TESIS DE LA UNIVERSIDAD DE ZARAGOZA

2011

11

Carlos Alvino Rovetto Ríos

Métodos basados en redes de Petri para el diseño de algoritmos de encaminamiento adaptativos mínimos libres de bloqueos

Departamento

Informática e Ingeniería de Sistemas

Director/es

Colom Piazuelo, José Manuel

ISSN 2254-7606

Prensas de la Universidad Universidad Zaragoza

Universidad Zaragoza
1542

Tesis Doctoral

MÉTODOS BASADOS EN REDES DE PETRI PARA EL DISEÑO DE ALGORITMOS DE ENCAMINAMIENTO ADAPTATIVOS MÍNIMOS LIBRES DE BLOQUEOS

Autor

Carlos Alvino Rovetto Ríos

Director/es

Colom Piazuelo, José Manuel

UNIVERSIDAD DE ZARAGOZA

Informática e Ingeniería de Sistemas

2011

Métodos basados en redes de Petri para el diseño de algoritmos de encaminamiento adaptativos mínimos libres de bloqueos.

Carlos Alvino Rovetto Ríos

Departamento de Informática e Ingeniería de Sistemas

TESIS DOCTORAL

Director: José-Manuel Colom Piazuelo

Universidad de Zaragoza
Zaragoza, España
Septiembre 2011

"Yo soy el Camino, la Verdad y la Vida. Nadie va al Padre sino por mí."

Evangelio según San Juan 14,6.

Dedicado a mis padres Bolivar y Catalina †.
A mi esposa Elia y a nuestro hijo Juan Carlos.

Agradecimientos

Ante todo doy gracias a Dios por haberme permitido finalizar los estudios doctorales y a la Santísima Virgen María en la advocación del Pilar de Zaragoza por su intercesión. Mi agradecimiento al director de la tesis Dr. José–Manuel Colom Piazuelo por su estrecha colaboración en la realización de esta tesis doctoral y al Departamento de Informática e Ingeniería de Sistemas de la Universidad de Zaragoza por todo el apoyo brindado. Quiero agradecer también a los profesores Dr. Manuel Silva Suárez director del grupo GISED, Dr. Fernando Tricas García, Dr. Javier Campos Laclaustra, Dr. Carlos Sagüés Blázquiz, Dr. Victor Viñals, Dr. Juan Domingo Tardós, Dr. Joaquín Ezpeleta Mateo, Dr. Enrique Teruel Doñate, Dr. Unai Arronategui Arribalzaga, Dr. Santiago Velilla Marco, Dra. Simona Bernardi, Dra. Sandra Baldassarri y Dr. Eduardo Mena Nieto. El agradecimiento también lo quiero extender para el personal de soporte informático: Ing. José Antonio Gutierrez Elipe, Ing. Carlos Gracia Heras, Ing. Angel Vargas Giménez, Ing. Alfonso García Pérez y el personal de administración del departamento quienes siempre nos atendieron amablemente, en especial al Sr. Alberto Ferrer Pérez y Sr. José Antonio Azorín Pons. A todos mis compañeros y ex-compañeros del programa de doctorado por su grata compañía y ejemplo de perseverancia, en especial para el Ing. Juan Pablo López–Grao, Dr. Darío Suárez Gracia, Ing. Raquel Trillo Lado, Ing. Carlos Bobed Lisbona, Ing. Javier Celaya Alastrué, Lic. María Pilar Albert, Dr. Hector Becerra Fermin, Ing. Diego Pérez–Palacín, Ing. Dorian Gálvez López y Ing. Javier Mauricio Antelis. Adicionalmente quiero agradecer a estas buenas personas que me han brindado una mano amiga, Sra. Betty Tejedor y los esposos Sr. Arturo López Martos y Sra. Conchi Vaquero Callejas de la Comunidad Familia, Evangelio y Vida.

Desde mi querido país, Panamá he sido siempre apoyado por varias personas entre las que debo resaltar al Ing. Salvador Rodriguez, Ing. Marcela Paredes, Dr. Victor Sánchez, Dr. Clifton Clunie, Ing. Raúl Barahona, Dr. Domingo Vega, Dr. Alexis Tejedor, Dr. Ignacio Chang, Ing. Lionel Pimentel, Lic. Alex Matus, Ing. José Mendoza, Ing. Bolivar Sanjur, Lic. Dariana Atencio, Lic. Velkis Araúz, Lic. Juana Aparicio, Lic. Lisenia Sanjur, Lic. Elena Mendoza, Lic. Damaris Nájera, Tec. María Edilma de Araúz, Sra. Olga Yolanda Beitia y la Sra. Nivia Gutierrez. Quiero expresar mi eterno agradecimiento a mi familia por su paciencia y apoyo incondicional a lo largo de estos años, a mi padre Bolivar Rovetto, mi hermano Franklin Rovetto y en especial a mi querida hermana Lic. Natalia V. Rovetto Ríos y su grupo de oración. Finalmente, darle las gracias a dos personas que han enriquecido mi vida; mi esposa Elia por su amor y compresión y a mi hijo Juan Carlos por su cariño.

Debo agradecer a las siguientes instituciones que han financiado parcialmente este trabajo: Universidad Tecnológica de Panamá, Secretaría Nacional de Ciencia, Tecnología e Innovación, Instituto para la Formación y Aprovechamiento de los Recursos Humanos, Universidad de Zaragoza, Fundación Carolina, Banco Santander Central Hispano. Este trabajo se realizó en el marco del European Community's Seventh Framework Programme bajo el proyecto DISC (Grant Agreement n. INFSO-ICT-224498) y el proyecto CICYT-FEDER DPI2006-15390.

Índice general

Índice de figuras

Índice de cuadros

Introducción

Los algoritmos de encaminamiento son ampliamente utilizados para la automatización de procesos de selección de rutas en áreas tan diversas de la ingeniería como las *redes de ordenadores, redes telefónicas, redes de microprocesadores, vehículos guiados automáticamente, sistemas de manufactura flexible entre otras*, sin embargo el proceso del encaminamiento puede generar situaciones indeseadas como los bloqueos. Este trabajo se enmarca en el desarrollo de una metodología para el diseño de algoritmos de encaminamiento adaptativos mínimos libres de bloqueos utilizando una visión *SAR* [Colom 03] para abstraer el sistema y las redes de Petri como herramienta formal de modelado.

Nuestro trabajo ofrece soluciones a las limitaciones que existen en las aproximaciones previas, que carecen de metodologías de modelado y métodos de corrección de los modelos. La gran mayoría de los trabajos que abordan este problema [Dally 87][Dally 93][Duato 95b][Taktak 08] utilizan grafos para representar el algoritmo de encaminamiento, sin embargo ante la presencia de un bloqueo no existen métodos sistemáticos de corrección. El diseño de algoritmos de encaminamiento es una tarea compleja [Dally 03] que debido a ello requiere la confiabilidad y la robustez de las soluciones que brindan los modelos matemáticos, por tal motivo utilizaremos las redes de Petri como la herramienta formal de modelado, análisis y síntesis del sistema.

Nuestra metodología permite abstraer el sistema utilizando una visión *SAR* que considera el encaminamiento de un mensaje u objeto como un proceso que demanda recursos en forma conservativa para desplazarse desde un origen hasta un destino. Esta metodología la aplicaremos en dos dominios de aplicación que utilizan algoritmos de encaminamiento en sus procesos como lo son las *las redes de interconexión para multicomputadores* y los *vehículos guiados automáticamente*. Apoyados en esta visión *SAR*, la primera etapa de la metodología llamada la *especificación* utiliza como información de entrada la dada por el diseñador como lo es la *topología, el algoritmo de encaminamiento y el control de flujo*. Esta información obtenida en la especificación se utiliza de forma sistemática en diversos grafos intermedios en la segunda etapa de la metodología denominada la *abstracción*. Para abstraer el sistema, y siguiendo nuestra visión *SAR* buscaremos hacer una clara diferencia entre los procesos y los recursos del

sistema. Los procesos de encaminamiento están formados de los diversos estados que alcanza un mensaje u objeto desde su origen hasta su destino a través de la red de interconexión. En cuanto a los recursos, claramente dependerá del dominio de aplicación que para las redes de interconexión serán los canales de comunicación y en los sistemas *AGV* los tramos recorridos por los vehículos.

Este proceso de abstracción da lugar a modelos con características estructurales particulares que nos permiten caracterizarlos y deducir propiedades estructurales sobre las que apoyarnos para realizar una análisis y síntesis del mismo. En el análisis se buscarán propiedades de buen comportamiento como la vivacidad del modelo que tiene una caracterización estructural para las redes que se obtienen con nuestra metodología. Adicionalmente, los modelos obtenidos tienen una gran riqueza semántica que nos permite caracterizar los procesos que ocurren en el sistema y así tener un conocimiento más profundo del comportamiento del mismo.

Los modelos de redes de Petri obtenidos con nuestra metodología en la etapa de obtención del *modelo formal*, presentan un orden particular en la adquisición y liberación de los recursos siguiendo nuestra visión *SAR*. Esta característica genera una estructura que se ha denominado *zonas de uso de un recurso* y que será ámpliamente utilizado para definir propiedades de una nueva clase de redes de Petri denominada $SOAR^2$ *(S^4PR with Ordered Allocation and Release of Resources)* y que caracteriza de forma precisa el encaminamiento de mensajes y objetos a través de una red.

En los procesos de encaminamiento se utilizan recursos que son tomados y liberados de forma unitaria, por tal motivo solo existen transiciones que adquieren o liberan recursos pero no en ambos casos en este tipos de redes de Petri. Estas zonas pueden solaparse entre ellas, debido a que un estado de un proceso de encaminamiento puede utilizar de forma simultánea diversos recursos por lo tanto en ese estado estarán solapadas estas zonas de recursos. Esta característica del solapamiento se formaliza en una propiedad y se definen los denominados *conjuntos máximos de zonas solapadas* restringidas a un conjunto de recursos en el que preservarán una relación de orden total. Posteriormente, se establece una relación entre estos conjuntos máximos de zonas solapadas y los cerrojos malos de la red de Petri que es una estructura que captura la espera circular que existe entre los recursos de un bloqueo. Por este motivo, los cerrojos tienen una gran importancia en la caracterización de la vivacidad del modelo de red de Petri en la etapa denominada *análisis* de nuestra metodología. Las zonas que se encuentran solapadas entre ellas tienen lugares particulares que una vez removidos $(podados)$ generan un cerrojo mínimo entre ellas. Esta fuerte relación es la base para verificar la vivacidad del modelo utilizando herramientas gráficas como el *grafo de poda entre zonas* que se desarrolla en esta tesis y el *grafo de poda de recursos*[Cano 10].

Para representar las zonas y su relación de podado se definirá el grafo de poda entre zonas, además realizando un proceso de aglomeración de las zonas del grafo de poda de zonas se puede generar el grafo de poda de recursos, del cual podemos determinar si existen cerrojos malos en la red. Para determinar si existen cerrojos malos en la red se verificará si el grafo de poda de recursos es acíclico o no. La existencia de ciclos en el grafo de poda de recursos es condición necesaria para la existencia de cerrojos malos en la red.

Posterior al análisis del modelo, nuestra metodología tiene una etapa de síntesis del modelo para garantizar la vivacidad a través de una política de control que utiliza recursos virtuales. En esta última etapa se buscará eliminar los ciclos en el grafo de poda de recursos a través de la introducción de recursos virtuales. Este novedoso método de control está fuertemente apoyado en los dos grafos antes mencionados debido a que permiten realizar una eliminación de los ciclos de forma eficiente. Adicionalmente, este método presenta retos debido a que en algunos casos se deben modificar la estructura de los procesos para garantizar la estructura de los mismos una vez introducido nuevos recursos. Estos retos serán resueltos utilizando algoritmos especializados y determinadas heurísticas para corregir el modelo y eliminar los cerrojos malos.

Esta memoria se organiza de la siguiente forma.

1. *Capítulo 1*: En este capítulo se presentará el problema que deseamos abordar y su uso en los diferentes dominios de aplicación con los que trabajaremos. Esto incluirá también la presentación de nuestro enfoque de modelado de nuestra metodología y una caracterización de la clase de redes obtenidas. Esta caracterización incluirá un pequeño ejemplo sobre el problema que abordamos utilizando las redes de Petri para modelarlo.

2. *Capítulo 2*: Aquí se introduce nuestra metodología de modelado para los sistemas de encaminamiento junto a la explicación detallada de cada una de sus etapas. Para explicar estas etapas se recurrirá a un ejemplo de una red de interconexión que presenta problemas de bloqueos. Con este ejemplo se obtendrá un modelo formal en redes de Petri que se sintetizará en capítulos posteriores.

3. *Capítulo 3*: Se define la nueva clase de redes de Petri denominada $SOAR^2$ y para la cual se definen características estructurales particulares sobre la cual razonar. Una de estas características es el concepto de zona que se relaciona de forma estrecha con el concepto de cerrojo. Ambos conceptos son muy importantes para apoyarnos en la caracterización de la vivacidad.

4. *Capítulo 4*: Este capítulo se dedica a presentar y desarrollar nuestra política de control para garantizar la vivacidad del modelo utilizando canales virtuales. Finalmente, el método de corrección se presentará aplicado al dominio de las redes de interconexión.

5. *Capítulo 5*: Se resumirán las contribuciones fundamentales de nuestro trabajo así como las conclusiones del mismo.

Finalmente se ha incluido tres anexos denominados *Anexo A, Anexo B* y *Anexo C*, que proporcionan terminología básica sobre las *Redes de Interconexión*, los *Vehículos Guiados Automáticamente* y sobre las *redes de Petri* respectivamente.

Capítulo 1

El problema de los algoritmos de encaminamiento

Resumen
En términos generales el proceso del encaminamiento de un objeto consiste en la selección de una ruta desde un origen hasta un destino. La utilización de algoritmos de encaminamiento es la mejor alternativa si se desea automatizar la selección de rutas, pero este proceso puede generar situaciones indeseadas como los **bloqueos**. Hoy en día los algoritmos de encaminamiento son ampliamente utilizados en diversos dominios de aplicación y su demanda crece en la medida en que se requieren sistemas automatizados para el movimiento automático de objetos *físicos o virtuales*. Diseñar algoritmos de encaminamiento libres de bloqueos es un área de investigación activa que puede ser abordada a través de modelos matemáticos por la confiabilidad y la robustez de las soluciones que brindan. En este trabajo abordaremos el problema de diseñar algoritmos de encaminamiento adaptativos mínimos libres de bloqueos a través de una metodología de diseño apoyada en un enfoque *SAR* del sistema. Nosotros utilizaremos las *Redes de Petri* como herramienta formal de modelado, análisis y síntesis del sistema en donde se presentarán ejemplos en dos dominios de aplicación.

1.1. Introducción

En términos generales el proceso del encaminamiento de un objeto implica la selección de una ruta desde un origen hasta un destino. La selección de la mejor ruta

es un problema que tiene usos en diferentes escenarios como por ejemplo: *encontrar direcciones automáticamente entre localizaciones físicas en un mapa, los procedimientos de optimización en investigación de operaciones, movimiento de información en redes de paquetes de datos, encaminamiento de vehículos guiados automáticamente, etcétera*.

Obtener sistemáticamente una solución óptima al problema del encaminamiento es un problema que se ha estado investigando a lo largo de muchos años. Para abordar este problema se han utilizado diversas herramientas, siendo los grafos la herramienta preferida para el razonamiento lógico del problema del encaminamiento. Los grafos se han preferido sobre otras herramientas por tener una estructura gráfica para la abstracción, modelado y representación matemática del problema del encaminamiento. Utilizando los grafos como modelo matemático se logró desarrollar el algoritmo del camino mas corto entre dos vertices de un grafo. Encontrar la ruta mas corta es un problema que fue resuelto por [Dijkstra 59] y es la base para muchos algoritmos de encaminamiento. Sin embargo existe el *Problema del Viajante* (Travelling Salesman Problem) [Bellman 62] que es un buen ejemplo que la complejidad que existe en encontrar la ruta mas óptima entre dos vértices de un grafo cumpliendo determinadas restricciones. Este problema pertenece a la categoría de los problemas NP-completos. La mejor solución es evaluar todas las posibles combinaciones de recorridos hasta encontrar la menor distancia, pero esto es computacionalmente prohibitivo en la mayoría de los casos reales. Debido a esta complejidad, la solución mas utilizada es aplicar distintas técnicas que en muchos casos incluyen heurísticas entre otras. Una técnica probabilista que da resultados casi óptimos al problema del viajante está inspirada en el comportamiento que presentan las hormigas [Dorigo 92] para encontrar las trayectorias desde la colonia hasta el alimento.

Durante el proceso del encaminamiento de un objeto en un sistema, tratamos de encontrar un camino entre dos puntos llamados *origen* y *destino*. Este proceso tiene usos en diferentes dominios de aplicación, pero el proceso de encaminamiento sigue siendo el mismo. Para abordar el problema de encaminamiento utilizaremos el enfoque de *Sistema de Asignación de Recursos (SAR)* apoyado en herramientas formales como las redes de Petri. En este trabajo desarrollaremos una metodología para el diseño de algoritmos de encaminamiento adaptativos mínimos libres de bloqueos en donde el enfoque *SAR* será ampliamente utilizado. Esta metodología tendrá aplicación en diversos dominios, en donde enfrentan problemas de bloqueos en el encaminamiento de objetos desde un origen a un destino utilizando algoritmos de encaminamiento adaptativos mínimos. En este capítulo se abordarán los siguientes temas. En la sección 1.2, presentaremos dos aplicaciones de los algoritmo de encaminamiento e incluiremos una breve introducción de los dos dominios de aplicación con los que trabajamos. Nuestro enfoque *SAR* de los algoritmos de encaminamiento será desarrollado en la sección 1.3. La presentación del problema de *bloqueo* en los algoritmos de encaminamiento se presentará en la sección

1.3.1. En esta sección mencionaremos otras aproximaciones metodológicas existentes para abordar este problema, así como nuestra motivación para abordarlo. Finalmente en la sección 1.4 haremos una breve introducción de la clase de redes de Petri denominada S^4PR. Esta clase de redes es fundamental debido a que aprovecharemos los resultados teóricos que se tienen para aplicar las políticas de control en la etapa de síntesis en nuestra metodología.

1.2. Aplicaciones de los algoritmos de encaminamiento

El uso y diseño de algoritmos de encaminamiento para la selección de rutas óptimas se utiliza en áreas tan diversas de la ingeniería como *redes de ordenadores, redes telefónicas, redes de microprocesadores, vehículos automáticos, sistemas de manufactura flexible entre otras*. Adicionalmente, en relación al tiempo y los recursos que se utilizan, el transporte es uno de las componentes más importantes y costosos de muchos sistemas para realizar esta actividad. Dentro de la gestión del transporte una de las decisiones operativas que deben tomarse es la selección de la mejor ruta para atender la demanda de encaminamiento que tienen los procesos en el sistema. Durante este proceso emergen problemas como son los **bloqueos**, siendo en muchos casos catastróficos para todo el sistema. El desarrollo de este trabajo se concentra en sistemas que utilizan algoritmos de encaminamiento adaptativos mínimos en sus procesos de encaminamiento. Para demostrar la viabilidad de nuestra solución nos basaremos en dos casos de estudio que abordan el problema del bloqueo en las *Redes de Interconexión* y los *Vehículos Guiados Automáticamente*.

1.2.1. Redes de interconexión

En general un sistema computacional está formado por tres bloques básicos denominados *lógica, memoria* y *comunicación*, en donde la comunicación suele ser el cuello de botella que impide obtener mejores índices de rendimiento. La tecnología *VLSI* ha permitido la construcción de sistemas informáticos cada vez más potentes en un solo circuito integrado. Esta tecnología ha logrado integrar cientos de procesadores idénticos en un solo circuito integrado denominados *CMPs*. Otro logro destacado es la integración de diversos tipos de procesadores en un solo circuito integrado denominados *SoCs*. Adicionalmente, existen versiones híbridas entre los grupos previos, pero en todos los casos se necesita que los circuitos estén comunicados de manera eficiente, flexible y escalable [Owens 07]. Una *Red de interconexión* es un sistema programable subyacente que permite la comunicación eficiente de datos entre terminales denominadas origen y

destino [Dally 03]. La tecnología que permite esta comunicación a través de las redes de interconexión es también conocida como *OCIN* o redes de microprocesadores *NoC*. Hoy en día las redes de interconexión han reemplazado muchos de los sistemas dedicados de comunicación porque brindan mayor escalabilidad y un uso eficiente de los recursos como los canales físicos de comunicación, buffers, memoria y ciclos de procesamiento. Existen muchos criterios para clasificar las redes de interconexión, pero tomando como referencia los dispositivos o componentes de los sistemas digitales es posible crear una taxonomía como sigue:

1. *Conectar Procesadores y Memorias*: Brinda la comunicación entre los procesadores y la(s) memoria(s) del sistema. Existen dos configuraciones. La primera es llamada **Pasillo del Baile** *(DanceHall en inglés)* como se muestra en la figura 1.1.a. Es utilizada en casos específicos debido a su menor coste, sin embargo presenta problemas debido a su ancho de banda limitado. La segunda es llamada **Integración de Nodos** como se muestra en la figura 1.1.b. Esta configuración concentra el procesador y la memoria en un nodo, siendo la más utilizada actualmente. La ventaja es que cada procesador es capaz de utilizar su propia memoria lo cual brinda escalabilidad.

2. *Sistemas de Computadores a Dispositivos de Entrada/Salida*: Es usado en sistemas de computadoras para conectar dispositivos de entrada/salida, discos duros, interfaces de red, memorias como se muestra en la figura 1.1.c. La diferencia con otros usos está en la velocidad de acceso a los dispositivos.

3. *Red de Encaminadores*: Es usada para reemplazar dipositivos de Internet, permitiendo incluir las unidades de conmutación en nodos de una red de conmutadores y encaminadores [Towles 05]. Esto se debe a la enorme demanda de ancho de banda producto de las nuevas aplicaciones y usuarios en Internet. Significa aplicar la tecnología de las redes de interconexión para microprocesadores en los encaminadores de Internet.

Actualmente, los ámbitos de aplicación más frecuentes de las redes de interconexión en los sistemas informáticos son: 1) *Clusters* o grupos de computadoras como Altavista, Google y Hotmail. 2) *Computadores masivamente paralelos* como CRAY T3D, RedStorm, Blue Gene L/P. y 3) *Encaminadores de altas prestaciones para Internet*. Claramente, su uso no está restringuido a estos sistemas, pero es en ellos en donde encontramos los beneficios mas tangibles de esta tecnología de comunicación.

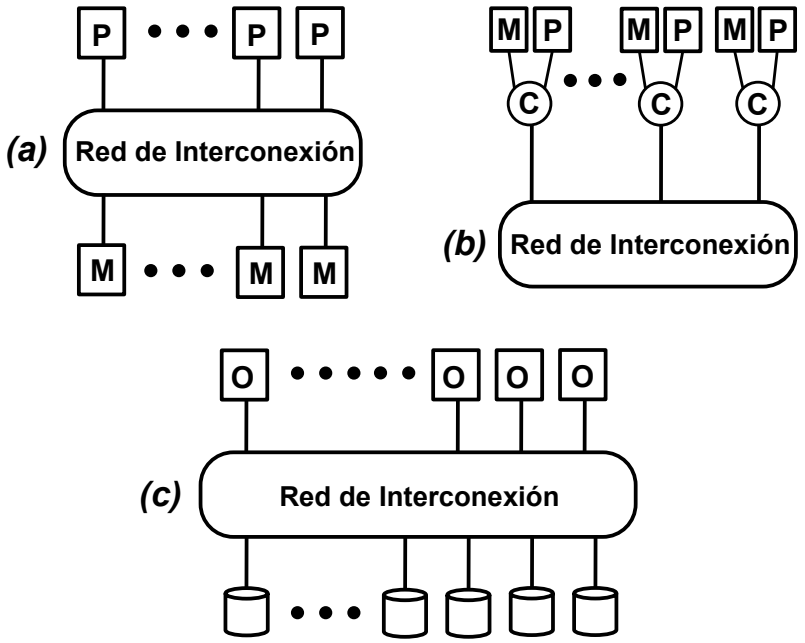

Figura 1.1: Tipos de redes de interconexión

Uno de los escenarios de nuestra investigación será las redes de interconexión para *multicomputadores*, pero los resultados obtenidos pueden ser llevados a otros tipos de redes en que se utilicen algoritmos de encaminamiento adaptativos mínimos. En la figura 1.1.b. se puede ver un esquema de una red de interconexión para multicomputadores en la que todos los nodos ejecutan la misma aplicación y cada procesador tiene su propia memoria. La aplicación de cada nodo modifica los datos que son compartidos entre los nodos de la red a través del paso de mensajes. Es preciso recordar que en esta arquitectura, cada nodo tiene su propia memoria lo que reduce significativamente el flujo de mensajes en la red vs memoria compartida. Para el diseño de los algoritmos de encaminamiento para una red de interconexión de multicomputadores se necesitan tomar en cuenta ciertos parámetros y requisitos de diseño como son:

1. *Topología*: Se refiere a la disposición geométrica de conexión y organización del conjunto de nodos y canales que forman la red de interconexión.

2. *Encaminamiento*: Es el método que se emplea para determinar la ruta utilizada por un mensaje para ir desde un origen a un destino en la red. Este proceso puede realizarse de forma incremental en cada nodo de la red o centralizado.

3. *Control de flujo*: Se encarga de la asignación de los canales a los mensajes que los solicitan. Las estrategias más utilizadas son:

   a) *store-and-forward*: Un mensaje que llega a un nodo se almacena completamente en el buffer antes de ser reenviado.

   b) *virtual cut-through*: Se interpreta solo la cabecera de cada mensaje que llega a un nodo y se reenvía, sin la necesidad de esperar a que se reciba en su totalidad. En caso de que el mensaje no se pueda reenviar, se almacena el nodo y se comporta igual que la técnica *store-and-forward*.

   c) *wormhole*: En esta técnica los paquetes son divididos en pequeñas unidades de control de flujo llamadas **flits**. Una vez que se recibe en el nodo el primer flit se reenvía inmediatamente hacia el puerto de salida antes de recibir el paquete completo. En el nodo no existe buffer para todo el paquete, motivo por el que esta técnica es prospensa a bloqueos, si no existe un canal de salida disponible en el nodo.

La técnica *wormhole* es una de las más utilizada en las redes de interconexión debido a su pequeña exigencia en buffer y menor latencia. Una desventaja con wormhole es que en caso de existir una parada en el flujo del paquete, no se tiene espacio para almacenarlo y los canales reservados no pueden ser utilizados por otros paquetes. En nuestra aproximación nos apoyaremos en las condiciones definidas en [Dally 86] para el comportamiento de los mensajes en una red de interconexión que utiliza control de flujo wormhole.

1. Un mensaje que llega a su destino es eventualmente consumido.
2. Un nodo puede generar mensajes destinados a cualquier otro nodo.
3. Para algoritmos no adaptativos, la ruta tomada por un mensaje es determinada únicamente por su nodo destino.
4. Un nodo puede generar mensajes de largo arbitrario. Los paquetes serán mayores que un flit en la mayoría de los casos.
5. Para una cola de un buffer, una vez que acepta el primer flit de un paquete, debe aceptar los flits restantes antes de aceptar flits de otro mensaje.
6. Una cola de buffer no puede escoger entre mensajes que la solicitan pero si intermediar entre ellos.
7. Los nodos pueden producir mensajes a cualquier tasa pero sujetos a las restricciones del buffer disponible en la cola fuente.

La topología de la red de interconexión es otro elemento fundamental a considerar para el buen diseño de los algoritmos de encaminamiento. Cada algoritmo de encaminamiento está diseñado para encaminar mensajes en determinados tipos de topologías. Las topologías pueden clasificarse según muchos criterios. Con base en la forma de utilizar el medio físico, se pueden clasificar en las que son de *medio compartido* y las que *no comparten el medio*. Las topologías de medio compartido tienen la ventaja de tener una arquitectura simple que reduce su costo de implementación, pero brindan bajas prestaciones por lo tanto son menos utilizadas que las que comparten el medio físico.

Según los tipos de nodos, las topologías se pueden clasificar en *directas* e *indirectas*. En las topologías directas, los nodos son de un mismo tipo y pueden emitir y recibir mensajes a otros nodos de la red. Las topologías indirectas pueden tener nodos que no realizan ambas funciones. Un buen ejemplo es la topología mariposa *(butterfly)* en la que los nodos internos son solo conmutadores que redirigen los mensajes pero no los consumen o los generan como el caso de los nodos de los extremos. Las topologías directas se subdividen en *ortogonales*, es decir, simétricas, y en redes *no ortogonales* o no simétricas. Las topologías ortogonales son preferidas por muchas razones de diseño e implementación, siendo las *mallas*, *toros* e *hipercubos* las más utilizadas. La figura 1.2 muestra un ejemplo de cada una de las topologías ortogonales. Los hipercubos han sido menos utilizados por su elevado costo de escalabilidad. Finalmente existen las topologías híbridas que combinan características de la clasificación antes mencionada.

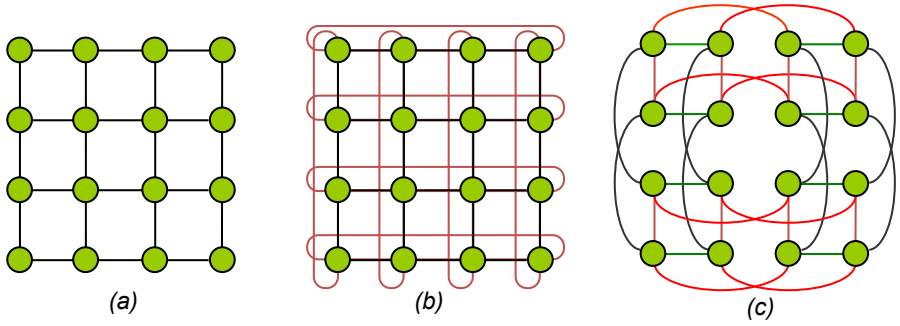

Figura 1.2: Topologías directas ortogonales

En este trabajo se crea una metodología de modelado formal para el diseño de algoritmos de encaminamiento adaptativos mínimos, desde una visión *SAR* del sistema. Con nuestra metodología podemos abstraer y representar el encaminamiento de un mensaje por la red de interconexión y así estudiar propiedades de buen comportamiento como la ausencia de bloqueos en el algoritmo de encaminamiento.

Los algoritmos de encaminamiento pueden clasificarse tomando en cuenta diversos criterios diferenciadores, sin embargo en muchos casos existen algoritmos que tienen características híbridas que inicialmente son diferenciadoras. Debido a estas situaciones hemos tomado en cuenta solo los aspectos más relevantes y claramente diferenciadores para hacer una clasificación de los algoritmos de encaminamiento en diferentes dominios de aplicación. La forma de tomar las decisiones del encaminamiento nos permite diferenciar los algoritmos de encaminamiento en *estáticos o deterministas* y aquellos que son *adaptativos o dinámicos*. Ambas estrategias son ampliamente utilizadas debido a que cada una brinda ventajas dependiendo de su implementación o el dominio de aplicación.

1. *Estático o Determinista*: Las rutas entre los nodos son calculadas con anticipación o *fuera de línea* y memorizadas en los nodos durante su inicialización. Estas rutas permanecen invariantes hasta que se vuelvan a calcular debido a la adición de nuevas rutas o nuevos destinos. Este tipo de algoritmo es poco tolerante a los fallos pero sigue siendo utilizado hoy en día por su simplicidad.

2. *Adaptativo o Dinámico*: Usan información del estado del sistema y en base a ésta, cambian *en línea* sus decisiones de encaminamiento. La información de estado puede reflejar modificaciones en las rutas debido a congestión, pérdida de enlaces, pérdida de destinos, etc.

El lugar en donde se toman las decisiones y el origen de la información intercambiada para seleccionar las rutas nos permite diferenciar los algoritmos de encaminamiento en *centralizado* o *distribuido*.

1. *Centralizado*: Si la información se centraliza en un controlador principal, estaremos hablando de un encaminamiento centralizado.

2. *Distribuido*: Si cada nodo es autónomo y solo utiliza información local para tomar las decisiones, estaremos hablando de un encaminamiento distribuido.

Los mensajes pueden tener uno o múltiples destinos, motivo por el cual los algoritmos de encaminamiento pueden clasificarse en aquellos que envian a un *destino único* o los que envían a *múltiples destinos*.

1. *Destino Único*: Existe un solo origen y destino para cada mensaje y es el caso más común en las redes de transporte de datos.

2. *Múltiples Destinos*: Los mensajes tienen múltiples receptores, optimizando el uso de los recursos en disfusiones masivas. Este método requiere protocolos especializados y mecanismos de subscripción por lo tanto es menos utilizado.

Una característica ampliamente utilizada para clasificar los algoritmos de encaminamiento es la distancia de la ruta seleccionada para enviar un mensaje desde un origen hasta un destino. La distancia desde un origen al destino, está dada por la cantidad de vértices o nodos entre ellos. Bajo este criterio podemos clasificar los algoritmos de encaminamiento en aquellos que son de rutas *mínimas* y los de rutas *no mínimas*.

1. *Mínimos*: Cuando la cantidad de nodos entre el origen y el destino es la menor.

2. *No mínimos*: Cuando la distancia de la ruta entre el origen y el destino no es la menor.

Es preciso aclarar que aunque la topología permita tener rutas no mínimas, será el algoritmo quien seleccione o no estas rutas para encaminar un mensaje. En nuestra metodología abordamos el caso de los algoritmos de encaminamiento adaptativos mínimos. En las subsecciones siguientes 1.2.2 y 1.2.3 abordaremos la utilización de los algoritmos de encaminamiento adaptativos mínimos en el dominio de aplicación de las redes de interconexión y los vehículos guiados automáticamente respectivamente.

1.2.2. Enfoques de diseño para las redes de interconexión

Las redes de interconexión tienen una historia que inicia con la transmisión y conmutación de datos en las centrales telefónicas. Hoy en día, su labor sigue siendo la misma, pero aplicada a diversos dominios de aplicación, distinguiéndose los usos para los sistemas informáticos. Actualmente, la comunicación se basa en el intercambio de mensajes entre los nodos de la red, que fue introducido en [Dally 86]. Este enfoque implica el encaminamiento de mensajes por la red, sin embargo pueden ocurrir problemas de bloqueo en el proceso del encaminamiento. Los bloqueos aparecen como un problema en el diseño de los algoritmos de encaminamiento, motivo por el cual su correcto diseño es de gran importancia. El diseño de algoritmos de encaminamiento es una tarea compleja, en donde los trabajos más relevantes han utilizado grafos, para modelar los algoritmos de encaminamiento [Dally 87][Duato 93][Duato 95b][Gregorio 95][Schwiebert 96], sin embargo carecen de procedimientos metodológicos de corrección. El trabajo de [Dally 87], estableció las condiciones para una prueba que garantizaba la ausencia de bloqueos a través de un grafo acíclico denominado *Grafo de Dependencias (GD)* entre canales. Este trabajo utilizó la técnica de control de flujo tipo *wormhole* para algoritmos de encaminamiento de tipo estático pero también fue utilizada en algoritmos de tipo adaptativo [Chien 95][Linder 91][Glass 92]. Es preciso mencionar que el control de flujo *wormhole* es más utilizado que otras técnicas por su baja latencia y reducido tamaño de los buffers, pero es más prospenso a generar bloqueos durante el encaminamiento, debido

a que no existe espacio suficiente en el buffer para almacenar todo el mensaje. En base al trabajo de [Dally 87], el *GD* permite verificar la existencia o no de bloqueos en el algoritmo de encaminamiento, a través de la presencia o no de ciclos en el *GD*. Si existen ciclos, éstos tienen que ser removidos utilizando canales virtuales [Dally 92][Dally 93] que se asignarán en estricto orden ascendente o descendente para así prevenir la existencia de bloqueos durante el encaminamiento. Apoyados en el *GD*, los trabajos de [Glass 92][Boura 93] brindaron un método mas flexible para eliminar los ciclos, evitando ciertos tipos de giros en el recorrido de los mensajes porque generan ciclos y por lo tanto bloqueos. El trabajo de [Dally 87], brindó una condición suficiente para la existencia de bloqueos en los algoritmos de encaminamiento, pero [Duato 93] probó que esto no se aplicaba a algoritmos de tipo adaptativos. Su estrategia consistió en definir el *Grafo de Dependencias Extendido (GDE)* entre canales, en donde el algoritmo de encaminamiento es libre de bloqueos aunque tenga ciclos en el *GD* pero no el *GDE*. Este trabajo cambió la forma de verificar la existencia de bloqueos en algoritmos de tipo adaptativos, lo cual influyó significativamente en los trabajos posteriores [Schwiebert 95a][Schwiebert 95b]. Es preciso mencionar que existen trabajos que abordan el problema del bloqueo para cierta clase de algoritmos como los inconscientes en [Schwiebert 01][Domke 11] o con suposiciones de comportamiento diferentes a las presentadas en [Dally 87] como en [Taktak 08]. Nuestro trabajo [Rovetto 10a] se enfoca en los algoritmos de encaminamiento de tipo adaptativo mínimo, utilizando nuestra metodología con la que somos capaces de modelar topologías regulares e irregulares.

1.2.3. Vehículos guiados automáticamente (AGVs)

Los aportes de nuestra metodología para el diseño de algoritmos de encaminamiento adaptativos mínimos libres de bloqueos, pueden ser llevados a otros dominios de aplicación en donde se utilizan algoritmos de encaminamiento para automatizar procesos de selección de rutas. En determinadas fábricas, almacenes de distribución, hospitales se requiere mover objetos *(piezas, partes, herramientas, insumos, etcétera)* desde un punto a otro de forma segura y automática. Para estos casos el sistema de transporte flexible más utilizado son los **Vehículos Guiados Automáticamente** o más conocidos por sus siglas en inglés como *AGVs - (Automated Guided Vehicles)* [Müller 83].

Los *AGVs* son robots móviles que carecen de un conductor permanente y están diseñados para diversas actividades motivo por el cual existe un gran variedad de ellos. Hoy en día, su uso va en aumento por las ventajas en la seguridad laboral y las reducciones en los costes de trabajo, así como la minimización de los daños a los materiales u objetos que se transportan [Chiba 06]. Adicionalmente, los *AGVs* pueden operar simulťaneamente con otros sistemas de transporte y tienen la ventaja funcional de poder modificar sus

trayectorias a bajo costo, en comparación a las cintas transportadoras, cadenas de montaje, etcétera. Análogo a los sistemas informáticos, los *AGVs* normalmente están compuestos por dos subsistemas de *hardware* y *software* que interactuan entre si [Ling 99]. En el hardware se encuentran los componentes físicos como los *AGV*, rutas, controladores, sensores, dispositivos de guiado, etc. En el software se encuentran los algoritmos que sistemáticamente administrarán los recursos de los *AGVs*, entre ellos la selección de las rutas. Los *AGVs* son administrados por un sistema de control que se puede subdividir en las siguientes tareas.

1. Control del tráfico.
2. Navegación.
3. Selección de las rutas.

Actualmente los *AGVs* se comunican con el resto de vehículos y con el sistema de control a través de señales de radio frecuencia [Lozoya 07], para realizar la navegación y el control del tráfico, aunque en algunos casos son utilizados rayos láser con el mismo objetivo. El *control del tráfico* está asociado a los procesos que se encargan de dirigir el tránsito de los *AGV* en el área de trabajo de forma segura, ordenada y rápida. La *navegación* es el proceso que tiene como objetivo orientar al vehículo por el área de trabajo [Rupp 98], para que sea guiado por el algoritmo de encaminamiento en la *selección de las rutas*. Las técnicas de navegación han evolucionado desde el uso de cables soterrados hasta la utilización de rayos láser. Hoy en día se siguen usando cables soterrados por ser una opción económica y fiable al igual que las cintas magnéticas sobre el piso y los métodos inertes como los imanes. Una técnica óptica menos utilizada es el uso de luz ultravioleta. El vehículo AGV sigue una ruta marcada sobre el suelo con una sustancia fluorescente que, al ser activada mediante una luz ultravioleta, es detectada por dos células fotoeléctricas en la base del vehículo. Al variar la frecuencia de la luz ultravioleta, la señal detectada es distinta y permite pasar de un segmento del circuito otro segmento. Uno de los problemas con este método son las irregularidades del suelo o ambientes contaminados que impidan detectar la sustancia fluorescente.

Otra técnica para la navegación ámpliamente utilizada es el uso de rayos láser por la ventaja de la precisión que ofrecen y su flexibilidad. En cada vehículo se instala un escáner láser que detecta puntos reflectantes que son colocados estratégicamente sobre el suelo o la pared. Un vez detectados estos puntos reflectantes son utilizados como puntos de referencia por el *AGV* que entonces es capaz de orientarse y de actualizar su posición actual con ayuda de un algoritmo de triangulación integrado en el software de administración. La utilización de los algoritmos de encaminamiento se ha destinado a la *selección de las rutas* por la libertad de movimiento que ofrece este metodo en el diseño de los sistemas.

La selección de la ruta es parte del sistema de control y es realizada por el *algoritmo de encaminamiento*, el cual es un software que dirige el vehículo desde su origen a su destino programado [Lee 98]. En términos generales, las tres tareas del sistema de control de los *AGVs* son comunes en todos los modelos existentes, sin embargo dicha taxonomía no es única, porque varía dependiendo del fabricante y el modelo del equipo. Para clasificar los *AGVs* existen diversos criterios y en nuestro caso lo haremos en base a la funcionavilidad de los modelos, aunque todos ellos comparten muchos de los componentes funcionales y del software de administración:

1. *AGV apilador*: Este modelo es utilizado para el movimiento y elevación de paletas *(pallets)*. Existen submodelos en base al tipo de brazo utilizado para sujetar la paleta o el movimiento que se realiza con ella.

2. *AGV remolque*: Es utilizado para arrastrar vagones que pueden tener diversos tamaños y pesos. Usualmente trabajan en circuitos cerrados.

3. *AGV para transporte de carros*: Es utilizado para desplazar cargas pequeñas a través pasillos hospitalarios y elevadores convencionales. Además, pueden realizar otras labores automáticas como la carga o la descarga del material.

4. *AGV para transporte de cargas pesadas*: Se utilizan en fábricas de maquinaria pesada o automóviles, debido a que los objetos pesan varias toneladas o son muy voluminosos.

Nuestro trabajo se enfocará en los *AGVs* tipo *remolque* como lo esquematiza la figura 1.3, y que son más conocidos por su término en idioma inglés como *Tow AGVs*. Podemos mencionar que los primeros vehículos *AGV* que se fabricaron en la década del cincuenta con fines comerciales, eran modelos tipo remolque. Su sistema de orientación estaba basado en un cable enterrado en el suelo de la fábrica para guiarse a través del campo magnético que generaba este cable. Los *Remolques AGVs* son una solución diseñada para arrastrar vagones de carga en las fábricas y almacenes de distribución, hospitales, etcétera. Estos *AGVs* tienen una movilidad reducida debido a que no operan en sentido inverso, por lo tanto es común que trabajen en circuitos. Debido a que los *AGVs* tipo *remolque* no pueden retroceder, les es imposible escapar de un bloqueo o colisión, dando marcha atrás como ocurre con otros modelos de *AGVs*. Esta limitación en la movilidad obliga a prestar mucha atención durante el diseño del algoritmo de encaminamiento del *AGV* para hacerlo libre de bloqueos. El conjunto formado por el primer vehículo hasta el último vagón es denominado el *Convoy AGV*. Para ser coherente con nuestra metodología de modelado y visión *SAR* del sistema, llamaremos al primer vehículo del convoy la *cabeza* y al último vagón la *cola*, como se puede apreciar en la figura 1.3.

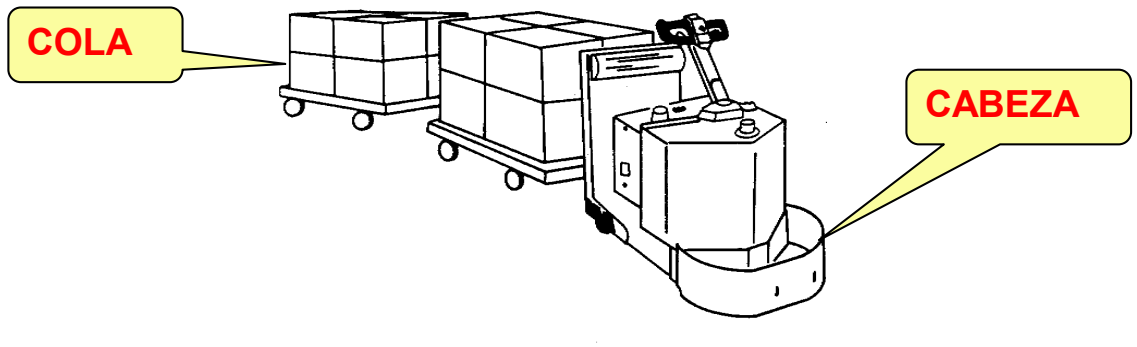

Figura 1.3: Vehículo Guiado Automáticamente tipo remolque

1.2.4. Enfoques de diseño para los AGVs

En nuestro enfoque metodológico del sistema generamos una visión abstracta de la fábrica, centro de distribución, hospital etcétera como una *plataforma de trabajo*. Desde nuestra perspectiva *SAR*, la plataforma de trabajo es un sistema programable para el movimiento de objetos entre estaciones de trabajo utilizando los *AGVs*. Las aproximaciones existentes de modelado y diseño para tratar los problemas que surgen durante el encaminamiento de los *AGVs* tienen diversos enfoques, debido a que se aborda una gran cantidad de problemas como las *control de colisiones, congestión, bloqueos, bloqueos activos, etc*. A continuación mostramos algunas aproximaciones relevantes sobre el problema del bloqueo en el encaminamiento en *AGVs*. La primera aproximación que trató el problema del encaminamiento, desde la perspectiva de encontrar una ruta incrementalmente entre un origen y un destino para los *AGVs* fue [Stephen C 88]. Desde una perspectiva similar el trabajo de [Kim 91] abordó la evitación de los bloqueos utilizando ventanas de tiempo para realizar las misiones de los vehículos *AGV*. Este enfoque es costoso en relación al tiempo necesario para hacer las misiones. Un enfoque mas dinámico y novedoso para abordar el problema de los bloqueos en los *AGVs*, fue presentado por [Taghaboni-Dutta 95], en donde se utiliza la información del estado de las estaciones vecinas y la plataforma misma. Este método tiene poco rendimiento cuando el número de vehículos crece significativamente. Las aproximaciones que abordan el problema de bloqueo, utilizando métodos formales como las redes de Petri y una visión *SAR*, la encontramos en [Reveliotis 96] y posteriormente [Reveliotis 00] y [Fanti 02]. Tomando como base estos trabajos, encontramos aproximaciones que abordan el bloqueo en el encaminamiento a través de las redes de Petri en [Wu 04] y [Wu 05]. Los enfoques previos brindan soluciones a los bloqueos en el encaminamiento pero carecen de un

enfoque estructurado de modelado de las rutas existentes en la plataforma de trabajo. La plataforma es un factor importante en el diseño de un sistema para movimiento de material. En la extensa literatura sobre la manipulación de materiales, se utiliza la clasificación de los esquemas de la plataforma de trabajo dada por [Kul 85] como sigue:

1. *Esquema de posición fija*: En este caso los procesos se realizan sobre un producto que es grande y pesado y por lo tanto permanece en una sola posición durante la mayoría de la fabricación. Por ejemplo, grandes barcos, maquinaria enorme.

2. *Esquema de proceso*: Todos los procesos de un mismo tipo o similares son realizados en una misma área. El material es movido a través de estas áreas.

3. *Esquema del producto*: El material o producto es deplazado consecutivamente por cada una de las ensambladoras o máquinas que lo tratan. Este esquema es ampliamente utilizado para productos estandarizados.

En nuestro trabajo nos concentraremos en los *Esquema de proceso* debido a que son los más complejos de administrar debido a la necesidad de transportar materiales o herramientas. En este trabajo se desarrollará una metodología para el diseño completo de algoritmos de encaminamiento adaptativos mínimos libres de bloqueos para *AGVs* que operan en una plataforma de trabajo bajo el esquema de proceso. Para seleccionar la ruta del *AGV*, el algoritmo de encaminamiento utiliza componentes electrónicos *(láser, radio)* o no *(imanes, etc)*, para guiar el vehículo dentro de la plataforma de trabajo. Las rutas de la plataforma de trabajo tienen divisiones denominadas *segmentos*, por donde se desplazará los *AGVs*. Los segmentos pueden converger o diverger entre ellos en puntos llamados los *puntos de decisión*. Es preciso mencionar que los puntos de decisión pueden ser utilizados para enviar más de una instrucción al *AGV*, como cambiar la velocidad del vehículo, hacer una pausa, etc. Estos puntos pueden dividirse en *pasivos* y *activos*.

1. *Pasivos*: Utiliza puntos de referencia sobre el piso o las paredes que el vehículo detecta para tomar una ruta programada previamente. Estos puntos pueden ser sensores, cintas magnéticas, cables, imanes e incluso láser.

2. *Activos*: Se basa en componentes electrónicos que emiten señales con información que es detectada e interpretada por el vehículo para tomar una decisión.

En la siguiente sección 1.3, abordaremos nuestra visión *SAR* de los algoritmos de encaminamiento adaptativos mínimos y el problema que abordamos. Explicaremos la motivación para abordar este problema y las estrategias existentes para abordarlo.

1.3. Visión SAR de los algoritmos de encaminamiento

El proceso del encaminamiento de un mensaje u objeto consiste en la selección de una ruta desde un origen hasta un destino. Los bloqueos en el encaminamiento de mensajes u objetos, emergen como consecuencia de una mala selección de los canales o segmentos, que son compartidos por diversos procesos de forma concurrente para mover los mensajes u objetos desde un origen a un destino en la red o plataforma de trabajo. Esto nos lleva a un tipo de sistema ampliamente conocido y estudiado llamado *Sistema de Asignación de Recursos (SAR)* [Colom 03] o por sus siglas en inglés *RAS* de *Resource Allocation Systems*.

Un *SAR* engloba un conjunto de procesos que siguen un plan predefinido en el que los procesos compiten por los recursos que son finitos. Desde la visión *SAR* se pueden agrupar los procesos según su estructura en *secuencial* o *no–secuencial*. En los procesos no–secuenciales, algunas partes se ejecutan en paralelo, lo cual no es el caso del encaminamiento de mensajes u objetos. El proceso del encaminamiento es siempre secuencial y se utilizan recursos en cada estado, que son compartidos en exclusión mutua con otros procesos. Esta relación de competición por los recursos entre los procesos puede causar problemas de bloqueos, los cuales son un fenónemo indeseable en el encaminamiento porque pueden detener todo el sistema. Existen diversas formas de abordar el problema de los bloqueos en los algoritmos de encaminamiento [Warnakulasuriya 97] y en nuestro caso desarrollaremos una metodología para la obtención de algoritmos de encaminamiento adaptativos mínimos, siguiendo el enfoque *SAR* presentado en [Colom 03] al que llamaremos *Visión de Asignación de Recursos*. El problema del bloqueo en el encaminamiento también ha sido estudiado desde la perspectiva *SAR* en los *Sistemas de Manufactura Flexible (FMS en inglés)* [Park 00a] debido a que existen planes de procesos y selección de rutas para encaminar los objetos. En términos generales, consideramos que un objeto *(mensaje, parte, AGVs, etc)* es encaminado en un sistema que tiene un conjunto de posibles rutas predefinidas, en donde para seguir una ruta el objeto pasa por un conjunto de estados.

Bajo esta visión un objeto se desplaza desde desde un estado inicial llamado origen, hasta un estado final llamado destino y para avanzar utiliza recursos de forma unitaria y secuencial. Adicionalmente, al existir encaminamiento adaptativo los procesos pueden tomar diferentes rutas para alcanzar su destino, pero siempre en una ruta mínima. Estas rutas son seleccionadas por el algoritmo de encaminamiento y podemos tener estados que se repitan, pero que llegaron a ellos por rutas diferentes, es decir que han utilizado recursos diferentes para llegar. El término *recurso* se usará en un sentido amplio de la palabra y se asociará al medio de cualquier tipo *(físico o virtual)* que en caso de necesidad, sirve para conseguir lo que se pretende.

Los recursos pueden clasificarse según muchos criterios pero basándonos en su comportamiento en el sistema los podemos agrupar en: *consumibles* o *reutilizables*.

1. *Consumible*: Es aquel que se destruye al ser adquirido por un proceso, por ejemplo: los paquetes de datos, la información de los buffers de entrada/salida. Esta clase de recursos se puede crear o bien destruir y no existe un límite para su existencia.

2. *Reutilizables*: Es aquel que no se destruye o se desgasta con su uso, como por ejemplo: un canal de entrada/salida o una zona de memoria, una impresora. Estos recursos solo los puede utilizar de forma segura un proceso en cada momento.

Siguiendo nuestra visión *SAR*, los canales y segmentos se considerarán como los recursos, que se utilizan de una manera conservativa *(que no se crean, ni se destruyen)* por los procesos. Los procesos en este caso es el flujo de los objetos desde un origen hasta alcanzar el destino. Cada proceso solicita los recursos de forma secuencial y en ese mismo orden los libera. Es posible que los recursos puedan tener una o múltiples instancias de si mismo, y además los recursos pueden ser compartidos por múltiples procesos, aunque usados en exclusión mutua. Adicionalmente, la existencia de múltiples instancias de un recurso da la posibilidad de que existan multiples rutas de igual distancia para llegar a un destino. Considerar esta situación incrementa la complejidad del problema del encaminamiento debido a que debemos evaluar que las solicitudes de un recurso no excedan las existencias. Es importante hacer notar que la repetición de estados similares genera una explosión de estados que solo puede ser abordada utilizando técnicas estructurales sobre los modelos para así evitar procesos costosos y muchas veces irrealizables. Finalmente, siguiendo la visión *SAR*, la construcción del modelo global del sistema se logrará por la composición de las diversas rutas, que en nuestro caso están organizadas por destinos. Es decir; para cada destino existirá un conjunto de rutas predefinidas que utilizan recursos y generan procesos de encaminamiento. Los diversos destinos tendrán en común la utilización de recursos en común, por los que los procesos competirán hasta utilizarlos de forma conservativa. Nuestro enfoque *SAR*, utilizado en nuestra metodología será abordado en detalle en el capítulo 2.

1.3.1. Motivación del problema

El interés de optimizar las prestaciones del sistema genera una competencia por los recursos que puede producir situaciones indeseadas en el sistema. Desde nuestra visión *SAR* del sistema, el encaminamiento de un objeto se realiza por la asignación de los recursos necesarios para su avance. El algoritmo de encaminamiento selecciona estos recursos que llevarán el objeto desde un origen hasta un destino.

La incorrecta selección de la ruta, por parte del algoritmo de encaminamiento genera problemas de bloqueos que es algo catastrófico para todo el sistema. Si durante el encaminamiento ocurre una utilización indebida de los recursos existentes, y estos recursos no son liberados oportunamente, puede generar una espera en otros procesos que solicitan los mismos recursos. Si esta espera se hace infinita en el tiempo entre un grupo de procesos en donde cada uno requiere un recurso que está siendo usado por otro del mismo grupo, entonces estaremos ante un problema ampliamente conocido y estudiado durante los últimos cuarenta años y llamado como el problema del **bloqueo**.

Los bloqueos en los algoritmos de encaminamiento son difíciles de detectar y aún existiendo la posibilidad de que se produzcan, el sistema puede sobrevivir durante largo tiempo sin que estos ocurran. Un bloqueo en el sistema es una interrupción permanente al procedimiento de asignación y liberación de un recurso. La figura 1.4, muestra estos eventos que son 1) Solicitud por parte de un proceso. 2) Asignación por del sistema operativo o software de administración. 3) Liberación del recurso por el proceso.

Figura 1.4: Eventos durante la asignación de un recurso.

La primera referencia al término bloqueo *(deadlock)* la encontramos en [Dijkstra 68], la cual incluyó procedimientos para su tratamiento. Este trabajo fue la base para el trabajo de [Coffman 71] que presentó las cuatro *condiciones necesarias para la existencia de un bloqueo*. Es preciso recordar que este problema emergió en los sistemas informáticos cuando los procesos cooperaban entre ellos, compartiendo los mismos recursos. Esto obedece a que la mejor utilización de los recursos es una demanda permanente que tiene mayor presencia en las ciencias de la computación, debido a que es comun paralelizar procesos en tareas más pequeñas. De la misma forma, los sistemas de bases de datos, enfrentan problemas de bloqueos porque acceden de forma concurrente muchos usuarios a la información almacenada. Estos accesos simultáneos pueden permitir la lectura, modificación o eliminación de la información, lo cual si no controlamos debidamente podría generar problemas que afectan la consistencia de la información. Finalmente los casos mas tangibles en donde ocurren bloqueos, nos lo proporcionan los sistemas de transporte terrestres, ferreviarios y aereos, en donde no es extraño escuchar de accidentes trágicos, debido a la mala planificación en la asignación de los recursos.

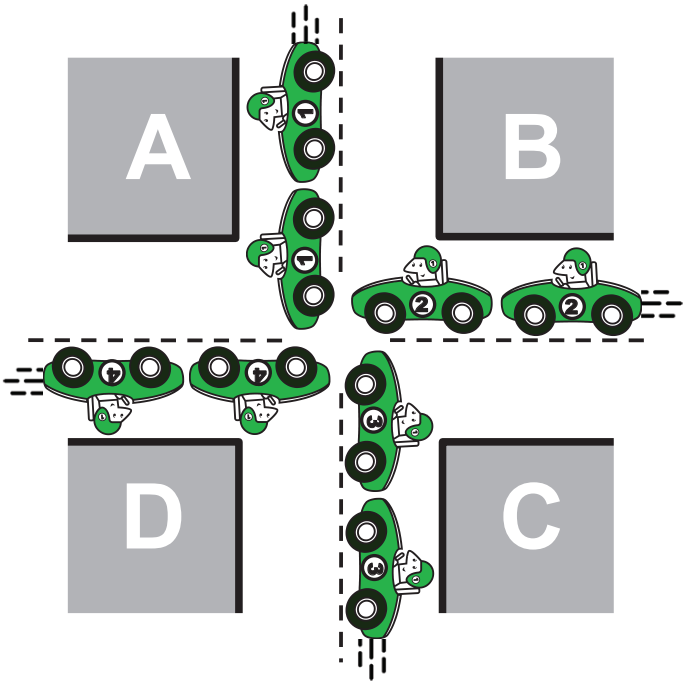

Figura 1.5: Ejemplo de un bloqueo.

A continuación se muestra de manera resumida un ejemplo de un bloqueo que ocurre en un sistema de transporte de automóviles y que está representado por la figura 1.5. En este ejemplo se puede ver cuatro zonas llamadas *A,B,C y D* que tienen a su alrededor las vías por donde circulan ocho automóviles numerados desde el número uno al número cuatro *(1,2,3 y 4)*. Estos automóviles viajan hacia adelante en dirección directa y llegan simultáneamente al cruce de las vías que pertenecen a las zonas. Los automóviles representan a los objetos *(procesos)* que demandan las zonas *(recursos)* que se encuentran ocupadas. Claramente se puede observar que ningún vehículo puede avanzar debido a que existe un ciclo de espera entre las zonas. En el cuadro 1.1 se muestra el recurso que está utilizando cada proceso y el que está demandando. Obviamente las zonas son suficiente para satisfacer las necesidades de los recursos pero debe existir un *algoritmo de encaminamiento* que coordine la debida asignación de los mismos. Para coordinar la correcta asignación de los recursos por el algoritmo de encaminamiento, se utilizan las *políticas de control* de los bloqueos. Nuestra metodología, utilizará el modelo de la Red de Petri para detectar la existencia de bloqueos, a través de una caracterización estructural de la vivacidad del modelo. Adicional, brindará políticas de control necesarias para la obtención de algoritmos de encaminamiento adaptativos mínimos libres de bloqueos.

Cuadro 1.1: Resumen del Bloqueo de la figura 1.5

Número de Autómovil	Zona Ocupada	Zona Demandada
1	A	D
2	B	A
3	C	B
4	D	C

Es preciso mencionar que existe una patología en los sistemas llamada *bloqueo activo (livelock)* y emerge cuando existen rutas no mínimas para llegar al destino. Puede entonces ocurrir que un objeto podría estar viajando alrededor del destino, sin llegar nunca a alcanzarlo. Esta situación se soluciona estableciendo solo rutas mínimas para alcanzar los destinos.

1.3.2. Tratamiento del bloqueo en los algoritmos de encaminamiento

Los métodos utilizados para controlar la aparición de los bloqueos en los algoritmos de encaminamiento es similar a los utilizados en otros dominios de aplicación que también enfrentan problemas de bloqueos en la asignación de los recursos. Para el tratamiento del bloqueo se requiere conocer información de los procesos, porque no existe una solución general, sin tomar en cuenta las demandas de recursos por parte de los procesos. El trabajo más relevante sobre la caracterización y tratamiento del bloqueo es [Coffman 71] debido a que estableció cuatro *condiciones necesarias* que se presentan cuando se alcanza el bloqueo. En base a este trabajo, el tratamiento del bloqueo se centró en *prevenir* o *evitar* que se dieran alguna de las cuatro condiciones que señalaba [Coffman 71]. Seguidamente mencionaremos las cuatro condiciones.

1. *Exclusión mutua*: Los recursos implicados deben usarse en exclusión mutua. Estos recursos solo los puede usar un proceso a la vez.

2. *Retención y espera*: Cuando no se puede satisfacer la petición de un proceso este se queda bloqueado a la espera de los recursos demandados, manteniendo los recursos que tenía previamente asignados.

3. *No expropiación*: No se deben expropiar los recursos. Un proceso solo debe liberar los recursos por voluntad propia.

4. *Espera circular*: Existe una lista cerrada de procesos, de tal manera que cada proceso posee al menos un recurso necesitado por el siguiente proceso de la lista.

Estas cuatro condiciones son la base para la construcción de métodos para controlar la aparición de bloqueos. Para detectar los bloqueos podemos utilizar métodos de dos tipos.

1. Método estático: Para detectar la aparición de los bloqueos se utilizará un modelo del sistema real y el análisis se realizará previamente sobre este modelo.

2. Método dinámico: La detección dinámica del bloqueo ocurre durante la evolución del sistema real. Se verificará el estado actual de los procesos y los recursos para saber si el sistema ha entrado en estado de bloqueo o no.

En nuestro caso utilizaremos un método estático, sobre un modelo formal que modela el sistema real. Una vez detectada la presencia de bloqueos, se debe atentar contra alguna de las cuatro condiciones establecidas por [Coffman 71]. Atentar contra estas condiciones, es lo que denominaremos *políticas de control* para el tratamiento de los bloqueos. Los métodos utilizados para controlar los bloqueos en los algoritmos de encaminamiento difieren poco de los utilizados en otros dominios de aplicación. La mayoría de la literatura que aborda el problema del bloqueo [Taktak 08][Peterson 89][Stallings 05], coinciden en clasificar las estrategias para tratar los bloqueos de la siguiente forma:

1. *Prevención*: Busca negar directamente alguna de las condiciones previas necesarias para la ocurrencia del bloqueo. Esto se logra modificando la estructura de los procesos o bien las condiciones para la asignación de los recursos.

2. *Evitación o Predicción*: Aunque esta técnica previene el bloqueo no modifica la estructura de los procesos. Se busca en avance de cada proceso asegurar que se tenga un estado seguro. Es evitar que se caiga en el estado de no retorno.

3. *Detección y Recuperación*: Intenta de forma dinámica detectar cuando el sistema ha entrado en estado bloqueo. Para la recuperación se busca anular forzosamente las condiciones del bloqueo utilizando técnicas de expropiación de los recursos o bien cancelación de los procesos. Esta es la mas traumática de las técnicas para tratar un bloqueo.

4. *Ignorar*: Esta técnica es la mas optimista respecto al bloqueo, porque ignora el problema. Es usada cuando se sabe que los bloqueos no son significativos o bien su tratamiento es más costoso que el mismo problema. Esta técnica es ampliamente utilizada en muchos sistemas operativos modernos.

1.4. Las redes de Petri para SAR

El desarrollo de este trabajo se concentrará en modelar los algoritmos de encaminamiento vistos desde una perspectiva *SAR*, utilizando las redes de Petri como una herramienta formal de modelado, análisis y síntesis del sistema. Las redes de Petri constituyen uno de los formalismos más aceptados en las aplicaciones de ingeniería en las que se tienen modelos de eventos discretos. Las redes de Petri son un campo de investigación activo en el que se han desarrollado subclases de redes de Petri para aprovechar las características propias y razonar sobre ellas. Las redes de Petri son una excelente herramienta para establecer un modelo y estudiar sus propiedades como la vivacidad. Una vez verificadas las propiedades del modelo se puede obtener una solución formal a los problemas encontrados, siendo entonces trasladados al sistema original que en nuestro caso se traducen como modificaciones al algoritmo de encaminamiento. Las redes de Petri fueron inventadas por el alemán *Karl Adam Petri* en su tesis doctoral titulada Comunicación con autómatas *(kommunikation mit automaten, en alemán)*, en donde estableció la teoría fundamental para representar distintos sistemas informáticos, industriales, de transporte, químicos, etcétera. Una red de Petri se representan por uno grafo bipartito orientado que está compuesto por dos tipos de nodos llamados *lugares* y *transiciones* y comunmente se referencia a sus siglas en inglés como P=Places y T=Transitions. Los arcos conectan un lugar a una transición o una transición a un lugar, pero no pueden conectar a dos del mismo tipo. Los arcos se etiquetan con un número entero *(salvo clases particulares)* que se denomina *peso del arco* y si no se etiqueta, se asume la unidad como valor. En el *Anexo 3* encontraremos información general sobre la notación, propiedades y características de las redes de Petri.

1.4.1. La clase de redes de Petri S^4PR

La clase S^4PR [Tricas 03] es adecuada para el modelado de una amplia variedad de sistemas de asignación de recursos. Esta clase atrae significativamente la atención de los investigadores en estos temas, por las características especiales que poseen. Con esta clase es posible el estudio de sistemas modelados desde una perspectiva estructural y así aprovechar la eficiente caracterización de los estados de bloqueos para esta clase de redes. Mediante esta clase de redes S^4PR, podemos modelar sistemas en los cuales los procesos pueden decidir entre rutas de ejecución alternativas a lo largo de todo un conjunto de rutas existentes para un destino. En este apartado se define formalmente la clase de red S^4PR, se presentan algunas de las características importantes de la misma y las propiedades abstractas que ésta posee. Finalmente, se presenta la caracterización de la vivacidad de la clase desde una perspectiva comportamental hasta su caracterización estructural.

Definición 1 ([Tricas 03])**.** La clase de red S^4PR.
Sea $I_{\mathcal{N}} = \{1,2,...,m\}$ un conjunto finito de índices. Una red S^4PR es una Red de Petri pura, generalizada y conexa, $\mathcal{N} = \langle P, T, \mathbf{C}\rangle$, donde:

1. $P = P_0 \cup P_S \cup P_R$ *es una partición tal que:*
   a) $P_S = \bigcup_{i\in I_{\mathcal{N}}} P_{S_i}$, $P_{S_i} \neq \emptyset$, $P_{S_i} \cap P_{S_j} = \emptyset$, $\forall i \neq j$.
   b) $P_0 = \bigcup_{i\in I_{\mathcal{N}}} \{p_{0i}\}$.
   c) $P_R = \{r_1, \ldots, r_n\}$, $n > 0$.
2. $T = \bigcup_{i\in I_{\mathcal{N}}} T_i$, $T_i \neq \emptyset$, $T_i \cap T_j = \emptyset$, *para todo* $i, j \in I_{\mathcal{N}}$ *tal que* $i \neq j$.
3. *Para todo* $i \in I_{\mathcal{N}}$, *la subred* $\mathcal{N}_i$ *generada por* $P_{S_i} \cup \{p_{0i}\} \cup T_i$ *es una máquina de estados fuertemente conexa, en la que cada ciclo contiene* p_{0i}.
4. *Para cada* $r \in P_R$ *existe un p–semiflujo mínimo,* $\mathbf{y}_r \in \mathbb{N}^{|P|}$, *tal que* $\{r\} = ||\mathbf{y}_r|| \cap P_R$, $\mathbf{y}_r[r] = 1$, $P_0 \cap ||\mathbf{y}_r|| = \emptyset$, *y* $P_S \cap ||\mathbf{y}_r|| \neq \emptyset$.
5. $P_S = \bigcup_{r\in P_R} (||\mathbf{y}_r|| \setminus \{r\})$.

En la definición anterior a los lugares pertenecientes al conjunto P_S se les denomina *lugares proceso* y modelan los cambios de estado durante el flujo del objeto, desde un origen a un destino. Los lugares p_{0i} perteneciente al conjunto P_0 y denominado *lugar de reposo* representan los estados inactivos de los procesos. Para representar los recursos en el modelo usaremos los *lugares recurso* pertenecientes al conjunto P_R. En la definición anterior es importante destacar la existencia de máquinas de estados fuertemente conexas, $\mathcal{N}_i$, (definición 1.3) en las cuales cada una modela un destino. A continuación se definirá el *marcado inicial aceptable* [Tricas 03] para completar la definición anterior. La inactividad del sistema o estado inicial está representado a través de este marcado, el mismo permite ejecutar aisladamente cada proceso secuencial.

Definición 2 ([Tricas 03])**.** *Sea $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C}\rangle$ una red S^4PR. Un marcado inicial $\mathbf{m_0}$ es aceptable para $\mathcal{N}$ si, y solo si:*

1. $\forall i \in I_{\mathcal{N}}$, $\mathbf{m_0}[p_{0i}] > 0$.
2. $\forall p \in P_S$, $\mathbf{m_0}[p] = 0$.
3. $\forall r \in P_R$, $\mathbf{m_0}[r] \geq \max\{\mathbf{y}_r[p] \mid p \in ||\mathbf{y}_r|| \setminus \{r\}\}$.

En la definición anterior, *el marcado inicial de* p_{0i} (condición 1): representa el número máximo de objetos destinados concurrentemente al mismo nodo o la misma estación destino. El *marcado inicial de los procesos p* (condición 2): indica que no existe ningún proceso activo en el sistema (es decir, en un estado intermedio de procesamiento) en su estado inicial. *El marcado inicial de los recursos r* (condición 3): indica la capacidad máxima que poseen los recursos.

Dado un recurso r y basado en los p–semiflujos mínimos $\mathbf{Y}_r$ el *conjunto de usuarios*[1] del recurso r se definen como el conjunto de los lugares proceso que pueden utilizar dicho recurso.

Definición 3. *Sea* $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ *una red* S^4PR. *Sea* $r \in P_R$. El conjunto de usuarios de r *es el soporte del p–semiflujo mínimo* $\mathbf{Y}_r$ *sin el lugar* $r : \mathscr{H}_r = ||\mathbf{Y}_r|| \setminus \{r\}$. *Esta definición puede extenderse de manera natural a los conjuntos de recursos* $A \subseteq P_R$: $\mathscr{H}_A = \bigcup_{r \in A} \mathscr{H}_r$.

1.4.2. Caracterización de la vivacidad

En los sistemas es importante garantizar la propiedad de *vivacidad*, porque así se tiene la certeza que desde cualquier estado alcanzable, todos las procesos de encaminamiento pueden llegar a realizarse. Utilizando los resultados obtenidos del análisis de la Red de Petri, podemos determinar la existencia o no de bloqueos mediante la propiedad de vivacidad en esta clase de redes. Es por esta razón que la propiedad de vivacidad es de gran significación para hacer una validación funcional del modelo.

El siguiente teorema presenta la caracterización de la vivacidad para la clase de red S^4PR la cual es completamente comportamental. Dado un marcado $\mathbf{m}$ en una red S^4PR, una transición t está habilitada por procesos en $\mathbf{m}$ (está deshabilitada por procesos en $\mathbf{m}$) si, y sólo si, está (no–está) habilitada por sus lugares proceso de entrada, y está habilitada por recurso en $\mathbf{m}$ (está deshabilitada por recurso en $\mathbf{m}$) si, y sólo si, todos *(alguno de)* sus lugares recurso de entrada tienen (no–tiene) suficientes marcas para su disparo.

Teorema 1 ([Tricas 03])**.**
Sea $\langle \mathcal{N}, \mathbf{m_0} \rangle$, $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ *una* S^4PR *admisiblemente marcada. El sistema es no–vivo si y solo si existe un marcado* $\mathbf{m} \in \mathrm{RS}(\mathcal{N}, \mathbf{m_0})$ *tal que el conjunto de transiciones habilitadas por proceso en* $\mathbf{m}$ *es no–vacio y cada una de esas transiciones está deshabilitada por recurso en* $\mathbf{m}$.

[1] Holders, $\mathscr{H}_r$, en terminología inglesa.

El Teorema 1 relaciona la no-vivacidad a la existencia de un marcado en donde los procesos activos están bloqueados. Las transiciones de salida de estos procesos necesitan unos recursos que no se encuentran disponibles y que además en el futuro tampoco se encontrarán disponibles. Esto sugiere que existe una espera circular entre un conjunto de procesos bloqueados. Este concepto de espera circular es capturado en el modelo por la existencia de un cerrojo (causa estructural para la no vivacidad en términos de redes de Petri) cuyos lugares recurso son los lugares que previenen el disparo de las transiciones de los lugares proceso habilitados.

El siguiente teorema caracteriza la no-vivacidad en términos de los cerrojos estableciendo un puente entre la estructura del modelo en redes de Petri y su comportamiento. Un cerrojo D es un conjunto de lugares tal que el conjunto de sus transiciones de entrada está contenido en el conjunto de sus transiciones de salida ${}^\bullet D \subseteq D^\bullet$. Si los lugares del cerrojo se vacían, éstos permanecerán vacíos para siempre. Debido a esto, todas las transiciones de salida de los lugares del cerrojo vacío estarán muertas *(no sensibilizadas)* para siempre porque al menos un lugar de entrada, que pertenece al cerrojo estará vacío para siempre.

Teorema 2 ([Tricas 03])**.**
Sea $\langle \mathcal{N}, \mathbf{m_0} \rangle$, $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ *una* S^4PR *admisiblemente marcada. La red es no–viva si y solo si existe un marcado* $\mathbf{m} \in \mathrm{RS}(\mathcal{N}, \mathbf{m_0})$ *y un cerrojo D tal que* $\mathbf{m}[P_S] > 0$ *y el disparo de cada transición habilitada por proceso en* $\mathbf{m}$ *es impedido por un conjunto de lugares recurso pertenecientes a D. Además, el cerrojo D es tal que:*

1. $D_R = D \cap P_R = \{r \in P_R | \exists t \in r^\bullet$ *tal que* $\mathbf{m}[r] < \boldsymbol{Pre}[r,t]$ *y* $\mathbf{m}[{}^\bullet t \cap P_S] > 0\} \neq \emptyset$*;*
2. $D_S = D \cap P_S = \{p \in \mathscr{H}_{D_R} | \mathbf{m}[p] = 0\} \neq \emptyset$*;*

Un cerrojo D y un marcado $\mathbf{m} \in \mathrm{RS}(\mathcal{N}, \mathbf{m_0})$ que verifican las propiedades del Teorema 2 se dirá que son un *cerrojo malo* y un *marcado D–bloqueado*, respectivamente.

Definición 4 ([Tricas 03])**.** *Sea* $\langle \mathcal{N}, \mathbf{m_0} \rangle$, $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ *una* S^4PR *marcada. Sea D un cerrojo de* $\mathcal{N}$*. Entonces,* $\mathscr{T}h_D = \mathscr{H}_{D_R} \setminus D_S$ *es el conjunto de ladrones de* D[2].

La anterior definición representa lugares de la red que usan recursos de los cerrojos y que no pertenecen a ellos. La siguiente caracterización de la vivacidad establece que cuando una S^4PR es no viva, existe un *marcado D–bloqueado* tal que todos sus procesos activos están *robando* marcas del conjunto de recursos de un cerrojo asociado D llamado *cerrojo malo*.

[2] Algunas veces se usa $\mathscr{T}h_{D_R}$ para mostrar la relación entre estos dos conjuntos.

Teorema 3 ([Tricas 03])**.** *Sea* $\langle \mathscr{N}, \mathbf{m_0} \rangle$, $\mathscr{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$, *una* S^4PR *marcada. La red es no–viva si y solo si existe un cerrojo D y un marcado* $\mathbf{m}_D \in \mathrm{RS}(\mathscr{N}, \mathbf{m_0})$ *tal que:*

1. $\mathbf{m}_D[P_S] > 0$.
2. $\mathbf{m}_D[P_S \setminus {}^{\bullet}\mathscr{T}h_D] = 0$.
3. $\forall p \in {}^{\bullet}\mathscr{T}h_{D_R}$ *tal que* $\mathbf{m}_D[p] > 0$, *el disparo de cada* $t \in p^{\bullet}$ *es impedido por un conjunto de lugares recurso pertenecientes a D.*

1.4.3. Ejemplo de modelado y síntesis

En esta subsección se mostrará un ejemplo de un sistema de transporte modelado a través de las redes de Petri como herramienta formal de modelado, análisis y síntesis. En la figura 1.6.a se muestra un sistema de transporte de objetos *(físicos o virtuales)* que está compuesto por tres nodos/estaciones y dos canales bidireccionales denominados C_A y C_B. Se puede deducir que si los nodos/estaciones de los extremos *(1 y 3)* desean enviarse objetos simultáneamente ocurrirá un bloqueo en el nodo intermedio. Es preciso mencionar que asumimos que el nodo/estación 2 está deshabilitado para enviar y recibir mensajes pero habilitado para reenviar los mensajes a los nodos/estaciones restantes.

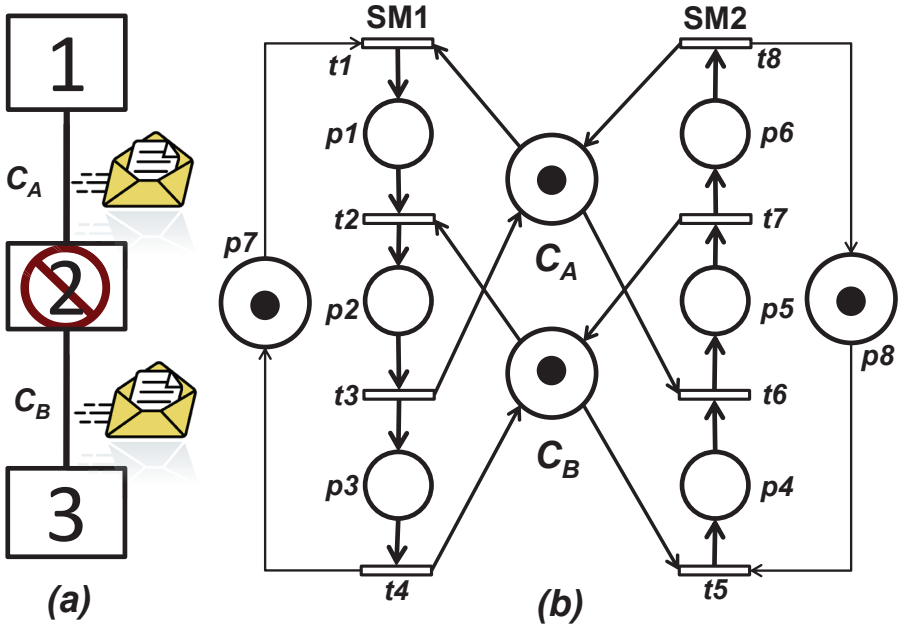

Figura 1.6: Sistema de transporte de objetos modelado por una red de Petri.

La figura 1.6.b muestra una Red de Petri con dos máquinas de estados, en donde la máquina $SM1$, modela el flujo de los objetos desde el nodo/estación 1 hasta el nodo/estación 3. La máquina $SM2$, modela el flujo en sentido inverso, es decir, los objetos desde el nodo/estación 3 hasta el nodo/estación 1. Desde nuestra perspectiva *SAR*, los recursos son los canales del sistema y son representados por los lugares recursos a los que hemos denominado C_A y C_B y tienen una marca que indica su disponibilidad o no.

Esta redes de Petri pertenece a la clase S^4PR [Tricas 03] que es adecuada para el modelado de una amplia variedad de Sistemas de Asignación de Recursos. Como hemos mencionado en el teorema 2, la vivacidad en esta clase de redes está relacionada con la existencia de cerrojos insuficientemente marcados en $\mathbf{m}$. La red de la figura 1.6.b tiene un cerrojo D que está formado por los siguientes lugares $D = \{p_2, p_3, p_5, p_6, C_A, C_B\}$, el cual se desmarca con el marcado $m = p_1 + p_2$. Bajo este marcado las transiciones de salida de los lugares en el cerrojo t_2 y t_6 están muertas y el cerrojo D desmarcado. Evidentemente, existe un bloqueo y el sistema no garantiza la propiedad de vivacidad como así lo demuestra la Red de Petri. Podemos forzar la vivacidad del modelo aplicando políticas de control que se abordarán en el capítulo 4.

1.5. Conclusión y plan de la tesis

En este capítulo se ha planteado el problema de los algoritmos de encaminamiento y se ha definido a los bloqueos como el objetivo a abordar en esta investigación. Se ha presentado lo que consideramos nuestra visión *SAR* de los sistemas tratados, así como la explicación de los dos dominios de aplicación sobre los que trabajamos. Para ambos dominios se ha dado una explicación de los tipos de recursos a considerar y su comportamiento en el sistema. Adicionalmente se ha presentado una taxonomía de los algoritmos de encaminamiento y las diversas técnicas que existen para tratar el bloqueo. Finalmente hemos presentado una breve introducción a la clase de redes de Petri S^4PR dentro de las cuales se enmarcan las redes de Petri a obtener en esta tesis, junto con su caracterización de la vivacidad que se aplica a una red de ejemplo. A continuación presentamos los temas que se abordarán en los capítulos sucesivos.

1. *Capítulo 2*: En este capítulo presentaremos nuestra metodología de modelado formal para los sistemas de encaminamiento incluyendo una detallada explicación de los diversos grafos que asisten al diseñador. Todos los grafos serán presentados juntos a sus respectivos algoritmos utilizados para obtenerlos. Adicionalmente, se desarrollará un ejemplo de una red de interconexión que no está libre de bloqueos en donde obtenemos los grafos indicados por la metodología. A partir de los grafos que indica nuestra metodología se obtiene la red de Petri, que seguidamente

procederemos a interpretar semánticamente los modelos obtenidos en base a nuestra perspectiva *SAR*. La interpretación semántica será de los elementos y propiedades dentro del modelo formal que corresponden al algoritmo de encaminamiento de la red de interconexión y que se utilizará a lo largo de este trabajo. Esta interpretación incluirá las características de los modelos obtenidos así como ciertas restricciones en su estructura. Finalmente, se incluirá otro ejemplo pero aplicado al dominio de aplicación de los vehículos guiados automáticamente. Igualmente, se desarrollará este ejemplo usando nuestra metodología de modelado hasta presentar la Red de Petri obtenida para este ejemplo.

2. *Capítulo 3*: Modelo teórico de la clase de redes de Petri que nos permite modelar y analizar sistemas de encaminamiento y posteriormente controlar el problema de los bloqueos en estos algoritmos de encaminamiento. En este capítulo presentaremos una caracterización más precisa de la clase de redes de Petri que obtenemos de la abstracción de los sistemas de encaminamiento a través de nuestra metodología. Está clase será definida por sus propiedades estructurales y nos apoyaremos en la caracterización de la vivacidad de las redes S^4PR para garantizar la vivacidad del modelo a través de nuestra política de control.

3. *Capítulo 4*: En este capítulo se sintetiza controles sobre la red de Petri que resuelven el problema del bloqueo para nuestro modelo de redes. Este objetivo se logra con la introducción de recuros virtuales que pueden modificar la estructura de los procesos y permite mayor flexibilidad a los mismos. Para asegurar que la red es viva se verificará que no existan cerrojos malos en la red a través de la aciclicidad del grafo de poda de recursos, que se apoya en la relación de poda entre zonas que se definirá. Finalmente, el método de corrección se presentará aplicado al dominio de las redes de interconexión.

4. *Capítulo 5*: En este capítulo presentaremos nuestras conclusiones y las aportaciones de la investigación.

Capítulo 2

Modelado de algoritmos de encaminamiento adaptativos mínimos

Resumen
En este capítulo presentamos una metodología de modelado formal para el diseño de algoritmos de encaminamiento adaptativos mínimos libres de bloqueos. En la metodología utilizaremos las Redes de Petri como herramienta formal para el modelado, análisis y síntesis de los algoritmos de encaminamiento adaptativos mínimo. También se hará uso de diversos grafos intermedios para guiarnos sistemáticamente en la metodología. La obtención de los grafos estará soportada por algoritmos especializados para tal fin. También se presentará la interpretación semántica del modelo desde nuestra perspectiva *SAR*, así como la caracterización de la clase de modelo de Red de Petri obtenido con la metodología. Esta metodología será explicada en detalle utilizando ejemplos en dos dominios de aplicación como son las *redes de interconexión* y los *vehículos guiados automáticamente*.

2.1. Introducción

El diseño de algoritmos de encaminamiento es una tarea compleja que requiere la asistencia de métodos formales de modelado para determinar de forma precisa si el algoritmo de encaminamiento es correcto y se comporta como se esperaba. El modelado formal es la acción de construir un *modelo formal* que es una abstracción del sistema

para describir su comportamiento. El modelo formal permite abordar determinados procesos para detectar problemas dentro del sistema, mientras que se omiten detalles que no son relevantes para describir los aspectos a estudiar del sistema. El modelado formal de algoritmos de encaminamiento es un elemento determinante para su buen diseño y se realiza para detectar problemas, tales como la existencia de bloqueos o estimar índices de rendimiento de los mismos. Modelar los algoritmos de encaminamiento para tratar estos problemas es de gran interés en diversas áreas de la ingeniería como *redes de ordenadores, vehículos automáticamente y sistemas de manufactura flexible* entre otras. Hoy en día los métodos para modelar algoritmos de encaminamiento están apoyados en la utilización de grafos como [Dally 87][Duato 93][Duato 95b][Taktak 08], sin embargo carecen de procedimientos metódicos de corrección. Adicionalmente estos métodos modelan la asignación de los recursos y solo capturan una imagen temporal *(snapshot)* del sistema como lo es el *Grafo de Asignación de Recursos (GAR)* [Peterson 89][Stallings 05], *Grafos de Dependencias (GD)* [Dally 87] o el *Grafo de Espera (WFGs en inglés)* [Warnakulasuriya 97]. Estos métodos impiden hacer una predicción sobre los nuevos estados del sistema y se ven forzados a esperar una nueva evolución del sistema para aplicar la misma técnica. Contrariamente nuestra metodología permite la obtención de algoritmos de encaminamiento adaptativos mínimos libres de bloqueos, obteniendo condiciones necesarias y suficientes para la existencia de los bloqueos. En términos generales nuestra metodología proporciona procedimientos sistemáticos para la *abstracción, análisis* y *síntesis* de los algoritmos de encaminamiento adaptativos mínimo como lo muestra gráficamente la figura 2.1.

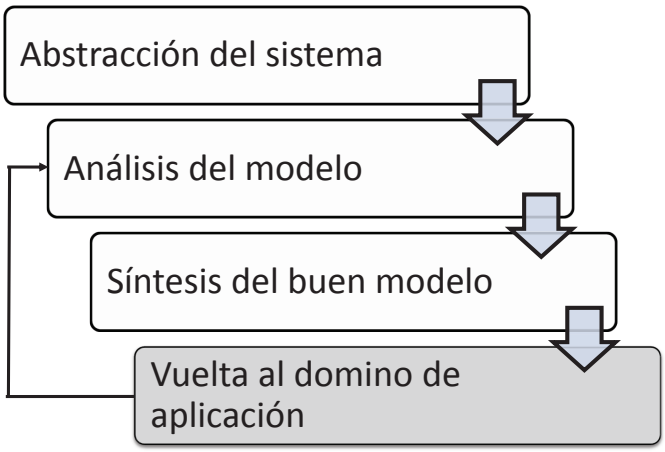

Figura 2.1: Pasos generales de una metodología de diseño.

En la *abstracción* se obtiene un modelo formal del sistema bajo estudio, sobre el cual se realizará un *análisis* en busca de propiedades de buen comportamiento. El análisis se realizará a través de las Redes de Petri que es una herramienta formal de modelado y permite utilizar métodos de *síntesis* para abordar el problema de los bloqueos. Las correcciones obtenidas durante la síntesis del modelo se trasladarán al dominio de aplicación para hacer los cambios necesarios y así garantizar las propiedades deseadas. Es posible que se requiera iterar para garantizar las propiedades de buen comportamiento por lo tanto la metodología es cíclica, sin embargo siempre existirá una condición de parada. La condición de parada está dada cuando se garantiza la propiedad de vivacidad del modelo de la Red de Petri, por lo tanto se elimina las condiciones que nos conducen a bloqueos en el sistema.

Las redes de Petri [Petri 62][Murata 89] son un conocido formalismo matemático y gráfico, para representar sistemas de eventos discretos con sincronizaciones describiendo las relaciones existentes de causa y efecto entre los eventos. Utilizando modelos basados en Redes de Petri nos permite buscar propiedades que caracterizan el sistema y se pueden dividir en las propiedades *Cuantitativas* y *Cualitativas*. El análisis de propiedades cuantitativas es ampliamente utilizado en la evaluación de prestaciones de los sistemas [Marsan 87] [Chiola 95]. Este análisis nace de la necesidad de cuantificar los índices de rendimiento de los sistemas para determinar si el sistema alcanzará los niveles de prestaciones para los cuales fue diseñado. Por otra parte en el análisis de propiedades cualitativas se realiza con la finalidad de hacer una validación funcional del modelo, siendo la propiedad de la *vivacidad* el objetivo principal en nuestro análisis. A través de la vivacidad se garantiza que desde cualquier estado alcanzable todas las actividades pueden llegar a realizarse. Esta propiedad abstracta es de gran importancia y significación en el contexto de los algoritmos de encaminamiento porque garantiza que los procesos de encaminamiento de mensajes u objetos no se bloquearán. Un modelo de Red de Petri está compuesto de una parte *estática* que es su estructura y una parte *dinámica* que es la evolución del marcado. En el análisis estructural se considera fundamentalmente la estructura del modelo y permite deducir las propiedades casi independientemente del marcado inicial. En nuestra metodología, el análisis para detectar la propiedad de la vivacidad se realizará utilizando técnicas estructurales porque permite tratar modelos de mayor tamaño.

En la sección 2.2 de este capítulo presentaremos un ejemplo de una red de interconexión con su respectivo algoritmo de encaminamiento adaptativo mínimo. Este algoritmo no está libre de bloqueos, por lo tanto incluiremos una configuración de bloqueo alcanzada por dicho algoritmo al intentar enviar cuatro mensajes de forma simultánea a cuatro destinos diferentes. En la sección 2.3 haremos un resumen de las aproximaciones existentes para solucionar el problema del bloqueo, incluyendo el modelado del ejemplo

previo. A continuación en la sección 2.4 presentaremos nuestra metodología, la cual utiliza el enfoque *SAR* [Peterson 89] para abordar el problema del diseño de algoritmos de encaminamiento adaptativos mínimos libres de bloqueos. Este enfoque se basa en la compartición de número limitado de *recursos*, por diversos *procesos* de forma concurrente. Por tal motivo desde la primera etapa de metodología, se distingue estos dos elementos del enfoque *SAR*.

En las subsecciones sucesivas se explicará en detalle cada etapa de la metodología, comenzando con la información proporcionada por el diseñador del algoritmo de encaminamiento adaptativo mínimo y los algoritmos especializados para la obtención de los grafos intermedios. Los grafos presentados corresponderán a la abstracción del ejemplo previo hasta conducirnos al modelo formal del sistema en Redes de Petri. Es preciso mencionar que siguiendo la perspectiva *SAR*, en la sección 2.5 asociaremos una interpretación semántica al modelo obtenido. Esta interpretación abarca la definición de conceptos como *recurso, proceso, ruta, conjunto de rutas, proceso de encaminamiento* entre otros. Adicionalmente procederemos a caracterizar estructuralmente estas interpretaciones semánticas dentro del modelo de la Red de Petri. Finalmente, en la sección 2.6 incluiremos un ejemplo de modelado para obtener algoritmos de encaminamiento adaptativos mínimos utilizando los *vehículos guiados automáticamente*. Este dominio de aplicación presenta analogías y diferencias que serán abordadas desde el punto de vista *SAR* de la metodología. Finalmente, en la sección 2.7 presentaremos nuestras conclusiones del capítulo.

2.2. Ejemplo de una red de interconexión

Hoy en día, muchos sistemas electrónicos digitales utilizan las redes de interconexión para las comunicaciones de datos entre procesadores. La fortaleza de estos sistemas es la comunicación paralela entre procesadores por medio de mensajes. Las redes de interconexión han evolucionado con la incorporación de técnicas eficientes en el encaminamiento de los mensajes por la red, lo cual ha provocado un incremento en la velocidad de transmisión y ha reducido los requerimientos operacionales. A continuación introducimos un ejemplo simple de una red de interconexión con su algoritmo de encaminamiento adaptativo mínimo, similar al introducido en el artículo [Dally 93] o en [Duato 95a]. La siguiente especificación ilustra una situación típica en el diseño de un algoritmo de encaminamiento adaptativo mínimo para una red de interconexión. Nuestra red de interconexión está definida por un sistema de nodos *ND* y un sistema de canales *CH* que interconectan los nodos. El patrón de la conexión entre nodos será llamado la *topología* de la red de interconexión. El ejemplo que estamos considerando es un anillo

unidireccional que gira en sentido de las manecillas del reloj como topología subyacente. Existe el conjunto de nodos ND=$\{n_0, n_1, n_2, n_3\}$ y están interconnectados por un conjunto de canales CH=$\{ca_0, ca_1, ca_2, cl_1, cl_2, cl_3\}$. Esta red de interconexión se representa de manera esquemática en la figura 2.2.a.

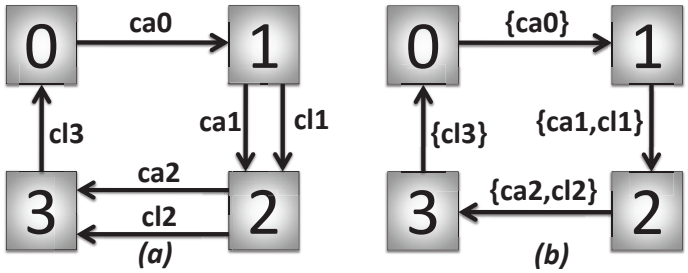

Figura 2.2: Topología y su Grafo de Interconexión en las figuras *a* y *b* respectivamente.

Obsérvese que debido a que estamos tratando con un algoritmo de encaminamiento adaptativo mínimo, si un nodo tiene dos canales de salida un mensaje puede utilizar cualesquiera de ellos, si el *algoritmo local de encaminamiento* así lo permite. El otro elemento de definición de la red de interconexión es el tipo de *control de flujo* que establece el sistema de asignación del canal. El tipo de control de flujo considerado por nosotros es el mecanismo agujero de gusano o mas conocido como **wormhole** en que un mensaje se descompone en uno o más unidades de control de flujo o $flits$. Estos flits fluyen consecutivamente a través de la red, debido a que generalmente un mensaje está compuesto por varios flits. Dado que el almacenamiento intermedio de los nodos solo puede almacenar un flit, un solo mensaje puede mantener reservado simultáneamente varios canales mientras alcanza el nodo destino. La cabeza (**head**) es el primer flit del mensaje y reserva los canales; la cola (**tail**) es el último flit del mensaje y libera el canal que acaba de atravesar y que había sido previamente reservado. Cada canal reservado soporta únicamente un flit en su almacenamiento temporal o $buffer$. Esta restricción obliga a que mensajes de más de un flit de largo ocupen diversos canales simultáneamente. En nuestro ejemplo solo consideramos mensajes compuestos por más de un flit entre los nodos.

Algoritmo de encaminamiento adaptativo mínimo

Cada nodo tiene una identidad local única y ejecuta una instancia del Algoritmo 1 de encaminamiento, el cual es de tipo adaptativo mínimo. El algoritmo de encaminamiento utilizará como parámetro de entrada el nodo destino que se incluye en el primer flit de cada mensaje. La función *destino(m)* obtiene el nodo destino del mensaje que se encuentra

en el primer flit de cada mensaje. Este algoritmo es iterativo y finaliza cuando no existen mensajes en la cola de mensajes del nodo actual. Si es imposible asignar un canal de salida a un mensaje, este es colocado en la cola de mensajes entrantes.

Algoritmo 1 Algoritmo de encaminamiento para el nodo i.

Entrada: El flit de la cabeza m desde la cola de mensajes.
Local: $C_i \subseteq CH$, conjunto de canales de salida para el nodo n_i
$F \subseteq C_i$, conjunto de canales no asignados
Salida: El próximo canal para ser usado por m.

```
1.  Si (destino(m) = i) Entonces
2.    Almacene el mensaje m en n_i
3.  Sino
4.    Si (ca_i ∈ F) Entonces
5.      usar ca_i
6.      F : =F\{ca_i}
7.    Sino
8.      Si ((destino(m) < i) ∧ (cl_i ∈ F)) Entonces
9.        usar cl_i
10.       F : =F\{cl_i}
11.     Sino
12.       colocar en cola el mensaje m
13.     Fin Si
14.   Fin Si
15. Fin Si
```

El Algoritmo 1 es un algoritmo de encaminamiento adaptativo mínimo y se puede expresar informalmente de la siguiente forma. Para llegar a un nodo de destino n_d diferente del nodo actual n_i, el algoritmo intenta asignar primeramente el canal de salida ca_i, si se trata de un canal libre en n_i. De lo contrario se asigna el canal cl_i como canal de salida si el nodo actual n_i satisface que $n_d < n_i$. Esta reserva la realiza el flit cabeza del mensaje al llegar al nodo actual de la red de interconexión y los siguientes flits intermedios seguirán a través de los canales reservados por el flit cabeza. Este proceso ocurre hasta que el flit cola libere todos los canales usados y que por tanto se sumarán al conjunto de canales libres F o no asignados. El diseño de algoritmos de encaminamiento puede llevar a soluciones en el que se alcancen estados de bloqueos en los mensajes. Un estado de bloqueo en una red de interconexión se presenta cuando un conjunto de mensajes en tránsito se detienen para siempre en los nodos intermedios de una red y ninguno de ellos ha alcanzado sus respectivos nodos destino. Todos estos mensajes están a la espera de la disponibilidad de

canales de salida en estos nodos intermedios y que han sido previamente asignados a otros mensajes que pertenecen a este mismo conjunto. Por lo tanto, ninguno de los mensajes implicados alcanzará a sus nodos destino.

El algoritmo de encaminamiento 1 de nuestro ejemplo presenta esta anomalía que se ilustra mediante la siguiente configuración de bloqueo de el cuadro 2.1. En el cuadro representamos cuatro mensajes en una configuración alcanzable porque cada uno está compuesto por más de un flit por lo tanto cada mensaje tendrá asignados simultáneamente más de un canal. En el cuadro 2.1, la columna **Flits** las siglas n_h y n_t representan los nodos actuales del flit cabeza y cola, respectivamente; **Canales asignados**: Canal asignado al mensaje; **Nodo Destino**: representa el nodo de destino del mensaje, **Algoritmo de encaminamiento**: Canal que se asignará al flit cabeza según el algoritmo de encaminamiento.

Número Mensaje	**Flits**		**Canales Asignados**	**Nodo Destino**	**Algoritmo de Encaminamiento**
	n_h	n_t			
ms_1	0	3	cl_3	1	ca_0
ms_2	1	0	ca_0	2	ca_1
ms_3	2	1	ca_1	3	ca_2
ms_4	3	2	ca_2	1	cl_3

Cuadro 2.1: Estado de bloqueo alcanzado en el ejemplo de cuatro mensajes.

Observe que la configuración presentada en el cuadro 2.1, es alcanzada por la red de interconexión de la figura 2.9.a, en donde todos los mensajes están en nodos intermedios, por lo tanto no han alcanzado su destino. Para avanzar en la red de interconexión, todos los mensajes necesitan canales *(los propuestos por el algoritmo de encaminamiento)* que todos ellos están ya asignados a otros mensajes en el mismo conjunto. Esto se puede apreciar al comparar las dos columnas **Canales Asignados** y **Algoritmo de Encaminamiento** en el cuadro 2.1. Por otra parte, ninguno de los flits cola puede liberar los canales, porque si alguno de estos flits cola avanza estará en el mismo nodo que el flit cabeza y esto no es posible. Por lo tanto, hemos llegado a un estado de bloqueo en que se cumplen todas las condiciones clásicas [Coffman 71] derivadas de la existencia de un bloqueo. Por último, se puede observar que, si bien estamos en un estado de bloqueo, existen dos canales, cl_1 y cl_2, que están libres pero que el algoritmo de encaminamiento no puede asignar ninguno de los dos a los cuatro mensajes de nuestro escenario.

2.3. Aproximaciones metodológicas existentes

Trabajos pioneros para aplicar métodos para el tratamiento de los bloqueos en los *algoritmos de encaminamiento* los encontramos en las redes de multicomputadores. Dos enfoques son los más utilizados; el enfoque de Dally & Seitz [Dally 87] que se utiliza para los algoritmos de encaminamiento estático y el enfoque de Duato [Duato 93][Duato 95b] para los algoritmos de encaminamiento adaptativos mínimos. Ambas metodologías son incompletas debido a que el diseñador carece de herramientas para modificar el algoritmo original si encuentra problemas de bloqueos. Esencialmente el diseñador propone una topología y un algoritmo de encaminamiento y el método sólo responde si está libre de bloqueos o no. No hay asistencia al diseñador, en el caso de la existencia de bloqueos, acerca de cómo solucionar los problemas y cómo obtener una versión corregida del algoritmo de encaminamiento. Por lo tanto estas metodologías pueden llevar al diseñador a iterar en el proceso de diseño sin encontrar una solución definitiva al problema del bloqueo. Adicionalmente estas metodologías abordan el problema utilizando métodos estáticos de representación como lo grafos para encontrar dependencias entre los recursos.

La utilización de grafos para representar la dependencias entre los recursos fue inicialmente introducido por [Holt 72] y fue adaptado para ser utilizado en la **detección** de bloqueos en los algoritmos de encaminamiento de las redes de multicomputadores. Estas **dependencias** se producen durante asignaciones sucesivas de recursos y se pueden agrupar en *directas* e *indirectas*. Asumiendo que existe un sistema de comunicación representado por un grafo G=(V,E), en donde V es el conjunto de vértices y E el conjunto de arcos entre los vértices. Los vértices representan los orígenes y los destinos de los objetos, que se mueven a través de los arcos. Por tanto los arcos son los *recursos* del sistema y el movimiento de los objetos son los *procesos* que necesitan ser servidos. Dado una ruta $\mathscr{P}$ desde un nodo inicial $n_i \in V$ y un nodo destino $n_d \in V$. Existen dependencias entre los recursos r_i y r_j, si se dan las siguientes condiciones.

1. *Directas*: Si existen los recursos r_i y r_j, ambos adyacentes y pueden ser utilizados en secuencia. Formalmente decimos que existe una ruta tal que $r_i \succ r_j$.

2. *Indirectas*: Es creada por la suposición de que se pueden utilizar simultáneamente varios recursos en un solo proceso. Si existe una ruta $\mathscr{P}$, para llegar a un nodo destino n_d desde un nodo inicial n_i, y el recurso r_i es el primer recurso en la ruta, seguido por un número indeterminado de recursos y finalmente hasta un último recurso r_j. Existe una relación indirecta entre los recursos extremos no adyacentes r_i y r_j, tal que $r_i \neq r_j$ y r_j no es el sucesor inmediato de r_i.

El trabajo de [Dally 87] proporcionó el primer método de *prevención* para obtener algoritmos de encaminamiento *deterministas* libres de bloqueos utilizando el denominado

Grafo de Dependencias entre Canales (GD) y la función de encaminamiento *R*. La *función de encaminamiento R*, describe más formalmente la relación $(mapeo)$ entre el conjunto de canales, que se obtienen del algoritmo de encaminamiento. En la definición 5, se define el *Grafo de Dependencias entre Canales (GD)*.

Definición 5 ([Dally 87])**.** *Un Grafo de Dependencias entre Canales (GD) de una red de interconexión* $I=\langle N,C\rangle$ *y una función de encaminamiento R es un grafo dirigido* $GD=\langle V,E\rangle$. *Los vértices* $V\subseteq C$ *representan los canales de I y los arcos E el par de canales* (c_i, c_j) *conectados por R tal que* $E=\{(c_i,c_j)|R(c_i,n)=c_j$ *para algún* $n\in N\}$. *Los vértices N representan los nodos de procesamiento en I y debido a que los canales no pueden enviarse mensajes a ellos mismos, no existirán ciclos en un mismo nodo.*

Este método se definió para redes de interconexión que utilizan el control de flujo wormhole y puede ser utilizado tanto en topologías regulares como irregulares, que utilizan algoritmos de encaminamiento deterministas. Esta última característica impide que la red de interconexión de la figura 2.2.a y su respectivo algoritmo de encaminamiento de tipo *adaptativo* mínimo pueda ser representado adecuadamente con este método. En el método, la estrategia de detección de los bloqueos se basa en la presencia de ciclos en el *GD* y la estrategia de corrección se basa en la utilización de canales virtuales para remover los ciclos. El algoritmo de encaminamiento para la red de interconexión, será libre de bloqueos si el *GD* es acíclico como se formaliza en el teorema 4.

Teorema 4 ([Dally 87])**.** *Dada una función de encaminamiento R, de una red de interconexión I, está libre de bloqueos si y solo si no existen ciclos en el Grafo de Dependencias entre Canales* (GD).

Si se detectan ciclos en el grafo, éstos deben ser eliminados dividiendo el canal físico en canales virtuales, mientras se mantiene la red conexa. Si estas condiciones se cumplen el algoritmo de encaminamiento determinista *(estático)* es libre de bloqueos. Esta estrategia consiste en la restricción del encaminamiento para prevenir la espera circular indicada por el *GD*. Si existen ciclos en el grafo, significa que existe dependencias cíclicas entre los canales y se puede interpretar que estamos ante un posible bloqueo. El método previene los bloqueos rompiendo esos ciclos por medio de un ordenamiento rígido durante la utilización de los canales en algoritmos de tipo determinista.

Un enfoque similar al de [Dally 87] fue utilizado para algoritmos de encaminamiento adaptativos en [Duato 95a][Duato 95b], y permite que el algoritmo de encaminamiento sea libre de bloqueos aunque tenga ciclos en el *GD* pero no el *Grafo de Dependencias Extendido (GDE)* de la definición 5. Este método permite tener dependencias cíclicas entre los canales y que a pesar de esto, el algoritmo de encaminamiento sea libre de bloqueos, debido a la existencia de *rutas de escape*. El concepto de rutas de escape,

garantiza que siempre que exista un bloqueo, los mensajes tendrán un conjunto de canales disponibles, para *evitar* el bloqueo. La definición del *GDE* incluyó una metodología para la construcción de algoritmos adaptativos libres de bloqueos, sin embargo la denominan metodología sin fin o *Duato's protocol* [Dally 03] porque no revela como aplicar sus ideas.

Teorema 5 ([Duato 03])**.** *Un Grafo de Dependencias Extendido (GDE) entre canales de una red de interconexión I y una sub–función de encaminamiento R_1, de una función de encaminamiento R, es un grafo dirigido GDE=$\langle C_1, E\rangle$. Los vértices C_1 son los canales proporcionados por la sub–función R_1 para algunos destinos. Los arcos E son los pares de canales (c_i, c_j) tal que existe una dependencia directa, indirecta, directa cruzada o indirecta cruzada desde c_i a c_j.*

Adicionalmente a las dependencias directas e indirectas entre los recursos, el *GDE* incluyó dos tipos de dependencias llamadas *directas cruzadas* o las *indirectas cruzadas* [Duato 03]. Estas dependencias se deben a una suposición metodológica del diseñador, que considera una sub-función de encaminamiento R_1, que se incluye dentro de la función de encaminamiento R. Informalmente podemos presentar la sub–función de encaminamiento y su relación con el *GDE* de la siguiente forma. Dado un conjunto de canales C, un nodo origen n_c, un nodo destino n_d, existe una función de encaminamiento R y una sub-función de encaminamiento R_1 que es una restricción a R y proporciona un conjunto de canales $C_1 \subset C$. R_1 se expresa en el grafo de dependencia extendido; debe ser conectada y no debe tener ciclos para que la función R sea libre de bloqueos. El teorema 6 brinda una condición *suficiente* para asegurar algoritmos de encaminamiento adaptativos libres de bloqueos.

Teorema 6 ([Duato 03])**.** *Una función de encaminamiento conectada y adaptativa R, de una red de interconexión I, está libre de bloqueos si existe una sub–función de encaminamiento R_1 que es conectada y no tiene ciclos en el Grafo de Dependencias Extendido (GDE).*

Las rutas que son de R pero no de R_1 se llaman relación complementaria $R_C = R \setminus R_1$ y R_1 es conocida como la *ruta de escape* de los bloqueos. El *GDE* tiene como vértices los canales C_1 y los arcos son las relaciones de dependencias directas e indirectas entre los canales C_1. La figura 2.3 muestra el *GDE* del ejemplo presentado en la sección 2.2 en donde se puede observar que existe más de un ciclo de dependencia entre los canales, por lo tanto el algoritmo de encaminamiento es propenso a bloquearse. Se puede observar que el canal cl_1 no participa en ningún ciclo por lo tanto no formará parte de los estados de bloqueo, sin embargo el canal cl_3 es muy demandado, motivo por el cual participará en los estados bloqueos como es nuestro caso. En este punto es incierto el procedimiento para eliminar de forma sistemática las dependencias entre los canales, por lo tanto se requiere la experiencia e intuición del diseñador [Dally 03].

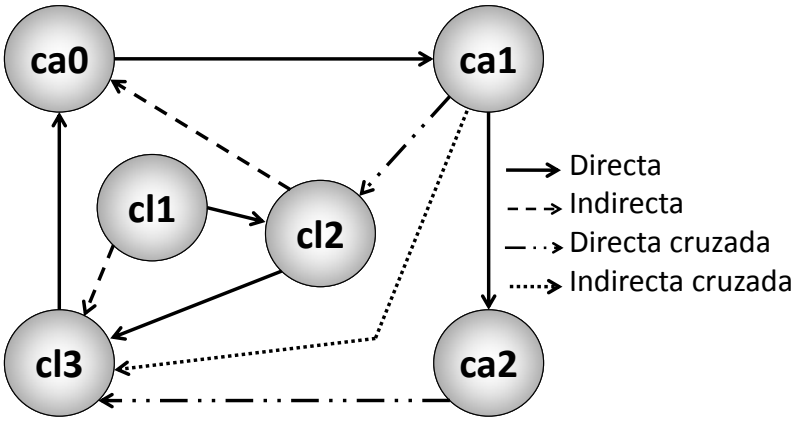

Figura 2.3: Grafo de Dependencias Extendido.

Otro aporte significativo para la creación de algoritmos adaptativos libres de bloqueos fue realizado en [Dally 93], usando canales virtuales a través de la inversión de dimensiones. Igualmente para algoritmos adaptativos de ruta mínima fue realizado un aporte por [Gravano 94] que se aplicaba a toros de n-dimensiones. Finalmente la implementación de técnicas de *recuperación* es poco utilizada en las redes de multicomputadores debido a que detectar el bloqueo en tiempo real es difícil y costoso. Sin embargo, esta técnica se utiliza si no existe otra opción o bien cuando se sabe que es poco probable que ocurra un bloqueo. Los trabajos mas representativos de esta técnica en las redes de multiprocesadores son [Anjan 95][Martínez 97] que utilizan la recuperación por medio de rutas de escape o buffers de almacenamiento en base a [Duato 93]. Es preciso mencionar que las aproximaciones existentes para abordar el problema del bloqueo en las redes de interconexión [Dally 87][Dally 93][Duato 93][Taktak 08], proporcionan solo condiciones para verificar la existencia o no de bloqueos. Esto es una limitación si se desea eliminar el bloqueo y corregir el algoritmo de encaminamiento. A continuación en la sección 2.4, presentaremos nuestra propuesta de metodología de diseño de algoritmos de encaminamiento adaptativos mínimos.

2.4. Propuesta de una metodología de diseño

El diseño de algoritmos de encaminamiento para las redes de interconexión es una tarea que desde el punto de vista metodológico es tediosa y compleja [Dally 03]. En esta sección presentamos nuestra metodología de diseño de algoritmos de encaminamiento adaptativos mínimos, que se muestra en la figura 2.4. En esta metodología el diseñador

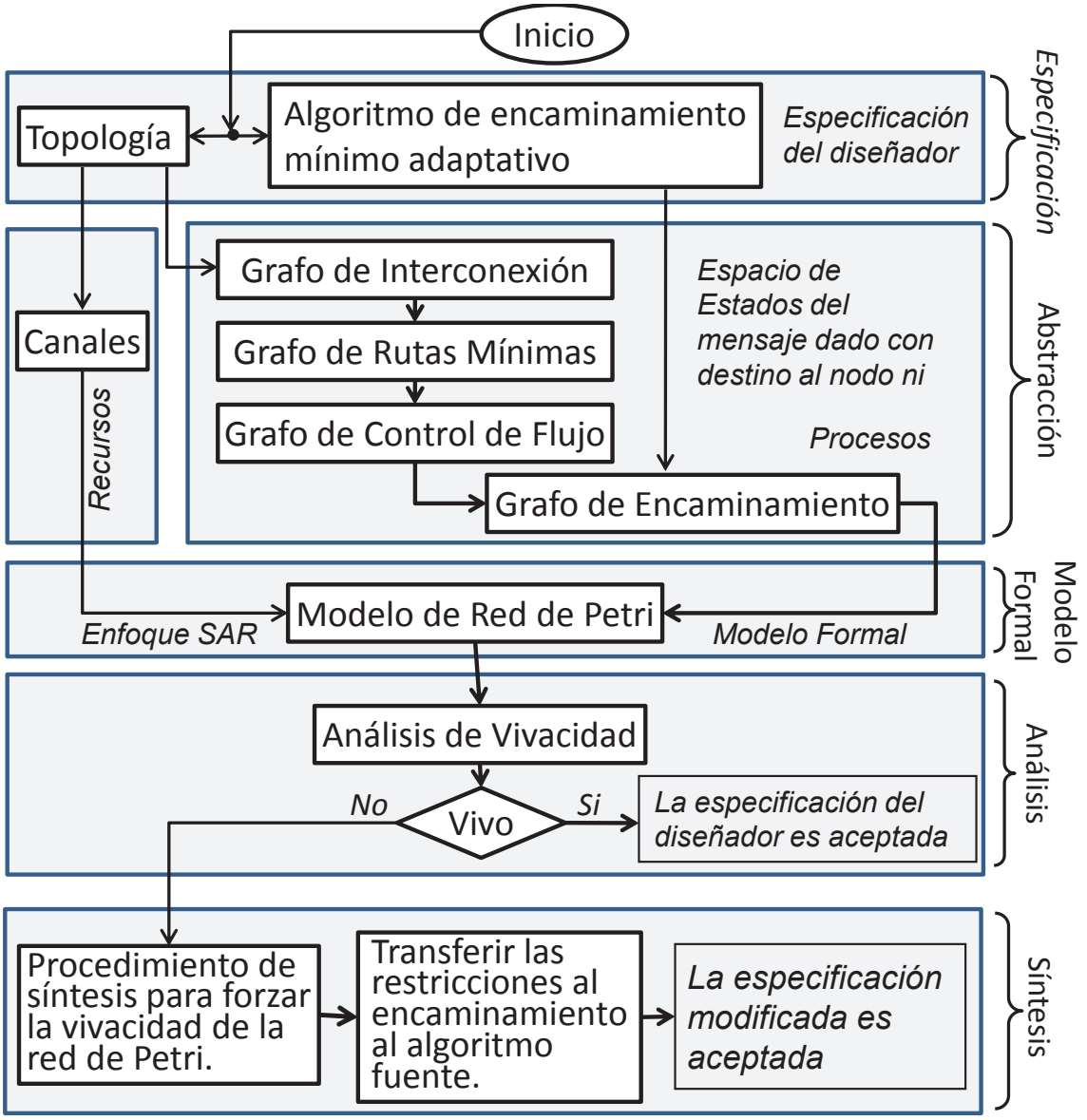

Figura 2.4: Metodología de diseño.

suministra como entrada una topología y un algoritmo de encaminamiento adaptativo mínimo. Con base en esta información se realiza una abstracción y posterior análisis del modelo obtenido, para aplicar un procedimiento de síntesis en caso de que así lo requiera el algoritmo de encaminamiento. Si en el análisis del modelo se detectan problemas de bloqueos, éste se corrige para que sea libre de bloqueos utilizando un método sistemático de síntesis. A fin de aplicar esta metodología se hará uso de las Redes de Petri como herramienta de modelado y análisis, conservando solo los aspectos relacionados con la aparición de los estados de bloqueos. En la etapa de análisis de nuestra metodología se determinará si existen problemas de bloqueos, basándonos en la caracterización estructural de la vivacidad del modelo de la Red de Petri. Si estos problemas existen es posible realizar los cambios necesarios en la etapa de síntesis de la metodología. Finalmente, los cambios aplicados al modelo se pueden trasladar al dominio de aplicación, para que sea aceptada por el diseñador del algoritmo de encaminamiento adaptativo mínimo. Esta metodología está asistida por la utilización de grafos que ayudarán al diseñador desde la especificación inicial hasta la obtención del modelo de Red de Petri. En el modelado formal que nos proporciona nuestra metodología, somos capaces de abstraer y representar el encaminamiento de un mensaje desde su nacimiento hasta su llegada a destino. Esta abstracción modela todos los estados alcanzados a través de la red de interconexión de multicomputadores desde una *perspectiva del mensaje* para posteriormente estudiar propiedades relacionadas a su estructura y comportamiento. La perspectiva del mensaje es coherente con nuestra visión *SAR* de la asignación de los canales en la red de interconexión por el algoritmo de encaminamiento. Hemos optado por utilizar redes de Petri [Murata 89], porque constituyen un paradigma muy potente para modelar, analizar y sintetizar sistemas concurrentes basados en eventos. Nos hemos apoyado en una sub–clase que tiene una caracterización estructural de la propiedad de vivacidad y deriva de la clase denominada S^4PR [Tricas 03]. Adicionalmente, en esta sub–clase podemos aplicar las políticas de control para corregir el modelo, y posteriormente el algoritmo de encaminamiento. Como entrada a la fase de diseño se deben suministrar los siguientes componentes:

1. Los patrones de la conexión entre los nodos que se llamará la *topología*.
2. El *algoritmo de encaminamiento*, que permite construir la ruta del mensaje.
3. El *control de flujo*, que se ocupa de la asignación de los recursos y las colisiones.

En nuestra metodología el diseñador es asistido por un conjunto de grafos que le permiten obtener el modelo de la Red de Petri. Posteriormente, se verifican propiedades cualitativas, como la existencia de bloqueos, hasta finalmente corregir el modelo. Esta metodología de diseño está dividida en niveles denominados *especificación, abstracción, modelo formal, análisis y síntesis*. La subsección 2.4.1 se dedica a los tres primeros niveles.

2.4.1. La especificación y la abstracción en la metodología

El primer nivel de la metodología es la **Especificación** que se realiza a partir de la topología y el algoritmo de encaminamiento propuesto por el diseñador, asumiendo siempre que trabajamos con el control de flujo tipo *wormhole*. En la especificación se introduce sin ambigüedades el algoritmo de encaminamiento y la topología de la red de interconexión. Estas informaciones se obtienen de manera precisa utilizando una notación tipo lenguaje de programación para establecer las caracteríticas del algoritmo de encaminamiento adaptativo mínimo y un grafo topológico que delimita la conectividad de los nodos de la red de interconexión, a la vez que enumera los canales entre ellos. Es preciso mencionar que podemos considerar topologías *regulares* e *irregulares* teniendo en consideración la minimalidad del algoritmo de encaminamiento que impide la aparición de ciclos en las rutas. Adicionalmente la topología debe ser conexa y el algoritmo de encaminamiento debe preservar esta característica topológica, para que sea considerado un algoritmo de encaminamiento válido en nuestra metodología. La información dada por la etapa de especificación es utilizada en el nivel sucesivo llamado **Abstracción** para obtener la enumeración de los canales de la topología; que en adelante serán considerados como los *recursos* del modelo. Los recursos son utilizados por los mensajes para moverse por la red de interconexión y hemos obviado la presencia de los buffers de los canales debido a que solo se consideran buffers de un flit de tamaño. Esta abstracción se resume en una visión que considera el sistema como:

1. Un conjunto de procesos secuenciales que tienen que seguir los mensajes.

2. Estos procesos secuenciales necesitan utilizar los recursos que son limitados y deben ser compartidos por otros procesos: los canales de interconexión.

A continuación procedemos a la definición y construcción de los *procesos* secuenciales que siguen los mensajes y que dependen del nodo de nacimiento, nodo destino, topología y el algoritmo de encaminamiento adaptativo mínimo. La topología de la interconexión de los nodos será usada junto con el algoritmo de encaminamiento para obtener lo sucesivos grafos. Cuanto más compleja es la topología tendremos subgrafos más complejos, como el caso de una topología en *malla, toro* o un *hipercubo* por citar algunos ejemplos de topologías en que existe más de una ruta para llegar a un nodo destino desde otro nodo.

Grafo de Interconexión (GI)

El primer grafo de la abstracción es el **Grafo de Interconexión** que formaliza la topología por medio de este grafo etiquetado $GI=(N,C)$, en donde N es el conjunto de nodos y C es el conjunto de arcos. El conjunto N es igual a ND y el conjunto

$C \subseteq N \times 2^{CH} \times N$, donde ND es el conjunto de los nodos de la red de interconexión (topología) y CH el conjunto de los canales. Cada arco en el GI etiquetado con $(n_1, s, n_2) \in C$ significa que existe un conjunto de canales $s \subseteq CH$ desde el nodo n_1 hasta el nodo n_2, tal que $n_1 \neq n_2$ y $s \neq \emptyset$ como se muestra en la figura 2.2.b. que representa la topología de la figura 2.2.a. En el GI somos capaces de representar canales en direcciones opuestas entre dos nodos, pero nunca canales que tengan un mismo nodo origen y destino. El GI es de gran utilidad para posteriormente determinar la minimalidad de una ruta debido a que simplifica la estructura redundante que determina la topología de la red al existir múltiples canales entre dos nodos. Observe que al tratar con algoritmos de encaminamiento adaptativos mínimos no existirán ciclos porque cada paso que da el mensaje es productivo debido a que se acerca al nodo destino.

Grafo de Rutas Mínimas (*GRM*)

El segundo grafo es el **Grafo de Rutas Mínimas** para el nodo destino n_i tal que $i \in \mathbb{N}$ y $i \leq \mid N \mid$. Este grafo captura todas las rutas de longitud mínima desde el nodo n_j hasta el nodo destino n_i tal que $n_i \neq n_j$. Esto significa que existirá un grafo por cada nodo destino n_i y que será un subgrafo del GI siendo un digrafo acíclico etiquetado $GRM_i = (V, E)$, donde $V = N$, y $E \subseteq C$, construidas a partir de GI siguiendo los siguientes pasos:

1. Todos los arcos de salida del nodo destino n_i en GI no pertenecen a E. Esta condición es coherente con las condiciones dadas en [Dally 87] donde dice que una vez que un mensaje llega a su destino es consumido en ese nodo.

2. La función $L_i : V \rightarrow \mathbb{N}$ se define como $L_i(n_i) = 0$ y $\forall n_j \neq n_i$, $L_i(n_j) = k$, donde k es la longitud de la ruta mínima desde n_j a n_i en el GI. Esto implica que $k < \mid V \mid$ porque de lo contrario estaríamos ante rutas no mínimas.

3. Todos los arcos $(n_1, s, n_2) \in C$ en GI, tal que $L_i(n_1) + 1 \neq L_i(n_2)$, no pertenecen a E. Es preciso recordar que $L_i(n_i) \geq 0$, por tal motivo solo se permiten relaciones entre vértices adyacentes.

Para obtener el GRM_i utilizamos la conectividad dada por el GI, partiendo desde el nivel cero que es el nodo destino n_i, hasta llegar a todos los otros nodos fuente $n_j \neq n_i$. Dada la condición de que los arcos de salida de n_i no son considerados, entonces nunca tendremos ciclos en el GRM_i. Tenga en cuenta que pueden existir múltiples arcos entre dos vértices, pero esto es almacenado en el conjunto $s \subseteq CH$. El GRM_i del ejemplo de la figura 2.2 está representado en la figura 2.5.

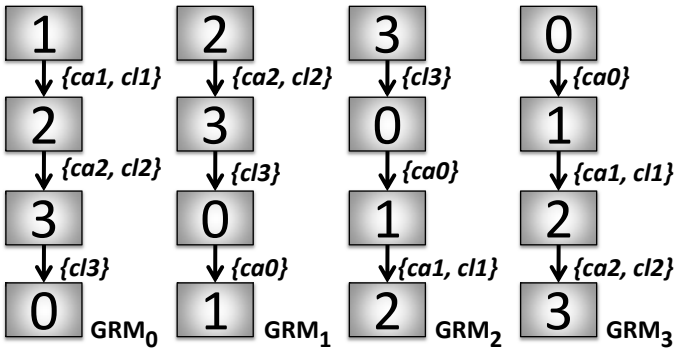

Figura 2.5: Grafo de Rutas Mínimas del GI de la figura 2.2.b

Grafo de Control de Flujo (GCF)

El tercer grafo es llamado el **Grafo de Control de Flujo** que detalla la información dada por cada *GRM* pero considerando el control de flujo. En el control de flujo del tipo **wormhole**, cada mensaje puede estar compuesto por uno o más flits. Como se considera tradicionalmente, nosotros solo evaluaremos mensajes de más de un flit debido a que en el caso de mensajes compuestos por un solo flit no pueden participar en un bloqueo. Esto obedece a que no se satisface la condición de *retención y espera* dada por [Coffman 71]. De esta forma en nuestro modelo se distinguen los estados de los mensajes según los nodos en donde se encuentra la cabeza y la cola representados por una **h** o una **t** respectivamente. El concepto de estado está relacionado con la cantidad de canales que se tienen reservados en cada instante, por lo tanto el avance del flit cabeza está condicionado a la existencia de al menos un canal físico que pueda ser asignado para ese movimiento. Por tanto el GCF_i=(Q,F) utiliza la información dada por el GRM_i=(V,E) y satisface las siguientes condiciones:

1. $Q \subseteq V \times V$, donde $\forall (n_h,n_t) \in Q$, n_h=n_t o $L(n_h) < L(n_t)$. Esto es, el primer componente de los estados definidos corresponde al nodo donde está el flit cabeza, y el segundo al nodo donde está el flit cola.

2. $F \subseteq Q \times \{A,R\} \times 2^{CH} \times Q$, donde F contendrá las siguientes aristas:

   a) Arcos de asignación $((n_{h1},n_t),A,s,(n_{h2},n_t))$, $\forall\, n_t \in V, \forall (n_{h1},s,n_{h2}) \in E$.

   b) Arcos de liberación $((n_h,n_{t1}),R,s,(n_h,n_{t2}))$, $\forall\, n_h \in V, \forall (n_{t1},s,n_{t2}) \in E$.

El GCF_i es un digrafo acíclico porque el GRM_i es también un digrafo acíclico. La figura 2.6 muestra el Grafo de Control de Flujo para el nodo destino 0 como GCF_0, que corresponde

al GRM_0 de la figura 2.5. El *GCF* se obtiene a partir de la información dada por el *GRM* y los vértices *Q* del *GCF* son un producto cartesiano de los vértices del *GRM*, pero con dos condiciones. La primera condición permite que la primera y segunda componente sean iguales, es decir que puedan estar en el mismo nodo la cabeza y la cola del mismo mensaje. Esto representa el nacimiento y la llegada de un mensaje a su destino. La segunda condición impide que la posición de la segunda componente, es decir la cola esté en un nivel inferior a la cabeza, debido a que esto es coherente con el comportamiento del control de flujo tipo *wormhole*, donde la cola nunca rebasará a la cabeza del mensaje.

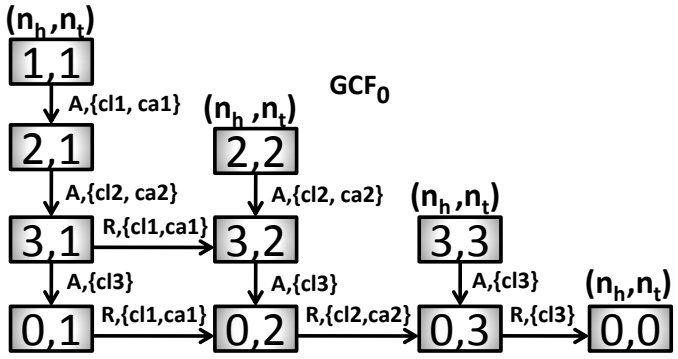

Figura 2.6: Grafo de Control de Flujo para el nodo destino 0.

Los arcos del *GCF* son un producto cartesiano de los vértices *Q*, pero solo se consideran vértices adyacentes que comparten canales. Los arcos representan dos tipos de eventos; los que *toman* los canales y aquellos que los *liberan*. Para simplificar la representación de las etiquetas de los eventos en los arcos de los grafos se utilizan las letras mayúsculas **A** y **R** que significan la reserva y la liberación respectivamente. Para saber si se trata de una reserva o una liberación se recurrirá a evaluar la primera o segunda componente de cada vértice adyacente. Si dos vértices adyacentes tienen la misma segunda componente, esto significa que entre ellos cambió solo la primera por tanto se trata de una reserva de un canal. Por otra parte si dos vértices adyacentes tienen la misma primera componente, esto significa que cambió la segunda componente por lo tanto se trata de una liberación de un canal.

Grafo de Encaminamiento

La abstracción finaliza con el grafo **Grafo de Encaminamiento (GE)** que se obtiene utilizando la información proporcionada por la topología de la red de interconexión y el algoritmo de encaminamiento adaptativo mínimo. El algoritmo de encaminamiento es una

función R: $N \times N \rightarrow 2^{CH}$, tal que si n_c *(current node en inglés)* es el nodo actual del flit cabeza y n_d es el nodo destino del mensaje, la función $R(n_c, n_d)$ determina el conjunto de canales de salida del nodo actual para seleccionar uno que sea asignado de forma tal que se aproxime o alcance el nodo destino. El modelo que estamos considerando es posibilista por lo tanto considera todas las opciones permitidas por el algoritmo de encaminamiento aunque muchas de ellas sean mutuamente excluyentes. Cada posibilidad corresponde a diferentes asignaciones de recursos para alcanzar el nodo destino. Por lo tanto para representar este algoritmo para cada GCF_i, nosotros construiremos el grafo llamado *Grafo de Encaminamiento* para cada nodo destino n_i. Como lo muestra la figura 2.7 para el destino cero partiendo de la información del GCF_0 de la figura 2.6.

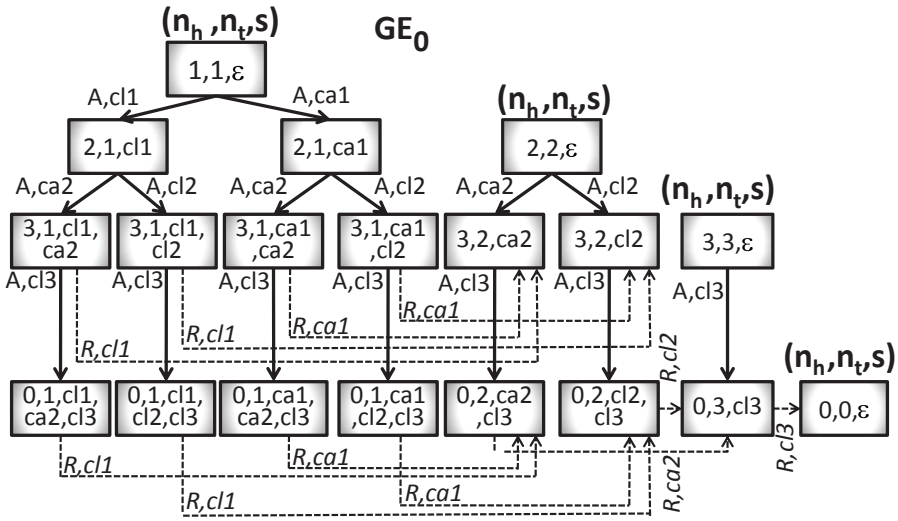

Figura 2.7: Grafo de Encaminamiento para el nodo destino 0.

El GE_i=(Q', F') donde $Q' \subseteq V \times V \times CH^*$ representa el conjunto de estados en que puede estar cada mensaje destinado al nodo n_i desde cada uno de los otros nodos de la red. Cada estado es caracterizado por la posición de la cabeza en los nodos, la posición de cola en los nodos así como la secuencia de canales que mantiene reservado ese estado del mensaje; $F' \subseteq Q' \times \{A, R\} \times CH \times Q'$ es el conjunto de arcos que representa las transiciones desde un estado a otro, dado por el avance de la cabeza o de la cola a su nodo adyacente. Siguiendo la estrategia del control de flujo *wormhole*, el movimiento de la cabeza reserva (A)*(allocates)* el canal especificado por el arco, que será liberado (R) *(release)* por el flit cola. El proceso de adquisición y liberación se realiza de forma unitaria y ordenadamente por el flit cabeza y por el flit cola de cada mensaje. Observe en el GE_i el siguiente camino que va desde un estado (n, n, ε), $n \neq n_i$, el cual corresponde al

Algoritmo 2 Construcción del $GE_i = (Q', F')$.

Entrada: $GCF_i = (Q, F)$
Salida: $GE_i = (Q', F')$

1. proximo-nivel := $\{(n,n,\varepsilon) | (n,n) \in Q\}$
2. Q' := proximo-nivel;
3. $F' := \emptyset$;
4. **Mientras que cada** proximo-nivel $\neq \emptyset$ **Hacer**
5. nivel-actual := proximo-nivel;
6. proximo-nivel := $\emptyset$;
7. **Para cada** $(n_1, n_2, r) \in$ nivel-actual **Hacer**
8. **Para cada** $((n_1,n_2), X, S, (n_3,n_4)) \in F$ **Hacer**
9. **Para cada** $c \in S$ **Hacer**
10. **Si** $(c \in R(n_1,n_i))$ **y** $(X = A)$ **Entonces**
11. proximo-nivel := proximo-nivel $\cup \{(n_3,n_4,r\&c)\}$;
12. $Q' := Q' \cup \{(n_3,n_4,r\&c)\}$;
13. $F' := F' \cup \{((n_1,n_2,r), X, c, (n_3,n_4,r\&c))\}$;
14. **Fin Si**
15. **Si** $(r = c\&t)$ **y** $(X = R)$ **Entonces**
16. proximo-nivel := proximo-nivel $\cup \{(n_3,n_4,t)\}$;
17. $Q' := Q' \cup \{(n_3,n_4,t)\}$;
18. $F' := F' \cup \{((n_1,n_2,c\&t), X, c, (n_3,n_4,t))\}$;
19. **Fin Si**
20. **Fin Para**
21. **Fin Para**
22. **Fin Para**
23. **Fin Mientras que que**

nacimiento de un mensaje en el nodo n, hasta el estado (n_i, n_i, ε). Este camino representa el encaminamiento de un mensaje en la red de interconexión desde el nodo fuente n al nodo destino n_i. Existen estados que no consumen recursos tales como el nacimiento y la llegada a destino del mensaje y por lo tanto se representan con la cadena ε. Obsérvese que para obtener este $GE_i = (Q', F')$ desde el correspondiente $GCF_i(Q, F)$, se aplicará el algoritmo 2 para obtener los arcos sólidos que representan la asignación del canal y los arcos punteados que son usados para representar la liberación del canal. Al ser un algoritmo posibilista debe explorar todas las opciones, procediendo nivel por nivel mientras etiqueta los arcos de asignación y liberación de los recursos. El algoritmo 2 construye el *GE* para cada nodo destino, utilizando la información dada por los vértices y los arcos etiquetados del *GCF*. Se procede entonces desde los niveles más altos hasta descender al nivel $\emptyset$, en donde se representa el nodo destino del grafo del *GCF*. El algoritmo 2 se utiliza para construir el *GE* con destino al nodo cero, que se muestra en la figura 2.7.

2.4.2. Obtención del modelo de red de Petri

En esta subsección construiremos el modelo de la Red de Petri, partiendo de la información que nos proporcionan los diversos *GE* obtenidos. El procedimiento de construcción de la Red de Petri será por composición de todos los *GE*, agregando las transiciones que *toman* o *liberan* los recursos así como los *lugares proceso, lugares resposo* y *lugares recurso*. Este procedimiento es coherente con nuestra visión *SAR* del problema, por lo tanto haremos una diferenciación entre los *procesos* y los *recursos*.

1. *Recursos*: El conjunto *CH* de los canales físicos que forman la topología. Los canales dados en la *especificación* son todos físicos.

2. *Procesos*: El movimiento de un mensaje desde su nacimiento hasta la llegada a su destino es considerado un proceso de encaminamiento, por tanto un proceso está formado por todos los encaminamiento posibles en el sistema hasta un nodo destino y es representado por cada *GE*.

Iniciamos la construcción del modelo de Red de Petri desde cada $GE_i(Q', F')$ con el que construimos la red $\mathcal{N}_i = \langle P_{0_i} \cup Ps_i, T_i, F_i \rangle$ la cual representa el espacio de estados alcanzado por cada mensaje con destino al nodo n_i, de la red de interconexión. Esta construcción procede según las siguientes reglas .

1. Agregar un lugar al conjunto Ps_i por cada vértice del GE_i, $(n_1, n_2, s) \in Q'$ tal que $n_1 \neq n_2$. El nombre del lugar será formado por la concatenación de los identificadores de la posición del flit cabeza y del flit cola, llamados n_1 y n_2 respectivamente, más la secuencia de canales que se mantienen reservados para este

estado y representados por $s \subseteq CH$. Estos lugares son llamados los *lugares proceso* y están desmarcados en el estado inicial M_0, porque en este estado no hay mensajes en tránsito.

2. Agregar un único lugar p_{0_i}, correspondiente a la fusión de los estados de la forma $(n,n,\varepsilon) \in Q'$, que pertenecerá el conjunto $P_{0_i} = \{p_{0_i}\}$ y será llamado *lugar reposo*. La marca inicial de este lugar será igual al número máximo de mensajes que puedan ser simultáneamente enviados al nodo de destino n_i. Si este número es ilimitado o es desconocido, entonces no se necesita agregar un lugar p_{0_i}, i.e. el número de mensajes con destino al nodo de n_i, en esta red, está únicamente limitado por los canales.

3. Agregar una transición al conjunto T_i por cada arco del grafo GE_i. Para cada arco $((n_1,n_2,s),X,c,(n_3,n_4,r)) \in F'$, el nombre de la transición será $n_1\&n_2\&s\&n_3\&n_4\&r$.

4. Para cada arco $((n_1,n_2,s),X,c,(n_3,n_4,r)) \in F'$, $n_1 \neq n_2$, agregar un arco desde el lugar $n_1\&n_2\&s$ hasta la transición $n_1\&n_2\&s\&n_3\&n_4\&r$, y un arco desde esta transición al lugar $n_3\&n_4\&r$.

5. Para cada arco $((n,n,s),X,c,(n_3,n_4,r)) \in F'$ agregar un arco desde el lugar $idle$, p_{0_i} (si existe este lugar) hasta la transición $n\&n\&s\&n_3\&n_4\&r$, y agregar un arco desde esta transición al lugar $n_3\&n_4\&r$.

Tenga en cuenta que la red $\mathscr{N}_i$, se obtiene después de aplicar las reglas de los párrafos precedentes, y da como resultado una máquina de estado fuertemente conexa. En efecto, por construcción, cada transición tiene únicamente un lugar de entrada y un lugar de salida porque se ha añadido una transición por cada arco en el grafo GE_i, y los lugares corresponden a ambos extremos del arco. Por otra parte, es una máquina de estado fuertemente conexa porque cada arco en GE_i, (n_1,n_2,s), es alcanzable por una ruta desde el vértice fuente (n,n,ε), puesto que la construcción del GE_i necesita que construyamos la secuencia de los canales asignados s desde el nodo fuente; y desde el vértice (n_1,n_2,s), siempre existirá un camino hasta el vértice destino (n,n,ε).

Tomando esto en consideración el lugar p_{0_i}, representa la fusión de todos los vértices (n,n,ε) del grafo GE_i, podemos concluir que la red $\mathscr{N}_i$ es fuertemente conexa. Adicionalmente, podemos ver que todos los circuitos de $\mathscr{N}_i$ contienen el lugar p_{0_i}; porque el grafo original GE_i es acíclico. Después de todas estas transformaciones se obtiene un conjunto de máquinas de estado fuertemente conexas $\mathscr{N}_i$, una por cada nodo destino. El último paso para obtener la perspectiva *SAR* de la red de interconexión es la adición de los recursos que en este caso son los canales físicos que conectan los nodos y su integración

con las máquinas de estado. Esto se puede hacer máquina a máquina y construir el modelo completo por la fusión de los lugares recurso que tengan el mismo nombre, siguiendo los pasos que mencionamos a continuación.

1. Agregar un lugar p_c al conjunto P_R por cada canal físico $c \in CH$ que se encuentra en la red de interconexión. El marcado inicial de este lugar será igual al máximo número de mensajes que pueden estar en tránsito simultáneamente en el canal $(bandwidth\ of\ the\ physical\ channel)$. Como se consideró en [Dally 87] un canal solo soporta un mensaje a la vez, por lo tanto su marcado es uno.

2. Para cada arco del grafo GE_i de la forma $((n_1,n_2,s),A,c,(n_3,n_4,r)) \in F'$, agregue un arco desde el lugar p_c, (el lugar que representa el canal físico c) hasta la transición $n_1\&n_2\&s\&n_3\&n_4\&r$. Este arco en la red de Petri representará la asignación del canal c. Para cada arco del grafo GE_i de la forma $((n_1,n_2,s),R,c,(n_3,n_4,r)) \in F'$ agregue un arco desde la transición $n_1\&n_2\&s\&n_3\&n_4\&r$ al lugar p_c. Este arco en la red de Petri representa la liberación del canal c.

Esta red se denominará $\mathcal{N}_i^R$, que representa el encaminamiento de un mensaje con destino al nodo n_i, y nacido en cualquiera de los otros nodos n_j distintos al nodo n_i. El lugar reposo, $IDLE$, tiene un marcado inicial que representa, el máximo número de mensajes que simultáneamente podemos enviar a este nodo destino n_i. Cada mensaje que tiene como destino el nodo n_i se representa en esta máquina de estados SM como una marca en alguno de los lugares proceso que competirá por los mismos canales de comunicación para poder alcanzar el nodo destino. Desde nuestro enfoque SAR que sigue la perspectiva del mensaje, consideramos estos mismos canales como recursos del sistema. El modelo completo es obtenido desde diferentes $\mathcal{N}_i^R$ por la fusión de los lugares recurso que tienen el mismo nombre que aparecen en diferentes $\mathcal{N}_i^R$.

La figura 2.8 representa de forma esquemática el modelo completo que corresponde al ejemplo de la sección 2.2. En esta figura los nombres de lugares *(procesos y recursos)* y de transiciones se han simplificado para facilitar la compresión del modelo. Cada lugar proceso se identifica con una etiqueta que representa el estado del mensaje asociado al mismo utilizando una notación simplificada $h_i{+}t_j$. Esta notación significa que la cabeza del mensaje está en el nodo i y la cola está en el nodo j. En las transiciones se indica el recurso/canal que toman o liberan, pero como se ha establecido cada transición realiza una u otra acción pero no ambas a la vez y para una mayor legibilidad se utilizará el nombre de los recursos/canales en letra mayúscula. La siguiente sección 2.5 se dedica a probar que las Redes de Petri obtenidas utilizando nuestra metodología de modelado de las redes de interconexión para multicomputadores, en las que se utilizan algoritmos encaminamiento adaptativos mínimos, pertenecen a la subclase de las redes de Petri denominadas S^4PR [Tricas 03][Tricas 99].

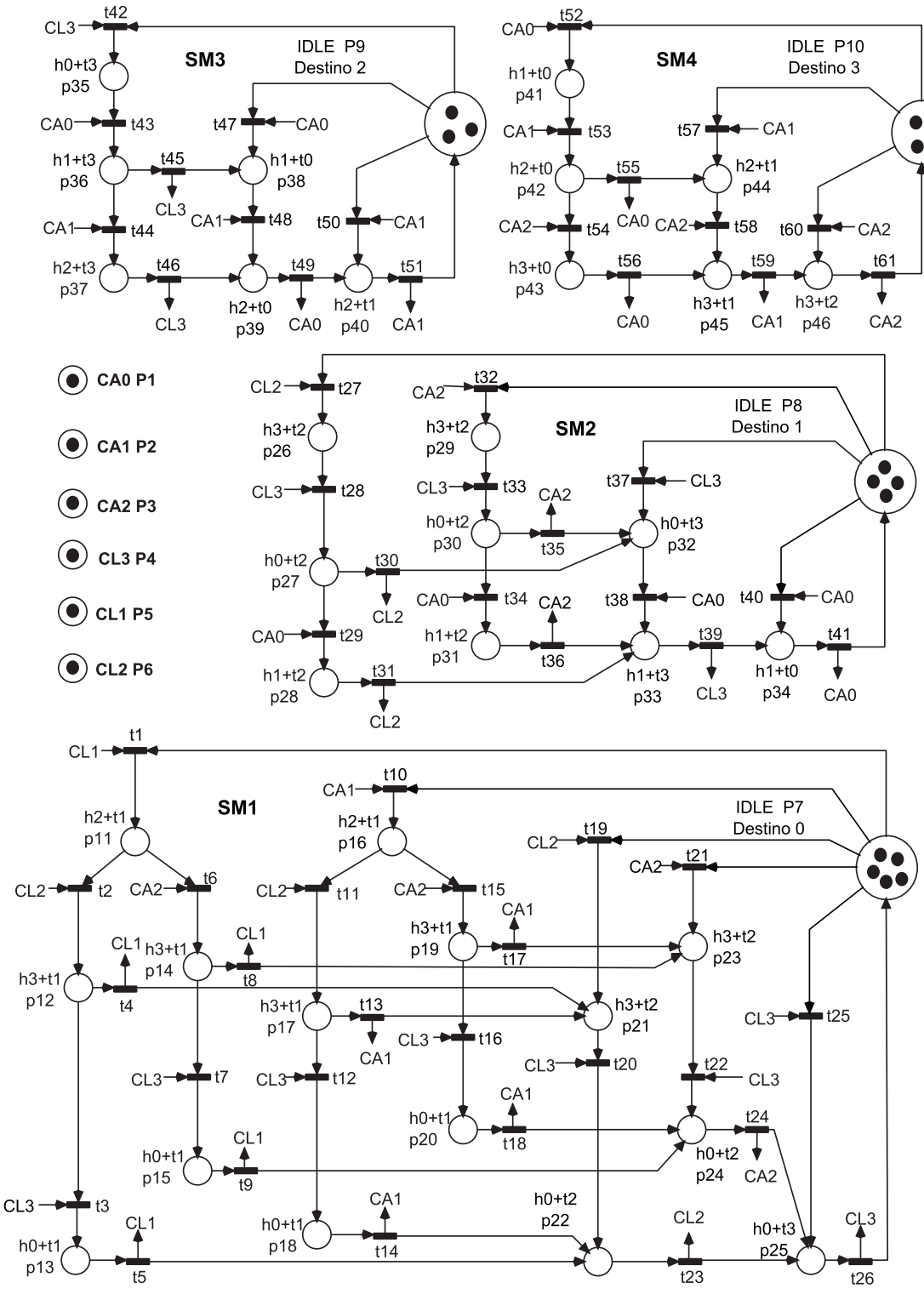

Figura 2.8: Red de Petri de la red de interconexión de la figura 2.2.

2.5. Interpretación de los modelos

En la sección 2.4 hemos explicado la construcción del modelo de la Red de Petri, siguiendo nuestra metodología. En esta sección formalizaremos la clase de modelos de Red de Petri obtenidos a través de la metodología así como sus propiedades estructurales más relevantes, como la caracterización de la vivacidad. Adicionalmente se presentará la interpretación semántica de los elementos y propiedades dentro del modelo formal que corresponden al algoritmo de encaminamiento de la red de interconexión y que se utilizará a lo largo de este trabajo. Estos mismos conceptos semánticos serán caracterizados estructuralmente y comportamentalmente en el modelo de la Red de Petri del tipo S^4PR.

2.5.1. Caracterización de la clase de redes de Petri obtenida

Los modelos de Redes de Petri obtenidos por nuestra metodología tienen características estructurales particulares que encajan en la clase de Redes de Petri conocida como las S^4PR. Las bien estudiadas características estructurales y comportamentales de las S^4PR nos siven de apoyo para realizar el estudio del sistema así como aplicar correcciones al mismo. Seguidamente demostraremos las características que identifican la clase de modelos obtenidos a través de nuestra metodología.

Lema 1. *Dado una red de interconexión especificada por medio de una topología y de un algoritmo de encaminamiento adaptativo mínimo, el modelo de la red de Petri obtenido con el método descrito en la sección 2.4, pertenece a la clase S^4PR.*

Esquema de la prueba: Después de las reglas para obtener las redes de Petri $\mathcal{N}_i$ desde el correspondiente grafo de encaminamiento GE_i, nosotros hemos probado que cada $\mathcal{N}_i$ es una máquina fuertemente conexa. Adicionalmente hemos probado que $\mathcal{N}_i$,$\mathcal{N}_j$, $i \neq j$, son conjuntos disjuntos de redes. También hemos probado que cada ciclo de cada máquina de estado fuertemente conexa $\mathcal{N}_i$ contiene un P_{0_i}. Por lo tanto, para terminar la prueba solo necesitamos probar la existencia de un único *p-semiflujo* $\mathbf{y}_r \in \mathbb{N}^{|P|}$ para cada recurso r. Pero esto es muy fácil de probar, porque para cada transición donde entra el recurso r *(se asigna el recurso)*, existe una única trayectoria, en la máquina de estado fuertemente conexa, hasta alcanzar a cada transición donde sale el recurso r *(se libera el recurso)*. Por otra parte, todas las transiciones donde r es un lugar de la salida de la máquina de estado $\mathcal{N}_i$ están conectadas por medio de una ruta mínima desde alguna transición donde r es un lugar de entrada. Por lo tanto, el recurso r más todos los lugares proceso definen las trayectorias mínimas que conectan las transiciones de la salida de r y las transiciones de la entrada de r, forman el *p-semiflow* que son únicas porque estamos tratando con las redes $\mathcal{N}_i$ que son máquinas de estado. □

Observe que el resultado anterior, también es verdadero para topologías no regulares, porque nosotros estamos considerando de una manera explícita los caminos hacia un nodo de destino. Por lo tanto, la no regularidad de la topología no afecta la Red Petri que se obtendrá. Sin embargo, la no minimalidad de los algoritmos de encaminamiento puede conducir a clases de redes más generales que la clase S^4PR en el caso de existencia de ciclos en la ruta seguida por algún mensaje. Una vez que hemos caracterizado el tipo de redes, podemos usar la teoría desarrollada para las Redes de Petri de la clase S^4PR, tratando de interpretar estos resultados desde el punto de vista de las redes de interconexión. En las siguientes subsecciones realizaremos la interpretación de los resultados obtenidos para el dominio de aplicación de las redes de interconexión.

2.5.2. Interpretación semántica de la clase para el modelo SAR

En esta subsección se presentará la interpretación semántica que se utilizará a lo largo de este trabajo, para cada uno de los elementos y propiedades de una Red de Petri que pertenece a la clase S^4PR. Esta Red de Petri será obtenida a partir del método presentado en la sección 2.4.

1. *Caracterización del conjunto de rutas mínimas hacia un destino determinado*: Es el conjunto de rutas mínimas que enlazan un nodo cualquiera de la red con un nodo destino concreto. Cada conjunto de rutas mínimas con destino modela las diferentes actividades llevadas a cabo en el procesamiento de mensajes hacia un destino. Por lo tanto un mensaje no perderá su identidad e integridad a lo largo del proceso. También, establece la forma en la cual los recursos son utilizados en dicho procesamiento. En nuestro caso, los mensajes son encaminados por el algoritmo de encaminamiento que le dirige para tomar los recursos y liberarlos de forma ordenada.

   El conjunto de rutas mínimas con destino puede ser modelado en Red de Petri a través de una máquina de estados fuertemente conexa, $\mathscr{N}$ que está cubierta por los t–semiflujos correspondientes al conjunto de rutas de encaminamiento que contiene. Desde el punto de vista de la aplicación, esto equivale al movimiento de un mensaje a partir del lugar del reposo P_0 siguiendo una ruta de encaminamiento hasta llegar nuevamente a P_0 en donde se completa el procesamiento de dicho mensaje. En la clase S^4PR, los lugares de $\mathscr{N}$, representados por un círculo, están relacionados con las diferentes operaciones de transporte llevadas a cabo sobre los mensajes. Estas operaciones son representadas por lugares que se les denomina lugares proceso.

   Cada mensaje que entra al sistema es un proceso. Su avance se modela como marcas que se mueven a través de $\mathscr{N}$. En un conjunto de rutas mínimas con

destino se pueden distinguir los puntos de nacimiento de los mensajes y los puntos de llegada de los mensajes. Un mensaje que entra en un plan de encaminamiento avanza a hacia su destino a través de las transiciones contenidas en $\mathscr{N}$. En donde, dichas transiciones (representadas por una barra) modelan los cambios de estados correspondientes a los tres tipos diferentes de eventos:

a) *Inicio de un proceso de encaminamiento*: Solicitud de los recursos para encaminar un nuevo mensaje. Esto lo conocemos como el nacimiento del mensaje en el sistema.

b) *Progreso de un proceso de encaminamiento*: El movimiento de un mensaje dentro de dos diferentes nodos intermedios en el sistema que representa el cambio del estado de los recursos debido a la toma o liberación de recursos.

c) *Finalización de un proceso de encaminamiento*: Cuando el mensaje alcanza su nodo destino es consumido por su receptor.

La ejecución de un conjunto de rutas mínimas con destino es realizado por la ejecución de una ruta de encaminamiento. En un mismo conjunto de rutas mínimas con destino pueden existir varias rutas de encaminamiento debido a la adaptabilidad de los algoritmos, muchas de ellas mutuamente excluyentes. Una ruta de encaminamiento es una secuencia de transiciones disparables en $\mathscr{N}$ cuya ocurrencia representa la encaminamiento de un objeto.

2. *Todas las rutas de encaminamiento son reproducibles*. Cada ciclo en una máquina de estados representa la secuencia de disparos con la cual se completa el procesamiento de un objeto a partir del lugar de reposo hasta el final. La naturaleza reproducible de esta secuencia es capturada en una Red de Petri por el concepto de t–semiflujo.

3. *La incorporación de los Recursos a cada conjunto de rutas mínimas con destino*: Los recursos son los elementos físicos que componen el sistema abordado *(canales de comunicación o segmentos)*.

   Cada tipo recurso es representado mediante un lugar, que posee una capacidad finita y es modelada por medio de un marcado inicial. Los arcos que unen a los lugares recurso con las transiciones modelan el cambio de estado de los recursos a medida que el mensaje es transportado en el sistema. Los arcos relacionados a los lugares recurso del sistema modelan la forma en la cual los recursos son utilizados durante el procesamiento de los mensajes. Por ejemplo, los arcos de salida desde un lugar recurso hacia una transición modelan la adquisición de los recursos realizado por el flit cabeza del mensaje, mientras que los arcos que entran de las transiciones a los lugares recurso modelan su liberación que es realizado por el flit cola del mensaje.

Cada estado en un conjunto de rutas mínimas con destino tiene asociado el conjunto de recursos requeridos en cada paso del procesamiento. Esto impone que cada paso del procesamiento use al menos un recurso. De igual forma, se establece que cada recurso debe ser reutilizado en serie, esto es, los recursos no pueden ser creados ni destruidos.

4. *Cada ruta de encaminamiento debe poder ser ejecutada aisladamente*. Esta propiedad se mantiene en la clase S^4PR debido a que el conjunto de rutas mínimas con destino tiene una estructura de máquina de estados fuertemente conexa (definición 1.3) y el marcado inicial del estado de reposo es no vacío (definición 2.1).

5. *Un mensaje nacido con destino a un nodo n_i alcanzará el destino, sea cual sea el nodo en donde nace.* Existe al menos una ruta desde origen a destino. Esta propiedad se mantiene en la clase S^4PR porque una máquina de estados fuertemente conexa puede mover todos los tokens a cualquier lugar de la red.

6. *El conjunto de rutas mínimas con destino usa los recursos en forma conservativa.* Esta propiedad está caracterizada en la clase S^4PR por la existencia de un p–Semiflujo mínimo para cada recurso, $\mathbf{y}_r$ tal que $\mathbf{y}_r[r] = 1$, impuesto por la definición de la clase (definición 1.4). Además, estos p–semiflujos mínimos $\mathbf{y}_r$ inducen una relación invariante de marcado, $(\forall \mathbf{m} \in \mathrm{RS}(\mathcal{N}, \mathbf{m_0}), \mathbf{y}_r \cdot \mathbf{m} = \mathbf{y}_r \cdot \mathbf{m_0} = \mathbf{m_0}[r])$, que implica la reusabilidad de los recursos. Esto es, los procesos no pueden cambiar los recursos que utilizan. Por lo tanto, los recursos no pueden ser creados ni destruidos.

7. *Todos los estados de un conjunto de rutas mínimas con destino usan algún recurso para llevar a cabo sus operaciones*. Esta propiedad es la última de la definición 1. Por lo tanto, los modelos de redes de esta clase SARs son conservativos.

8. *Un conjunto de rutas mínimas con destino y sus recursos está bien definido*. Esto significa que cuando no hay actividad en el sistema, cada posible ruta de encaminamiento tiene los recursos suficientes para ejecutarse aisladamente (definición 2) [Tricas 99].

La clase de *SARs* que pueden ser modeladas con las redes S^4PR permiten modelar decisiones en línea o rutas adaptativas y el uso de múltiples recursos en cada paso del procesamiento.

2.6. Aplicación a vehículos guiados automáticamente

Existen otros dominios de aplicación que tienen paralelismos obvios con las redes de interconexión para multicomputadores, lo cual nos permite aprovechar nuestra metodología de modelado y corrección. Así vemos que en los *vehículos guiados automáticamente (AGVs)* [1][Müller 83] se utilizan algoritmos de encaminamiento para encaminar objetos y llevarlos a sus respectivos destinos. Los *AGVs* son *robots* que se utilizan para mover objetos de forma automática en diferentes tipos de industrias o negocios como *fábricas, centros de distribución, hospitales, etcétera*. Los *AGVs* nacieron de la necesidad de apoyar tareas repetitivas como la alimentación de materias primas entre máquinas de una célula de fabricación robotizada, acarreo de materiales o componentes entre dos puntos de una fábrica, siendo la industria del automóvil un buen ejemplo de estos casos. Funcionan con baterías recargables de larga duración que les ofrecen gran autonomía y contribuyen a que sean muy eficientes. Hoy en día se ha recurrido a la utilización de algoritmos de encaminamiento para seleccionar la mejor ruta que permita al *AGV* alcanzar a su destino por la flexibilidad que proporciona este método [Reveliotis 00]. A continuación presentaremos una aproximación para tratar el problema del bloqueo en el encaminamiento de los *AGVs*, a través de nuestra metodología, presentada en la sección 2.4 y su visión *SAR* del sistema.

2.6.1. Analogías y diferencias

Las analogías entre las redes de interconexión para multicomputadores y los *AGVs* nos permiten aprovechar la teoría de las Redes de Petri para abordar los problemas de bloqueos en este dominio de aplicación. De igual forma existen diferencias que no pueden ser ignoradas, como la *interpretación del modelo* y las *técnicas de corrección*.

A continuación incluiremos las analogías entre ambos dominios de aplicación y que se utilizará para el diseño de algoritmos de encaminamiento adaptativos mínimos libres de bloqueos en los *AGVs*. Estas analogías se detallan en el cuadro 2.2.

El hecho de que existen diferencias entre ambos dominios de aplicación obliga a establecer condiciones al funcionamiento de los *AGVs*. Por lo tanto para el modelado de los *AGVs* hemos utilizado las siguientes condiciones de funcionamiento, las cuales aseguran la compatibilidad de la estrategia de modelado entre ambos dominios de aplicación. Estas condiciones no restringen el buen funcionamiento del *AGV* pero si situaciones particulares que atentan contra un uso óptimo de los recursos existentes.

[1] Siglas por terminología inglesa

Analogías entre los sistemas	
Redes de Interconexión	**Sistemas AGVs**
Red	Plataforma
Canales de Comunicación	Segmentos
Mensaje	Convoy
Nodo	Estación
Flit	Vagón
Wormhole	Control del Convoy

Cuadro 2.2: Analogías entre dos dominios de aplicación.

1. Un $AGV's$ llegado a su estación destino es recibido en su totalidad.
2. Una estación de trabajo puede enviar objetos a todas las demás estaciones excepto a ella misma.
3. Un convoy válido debe tener al menos un vagón de carga.
4. El largo del convoy debe ser menor que el largo de la ruta seleccionada en la plataforma.
5. La topología de la plataforma de trabajo no tendrá cruces de vías sin una estación.
6. Todas las rutas en la plataforma de trabajo serán mínimas.
7. Cada estación de trabajo ejecutará un algoritmo de encaminamiento.
8. Solo la estación de destino será la única condición para seleccionar la ruta del AGV.
9. En cada estación de trabajo atenderá únicamente un AGV a la vez.

Las analogías nos permiten aprovechar la teoría de las Redes de Petri a través de la metodología y las diferencias nos permite hacer una nueva interpretación de los modelos durante la etapa de síntesis del modelo. Una diferencia significativa que existe entre ambos dominios de aplicación está dada por el tipo de corrección que se realiza en cada sistema. En las redes de interconexión existe un grado de flexibilidad adicional dado por la existencia de recursos virtuales $(canales)$, que no existen en el sistema de los AGV. Esta diferencia cambia la método de corrección durante la etapa de síntesis del modelo, cuando es aplicada a los $AGVs$, el cual se abordará en detalle en el capítulo 4. A continuación presentaremos los trabajos que abordan el problema del bloqueo en los $AGVs$, dando énfasis al método utilizado para modelar el sistema.

2.6.2. Otras aproximaciones de modelado para los AGVs

Los *AGVs* son muy importantes en diversos tipos de industrias de tamaño medio y grande que incluyen a los sistemas de manufactura flexible (FMSs), centros de distribución automatizados, lavanderías, etcétera. En el diseño de los sistemas para *AGVs* se toma en consideración muchos aspectos, dependiendo del nivel de decisión con que se esté trabajando. La plataforma de trabajo es uno de los aspectos de diseño más importante, que según [Ganesharajah 98] se puede dividir en 1) Ciclo único, 2) Tipo escalera y 3) Redes complejas. A los *AGVs* capaces de moverse libremente a través de rutas en una plataforma de red compleja se les conoce como *AGVs de rango libre*. Los *AGVs* modernos son en su mayoría de tipo rango libre por la flexibilidad de movimiento que ofrecen, pero requieren la utilización de algoritmos de encaminamiento en la selección de las rutas para alcanzar su destino. Existen diversas aproximaciones y métodos para modelar el sistema *AGV*, siendo [Broadbent 85] uno de los primeros trabajos. Este trabajo utilizó un grafo para modelar la plataforma de trabajo y el algoritmo de caminos más cortos de *Dijkstra* para alimentar una matriz con los tiempos de ocupación del *AGV* en cada ruta. Esta estrategia también permitió controlar la aparición de bloqueos, estableciendo una planificación que evite los conflictos entre los *AGVs*. Un enfoque similar de modelado fue dado por [Gaskins 87] utilizando programación entera binaria para encontrar una ruta que minimizara el viaje de los vehículos. Este trabajo fue la base de otras aproximaciones como la presentada en [Goetz 90] que utilizó programación entera lineal para encontrar la mejor ruta para el *AGV*. Adicionalmente, en en [Goetz 90] se utilizó heurísticas para aprovechar al máximo la estructura de la plataforma y reducir la cantidad de restricciones al modelo. Otro enfoque fue presentado en [Bozer 91] a través de la utilización de zonas de un plataforma, a la que denominaron configuración *Tandem*, en donde existen zonas y cada vehículo sirve a una zona en particular. Las zonas tienen recorridos cíclicos, que son servidas por un solo *AGV* y las estaciones son interfaces entre las zonas. Adicionalmente, en [Sinriech 97] se presentó un concepto más robusto de las zonas, porque se segmentaron en zonas que no se solapan entre si. Este enfoque fue muy similar a la configuración tandem, pero se demostró que era más eficiente que los enfoques anteriores.

Posteriormente, seguimos encontrando la utilización de zonas, pero desde una visión *SAR* y asistidas por la utilización de grafos y las Redes de Petri como modelo formal para el análisis en [Reveliotis 00]. Es preciso mencionar que los trabajos en los que se modelan los *AGVs* y se utilizan las Redes de Petri[Reveliotis 00][Fanti 02][Wu 04][Wu 05], carecen de una perspectiva de modelado. Este hecho provoca que las Redes de Petri no tengan una estructura interna a diferencia de los modelos que se obtienen con nuestra metodología. El trabajo de [Reveliotis 00] se apoyó en un grafo $G = (V, E)$, en donde los vértices V, corresponden a las intersecciones entre las estaciones. Los arcos E, corresponden a las

zonas que desde la visión *SAR* son considerados los recursos del sistema. Este trabajo se apoyó en el concepto de zonas que se aplica a los *Sistemas de Manufactura Flexible (SMF)* para desarrollar una política de control para *AGVs* basada en el algoritmo del banquero. Una estrategia de modelado que también se apoyó en el concepto de zonas disjuntas se presentó en [Fanti 02], pero con un nivel de abstracción superior debido a que se trabajó con Redes de Petri coloreadas. Finalmente tenemos el enfoque presentado en [Wu 04], con Redes de Petri coloreadas en donde también se utilizó el concepto de las zonas. Un grafo representa la plataforma de trabajo, similar al enfoque presentado en [Reveliotis 00]. En [Wu 04] la intersección de las zonas es representado por los lugares de la Red de Petri, y cada arco de entrada a los lugares representa una opción de entrada a la zona que proporciona la plataforma de trabajo. La finalidad de este trabajo es abordar el control de los conflictos entre los *AGVs*, utilizando una política de evitación.

A continuación presentaremos un ejemplo de una plataforma de trabajo de *AGVs*, con su respectivo algoritmo de encaminamiento adaptativo mínimo, pero este algoritmo de encaminamiento es prospenso a tener bloqueos en sus procesos. Este ejemplo será modelado siguiendo nuestra metodología para el diseño de algoritmos de encaminamiento adaptativos mínimos libres de bloqueos.

2.6.3. Ejemplo de una plataforma para AGVs

En nuestro trabajo nos enfocaremos en los *AGV remolque*, que son utilizados para arrastrar vagones de carga que pueden tener diversos tamaños y pesos. Estos equipos son de gran utilidad por la flexibilidad de carga que permiten transportar, pero no pueden operar en sentido inverso lo cual les impide recuperarse de situaciones bloqueo. A continuación introducimos un ejemplo simple de una plataforma de trabajo, similar al introducido en el artículo [Reveliotis 00] o en [Fanti 02]. La siguiente especificación ilustra una situación típica en el diseño de un algoritmo de encaminamiento para un sistema *AGV*. Nuestra plataforma está definida por un sistema de estaciones *ES* y un sistema de segmentos *SE* que interconectan las estaciones. El patrón de la conexión entre nodos será llamado la *topología* de la plataforma. El ejemplo que estamos considerando es una malla como la topología subyacente, con el siguiente conjunto de estaciones $ES=\{es_0, es_1, es_2, es_3\}$ y están connectados por un conjunto de segmentos $SE=\{se_1, se_2, se_3, se_4, se_5, se_6, se_7, se_8\}$. Esta plataforma se representa de manera esquemática en la figura 2.9.a.

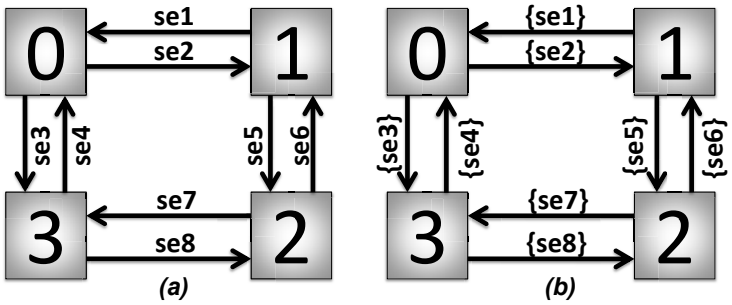

Figura 2.9: Topología para AGV y su Grafo de Interconexión en las figuras *a* y *b* respectivamente.

Como trabajamos con algoritmos de encaminamiento adaptativos mínimos existirá más de un camino para alcanzar al destino y el vehículo puede utilizar cualesquiera de ellos, si el *algoritmo local de encaminamiento* así lo permite. El otro elemento de definición de la red de interconexión es el tipo de *control del convoy* que establece el sistema de asignación del segmento. Como el *AGV* está compuesto por más de un elemento y ellos viajan en estilo convoy el *control del convoy* se encarga de asignar los segmentos al primer vehículo que será liberado por el último vagón del convoy. Debido a esta situación el primer vehículo se le conoce como la cabeza **(head)** que reserva los segmentos y al último como la cola **(tail)** que libera el segmento que acaba de atravesar. Dado el comportamiento de estilo convoy, el *AGV* puede mantener reservado simultáneamente varios segmentos mientras alcanza la estación destino. En nuestro ejemplo cada estación controla coordinadamente un segmento y ejecuta una instancia del algoritmo de encaminamiento adaptativo mínimo. Cada estación tiene una identidad local única y el algoritmo de encaminamiento utilizará como parámetro de entrada la estación destino que se conoce cuando llega el primer vehículo a la estación actual de trabajo.

El Algoritmo 3 es un algoritmo de encaminamiento adaptativo mínimo que mezcla el conocido algoritmo estático *XY* y una libre selección de rutas para determinadas estaciones. Se puede expresar informalmente de la siguiente forma: Si el convoy está destinado para la estación actual, se debe recibir el convoy. Si no es el caso, se debe preguntar si la estación destino es par y en tal caso puede usar cualquier canal disponible de salida de forma adaptativa. Si la estación destino no es par entonces utilizará las rutas verticales Y y después las horizontales X, hasta llegar a la estación destino. Esta reserva la realiza el vehículo cabeza del convoy al llegar al nodo actual de la plataforma y los siguientes vagones intermedios seguirán a través de los segmentos reservados. Este proceso ocurre hasta que el vagón cola libere todos los segmentos usados y que por tanto se sumarán al conjunto de segmentos libres F o no asignados.

Algoritmo 3 Algoritmo de encaminamiento para la estación i.

Entrada: El destino dado por la cabeza el convoy co.
Local: $SE_i \subseteq SE$, conjunto de segmentos de salida para la estación es_i.
$L \subseteq SE_i$, conjunto de segmentos no asignados.
Salida: El próximo segmento para ser usado por co

1. **Si** $(destino(co) = es_i)$ **Entonces**
2. *almacene el* **co** *en* es_i
3. **Sino**
4. **Si** $(\ destino(co) mod 2 = 0)$ **Entonces**
5. Elegir un canal de SE_i.
6. **Sino**
7. Según política YX, seleccionar un canal de es_i.
8. **Fin Si**
9. **Fin Si**

El diseño de algoritmos de encaminamiento para los $AGVs$, puede llevar a soluciones en las que se alcancen estados de bloqueos en el movimiento de los vehículos. Un estado de bloqueo en una plataforma se presenta cuando un conjunto de convoys en tránsito se detienen para siempre en las estaciones intermedias de una plataforma y ninguno de ellos ha alcanzado sus respectivas estaciones destino. Todos estos vehículos están a la espera de la disponibilidad de segmentos de salida en estas estaciones intermedias y que han sido previamente asignados a otros convoys que pertenecen a este mismo conjunto. Por lo tanto, ninguno de los vehículos implicados alcanzará a sus estaciones destino.

El algoritmo 3, de nuestro ejemplo presenta esta anomalía que se ilustra mediante la siguiente configuración de bloqueo en el cuadro 2.3. En este cuadro representamos cuatro vehículos en una configuración alcanzable porque cada uno está compuesto por más de un vagón por lo tanto cada convoy ocupará simultáneamente más de un segmento. En el cuadro 3, la columna **Vagones** las siglas e_h y e_t representan las estaciones actuales del vagón cabeza y cola, respectivamente; **Segmentos asignados**: Segmento asignado al convoy; **Estación Destino**: representa la estación destino del convoy, **Algoritmo de encaminamiento**: Segmento que se asignará al vagón cabeza según el algoritmo de encaminamiento.

Observe que todos los vehículos están en las estaciones intermedias y con el fin de avanzar en la plataforma, todos los vehículos necesitan segmentos *(los propuestos por el algoritmo de encaminamiento)* y que todos ellos están ya asignados por otros vehículos en el mismo conjunto. Esto se puede apreciar al comparar las dos columnas **Segmentos Asignados** y **Algoritmo de Encaminamiento** en el cuadro 2.3. Por otra parte, estamos asumiendo que el convoy utiliza más de un segmento y por lo tanto ninguno de los vagones

Número Convoy	Vagón		Segmentos Asignados	Estación Destino	Algoritmo de Encaminamiento
	e_h	e_t			
co_1	1	0	se_2	2	se_5
co_2	2	1	se_5	3	se_7
co_3	3	2	se_7	0	se_4
co_4	0	3	se_4	1	se_2

Cuadro 2.3: Estado de bloqueo alcanzado por los vehículos de la plataforma.

cola puede liberar los segmentos, porque si alguno de estos flits cola avanza estará en la misma estación que la cabeza. Por lo tanto, hemos llegado a un estado de bloqueo en que se cumplen todas las condiciones clásicas [Coffman 71] para la existencia de un bloqueo. Por último, se puede observar que, si bien estamos en un estado de bloqueo, existen cuatro canales, se_1, se_3, se_6 y se_8, que están libres pero que el algoritmo de encaminamiento no puede asignar ninguno de estos a los cuatro vehículos de nuestro escenario.

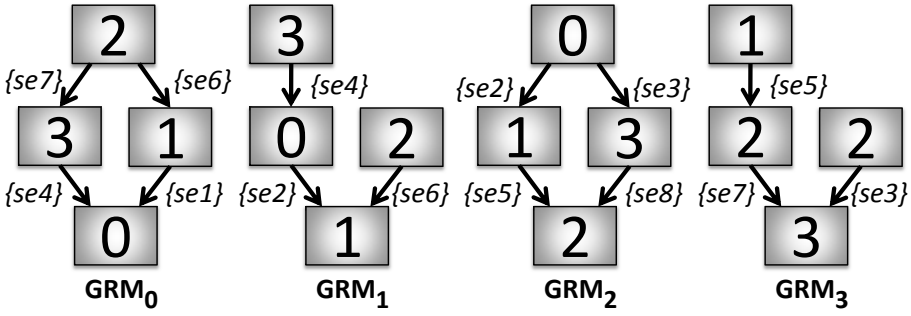

Figura 2.10: Grafo de Rutas Mínimas del ejemplo de la subsección 2.6.3.

En el Grafo de Rutas Mínimas se puede apreciar la adaptabilidad del algoritmo de encaminamiento, por la existencia de más de una ruta mínima para llegar al destino como es el caso de las estaciones destino *cero* y *dos*. La clase de *SARs* que pueden ser modeladas con las redes S^4PR permiten modelar decisiones en línea y rutas adaptativas y el uso de múltiples recursos en cada paso del procesamiento. El grafo de encaminamiento de nuestra metodología para este ejemplo los hemos omitido debido a la simplicidad del mismo y gran parecido al grafo de Control de Flujo. La figura 2.11 muestra el modelo de la red de Petri obtenido usando nuestra metodología que abstrae el sistema *AGVs* con una visión *SAR* del mismo. Es preciso hacer notar que las múltiples rutas para alcanzar un destino están representadas en la Red de Petri.

Es preciso mencionar que siguiendo nuestra metodología, cada segmento de la plataforma equivale a un recurso del sistema que es modelado con un lugar recurso $p_1 ... p_8$, que tendrá una marca indicando su disponibilidad o no. Esta definición está conforme a la definición 2, para el marcado aceptable de la clase de redes de Petri con la que trabajamos. Lo mismo lo podemos observar en los lugares reposo p_{29}, p_{30},p_{31} y p_{32}, en los cuales las marcas representan la cantidad de procesos de encaminamiento que se pueden ejecutar en aislamiento. Cada uno de estos lugares representa el encaminamiento a una estación destino de la plataforma de trabajo que en este caso son cuatro estaciones de trabajo. Finalmente, los lugares proceso están vacios debido a que no existe ninún proceso de encaminamiento de objetos representándose en la red de Petri de la figura 2.11, y es coherente con la totalidad de marcas en los lugares resposo y de recursos de la misma figura.

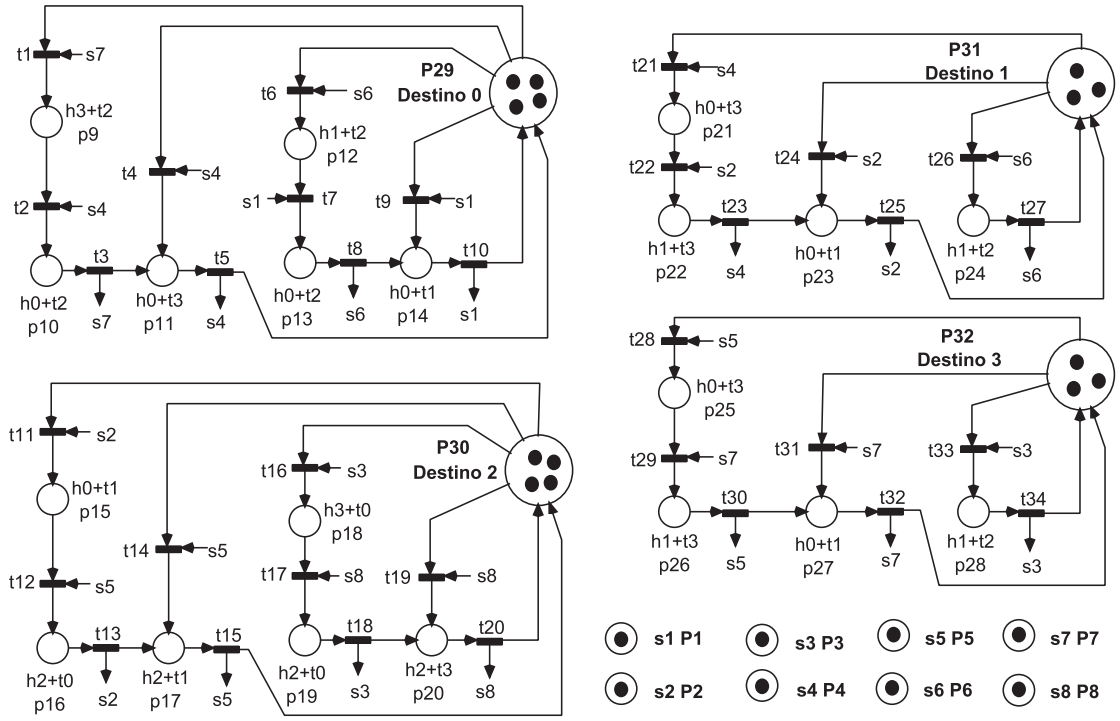

Figura 2.11: Red de Petri del AGV del ejemplo de la subsección 2.6.3.

2.7. Conclusión

Las diferentes aproximaciones existentes para abordar el problema del bloqueo, en los algoritmos de encaminamiento carecen de una metodología de diseño completa. El diseño de algoritmos de encaminamiento libres de bloqueos es una tarea *compleja* y *tediosa* [Dally 03], para la cual existen actualmente métodos con técnicas del tipo *ensayo y error* para saber si existen bloqueos o no. En este capítulo hemos presentado nuestra metodología para el diseño de algoritmos de encaminamiento adaptativos mínimos libres de bloqueos, en la que se ha utilizado las Redes de Petri como herramienta formal de modelado, análisis y síntesis de los modelos obtenidos. El primer paso de esta metodología consiste en la abstracción del sistema desde una visión *SAR*, para conservar sólo los elementos que nos permiten al estudio de la aparición de bloqueos. Estos elementos son los canales de comunicación para el caso de las redes de interconexión y los segmentos para el caso de los vehículos guiados automáticamente. La asignación de los recursos para el encaminamiento de mensajes u objetos dependerá del algoritmo de encaminamiento del tipo adaptativo mínimo.

Las diversas etapas del método de contrucción de la red de Petri, están asistidas por el uso de grafos, con sus respectivos algoritmos de construcción, hasta la obtención de la Red de Petri. La clase de Redes de Petri obtenida nos proporciona una caracterización de la propiedad de la vivacidad. En el caso de la no vivacidad del modelo, la etapa de síntesis de la metodología permite forzar la propiedad de vivacidad. Tomando en cuenta el tipo de modelos obtenidos con nuestra metodología, hemos presentado la interpretación semática de ellos, así como ciertas caracteríticas propias de los modelos. Estas características nos permiten aprovechar las analogías y diferencias con el dominio de aplicación de los *vehículos guiados automáticamente*. Finalmente, se presentó un ejemplo para este dominio de aplicación en el que se obtuvo su respectivo modelo en Redes de Petri, siguiendo nuestro método de construcción.

Capítulo 3

La clase de redes $SOAR^2$: definición y propiedades

Resumen
El encaminamiento de mensajes u objetos en las *redes de interconexión* y los *vehículos guiados automáticamente* respectivamente, se puede modelar con las Redes de Petri, utilizando una abstracción tipo asignación de recursos (SAR). Los modelos de red obtenidos para estos sistemas pertenecen a la clase de redes S^4PR, sin embargo tienen características particulares, motivo por el cual es conveniente definir una clase de Red de Petri más precisa para así aprovechar sus características comportamentales y estructurales apropiadamente. Atendiendo este objetivo, en este capítulo se definirá la clase de Redes de Petri $SOAR^2$ y se presentará sus características y propiedades estructurales más relevantes. Esto incluirá la definición de conceptos propios de esta clase como lo son las zonas de recursos y el solapamiento de las mismas. Finalmente presentaremos las propiedades más relevantes para esta clase de redes.

3.1. Introducción

El modelado de sistemas de encaminamiento ya sean algoritmos de encaminamiento de mensajes en redes de interconexión de multicomputadores o redes para el desplazamiento de objetos mediante vehículos guiados automáticamente, requiere un proceso de abstracción que retenga del sistema real sólo aquellos aspectos relacionados con el proceso

mismo de encaminamiento de mensajes u objetos. En este sentido, en [Rovetto 10a] y [Rovetto 10b] se presentan sendos procesos de abstracción que permiten obtener Redes de Petri como modelos que representan estos sistemas para estudiar los problemas en el encaminamiento de mensajes u objetos como consecuencia de la compartición de recursos (canales o vías) que deben ser utilizados en exclusión mutua. Esto es, las redes de Petri obtenidas son una Abstracción tipo Sistema de Asignación de Recursos *SAR* (Resource Allocation System Abstraction - RAS Abstraction) del sistema real a estudiar. En estos artículos, los autores prueban que las redes obtenidas pertenecen a la bien conocida clase de las redes S^4PR, siguiendo una metodología presentada por ellos.

En este capítulo trataremos de obtener una caracterización más precisa de la clase de redes que se pueden obtener de la abstracción de sistemas de encaminamiento. El objetivo es aprovechar las particularidades de la nueva clase a definir para obtener resultados de análisis más potentes que permitan mejorar las prestaciones de los algoritmos involucrados en el análisis de la propiedad de ausencia de bloqueos y los consecuentes algoritmos para forzar la propiedad de vivacidad en los modelos de partida. En otras palabras, vamos a definir una subclase de las redes S^4PR que denominaremos $SOAR^2$ (S^4PR *with Ordered Allocation and Release of the Resources*), es decir, redes S^4PR con Asignación y Liberación Ordenada de Recursos. El orden en la asignación y liberación de los recursos está establecido por el comportamiento de los mensajes u objetos cuando se desplazan por la red de interconexión o la plataforma de trabajo de los *AGV s* respectivamente.

Este capítulo se organiza de la siguiente forma. Las características comportamentales que fundamentan la definición de la clase se detallan en la sección 3.2. Estas características se traducen en restricciones estructurales de la clase de redes S^4PR, utilizando el concepto denominado *zona de uso continuado de un recurso*. El solapamiento de estas zonas es fundamental para describir el comportamiento ordenado en la adquisición y liberación de los recursos y es utilizado en la definición estructural de la clase de redes $SOAR^2$. Las propiedades de estas zonas en la clase de redes $SOAR^2$ y su relación con los cerrojos se desarrollará en la sección 3.3. En esta sección también se definirá y utilizará los conjuntos máximos de zona solapadas pero restringidas a un subconjunto de recursos a tal fin de generar los cerrojos mínimos de la red para la clase de redes $SOAR^2$.

3.2. La clase $SOAR^2$

En esta sección definiremos la clase de Redes de Petri $SOAR^2$, utilizando argumentos estructurales exclusivamente para el modelado de sistemas relacionados con

el encaminamiento de objetos/datos entre una estación/nodo fuente y una estación/nodo destino a través de vías para el guiado de vehículos o una red de interconexión compuesta por canales de comunicación.

3.2.1. Justificación de las características a modelar con las redes *SOAR*^{2}

Para justificar las características especiales que motivan la definición de la clase de redes *SOAR*^{2}, como una subclase de las redes S^4PR, utilizaremos nuestra visión *SAR* del sistema.

1. El orden en el que los recursos (canales de comunicación de datos o vías para el guiado de vehículos) son solicitados y asignados al seguir una ruta determinada, es el mismo orden en el que estos recursos son posteriormente liberados. Por ejemplo, en un algoritmo de encaminamiento de mensajes en una red de interconexión de multicomputadores el flit de cabeza reserva los canales de comunicación necesarios para avanzar por la red hacia su destino conforme al algoritmo de encaminamiento implementado tipo *wormhole*. Los canales reservados y posteriormente asignados al mensaje son mantenidos mientras existan flits intermedios del mensaje que deban avanzar siguiendo al flit de cabeza. El flit de cola es el encargado de liberar los canales previamente asignados, y dado el caracter secuencial del movimiento de la cadena de flits por la red el orden de liberación de dichos canales será el mismo orden en el que fueron asignados: el primer canal que fue asignado al flit de cabeza será también el primer canal liberado por el flit de cola.

2. Un objeto o un dato siguiendo una ruta puede tener asignados, y por lo tanto estar usando, simultáneamente un conjunto de recursos. En el caso de redes de interconexión de multicomputadores con algoritmos de encaminamiento tipo *wormhole*, el mensaje se descompone en un conjunto de unidades más elementales denominadas flits y cada uno puede avanzar en paralelo con el resto siguiendo la misma ruta. Para lograr esta concurrencia en el avance, el mensaje debe tener asignados varios canales y el número de canales asignados será lo que permita definir el grado de paralelismo en el avance de flits por la red. De la misma forma si se piensa en redes de vehículos guiados automáticamente, estos vehículos están compuestos por varias unidades de forma que constituye un convoy de vehículos más pequeños cada uno de los cuales ocupa un segmento de la vía de forma exclusiva. El conjunto de segmentos ocupados por el convoy que forma el vehículo es el conjunto de recursos asignados simultáneamente al vehículo.

3. El objeto o datos que está moviéndose a través de la red de interconexión no puede realizar más que una operación de asignación o de liberación de una una única copia de un tipo de recurso. Es decir, no puede existir simultaneidad en las operaciones de asignación y liberación de recursos, y en cada operación permitida sólo hay implicada una copia de un tipo de recurso (un canal o una vía de la red de comunicación). Dicho de otra forma, dado que las operaciones de asignación y liberación son realizadas por partes distintas del mensaje (la asignación se realiza por el flit de cabeza y la liberación por el flit de cola) y separadas geográficamente (cada flit se encuentra en un nodo diferente y separado en la red de interconexión) dichas operaciones se realizarán asíncronamente y además separadamente (ausencia de simultaneidad).

4. Para cada tipo de recurso (canal de comunicación o vía en la red de interconexión) existe una única copia del mismo, o si se quiere cada recurso del sistema se configurará como un tipo diferente de recurso y él mismo es la única copia capaz de satisfacer la solicitud del recurso.

Obviamente, las características anteriores se pueden transformar fácilmente en restricciones estructurales sobre la clase de redes S^4PR, como se muestra informalmente a continuación,

Restricción 1. En cada circuito de cada máquina de estado de la red S^4PR, los recursos son liberados en el mismo orden en el que fueron asignados.

Restricción 2. En cada transición de la red solo se puede encontrar un recurso como máximo conectado con esta transición. Si el arco es de entrada a la transición desde el lugar recurso, se trata de una operación de asignación del recurso al proceso. Si se trata de un arco de salida desde la transición al lugar recurso, es una operación de liberación del recurso por parte del proceso.

Restricción 3. Los arcos de entrada y de salida de los lugares recurso son todos ordinarios, es decir, todos tienen peso asociado igual a uno.

Restricción 4. El marcado inicial admisible de todos los recursos es igual uno, indicando que solo existe una copia del tipo de recurso que modela el lugar y además ésta se encuentra disponible en el estado inicial del sistema de encaminamiento (en el estado inicial no existen mensajes o vehículos en tránsito).

De todas las restricciones anteriores, las tres últimas son relativamente fáciles de introducir ya que en realidad se trata de restringir estructuralmente los apartados

tradicionales en la definición de las redes S^4PR. No obstante, la primera restricción resulta especialmente difícil de introducir en la definición de una red S^4PR por dos motivos. El primero es que se trata de una restricción que se aplica a un tipo de objeto cuyo número puede ser extraordinariamente grande en una red S^4PR, dado que el número de circuitos puede ser un número exponencial con respecto al tamaño de la máquina de estados de entrada. Por lo tanto, la verificación de tal propiedad puede ser muy onerosa desde el punto de vista computacional. La segunda razón es que una definición basada en esta propiedad requiere definir una relación de orden sobre el conjunto de recursos que podemos encontrar sobre un circuito y esto no se puede hacer de forma independiente del resto de los circuitos.

Para evitar los problemas anteriores, y de cara a construir la definición de la clase de redes $SOAR^2$, en las subsecciones siguientes definimos un nuevo concepto en el seno de la clase de redes S^4PR que denominaremos: *zona de uso continuado de un recurso en un proceso de encaminamiento* o simplemente *zona de un recurso*. Las propiedades de estos objetos individualmente considerados y las relaciones que definiremos en el conjunto de zonas en redes S^4PR, nos permitirán definir las restricciones que tienen que cumplir las zonas para considerar una red como perteneciente a la clase de las redes $SOAR^2$.

3.2.2. Zona de uso continuado de un recurso en redes S^4PR y su cálculo

Una zona de uso continuado de un recurso r es una subred conexa máxima en la que se han asignado copias del recurso r a un proceso activo que representa un mensaje o un vehículo en movimiento. A continuación se formaliza dicha definición para las Redes de Petri de la clase S^4PR.

Definición 6. *Sea $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ una red S^4PR y $r \in P_R$ un lugar recurso de $\mathcal{N}$. Una zona j de uso continuado del recurso r en la i-ésima máquina de estados de $\mathcal{N}$ es el conjunto máximo de lugares usuarios de r, $Z^r_{i,j} \subseteq H_r \cap P_{S_i}$, tal que la subred generada por $Z^r_{i,j} \cup ((Z^r_{i,j})^\bullet \cap {}^\bullet(Z^r_{i,j}))$ es una máquina de estados conexa.*

Cuando por el contexto se sobreentienda, y de cara a aligerar la notación, nos referiremos a una zona j de uso continuado del recurso r en la i-ésima máquina de estados de $\mathcal{N}$, como zona j de r en la máquina i. En el caso que sólo exista una zona de r en la máquina i, prescindiremos del índice j y si además se sobreentiende la máquina de estados a la que estamos haciendo referencia prescindiremos también del índice i.

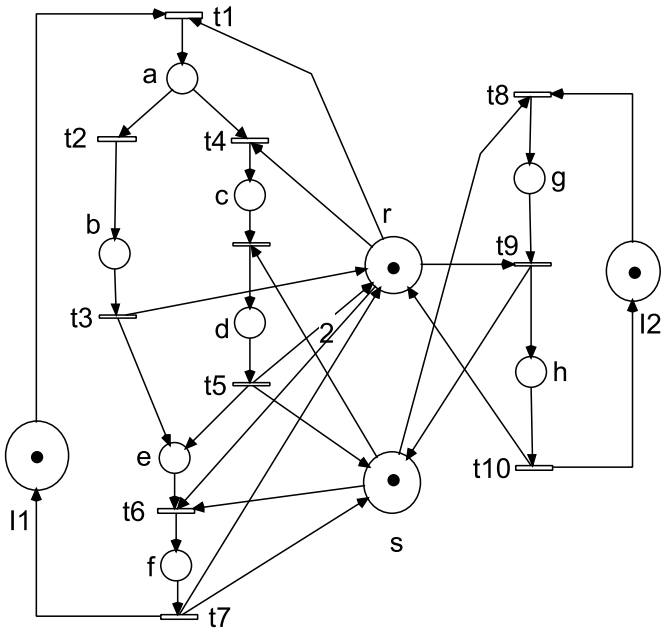

Figura 3.1: Una red S^4PR.

Es importante resaltar que el concepto de zona de un recurso no viene determinado por la máquina de estado en la que se encuentra. De hecho para un mismo recurso podemos encontrar varias zonas definidas sobre una misma máquina de estado, aunque obviamente estas zonas deben ser disconexas, es decir, no pueden compartir ningún lugar.

En la red de Petri S^4PR de la figura 3.1 se pueden identificar hasta tres zonas diferentes para el recurso r: $Z^r_{1,1} = \{a,b,c,d\}$, $Z^r_{1,2} = \{f\}$ y $Z^r_{2,1} = \{h\}$. Así mismo, para el recurso s también se pueden identificar 3 zonas distintas: $Z^s_{1,1} = \{d\}$, $Z^s_{1,2} = \{f\}$ y $Z^s_{2,1} = \{g\}$. A partir de la definición de zona y visto el ejemplo anterior, es fácil deducir que un mismo lugar puede pertenecer a 2 zonas distintas pero estas zonas deben corresponder a recursos diferentes, ya que zonas distintas de un recurso son disjuntas. Por otra parte, por la Definición 6, se verifica que un lugar reposo nunca puede pertenecer a una zona ya que los lugares reposo no pueden ser usuarios de ningún recurso.

De acuerdo con las propiedades anteriores, a continuación se presentan las cotas superior e inferior del número de zonas de un recurso que podemos encontrar en una red S^4PR.

Lema 2. *Sea $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ una red S^4PR y sea $I_{\mathcal{N}}$ el conjunto de índices que identifican cada una de las máquinas de estado de $\mathcal{N}$. Si $\mathscr{Z}^r$ es el conjunto de zonas del recurso r en $\mathcal{N}$, entonces el cardinal de $\mathscr{Z}^r$ está acotado por los siguientes valores:*

$$1 \leq |\mathscr{Z}^r| \leq \sum_{i \in I_{\mathcal{N}}} \left\lceil \frac{|P_i|}{2} \right\rceil$$

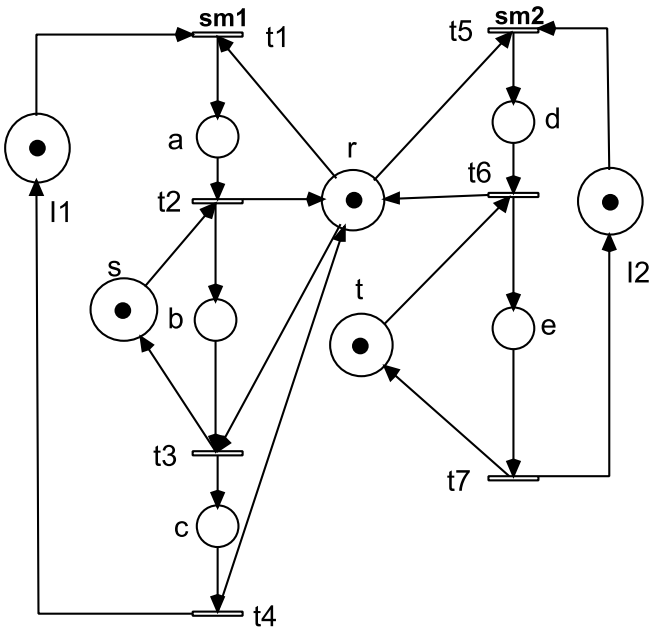

Figura 3.2: Una red S^4PR que permite verificar que las cotas establecidas en el Lema 2 son alcanzables.

Las cotas anteriores son alcanzables como se puede comprobar fácilmente en la red de la figura 3.2 que tiene dos máquinas de estado denominadas sm_1 y sm_2 y dos recursos denominados *r* y *s*.

$$\begin{aligned} |\mathscr{Z}^r| &= \left\lceil \frac{3}{2} \right\rceil + \left\lceil \frac{2}{2} \right\rceil = 3 \\ |\mathscr{Z}^s| &= |\mathscr{Z}^t| = 1 \end{aligned}$$

A partir de esta acotación podemos concluir que en el peor de los casos el número máximo de zonas que podemos encontrar en una red S^4PR viene acotado por los valores indicados en el siguiente resultado.

Lema 3. *Sea $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ una red S^4PR y sea $I_{\mathcal{N}}$ el conjunto de índices que identifican cada una de las máquinas de estado de $\mathcal{N}$. Si $\mathscr{Z}^{P_R}$ es el conjunto de zonas de la red $\mathcal{N}$, entonces el cardinal de $\mathscr{Z}^{P_R}$ está acotado por los siguientes valores:*

$$|P_R| \quad \leq \quad |\mathscr{Z}^{P_R}| \leq |P_R| \cdot \left(\sum_{i \in I_{\mathcal{N}}} \left\lceil \frac{|P_i|}{2} \right\rceil \right)$$

Esta cota del número de zonas resuelve la primera dificultad planteada en la *Restricción 1* presentada anteriormente para definir la subclase de redes $SOAR^2$. En efecto, esta cota es del orden de $|P_R| \cdot |P_S|$, mientras que los circuitos de una máquina de estados puede resultar en un número combinatorio con respecto al número de lugares de la misma (número prohibitivo en algunos casos). Es por ello que en la definición de la subclase utilizaremos las zonas definidas en esta sección en lugar de los circuitos, aunque estos últimos puedan tener una interpretación más intuitiva.

Para calcular el conjunto de zonas de cada recurso $r \in P_R$, se presenta el algoritmo 4 que procede de una forma iterativa a partir del conjunto soporte del p-semiflujo del recurso r. La idea básica consiste en que todas las zonas son conjuntos de lugares usuarios del recurso r y por lo tanto, todas las zonas estarán contenidas en el soporte del p-semiflujo mínimo de r.

3.2.3. Zonas binarias de recursos

Si se considera la subred generada por una zona Z^r y las transiciones $(Z^r)^\bullet \cap {}^\bullet(Z^r)$, en el caso de redes S^4PR generales, se puede encontrar casos en los que transiciones $t \in (Z^r)^\bullet \cap {}^\bullet(Z^r)$ puede tener el lugar r como entrada o como salida. En el caso de los sistemas que estamos considerando, esto es, sistemas de encaminamiento de datos o de vehículos, se ha establecido anteriormente que en cada transición sólo se puede tener como máximo un recurso de entrada o de salida pero no ambas cosas a la vez. Por otra parte, también se ha establecido que en las operaciones de asignación y liberación de un recurso (que se representan por la ocurrencia de una transición) sólo puede haber implicada una copia del recurso (es decir, sólo se asigna o libera un canal que corresponde al canal que necesita el dato para realizar el movimiento dentro de la red de interconexión). Todos estos comentarios permiten poner de manifiesto que las zonas de recursos en S^4PR que representan sistemas de encaminamiento tienen una estructura muy particular que las hace

Algoritmo 4 Cálculo del conjunto de zonas $\mathscr{Z}$ de una red S^4PR.

Entrada:

1. La red $\mathscr{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ que es una red S^4PR;
2. El conjunto de p-semiflujos mínimos de la red $\mathscr{N}$ correspondientes a los recursos: $\{\mathbf{y}_r | r \in P_R$ e $\mathbf{y}_r$ es un p-semiflujo mínimo de$\mathscr{N}\}$;

Salida:

3. El conjunto de zonas de $\mathscr{N}$:
4. $\mathscr{Z} = \{Z^r_{i,j} |$ es la j-ésima zona de $r \in P_R$, en la i-ésima máquina de estado de $\mathscr{N}\}$;
5. **Inicio**
6. $\mathscr{Z} = \emptyset$;
7. **Para todo** $r \in P_R$ **Hacer**
8. $\quad WS = \|\mathbf{y}_r\| \setminus \{r\}$;
9. $\quad$ Seleccionar un lugar $p \in WS$;
10. $\quad Z = \{p\}$; $WS = WS \setminus \{p\}$;
11. $\quad$ **Repetir**
12. $\quad\quad A = (Z^{\bullet\bullet} \cup {}^{\bullet\bullet}Z) \cap WS$;
13. $\quad\quad$ **Si** $A \setminus Z \neq \emptyset$ **Entonces**
14. $\quad\quad\quad WS = WS \setminus A$; $Z = Z \cup A$;
15. $\quad\quad$ **Sino**
16. $\quad\quad\quad \mathscr{Z} = \mathscr{Z} \cup \{Z\}$;
17. $\quad\quad\quad$ Seleccionar un lugar $p \in WS$;
18. $\quad\quad\quad Z = \{p\}$; $WS = WS \setminus \{p\}$;
19. $\quad\quad$ **Fin Si**
20. $\quad$ **Hasta que** $WS = \emptyset$
21. $\quad \mathscr{Z} = \mathscr{Z} \cup \{Z\}$;
22. **Fin Para**
23. **Fin**

especialmente simples dado que las zonas surgen de p–semiflujos mínimos de los recursos que son binarios, es decir, p–semiflujos $\mathbf{y}_r \in \{0,1\}^{|P_0 \cup P_S \cup P_R|}$, para todo $r \in P_R$.

Esta forma particular de las zonas se debe, por tanto, a que están formadas por lugares proceso que son usuarios de un recurso cuyo p-semiflujo mínimo, de acuerdo con la definición de redes S^4PR, tiene sus pesos dentro del conjunto $\{0,1\}$. Esta propiedad se formula en el siguiente resultado que será de capital importancia a la hora de construir la clase $SOAR^2$. Las zonas así caracterizadas se denominarán *zonas binarias de uso continuado de recursos* o simplemente *zonas binarias de recursos*.

Lema 4 (Zonas binarias de uso continuado recursos)**.** *Sea $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ una red S^4PR y sea r un lugar recurso de la red $\mathcal{N}$ tal que el correpondiente p-semiflujo mínimo de r verifica que $\mathbf{y}_r \in \{0,1\}^{|P_0 \cup P_S \cup P_R|}$. Toda zona del recurso r, $\mathcal{Z}^r$, verifica que,*

1. *El conjunto de transiciones de la subred generada por Z^r y $\tau = (Z^r)^\bullet \cap {}^\bullet(Z^r)$ no tienen al lugar r como lugar de entrada o de salida: $r \notin {}^\bullet\tau$, y $r \notin \tau^\bullet$.*

2. *Para toda transición $t \in {}^\bullet(Z^r) \setminus \tau$, $r \in {}^\bullet t$.*

3. *Para toda transición $t \in (Z^r)^\bullet \setminus \tau$, $r \in t^\bullet$.*

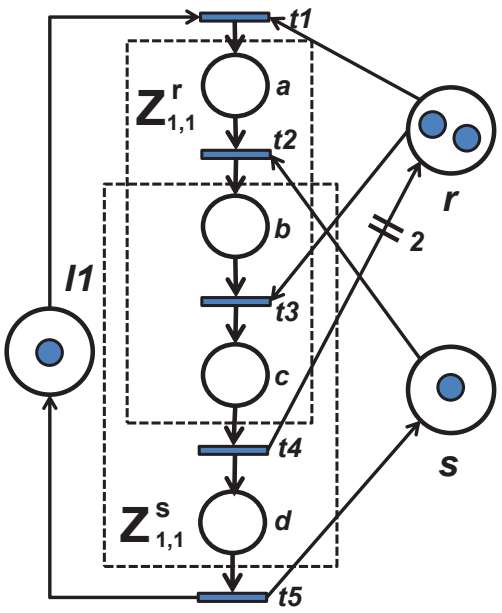

Figura 3.3: Ejemplos de zonas de uso continuo de recursos y sus relaciones.

Caracterizada la estructura particular de las zonas binarias de recursos en las redes S^4PR obtenidas de los sistemas de encaminamiento, en la sección siguiente analizamos como estudiar a partir de las zonas binarias la relación de orden en la asignación de un conjunto de recursos y la verificación que esta relación de orden se respeta en la liberación de los recursos.

3.2.4. Solapamiento de zonas binarias de recursos en redes S^4PR

Esta sección está dedicada a capturar las cuestiones relacionadas con la preservación del orden de asignación de un conjunto de recursos en la liberación de los mismos. Esta preservación se va a caracterizar por medio de las relaciones existentes entre las zonas asociadas a esos recursos. Hay casos de redes S^4PR en los que no existe una relación de orden en la asignación de recursos, y esto es debido, en algunos casos a la existencia de zonas de uso continuo de recursos en las que se toma más de una copia del recurso y esta zona se encuentra entrelazada con otras zonas. La red de la figura 3.3 ilustra una de estas situaciones. En efecto, en esta red existen dos zonas de recursos, $Z^r_{1,1}$ y $Z^s_{1,1}$, se puede apreciar que el recurso r se asigna antes y después que la asignación del recurso s dado que la ejecución de un proceso requiere dos copias del recurso tipo r y una copia del recurso tipo s. Esto implica que no podamos ordenar la asignación de los recursos de r y s con una relación de orden total, y mucho menos, tratar de analizar la preservación de la relación de orden. Para los sistemas de encaminamiento esta situación implica que un canal de comunicación o segmento de vía es utilizado consecutivamente dos veces por un mensaje u objeto. Esta situación es lógicamente contraria a la realidad que existe en estos dominios de aplicación. Por el contrario, en estos dominios de aplicación, los recursos son liberados en el mismo orden en que han sido adquiridos de forma unitaria por los procesos de encaminamiento de mensajes u objetos.

Por las razones anteriores, y atendiendo a que estos casos no se pueden presentar en las redes de Petri derivadas de los sistemas de encaminamiento, restringiremos nuestro estudio de las relaciones de orden en la asignación de recursos y su preservación en las operaciones de liberación al caso de las zonas binarias.

En la figura 3.4 se presentan algunos ejemplos de zonas binarias de recursos y las relaciones entre ellas. En la figura 3.4.a las tres zonas binarias de recursos diferentes que existen son: $Z^r_{1,1} = \{a,b,c\}$, $Z^s_{1,1} = \{b,c,d\}$ y $Z^t_{1,1} = \{c,d,e\}$. Cada una de estas zonas binarias representan el uso continuado de un recurso a lo largo de varios lugares proceso vecinos. En la figura se representa una parte de una ruta que arranca en la transición t_1 y termina en la transición t_6. Se puede observar que al iniciar esta ruta se toman los recursos r, s y t y además en este orden. Estas operaciones de adquisición de los recursos se realizan mediante el disparo sucesivo de las transiciones t_1, t_2 y t_3, respectivamente. Este orden

de asignación de los recursos se respeta cuando este conjunto de recursos se libera. Este extremo se puede verificar en la red porque de cara a liberar estos recursos se debe disparar sucesivamente las transiciones t_4, t_5 y t_6, que liberan los recursos r, s y t, respectivamente. Desde el dominio de aplicación, esta adquisición consecutiva de tres recursos, se puede interpretar como un mensaje u objeto que utiliza simultáneamente estos tres recursos.

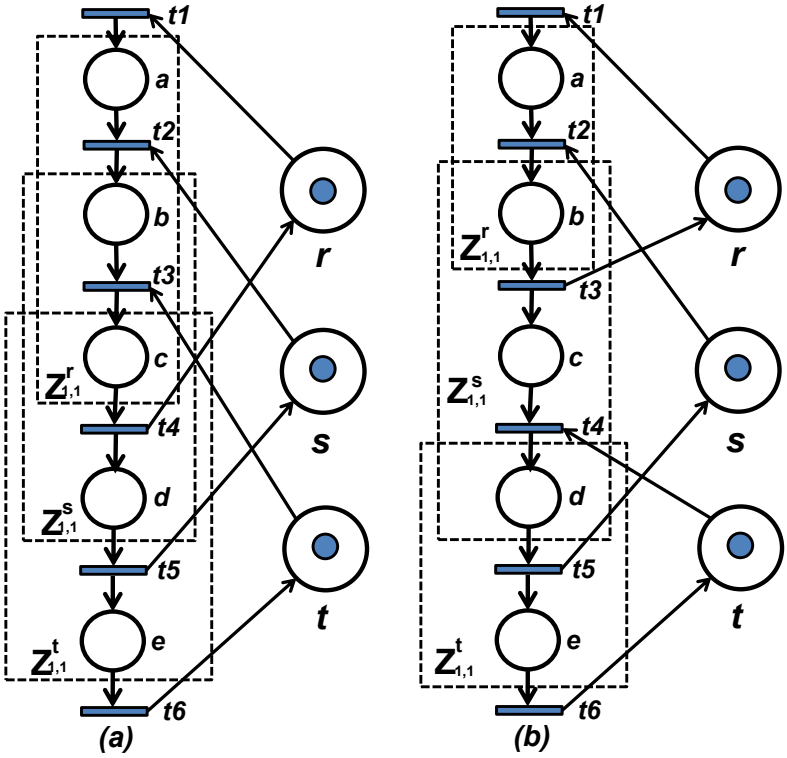

Figura 3.4: Ejemplos de zonas de uso continuo de recursos y sus relaciones.

En la figura 3.4.b existen también tres zonas binarias de recursos diferentes: $Z^r_{1,1} = \{a,b\}$, $Z^s_{1,1} = \{b,c,d\}$ y $Z^t_{1,1} = \{d,e\}$; pero estas están relacionadas de forma diferente. En efecto, en la red de la figura 3.4.a las tres zonas llegaban a estar simultáneamente solapadas en el lugar c de la red, sin embargo esto no ocurre en la red de la figura 3.4.b. Esto significa que a la hora de analizar el orden de asignación o liberación de los recursos solo tendremos que estudiarlo entre los recursos del conjunto $\{r,s\}$ y entre los recursos del conjunto $\{s,t\}$. En otras palabras, el recurso r y el recurso t no hay que analizar la preservación de la relación de orden de la asignación en la liberación dado que no hay ningún estado del proceso representado en la figura en el que se usen simultáneamente una copia de r y una

copia de t. Desde el dominio de aplicación, la adquisición consecutiva de dos recursos en la figura 3.4.b, se puede interpretar como un mensaje u objeto que su longitud exige como máximo dos recursos simultáneamente.

A partir de los ejemplos anteriores queda claro que hay dos temas que deben ser analizados para caracterizar la relación de orden que hay que mantener desde la adquisición de los recursos hasta su liberación. Estos dos temas son:

A. Determinar el conjunto de recursos sobre el que hay que definir la relación de orden en la asignación de los mismos, y que hay que respetar en la liberación de los mismos.

B. Determinar el procedimiento a utilizar para verificar que esta relación de orden se respeta sobre este conjunto en la asignación y la liberación.

Con respecto al primer tema, diremos que dos recursos pertenecen a un mismo conjunto de recursos en el que es necesario determinar el orden de asignación de los recursos de cara a verificar que este orden es respetado en la fase de liberación de esos recursos, si estos dos recursos son usados simultáneamente por un proceso activo en un estado determinado alcanzado por ese proceso activo. Siendo la condición apuntada una condición necesaria, la mera pertenencia al conjunto no garantiza que la relación de orden en la asignación sea la misma que la relación de orden en la liberación. Para caracterizar esta situación utilizaremos las zonas de estos recursos en lugar de los recursos directamente. Además, en todo lo que sigue supondremos que las zonas que estamos considerando son todas zonas binarias de recurso, salvo que se especifique lo contrario.

De acuerdo con esto, diremos que dos recursos pertenecen a un mismo conjunto de recursos a ordenar si existen dos zonas, cada una asociada a uno de los recursos bajo consideración, que tienen una intersección no nula. Esta condición evidentemente implica la condición anteriormente anunciada: existe un estado en el que se llega a encontrar un proceso activo y en el que los dos recursos son usados simultáneamente por el proceso activo. La condición anterior se refiere a dos recursos, pero en nuestro caso estamos interesados en detectar conjuntos máximos de recursos sobre los cuales haya que definir y garantizar la relación de orden de asignación y/o liberación. Esta extensión quiere decir que estamos interesados en conjuntos máximos de zonas de recursos cuya intersección sea no nula, lo cual querrá decir que existe un estado en el que un proceso activo usa simultáneamente todos los recursos del conjunto y ninguno más fuera del conjunto (en este sentido es un conjunto máximo). A cada uno de estos conjuntos máximos de recursos lo denominaremos *unidad de recursos* (RU, Resource Unit). Con estas ideas en mente, a continuación se formalizan estos objetos utilizando como base el concepto de *conjunto de zonas solapadas*.

Definición 7. *Sea $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ una red S^4PR y sea $\mathcal{Z}$ el conjunto de zonas asociadas a los recursos de $\mathcal{N}$. Un conjunto de zonas $A \subseteq \mathcal{Z}$ se dice que es un conjunto de zonas solapadas si y solo si $\bigcap_{\mathcal{Z}^r \in A} \mathcal{Z}^r \neq \emptyset$ y $A \neq \emptyset$.*

Obsérvese que en la definición anterior, dado un conjunto de zonas solapadas, la intersección de todas las zonas del conjunto será el conjunto no vacío de lugares proceso en los que un proceso activo (marca contenida en esos lugares) que se encuentra en esos lugares está utilizando simultáneamente todos los recursos asociados a las zonas que pertenecen a ese conjunto.

Además, se verifica que un conjunto de zonas solapadas siempre está definido sobre una misma máquina de estado de la red S^4PR. Esto es así ya que las zonas son conjuntos de lugares proceso pertenecientes a la misma máquina de estado (ver la definición de zona de uso continuado de un recurso) y dado que los lugares de dos máquinas de estado diferentes son conjuntos disjuntos, se sigue que todas las zonas de un conjunto de zonas solapadas (cuya intersección es no nula) deberán estar incluidas en la misma máquina de estado.

Finalmente, y por convención, asumiremos que un conjunto de zonas solapadas formado por una sola zona es válido y además los lugares proceso que pertenecen a la intersección son todos los lugares de la zona del conjunto. En otras palabras, consideraremos que la relación de solapamiento de zonas es reflexiva. La siguiente definición formaliza los aspectos comentados sobre los lugares proceso pertenecientes a la intersección de las zonas de un conjunto de zonas solapadas, así como los recursos implicados.

Definición 8. *Sea $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ una red S^4PR y sea $\mathcal{Z}$ el conjunto de zonas binarias asociadas a los recursos de $\mathcal{N}$. Si $A \subseteq \mathcal{Z}$ es un conjunto de zonas solapadas, denominamos,*

1. *Solapamiento de A, $sol(A)$, al conjunto de lugares proceso pertenecientes a la intersección de todas las zonas de A,*

$$sol(A) \quad = \quad \begin{cases} \bigcap_{\mathcal{Z}^r \in A} \mathcal{Z}^r, & si\ |A| \geq 2 \\ \mathcal{Z}^r, & si\ A = \{\mathcal{Z}^r\} \end{cases}$$

2. *Recursos de A, $rec(A)$, al conjunto de lugares recurso que son utilizados en cada uno de los lugares del solapamiento de A, $sol(A)$,*

$$rec(A) \quad = \quad \{r | r \in P_R \text{ y } \mathcal{Z}^r \in A\}$$

Como ya se estableció previamente, estamos interesados en conjuntos de zonas solapadas máximas, dado que estos conjuntos caracterizan conjuntos de estados en los que se utilizan conjuntos máximos de recursos y por lo tanto se necesita estudiar el orden en el que son adquiridos para comprobar que este orden se respeta en la liberación.

Definición 9. *Sea $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C}\rangle$ una red S^4PR y sea $\mathscr{Z}$ el conjunto de zonas binarias asociadas a los recursos de $\mathcal{N}$. Sea $A \subseteq \mathscr{Z}$ un conjunto de zonas solapadas. Se dice que A es un conjunto máximo de zonas solapadas o unidad de recursos si y solo si no existe otro conjunto de zonas solapadas $B \neq A$, tal que $A \subseteq B$.*

En la sección siguiente se presenta un algoritmo para el cálculo de estos conjuntos máximos.

3.2.5. Cálculo de las unidades de recursos en redes S^4PR

A continuación se presenta un algoritmo para el cálculo de los conjuntos máximos de zonas solapadas. Dicho algoritmo se aplica a redes S^4PR generales, aunque nosotros solo estaremos interesados en su aplicación en redes para las que todas sus zonas son binarias, ya que es este tipo de redes las que resultan del proceso de modelado de los sistemas de encaminamiento. El algoritmo que se presenta, aprovecha la circunstancia de que en un conjunto de zonas solapadas nunca podrán existir dos zonas distintas que pertenezcan cada una a una máquina de estados diferente de la red S^4PR. Por esta razón, en el algoritmo, inicialmente se clasifican las zonas según la máquina a la que pertenecen y a partir de ahí se hace el cálculo independientemente para cada conjunto de zonas de una máquina de estado.

A continuación aplicamos el Algoritmo 5 al cálculo de los conjuntos máximos de zonas solapadas en la red de la figura 3.5. Después de aplicar el Algoritmo 4 a la red S^4PR de la figura 3.5 se obtiene el siguiente conjunto de zonas:

$$\begin{array}{lll} Z^r_{1,1} = \{p_1, p_2\}; & Z^r_{1,2} = \{p_6, p_7\}; & Z^r_{2,1} = \{q_3, q_4, q_5\}; \\ Z^s_{1,1} = \{p_2, p_3, p_4\}; & Z^s_{2,1} = \{q_2, q_3, q_4\}; & \\ Z^t_{1,1} = \{p_4, p_5, p_6\}; & Z^t_{2,1} = \{q_1, q_2, q_3\}; & \end{array}$$

La aplicación del algoritmo hace que al principio de la primera iteración del bucle de la línea 9 del algoritmo tengamos que,

$$\mathscr{Z}^{w_1} = \{\{Z^r_{1,1}\}, \{Z^r_{1,2}\}, \{Z^r_{2,1}\}, \{Z^s_{1,1}\}, \{Z^s_{2,1}\}, \{Z^t_{1,1}\}, \{Z^t_{2,1}\}\}$$

El algoritmo procede iterando hasta que el conjunto $\mathscr{Z}^{w_1}$ es igual al conjunto vacío. En ese instante se tiene al final del bucle en la línea 25 del algoritmo,

$$\begin{array}{rcl} \mathscr{Z}^{w_1} & = & \emptyset; \\ \mathscr{Z}^{auxt} & = & \{\{Z^r_{1,1}; Z^s_{1,1}\}, \{Z^r_{1,2}; Z^t_{1,1}\}, \{Z^r_{2,1}; Z^s_{2,1}\}, \{Z^s_{1,1}; Z^t_{1,1}\}, \{Z^s_{2,1}; Z^t_{2,1}\}\} \\ \mathscr{Z}^{\text{máx}} & = & \{\{Z^t_{1,1}\}\} \end{array}$$

Algoritmo 5 Cálculo de los conjuntos máximos de zonas solapadas en redes S^4PR.

Entrada:

1. La red S^4PR $\mathscr{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$;
2. El conjunto de zonas de $\mathscr{N}$:
3. $\mathscr{Z} = \{Z^r_{i,j} |$ es la j-ésima zona de $r \in P_R$, en la i-ésima máquina de estado de $\mathscr{N}\}$;

Salida:

4. Los conjuntos máximos de zonas solapadas de la red $\mathscr{N}$:
5. $\mathscr{Z}^{\text{máx}} = \{C_Z | C_Z \in 2^{\mathscr{Z}}$ y es un conjunto máximo de zonas solapadas de $\mathscr{N}\}$;
6. **Inicio**
7. $\mathscr{Z}^{\text{máx}} = \emptyset$; $\mathscr{Z}^{auxt} = \emptyset$;
8. Añadir a $\mathscr{Z}^{w_1} \subseteq 2^{\mathscr{Z}}$ un conjunto por cada zona que pertenezca a $\mathscr{Z}$ compuesto por la propia zona;
9. **Mientras que** $\mathscr{Z}^{w_1} \neq \emptyset$ **Hacer**
10. Seleccionar un elemento de $\mathscr{Z}^{w_1}$ y almacenarlo en la variable $C_1 \subseteq \mathscr{Z}$ que será un conjunto de zonas solapadas;
11. $\mathscr{Z}^{w_1} = \mathscr{Z}^{w_1} \setminus \{C_1\}$;
12. $\mathscr{Z}^{w_2} = \mathscr{Z}^{w_1}$; $\mathscr{Z}^{auxp} = \emptyset$;
13. **Mientras que** $\mathscr{Z}^{w_2} \neq \emptyset$ **Hacer**
14. Seleccionar un elemento de $\mathscr{Z}^{w_2}$ y almacenarlo en la variable $C_2 \subseteq \mathscr{Z}$ que será un conjunto de zonas solapadas;
15. $\mathscr{Z}^{w_2} = \mathscr{Z}^{w_2} \setminus \{C_2\}$;
16. **Si** $sol(C_1) \cap sol(C_2) \neq \emptyset$ **Entonces**
17. $\mathscr{Z}^{auxp} = \mathscr{Z}^{auxp} \cup \{C_1 \cup C_2\}$;
18. **Fin Si**
19. **Fin Mientras que**
20. **Si** $\mathscr{Z}^{auxp} = \emptyset$ **Entonces**
21. $\mathscr{Z}^{\text{máx}} = \mathscr{Z}^{\text{máx}} \cup \{C_1\}$
22. **Sino**
23. $\mathscr{Z}^{auxt} = \mathscr{Z}^{auxt} \cup \mathscr{Z}^{auxp}$;
24. **Fin Si**
25. **Si** $\mathscr{Z}^{w_1} = \emptyset$ **Entonces**
26. $\mathscr{Z}^{w_1} = \mathscr{Z}^{auxt}$; $\mathscr{Z}^{auxt} = \emptyset$;
27. **Fin Si**
28. **Fin Mientras que**
29. Eliminar todos los conjuntos de zonas solapadas $C \in \mathscr{Z}^{\text{máx}}$ tales que existe $C' \in \mathscr{Z}^{\text{máx}}$ y $C \subseteq C'$.
30. **Fin**

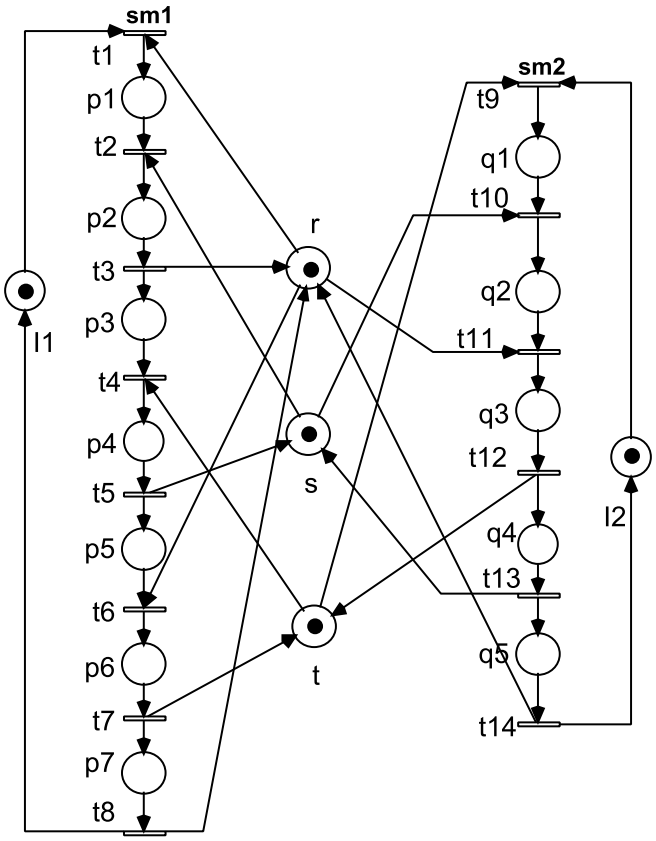

Figura 3.5: Red *SOAR*^{2} para ilustrar el cálculo de los conjuntos máximos de zonas solapadas.

Al inicio de la siguiente iteración del bucle que empieza en la línea 9, $\mathscr{Z}^{w_1}$ es igual al conjunto $\mathscr{Z}^{auxt}$ anterior. Una vez más, el algoritmo itera hasta que el conjunto $\mathscr{Z}^{w_1}$ es igual al conjunto vacío y ahora los valores que se alcanzan son,

$$\begin{aligned}
\mathscr{Z}^{w_1} &= \emptyset; \\
\mathscr{Z}^{auxt} &= \{\{Z^r_{2,1}; Z^s_{2,1}; Z^t_{2,1}\}\} \\
\mathscr{Z}^{\text{máx}} &= \{\{Z^t_{1,1}\}, \{Z^r_{1,1}; Z^s_{1,1}\}, \{Z^r_{1,2}; Z^t_{1,1}\}, \{Z^s_{1,1}; Z^t_{1,1}\}, \{Z^s_{2,1}; Z^t_{2,1}\}\}
\end{aligned}$$

La siguiente iteración del bucle que comienza en la línea 9 del algoritmo, hace que el algoritmo termine dado que $\mathscr{Z}^{w_1}$ tiene un único elemento y se obtiene,

$$\mathscr{Z}^{\text{máx}} = \{\{Z^t_{1,1}\},\{Z^r_{1,1};Z^s_{1,1}\},\{Z^r_{1,2};Z^t_{1,1}\},\{Z^s_{1,1};Z^t_{1,1}\},\{Z^s_{2,1};Z^t_{2,1}\},\{Z^r_{2,1};Z^s_{2,1};Z^t_{2,1}\}\}$$

Sobre este último conjunto es sobre el que se aplica el paso de la línea 29 del algoritmo que elimina los conjuntos no máximos de zonas solapadas, dándonos el resultado final buscado compuesto por 4 conjuntos máximos,

$$\mathscr{Z}^{\text{máx}} = \{\{Z^r_{1,1};Z^s_{1,1}\},\{Z^r_{1,2};Z^t_{1,1}\},\{Z^s_{1,1};Z^t_{1,1}\},\{Z^r_{2,1};Z^s_{2,1};Z^t_{2,1}\}\}$$

En este punto ya hemos respondido al tema A planteado anteriormente sobre cuáles son los conjuntos de recursos sobre los que hay que definir la relación de orden que debe ser mantenida de su asignación a su liberación. Merece en este punto detallar la respuesta,

a) Los conjuntos de recursos sobre los que hay que definir la relación de orden son múltiples y se han caracterizado mediante los conjuntos máximos de zonas solapadas. Dado uno de estos conjuntos, C, el conjunto de recursos sobre el que hay que verificar la relación de orden es $rec(C)$.

b) Un mismo recurso puede estar implicado en múltiples conjuntos de recursos a ordenar en su asignación/liberación. En el ejemplo desarrollado anteriormente se puede observar que el recurso r está implicado en dos conjuntos diferentes en la máquina de estados de la izquierda (numerada como 1) y en un conjunto definido por los solapamientos de las zonas de la máquina de estados de la derecha (numerada como máquina 2). En cada uno de estos conjuntos de recursos, el recurso r puede ser ordenado de formas diferentes con respecto a los otros recursos contenidos en los conjuntos.

c) El método de cálculo de todos los conjuntos de recursos en los que hay que estudiar la relación de orden en la asignación/liberación está basado en el cálculo de los conjuntos máximos de zonas solapadas planteado en el Algoritmo 5.

El siguiente punto a desarrollar es la caracterización de la relación de orden en la asignación y en la liberación de los recursos implicados en cada conjunto máximo de zonas solapadas. Obsérvese que nuestro problema no es tanto conocer la relación de orden concreta definida entre los recursos implicados en un conjunto máximo de zonas solapadas, como conocer si la relación de orden aplicada en la asignación es la misma que se aplica en la liberación de los recursos.

3.2.6. Relación de orden sobre los recursos de zonas solapadas

La existencia de determinadas transiciones y lugares en las zonas, nos permiten establecer fronteras de asignación y liberación de los recursos que se presentarán formalmente a través de diversas propiedades en esta subsección. Se debe tener en cuenta que una zona $Z^r_{i,j}$ de un recurso r en la i-ésima máquina de estados es un conjunto máximo de lugares usuarios de r, verificándose que $Z^r_{i,j} \cup ((Z^r_{i,j})^\bullet \cap {}^\bullet(Z^r_{i,j}))$ es una máquina de estados conexa. Por lo tanto, de esta definición se pueden extraer las siguientes conclusiones:

a) En la máquina de estados conexa definida a partir de $Z^r_{i,j} \cup ((Z^r_{i,j})^\bullet \cap {}^\bullet(Z^r_{i,j}))$ existe un conjunto de lugares

$$F \;=\; Z^r_{i,j} \setminus ((Z^r_{i,j})^\bullet \cap {}^\bullet(Z^r_{i,j}))^\bullet$$

que verifican que: (1) Son lugares fuente en la máquina de estados conexa definida por la zona, es decir, no tienen transiciones de entrada dentro de la máquina de estados; (2) Las transiciones de entrada a estos lugares, ${}^\bullet F$, son transiciones de salida del recurso r, dado que los lugares de la zona son usuarios de r. Además, como estamos considerando zonas binarias, estas transiciones son las únicas que son de salida del recurso ya que ninguna transición $t \in ((Z^r_{i,j})^\bullet \cap {}^\bullet(Z^r_{i,j}))$ puede pertenecer a $r^\bullet$. En otras palabras, y este es el aspecto que nos interesa aquí, ${}^\bullet F$ es el conjunto de transiciones que realizan la operación de asignación del recurso r en la zona $Z^r_{i,j}$.

b) De forma análoga al punto anterior se puede definir el conjunto de lugares,

$$E \;=\; Z^r_{i,j} \setminus {}^\bullet((Z^r_{i,j})^\bullet \cap {}^\bullet(Z^r_{i,j}))$$

que verifican que: (1) Son lugares sumidero en la máquina de estados conexa definida por la zona, es decir, no tienen transiciones de salida dentro de la máquina de estados asociada a la zona; (2) Las transiciones de salida de estos lugares, $E^\bullet$, son transiciones de entrada al recurso r, dado que una vez más, los lugares de zona son usuarios del recurso r. Además, como estamos considerando zonas binarias, estas transiciones son las únicas que son de entrada al recurso r ya que ninguna transición $t \in ((Z^r_{i,j})^\bullet \cap {}^\bullet(Z^r_{i,j}))$ puede pertenecer a ${}^\bullet r$. En otras palabras, $E^\bullet$ es el conjunto de las transiciones que realizan la operación de liberación del recurso r cuando un proceso activo abandona la zona $Z^r_{i,j}$.

De acuerdo con la discusión anterior, formalizamos los conjuntos de lugares fuente y sumidero anteriores para una zona dada a través de los conceptos de *Frontera de asignación* y *Frontera de liberación*.

Definición 10. *Sea $\mathscr{N}$ una Red de Petri de la clase S^4PR, $\mathscr{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ y sea $\mathscr{Z}^r$ una zona binaria asociada al recurso $r \in P_R$. Se denomina:*

1. *Frontera de asignación de r a un proceso que entra en la zona $\mathscr{Z}^r$, $Fa(\mathscr{Z}^r) \subseteq \mathscr{Z}^r$, al conjunto de lugares*

$$Fa(\mathscr{Z}^r) \quad = \quad Z^r \setminus ((Z^r)^\bullet \cap {}^\bullet(Z^r))^\bullet$$

2. *Frontera de liberación de r por un proceso que sale de la zona $\mathscr{Z}^r$, $Fl(\mathscr{Z}^r) \subseteq \mathscr{Z}^r$, al conjunto de lugares*

$$Fl(\mathscr{Z}^r) \quad = \quad Z^r \setminus {}^\bullet((Z^r)^\bullet \cap {}^\bullet(Z^r))$$

El siguiente resultado pone en evidencia la importancia de las zonas de uso continuado de recursos a la hora de determinar que el orden de asignación de recursos en un conjunto de zonas solapadas es el mismo orden que se usa en la liberación de los mismos.

Lema 5. *Sea $\mathscr{N}$ una red de Petri de la clase S^4PR, $\mathscr{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ y sea Z^r una zona binaria asociada al recurso $r \in P_R$. Para todo camino $t_0 p_1 t_1 p_2 t_2 .....p_k$ perteneciente a la máquina de estados de la zona Z^r tal que: (1) no existen nodos repetidos; (2) $t_0 \in {}^\bullet Fa(Z^r)$; y (3) p_k es el lugar reposo de la máquina de estados del camino; se cumple que existe un prefijo único de este camino: $t_0 p_1 t_1 p_2 t_2 ... p_n t_n$ tal que: (1) $t_n \in Fl(Z^r)^\bullet$; (2) $p_i \in Z^r$ para todo $i = 1...n$; y (3) $t_j \in (Z^r)^\bullet \cap {}^\bullet(Z^r)$ para todo $j = 1...(n-1)$.* □

Demostración. Se sigue directamente del hecho de que Z^r es el conjunto máximo de holders tales que Z^r y $(Z^r)^\bullet \cap {}^\bullet(Z^r)$ forman una máquina de estados conexa, y todos los lugares de Z^r pertenecen al p–semiflujo mínimo binario del recurso r. □

El lema anterior especifica que si un mensaje entra en la zona Z^r por una transición t de entrada a la frontera de asignación de Z^r, $t \in {}^\bullet Fa(Z^r)$, el mensaje inevitablemente sigue un camino por el interior de la zona Z^r hasta que la abandona al disparar una transición t' de salida de la frontera de liberación de Z^r, $t' \in Fl(Z^r)^\bullet$. En otras palabras, un mensaje al que se va a asignar un recurso r en la zona Z^r, inicia su movimiento desde la frontera de asignación por el interior de la zona Z^r hasta que la abandona por la frontera de liberación: de allí el nombre de zona de uso continuado de un recurso r.

A partir de la frontera de asignación y la frontera de liberación definidas anteriormente y teniendo en cuenta la propiedad expresada en el lema anterior 5, si tenemos dos zonas solapadas referidas a dos recursos diferentes, podemos estudiar con estos conceptos de fronteras el orden de asignación de los dos recursos implicados para determinar si coincide con el orden de liberación de estos recursos. Obsérvese que las dos zonas bajo

consideración deben corresponder a dos recursos diferentes ya que dos zonas de un mismo recurso, aunque estén definidas sobre la misma máquina de estados de la red S^4PR, siempre son disjuntas y por tanto nunca podrán solaparse.

A continuación se define una relación entre dos zonas solapadas de recursos.

Definición 11. *Sea $\mathcal{N}$ una red de Petri de la clase S^4PR, $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ y sean Z^r y Z^s dos zonas binarias solapadas de $\mathcal{N}$ asociadas a los recursos $r,s \in P_R$ y $r \neq s$. Diremos que la zona Z^r precede a la zona Z^s, denotado como $Z^r < Z^s$, si y solo si*

a) $Fa(Z^r) \subseteq Z^r \setminus Z^s$

b) $Fl(Z^s) \subseteq Z^s \setminus Z^r$

c) $Fl(Z^r) \subseteq Z^s$

□

El resultado siguiente pone de manifiesto que la relación de precedencia anterior es irreflexiva y asimétrica, pero en general no es transitiva lo cual no es algo deseable en el contexto en el que estamos ya que pretendemos una relación de orden total estricta entre las zonas solapadas de un conjunto máximo.

Lema 6. *La relación de precedencia, $<$, definida entre zonas binarias solapadas de una red $\mathcal{N}$ de la clase S^4PR es irreflexiva y asimétrica.* □

Demostración. Para demostrar que es irreflexiva debemos verificar que dada una zona binaria Z^r, es falso que $Z^r < Z^r$. En efecto, de acuerdo con la definición de uso continuado de recurso r, se cumple que $Fa(Z^r) \neq \emptyset$. Por lo tanto, la condición $a)$ de la definición 11 no se cumple ya que $Z^r \setminus Z^r = \emptyset$. Para demostrar que la relación $<$ es asimétrica debemos probar que si Z^r y Z^s son dos zonas solapadas tales que $Z^r < Z^s$, entonces no se cumple que $Z^s < Z^r$. En efecto, si $Z^r < Z^s$, de acuerdo con la condición $b)$ de la definición 11, $Fl(Z^s) \subseteq Z^s \setminus Z^r$ luego nunca podría cumplirse que $Fl(Z^s) \subseteq Z^r$ que sería la condición $c)$ si se cumpliese que $Z^s < Z^r$. □

En general la relación de precedencia entre zonas solapadas no es una relación transitiva como el ejemplo de la figura 3.6 pone de manifiesto. En efecto se puede verificar fácilmente que $Z^r < Z^s$ y $Z^s < Z^t$ pero no se cumple que $Z^r < Z^t$ ya que $Fl(Z^r) \not\subseteq Z^t$ ya que $Fl(Z^r) = \{p_2, p_3\}$ y $Z^t = \{p_3, p_5, p_6, p_7, p_8\}$, *(es la condición c) de la definición 11).*

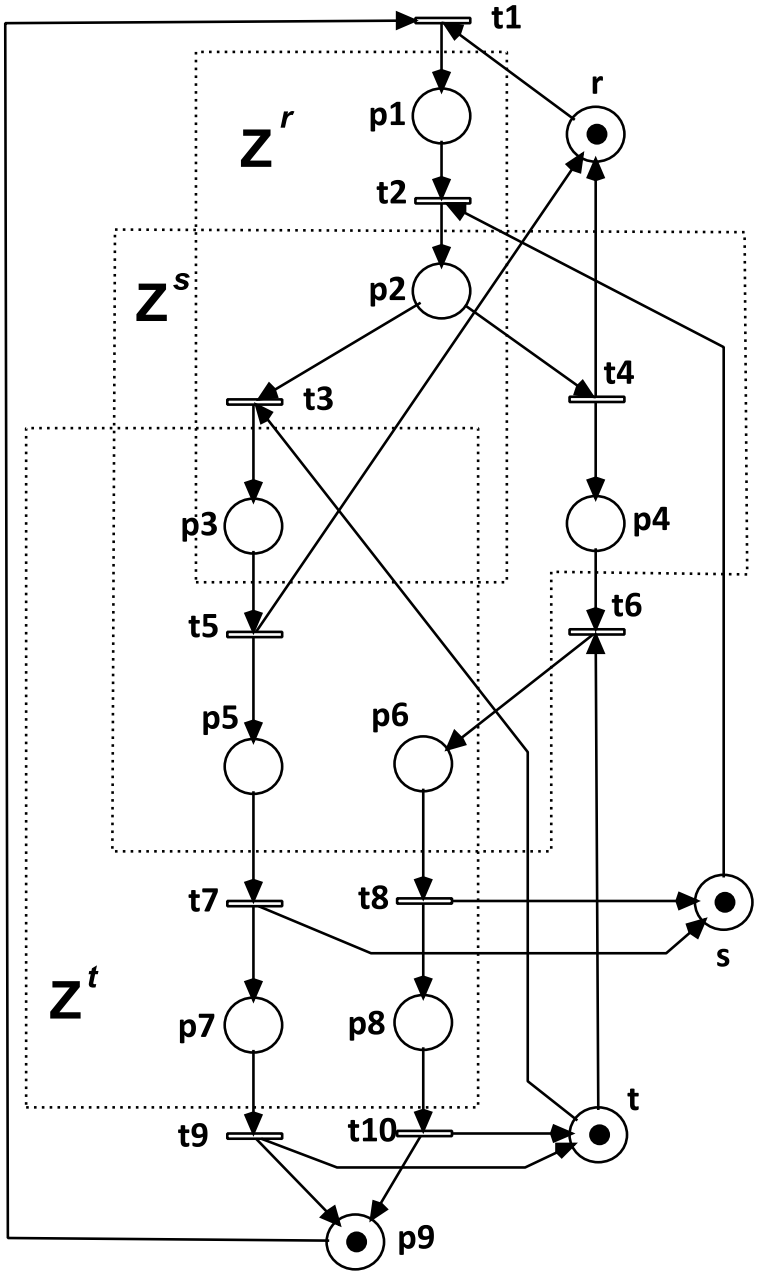

Figura 3.6: Red S^4PR que muestra que la relación de precedencia no es transitiva: $Z^r < Z^s$ y $Z^s < Z^t$ pero no se cumple $Z^r < Z^t$.

De la definición 11 se deduce que dos zonas solapadas Z^r y Z^s no precede una a la otra si por ejemplo:

a) $Fa(Z^r) \cap Fa(Z^s) \neq \emptyset$ ya que en este caso existe al menos un $p \in Fa(Z^r) \cap Fa(Z^s)$ y por la definición de frontera de asignación existe una transición $t \in {}^\bullet p$ tal que $r \in {}^\bullet t$ y $s \in {}^\bullet t$. Es decir, los dos recursos se toman simultáneamente al disparar t y por tanto no se puede ordenar la asignación de estos recursos porque se asignan concurrentemente.

b) $Fl(Z^r) \cap Fl(Z^s) \neq \emptyset$, que resulta ser el mismo caso que el anterior pero referido a las fronteras de liberación de los recursos.

c) Las fronteras de asignación de las zonas Z^r y Z^s se cruzan, es decir, $Fa(Z^r) \cap (Z^r \setminus Z^s) \neq \emptyset$ y $Fa(Z^s) \cap (Z^s \setminus Z^r) \neq \emptyset$ pero $Fa(Z^r) \not\subseteq Z^r \setminus Z^s$ y $Fa(Z^s) \not\subseteq Z^s \setminus Z^r$. Este es el caso de la red de Petri presentada en la figura 3.7. Efectivamente, $Fa(Z^r) = \{p_2, p_5\}$ y $Fa(Z^s) = \{p_3, p_4\}$. Por lo tanto, $Fa(Z^r) \cap (Z^r \setminus Z^s) = \{p_2, p_5\} \cap \{p_2\} = 0$ y $Fa(Z^s) \cap (Z^s) \cap (Z^s \setminus Z^r) = \{p_3, p_4\} \cap \{p_3, p_7\} = \{p_3\} \neq 0$; pero como se puede apreciar $Fa(Z^r) \not\subseteq Z^r \setminus Z^s$ por causa de $p_5 \in Fa(Z^r)$ y $p_5 \notin Z^r \setminus Z^s$; y $Fa(Z^s) \not\subseteq Z^s \setminus Z^r$ por causa de $p_4 \in Fa(Z^s)$ y $p_4 \notin Z^s \setminus Z^r$. En este caso, lo que se observa es que los recursos r y s deben estar ordenados de la misma manera en la asignación y la liberación ya que son recursos por los que Z^r y Z^s están solapados, es decir, existen los lugares p_4, p_5 y p_6 en los que un mensaje utiliza simultáneamente ambos recursos. Por lo tanto cuando el mensaje alcance un lugar como el p_6 lo habrá hecho de forma que r y s hayan sido asignados en un orden determinado para que en la liberación este orden sea el mismo. Obsérvese, que un mensaje que alcance el lugar p_6 ha podido venir por la rama t_1, t_3 y t_5, en cuyo caso el orden de asignación es r seguido de s, o ha podido venir por la rama t_2, t_4 y t_6, en cuyo caso el orden de asignación es s seguido de r. Por lo tanto, en p_6 el lugar/mensaje ha obtenido los mensajes en cualquiera de los órdenes posibles y no se ha memorizado cual es el orden que realmente se ha utilizado. Si además se observa que la liberación se realiza en un orden: primero r y después s; concluimos que en el camino $t_2, p_3, t_4, p_5, t_6, p_6, t_7, p_7, t_8$ la asignación de los recursos r y s se realiza en un orden distinto al que se usa en la liberación.

d) Las fronteras de liberación de las zonas Z^r y Z^s se cruzan. Este caso es parecido al anterior pero referido en este caso a las fronteras de liberación de las zonas Z^r y Z^s.

La relación de precedencia establecida garantiza que en todo camino conducente a lugares que usan simultáneamente los dos recursos en las zonas solapadas Z^r y Z^s de los dos recursos, el orden de asignación de esos dos recursos es el mismo que el de su liberación. No obstante, esto no prejuzga relaciones de inclusión especiales entre partes de las zonas ordenadas por la relación de precedencia.

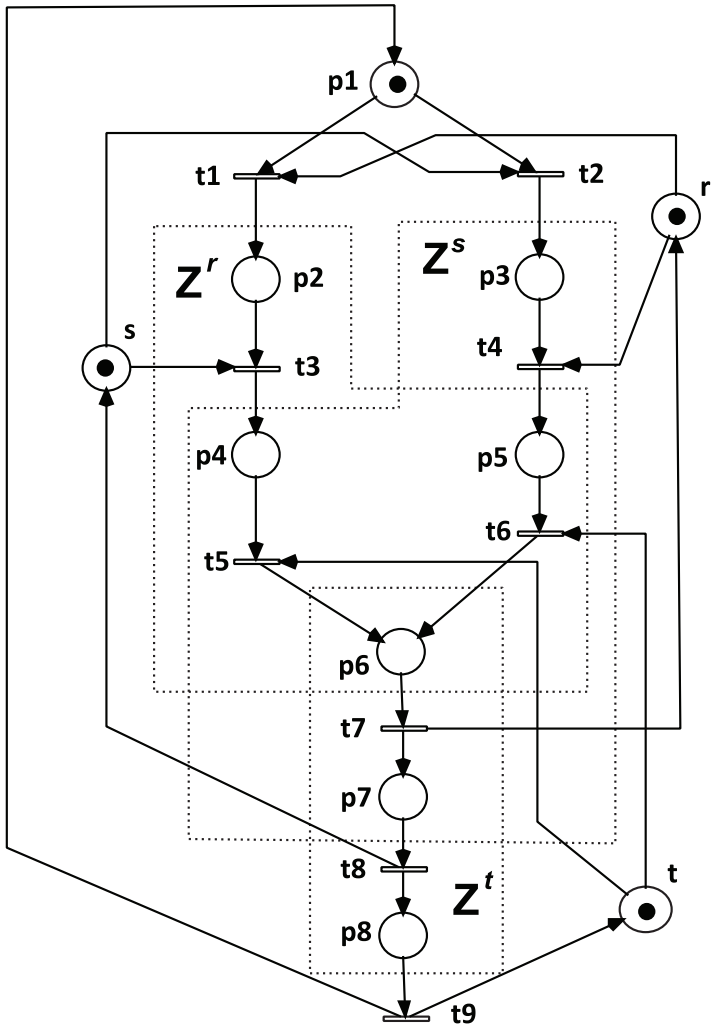

Figura 3.7: Red S^4PR en la que las fronteras de asignación de Z^r y Z^s se cruzan.

Efectivamente, en la red de la figura 3.5 se observa que $Z^r_{1,1} < Z^s_{1,1}$. Además, se verifica que $Z^r_{1,1} \setminus (Z^r_{1,1} \setminus Z^s_{1,1}) \subset Z^s_{1,1}$. Es decir, que una vez atravezada las fronteras de $Z^r_{1,1}$ y $Z^s_{1,1}$ el resto de la zona $Z^r_{1,1}$ está completamente contenida en la zona $Z^s_{1,1}$ con lo que se garantiza *(condición suficiente)* que el camino desde la frontera de r hasta el lugar reposo asignará primero r y luego s y la liberación será en el mismo orden.

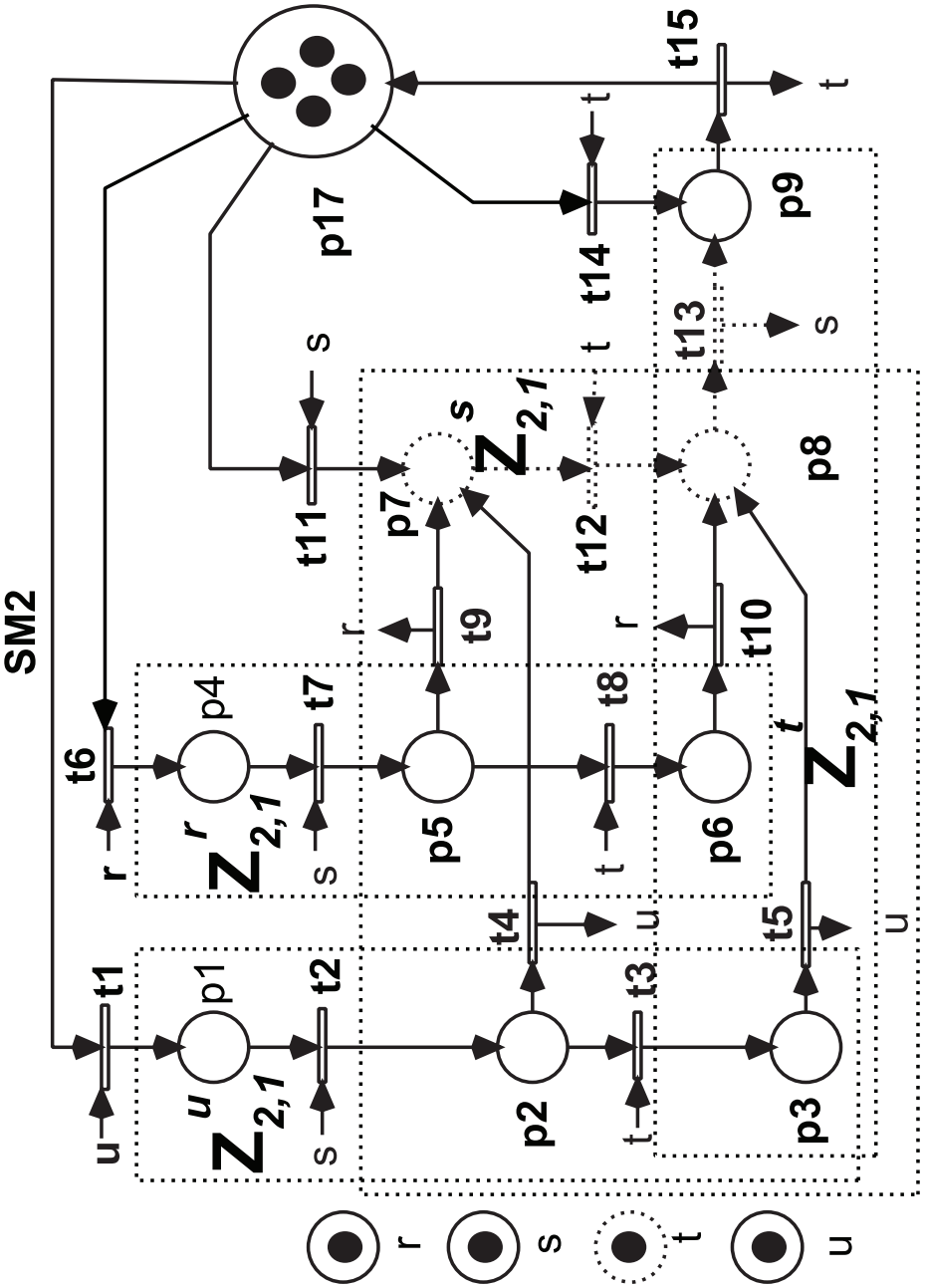

Figura 3.8: Red de Petri S^4PR mostrando distintos grados de solape entre zonas en relación de precedencia.

No obstante, otras relaciones de inclusión son posibles entre partes de las zonas implicadas respetando que el orden de asignación sea el mismo que el orden de liberación.

Así por ejemplo, en la red de Petri de la figura 3.8, se puede observar que $Z^r_{2,1} < Z^s_{2,1}$ ya que se trata de dos zonas solapadas $(Z^r_{2,1} \cap Z^2_{2,1}) = \{p_5, p_6\}$ y además se cumple que:

1. $Fa(Z^r_{2,1}) \subseteq Z^r_{2,1} \setminus Z^s_{2,1}$; ya que $Fa(Z^r_{2,1}) = \{p_4\}$ y $Z^r_{2,1} \setminus Z^s_{2,1} = \{p_4\}$.
2. $Fl(Z^s_{2,1}) \subseteq Z^s_{2,1} \setminus Z^r_{2,1}$; ya que $Fl(Z^s_{2,1}) = \{p_8\}$ y $Z^s_{2,1} \setminus Z^r_{2,1} = \{p_2, p_3, p_7, p_8\}$.

Sin embargo la forma en la que se solapan estas zonas es diferente a la forma en la que lo hacían las zonas de la figura 3.5 comentadas anteriormente.

Efectivamente, en este caso tenemos que como se esperaba todo camino que empieza asignando r al entrar en la zona $Z^r_{2,1}$ a continuación procederá a asignar el recurso s al entrar inevitablemente en la zona $Z^s_{2,1}$ y la liberación se realizará en el mismo orden. No obstante, existen caminos de entrada a la zona $Z^s_{2,1}$ que no vienen precedidas por una entrada a la zona $Z^r_{2,1}$. Ahora bien, estos caminos nunca podrán atravesar la zona $Z^r_{2,1}$ porque en ese caso no se cumpliría que $Z^r_{2,1}$ esté incluido en el soporte del p–semiflujo mínimo de r.

Para terminar esta sección, se presenta el resultado que señala que si una zona precede a otra, entonces todo camino desde la entrada de la zona precedente hasta la salida de ambas zonas presenta el mismo orden de asignación de los recursos que el de liberación de los mismos.

Lema 7. *Sea $\mathcal{N}$ una red de Petri de la clase S^4PR, $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C}\rangle$ y sean Z^r y Z^s dos zonas solapadas binarias asociadas a los recursos $r,s \in P_R$, $r \neq s$, y tales que Z^r precede a la zona Z^s, es decir $Z^r < Z^s$. Para todo camino $t_0p_1t_1p_2t_2...p_k$ perteneciente a la máquina de estados de las zonas solapadas Z^s y Z^r, tal que: (1) no existen nodos repetidos; (2) $t_0 \in {}^\bullet Fa(Z^r)$; (3) p_k es el lugar reposo de la máquina de estados del camino; se cumple que existe un prefijo único de este camino, $t_0p_1t_1p_2t_2...p_nt_n$ que se puede descomponer en tres partes $uxz = t_0p_1t_1p_2t_2...p_nt_n$ que cumplen las siguientes condiciones:*

1. *Tramo $u = t_0p_1t_1p_2t_2...p_at_a$, tal que: (1) $t_0 \in {}^\bullet Fa(Z^r)$; (2) $t_a \in {}^\bullet Fa(Z^s)$ con $a \geq 1$; (3) $P_i \in Z^r \setminus Z^s$ para todo $i = 1..a$; (4) $t_j \in (Z^r)^\bullet \cap {}^\bullet(Z^r)$, $j = 1..a$.*
2. *Tramo $x = P_{a+1}t_{a+1}...P_bt_b$: tal que (1) $P_{a+1} \in Fa(Z^s) \cap Z^r$; (2) $P_i \in Z^r \cap Z^s$ para todo $i = a+1...b$, con $b \geq a+1$; (3) $t_j \in ((Z^r)^\bullet \cap {}^\bullet(Z^r)) \cap ((Z^s)^\bullet \cap {}^\bullet(Z^s))$ para todo $j = a+1...b-1$; (4) $t_b \in Fl(Z^r)^\bullet[$ y además $t_b \in ((Z^s)^\bullet \cap {}^\bullet(Z^s)) \setminus ((Z^r)^\bullet \cap {}^\bullet(Z^r)]$.*
3. *Tramo $z = P_{b+1}t_{b+1}...P_ct_c$, tal que: (1)$P_{b+1} \in Z^s \setminus Z^r$ y $P_{b+1} \in Fl(Z^r)^{\bullet\bullet} \cap Z^s$; (2) $P_c \in Fl(Z^s)$ con $c \geq b+1$; (3) $t_c \in Fl(Z^s)^\bullet$; (4)$P_i \in Z^s \setminus Z^r$ para todo $i = b+1...c$; (5) $t_j \in (Z^s)^\bullet \cap {}^\bullet(Z^s)$ para todo $j = b+1...c-1$.*

□

Demostración. Un camino $t_0 p_1 t_1 p_2 t_2 ... p_k$ en el que $t_0 \in {}^\bullet Fa(Z^r)$ y p_k es el lugar reposo y no se repiten nodos, de acuerdo con el lema 5 tiene un prefijo único: $\Pi : t_0 p_1 t_1 p_2 t_2 ... P_b t_b$ tal que (1) $t_b \in Fl(Z^r)^\bullet$; (2) $P_i \in Z^r$ para todo $i = 1...b$; y (3) $t_j \in (Z^r)^\bullet \cap {}^\bullet(Z^r)$ para todo $j = 1...b-1$. Dado que $t_0 \in {}^\bullet Fa(Z^r)$ se cumple también que $p_1 \in Fa(Z^r)$. Adicionalmente, ya que $Z^r < Z^s$, por la definición 11 se cumple que $Fa(Z^r) \subseteq Z^r \setminus Z^s$ luego por lo tanto, en Π encontramos un prefijo $u = t_0 p_1 t_1 p_2 t_2 ... P_a t_a$ que verifica que todas las condiciones establecidas en el enunciado para el tramo u. Obsérvese que el subcamino de Π resultante de la eliminación del prefijo u, es un camino $x = P_{a+1} t_{a+1} ... P_b t_b$ que con arreglo al lema 5 podemos completar con un sufijo $z = P_{b+1} t_{b+1} ... P_c t_c$ para terminar de atravesar la zona Z^s y que cumplirá todas las condiciones del enunciado del lema dado que simplemente corresponden a una transcripción de las condiciones del lema 5. □

Por lo tanto, con la relación de precedencia definida sobre zonas solapadas de recursos impuesta de forma normativa a los conjuntos máximos de zonas solapadas podemos garantizar que el orden de asignación de recursos que llegan a usarse simultáneamente es el mismo orden con el que son liberados. Por lo tanto, en la sección siguiente procederemos a definir la clase de redes *SOAR*^{2} que cumplen esta propiedad usando estas herramientas.

3.2.7. Definición de la clase de redes *SOAR*^{2}

Una vez introducidos los conceptos de zonas, zonas solapadas y conjuntos máximos de zonas solapadas, estamos en condiciones de culminar esta sección con la definición de la clase *SOAR*^{2} de acuerdo con los requisitos establecidos en la subsección 3.2.1 indicando las cuatro condiciones allí establecidas. La definición como ya se estableció es una restricción de la clase de redes S^4PR y las restricciones se establecen a nivel estructural en tres puntos fundamentales.

1. Las transiciones solo están conectadas con un recurso y por lo tanto se particionan en dos conjuntos: transiciones de asignación de recurso y transiciones de liberación de recurso.

2. Los p–semiflujos mínimos que contienen un recurso tienen pesos que pertenecen al conjunto $\{0, 1\}$, es decir, son p–semiflujos binarios.

3. Se impone un orden total sobre las zonas que pertenecen a los conjuntos máximos de zonas solapadas a través de la relación de precedencia definida anteriormente.

A continuación se formaliza la definición anunciada.

Definición 12. [La clase de redes *SOAR*^{2}]. *Sea $I_{\mathcal{N}} = \{1,2,...,m\}$ un conjunto finito de índices. Una Red de Petri de la clase SOAR^2, es una red lugar/transición ordinaria, pura y conexa $\mathcal{N} = \langle P,T,\mathbf{C}\rangle$, tal que:*

1. *$P = P_0 \cup P_S \cup P_R$ es una partición tal que:*
   a) *$P_S = \bigcup_{i\in I_{\mathcal{N}}} P_{S_i}$, $P_{S_i} \neq \emptyset$, $P_{S_i} \cap P_{S_j} = \emptyset$, $\forall i \neq j$.*
   b) *$P_0 = \bigcup_{i\in I_{\mathcal{N}}} \{p_{0i}\}$.*
   c) *$P_R = \{r_1,\ldots,r_n\}$, $n > 0$.*
2. *$T = T_a \cup T_r$ es una partición tal que:*
   a) *$T_a = \bigcup_{i\in I_{\mathcal{N}}} T_{ai}, T_{ai} \neq \emptyset$, $T_a = P_R{}^{\bullet}$, para todo $i,j \in I_{\mathcal{N}}, i \neq j$, $T_{ai} \cap T_{aj} = \emptyset$; y $\forall t \in T_a, |\ {}^{\bullet}t \cap P_R\ | = 1$, $|\ t^{\bullet} \cap P_R\ | = 0$.*
   b) *$T_r = \bigcup_{i\in I_{\mathcal{N}}} T_{ri}, T_{ri} \neq \emptyset$, $T_r = {}^{\bullet}P_R$, para todo $i,j \in I_{\mathcal{N}}, i \neq j$, $T_{ri} \cap T_{rj} = \emptyset$, y $\forall t \in T_r, |\ t^{\bullet} \cap P_R\ | = 1$, $|\ {}^{\bullet}t \cap P_R\ | = 0$.*
3. *Para todo $i \in I_{\mathcal{N}}$, la subred $\mathcal{N}_i$ generada por $P_{S_i} \cup \{p_{0_i}\} \cup T_{ai} \cup T_{ri}$ es una máquina de estados fuertemente conexa, en la que cada ciclo contiene a p_{0i}.*
4. *Para cada $r \in P_R$ existe un único p–semiflujo mínimo, $\mathbf{y}_r \in \{0,1\}^{|P|}$, tal que $\{r\} = \|\mathbf{y}_r\| \cap P_R$, $P_0 \cap \|\mathbf{y}_r\| = \emptyset$, y $P_S \cap \|\mathbf{y}_r\| \neq \emptyset$.*
5. *$P_S = \bigcup_{r\in P_R} (\|\mathbf{y}_r\| \setminus \{r\})$.*
6. *Para cada conjunto máximo de zonas solapadas la relación de precedencia, $<$, es una relación de orden total estricta.*

En la definición anterior a los lugares pertenecientes al conjunto P_S se les denomina *lugares proceso* y modelan los cambios de estado durante el flujo del objeto, desde un origen a un destino. Los lugares p_{0i} del conjunto P_0 se denominan *lugar de reposo* de la máquina de estados i–ésima y representan los estados inactivos de los procesos. Para representar los recursos en el modelo usaremos los *lugares recurso* pertenecientes al conjunto P_R. Obviamente como en las redes *SOAR*^{2} solo existen p–semiflujos mínimos de los recursos que son binarios, las zonas de recursos también son binarias conforme a la definición realizada anteriormente.

A partir de la definición 12, resulta trivial verificar que toda la red *SOAR*^{2} es una red S^4PR, aunque una red S^4PR en general no es una red *SOAR*^{2}. Este resultado se formaliza en el lema 8.

Lema 8. *Toda Red de Petri $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ de la clase SOAR^2 es una red de la clase S^{4}PR, es decir, SOAR$^2 \subset$ S^{4}PR.*

A continuación se definirá el *marcado inicial aceptable* para completar la definición 12 con arreglo a los mismos criterios y con las mismas consecuencias que cuando se definen los marcados iniciales aceptables en la clase general de las S^4PR. La inactividad del sistema o estado inicial está representado a través de este marcado, el mismo permite ejecutar aisladamente cada proceso secuencial como ocurren el las redes S^4PR [Tricas 03]. De acuerdo con los requisitos establecidos en la subsección 3.2.1 solo se permitía una marca en cada lugar recurso de una red *SOAR*^{2} para que el marcado inicial sea admisible.

Definición 13. *Sea $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ una red SOAR^2. Un marcado inicial $\mathbf{m_0}$ es aceptable para $\mathcal{N}$ si, y solo si:*

1. $\forall i \in I_{\mathcal{N}}$, $\mathbf{m_0}[p_{0i}] > 0$.
2. $\forall p \in P_S$, $\mathbf{m_0}[p] = 0$.
3. $\forall r \in P_R$, $\mathbf{m_0}[r] = 1$.

En la definición 13, *el marcado inicial de p_{0i}* (condición 13.1): representa el número máximo de mensajes u objetos que pueden estar destinados concurrentemente al mismo nodo o la misma estación destino. El *marcado inicial de los lugares procesos* (condición 13.2) indica que no existe ningún objeto o dato en tránsito en el sistema en su estado inicial. *El marcado inicial de los recursos r* (condición 13.3): indica la capacidad máxima que poseen los recursos.

Los lugares proceso que usan un recurso son denominados *usuarios* de este recurso. Según la definición 14, dado un recurso r y basado en los p–semiflujos mínimos $\mathbf{y}_r$ el *conjunto de usuarios* del recurso r se definen como el conjunto de los lugares proceso del soporte de $\mathbf{y}_r$ que pueden utilizar dicho recurso.

Definición 14. *Sea $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ una red SOAR^2. Sea $r \in P_R$.* El conjunto de usuarios de *r es el soporte del p–semiflujo mínimo $\mathbf{y}_r$ sin el lugar r* : $\mathscr{H}_r = \|\mathbf{y}_r\| \setminus \{r\}$.

Esta definición puede extenderse de manera natural a los conjuntos de recursos $A \subseteq P_R$: $\mathscr{H}_A = \bigcup_{r \in A} \mathscr{H}_r$.

3.3. Propiedades estructurales de las redes *SOAR*^{2} basadas en las zonas

Como se ha visto en la sección anterior, las zonas de uso continuado de un recurso resultan muy útiles a la hora de definir la clase de redes *SOAR*^{2} que aparecen en algoritmos

de encaminamiento mínimo de tipo wormhole. Es decir, las zonas nos permiten capturar los requisitos de modelado sobre el orden de asignación y liberación de recursos de una forma natural y esto conduce a una definición compacta de la clase basada en la relación de orden definida entre las zonas solapadas.

3.3.1. Propiedades básicas de los cerrojos mínimos en redes *SOAR*^{2}

A continuación se analizarán un conjunto de propiedades de estas redes que se derivan de la definición de zona y las relaciones que existen entre ellas en la clase de redes *SOAR*^{2}. La primera de las propiedades relaciona las zonas asociadas a un recurso con el soporte de un p–semiflujo de este mismo recurso.

Lema 9. *Sea $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C}\rangle$ una red SOAR^2 y sea $r \in P_R$ un recurso de $\mathcal{N}$. El conjunto de las zonas de uso continuado de r satisfacen,*

1. *$\mathscr{H}_r = \bigcup_{i\in I}(\bigcup_{j\in J_{r,i}} \mathscr{Z}^r_{i,j})$; es decir, la unión de todas las zonas asociadas a r, es igual al conjunto de los lugares* usuarios *de r;*
2. *$\|\mathbf{y}_r\| = \{r\} \cup \mathscr{H}_r = \{r\} \cup (\bigcup_{i\in I}(\bigcup_{j\in J_{r,i}} \mathscr{Z}^r_{i,j}))$;*
3. *$\forall i, i' \in I; \forall j \in J_{r,i}$ y $\forall j' \in J_{r,i'}$ tal que $i \neq i'$ ó $j \neq j'$; $\mathscr{Z}^r_{i,j} \cap \mathscr{Z}^r_{i',j'} = \emptyset$.*

Donde I es el conjunto de índices que identifica las máquinas de estado de la red $\mathcal{N}$, $J_{r,i}$ es el conjunto de índices que identifica las zonas de uso continuado del recurso r en la máquina de estados i–ésima, y $\mathscr{Z}^r_{i,j}$ es la j–ésima zona de uso continuado de r en la máquina de estados i–ésima.

Demostración.

1. Por la definición 6 de zona de uso continuado de un recurso se verifica que $\mathscr{Z}^r_{i,j} \subseteq \mathscr{H}_r \cap P_{Si}$. Dado que el algoritmo 4 procede exhaustivamente en el cálculo de las zonas de r sobre el conjunto $\|\mathbf{y}_r\| \setminus \{r\}$ se concluye que la unión de todas las zonas de r es el conjunto de lugares usuarios de r.
2. Se deduce de la definición de lugares usuarios y de la propiedad anterior.
3. De la definición 6 de zona de uso continuado de un recurso r se cumple que el conjunto de lugares $\mathscr{Z}^r_{i,j}$ es máximo y la subred generada por $\mathscr{Z}^r_{i,j} \cup ((\mathscr{Z}^r_{i,j})^\bullet \cap {}^\bullet(\mathscr{Z}^r_{i,j}))$ es conexa. Por lo tanto, se cumple que si la intersección de las zonas fuese no–vacía, estas zonas no podrán ser conjuntos de lugares máximos.

□

Como consecuencia de que la relación de precedencia sobre las zonas de un conjunto máximo de zonas solapadas es una relación de orden total se deriva el siguiente resultado.

Lema 10. *Sea $\mathscr{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ una red SOAR^2 y sea $\mathscr{Z} = \{Z_1, Z_2, .., Z_k\}$ un conjunto máximo de zonas solapadas tales que la relación de precedencia define el siguiente orden total: $Z_1 < Z_2 < ... < Z_k$. Se cumple que $Fl(Z_i) \subseteq Z_j$ para todo $i < k$ y para todo $j > i$.* □

Demostración. Por la definición de la relación de precedencia si $Z_u < Z_{u+1}$ entonces $Fl(Z_u) \subseteq Z_{u+1}$. Si $Z_u < Z_v < Z_x$ con $u < v < x$, dado que la relación de precedencia es una relación de orden total estricta se cumple que $Z_u < Z_x$ y por tanto $Fl(Z_u) \subseteq Z_x$ para todo $x > u$. □

De manera informal, a partir del resultado anterior, se puede decir que el conjunto de zonas de uso continuado de un recurso r, es el conjunto de los conjuntos de lugares proceso cada uno de los cuales es una parte máxima del soporte del p–semiflujo en el sentido que la subred generada por la zona y sus transiciones internas forma una red conexa.

Los siguientes resultados analizan las propiedades de las zonas en relación con los cerrojos de una red *SOAR*^{2}. En este sentido conviene recordar aquí que la clase de redes *SOAR*^{2} se trata de una subclase estricta de las redes S^4PR y en este sentido sus cerrojos verifican las propiedades que se demostraron para este objeto en [Cano 10]. A continuación se recuerdan estas propiedades con el objeto de dar completitud al trabajo aquí presentado.

Lema 11 ([Cano 10])**.** *Sea $\mathscr{N}$ una red S^4PR y $D \subseteq P$ un cerrojo mínimo de $\mathscr{N}$. $D \cap P_R \neq \emptyset \Rightarrow D \cap P_S \neq \emptyset$.*

Lema 12 ([Cano 10])**.** *Sea $\mathscr{N}$ una red S^4PR y $D \subseteq P$ un cerrojo mínimo no–vacío de $\mathscr{N}$ conteniendo al menos un lugar recurso. D es el único cerrojo mínimo de $\mathscr{N}$ conteniendo exactamente el conjunto de recursos $D_R = D \cap P_R$.*

A diferencia de la clase S^4PR, en la clase *SOAR*^{2} los cerrojos pertenecientes a la clase $\mathscr{D}^1$, es decir cerrojos mínimos que contienen exactamente un recurso, no pueden estar estrictamente contenidos en el soporte del p–semiflujo del recurso que contiene. Mostramos este resultado en el siguiente lema 13.

Lema 13. *Sea $\mathscr{N}$ una red SOAR^2. Para cada $r \in P_R$ existe un cerrojo mínimo, $D_r \in \mathscr{D}^1$, tal que $D_r = \|\mathbf{Y}_r\|$.*

Demostración. El soporte de $\mathbf{y}_r$ es un cerrojo por construcción ya que ${}^\bullet\|\mathbf{y}_r\| \subseteq \|\mathbf{y}_r\|^\bullet$ y contiene exactamente un recurso r. Demostramos a continuación que $\|\mathbf{y}_r\|$ es un cerrojo mínimo por contradicción. Supongamos que $\|\mathbf{y}_r\|$ es no mínimo. Esto quiere decir que existen al menos un lugar $p \in \|\mathbf{y}_r\|$ que es no esencial para el cerrojo, es decir, que si se elimina de $\|\mathbf{y}_r\|$ lo que queda es un cerrojo.

Ahora bien, p no puede ser el recurso r ya que en este caso el cerrojo $\|\mathbf{y}_r\| \setminus \{r\}$ estaría formado solo por lugares proceso. Sea q uno de estos lugares proceso, por la propiedad de cerrojo y teniendo en cuenta que todos los circuitos de la máquina de estados pasan por el lugar reposo, llegaríamos a que el lugar reposo pertenece a $\|\mathbf{y}_r\| \setminus \{r\}$ pero esto no es posible por la definición de redes S^4PR que dice que el lugar reposo nunca puede pertenecer a un p–semiflujo mínimo con recurso.

El lugar p tampoco puede ser un lugar proceso porque para que llegue a ser no esencial se requiere que para toda transición $t \in p^\bullet$ tal que $t^\bullet \cap \|\mathbf{y}_r\| \setminus \{p\} \neq \emptyset$ exista otro lugar $q \in {}^\bullet t$ que también pertenezca a $\|\mathbf{y}_r\| \setminus \{p\}$. La única posibilidad es que el lugar q anterior sea el lugar recurso r ya que $\|\mathbf{y}_r\| \setminus \{r\}$ son lugares proceso que pertenecen a una máquina de estados fuertemente conexa. Ahora bien, esto es imposible dado que tendríamos que la transición t verificaría:

1. ${}^\bullet t \cap \|\mathbf{y}_r\| = \{p, r\}$.
2. $t^\bullet \cap \|\mathbf{y}_r\| = \{s\}, s \in P_S$.

Observe que s solo puede ser un lugar proceso ya que r es el único recurso que pertenece a $\|\mathbf{y}_r\|$. Por lo tanto; si tenemos que anular la columna t, es decir, $\mathbf{y}_r^t \cdot C[t] = 0$ no podemos tener que $\mathbf{y}_r \in \{0,1\}^{|P|}$ ya que $-\mathbf{y}_r[p] - \mathbf{y}_r[r] + \mathbf{y}_r[s] = 0$ no tiene ninguna solución binaria con las tres variables distintas de cero.

Una vez demostrado que $\|\mathbf{y}_r\|$ es mínimo, por el lema 12 anterior se trata de único cerrojo mínimo que contiene al conjunto de recursos r. □

A partir de estos resultados y en particular debido al lema 12 presentado en [Cano 10], para la clase de redes S^4PR se puede establecer las siguientes cotas alcanzables del número de cerrojos mínimos, n_D.

$$n_I + n_R \leq n_D \leq n_I + 2^{n_R - 1} \tag{3.1}$$

Donde n_I es el número de máquinas de estado fuertemente conexas de la red, y $n_R = \mid P_R \mid$, i,e. el número de lugares recurso de la red. Finalmente, se recuerdan tres propiedades adicionales presentadas en [Cano 10].

Lema 14 ([Cano 10])**.** *Sea $\mathcal{N}$ una red S^4PR, $D \subseteq P$ un cerrojo mínimo no–vacío de $\mathcal{N}$ y $D_R = D \cap P_R \neq \emptyset$. $D \subseteq \bigcup_{r \in D_R} \|\mathbf{y}_r\|$.*

En [Cano 10] se presentaba un lema previo a este anterior para la clase de redes S^4PR en la que en lugar de la relación de inclusión anterior se escribía $D \subseteq \bigcup_{r \in D_R} D_r$, $D_r \in \mathscr{D}^1$. Obsérvese que aunque es cierto para el caso de las redes $SOAR^2$, no resulta interesante para la clase de redes $SOAR^2$, ya que después del lema 13, la última relación de inclusión se convierte en la relación de inclusión del lema 14, esto es debido a que se cumple que $D_r = \|\mathbf{y}_r\|, \forall r \in P_R$.

Lema 15 ([Cano 10])**.** *Sea $\mathscr{N}$ una red S^4PR, y $D \subseteq P$ un cerrojo mínimo no–vacío de $\mathscr{N}$ conteniendo al menos un lugar recurso, $D \cap P_0 = \emptyset$.*

3.3.2. Zonas de recursos y cerrojos mínimos en $SOAR^2$: relaciones y propiedades

El punto importante a considerar ahora es la relación existente entre cerrojos de una red $SOAR^2$ y las correspondientes zonas. El siguiente resultado pone de manifiesto una primera relación simple.

Lema 16. *Sea $\mathscr{N}$ una red $SOAR^2$, y $D \subseteq P$ un cerrojo mínimo no–vacío de $\mathscr{N}$ tal que $\mid D_R \mid \geq 2$, donde $D_R = D \cap P_R$ es el conjunto de lugares recursos contenidos en D.*

1. *Para cada $r \in D_R$ existe al menos una zona de uso continuado de r, $\mathscr{Z}^r_{i,j}$, tal que $\mathscr{Z}^r_{i,j} \not\subseteq D$.*
2. *Para toda zona de uso continuado de un recurso $r \in D_R$, $\mathscr{Z}^r_{i,j}$, se verifica que $\mathscr{Z}^r_{i,j} \cap D \neq \emptyset$.*

Demostración. Lo demostraremos por contradicción.

1. Si toda zona de uso continuado de r estuviese contenida en D, $\mathscr{Z}^r_{i,j} \subseteq D, \forall i, j$; de acuerdo con los lemas 9 y 13 se verificaría.

$$D_r = \{r\} \cup (\cup_{i \in I}(\cup_{j \in J_{r,i}} \mathscr{Z}^r_{i,j})) \subseteq D \tag{3.2}$$

   Es decir, que el cerrojo mínimo D_r está contenido en D, contradiciendo que D es un cerrojo mínimo.

2. Supongamos que existe una zona $\mathscr{Z}^r_{i,j}$, para algún $r \in D_R$ tal que $\mathscr{Z}^r_{i,j} \cap D = \emptyset$. Sea p un lugar proceso tal que $p \in Fl(\mathscr{Z}^r_{i,j}) = \mathscr{Z}^r_{i,j} \setminus {}^\bullet((\mathscr{Z}^r_{i,j})^\bullet \cap {}^\bullet(\mathscr{Z}^r_{i,j}))$. Dado que $\mathscr{Z}^r_{i,j} \cap D = \emptyset$ se cumple que $p \notin D$. Por la definición de zona de uso continuado de un recurso, se cumplirá que $r \in (p^\bullet)^\bullet$ y $r \in D$. Por lo tanto llegamos a la contradicción de que $r \notin D$ ya que para que se cumpla la propiedad de cerrojo relativa a r o bien

$p \in D$ o bien existe otro recurso $r' \neq r$ tal que $r' \in {}^\bullet(p^\bullet)$ y $r' \in D$, cosa que no puede ocurrir ya que en las redes *SOAR*^{2} cada transición solo puede estar conectada con un recurso (mediante un arco de entrada o de salida).

□

De manera informal, del resultado anterior podemos concluir que no todas las zonas de un recurso de un cerrojo mínimo pueden estar incluidas en el cerrojo. En otras palabras un cerrojo mínimo con más de un recurso se cumplirá que algún lugar proceso perteneciente a una zona de un recurso del cerrojo llega a ser no esencial gracias a algún otro recurso del cerrojo y es por ello que alguna zona no puede llegar a estar incluida en el cerrojo si queremos respetar su minimalidad. La parte segunda del lema anterior establece que toda zona de los recursos del cerrojo mantienen al menos un lugar dentro del cerrojo y por lo tanto a través de los lugares proceso del cerrojo se puede rastrear todas las zonas de todos los recursos del cerrojo.

En los resultados posteriores vamos a tratar de ser más precisos en cuanto a los lugares de las zonas de los recursos de un cerrojo mínimo que llegan a ser no–esenciales para construir un cerrojo mínimo que contenga dos o más recursos. Recordamos aquí nuevamente que los cerrojos mínimos que contienen solo un recurso son los soportes de los p–semiflujos mínimos del recurso que contienen. Por lo tanto ningún lugar de ninguna zona del recurso del cerrojo llega a ser no–esencial.

En primer lugar procederemos a formalizar el concepto de conjunto máximo de zonas solapadas de recursos pertenecientes a un cierto conjunto, en particular al conjunto de recursos que pertenecen a un cerrojo mínimo.

Definición 15. *Sea $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ una red SOAR^2 y sea $\mathscr{Z}$ el conjunto de zonas de uso continuado de los recursos de un conjunto $A \subseteq P_R$. Sea $\mathscr{Z}_A \subseteq \mathscr{Z}$ un conjunto no vacío de zonas solapadas, i.e. $\bigcap_{\mathscr{Z}^r \in \mathscr{Z}_A} \mathscr{Z}^r \neq \emptyset$ y $\mathscr{Z}_A \neq \emptyset$. Se dice que $\mathscr{Z}_A$ es un conjunto máximo de zonas solapadas o* unidad de recursos de *A si y solo si no existe otro conjunto de zonas solapadas $\mathscr{Z}_B \subseteq \mathscr{Z}$, $\mathscr{Z}_B \neq \mathscr{Z}_A$, tal que $\mathscr{Z}_A \subseteq \mathscr{Z}_B$.*

Obviamente esta es una condición muy próxima a la definición 9 de la sección 3.2 de este capítulo y la única diferencia introducida es que allí se hacía referencia al conjunto completo de recursos P_R para definir los conjuntos máximos de zonas solapadas, mientras que aquí se considera cualquier subconjunto de recursos tanto para definir el solapamiento como para caracterizar los conjuntos máximos. El interés de ello radica en que posteriormente nos vamos a centrar en conjuntos de recursos que sean los contenidos en un cerrojo mínimo. Es decir, el conjunto A de la definición 15 será un conjunto $D_R \subseteq P_R$ tal que $D_R = D \cap P_R$ y D es un cerrojo mínimo de la red *SOAR*$^2, \mathcal{N}$.

Obsérvese también que para calcular estos conjuntos máximos de zonas solapadas de A podemos utilizar el algoritmo 5 presentado anteriormente en este capítulo alimentandolo en la entrada con la red *SOAR*^{2}, $\mathscr{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$, que obviamente se trata de una red S^4PR como el algoritmo requiere; y el conjunto de zonas $\mathscr{Z}_A = \{\mathscr{Z}^r_{i,j} \mid$ es la j–ésima zona de $r \in A$, en la i–ésima máquina de estados de $\mathscr{N}\}$. Para calcular el conjunto de zonas $\mathscr{Z}_A$ se puede utilizar el algoritmo 4 de este capítulo alimentando la entrada con una red $\mathscr{N}$ que sea la red original *SOAR*^{2} pero en la que solo se hayan mantenido los recursos del conjunto A. Adicionalmente, solo se suministrarán los p–semiflujos mínimos de la red correspondientes a los recursos del conjunto A para los que pretendemos calcular el conjunto de zonas.

No obstante, de los comentarios anteriores parece inferirse que la única forma de calcular los conjuntos máximos de zonas solapadas de un conjunto de recursos sea reiniciar el algoritmo 5 con cada nuevo conjunto sin poder reaprovechar ningún cálculo previo. El siguiente resultado persigue realizar este cálculo no ejecutando el algoritmo 5 cada vez que tengamos un conjunto nuevo de recursos, sino que el algoritmo 5 se ejecutará tan solo una vez para todos los recursos de P_R *(es decir, ejecutar el algoritmo 5 tal y como aparece anteriormente en el capítulo)* y a partir de los conjuntos máximos de zonas solapadas del conjunto de recursos P_R calcular los conjuntos máximos de zona solapadas pero restringidas a un subconjunto de recursos dado. La operación que se define a continuación justifica y formaliza esta estrategia y puede ser vista de una manera intuitiva como una operación de restricción o proyección referida a un subconjunto de recursos. La propiedad como se verá a través de un ejemplo posteriormente, se verifica para redes de la clase *SOAR*^{2} pero para clases de redes S^4PR más generales que las *SOAR*^{2} no se cumple.

Lema 17. *Sea $\mathscr{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ una red SOAR^2 y sea $\mathscr{Z}$ el conjunto de zonas de uso continuado de la red $\mathscr{N}$. Sea $\mathscr{Z}_A \in 2^{\mathscr{Z}}$ un conjunto máximo de zonas solapadas de $\mathscr{N}$. Sea $B \subseteq P_R$ un conjunto de recursos. El conjunto de zonas.*

$$\mathscr{Z}_B = \{\mathscr{Z}^r_{i,j} \mid \mathscr{Z}^r_{i,j} \in \mathscr{Z}_A \text{ y } r \in B\} \tag{3.3}$$

Es un conjunto máximo de zonas solapadas de recursos de B.

Demostración. En primer lugar demostraremos que si $\mathscr{Z}_A$ es un conjunto de zonas solapadas entonces $\mathscr{Z}_B$, también es un conjunto de zonas solapadas. Por la definición de conjunto de zonas solapadas para $\mathscr{Z}_A$ se cumple que,

$$\bigcap_{\mathscr{Z}^r \in \mathscr{Z}_A} \mathscr{Z}^r \neq \emptyset \tag{3.4}$$

Teniendo en cuenta la definición de $\mathscr{Z}_B$ se cumple que $\mathscr{Z}_B \subseteq \mathscr{Z}_A$ y por la asociatividad de la intersección de conjuntos.

$$(\bigcap_{\mathscr{Z}^r \in \mathscr{Z}_B} \mathscr{Z}^r) \cap (\bigcap_{\mathscr{Z}^s \in \mathscr{Z}_A \setminus \mathscr{Z}_B} \mathscr{Z}^r) \neq \emptyset \tag{3.5}$$

Para que la intersección anterior sea distinta de vacío se requiere que cada uno de los paréntesis sea distinto de vacío y por lo tanto:

$$\bigcap_{\mathscr{Z}^r \in \mathscr{Z}_B} \mathscr{Z}^r \neq \emptyset \tag{3.6}$$

O lo que es lo mismo, $\mathscr{Z}_B$ es un conjunto de zonas solapadas.

A continuación demostraremos que si $\mathscr{Z}_A$ es máximo cuando consideramos el conjunto P_R, entonces $\mathscr{Z}_B$ es también máximo pero referido al conjunto de recursos de B. Efectivamente, si $\mathscr{Z}_A$ es un conjunto máximo de zonas solapadas sobre P_R no se puede añadir ninguna zona adicional que mantenga la intersección de las zonas distinta de vacío por lo tanto cuando nos restringimos al conjunto de recursos de B no se puede añadir ninguna zona asociada a recursos de B porque en ese caso $\mathscr{Z}_A$ no sería máximo. □

Por lo tanto una vez aplicado el algoritmo 5 para calcular los conjuntos máximos de zonas solapadas de la red $\mathscr{N}$, si queremos restringirnos a los conjuntos máximos de zonas solapadas referidas a un conjunto de recursos $B \subseteq P_R$, basta con eliminar de cada uno de los conjuntos devueltos por el algoritmo 5 aquellas zonas asociadas a recursos que no aparecen en el conjunto B y estos conjuntos serán máximos referidos al conjunto B. Uno de los problemas que puede aparecer que puede requerir un filtrado posterior a la operación de proyección anterior es que nos aparezcan conjuntos máximos repetidos después de la eliminación de algunas zonas de los conjuntos máximos originales.

En otras palabras si ponemos en correspondencia los conjuntos máximos de zonas solapadas de $\mathscr{N}$ y los restringidos a B por la operación de proyección, la función resultante es suprayectiva.

Otra propiedad relacionada con los conjuntos máximos de zonas solapadas restringidas a un conjunto de recursos es que preservarán la relación de orden total entre las zonas que existía en el conjunto máximo de zonas solapadas de la red $\mathscr{N}$.

Lema 18. *Sea $\mathscr{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ una red SOAR^2. La operación de restricción de los conjuntos máximos de zonas solapadas de $\mathscr{N}$ a un conjunto de recursos $B \subseteq P_R$ preserva la relación de orden total entre las zonas solapadas de B.*

Demostración. Dada una zona máxima de zonas solapadas de $\mathscr{N}$, $\mathscr{Z} = \{Z_1, ...Z_k\}$ en la clase de redes *SOAR*^{2} se cumple que estas zonas están totalmente ordenadas por la relación de precedencia, y por lo tanto cualquier subconjunto de $\mathscr{Z}$ mantendrá el orden entre sus elementos que tenía en el conjunto $\mathscr{Z}$. □

3.3.3. Conjuntos máximos de zonas solapadas de recursos y cerrojos mínimos en redes *SOAR*^{2}

A continuación se presenta un lema técnico referido a la selección entre conjuntos máximos de zonas solapadas de un conjunto de recursos D_R y el cerrojo mínimo que contiene exactamente ese conjunto de recursos D_R. Dicha relación establece que de los lugares de las zonas del conjunto máximo solo encontraremos como mucho los lugares de la zona a la que todas las demás preceden.

Lema 19. *Sea $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ una red SOAR^2. Sea D un cerrojo mínimo que contiene el conjunto de recursos $D_R \neq \emptyset$ y $\mathscr{Z}$ un conjunto máximo de zonas solapadas de los recursos de D_R. Sea $Z^r \in \mathscr{Z}$, $r \in D_R$, la zona máxima según la ordenación total de las zonas $\mathscr{Z}$ mediante la relación $<$, es decir, $\forall Z^s \in \mathscr{Z}$, $Z^s \neq Z^r$ se cumple que $Z^s < Z^r$. En estas condiciones se verifica que,*

$$((\bigcup_{Z^s \in \mathscr{Z}} Z^s) \setminus Z^r) \cap D = \emptyset \text{ y } Z^r \cap D \neq \emptyset \tag{3.7}$$

Demostración. Sea p un lugar que pertenece a la unión de las zonas de $\mathscr{Z}$ pero no pertenece a Z^r, i.e. $p \in (\bigcup_{Z^s \in \mathscr{Z}} Z^s) \setminus Z^r$. Demostramos que p no es esencial para el cerrojo mínimo D y por lo tanto $p \notin D$.

Si $p \in (\bigcup_{Z^s \in \mathscr{Z}} Z^s) \setminus Z^r$ teniendo en cuenta que Z^r es el máximo de la ordenación total de las zonas $\mathscr{Z}$ según la relación de orden $<$, entonces p tiene que encontrarse en algún camino desde un lugar de la frontera de asignación de una $Z^s \neq Z^r$ hasta un lugar de la frontera de asignación Z^r *(sin pertenecer a esta frontera porque sino pertenecería a Z^r)*. Obsérvese que todos estos caminos terminan en la frontera de asignación Z^r cuyas transiciones de entrada tienen como lugar recurso de entrada al recurso r y un lugar proceso de alguna de las zonas $Z^s \in \mathscr{Z}$ pero $Z^s \neq Z^r$. Además, en cada uno de estos caminos solo encontramos transiciones de asignación de recursos y nunca de liberación de recursos, porque $\mathscr{Z}$ es un conjunto máximo de zonas solapadas de D_R de una red *SOAR*^{2} donde estas zonas están totalmente ordenadas por la relación $<$. De todo lo anterior se deduce que ${}^\bullet Fa(\mathscr{Z}^r)$ son transiciones que tienen como lugares de entrada el recurso r que pertenece a D_R y los lugares proceso que pertenecen a zonas $Z^s \neq Z^r$. Por lo tanto, si existe un lugar de salida de estas transiciones que pertenece a D *(y que solo puede ser un lugar proceso ya que tiene como entrada al lugar recurso r)*, para que se cumpla la propiedad cerrojo ya está el lugar r que pertenece al cerrojo mínimo, en otras palabras, el lugar proceso es no esencial. Tampoco serán esenciales ninguno de los lugares proceso en los caminos desde las fronteras de las zonas $Z^s \in \mathscr{Z}$, $Z^s \neq Z^r$ hasta la frontera Z^r porque no hay ningún lugar recurso de salida de las transiciones visitadas por los caminos que pertenecen al cerrojo. □

A partir del lema anterior se puede concebir un método para la construcción de cerrojos que se apoyen en la utilización de los conjuntos máximos de zonas solapadas. Es decir; aprovechando que conocemos qué lugares de las zonas en un conjunto máximo de ellas seguro que no pertenecen a un cerrojo mínimo podemos ir seleccionando las zonas que contribuyan a la formación del cerrojo.

Por ejemplo vamos a analizar las distintas situaciones que pueden aparecer a partir de los conjuntos máximos de zonas solapadas de la red de la figura 3.5. Los conjuntos máximos y las relaciones de orden se describen en el cuadro 3.1.

Conjuntos máximos de zonas solapadas	Relación de orden
$\{Z^r_{1,1}; Z^s_{1,1}\}$	$Z^r_{1,1} < Z^s_{1,1}$
$\{Z^r_{1,2}; Z^t_{1,1}\}$	$Z^t_{1,1} < Z^r_{1,2}$
$\{Z^s_{1,1}; Z^t_{1,1}\}$	$Z^s_{1,1} < Z^t_{1,1}$
$\{Z^r_{2,1}; Z^s_{2,1}; Z^t_{2,1}\}$	$Z^t_{2,1} < Z^s_{2,1} < Z^r_{2,1}$

Cuadro 3.1: Conjuntos máximos de zonas solapadas de la red de la figura 3.5 y su ordenación.

Por lo tanto si deseamos determinar si existe un cerrojo mínimo que contiene los recursos $D_r = \{r,s\}$, procederemos en primer lugar a restringir los conjuntos máximos anteriores a este conjunto de recursos, quedando en este caso el cuadro anterior reducido al siguiente cuadro 3.2

Conjuntos máximos de zonas solapadas de {r,s}	Relación de orden
$\{Z^r_{1,1}; Z^s_{1,1}\}$	$Z^r_{1,1} < Z^s_{1,1}$
$\{Z^r_{1,2}\}$	$Z^r_{1,2}$
$\{Z^s_{1,1}\}$	$Z^s_{1,1}$
$\{Z^r_{2,1}; Z^s_{2,1}\}$	$Z^s_{2,1} < Z^r_{2,1}$

Cuadro 3.2: Conjuntos máximos de zonas solapadas de los recursos r y s de la red de la figura 3.5 y su ordenación.

Obsérvese que si todos los conjuntos máximos de zonas solapadas restringidos a D_R hubieran quedado como el segundo y el tercero de la tabla 3.2, estaría claro que no podríamos generar un cerrojo mínimo conteniendo solo $D_R = \{r,s\}$, ya que en este caso no eliminaríamos ningún lugar de los conjuntos máximos de las zonas solapadas por lo tanto el cerrojo contendría los cerrojos mínimos de r o de s *(cerrojos de un solo recurso)*. En

esencia estamos aplicando el lema 19 que nos da una condición necesaria para que pueda existir un cerrojo mínimo conteniendo un conjunto de recursos determinados.

De acuerdo con los comentarios anteriores, nos fijamos en el conjunto máximo de zonas solapadas $\{Z^r_{1,1}; Z^s_{1,1}\}$ en el que la ordenación total de estas zonas es $Z^r_{1,1} < Z^s_{1,1}$, y conforme al lema 19, se concluye que los lugares $Z^r_{1,1} \setminus Z^s_{1,1} = \{p_1\}$ no pueden pertenecer al cerrojo que contenga los recursos $D_R = \{r,s\}$. Además, el lema establece que al menos uno de los lugares $Z^s_{1,1}$ debe pertenecer al cerrojo que contenga D_R. El cerrojo conteniendo r, D^r, ya no podrá estar contenido en el cerrojo que contenga D_R porque el lugar $p_1 \in D^r$ ha resultado ser no esencial *(en [Cano 10] se define que un cerrojo D^s poda al cerrojo D^r el lugar p_1)*. Ahora bien, esto no es suficiente para garantizar la minimalidad del cerrojo conteniendo $\{r,s\}$ ya que en lo visto hasta la fecha no hemos demostrado que exista un lugar del cerrojo D^s que resulte no esencial y por lo tanto el cerrojo D^s no está incluido en el cerrojo que contenga $\{r,s\}$. No obstante esto queda puesto de manifiesto a través del cuarto conjunto máximo de zonas solapadas de $D_R = \{r,s\}$, $\{Z^r_{2,1}; Z^s_{2,1}\}$ en el que la ordenación total de estas zonas es $Z^s_{2,1} < Z^r_{2,1}$ *(estas zonas de los recursos r y s están ordenadas en sentido contrario a las zonas que hemos considerado anteriormente)*.

Conforme al lema 19 se concluye que los lugares $Z^s_{2,1} \setminus Z^r_{2,1} = \{q_2\}$ no pueden pertenecer al cerrojo que contenga los recursos $D_R = \{r,s\}$. Además, el lema establece que al menos uno de los lugares $Z^r_{2,1}$ debe pertenecer al cerrojo que contenga D_R. En otras palabras, y de manera análoga al análisis realizado con el otro conjunto máximo de zonas solapadas, el cerrojo conteniendo solo s, D^s, ya no podrá estar contenido en el cerrojo que contenga D_R porque el lugar $q_2 \in D^s$ ha resultado ser no esencial *(en [Cano 10] se diría que el cerrojo D^r poda al cerrojo D^s el lugar q_2)*.

Por lo tanto para este ejemplo concreto observamos que a partir de los conjuntos máximos de zonas solapadas de $D_R = \{r,s\}$ se puede construir el cerrojo mínimo *(que será único)* que contiene precisamente D_R mediante la siguiente ecuación que es un cerrojo mínimo conteniendo los recursos r y s.

$$
\begin{aligned}
D = ((D_R \cup (Z^r_{1,1} \cup Z^r_{1,2} \cup Z^r_{2,1}) \cup \\
\cup (Z^s_{1,1} \cup Z^s_{2,1})) \setminus ((Z^r_{1,1} \cup Z^s_{1,1}) \setminus Z^s_{1,1})) \\
\setminus ((Z^r_{2,1} \cup Z^s_{2,1}) \setminus Z^r_{2,1}) = \\
(((\{r,s\} \cup \{p_1, p_2, p_6, p_7, q_3, q_4, q_5\} \cup \\
\cup \{p_2, p_3, p_4, q_2, q_3, q_4\}) \setminus (\{p_1\})) \setminus (\{q_2\}) = \\
= \{r, s, p_2, p_6, p_7, q_3, q_4, q_5\}
\end{aligned}
\tag{3.8}
$$

Conjuntos máximos de zonas solapadas de {r,t}	Relación de orden
$\{Z^r_{1,1}\}$	$Z^r_{1,1}$
$\{Z^r_{1,2}; Z^t_{1,1}\}$	$Z^t_{1,1} < Z^r_{1,2}$
$\{Z^t_{1,1}\}$	$Z^t_{1,1}$
$\{Z^r_{2,1}; Z^t_{2,1}\}$	$Z^t_{2,1} < Z^r_{2,1}$

Cuadro 3.3: Conjuntos máximos de zonas solapadas de los recursos r y t de la red de la figura 3.5 y sus ordenaciones.

Si reproducimos los cálculos anteriores pero ahora considerando el conjunto de recursos $D_R = \{r,t\}$ obtenemos los siguientes conjuntos máximos de zonas solapadas D_R y las ordenaciones correspondientes.

Aplicando el lema 19 al segundo conjunto $\{Z^r_{1,2}; Z^t_{1,1}\}$ en el que $Z^t_{1,1} < Z^r_{1,2}$ se concluye que los lugares $Z^t_{1,1} \setminus Z^r_{1,2} = \{p_4, p_5\}$ no pueden pertenecer al cerrojo que contenga $D_R = \{r,t\}$. De la misma manera aplicando el lema 19 al cuarto conjunto $\{Z^r_{2,1}; Z^t_{2,1}\}$ en el que $Z^t_{2,1} < Z^r_{2,1}$ se concluye que los lugares $Z^t_{2,1} \setminus Z^r_{2,1} = \{q_1, q_2\}$. Es decir, en ambos casos los únicos lugares que llegan a ser no esenciales pertenecen a zonas del recurso t y por lo tanto el cerrojo mínimo D^r que contiene solo el recurso r está completamente contenido en el cerrojo que contiene $D_R = \{r,t\}$ ya que el recurso t no ha llegado a hacer no esencial ningún lugar del cerrojo mínimo D^r. En conclusión, en este caso no existe un cerrojo mínimo que contenga exactamente a los recursos $D_R = \{r,t\}$.

Finalmente, podemos hacer los cálculos para el otro posible cerrojo mínimo conteniendo exactamente dos recursos que son $D_R = \{s,t\}$. Los conjuntos máximos de zonas solapadas de D_R y sus ordenaciones se representan en el cuadro 3.4.

Conjuntos máximos de zonas solapadas de {s,t}	Relación de orden
$\{Z^s_{1,1}\}$	$Z^s_{1,1}$
$\{Z^t_{1,1}\}$	$Z^t_{1,1}$
$\{Z^s_{1,1}; Z^t_{1,1}\}$	$Z^s_{1,1} < Z^t_{1,1}$
$\{Z^s_{2,1}; Z^t_{2,1}\}$	$Z^t_{2,1} < Z^s_{2,1}$

Cuadro 3.4: Conjuntos máximos de zonas solapadas de los recursos s y t de la red de la figura 3.5 y sus ordenaciones.

Aplicando el lema 19 al tercer conjunto $\{Z^s_{1,1}; Z^t_{1,1}\}$ en el que $Z^s_{1,1} < Z^t_{1,1}$, se concluye que los lugares $Z^s_{1,1} \setminus Z^t_{1,1} = \{p_2, p_3\}$ no pueden pertenecer al cerrojo que contenga $D_R =$

$\{s,t\}$. De la misma manera, aplicando el lema 19 al cuarto conjunto $\{Z^s_{2,1};Z^t_{2,1}\}$ en el que $Z^t_{2,1} < Z^s_{2,1}$ se concluye que los lugares $Z^t_{2,1} \setminus Z^s_{2,1} = \{q_1\}$. Es decir, en este caso el cerrojo que contiene los recursos $D_R = \{s,t\}$ es mínimo y se puede obtener de la misma manera que antes a partir de los conjuntos máximos de zonas solapadas.

$$\begin{aligned} D = D_R \cup ((Z^s_{1,1} \cup Z^t_{1,1} \cup Z^s_{2,1} \cup Z^t_{2,1}) \\ \setminus (Z^s_{1,1} \setminus Z^t_{1,1})) \setminus (Z^t_{2,1} \setminus Z^s_{2,1}) = \\ = \{s,t,p_4,p_5,p_6,q_2,q_3,q_4\} \end{aligned} \tag{3.9}$$

Para terminar este ejemplo de cálculo de los cerrojos mínimos a partir de los conjuntos máximos de zonas solapadas de un conjunto de recursos, consideraremos el cálculo del cerrojo mínimo que contenga los tres recursos, i.e. $D_R = \{r,s,t\}$. En este caso, la tabla se ha presentado anteriormente incluyendo la ordenación de las zonas de cada uno de los conjuntos máximos. El cálculo conforme al lema 19 de los conjuntos de los lugares que no pueden pertenecer al cerrojo mínimo conteniendo $D_R = \{r,s,t\}$ para cada uno de los conjuntos máximos de zonas solapadas de D_R nos permite obtener los siguientes resultados en el cuadro 3.5.

Conjuntos máximos de zonas solapadas de {r,s,t}	Lugares no esenciales por el lema 19
$\{Z^r_{1,1};Z^s_{1,1}\}(Z^r_{1,1} < Z^s_{1,1})$	$Z^r_{1,1} \setminus Z^s_{1,1} = \{p_1\}$
$\{Z^r_{1,2};Z^t_{1,1}\}(Z^t_{1,1} < Z^r_{1,2})$	$Z^t_{1,1} \setminus Z^r_{1,2} = \{p_4,p_5\}$
$\{Z^s_{1,1};Z^t_{1,1}\}(Z^s_{1,1} < Z^t_{1,1})$	$Z^s_{1,1} \setminus Z^t_{1,1} = \{p_2,p_3\}$
$\{Z^r_{2,1};Z^s_{2,1};Z^t_{2,1}\}(Z^t_{2,1} < Z^s_{2,1} < Z^r_{2,1})$	$(Z^t_{2,1} \cup Z^s_{2,1}) \setminus Z^r_{2,1} = \{q_1,q_2\}$

Cuadro 3.5: Conjuntos máximos de zonas solapadas de los recursos r,s y t de la red de la figura 3.5, su ordenación y lista de lugares no esenciales.

Es decir, según los resultados anteriores y el método de cálculo que venimos realizando sobre este ejemplo obtendríamos como cerrojo mínimo conteniendo $D_R = \{r,s,t\}$.

$$\begin{aligned} D' = D_R \cup \\ \cup (Z^r_{1,1} \cup Z^s_{1,1}) \setminus (Z^r_{1,1} \setminus Z^s_{1,1}) \cup \\ \cup (Z^r_{1,2} \cup Z^t_{1,1}) \setminus (Z^t_{1,1} \setminus Z^r_{1,2}) \cup \\ \cup (Z^s_{1,1} \cup Z^t_{1,1}) \setminus (Z^s_{1,1} \setminus Z^t_{1,1}) \cup \\ \cup (Z^r_{2,1} \cup Z^s_{2,1} \cup Z^t_{2,1}) \setminus ((Z^t_{2,1} \cup Z^s_{2,1}) \setminus Z^r_{2,1}) = \end{aligned}$$

$$= \{r,s,t\} \cup \{p_1,p_2,p_3,p_4\} \setminus \{p_1\} \cup$$
$$\cup \{p_4,p_5,p_6,p_7\} \setminus \{p_4,p_5\} \cup$$
$$\cup \{p_2,p_3,p_4,p_5,p_6\} \setminus \{p_2,p_3\} \cup$$
$$\cup \{q_1,q_2,q_3,q_4,q_5\} \setminus \{q_1,q_2\} =$$
$$= \{r,s,t\} \cup \{p_2,p_3,p_4\} \cup \{p_6,p_7\} \cup \{p_4,p_5,p_6\} \cup \{q_3,q_4,q_5\}$$
$$= \{r,s,t,p_2,p_3,p_4,p_5,p_6,p_7,q_3,q_4,q_5\} \tag{3.10}$$

Como puede observar D' no es un cerrojo mínimo ya que contiene al lugar p_3 que es no esencial y por lo tanto D' contiene al verdadero cerrojo mínimo $D = \{r,s,t,p_2,p_4,p_5,p_6,p_7,q_3,q_4,q_5\}$ que contiene exactamente a los recursos r,s y t. El problema surge como consecuencia de las propias operaciones de cálculo y eliminación de los lugares no esenciales que están circunscritas a los conjuntos máximos de zonas solapadas de un conjunto de recursos. Es decir, un lugar puede llegar a ser no esencial *(aunque en la unión de las zonas solapadas de un conjunto máximo sea esencial)* a causa de otra zona que pertenece a otro conjunto máximo de zonas solapadas. Este es el caso en nuestro ejemplo donde el lugar p_3 es esencial cuando se realiza la unión de las zonas solapadas del conjunto $\{Z^r_{1,1};Z^s_{1,1}\}$ *(p_3 es esencial para p_4 y p_4 es esencial para s)*, pero la zona t, $Z^t_{1,1}$ hace que el lugar p_3 deje de ser esencial debido al lugar t; y como puede observarse $Z^t_{1,1}$ pertenece a otro conjunto máximo de zonas solapadas $\{Z^s_{1,1};Z^t_{1,1}\}$.

Una posibilidad para la eliminación de estos lugares no esenciales podría consistir en la eliminación de todos los lugares calculados como no esenciales a través del lema 19, de la unión de todas las zonas de los recursos del conjunto dado incialmente. Como se puede apreciar en el ejemplo que venimos desarrollando en los últimos párrafos esto conduce a un resultado erróneo como el siguiente.

$$D'' = D_R \cup$$
$$\cup (Z^r_{1,1} \cup Z^r_{1,2} \cup Z^s_{1,1} \cup Z^t_{1,1} \cup Z^r_{2,1} \cup Z^s_{2,1} \cup Z^t_{2,1})$$
$$\setminus ((Z^r_{1,1} \setminus Z^s_{1,1}) \cup (Z^t_{1,1} \setminus Z^r_{1,2}) \cup (Z^s_{1,1} \setminus Z^t_{1,1}) \cup ((Z^t_{2,1} \cup Z^s_{2,1}) \setminus Z^r_{2,1})) =$$
$$= \{r,s,t\} \cup \{p_1,p_2,p_3,p_4,p_5,p_6,p_7,q_1,q_2,q_3,q_4,q_5\} \setminus$$
$$(\{p_1\} \cup \{p_4,p_5\} \cup \{p_2,p_3\} \cup \{q_1,q_2\}) =$$
$$= \{r,s,t,p_6,p_7,q_3,q_4,q_5\} \tag{3.11}$$

Que como se puede apreciar ni siquiera es un cerrojo ya que se requieren como mínimo para llegar a ser un cerrojo los lugares adicionales p_2, p_4 y p_5. Esta anomalía se debe a que el conjunto $\{p_2,p_3\}$ se elimina a partir de la información del conjunto máximo de zonas

solapadas $\{Z^s_{1,1}; Z^t_{1,1}\}$ y considerando aisladamente este conjunto de zonas y no se tiene en cuenta el solape que representa el conjunto máximo $\{Z^r_{1,1}; Z^s_{1,1}\}$. Obsérvese que es por ello que p_2 es necesario. Adicionalmente, conviene resaltar que esta situación se hubiera detectado al observar que la zona $Z^s_{1,1}$ aparece en los dos conjuntos máximos de zonas solapadas implicadas, aunque realizando funciones diferentes. En el conjunto $\{Z^r_{1,1}; Z^s_{1,1}\}$ es el máximo en ordenación del conjunto y por lo tanto alguno de sus lugares debe ser retenido en el cerrojo final (lema *19)*; mientras que en la zona $\{Z^s_{1,1}; Z^t_{1,1}\}$ no es el máximo y por lo tanto podría eliminarse completamente.

3.3.4. Grafos de poda de zonas y transformación al grafo de poda de recursos.

En las secciones anteriores se han establecido las relaciones existentes entre las zonas de uso continuado de recursos y los cerrojos de una red de la clase *SOAR*^{2}. En esta sección ahondaremos en esta relación pero lo haremos a través del grafo de poda introducido para redes S^4PR en [Cano 10].

Para ello nos basaremos en tres propiedades que se han mostrado anteriormente.

1. Los cerrojos mínimos conteniendo un único recurso *(pertenecientes a la clase $\mathscr{D}^1$)* son los soportes de los p–semiflujos mínimos de los recursos *(lema 13)*.

2. La unión de las zonas de uso continuado de un recurso r forman el conjunto de lugares usuario (holders) de r, que junto al propio recurso dan lugar al soporte del p–semiflujo de r *(lema 9)*.

3. Dentro de un conjunto máximo de zonas solapadas de un conjunto de recursos, dos zonas Z^r y Z^s que satisfacen la relación de orden $Z^s < Z^r$, verifican que el conjunto de lugares $Z^s \setminus Z^r$ son los lugares no esenciales en un cerrojo mínimo que contuviese a los recursos r y s *(lema 19)*.

De la tercera propiedad anterior concluimos que existe una estrecha analogía entre los conjuntos de lugares que un cerrojo mínimo poda de otro cerrojo mínimo y que queda puesto de manifiesto a través de la relación de poda y visualizado gráficamente en el grafo de poda de la red. Para alcanzar esta interpretación vamos a proceder a introducir un grafo análogo al grafo de poda de las redes S^4PR pero en el que vamos a representar como vértices de este grafo las zonas de la red.

Definición 16 (Relación de poda entre zonas de uso continuado de recursos en redes *SOAR*^{2})**.** *Sea $\mathscr{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C}\rangle$ una red SOAR^2 y sean Z^r y Z^s dos zonas de uso continuado de recursos r y s, respectivamente, tales que ambas pertenecen a un conjunto*

máximo de zonas solapadas. La zona Z^r está en relación de poda con la zona Z^s, o simplemente la zona Z^r poda a la zona Z^s, si y solo si $Z^r < Z^s$. □

La relación de poda entre zonas como se ha definido anteriormente da lugar a que por cada conjunto máximo de zonas solapadas se genera una " cadena " de zonas relacionadas debido a que las zonas de este conjunto están totalmente ordenadas por la relación " precede "$<$.

Ahora bien, debemos recordar que una misma zona puede pertenecer a varios conjuntos máximos.

Desde el punto de vista de la notación, debemos señalar que aunque hablamos de la relación de poda referida a cerrojos y a zonas, no existirá confusión dado que en cada caso, por el contexto, quedará claro a cual de las relaciones de poda nos estamos refiriendo.

Lema 20. *Sea $\mathcal{N}$ una red SOAR^2 y Z^r, Z^s dos zonas relacionadas por la relación de poda de forma que $Z^s < Z^r$ dentro del mismo conjunto máximo de zonas solapadas. Los lugares que poda la zona Z^r a la zona Z^s en los cerrojos que contienen simultáneamente a r y a s son $Z^s \setminus Z^r$.*

Demostración. Se deriva directamente del lema 19. □

Para realizar una presentación más intuitiva y aproximarnos a los grafos de poda entre cerrojos en las redes S^4PR, representamos la relación de poda sobre las zonas asociadas a un conjunto de recursos mediante un grafo que denominaremos *Grafo de poda completo entre zonas* o simplemente *Grafo de poda completo* cuando por el contexto se sobreentienda que nos estamos refiriendo a zonas y no a cerrojos.

Definición 17 (Grafo de poda completo entre zonas)**.** *Sea $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ una red SOAR^2. Sea $A \subseteq P_R$ un conjunto de recursos y $\mathcal{Z}$ el conjunto de zonas de A. El grafo de poda completo de $\mathcal{N}$ para el conjunto de recursos de A es un grafo $G = (V, E)$ donde,*

1. *$V = \mathcal{Z}$; y*

2. *$E \subseteq V \times V$ y para todo $Z^r, Z^x \in \mathcal{Z}$, $Z^r \neq Z^x$, $(Z^r, Z^x) \in E$ si, y solo si, la zona Z^r poda a la zona Z^x, es decir, $Z^r < Z^x$.*

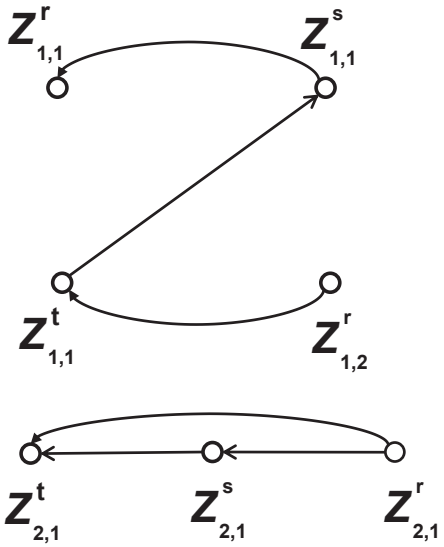

Figura 3.9: Grafo de poda completo para la red de la figura 3.5, considerando el conjunto de recursos P_R.

Para la red de la figura 3.5, el grafo de poda completo de $\mathscr{N}$ para el conjunto de recursos P_R se representa por la figura 3.9. A la vista del grafo completo de la figura 3.9 podemos hacer las observaciones iniciales siguientes.

1. En general, el grafo de poda completo de un conjunto de recursos es un grafo disconexo, existiendo al menos una componente conexa para cada una de las máquinas de estado de la red, dado que las zonas de máquinas distintas son disjuntas. Podrán existir más componentes disconexas pertenecientes a la misma máquina si éstas relacionan zonas completamente disjuntas. En la figura, existen dos componentes conexas, cada una correspondiente a una de las máquinas de estado.

2. La organización del grafo está basada en las cadenas de zonas completamente ordenadas de cada uno de los conjuntos máximos de zonas solapadas del conjunto de recursos dado. Se añaden arcos adicionales por la transitividad entre los elementos completamente ordenados de cada uno de estos conjuntos máximos. En la figura, la primera de las componentes conexas corresponde a las cadenas ordenadas de zonas de cada uno de los tres conjuntos máximos de zonas solapadas de recursos. Obsérvese que no hay arcos extras en esta componente ya que cada conjunto máximo de zonas solapadas solo tiene dos zonas. Sin embargo la segunda componente conexa, que corresponde al conjunto máximo $\{Z^t_{2,1}, Z^s_{2,1}, Z^r_{2,1}\}$, y sobre el que

la cadena ordenada de zonas es $Z^t_{2,1} < Z^s_{2,1} < Z^r_{2,1}$, además de los dos arcos obvios $(Z^t_{2,1}, Z^s_{2,1}) \in E$ y $(Z^s_{2,1}, Z^t_{2,1}) \in E$ derivados directamente de la cadena completamente ordenada, existe otra relación de poda indirecta que es $(Z^t_{2,1}, Z^r_{2,1}) \in E$ que también se ha representado en el grafo de poda completo. Como se verá más adelante estas relaciones de arcos indirectas no son necesarias para construir el grafo de poda de cerrojos conteniendo un único recurso que es el objeto con el que queremos llegar a relacionarnos en esta sección.

3. Una componente conexa puede estar formada por un único vértice aislado. Corresponde a un conjunto máximo formado por una única zona que además no aparece en ningún otro conjunto máximo.

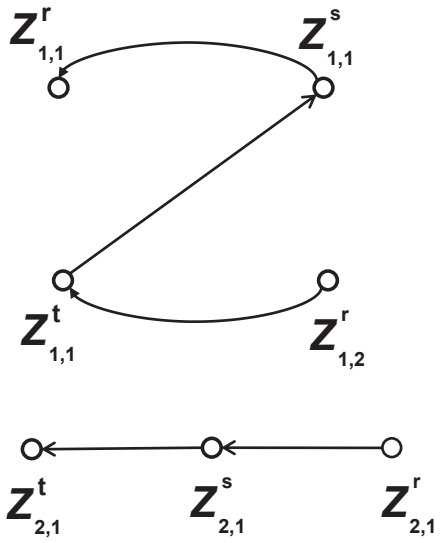

Figura 3.10: Grafo de poda simple para la red de la figura 3.5, considerando el conjunto de recursos P_R.

Dado un grafo de poda completo $G = (V, E)$ de zonas de un conjunto de recursos, definimos el grafo de poda simple ó grafo de poda sin más, como el grafo de poda en el que se han añadido nada más que los arcos que corresponden a dos zonas consecutivas en la cadena ordenada de zonas dentro de un conjunto máximo de zonas solapadas de un conjunto de recursos. Asi por ejemplo, el grafo de poda simple de la red de la figura 3.5, con respecto al conjunto de recursos P_R, es el mismo que el presentado en la figura 3.9, excepto que no tiene el arco $(Z^t_{2,1}, Z^r_{2,1})$. Asociado al grafo de poda completo o simple definimos las siguientes tres funciones de etiquetado de los conjuntos V y E.

Definición 18. *Sea $\mathscr{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ una red SOAR^2 y $G = (V,E)$ un grafo de poda simple (o completo) de $\mathscr{N}$ con respecto a un conjunto de recursos A.*

1. *La función de etiquetado de los vértices $S{:}V \to \mathscr{Z}$, donde para todo $v{\in}V$, tal que $v{=}Z^s{\in}\mathscr{Z}$ $S(v){=}Z^s$. Obsérvese que esta es una función formal para distinguir entre el nodo del grafo y la propia etiqueta, aun cuando ambos objetos en este caso se refieren a lo mismo.*

2. *La función de etiquetado de los arcos, $L : E \to 2^{T \times P_S}$, donde $\forall (Z^r, Z^x) \in E$, $L((Z^r, Z^x)) = \{(t,p) \mid t \in {}^\bullet Fa(\mathscr{Z}^r); p \in {}^\bullet t \cap P_S\}$.*

3. *El conjunto de poda $K_G : V \to 2^{P_S}$, donde para todo $v \in V$, tal que $v = Z^r \in \mathscr{Z}$ $K_G(v) = K_G(Z^r) \subseteq P_S$ y $K_G(Z^r) = Z^r \setminus (\bigcup_{Z^x \in \mathscr{Z};\, (Z^x, Z^r) \in E} Z^x)$.*

□

Para la red de la figura 3.5, el grafo de poda simple junto con las funciones de etiquetado S, L y K_G se presenta en la figura 3.10 y el cuadro 3.6 respectivamente. De las funciones de etiquetado S, L y K_G podemos realizar las siguientes observaciones.

1. La función S en realidad se trata de una función identidad que a cada zona le asigna ella misma. Se ha introducido tan solo desde el punto de vista formal para distinguir una zona cuando es tratada como el vértice del correspondiente grafo de poda y cuando es tratada como un conjunto de lugares que definen la zona de un recurso.

2. La función L permite etiquetar cada arco $(Z^r; Z^x) \in E$ con el conjunto de pares (t,p) representando un lugar candidato a ser podado cuando construyamos el cerrojo que contiene los recursos r y s. Evidentemente, estos son los lugares semilla que se utilizan en [Cano 10] para calcular el conjunto K_G en el grafo de poda de cerrojos mínimos conteniendo un único recurso. La razón de definir esta función por este grafo es por formalizar más adelante la transformación que nos permite pasar de un grafo de poda de zonas al grafo de poda de cerrojos de un recurso presentado en [Cano 10].

3. La función de etiquetado K_G es una función que tiene que ser calculada para el grafo de poda de zonas de un conjunto de recursos. Como veremos a continuación estos son los lugares que resultan ser no esenciales cuando construimos un cerrojo conteniendo los recursos para lo que se ha definido el grafo de poda de zonas.

A continuación presentamos como obtener el grafo de poda de cerrojos de $\mathscr{D}^1$ a partir de grafo de poda de zonas. La transformación va a dar lugar no al grafo de poda de la red, sino al subgrafo de poda de la red generado por el conjunto de recursos para el cual fue definido el grafo de poda de zonas de partida.

$S(Z^r_{1,1}) = Z^r_{1,1} = \{p_1, p_2\}$
$S(Z^s_{1,1}) = Z^s_{1,1} = \{p_2, p_3, p_4\}$
$S(Z^t_{1,1}) = Z^t_{1,1} = \{p_4, p_5, p_6\}$
$S(Z^r_{1,2}) = Z^r_{1,2} = \{p_6, p_7\})$
$S(Z^s_{2,1}) = Z^s_{2,1} = \{q_2, q_3, q_4\}$
$S(Z^t_{2,1}) = Z^t_{2,1} = \{q_1, q_2, q_3\}$
$S(Z^r_{2,1}) = Z^r_{2,1} = \{q_3, q_4, q_5\}$
$L((Z^s_{1,1}; Z^r_{1,1})) = \{(t_2, p_1)\}$
$L((Z^r_{2,1}; Z^s_{2,1})) = \{(t_{11}, q_2)\}$
$L((Z^t_{1,1}; Z^s_{1,1})) = \{(t_4, p_3)\}$
$L((Z^s_{2,1}; Z^t_{2,1})) = \{(t_{10}, q_1)\}$
$L((Z^r_{1,2}; Z^t_{1,1})) = \{(t_6, p_5)\}$
$K_G(Z^r_{1,1}) = Z^r_{1,1} \setminus Z^s_{1,1} = \{p_1\}$
$K_G(Z^t_{1,1}) = Z^t_{1,1} \setminus Z^r_{1,2} = \{p_4, p_5\}$
$K_G(Z^t_{2,1}) = Z^t_{2,1} \setminus Z^s_{2,1} = \{q_1\}$
$K_G(Z^s_{1,1}) = Z^s_{1,1} \setminus Z^t_{1,1} = \{p_2, p_3\}$
$K_G(Z^s_{1,1}) = Z^s_{1,1} \setminus Z^r_{1,1} = \{q_2\}$
$K_G(Z^r_{1,2}) = \emptyset$
$K_G(Z^r_{2,1}) = \emptyset$

Cuadro 3.6: Funciones de etiquetado S, L y K_G del Grafo de poda simple de la figura 3.10.

Definición 19 (Transformación del grafo de poda de zonas por aglomeración de zonas)**.** *Sea $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ una red SOAR^2, $A \subseteq P_R$ un conjunto de recursos y $\mathcal{Z}$ el conjunto de zonas de A. Sea $G=(V,E)$ el grafo de poda simple de zonas de los recursos de A. Definimos la aglomeración del grafo de poda de zonas G como el grafo $G'=(V',E')$ donde,*

1. $V' = A$
2. $E' \subseteq V' \times V'$ *de forma que* $(r,x) \in E'$*, con* $r,x \in P_R$ *si y solo si* $\exists Z^r, Z^x \in \mathcal{Z}$ *tal que* $(Z^r; Z^x) \in E$.
3. $S' : V' \to 2^{P_S}, \forall r \in V'$, $S'(r) = (\bigcup_{Z^r_{i,j} \in \mathcal{Z}} Z^r_{i,j}) \cup \{r\}$.
4. $L' : E' \to 2^{T \times P_S}, \forall (r,x) \in E'$, $L'(r,x) = \bigcup_{(Z^r_{i,j}; Z^x_{i',j'}) \in E} L((Z^r_{i,j}; Z^x_{i',j'}))$.
5. $K_{G'} : V' \to 2^{P_S}$ *donde* $\forall r \in V'$, $K_{G'}(r) = \bigcup_{Z^r_{i,j} \in \mathcal{Z}} K_G(Z^r_{i,j})$

□

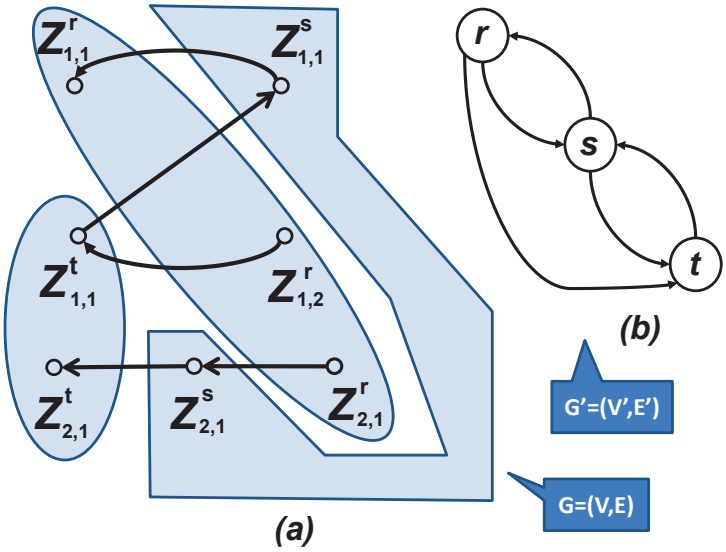

Figura 3.11: Grafo aglomerado a partir del grafo de poda de zonas de los recursos r, s y t de la red de la figura 3.5.

Funciones de etiquetado de la figura 3.11
$S'(r) = \{r\} \cup Z^r_{1,1} \cup Z^r_{1,2} \cup Z^r_{2,1} = \{r, p_1, p_2, p_6, p_7, q_3, q_4, q_5\} = \|\mathbf{y}_r\|$.
$S'(s) = \{s\} \cup Z^s_{1,1} \cup Z^r_{2,1} = \{s, p_2, p_3, p_4, q_2, q_3, q_4\} = \|\mathbf{y}_s\|$.
$S'(t) = \{t\} \cup Z^t_{1,1} \cup Z^t_{2,1} = \{t, p_4, p_5, p_6, q_1, q_2, q_3\} = \|\mathbf{y}_t\|$.
$L'((s,r)) = L((Z^s_{1,1}; Z^r_{1,1})) = \{(t_2, p_1)\}$
$L'((r,s)) = L((Z^r_{2,1}; Z^s_{2,1})) = \{(t_{11}, q_2)\}$
$L'((t,s)) = L((Z^t_{1,1}; Z^s_{1,1})) = \{(t_4, p_3)\}$
$L'((s,t)) = L((Z^s_{2,1}; Z^t_{2,1})) = \{(t_{10}, q_1)\}$
$L'((r,t)) = L((Z^r_{1,2}; Z^t_{1,1})) = \{(t_6, p_5)\}$
$K_{G'}(r) = K_G(Z^r_{1,1}) \cup K_G(Z^r_{1,2}) \cup K_G(Z^r_{2,1}) = \{p_1\}$
$K_{G'}(s) = K_G(Z^s_{1,1}) \cup K_G(Z^s_{2,1}) = \{p_2, p_3, q_2\}$
$K_{G'}(t) = K_G(Z^t_{1,1}) \cup K_G(Z^t_{2,1}) = \{p_4, p_5, q_1\}$

Cuadro 3.7: Funciones de etiquetado S, L y K_G del Grafo aglomerado de la figura 3.11.

Si aplicamos la transformación descrita en la definición 19 al grafo de poda de zonas de la red de la figura 3.5 tal y como se describe en la figura 3.10, se obtiene el grafo descrito en la figura 3.11 junto con las nuevas funciones de etiquetado que se encuentran en el cuadro 3.7 de nodos y arcos tal como se han definido.

En la figura 3.11.a observamos el grafo de poda de zonas describiendo los vértices que se proceden a aglomerar por la transformación. En la figura 3.11.b vemos el grafo de poda aglomerado a partir del grafo de poda de zonas. Las funciones de etiquetado S, L y K_G de los arcos y vértices se describen en el siguiente cuadro 3.7.

El resultado presentado a continuación establece la relación buscada entre los grafos de poda de zonas y los cerrojos $\mathscr{D}^1$.

Lema 21. *Sea $\mathscr{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ una red SOAR^2. Sea $A \subseteq P_R$ un conjunto de recursos y $\mathscr{Z}$ el conjunto de zonas de uso continuado de los recursos del conjunto A. Sea $G = (V,E)$ el grafo de poda simple de zonas de los recursos de A y sea $G' = (V',E')$ el grafo aglomerado obtenido del grafo G con arreglo a la definición 19. G' es el subgrafo de poda de cerrojos $\mathscr{D}^1$ inducido por el conjunto de recursos de A.* □

Demostración. Dado el grafo $G = (V,E)$ de poda simple de zonas, al aplicar la aglomeración establecida en la Definición 19, se obtiene un grafo en el que:

1. El conjunto de vértices es igual al conjunto de recursos sobre el que se va a construir el grafo de poda, $V' = A$, donde A es el conjunto de recursos.

2. La función de etiquetado S' de los vértices del grafo de poda de recursos asocia a cada vértice $r \in V'$ (recurso del conjunto A), de acuerdo con la definición 19, $S'(r) = (\bigcup_{Z^r_{i,j} \in \mathscr{Z}} Z^r_{i,j}) \cup \{r\}$, que de acuerdo con el lema 7.2 es el soporte del p–semiflujo mínimo del recurso r, es decir, $S'(r) = \|\mathbf{y}_r\|$. Además, por el lema 10, se cumple que $D^r = \|\mathbf{y}_r\|$, es decir, el cerrojo mínimo conteniendo el recurso r, y solo el recurso r, es precisamente el soporte del p–semiflujo mínimo de r. Por lo tanto, la función de etiquetado S' está asociando a cada vértice del grafo de poda resultante de la aglomeración, que es un recurso de A, el cerrojo mínimo que contiene exactamente el recurso asociado al vértice. Es decir, por el conjunto de vértices y la función S' el grafo G' es el grafo de poda de recursos del conjunto A.

3. De acuerdo con la definición del grafo de poda de recursos [Cano 11], existe un arco $(r,x) \in E'$, si y sólo si, el cerrojo $D^r \in \mathscr{D}^1$ poda al cerrojo $D^x \in \mathscr{D}^1$, es decir,

$$U_r = r^\bullet \cap T_{rx} \cap ({}^\bullet T_{rx} \cap D^x \cap P_s)^\bullet \neq \emptyset$$

con

$$T_{rx} = (D^r)^\bullet \cap (D^x)^\bullet.$$

Teniendo en cuenta, según el número anterior que $S'(r) = D^r = \|\mathbf{y}^r\|$ y $S'(x) = D^x = \|\mathbf{y}^x\|$ se cumplirá que $T_{rx} = (\|\mathbf{y}^r\|)^\bullet \cap (\|\mathbf{y}^x\|)^\bullet$. Además, si tenemos en cuenta que $\|\mathbf{y}^r\| = (\bigcup_{Z^r_{i,j} \in \mathscr{Z}} Z^r_{i,j}) \cup \{r\}$ y $\|\mathbf{y}^x\| = (\bigcup_{Z^x_{i,j} \in \mathscr{Z}} Z^x_{i,j}) \cup \{x\}$, $T_{rx} \neq \emptyset$ si y sólo si existen al menos una zona de r y una zona de x, tales que $Z^r_{i,j} \cap Z^x_{i,j} \neq \emptyset$ ya que entonces $(Z^r_{i,j})^\bullet \cap (Z^x_{i,j})^\bullet \neq \emptyset$. En otras palabras, si existen dos zonas solapadas, es más, T_{rx} deberá estar formado por todas las transiciones que son salida de las de los conjuntos de solape de cada pareja de zonas solapadas $(Z^r_{i,j}, Z^x_{i',j'})$. Por lo tanto, de acuerdo en la forma en la que genera el grafo de poda de recursos por la aglomeración definida en la definición 19.2 en la que se añade un arco $(r,x) \in E', r,x \in V'$, si y solo si existen dos zonas Z^r y $Z^x \in \mathscr{Z}$ tal que $(Z^r, Z^x) \in E$ o lo que es lo mismo se trata de dos zonas solapadas que además $Z^r > Z^x$, es decir, la zona Z^r poda a la zona Z^x (Definición relación de poda entre zonas). En este punto, hemos demostrado que la estructura del resultado de la aglomeración del grafo de poda de zonas y la función de etiquetado de los vértices, S', son el grafo de poda de recursos de A.

4. De acuerdo con la definición del grafo de poda de recursos del conjunto de recursos A [Cano 11], si $(r,x) \in E'$, la función de etiquetado $L' : E' \to 2^{T \times P_s}$ está definida como, $L'((r,x)) = \{(t,p) | t \in r^\bullet \cap T_{rx} \cap ({}^\bullet T_{rx} \cap S'(x) \cap P_s)^\bullet;\ T_{rx} = S'(r)^\bullet \cap S'(x)^\bullet;\ \{p\} = {}^\bullet t \cap P_s\}$.

   Obsérvese que T_{rx} contendrá para cada par de zonas solapadas $Z^x_{i',j'}$ y $Z^r_{i,j}$ tales que $Z^x_{i',j'} < Z^r_{i,j}$, todas las transiciones de salida de los lugares proceso que pertenecen al solape $Z^x_{i',j'} \cap Z^r_{i,j}$ más las transiciones de salida del recurso r (véase la figura 3.12). Por lo tanto $r^\bullet \cap T_{rx}$ será precisamente las transiciones de salida de r que obviamente pertenecen a la entrada de la frontera de asignación del recurso r, (en la figura 3.12 son las transiciones t y t'). Además, ${}^\bullet T_{rx} \cap P_s$ son los lugares que pertenecen al solape de las dos zonas, además de los lugares proceso de entrada a las transiciones de entrada a la frontera de asignación del recurso r (en la figura 3.12 los lugares p y p'). Esto quiere decir que, en el término $({}^\bullet T_{rx} \cap S'(x) \cap P_s)^\bullet$ no sólo se incluyen las transiciones de salida de la intersección de zonas sino que también se incluyen las transiciones de salida de r, con lo cual al realizar la intersección con $r^\bullet \cap T_{rx}$ obtenemos exactamente las transiciones de salida de r que son las transiciones de entrada a los lugares de la frontera de asignación de r. Por lo tanto, los pares de la función $L'((r,x))$, son pares (t,p) en las que $t \in {}^\bullet F_a(Z^r_{i,j})$ y $p \in {}^\bullet t \cap P_s$, que es exactamente lo que se realiza en la operación de aglomeración . En efecto, según la definición 19.4, L' se construye mediante la unión de las L que se han definido en el grafo de poda de zonas para cada par de zonas $Z^r_{i,j}$ y $Z^x_{i',j'}$ que están solapadas con $Z^x_{i',j'} < Z^r_{i,j}$, es decir, la zona $Z^r_{i,j}$ poda a la zona $Z^x_{i',j'}$. Para cada una de estas parejas existe un arco en el grafo de poda de zonas y además la función L para ese

arco $L((Z^r_{i,j}, Z^x_{i,j}))$ es igual a $\{(t,p)|t \in {}^\bullet F_a(Z^r_{i,j}); p \in {}^\bullet t \cap P_s\}$ que es lo que se ha establecido antes que era necesario para la construcción de la función L' en el grafo de poda de recursos del conjunto de recursos A.

5. Finalmente probamos que la función K'_G construida en la definición 19.5 es precisamente la función de etiquetado del grafo de poda de recursos del conjunto de recursos A que se especifica en [Cano 11]. La función de poda $K_{G'}$ se calcula mediante un algoritmo presentado en [Cano 11] que toma como semillas los pares (t,p) calculados para la función L' en los que p es el primer lugar no esencial del cerrojo D^x debido al recurso r porque r es un lugar de entrada a la transición t. Obviamente, el algoritmo tiene en cuenta que cuando p llega a ser no esencial, esto puede provocar que otros lugares de D^x lleguen a ser no esenciales ya que lo eran por culpa de p, y así sucesivamente. Este algoritmo de localización de lugares no esenciales por propagación cuando se alcanzan lugares a los que a sus transiciones de entrada entra el recurso x. En nuestro caso, y para las redes $SOAR^2$, si queremos calcular $K_{G'}(r)$ es decir calcular los luagres que hay que podar del cerrojo del recurso r debemos tener en cuenta todos los pares (t,p) asociados a las etiquetas $L'((x,r))$ asociadas a los arcos $(x,r) \in E'$ que existen en el grafo de poda de recursos del conjunto de recursos A. Teniendo en cuenta que las zonas del recurso r son disjuntas y que en cada par (t,p) anterior utilizado como semilla, el lugar p sólo puede pertenecer a una y solo a una zona $Z^r_{i,j}$, entonces la propagación buscando lugares no esenciales para cada semilla se puede restringir a cada zona $Z^r_{i,j}$ y los pares (t,p) en los que el lugar pertenece a la zona. Posteriormente, los lugares no esenciales detectados en cada una de las zonas $Z^r_{i,j}$ serán no esenciales en el cerrojo completo. En otras palabras, esto es lo que se especifica en la Definición 19.5 cuando se especifica que $K_{G'}(r) = \bigcup_{Z^r_{i,j} \in \mathscr{Z}} K_G(Z^r_{i,j})$, es decir, el conjunto de poda en el grafo de recursos para el recurso r es la unión de los conjuntos de poda en el grafo de poda de zonas para cada una de las zonas disjuntas del recurso r. Sólo nos queda ver que el conjunto $K_G(Z^r_{i,j})$ tal y como se define en el grafo de poda de zonas es el conjunto que calcularía el algoritmo de propagación de lugares no esenciales presentado en [Cano 11] a partir de las semillas constituidas por lugares pertenecientes a la zona $Z^r_{i,j}$. Para ello basta observar que la propagación de la no esencialidad se hace desde los lugares de entrada a las transiciones de entrada a la frontera de asignación del recurso r, que llegan a ser no esenciales a causa de r. Obviamente, estos lugares hacen no esenciales todos los lugares de entrada a sus transiciones de entrada ya que éstas últimas eran esenciales precisamente por mantener la propiedad cerrojo con los lugares no esenciales usadas como semilla. Los lugares que vamos a ir convirtiendo en no esenciales son los lugares proceso

pertenecientes a la zona $Z^x_{i',j'}$ que no pertenecen a la zona $Z^r_{i,j}$ hasta la frontera de asignación de x, es decir, que si consideramos ahora todas las zonas de recursos de A que se solapan con $Z^x_{i',j'}$ y por tanto podan a $Z^x_{i',j'}$ llegamos a la conclusión que los lugares que llegan a ser no esenciales calculados por el algoritmo de cálculo de K_G en [Cano 11] son los mismos establecidos en la ecuación de la definición 18.3 que define el K_G del grafo de poda de zonas: $K_G(Z^x) = Z^x \setminus (\bigcup_{Z^r \in \mathscr{Z};\,(Z^r,Z^x)\in E} Z^r)$ con lo cual el grafo obtenido después de la aglomeración de la definición 19 partiendo del grafo de poda de zonas de los recursos del conjunto A, es el grafo de poda de recursos pertenecientes al conjunto A.

□

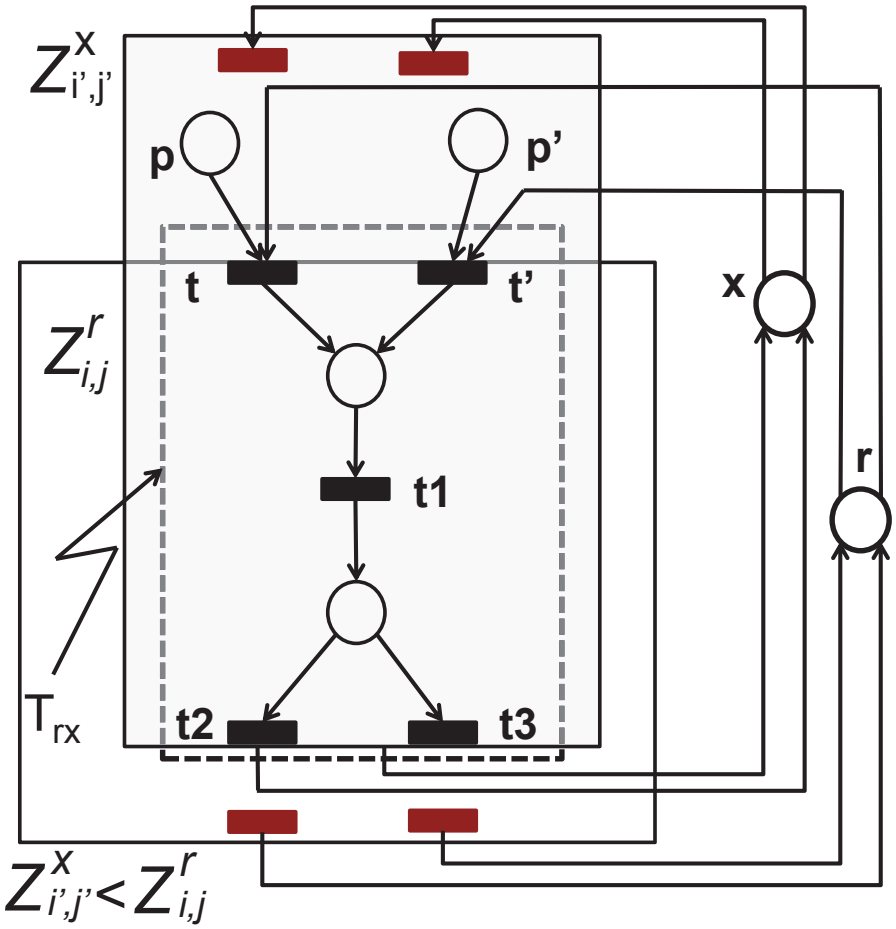

Figura 3.12: Esquema para ilustrar el paso 4 de la demostración del lema 21.

El resultado anterior nos permite concluir que en redes *SOAR*^{2} el cálculo de los grafos o subgrafos de poda se puede hacer de manera más eficiente que en el caso de las redes S^4PR tal y como se planteaba en [Cano 11]. Efectivamente, si se conoce el conjunto de zonas de uso continuado de recursos y su grafo de poda, el cálculo del grafo de poda de recursos se puede realizar aplicando cálculos algebraicos sobre los conjuntos que representan la definición de las zonas como pone de manifiesto la definición 19 y sobre la que la demostración anterior ha trabajado intensivamente. Esto quiere decir, que no es necesario volver a recalcular los K_G utilizando para ello los algoritmos de propagación de la no esencialidad sobre la red de Petri de partida como se hacía en [Cano 11], sino que en este caso tan solo se requieren simples cálculos conjuntistas.

A partir de ahí la determinación, por ejemplo, de los cerrojos mínimos de la red puede seguir los mismos métodos que se desarrollaron en [Cano 11] aplicados sobre el grafo de poda de recursos obtenido a través del procedimiento de aglomeración expuesto en la definición 19.

Obviamente, las caracterizaciones de cerrojos mínimos obtenidas para el grafo de poda de recursos podrían traducirse en términos de grafo de poda de zonas atendiendo a los ecuaciones conjuntistas de la definición 19. No obstante, consideramos que este resultado no aporta conocimiento nuevo más allá de una reformulación de los resultados de [Cano 11] en términos de zonas de uso continuado de recursos. En particular, podría traducirse al teorema de vivacidad de las redes *SOAR*^{2} formulado como si fueran redes S^4PR en términos de cerrojos, a una formulación que utilizase exclusivamente el conjunto de zonas.

Desde un punto de vista algorítmico, trabajar con el grafo de poda de zonas en lugar de el grafo de poda de recursos para el cálculo de cerrojos o determinación de la vivacidad, tampoco aporta a priori ventajas. En efecto, el número de vértices del grafo de poda de zonas es mayor o igual, siempre, al número de vértices del grafo de poda de recursos y lo mismo ocurre con los arcos de uno y otro de los grafos. Por lo tanto, los algoritmos no mejorarán la complejidad ya que el número de zonas de uso continuado de recursos siempre será mayor o igual que el número de recursos.

No obstante, el grafo de poda de zonas de recursos es de una importancia capital a la hora de desarrollar las técnicas de prevención de bloqueos que se presentarán en el capítulo siguiente. Ello es debido a que a través del grafo de poda de zonas tenemos información de las causas de no vivacidad localizadas en las máquinas de estado implicadas a través del concepto de zona.

Avanzamos someramente la idea. La no vivacidad en redes *SOAR*^{2} está asociada, como redes S^4PR que son, a la existencia de un cerrojo malo a la par que un marcado alcanzable

bajo el cual el cerrojo llega a ser deficitario en marcas. Obviamente, si no existieran cerrojos malos, las redes S^4PR con un marcado inicial admisible serían redes vivas.

La identificación de un cerrojo malo de los que se requieren para la no vivacidad, en términos del grafo de poda de recursos requiere la existencia de un subgrafo fuertemente conexo junto con algunas propiedades adicionales sobre las funciones de etiquetado.

La técnica que se propondrá será una que transforme el grafo de poda de recursos, transformando para ello la red de Petri de partida, de tal forma que no se puede formar ningún subgrafo fuertemente conexo de más de un vértice. En otra palabras, lo que se pretende es transformar la red de Petri de forma que el grafo de poda de recursos resultante sea acíclico y de esta manera tenemos garantizado (condición suficiente) que no va a existir ningún subgrafo fuertemente conexo de más de un vértice, y por tanto, no hay cerrojos malos.

El grafo de poda de zonas nos va a dar la información necesaria para transformar la red de Petri con el objetivo antes apuntado. La idea es eliminar arcos del grafo de poda de recursos, pero para saber como eliminarlos debemos saber donde se originaron, o en otras palabras, cual es la región concreta de la red de Petri (o regiones, porque pueden ser varias regiones a la vez) que de lugar a la aparición de ese arco con cuya eliminación vamos camino de hacer acíclico el grafo de poda de recursos. La respuesta a esa cuestion es que un arco, como hemos visto en la definición 19, del grafo de poda de recursos tiene su origen en un arco del grafo de poda de zonas. Precisamente, la existencia de un arco en el grafo de poda de zonas, entre dos zonas de recursos en las que una poda a la otra, ésta la existencia de un arco entre los recursos asociados a estas dos zonas en el grafo de poda de recursos. Por tanto, la eliminación del arco en el grafo de poda de recursos requerirá una intervención local o localizada en las dos zonas de recursos que están conectadas en el grafo de poda de zonas. Esta idea de localidad en la corrección es un requisito fundamental en las técnicas para forzar la vivacidad en las que no podemos tener un acceso a información global del sistema (imposibilidad ya sea conceptual o práctica). En otras palabras, se trata de desarrollar técnicas de prevención de bloqueos en las que se respete el caracter no centralizado del sistema desarrollando intervenciones que actuen localmente para la prevención del bloqueo.

Resumiendo, amén de permitir construir de forma más eficiente el grafo de poda de recursos de una red $SOAR^2$, el grafo de poda de zonas nos va a permitir interpretar las causas de la aparición de bloqueos y planificar las modificaciones locales de la red de Petri para que no existan estos bloqueos.

3.4. Conclusiones

El comportamiento particular que presentan los sistemas de encaminamiento en la asignación y liberación de los recursos es capturado de forma precisa a través de la redes de Petri de la clase *SOAR*^{2}. La definición de esta clase se ha realizado en este capítulo por medio de elementos estructurales de alto nivel como las zonas de uso continuado de recursos. La utilización de zonas y la definición de los conjuntos máximos de zonas solapadas nos permiten establecer las relaciones de orden entre estas zonas para así modelar formalmente el orden que existe en la asignación y liberación de los recursos en los sistemas tratados.

Adicionalmente, los conjuntos máximos de zonas solapadas son ampliamente utilizados por la relación que se estableció entre estos y los cerrojos mínimos para esta clase de redes. De esta relación es posible determinar cuando un conjunto máximo de zonas solapadas generan un cerrojo mínimo malo que es la causa estructural de la existencia de bloqueos en la red. Esta misma relación entre las zonas y los cerrojos nos permite derivar diversas propiedades estructurales de los cerrojos para esta clase de redes. También se definió lo que denominamos relación de poda entre zonas solapadas y se representó a través del grafo de poda entre zonas. Utilizando una transformación por aglomeración es posible obtener el grafo de poda de recursos que nos permite determinar si existen o no cerrojos mínimos malos en la red. Posteriormente, se utilizará ampliamente estos conceptos para aplicar la política de control que garantice la vivacidad del modelo.

Capítulo 4

Síntesis de redes $SOAR^2$ vivas con la introducción de canales virtuales

Resumen
En este capítulo nos concentraremos en realizar la síntesis de redes $SOAR^2$ libres de bloqueos mediante la utilización de un método no–iterativo que se apoya en técnicas que introducen recursos virtuales para garantizar la vivacidad de la red. A diferencia de los métodos existentes para controlar los bloqueos, este nuevo enfoque no limita la concurrencia de los procesos y permite una solución no centralizada al problema de los bloqueos en el contexto de las redes de interconexión. Para asegurar que la red es viva se verificará que no existan cerrojos malos en la red a través de la aciclicidad del grafo de poda de recursos, que se apoya en la relación de poda entre zonas que se definió previamente. La utilización de recursos virtuales garantiza que es posible producir un grafo de poda de recursos acíclico para las redes $SOAR^2$, siguiendo el método de corrección que presentaremos y que se utilizará para corregir el ejemplo de la red de interconexión con bloqueos que se presentó en la sección 2.2.

4.1. Introducción

Los algoritmos de encaminamiento de datos u objetos a través de una red de interconexión entre centros de procesamiento son esenciales para alcanzar la funcionalidad global perseguida al distribuir las tareas entre diversos centros de procesamiento.

Como se vió en el capítulo 1, el diseño de estos algoritmos no solo depende de la estrategia de encaminamiento elegida *(el algoritmo de encaminamiento propiamente dicho)* sino que depende también de otros factores ligados a la plataforma de ejecución y a los medios físicos utilizados por el transporte propiamente dicho.

En definitiva se trata de una tarea compleja que hace que la aparición de errores de diseño o simplemente mal funcionamiento debido a una sobresimplificación de los detalles del entorno de implementación, sea algo frecuente. Las metodologías de diseño que han aparecido hasta la fecha han tratado de salvar este problema. Para ello han introducido modelos formales que después de un proceso de abstracción del sistema a diseñar han dado lugar a un modelo de cuyo análisis *(mediante técnicas formales o mediante simulación)* se sacan conclusiones acerca del cumplimiento de la especificación de partida. Sin embargo estas metodologías como se han puesto de manifiesto en el capítulo 2 se quedan en este punto. Es decir, el diseñador produce un algoritmo de encaminamiento tomando las decisiones de diseño que tiene en cuenta no solo los objetivos de diseño sino que además tiene en cuenta las restricciones de la implementación y las características relevantes de las tecnologías disponibles.

A partir de ahí mediante técnicas formales o técnicas operacionales determina si su diseño es conforme a la especificación de partida, si la respuesta es afirmativa se puede proceder a la implementación o al menos a la siguiente fase del diseño. El problema aparece cuando la respuesta es negativa y el diseño presentado al sistema de análisis no cumple con las especificaciones de partida ya que ninguna de las metodologías reportadas en la literatura dan una respuesta satisfactoria a qué hacer ante la detección de un error de diseño. Es en este último contexto donde se desenvuelven las aportaciones a realizar en este capítulo.

Más en concreto. En capítulos anteriores hemos presentado una metodología de modelado y análisis de sistemas de encaminamiento de datos y objetos en redes de interconexión entre centros de procesamiento. Incluso para determinados tipos de sistemas hemos caracterizado un conjunto de modelos que poseen características particulares que facilitan en diversos aspectos las tareas de análisis. En este proceso una de las especificaciones de partida que debe satisfacer cualquier diseño a desarrollar es que debe tratarse de un diseño libre de bloqueos. Es decir; cualquier mensaje u objeto que se encuentre en ruta deberá alcanzar su destino asumiendo una propiedad de progreso en el sistema como que si una acción está lista para su ejecución en el futuro acabará ejecutándose. Obsérvese, que esta propiedad no significa que el sistema de encaminamiento completo se bloquea dejando de transportarse mensajes u objetos, el sistema en general puede mantener tráfico entre algunas partes de la red de interconexión

pero algunos mensajes u objetos no pueden alcanzar nunca su destino porque los recursos *(canales de comunicación o vías)* que necesitan están asignados a otros mensajes u objetos que tampoco pueden progresar porque necesitan también recursos en posesión de los mensajes bloqueados.

Por lo tanto, asumiremos que el diseñador ha producido un algoritmo de encaminamiento, se ha obtenido un modelo de la clase *SOAR*^{2} como las presentadas en el capítulo 3 y como consecuencia del análisis se ha determinado que el sistema presenta problemas de bloqueo como los comentados anteriormente. En nuestra terminología de redes de Petri diremos que la red es no viva. Es en este punto en el que se inicia el trabajo de los métodos presentados en este capítulo. Se trata de,

1. Interpretar los resultados del análisis con el ánimo de determinar las causas de la aparición de estas situaciones o estados indeseables que aparecen como consecuencia del bloqueo indefinido de un conjunto de mensajes. Este análisis de las causas ya se ha planteado por ejemplo en [Dally 86] y posteriormente mejorado en [Duato 95b][Duato 95a] cuando aseguran que la causa es que el grafo de dependencias $(extendido)$ no es acíclico.

   Obviamente en este trabajo se buscarán causas que pueden estar relacionadas con puntos concretos del algoritmo de encaminamiento para su posterior modificación, ya que en nuestra opinión, la aciclicidad o no del grafo de dependencias no es una causa operativa que permita al diseñador del algoritmo tomar medidas inmediatas para la subsanación de los problemas sino más bien deja al diseñador desasistido en el proceso de construir algoritmos de encaminamiento bien comportados y lo aboca prácticamente a reiniciar el proceso de diseño con la esperanza de obtener un algoritmo libre de bloqueos en la siguiente iteración.

2. A partir de la identificación de las causas operativas en el algoritmo de encaminamiento o en el modelo de redes de Petri construido para ello, se tratará de corregir o al menos proponer al diseñador una corrección de su algoritmo que prevenga la aparición de estos problemas de bloqueo. En otras palabras, vamos buscando las causas directas en el algoritmo de la aparición de bloqueos y en el capítulo buscar unas técnicas que para cada una de las posibles causas proponga un método "terapéutico" que elimine el problema al erradicar la causa. Es en este sentido en el que las técnicas de corrección que se proponen en este capítulo caen dentro del grupo de técnicas para tratar con bloqueos conocidas como técnicas de prevención de bloqueos.

3. El diseño de la propia técnica de corrección. Una vez identificada la causa del problema y desarrollada la modificación del algoritmo, el método que se

propondrá está basado en la adición de determinados lugares a la red que denominaremos recursos virtuales. Se trata de acogerse al viejo principio, en el contexto del tratamiento de bloqueos debidos a recursos, que establece que en un entorno de recursos ilimitados no pueden aparecer bloqueos debido a una asignación incorrecta de los recursos compartidos *(considérese el caso del uso de ficheros UNIX como recursos compartidos entre un conjunto de procesos concurrentes)*.

Obviamente, no se puede disponer de una cantidad ilimitada de recursos en el contexto de una red de interconexión, por ejemplo en un sistema multicomputador, dado que los canales físicos de comunicación que tiene la red no van a poder modificarse de una forma arbitraria e ilimitada. No obstante, si que se puede tener el margen de maniobra de introducir canales de comunicación virtuales sobre un único canal físico de comunicación siempre que el ancho de banda disponible dentro del canal físico lo permita. Por lo tanto, nos encontramos en un entorno en el que es posible proponer el incremento de recursos compartidos disponibles aunque dentro de unos límites que llevan a tener que hacer un análisis detallado sobre su viabilidad. La razón de llamarlos por tanto, recursos virtuales a aquellos que vamos a incluir para subsanar el problema del bloqueo viene de la analogía con los canales de comunicación en las redes de interconexión.

Obsérvese que esta técnica resulta más complicada *(sino inviable)* cuando nos referimos a sistemas de tipo *AGV* ya que la técnica de adición de recursos virtuales que hemos sugerido en los párrafos anteriores transplantada en este contexto nos llevaría a tener que construir segmentos de vías adicionales *(los recursos en este tipo de sistemas)* que en muchos casos no sería posible o introduciría un coste que sería necesario evaluar. Obviamente el diseño de este recurso virtual requerirá determinar el número de copias disponibles, definir los puntos donde se realizarán las operaciones de adquisición y liberación dentro del algoritmo.

A modo de resumen del planteamiento realizado, este capítulo se entronca dentro de la línea de trabajo agrupada bajo la denominación "técnicas para forzar la vivacidad de redes de Petri", ó "Técnicas de síntesis de modelos vivos mediante la adición de monitores". El repertorio de referencias en esta línea es bastante amplio [Park 00b][Ezpeleta 95][Tricas 03]. No obstante, aunque desde el punto de vista global o finalista en cuanto a los objetos de los que se sirve la técnica: los recursos virtuales; para resolver el problema de bloqueo sea similar a las anteriormente citadas, queremos resaltar ya desde el principio que las técnicas que aquí se proponen son radicalmente distintas, como se verá en las secciones siguientes, por dos motivos bien distintos:

1. Las técnicas que aquí se proponen pueden llegar a modificar las máquinas de estado que caracterizan los procesos que compiten por recursos. Es decir, estas técnicas

pueden llegar a modificar el flujo de control de los algoritmos de encaminamiento que propone el diseñador. Esto no se ha utilizado en lo que nosotros hemos conocido hasta la fecha, nunca antes ya que todas las técnicas conocidas se basan en un principio no escrito, según el cual las máquinas de estado o procesos eran intocables en su estructura y la única posibilidad de corrección pasa por la incorporación de recursos adicionales que en realidad restringen las asignaciones posibles de los recursos originales.

2. Las técnicas que aquí se proponen no entroncan con las técnicas encaminadas a la "prohibición de estados" mediante la introducción de relaciones de exclusión mutua generalizada [Guia 92, Giua 93] *(que es la forma en la que se pueden interpretar los recursos adicionales introducidos para prevenir los estados de bloqueo)*. Por el contrario son técnicas que mantienen en un sentido preciso que se especificará más tarde, los estados de bloqueo de mensajes pero introducen rutas alternativas de estos mensajes gracias a los canales virtuales que salvan la situación de bloqueo. En este sentido, podemos decir que se trata de técnicas de "ampliación de estados " mediante la introducción de recursos virtuales que se usan de una manera más restringida o privada que los originales para salvar los bloqueos *(garantía de recursos extra en caso de aparición de bloqueos)*.

El capítulo se organiza como sigue. En la sección 4.2 se hace una revisión de las técnicas de prevención de bloqueos mediante la adición de monitores, fundamentalmente en redes S^4PR, la clase de referencia de nuestras redes $SOAR^2$ para el modelado de algoritmos de encaminamiento. En la sección 4.3 se presentará mediante un ejemplo las claves del método que se propone en este capítulo. La sección 4.4 describe fundamentalmente el método y la sección 4.5 está dedicada al análisis del método en si mismo, y las propiedades de las soluciones alcanzadas. La sección 4.6 presenta un ejemplo completo no trivial, incluyendo la propagación de la solución alcanzada a nivel de la red al algoritmo que da origen al modelo.

4.2. Revisión de las técnicas para forzar la vivacidad en redes S^4PR y su aplicabilidad en redes $SOAR^2$

Las técnicas para forzar la vivacidad en redes S^4PR están basadas todas ellas en las caracterizaciones diferentes que existen de la vivacidad. Tres ejemplos de caracterizaciones diferentes de la vivacidad se presentan en los teoremas $1, 2$ y 3 del capítulo 1.

La primera de las técnicas está basada en la forma de los marcados que garantizan la existencia de mensajes bloqueados: marcados en los que el conjunto de transiciones sensibilizado por lugares proceso es no nulo y todas ellas están desensibilizadas por medio de algún recurso. La técnica para forzar la vivacidad utilizando esta caracterización se basa en forzar "exclusiones mutuas generalizadas" que prohiban cada uno de estos marcados. En [López-Grao 11] se presenta esta técnica por primera vez y se basa en una técnica iterativa en la que en cada iteración se calcula un marcado de los que caracterizan la no vivacidad según el teorema 1 del capítulo 1. El marcado se calcula a partir de la ecuación de estado y las restricciones lineales adicionales que caracterizan los marcados del teorema 1. Adicionalmente existen algunas heurísticas para seleccionar marcados que posteriormente nos permitan eliminar el marcado calculado además del mayor número de ellos que también sean malos. Heurísticas presentadas en [López-Grao 11] son por ejemplo seleccionar marcados del teorema 1 que contengan el mayor número posible de recursos. Una vez selecionado un marcado se calcula un lugar mediante combinación lineal de las filas de la matriz de incidencia correspondiente a los lugares que están marcados en el marcado calculado. El marcado inicial que se coloca a este lugar es el que le correspondería para un lugar implícito con un conjunto de lugares implicantes como el utilizado para calcularlo, menos una unidad. Con esto se garantiza que el marcado calculado ya no será alcanzable.

Otro conjunto nutrido de técnicas para forzar la vivacidad utilizan la segunda caracterización de la no vivacidad presentada en el capítulo 1. El número de artículos dedicados a este tipo de técnicas es enormemente alto y de hecho a día de hoy son las más populares en el dominio. En este caso la caracterización requiere la existencia de un marcado y un cerrojo de forma que las transiciones habilitadas por proceso están deshabilitadas por los recursos de un cerrojo determinado *(no necesariamente mínimo)*.

Las estrategias para proceder con este método tienen en común restringir el disparo de las transiciones de los cerrojos malos para que no lleguen a un contenido de marcas deficitario que lleve a bloquear mensajes entorno a estos cerrojos. Con esta idea en mente, algunas técnicas calculan inicialmente todos los cerrojos mínimos de la red y entonces proceden a controlar estos cerrojos, cada uno mediante un lugar monitor.

Otra estrategia consiste en la construcción de un sistema de ecuaciones lineales en números enteros que contiene la ecuación de estado de la red, para describir los marcados potencialmente alcanzables, que contiene un conjunto de ecuaciones cuyas soluciones son los cerrojos de la red y que contiene un conjunto de ecuaciones que establecen las condiciones que debe cumplir el marcado que bloquea mensajes y su relación con algún cerrojo deficitario en marcas de la red. La resolución del sistema permite obtener una solución que contiene un cerrojo y un marcado con los que de manera similar a la estrategia basada en la enumeración exhaustiva de los cerrojos, se calcula un monitor que elimina

los marcados y controla el cerrojo para que en ningún caso llegue a ser deficitario.

Todas estas técnicas basadas en los cerrojos de la red, bien sea por la enumeración exhaustiva, bien sea mediante la descripción compacta a través de un sistema de ecuaciones y desigualdades lineales son técnicas de caracter iterativo. Es decir, una vez controlados los cerrojos originales de la red mediante la adición de un lugar monitor, es necesario volver a calcular *(o incorporar a la descripción lineal el nuevo lugar)* los cerrojos de la nueva red dado que pueden aparecer nuevos cerrojos que sea necesario controlar.

Este caracter iterativo de todas estas estrategias se debe a que los marcados que caracterizan la no vivacidad son terminales en el sentido que una vez alcanzados hay un mensaje que está ya bloqueado. No obstante, pueden existir otros marcados que siendo igualmente indeseables no manifiestan la existencia de mensajes completamente bloqueados. Lo que ocurre es que estos marcados son indeseables porque conducen inevitablemente al bloqueo de al menos un mensaje. Paulatinamente, conforme se van controlando cerrojos desaparecen marcados terminales no vivos y van quedando como terminales lo que anteriormente eran de la clase de los que inevitablemente llevaban a bloqueo.

La condición de parada de este método iterativo de corrección es que no existan marcados no vivos, i.e. marcados en los que existan mensajes bloqueados, entre los marcados alcanzables en la red junto con todos los monitores añadidos. Esta es una debilidad importante de todos estos métodos aplicados a las S^4PR. En efecto, además del coste computacional ya sea de calcular todos los cerrojos de la red ya sea de resolver un problema de programación lineal entera debemos multiplicar el coste computacional de los problemas anteriores, por el número de iteraciones que es necesario realizar hasta la completa eliminación de los marcados que inevitablemente conducen a marcados donde hay mensajes bloqueados. Este número de iteraciones puede ser muy grande, en el peor de los casos del orden del tamaño del propio grafo de alcanzabilidad de la red y con problemas cada vez mayores a causa de los lugares monitores incorporados en cada iteración.

Todos los métodos presentados hasta la fecha para forzar la vivacidad de la red de Petri en redes S^4PR se basan en la idea de prohibir todos los estados malos *(estados en los que se bloquea un mensaje o inevitablemente conducen a uno de estos estados)* y a ser posible solo los estados malos. Esta idea subyacente es muy propia o se puede decir que está entroncada con la tradición de la teoría de control y que es conocida como el "problema de los estados prohibidos" en el contexto de la "teoría de control supervisor" introducida en [Ramadge 87]. Es decir, los lugares monitor añadidos son lugares de control que restringen el comportamiento del sistema impidiendo que algunas secuencias sean disparables: aquellas que conducen a marcados o estados malos; que ya nunca serán alcanzables.

Es importante también añadir que las técnicas anteriormente descritas que se basan en lugares monitores no siempre permiten eliminar solamente los marcados malos. A veces, para eliminar marcados malos también es necesario eliminar algunos marcados legales o buenos que de otra manera podrían permitirse sin afectar a la propiedad de ausencia de bloqueos. Su eliminación se debe a la propia técnica para forzar la vivacidad: *la adición de lugares monitores*. Por ello muchos autores han dedicado muchos esfuerzos y trabajos a desarrollar heurísticas que eliminen el menor número posible de estados legales, o como aparece indicado en estos artículos se dedican a desarrollar heurísticas para las políticas de control para que sean los más permisivas posibles. El concepto de permisividad es pues un criterio de calidad a la hora de comparar políticas de control para forzar la vivacidad en las redes S^4PR: Cuantos más marcados legales permita la política de control, mayor es su calidad o bondad.

Finalmente, como una propiedad adicional e intrísecamente relacionada con la estrategia de adición de lugares monitores que eliminan los estados, es que estas políticas de control tienen como efecto la reducción de la concurrencia en el sistema. En otras palabras, las restricciones impuestas a la forma en la que se asignan los recursos en el sistema provoca que haya un menor número de mensajes en tránsito y por lo tanto reduciendo la concurrencia en el sistema. En otras palabras la secuencialización está detrás de la estrategia de corrección lo cual hace que desaparezcan los problemas de solicitud de recursos que están asignados a otros procesos para poder continuar.

Esta secuencialización o reducción de la concurrencia puede ser llevada al límite en redes S^4PR dado que en estas redes el marcado inicial que se coloca pertenece a la clase de los marcados admisibles que tiene como propiedad que todos los t–semiflujos *(que representan el tránsito de un mensaje individual desde su nodo fuente hasta su destino)* son disparables de una manera aislada. Por lo tanto, y dado que la técnica iterativa que se aplica siempre se hace sobre la clase de las S^4PR con marcados iniciales admisibles; se puede llegar a redes S^4PR vivas pero completamente secuenciales: *solo un mensaje en tránsito cada vez.*

Finalmente para ilustrar de una manera informal las técnicas resumidas hasta el momento se presenta a continuación un ejemplo sencillo para forzar la vivacidad en una subclase de las S^4PR: la clase S^3PR. Cada una de las figuras presenta la red y tiene su grafo de alcanzabilidad que muestra todos los marcados alcanzados por la red y las transiciones disparables desde cada uno de estos marcados desde un marcado inicial admisible. Los marcados malos *(aquellos en donde la red está bloqueada o bien los que conducen a estos estados)* están resaltados de los demás estados de la red.

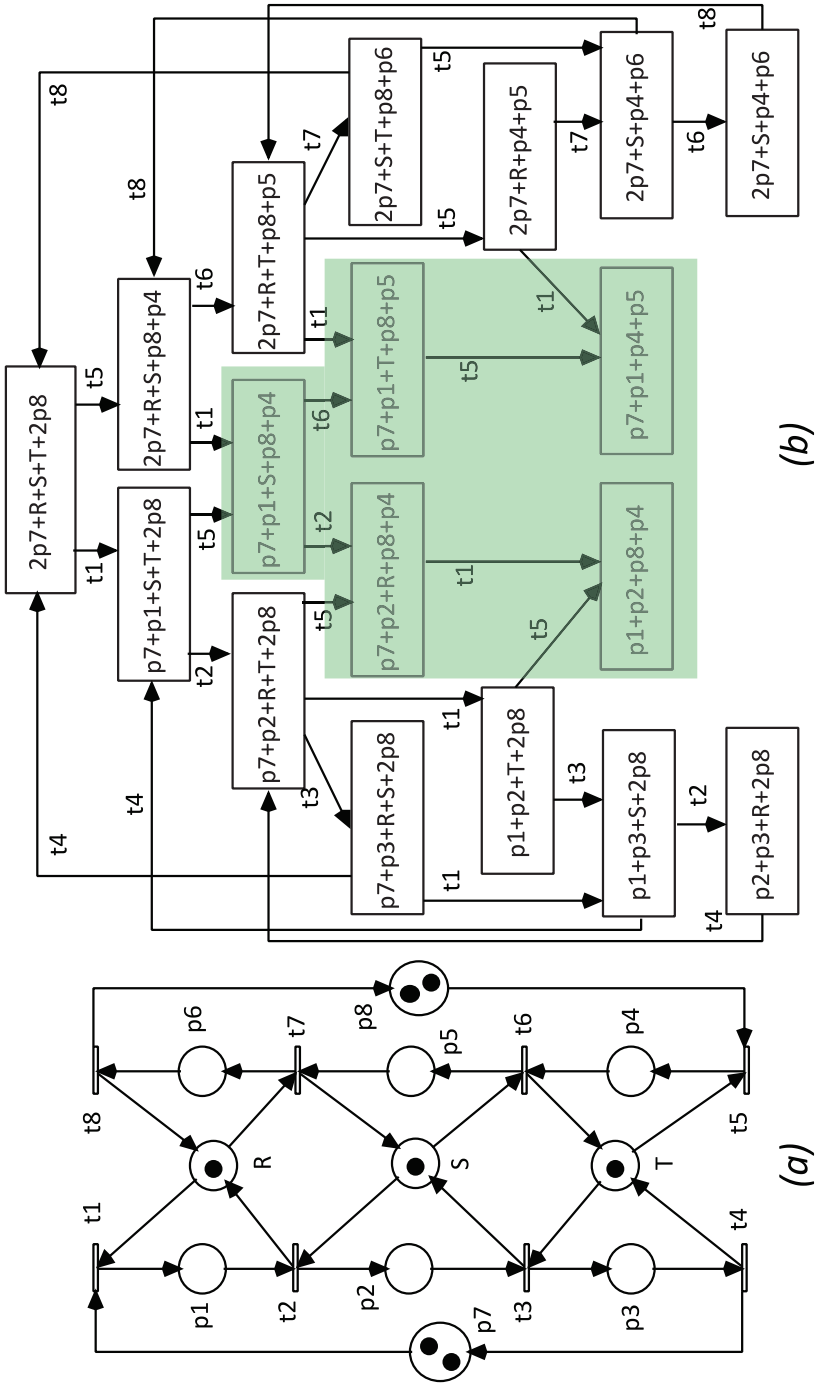

Figura 4.1: Red de Petri S^4PR con su grafo de alcanzabilidad mostrando estados con mensajes bloqueados y estados que inevitablemente llevan a uno de estos estados.

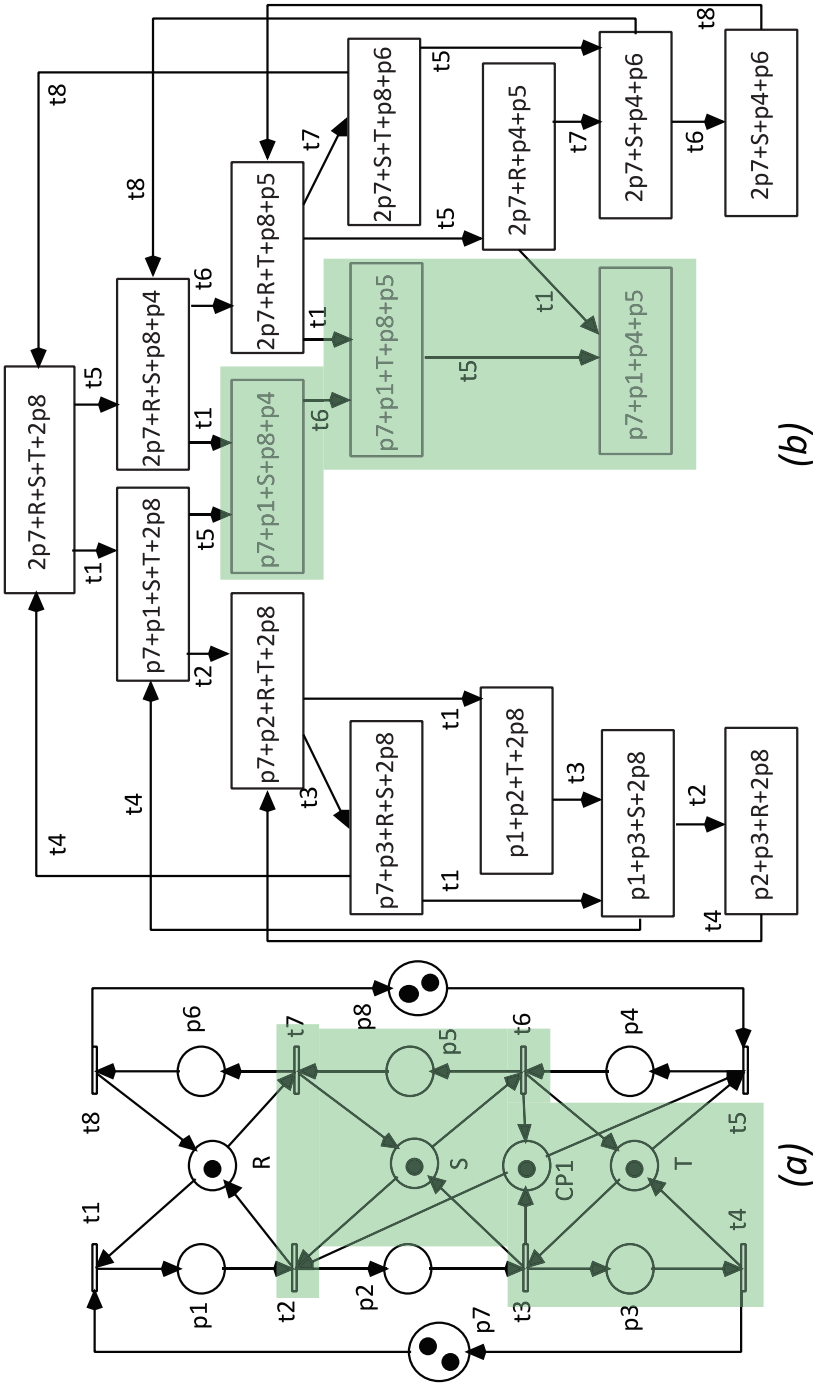

Figura 4.2: Red de Petri S^4PR en la que se ha controlado el cerrojo $\{p_3, p_5, S, T\}$ con el lugar CP_1 y se han eliminado algunos marcados de la red de la figura 4.1.

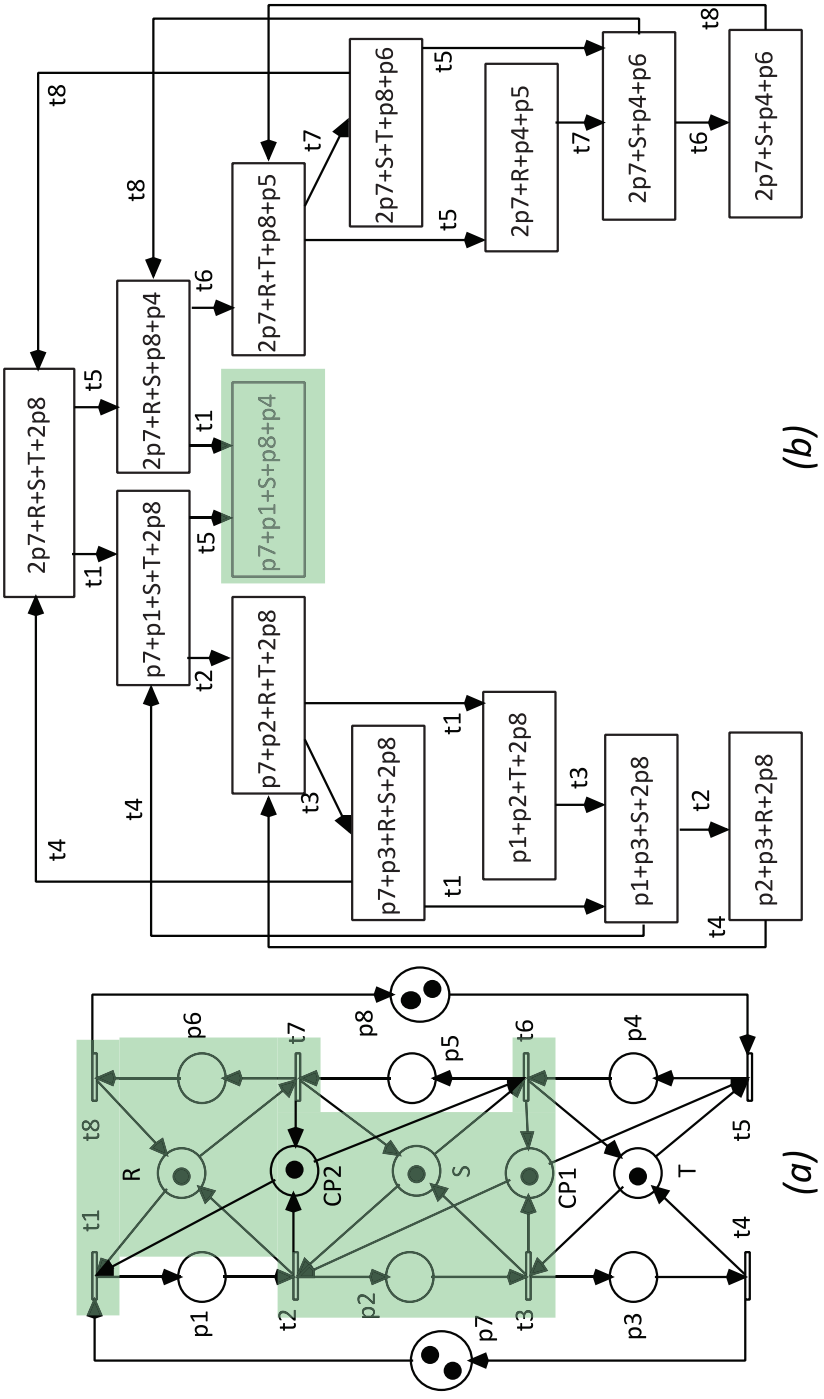

Figura 4.3: Red de Petri S^4PR en la que se ha controlado el cerrojo $\{p_2, p_6, R, S\}$ con el lugar CP_2 y se han eliminado algunos marcados de la red de la figura 4.2.

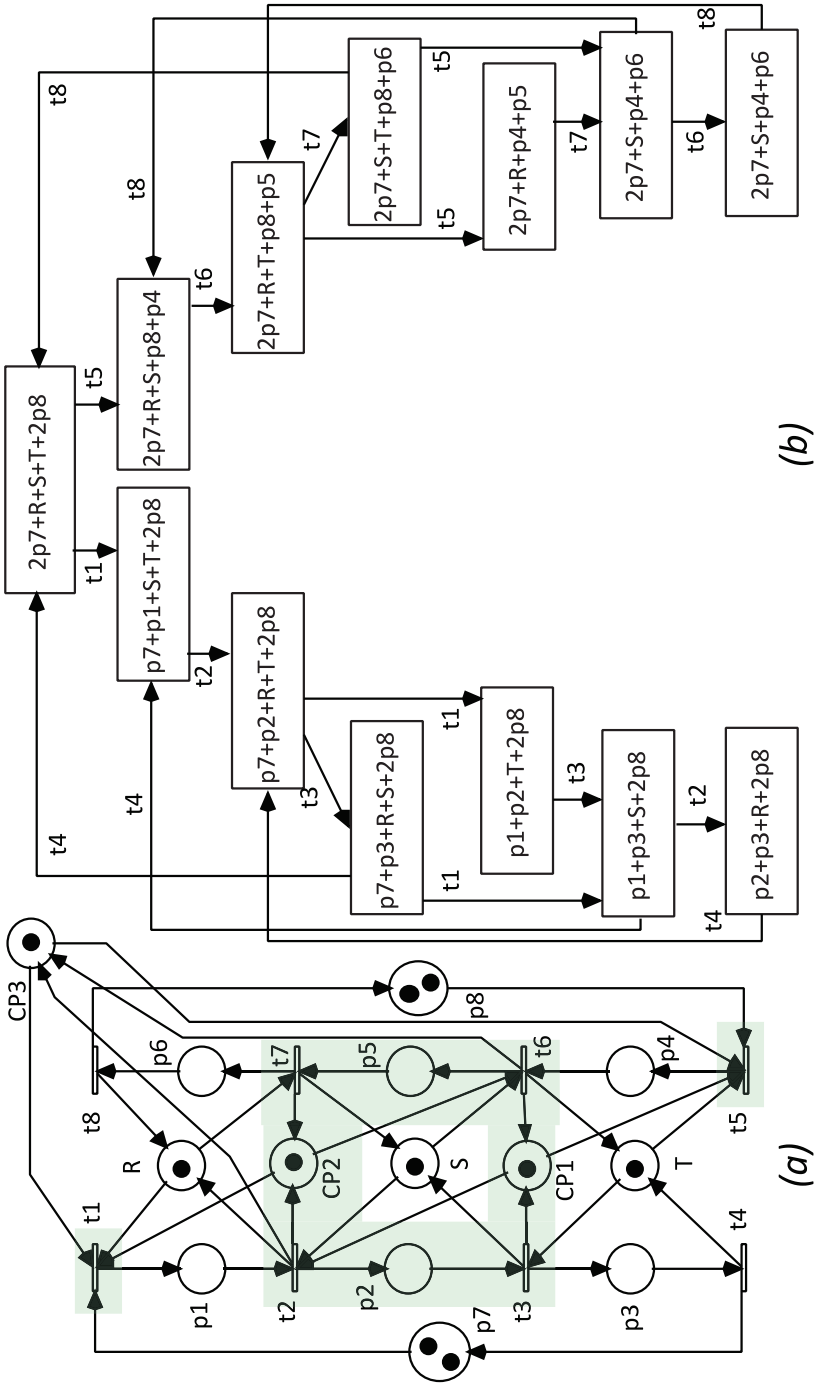

Figura 4.4: Red de Petri S^4PR en la que se ha controlado el cerrojo $\{p_2, p_5, CP_1, CP_2\}$, que no existía en la red original, con el lugar CP_3 resultando una red viva.

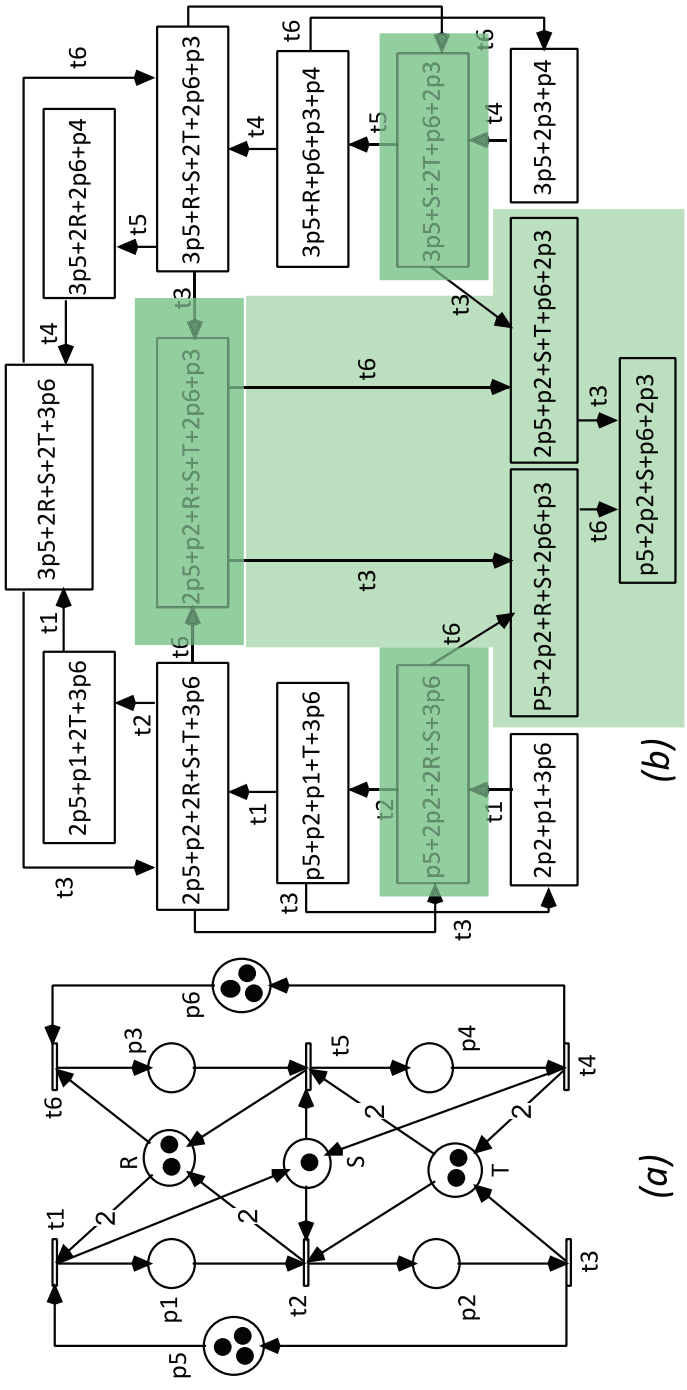

Figura 4.5: Red de Petri S^4PR en la que por eliminar el estado $2p_5 + p_2 + R + S + T + 2p_6 + p_3$ que inevitablemente lleva a un estado con un mensaje bloqueado es necesario eliminar los marcados legales $p_5 + 2p_2 + 2R + S + 3p_6$ y $3p_5 + S + 2T + p_6 + 2p_3$.

En la figura 4.2 y 4.3 se ilustra el control de los cerrojos de la red original de la figura 4.1 que se necesitaban controlar. Esto requiere la adición de los lugares CP_1 y CP_2. El resto de los cerrojos originales no necesitan ser controlados por lo que se da por concluido la primera iteración de la técnica para forzar la vivacidad. Sin embargo, como se puede apreciar en la figura 4.3 todavía queda un marcado malo. Para su eliminación necesitamos empezar la iteración dos calculando los cerrojos que no estuviesen en la red original y que aparecen como consecuencia de la adición de los lugares monitores CP_1 y CP_2. Esto se ilustra en la figura 4.4 donde aparece un cerrojo en la segunda iteración compuesto por $\{p_2, p_5, CP_1, CP_2\}$ y cuyo control nos permite eliminar el último de los marcados malos. Es decir, la red resultante es viva.

Finalmente, en la figura 4.5 se ilustra el caso en el que la eliminación de un marcado malo mediante la adición de lugares monitores no se puede realizar a menos que eliminemos simultáneamente marcados legales [García-Vallés 99] y por lo que la permisividad de una política de control es un criterio de calidad de la misma.

Como apartado final de esta sección queremos hacer algunos comentarios acerca de la aplicabilidad de las técnicas para forzar la vivacidad en redes S^4PR a las redes derivadas del modelado de algoritmos de encaminamiento.

Las técnicas desarrolladas para forzar la vivacidad en redes S^4PR son difícilmente transplantables *(sino imposible)* al contexto de las redes que modelan sistemas de encaminamiento de mensajes o de objetos como a continuación se discute.

Por la propia estrategia de corrección utilizada en las técnicas descritas anteriormente y basadas en lugares monitor que se añaden a la red para garantizar que los cerrojos no lleguen a ser deficitarios en contenido de marcas, la solución resulta inviable. Efectivamente, la utilización de estos monitores no tiene en cuenta que en el contexto en el que estamos no se puede implementar como método de control. El cálculo de estos lugares considera o usa un cerrojo completo que puede afectar a muchas partes *(incluso geográficamente alejadas)* del sistema y estar relacionado con muchos procesos del sistema. En otras palabras, el método de corrección propuesto es un método global que no tiene en cuenta las estrictas condiciones de localidad de los algoritmos de encaminamiento.

Si uno de estos recursos virtuales es asignado en el contexto de un nodo de la red de interconexión, no es posible que dicho recurso virtual sea devuelto en el contexto de otro nodo ya que las operaciones de asignación y liberación son locales a los nodos que comparten los canales y estas operaciones son llevadas a cabo por las copias de los algoritmos de encaminamiento que se ejecutan en cada nodo. Obsérvese que si se trata de implementar uno de estos recursos virtuales con asignación y liberación en nodos diferentes, las decisiones no podrían ser tomadas localmente por los algoritmos de encaminamiento que se ejecutan en los nodos y sería necesario intercambiar información entre nodos acerca de las operaciones de asignación y liberación. Esto conlleva dos

aspectos. El primero es inasumible y se refiere a la aparición de mensajes de control referidos a la política de control implementada para eliminar bloqueos, que deberían convivir con los mensajes que itercambian las aplicaciones que se ejecutan en cada uno de los nodos de la red, reduciendo prestaciones de manera muy significativa, puesto que las decisiones de los algoritmos de encaminamiento estarían condicionados por la velocidad de la propia red de interconexión y el grado de congestión que puede llegar a existir.

El segundo de los problemas se refiere a la necesidad de introducir mecanismos extra para el mantenimiento de la consistencia del estado del recurso virtual en todos los nodos de la red que requerirán conocer su estado para tomar un decisión consistente *(ya que una implementación centralizada a la cual recurren todas las instancias de los algoritmos locales de encaminamiento de nodos implicados no tiene sentido en el contexto de redes de interconexión de multicomputadores)*.

Estos algoritmos de mantenimiento de la consistencia todavía empeoran más la situación ya que por mantener dicha consistencia el número de mensajes de control todavía crece más agravando de una manera extraordinaria los problemas de congestión de la red. Podría pensarse en mantener una red de interconexión alternativa a la de comunicación de datos para intercambiar la información de control. No obstante, esta solución tecnológicamente no es asumible desde el punto de vista de las redes de interconexión existentes y además los problemas de ralentización del encaminamiento no se eliminarían sino que tan solo se aliviarían parcialmente. Otro aspecto importante, derivado de lo anterior, y por lo que las técnicas descritas en esta sección son de difícil/improbable aplicación en este contexto de redes de interconexión para el encaminamiento de mensajes es que las relaciones a las que conduce es a la construcción de algoritmos cuya estructura e información que manejan es fuertemente dependiente del nodo o router donde se alojan. En otras palabras, estas estrategias de corrección conducen a soluciones altamente heterogéneas, lo cual está en franca oposición con la realidad tecnológica actual en la que la homogeneidad de las redes y la simetría de las soluciones son objetivos de diseño perseguidos.

Los comentarios anteriores son suficientes para descartar la aplicabilidad de las técnicas conocidas para forzar la vivacidad de modelos de redes para el encaminamiento de mensajes, pero es que adicionalmente tenemos una dificultad añadida. Ello es debido a la estrategia global subyacente en los propios métodos de corrección que está basada en la secuencialización de los mensajes en tránsito, o en otras palabras la reducción del número de mensajes que pueden estar simultáneamente en tránsito en la red de interconexión. La reducción de la concurrencia es diametralmente contraria a la propia razón de ser de las redes de interconexión en sistemas multicomputadores en los que la maximización del

"throughput" es lo que da sentido a estas soluciones tecnológicas y lo que hace a los multicomputadores una solución competitiva en el contexto de la computación de altas prestaciones. Por ello una solución de corrección que reduzca la concurrencia sería la última solución a adoptar cuando ninguna otra nos hubiera podido permitir el diseño de algoritmos de encaminamiento libres de bloqueos.

Por todo lo anterior en la sección siguiente se establecen los objetivos y requisitos de la política de control que se desarrolla en este capítulo para forzar la vivacidad en redes que modelan el encaminamiento de mensajes en redes de interconexión para multicomputadores.

4.3. Planteamiento y motivación de las técnicas basadas en la introducción de canales de comunicación virtuales

En la figura 4.6 se representa una red de la clase $SOAR^2$. Utilizaremos esta red para ilustrar el procedimiento de corrección de la red para forzar la vivacidad que se formalizará en secciones sucesivas y sobre todo para motivar la necesidad del mismo por la insuficiencia que presentan los métodos conocidos hasta la fecha en el contexto de las S^4PR. En esta red se identifican las siguientes zonas de uso continuado de recursos como se muestra en el cuadro 4.1.

Zonas de recursos
$Z^s_{1,1} = \{p_{10}, p_{11}\}$
$Z^r_{2,1} = \{p_4, p_5, p_6\}$
$Z^r_{1,1} = \{p_{11}, p_{12}\}$
$Z^s_{2,1} = \{p_2, p_3, p_5, p_6, p_7, p_8\}$
$Z^u_{2,1} = \{p_1, p_2, p_3\}$
$Z^t_{2,1} = \{p_3, p_6, p_8, p_9\}$
$Z^t_{3,1} = \{p_{13}, p_{14}\}$
$Z^u_{3,1} = \{p_{14}, p_{15}\}$

Cuadro 4.1: Zonas de recursos de la red de la figura 4.6.

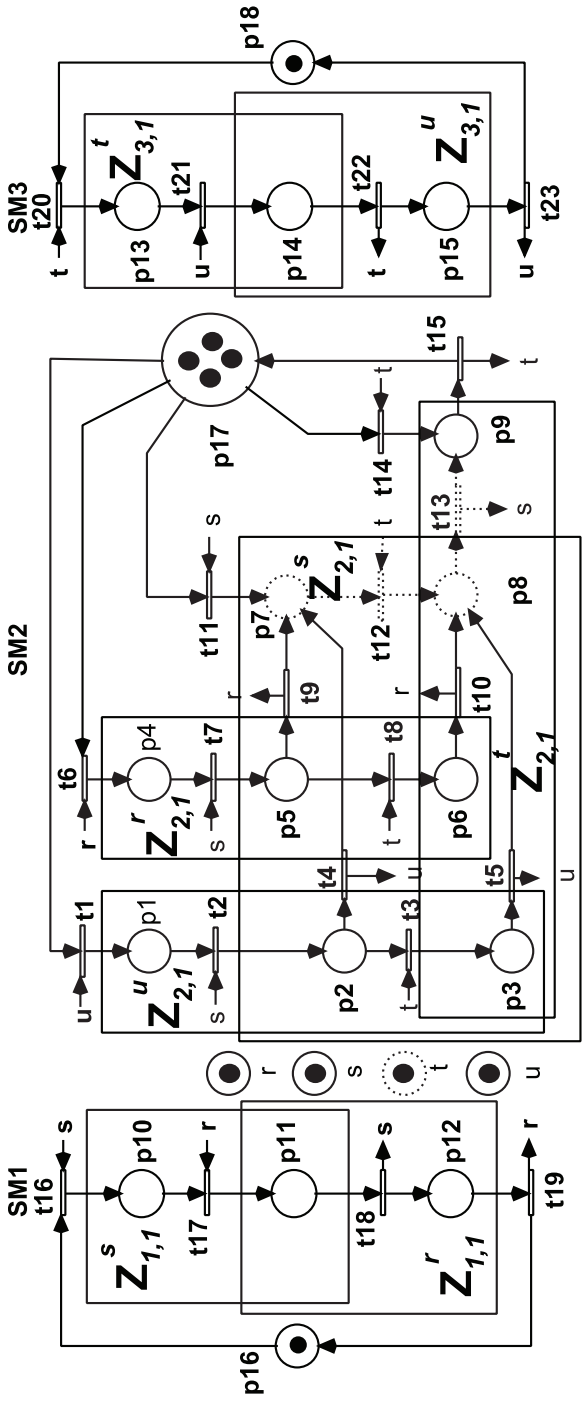

Figura 4.6: Red *SOAR*^{2} en la que para los lugares recurso solo se ha indicado a que transiciones están conectados, sin dibujar el arco para hacer más clara la figura.

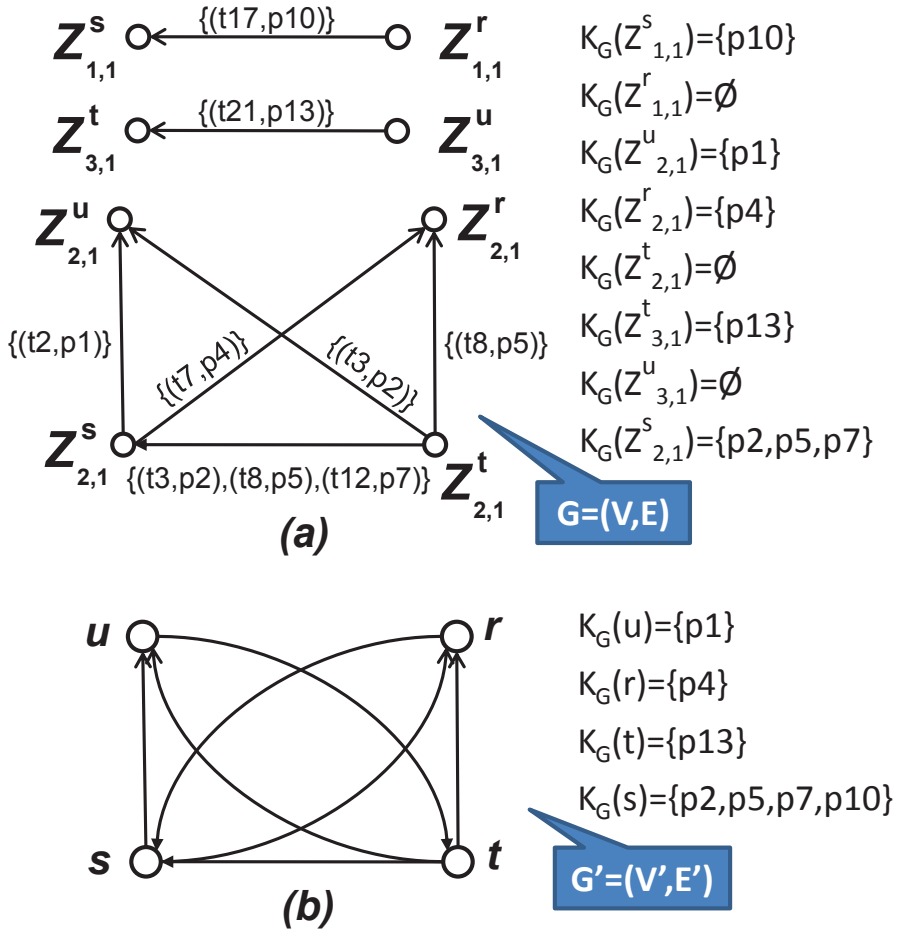

Figura 4.7: Grafo de poda simple de las zonas de P_R y el grafo de poda de los recursos de la red de la figura 4.6.

En la figura 4.7.a se presenta el grafo de poda simple de las zonas de los recursos de P_R, y en la figura 4.7.b el grafo de poda de los cerrojos que contienen un solo recurso. Siendo la red de la clase S^4PR y a la vista del grafo de poda de recursos de la figura 4.7.b, llegaríamos a las siguientes conclusiones.

1. En el grafo de poda existen diversos subgrafos fuertemente conexos que potencialmente pueden dar lugar a cerrojos malos.

2. Por ser una red S^4PR podemos aplicar los métodos para forzar la vivacidad consistentes en controlar los cerrojos mediante un lugar monitor. Por ejemplo, el cerrojo $D = \{s, r, p_2, p_3, p_5, p_6, p_7, p_8, p_{11}, p_{12}\}$.

3. Obsérvese que el lugar de control del cerrojo D que se obtiene como la combinación lineal de los lugares que pertenecen al cerrojo, necesariamente tendrá arcos de entrada y de salida a las dos máquinas de estado $SM1$ y $SM2$. Obviamente este lugar entrará a la transición t_{16} donde también entra el recurso s, y entrará a la transición t_6 donde también entra el recurso r. Con lo cual al menos, después de agregar este lugar de control la red deja de pertenecer a la clase $SOAR^2$ y por lo tanto resulta difícil la implementación en el algoritmo de encaminamiento, la asignación simultánea de dos recursos en la misma transición.

4. Adicionalmente, el lugar comentado anteriormente está relacionado con zonas de dos máquinas de estado diferentes con lo que no hay garantías de que las decisiones sobre su asignación o sobre su liberación puedan ser tomados localmente.

El ejemplo anterior evidencia los comentarios del final de la sección anterior en los que señalábamos que aunque consigamos hacer viva la red de Petri que modela el algoritmo de encaminamiento proporcionado por el diseñador, no se puede en general, llevar desde el modelo al sistema real la solución alcanzada. Esto es lo que hemos pretendido ilustrar con el lugar de control asociado al cerrojo D anterior de la red de la figura 4.6.

La estrategia que se pretende desarrollar en este capítulo parte de un principio diferente. Efectivamente, asumiremos o consideraremos inicialmente la caracterización de la no vivacidad expuesta en el teorema 2 del capítulo 1 basada en los cerrojos de la red *(las redes SOAR^2 son redes S^4PR y por lo tanto podemos aplicar el teorema 2)*. No obstante, en lugar de considerar directamente los cerrojos de la red y proceder a su control procederemos de una manera diferente.

Para ello nos apoyaremos en primer lugar en el grafo de poda de los recursos de la red. Este grafo se utiliza como generador de todos los cerrojos de una red S^4PR [Cano 10]. Ello es debido a que todo cerrojo mínimo conteniendo al menos un recurso tiene asociado un subgrafo fuertemente conexo de este grafo de poda cumpliendo algunas propiedades adicionales las funciones de etiquetado del grafo.

Teniendo en cuenta este hecho, vamos a estudiar este grafo con el ánimo de seleccionar un conjunto de arcos de forma que su eliminación garantice que ya no va a existir ningún subgrafo fuertemente conexo del grafo de poda, es decir, no va a existir ningún cerrojo malo. Por lo tanto nuestra estrategia consiste no en controlar los cerrojos de la red, sino en modificar la red para que no existan cerrojos malos. La forma de hacerlo será conseguir que el grafo de poda de los recursos de la red sea acíclico eliminando algunos arcos del grafo. En el grafo de poda de los recursos de la figura 4.7.b se puede encontrar subgrafos fuertemente conexos diferentes.

1. El propio grafo de poda.

2. El subgrafo formado por los vértices r y s y todos los arcos que los conectan.

3. El subgrafo formado por los vértices u y t.

4. El subgrafo fuertemente conexo formado por los vértices u,t y s.

Esto quiere decir que potencialmente deberíamos considerar el control de cuatro cerrojos. Sin embargo podemos considerar que si eliminamos tan solo dos arcos del grafo de poda, como por ejemplo el arco (s,r) y el arco (u,t) entonces el grafo de poda de los recursos es acíclico y por lo tanto la red no contiene cerrojos malos con más de un recurso. En otras palabras, si somos capaces de hacer esto hemos obtenido una red de Petri viva ya que está admisiblemente marcada y no existen cerrojos malos que son necesarios para la existencia de mensajes bloqueados *(teorema 2 del capítulo 1)*.

Definida la estrategia, a continuación nos centramos en presentar la técnica de eliminación de un arco del grafo de poda de recursos mediante transformación de la red de Petri. Para ello en primer lugar hay que identificar en la red de Petri que significa la existencia de un arco en el grafo de poda de recursos. Como se pudo apreciar en la figura 4.7, existe un arco de un recurso r a un recurso x si existen al menos dos zonas solapadas Z^r y Z^x tales que la zona Z^x precede a la zona Z^r según la relación de precedencia $<$ definida para la clase de redes *SOAR*^{2}. En este caso, obviamente en la zona que precede a la otra, algunos de los lugares dejan de ser esenciales para el cerrojo. Por lo tanto, la eliminación de un arco en el grafo de poda pasará por tratar las parejas de zonas solapadas que sustancian este arco del grafo de poda de recursos. En nuestro ejemplo para los dos arcos que estamos tratando de eliminar.

1. El arco (s,r) existe porque existen las dos zonas solapadas $Z^s_{2,1}$ y $Z^r_{2,1}$ *(solapadas en los lugares p_5 y p_6)*; y además la zona $Z^r_{2,1}$ precede a la zona $Z^s_{2,1}$, es decir $Z^r_{2,1}<Z^s_{2,1}$, y por lo tanto la zona $Z^s_{2,1}$ hace no esencial el lugar p_6 de la zona $Z^r_{2,1}$.

2. El arco (u,t) existe porque existen las dos zonas solapadas $Z^t_{3,1}$ y $Z^u_{3,1}$ *(que se solapan en el lugar p_{14})*; y además la zona $Z^t_{3,1}$, precede a la zona $Z^u_{3,1}$, es decir, $Z^t_{3,1} < Z^u_{3,1}$ y por lo tanto la zona $Z^u_{3,1}$ hace no esencial el lugar p_{13} de la zona $Z^t_{3,1}$.

En este caso solo tenemos que tratar con una pareja de zonas por cada uno de los arcos que vamos a eliminar. Si existiesen más parejas de zonas para alguno de los arcos del grafo de poda de recursos, trataríamos cada pareja como se hará para una a continuación.

Consideramos en primer lugar la pareja de zonas $Z^t_{3,1}$ y $Z^u_{3,1}$ que dan lugar al arco (u,t) del grafo de poda de recursos. Para determinar la transformación a realizar observamos que

al ser dos zonas solapadas y ordenadas ($Z^t_{3,1} < Z^u_{3,1}$) en primer lugar se trata de operaciones locales a un nodo u otro de la red de interconexión dado que se tratan de zonas solapadas y ordenadas y por lo tanto las decisiones de asignación son locales al mismo nodo. Por lo tanto las modificaciones que hagamos en estas zonas se manifestarán exclusivamente en la copia del algoritmo en el nodo correpondiente. Para eliminar el arco basta con eliminar el recurso de la zona mayor, en nuestro caso $Z^u_{3,1}$, e introducir un nuevo recurso u' que se asignará y se liberará en los mismos puntos en los que se hacía por el recurso u. Esta operación de adición del recurso virtual u' tiene los siguientes efectos.

1. Las necesidades de recursos *(canales de comunicación)* que tuviese un mensaje que atraviese la pareja de zonas $Z^t_{3,1}$ y $Z^u_{3,1}$ son satisfechas de igual manera en la red con el nuevo recurso virtual u' que sustituye a u en la zona $Z^u_{3,1}$. Efectivamente, para toda secuencia que contenga las transiciones t_{21} y t_{23} en una y otra red la única diferencia es que en lugar de asignar/liberar u se hará con u'.

2. La zona $Z^u_{3,1}$ desaparece en la nueva red ya que a t_{21} entra u' y de t_{23} sale u'.

3. En el grafo de poda de zonas de la figura 4.7, el nodo $Z^u_{3,1}$ se transforma en el nodo $Z^{u'}_{3,1}$, pero las funciones de etiquetado mantienen los valores que teníamos previamente, salvo el cambio de nombre de la zona $Z^u_{3,1}$.

4. En el grafo de poda de recursos el arco (u,t) desaparece dado que al realizar la aglomeración de los vértices que representan zonas del mismo recurso ya no tenemos $Z^u_{3,1}$, y por lo tanto el arco (u,t) no puede aparecer.

5. En el grafo de poda de recursos aparece un nuevo vértice debido al nuevo recurso u' para el que existe una única zona, la nueva $Z^u_{3,1}$ y un nuevo arco (u',t) debido al arco existente entre las zonas $Z^t_{3,1}$ y $Z^{u'}_{3,1}$ del grafo de poda de zonas.

6. La red resultante con esta transformación sigue siendo de la misma clase *SOAR*^{2}. Esto es debido a que cada zona, tal y como se ha definido, con su recurso es un p–semiflujo de la red que contiene la máquina de estado sobre la que se define la zona. Por lo tanto, el nuevo recurso pertenece a un p–semiflujo mínimo binario que lo contiene a él y la zona que ha servido para su definición. Además, la nueva zona no altera la relaciones existentes previamente entre zonas ya que tan solo ha cambiado de nombre al cambiar el nombre del recurso que sirve para definirla.

7. Vamos camino de conseguir la vivacidad de la red debido a que hemos eliminado un arco del grafo de poda de recursos que era esencial para formar un circuito en el grafo y por lo tanto se han eliminado subgrafos fuertemente conexos que daban lugar a cerrojos en la red

Para eliminar el arco (s,r) podemos proceder de forma análoga a como lo hemos hecho para el caso del arco (u,t). Ahora tenemos que trabajar con la pareja de zonas $Z^s_{2,1}$ y $Z^r_{2,1}$, sabiendo que $Z^r_{2,1} < Z^s_{2,1}$ y por lo tanto hay que eliminar el recurso s de la zona $Z^s_{2,1}$ y añadir un nuevo recurso virtual s' que debería estar conectado a las mismas transiciones a las que estaba conectado el recurso s en la zona $Z^s_{2,1}$.

Sin embargo esto es erróneo ya que a diferencia del primer caso existen transiciones de asignación del recurso s, como la t_2 o la t_{11}, a la entrada de la zona $Z^s_{2,1}$ que no están precedidas por una transición de asignación del recurso r en la zona $Z^r_{2,1}$. Por lo tanto, tan solo hay que cambiar en aquellas transiciones que pertenecen a la zona $Z^r_{2,1}$ y que son de asignación de s, y el cambio consistirá en asignar el recurso s'. En otras palabras, la condición de cambio del recurso s al recurso s', se realizará $\forall t \in Z^s_{2,1} \cap Z^r_{2,1} \cap s^\bullet$; en palabras, transiciones dentro de la zona que precede donde se asigna el recurso de la zona precedida. En este caso el cambio de s a s' solo hay que hacerlo a la entrada de la transición t_7.

En cuanto a las transiciones de liberación de s, en este caso tenemos el problema que existirán caminos a esta transición de liberación t_{13} que se originan en transiciones que asignan el recurso s, pero también tenemos caminos que se originan en transiciones que asignan el recurso s'. Dado que la transición t_{13} solo puede liberar s ó s' pero no ambas *(en otras palabras porque nunca sabría qué tendría que liberar, si s ó s')* hay que proceder a distinguir cual es el recurso que hay que liberar en función de cual fue el recurso que se asignó. Para realizar esta distinción *(memorizar qué se asignó)* se replicarán todos los lugares proceso y transiciones correspondientes a caminos que arranquen de las transiciones $t \in (Z^r_{2,1} \cap Z^s_{2,1})^\bullet$ hasta el lugar $((Z^s_{2,1})^\bullet \cap {}^\bullet s)^\bullet \cap P_S$. La transición de inicio y el lugar final de estos caminos no se replicará. Las transiciones replicadas mantendrán conexiones a los mismos recursos a las que estaban conectadas las anteriores. Adicionalmente, se eliminarán de la red los arcos de salida de las transiciones $t \in (Z^r_{2,1} \cap Z^s_{2,1})^\bullet$ a los lugares proceso originales.

Para el ejemplo que venimos desarrollando se requiere replicar los lugarse p_7 y p_8 y las transiciones t_{12} y t_{13}.

Obsérvese que la transición $t_{12'}$ también tiene como entrada el recurso t como ocurriría para la transición t_{12}. Ahora es posible distinguir cual es el recurso que hay que devolver ya que los caminos que arrancaban con la asignación de s' acaban en la transición $t_{13'}$ que liberará el recurso s', y los otros caminos acaban en t_{13} que liberará s. En el caso del arco (u,t) esto no ha sido necesario debido a que todas las transiciones de asignación del recurso u estaban en el interior de la zona de t, que precedía a la zona de u y por lo tanto no era necesario distinguir cual es el recurso asignado.

El resultado final de la transformación de la red se presenta en la figura 4.8. Así mismo en la figura 4.9 se presentan los grafos de poda de zonas y de recursos para la red de la figura 4.8.

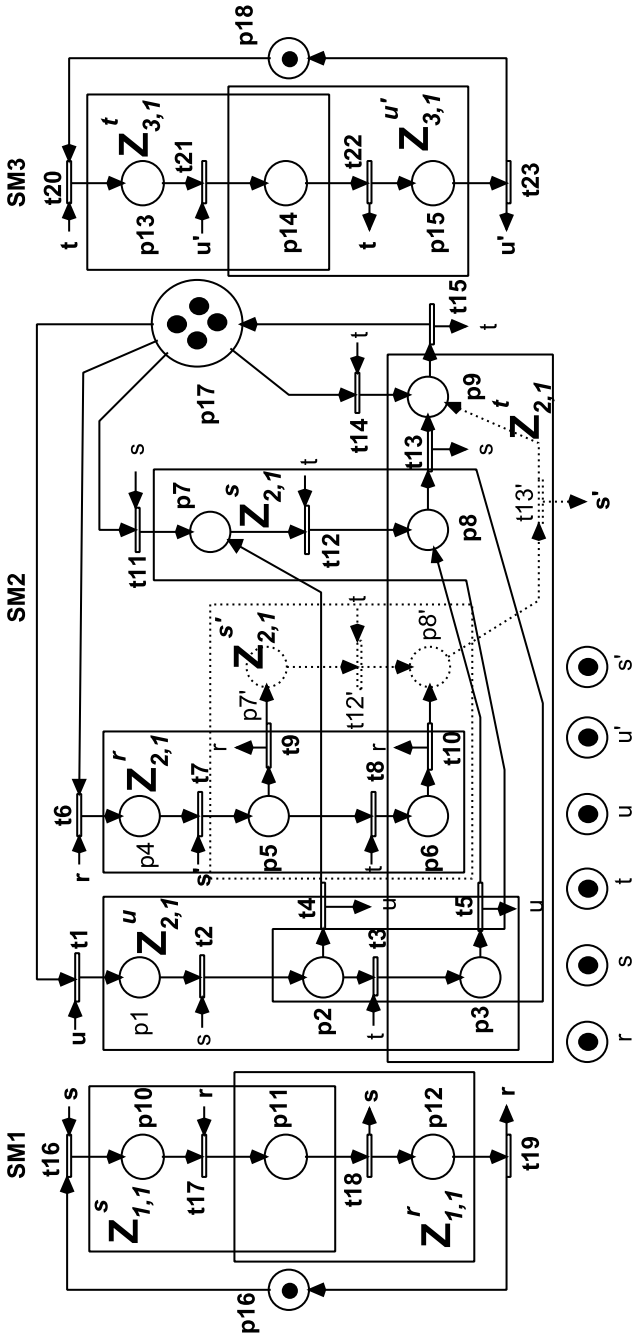

Figura 4.8: Red de Petri de la figura 4.7 transformada para conseguir que sea viva.

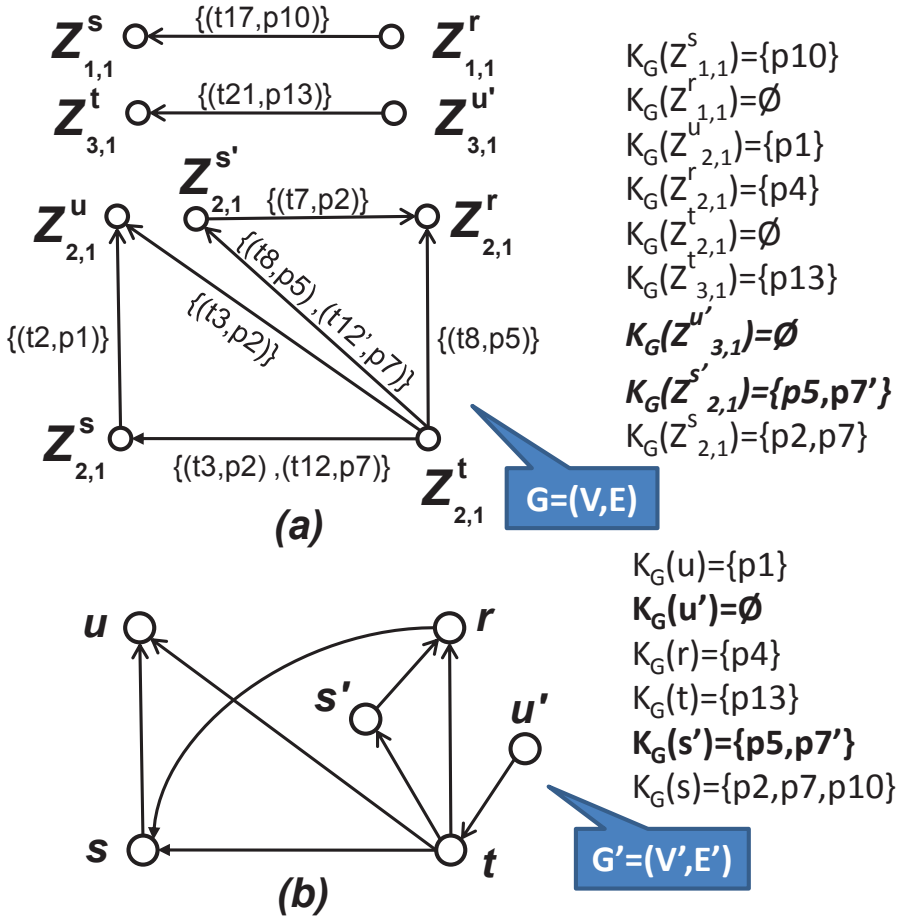

Figura 4.9: Grafo de poda simple y grafo de poda de recursos de la red de la figura 4.8.

Para terminar esta sección de introducción intuitiva del método de corrección mediante la eliminación de arcos del grafo de poda de recursos por introducción de recursos virtuales en los pares de zonas que den origen al arco, vamos a presentar un fenómeno que hace que haya que completar el procedimiento básico presentado en el ejemplo anterior.

Este fenómeno aparece cuando al tratar un par de zonas (Z^r, Z^s), la zona Z^r no es el máximo, según la relación de orden basada en la relación de precedencia, del conjunto o conjuntos máximos de zonas solapadas a las que pertenece Z^r. En efecto, en este caso cuando se introduce el recurso virtual r', el grafo de poda de zonas queda inalterado en estructura y funciones y tan solo hay que cambiar el nombre Z^r por el nombre $Z^{r'}$ allí donde aparezca. Si Z^r era el máximo de todos los conjuntos máximos de zonas solapadas a las que pertenecía no tiene ningún predecesor en el grafo de poda de zonas, y al ser ahora una

zona de un nuevo recurso en el grafo de poda de recursos aparecerá un nuevo nodo r' pero ahora no tendría ningún predecesor, así que a través de r' en el grafo de poda de recursos no se podrá cerrar ningún circuito. Además entre el nodo r y el nodo r' no habrá ningún arco, todos los descendientes de r han sido heredados por r' (por que se ha actuado de la misma manera para todos los pares que pudieran haber dado lugar al arco (r,s)). Por lo tanto, se ha conseguido el objetivo de eliminar el arco (r,s) en el grafo de poda de recursos, por introducción de un vértice r' que no tiene predecesores. Sin embargo, si Z^r no es el máximo en todos los conjuntos máximos de zonas solapadas entonces, en el grafo de poda de recursos el nuevo recurso virtual puede tener arcos de entrada que lo conectan a otros vértices y por lo tanto subsistir algún circuito en el grafo de poda de recursos.

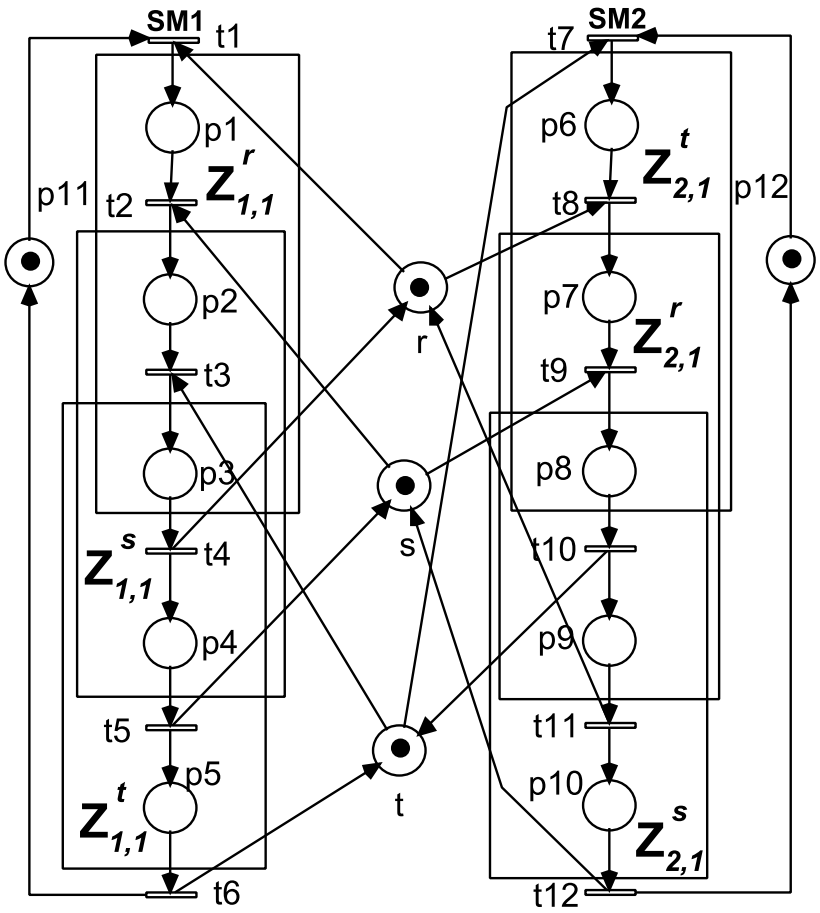

Figura 4.10: Red *SOAR*^{2}.

Para ilustrar esta situación considérese el ejemplo de red *SOAR*^{2} mostrado en la figura 4.10. El grafo de poda de zonas se muestra en la figua 4.11 que pone de manifiesto la existencia de dos conjuntos máximos de zonas solapadas. En la figura 4.12 se muestra el grafo de poda de recursos que contiene exactamente un circuito. Para hacerlo acíclico elegimos como arco víctima el arco (r,t) en el grafo de la figura 4.12. Este arco tiene su origen en el par de zonas $(Z^r_{2,1}, Z^t_{2,1})$. Obsérvese que la zona $Z^r_{2,1}$ no es el máximo del conjunto máximo de zonas solapadas al que pertenece y en el que se verifica la siguiente relación de orden $Z^t_{2,1} < Z^r_{2,1} < Z^s_{2,1}$.

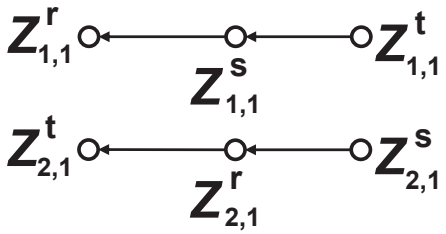

Figura 4.11: Grafo de poda de zonas de la red de la figura 4.10.

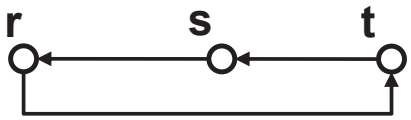

Figura 4.12: Grafo de poda de recursos de la red de la figura 4.10.

Al introducir el recurso virtual r', la zona $Z^r_{2,1}$ desaparece y aparece una nueva zona $Z^{r'}_{2,1}$ exactamente igual y con las mismas relaciones. Esto puede observarse en el grafo de poda de zonas de la red con el recurso r' en la figura 4.13. Al generar ahora el grafo de poda de recursos observamos que el nuevo vértice r', que corresponde al nuevo recurso virtual r', tiene un arco de entrada que tiene su origen precisamente en el arco de entrada que tiene la zona $Z^{r'}_{2,1}$ en el grafo de poda de zonas de la red con r'. Obviamente, si $Z^r_{2,1}$ hubiera sido el máximo de su conjunto máximo de zonas solapadas esto no hubiera pasado. La propiedad importante a retener es que los únicos arcos de entrada que puede tener el nuevo recurso virtual sólo son debidos a los arcos de entrada que tiene la zona nueva en el grafo de poda de zonas.

El efecto negativo es que no hemos conseguido eliminar el circuito. No obstante, tenemos un buen candidato para lograr el objetivo y este es precisamente el arco de

entrada del nuevo recurso virtual r' que tiene su origen en el arco de entrada en el grafo de poda de zonas donde la zona $Z^r_{2,1}$ no era máximo. Repetiremos el proceso cuantas veces sea necesario, pero por el procedimiento que estamos aplicando este termina en cuanto alcancemos el máximo del conjunto máximo de zonas solapadas al que pertenece $Z^r_{2,1}$. Esta condición de terminación siempre existe dado que el grafo de poda de zonas es acíclico como se demuestra en la sección 4.5.

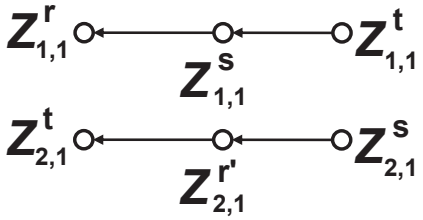

Figura 4.13: Grafo de poda de zonas después de haber intervenido en el par de zonas $(Z^r_{2,1}, Z^t_{2,1})$ y haber añadido el recurso virtual r'.

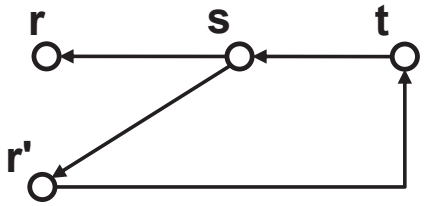

Figura 4.14: Grafo de poda de recursos después de haber introducido el recurso virtual r' para el par de zonas $(Z^r_{2,1}, Z^t_{2,1})$.

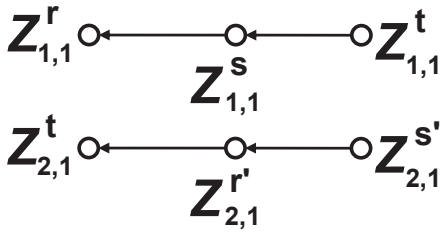

Figura 4.15: Grafo de poda de zonas después de haber intervenido en el par de zonas $(Z^{r'}_{2,1}, Z^s_{2,1})$ y haber añadido el recurso virtual s'.

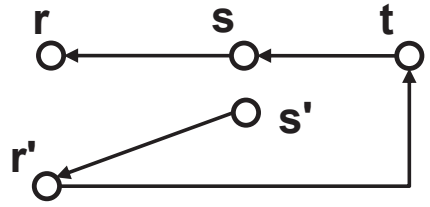

Figura 4.16: Grafo de poda de recursos después de haber introducido el recurso virtual s' para el par $(Z^{r'}_{2,1}, Z^{s}_{2,1})$.

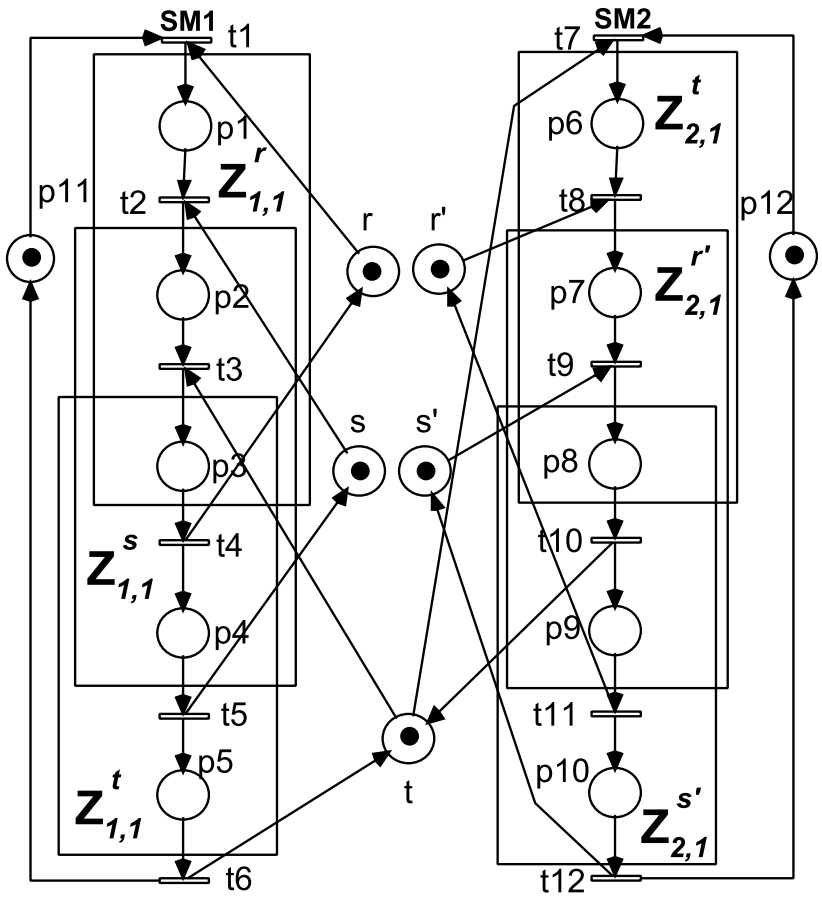

Figura 4.17: Red de Petri corregida desde la red de Petri de la figura 4.10.

Efectivamente, si ahora se considera el arco de grafo de poda de recursos de la figura 4.14, (s, r') que tiene su origen en el par de zonas $(Z^s_{2,1}, Z^{r'}_{2,1})$ y procedemos de la misma manera, entonces se obtienen los siguientes resultados:

a) La zona $Z^s_{2,1}$ se renombra a la zona $Z^{s'}_{2,1}$ por introducción del recurso virtual s'. El nuevo grafo de poda de zonas se presenta en la figura 4.15.

b) El grafo de poda de recursos será ahora acíclico ya que la zona $Z^s_{2,1}$ era el máximo del conjunto máximo de zonas solapadas al que pertenecía y por lo tanto el vértice no tendrá arcos de entrada que permitirían cerrar circuitos. Este grafo acíclico obtenido se presenta en la figura 4.16.

Después de esta última corrección la red de Petri obtenida que se muestra en la figura 4.17 es viva porque su grafo de poda de recursos es acíclico y la red está inicialmente marcada con un marcado inicial admisible. Por lo tanto, al procedimiento que presentamos al inicio de esta sección es necesario completarlo con una última acción consistente en determinar si la zona en la que se ha introducido el recurso virtual tiene algún arco de entrada en el grafo de poda de zonas, entonces seleccionar todos los pares de zonas que tengan como segunda componente la nueva zona y como primera componente el vértice origen del arco incidente como nuevos pares en los que hay que intervenir con el mismo procedimiento que se describió al comienzo de esta sección.

Obviamente, el fenómeno descrito para el primer tipo de intervención sobre pares de zonas también se presenta en el caso del segundo tipo de intervención descrito en esta sección (en el que teníamos duplicación de nodos) y la actuación será del mismo tipo.

4.4. Descripción del método

En esta sección vamos a proceder a formalizar en toda su extensión el método descrito mediante un ejemplo en la sección anterior. La presentación se realizará a través de las distintas fases a cubrir hasta conseguir una red viva.

El contexto en el que se desarrolla el método es dentro de la clase de redes de Petri *SOAR*^{2} definida en el capítulo 3 y utilizadas para modelar algoritmos de encaminamiento mínimos con una estrategia tipo "wormhole". En este contexto el problema a resolver es corregir algoritmos de encaminamiento que presentan problemas debido a mensajes u objetos bloqueados para siempre y sin posibilidad de alcanzar su destino, interviniendo a nivel del modelo de red de Petri construido y trasladando la corrección de la red a nivel de algoritmo. Es decir, se trata de completar el ciclo de diseño de este tipo de algoritmos propuesto en el capítulo 2 y basado en la utilización intensiva de redes de Petri como modelo formal de los sistemas a diseñar. El desarrollo de esta etapa del ciclo de diseño,

como ya se apuntó en el capítulo 2, es una novedad metodológica importante en cuanto que las metodologías conocidas hasta la fecha no cubren esta parte de corrección de los problemas de un diseño propuesto por el diseñador.

La base del método de corrección que se propone consiste en el caso que tengamos estados de bloqueos, en transformar la red de Petri de forma tal que el grafo de poda de recursos de la red transformada resulte un grafo acíclico. Por lo tanto, y de acuerdo con la caracterización de los cerrojos mínimos de una red S^4PR (las redes $SOAR^2$ son una subclase de las S^4PR) sobre el grafo de poda de recursos, no existirá ningún cerrojo mínimo que contenga más de un recurso ya que no contendrá ningún subgrafo fuertemente conexo con más de un vértice. Es la base del método porque de esta forma forzamos a que la red sea viva ya que la red posee un marcado inicial admisible y no existe ningún cerrojo malo (cerrojos mínimos conteniendo más de un recurso).

Para llegar a este objetivo final se seguirá un procedimiento desglosado en una serie de fases a cada una de las cuales se dedica una subsección en lo que sigue. Estas fases se enuncian a continuación:

*Paso*1 : Construcción del grafo de poda de zonas de uso continuado de recursos de la red $SOAR^2$.

*Paso*2 : Construcción del grafo de poda de recursos de la red $SOAR^2$.

*Paso*3 : Determinación de un conjunto de arcos cuya eliminación hace acíclico el grafo de poda de recursos.

*Paso*4 : Eliminación de arcos del grafo de poda de recursos por adición de recursos virtuales entre las zonas solapadas que generan estos arcos.

Con el paso 4 se alcanzará un modelo de red de Petri perteneciente a la clase $SOAR^2$ que será vivo para el marcado inicial admisible calculado. La última tarea a realizar será la traslación de la solución alcanzada para forzar la vivacidad del modelo de red de Petri a el diseño concreto suministrado por el diseñador y en el lenguaje propio del diseño. Dado que la corrección introducida es local y la obtención de la red de Petri ha seguido unas pautas constructivas muy detalladas en el capítulo 2 es posible realizar esta traslación aunque es fuertemente dependiente de los lenguajes de descripción utilizados para representar los diseños suministrados. Es por ello, que en lugar de desarrollar un algoritmo concreto para un lenguaje de descripción de los diseños concretos, hemos preferido presentar las pautas de esta traslación a través del ejemplo que se desarrolla en la sección 4.6.

A continuación se formalizan cada uno de los pasos listados anteriormente.

4.4.1. Paso 1: Construcción del grafo de poda de zonas de uso continuado de recursos

La entrada al proceso de corrección es un modelo de redes de Petri perteneciente a la clase de redes *SOAR*^{2} que se ha obtenido a partir de un algoritmo de encaminamiento especificado por el diseñador, a través de un proceso de abstracción como las descritos en el capítulo 2 de esta tesis.

Tanto para la fase de análisis como para la fase de corrección se requiere disponer del grafo de poda de zonas. Esto requerirá previamente el cálculo de todas las zonas de uso continuado de todos los recursos de la red. Posteriormente será necesario calcular los conjuntos máximos de zonas solapadas de los recursos de la red *SOAR*^{2}. Finalmente, se construirá el grafo de poda de zonas y las funciones S, L y K que etiquetan los vértices y los arcos del grafo de poda de zonas de los recursos de la red. Todas estos cálculos han sido detallados y formalizados en el capítulo 3 de esta tesis, razón por la cual a continuación se listan las referencias detalladas a las partes de ese capítulo donde se realizan los cálculos más arriba requeridos,

- Algoritmo 4, sección 3.2.3, para el cálculo del conjunto de zonas. Dado que partimos de una red *SOAR*^{2} como entrada del algoritmo, todas las zonas calculadas son zonas binarias.
- Algoritmo 5, sección 3.2.5, para el cálculo de los conjuntos máximos de zonas solapadas. Dado que partimos de una red *SOAR*^{2} los conjuntos máximos obtenidos están totalmente ordenados por la relación de precedencia $<$ entre zonas solapadas (Definición 12 del capítulo 3).
- Definición 17 y comentarios posteriores en la sección 3.3.4, para el cálculo de grafo de poda simple de zonas de los recursos de la red *SOAR*^{2}.
- Definición 18 en la sección 3.3.4, para el cálculo de las funciones de etiquetado del grafo de poda simple de zonas de la red *SOAR*^{2} denominadas S, L y K.

La aplicación de los procedimientos referenciados anteriormente nos da como resultado, para su uso posterior,

- El grafo de poda de zonas de la red *SOAR*^{2} comprendiendo su estructura y las tres funciones de etiquetado S, L y K.
- La ordenación total de las zonas pertenecientes a cada conjunto máximo de zonas solapadas según la relación de precedencia, $<$, entre zonas solapadas.

4.4.2. Paso 2: Construcción del grafo de poda de recursos de la red *SOAR*^{2}

En el camino hacia la corrección de la red de Petri, el grafo de poda de recursos de la red será utilizado, de acuerdo con la caracterización de la vivacidad en redes S^4PR, para determinar cómo forzar la vivacidad. Esto es debido a que como se verá en los pasos siguientes, la pretensión es llegar a obtener un grafo de poda de recursos acíclico por transformación de la red de Petri de partida.

Obviamente si el grafo de poda de recursos de partida ya es acíclico no es necesario seguir el procedimiento, dado que con el marcado inicial admisible suministrado para esta red, ya es viva y no se requiere corrección alguna.

Para el cálculo de este grafo no se partirá ex novo desde la red de Petri suministrada por el diseñador, utilizando para ello los procedimientos presentados en [Cano 11] para la clase de redes S^4PR. En su lugar, calcularemos el grafo de poda de recursos a partir del grafo de poda de zonas calculado en el paso 1 del presente procedimiento. Para ello, se aplicará sobre el grafo de poda de zonas el método de aglomeración descrito en la definición 19 en la sección 3.3.4, que como se desmostró allí permite obtener el grafo de poda de recursos de la red *SOAR*^{2}.

Por lo tanto, la aplicación del referido procedimiento nos da como resultado, para su uso posterior,

- El grafo de poda de recursos de la red *SOAR*^{2} comprendiendo su estructura y las tres funciones de etiquetado S', L' y K'.

4.4.3. Paso 3: Determinación de un conjunto de arcos cuya eliminación hace acíclico el grafo de poda de recursos

El procedimiento de corrección de la red de Petri propiamente dicho empieza en este paso. Efectivamente, dado el grafo de poda de recursos de la red se trata de llegar a hacerlo acíclico por transformación de la red de Petri. Para ello, se necesita seleccionar un conjunto de arcos del grafo de poda de recursos que,

1. Su eliminación garantice que el grafo resultante sea acíclico y por tanto habremos alcanzado una red sin cerrojos malos.

2. Para cada uno seleccionado como víctima tengamos un procedimiento que nos permita identificar la región de la red de Petri en la que mediante una transformación adecuada haga desaparecer el arco.

3. Tengamos un procedimiento efectivo de transformación de la red de Petri en la región localizada que garantice la eliminación del arco seleccionado como víctima.

Estos tres puntos enumerados no son independientes y se influyen mutuamente a la hora de determinar un buen conjunto de arcos víctima del grafo de poda de recursos que si se eliminan hacen acíclico el grafo. Por ejemplo, un arco entre dos recursos en el grafo de poda de recursos que tenga su origen en varias parejas de zonas de esos recursos donde cada pareja de zonas está conectada mediante un arco en el grafo de poda de zonas requerirá una modificación de la red de Petri más amplia que otro arco que sólo esté originado por un par de zonas conectadas. Efectivamente, como vimos en la sección 4.3 una pareja de zonas solapadas relacionadas en el grafo de poda de zonas define una región localizada que habrá que transformar de cara a extinguir el arco del grafo de poda de recursos. Por tanto, la elección de un arco puede dar lugar a tener una o varias regiones localizadas en la red de Petri sobre las que hay que intervernir. De la misma manera, y como se vio en la sección 4.3 la selección del arco puede hacer que la transformación sea más o menos complicada en función de la relación de solape entre el par de zonas implicadas en la generación del arco del grafo de poda de recursos. Recuérdese que en el ejemplo presentado en la sección 4.3, la transformación de la red de Petri en la región definida por las zonas $Z^r_{2,1}$ y $Z^s_{2,1}$ requerirá la duplicación de un conjunto particular de lugares y transiciones de la zona $Z^s_{2,1}$; mientras que la transformación de la región definida por las zonas $Z^t_{3,1}$ y $Z^u_{3,1}$ no requería duplicación alguna de lugares ni de transiciones. Por lo tanto, también el propio procedimiento de transformación de una región de la red de Petri tiene influencia a la hora de optar por un arco víctima u otro porque la transformación será más simple o más compleja (con duplicación de algunos lugares y transiciones).

Por todo ello, y a sabiendas de esta íntima relación entre los tres apartados antes mencionados, en esta subsección nos concentramos en algoritmos adecuados para la selección de un conjunto de arcos cuya eliminación hace acíclico el grafo de poda de recursos. La influencia de los otros aspectos comentados quedará encapsulado en un conjunto de heurísticas que permitirán decidir entre diversas alternativas en el algoritmo voraz que proponemos. La justificación de estas heurísticas se realizará en la siguiente subsección.

El problema que consideramos aquí es la búsqueda de un conjunto de arcos en un grafo dirigido que sí los eliminamos, el grafo resultante es acíclico. Este problema pertenece al grupo de los denominados "feedback arc set problems" debido a que si el conjunto de arcos seleccionados se invierte su sentido en el grafo, entonces el grafo resultante es acíclico. Entre los problemas más famosos de este grupo se encuentra el *"minimum feedback arc set problem"* [Garey 83] que consiste en encontrar el conjunto de arcos de cardinal mínimo cuya eliminación hace acíclico el grafo dirigido. Este problema es un problema catalogado como NP-completo.

Algoritmos para resolver el problema del *"minimum feedback arc set problem"* existen varios publicados en la literatura, basados en diferentes estrategias: "depth first search" [Tarjan 82], aproximaciones "divide y vencerás" [Eades 90] y algoritmos voraces [Kaufmann 01]. Se ha demostrado que los algoritmos voraces son de entre los métodos no exactos los que tienen un comportamiento más rápido produciendo soluciones aproximadas de buena calidad. Adicionalmente, ya en [Kaufmann 01] se establece que una heurística voraz mejorada puede ser introducida con el objetivo de calcular un conjunto de arcos satisfaciendo un conjunto de restricciones particulares. Este último aspecto nos interesa especialmente después de la discusión anterior acerca de las características no escalares asociadas a un arco y que les puede hacer preferibles a otros arcos en el proceso de elección de los arcos que realice el algoritmo voraz (como número de regiones de la red de Petri implicadas en la definición del arco del grafo de poda de recursos; o la relación de inclusión existente entre la pareja de zonas que dan lugar al arco).

Antes de presentar el algoritmo introducimos la siguiente terminología. Dado un grafo dirigido $G = (V,E)$ diremos que un vértice $v \in V$ es un "vértice fuente" si no existe un arco $(x,v) \in E$, con $x \in V$. De la misma manera diremos que un vértice $v \in V$ es un "vértice sumidero" si no existe ningún arco $(v,x) \in E$, con $x \in V$. Dado un vértice $v \in V$, denominamos "grado de entrada" de v, $ge(v)$, al número de arcos de entrada al vértice v, i.e. número de arcos $(x,v) \in E$ con $x \in V$. De una forma análoga, dado $v \in V$, denominamos "grado de salida" de v, $gs(v)$, al número de arcos de salida del vértice v, i.e. número de arcos $(v,x) \in E$ con $x \in V$.

El algoritmo 6 presentado a continuación está inspirado en el algoritmo de [Kaufmann 01] incorporando heurísticas asociadas a características no escalares de los arcos a seleccionar.

El algoritmo 6 trabaja de forma que todos los vértices del grafo de poda de recursos son visitados solo una vez. Esto se consigue porque cada vez que un vértice es tratado se elimina del grafo junto con todos los arcos de entrada y salida al mismo. Mientras que existan vértices sin visitar, primero se eliminan los vértices fuente y sumidero porque estos no pueden formar parte de circuitos en lo que nos quede de grafo en la iteración en la que estemos. Cuando ya no se pueden eliminar más vértices de este tipo, cualquiera de los vértices que quede (si queda alguno debe formar parte de algún circuito). Por lo tanto, el algoritmo procede a buscar un vértice conforme a una heurística de forma que el conjunto de sus arcos de entrada pasa a formar parte del conjunto de arcos del grafo de poda de recursos cuya eliminación lo hace acíclico. La heurística consiste en seleccionar en primera instancia los vértices en los que la diferencia entre arcos de salida y arcos de entrada sea máxima. Esta heurística es heredada del trabajo presentado en [Kaufmann 01]. Si el número de vértices así seleccionados es igual a 1, entonces memorizamos sus arcos de entrada en el conjunto F y se eliminan todos los arcos de entrada y de salida y el mismo

Algoritmo 6 Selección de un conjunto de arcos del grafo de poda de recursos de una red *SOAR*^{2} cuya eliminación lo hace acíclico.

Entrada:

1. Grafo de poda de zonas $G = (V,E)$ de la red;
2. Grafo de poda de recursos $G' = (V',E')$ de la red

Salida:

3. Conjunto de arcos $F \subseteq E'$ de G' tales que $G'' = (V',E' \setminus F)$ es un grafo dirigido acíclico.
4. **Inicio**
5. $F = \emptyset$;
6. **Mientras que** $V' \neq \emptyset$ **Hacer**
7. **Mientras que** V' contenga un vértice fuente v **Hacer**
8. $E' = E' \setminus \{(v,x) \in E' | x \in V'\}$;
9. $V' = V' \setminus \{v\}$;
10. **Fin Mientras que**
11. **Mientras que** V' contenga un vértice sumidero v **Hacer**
12. $E' = E' \setminus \{(x,v) \in E' | x \in V'\}$;
13. $V' = V' \setminus \{v\}$;
14. **Fin Mientras que**
15. $W = \{v \in V' | gs(v) - ge(v) = max_{x \in V'}\{gs(x) - ge(x)\}\}$;
16. **Si** $|W| = 1$ **Entonces**
17. $F = F \cup \{(x,v) \in E' | v \in W \text{ y } x \in V'\}$;
18. $E' = E' \setminus \{(x,v) \in E' | v \in W \text{ y } x \in V'\}$;
19. $E' = E' \setminus \{(v,x) \in E' | v \in W \text{ y } x \in V'\}$;
20. $V' = V' \setminus \{v \in W\}$;
21. **Fin Si**
22. **Si** $|W| > 1$ **Entonces**
23. Mantener en W solamente los vértices $v \in W$ tales que el número de arcos del grafo de poda de zonas, G, que de origen a los arcos de entrada a v en el grafo de poda de recursos sea mínimo.
24. **Si** $|W| > 1$ **Entonces**
25. Mantener en W un vértice $v \in W$ tal que el número de arcos (Z^r, Z^s) del grafo de poda de zonas, G, que de origen a los arcos de entrada a v en el grafo de poda de recursos tales que $Fa(Z^r) \subseteq Z^s$ sea máximo.
26. **Fin Si**
27. $F = F \cup \{(x,v) \in E' | v \in W \text{ y } x \in V'\}$;
28. $E' = E' \setminus \{(x,v) \in E' | v \in W \text{ y } x \in V'\}$;
29. $E' = E' \setminus \{(v,x) \in E' | v \in W, x \in V'\}$;
30. $V' = V' \setminus \{v \in W\}$;
31. **Fin Si**
32. **Fin Mientras que**
33. **Fin**

vértice del grafo G'. En caso que hubiera más de un vértice en el conjunto W, entonces se seleccionarían aquellos vértices que implicasen el menor número de arcos entre zonas para generar los arcos de entrada a los vértices de W. Si aún así quedasen varios se seleccionaría uno de los vértices que requiriese duplicar el menor número de lugares y transiciones como se explicó en la sección 4.3 sobre el ejemplo.

La aplicación del algoritmo anterior al grafo de poda de recursos de la figura 4.7 nos llevaría a los siguientes resultados:

1. En la primera iteración del bucle externo no se encuentran ni vértices fuente ni sumidero. La parte final del bucle selecciona un conjunto $W = \{t\}$ ya que $gs(t) - ge(t) = 2$, que es el máximo cuando se hace este cálculo para todos los nodos del grafo. Como el conjunto W solo contiene un elemento entonces se eliminan todos los arcos de entrada y de salida de t y el mismo vértice t. El conjunto $F = \{(u,t)\}$. En la figura 4.18 se ilustra el grafo resultante de esta iteración.

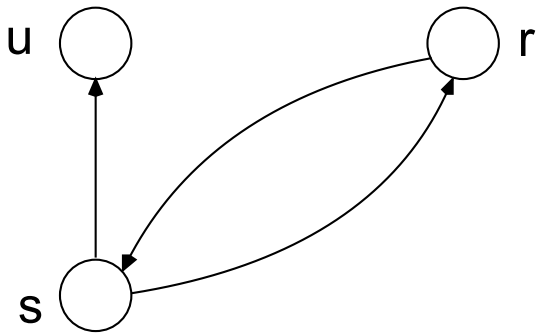

Figura 4.18: Grafo resultante de la primera iteración del algoritmo 6 aplicado al grafo de poda de recursos de la figura 4.7.

2. En la segunda iteración existe un nodo sumidero que es el nodo u que será eliminado junto con sus arcos de entrada. Al final del bucle principal el conjunto W es igual a $\{s,r\}$, dado que ambos casos la diferencia entre arcos de salida y arcos de entrada es la misma. En ambos casos, el número de arcos del grafo de poda de zonas que dan lugar a sus arcos de entrada es el mismo: para el arco (s,r) el arco $(Z^s_{2,1}, Z^r_{2,1})$ y para el arco (r,s) el arco $(Z^r_{1,1}, Z^s_{1,1})$. Ahora bien, este último par de zonas es el que requiere duplicar el menor número de lugares y transiciones ya que en este caso se cumple la condición expresada al final: $Fa(Z^r_{1,1}) \subseteq Z^s_{1,1}$. Esto quiere decir que en W retenemos solo el vértice s y por lo tanto $F = \{(u,t),(r,s)\}$.

Obsérvese, que no es esta la solución que se adoptó en la sección 4.3 ya que allí se optó por el arco (s,r) en lugar del arco (r,s). La diferencia estriba que con la elección que allí hicimos tuvimos que duplicar un conjunto de lugares y transiciones, mientras que con la solución dada por el algoritmo 6, no tenemos que duplicar ningún lugar ni ninguna transición y por lo tanto la corrección es menos costosa. (Nota: en la sección 4.3 se adoptó el arco (s,r) con el objetivo de ilustrar el proceso de duplicación de lugares y transiciones necesario en los casos en los que la frontera de asignación de la zona que poda no está incluida en la zona podada).

El resultado final de este paso nos da como resultado, para su uso posterior,

- Un conjunto de arcos del grafo de poda de recursos que si se eliminan hacen acíclico este grafo. El conjunto en general no es mínimo y trata de reducir el número de transformaciones y complejidad de las mismas sobre la red de Petri.

4.4.4. Paso 4: Eliminación de arcos del grafo de poda de recursos por adición de recursos virtuales entre las zonas solapadas que generan estos arcos

Este constituye el último paso del procedimiento de corrección de la red de Petri de la clase *SOAR*^{2} para lograr que sea viva. Se parte de un conjunto de arcos, que se han calculado en el paso 3, pertenecientes al grafo de poda de recursos de la red y cuya eliminación hará que el grafo sea acíclico.

A partir de la información de estos arcos dos son las tareas que hay que emprender en este punto y que ya han sido anunciadas en varias ocasiones anteriores,

1. Localizar las regiones de la red de Petri donde efectivamente hay que intervenir realizando transformaciones. Estas regiones se localizan a través de los arcos del grafo de poda de recursos que suministra el paso 3 y la transformación persigue su eliminación.

2. Definir y aplicar la transformación propiamente dicha a realizar sobre la región localizada a través del arco del grafo de poda de recursos que logre eliminar el arco del grafo de poda de recursos.

En lo que concierne a la primera tarea, las regiones de la red de Petri donde hay que intervenir y determinadas por un arco de los seleccionados del grafo de poda de recursos se obtienen a partir de los pares de zonas solapadas de recursos conectadas en el grafo de poda de zonas y que dan lugar al arco seleccionado como víctima de acuerdo con el procedimiento de aglomeración establecido en la definición 19 del capítulo 3. Más en

concreto, si (r,s) es uno de los arcos seleccionados como víctima donde r,s son vértices del grafo de poda de recursos y por tanto cada uno de ellos son recursos de la red $SOAR^2$, entonces por la antedicha definición 19 existe un conjunto no vacío de pares de zonas $Z^r_{i,j}$ y $Z^s_{i',j'}$ en el grafo de poda de zonas tales que $(Z^r_{i,j};Z^s_{i',j'})$ es un arco en el grafo de poda de zonas de la red.

Denominamos $G=(V,E)$ el grafo de poda de zonas de la red, $G'=(V',E')$ el grafo de poda de recursos de la red, y Q el conjunto de arcos de G' obtenidos en el paso 3 del procedimiento de corrección que estamos formalizando.

Denotaremos como R al conjunto de todos los pares de zonas que representan un arco del grafo G que de origen a un arco víctima de Q conforme al procedimiento de aglomeración de la definición 19 del capítulo 3. Es decir;

$$R=\{(Z^r_{i,j};Z^s_{i',j'})\in E \mid (r,s)\in Q\subseteq E' \text{ y } r,s\in P_R\}$$

Por lo tanto, del conjunto de entrada Q dado en el paso 3, calculamos el conjunto R que son en esencia (o pueden ser vistos) pares de zonas, cada una de las cuales define una región de intervención en la red de Petri para su transformación.

Obsérvese que la intervención deberá realizarse en todos y cada uno de estos pares de zonas ya que si no se hiciese así y quedase alguno de estos pares de zonas como un arco del grafo G original, entonces por el procedimiento de aglomeración de la definición 19 haría que el arco de G' conectando los recursos asociados a las zonas del par subsistiese en el grafo G' y por tanto no habríamos logrado nuestro objetivo de hacer acíclico el grafo G'.

A continuación justificamos la intervención sobre cada par de zonas del conjunto R y el procedimiento de intervención.

La razón por la que aparece un arco como $(Z^r,Z^s)\in R$ se debe a que la zona Z^r poda a la zona Z^s, es decir, la zona Z^r consigue que algunos de los lugares privados de Z^s lleguen a ser no esenciales en cualquier cerrojo que llegase a contener simultáneamente los recursos r y s. Esta capacidad de poda de la zona Z^r sobre la zona Z^s se consigue precisamente gracias al recurso r que entra en las transiciones de entrada a los lugares de la frontera de asignación de la zona Z^r haciendo que los lugares proceso de entrada a estas transiciones que pertenecen a Z^s lleguen a ser no esenciales. Por lo tanto la capacidad de poda de Z^r sobre Z^s, que es lo mismo que la aparición del arco (Z^r,Z^s) en G, es debida a que

1. Las zonas Z^r y Z^s están solapadas y además $Z^s < Z^r$ según la relación de precedencia.

2. El recurso r hace no esenciales alguno lugares proceso de $Z^s \setminus Z^r$.

En esta justificación se encuentra la base del método de eliminación del arco (Z^r, Z^s). La idea es sustituir el recurso r utilizado en la zona Z^r (solapada con Z^s) de forma continuada por un recurso virtual nuevo r' que tendrá las siguientes propiedades,

a) Será un recurso que solo será utilizado por lugares de la zona Z^r que conviertan en no esenciales los mismos lugares que hacía no esenciales el recurso r, y además r' no será utilizado en ninguna otra parte de la red. Se trata de un recurso privado para esos lugares proceso pertenecientes a la zona Z^r.

b) Será un recurso del cual existirá una única copia inicialmente. Es decir, su marcado inicial será igual a la unidad de la misma manera que se definió para el resto de lugares recurso de una red *SOAR*^{2} el marcado inicial admisible.

El algoritmo 7 que se detalla a continuación debe contemplar adicionalmente dos aspectos a la hora de realizar la transformación,

A) El primer aspecto concierne a la conexión del nuevo recurso virtual r' con las transiciones de la red y la construcción de la nueva zona de uso continuado del recurso r'. Como ya se ilustró en el ejemplo de la sección 4.3, tenemos dos casos que se distinguen en función de que la frontera de asignación de Z^r, en el par (Z^r, Z^s) con $Z^s < Z^r$), esté incluida o no en Z^s. Si la frontera de asignación está incluida, $Fa(Z^r) \subseteq Z^s$ entonces el lugar r' se conecta a las mismas transiciones que el lugar r relacionadas con la zona Z^r. Es decir, $(r')^\bullet = {}^\bullet(Fa(Z^r))$ y ${}^\bullet(r') = (Fl(Z^r))^\bullet$. La nueva zona generada $Z^{r'}$ es exactamente igual que la zona Z^r anterior (contiene los mismos lugares) y las relaciones de $Z^{r'}$ con otras zonas son las mismas que tenía Z^r con las zonas. Ahora bien, si no se cumple que $Fa(Z^r) \subseteq Z^s$, entonces $(r')^\bullet = {}^\bullet(Fa(Z^r) \cap Z^s)$ y el resto de transiciones ${}^\bullet(Fa(Z^r) \setminus Z^s)$ mantendrán como lugar de entrada al recurso original r. Ahora bien, esto significa que alguna de las transiciones de liberación del recurso r antiguo no será capaz de discriminar qué recurso fue asignado al entrar en la antigua zona Z^r, si el recurso r o el recurso r'. Para poder distinguir cuál fue el recurso asignado habrá que duplicar los lugares y transiciones que saliendo de la zona Z^s conducen hasta una transición de salida de la frontera de liberación en cuyo caso liberarán el recurso r' que es el que fue asignado. El resto de las transiciones de salida de la frontera de liberación de la antigua zona Z^r liberarán el antiguo recurso r.

B) El segundo aspecto concierne al propio proceso de eliminación de los arcos. Si bien es cierto que en el proceso de eliminación de un arco $(Z^r, Z^s) \in R$, éste desaparece formalmente del grafo G de poda de zonas, en realidad la estructura de este grafo se ve inalterada porque se ha introducido el nuevo vértice $Z^{r'}$ que deja esencialmente

inalterada la estructura del grafo y las funciones de etiquetado. La única alteración que se ha producido en G es un cambio de nombre, hemos pasado de Z^r a $Z^{r'}$. Este cambio de nombre también podría tener que afectar a pares de zonas almacenadas en R que tuvieran como segunda componente Z^r y todavía no hubieran sido tratadas. En este caso, tendríamos que actualizar también el conjunto R. Para evitar esta actualización de R cuando cambiamos una zona Z^r por la correspondiente $Z^{r'}$ por introducción del recurso virtual r', es por lo que al principio del algoritmo se seleccionan pares de zonas $(Z^r, Z^s) \in R$ tales que no tiene un par predecesor (Z^t, Z^r) y así evitar la actualización de R cuando se intervenga en el par (Z^r, Z^s).

C) El tercer aspecto a tener en cuenta se refiere al fenómeno comentado al final de la sección 4.3 anterior. Cuando se intervenga en una zona (Z^r, Z^s) en la que la zona Z^r no sea la zona máxima en los conjuntos máximos de zonas solapadas a las que pertenece, entonces después de tratar el par de zonas por alguno de los procedimientos se incorporarán al conjunto R todos los pares de zonas predecesoras de el par (Z^r, Z^s) en el grafo de poda de zonas. Los pares se incorporarán con el nuevo nombre de la zona puesto que se ha introducido el recurso virtual r' y la zona de interés ahora es $Z^{r'}$. La terminación del algoritmo depende de que el conjunto R llegue a ser vacío. Esta adición de pares extra al conjunto R no pone en cuestión la terminación dado que la adición de pares termina cuando se alcanzan los pares (Z^r, Z^s) en los que Z^r son zonas máximas en los conjuntos máximos a los que pertenecen. En ese momento no se añaden más y dado que el grafo de poda de zonas es acíclico (como se verá en la siguiente sección) el número de estas adiciones será finito.

El algoritmo 7 implementa la transformación de la red de Petri a partir del conjunto de pares de zonas R en las que hay que intervenir para llegar a obtener un grafo acíclico.

En el algoritmo anterior, además de incluir la transformación de la red de Petri se ha incluido también el cálculo de las nuevas zonas generadas por los recursos virtuales añadidos y la transformación del grafo de poda de zonas de la red para en todo momento disponer del grafo de la red de Petri transformada. Esto se ha hecho así para facilitar la presentación de los resultados de la sección siguiente relativos a la terminación del algoritmo y la demostración sobre la vivacidad de la red de Petri final obtenida por el algoritmo.

Algoritmo 7 Transformación de la red de Petri para la eliminación del conjunto de arcos R del grafo de poda de zonas mediante recursos virtuales.

Entrada:

1. $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ la red de Petri $SOAR^2$ que es necesario transformar para forzar su vivacidad, $\mathbf{m_0}$ es su marcado inicial aceptable.
2. $\mathscr{Z}$ el conjunto de zonas de uso continuado de recursos de $\mathcal{N}$.
3. $G = (V, E)$ el grafo de poda de zonas de recursos de $\mathcal{N}$.
4. $R = \{(Z^r_{i,j}; Z^s_{i',j'}) \in E |$ son arcos que hay que eliminar de $G\}$

Salida:

5. $\mathcal{N}^L$ la red de Petri $SOAR^2$ transformada de forma que con el marcado inicial aceptable $\mathbf{m_0}^L$ es viva.
6. $\mathscr{Z}^L$ el conjunto de zonas de uso continuado de recursos de $\mathcal{N}^L$
7. $G^L = (V^L, E^L)$ el grafo de poda de zonas de recursos de $\mathcal{N}^L$.
8. **Inicio**
9. $\mathcal{N}^L = \mathcal{N}$; $\mathbf{m_0}^L = \mathbf{m_0}$; $\mathscr{Z}^L = \mathscr{Z}$; $G^L = G$;
10. **Mientras que** $R \neq \emptyset$ **Hacer**
11. Seleccionar un par $(Z^r, Z^s) \in R$ tal que no existe otro par $(Z^t, Z^r) \in R$;
12. $R = R \setminus \{(Z^r, Z^s)\}$;
13. **Si** $Fa(Z^r) \subseteq Z^s$ **Entonces**
14. Eliminar de $\mathcal{N}^L$ todos los arcos del lugar r a las transiciones ${}^\bullet Fa(Z^r)$
15. Eliminar de $\mathcal{N}^L$ todos los arcos de las transiciones $Fl(Z^r)^\bullet$ al lugar r
16. Añadir a $\mathcal{N}^L$ un nuevo lugar r' tal que $(r')^\bullet = {}^\bullet(Fa(Z^r))$ y ${}^\bullet(r') = (Fl(Z^r))^\bullet$.
17. Añadir a $\mathbf{m_0}^L$ el marcado inicial de r' con valor igual a la unidad
18. $Z^{r'} = Z^r$;
19. $\mathscr{Z}^L = (\mathscr{Z}^L \setminus \{Z^r\}) \cup \{Z^{r'}\}$;
20. Renombrar el vértice Z^r en G^L dándole el nombre $Z^{r'}$, realizar este renombramiento en las funciones S, L y K
21. **Para todo** $(Z^x, Z^{r'}) \in E^L$ **Hacer**
22. $R = R \cup \{(Z^x, Z^{r'})\}$;
23. **Fin Para**

24. **Sino**
25. Replicar en $\mathscr{N}^L$ todas las transiciones, arcos y lugares proceso que aparecen en algún camino simple entre una $t \in Fl(Z^s)^\bullet$ y un lugar $p \in (Fl(Z^r)^\bullet)^\bullet \cap P_S$ a excepción de los propio t y p.
26. Conectar en $\mathscr{N}^L$ los recursos diferentes a r a las transiciones replicadas de las misma manera a como estaban conectadas a las transiciones originales.
27. Eliminar los arcos de $\mathscr{N}^L$ que conectan un $t \in Fl(Z^s)^\bullet$ y un lugar $p \in Z^r \setminus Z^s$.
28. Añadir a $\mathscr{N}^L$ un nuevo lugar r' tal que $(r')^\bullet = {}^\bullet(Fa(Z^r) \cap Z^s)$ y ${}^\bullet(r') = \{t' | t'$ es una transición replicada de una transición $t \in Fl(Z^s)^\bullet\}$
29. Añadir a $\mathbf{m_0}^L$ el marcado inicial de r' con valor igual a la unidad
30. Crear la nueva $Z^{r'} = Z^r \cap Z^s \cup \{p | p$ es un lugar proceso replicado $\}$
31. Actualizar la zona $Z^r = Z^r \setminus (Z^r \cap Z^s)$ en $\mathscr{Z}$
32. $\mathscr{Z}^L = \mathscr{Z}^L \cup \{Z^{r'}\}$
33. Replicar en G^L el vértice Z^r con todos sus arcos de entrada y salida dándole a la réplica el nombre $Z^{r'}$
34. Actualizar los valores $S(Z^r)$ y $S(Z^{r'})$ conforme a los nuevos valores calculados para Z^r y $Z^{r'}$
35. Eliminar de G^L el arco (Z^r, Z^s)
36. Actualizar los siguientes arcos de G^L con arreglo a los valores calculados de Z^r y $Z^{r'}$: $(Z^r, Z^x) \in E^L$ y $(Z^x, Z^r) \in E^L$ con $x \neq s$; $(Z^{r'}, Z^x) \in E^L$ y $(Z^x, Z^{r'}) \in E^L$ con $x \neq s$
37. Actualizar las funciones L y K de G^L con arreglo a los nuevos valores calculados de Z^r y $Z^{r'}$
38. **Para todo** $(Z^x, Z^{r'}) \in E^L$ **Hacer**
39. $R = R \cup \{(Z^x, Z^{r'})\}$;
40. **Fin Para**
41. **Fin Si**
42. **Fin Mientras que**
43. **Fin**

4.5. Propiedades del método de corrección, vivacidad del modelo resultante y condiciones de terminación

En esta sección se enuncian las propiedades y resultados fundamentales relacionados con el método de corrección propuesto en la sección anterior junto con las correspondientes demostraciones.

Denominaremos $G = (V,E)$ al grafo de poda de zonas de la red, $G' = (V',E')$ al grafo de poda de recursos de la red, y Q al conjunto de arcos de G' obtenidos en el paso 3, del procedimiento de corrección. En el algoritmo 7 se denomina R al conjunto de todos los pares de zonas que representan un arco del G que de origen a un arco víctima del conjunto Q conforme al procedimiento de aglomeración de la definición 19 del capítulo 3. Es decir;

$$R = \{(Z^r_{i,j}); (Z^s_{i',j'}) \in E \mid (r,s) \in Q \subseteq E' \; y \; r,s \in P_R\}$$

Las transformaciones de la red de Petri presentadas en el algoritmo 7 son de dos tipos, dependiendo que en el par de zonas implicadas (Z^r, Z^s) se cumple o no la condición $Fa(Z^r) \subseteq Z^s$. Las diferencias entre un tipo de transformación y otra resultan ostensibles a la vista del algoritmo, pero conviene observar que la primera *(cuando se cumple que* $Fa(Z^r) \subseteq Z^s$*)* es simplemente el caso límite de la otra transformación.

La primera de las transformaciones es bastante simple dado que la antigua zona Z^r del par (Z^r, Z^s) desaparece completamente y aparece la nueva zona $Z^{r'}$ asociada al nuevo recurso virtual r' que sustituye al recurso r en relación con los lugares de la zona Z^r y sus transiciones anexas. Desde el punto de vista del grafo de poda de zonas, éste no cambia en su estructura, ni tampoco cambian las funciones S, L ni K salvo en lo que se refiere a un cambio de nombre que sustituye el nombre antiguo de la zona $Z^{r'}$ ya que incluso se cumple que $S(Z^r) = S(Z^{r'})$.

Está claro que si la única modificación introducida por esta transformación sobre el grafo de poda de zonas es cambiar el nombre de una zona del recurso r de Z^r a $Z^{r'}$, y además esta última es la única zona existente del recurso virtual r' introducido como nuevo, el circuito o circuitos que se cerraban a través del arco (r,s) en el grafo de poda de recursos ha desaparecido. Obviamente, lo anterior es cierto si ese arco (r,s) tuviera su origen exclusivamente en el par (arco) de zonas (Z^r, Z^s). Lograremos la completa desaparición del arco (r,s) si se realizan las correspondientes transformaciones sobre todo el resto de pares de zonas (Z^r_a, Z^s_b) que den origen al arco (r,s), introduciendo un recurso virtual diferente por cada uno de los pares implicados sobre el arco (r,s) del grafo de poda de recursos. Obsérvese que a cada par de zonas que se aplica la transformación descrita, el efecto que se produce sobre el arco (r,s) es que las cardinalidades de $L'((r,s))$ y $K'_G(s)$ disminuyen hasta que el último par transformado llegan a ser conjuntos vacíos.

El otro aspecto que debemos analizar para determinar que la transformación es efectiva camino de obtener un grafo de poda de recursos refiere a la conectividad que tenemos con el nuevo vértice del grafo de poda de recursos que está asociado precisamente al nuevo recurso virtual r'. Efectivamente, según hemos visto el arco (r,s) desaparece del grafo de poda de recursos cuando hemos intervenido en todas las parejas de zonas que dan origen al arco (r,s). No obstante, por cada pareja de zonas hemos añadido un nuevo recurso virtual. Sea r' el recurso virtual añadido por la pareja de zonas (Z^r, Z^s). En el grafo de poda de zonas la estructura, es decir, la conectividad de la zona $Z^{r'}$ con otras zonas de la red es exactamente la misma que la que tenía la zona Z^r, entre otras cosas porque se trata del mismo conjunto de lugares. De acuerdo con esto, el nuevo vértice r' en el grafo de poda de recursos tendrá las siguientes propiedades en cuanto a su conectividad con los otros vértices del grafo de poda de recursos.

a) El vértice r' está conectado a todos los vértices x del grafo de poda de recursos mediante un arco (r', x), que previamente hubiera estado conectado el vértice r mediante un arco (r, x). Esto es así debido a que la zona $Z^{r'}$ mantiene la misma conectividad que la zona Z^r en el grafo de poda de zonas y por lo tanto tenemos las mismas parejas de zonas que antes pero renombradas, es decir, los arcos que señalábamos (r', x) que se encuentran en el grafo de poda de recursos se deben a pares de zonas $(Z_c^{r'}, Z_c^x)$ que existen en el grafo de poda de zonas porque previamente existía un par de zonas (Z_c^r, Z_c^x) en este grafo.

b) Si la zona Z_c^r era una zona máxima en todos los conjuntos máximos de zonas solapadas, con respecto a la relación de precedencia *(que en redes SOAR^2 ordena totalmente estos conjuntos máximos)*, entonces en el grafo de poda de recursos no existirá ningún arco (x, r') de entrada al recurso r' por dos razones. La zona $Z_c^{r'}$ no tiene ninguna zona tal que el par $(Z_c^x, Z_c^{r'})$ sea un arco de este grafo. La segunda razón es que $Z_c^{r'}$ es la única zona que existe por el recurso r'. Por lo tanto, al aplicar el procedimiento de agregación no existirá ningún arco (x, r'), con el cual conseguimos el objetivo de eliminar completamente el arco (r,s) del grafo de poda y además que r' no introduzca ningún circuito nuevo por el que haya que tomar acciones adicionales para su eliminación.

c) Si la zona Z_c^r no era una zona máxima en algún conjunto máximo de zonas solapadas, entonces existirá un par de zonas $(Z_c^x, Z_c^{r'})$ que dará lugar a un arco de entrada a r' en el grafo de poda de recursos que puede mantener circuitos que antes atravesaban el vértice r y ahora mantienen el mismo circuito pero a través de r'. A pesar de la existencia de este arco (x, r') debido al par de zonas $(Z_c^x, Z_c^{r'})$ hay que decir que este par es el único que puede dar origen al arco (x, r') del grafo de poda de recursos. Esto es así, debido a que una vez más la zona $Z_c^{r'}$ es única y entre zonas de x y la única zona de r' solo puede

existir una zona de x solapada en el mismo conjunto máximo de zonas solapadas. Por lo tanto, la eliminación de este par de zonas garantiza que se eliminará el arco (x, r'), pero probablemente a costa de introducir de un arco (y, x') que nos mantiene en la misma situación descrita en este apartado. No obstante, este proceso de eliminación de pares de zonas por culpa de que las zonas implicadas no sean máximas en todos los conjuntos máximos de zonas solapadas a las que pertenecen no puede continuar indefinidamente. Esto es debido a que el grafo de poda de zonas es acíclico y la propagación hacia atrás en la eliminación de arcos se detendrá en el momento que se alcance el máximo del conjunto máximo de zonas solapadas que se esté recorriendo.

Finalmente hay que señalar que la transformación apuntada para un par de zonas (Z^r, Z^s) en la que $Fa(Z^r) \subseteq Z^s$, garantizan que no salimos de la clase de redes *SOAR*^{2} a pesar de haber introducido un recurso virtual adicional. La clave para ello es que se ha mantenido la misma conectividad para el nuevo recurso r' que la que tenía el recurso r. Por lo tanto, el nuevo recurso r' forma un p–semiflujo mínimo con los lugares de la zona $Z^{r'}$ porque así lo hacía el lugar r cuando nos restringíamos a considerar exclusivamente las columnas de la matriz de incidencia correspondientes a las transiciones que pertenecen a la máquina de estados donde está alojada la zona $Z^{r'}$. El otro efecto observado es que el p–semiflujo mínimo asociado al recurso r se ha visto disminuido en los lugares pertenecientes a la zona Z^r *(todos ellos ahora forman la zona* $Z^{r'}$*)*.

En el algoritmo 7 también se puede apreciar que el marcado inicial de la red sigue siendo un marcado inicial admisible. Además tampoco se han modificado ni las zonas *(solo se han renombrado algunas)* no sus relaciones, por lo que se mantienen los mismos conjuntos máximos de zonas solapadas y las mismas relaciones de precedencia que se tenían anteriormente a la transformación de la red, aunque el nombre de algunas zonas ha cambiado.

A continuación se formalizan todos los comentarios anteriores referidos a la transformación primera sobre el par de zonas (Z^r, Z^s) en las que $Fa(Z^r) \subseteq Z^s$.

El primer resultado que se presenta establece que en redes *SOAR*^{2} no es posible tener ciclos en el grafo de poda de zonas y por lo tanto siempre existirá un único máximo en cada conjunto máximo de zonas solapadas según la relación de precedencia entre zonas.

Lema 22. *Sea* $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ *una red SOAR*^{2}*. El grafo de poda de zonas de* $\mathcal{N}$*,* $G = (V, E)$*, no contiene ciclos.* □

Demostración. La demostración se realiza por contradicción. Supongamos que existe un ciclo de zonas $Z_0, Z_1, ..., Z_k$ tal que $Z_0 = Z_k$. Entre dos zonas consecutivas de este ciclo, Z_i y Z_{i+r} con $i = 0..k-1$, se verifica que existe un arco (Z_i, Z_{i+1}) lo cual implica que $Z_{i+1} < Z_i$, i.e. Z_{i+1} precede a Z_i, por lo tanto concluimos que $Z_0 < Z_0$, pero como ya se demostró la relación de precedencia es irreflexiva, con lo cual hemos alcanzado la contradicción. □

Del lema anterior se deriva el siguiente lema según el cual, el proceso de selección que se realiza sobre el conjunto $R \neq 0$ al comienzo del bucle principal del algoritmo 7 siempre devuelve un resultado.

Lema 23. *Sea $\mathcal{N}$ una red de Petri SOAR^2 y R un conjunto no vacío de pares de zonas del grafo de poda de zonas, $G = (V,E)$, conectadas mediante un arco. Siempre existe un par $(Z^r, Z^s) \in R \subseteq E$ tal que no existe $(Z^x, Z^r) \in R$.* □

Demostración. Los pares contenidos en R son arcos del grafo G. Procedemos a calcular el camino más largo de G, en el que todos los arcos recorridos son pares contenidos en R. Este camino no puede ser un circuito por el lema 22, y como se trata de una red *SOAR*^{2} está totalmente ordenado por la relación de precedencia. Por lo tanto las dos zonas máximas según este ordenamiento definen el arco o el par de zonas que cumple la propiedad establecida en el enunciado del lema. □

El siguiente resultado establece que la modificación de la red de Petri al intervenir por el par de zonas (Z^r, Z^s) nos permite obtener una nueva red de Petri que pertenece a la clase de *SOAR*^{2} si la red original de la que partíamos era una red *SOAR*^{2}.

Lema 24. *Sea $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ una red SOAR^2. Sea (Z^r, Z^s) un par de zonas que definen un arco del grafo de poda de zonas de $\mathcal{N}$ tales que $Fa(Z^r) \subseteq Z^s$. Sea $\mathcal{N}^L$ la red de Petri obtenida de $\mathcal{N}$ mediante los siguientes pasos:*

1. *Añadir un recurso r' al conjunto P_R.*
2. *Eliminar todos los arcos de r a las transiciones ${}^\bullet Fa(Z^r)$.*
3. *Eliminar todos los arcos desde las transiciones $Fl(Z^r)^\bullet$ al lugar r.*
4. *Añadir los arcos que conectan r' con las transiciones de la red de la forma que $(r')^\bullet = {}^\bullet(Fa(Z^r))$ y ${}^\bullet(r') = (Fl(Z^r))^\bullet$.*

Entonces $\mathcal{N}^L$ es una red de Petri SOAR^2. □

Demostración. La forma en la que se ha conectado el lugar r' a las transiciones de la red y los arcos que se han eliminado de o hacia r hacen que la fila de la matriz de incidencia correspondiente al lugar r, después de la transformación se haya descompuesto en dos filas. Una fila correspondiente al lugar r que será igual a la anterior fila de r excepto que $C[r,t] = 0$ para toda transición t perteneciente a ${}^\bullet Fa(Z^r) \cup Fl(Z^r)^\bullet$, lo cual representa los arcos eliminados que conectaban con r desde las fronteras de Z^r. La otra fila corresponde al nuevo recurso añadido r' que será igual a cero excepto $C[r',t] = -1$, $\forall t \in {}^\bullet Fa(Z^r)$ y $C[r',t] = 1, \forall t \in Fl(Z^r)^\bullet$ *(los mismos valores y para las mismas transiciones que teníamos*

para r antes de la transformación de la red Petri. Obsérvese, que $C[r]+C[r']$ en la nueva red es igual a la antigua fila de r en la matriz de incidencia. Por lo tanto, $\mathbf{y}_{r'}$ es un p–semiflujo mínimo binario de la red con soporte $\|\mathbf{y}_{r'}\| = Z^r \cup \{r'\}$ porque este vector anulaba las columnas correspondientes a la máquina de estados que contiene Z^r. Además, $\mathbf{y}_r \in \{0,1\}^{|P|}$ con soporte igual al anterior p–semiflujo mínimo de r pero ahora eliminando los lugares Z^r es un p–semiflujo mínimo binario del recurso r porque anulaba todas las columnas de C menos las columnas de las transiciones de la máquina de estados que contiene Z^r *(ya que para anular estas columnas se necesitan las filas correspondientes a las lugares de* Z^r*).*

La forma en la que se ha reconectado r' a las transiciones del entorno de Z^r, hacen que la zona de r', $Z^{r'}$, sea igual a la zona Z^r antes de la transformación, i.e. el conjunto de lugares proceso de Z^r y $Z^{r'}$ es el mismo. Por lo tanto, las relaciones de solape y de poda serán exactamente igual a las que se tenían antes de la transformación la única diferencia es que allí donde aparecía el nombre de la vieja zona, Z^r, ahora deberá aparecer el nombre $Z^{r'}$.

Esto quiere decir que estructuralmente el grafo de poda de zonas no ha cambiado, manteniendo todas las relaciones existentes antes de la trasformación. En particular, se mantendrán los mismos conjuntos máximos de zonas solapadas de recursos *(con el renombramiento de la zona* Z^r *con el nombre* $Z^{r'}$*)* y se mantendrán los mismos ordenamientos totales previos en este conjunto de zonas debido a que el grafo no ha cambiado, los lugares de la zona $Z^{r'}$ y Z^r son los mismos, y el recurso r' se conecta a sus transiciones de entrada y de salida de la misma manera que lo hacía el recurso r antes de la transformación por lo que todas la relaciones de precedencia que mantenía $Z^{r'}$ y con las mismas características. Por lo tanto, la red $\mathscr{N}^L$ es una red *SOAR*^{2}. □

El resultado anterior es importante para garantizar que tras cada iteración del bucle principal del algoritmo 7 estamos en las mismas condiciones del inicio del bucle, es decir, a pesar de la transformación continuamos dentro de la clase *SOAR*^{2} y por lo tanto podemos iniciar una nueva iteración para proceder con otro par de zonas del conjunto R en las mismas condiciones que la iteración anterior.

El siguiente resultado simplemente es un corolario de los resultados anteriores referidos a los cambios producidos en el conjunto de zonas de la red modificada y en el grafo de poda de zonas de la red.

Lema 25. *Sea* $\mathscr{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C}\rangle$ *una red SOAR*^{2}*. Sea* (Z^r, Z^s) *un par de zonas que definen un arco del grafo de poda de* $\mathscr{N}$ *y sea* $\mathscr{N}^L$ *la red obtenida mediante la transformación apuntada en el lema 24 por adición del recurso virtual* r' *con relación a la zona* Z^r*. El conjunto de zonas de* $\mathscr{N}^L$ *es el mismo que el conjunto de zonas de* $\mathscr{N}$ *salvo*

que la zona Z^r es renombrada como $Z^{r'}$. El grafo de poda de zonas de $\mathscr{N}^L$ es el mismo que el grafo de poda de $\mathscr{N}$ salvo el renombramiento anterior.

El segundo tipo de transformación de la red de Petri propuesto en el algoritmo 7 concierne pares de zonas (Z^r, Z^s) en las que $Fa(Z^r) \not\subseteq Z^s$. En este caso, se requiere la replicación de algunos lugares, transiciones y arcos de la red original con el objetivo de memorizar que recurso se asignó a la entrada de la antigua zona Z^r: o bien el recurso r o bien el recurso r' *(ya que $Fa(Z^r) \not\subseteq Z^s$ y el lugar r' nuevo solo entra a las transiciones $^\bullet(Fa(Z^r) \cap Z^s)$ y el lugar r sigue entrando a las transiciones $^\bullet(Fa(Z^r) \setminus Z^s)$)* y por lo tanto liberar correctamente el recurso que fue asignado.

El efecto que esto produce es que la antigua zona Z^r, ahora da lugar a dos zonas nuevas distintas de la zona original Z^r,

a) La primera de las zonas es la nueva zona $Z^{r'}$ que estará formada por los lugares que pertenecen a ambas zonas $Z^r \cap Z^s$ más los lugares que se han replicado que son los lugares que van desde las transiciones de salida de la zona Z^r a las transiciones de salida de la zona Z^s a través de caminos simples. Obsérvese que esta nueva zona pertenecerá a los mismos conjuntos máximos de zonas solapadas a las que pertenecían las antiguas zonas Z^r y la Z^s ya que contiene los lugares $Z^r \cap Z^s$.

b) La segunda zona que se crea es la correspondiente al recurso r y formada por el resto de lugares de Z^r que quedan después de la eliminación de los lugares para formar la nueva zona $Z^{r'}$. Es decir, la nueva zona Z^r es igual a $Z^r \setminus (Z^r \cap Z^s)$. Obviamente esta nueva zona pertenece a los mismos conjuntos máximos de recursos a los que pertenecía la antigua zona Z^r pero no pertenecía la zona Z^s.

El primer lema que se presenta a continuación es análogo al lema 24 presentado para el primer tipo de transformación pero ahora referido al segundo tipo de transformación.

Lema 26. *Sea $\mathscr{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C}\rangle$ una red SOAR^2. Sea (Z^r, Z^s) un par de zonas que definen un arco del grafo de poda de zonas de $\mathscr{N}$ tales que $Fa(Z^r) \not\subseteq Z^s$. Sea $\mathscr{N}^L$ la red de Petri obtenida de $\mathscr{N}$ mediante los pasos descritos en el algoritmo 7 en el apartado "sino" correspondiente al caso $Fa(Z^r) \not\subseteq Z^s$. La red $\mathscr{N}^L$ es una red de Petri SOAR^2.*

Demostración. La forma en la que se ha conectado el recurso virtual r' a las transiciones de la red y las replicadas hace que tengamos un p–semiflujo mínimo binario formado por r', los lugares $Z^r \cap Z^s$ y los lugares replicados. Esto es trivial, si no se tiene en cuenta que se han eliminado los arcos de los lugares replicados que conectan con algún lugar de la zona Z^r que está en algún camino desde las transiciones de la frontera que no pertenecen a la nueva zona. De la misma manera se concluye con respecto al nuevo p–semiflujo de

r que sigue siendo binario y mínimo pero del que se han eliminado todos los lugares de $Z^r \cap Z^s$ y es un p–semiflujo gracias a que se han eliminado todos los arcos de lugares que no pertenecen a $Z^r \cap Z^s$ y que conectaban mediante algún camino con lugares de la frontera de la antigua Z^r que están en la nueva zona $Z^{r'}$.

La forma en la que se ha reconectado los recursos r y r' hacen que las nuevas zonas Z^r y $Z^{r'}$ mantengan las siguiente relaciones con el resto de la red:

A) $Z^{r'}$ pertenecerá a los mismos conjuntos máximos de zonas solapadas a las que pertenecían simultáneamente la antigua Z^r y la zona Z^s y esto es debido a que la nueva zona $Z^{r'}$ se define a partir de los lugares $Z^r \cap Z^s$ *(donde aquí se refiere a la zona antigua Z^r)*.

B) La nueva zona Z^r pertenecerá a los mismos conjuntos máximos de zonas solapadas a las que pertenecía la antigua zona Z^r pero no pertenecía la zona Z^s y esto es debido a que la zona Z^r actualizada es la antigua zona Z^r menos los lugares que pertenecen a $Z^r \cap Z^s$ luego la nueva zona Z^r no puede estar solapada con Z^s

De lo anterior se deduce que el grafo de poda de zonas de $\mathscr{N}$, los caminos máximos conteniendo la zona Z^r *(que corresponden a los conjuntos máximos de zonas solapadas)*, se particionan en dos grupos de caminos máximos.

a) Los antiguos caminos que contenían Z^r pero no contenían Z^s que ahora se mantienen de la misma forma en el grafo de poda de $\mathscr{N}^L$ con la única salvedad que la zona Z^r se refiere ahora a la nueva zona Z^r.

b) Los antiguos caminos máximos que contenían simultáneamente las zonas Z^r y Z^s que ahora contendrán las mismas zonas y arcos salvo en lo que concierne a Z^r que será ahora el nuevo vértice del grafo $Z^{r'}$.

Por lo tanto, todos los conjuntos máximos están completamente ordenados por la relación de precedencia y la red $\mathscr{N}^L$ es una red *SOAR*^{2}. □

El siguiente colorario se deriva directamente del resultado anterior y está referido al conjunto de zonas de la red modificada y el grafo de poda de zonas de la red.

Lema 27. *Sea $\mathscr{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ una red SOAR^2. Sea (Z^r, Z^s) un par de zonas que definen un arco del grafo de poda de zonas de $\mathscr{N}$ y sea $\mathscr{N}^L$ la red obtenida mediante la transformación apuntada en el lema 26 para la adición del recurso virtual r' con relación a la zona Z^r. El conjunto de zonas de $\mathscr{N}^L$ es el conjunto de zonas al que se le ha añadido la nueva zona $Z^{r'}$ como se define en el algoritmo 7, $Z^{r'} = Z^r \cap Z^s \cup \{p \mid p$* es un lugar proceso replicado y la zona Z^r se actualiza eliminando los lugares $Z^r \cap Z^s\}$. *El grafo de*

poda de zonas separa los caminos máximos que pasaban por el vértice Z^r en dos grupos: (1) los que pasaban por Z^s que ahora no pasan por Z^r y pasan por $Z^{r'}$; y (2) los que no pasaban por el vértice Z^s que siguen pasando por el vértice Z^r. Actualizándose las funciones S, L y K conforme a los especificado en el algoritmo 7.

Los resultados presentados a continuación trasladan las transformaciones de la red de Petri presentadas anteriormente y del grafo de poda de zonas, a nivel del grafo de poda de recursos.

Este es nuestro objetivo final dado que el método de corrección está basado en hacer acíclico este grafo por transformaciones en la red de Petri original.

El primer resultado establece las condiciones que garantizan que un arco (r,s) del grafo de poda de recursos desaparece del grafo de poda de recursos correspondiente a la red de Petri transformada según los procedimientos anteriores.

Lema 28. *Sea $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C}\rangle$ una red SOAR^2. Sea (r,s) un arco del grafo de poda de recursos de $\mathcal{N}$. Si para cada par de zonas (Z^r, Z^s) que da origen al arco (r,s), se aplican las transformaciones de la red de Petri especificadas en los lemas 24 y 26, cumpliéndose en cada uno de estos pasos que Z^r es una zona máxima en todos los conjuntos máximos de zonas solapadas a las que pertenece, según la relación de precedencia, entonces.*

a) Cada uno de los vértices nuevos r' del grafo de poda de recursos correspondiente a cada nuevo recurso virtual asociado a la nueva zona $Z^{r'}$ derivada del par (Z^r, Z^s), no tiene ningún arco de entrada en dicho grafo.

b) En el grafo de poda de recursos de la red transformada no existe ninún circuito que contenga simultáneamente los recursos r y s o bien simultáneamente los recursos r' y s, para todo r' añadido.

□

Demostración. Razonamos en primer lugar para un par de zonas (Z^r, Z^s). Si Z^r es una zona máxima para todo conjunto máximo de zonas solapadas, cualquiera de ambas transformaciones mantiene la estructura de caminos máximos en el grafo de poda de zonas. En la segunda de las transformaciones lo único que cambia es que los caminos máximos que originalmente pasaban por Z^r ahora se han particionado en dos grupos, los que siguen pasando por Z^r y los que pasan por $Z^{r'}$. En ambos casos, en el grafo de poda de zonas, debido a que Z^r es un máximo para todo conjunto máximo de zonas solapadas, el vértice $Z^{r'}$ no tiene ningún arco predecesor y dado que $Z^{r'}$ es la única zona existente asociada al recurso r', al aplicar el método de aglomeración de la definición 19 del capítulo 3, en el

grafo de poda de recursos el nuevo vértice r' no tendrá ningún arco de entrada. Como esto sucede para todo par de zonas (Z^r, Z^s) que da origen al arco (r,s) del grafo de poda de recursos, no existirá ningún circuito en este nuevo grafo que contenga simultáneamente al vértice s y a uno de estos nuevos recursos virtuales r'.

Tanto en la primera como en la segunda transformación de la red de Petri, no tenemos ningún arco (Z^r, Z^s) que permanezca después de la transformación. En efecto, en las primeras de las transformaciones porque por cada par (Z^r, Z^s) desaparece la zona Z^r y por lo tanto no nos quedará ningún par (Z^r, Z^s) que pueda dar origen al arco (r,s) en el grafo de poda de recursos. En la segunda transformación si que permanece una zona Z^r para cada par (Z^r, Z^s) aunque ha sido actualizada convenientemente y lo que hemos visto es que en el grafo de poda de zonas los únicos caminos máximos que atraviesan el vértice actualizado Z^r son precisamente los que no pertenecen a Z^s. Por lo tanto, también en este caso nos encontramos con que al aplicar el procedimiento de aglomeración de la definición 19 del capítulo 3, tampoco puede aparecer un arco (r,s). Es decir, tampoco existirán circuitos en el grafo de poda de recursos de la red transformada que contengan simultáneamente los vértices r y s, □

Parece que el resultado anterior impone condiciones extraordinariamente estrictas para eliminar un arco (r,s) del grafo de poda de recursos en el sentido que la posibilidad de que se de esta circunstancia se vea como remota.

El siguiente resultado justifica el procedimiento implementado en el algoritmo 7 para aquellos casos en los que en algún par (Z^r, Z^s) que de origen al arco (r,s) que se está tratando de eliminar, la zona Z^r no sea máxima por algún conjunto máximo de zonas solapadas al que pertenece.

Lema 29. *Sea $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ una red SOAR^2. Sea (Z^r, Z^s) un par de zonas que da origen al arco (r,s) del grafo de poda de recursos que quiere ser eliminado, tal que Z^r no es una zona máxima para algún conjunto máximo de zonas solapadas de $\mathcal{N}$. Si se aplican las transformaciones especificadas en los lemas 24 ó 26 se cumplirá que el vértice r' del grafo de poda de recursos correspondiente al nuevo recurso virtual asociado a la nueva zona $Z^{r'}$ derivada del par (Z^r, Z^s), tendrá al menos un arco (x, r') que tendrá su origen en un único par de zonas $(Z^x, Z^{r'})$ que cumplen que $Z^{r'} < Z^x$.* □

Demostración. Si (Z^r, Z^s) es un par de zonas tales que Z^r no es una zona máxima en algún conjunto máximo de zonas solapadas, entonces en ese conjunto existe una zona Z^x tal que $Z^r < Z^x$. Como el conjunto máximo de zonas solapadas está totalmente ordenado por la relación de precedencia, y teniendo en cuenta que el grafo de poda de zonas con el que trabajamos es "simple", entonces la zona anterior Z^x es única en ese conjunto máximo de zonas solapadas. Cualquiera de los métodos de transformación mantienen caminos que

incluyen $Z^{r'}$ y Z^s, y por lo tanto el arco $(Z^x, Z^{r'})$ existirá en el grafo de poda de zonas que al aplicar el procedimiento de aglomeración a la red transformada nos permitirá obtener el arco (x, r'), ahora bien no existirá ningún otro par de zonas $(Z_b^x, Z_b^{r'})$ que de origen a este arco dado que para r' solo existe una zona $Z^{r'}$ y solo existe un arco conectando una y solo una zona de x con la zona r'. □

A partir del resultado anterior, resulta evidente que el procedimiento implantado en el algoritmo 7 nos permitirá eliminar todos los circuitos que contengan simultáneamente los recursos r y s *(es decir, pretendemos eliminar el arco (r,s) del grafo de poda de recursos)* aunque la estrategia es diferente según podamos aplicar el lema 28 o el lema 29.

Efectivamente, en el caso del lema 28 la estrategia aplicada consigue eliminar de forma efectiva el arco (r, s) y además aunque tenemos el arco (r', s) donde r' representa a cada uno de los recursos virtuales añadidos, garantizamos que estos vértices r' no tienen ningún arco de entrada por lo tanto todos los circuitos que pudieran contener los recursos s y uno de los recursos r ó r' no pueden existir. La segunda estrategia *(lema 29)* no garantiza este último apartado y por alguno de estos recursos virtuales r' podemos tener algún arco de entrada que mantuviese algún circuito conteniendo r' y s en el grafo de poda de recursos. En este caso, y dado que este arco de entrada a r', arco (x, r') tiene su origen en un único par de zonas, $(Z^x, Z^{r'})$ el procedimiento es extendido haciendo que este par de zonas sea también un arco a eliminar por alguno de los procedimientos de transformación de la red de Petri. Esta forma de proceder termina, dado que los nuevos pares de zonas que se añaden al conjunto R del algoritmo 7 que hay que eliminar son finitos ya que terminarán cuando alcancemos las zonas máximas de los conjuntos máximos de zonas solapadas. Los comentarios anteriores se resumen en los resultados siguientes.

Teorema 7. *Sea $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ una red SOAR^2. El algoritmo 7 aplicado a la red $\mathcal{N}$ termina en el número de iteraciones del orden del número de arcos del grafo de poda de $\mathcal{N}$.*

Demostración. Inicialmente el conjunto R contiene un número finito y no vacío de arcos del grafo de poda de zonas. En cada iteración se procesa uno de estos arcos que se elimina del conjunto, R y cuya cardinalidad define el número de iteraciones que ejecuta el algoritmo 7. Si en la eliminación, tratamiento de un par de zonas (Z^r, Z^s) se verifica que Z^r no es una zona máxima por algún conjunto máximo de zonas solapadas, se añaden al conjunto, R todos los pares (Z^x, Z^r) que aparecen en el grafo de poda de zonas de R. Dado que el grafo de poda de zonas es acíclico y en la forma en que se añaden estos nuevos arcos al conjunto R siempre se procede en el mismo sentido seleccionando arcos de entrada de vértices del grafo de poda de zonas, resulta evidente que nunca procesaremos más de una vez un par de zonas y en el peor de los casos deberemos procesar todos los arcos del grafo de poda de zonas. □

De acuerdo con el resultado anterior, en particular su demostración, el procesamiento siempre se culmina en pares (Z^r, Z^s) donde Z^r es una zona máxima para todos los conjuntos máximos de zonas solapadas a las que pertenece. Es decir, podemos aplicar el lema 28 que es el que garantiza la desaparición sistemática de circuitos en el grafo de poda de zonas de cada una de las redes transformadas *(que siempre serán de la clase SOAR^2 ya que las transformaciones realizadas preservan la pertenencia a la clase)*.

El resultado siguiente demuestra el objetivo perseguido en la estrategia planteada para la corrección de la red de Petri transformándola de manera que el grafo de poda de recursos resultante sea acíclico.

Teorema 8. *Sea $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ una red SOAR^2 con marcado inicial admisible m_0. El algoritmo 7 aplicado a la red de Petri $\mathcal{N}$, permite obtener una red de Petri $\mathcal{N}^L$ de la clase SOAR^2 con marcado inicial admisible m_0^L y para la que el grafo de poda de recursos de $\mathcal{N}^L$ es acíclico.*

Demostración. La red resultante N^L es de la clase *SOAR^2* ya que ambos tipos de transformación de la red hacen que el resultado pertenezca a la clase *SOAR^2 (lemas 24 y 26)*. El marcado inicial m_0^L es admisible ya que el marcado inicial de los lugares reposo no se ve alterado por el algoritmo, y cada nuevo lugar recurso virtual que se añade a la red contiene una única marca.

Finalmente, el grafo de poda de recursos que se obtiene es acíclico dado que se seleccionó un conjunto de arcos de este grafo de forma que si se eliminaban el grafo resultante era acíclico. Estos arcos (Z^r, Z^s) han sido seleccionados porque evidentemente existía algún circuito que incluía simultáneamente a los vértices r y s. Los procedimientos de transformación conseguían eliminar estos arcos del grafo de poda de recursos como pone de manifiesto el lema 24 sin introducir circuitos adicionales como consecuencia de la adición de los recursos virtuales. En el caso que no se pudiese aplicar el lema 24 directamente debido a que los nuevos recursos virtuales introducen nuevos circuitos, entonces se procede a eliminar estos nuevos circuitos mediante el procesamiento de los arcos de entrada a la nueva zona creada asociada al recurso virtual. Este proceso termina porque el grafo de poda de zonas es acíclico y al final en todas las ramas cubiertas en la propagación hacia atras en el grafo de poda de zonas acabará siendo de aplicación el lema 24. Por lo tanto, todos los circuitos originales habrían sido eliminados y ningún circuito nuevo aparecerá como consecuencia de la introducción de los nuevos recursos virtuales □

Finalmente, para terminar el siguiente resultado demuestra que la red de Petri alcanzada tras la aplicación del algoritmo 7 es una red de Petri de la clase *SOAR^2* y viva.

Teorema 9. *Sea $\mathcal{N} = \langle P_0 \cup P_S \cup P_R, T, \mathbf{C} \rangle$ una red SOAR^2 con marcado inicial admisible m_0. Tras la aplicación del algoritmo 7 la red $\mathcal{N}^L$ con el marcado inicial admisible m_0^L es viva.*

Demostración. Por el teorema 8 el grafo de poda de recursos de la red $\mathscr{N}^L$ es acíclico por lo tanto no existen cerrojos que contengan más de un recurso que sean mínimos. Es decir, no existen cerrojos malos [Cano 11]. Por lo tanto, como la red está inicialmente marcada con un marcado inicial admisible por el teorema 2 del capítulo 1 la red es viva. □

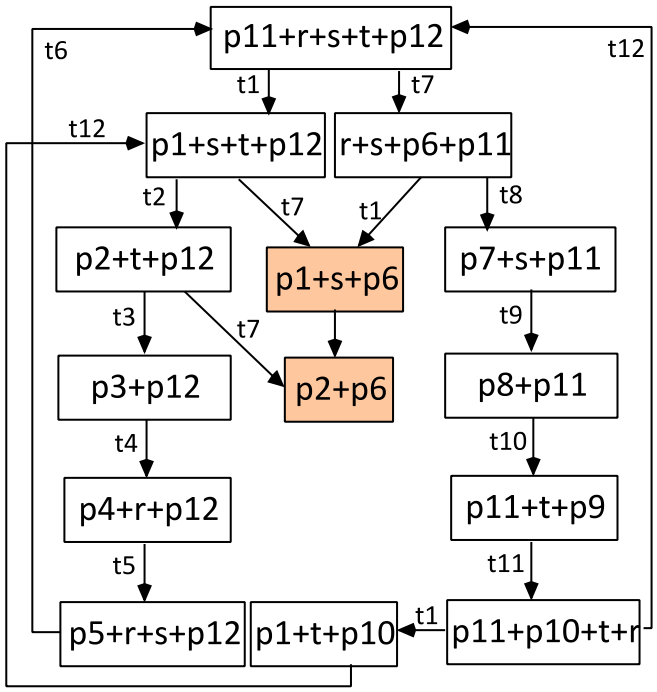

Figura 4.19: Grafo de marcados de la red de Petri de la figura 4.10.

Para terminar ilustraremos mediante un ejemplo, la red de Petri presentada en la figura 4.10, el efecto que tiene el método de corrección propuesto sobre el espacio de estados de la red de Petri no viva que inicialmente proporciona el diseñador. En efecto, en la figura 4.19 se presenta el grafo de marcados de la red *SOAR*^{2} presentada en la figura 4.10, que como se puede apreciar presenta un estado de bloqueo total correspondiente al marcado $p_2 + p_6$. En la figura 4.20 se presenta el grafo de marcados de la red de Petri de la figura 4.17 que es la red de Petri corregida con el método propuesto en este capítulo de la red de Petri de la figura 4.10. Obsérvese, que en este grafo de marcados no existe ningún estado de bloqueo y todas las transiciones son vivas *(esto se puede demostrar viendo el grafo de marcados fuertemente conexo y todas las transiciones aparecen en al menos un arco del grafo de marcados)*.

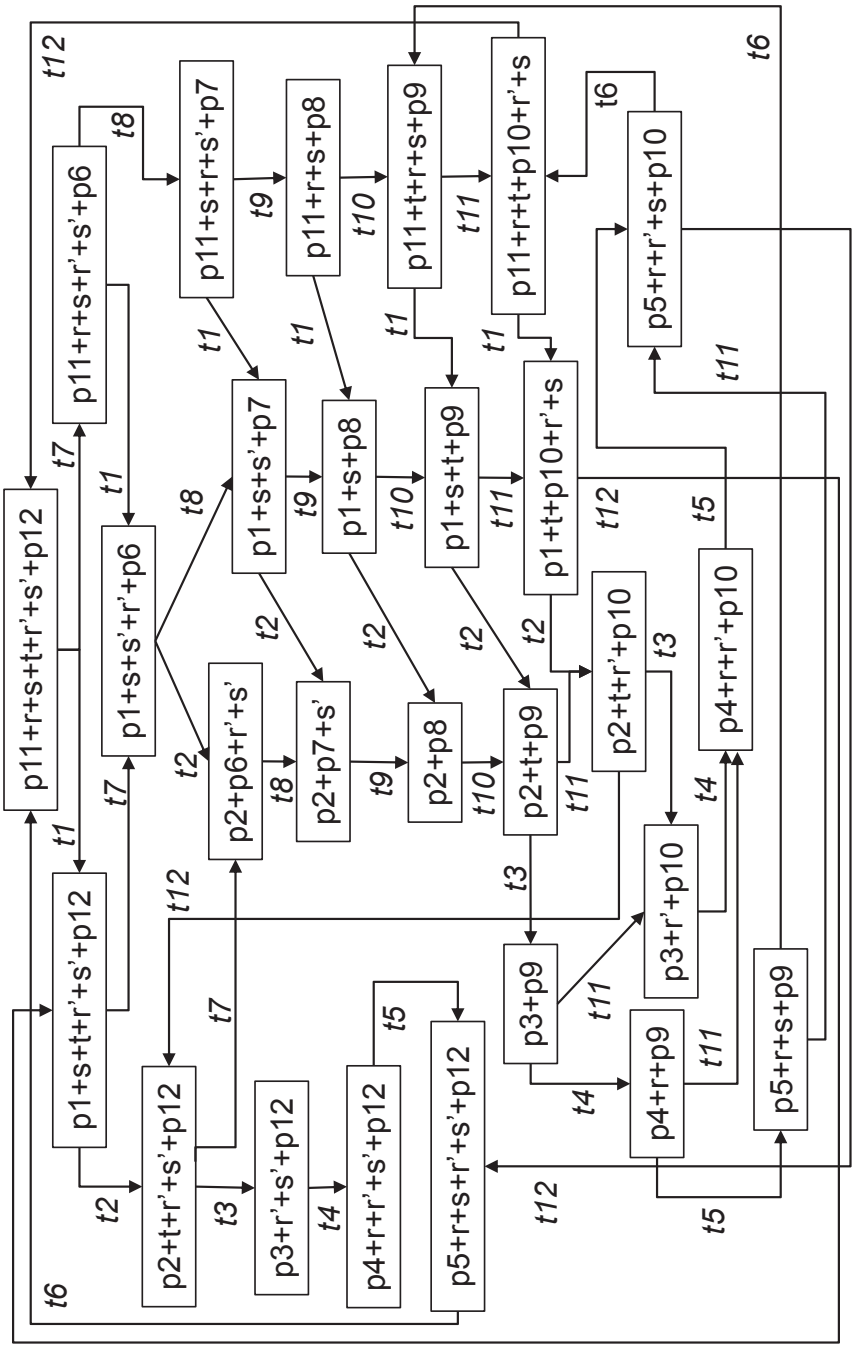

Figura 4.20: Grafo de marcados de la red de Petri de la figura 4.17.

El método de corrección ha introducido dos recursos virtuales nuevos, el r' y el s', lo cual ha producido los siguientes efectos observables.

a) Todos los marcados que teníamos en la red de Petri sin corregir aparecen en el grafo de marcados de la red de Petri corregida, aunque con un número mayor de recursos disponibles debido a la introducción de los nuevos recursos virtuales r' y s'.

b) Se introducen nuevos marcados alcanzables en la red de Petri corregida que hacen que la red de Petri corregida sea viva.

c) Los recursos virtuales introducidos son los mínimos imprescindibles para hacer la red viva, pues como se puede apreciar el marcado alcanzable $p_2 + p_8$ o en el marcado $p_3 + p_9$ todos los recursos inicialmente disponibles (r, s, t, r', s') se encuentran en uso *(no hay recursos libres en los dos marcados señalados)*.

d) Se ha incrementado la concurrencia en la red de Petri. Es decir, en la red de Petri corregida es posible disparar transiciones concurrentemente, que en la red de Petri original antes no era posible. Por ejemplo, ahora las transiciones t_2 y t_9 ó t_1 y t_{10} pueden llegar a dispararse concurrentemente.

Como comentario final, a través de este ejemplo queda patente que el método de corrección propuesto no se basa en las técnicas de prevención habituales que persiguen prohibir el conjunto de estados malos de la red *(en el ejemplo anterior se trataría de cortar los dos estados $p_1 + s + p_6$ y $p_2 + p_6$: el primero es malo porque inevitablemente conduce a bloqueo y el segundo porque es un estado de bloqueo total de la red.)* Esto podría haberse hecho pero si se observa el grafo de marcados que se obtiene después de prohibir estos dos estados es completamente secuencial, es decir, primero se ejecuta una máquina de estados y hasta que no se termina no se puede empezar ninguna acción de la otra máquina de estados, *(Salvo el estado $p_{11} + r + t + p_{10}$ que antes de acabar la máquina de estados segunda se puede disparar la primera acción de la máquina de estados primera)*.

Por el contrario, el método aquí propuesto se trata de introducir recursos virtuales extra pero de uso privado por los procesos cuando estos se encuentran en los estados (lugares) asociados a la zona de uso continuado de recursos que sirvió para introducir el nuevo recurso virtual. Cuando se dice que se introduce un nuevo recurso virtual pero de uso privado, queremos decir lo siguiente,

a) Que no se está incrementando el contenido de marcas de un lugar recurso para que desaparezca el bloqueo. Si esto se hiciese así en el ejemplo anterior se observaría que las causas estructurales de la aparición del bloqueo permanecerían y para manifestarse bastaría con incremetar el marcado de los lugares reposo a un valor suficientemente alto para que volviesen a aparecer problemas de bloqueos.

b) Que la técnica si que incrementa el número de recursos, pero lo hace de forma que estos recursos no están compartidos entre los mismos usuarios que antes. Estos recursos se usan privadamente por solo algunos estados del sistema. Obsérvese en el ejemplo que antes el recurso r podía ser usado en los lugares p_1, p_2, p_3 y p_7, p_8, p_9. Al introducir el nuevo recurso r' hemos hecho que la antigua copia de r sea usada privadamente en los estados p_1, p_2 y p_3 mientras que el nuevo recurso virtual solo se usa en los estados p_7, p_8 y p_9 *(uso privado para estos estados)*.

c) Que la técnica supone un corrección estructural profunda ya que para todo marcado inicial admisible *(incluso incrementando el marcado de los lugares reposo a una cantidad arbitrariamente grande)* la red seguirá siendo viva. Esto es debido a que se ha hecho desaparecer todos los cerrojos malos de la red que eran la causas estructurales que estaban detrás de la aparición del bloqueo de mensajes en esta red.

El método evidentemente puede ser aplicado en el dominio de la redes de interconexión para el encaminamiento de mensajes ya que los recursos virtuales introducidos tienen una implementación fácil a través del concepto de canal virtual sobre un canal físico, siempre que el ancho de banda de este último lo permita. Pero sobre todo es posible implementarlo porque la introducción del recurso virtual se realiza de tal manera que respeta el principio de localidad necesario en estos sistemas distribuidos de alto débito en el que no es posible el intercambio de información entre los algoritmos de encaminamiento para tomar decisiones coordinadas sobre recursos de uso global entre procesos geográficamente dispersos.

En la sección siguiente se desarrolla un ejemplo más elaborado de una red de interconexión para el encaminamiento de mensajes de acuerdo con un algoritmo concreto suministrado por el diseñador. Se aplica la metodología completa desarrollada en esta tesis para corregir los modelos mediante la introducción de canales virtuales, para finalmente dar algunas pautas de como trasladar las correcciones introducidas a nivel de modelo al algoritmo concreto suministrado por el diseñador.

4.6. Ejemplo en el contexto de las redes de interconexión

En esta sección abordaremos la corrección de la red *SOAR*^{2} que corresponde al ejemplo de la red de interconexión presentado en la sección 2.2, utilizando la caracterización de la vivacidad presentada en el teorema 2, del capítulo 1 y el grafo de poda de recursos [Cano 11] como herramienta de corrección. A través del grafo de poda de recursos podemos determinar si existen cerrojos malos en la red debido a la existencia de ciclos que se eliminan mediante la introducción de recursos virtuales que en este

contexto denominamos *canales virtuales*. Es preciso mencionar que en las redes de interconexión existe un grado de flexibilidad adicional porque éstas utilizan algoritmos de encaminamiento para la asignación y liberación de los recursos, por lo tanto es posible modificar la estructura de los procesos a nivel lógico. Esta flexibilidad no existe en muchos otros dominios de aplicación en donde para cambiar los procesos se necesita realizar cambios profundos y costosos en la arquitectura física del sistema o su entorno de trabajo que en la mayoría de los casos es una solución inviable.

4.6.1. Redes de interconexión

La red de Petri de la figura 2.8 es el modelo que corresponde al ejemplo presentado en la sección 2.2 en donde existe una red de interconexión que está definida por un sistema de nodos *ND* y un sistema de canales *CH* que interconectan los nodos. La topología de la red es un anillo unidireccional en donde *ND*=$\{n_0, n_1, n_2, n_3\}$ son los nodos de la red y están interconnectados por un conjunto de canales *CH*=$\{ca_0, ca_1, ca_2, cl_1, cl_2, cl_3\}$ que son los recursos del sistema como se muestra en la figura 2.2.a. Esta red utiliza el algoritmo de encaminamiento 1 de tipo adaptativo mínimo para encaminar los mensajes desde un nodo origen a un nodo destino, sin embargo este algoritmo presenta problemas de bloqueos, como se mostró en la configuración del cuadro 2.1. y que se solucionará en esta sección a través de nuestra metodología y la utilización de recursos virtuales. Es preciso recordar que trabajamos con control de flujo tipo wormhole en donde los paquetes son divididos en pequeñas unidades que fluyen de forma consecutiva $(pipeline\ fashion)$ y por este motivo pueden ocupar simultáneamente multiples canales de comunicación.

Utilizaremos nuestro modelo en redes de Petri de la red de interconexión, obtenido utilizando nuestra metodología para analizar y sintetizar este modelo por medio de nuestro método de corrección. Procederemos siguiendo los pasos establecidos para el método de corrección en la sección 4.4 más una explicación sobre el nuevo algoritmo de encaminamiento libre de bloqueos. El primer paso de este método consiste en la obtención del grafo de poda completo de zonas de la definición 17 que se muestra en la figura 4.21 y corresponde a la red de la figura 2.8. En este grafo podemos ver que existen cuatro componentes conexas *a,b,c,d* que corresponden a cada una de las máquinas de estado de la red *SM1,SM2,SM3,SM4* que a su vez representan los diversos nodos destinos n_0, n_1, n_2 y n_3 de los mensajes respectivamente *(en las referencias al nodo se omitirá la letra n para mayor claridad)*. Es decir, una componente conexa que corresponde al conjunto máximo $\{Z_{3,1}^{cl_3}; Z_{3,1}^{ca_0}; Z_{3,1}^{ca_1}\}$ y sobre el que existe una cadena ordenada de zonas $\{Z_{3,1}^{cl_3} < Z_{3,1}^{ca_0} < Z_{3,1}^{ca_1}\}$ correspondería a las cadenas ordenadas de zonas solapadas de recursos necesarios para que un mensaje alcance el nodo destino 2 desde el nodo origen 3.

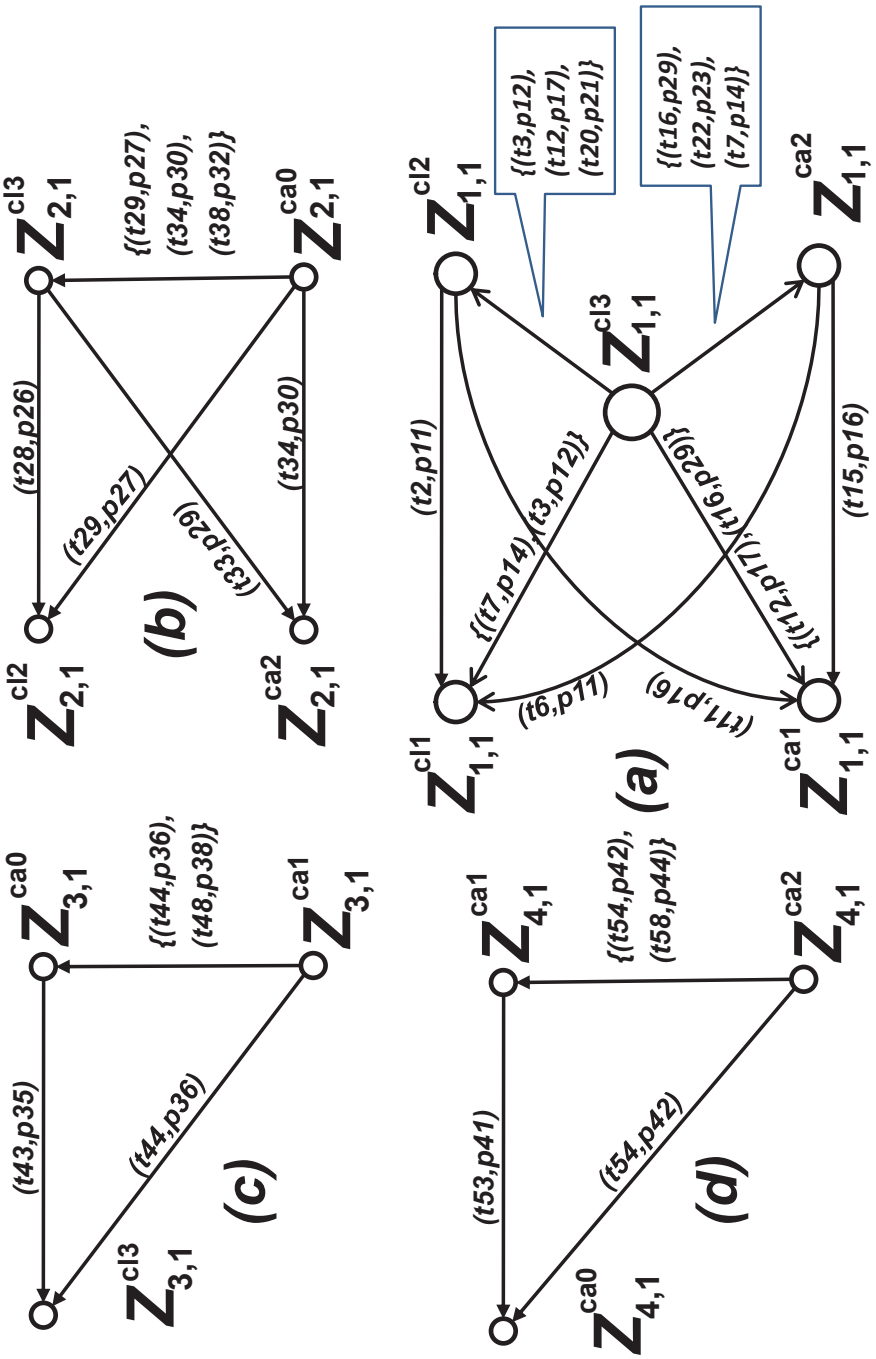

Figura 4.21: Grafo de poda completo de la red de Petri de la figura 2.8 con sus funciones de etiquetado *L*.

La decisión de asignar estos recursos es determinada por el algoritmo de encaminamiento 1, de la red de interconexión estableciendo relaciones entre los recursos del sistema que se resume en las funciones del cuadro 4.2. En el grafo 4.21 se han incluido las etiquetas de los arcos dadas por la función L para cada arco con el conjunto de pares (t,p) representando un lugar candidato a ser podado cuando construyamos el cerrojo con los recursos indicados en los vértices del grafo. En el cuadro 4.2 se ha incluido la función S, que es una función de identidad de cada zona y la función K_G que indica los lugares que resultan ser no esenciales cuando construimos un cerrojo de esta zona, sin embargo existirán zonas que no tienen lugares para podar como la zona $K_G(Z_{1,1}^{cl_3})$=0.

Funciones de etiquetado S de las zonas en cada máquina de estados	
$S(Z_{3,1}^{cl_3})=Z_{3,1}^{cl_3}=\{p_{35},p_{36},p_{37}\}$ $S(Z_{3,1}^{ca_0})=Z_{3,1}^{ca_0}=\{p_{36},p_{37},p_{38},p_{39}\}$ $S(Z_{3,1}^{ca_1})=Z_{3,1}^{ca_1}=\{p_{37},p_{39},p_{40}\}$	$S(Z_{4,1}^{ca_0})=Z_{4,1}^{ca_0}=\{p_{41},p_{42},p_{43}\}$ $S(Z_{4,1}^{ca_1})=Z_{4,1}^{ca_1}=\{p_{42},p_{43},p_{44},p_{45}\}$ $S(Z_{4,1}^{ca_2})=Z_{4,1}^{ca_2}=\{p_{43},p_{45},p_{46}\}$
$S(Z_{2,1}^{cl_2})=Z_{2,1}^{cl_2}=\{p_{26},p_{27},p_{28}\}$ $S(Z_{2,1}^{ca_2})=Z_{2,1}^{ca_2}=\{p_{29},p_{30},p_{31}\}$ $S(Z_{2,1}^{cl_3})=Z_{2,1}^{cl_3}=\{p_{27},p_{28},p_{30},p_{31},p_{32},p_{33}\}$ $S(Z_{2,1}^{ca_0})=Z_{2,1}^{ca_0}=\{p_{28},p_{31},p_{32},p_{34}\}$	$S(Z_{1,1}^{cl_1})=Z_{1,1}^{cl_1}=\{p_{11},p_{12},p_{13},p_{14},p_{15}\}$ $S(Z_{1,1}^{ca_1})=Z_{1,1}^{ca_1}=\{p_{16},p_{17},p_{18},p_{19},p_{20}\}$ $S(Z_{1,1}^{cl_2})=Z_{1,1}^{cl_2}=\{p_{12},p_{13},p_{17},p_{18},p_{21},p_{22}\}$ $S(Z_{1,1}^{ca_2})=Z_{1,1}^{ca_2}=\{p_{14},p_{15},p_{19},p_{20},p_{23},p_{24}\}$ $S(Z_{1,1}^{cl_3})=Z_{1,1}^{cl_3}=\{p_{13},p_{15},p_{18},p_{20},p_{22},p_{24},p_{25}\}$
Funciones K_G de las zonas de los recursos en cada máquina de estados	
$K_G(Z_{3,1}^{cl_3})=Z_{3,1}^{cl_3}\setminus Z_{3,1}^{ca_0}=\{p_{35}\}$ $K_G(Z_{3,1}^{ca_0})=Z_{3,1}^{ca_0}\setminus Z_{3,1}^{ca_1}=\{p_{36},p_{38}\}$ $K_G(Z_{3,1}^{ca_1})=\emptyset$	$K_G(Z_{4,1}^{ca_0})=Z_{4,1}^{ca_0}\setminus Z_{4,1}^{ca_1}=\{p_{41}\}$ $K_G(Z_{4,1}^{ca_1})=Z_{4,1}^{ca_1}\setminus Z_{4,1}^{ca_2}=\{p_{42},p_{44}\}$ $K_G(Z_{4,1}^{ca_2})=\emptyset$
$K_G(Z_{2,1}^{cl_2})=Z_{2,1}^{cl_2}\setminus Z_{2,1}^{cl_3}=\{p_{26}\}$ $K_G(Z_{2,1}^{ca_2})=Z_{2,1}^{ca_2}\setminus Z_{2,1}^{cl_3}=\{p_{29}\}$ $K_G(Z_{2,1}^{cl_3})=Z_{2,1}^{cl_3}\setminus Z_{2,1}^{ca_0}=\{p_{27},p_{30},p_{32}\}$ $K_G(Z_{2,1}^{ca_0})=\emptyset$	$K_G(Z_{1,1}^{cl_1})=Z_{1,1}^{cl_1}\setminus Z_{1,1}^{cl_2}=\{p_{11}\}$ $K_G(Z_{1,1}^{cl_1})=Z_{1,1}^{cl_1}\setminus Z_{1,1}^{ca_2}=\{p_{11}\}$ $K_G(Z_{1,1}^{cl_2})=Z_{1,1}^{cl_2}\setminus Z_{1,1}^{cl_3}=\{p_{12},p_{17},p_{21}\}$ $K_G(Z_{1,1}^{ca_2})=Z_{1,1}^{ca_2}\setminus Z_{1,1}^{cl_3}=\{p_{14},p_{19},p_{23}\}$ $K_G(Z_{1,1}^{cl_3})=\emptyset$ $K_G(Z_{1,1}^{ca_1})=Z_{1,1}^{ca_1}\setminus Z_{1,1}^{cl_2}=\{p_{16}\}$ $K_G(Z_{1,1}^{ca_1})=Z_{1,1}^{ca_1}\setminus Z_{1,1}^{ca_2}=\{p_{16}\}$

Cuadro 4.2: Funciones de etiquetado S y K_G del grafo de poda completo de la figura 4.21.

Las relaciones entre las zonas de recursos pueden ser *directas* e *indirectas*, en donde los vértices adyacentes del grafo de poda completo establecen relaciones directas entre las zonas que se solapan y los vértices no adyacentes y conexos establecen relaciones indirectas entre estas zonas que se solapan como las zonas $Z_{3,1}^{cl_3}$ y $Z_{3,1}^{ca_1}$ en la figura 4.21.c. Se ha explicado que las relaciones indirectas no son necesarias para construir el grafo de poda de cerrojos conteniendo un único recurso, por tal motivo se remueven estas relaciones indirectas como se muestra en el grafo de poda simple de la figura 4.22 que se definió en la definición 18. Este grafo de poda simple se obtiene del Grafo de poda completo de la figura 4.21, removiendo todas las relaciones indirectas entre las zonas, como la mencionada anteriormente.

Este cambio influye en las funciones S, L y K_G que se muestran el el cuadro 4.3, en donde las funciones L están organizadas por las distintas máquinas de estado. Contrariamente, en las funciones S y K_G del cuadro 4.3, los lugares obtenidos de estas funciones son el resultado de la unión de las diferentes máquinas de estado. Es preciso hacer notar que en base al lema 9.2, la unión de todas las zonas asociadas a un recurso r más el recurso r es igual al soporte del p–semiflujo del recurso como se muestra con la función S del cuadro 4.3. El grafo de poda simple será determinate para encontrar los puntos estratégicos en donde aplicar las correcciones al modelo de la red de Petri mediante la introducción de canales virtuales debido a que podemos determinar la posición menos invasiva para agregar los canales virtuales en base a determinadas heurísticas que toman en consideración las zonas a corregir.

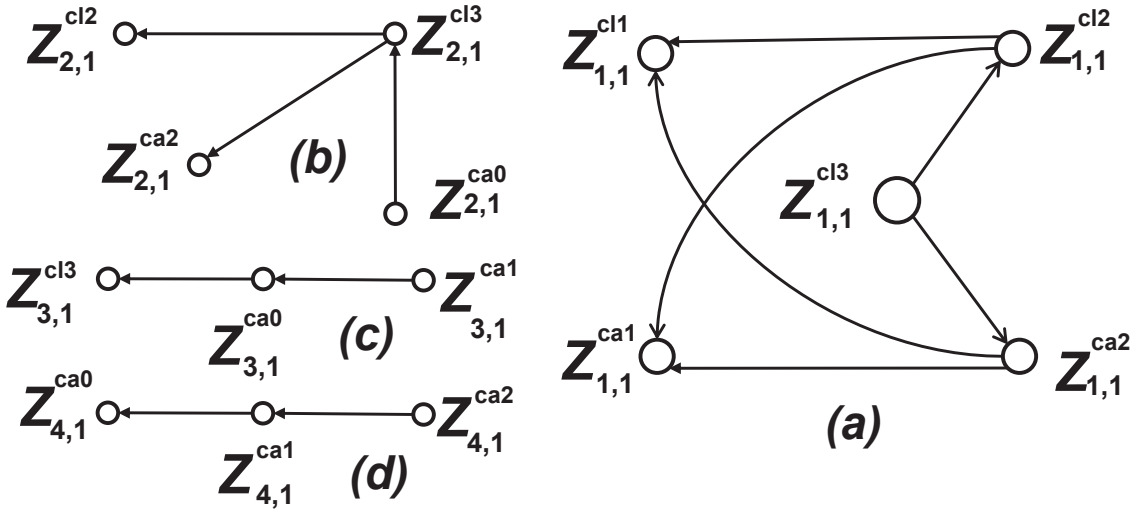

Figura 4.22: Grafo de poda simple de la red de Petri de la figura 2.8.

Funciones de etiquetado S, L y K_G de la figura 4.22
$S(ca_0) = \{ca_0\} \cup Z^{ca_0}_{2,1} \cup Z^{ca_0}_{3,1} \cup Z^{ca_0}_{4,1} = \{ca_0, p_{28}, p_{31}, p_{33}, p_{34}, p_{36}, p_{37}, p_{38}, p_{39}, p_{41}, p_{42}, p_{43}\} = \|\mathbf{y}_{ca_0}\|$.
$S(ca_1) = \{ca_1\} \cup Z^{ca_1}_{1,1} \cup Z^{ca_1}_{3,1} \cup Z^{ca_1}_{4,1} = \{ca_1, p_{16}, p_{17}, p_{18}, p_{19}, p_{20}, p_{37}, p_{39}, p_{40}, p_{42}, p_{43}, p_{44}, p_{45}\} = \|\mathbf{y}_{ca_1}\|$
$S(ca_2) = \{ca_2\} \cup Z^{ca_2}_{1,1} \cup Z^{ca_2}_{2,1} \cup Z^{ca_2}_{4,1} = \{ca_2, p_{14}, p_{15}, p_{19}, p_{20}, p_{23}, p_{24}, p_{29}, p_{30}, p_{31}, p_{43}, p_{45}, p_{46}\} = \|\mathbf{y}_{ca_2}\|$
$S(cl_1) = \{cl_1\} \cup Z^{cl_1}_{1,1} = \{cl_1, p_{11}, p_{12}, p_{13}, p_{14}, p_{15}\} = \|\mathbf{y}_{cl_1}\|$
$S(cl_2) = \{cl_2\} \cup Z^{cl_2}_{1,1} \cup Z^{cl_2}_{2,1} = \{cl_2, p_{12}, p_{13}, p_{17}, p_{18}, p_{21}, p_{22}, p_{26}, p_{27}, p_{28}\} = \|\mathbf{y}_{cl_2}\|$
$S(cl_3) = \{cl_3\} \cup Z^{cl_3}_{1,1} \cup Z^{cl_3}_{3,1} = \{cl_3, p_{13}, p_{15}, p_{18}, p_{20}, p_{22}, p_{24}, p_{25}, p_{27}, p_{28}, p_{30}, p_{31}, p_{32}, p_{33}, p_{35}, p_{36}, p_{37}\} = \|\mathbf{y}_{cl_3}\|$
$L((ca_1, ca_0)) = L((Z^{ca_1}_{3,1}; Z^{ca_0}_{3,1})) = \{(t_{44}, p_{36}), (t_{48}, p_{38})\}$
$L((ca_0, cl_3)) = L((Z^{ca_0}_{3,1}; Z^{cl_3}_{3,1})) = \{(t_{43}, p_{35})\}$
$L((ca_2, ca_1)) = L((Z^{ca_2}_{4,1}; Z^{ca_1}_{4,1})) = \{(t_{54}, p_{42}), (t_{58}, p_{44})\}$
$L((ca_1, ca_0)) = L((Z^{ca_1}_{4,1}; Z^{ca_0}_{4,1})) = \{(t_{53}, p_{41})\}$
$L((ca_0, cl_3)) = L((Z^{ca_0}_{2,1}; Z^{cl_3}_{2,1})) = \{(t_{29}, p_{27}), (t_{34}, p_{30}), (t_{38}, p_{32})\}$
$L((cl_3, ca_2)) = L((Z^{cl_3}_{2,1}; Z^{ca_2}_{2,1})) = \{(t_{33}, p_{29})\}$
$L((cl_3, cl_2)) = L((Z^{cl_3}_{2,1}; Z^{cl_2}_{2,1})) = \{(t_{28}, p_{26})\}$
$L((cl_3, cl_2)) = L((Z^{cl_3}_{1,1}; Z^{cl_2}_{1,1})) = \{(t_3, p_{12}), (t_{12}, p_{17}), (t_{20}, p_{21})\}$
$L((cl_3, ca_2)) = L((Z^{cl_3}_{1,1}; Z^{ca_2}_{1,1})) = \{(t_7, p_{14}), (t_{16}, p_{19}), (t_{22}, p_{23})\}$
$L((ca_2, cl_1)) = L((Z^{ca_2}_{1,1}; Z^{cl_1}_{1,1})) = \{(t_6, p_{11})\}$
$L((ca_2, ca_1)) = L((Z^{ca_2}_{1,1}; Z^{ca_1}_{1,1})) = \{(t_{15}, p_{16})\}$
$L((cl_2, ca_1)) = L((Z^{cl_2}_{1,1}; Z^{ca_1}_{1,1})) = \{(t_{11}, p_{16})\}$
$L((cl_2, cl_1)) = L((Z^{cl_2}_{1,1}; Z^{cl_1}_{1,1})) = \{(t_2, p_{11})\}$
$K_G(ca_0) = K_G(Z^{ca_0}_{2,1}) \cup K_G(Z^{ca_0}_{3,1}) \cup K_G(Z^{ca_0}_{4,1}) = \{p_{36}, p_{38}, p_{41}\}$
$K_G(ca_1) = K_G(Z^{ca_1}_{1,1}) \cup K_G(Z^{ca_1}_{3,1}) \cup K_G(Z^{ca_1}_{4,1}) = \{p_{16}, p_{42}, p_{44}\}$
$K_G(ca_2) = K_G(Z^{ca_2}_{1,1}) \cup K_G(Z^{ca_2}_{2,1}) \cup K_G(Z^{ca_2}_{4,1}) = \{p_{29}, p_{23}, p_{14}\}$
$K_G(cl_1) = K_G(Z^{cl_1}_{1,1}) = \{p_{11}\}$
$K_G(cl_2) = K_G(Z^{cl_2}_{1,1}) \cup K_G(Z^{cl_2}_{2,1}) = \{p_{12}, p_{17}, p_{21}, p_{26}\}$
$K_G(cl_3) = K_G(Z^{cl_3}_{1,1}) \cup K_G(Z^{cl_3}_{3,1}) = \{p_{35}\}$

Cuadro 4.3: Funciones de etiquetado S, L y K_G del grafo simple de la figura 4.22.

Siguiendo con el paso 2 del método de corrección definido en la sección 4.4, se utilizará el grafo de poda simple para generar el grafo de poda de cerrojos $G' = (V', E')$, de la definición 19. Esta transformación ocurre por la aglomeración de zonas comunes, obteniendo de esta manera grafo de poda entre recursos de la figura 4.23. Una vez obtenido el grafo de poda de recursos de la figura 4.23 procedemos con el tercer paso definido en los pasos necesarios para realizar la corrección del modelo que consiste en determinar que conjunto de arcos es necesario eliminar para que el grafo de poda de recursos sea acíclico.

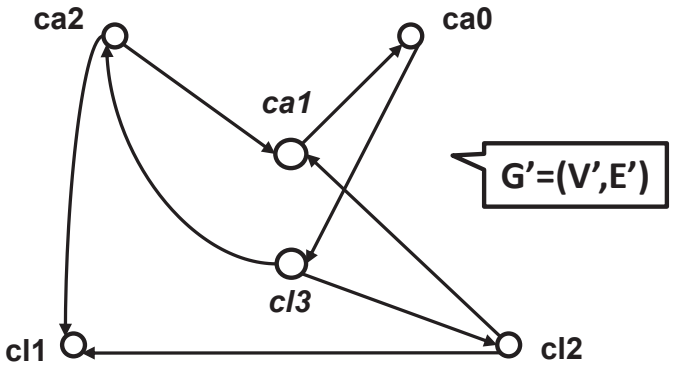

Figura 4.23: Grafo aglomerado a partir del grafo de poda de zonas de la figura 4.22.

Los pasos posteriores harán uso intensivo de los algoritmos 6 y 7, sin embargo debido al tamaño de la red solo mencionaremos los resultados parciales de las operaciones de selección y transformación de la red. Esto lo haremos presentando los resultados de determinadas operaciones relevantes en los algoritmos y las modificaciones respectivas en el grafo de poda de zonas y de recursos así como en la red de Petri obtenida en base a estas modificaciones.

Inicialmente utilizaremos el algoritmo 6 que nos permite realizar una mejor selección del(los) arco(s) que hacen acíclico en grafo de poda de recursos para así remover todos los cerrojos por la introducción de recursos virtuales en la red. Es preciso mencionar que el algoritmo para la selección de los arcos tiene heurísticas que se han basado en el trabajo de [Kaufmann 01]. Una de las primeras acciones heurísticas del algoritmo es la eliminación de vértices no productivos en términos de corrección como lo son los vértices fuentes o los vértices sumideros que es una verificación que se repite en cada iteración. En nuestro grafo tenemos que el vértice cl_1 es un vértice sumidero y por lo tanto no pertenecerá a un ciclo en el grafo de poda de recursos como se muestra en la figura 4.24.a. Es preciso mencionar que el recurso cl_1 no forma parte de ninguno de los subgrafos fuertemente conexos porque solo tiene arcos que inciden en él. Es decir; no existe ninguna zona de este recurso que pode lugares no esenciales a otra zona para la obtención de un cerrojo malo.

En términos técnicos podríamos mencionar que este recurso nunca formará parte de una espera circular en un estado de bloqueo de la red. El extremo opuesto a este caso lo vemos con los recursos ca_2 y cl_3 que tienen diversos arcos incidentes y salientes, motivo por el cual al menos cada uno de ellos participará en uno de los estados de bloqueos alcanzados por la red de interconexión.

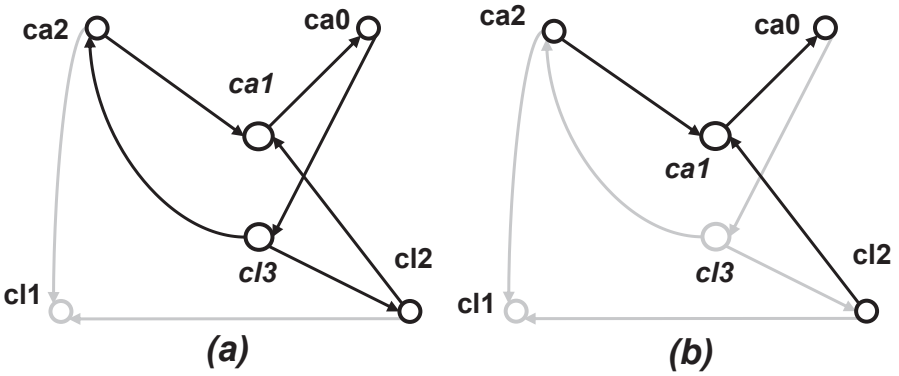

Figura 4.24: Grafo resultante después de la primer y segunda iteración del algoritmo 6 aplicado al grafo de poda de recursos de la figura 4.23.

Procedemos entonces a seleccionar los vértices en que la diferencia entre sus vértices de entrada y sus vértices salida sea máxima, como es el caso del vértice, cl_3 que tiene un valor de 1, aplicando $gs(2) - ge(1) = 1$ y omitiendo el vértice sumidero cl_1 que ya fue removido del grafo de poda de recursos como se muestra en la figura 4.24.a. de color opaco respecto a los otros vértices.

Con esta acción almacenamos en F todos los arcos que inciden en el vértice seleccionado, con lo cual tenemos que $F = \{(ca_0, cl_3)\}$ y el conjunto $W = \{cl_3\}$. En la siguiente iteración solo quedan vértices fuente como los vértices ca_2 y cl_2, o vértices sumideros como el vértice ca_0 que se eliminan en la primera parte de la segunda iteración del algoritmo 6 como se muestra en la figura 4.24.b en con líneas negras. De igual manera también se elimina el vértice cl_2 al quedar como un vértice disconexo de los otros. Una vez finalizado el algoritmo 6, obtenemos la información de entrada para el algoritmo 7 de transformación de la red como el conjunto R que contiene la siguiente información.

$$R = \{(Z_{2,1}^{ca_0}; Z_{2,1}^{cl_3}), (Z_{3,1}^{ca_0}; Z_{3,1}^{cl_3})\}$$

Aplicando el algoritmo 7, procederemos ha hacer los cambios en el grafo de recursos para que este sea acíclico, tomando en cuenta la información de las zonas de los arcos

seleccionados en el conjunto R. Hemos seleccionado el arco entre ca_0 y cl_3 del grafo de poda de recursos, que en base a la función L del grafo de poda de zonas nos indica las zonas implicadas:

$$L((ca_0, cl_3)) = L((Z_{3,1}^{ca_0}; Z_{3,1}^{cl_3})) = \{(t_{43}, p_{35})\}$$

$$L((ca_0, cl_3)) = L((Z_{2,1}^{ca_0}; Z_{2,1}^{cl_3})) = \{(t_{29}, p_{27}), (t_{34}, p_{30}), (t_{38}, p_{32})\}$$

Siguiendo el algoritmo 7 debemos seleccionar un par, que en nuestro caso ha sido el par $L((Z_{3,1}^{ca_0}; Z_{3,1}^{cl_3}))$, sin embargo no aplica para condición de la línea ♯11 de este algoritmo. El par restante, satisface esta condición pero no se satisface que $Fa(Z^r) \subseteq Z^s$, *(lema 26)* motivo por el cual debemos realizar la transformación de la red como lo indica el algoritmo en los pasos posteriores a la línea ♯24. Para este caso es necesario agregar un nuevo recurso denominado $ca_{0'}$ con su respectivo marcado inicial y además replicar ciertos lugares proceso. Este nuevo recurso virtual se ha agregado en el grafo de poda de recursos de la figura 4.25, que ahora aparece sin ciclos entre los vértices. Hemos omitido presentar el grafo de poda de zonas que también deberá ser actualizado con este nuevo recurso en el respectivo grafo conexo.

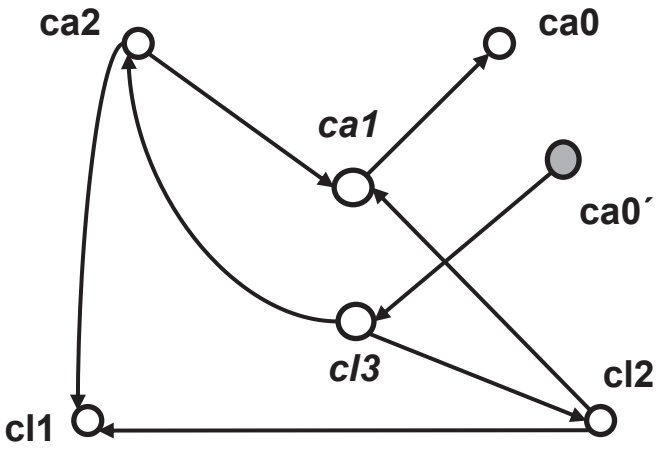

Figura 4.25: Grafo acíclico después de aplicar el algoritmo 7 al grafo de la figura 4.24.b

Los cambios aplicados por el algoritmo han modificado las transiciones t_{29},t_{34} y (t_{38} la máquina de estados $SM2$, con la entrada al nuevo recurso virtual. Adicionalmente, fue necesario agregar el nuevo lugar proceso $p_{34'}$ y la nueva transición $t_{41'}$ que libera

el nuevo recurso virtual como se indica en la red de la figura 4.26. El lugar $p_{34'}$ y la transición $t_{41'}$ son necesarios porque memorizan la ruta particular para el nuevo recurso virtual $ca_{0'}$. Los lugares y transiciones originales seguirán utilizando el viejo recurso ca_0 en esta y las otras máquinas de estado. La función S del nuevo recurso virtual $ca_{0'}$ es la siguiente: $S(ca_{0'})=\{ca_{0'}\} \cup Z_{2,1}^{ca_{0'}}=\{ca_{0'},p_{28},p_{31},p_{33},p_{34'}\} = \|\mathbf{y}_{ca_{0'}}\|$. Adicionalmente, la zona del recurso anterior será la siguiente. $Z_{2,1}^{ca_0}=Z_{2,1}^{ca_0} \setminus Z_{2,1}^{ca_{0'}}$. Este nuevo recurso tendrá un marcado igual a los otros recursos y será de uso privado para esta máquina de estados, es decir para este nodo destino en particular.

Los cambios realizados sobre el grafo de poda de recursos los podemos ver en la red de Petri de la figura 4.26 que muestra solo la máquina de estados $SM2$ de toda la red presentada en la figura 2.8, debido a que las otras máquinas de estado permanecen sin cambios, pero siguen perteneciendo a la clase $SOAR^2$. Para mayor legibilidad se utilizará el nombre de los canales en letra mayúscula en la red de Petri de esta figura.

En términos técnicos podemos comentar que el recurso virtual $ca_{0'}$, junto con el lugar proceso $p_{34'}$ y la transición $t_{41'}$ forman parte de una *ruta de escape del bloqueo* en la red de interconexión [Duato 93]. Visto desde nuestro modelo, está claro que se ha eliminado el cerrojo que es la causa estructural detras de los bloqueos que inducen una espera circular entre los recursos implicados en un bloqueo y por lo tanto al eliminarlo se puede garantizar la vivacidad del modelo.

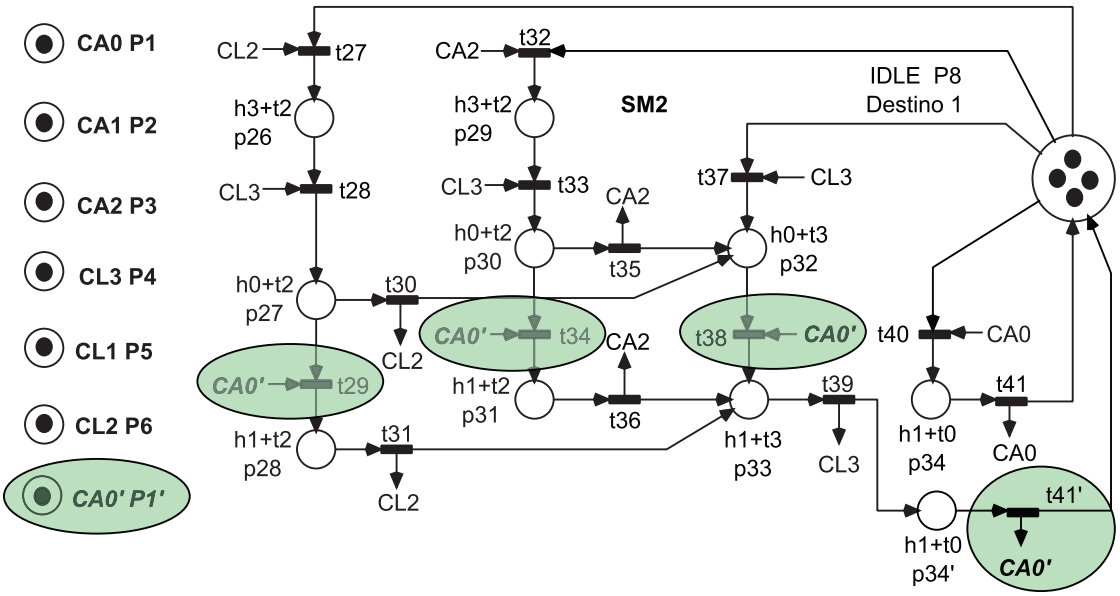

Figura 4.26: Red de Petri viva después de aplicar el algoritmo 7.

Siguiendo nuestra metodología, presentada en la figura 2.4 del capítulo 2, en la etapa de síntesis se realiza el proceso de *transferir las restricciones de encaminamiento al algoritmo fuente*, que en este caso consiste en agregar una restricción para la correcta utilización del canal virtual $ca_{0'}$ en el algoritmo 8 que será libre de bloqueos. Es preciso mencionar que no existe un método general para implantar los cambios de la red de Petri en el algoritmo fuente, debido a que los lenguajes utilizados para describir los algoritmos de encaminamiento son diversos. Adicionalmente, aunque el lenguaje de descripción fuese estándar, los criterios utilizados para tomar las decisiones para el encaminamiento son innumerables y dependerán del dominio de aplicación o cualquier otro criterio que el diseñador considere importante para el encaminamiento.

Algoritmo 8 Algoritmo de encaminamiento libre de bloqueos para el nodo i.

Entrada: El flit de la cabeza m desde la cola de mensajes.
Local: $C_i \subseteq CH$, conjunto de canales de salida para el nodo n_i
$F \subseteq C_i$, conjunto de canales no asignados
Salida: El próximo canal para ser usado por m.

1. **Si** $(destino(m) = i)$ **Entonces**
2. Almacene el mensaje m en n_i
3. **Sino**
4. **Si** $(entradac(m)=cl_3 \wedge destino(m)=1)$ **Entonces**
5. usar $ca_{0'}$
6. **Fin Si**
7. **Si** $(ca_i \in F)$ **Entonces**
8. usar ca_i
9. $F := F \backslash \{ca_i\}$
10. **Sino**
11. **Si** $((destino(m) < i) \wedge (cl_i \in F))$ **Entonces**
12. usar cl_i
13. $F := F \backslash \{cl_i\}$
14. **Sino**
15. colocar en cola el mensaje m
16. **Fin Si**
17. **Fin Si**
18. **Fin Si**

En este punto nosotros sabemos que una forma eficaz para garantizar la utilización del nuevo recurso virtual y prevenir los bloqueos en el nuevo algoritmo de encaminamiento, es considerar la cola de entrada del mensaje y el destino del mismo. En nuestro ejemplo se

agrega la función *entradac(m)* que nos brinda la cola por la que está entrando el mensaje al nodo actual. Esta función junto con el nodo destino del mensaje sirve para condicionar de forma unívoca la utilización del nuevo recurso virtual. En nuestro algoritmo, las líneas $4-6$ establecen esta condición para el uso del nuevo recurso virtual. Esta restricción solo altera a los mensajes con destino al nodo 1 y que entran por el canal cl_3 lo cual significa que es un cambio mínimo con respecto a todo el conjunto de destinos de la red. A continuación presentamos nuestra conclusión para este capítulo.

4.7. Conclusión

En este capítulo nos hemos concentrado en definir y aplicar un nuevo método de corrección/síntesis que garantiza la vivacidad de las redes $SOAR^2$, mediante la introducción de canales virtuales en puntos estratégicos del modelo. Este proceso corresponde a la última etapa de nuestra metodología, presentada en el capítulo 2 de este trabajo. Este método de control es un enfoque novedoso en comparación a todos los métodos de control existentes hasta la fecha, debido a que se apoya en la modificación de la estructura del proceso y no en restringir el uso de los recursos entre los procesos que los solicitan.

Para modificar los procesos se ha recurrido a agregar recursos virtuales y en algunos casos a replicar ciertos lugares proceso y transiciones que se requieran para garantizar la correcta estructura de los procesos en el modelo. A diferencia de los métodos tradicionales, aquí no se restringen los marcados muertos en el modelo y al contrario se generan nuevos marcados que proporcionan mayor concurrencia entre los procesos.

Adicionalmente, hemos hecho uso intensivo de los grafos de poda de zonas y el grafo de poda de recursos para determinar la existencia de cerrojos en la red y aplicar los cambios necesarios para eliminarlos del modelo. Este proceso se ha optimizado a través de dos algoritmos especializados que utilizan heurísticas para sus decisiones. El primero de los algoritmos selecciona los arcos del grafo de poda de recursos a ser podados y el segundo establece de forma estructural los lugares proceso y las transiciones que se deben modificar en la red de Petri. La estrategia para realizar la menor cantidad de cambios posibles es determinada por la cantidad de zonas implicadas en cada arco seleccionado para remover del grafo de poda de cerrojos. Esto es de gran utilidad y significación porque permite alcanzar soluciones óptimas al modelo tratado. Finalmente, se ha retomado el ejemplo de la red de interconexión presentado en el capítulo 2, para aplicar este método de control y hacer vivo el modelo. Estos cambios se han trasladado manualmente al algoritmo de encaminamiento fuente como pequeñas restricciones al encaminamiento, que garantizarán que el algoritmo de encaminamiento será libre de bloqueos.

Capítulo 5

Conclusiones

La presente tesis doctoral se ha centrado en el desarrollo de una metodología completa para la construcción de algoritmos de encaminamiento mínimos y adaptativos de tipo wormhole que estén libres de bloqueo. El buscar una metodología completa que cubre desde la fase de especificación, pasando por el análisis y llegando hasta la síntesis que introduce correcciones en caso de haber detectado errores en el diseño original, es en sí mismo una aportación en cuanto que las aproximaciones existentes no cubren sobre todo las últimas fases.

Nos hemos concentrado en los problemas de bloqueos, entendidos estos como la situación que aparece cuando un mensaje en tránsito desde un origen a un destino no puede alcanzar nunca su destino. Para abordar este problema se ha adaptado un tipo de abstracción del diseño que se denomina “abstracción como Sistema de Asignación de Recursos” ó “abstracción SAR”. Desde un punto de vista instrumental hemos utilizado como herramienta formal, las redes de Petri, por tratarse de un paradigma de modelado formal para el que existe un rico cuerpo de resultados para todas las fases del ciclo de diseño.

En este contexto las aportaciones realizadas al problema planteado se han presentado en tres capítulos cada uno dedicado a una de las fases de la metodología de diseño propuesta.

En el capítulo 2, nos hemos centrado en la obtención de un modelo de redes de Petri, a partir del diseño suministrado. La especificación de partida consiste en la topología de la red de interconexión y el propio algoritmo de encaminamiento mínimo adaptativo y tipo wormhole. La abstracción SAR ha consistido en la identificación sobre estos sistemas de los dos elementos básicos para estudiar los problemas de bloqueos debidos a recursos:

1. Los procesos, que en nuestro caso están constituídos por las rutas mínimas que conducen desde cualquier nodo origen de la red de interconexión hasta un

nodo destino concreto siguiendo el algoritmo de encaminamiento suministrado. Habrá tantos procesos como destinos posibles.

2. Los recursos, que en nuestro caso son los distintos canales que interconectan los nodos de la red de interconexión y que son asignados a los mensajes en tránsito hacia su destino final.

A partir de la abstracción propuesta y a través del procedimiento sistemático descrito en el capítulo 2 hemos obtenido el modelo de red de Petri para el estudio de la existencia de bloqueos. Hay que insistir que el procedimiento desarrollado no presupone ninguna restricción especial a las topologías y algoritmos de encaminamiento. No obstante, la restricción a algoritmos adaptativos, de ruta mínima con control de flujo wormhole nos ha llevado a demostrar que las redes obtenidas pertenecen a la bien conocida clase de las S^4PR dentro del estudio de Sistemas de Asignación de Recursos.

En el capítulo 3 se ha profundizado en el tipo de modelos de redes de Petri que se obtienen para esta clase de sistemas, llegando a determinar que en realidad se obtiene una subclase estricta de las S^4PR caracterizada porque para cada conjunto máximo de recursos que se utilizan simultáneamente en el sistema, el orden de asignación de estos recursos es el mismo orden de su liberación. Esta propiedad es particular de los sistemas de encaminamiento en redes de interconexión o en sistemas de vehículos guiados automáticamente. Esta restricción se ha caracterizado estructuralmente sobre las redes S^4PR a través del concepto zona de uso continuado de un recurso y la relación de precedencia entre estas zonas. El resultado ha sido la caracterización de un nueva subclase de redes de Petri denominada $SOAR^2$ (S^4PR with Ordered Allocation and Release of Resources). Se han estudiado las propiedades de esta clase y se han dado métodos de análisis especializados que mejoran en algunos sentidos los generales conocidos para la clase S^4PR, por ejemplo, para el cálculo del grafo de poda de recursos.

En el capítulo 4 se han desarrollado técnicas nuevas para la corrección de los modelos en caso de que existan bloqueos aprovechando extensivamente la teoría desarrollada para la clase $SOAR^2$. Estas técnicas, desde el punto de vista del objeto utilizado para la corrección, no difieren de las técnicas conocidas para forzar la vivacidad en redes S^4PR ya que ambas utilizan recursos virtuales. No obstante, como se ha demostrado son totalmente diferentes dado que las conocidas no son aplicables (por ser no implementables, en general) en el contexto de las redes de interconexión en multicomputadores. Estas técnicas a través del concepto de zona de uso continuado de recurso que garantiza la localidad de la solución alcanzada, permiten hacer viva la red por la introducción de nuevos estados que representan rutas de escape a través de los recursos virtuales (canales virtuales implementados sobre canales físicos) introducidos que son usados de manera más privada. La solución alcanzada permite incluso incrementar la concurrencia en muchos casos al

contrario que los métodos basados en la eliminación de estados prohibidos mediante relaciones de exclusión mutua generalizadas que secuencializan el sistema al introducir restricciones en el uso de una forma compartida de los recursos.

Finalmente, se enumeran algunas extensiones inmediatas que podrían constituir un continuación del trabajo presentado en esta memoria.

1. Ampliación del estudio de este tipos de redes con la finalidad de capturar otros problemas que enfrentan los algoritmos de encaminamiento como los bloqueos activos.

2. Profundización en los métodos de control para asegurar la vivacidad de estos modelos en entornos que no permiten la utilización de recursos virtuales.

3. Desarrollar métodos sistemáticos para trasladar las correcciones del modelo al algoritmo de encaminamiento fuente.

4. Estudio de otros tipos de algoritmos de encaminamiento como los que utilizan rutas no mínimas para alcanzar su destino.

Bibliografía

[Anjan 95] K.V. Anjan & T.M. Pinkston. *DISHA: a deadlock recovery scheme for fully adaptive routing*. In Parallel Processing Symposium, 1995. Proceedings., 9th International, pages 537 –543, April 1995.

[Bellman 62] Richard Bellman. *Dynamic Programming Treatment of the Travelling Salesman Problem*. J. ACM, vol. 9, pages 61–63, January 1962.

[Boura 93] Younes M. Boura & Chita R. Das. *A Class of Partially Adaptive Routing Algorithms for n–dimensional Meshes*. In Proceedings of the 1993 International Conference on Parallel Processing - Volume 03, ICPP '93, pages 175–183, Washington, DC, USA, 1993. IEEE Computer Society.

[Bozer 91] Yavuz A. Bozer & Mandyam M. Srinivasan. *Tandem Configurations for Automated Guided Vehicle Systems and the Analysis of Single Vehicle Loops*. IIE Transactions, vol. 23, no. 1, pages 72–82, 1991.

[Broadbent 85] A J Broadbent, C B Besant, S K Premi & S P Walker. *Free ranging AGV systems: promises, problems and pathways*. In Proc. of 2 nd Int. Conf. on Automated Material Handling, (IFS Publ. Ltd, pages 221–237, 1985.

[Cano 10] E.E. Cano, C.A. Rovetto & J. Colom. *On the computation of the minimal siphons of S4PR nets from a generating family of siphons*. In Emerging Technologies and Factory Automation (ETFA), 2010 IEEE Conference on, pages 1 –8, 2010.

[Cano 11] E. Cano Elia. *Teoría de Grafos Aplicada al Análisis de Redes S^4PR y Subclases. El Problema del Cálculo de los Cerrojos Mínimos*. PhD thesis, Zaragoza. España, Departamento de Ingeniería e Informática, Universidad de Zaragoza, Octubre 2011.

[Chiba 06] R. Chiba, J. Ota & T. Aral. *AGV System Design using Competitive and Cooperative Co-evolution*. In SICE-ICASE, 2006. International Joint Conference, pages 4256 –4259, oct. 2006.

[Chien 95] Andrew A. Chien & Jae H. Kim. *Planar-adaptive routing: low-cost adaptive networks for multiprocessors*. J. ACM, vol. 42, pages 91–123, January 1995.

[Chiola 95] G. Chiola, G. Franceschinis, R. Gaeta & M. Ribaudo. *GreatSPN 1.7: GRaphical Editor and Analyzer for Timed and Stochastic Petri Nets*. vol. 24, pages 47–68, 1995.

[Coffman 71] E. G. Coffman, M. Elphick & A. Shoshani. *System Deadlocks*. ACM Comput. Surv., vol. 3, pages 67–78, June 1971.

[Colom 03] José-Manuel Colom. *The Resource Allocation Problem in Flexible Manufacturing Systems*. In W.M.P. van der Aalst & E. Best, editeurs, Proc. of the 24th Int. Conf. on Applications and Theory of Petri Nets, volume 2679 of *Lecture Notes in Computer Science*, pages 23–35, Eindhoven, Netherlands, June 2003. Springer–Verlag.

[Dally 86] William J. Dally & Charles L. Seitz. *The Torus Routing chip*. Distributed Computing, vol. 1, pages 187–196, 1986. 10.1007/BF01660031.

[Dally 87] W.J. Dally & C.L. Seitz. *Deadlock-Free Message Routing in Multiprocessor Interconnection Networks*. IEEE Transactions on Computers, vol. 36, no. 5, pages 547–553, May 1987.

[Dally 92] W.J. Dally. *Virtual-Channel Flow Control*. IEEE Transactions on Parallel and Distributed Systems, vol. 3, no. 2, pages 194–205, March 1992.

[Dally 93] W.J. Dally & H. Aoki. *Deadlock-Free Adaptive Routing in Multicomputer Networks Using Virtual Channels*. IEEE Transactions on Parallel and Distributed Systems, vol. 4, no. 4, pages 466–475, April 1993.

[Dally 03] William Dally & Brian Towles. Principles and practices of interconnection networks. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.

[Dijkstra 59] E. W. Dijkstra. *A note on two problems in connexion with graphs*. Numerische Mathematik, vol. 1, pages 269–271, 1959. 10.1007/BF01386390.

[Dijkstra 68] Edsger W. Dijkstra. *Cooperating sequential processes*. In F. Genuys, editeur, Programming Languages: NATO Advanced Study Institute, pages 43–112. Academic Press, 1968.

[Domke 11] J. Domke, T. Hoefler & W. Nagel. *Deadlock–Free Oblivious Routing for Arbitrary Topologies*. In Proceedings of the 25th IEEE International Parallel and Distributed Processing Symposium (IPDPS), pages 613–624. IEEE Computer Society, May 2011.

[Dorigo 92] Marco Dorigo. *Optimization, Learning and Natural Algorithms (in Italian)*. PhD thesis, Dipartimento di Elettronica, Politecnico di Milano, Milan, Italy, 1992.

[Duato 93] J. Duato. *A new theory of deadlock-free adaptive routing in wormhole networks*. Parallel and Distributed Systems, IEEE Transactions on, vol. 4, no. 12, pages 1320 –1331, December 1993.

[Duato 95a] J. Duato. *A Necessary and Sufficient Condition for Deadlock-Free Adaptive Routing in Wormhole Networks*. IEEE Transactions on Parallel and Distributed Systems, vol. 6, no. 10, pages 1055–1067, October 1995.

[Duato 95b] J. Duato. *A Theory of Deadlock-Free Adaptive Multicast Routing in Wormhole Networks*. IEEE Transactions on Parallel and Distributed Systems, vol. 6, no. 9, pages 976–987, September 1995.

[Duato 03] J. Duato, S. Yalamanchili & L. Ni. Interconnection networks: An engineering approach. Morgan Kaufmann Publishers, Los Altos, CA 94022, USA, 2003.

[Eades 90] P. Eades & K. Sugiyama. *How to draw a directed graph*. Journal of Information Processing, vol. 13, no. 4, 1990.

[Ezpeleta 95] Joaquin Ezpeleta, José-Manuel Colom & Javier Martínez. *A Petri net based deadlock prevention policy for flexible manufacturing systems*. IEEE Transactions on Robotics and Automation, vol. 11, no. 2, pages 173–184, April 1995.

[Fanti 02] Maria Pia Fanti. *A Deadlock Avoidance Strategy for AGV Systems Modelled by Coloured Petri Nets*. In Proceedings of the Sixth International Workshop on Discrete Event Systems (WODES'02), pages 61–, Washington, DC, USA, 2002. IEEE Computer Society.

[Ganesharajah 98] Tharma Ganesharajah, Nicholas G. Hall & Chelliah Sriskandarajah. *Design and operational issues in AGV-served manufacturing systems*. Annals of Operations Research, vol. 76, pages 109–154, 1998. 10.1023/A:1018936219150.

[García-Vallés 99] F. García-Vallés. *Contributions to the structural and symbolic analysis of place/transition nets with applications to flexible manufacturing systems and asynchronous circuits*. PhD thesis, University of Zaragoza, Zaragoza, April 1999.

[Garey 83] M. Garey & D. S. Johnson. *Crossing number is NP-complete*. SIAM Journal of Algebraic and Discrete Methods, vol. 4, no. 3, 1983.

[Gaskins 87] R. J. Gaskins & J. M. A. Tanchoco. *Flow path design for automated guided vehicle systems*. International Journal of Production Research, vol. 25, pages 667–676, 1987.

[Giua 93] A. Giua, F. DiCesare & M. Silva. *Petri nets supervisors for generalized mutual exclusion constraints*. In Proc. 1993 IFAC World Congress, volume 1, pages 267–270, Sidney, Australia, July 1993.

[Glass 92] C.J. Glass & L.M. Ni. *The turn model for adaptive routing*. In Procs of the 19th annual International Symposium on Computer Architecture, pages 278–287, Queensland, Australia, May 19-21 1992. ACM Press.

[Goetz 90] William G. Goetz & Pius J. Egbelu. *Guide path design and location of load pick-up/drop-off points for an automated guided vehicle system*. International Journal of Production Research, vol. 28, no. 5, pages 927–941, 1990.

[Gravano 94] Luis Gravano, Gustavo D. Pifarré, Pablo E. Berman & Jorge L. C. Sanz. *Adaptive Deadlock- and Livelock-Free Routing with All Minimal Paths in Torus Networks*. IEEE Trans. Parallel Distrib. Syst., vol. 5, pages 1233–1251, December 1994.

[Gregorio 95] J. A. Gregorio, F. Vallejo, R. Beivide & C. Carrión. *Petri net modeling of interconnection networks for massively parallel architectures*. In Proceedings of the 9th international conference on Supercomputing, ICS '95, pages 107–116, New York, NY, USA, 1995. ACM.

[Guia 92] A. Guia, F. DiCesare & M. Silva. *Generalized mutual exclusion constraints for nets with uncontrollable transitions*. In Proc. IEEE Int. Conf. on Systems, Man, and Cybernetics, pages 974–979, Chicago, USA, October 1992.

[Holt 72] Richard C. Holt. *Some Deadlock Properties of Computer Systems*. ACM Comput. Surv., vol. 4, pages 179–196, September 1972.

[Kaufmann 01] M. Kaufmann & D. Wagner. *Drawing graphs: methods and models*. Springer Verlag, 2001.

[Kim 91] Chang Kim & J. M. A. Tanchoco. *Conflict-free shortest-time bidirectional AGV routeing*. International Journal of Production Research, vol. 29, pages 2377–2391, 1991.

[Kul 85] Materials handling handbook. Wiley-Interscience; 2 edition (January 4, 1985), 1985.

[Lee 98] Jung Hoon Lee, Beom Hee Lee & Myoung Hwan Choi. *A real-time traffic control scheme of multiple AGV systems for collision free minimum time*

motion: a routing table approach. Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on, vol. 28, no. 3, pages 347 –358, may 1998.

[Linder 91] D.H. Linder & J.C. Harden. *An Adaptive and Fault Tolerant Wormhole Routing Strategy for k-ary n-cubes*. IEEE Transactions on Computers, vol. 40, no. 1, pages 2–12, January 1991.

[Ling 99] Gui Ling, Hsu Wen-jing, Qiu Ling, Qiu Ling & Hsu Wen Jing. *Scheduling and Routing Algorithms for AGVs: a Survey*. 1999.

[López-Grao 11] Juan Pablo López-Grao. *Contributions to the deadlock problem in multithreaded software applications approached as Resource Allocation Systems*. PhD thesis, Zaragoza. España, Departamento de Informática e Ingeniería de Sistemas, Universidad de Zaragoza, 2011.

[Lozoya 07] C. Lozoya, P. Marti, M. Velasco & J.M. Fuertes. *Effective Real-Time Wireless Control of an Autonomous Guided Vehicle*. In Industrial Electronics, 2007. ISIE 2007. IEEE International Symposium on, pages 2876 –2881, june 2007.

[Marsan 87] M. Ajmone Marsan, G. Balbo & G. Conte. Performance models of multiprocessor systems. MIT Press, Cambridge, MA, USA, 1987.

[Martínez 97] Juan Miguel Martínez, P. L. López, José Duato & Timothy Mark Pinkston. *Software-Based Deadlock Recovery Technique for True Fully Adaptive Routing in Wormhole Networks*. In Proceedings of the international Conference on Parallel Processing, ICPP '97, pages 182–189, Washington, DC, USA, 1997. IEEE Computer Society.

[Müller 83] Thomas. Müller. Automated guided vehicles. IFS (Publications), 1983.

[Murata 89] Tadao Murata. *Petri nets: Properties, analysis and applications*. Proceedings of the IEEE, vol. 77, no. 4, pages 541–580, April 1989.

[Nof 99] S.Y. Nof. Handbook of industrial robotics. no. v. 1 in Electrical and electronic engineering. John Wiley, 1999.

[Owens 07] J.D. Owens, W.J. Dally, R. Ho, D.N. Jayasimha, S.W. Keckler & Li-Shiuan Peh. *Research Challenges for On-Chip Interconnection Networks*. Micro, IEEE, vol. 27, no. 5, pages 96 –108, sept.-oct. 2007.

[Park 00a] J. Park & S. A. Reveliotis. *Enhancing the flexibility of algebraic deadlock avoidance policies through petri net structural analysis*. In In Proc. of the 2000 IEEE Int. Conf. on Robotics and Automatization, pages 3371–3376, San, Francisco, USA, April 2000.

[Park 00b] Jonghun Park & Spyros A. Reveliotis. *A Polynomial-Complexity Deadlock Avoidance Policy for Sequential Resource Allocation Systems with Multiple Resource Acquisitions and Flexible Routings*. In Proceedings of the IEEE International Conference on Decision & Control, pages 2663–2669, Australia, December 2000. IEEE.

[Peterson 89] J. L. Peterson & A. Silberschatz. Sistemas operativos. Conceptos fundamentales. Editorial Reverté, 1989.

[Petri 62] C. A. Petri. *Kommunikation mit Automaten*. PhD thesis, Institut für instrumentelle Mathematik, Bonn, 1962.

[Ramadge 87] P. J. Ramadge & W. M. Wonham. *Supervisory control of a class of discrete event processes*. SIAM J. Control Optim., vol. 25, pages 206–230, January 1987.

[Reveliotis 96] S.A. Reveliotis & P.M. Ferreira. *Deadlock avoidance policies for automated manufacturing cells*. Robotics and Automation, IEEE Transactions on, vol. 12, no. 6, pages 845 –857, dec 1996.

[Reveliotis 00] Spyros A. Reveliotis. *Conflict resolution in AGV systems*. IIE Transactions, vol. 32, pages 647–659, 2000. 10.1023/A:1007663100796.

[Rovetto 10a] C.A. Rovetto, E.E. Cano & J.-M. Colom. *Deadlock analysis in minimal adaptive routing algorithms using Petri Nets*. In Systems Man and Cybernetics (SMC), 2010 IEEE International Conference on, pages 2619 –2626, 2010.

[Rovetto 10b] C.A. Rovetto, E.E. Cano & J.-M. Colom. *Deadlock Control Software for Tow Automated Guided Vehicles using Petri Nets*. pages 125–140, Braga, Portugal, 06/2010 2010. University of Hamburg Department of Informatics.

[Rupp 98] T. Rupp, T. Cord, R. Lohnert & D.E. Lazic. *Positioning and communication system for autonomous guided vehicles in indoor environment*. In Electrotechnical Conference, 1998. MELECON 98., 9th Mediterranean, volume 1, pages 187 –191 vol.1, may 1998.

[Schwiebert 95a] Loren Schwiebert & D. N. Jayasimha. *Optimal Fully Adaptive Minimal Wormhole Routing for Meshes*. Journal of Parallel and Distributed Computing, vol. 27, pages 56–70, 1995.

[Schwiebert 95b] Loren Schwiebert & D. N. Jayasimha. *A Universal Proof Technique for Deadlock-Free Routing in Interconnection Networks*. In In 7 th Annual ACM Symposium on Parallel Algorithms and Architectures, pages 175–184, 1995.

[Schwiebert 96] Loren Schwiebert & D. N. Jayasimha. *A necessary and sufficient condition for deadlock-free wormhole routing*. J. Parallel Distrib. Comput., vol. 32, pages 103–117, January 1996.

[Schwiebert 01] L. Schwiebert. *Deadlock-free oblivious wormhole routing with cyclic dependencies*. Computers, IEEE Transactions on, vol. 50, no. 9, pages 865 –876, sep 2001.

[Silva 85] Manuel Silva. Las redes de petri, en la automática y en la informática. Editorial AC, 1985.

[Sinriech 97] David Sinriech & J.M.A. Tanchoco. *Design procedures and implementation of the segmented flow topology (SFT) for discrete material flow systems*. IIE Transactions, vol. 29, no. 4, pages 323–335, 1997.

[Stallings 05] William Stallings. Sistemas operativos: Aspectos internos y principios de diseño. PEARSON, 2005.

[Stephen C 88] Daniels Stephen C. *Real Time Conflict Resolution in Automated Guided Vehicle Scheduling*. In Department of Industrial Engineering, Pennsylvania State University, Phd, Thesis, USA, May 1988.

[Taghaboni-Dutta 95] F. Taghaboni-Dutta & J. M. A. Tanchoco. *Comparison of dynamic routeing techniques for automated guided vehicle system*. In International Journal of Production Research, pages 2653–2669. Taylor & Francis, 1995.

[Taktak 08] Sami Taktak, Jean-Lou Desbarbieux & Emmanuelle Encrenaz. *A tool for automatic detection of deadlock in wormhole networks on chip*. ACM Trans. Des. Autom. Electron. Syst., vol. 13, pages 6:1–6:22, February 2008.

[Tarjan 82] R. Tarjan. *Depth first search and linear graph algorithms*. SIAM Journal of Computing, vol. 1, no. 2, 1982.

[Towles 05] Brian Towles. *Distributed Router Fabrics*. PhD thesis, Stanford University, 2005.

[Tricas 99] Fernando Tricas, José-Manuel Colom & Joaquin Ezpeleta. *A solution to the problem of deadlocks in concurrent systems using Petri nets and integer linear programming*. In G. Horton, D. Moller & U. Rude, editeurs, Proc. of the 11th European Simulation Symposium, pages 542–546, Erlangen, Germany, October 1999. The society for Computer Simulation Int.

[Tricas 03] Fernando Tricas. *Analysis, Prevention and Avoidance of Deadlocks in Sequential Resource Allocation Systems*. PhD thesis, Zaragoza. España, Departamento de Ingeniería Eléctrica e Informática, Universidad de Zaragoza, May 2003.

[Warnakulasuriya 97] Sugath Warnakulasuriya & Timothy Mark Pinkston. *Characterization of Deadlocks in Interconnection Networks*. In Proceedings of the 11th International Symposium on Parallel Processing, IPPS '97, pages 80–86, Washington, DC, USA, 1997. IEEE Computer Society.

[Wu 04] Naiqi Wu & MengChu Zhou. *Modeling and deadlock control of automated guided vehicle systems*. Mechatronics, IEEE/ASME Transactions on, vol. 9, no. 1, pages 50 –57, march 2004.

[Wu 05] Naiqi Wu & MengChu Zhou. *Modeling and deadlock avoidance of automated manufacturing systems with multiple automated guided vehicles*. Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on, vol. 35, no. 6, pages 1193 –1202, dec. 2005.

Apéndice A

Definiciones formales sobre las redes de interconexión

En este apéndice presentamos las definiciones formales de los conceptos básicos de las Redes de Interconexión utilizados en esta memoria. Este apéndice se apoya principalmente de los términos utilizados en los trabajos de [Dally 87] y [Duato 95b] y [Duato 95a]. Para profundizar sobre estos términos, le recomendamos al lector la siguiente literatura [Dally 03] [Duato 03].

Una *Red de Interconexión (I)*, es un multigrafo dirigido fuertemente conexo $I = G(N,C)$, en donde, N son los vértices del multigrafo que representan los nodos de procesamiento. Los arcos C del multigrafo representan los canales de comunicación, que pueden ser mas de uno entre un par de nodos. Cada canal es asociado con una capacidad $cap(c_i)$ de almacenamiento $(buffer)$. El nodo fuente y destino de un canal c_i, se denomina s_i y d_i respectivamente.

Una Red de Interconexión se representa gráficamente mediante un multigrafo orientado, en donde los lugares representan los canales y los vértices los nodos de procesamiento. Los nodos se pueden interpretar como microprocesadores de una red de interconexión. El patrón de interconexión entre nodos y canales se conoce como la *topología* de la red de interconexión.

Sea $F \subseteq C$ el conjunto de los canales libres $(free)$ del conjunto de canales C de la red de interconexión. El conjunto $F_i \subseteq C_i$, son los canales libres en el nodo n_i. Cada instrucción *usar* del algoritmo de encaminamiento, elimina el canal c_i del conjunto F.

El algoritmo de encaminamiento se representa como una relación de encaminamiento **R**, en donde ρ es una función selección. R retorna las rutas (o canales para los algoritmos de tipo incremental) y ρ selecciona entre estas rutas o canales.

1. $R = N \times N \mapsto \rho(P)$ Se selecciona la ruta completa desde el nodo origen y se encadena al paquete.

2. $R = N \times N \mapsto \rho(C)$ Esta relación de encaminamiento es incremental porque se realiza en cada nodo intermedio.

3. $R = C \times N \mapsto \rho(C)$ Esta relación de encaminamiento es incremental porque se realiza en cada nodo intermedio pero solo se toma en cuenta los canales previamente utilizados.

La función de selección para algoritmos de tipo incremental, se detalla en estos términos $S : \rho(C \times F) \mapsto C$. Esta función selecciona entre los canales libres seleccionados por el algoritmo de encaminamiento, tomando en cuenta el estado de estos canales.

Una función de encaminamiento R de una red de interconexión I está conectada si y solo si.

$$\forall x, y \in N, x \neq y, \exists c_1, c_2, ..., c_k \in C \; tal \; que = \begin{cases} c_i \in R(x,y) \\ c_m \in R(d_{m-1}, y), m = 2, ..., k \\ d_k = y \end{cases}$$

Esto significa que es posible establecer una ruta $P(x,y) \in \rho(C)$ entre los nodos x y y utilizando los canales brindados por R.

Dada una red de interconexión I, una función de encaminamiento R y un par de canales adyacentes denominados c_i y $c_j \in C$. Existe dependencia directa entre c_i y c_j si y solo si.

$\exists x \in N$ tal que $c_i \in R(s_i, x)$ y $c_j \in R(d_i, x)$.

Esto significa que el canal c_j, puede ser solicitado inmediatamente después de usar c_i para algún nodo destino x. Es preciso notar que c_i y c_j son adyacentes.

Una subfunción de encaminamiento R_1, de una función de encaminamiento R, se define sobre el dominio de R que brinda un conjunto de canales.

$R_1(x,y) \subseteq R(x,y), \forall x, y \in N$.

La función de encaminamiento R brindá un conjunto de canales C. La subfunción $R_1 \subseteq R$ brinda un conjunto de canales $C_1 \subseteq C$. Las rutas de R, que no están en R_1 se llaman relación complementaria $R_C = R - R_1$.

Dada una red de interconexión I, una función de encaminamiento R, una subfunción de encaminamiento R_1 y un par de canales no adyacentes denominados c_i y $c_j \in C_1$. Existe dependencia indirecta entre c_i y c_j si y solo si.

$$\forall x,y \in N, x \neq y, \exists c_1, c_2, ..., c_k \in C \; tal \; que = \begin{cases} c_i \in R(x,y) \\ c_m \in R(d_{m-1}, y), m = 2, ..., k \\ d_k = y \end{cases}$$

Esto significa que el canal c_j, puede ser solicitado inmediatamente después de usar c_i para algún nodo destino x. Es preciso notar que c_i y c_j son adyacentes.

Una Grafo de Dependencia entre Canales (GD) de una red de interconexión I y una función de encaminamiento R es un grafo dirigido GD=G(V,E). Los vértices V son los canales de I. Los arcos E, son los pares de canales c_i, c_j de forma tal que existe una dependencia directa desde c_i a c_j.

Una Grafo de Dependencia Extendido entre Canales (GDE) de una red de interconexión I y una subfunción de encaminamiento R_1 de una función de encaminamiento R es un grafo dirigido GDE=$G(V_1, E_E)$. Los vértices V_1 son los canales brindados por R_1 para algunos destinos. Los arcos E_E, son los pares de canales c_i, c_j de forma tal que existe algún tipo de dependencia desde c_i a c_j.

El término *VLSI* es la sigla en idioma inglés de *Very Large Scale Integration*, que significa integración a escala muy grande de circuitos integrados.

El término *CMPs* es la sigla en idioma inglés de *Chip of Multiprocessors*, que significa integración de nucleos similares de microprocesadores como un solo circuito integrado.

El término *SoCs* es la sigla en idioma inglés de *Systems of Chips*, que significa integración de diferentes microprocesadores como un solo circuito integrado.

El término *OCIN* es la sigla en idioma inglés de *On-Chip Inteconnection Network*, que significa red de interconexión entre circuitos integrados.

El término *NoC* es la sigla en idioma inglés de *Network of Chips*, que significa red de circuitos integrados.

La *Latencia* es la suma de todos los retardos que acumula un mensaje que se desplaza por la red. Es el tiempo total que tarda un mensaje para ir desde un origen a un destino de la red.

Un *Flit* es una unidad de control de flujo y la menor unidad de asignación de recursos en un encaminador. Los flits pueden subdividirse en *phits* para ser manipulados por el encaminador.

El *Control de Flujo* es la planificación y asignación de los recursos de la red de interconexión.

Un *Paquete* es la unidad de encaminamiento dentro de la red de interconexión. Los paquetes son divididos en unidades más pequeñas pero todos siguen la misma ruta hasta alcanzar el destino.

Un *Mensaje* es la unidad lógica de transferencia de la interface red. Los mensajes son divididos en unidades más pequeñas llamadas paquetes para ser manipulados por la red.

Apéndice B

Definiciones sobre los Vehículos Guiados Automáticamente

En este apéndice presentamos las definiciones de los conceptos básicos de los vehículos guiados automáticamente (AGVs). Este apéndice se apoya principalmente de los términos utilizados en los trabajos de [Ling 99], [Reveliotis 00]. Para profundizar sobre estos términos le recomendamos al lector la siguiente literatura [Nof 99].

AGV: Vehículo guiado automáticamente.

AGVS: Sistema de Vehículos guiados automáticamente.

Almacén de Distribución para AGVS: Mas conocido como Warehouse Distribution Center (en idioma inglés). Es un depósito automatizado en donde se movilizan los *AGV*. Usualmente está dividido en tres áreas llamadas entrada, almacenamiento y depósito.

Tractor AGV: Es conocido como Tow AGV o Tugger AGV. Es un tipo de *AGV* capaz de arrastrar vagones y usualmente operan en circuitos. Tiene la particularidad de que no puede desplazarse en sentido inverso.

Sistema Administrador AGVS: Es un sistema controlador utilizado para coordinar el tráfico y administrar el conjunto de operaciones del *AGVS*.

Ruta Virtual: Es la ruta creada vía inalámbrica para que circulen los *AGV*. Se asume que las rutas deben de ser de mayor longitud que los *AGV*.

Semento de la Ruta: Es la unidad de división de una ruta, que inicia en una estación y termina en otra estación.

Punto de Decisión: Es cualquiera ubicación en donde los segmentos divergen, aunque no exista una estación.

Marcas de Navegación: Son dispositivos instalados en puntos predeterminados a lo largo de las rutas. Pueden ser de tipo magnético, electrónico como laser etc. Son usados

como puntos de referencia para corregir la navegación de los vehículos.

Código de Marcas de Navegación: Son códigos enviados a los puntos de decisión para informar al *AGV* sobre un cambio en la velocidad, una parada programada etc. Claramente, solo se aplica para las marcas de navegación electrónica.

Punto de Iniciación: Son los lugares en donde los *AGV* son introducidos al sistema. Estos puntos pueden ser también, puntos de servicio y recarga de las baterias.

Distancia de Acarreo: Se define como la distancia que el *AGV* tiene que viajar desde la estación origen (punto de carga) hasta la estación destino (punto de descarga).

Tiempo de Recogida y Depósito: Es el tiempo requerido por el vehículo para recoger y descargar la carga en los puntos de carga y descarga respectivos. Este tiempo dependerá del tipo de vehículo *AGV* y usualmente oscila entre 10 y 40 segundos.

Tiempo de Manipulación de Material: Es el tiempo requerido para recoger y descargar todas las cargas. Está dado por Tiempo de Manipulación de Material= Tiempo de Recogida y Deposito x Número total de cargas movidas por 2.

Velocidad del Vehículo AGV: Es la velocidad a la cual se mueve el vehículo dentro del área de trabajo.

Factor de Retardo de Viaje Cargado/Descargado: Es el porcentaje en que se ve reducida la velocidad del *AGV* cuando está cargado/descargado. Generalmente, los *AGV* cargados tardan más en moverse, pero esa diferencia es mínima.

Tiempo de Viaje Total: Es el tiempo total requerido por el *AGV* para satisfacer los requerimientos de movimiento. Está dado por Tiempo de Viaje Total= Factor de Retardo de Viaje Cargado + Factor de Retardo de Viaje Descargado + Tiempo de Manipulación de Material.

Rango Libre AGVs: El término es más conocido en inglés por *Free Ranging AGVs*, y se aplica a los vehíclos *AGV s* que son capaces de moverse por rutas virtuales.

Acoplado para AGVs: El término es más conocido en inglés por *Tandem guided-path*. Se refiere a un enfoque de modelado para AGVs en donde cada uno sirve una zona del área de trabajo.

Apéndice C

Definiciones básicas y terminología sobre las redes de Petri

En este apéndice se presentan las principales definiciones de los elementos relacionados con las Redes de Petri utilizados en esta memoria. El lector interesado en profundizar en este tema puede encontrar mayor información en: [Silva 85][Murata 89][Petri 62].

C.1. Conceptos básicos y terminología

Una *Red de Petri generalizada (RdPG),* $\mathcal{N}$ es una cuadrupla $\mathcal{N} = \langle P, T, Post, Pre \rangle$, donde,

1. P es un conjunto finito y no vacío de elementos llamados *lugares*.
2. T es un conjunto finito y no vacío de elementos llamados *transiciones*, tal que, $P \cap T = \emptyset$.
3. Pre (Post) es la función de incidencia previa (posterior):
   - Pre: $(P \times T) \rightarrow \mathbb{N}$.
   - Post: $(P \times T) \rightarrow \mathbb{N}$.

Una Red de Petri se representa gráficamente mediante un grafo orientado con dos tipos de nodos: lugares y transiciones. Los lugares se representan mediante círculos y las transisiones mediante rectángulos. Existe un arco orientado de un lugar p a una transición t sii $Pre(p,t) \neq 0$. Análogamente, existe un arco de una transición t a un lugar p sii

$Post(p,t) \neq 0$. Cada arco orientado se etiqueta por un entero natural, $Pre(p,t)$ ó $Post(p,t)$ que se denomina *peso del arco*. Por convenio, un arco no etiquetado posee un peso unitario.

Sea $\mathscr{N} = \langle P, T, Post, Pre \rangle$ una RdPG. Entonces, se puede definir alternativamente como $\mathscr{N} = \langle P, T, \Gamma, V \rangle$ donde:

1. $\Gamma \subseteq (P \times T) \cup (T \times P)$ es el grafo subyacente, de manera que existe un arco $(p,t) \in \Gamma$ sii $Pre(p,t) \neq 0$. Análogamente, existe un arco $(t,p) \in \Gamma$ sii $Post(p,t) \neq 0$.

2. $V : \Gamma \to \mathbb{N}$ se denomina una valuación de arcos, y se define como $V(p,t) = Pre(p,t)$ y $V(t,p) = Post(p,t)$.

Sea $\mathscr{N} = \langle P, T, Post, Pre \rangle$ una RdPG, y sean $t \in T$ y $p \in P$. Se definen los siguientes conjuntos:

1. Conjunto de lugares de entrada de t : ${}^\bullet t = \{p \in P | Pre(p,t) \neq 0\}$.

2. Conjunto de lugares de salida de t : $t^\bullet = \{p \in P | Post(p,t) \neq 0\}$.

3. Conjunto de transiciones de entrada de p : ${}^\bullet p = \{t \in T | Post(p,t) \neq 0\}$.

4. Conjunto de transiciones de salida de p : $p^\bullet = \{t \in T | Pre(p,t) \neq 0\}$.

5. Si $E \subseteq P \cup T$, ${}^\bullet E = \bigcup_{x \in E} {}^\bullet x$, y $E^\bullet = \bigcup_{x \in E} x^\bullet$.

Una RdP $\mathscr{N} = \langle P, T, Post, Pre \rangle$ se dice ordinaria sii $\forall p \in P, t \in T, Post(p,t) \in \{0,1\}$ y $Pre(p,t) \in \{0,1\}$.

Una *máquina de estados* es una red ordinaria tal que para cada transición $t \in T$, $|{}^\bullet t| = |t^\bullet| = 1$.

Sea $\mathscr{N} = \langle P, T, Post, Pre \rangle$ una RdPG. La matriz $C = Post - Pre$ se denomina *matriz de incidencia* de la red. También se suele denominar *matriz de flujo* de la red.

1. Un lugar se dice puro sii ${}^\bullet p \cap p^\bullet = \emptyset$.

2. Una transición se dice pura sii ${}^\bullet t \cap t^\bullet = \emptyset$.

3. Una RdP se dice pura sii todos sus lugares (o transiciones) son puros.

Un *camino* π de $\mathcal{N}$ es una secuencia de nodos $\pi = x_1\ x_2\ ...\ x_n$ pertenecientes a $P \cup T$ tal que $x_{i+1} \in x^\bullet$, para todo $i \in \{1,...,n-1\}$. Un camino es *simple* si todos los nodos son diferentes. Un *circuito* es un camino tal que $x_1 = x_n$. Un *circuito simple* es un circuito tal que todos los nodos, excepto el primero y el último son diferentes.

C.1.1. Funcionamiento de una red de Petri

Un *marcado m* de una RdP $\mathcal{N}$ es una función: $m : P \to \mathbb{N}$.

Las marcas son representadas por puntos negros dentro de los lugares.

El *soporte* de un marcado, $\|m\|$, es el conjunto de lugares los cuales están marcados en m, i.e. $\|m\| = \{p \in P \mid m[p] \neq 0\}$.

Una *red de Petri marcada* es el par $\langle \mathcal{N}, m_0 \rangle$, donde $\mathcal{N}$ es una RdP y m_0 es un *marcado inicial* de la red.

Se dice que $\mathcal{N}$ es la estructura del sistema mientras que m_0 representa el estado del sistema.

Sea $\langle \mathcal{N}, m_0 \rangle$ una RdP marcada. Una transición $t \in T$ está *sensibilizada* (también *habilitada*) sii $\forall p \in {}^\bullet t\ .\ m_0[p] \geq Pre(p,t)$, lo cual es denotado por $m_0[t\rangle$.

El *disparo* de una transición sensibilizada t cambia el estado del sistema a $\langle \mathcal{N}, m_1 \rangle$, donde $\forall p \in P\ .\ m_1[p] = m_0[p] + C[p,t]$, esto es denotado por $m_0[t\rangle m_1$.

Una *secuencia de disparos* σ de $\langle \mathcal{N}, m_0 \rangle$ es una secuencia de transiciones no–vacia $\sigma = t_1 t_2 ... t_k$ tal que $m_0[t_1\rangle m_1[t_2\rangle ... m_{k-1}[t_k\rangle$. El disparo de σ es denotado por $m_0[\sigma\rangle t_k$.

Se denomina vector *característico* asociado $\bar{\sigma}$ de una secuencia de disparo σ a la aplicación $\bar{\sigma} : T \to \mathbb{N}^{|T|}$ tal que $\bar{\sigma}(t)$ es el número de veces que la transición t aparece en la secuencia. $\bar{\sigma}$ se denomina también *vector contador de disparos*. El soporte de $\bar{\sigma}$ es denotado por $\|\bar{\sigma}\|$.

Un marcado m es *alcanzable* de $\langle \mathcal{N}, m_0 \rangle$ sii existe una *secuencia de disparos* σ tal que $m_0[\sigma\rangle m$. El *conjunto de alcanzabilidad* $RS(\mathcal{N}, m_0)$ es el conjunto de marcados alcanzables, i.e. $RS(\mathcal{N}, m_0) = \{m \mid \exists\ \sigma\ .\ m_0[\sigma\rangle m\}$.

C.1.2. Componentes estructurales de redes de Petri

Un *p-semiflujo* (*t-semiflujo*) es un vector $Y \in \mathbb{N}^{|P|}$, $Y \neq \mathbf{0}$ ($X \in \mathbb{N}^{|T|}$, $X \neq \mathbf{0}$), el cual es un anulador izquierdo (derecho) de la matriz de incidencia, $Y \cdot C = \mathbf{0}$ ($C \cdot X = \mathbf{0}$).

El soporte de un p-semiflujo (t-semiflujo) es denotado $\|Y\|$ ($\|X\|$), y sus lugares (transiciones) están cubiertas por Y (X).

Un *p-semiflujo mínimo* (*t-semiflow mínimo*) es un p-semiflujo (t-semiflujo) tal que el máximo común divisor de sus componentes no–nulas es uno y su soporte $\|Y\|$ ($\|X\|$) no contiene el soporte de ningún otro p-semiflujo (t-semiflujo).

Sea $\mathcal{N} = \langle P, T, C \rangle$ una RdP. Un subconjunto de lugares $D \subseteq P$ es un *cerrojo* si, y solo si, ${}^\bullet D \subseteq D^\bullet$.

Sea $\mathcal{N} = \langle P, T, C \rangle$ una RdP. Un subconjunto de lugares $D \subseteq P$ es una *trampa* si, y solo si, $D^\bullet \subseteq {}^\bullet D$.

C.2. Propiedades de las Redes de Petri

C.2.1. Propiedades estructurales

Una RdP $\mathcal{N}$ es *conservativa* (*consistente*) sii cada lugar (transición) está cubierto por un p-semiflujo (t-semiflujo).

Un p–semiflujo, y define la siguiente propiedad de invarianza:

$$\forall \mathbf{m_0} \cdot \forall \mathbf{m} \in RS(\mathcal{N}, \mathbf{m_0}) \cdot y \cdot \mathbf{m} = y \cdot \mathbf{m_0}$$

(ley de comportamiento cíclico).

C.2.2. Propiedades dinámicas

Una transición $t \in T$ es *viva* sii para cada marcado alcanzable $m \in RS(\mathcal{N}, m_0)$, $\exists m' \in RS(\mathcal{N}, m)$ tal que $m'[t\rangle$.

El sistema $\langle \mathcal{N}, m_0 \rangle$ es *vivo* sii cada transición es viva. De otra manera, $\langle \mathcal{N}, m_0 \rangle$ es *no–viva*.

Una transición $t \in T$ es *muerta* sii no existe un marcado alcanzable $m \in RS(\mathcal{N}, m_0)$ tal que $m[t\rangle$.

El sistema $\langle \mathcal{N}, m_0 \rangle$ es un *bloqueo total* sii cada transición está muerta, i.e. ninguna transición está sensibilizada.

El sistema $\langle \mathcal{N}, m_0 \rangle$ está *libre de bloqueo* sii $\forall m \in RS(\mathcal{N}, mo)$ existe al menos una transición disparable.