

# Anexos

## A. Datos calculados para el análisis

### A.1. Representatividad de los componentes gigantes

En la tabla 1 se muestran los porcentajes que representa el componente gigante débilmente conexo con respecto a la totalidad de la red. Los componentes gigantes de #16m y #22m constituyen un porcentaje más bajo del total que el resto. Esto se debe a que han quedado fuera muchos mensajes que, a pesar de contener el *hashtag*, no tenían relación con el proceso de comunicación del movimiento. Por ejemplo, #16m puede referirse al 16 de mayo por otro motivo ajeno a la movilización o incluso significar algo totalmente distinto (16 minutos, 16 millones...). Todos los demás son más exclusivos y dejan fuera *tweets* que no han interactuado con el grueso de la red.

<i>Hashtag</i>	Porcentaje nodos	Porcentaje aristas
#15msectorradical	91,73	96,93
#16m	<b>58,21</b>	<b>70,87</b>
#alabolsa	76,35	85,84
#asambleafcom	95,67	98,88
#bankiaesnuestra	84,64	92,95
#cierrabankia	90,05	95,86
#desalojosol	93,08	97,47
#larimaia	96,36	99,07
#nurembergfinanciero	95,92	97,74
#planderescateciudadano	87,11	94,09
#prima500	75,1	87,07
#12mglobal	72,89	83,15
#22m	<b>65,34</b>	<b>76,96</b>
#alaplaza12m	90,95	96,34
#es15m	83,65	92,72
#feliz15m	76,74	89,93
#huelgadeclase	78,23	89,22
#yovoy12m	87,41	95,19
#12m15m	93,94	98,47
#15mparato	89,7	92,58
#lacaixaesmordor	97,69	99,59
#occupymordor		

Cuadro 1: Comparación de tamaños entre componente gigante y red completa.

## A.2. Coeficientes de clustering y distancias promedio

Se recogen los datos de las redes reales ( $C$  y  $L$ ) y sus equivalentes en una red aleatoria del mismo tamaño y grado medio. Las relaciones permiten comprobar su estructura de “mundo pequeño”.

<i>Hashtag</i>	Tipo	$C$	$C_{random}$	$C/C_{random}$	$L$	$L_{random}$	$L/L_{random}$
#15msectorradiacal	A	0,064	0,00333	<b>19,23</b>	4,673	2,742	<b>1,7</b>
#16m	A	0,087	0,00048	<b>181,2</b>	6,071	2,821	<b>2,15</b>
#alabolsa	A	0,091	0,00301	<b>30,23</b>	1,534	2,356	<b>0,65</b>
#asambleafcom	A	0,172	0,00101	<b>172</b>	4,258	3,676	<b>1,16</b>
#bankiaesnuestra	A	0,081	0,00099	<b>81,26</b>	3,681	2,672	<b>1,38</b>
#cierrabankia	A	0,047	0,00042	<b>112,84</b>	6,725	2,741	<b>2,45</b>
#desalojosol	A	0,170	0,00034	<b>498,47</b>	4,701	3,023	<b>1,56</b>
#larimaia	A	0,096	0,00096	<b>99,97</b>	4,170	4,011	<b>1,04</b>
#nurembergfinanciero	A	0,079	0,00095	<b>83,54</b>	4,205	2,773	<b>1,52</b>
#planderescateciudadano	A	0,101	0,00012	<b>835,66</b>	3,713	2,872	<b>1,29</b>
#prima500	A	0,044	0,00022	<b>202,61</b>	4,483	2,425	<b>1,85</b>
#12mglobal	B	0,059	0,00030	<b>198,63</b>	2,715	2,464	<b>1,1</b>
#22m	B	0,082	0,00014	<b>590,2</b>	9,201	2,607	<b>3,53</b>
#alaplaza12m	B	0,055	0,00030	<b>181,45</b>	5,369	2,944	<b>1,82</b>
#es15m	B	0,083	0,00025	<b>327,68</b>	6,720	2,797	<b>2,4</b>
#feliz15m	B	0,105	0,00020	<b>530,26</b>	6,761	3,138	<b>2,15</b>
#huelgadeclase	B	0,047	0,00018	<b>258,17</b>	7,074	2,639	<b>2,68</b>
#yovoy12m	B	0,079	0,00015	<b>519,83</b>	6,299	3,072	<b>2,05</b>
#12m15m	A-B	0,127	0,00010	<b>1211,77</b>	5,081	4,633	<b>1,1</b>
#15mparato	A-B	0,183	0,00040	<b>452,64</b>	4,023	5,823	<b>0,69</b>
#lacaixaesmordor #occupymordor	A-B	0,149	0,00027	<b>559,56</b>	4,733	3,800	<b>1,25</b>

Cuadro 2: Coeficientes de clustering y distancias promedio.

### A.3. Exponentes de las distribuciones de ley de potencias

Según la distribución de ley de potencias,  $P(k) \sim ck^{-\gamma}$ , el exponente determinará la pendiente de la recta obtenida mediante la aproximación por mínimos cuadrados. En la tabla 3 se recogen los resultados para todos los sistemas, ordenados por tipos.

<i>Hashtag</i>	Tipo	$\gamma_{in}$	$\gamma_{out}$
#15msectorradical	A	-1,5143	-2,3313
#16m	A	-1,5702	-2,4898
#alabolsa	A	-1,3418	-2,9698
#asambleafcom	A	-1,6371	-2,2812
#bankiaesnuestra	A	-1,7238	-2,6353
#cierrabankia	A	-1,7435	-2,5601
#desalojosol	A	-1,5025	-2,6652
#larimaia	A	-1,5490	-2,2041
#nurembergfinanciero	A	-1,5087	-2,4427
#planderescateciudadano	A	-1,5919	-2,6642
#prima500	A	-1,6223	-2,9313
#12mglobal	B	-1,7346	-3,0761
#22m	B	-1,8946	-2,9212
#alaplaza12m	B	-1,7996	-2,4963
#es15m	B	-1,7076	-2,4739
#feliz15m	B	-1,8634	-2,3651
#huelgadeclase	B	-1,7699	-2,8047
#yovoy12m	B	-1,6883	-2,4725
#12m15m	A-B	-1,7759	-2,3201
#15mparato	A-B	-1,6136	-2,3636
#lacaixaesmordor	A-B	-1,61	-1,94
#occupymordor	A-B	-1,61	-1,94

Cuadro 3: Exponentes de las distribuciones de ley de potencias.

#### A.4. Código de adquisición de datos

Este es el código utilizado para realizar barridos en el motor de búsqueda de Twitter de forma automática, a través de ScraperWiki. Se accede a los datos mediante Search API. Todos ellos se almacenan en un archivo .json que se puede descargar directamente en formato .csv, más fácil de tratar.

```
import scraperwiki
import simplejson
import sys
# Extraer una página
base_url = 'http://search.twitter.com/search.json?q='
q = 'termino_busqueda'
# q es una variable que identifica el término que se quiere buscar,
# en nuestro caso, el hashtag
options = '&rpp=100&page='
page = 1
while 1:
    try:
        url = base_url + q + options + str(page)
        html = scraperwiki.scrape(url)
        print html
        soup = simplejson.loads(html)
        for result in soup['results']:

            # Guardar los datos en la base de datos de ScraperWiki
            scraperwiki.sqlite.save(["id"], result)
        page = page + 1
    except:
        print str(page) + ' pages scraped'
        break
```

