Escola d'Enginyeria de Telecomunicació i Aeroespacial de Castelldefels

UNIVERSITAT POLITÈCNICA DE CATALUNYA

TREBALL DE FI DE CARRERA

TÍTOL DEL TFC: Estudi d'un model de generació i optimització de xarxes complexes basat en l'excentricitat

TITULACIÓ: Enginyeria Tècnica de Telecomunicació, especialitat Sistemes de Telecomunicació

AUTORS: Laura Retamar
Jonathan Galvez

DIRECTOR: Francesc Comellas Padró

SEGON DIRECTOR: Alicia Miralles de la Asunción

DATA: 13 de setembre de 2011

Títol: Estudi d'un model de generació i optimització de xarxes complexes basat en l'excentricitat

Autor: Jonathan Galvez, Laura Retamar

Director: Francesc Comellas Padró

Segon director: Alicia Miralles de la Asunción

Data: 13 de setembre de 2011

Resum

Tot just ara es comença a comprendre bé l'estructura de la WWW i d'altres xarxes reals, com Internet, la xarxa telefònica, l'elèctrica, sistemes de transport -incloent les rutes aèries-, o xarxes associades a sistemes biològics o de relacions socials.
Les característiques comunes per a la majoria d'aquestes xarxes són un diàmetre petit, logarítmic amb el nombre de nodes, una concentració local de nodes alta o "clustering" (els nodes tenen molts veïns comuns) i una distribució del nombre d'enllaços per node que segueix una llei potencial (la xarxa és "scale-free") o és exponencial.
La majoria de models que s'han considerat per explicar aquesta estructura es basen en un creixement condicionat de la xarxa. El més clàssic és el que considera que els nodes que es van afegint a la xarxa s'uneixen preferentment en aquells ja existents que tenen moltes connexions. Aquest model, introduït el 1999 per A.L. Barabási i R. Albert, resulta en una distribució de graus potencial i és conegut com d'adjunció preferent o "preferential attachment".
En aquest TFC, presentem nous models per generar xarxes basats en l'excentricitat dels nodes (distància al node més allunyat). En un primer model, quan s'introdueix un nou node a la xarxa aquest té més probabilitat de connectar-se amb nodes existents de baixa excentricitat. Analitzem la mena de xarxes que apareixen i les comparem tant amb el model de Barabási-Albert com amb xarxes reals conegudes. També estudiem una variant que no afegeix nous nodes i que podria modelar xarxes, com per exemple la de les rutes aèries, en que no s'introdueixen nous nodes (p.e. nous aeroports) sinó que es connecten de forma diferent. Analitzem el cas geogràfic en el qual els nodes es troben localitzats en una superfície i els enllaços tenen una certa longitud física que cal considerar, conjuntament amb l'excentricitat, per optimitzar la xarxa.

Títol:

A model for the generation and optimization of complex networks based on the eccentricity.

Autor: Jonathan Galvez, Laura Retamar

Director: Francesc Comellas Padró
Segon director: Alicia Miralles de la Asunción

Data: 13 de setembre de 2011

Overview

Just now we are starting to fully understand the structure of the WWW and other real life networks like the Internet, the telephone network, the power grid, transportation systems (including air routes), or networks associated with biological systems or social relationships.
Common features for most of these networks are a small diameter, logarithmic with the number of nodes, a high local concentration of nodes or "clustering" (nodes have many common neighbors) and a link distribution which follows a power law (the network is scale-free) or is exponential.
Most of the models considered to explain this network structure use a growing network method. In a classical method, when a new node is introduced to the network, it is joined to other existing nodes with a probability that is proportional to the number of links that these nodes already have. This model, introduced by A.L. Barabási and R. Albert in 1999, results in a power law distribution of degrees and is known as "preferential attachment".
In this TFC we introduce and study new network models, based on the eccentricity of the nodes (distance to the node which is farthest away). In a first model, when a new node is introduced to the network it is joined to other existing nodes with a probability that is inversely proportional to their eccentricity. We study the networks that result and we compare them both with the Barabási-Albert model and with known real networks. We also study a model which does not add new nodes to the network but existing nodes are connected differently. The method can be used to model, for example, air routes. in which no new airports are introduced, but the connections are changed. We consider the particular "geographical" case where nodes have a fix location and links have a certain physical length which has to be considered, together with the eccentricity, to optimize the network.

ÍNDEX

II INTRODUCCIÓ

L'objectiu d'aquest treball final de carrera es introduir i estudiar models de grafs, basats en l'excentricitat, que permetin explicar les propietats de certes xarxes reals.

El projecte es divideix en dues parts, la primera part consisteix en generar grafs aleatoris basant-nos en la propietat de l'excentricitat i comparar-los amb xarxes reals i amb grafs obtinguts mitjançant el model clàssic de Barabási-Albert. El nostre model fa servir un mètode de creixement del graf similar al que han considerat Barabási-Albert[4], però en comptes de basar el procés de construcció del graf en el grau dels nodes, fa servir la seva excentricitat (distància que hi ha de un node al node mes llunyà), és a dir, quan un nou node s'introdueix a la xarxa i es connecta a diversos nodes ja existents, ho fa amb probabilitat inversament proporcional a l'excentricitat que tenen aquests nodes.

A la segona part del projecte es fa servir un procés d'optimització per modificar un graf inicial i obtenir-ne un de nou. Es comença amb una malla rectangular. El procés consisteix en, a cada pas i de forma aleatòria, reconnectar un enllaç, eliminar-lo o bé crear-ne un de nou de manera que el graf que resulti sigui "millor" segons la mesura d'una certa funció de cost que es basa en l'excentricitat dels nodes i la llargada física dels enllaços. L'algoritme d'optimització que s'ha considerat és el "simulated annealing". Els grafs obtinguts també es comparen amb xarxes reals.

III QUE ES UN GRAF?

Un graf és un conjunt d'objectes anomenats nodes i una selecció d'enllaços entre ells o arestes que poden ser orientades o no. Típicament un graf es representa mitjançant punts (nodes) connectats per línees (arestes). A la vida diària estem en contacte amb multitud de xarxes que podrien ser modelades com un graf, com ara poden ser la xarxa de carreteres, Internet, la xarxa elèctrica o la xarxa de drenatge d'una ciutat entre d'altres. A la Figura 1 tenim un exemple d'un graf aleatori de 10 nodes i 24 arestes.

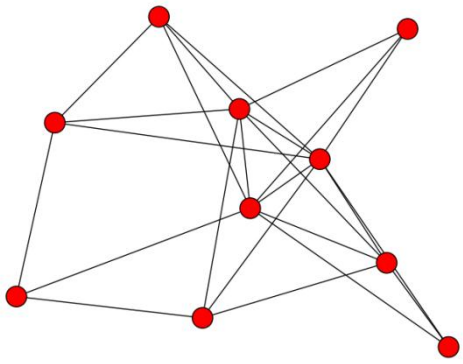


Figura 1.- Graf aleatori de 10 nodes i 24 arestes

IV XARXES COMPLEXES

Durant mes de 40 anys, la ciència tractava a totes les xarxes complexes com xarxes completament aleatòries. Aquest pensament té el seu origen en el treball que van presentar Paul Erdös i el seu col·lega Alfred Rényi l'any 1959.

La hipòtesis que proposaven era que aquest tipus de sistemes podien ser modelats connectant els seus nodes amb enllaços escollits aleatòriament. Un resultat important que es va extreure fou que tot i la disposició aleatòria dels enllaços, la majoria dels nodes del sistema resultant tenien el mateix nombre de enllaços aproximadament. D'aquesta manera el nombre d'enllaços dels nodes seguien una distribució de Poisson ben determinada i era molt difícil trobar nodes amb un nombre d'enllaços molt superior o inferior a la mitjana.

L'any 1998 Albert-laszló Barabási, Eric Bonabeau, Hawoong Jeong i Réka Albert van començar un projecte per mapejar la World Wide Web[1,2] amb la idea de trobar una xarxa aleatòria. Però aquest no va ser el resultat obtingut.

El software dissenyat per analitzar la www saltava d'una pàgina a l'altre i guardava tots els enllaços del camí que feia. Tot i què només s'analitzava una petita part de la immensitat de la Web, els resultats obtinguts van revelar que la www es sostenia per poques pàgines amb un nombre elevat d'enllaços. Mes del 80% de les pàgines tenien menys de 4 enllaços, i menys del 0,01% tenien mes de 1000.

Fent un recompte del nombre de pàgines web que tenien *k* enllaços es va demostrar que la distribució de graus seguia una llei potencial (power-law). La probabilitat de que qualsevol node estigués connectat amb *k* nodes era proporcional a $1/k^n$ *on* el valor de *n* era aproximadament 2. Es tractava d'un altre tipus de distribució, era una xarxa lliure d'escala[3,4], d'aquí el nom Scale-Free.

IV.1 **Power law**

Principalment podem identificar una xarxa SF a partir de la distribució de probabilitat[5] p(k) dels graus dels nodes. Com s'ha mencionat a l'apartat anterior, la distribució dels graus es regeixen per una llei potencial:

$$p(k) \sim ck^{-\gamma} \text{ as } k \to \infty$$

On el paràmetre γ de la funció exponencial pren valors dins del rang $2< \gamma<3$. Diversos estudis han comprovat que aquests valors són els que determinen una xarxa scale-free.

Però no totes les xarxes reals mostren un comportament de la distribució de graus de tipus scale-free.

A la Figura 2 es mostra la distribució de graus acumulada de diferents xarxes reals.
En aquest tipus de representació el que caracteritza les xarxes scale-free és una recta amb pendent negatiu on el valor absolut del pendent és el valor de γ.
Veiem doncs que en els gràfics a) i b) no es verifica aquest comportament.

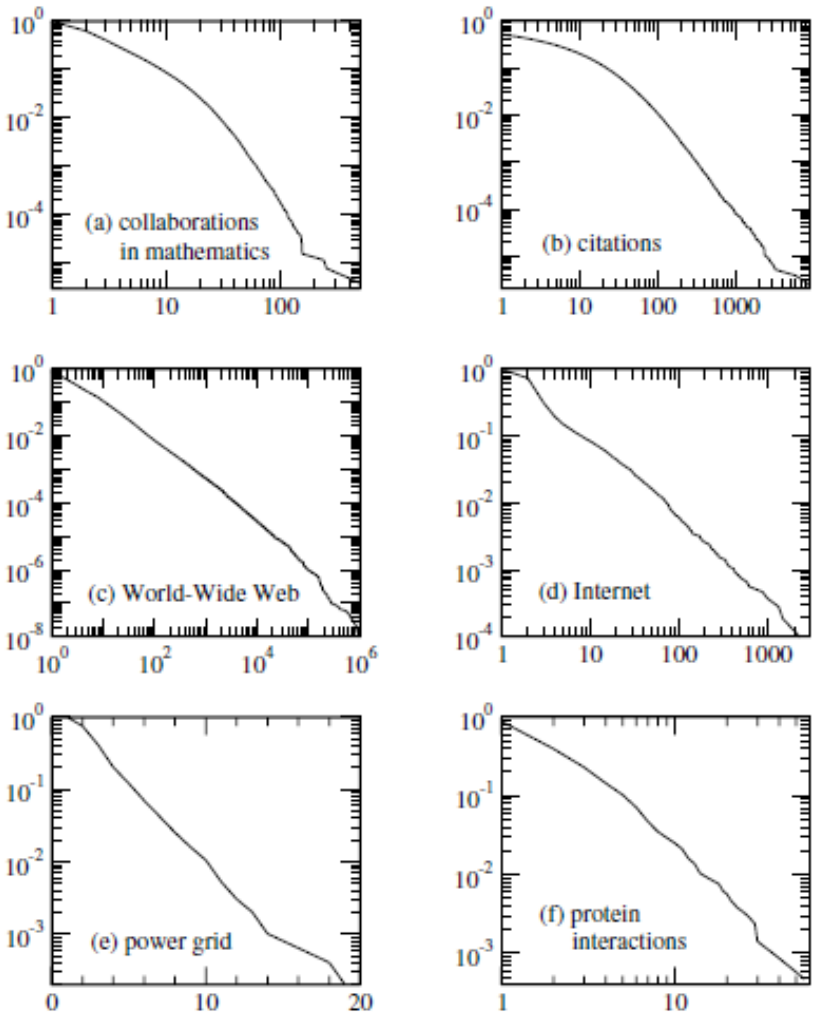


Figura 2.- Distribució de graus per a sis xarxes reals diferents. L'eix horitzontal correspon al grau k i l'eix vertical és la probabilitat acumulada de la distribució de grau. Les xarxes (c), (d) i (f) semblen tenir una distribució potencial, (b) té una distribució de grau potencial per graus elevats, però per graus petits aquesta es desvia lleugerament. La xarxa (e) té una distribució de grau exponencial i, per últim, (a) sembla tenir una distribució de grau potencial truncada, o dos distribucions de grau amb diferents exponents[6].

IV.2 **Efecte petit-món**

Els nodes amb menor grau d'una xarxa pertanyen a grans densitats de sub-grafs i aquests subgrafs són connectats entre ells a traves dels "hubs": nodes amb el grau més elevat que la majoria. A continuació es descriu aquest fenòmen en un exemple:

Considerem una xarxa social on els nodes són persones i els enllaços són la relació que els uneix amb altres persones. En una comunitat (graf complet) tothom es coneix amb tothom, però cada persona individualment coneix a gent fora del cercle de la comunitat. Algunes persones, a més a més, estan connectades amb varies comunitats diferents. Aquestes persones són les que es considerarien hubs. Aquest efecte que veiem es el que anomenarem fenomen "small-world" o petit-mon[6].

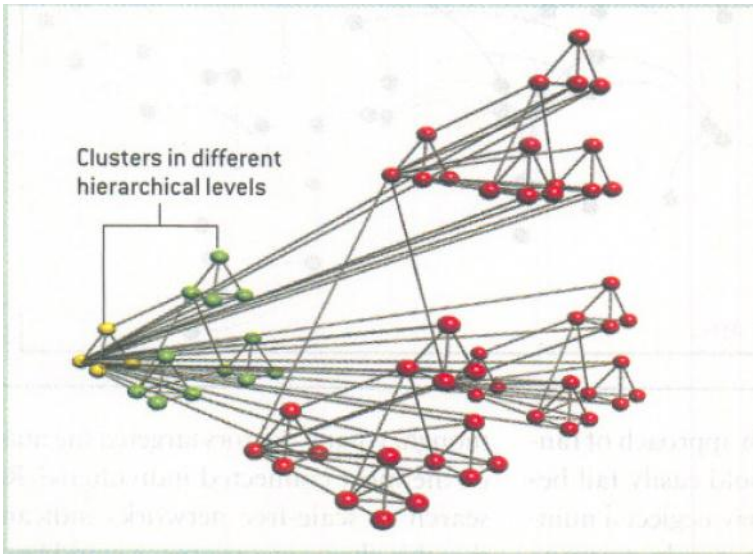


Figura 3.- Exemple de la formació d'un petit-món. Els membres de la comunitat (grocs), estan connectats amb altres persones (verds) que poden ser cosins, amics fora de la comunitat,... a la vegada aquestes persones estan connectades a d'altres comunitats (vermells).

IV.3 **Robustesa i vulnerabilitat**

Un altre característica notable de les xarxes SF es la tendència que tenen a la creació de "hubs".
Les xarxes SF són a la vegada molt robustes i molt vulnerables, depenent del tipus d'atac al que estan sotmeses. La majoria de "hubs" estan connectats a uns altres de grau mes petit. Aquests, al mateix temps, estan connectats a d'altres nodes amb un grau encara més petit i així successivament. Aquest tipus de jerarquia fa que aquestes xarxes tinguin un comportament tolerant a caigudes. En el cas de produir-se la caiguda d'un node a l'atzar, la probabilitat que un "hub" de grau més elevat sigui atacat és molt petita, degut a la gran quantitat de nodes amb grau molt petit. Si cauen aquests nodes menys connectats, només tenim problemes de forma local, la resta de la xarxa no es veu afectada. Per tant, en aquest cas la xarxa presenta una gran robustesa. Per altre banda, si s'elimina un dels principals "hubs", la xarxa es converteix en diferents grafs aïllats, en aquest cas les conseqüències son caòtiques.

Aquestes propietats han sigut estudiades analíticament per Cohen[7, 8] i per Callaway[9].

IV.1 CONCEPTES BÀSICS EN UN GRAF

La representació d'un graf ve donada per un conjunt de punts (vèrtex o nodes) units (alguns d'ells) per línies (arestes o enllaços). Formalment un graf *G = (V, E)* és un parell de conjunts *V* i *E*, on *V* és el conjunt dels vèrtex i *E* és el conjunt d'arestes.

Donarem unes definicions bàsiques que ens permetran tant interpretar els grafs com facilitar la comprensió i lectura d'aquest TFC.

- Dos nodes són **adjacents** si hi ha una aresta que els uneix.
- **L'ordre** d'un graf és el nombre de nodes que té: n = |V|.
- La **mida** d'un graf és el nombre d'arestes que té: m = | E |.
- El **grau** d'un node és el nombre d'arestes que cauen sobre ell.
- Un enllaç es **incident** a un node si aquest l'uneix a un altre node.

A continuació es descriuen quines són les propietats[10] que s'han tingut en compte per analitzar les xarxes en l'estudi d'aquest TFC.

IV.1.1 "Clustering"

El clustering mesura el grau de connectivitat local d'un graf, el coeficient varia entre 0 i 1. Un valor proper a 0 indica que molts dels nodes que són adjacents a un node no ho són entre si, mentre que valors propers a 1 ens indica que els veïns d'un node també estan connectats entre si.

IV.1.2 Diàmetre i distància

La distància entre dos nodes, es el nombre d'enllaços que conte el camí mes curt que uneix aquests dos nodes. Al treball contarem amb unitats cada enllaç sigui mes llarg o mes curt.
El diàmetre del graf es la màxima distància possible entre qualsevol parella de nodes d'un graf.
En aquest TFC s'ha tingut en compte la distància màxima, mínima i mitjana, de cada graf, també el diàmetre màxim mínim i s'ha calculat la mediana d'aquest.

IV.1.3 Correlació

El coeficient de correlació ens indica com estan distribuïts els graus per tot el graf, sent 1 quant un node esta connectat a un altre node del mateix grau, i

sent -1 quant un node esta connectat a un altre de grau superior o inferior. Enel cas que el graf estigui totalment incorrelat, coeficient de correlació pròxim a 0, voldrà dir que els nodes estan connectats a alguns nodes del mateix grau i a un altres de diferents.
Un altre cosa a tenir en compte es que, si el coeficient de correlació es proper a -1, un node amb un grau alt estarà connectat a un node de grau baix, tant el clustering com el grau mig disminuiran en funció del grau.

IV.1.4 Grau

El grau es defineix com el numero d'enllaços que surten d'un node, és a dir, si un node té 5 enllaços incidents direm que el grau d'aquest node serà 5. En aquest TFC es té en compte el grau màxim, grau mínim i grau mitjà.
A nivell de node, quant mes elevat sigui el grau mes catastròfica seria la seva caiguda per la connectivitat de la xarxa, tanmateix si el grau d'un node és petit vol dir que està pitjor connectat i la seva caiguda no afectarà a tants nodes, sempre i quant aquests estiguin connectats a d'altres nodes.
A nivell de graf com més elevat sigui el grau mitjà voldrà dir que més robust és el graf a possibles agressions de caigudes, tant d'enllaços com de nodes, tenint en compte l'optimització dels recursos s'haurà de trobar un equilibri en el que el grau mitjà sigui menor i el clustering sigui més proper a 1 així com reduir al màxim el nombre d'enllaços.

A continuació hi ha un exemple d'un graf aleatori de 10 nodes i 24 arestes, aprofitant la Figura 1, en el que estan indicats el grau de cada node.

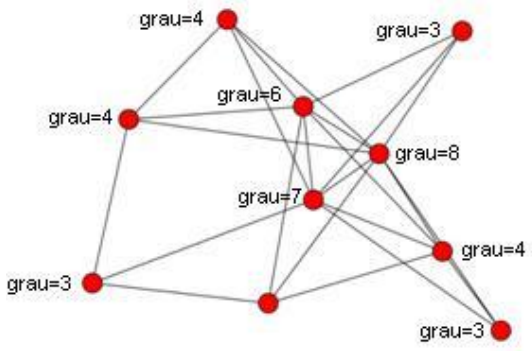


Figura 4.- Graf de 10 nodes i 24 arestes, amb graus indicats.

IV.1.5 Excentricitat

L'excentricitat d'un node es la distància més curta que hi ha des d'aquest node al node més llunya. En termes d'optimització com més baixa sigui l'excentricitat millor connectat està aquest node amb el seu node més llunya, i com més

baixa sigui aquesta distància millor connectat estarà el node amb la resta de nodes.
A continuació hi ha un exemple d'un graf aleatori de 10 nodes i 24 arestes, aprofitant la Figura 1, en el que estan indicats l'excentricitat de cada node.

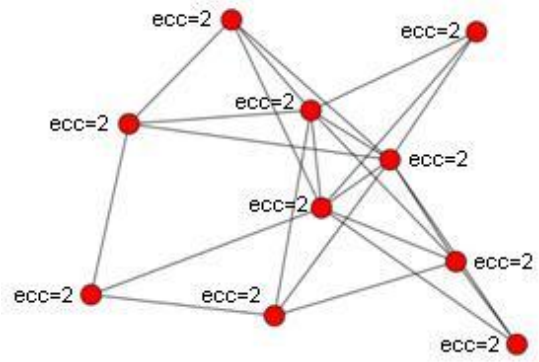


Figura 5.- Graf de 10 nodes i 24 arestes, amb excentricitat indicats.

En aquest cas podem observar que la xarxa en quant a excentricitat està perfectament connectada.

IV.2 **Model Barabási-Albert**

En el model de Barabási-Albert es construeix el graf de manera dinàmica a partir de la incorporació successiva de nodes. Cada node s'afegeix al graf connectant-lo a nodes existents seleccionats proporcionalment al seu grau.

La xarxa es comença a construir amb un conjunt m_o nodes connectats aleatòriament. En el cas de triar un node amb $m_o = 3$ es començarà a construir el graf amb 1 node amb grau 3 com veiem a la figura.

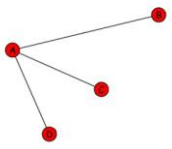


Figura 6.- Primer node per construir un graf amb $m_o = 3$

Tenint en compte que $m_o \geq 2$ i el grau de cada node de la xarxa inicial ha de ser de almenys 1, si no fos així el creixement de la xarxa podria fer que aquests nodes es quedessin desconnectats completament de la xarxa.
La probabilitat que el nou node es connecti a un node existent *i* depèn del seu grau (preferential attachment), com més gran sigui el grau més probabilitat tindrà el nou node de connectar-se a un node existent.

A la figura 7 es pot veure un exemple d'adjunció d'un nou node al graf de la Figura 6, basant-se en el preferential attachment.

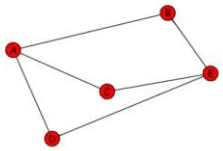


Figura 7.- Exemple d'adjunció d'un nou node "E" al graf inicial de Figura 2

Per explicar en què consisteix el Preferential attachment es té en compte el mateix exemple que va seguir Barabási-Albert.
Des del seu llançament al 1990 amb una única pàgina, la www ha experimentat un enorme creixement, amb més de tres bilions de entrades al 2003 i sobre 19,2 bilions al 2005 (Yahoo). El model de creixement però, s'ha mantingut al llarg dels anys, i el que és més important, la seva distribució potencial no ha canviat.
Normalment l'autor d'una pàgina web que ha d'enllaçar aquesta a una altre sempre ho farà amb més probabilitat a la que tingui més entrades (grau més elevat). Per tant la probabilitat que node "i" es connecti a un node "j" existent és:

$$Pv(j) = \frac{K_i(grau\ node\ i)}{k_j(grau\ node\ j)}$$

Si simulem qualsevol xarxa basada en aquest model ens donarem compte que sempre hi hauran alguns nodes amb el grau molt superior a d'altres, els quals s'anomenen "hubs".
A continuació es simula un graf petit basat en aquest model:

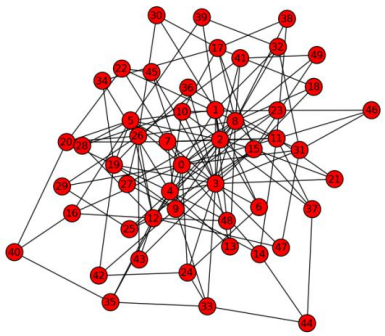


Figura 8.- Graf aleatori model Barabási-Albert de 50 nodes.

IV.3 **Model basat en l'excentricitat**

En el model basat en l'excentricitat, com en el model de Barabási-Albert, també es construeix el graf de manera dinàmica a partir de la incorporació successiva de nodes, però en aquest cas tindrem en compte l'excentricitat dels nodes existents.

Es tria l'exemple anterior per començar amb $m_o = 3$, en aquest cas es tindrà en compte l'excentricitat per adjuntar els nous nodes. Un node que tingui la excentricitat mes elevada tindrà menys probabilitat de que se li connectin nous nodes, mentre que un node amb excentricitat baixa tindrà mes probabilitat que se li connectin els nous nodes.
En el cas de començar amb $m_o = 3$ tindríem el mateix punt inicial que a la Figura 6.
Es proposa una xarxa simple, veure Figura 9, per explicar les probabilitats d'adjunció de nous nodes a una xarxa en construcció. A continuació s'explica amb de manera gràfica i numèrica, com es calcula la probabilitat de connexió.

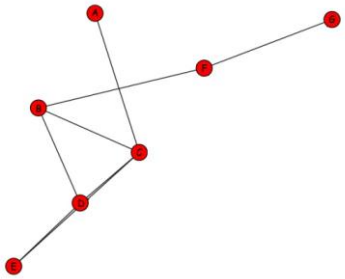


Figura 9.- Xarxa simple de 7 nodes i 9 enllaços

A la taula següent queden identificats tots els nodes amb les seves perspectives excentricitats.

Node	A	B	C	D	E	F	G
Excentricitat	4	2	3	3	4	3	4

La probabilitat de que un nou node es connecti a un altre existent es:

$$\Pr(v) = \frac{1/_{ecc(v)}}{Ecc(G)}$$

Sent "r" el nou node, ecc(v) la excentricitat de un node que ja pertany al graf i Ecc(G) la excentricitat total del graf.
S'observa que contra mes gran sigui l'excentricitat del node *v*, mes gran serà el denominador, per tant, mes petita la probabilitat de connectar-ne el node *v* al nou node *r*. L'excentricitat total del graf serà:

$$Ecc(G) = \sum_{K=0}^{K=n} \frac{1}{ecc(k)}$$

Aprofitant l'exemple de la Figura 9, es calcula la probabilitat de que un nou node es connecti a algun node ja existeixen al graf.
En el càlcul de la probabilitat de connexió d'un nou node amb un d'existent, primerament es calcula la Ecc(G), sumant les inverses de les excentricitats de tots els nodes.

$$Ecc(G) = \sum_{K=0}^{K=n} \frac{1}{ecc(k)} = \frac{1}{4} + \frac{1}{2} + \frac{1}{3} + \frac{1}{3} + \frac{1}{4} + \frac{1}{3} + \frac{1}{4} = \frac{3}{4} + \frac{3}{3} + \frac{1}{2} = \frac{54}{24}$$

Càlcul de la probabilitat de que es connecti a un node *v*:

$$\Pr(v) = \frac{1/_{ecc(v)}}{Ecc(G)} \rightarrow \Pr(A) = \frac{1/_4}{54/_{24}} = \frac{24}{54 \cdot 4} = \frac{6}{54}$$

$$\Pr(B) = \frac{1/_2}{54/_{24}} = \frac{12}{54} \qquad \Pr(C) = \frac{1/_3}{54/_{24}} = \frac{8}{54} \qquad \Pr(D) = \frac{1/_3}{54/_{24}} = \frac{8}{54}$$

$$\Pr(E) = \frac{1/_4}{54/_{24}} = \frac{6}{54} \qquad \Pr(F) = \frac{1/_3}{54/_{24}} = \frac{8}{54} \qquad \Pr(G) = \frac{1/_4}{54/_{24}} = \frac{6}{54}$$

Com es pot veure els nodes que tenen l'excentricitat més elevada son els que tenen menys probabilitat de connectar-se a un nou node, en aquest cas serien els nodes *A*, *E* i *G*.

A continuació es simula un graf petit basat en aquest model:

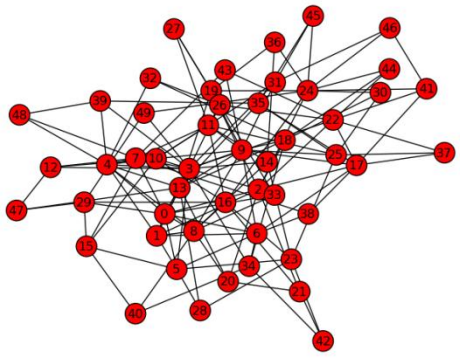


Figura 10.- Graf aleatori model basat en l'excentricitat de 50 nodes.

IV.3.1 Comparativa entre model de Barabási-Albert i model basat en l'excentricitat.

Per fer aquesta comparativa es realitzen 20 simulacions diferents per cada nombre de nodes variant la "m". S'han triat els valors m= 3, 4, 5, 8 , 9 i les dimensions dels grafs de 50, 100, 1000 i 10000 nodes, degut al gran cost computacional que té la generació d'un graf tenint en compte l'excentricitat no s'han fet simulacions amb major nombre de nodes.
Per confeccionar les taules s'han tingut en compte les característiques explicades en l'apartat *V1.1* a *V1.5* (grau, distància, diàmetre, clustering i coeff. correlació)

Resultats obtinguts amb el model de Barabási-Albert:

m	Nº rep.	Nodes	Edges	Grau			Distància			Diàmetre			Dist. Grau	Clust.	Coeff. Correlació
				Mig	Max	Min	Mitjana	Max	Min	Mig	Max	Min			
3	20	50	144	5.76	21.40	3	2.263	2.303	2.142	4	4	3	1.304	0.26821	-0.209055
4	20	50	190	7.60	23.05	4	2.074	2.119	2.029	3	4	3	1.203	0.28870	-0.152164
5	20	50	235	9.40	26.95	5	1.930	1.962	1.893	3	3	3	1.174	0.33313	-0.143615
8	20	50	364	14.56	31.75	8	1.717	1.725	1.711	3	3	3	0.992	0.42358	-0.105945
9	20	50	405	16.20	33.35	9	1.675	1.681	1.671	3	3	3	0.878	0.46122	-0.153431
3	20	100	294	5.88	30.20	3	2.584	2.624	2.505	4	5	4	1.477	0.15541	-0.140576
4	20	100	390	7.80	34.30	4	2.364	2.393	2.311	4	4	4	1.421	0.18307	-0.113364
5	20	100	485	9.70	39.20	5	2.203	2.231	2.160	4	4	3	1.373	0.21157	-0.120979
8	20	100	764	15.28	45.15	8	1.943	1.959	1.924	3	3	3	1.165	0.27176	-0.064234
9	20	100	855	17.10	49.10	9	1.886	1.896	1.875	3	3	3	1.151	0.29884	-0.074006
3	20	1000	2994	5.98	95.75	3	3.468	3.524	3.425	6	6	6	1.763	0.03289	-0.077673
4	20	1000	3990	7.98	106.35	4	3.180	3.203	3.155	5	5	5	1.794	0.03662	-0.056913
5	20	1000	4985	9.97	125.05	5	2.968	2.989	2.930	5	5	5	1.781	0.04238	-0.053310
8	20	1000	7964	15.92	153.4	8	2.662	2.674	2.647	4	4	4	1.715	0.05713	-0.034499
9	20	1000	8955	17.91	158.5	9	2.604	2.613	2.593	4	4	4	1.700	0.06084	-0.030055
3	5	10000	29994	5.99	363.0	3	4.259	4.275	4.230	7	7	7	1.857	0.00593	-0.034104
4	5	10000	39990	7.99	369.25	4	3.891	3.909	3.875	6	6	6	1.960	0.00655	-0.031601
5	5	10000	49985	9.99	401.75	5	3.652	3.666	3.640	6	6	6	1.947	0.00734	-0.027012
8	5	10000	79964	15.99	448.25	8	3.242	3.246	3.235	5	5	5	1.963	0.00967	-0.013062
9	5	10000	89955	17.99	487.5	9	3.138	3.140	3.133	5	5	5	1.941	0.01070	-0.014874

Taula 1.- Valors obtinguts amb simulacions del model de Barabási-Albert

Resultats obtinguts amb el model basat en l'excentricitat:

m	Nº rep.	Nodes	Edges	Grau			Distància			Diàmetre			Dist. Grau	Clust.	Coeff. Correlació
				Mig	Max	Min	Mitjana	Max	Min	Mig	Max	Min			
3	20	50	144	5.76	15.40	3	2.370	2.411	2.339	4	5	4	1.322	0.15764	0.287812
4	20	50	190	7.60	18.60	4	2.125	2.102	2.144	4	4	3	1.192	0.20185	0.048819
5	20	50	235	9.40	21.55	5	1.963	1.983	1.937	3	4	3	1.140	0.24728	0.026074
8	20	50	364	14.56	26.90	8	1.722	1.729	1.715	3	3	3	0.879	0.36136	0.065098
9	20	50	405	16.20	29.30	9	1.678	1.684	1.675	3	3	3	0.872	0.39257	0.073603
3	20	100	294	5.88	18.50	3	2.746	2.782	2.689	3	3	3	1.534	0.08473	0.090895
4	20	100	390	7.80	21.95	4	2.455	2.474	2.444	4	4	4	1.396	0.10990	0.112962
5	20	100	485	9.70	26.90	5	2.277	2.292	2.267	4	4	4	1.359	0.13111	0.140893
8	20	100	764	15.28	35.90	8	1.963	1.971	1.952	3	3	3	1.109	0.20800	0.110513
9	20	100	855	17.10	38.55	9	1.904	1.911	1.897	3	3	3	1.068	0.23113	0.111477
3	20	1000	2994	5.98	29.10	3	3.947	3.961	3.935	7	7	6	2.162	0.00792	0.194886
4	20	1000	3990	7.98	35.5	4	3.506	3.518	3.501	6	6	6	1.994	0.01208	0.226142
5	20	1000	4985	9.97	42.6	5	3.242	3.247	3.237	5	5	5	1.960	0.01467	0.252033
9	20	1000	8955	17.91	66.0	9	2.709	2.711	2.707	4	4	4	1.742	0.02746	0.267153
3	5	10000	29994	5.99	38.5	3	5.087	5.091	5.081	8	8	8	2.663	0.00925	0.221533
4	5	10000	39990	7.99	47.25	4	4.509	4.513	4.506	7	7	7	2.617	0.00125	0.256316
5	5	10000	49985	9.99	60.75	5	4.156	4.159	4.154	6	6	6	2.640	0.00160	0.279500
8	5	10000	79964	15.99	84.75	8	3.580	3.581	3.579	5	5	5	2.383	0.00253	0.306427
9	5	10000	89955	17.99	97.00	9	3.472	3.473	3.471	5	5	5	2.481	0.00289	0.316863

Taula 2.- Valors obtinguts amb simulacions del model basat en l'Excentricitat

Les primeres observacions que es poden fer són per les simulacions de 50 i 100 nodes. Els resultats no poden ser considerats com a rellevants, degut al nombre tant petit de nodes, no es pot fer una aproximació a cap tipus de xarxa complexa. Tot hi així ja es comença a veure que la distribució de graus no segueix una llei potencial. Els que si s'estudiaran són els resultats obtinguts per el numero de nodes més gran:

m	Nº rep.	Nodes	Edges	Grau			Distància			Diàmetre			Dist. Grau	Clust.	Coeff. Correlació
				Mig	Max	Min	Mitja	Max	Min	Mitja	Max	Min			
3	5	10000	29994	5.99	38.5	3	5.087	5.091	5.081	8	8	8	2.663	0.00925	0.221533
4	5	10000	39990	7.99	47.25	4	4.509	4.513	4.506	7	7	7	2.617	0.00125	0.256316
5	5	10000	49985	9.99	60.75	5	4.156	4.159	4.154	6	6	6	2.640	0.00160	0.279500
8	5	10000	79964	15.99	84.75	8	3.580	3.581	3.579	5	5	5	2.383	0.00253	0.306427
9	5	10000	89955	17.99	97.00	9	3.472	3.473	3.471	5	5	5	2.481	0.00289	0.316863

Taula 3.- Simulacions de 10000 nodes amb el model basat en l'Excentricitat

m	Nº rep.	Nodes	Edges	Grau			Distància			Diàmetre			Dist. Grau	Clust.	Coeff. Correlació
				Mig	Max	Min	Mitjana	Max	Min	Mig	Max	Min			
3	5	10000	29994	5.99	363.0	3	4.259	4.275	4.230	7	7	7	1.857	0.00593	-0.034104
4	5	10000	39990	7.99	369.25	4	3.891	3.909	3.875	6	6	6	1.960	0.00655	-0.031601
5	5	10000	49985	9.99	401.75	5	3.652	3.666	3.640	6	6	6	1.947	0.00734	-0.027012
8	5	10000	79964	15.99	448.25	8	3.242	3.246	3.235	5	5	5	1.963	0.00967	-0.013062
9	5	10000	89955	17.99	487.5	9	3.138	3.140	3.133	5	5	5	1.941	0.01070	-0.014874

Taula 4.- Simulacions de 10000 nodes amb el model de Barabási-Albert

Com era d'esperar els dos models, al tenir la mateixa manera de construir el graf (de manera dinàmica a partir de la incorporació successiva de nodes), tenen com a resultat el mateix número d'enllaços incidents *m.*

En quant al grau, a primera vista, podem comprovar com en el model basat en l'excentricitat no es creen hubs, nodes que tenen un elevat grau (nombre de enllaços incidents), d'aquesta manera s'elimina un dels principals problemes del model de Barabási-Albert i és que si caigués un dels hubs que es creen, una part local de la xarxa es quedaria incomunicada.
La distància i el Diàmetre són força semblants, els grafs generats amb el model de l'excentricitat són una mica més grans que els de Barabási-Albert, o més ben dit, els enllaços estan més repartits.

La connectivitat del graf (Clustering) varia en funció de la *m* que es determina per la construcció del graf. En tots dos casos aquest paràmetre augmenta conforme augmentem la *m*, però per BA el clustering augmenta amb diferencia més significativa que per el model basat en l'excentricitat.

El coeficient de correlació és més pròxim a 1 en el cas del model de l'excentricitat, això vol dir que hi han més nodes connectats amb el mateix grau que en el model de Barabási-Albert. Aquest resultat era d'esperar degut a la presència de hubs per el model de Barabási-Albert.

Quan s'analitza la distribució de grau, en tots els grafs generats observem el fet comú que els dos models treballats no segueixen la mateixa llei. En els grafs

generats amb el model de Barabasi es veu clarament que la distribució segueix una llei potencial tan per la probabilitat p(k) com per la probabilitat acumulada P(k) mentre que no és així amb l'altre model. Per mostrar aquests fets hem triat dues simulacions realitzades amb els mateixos paràmetres m=4 i un nombre de 1000 nodes, tenint en compte els dos models esmentats anteriorment. A continuació mostrem aquestes diferències. Els dos primers gràfics mostren les probabilitat i probabilitats acumulades pel model de Barabasi mentre que els dos gràfics següents corresponen al graf generat amb el model de l'excentricitat.

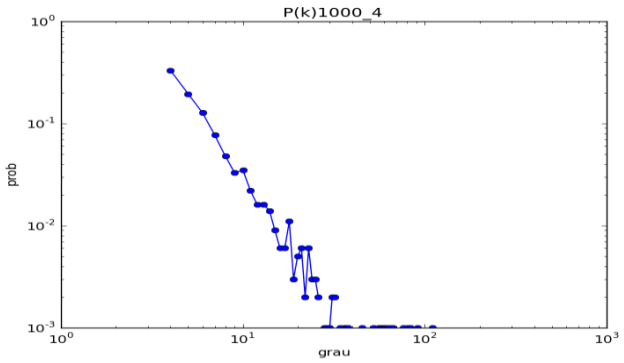


Figura 11.- Distribució de grau p(k) pel model de Barabasi. El pendent de la recta de regressió és -3

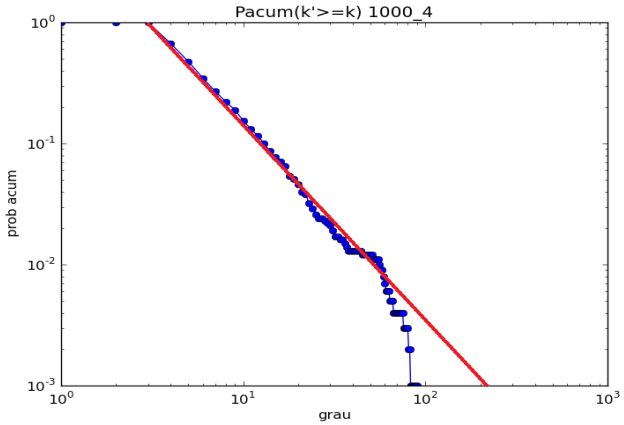


Figura 12.- Distribució de grau P(k) pel model de Barabasi. El pendent de la recta de regressió és -2

Observem el fet que caracteritza a les xarxes “scale-free”: la probabilitat que un node tingui grau k segueix la llei $p(k) \sim k^{-\gamma}$ i la probabilitat que un node tingui un grau superior o igual a k és $P(k) \sim k^{1-\gamma}$.
A les anteriors figures el valor de gamma és aproximadament 3.

A les gràfiques obtingudes seguint el model de la excentricitat no observem aquest fet.

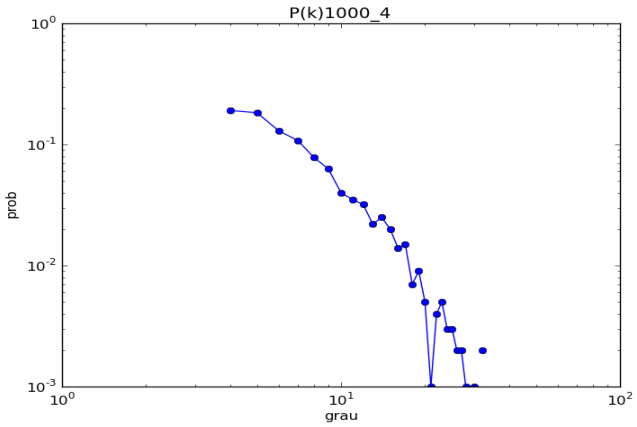


Figura 13.- Distribució de grau p(k) seguint el model de l'excentricitat

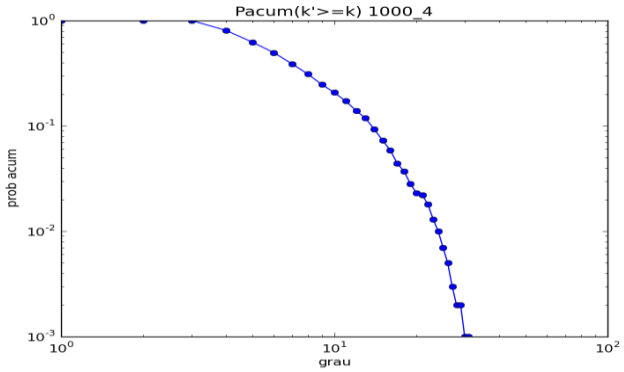


Figura 14.- Distribució de grau P(k) seguint el model de l'excentricitat

Hem analitzat altres tipus de gràfics.
Al representar la probabilitat p en funció de k hem observat una dependència exponencial que sembla del tipus $p(k) = c \cdot \exp(-\alpha \cdot k)$ on c i α són constants. Ho hem comprovat considerant l'expressió:

$$\log\big(p(k)\big) = \log(c) - \alpha \cdot k$$

Al representar $log(p(K))$ en funció de k obtenim una recta on podem determinar els valors de c i de α. En el nostre cas obtenim c~ 0.78 i α ~ 0.09.

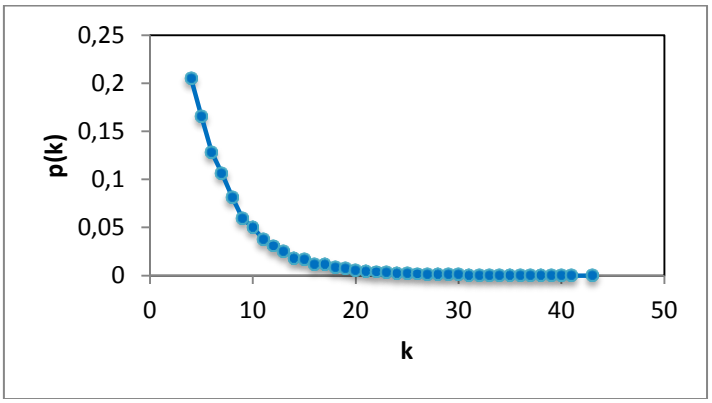


Figura 15.- Exemple distribució de grau mitjana p(k) per a 1000 nodes amb 20 simulacions.

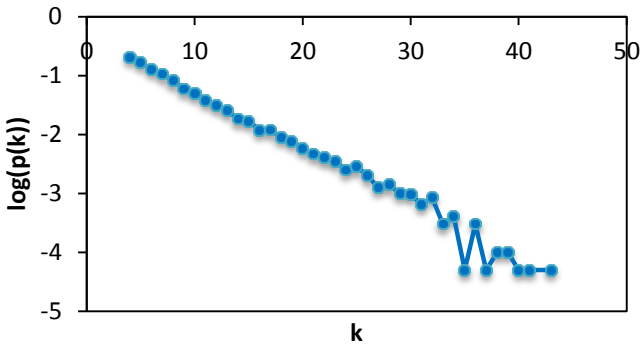


Figura 16.- Distribució de grau acumulada P(k). Es veu com dona una recta, la qual ens proporciona el paràmetre α

V XARXES COMPLEXES REALS

Gràcies als avenços en els càlculs i a la realització d'estudis mes exhaustius s'ha pogut demostrar que moltes de les xarxes que representen sistemes complexos no tenen una estructura aleatòria com s'havia suposat fins aleshores.

V.1 Tipus de xarxes complexes[11]

En aquests apartat s'explicaran alguns dels casos de xarxes reals, la seva estructura, propietats i exemples.
S'expliquen els 4 grans grups de xarxes reals: Socials, d'Informació, Tecnològiques i Biològiques.

V.1.1 Xarxes socials

Es considera una xarxa social com un conjunt de persones o grups de persones que segueixen un patró de contactes o interaccions entre ells (amistat,negocis, etc.). Tot i que aquest tipus de xarxes s'han estat investigant des dels anys 30, estudis més recents s'han centrat més en fer anàlisis a "xarxes de col·laboració" Petit Mon[12], que es definiran en el proper capítol de Xarxes Scale-free.
Un exemple n'és la xarxa de col·laboració que hi ha entre els actors de Hollywood, documentat exhaustivament a www.imdb.com, i des de on a partir del nom de dos actors que apareixen a la web, es pot calcular la distància en nombre de salts que hi ha entre ells a partir de les seves col·laboracions en pel·lícules[13].

V.1.2 Xarxes d'informació

També anomenades xarxes de coneixement ("knowledge networks"). Dins de les xarxes d'informació es poden destacar dos exemples:
El primer seria la relació de cites que hi ha entre publicacions acadèmiques, a partir de les publicacions i les referències entre elles. Cal comentar però, que la xarxa resultant és directiva i sense cercles, donat que no es pot fer referència a un article que encara no s'ha escrit.
Un altre exemple és la xarxa www[14] considerant el conjunt de pàgines web i els seus hiperlinks com a vèrtexs i arestes respectivament. En aquest cas la xarxa resultant és també Petit Mon, i el graf també és directiu, amb la diferència que alguns enllaços son bidireccionals i es poden donar cercles.

V.1.3 Xarxes tecnològiques

Aquest tipus de xarxes han estat, almenys en principi, dissenyades per l'home per a distribuir recursos. Hi ha diversos exemples: les xarxes elèctriques, les xarxes de rutes aèries, ferroviàries, xarxa telefònica, correus...fins i tot els circuits electrònics son una forma de xarxa de distribució tecnològica. Un dels casos més interessants és Internet, sobretot en quan al funcionament de les connexions entre els grans routers.
Per estudiar la xarxa primer cal tenir en compte una sèrie d'aspectes: La infraestructura física que permet aquestes connexions no és fàcil d'implementar ja que depèn de diferents organitzacions. La forma de fer-ho és mitjançant programes que realitzen traces entre dos punts. Les dades dels nodes entremitjos son emmagatzemades fins que es determina l'estructura global de la xarxa. Ara bé, com a conseqüència d'aquesta metodologia sempre hi ha un conjunt de vèrtexs (així com un conjunt de enllaços) que mai son mostrejats, de forma que aquesta aproximació pot arribar a ser prou bona, però no perfecte[15].
Un dels punts més interessants d'aquestes arquitectures és que les seves estructures estan governades per l'espai i la geografia, de forma que la distància entre els seus nodes juga un paper important, tot i que encara no es sap fins a quin punt.

V.1.4 Xarxes biològiques

Hi ha nombrosos sistemes biològics que també formen xarxes complexes. L'exemple més característic son els camins metabòlics[16], basats en la representació dels substrats i amb enllaços unidireccionals que representen les reaccions metabòliques existent i donen lloc als productes. Una altra tipus de xarxa biològica és la xarxa reguladora genètica. L'expressió d'un gen, la producció per transcripció i traducció de les proteïnes per a les quals es codifica el gen, pot ser controlada per la presència d'unes altres proteïnes, tant activadors com inhibidors, de manera que el genoma per si mateix forma una xarxa de commutació amb vèrtexs que representen les proteïnes i enllaços dirigits que representen les dependències entre elles.
Per tancar el capítol es mostra una taula on es poden observar els diferents

tipus de xarxes que hi ha a la realitat amb les propietats més importants definides anteriorment així com una petita taula comparativa de les xarxes reals amb el model basat en l'excentricitat.

A continuació simulem una xarxa real en concret la Marine food Web.

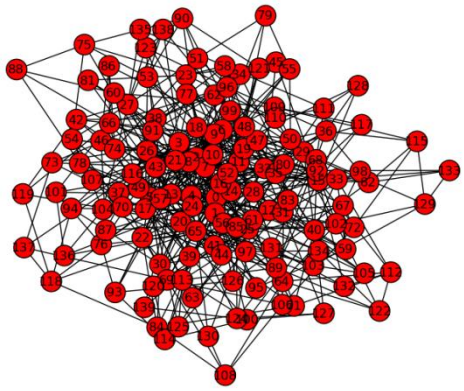


Figura 17.- Simulació xarxa real, Marine food Web

	Network	Type	n	m	z	ℓ	α	$C^{(1)}$	$C^{(2)}$
Social	film actors	undirected	449 913	25 516 482	113.43	3.48	2.3	0.20	0.78
	company directors	undirected	7 673	55 392	14.44	4.60	–	0.59	0.88
	math coauthorship	undirected	253 339	496 489	3.92	7.57	–	0.15	0.34
	physics coauthorship	undirected	52 909	245 300	9.27	6.19	–	0.45	0.56
	biology coauthorship	undirected	1 520 251	11 803 064	15.53	4.92	–	0.088	0.60
	telephone call graph	undirected	47 000 000	80 000 000	3.16		2.1		
	email messages	directed	59 912	86 300	1.44	4.95	1.5/2.0		0.16
	email address books	directed	16 881	57 029	3.38	5.22	–	0.17	0.13
	student relationships	undirected	573	477	1.66	16.01	–	0.005	0.001
	sexual contacts	undirected	2 810				3.2		
Information	WWW `nd.edu`	directed	269 504	1 497 135	5.55	11.27	2.1/2.4	0.11	0.29
	WWW Altavista	directed	203 549 046	2 130 000 000	10.46	16.18	2.1/2.7		
	citation network	directed	783 339	6 716 198	8.57		3.0/–		
	Roget's Thesaurus	directed	1 022	5 103	4.99	4.87	–	0.13	0.15
	word co-occurrence	undirected	460 902	17 000 000	70.13		2.7		0.44
Technological	Internet	undirected	10 697	31 992	5.98	3.31	2.5	0.035	0.39
	power grid	undirected	4 941	6 594	2.67	18.99	–	0.10	0.080
	train routes	undirected	587	19 603	66.79	2.16	–		0.69
	software packages	directed	1 439	1 723	1.20	2.42	1.6/1.4	0.070	0.082
	software classes	directed	1 377	2 213	1.61	1.51	–	0.033	0.012
	electronic circuits	undirected	24 097	53 248	4.34	11.05	3.0	0.010	0.030
	peer-to-peer network	undirected	880	1 296	1.47	4.28	2.1	0.012	0.011
Biological	metabolic network	undirected	765	3 686	9.64	2.56	2.2	0.090	0.67
	protein interactions	undirected	2 115	2 240	2.12	6.80	2.4	0.072	0.071
	marine food web	directed	135	598	4.43	2.05	–	0.16	0.23
	freshwater food web	directed	92	997	10.84	1.90	–	0.40	0.48
	neural network	directed	307	2 359	7.68	3.97	–	0.18	0.28

Taula 5.- Propietats de diferents xarxes reals[11]

V.2 Comparació entre xarxes reals i el model basat en l'excentricitat

Per realitzar la següent taula s'han triat les xarxes reals a les que més ens podríem aproximar en numero de nodes i enllaços a l'hora de generar el graf, degut a l'alt nivell computacional del model que estem fent servir.

TX	m	Nº rep.	Nodes	Edges	Grau			Distància			Diàmetre			Dist. Grau	Clust.	Coeff. Correlació
					Mig	Max	Min	Mitjana	Max	Min	Mitjà	Max	Min			
EX	10	40	100	945	18.9	41.05	10	1.857	1.864	1.850	3	3	3		0.2525	0.1026
XR	Freshwater food web		92	997	10.84			1.90						-	0.20	-0.326
EX	4	40	140	550	7.85	24.64	4	2.607	2.634	2.588	4	5	4		0.0788	0.1251
XR	Marine food web		135	598	4.43			2.05						-	0.16	-0.263
EX	8	40	300	2364	15.76	47.35	8	2.389	2.393	2.383	4	4	4		0.0744	0.2427
XR	Neural Network		307	2359	7.68			3.97						-	0.18	-0.226
EX	5	40	740	3685	9.95	39.02	5	3.118	3.128	3.110	5	5	5		0.0198	0.2416
XR	Metabolic Network		765	3686	9.64			2.56						2.2	0.090	-0.240
EX	5	40	1020	5085	9.97	41.94	5	3.249	3.244	3.254	5	5	5		0.0146	0.2490
XR	Roget's Theasaurus		1022	5103	4.99			4.87						-	0.13	0.157
EX	2	40	1100	2197	3.99	21.92	2	4.928	4.976	4.880	9	9	8		0.0044	0.1409
XR	Software classes		1377	2213	1.61			1.51						-	0.033	-0.119
EX	7	4	7950	55622	13.99	77.75	7	3.639	3.640	3.638	5	5	5		0.0027	0.2986
XR	Company directors		7673	55392	14.44			4.60						-	0.59	
EX	3	3	10690	32064	5.99	41.66	3	5.122	5.127	5.118	8	8	8		0.0009	0.2266
XR	Internet		10697	31992	5.98			3.31						2.5	0.035	
EX	3	3	17000	50994	5.99	42.33	3	5.344	5.349	5.338	8	8	8		0.0004	0.2288
XR	Email adress book		16881	57029	3.38			5.22							0.17	
EX	2	2	25000	49997	3.99	31.5	2	6.860	6.865	6.856	11	11	11	3.17	0.0002	0.1628
XR	Electronic circuit		24097	53248	4.34			11.05						3.0	0.010	

Taula 6.- Taula comparativa entre xarxes real i grafs generats amb el model basat en l'excentricitat

A la Taula 6 podem observar que per a xarxes "petites", de 100 a 1000, els valors de la distància i clustering són molt similars. Cal tenir en compte en interpretar la taula que en alguns casos es compara una xarxa real dirigida amb un graf no dirigit i això afecta el valor del grau mitjà. A les xarxes reals (*Freshwater food web, Marine food web, Neutral Network i Metabolic Network*) el coeficient de correlació és negatiu, hi ha molts nodes connectats a altres que no tenen el mateix graus.

A continuació es mostren unes gràfiques de les probabilitats de distribucions de grau de Internet, dades tretes del 2000[17] , en les quals es pot veure que segueix un comportament tipus xarxa Scale-Free.

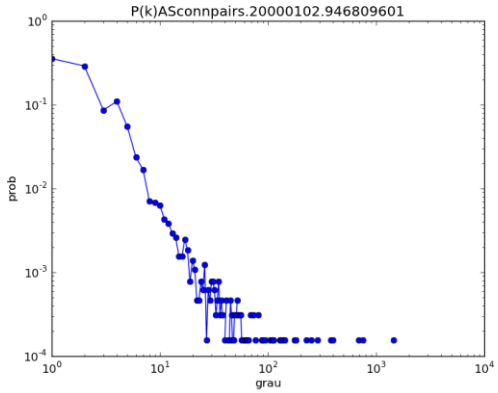


Figura 18.- Prob de dist. de graus, Internet 2000

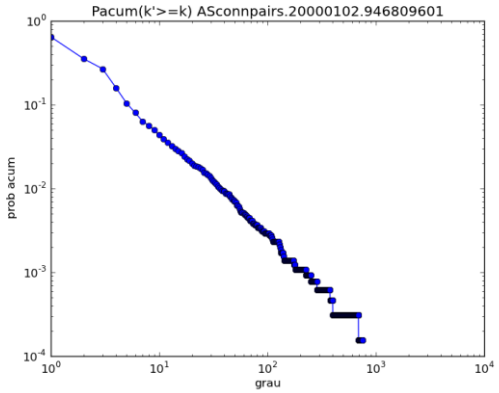


Figura 19.- Proba. acum. dist. grau Internet 2003

I també s'han simulat les distribucions de grau de les dades obtingudes el 2003[17], on veiem que ja no és tipus scale-free.

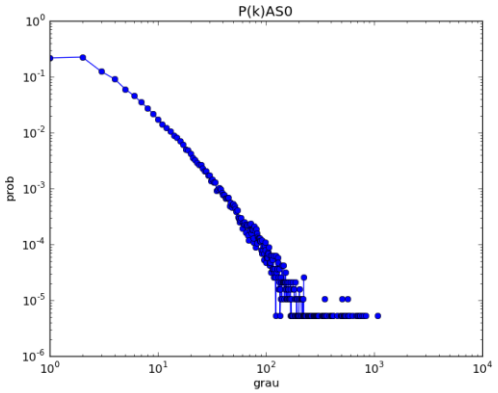


Figura 20.- Prob. dist. graus, Internet 2003

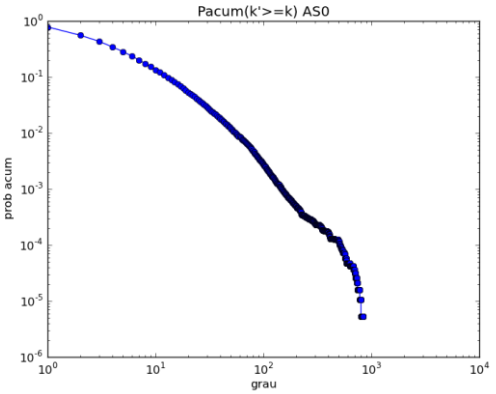


Figura 21.- Prob acum. dist. graus, Internet 2003

Com podem observar les gràfiques de la probabilitat de distribució de graus en aquest últim cas són més semblants a les obtingudes amb el model basat en l'excentricitat, que les obtingudes basat en el model Barabási-Albert. La probabilitat acumulada en les últimes dades que disposem de Internet fa mes forma de "panxa", com en el cas del model de l'excentricitat tal com hem vist en a les Figures 15 i 16. No observem una recta com en el model de Barabasi.

Amb *Internet* les diferències són mínimes mateix grau mig, distància una miqueta superior i clustering molt semblant, en quant a *Electronic circuit* el grau mitjà és molt semblant com la distribució de graus i el clustering però la distància mitjana entre els nodes es molt inferior en el cas del model basat en l'Excentricitat.

Hem analitzat unes altres dades on les distribucions de grau són encara més semblants a les obtingudes amb el model basat en l'excentricitat. Corresponen a la xarxa de proteïnes de l'interactoma humà[18]. Els nodes corresponen a les proteïnes humanes, i les branques a les relacions entre elles.

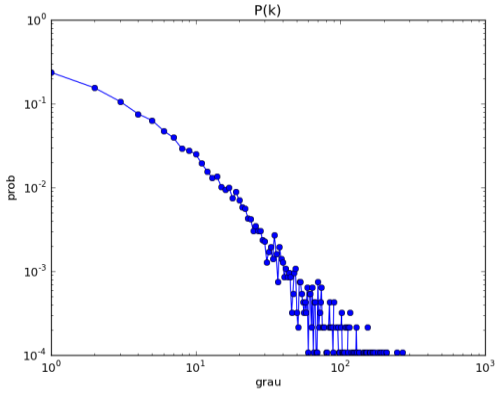


Figura 22.- Prob. dist. grau de la xarxa de proteïnes

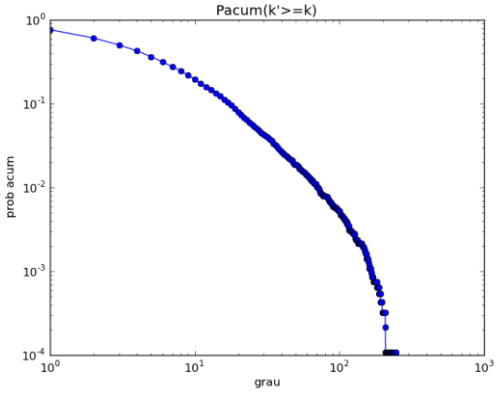


Figura 23.- Prob. acum. dist. grau xarxa de proteïnes

En aquest cas la distribució de graus segueix el model basat en l'excentricitat.

Les gràfiques de les Figura 18 a 21 s'assemblen força a la distribució Weibull[19] (que correspon a la distribució de vida d'objectes), que ve donada per la funció de probabilitat:

$$P(x) = \alpha\beta^{-\alpha}x^{\alpha-1}e^{-(x/\beta)^{\alpha}} \; for \; x \in [0, \infty)$$

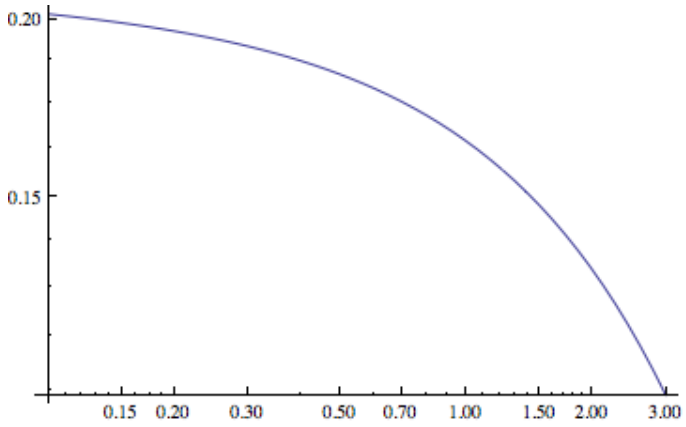


Figura 24.- Distribució Weibull. Exemple fet amb mathematica per $\alpha = 0.99$ $\beta = 5$

La distribució de Weibull ens dona la distribució de vida d'objectes, utilitzada per l'anàlisi dels sistemes que es veuen involucrats amb enllaços mes dèbils.

VI ALGORITMES D'OPTIMITZACIÓ COMBINATORIA

Cada dia, enginyers i científics s'enfronten amb problemes d'optimització de complexitat creixent. Aquest problemes complexos no és poden resoldre ràpidament aplicant algoritmes de força bruta, com és en el cas dels problemes lineals. Aquests problemes de complexitat exponencial no tenen una solució òptima en un interval de temps raonable. Degut això, s'han buscat algoritmes que obtinguin una solució el més òptima possible en un marge de temps raonable.
Aquest mètodes amb els quals trobem una solució aproximada acceptable, no necessariament la més òptima, en un temps reduït són els anomenats mètodes *metaheuristics.* Dintre d'aquests mètodes d'optimització, tenim el "simulated annealing", els algoritmes genètics, la "tabu search", etc. El primer de tots és el que implementem en el nostre projecte.

VI.1 SIMULATED ANNEALING

El mètode del "simulated annealing" va ser introduït l'any 1983 per S. Kirkpatrick, aquest mètode esta basat en l'algoritme dissenyat per Nicholas Metropolis[20] i els seus col·laboradors l'any 1953, el qual simulava el refredament del material. Un exemple fàcil per explicar el seu funcionament és la formació de cristalls. En una solució d'aigua calenta i sucre les molècules es mouen aleatòriament. Si la temperatura decreix ràpidament, les molècules de sucre es solidifiquen en estructures complicades i caòtiques. Però, si la temperatura baixa lentament, aquestes molècules formen un enorme cristall ordenat, el qual pot ser bilions de vegades més gran que les molècules. També, a part de estar cada molècula immòbil, aquestes és trobaran en el seu estat d'energia més baix.
Les molècules estan naturalment distribuïdes en un rang de energia. A mida que la temperatura es refreda, l'energia mitjana també ho fa. Però, ha mida que baixa la temperatura, no totes les molècules es reordenen per crear el cristall, encara hi ha algunes que es mantenen en el seu estat màxim de temperatura. Entre aquestes molècules, i les que estan en un estat inferior tot i pertànyer a un de superior, es produeix un intercanvi i una reordenació que fa que cadascuna d'elles vagi al nivell que li pertoca, minimitzant l'energia en conjunt i completant el cristall. Aquest intercanvi d'energia entre molècules afavoreix a la reordenació i la creació del cristall, però necessita d'un temps per realitzar-se. Aquest temps és important perquè si hi ha un descens molt brusc de la temperatura, poc temps, la formació del cristall no serà tan estructurat.

VI.1.1 Funcionament de l'algoritme

Una de les principals característiques del "simulated annealing"[21] (SA) és la seva capacitat per evitar quedar atrapat en un mínim local. Això es degut a que l'algoritme no només accepta les millors solucions, sinó que també es queda amb les pitjors amb una certa probabilitat donada.

El principal desavantatge que té és la definició empírica, per part de l'usuari, de certs paràmetres de control (temperatura inicial, velocitat de refredament, etc). D'aquesta manera és fàcil que SA funcioni, el més complicat es que funcioni bé.

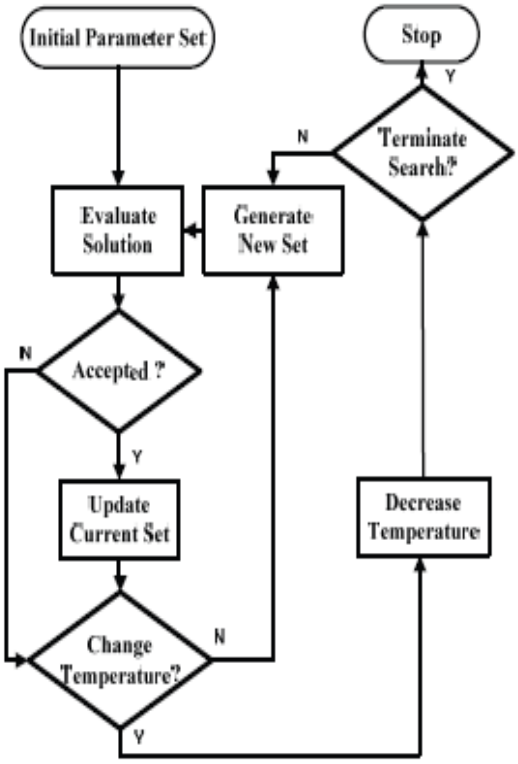


Figura 25.- Diagrama de flux per SA[22]

Un altre característica del SA és la capacitat que té d'evitar optimitzacions locals basades en l'acceptació de la norma d'una solució proposta. Si la solució actual (f_{new}) millora respecte a l'antiga (f_{old}), llavors el nou resultat s'accepta. D'altre banda, si el nou resultat obtingut és pitjor que l'antic, aquest pot ser acceptat si el valor donat per la distribució de Boltzmann es millor que un numero aleatori en l'interval [0,1], on *T* és la 'temperatura' (paràmetre de control).

A continuació s'exposen les diferents decisions que s'han de prendre per poder ajustar l'algoritme de manera que s'arribi al millor resultat possible.

$$e^{-\frac{f_{new}-f_{old}}{T}}$$

VI.1.2 Decisions genèriques.

Es refereixen principalment als paràmetres de "refredament". Des de controlar la variació de temperatura, el número d'iteracions abans del decreixement de la temperatura, i les condicions que ens permetran considerar quan el sistema s'ha "refredat", condicions de parada:

Temperatura inicial: Donat que l'objectiu és que la qualitat de la solució final sigui independent de la solució inicial de la que es parteix, la temperatura inicial hauria de ser independent de la solució inicial i suficientment alta com per acceptar lliurament les solucions de l'entorn.

Si es tracta d'un problema ben estructurat i tenim dades suficients com per estimar el valor, és possible partir d'un valor t_0 que accepti un canvi amb una determinada probabilitat. En cas de no ser conegut caldrà fixar un t_0 mes elevada, de forma que el sistema s'escalfarà per evitar condicionaments abans de començar el procés d'optimització.

Velocitat de refredament: La velocitat a la que produeix el refredament d'un altre factor clau a tenir en compte. La teoria suggereix que el sistema estigui a prop del seu estat estacionari corresponent a la temperatura actual, abans de reduir aquesta, i també que la temperatura vagi descendent gradualment al valor 0. En diversos estudis s'ha comprovat que els millors resultats obtinguts han sigut amb un descens geomètric de la velocitat:

$$t \to \alpha \cdot t, amb\ \alpha < 1$$

S'ha demostrat empíricament que els millors resultats s'obtenen per valors α de entre 0,8 i 0,99.

Número d'iteracions: El número de repeticions s'escull en funció de la magnitud del problema. El valor òptim per a tenir suficients mostres a cada interval de temperatura es de NxN, on N es el nombre d'enllaços. D'altre banda en alguns casos es pot escollir un valor mes baix al principi i anar augmentat aquest segons baixa la temperatura, d'aquesta manera aconseguim dedicar més temps al tram final del procés quan hi ha menys nombre de canvis que milloren la solució i cal tenir en compte més probabilitats.

Temperatura final: Aquest paràmetre és el que determina quan l'algoritme s'ha d'aturar. El valor final de la temperatura teòricament hauria de ser 0, però a la pràctica s'ha observat que s'arriba a una solució òptima final bastant abans. Per lo tant, la temperatura es fixada molt baixa. Diversos estudis[16] han arribat a la conclusió que per:

$$t \leq \varepsilon / ln \left[{|S| - 1}/{\theta} \right]$$

On θ és la probabilitat que una solució estigui a menys distància que ε de l'òptim. D'altres suggereixen aturar l'algoritme quan s'hagi produït un numero determinat d'iteracions sense cap acceptació.

VI.1.3 Decisions especifiques per a cada problema

Les decisions específiques és refereixen sobre tot a la definició de l'espai de solucions, l'estructura dels entorns y la funció de cost, així com la elecció de la solució final.
Està generalment acceptat que la solució inicial s_0 ha de ser generada de manera aleatòria. Per la resta no hi ha regles definitives però s'han de tenir en compte diversos factors.
Cal exigir que la solució pugui ser obtinguda des de qualsevol altre, a través d'una cadena de moviments vàlids.
Per garantir que el temps de càlcul s'utilitza eficientment, es important que les rutines utilitzades més freqüentment siguin el més ràpides possibles, pe: càlcul del cost d'una solució, generació de la nova solució aleatòria... El que tracta és relaxar algunes de les restriccions en la definició de les solucions, y penalitzar per el contrari mitjançant la funció de cost les violacions de les restriccions considerades.
Un altre aspecte que millora el temps de computació resideix en el càlcul de la funció de Boltzmann. Es proposa crear una taula on mirar directament els valors de l'exponencial sense necessitat d'haver-los de calcular a cada iteració. El valor d'aquesta taula es fixa en 1000 posicions, per cada t, $\delta \in [\frac{t}{200}, 5t]$. Cada posició representa el $\delta = i \cdot (\frac{t}{200})$, i per lo tant, a la posició i s'emmagatzemarà el valor $e^{-\delta/t} = e^{-i/200}$. L'índex per mirar a la taula s'obté calculant $\delta(200/t)$, acotant els valors entre 1 i 1000.

VII CREACIÓ DE L'ALGORITME

L'objectiu d'aquesta segona part del projecte es optimitzar un graf donat, fent modificacions en les connexions dels nodes existents, per obtenir un nou graf que optimitzi una certa funció de cost.

En el següent estudi analitzarem els principals paràmetres dels grafs obtinguts i comentarem la importància de considerar l'excentricitat a l'hora de realitzar aquest procés d'optimització.

VII.1 Paràmetres i funcions principals

Per tal de poder observar l'evolució del graf inicial al graf final, mitjançant el SA s'han de tenir en compte una sèrie de paràmetres importants en l'anàlisi de xarxes. Els paràmetres que es tindran en compte són:

- *Cost inicial*: En el següent apartat s'explica mes detalladament les funcions de cost utilitzades.
- *Distància mitjana i Diàmetre:* La distància mitjana es calcula com el promig de salts que hi ha per arribar d'un node a qualsevol altre del graf. El diàmetre ve determinat per el valor màxim de la distància en nombre de salts.
- *Clustering:* Per el càlcul del clustering primer es marquen els enllaços que estan connectats al node. Posteriorment, es compten tots els enllaços possibles que es poden donar entre la resta de parelles de nodes del graf i es da el quocient amb els que realment existeixen. Un cop fet això, es treu el promig per a tots els nodes.
  Aquest valor ens dóna informació important per veure la millora del grau de connectivitat que te el graf. Aquest valor només el calculem per el grau inicial i el grau final.
- *Excentricitat:* El càlcul de la excentricitat es imprescindible en el nostre programa, ja que intervé al llarg de tot el procés d'optimització. Aquest és calculat a cada iteració, i es un dels paràmetres que intervé en la funció de cost.
  Per a cada node, es calcula el nombre de salts que hi ha d'aquest a tota la resta. Per a cada parell de nodes, es mesura el camí mes curt del node inicial al node final. Després, de tots els camins calculats als diferents nodes es tria el valor màxim que és el que determinarà l'excentricitat del node, es a dir, busquem el camí mes curt al node mes llunyà.
- *Grau mig:* Es pren el valor de grau a nivell de xarxa. Aquest ve donat per el promig de tots els graus del graf. Inicialment ha de ser proper a 4, ja que tenim una malla on la majoria de nodes prenen el valor de grau 4. Ens determinarà si hi ha una millora a la robustesa del graf.

Per altre banda també s'ha de tenir en compte una sèrie de funcions que seran rellevants en l'algoritme:

Connexitat (connex()): La connexitat del graf és una restricció que tenim en compte en el procés d'optimització, ja que qualsevol parella de nodes ha de quedar connectada a la xarxa. En cas de que no fos així l'estudi no es podria tirar cap endavant.
Per no forçar aquesta condició i modificar les propietats estadístiques, el que es fa es només calcular la connexitat del graf al final de la modificació d'un enllaç, com a mode de comprovació. En cas de que el graf no fos connex es rebutjaria aquella modificació i es tornaria a fer una de nova.

Distància física (distància()): Es la distància real que tenen els enllaços.. Aquest es l'altre factor important a tenir en compte a l'optimització, es l'altre paràmetre que intervé en la nostra funció de cost i es veu modificat en cada canvi que sofreix el graf.
El cost d'un enllaç es proporcional a la seva capacitat i a la seva distància. El procés avalua la viabilitat d'introduir enllaços mes llargs i menys capacitat enfront a enllaços més curts i de mes grau d'utilització.
La distància és calcula directament a partir de les posicions que ocupen els nodes dintre de la malla prefixada i se'ls hi dona un valor prefixat d'1, per tots els enllaços igual.

Modificar el graf:

- *Canviar_enllaç():*Aquesta funció el que fa és seleccionar un enllaç a l'atzar i canviar un dels extrems, connectant-lo a un altre aleatòriament, amb la condició que el nou node al que connectem no estigui connectat a al primer node per un altre enllaç.

- *Eliminar enllaç():* Es selecciona un enllaç existent a la xarxa, el qual serà utilitzar per connectar-lo de nou a dos nodes diferents.

- *Afegir enllaç:* Es selecciona dos nodes aleatoris, amb la condició que no estiguin ja connectats per un enllaç, és connecten mitjançant un nou enllaç.

Fitxers de sortida: Per poder aprofitar el màxim les dades que podem extreure del graf, es creen diversos fitxer.
Apart del fitxer que emmagatzema les paràmetres inicials i finals del graf (funció de cost, clustering, distància mitjana...), es crea un fitxer amb el graf inicial i el graf final. Aquest te un format ".grf" que ens permet llegir-lo amb Python, utilitzant el paquet Network-X. D'aquí s'extreuen totes les gràfiques i representacions del diferents grafs.

VII.1.1 Funcions de cost

A l'hora d'escollir la funció de cost que modela les xarxes SF, s'ha fet servir diferents hipòtesis basades en les moltes investigacions que s'han realitzat sobre l'optimització de les SFNs.

En estudis anteriors s'ha comprovat que la escalabilitat dels nodes segueix una distribució potencial, amb nodes molt ben connectats amb les funcions de hub i d'altres més aïllats i sense tant de pes.
A partir d'aquesta hipòtesis s'han decidit utilitzar les següents funcions de cost:

- f=distància*excentricitat
- f=distància^2*excentricitat

Les funcions de cost escollides donen importància a la distància física dels enllaços, ja que el cost d'un enllaç be donat per la seva capacitat i la seva distància.
El cost final es calculat durant tota la execució de l'algoritme per cada iteració, perquè ens determina si el canvi que s'ha fet en aquella iteració fa que millori o empitjori la nostra xarxa.

VII.1.2 Eleccions dels paràmetres del SA

Com ja s'ha exposat en l'apartat V1.1 hi ha unes decisions a prendre a l'hora de programar el nostre algoritme d'optimització.

- La temperatura inicial (t_0) es determina en 650.
- Paràmetre α el determinarem en 0.9.
- El numero d'iteracions(repeticions) que farà per cada temperatura serà de 1000.
- Temperatura final (t_f) es determina el mes petit possible a 0.01

Partint d'aquest paràmetres una dada important a tenir en compte es el nombre de vegades que executem el procés del SA, per cada variació de temperatura fem NK repeticions (NK=1000 en aquest cas).
Cada vegada que executem l'algoritme d'optimització es fan 105183 iteracions/modificacions, de les quals algunes les dona com a bones i unes altres no.

VII.1.2.1 Estructura del programa

Com ja s'ha comentat a l'apartat anterior s'utilitzaran dos funcions de cost diferents, per aquest motiu s'ha d'ajustar l'algoritme del SA per a cada cas.

A continuació es detalla el procediment que fa el programa:

Primerament és genera el graf inicial, del qual es calculen els paràmetres principals (llistades al apartat VII.1.2) i es guarden en un fitxer .txt
Posteriorment es procedeix a inicialitzar les dades del SA:

- Temperatura pren valor inicial t_0
- Executa mentre temperatura>t_f
    - Baixem paulatinament valor temperatura (0.9*temperatura)
    - Repetir NK vegades
        - Modificar lleugerament el graf (canviar_enllaç())
        - Comprovar que el graf sigui connex, si connex()

- Calcular cost_ actual (cost())
- Δcost = cost_inicial – cost_actual
- Si Δcost<0 el canvi empitjora
    - Triar numero aleatori entre 0 i 1
    - Si numero aleatori < $e^{\left(\frac{\Delta cost}{Temperatura}\right)}$
        - Accepta el canvi
    - Si NO
        - Rebutja el canvi
- Si Δcost>0 el canvi millora, s'accepta

▪ Si NO connex(), rebutja el canvi
▪ Si rebutja el canvi
- Desfer la modificació, torna_endarrere()

○ Tornem a entrar al bucle, Repetir NK vegades
○ Disminuir la temperatura

- Fi SA

Un cop sortim del SA és tornen a calcular els paràmetres principals, per el graf final, i es guarden al final del fitxer creat amb els paràmetres del graf inicial.

VIII SIMULACIONS I RESULTATS

Comentem a continuació les simulacions efectuades i els seus resultats.
En totes les simulacions s'ha utilitzat una malla inicial 30x10 ja que d'aquesta manera s'aconsegueix tenir un graf relativament petit (la qual cosa comporta simulacions més ràpides), al mateix temps que hi ha un nombre d'enllaços comparable al de certes xarxes reals.
La malla inicial 30x10 considerada es mostra a continuació:

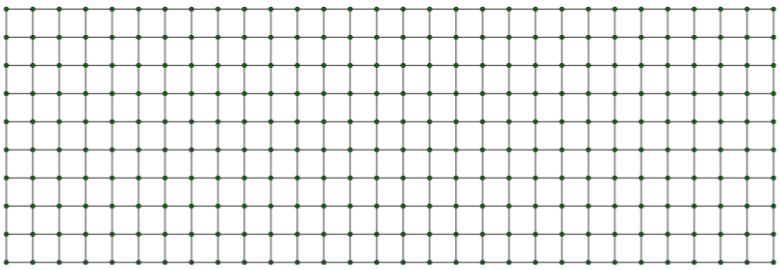


Figura 26.- Malla inicial de dimensions 30x10, 560 enllaços i 300 nodes

Les propietats d'aquesta malla inicial són les següents:

	COST INICIAL	NUM_ENLLAÇOS	CLUSTERING	DISTÀNCIA MITJANA	DIAMETRE	GRAU MAXIM
RESULTATS MALLA INCIAL	32220	560	0	13,3333333	38	4

Taula 7.- Resultats malla inicial 30x10

VIII.1 Funció de cost: distància * excentricitat

S'han realitzat 50 simulacions, fent servir l'algoritme descrit a la secció anterior amb les següents probabilitats de modificació del graf: reconnexió de l'enllaç 0.8, eliminació de l'enllaç 0.1, creació d'un nou enllaç 0.1. La primera funció de cost considerada ha estat "distància*excentricitat".
A la taula següent es mostren els resultats obtinguts:

	COST FINAL	NUM_ENLLAÇOS	CLUSTERING	DISTÀNCIA MITJANA	DIAMETRE	GRAU MAXIM
RES. MITJANES:	7595,900682	300,4	0	6,91290708	11,6	11,32
MAX	8714,355383	309	0	7,448785	13	16
MIN	6984,641835	299	0	6,456276	10	8

Taula 8.- Resultats malla 30x10, amb funció de cost= distància * excentricitat

Observem que el graf es va modificant progressivament i es converteix en un arbre o en un arbre amb alguns cicles. Això es degut a què hi ha una probabilitat (encara que sigui petita) d'eliminar enllaços i la funció de cost considerada es redueix sempre que s'elimina un enllaç.
El clustering segueix éssent 0 degut a la topologia del graf resultant. La distància mitjana i el diàmetre es redueixen a la meitat.

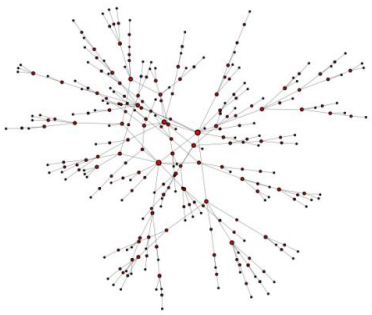


Figura 27.- Exemple de graf obtingut per SA considerant la funció "distància*excentricitat" a partir d'una malla inicial 30x10.

Si ens fixem en la distribució i distribució acumulada de graus d'una de les simulacions fetes (Figura 28), podem veure que suggereixen una distribució exponencial.

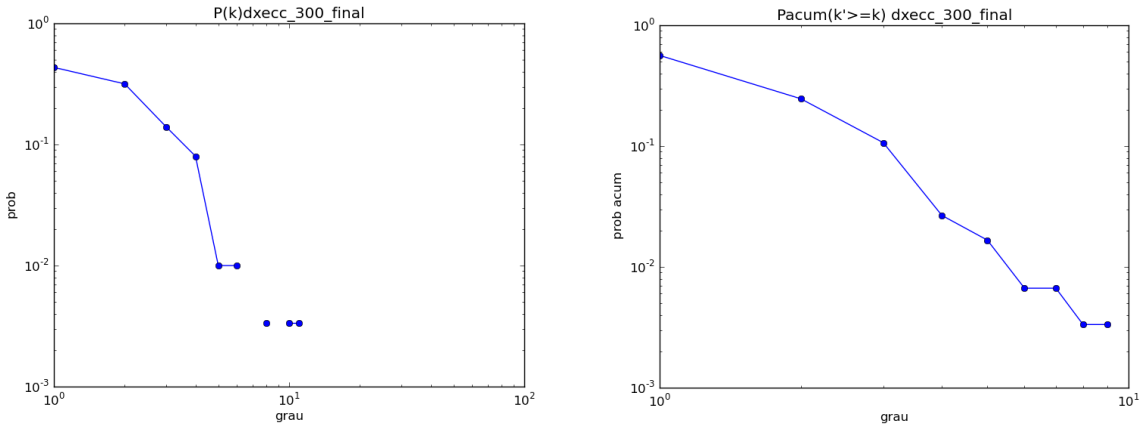


Figura 28.- Gràfiques de les distribució i distribució acumulada de graus per a un graf obtingut considerant la funció de cost "distancia * excentricitat".

Per determinar els paràmetres d'aquesta distribució exponencial, considerem el promig de les 50 simulacions. Es pot observar a la Figura 29 com la distribució de graus d'aquest promig té la clàssica forma de "panxa".

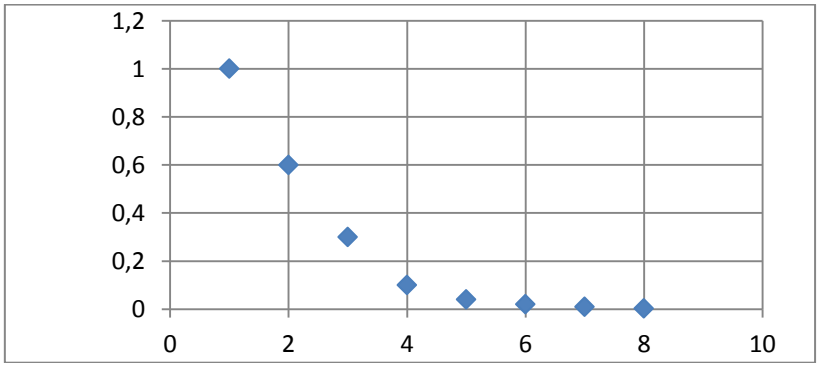


Figura 29.- Distribució acumulada dede grau P(k) per al promig de 50 simulacions.

Amb l'anàlisi lligat a la Figura 30 s'ha determinat el paràmetre α que pren un valor de 0.375, valor semblant a l'obtingut en l'apartat IV.3.1.

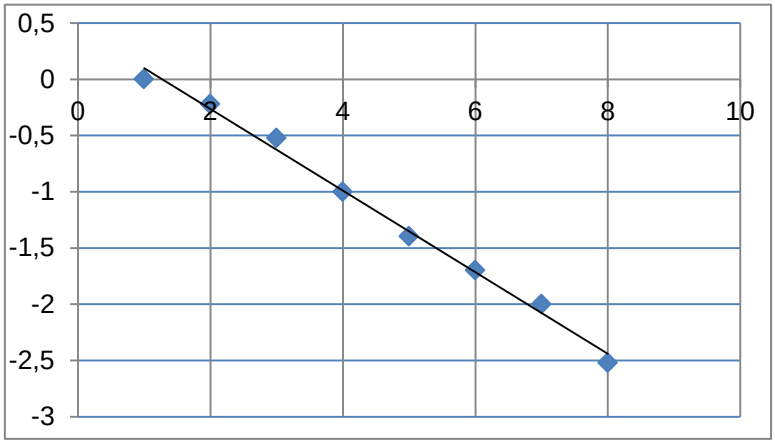


Figura 30.- Representació de log(P(k)) que permet calcular el valor α=0.375 per a la distribució exponencial.

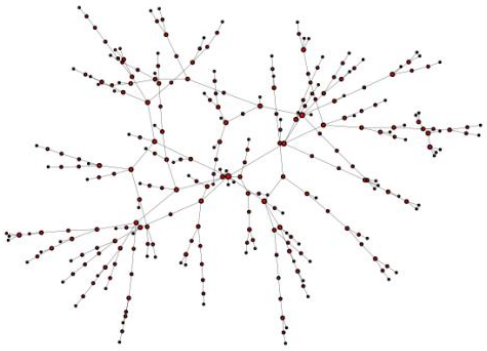
VIII.2 Funció de cost: distància^2 * excentricitat

També s'han realitzat 50 simulacions més amb la funció de cost "distància^2*excentricitat".
Els resultats obtinguts són a la Taula 9

	COST FINAL	NUM_ENLLAÇOS	CLUSTERING	DISTÀNCIA MITJANA	DIAMETRE	GRAU MAXIM
RESULTATS MITJOS:	14488,61	299,34	0	9,86168518	18,14	6,58
MAX	16858,00	303	0	10,986867	22	9
MIN	12541,99	299	0	9,020289	16	5

Taula 9.- Resultats obtinguts amb la funció de cost= distància^2 * excentricitat

Els paràmetres trobats són molt semblants als obtinguts amb la funció de cost anterior i la distribució de graus segueix indicant una funció exponencial. La topologia del graf resultant també és pròxima a un arbre.



.

Figura 31.- Graf resultant amb la funció de cost= distància2 * excentricitat

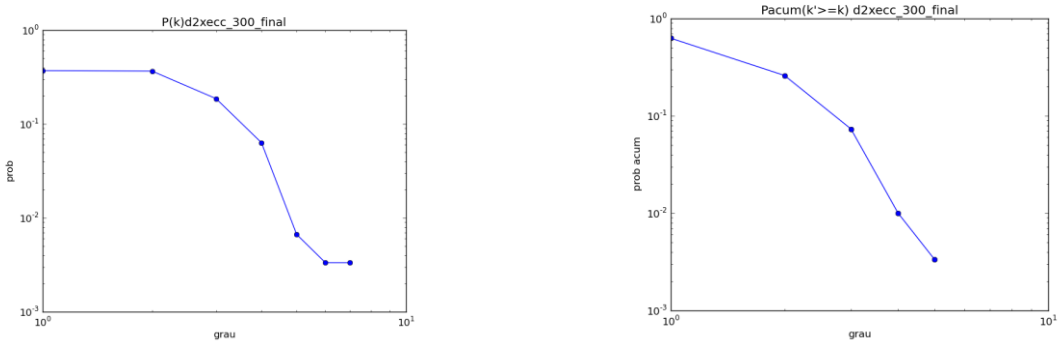


Figura 32.- Gràfiques de les distribució i distribució acumulada de graus per al graf obtingut considerant la funció de cost "distancia^2 * excentricitat".

VIII.1 Simulacions sense eliminació d'enllaços

A partir dels resultats obtinguts amb les simulacions anteriors, es va decidir canviar la funció de l'algoritme SA que realitza la modificació del graf. Es van eliminar les funcions afegir i eliminar enllaç, ja que d'aquesta manera en modificar enllaços ja existents, sense possibilitat d'eliminar-los, s'obtindrà sempre un graf diferent a un arbre. S'han realitzat 20 simulacions amb les funcions de cost "distància* excentricitat" i "distància^2 * excentricitat" per a un graf amb 1000 nodes. En aquest cas els resultats trobats suggereixen en ambdós casos que el graf resultant és parcialment "scale-free", encara que aquest fet podria ser degut a que no s'han realitzat prou iteracions.

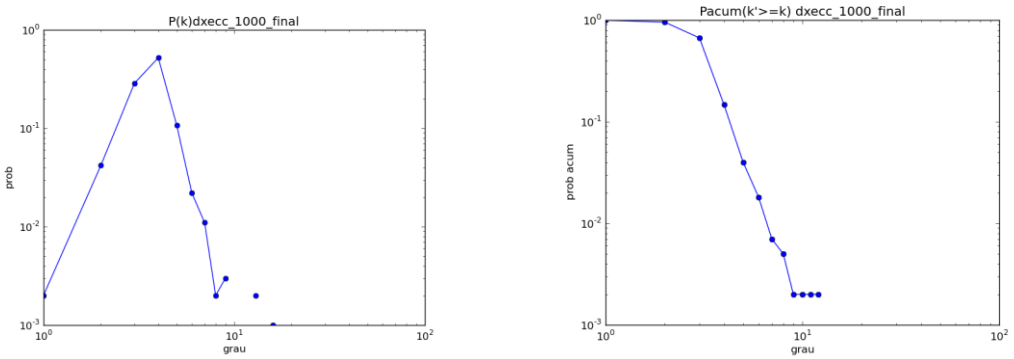


Figura 32.- Gràfiques de les distribució i distribució acumulada de graus per a la xarxa obtinguda considerant la funció de cost "distància*excentricitat".

A la gràfica de l'esquerra per la distribució de graus p(k), no es pot identificar un graf amb característiques totalment xarxa "scale-free" però tampoc una distribució exponencial. A la dreta es mostra la distribució acumulada de graus on una part podria interpretar-se com a llei potencial. El mateix s'observa a les gràfiques de la Figura 36.

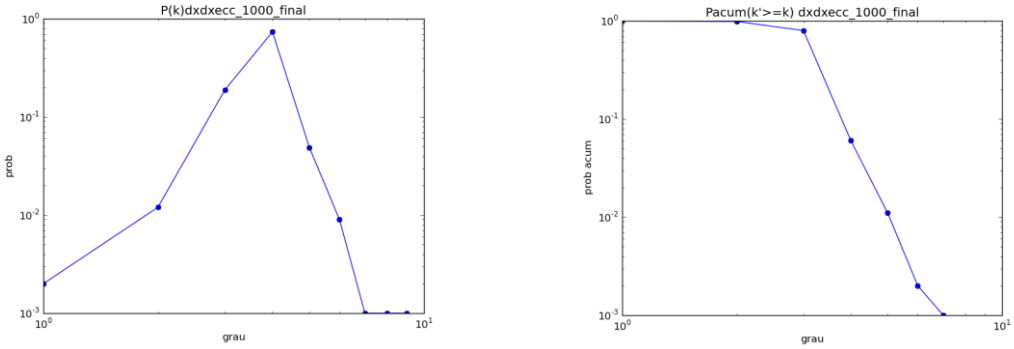


Figura 33.- Gràfiques de les distribució i distribució acumulada de graus per a la xarxa obtinguda considerant la funció de cost "distancia^2 * excentricitat".

Per confirmar si la distribució de graus correspon finalment a una distribució exponencial , convindria executar l'algoritme realitzant moltes més iteracions i també estudiar malles inicials de dimensions diferents.

IX CONCLUSIONS

En aquest TFC hem presentat i estudiat nous models per generar xarxes complexes. Aquests models produeixen xarxes amb una distribució de grau de tipus exponencial. En l'anàlisi de xarxes reals ja havia estat observat sovint aquest comportament del grau, però models clàssics com el de Bararbási-Albert no el poden explicar ja que generen xarxes amb una distribució de graus potencial (tipus "scale-free"). Fins i tot, la xarxa de routers (Intenet) que en estudis fets els anys 1999 i 2000 es considerava que tenia propietats "scale-free", ha anat evolucionant en el temps de tal manera que l'any 2003 presentava ja un comportament que no s'explicava amb el model de Barabási-Albert. El primer model que hem introduït, basat en un creixement del graf en funció de l'excentricitat, aproxima millor Internet i justifica bé les seves propietats així com les d'altres xarxes reals rellevants (xarxa elèctrica, xarxes de col·laboracions, etc.) que models considerats habitualment no poden explicar convenientment.

El model estudiat en la segona part del projecte optimitza, en base a l'excentricitat i la longitud dels enllaços, un graf inicial amb nodes de grau semblant i enllaços de la mateixa longitud física (una malla). El procés d'optimització, que fa servir "simulated annealing", obté –quan es permet l'eliminació d'enllaços- un graf tipus arbre amb una distribució de graus que també és exponencial. Si el procés sols permet la reconnexió dels enllaços, els resultats obtinguts no són concluents degut a que caldria un estudi més complet.

El fet que el nostres models aproximin bé xarxes reals amb un comportament exponencial en la distribució de graus, afegeix una nova metodología per a l'estudi d'altres aspectes d'aquestes xarxes, com per exemple les seves propietats dinàmiques: sincronització, difusió d'informació, temps de primer pas, etc.

X REFERENCIES

[1]G.B. West, J.H. Brown, B.J. Enquist. A general model for the origin of allometric scalingl laws in biology. (1997) Science 276 pp. 122-126.

[2]D.Cohen, All the world's a net. (2002). NewScientist vol 174,issue2338, page 24

[3]A.-L.Barabasi, E. Bonabeau. Scale-free networks. (2003), Scientific American nº5 50-59

[4]R.Albert A L Barabási. Statistical mechanics of complex networks.(2002) Reviews of modern physics, vol 74

[5]Albert-Laszlo Barabasi, Reka Albert, Hawoong Jeong. Mean-field theory for scale-free random networks. Physica A 272 (1999) 173-187

[6]MEJ. Newman, The structure and function of complex Networks. (2003) Society for Industrial and Applied Mathematics review vol 45 nº2 pp 167-256.

[7]Cohen, Reoven, K. Erez, D. ben-Avraham and S. Havlin. Resilience of the Internet to Random Breakdowns.(2000) Phys. Rev. Lett. 85: 4626–8

[8]Cohen, Reoven; K. Erez, D. ben-Avraham and S. Havlin. Breakdown of the Internet under Intentional Attack. (2001) Phys. Rev. Lett. 86: 3682–5

[9]Callaway, Duncan S.; M. E. J. Newman, S. H. Strogatz and D. J. Watts. Network Robustness and Fragility: Percolation on Random Graphs. (2000) Phys. Rev. Lett. 85: 5468–71.

[10]D.B.West, Introduction to Graph Theory. Prentice-Hall. (2001).Saddle river, NJ,

[11]MEJ. Newman, The structure and function of complex Networks. (2003) Society for Industrial and Applied Mathematics review vol 45 nº2 pp 167-256.

[12] J.Travers And S Milgram, An expermental study of the small World problem. (1969), sociometry, 32 pp 425-443.

[13]http://oracleofbacon.org/

[14]R.Albert, H.Jeong, A.-LBarabási, Diameter of the world wide web.(1999) Nature 130-131.

[15]Q.Chen H.Chang, R Govindan, SJamin, S.J Shenker, And W. Willinger, The origin of power laws in Internet topologies revisited.(2002) in Proceedings of the 21st Anual Joint Conference of the IEEE Computer and Communications Societies, IEEE.

[17]http://www.caida.org/tools/measurement/skitter/router_topology/index.xml

[18]http://www.hprd.org/download

[19]http://mathworld.wolfram.com/WeibullDistribution.html

[20]Shawn Carlson, The Amateur Scientist. (1997), Scientific American, pp 121-123

[21]Kathryn A.Dowsland, Heuristic design and fundamentals of the Simulated Annealing. ASAP Research Group School of Computer Science and Information Technology University of Nottingham.

[22]Rui Chivanete, Simulated Annealing Theory with Applications. Sciyo.com, page 2

XI ANNEX

Codi en programació C++ de l'algoritme Simulated Annealing tenint en compte l'excentricitat i la distancia.

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <time.h>
#include <string.h>


#define  DIMENSIO_F   50
#define  DIMENSIO_C   20
#define  MAX_NODES    1000
#define  NK          1000
#define  TEMP_INICIAL 650
#define  TEMP_MINIMA  0.01
#define  MAX_ENLLACOS 2000
#define  MAX         2000
#define  TRUE 1
#define  FALSE 0


typedef struct{
          int A;
          int B;
          float distància;
          double bt;
          float eccA;
   float eccB;
   float AeccA;
   float AeccB;
}Tenllac;

typedef struct{
   int num_enll;
          int posicio[2];
   int enllacos[MAX_ENLLACOS];
   float clust;
   float ecc;
}Tnode;

typedef struct{
          int malla[DIMENSIO_F][DIMENSIO_C];
   Tnode nodes[MAX_NODES];
          Tenllac arestes[MAX_ENLLACOS];
```

```
        int distància[MAX_NODES][MAX_NODES];
        int diametre;
        float clustering;
        float distància_mitja;
        int dist_ecc[MAX_NODES][MAX_NODES];
        float graumig;
}Tgraf;

typedef struct{
    int node;
    float dist;
}t_dist;


typedef struct {
        int     slot[MAX];
   int   dimensiones;
   int   head;
   int   tail;
} Tfifo;


void init_fifo(Tfifo *f, int dimensiones)
{
        f->tail=0;
        f->head=0;
        f->dimensiones=dimensiones;
}

int fifo_llena(Tfifo *f)
{
        if (f->head>f->tail)
        {
                if(MAX-f->head+f->tail==f->dimensiones)
                return(TRUE);
        else
                return(FALSE);}
        else
        {
                if(f->tail-f->head==f->dimensiones)
                        return(TRUE);
                else
                        return(FALSE);
        }
}

int fifo_vacia(Tfifo *f)
{
        if(f->head==f->tail)
                return(TRUE);
        return(FALSE);
}

int fifo_elementos(Tfifo *f)
{
        if(f->head<=f->tail)
                return(f->tail-f->head);
        else
                return(MAX-f->head+f->tail);
}
```

```
int put_fifo(Tfifo *f, int *nuevo)
{
  if(fifo_llena(f)==TRUE)
  {
            printf("stack overflow");
            exit(1);
  }
        f->slot[f->tail]=*nuevo;
        f->tail=(f->tail+1)%MAX;
        return(TRUE);
}


int get_fifo(Tfifo *f, int *nuevo)
{
        if(fifo_vacia(f)==TRUE)
                return(FALSE);
        *nuevo=f->slot[f->head];
        f->head=(f->head+1)%MAX;
        return(TRUE);
}


Tgraf mon;
float f,temperatura;
double cost_inicial,cost_actual,bttotal, ecc_total;
int NUM_ENLLACOS;
int recompte[MAX_NODES],graumax[2];
t_dist nd[MAX_NODES];


void networkx(char *arxiv,char *cua)
{
        int A,B,n;
        FILE *fitxer;
        char arxiu[25];
        strcpy(arxiu,arxiv);
        strcat(arxiu,cua);
        fitxer = fopen(arxiu,"w+");
        for (n=0;n<MAX_NODES;n++)
        {
                fprintf(fitxer,"\n%i ",n);
                if(mon.nodes[n].num_enll>0)
                {
                        for(int e=0;e<mon.nodes[n].num_enll;e++)
                        {
                                A=mon.arestes[mon.nodes[n].enllacos[e]].A;
                                B=mon.arestes[mon.nodes[n].enllacos[e]].B;
                                if(B>n)
                                        fprintf(fitxer,"%d ",B);
                                else
                                {
                                        if (A>n)
                                                fprintf(fitxer,"%d ",A);
                                }
                        }
                }
        }
fclose(fitxer);
```

```
}

void comptar()
{
	graumax[0]=0;
	graumax[1]=0;
	bttotal=0;
	ecc_total=0;
	float eccA,eccB;

	for(int e=0;e<=MAX_NODES;e++)
	{
		recompte[e]=0;
	}
	for(int j=0;j<MAX_NODES;j++)
	{
      if(graumax[0]<mon.nodes[j].num_enll)
		{
			graumax[0]=mon.nodes[j].num_enll;graumax[1]=j;
		}
      recompte[mon.nodes[j].num_enll]++;
	}

	for(int e=0;e<NUM_ENLLACOS;e++)
	{
		eccA=mon.nodes[mon.arestes[e].A].ecc;
		eccB=mon.nodes[mon.arestes[e].B].ecc;
		ecc_total=+(eccA+eccB);
		bttotal+=mon.arestes[e].bt;
	}
}

void escriu(char *arxiv,int cas)
{
	FILE *fitxer;
	char arxiu[25];
	strcpy(arxiu,arxiv);
	strcat(arxiu,".txt");
	switch(cas)
	{
		case 0:
			comptar();
			fitxer = fopen(arxiu,"wb");
			fprintf(fitxer,"\nDIMENSIO_F:%i
DIMENSIO_C:%i\nMAX_NODES:%i\nNUM_ENLLACOS:%i",DIMENSIO_F,DIMENSIO_C,MAX
_NODES,NUM_ENLLACOS);
			fprintf(fitxer,"\ncost_inicial:%f\nclustering:%f\ndistància
mitja:%f\ndiametre:%i\ngrau mig:%f\ngrau
maxim:\tnode:%i\tgrau:%i\n",cost_inicial,mon.clustering,mon.distància_mitja,mon.diametre,mon.
graumig,graumax[1],graumax[0]);
			fprintf(fitxer,"\nrecompte");

			for(int x=0;x<=(MAX_NODES)/2;x++)
			{
				fprintf(fitxer,"\n%d\t%d",x,recompte[x]);
			}
			fprintf(fitxer,
"\n**********************************ECCENTRICITATS**********************************************
*****\n");
```

```
	for (int i=0; i<MAX_NODES;i++)
	{
	fprintf(fitxer,"[nodo:%i  ecc:%f] \n" , i, mon.nodes[i].ecc);
	}

				fprintf(fitxer,
"\n*****************************************************************************************************
**********************\n");

				fclose(fitxer);
				break;

			case 1:
				comptar();
				fitxer = fopen(arxiu,"ab");
				fprintf(fitxer,"\nDIMENSIO_F:%i
DIMENSIO_C:%i\nMAX_NODES:%i\nNUM_ENLLACOS:%i",DIMENSIO_F,DIMENSIO_C,MAX
_NODES,NUM_ENLLACOS);
				fprintf(fitxer,"\ncost_inicial:%f\nclustering:%f\ndistància
mitja:%f\ndiametre:%i\ngrau mig:%f\ngrau
maxim:\tnode:%i\tgrau:%i\n",cost_inicial,mon.clustering,mon.distància_mitja,mon.diametre,mon.
graumig,graumax[1],graumax[0]);

				for(int x=0;x<=(MAX_NODES/2);x++)
				{
					fprintf(fitxer,"\n%d\t%d",x,recompte[x]);
				}


				fprintf(fitxer,
"\n*********************************ECCENTRICITATS*************************************************
*****\n");

				for (int i=0; i<MAX_NODES;i++)
	{
					fprintf(fitxer,"[nodo:%i  ecc:%f] \n" , i, mon.nodes[i].ecc);
	}

				fprintf(fitxer,
"\n*****************************************************************************************************
**********************\n");

				fclose(fitxer);
				break;
	}
}

void clustering()
{
	int aux,ereals,epossibles;
	int enllacos[NUM_ENLLACOS];
	Tfifo veins;

	for(int i=0;i<NUM_ENLLACOS;i++)
		enllacos[i]=0;

	init_fifo(&veins,MAX_NODES);

	for (int i=0;i<MAX_NODES;i++)
	{
```

```
		for(int p=0;p<NUM_ENLLACOS;p++)
			enllacos[p]=0;
		ereals=0;
		epossibles=0;
		if(mon.nodes[i].num_enll>1)
		{
			for(int j=0;j<mon.nodes[i].num_enll;j++)
			{
				if(mon.arestes[mon.nodes[i].enllacos[j]].A==i)

	put_fifo(&veins,&(mon.arestes[mon.nodes[i].enllacos[j]].B));

				else

	put_fifo(&veins,&(mon.arestes[mon.nodes[i].enllacos[j]].A));
			}
			while(fifo_vacia(&veins)== FALSE)
			{
				get_fifo(&veins,&aux);

				for(int k=0;k<mon.nodes[aux].num_enll;k++)
				{
					enllacos[mon.nodes[aux].enllacos[k]]++;
				}
			}

			for(int c=0;c<NUM_ENLLACOS;c++)
			{
				if(enllacos[c]==2)
					ereals++;
			}
			epossibles=(mon.nodes[i].num_enll)*(mon.nodes[i].num_enll-1)/2;
			mon.nodes[i].clust =(float) ereals/epossibles;
		}
		else
			mon.nodes[i].clust=1;
	}
	for(int k=0;k<MAX_NODES;k++)
	{
		mon.clustering += mon.nodes[k].clust;}
	mon.clustering = (mon.clustering/MAX_NODES);
	}

void distànciametre()
{
	int i,w,l,grau;
	Tfifo cua;
	for(int j=0;j<MAX_NODES;j++)
	{
		for(int k=0;k<MAX_NODES;k++)
		{
			mon.distància[j][k]=-1;
		}
	}
	for(int x=0;x<MAX_NODES;x++)
	{
		init_fifo(&cua,MAX_NODES);
		put_fifo(&cua,&x);
		mon.distància[x][x]=0;
		while (!fifo_vacia(&cua))
```

```
                {
                        get_fifo(&cua,&i);
                        l=mon.distància[x][i];
                        for (int e=0;e<mon.nodes[i].num_enll;e++)
                        {
                                if (mon.arestes[mon.nodes[i].enllacos[e]].A==i)
                                        w=mon.arestes[mon.nodes[i].enllacos[e]].B;

                                else
                                        w=mon.arestes[mon.nodes[i].enllacos[e]].A;

                                if (mon.distància[x][w]==-1)
                                {
                                        mon.distància[x][w]=l+1;
                                        put_fifo(&cua,&w);}
                                }
                        }
        }
        mon.distància_mitja=0;
        mon.diametre=0;
        for (int j=0;j<MAX_NODES;j++)
        {
                for (int k=0;k<MAX_NODES;k++)
                {
                        mon.distància_mitja +=mon.distància[j][k];

                        if(mon.distància[j][k]>mon.diametre)
                                mon.diametre=mon.distància[j][k];
                }
        }
        mon.distància_mitja= mon.distància_mitja/(MAX_NODES*(MAX_NODES-1));

        grau=0;;
        for (int j=0;j<MAX_NODES;j++)
        {
                grau+=mon.nodes[j].num_enll;
        }
        mon.graumig=grau/MAX_NODES;

}

void eccent_node()
{
   int vec[MAX_NODES];
   int i,w,l,grau,max;
        Tfifo cua;
        for(int j=0;j<MAX_NODES;j++)
        {
                for(int k=0;k<MAX_NODES;k++)
                {
                        mon.dist_ecc[j][k]=-1;
                }
        }
        for(int x=0;x<MAX_NODES;x++)
        {
                init_fifo(&cua,MAX_NODES);
                put_fifo(&cua,&x);
                mon.dist_ecc[x][x]=0;
                while (!fifo_vacia(&cua))
                {
```

```
                        get_fifo(&cua,&i);
                        l=mon.dist_ecc[x][i];

                        for (int e=0;e<mon.nodes[i].num_enll;e++)
                        {
                                if (mon.arestes[mon.nodes[i].enllacos[e]].A==i)
                                        w=mon.arestes[mon.nodes[i].enllacos[e]].B;
                                else
                                        w=mon.arestes[mon.nodes[i].enllacos[e]].A;
                                if (mon.dist_ecc[x][w]==-1)
                                {
                                        mon.dist_ecc[x][w]=l+1;
        put_fifo(&cua,&w);
                                }
                        }
                }
        }

        for (int j=0; j<MAX_NODES;j++)
   {
      for (int k=0; k<MAX_NODES;k++)
      {
         vec[k]=mon.dist_ecc[j][k];
      }

      max=0;
      for (i=0; i<MAX_NODES; i++)
      {

        if (vec[i]>max)
          {
            max=vec[i];
          }
                }
                mon.nodes[j].ecc=max;
        }

}

int connex()
{
        Tfifo cua2;
   int *nou,c;
   int trobat[MAX_NODES];

   for (int i=0;i<MAX_NODES;i++)
          trobat[i]=0;

   init_fifo(&cua2,MAX);
   nou=(int*)malloc(sizeof(int));

        if(nou==NULL)
        {
                printf("error malloc");
                exit(1);
        }
   do
        {
                *nou=rand()%(MAX_NODES-1);
        }
```

```
        while(mon.nodes[*nou].num_enll==0);
    put_fifo(&cua2,nou);

    while(fifo_vacia(&cua2)==FALSE)
        {
                get_fifo(&cua2,nou);
                trobat[*nou]=1;

                for(int aux=0;aux<mon.nodes[*nou].num_enll;aux++)
                {
                        if (mon.arestes[mon.nodes[*nou].enllacos[aux]].A == *nou)
                        {
                                if (trobat[mon.arestes[mon.nodes[*nou].enllacos[aux]].B]==0)
                                {

        put_fifo(&cua2,&(mon.arestes[mon.nodes[*nou].enllacos[aux]].B));
             trobat[mon.arestes[mon.nodes[*nou].enllacos[aux]].B]=1;
                                }
                        }

                        if (mon.arestes[mon.nodes[*nou].enllacos[aux]].B == *nou)
                        {
                                if (trobat[mon.arestes[mon.nodes[*nou].enllacos[aux]].A]==0)
                                {

        put_fifo(&cua2,&(mon.arestes[mon.nodes[*nou].enllacos[aux]].A));

        trobat[mon.arestes[mon.nodes[*nou].enllacos[aux]].B]=1;
                                }
                        }
                }
        }
        c=0;
        for (int v=0;v<MAX_NODES;v++)
        {
                if(trobat[v]!=1)
                {
                        c=1;
                }
        }
        free(nou);
        if (c==1)
                return 0;

        else
                return 1;
}


float distància(int cap1, int cap2)
{
        int n1i,n1j,n2i,n2j,disti,distj;

        n1i=mon.nodes[cap1].posicio[0];
        n1j=mon.nodes[cap1].posicio[1];
        n2i=mon.nodes[cap2].posicio[0];
        n2j=mon.nodes[cap2].posicio[1];
        disti=abs(n1i-n2i);
        distj=abs(n1j-n2j);
```

```
        return (sqrt((disti*disti)+(distj*distj)));
}

void inicialitza_mon_quadricula()
{
        int existeix,aux,i,j,n,n1,n2,e,f;
        i=j=n1=n2=e=aux=n=0;

        for (i=0;i<DIMENSIO_F;i++)
  {
                for(j=0;j<DIMENSIO_C;j++)
    {
      mon.malla[i][j]=n;
      mon.nodes[n].posicio[0]=i;
      mon.nodes[n].posicio[1]=j;
      mon.nodes[n].num_enll=0;
      n++;
    }
  }


  for(j=0;j<DIMENSIO_F*DIMENSIO_C;j+=DIMENSIO_C)
  {
    for(i=0;i<DIMENSIO_C-1;i++)
    {
      mon.nodes[j+i].enllacos[mon.nodes[j+i].num_enll]=e;
      mon.nodes[j+i+1].enllacos[mon.nodes[j+i+1].num_enll]=e;
      mon.nodes[j+i].num_enll++;
      mon.nodes[j+i+1].num_enll++;
      mon.arestes[e].A=j+i;
      mon.arestes[e].B=j+i+1;
      mon.arestes[e].distància=distància(j+i,j+i+1);
      e++;
    }
  }

  for(j=0;j<(DIMENSIO_F-1)*DIMENSIO_C;j+=DIMENSIO_C)
  {
    for(i=0;i<DIMENSIO_C;i++)
    {
      mon.nodes[j+i].enllacos[mon.nodes[j+i].num_enll]=e;
      mon.nodes[j+i+DIMENSIO_C].enllacos[mon.nodes[j+i+DIMENSIO_C].num_enll]=e;
      mon.nodes[j+i].num_enll++;
      mon.nodes[j+i+DIMENSIO_C].num_enll++;
      mon.arestes[e].A=j+i;
      mon.arestes[e].B=j+i+DIMENSIO_C;
      mon.arestes[e].distància=distància(j+i,j+i+DIMENSIO_C);
      e++;
      }
  }
  NUM_ENLLACOS=e;
}

void mostra_mon()
{
  for(int i=0;i<DIMENSIO_F;i++)
  {
    for(int j=0;j<DIMENSIO_C;j++)
    {
```

```
            if(mon.malla[i][j]!=-1)
                printf(" %i",mon.malla[i][j]);
            else
                printf(" .");
        }
     printf("\n");
    }
}

void mostra_posicions()
{
    printf ("\nNODE:\ti\tj\n\n");

    for (int i=0;i<MAX_NODES;i++)
        printf("%i:\t%i\t%i\n",i,mon.nodes[i].posicio[0],mon.nodes[i].posicio[1]);
}


Tenllac canviar_enllac(int *index_antic)
{
    Tenllac antic;
    int e,A,nouA,j,repetit;
    do
    {
        e=rand()%(NUM_ENLLACOS);
    }
    while(mon.nodes[mon.arestes[e].A].num_enll<2 || mon.nodes[mon.arestes[e].B].num_enll ==
(MAX_NODES-1));

    A=mon.arestes[e].A;

    *index_antic=e;
    antic.A=mon.arestes[e].A;
    antic.B=mon.arestes[e].B;
    antic.distància=mon.arestes[e].distància;

    for(int i=0,j=0;i<mon.nodes[A].num_enll;i++)
    {
        if(mon.nodes[A].enllacos[j]==e)
        j++;
        mon.nodes[A].enllacos[i]=mon.nodes[A].enllacos[j];
        j++;
    }

    mon.nodes[A].num_enll=mon.nodes[A].num_enll-1;

    do
    {
        nouA=rand()%(MAX_NODES);
        repetit=FALSE;
        if(nouA == antic.B)
            repetit = TRUE;

        for(int rep=0;rep<NUM_ENLLACOS;rep++)
        {
            if((mon.arestes[rep].A==nouA &&
mon.arestes[rep].B==antic.B)||(mon.arestes[rep].A==antic.B && mon.arestes[rep].B==nouA))
            {
                repetit=TRUE;
            }
```

```
        }
    }
    while(repetit==TRUE);
    mon.arestes[e].A=nouA;
    mon.arestes[e].distància=distància(nouA,mon.arestes[e].B);
    mon.nodes[nouA].enllacos[mon.nodes[nouA].num_enll]=e;
    mon.nodes[nouA].num_enll++;
    return antic;
}

void torna_endarrera(int index_antic,Tenllac antic)
{
    int A, nouA;
    A=mon.arestes[index_antic].A;
    for(int i=0,j=0;i<mon.nodes[A].num_enll;i++)
    {
        if(mon.nodes[A].enllacos[j]==index_antic)
            j++;
        mon.nodes[A].enllacos[i]=mon.nodes[A].enllacos[j];
        j++;
    }

    mon.nodes[A].num_enll=mon.nodes[A].num_enll-1;
    nouA=antic.A;
    mon.arestes[index_antic].A=nouA;
    mon.arestes[index_antic].distància=antic.distància;
    mon.nodes[nouA].enllacos[mon.nodes[nouA].num_enll]=index_antic;
    mon.nodes[nouA].num_enll++;
}

double  cost()
{
    double  cost_total=0;
    double pA,pB;
    pA=pB=0;
    eccent_node();

    for(int k=0; k<NUM_ENLLACOS; k++)
    {
        pA=pow(mon.arestes[k].distància,2)*mon.nodes[mon.arestes[k].A].ecc;
        pB=pow(mon.arestes[k].distància,2)*mon.nodes[mon.arestes[k].B].ecc;
        cost_total+=(pA+pB);
    }

    return cost_total;
}

void mostra_enllacos()
{
    int e;
    for (int i=0;i<MAX_NODES;i++)
    {
        if(mon.nodes[i].num_enll==0)
        {
            printf("NODE AILLAT:%i\n",i);
        }
    }
    printf("\nENLLAC\tnodeA\tnodeB\tdistància\n");
    for (e=0;e<NUM_ENLLACOS;e++)
    {
```

```
        printf("%i:\t%i\t%i\t%f\n",e,mon.arestes[e].A,mon.arestes[e].B,mon.arestes[e].distància);
    }
}

void elimina_ultim()
{
    NUM_ENLLACOS-=1;
    mon.nodes[mon.arestes[NUM_ENLLACOS].A].num_enll-=1;
    mon.nodes[mon.arestes[NUM_ENLLACOS].B].num_enll-=1;
    mon.nodes[NUM_ENLLACOS].ecc=0;
    mon.arestes[NUM_ENLLACOS].distància=0;
    mon.arestes[NUM_ENLLACOS].A=-1;
    mon.arestes[NUM_ENLLACOS].B=-1;
}

Tenllac elimina_enllac(int *baixa)
{
    Tenllac antic;
    int e;
    if(*baixa!=-1)
    {
        e=*baixa;
        mon.arestes[e].AeccA=mon.nodes[mon.arestes[e].A].ecc;
        mon.arestes[e].AeccB=mon.nodes[mon.arestes[e].B].ecc;
        antic.A=mon.arestes[e].A;
        antic.B=mon.arestes[e].B;
        antic.eccA=mon.arestes[e].AeccA;
        antic.eccB=mon.arestes[e].AeccB;
        antic.distància=mon.arestes[e].distància;
        for(int s=0;s<mon.nodes[mon.arestes[e].A].num_enll;s++)
        {
            if(mon.nodes[mon.arestes[e].A].enllacos[s]==e)
            {

mon.nodes[mon.arestes[e].A].enllacos[s]=mon.nodes[mon.arestes[e].A].enllacos[mon.nodes[m
on.arestes[e].A].num_enll-1];
                mon.nodes[mon.arestes[e].A].enllacos[mon.nodes[mon.arestes[e].A].num_enll-1]=e;
            }
        }
        mon.nodes[mon.arestes[e].A].num_enll-=1;
        for(int s=0;s<mon.nodes[mon.arestes[e].B].num_enll;s++)
        {
            if(mon.nodes[mon.arestes[e].B].enllacos[s]==e)
            {

mon.nodes[mon.arestes[e].B].enllacos[s]=mon.nodes[mon.arestes[e].B].enllacos[mon.nodes[m
on.arestes[e].B].num_enll-1];
                mon.nodes[mon.arestes[e].B].enllacos[mon.nodes[mon.arestes[e].B].num_enll-1]=e;
            }
        }

        mon.nodes[mon.arestes[e].B].num_enll-=1;
        mon.arestes[e].A=mon.arestes[NUM_ENLLACOS-1].A;
        mon.arestes[e].B=mon.arestes[NUM_ENLLACOS-1].B;
        mon.nodes[mon.arestes[e].A].ecc=mon.nodes[NUM_ENLLACOS-1].ecc;
        mon.nodes[mon.arestes[e].B].ecc=mon.nodes[NUM_ENLLACOS-1].ecc;
        mon.arestes[e].distància=mon.arestes[NUM_ENLLACOS-1].distància;

        for(int s=0;s<mon.nodes[mon.arestes[NUM_ENLLACOS-1].A].num_enll;s++)
        {
```

```
            if(mon.nodes[mon.arestes[NUM_ENLLACOS-1].A].enllacos[s]==NUM_ENLLACOS-1)
            {
               mon.nodes[mon.arestes[NUM_ENLLACOS-1].A].enllacos[s]=e;
            }
         }

         for(int s=0;s<mon.nodes[mon.arestes[NUM_ENLLACOS-1].B].num_enll;s++)
         {
            if(mon.nodes[mon.arestes[NUM_ENLLACOS-1].B].enllacos[s]==NUM_ENLLACOS-1)
            {
               mon.nodes[mon.arestes[NUM_ENLLACOS-1].B].enllacos[s]=e;
            }
         }
         NUM_ENLLACOS--;
         mon.nodes[mon.arestes[NUM_ENLLACOS].A].ecc=0;
         mon.nodes[mon.arestes[NUM_ENLLACOS].B].ecc=0;
         mon.arestes[NUM_ENLLACOS].distància=0;
         mon.arestes[NUM_ENLLACOS].A=-1;
         mon.arestes[NUM_ENLLACOS].B=-1;
         }
         return antic;
         }

void recupera_eliminat(Tenllac antic)
{
   mon.nodes[antic.A].enllacos[mon.nodes[antic.A].num_enll]=NUM_ENLLACOS;
   mon.nodes[antic.B].enllacos[mon.nodes[antic.B].num_enll]=NUM_ENLLACOS;
   mon.nodes[antic.A].num_enll++;
   mon.nodes[antic.B].num_enll++;
   mon.arestes[NUM_ENLLACOS].A=antic.A;
   mon.arestes[NUM_ENLLACOS].B=antic.B;
   mon.arestes[NUM_ENLLACOS].distància=antic.distància;
   mon.nodes[antic.A].ecc=antic.eccA;
   mon.nodes[antic.B].ecc=antic.eccB;
   NUM_ENLLACOS+=1;
}

Tenllac afegeix_enllac()
{
   Tenllac nou;
   int n1,n2,existeix;
   n1=n2=0;
   do
   {
      n1=rand()%(MAX_NODES);
      n2=rand()%(MAX_NODES);
      existeix=FALSE;
      if(n1!=n2)
      {
         for(int z=0;z<=NUM_ENLLACOS;z++)
         {
            if ((mon.arestes[z].A==n1 && mon.arestes[z].B==n2)||(mon.arestes[z].A==n2 &&
mon.arestes[z].B==n1))
            {
               existeix=TRUE;
               break;
            }
         }
      }
   }
```

```
    while(n1==n2 || existeix==TRUE);

    mon.nodes[n1].enllacos[mon.nodes[n1].num_enll]=NUM_ENLLACOS;
    mon.nodes[n2].enllacos[mon.nodes[n2].num_enll]=NUM_ENLLACOS;
    mon.nodes[n1].num_enll++;
    mon.nodes[n2].num_enll++;

    if (n1<n2)
    {
        mon.arestes[NUM_ENLLACOS].A=n1;
        mon.arestes[NUM_ENLLACOS].B=n2;
    }
    else
    {
       mon.arestes[NUM_ENLLACOS].A=n2;  mon.arestes[NUM_ENLLACOS].B=n1;
    }
    mon.arestes[NUM_ENLLACOS].distància=distància(n1,n2);
    NUM_ENLLACOS+=1;

    return mon.arestes[NUM_ENLLACOS-1];
}

void ensenya()
{
    for(int x=0;x<MAX_NODES;x++)
    {
       printf("\nnode:%i-> ",x);
       for(int s=0;s<mon.nodes[x].num_enll;s++)
       {
       printf("%i ",mon.nodes[x].enllacos[s]);
       }
    }
}

int main(int argc, char *argv[])
{
    Tenllac antic;
    int index,acceptat,no_acceptat,iscon,bons,dolents,s,node,enllac;

    float p=0;
    srand(time(NULL));
    char net[25];
    char math[25];
    if(argc!=2)
    {
       printf("\nescriu la capçalera del fixer de sortida");
       exit(1);
    }
   strcpy(net,argv[1]);
   strcpy(math,argv[1]);

   inicialitza_mon_quadricula();
   cost_inicial=cost();
   distànciametre();
   clustering();
   escriu(argv[1],0);
   networkx(net,"_netx_ini.grf");

   bons=acceptat=dolents=0;
   temperatura=TEMP_INICIAL;
```

```
while(temperatura>TEMP_MINIMA)
{
    temperatura=0.9*temperatura;
    for(int Z=0;Z<NK;Z++)
    {
       p=float(rand())/float(RAND_MAX);
       if(p<0.8)
       {
          s=0;
    }
    else
       if(p<0.9)
       {
          s=1;
       }
       else
       {
          s=2;
       }
       switch(s)
       {
          case 0:
             antic=canviar_enllac(&index);
             break;

          case 1:
             enllac=rand()%(NUM_ENLLACOS-1);
             antic=elimina_enllac(&enllac);
             break;

          case 2:
             if(NUM_ENLLACOS<MAX_ENLLACOS-1)
             {
                afegeix_enllac();
             }
             break;
       }

       if(connex()==1)
       {
          no_acceptat=0;
          cost_actual=cost();
          f=cost_inicial-cost_actual;

          if(f<0)
          {
             dolents ++;
             if((exp(f/(0.001*temperatura)))>(float(rand())/float(RAND_MAX)))
             {
                acceptat++;
                cost_inicial=cost_actual;
             }
             else
             {
                no_acceptat=1;
             }
          }
          else
          {
```

```
                bons++;cost_inicial=cost_actual;
            }
        }
        else{no_acceptat=1;
        }
        if(no_acceptat==1)
        {
            switch(s)
            {
                case 0:
                    torna_endarrera(index,antic);
                    break;

                case 1:
                    recupera_eliminat(antic);
                                break;

                case 2:
                    elimina_ultim();
                    break;
            }
        }
      }
      acceptat=bons=dolents=0;
   }

   distànciametre();
   clustering();
   escriu(argv[1],1);
   networkx(net,"_netx_fi.grf");
   exit(1);
}
```