OPTIMITZACIÓ COMBINATÒRIA I DISSENY DE XARXES D'INTERCONNEXIÓ

FRANCESC COMELLAS I EDUARD PALLARÈS
Departament de Matemàtica Aplicada i Telemàtica
Escola Tècnica Superior d'Enginyers de Telecomunicació
Universitat Politècnica de Catalunya
Campus Nord, Edifici C3, 08071 Barcelona

1. Introducció

En els darrers anys s'han considerat tècniques noves d'optimització amb ordinador que han donat bons resultats en el tractament de problemes combinatoris complexos. Aquests problemes consisteixen a trobar un mínim o màxim global d'una funció de cost definida en un conjunt d'estats (o de solucions possibles). D'entre aquestes tècniques destaquem les xarxes neuronals, la recuita simulada (*simulated annealing*) i els algorismes genètics. En aquest article es mostra la seva aplicabilitat al disseny de xarxes d'interconnexió i en concret al problema d'obtenir una xarxa d'interconnexió plana a partir d'una xarxa possiblement no plana. Es fa èmfasi, en particular, en la recuita simulada, mètode basat en l'analogia entre les configuracions possibles d'un problema d'optimització combinatòria i els estats d'un sistema físic del qual es busca l'estat d'energia mínima. La tècnica s'ha emprat per a determinar una funció de cost adequada als problemes d'aplanament. També, a la darrera secció, s'ha comparat la seva efectivitat en relació als algorismes genètics.

2. Optimització combinatòria

En l'optimització combinatòria es considera un espai d'estats, o conjunt discret de possibles solucions a un problema donat, juntament amb una funció de cost que assigna un nombre real a cadascuna de les solucions. Es tracta de trobar la solució de cost òptim (mínim en aquest article) d'entre totes les possibles solucions que pot tenir el problema.

Aquest tipus de qüestions es plantegen en diversos camps de la ciència i l'enginyeria. Potser l'exemple més conegut és el problema del viatjant que se sol considerar com a model per a comprovar l'eficàcia dels diferents mètodes de resolució [11]. En la formulació més coneguda d'aquest problema, un viatjant ha de visitar un cert nombre de ciutats passant únicament una vegada per cadascuna d'elles i retornant a la ciutat de partença. Es tracta de trobar aquell trajecte que faci mínima la distància total recorreguda. Si el nombre de ciutats considerades és N, el tractament exhaustiu d'aquest problema comporta estudiar $(N-1)!/2$ recorreguts. Aquest nombre, que creix més de pressa que qualsevol potència finita de N, fa que el problema esdevingui rapidament intractable. El problema del viatjant pertany a una classe de problemes anomenats *NP-complets* [9]. Per a aquests problemes no es coneixen algorismes que garanteixin que la solució òptima pugui trobar-se en un temps raonable d'execució d'un programa.

Atès que molts problemes d'optimització combinatòria són del tipus NP-complet on la cerca exhaustiva no és factible, molt sovint hom es comforma a trobar una solució quasi-òptima. Diverses alternatives són aleshores possibles.

Per a problemes NP-complets concrets, s'han desenvolupat algorismes heurístics capaços de trobar una solució acceptable en un temps limitat, tot i que no s'assegura que trobin la solució òptima. Tanmateix, els algorismes heurístics depenen del problema considerat. Un algorisme heurístic pot ésser molt eficient per a trobar una solució quasi-òptima d'un problema determinat i resultar totalment inútil en un altre. A més, molt sovint els algorismes heurístics no poden escapar dels mínims locals.

Les estratègies dels algorismes heurístics responen a diversos estils: tècniques constructives, mètodes de partició i mètodes de millora iterativa.

Els algorismes constructius creen la solució directament tenint en compte les característiques pròpies del problema.

Els mètodes de partició, o també de *dividir per a vèncer*, separen el problema en problemes més petits dels quals és més fàcil trobar una solució. Després es genera la solució global a partir de les solucions parcials trobades. Aquests mètodes són efectius, lògicament, si els subproblemes són disjunts.

Els mètodes de millora iterativa són de més interès ja que es poden aplicar a un conjunt gran de problemes. El més conegut és la *cerca local*. Com el seu nom indica, es tracta de partir d'una solució subòptima i intentar trobar una solució millor explorant iterativament a partir d'ella l'espai d'estats mitjançant petits canvis. L'algorisme sol mantenir la millor solució trobada i no n'accepta cap que no sigui superior. El procés continua fins que els canvis no aconsegueixen cap millora. Els mètodes de millora successiva més emprats acostumen a ésser descendents: a cada iteració el cost de la funció objectiva ha de disminuir respecte al cost anterior. Això comporta molt sovint que el mètode s'encalli en un mínim local. Per aquest motiu, a la pràctica s'acostuma a executar diferents vegades l'algorisme a partir de configuracions inicials aleatòries per a guardar la millor solució que es trobi. És evident que per a problemes grans aquest

procés no només és impracticable en temps sinó que a més no queda garantit que es pugui aconseguir la millor solució de tot l'espai d'estats. Una altra variant de la millora iterativa és la *cerca aleatòria,* la qual genera una mostra a l'atzar de l'espai d'estats, avalua el cost per a cadascuna de les solucions considerades i es queda amb la solució de menor cost. Tots aquests mètodes resulten, en la pràctica, poc eficients per a trobar bones solucions en el cas de problemes NP-complets.

Existeixen altres mètodes, que poden relacionar-se també amb la millora iterativa i que, com ella, són d'aplicabilitat general. Resulten particularment interessants pel que fa a la seva eficiència i al fet que, a la vegada que són relativament simples de programar, es poden adaptar fàcilment per a resoldre problemes molt diversos.

Curiosament, tots ells tenen en comú un principi de funcionament que és inspirat amb fenòmens de la naturalesa. Entre ells podem esmentar les xarxes neuronals, els algorismes genètics i la recuita simulada (*simulated annealing*).

Les xarxes neuronals, basades en el model de Hopfield i Tank [5], són utilitzades per a tractar diversos problemes combinatoris. Una xarxa neuronal és formada per un cert nombre d'elements o neurones artificials les quals cooperen a buscar l'estat corresponent al mínim (o màxim) local o global. Una descripció detallada la podeu trobar en l'article de Pau Bofill publicat en aquest mateix volum.

Els algorismes genètics [4] formen una altra família d'algorismes amb possibilitats d'utilització molt generals. Foren introduïts per J.H. Holland els anys 60 i han estat aplicats amb èxit a un gran nombre de problemes diversos (vegeu [6] per a una descripció detallada amb bibliografia). En un algorisme genètic el punt de partença és una col·lecció de possibles solucions (o individus) generades aleatòriament i que rep el nom de *població* o *generació* . La codificació concreta de les solucions constitueix un primer aspecte important de l'algorisme i n'afecta l'eficiència. Una vegada construïda la primera generació es determina el cost de cadascuna de les seves solucions. Llavors es procedeix a crear una nova generació mitjançant la *reproducció, encreuament* i *mutació.* Les solucions millors tenen una probabilitat més alta de participar en aquest procés. El encreuament de solucions es realitza de manera que dues solucions pare donen dues solucions fill mitjançant l'intercanvi aleatori de fragments dels pares, d'acord amb la codificació emprada. Un altra aleatorietat s'introdueix mitjançant la mutació que modifica lleugerament les solucions generades de cara a evitar que l'algorisme es quedi en un mínim local. Les operacions de encreuament i mutació es realitzen amb una certa probabilitat, paràmetres importants a l'algorisme. El fet que no totes les solucions d'una generació participin en el procés comentat garanteix que algunes solucions de la generació actual continuïn presents a la següent generació. Un cop la nova generació s'ha creat, el cost de totes les solucions o individus es torna a avaluar i el procés es repeteix. A cada generació es guarda la millor solució. L'algorisme

acaba quan el resultat s'estabilitza o es troba la solució òptima si aquesta pot ésser identificada.

En la següent secció comentarem en detall la recuita simulada, una altra tècnica que, com les comentades, es mostra també molt adequada per a la resolució d'un ventall ampli de problemes combinatoris.

3. La recuita simulada

L'any 1953, Nicolas Metropolis, Arianna i Marshall Rosenbluth i Augusta i Edward Teller [12] proposaren un algorisme que evitava mínims locals en la cerca de configuracions estables d'un conjunt d'àtoms a una certa temperatura. La base de l'algorisme és que s'accepten, a més de canvis aleatoris que disminueixen l'energia del sistema, també canvis que l'augmenten amb una probabilitat donada pel factor de Boltzmann $e^{-\Delta E/KT}$. Com més alta és la temperatura, més probable és que s'accepti el canvi en l'estat del sistema. El sistema evoluciona, en aquesta simulació, cap a l'equilibri tèrmic, i els paràmetres macroscòpics varien segons la distribució de Boltzman corresponent a la temperatura T.

Aquest algorisme fou adaptat a la resolució de problemes combinatoris, i és ara conegut com a recuita simulada o *simulated annealing*, per S. Kirkpatrich, C.D. Gelatt i M.P. Vecchi l'any 1983 [10]. De fet és una variant d'un algorisme de millorament iteratiu: s'accepten sempre canvis que comporten una reducció del cost, però, amb una probabilitat relacionada amb el factor de Boltzmann, també s'accepten solucions que comporten un augment del cost per tal d'evitar de caure en un mínim local.

La tècnica de la recuita simulada prové, doncs, de l'analogia que hom fa entre el problemes de la mecànica estadística de trobar l'estat bàsic o fonamental d'un sistema de diversos cossos (per exemple un líquid), i el de trobar el mínim (o màxim) global d'una funció de cost en un problema d'optimització combinatoria. Si la temperatura en la interacció molecular d'un líquid es reduís sobtadament per sota del punt de fusió, el resultat seria un estat desordenat pseudo-cristal·lí amb una energia més alta que la de l'estat crital·lí real. De fet, les molècules es trobarien en un mínim local d'energia. En canvi, si la temperatura del líquid es redueix lentament (recuita) d'acord amb una pauta de refredament adequada, es tendeix cap l'equilibri i el líquid es congela a través d'un procés que condueix a un estat cristal·lí amb energia global mínima.

En la recuita simulada, els paràmetres que es varien del problema d'optimització combinatòria són equiparats amb les posicions de les molècules del líquid i la funció de cost a optimitzar s'identifica amb l'energia. La temperatura s'hi defineix com un paràmetre de control relacionat amb la probabilitat que s'acceptin canvis a un estat pitjor.

Una iteració elemental consta de dues etapes: la generació d'un nou estat, mitjançant una pertorbació petita de la solució actual, i l'aplicació del criteri d'acceptació. Per a una temperatura donada, l'algorisme accepta sempre els canvis d'estat que fan disminuir el cost; si el cost augmenta, el canvi és acceptat amb una certa probabilitat que depèn de la temperatura del sistema d'acord amb l'equació $e^{-\Delta f/T}$ on $\Delta f = f(j) - f(i)$ mesura la variació del valor de la funció de cost entre els estats i i j i T és la temperatura del sistema. El nombre total d'iteracions que es fan per a aquesta temperatura ha d'assegurar que l'espai ha estat suficientment explorat.

Una vegada realitzades les iteracions previstes per a la temperatura considerada, aquesta es decrementa i es repeteix el procés. D'aquesta forma el sistema es va refredant fins a aturar-se d'acord amb algun criteri predeterminat (s'ha assolit la temperatura final, s'han acceptat pocs canvis o no hi ha hagut reducció del cost a la darrera temperatura, etc.).

Una característica típica de la tècnica de la recuita simulada és que per a valors inicials de la temperatura, i pel fet que aquesta és relativament alta, s'accepten empitjoraments importants del cost; després, a mesura que la temperatura va disminuint, s'accepten empitjoraments cada vegada menors, i finalment, quan la temperatura s'apropa a zero, la probabilitat d'acceptar un augment de cost és pràcticament nul·la. Aquest fet és el que permet escapar dels mínims locals i assegura la bona convergència de la tècnica.

La figura 1 presenta en pseudo-codi una possible descripció de l'algorisme de recuita simulada on T_k denota el paràmetre de control o temperatura i N_k el nombre d'iteracions que es fan en aquesta temperatura. La convergència de l'algorisme dependrà, lògicament, de la tria adequada d'aquests paràmetres. A la secció següent es discutiran quins són els valors més idonis per a T_k i N_k en el cas del problema d'aplanament estudiat.

```
INICIALITZAR (i_inicial, T_0, N_0);
k := 0;
i := i_inicial;

repetir
   Per a l := 1 fins N_k fer
      GENERAR (j a partir de Estats(i));
      Si f(j) <= f(i)
         Aleshores i := j;
      Altrament
         Si exp ((f(i) - f(j)) / Tk) < aleatori [0,1)
            Aleshores i := j;
   k := k + 1;
   CALCULAR_ITERACIO (N_k);
   CALCULAR_TEMPERATURA (T_k);
fins CRITERI_D'ATURADA
```

Fig. 1. L 'algorisme de recuita simulada.

Finalment cal comentar que la convergència de l'algorisme de la recuita simulada cap a la solució òptima global pot ésser justificada, com ja s'ha esmentat, a partir de la seva correspondència amb la física estadística. Una demostració més formal es pot realitzar considerant la teoria de cadenes finites de Markov [13, 1, 14, 15].

4. Aplanament de grafs

En aquest article estudiem l'aplicació i l'eficiència de la recuita simulada en el disseny de xarxes d'interconnexió. Ens centrarem en l'obtenció de xarxes que puguin representar-se en un pla sense talls. Com se sap, les xarxes d'interconnexió es poden modelar mitjançant grafs. Recordem que un graf $G(V,E)$ és una estructura constituïda per un conjunt finit d'elements V anomenats vèrtexs i un conjunt també finit E de parells no ordenats de vèrtexs anomenats branques. Els nodes de la xarxa s'associen als vèrtexs del graf i les connexions a les branques. Així, en termes de Teoria de Grafs, el problema concret que hem estudiat consisteix en la cerca d'un subgraf pla maximal a partir d'un graf donat en general no pla. El subgraf ha de tenir, doncs, el màxim nombre possible de branques del graf original sense que hi hagi talls entre elles. Aquest problema, a més del seu interés teòric, es relaciona directament amb el disseny de plaques de circuit imprès tant pel que fa al camí que han de recórrer les pistes elèctriques, com en l'aspecte de la distribució òptima dels diversos components sobre la placa. També afecta el disseny de circuits VLSI, així com altres qüestions de tipus tècnic.

L'aplanament de grafs encara no havia estat estudiat mitjançant la recuita simulada, la qual s'havia aplicat amb èxit a d'altres problemes NP-complets de Teoria de Grafs que tenen també aplicacions pràctiques evidents. Destaquem [13]:

- *Problema del conjunt de tall maximal (o minimal).* Consisteix a buscar per un graf amb pesos a les branques $G(V,A)$, una partició dels vèrtexs V en dos subconjunts V_0 i V_1 tal que $V = V_0 \cup V_1$ i $V_0 \cap V_1 = \emptyset$ i en la qual la suma dels pesos corresponents a branques que uneixen els dos conjunts sigui màxima (o mínima). El problema de la partició de grafs correspon al cas particular en què tots els arcs del graf tenen el mateix pes.

- *Problema de conjunt independent.* Consisteix a buscar un conjunt independent maximal de vèrtexs $V' \subseteq V$ tal que entre dos vèrtexs qualssevol de V' no hi hagi cap branca que els uneixi.

- *Coloració de grafs.* Consiteix a buscar la coloració mínima d'un graf $G(V,E)$. És a dir: donat un conjunt de colors C trobar una aplicació de

V en C tal que dos vèrtexs adjacents tinguin diferent color i C sigui de cardinalitat el més petita possible.

- *Problema de l'arbre de Steiner.* Donat un graf no dirigit amb branques amb pesos $G(V,A)$ i un subconjunt convenient V' de V, buscar l'arbre generador de mínim pes que cobreixi els vèrtexs de V' i si cal d'altres no pertanyents a V.

El problema d'aplanament de grafs que ens ocupa, tal com s'ha indicat, és del tipus NP-complet. Per aquest motiu interessa donar mètodes que, encara que no puguin garantir la solució millor, permetin trobar solucions quasi-òptimes. Jayakumar i altres a [7] van presentar un algorisme heurístic d'ordre $O(N^2)$, on N és el nombre de vèrtexs. L'aplicació de l'algorisme al graf de 22 branques de la figura 2 els permetia de trobar un subgraf pla amb 19 branques.

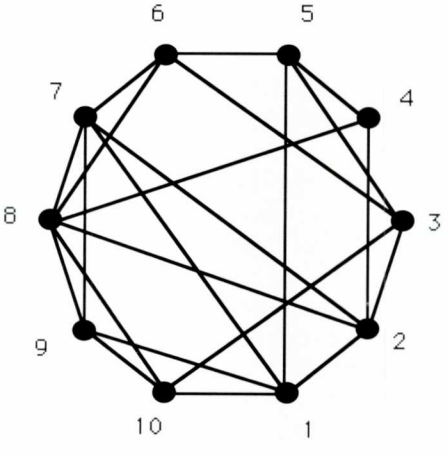

Fig. 2. El graf de 22 branques estudiat per Jayakumar *et al.*.

Takefuji i K.C. Lee a [8] i emprant una xarxa neuronal $N \times N$ sobre el mateix graf obtingueren una solució amb 20 branques. Un resultat equivalent [2] s'obté utilitzant algorismes genètics amb l'avantatge de treballar amb un algorisme conceptualment més simple.

En aquesta Secció, es considera la recuita simulada per al tractament d'aquest problema. Per a la seva aplicació s'ha de fer una representació adequada del graf per a poder explorar fàcilment l'espai d'estats. D'altra banda també cal determinar la funció de cost que faci l'algorisme el màxim d'eficient.

Per a la solució d'aquest problema d'aplanament mitjançant la recuita simulada, l'espai d'estats de les solucions possibles s'obté del graf original dibuixant-ne els vèrtexs en una única fila. Les branques, quan es consideren, poden dibuixar-se per sobre o bé per sota de la fila. D'aquesta manera poden aparèixer talls entre branques (vegeu la figura 3).

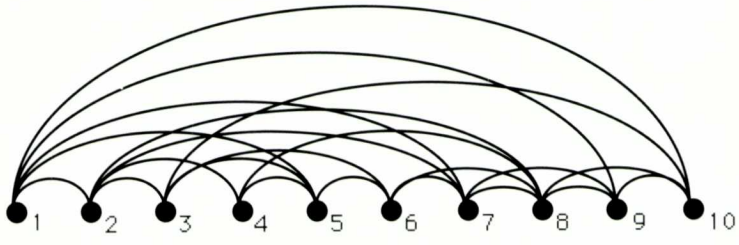

Fig. 3. Representació en fila del graf de Jayakumar.

Una solució de l'espai de'estats consistirà en una llista on cada element correspon a una branca del graf original i pren un valor que indica si la branca passa per sobre de la fila de vèrtexs, per sota o no es considera.

L'algorisme tractarà d'obtenir a partir d'una solució inicial generada a l'atzar, i a través dels mecanismes de la recuita simulada, una solució sense talls amb el màxim nombre de branques presents del graf original que volem aplanar.

Un aspecte important és la funció de cost, la qual es defineix sobre l'espai d'estats i assigna un nombre real, el cost, a cadascun dels estats. La tria de la funció de cost afecta directament l'eficàcia de l'algorisme. Una certa funció de cost pot accentuar excessivament les diferències entre estats pròxims i facilitar que l'algorisme s'encalli en mínims locals. Així part d'aquest estudi s'ha dedicat a la determinació d'una funció de cost adequada per a problemes d'aplanament de grafs i a donar els valors adients per als coeficients que puguin intervenir en aquesta funció. Una funció de cost que ha resultat efectiva és la següent:

$$C = (N - B) + \alpha X$$

on N és el nombre de branques del graf original, B és el nombre de branques del subgraf corresponent, X indica el nombre de talls entre branques i α és un coeficient que pondera la importància que cal donar als talls respecte a les branques a l'hora de calcular el cost. La determinació de les variables ve facilitada per la reprentació en fila del graf.

La funció de cost és sempre positiva ($\alpha > 0$). El problema és, doncs, un problema de minimització i la fita és aconseguir, si això és possible, una solució amb cost nul que correspondria a haver col·l ocat totes les branques del graf original de manera plana.

D'altra banda, la funció depèn d'α, únic paràmetre que no es pot trobar a partir de la representació emprada. Una condició que ha de verificar aquest paràmetre és que $\alpha > 1$. D'aquesta manera es dóna més importància al fet que no hi hagi talls respecte a la presència de branques. Per a determinar un valor d'α adequat al problema s'han realitzat unes proves per a diferents grafs aleatoris amb ordres entre 10 i 50 i nombre de branques entre 20 i 100. Per a cada graf considerat s'ha executat 10 vegades l'algorisme, per a cadascun dels valors d'α estudiats, i s'ha guardat el nombre total d'iteracions efectuades, canvis acceptats, iteracions que s'han necessitat per a trobar les millors solucions i el nombre de branques d'aquestes. Com a exemple, la taula 1 mostra el resultat d'una d'aquestes proves. A partir del conjunt de resultats es pot comprovar que un conjunt de valors adequat per a aquest coeficient és:

$$1.10 \leq \alpha \leq 1.20$$

alfa	<iteracions>	<canvis>	1r millor	branques de la solucio 62	64	66	68	70	72
1.01	189729	97513	94855			1		9	
1.05	151374	89604	99206					10	
1.10	133475	84034	90228					10	
1.15	125548	84076	92535			2		8	
1.20	120179	83173	98500					10	
1.25	116854	83487	102621					10	
1.30	112252	81751	99165				1	9	
1.35	109951	81767	99532			1	3	6	
1.40	103558	77172	98024			2	5	3	
1.45	104837	80705	98065		2	4	3	1	
1.50	100745	77086	95625	2	4	3	1		

Taula 1. Estudi de la convergència de l'algorisme per a diferents valors de Graf aleatori d'ordre 30 i 72 branques.

Un altre paràmetre important de l'algorisme és la temperatura inicial. Si s'escull una temperatura inicial molt elevada, l'algorisme acceptarà fàcilment qualsevol canvi d'empitjorament de la funció de cost, de manera que el cost oscil·larà constantment amunt i avall fins que no disminueixi la temperatura

i el nombre d'iteracions totals augmentarà innecessàriament. En canvi, l'elecció d'una temperatura inicial massa freda comporta que no s'accepti gairebé mai un empitjorament de la funció de cost, la qual cosa fa que l'algorisme pugui quedar encallat en un mínim local. Així doncs, cal establir un compromís en l'elecció de la temperatura inicial. En aquest problema ha donat bons resultats considerar aquella que accepta un increment mitjà del cost amb probabilitat $p = 0.2$. Per a determinar quin és aquest increment mitjà es fan, sobre el subgraf inicial generat a l'atzar, un cert nombre de pertorbacions aleatòries i s'obté la mitjana dels diferents increments del cost. A partir del factor de Boltzman es troba $T_0 = -(\Delta f)_m / \ln 0.2$.

Es important també d'establir una temperatura final, la qual ens dóna un dels criteris per a aturar l'algorisme. S'ha proposat aquella que fa molt poc probable l'acceptació d'un empitjorament del cost. En concret s'ha triat com temperatura final la que fa que si a un subgraf donat se li afegís una branca i el nombre de talls augmentés en una unitat, l'empitjorament del cost fos acceptat amb probabilitat $p = 0.005$. Com abans i ja que l'increment de cost és en aquest cas $\Delta f = \alpha - 1$ es troba que $T_f \approx (\alpha - 1)0.2$.

Entre la temperatura inicial i la final se segueix una determinada pauta de refredament. De les pautes considerades a la literatura s'ha triat la més habitual: el refredament exponencial. En aquest cas $T_k = T_0 r^k$ on T_k és el valor de la temperatura un cop refredada k vegades, T_0 és la temperatura inicial i r és la raó de refredament que sol prendre un valor a l'entorn de 0.9.

Un altre paràmetre característic de la recuita simulada és N_k, el nombre d'iteracions que es fan a una temperatura T_k. Aquest paràmetre ha d'assegurar que l'espai d'estats sigui suficientment explorat. Ha resultat adequat el valor considerat per F. Darema i altres a [3]: $N_k = M(M-1)/2$, on M és un indicador de la mida del problema i en el nostre cas és el nombre de branques del graf original. També, per a cada temperatura, ha d'haver-hi un nombre mínim de canvis acceptats. S'ha fixat aquest en la dècima part de les iteracions realitzades a la temperatura considerada, així doncs: $m_k = N_k / 10$.

Aquests dos paràmetres ens proporcionen nous criteris per a l'aturada de l'algorisme. Així, si un cop s'han fet N_k iteracions a una temperatura, el nombre d'acceptacions del cost no ha arribat a m_k, s'incrementa el nombre d'iteracions en aquesta temperatura fins que s'aconsegueixi el nombre mínim de canvis. Si, tot i així, el nombre d'iteracions a una temperatura arriba a $2N_k$ sense haver aconseguit aquest nombre de canvis m_k, es considera que la solució s'ha estabilitzat en un mínim de l'espai d'estats i l'algorisme s'atura.

Aquests criteris d'aturada són importants per a l'eficiència de l'algorisme. Una parada prematura pot donar una solució no òptima, però allargar l'algorisme un temps considerable pot no dur-nos necessàriament a una solució millor.

El darrer criteri d'aturada considerat és obvi. Si s'aconsegueix una solució de cost nul l'algorisme no continua. És clar que s'ha trobat la solució desitjada.

5. Resultats. Comparació amb un algorisme genètic

La tècnica de la recuita simulada ha estat aplicada a un conjunt de grafs de diferents mides i ha donat resultats bons de manera eficient. Per exemple, a partir del graf aleatori d'ordre 12 i 28 branques de la figura 4 ha estat possible trobar en 10213 iteracions el graf pla de la figura 5 amb 19 branques. La figura 6 mostra la variació del cost en la cerca d'aquest subgraf pla.

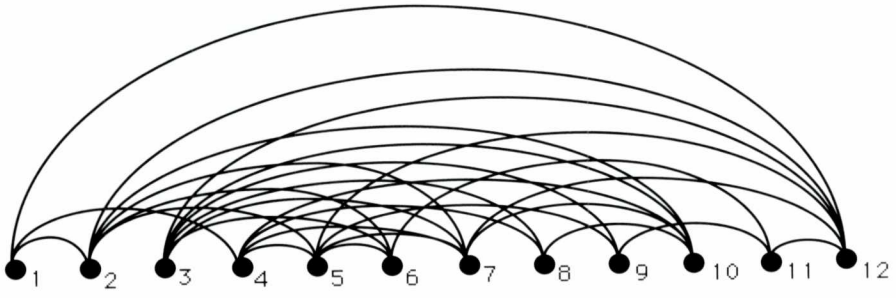

Fig. 4. Graf aleatori de 12 vèrtexs i 28 branques.

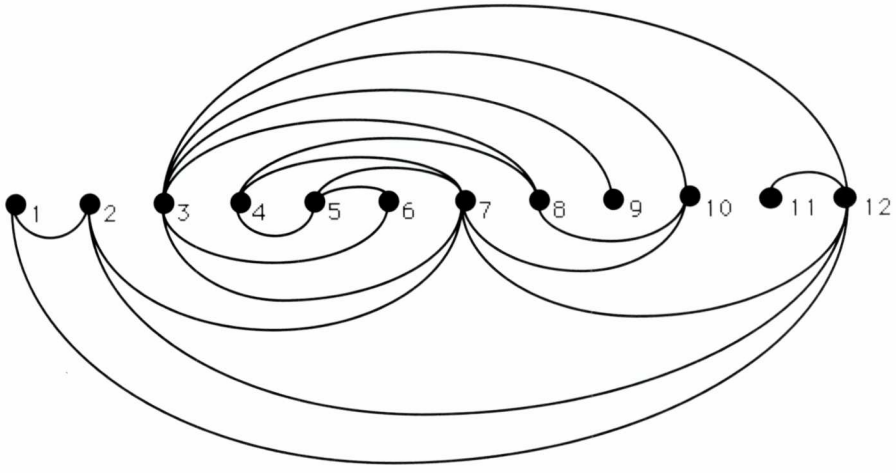

Fig. 5. Graf pla obtingut per recuita simulada pel graf de la figura anterior.

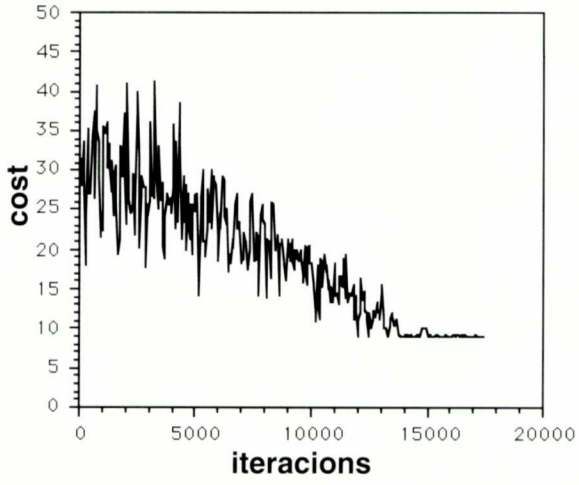

Fig. 6. Variació del cost del graf a cada iteració. Graf aleatori d'ordre 12 i 28 branques (a = 1.13).

El graf considerat per Jayakumar *et al.* també ha estat estudiat amb la tècnica de la recuita simulada. En aquest cas 3310 iteracions han permés de trobar una solució amb 20 branques a partir del graf inicial d'ordre 10 i 22 branques de la figura 2 (α =1.13).

La tècnica de la recuita simulada ha resultat molt adequada per a aquests problemes d'aplanament. Hem volgut, finalment, comparar-la amb una altra tècnica que ha demostrat ésser també d'interès per al tractament de problemes NP-complets: els algorismes genètics. Per a això hem emprat un algorisme genètic, descrit a [2], fent servir els mateixos grafs que amb la recuita simulada. Aquí presentem les comparacions correponents al cas dels grafs de les figures 2 i 5. Els resultats es mostren a les figures 7 i 8.

Aquestes figures indiquen a les ordenades el nombre de vegades que els algorismes han trobat la solució òptima dins del marge d'iteracions indicat a les abscisses (intervals de 1000 iteracions). En total s'ha executat 100 vegades cadascun dels algorismes per a cada graf.

Els resultats indiquen una millor eficiència de la recuita simulada ja que convergeix cap la solució òptima un nombre elevat de vegades amb menys iteracions que l'algorisme genètic considerat. En concret, i per al grafs de Jayakumar i l'aleatori de la figura 3, la recuita simulada troba la solució òptima el 95 % i el 100 % de les vegades, respectivament, enfront del 64 % i el 62 % de l'algorisme genètic. En mitjana la recuita simulada ha trobat els bons grafs a les 5883 i 12946 iteracions davant dels 6714 i 22472 càlculs de la funció de cost que ha d'efectuar l'algorisme genètic. Tanmateix, en alguns casos l'algorisme genètic ha trobat la solució amb menys càlculs de la funció de cost (1800 i 5900 enfront dels 3350 i 10250 efectuats per la recuita simulada).

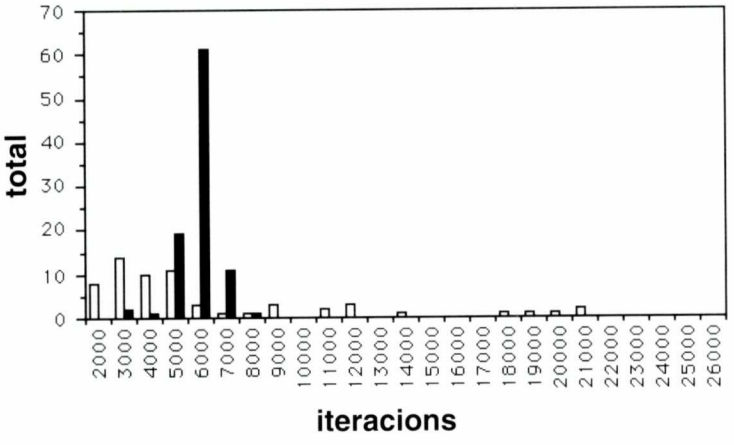

Fig. 7. Graf de Jayakumar. Comparació entre la recuita simulada (■) i l'algorisme genètic (□).

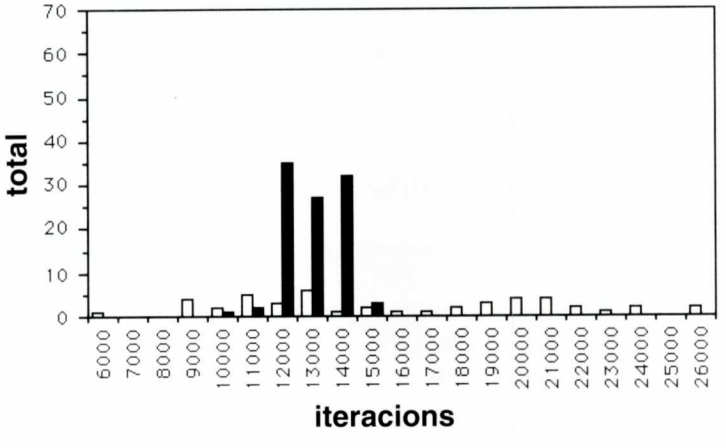

Fig. 8. Graf aleatori de 12 vèrtexs i 28 branques. Comparació entre la recuita simulada (■) i l'algorisme genètic (□).

La recuita simulada també té l'avantatge de necessitar menys memòria, ja que sols ha de guardar un únic graf durant l'execució de l'algorisme. En canvi, l'algorisme genètic ha de disposar de memòria suficient per a guardar tota una població de grafs.

Un altre avantatge de la tècnica de la recuita simulada és la senzillesa amb què es pot programar l'algorisme i, en conseqüència, la facilitat d'aplicar aquesta tècnica a problemes molt diversos d'optimització combinatòria.

BIBLIOGRAFIA

[1] E. AARTS AND J. KORST. *Simulated Annealing and Boltzmann Machines.* John Wiley & Sons. Chichester, 1989. ISBN 0-471-92146-7

[2] F. COMELLAS. Using genetic algorithms for planarization problems. *Computational and Applied Mathematics I: Algorithms and Theory.* Elsevier Science Publishers B.V., pp. 93-100, 1992. ISBN 0-444-89701-1.

[3] F. DAREMA, S. KIRKPATRICK AND V.A. NORTON. Parallel algorithm for chip placement by simulated annealing. *IBM J. Res. Develop.*, **31**, pp. 391-402, 1987.

[4] J.H. HOLLAND. Genetic algorithms. *Scientific American*, **267**-1, pp. 44-50, Juliol 1992.

[5] J.J. HOPFIELD AND D.W. TANK. Neural computation of decisions in optimization problems. *Biol. Cybernet.*, **52**, pp. 141-152, 1985.

[6] D. E. GOLDBERG, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley, 1989. ISBN 0-201-15767-5.

[7] R. JAYAKUMAR, K. THULASIRAMAN AND M.N.S. SWAMY, $O(n_2)$ algorithms for graph planarization, IEEE Trans. Computer-Aided Design, 8 (1989), 257-267.

[8] Y. TAKEFUJI AND K.C. LEE, A near-optimum parallel planarization algorithm, Science, 245 (1989), 1221-1223.

[9] M.R. GAREY AND D.S. JOHNSON, Computers and Intractability: A Guide to the Theory of NP-Completeness, Freeman, New York, 1979. ISBN 0-7167-1044-7.

[10] S. KIRKPATRICK, C.D. GELATT AND M.P. VECCHI, Optimization by Simulated Annealing. *Science*, **220**, 671-680, 1983.

[11] E.L. LAWLER, J.K. LENSTRA, A.H.G. RINNOOY KAN AND D.B. SHMOYS. *The Traveling Salesman Problem. A Guided Tour of Combinatorial Optimization.* John Wiley & Sons. Chichester, 1985. ISBN 0-471-90413-9.

[12] N. METROPOLIS, A. ROSENBLUTH, M. ROSENBLUTH, A. TELLER AND E. TELLER, *J. Chem. Phys.*, **21**, 1087, 1953.

[13] R.H.J.M. OTTEN AND L.P.P.P. VAN GINNEKEN. *The Annealing Algorithm.* Kluwer Academic Publishers, Boston, 1989. ISBN 0-7923-9022-9.

[14] F. ROMEO I A. SANGIOVANNI-VINCENTELLI. A theoretical framework for simulated annealing. *Algorithmica*, **6**, pp. 302-345, 1991.

[15] P.N. STRENSKI I S. KIRKPATRICK. Analysis of finite length annealing schedules. *Algorithmica*, **6**, pp. 346-366, 1991.