Contribucions

Lazlo Lovász i Avi Widergson: Premis Abel 2021

Juanjo Rué,
Departament de Matemàtiques, UPC

Fotografia dels dos guanyadors del premi Abel 2021

El dia 17 de març l'Acadèmia de Ciències Noruega va anunciar que el Premi Abel 2021 s'atorgava a Laszló Lovász i Avi Widgerson per, segons es llegeix de la *laudatio* del premi, "... les seves contribucions fonamentals a la informàtica teòrica i les matemàtiques discretes, i el seu paper principal en la seva configuració en camps centrals de les matemàtiques modernes..." Aquesta curta frase no podria resumir millor les contribucions fonamentals dels dos investigadors. Widgerson (Haifa, 1956) es va criar en el si d'una família jueva d'origen polonès supervivent de l'holocaust nazi. Segons recorda en més d'una entrevista, Widgerson va viure una infància i una adolescència molt tranquill·les, jugant a futbol i gaudint de la platja. El 1983 va obtenir a Princeton el seu doctorat en l'àrea de la complexitat computacional. Des d'aleshores, la seva carrera ha estat meteòrica, i ha fonamentat sobretot la teoria de la complexitat computacional. El Premi Abel és l'últim d'una llarga llista de reconeixements als innovadors i originals treballs de Widgerson en la fonamentació de la informàtica teòrica, entre els quals, el Premi Nevanlinna, el Premi Gödel i el Premi Knuth.

Per la seva banda, Lovász (Budapest, 1948) va ser un nen prodigi de les matemàtiques: per exemple, va guanyar tres medalles d'or a les Olimpíades Matemàtiques internacionals, en dues ocasions amb un reconeixement especial del jurat. Lovász pertany, de fet, a una generació de joves matemàtics hongaresos brillants i estimulats per l'ambient matemàtic de la Budapest de postguerra. En aquest punt, és necessari esmentar l'influència més gran per al jove Lovász: la del ja mític matemàtic Paul Erdős, amb qui va establir una col·laboració molt fructífera des de l'adolescència. De la mateixa manera que Widgerson, el Premi Abel és el darrer d'un seguit de reconeixements: els Premis Gödel i Knuth, així com el Premi Wolf, el Premi Kyoto i fins i tot, a casa nostra, el Premi Hipàtia.

No ens allargarem en els detalls biogràfics i passarem a la ciència. En farem un tast de quatre contribucions diferents (i prou representatives), íntimament lligades per aquest principi d'interacció fonamental entre la matemàtica discreta i la informàtica teòrica que impregna la seva obra.

Contribució conjunta de Lovász i Wigderson a l'algorísmia

Josep Díaz, Departament de Ciències de la Computació, UPC

És un plaer escriure sobre l'impacte científic en el camp de l'algorísmia de dos investigadors que admiro molt, els recents guardonats amb el Premi Abel, professors Laszló Lovász i Avi Wigderson. Els seus treballs han estat una contribució important al desenvolupament dels mètodes probabilístics en algorísmia. Per exemple, el *Lema local de Lovász*, que té una versió constructiva molt útil per dissenyar algorismes basats en mostreig, o les publicacions de Wigderson sobre paral·lelisme. A causa de les limitacions d'espai, en aquest article, únicament esbossaré la contribució de Lovász i Wigderson a un problema particular: la *k-vèrtex connectivitat d'un graf*. La seva contribució va ser donar un algorisme aleatoritzat més eficient que els existents l'any 1986, tot i que anys després altres investigadors van continuar les millores en el temps de resolució. He escollit aquest problema perquè és un dels dos únics treballs coll·laboratius, i perquè les tècniques emprades són diferents de les utilitzades per altres autors per resoldre el mateix problema.

Abans d'entrar en l'algorisme de Linial, Lovász i Wigderson [11], faré una breu introducció al concepte d'algorisme aleatoritzat. En l'actualitat el mot algorisme ha esdevingut omnipresent i conegut. Un algorisme és simplement una llista d'instruccions per processar informació i resoldre problemes. Els algorismes existeixen des del començament dels organismes cel·lulars, ja que l'ADN i l'ARN són algorismes amb les instruccions per produir proteïnes.[2] Els humans han dissenyat i implementat algorismes al llarg de més de 25 segles. Un dels paràmetres importants associats amb un algorisme és el "temps" o nombre de passos que triga a finalitzar el càlcul per a la instància "pitjor" del problema. Aquest "temps" s'expressa com a funció de la grandària n de l'entrada. Fent un abús de nomenclatura, em referiré a la complexitat d'un algorisme com al nombre de passos o temps que l'algorisme triga a resoldre un problema. Amb la tecnologia actual, un algorisme que tingui una complexitat polinòmica de n^6, per a valors de n prou grans pot trigar dies a donar la solució. Per exemple, un graf dirigit que vulgui simular la xarxa de Facebook haurà de tenir 1.6×10^9 vèrtexs. Per accelerar el temps de resolució de problemes, els darrers anys s'han utilitzat avenços en sistemes processadors més ràpids, conjuntament amb canvis en el disseny d'algorismes utilitzant conceptes matemàtics i físics. La utilització de tècniques probabilístiques ha esdevingut clau en el camp de l'algorísmia. Dues de les seves aplicacions més conegudes són l'anàlisi de la complexitat dels algorismes i el disseny d'algorismes més eficients. La idea bàsica és que un algorisme determinista pot tenir una complexitat bona per a la majoria de les possibles entrades, però trigar molt més per a un nombre petit d'entrades. Emprar mètodes probabilístics pot evitar que les entrades patològiques distorsionin la complexitat de l'algorisme.

Per exemple, imaginem que volem ordenar de manera creixent una cadena de n enters. L'algorisme de dividir i vèncer anomenat *quicksort* agafa el primer element, a, de la cadena, el compara amb la resta dels elements i el col·loca a la posició que li correspondria si la cadena estigués ordenada, posant els enters $\leq a$ a la subcadena esquerra de a i els enters $> a$ a la subcadena dreta. Després aplica l'algorisme de manera recursiva a les dues subcadenes desordenades. Si l'entrada està força desordenada, cada subcadena tindrà aproximadament la meitat de la grandària de la del pas previ, i la complexitat d'aquest algorisme és $O(n \log n)$. Però si l'entrada està ordenada, o quasi ordenada, una de les subcadenes de la crida recursiva serà buida i l'altra subcadena tindrà grandària igual que la grandària de la crida prèvia menys 1, és a dir, per aquesta mena d'entrades, *quicksort* tindrà complexitat $O(n^2)$. Notem que un *adversari* pot donar com a entrada a *quicksort* la cadena ja ordenada. Aleshores, per estudiar la complexitat de l'algorisme té sentit considerar una distribució uniforme de les $n!$ cadenes, escollir-ne aleatòriament una i estudiar la complexitat de *quicksort* sobre aquesta entrada. Fàcilment es pot demostrar que el temps esperat serà $O(n \log n)$, i amb *alta probabilitat* que sigui així. Per tant, per valors

[2] Els vaccins mRNA contra la Covid-19 són algorismes.

de n molt grans, la probabilitat d'ensopegar amb una entrada *costosa* (ordenada) tendeix a 0, i aquesta darrera part ja no és fàcil de demostrar.

A partir d'aquest resultat podem dissenyar el següent *algorisme aleatoritzat*, que és una senzilla modificació del *quicksort*: Donada com a entrada una cadena, a cada crida recursiva, en lloc d'agafar com a pivot a fixar l'element més a la dreta de la subcadena, agafem com a pivot un element aleatori de la subcadena. Sigui quina sigui l'entrada, el temps esperat d'aquest algorisme aleatoritzat és $O(n \log n)$. Aquest algorisme s'anomena *rand-quicksort*. Un *algorisme aleatoritzat* és un procediment que utilitza probabilitat per prendre decisions de com continuar el procés. L'exemple de *rand-quicksort* és un algorisme molt senzill (en general, els algorismes aleatoritzats poden ser força més complicats). Com veurem després, una part dels algorismes aleatoritzats poden donar algunes solucions incorrectes a costa de funcionar ràpid; en aquest cas, es tracta de dissenyar l'algorisme de manera que la probabilitat de donar un solució errònia sigui mínima. Aquest tipus d'algorismes s'anomenen Monte-Carlo.

El problema de la k-vèrtex-connectivitat d'un graf

Donat un graf connex i no dirigit $G = (V, A)$ amb $|V| = n$ i $|A| = m$, es diu que G és *k-vèrtex-connex* si s'han d'eliminar com a mínim k vèrtexs per desconnectar G. Per exemple, un cicle amb n vèrtexs és 2-vèrtex-connex. El mínim conjunt dels k vèrtexs que desconnecten G s'anomena el *tall mínim de vèrtexs*. Donat un graf G i una k, $n > k \geq 1$, el problema de la *k-vèrtex-connectivitat de G* consisteix a determinar si G és k-vèrtex-connex.

Considerem el problema relacionat de, *donat G,* **trobar** *el valor mínim $k > 0$ tal que eliminant k vèrtexs es desconnecta G*. Des del punt de vista de la complexitat, tots dos problemes són $\log n$-equivalents, en el sentit que si tenim un algorisme **AlgKvc** per resoldre el problema de la k-vèrtex connectivitat, aleshores fent una *cerca binària* sobre els $n - 1$ possibles valors de k, com que el nombre màxim de crides de la cerca binària és $O(\log n)$, aleshores per trobar el valor de k correcte s'ha de repetir **AlgKvc** com a màxim $O(\log n)$ cops.

Per tant, parlarem únicament del problema de determinar si G és k-vèrtex-connex. Un enunciat alternatiu d'aquest problema és comprovar si el tall mínim té un valor igual a k. Intuïtivament aquest enunciat indica la possibilitat d'utilitzar algorismes del tipus *maxflow-mincut* (vegeu per exemple, el capítol 5 a [4]). I, de fet, la majoria dels algorismes que s'han dissenyat per resoldre el problema estan basats en la següent estratègia bàsica: per cada parell de vèrtexs $\{u, v\}$ a G, es genera un digraf, i s'orienten les arestes a G perquè vagin de $u \to v$, amb totes les arestes amb capacitat 1, i aplicar l'algorisme de Dinic per trobar el flux màxim que dona el valor del tall mínim (vegeu la secció 5.2 a [4]). Aquest procés s'itera sobre els $\binom{n}{2}$ parells de vèrtexs a G. Com que l'algorisme de Dinic té una complexitat $O(m\sqrt{n})$, el temps total d'aplicar l'algorisme descrit per a resoldre el problema de determinar la k-vèrtex connectivitat de G és $O(mn^{5/2})$. Fonamentalment l'evolució dels algorismes pel que fa a temps de computació es va basar majoritàriament en diferents refinaments en la selecció de parells. El 1980 va aparèixer un algorisme determinista que resol el problema amb complexitat $(n^2+k^2)mk$ [5], i un algorisme aleatoritzat que crida Dinic un nombre esperat de $O(n \log n)$ cops, per tant la complexitat esperada és $O(mn^{3/2} \log n)$ [2].

El 1986, Linial, Lovász i Wigderson (d'ara endavant, LLW) van utilitzar tècniques algebraiques, totalment diferents de les que usen l'estudi del flux màxim, per dissenyar un algorisme aleatoritzat que millorava les fites de complexitat dels algorismes previs [11, 12]. L'algorisme de LLW està basat a donar una caracterització algebraica de les condicions necessàries i suficients per a que donats G i k, el graf G sigui *k-vèrtex-connex*.

Sigui $G = (V, A)$ un graf. Per a tot $v \in V$, definim el conjunt de veïns de v com $\mathcal{N}(v) = \{u \in V : (u, v) \in A\}$. Notem que si G és *k-vèrtex-connex*, $\forall u \in V$ es té que $|\mathcal{N}(v)| \geq k$. Així mateix, donats G i r, per a tot $v \in V$, definim $\mathcal{N}_r(v)$ com un subconjunt de $\mathcal{N}(v)$ amb $|\mathcal{N}_r(v)| = r$. Finalment, donats G i $X \subset V$ amb $|X| = k$, direm que G té una *X-immersió convexa* si existeix una funció $f : V \to \mathbb{R}^{k-1}$, tal que $\forall v \in V \backslash X$, tenim que $f(v)$ pertany a *l'envolupant convexa de $f(\mathcal{N}(v))$*. Intuïtivament

si, per exemple, tenim que $|X| = 4$, podem pensar en una X-immersió convexa com una funció f que porta tot els vèrtexs de $V \backslash X$ a dintre d'un tetraedre a $\mathbb{R}^3$ que té per vèrtexs els elements de $f(X)$. Per evitar degeneracions, definim que una X-immersió convexa de G és en *posició general*, si cap hiperplà conté les imatges de més de $k-1$ vèrtexs de V. Formalment, la immersió f ve donada per una matriu de rang k, i $f(X)$ forma una base de dimensió $k-1$ a $\mathbb{R}^{k-1}$.

El teorema de caracterització que tenim és el següent [11, 12]: *Donats G i k, G és k-vèrtex connex si i sols si per a tot $X \subset V$ amb $|X| = k$, G té una X-immersió convexa en posició general.* Intuïtivament, aquest teorema ens indica que, si $v \in V \backslash X$, per a separar v de X necessitem eliminar com a mínim k vèrtexs, fet que no és sorprenent si considerem el teorema de Menger (vegeu, per ex., el teorema 6.4 a [4]).

Una part important de la demostració d'aquest resultat és una construcció algorísmica de la X-immersió convexa en posició general. És a dir, volem construir $f(V \backslash X)$ dintre d'un $k-1$-símplex unitat, on els vèrtexs del símplex siguin $f(X)$. La tècnica bàsica és una generalització del *mètode de les molles*, implícitament utilitzat a [25] per dibuixar de manera planar grafs 3-connexos, que admetin una representació planar.

Per a tot $(u, v) \in E$, tal que u i v no pertanyen a X, considerem que cada aresta $(f(u), f(v))$ és una molla. Les arestes entre els vèrtexs del símplex $(f(X))$ són les arestes rígides del $k-1$ símplex. Recordeu que per cada $v \in V$, $f(v)$ té $k-1$ components.La llei de Hooke ens diu que la força F per modificar una molla una distancia d és $F = cx$, on c és el *coeficient d'elasticitat* de la molla i x és la posició. En el nostre cas, cada molla corresponent a (u, v) tindrà el seu propi coeficient d'elasticitat $c_{uv} > 0$. Per la llei de Hooke, el sistema tendeix a un equilibri de mínima energia dintre del $k-1$ símplex, aquest equilibri és la X-immersió. El que no és sempre cert, és que aquesta immersió estigui en posició general (vegeu, per ex., la secció 3.3 a [14]). LLW demostren que, si escollim cada valor $c_{u,v} > 0$ de manera aleatòria, aleshores amb probabilitat tendint a 1, el sistema evoluciona cap a l'equilibri i finalitzarà en posició general.

Sigui $f_{\mathbf{c}}$ la immersió final quan el sistema s'ha estabilitzat en equilibri, on $\mathbf{c}$ és el vector de coeficients d'elasticitat escollits. Per cada $v \in V \backslash X$ la posició final $f_{\mathbf{c}}(v)$, ve determinada per un sistema homogeni de $n-k$ equacions lineals, on cada equació correspon a un vèrtex $v \in V \backslash X$, i l'equació és: $\sum_{(u,v) \in E} c_{uv}(f(u) - f(v)) = 0$.

Per al vector aleatori $\mathbf{c}$ de les molles, aquest sistema d'equacions té una solució única. Per trobar la $f_{\mathbf{c}}$ de l'equilibri, hem de resoldre el sistema. Però cada $f(v)$ té $k-1$ coordenades i, per tant, el sistema té $(n-k) \times (k-1)$ variables. Per trobar la solució s'ha d'invertir una matriu d'aquestes dimensions, i això fa que la solució no sigui eficient. LLW van fer una hàbil manipulació, i van reduir el càlcul de $f_{\mathbf{c}}(v)$ a una multiplicació de matrius molt més assequibles [12].

Com que les equacions estan definides sobre $\mathbb{R}^{k-1}$, un problema podia ser la precisió de les computacions amb coma flotant. Per solucionar aquest tema, LLW en lloc de resoldre les equacions sobre $\mathbb{R}$, ho fan sobre un cos finit, introduint el concepte de *X-immersió modular i aleatòria.* Aquest canvi fa perdre el significat geomètric de la immersió, però en manté l'estructura algebraica. La X-immersió modular i aleatòria funciona de la manera següent: donats G amb $|V| = n, |A| = m$ i $X \subset V$, escollim aleatòriament un nombre primer $p < n^5$, després escollim aleatòriament el vector de coeficients d'elasticitat $\mathbf{c} \in (\mathbb{Z}/p\mathbb{Z})^m$ i amb aquests valors resolem el sistema d'equacions sobre $\mathbb{Z}/p\mathbb{Z}$ per obtenir $f_{\mathbf{c}}$. Per escollir p existeixen algorismes aleatoris força ràpids basats en Miller-Rabin (vegeu, per ex., 10.8 a [16]).

L'algorisme **AlgKvc** de LLW, que comprova si G és *k-vèrtex-connex*, està basat en la següent conseqüència del teorema de caracterització: Donats G i $k > 0$, G és k-vèrtex-connex si i sols si hi ha com a mínim k vèrtexs diferents $u \in V$, tals que G té una $\mathcal{N}_k(u)$-immersió convexa f en què $\forall v \in V \backslash \mathcal{N}_k(u)$, el rang de $f(\mathcal{N}_k(v))$ és k.

L'algorisme **AlgKvc** utilitza una subrutina $\mathbf{Test}_k(G, u)$ que, donats un G i un $u \in V$, computa una $\mathcal{N}_k(v)$-immersió aleatòria en

posició general per als $v \in \mathcal{N}(u)$. Recordeu que tota l'estona $m = |A|$.

Test$_k(G, u)$
Tria aleatòriament $p < n^5$ i $\mathbf{c} \in (\mathbb{Z}/p\mathbb{Z})^m$, p primer
Computa la $\mathcal{N}_k(u)$-immersió modular $f = f_{\mathbf{c}}$
$\forall v \in V \backslash \mathcal{N}_k(u)$ comprova si $\text{rang}(f(\mathcal{N}_k(v))) = k$
  si $\forall v$ és compleix la condició, retornar *cert*
  altrament retornar *fals*

Finalment, l'algorisme MonteCarlo per determinar si G és *k-vèrtex-connex*,

AlgKvc(G, k)
Repetir $(n \log n)/(n-k)$ passos
  escollir aleatòriament $u \in V$
  si **Test**$_k(G, u)$ torna *fals*,
    escriure G no és *k-vèrtex-connex*
Si per tots els u triats, **Test**$_k(G, u)$ torna *cert*,
  escriure G és *k-vèrtex-connex*
altrament escriure G NO és *k-vèrtex-connex*

LLW demostren que **AlgKvc**(G, k) retorna la resposta correcta amb probabilitat $> 1 - \frac{1}{n}$. La complexitat de l'algorisme és $O(n^{2.37} + nk^{2.37})$, on l'exponent 2.37 ve de la multiplicació de matrius. Als anys 80, **AlgKvc**(G, k) va ser l'algorisme més eficient per valors de $k \geq \sqrt{n}$.

Respecte a l'evolució posterior de la complexitat del problema, l'any 1991 Cheriyan i Reif van estendre el resultat de LLW a digrafs, utilitzant les mateixes tècniques i amb la mateixa complexitat [3]. El 2000, Henzinger *et al.* [7] tornant a utilitzar tècniques de fluxos van produir un algorisme aleatoritzat amb una millora de la complexitat a $O(kn^2)$. El 2019 Nanogkai *et al.* van dissenyar un algorisme aleatoritzat amb una complexitat $O(m+k^{7/3}n^{4/3})$ per a valors de k fins a $\sqrt{n}$ [17]. El que és remarcable en aquest darrer treball és que és el primer algorisme que, per $m = O(n)$, trenca la barrera de complexitat $O(n^2)$, i la rebaixa a $O(n^{4/3})$. Als anys 70, es va conjecturar que el problema es pot resoldre en temps $\Theta(m)$. Molt recentment, Li *et al.* [10] han donat una solució al problema en temps $O(m^\alpha)$, on $\alpha \geq 1$ és l'exponent de l'algorisme més eficient que es coneix per computar el màxim flux, actualment $O(m^\alpha)$, amb $\alpha = \frac{4}{3} + o(1)$ [6]. L'algorisme de Li *et al.* queda molt a prop de resoldre la conjectura oberta des de l'any 70.

L'algorisme LLL

Jordi Guàrdia, Departament Matemàtiques, UPC

L'algorisme LLL va ser dissenyat el 1981 per tres "L" insignes de les matemàtiques: els germans Hendrik i Arjen Lenstra i Lazlo Lovász. Tot i que el seu origen remot és un problema de programació lineal i que la primera aplicació que en van fer els autors va ser un problema aritmètic, d'aleshores ençà ha anat revolucionant molts i diversos camps de les matemàtiques. Si algun moment heu de resoldre un problema amb xarxes, molt probablement l'algorisme LLL us serà molt útil.

Descrit molt breument, l'algorisme ens permet trobar bases *bones* de xarxes. Una *xarxa*[3] en $\mathbb{R}^n$ és el conjunt format per totes les combinacions lineals enteres de n vectors linealment independents. Un exemple molt senzill seria la xarxa generada pels vectors $(1, 0), (0, 1)$ en $\mathbb{N}^2$, que està formada pels punts de $\mathbb{R}^2$ amb coordenades enteres. Naturalment, una xarxa donada admet moltes bases i algunes són molt més convenients que d'altres. Per exemple, la xarxa que acabem de descriure també admet la base $(5341, 393), (4213, 310)$, però amb aquesta base serà bastant més difícil treballar. Per decidir quines bases són millors que d'altres solem tenir en compte la mida dels seus vectors. Ara bé, aquesta "mida" pot ser el seu mòdul en la mètrica euclidiana habitual de $\mathbb{R}^n$ o, més en general, la mida respecte a una altra mètrica qualsevol, donada per una forma quadràtica definida positiva.

Les xarxes són una eina molt potent per a resoldre problemes pràctics d'àmbits molt variats. És molt important, doncs, tenir un bon algorisme per trobar una base "petita" d'una xarxa donada i això és el que fa justament l'algorisme LLL. En particular, l'algorisme calcula una *bona aproximació* del vector més

[3] En altres àmbits, són el que s'anomenen reticles, o *lattices* de l'anglès