Butlletí de la Societat Catalana de Matemàtiques
Vol. 15, núm. 2, 2000. Pàg. 27–35

Aspectes lògics de la complexitat computacional*

Jörg Flum

1 Introducció

Els resultats obtinguts en els darrers vint-i-cinc anys ens mostren que moltes vegades la complexitat computacional d'un problema està relacionada amb la seva complexitat descriptiva, és a dir, amb la complexitat d'un llenguatge formal que permeti descriure el problema. En aquest article, considerant alguns exemples concrets, volem donar una visió de la naturalesa d'aquests resultats (el lector interessat pot consultar [3] per a un estudi sistemàtic). Al mateix temps, volem documentar la importància per a la complexitat computacional de conceptes de la teoria de grafs, que recentment han estat introduïts per Robertson i Seimour en la seva profunda anàlisi de l'estructura dels grafs.

2 Grafs i arbres

Un graf $\mathcal{G} = (V, A, I)$ consisteix en un conjunt finit V de *vèrtexs* o *punts*, un conjunt A d'*arestes* i una relació I, $I \subseteq A \times V$, d'*incidència*. Per cada aresta $a \in A$ existeixen exactament dos vèrtexs $u, v \in V$ que incideixen amb a, això és, amb $(a, u) \in I$ i $(a, v) \in I$. I per cada dos vèrtexs $u, v \in V$ amb $u \neq v$ hi ha, com a màxim, una aresta $a \in A$ que incideix amb u i v. Llavors també denotarem l'aresta a per $\{u, v\}$.

Moltes vegades il·lustrarem un graf amb una figura: els punts correspondran als vèrtexs i les línies, a les arestes. Por exemple, la figura 1 representa un graf que consisteix en quatre vèrtexs i arestes entre cada dos d'aquests vèrtexs.

Considerem el graf que té per vèrtexs les places i les cruïlles d'una ciutat i per arestes els corresponents trams dels carrers. Suposem que la companyia telefònica està disposada a instal·lar k cabines telefòniques. A fi que siguin fàcilment accessibles a tots els habitants de la ciutat, la companyia buscarà un «conjunt dominant» de cardinalitat k. Donat un graf, diem que un conjunt D de vèrtexs és un *conjunt dominant* si, per a tot vèrtex $v \notin D$, existeix $u \in D$ tal que $\{u, v\}$ és una aresta. Evi-

*Traducció de Frederic Utzet.

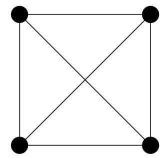

FIGURA 1: Un graf.

dentment, si D és un conjunt dominant, llavors cada conjunt E amb $D \subseteq E$ també serà dominant. O sigui que en el cas de la companyia telefònica hauríem de resoldre un problema del tipus

DOM$_k$	*Input*:	Un graf $\mathcal{G}$.
	Pregunta:	Existeix en $\mathcal{G}$ un conjunt dominant de cardinalitat k?

Busquem un algorisme ràpid per a resoldre aquest problema: direm d'un algorisme que és *lineal*, si el nombre $n(\mathcal{I})$ de passos que ha d'efectuar l'algorisme per a resoldre'l és proporcional a la mida $||\mathcal{I}||$ de l'input $\mathcal{I}$, $n(\mathcal{I}) = O(||\mathcal{I}||)$. La mida $||\mathcal{G}||$ d'un graf $\mathcal{G} = (V, A, I)$ es podria definir com $|V| + |A| + |I|$ (on $|X|$ és el nombre d'elements de X); llavors $O(||\mathcal{G})||) = O(|V| + |A|)$.

Direm d'un algorisme que és *polinomial* si per a algun $s \geq 1$ tenim que $n(\mathcal{I}) = O(||\mathcal{I}||^s)$, això és, el nombre de passos està afitat per algun polinomi. Evidentment, un algorisme lineal és també polinomial. Es diu que un problema és *intractable* si no existeix un algorisme polinomial que el resolgui.

L'algorisme obvi per a resoldre DOM$_k$ consisteix a comprovar, per a cada conjunt de vèrtexs de $\mathcal{G}$ de cardinalitat k, si es tracta d'un conjunt dominant. En el pitjor cas el nombre de passos és $\binom{|V|}{k}$, o sigui que no es tracta d'un algorisme lineal. La situació canvia radicalment si només considerem grafs que siguin arbres. Un *arbre* és un graf connex sense cicles. Un graf és *connex* si entre dos vèrtexs u i v diferents existeix un *camí* que els uneix, això és, existeixen $n \geq 1$ vèrtexs $w_0, \ldots, w_n$ tals que

$$u = w_0,\ v = w_n \ \text{ i } \ \{w_i, w_{i+1}\} \in A \ \text{ per a } \ i = 0, \ldots, n-1.$$

Un *cicle* és un camí $w_0, \ldots, w_n$ de punts distints dos a dos amb $n \geq 2$ i $\{w_n, w_0\} \in A$.

Sigui $\mathcal{T} = (T, A, I)$ un arbre (en el que segueix denotarem un arbre amb la lletra $\mathcal{T}$ i el seu conjunt de vèrtexs per T). Escollim un element qualsevol $r \in T$ i el declarem *arrel* de l'arbre. Llavors obtenim l'arbre de la figura 2 com a representació de $\mathcal{T}$: el nivell 0 només conté l'arrel r, i el nivell 1 tots els elements u tals que $\{r, u\}$ és una aresta. Si v és un element del nivell $n+1$ amb «antecessor» u en el nivell n, llavors els «successors» de v al nivell $n+2$ són els elements del conjunt $\{w \mid w \neq u \text{ i } \{v, w\} \in A\}$. Les *fulles* són els elements sense successors. Si $u \in T$ és un vèrtex, denotarem per $\mathcal{T}_u$ l'arbre que té per vèrtexs u, llurs successors, els successors de llurs successors...

És fàcil obtenir un conjunt dominant de cardinalitat mínima per a un «arbre amb arrel»: partint de les fulles, assignem a cada vèrtex u de l'arbre un parell (e, f)

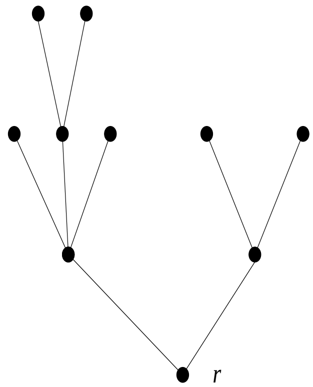

FIGURA 2: Un arbre.

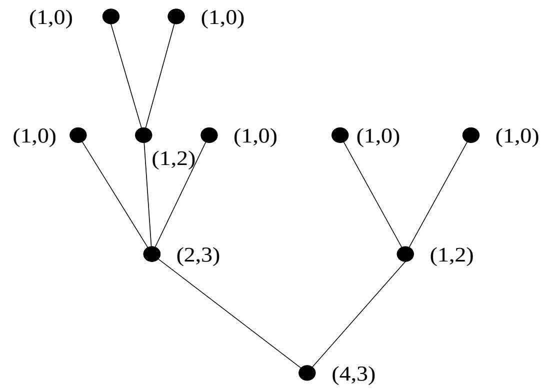

FIGURA 3: L'algorisme.

de nombres naturals: e és el mínim de les cardinalitats de conjunts dominants de $\mathcal{T}_u$ que continguin u, i f el mínim de les cardinalitats de conjunts dominants de $\mathcal{T}_u$ que no continguin u. Per definició, f serà igual a 0 per a les fulles. Assignem, per tant, $(1, 0)$ a les fulles. Fàcilment es verifica que per a u amb successors $u_1, \dots, u_n$ tenim que $e = 1 + \sum_{i=1}^{n} \min\{e_i, f_i\}$ i $f = \sum_{i=1}^{n} e_i$, si (e_i, f_i) és el parell assignat a u_i. Per a l'arbre abans esmentat obtenim els valors indicats a la figura 3. En el cas general, l'arbre tindrà un conjunt dominant de cardinalitat k si un dels valors assignats a l'arrel és $\leq k$ (i l'arbre té almenys k elements). O sigui, per a DOM_k tenim un algorisme que és lineal per a arbres.

Vegem dos problemes més per a grafs: un graf és 3-*colorable* si, utilitzant tres colors, podem colorar els vèrtexs del graf de tal forma que dos vèrtexs u i v adjacents (això és, on $\{u, v\}$ és una aresta) siguin de color distint.

El graf $\mathcal{K}_n$ és un *clique* (o *graf complet*) de n elements si consisteix en n punts i arestes entre cada dos d'aquests punts. Evidentment, $\mathcal{K}_3$ és 3-colorable, però no ho és cap $\mathcal{K}_n$ per a $n > 3$. Considerem el problema

3-COL	*Input*:	Un graf $\mathcal{G}$.
	Pregunta:	És $\mathcal{G}$ 3-colorable?

No se sap si existeix un algorisme polinomial per a 3-COL (més ben dit, 3-COL és intractable, si PTIME $\neq$ NPTIME). El mateix passa amb el problema

HAM	*Input*:	Un graf $\mathcal{G}$.
	Pregunta:	Conté $\mathcal{G}$ un cicle hamiltonià?

Un cicle en $\mathcal{G}$ és *hamiltonià* si passa per tots els vèrtexs de $\mathcal{G}$.

Ambdós problemes, 3-COL i HAM, són trivials per a arbres. Cap arbre té un cicle hamiltonià (ja que per definició no conté cap cicle) i cada arbre és 3-colorable (dos colors són suficients: ni ha prou alternant els colors de nivell en nivell).

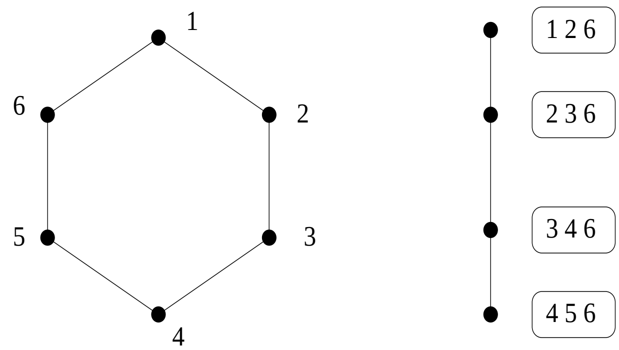

FIGURA 4: Un cicle i una descomposició arbòria.

3 Grafs d'amplada arbòria fitada

El que hem vist per a tres problemes (DOM_k, 3-COL i HAM) és un fenòmen força generalitzat: problemes algorísmicament difícils en la classe de tots els grafs són fàcils de resoldre per a arbres. Fins a quin punt podem transferir aquestes propietats d'arbres a grafs més generals, grafs que si bé no són arbres «tenen una descomposició al llarg d'arbres»? La noció adient de descomposició fou introduïda per Robertson i Seimour [8] en els seus estudis estructurals de grafs. En aquesta secció ens familiaritzarem amb aquesta noció i en la secció següent veurem com podem generalitzar els resultats obtinguts per a arbres.

1 DEFINICIÓ *Sigui* $\mathcal{G} = (V, A, I)$ *un graf. Una* descomposició arbòria *és un parell* $(\mathcal{T}, (V_t)_{t\in T})$ *on* $\mathcal{T}$ *és un arbre i cada* V_t *és un conjunt de vèrtexs de* $\mathcal{G}$ *en què*

- *per a tota aresta* $\{u, v\}$ *de* $\mathcal{G}$ *existeix un* $t \in T$ *tal que* $u, v \in V_t$*;*
- *per a tot vèrtex* u *de* $\mathcal{G}$ *el conjunt* $\{t \in T \mid u \in V_t\}$ *és un subconjunt no buit i connex de vèrtexs de* $\mathcal{T}$*.*

Els conjunts V_t *són els* blocs *de la descomposició. L'*amplada *de la descomposició* $(\mathcal{T}, (V_t)_{t\in T})$ *és* $\max\{|V_t| - 1 \mid t \in T\}$*. L'*amplada arbòria $\mathrm{aa}(\mathcal{G})$ *de* $\mathcal{G}$ *és el mínim de les amplades de les descomposicions arbòries de* $\mathcal{G}$*.*

L'amplada arbòria d'un graf ens indica en certa manera quant diferent és d'un arbre. Cada arbre (amb al menys dos elements) té amplada 1; en efecte, l'arbre $\mathcal{T}$ de la figura 2 té la descomposició arbòria $(\mathcal{T}, (V_t)_{t\in T})$ on $V_r = \{r\}$ i $V_t = \{t, \text{antecessor de } t\}$ per a $t \neq r$. Suposem que el graf $\mathcal{G}$ és un cicle, amb $\{u_0, \dots, u_n\}$ el seu conjunt de vèrtexs i $\{\{u_i, u_{i+1}\} \mid i = 0, \dots, n-1\} \cup \{\{u_n, u_0\}\}$ el seu conjunt d'arestes. Llavors obtenim la següent descomposició arbòria $(\mathcal{T}, (V_t)_{t\in T})$ d'amplada dos (vegeu la figura 4): $T := \{1, \dots, n\}$, i en $\mathcal{T}$ solament hi ha arestes entre i i $i+1$ per a $i = 1, \dots, n-1$, i sigui $V_i := \{u_n, u_{i-1}, u_i\}$ per a $i = 1, \dots, n$. En efecte, $\mathrm{aa}(\mathcal{G}) = 2$ per a cada cicle.

Cada graf $\mathcal{G} = (V, A, I)$ té una descomposició arbòria, l'arbre de la qual només conté un vèrtex (amb bloc V). Per tant, veiem que $\mathrm{aa}(\mathcal{G}) \leq |V| - 1$. Per a obtenir fites inferiors per a l'amplada arbòria, la caracterització de l'amplada arbòria mitjançant el joc que presentarem a continuació és sovint molt útil.

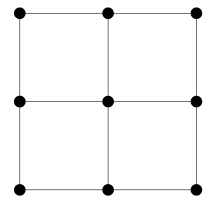

FIGURA 5: La reixa $\mathcal{R}_3$.

Donat un graf $\mathcal{G}$ i un nombre natural $k \geq 1$, considerem un joc $J_k(\mathcal{G})$ entre k policies i un lladre. Després de cada jugada, cadascun dels policies i el lladre es troben en un vèrtex del graf. A la primera jugada d'una partida, cada policia elegeix la seva situació en el graf i a continuació el lladre la seva. En cada jugada posterior, un dels policies puja a un helicòpter i es desplaça a un nou vèrtex. Abans que l'helicòpter aterri, el lladre (utilitzant també la informació d'on aterrarà l'helicòpter) pot desplaçar-se al llarg d'un camí de $\mathcal{G}$ cap a un nou vèrtex, però no pot passar per vèrtexs ocupats per policies. Els policies *guanyen* la partida si finalment s'obté una situació en la qual el lladre i (almenys) un policia ocupin el mateix vèrtex. Si els policies poden jugar cada partida de tal forma que la guanyen, direm que *la policia guanya* $J_k(\mathcal{G})$.

Per exemple, per a cada arbre $\mathcal{T}$ la policia guanya $J_2(\mathcal{T})$: en efecte, en la primera jugada un policia es situa en «l'arrel» de l'arbre; en les jugades següents els policies es comporten de forma que finalment el lladre ha de situar-se en una fulla tal que el seu antecessor l'ocuparà el policia en l'helicòpter. A la jugada següent, els policies guanyaran la partida.

Evidentment, per al clique $\mathcal{K}_n$ de n elements, la policia guanya $J_n(\mathcal{K}_n)$ però no guanya $J_{n-1}(\mathcal{K}_n)$. Considerem la reixa $\mathcal{R}_n$ (vegeu la figura 5). Es tracta del graf amb conjunt de vèrtexs $\{(i,j) \mid i,j = 1,\ldots,n\}$; hi ha una aresta entre (i,j) i (i',j') si i només si $|i-i'|+|j-j'| = 1$. La policia no guanya $J_{n-1}(\mathcal{R}_n)$: qualssevol que siguin les posicions dels policies després d'una jugada, sempre hi haurà una barra horitzontal, diguem la i-èsima, i una barra vertical, diguem la j-èsima, sense policies. El lladre llavors es situarà en la posició (i,j). El lector pot verificar que la policia guanya $J_n(\mathcal{R}_n)$.

A conseqüència del teorema següent, les consideracions precedents ens mostren que $\mathrm{aa}(\mathcal{K}_n) = n-1$ i que $\mathrm{aa}(\mathcal{R}_n) = n-1$.

2 TEOREMA ([9]) *Per a* $n \geq 2$ *i tot graf* $\mathcal{G}$ *tenim que* $\mathrm{aa}(\mathcal{G}) \leq n-1$ *si i només si la policia guanya* $J_n(\mathcal{G})$.

4 El marc lògic

Per a $n \geq 1$, denotem per C_n la classe de grafs d'amplada arbòria $\leq n$. En particular, C_1 conté tots els arbres. En la primera part hem vist que els problemes DOM_k, 3-COL i HAM eren resolubles en temps lineal per a arbres. Veurem que per a cada n també existeixen algorismes lineals que resolen aquests problemes per a grafs de C_n. Per quina classe de problemes algorísmics es compleix aquest resultat? Els llenguatges formals introduïts en lògica matemàtica formen el marc per respondre aquesta pregunta.

Considerem el *llenguatge monàdic de segon ordre* que denotem per MSO. Les seves fórmules o sentències s'obtenen a partir dels signes següents:

$u, v, w, \ldots$	les variables per a vèrtexs;
$a, b, c, \ldots$	les variables per a arestes;
$X, Y, Z, \ldots$	les variables per a conjunts de vèrtexs;
$E, F, G, \ldots$	les variables per a conjunts d'arestes;
$\neg, \wedge, \vee, \rightarrow, \leftrightarrow$	els connectius;
$\forall, \exists$	els quantificadors;
$=, \in$	el signe d'igualtat i el de pertinença;
I	el signe per a la relació d'incidència;
$), (, ,.$	els parèntesis i la coma.

Les *sentències* (o *fórmules*) del *llenguatge monàdic de segon ordre* són certes tires d'aquests signes: s'obtenen juxtaposant de forma raonable mitjançant els connectius i quantificadors tires de la forma $u \in X$ («el vèrtex u pertany a X»), $a \in E$ («l'aresta a pertany a E»), $(a, u) \in I$ («la aresta a incideix amb el vèrtex u»), $u = v$ i $a = b$. En lloc de donar una definició exacta, vegem alguns exemples concrets (vegeu [4] per als detalls). La 3-colorabilitat es pot expressar amb la següent sentència $\varphi_{3-\mathrm{COL}}$ de MSO (X, Y, Z representen els conjunts de vèrtexs dels tres colors distints):

$$\begin{aligned}
&\exists X \exists Y \exists Z (\forall u (u \in X \vee u \in Y \vee u \in Z) \wedge \forall a \forall u \forall v \\
&(((a,u) \in I \wedge (a,v) \in I \wedge \neg u = v) \rightarrow (\neg(u \in X \wedge v \in X) \\
&\wedge \neg(u \in Y \wedge v \in Y) \wedge \neg(u \in Z \wedge v \in Z)))).
\end{aligned}$$

Per simplificar la sentència no hem demanat que X, Y, Z siguin disjunts, però fàcilment es comprova que llavors $X, Y \setminus X$ i $Z \setminus (X \cup Y)$ és una coloració coherent amb 3-COL.

Si una sentència φ de MSO és vàlida en un graf $\mathcal{G}$, direm que $\mathcal{G}$ és un *model* de φ. Aleshores tenim que $\mathcal{G}$ és un model de $\varphi_{3-\mathrm{COL}}$ si i només si $\mathcal{G}$ és 3-colorable. Anàlogament, $\mathcal{G}$ conté un conjunt dominant de cardinalitat k si i només si $\mathcal{G}$ és un model de la sentència φ_k de MSO, on φ_k és:

$$\begin{aligned}
&\exists u_1 \ldots \exists u_k \forall v ((\neg v = u_1 \wedge \ldots \wedge \neg v = u_k) \rightarrow \\
&\exists a ((a,v) \in I \wedge ((a,u_1) \in I \vee \ldots \vee (a,u_k) \in I))).
\end{aligned}$$

Finalment, és senzill comprovar que $\mathcal{G}$ conté un cicle hamiltonià si i només si existeix un conjunt E d'arestes tal que cada vèrtex de $\mathcal{G}$ és incident amb exactament dues arestes d'E i el conjunt de tots els vèrtexs de $\mathcal{G}$ és l'únic conjunt no buit de vèrtexs amb la propietat de contenir els dos vèrtexs d'una aresta d'E si almenys en conté un. O sigui que $\mathcal{G}$ conté un cicle hamiltonià si i només si $\mathcal{G}$ és un model de φ_{HAM} on φ_{HAM} és la sentència de MSO següent:

$$\begin{aligned}
&\exists E (\forall u \exists a \exists b (\neg a = b \wedge a \in E \wedge b \in E \wedge (a,u) \in I \wedge (b,u) \in I \\
&\wedge \forall c ((c \in E \wedge (c,u) \in I) \rightarrow (c = a \vee c = b))) \wedge \\
&\forall X ((\exists u u \in X \wedge \forall v \forall w \forall a ((v \in X \wedge a \in E \wedge (a,v) \in I \wedge (a,w) \in I) \rightarrow w \in X)) \\
&\rightarrow \forall u u \in X)).
\end{aligned}$$

Amb el teorema següent veurem en especial que, donat $n \geq 1$, els problemes DOM_k, 3-COL i HAM són resolubles en temps lineal per a grafs de C_n.

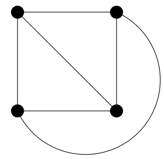

FIGURA 6: Un graf planar.

3 TEOREMA ([2]) *Donats $n \geq 1$ i una sentència φ de MSO, existeix un algorisme lineal que per cada graf $\mathcal{G}$ d'amplada arbòria $\leq n$ decideix si $\mathcal{G}$ és un model de φ.*

L'algorisme corresponent construeix primer una descomposició arbòria del graf donat. Aquí s'utilitza el següent resultat sobre descomposicions arbòries de Bodlaender:

4 TEOREMA ([1]) *Per a tot $n \geq 1$ existeix un algorisme lineal que per cada graf $\mathcal{G}$ decideix si* $\mathrm{aa}(\mathcal{G}) \leq n$*, i en cas positiu ens dóna una descomposició arbòria de $\mathcal{G}$ d'amplada $\leq n$.*

Considerem la classe P de grafs planars. Un graf és *planar* si podem representar els seus vèrtexs per punts del pla i les seves arestes per corbes en el pla de tal forma que dues arestes coincideixin com a màxim en llurs punts terminals. Per exemple, la figura 6 ens mostra que el clique $\mathcal{K}_4$ és un graf planar; $\mathcal{K}_5$ no és planar. La classe P de grafs planars no està continguda en cap C_n. En efecte, les reixes $\mathcal{R}_n$ són totes planars però havíem vist que $\mathrm{aa}(\mathcal{R}_n) = n - 1$ i per tant $\mathcal{R}_{n+2} \notin C_n$.

És sabut que per a grafs planars el problema DOM_k és resoluble en temps lineal, mentre que 3-COL és NPTIME-complet i per tant no lineal (si PTIME $\neq$ NPTIME). Analitzem aquesta diferència des del punt de vista lògic. Diem que una fórmula de MSO és de primer ordre si no conté variables per a conjunts (ni per a conjunts de vèrtexs ni per a conjunts d'arestes). La fórmula φ_k que hem introduït abans per a DOM_k és de primer ordre, però no ho són ni $\varphi_{3-\mathrm{COL}}$ ni φ_{HAM}.

5 TEOREMA ([6]) *Sigui φ una fórmula de primer ordre. Existeix un algorisme lineal que per cada graf planar $\mathcal{G}$ decideix si $\mathcal{G}$ és un model de φ.*

El problema 3-COL ens mostra que aquest resultat no és vàlid per a una fórmula de MSO qualsevol. El teorema és un cas especial d'un teorema molt més general demostrat per Frick i Grohe [6] per a la classe de grafs d'«amplada arbòria localment fitada».

5 Immersions i homomorfismes

Un graf $\mathcal{H} = (W, B, J)$ és un *subgraf* de $\mathcal{G} = (V, A, I)$, si $W \subseteq V$, $B \subseteq A$ i $J \subseteq I$. Els grafs $\mathcal{H}_1$ i $\mathcal{H}_2$ són subgrafs de $\mathcal{G}_0$ (vegeu la figura 7).

$\mathcal{H}$ és *immersible* en $\mathcal{G}$, si $\mathcal{H}$ és isomorf a un subgraf de $\mathcal{G}$. Donat $\mathcal{H}$, és fàcil escriure una sentència $\varphi_{\mathcal{H}}$ de primer ordre tal que per a qualsevol graf $\mathcal{G}$ es compleixi

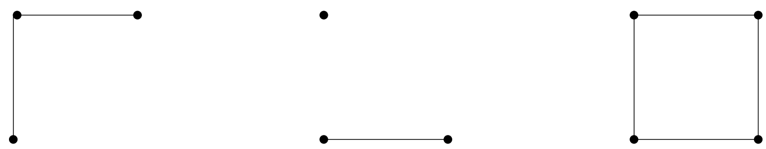

FIGURA 7: Els grafs $\mathcal{H}_1$, $\mathcal{H}_2$ i $\mathcal{G}_0$.

$\mathcal{G} \models \varphi_{\mathcal{H}}$ si i nomes si $\mathcal{H}$ és immersible en $\mathcal{G}$. En efecte, per a $\mathcal{H}_1$ prendríem

$$\begin{aligned}\varphi_{\mathcal{H}_1} \quad := \quad & \exists u \exists v \exists w \exists a \exists b (\neg u = v \wedge \neg u = w \wedge \neg v = w \\ & \wedge (a, u) \in I \wedge (a, v) \in I \wedge (b, v) \in I \wedge (b, w) \in I).\end{aligned}$$

Del teorema 3 es dedueix:

> Per a tot $n \geq 1$ i qualsevol graf $\mathcal{H}$ existeix un algorisme lineal que per a tot graf $\mathcal{G}$ d'amplada arbòria $\leq n$ decideix si $\mathcal{H}$ és immersible en $\mathcal{G}$.

L'algorisme canònic per a decidir si $\mathcal{H} = (W, B, J)$ és immersible en $\mathcal{G} = (V, A, I)$ comprovaria per cada funció de W a V si és una immersió de $\mathcal{H}$ en $\mathcal{G}$. En el pitjor dels casos el nombre de passos seria $\geq |V|^{|W|}$. O sigui, si $\mathcal{H}$ és un cicle amb 1000 vèrtexs tindríem $|V|^{1000}$ passos. El resultat següent ens diu que per a cada cicle podem resoldre el problema en $O(|V|^3)$ passos (ja que havíem vist que els cicles tenen amplada arbòria dos).

6 TEOREMA ([7]) *Sigui $\mathcal{H} = (W, B, J)$ un graf d'amplada arbòria $\leq s$. Existeix un algorisme que per a qualsevol graf $\mathcal{G} = (V, A, I)$ decideix si $\mathcal{H}$ és immersible en $\mathcal{G}$ en $O(|V|^{s+1})$ passos.*

Resulta que obtenim la mateixa fita per al nombre de passos si volem saber si existeix un homomorfisme de $\mathcal{H} = (W, B, J)$ en $\mathcal{G} = (V, A, I)$. Una funció $f : W \to V$ és un *homomorfisme* de $\mathcal{H}$ en $\mathcal{G}$ si $v, w \in W$ i $\{v, w\} \in B$ impliquen $\{f(v), f(w)\} \in A$. Fixem-nos que existeix una sentència $\psi_{\mathcal{H}}$ de primer ordre tal que per a qualsevol graf $\mathcal{G}$ tenim que $\mathcal{G} \models \psi_{\mathcal{H}}$ si i només si existeix un homomorfisme de $\mathcal{H}$ en $\mathcal{G}$ (per a $\mathcal{H} = \mathcal{H}_1$ només cal ometre en $\varphi_{\mathcal{H}_1}$ les «desigualtats»).

El marc lògic ens permet generalitzar aquests resultats sobre immersions (vegeu el teorema 6) i homomorfismes; sense entrar en detalls tècnics, volem mencionar que, per poder fer això, a cada sentència φ de primer ordre se li associa un graf $\mathcal{H}_\varphi$ (essencialment els vèrtexs de $\mathcal{H}_\varphi$ són les variables de la fórmula φ, i existeix una aresta entre dues variables si ambdues apareixen en una «part atòmica que no sigui una desigualtat»). Per definició, l'amplada arbòria de φ és la del graf $\mathcal{H}_\varphi$. Llavors la generalització esmentada seria:

7 TEOREMA ([5]) *Sigui φ una sentència «existencial i prenexa» de primer ordre d'amplada arbòria s. Llavors existeix un algorisme que per cada graf $\mathcal{G} = (V, A, I)$ decideix si $\mathcal{G}$ és un model de φ en $O(|V|^{s+1})$ passos.*

Referències

[1] BODLAENDER, H. L. «A linear-time algorithm for finding tree-decompositions of small treewidth», *SIAM J. on Comp.*, 25(1996): 1305–1317.

[2] COURCELLE, B. «Graph rewriting: An algebraic and logic approach», J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volum 2, 194–242. Elsevier Science Publishers, 1990.

[3] EBBINGHAUS, H.-D., FLUM, J. *Finite Model Theory*. Springer-Verlag, 1995.

[4] EBBINGHAUS, H.-D., FLUM, J., THOMAS, W. *Mathematical Logic*. Springer-Verlag, 1984.

[5] FLUM, J., GROHE, M. «Fixed-parameter tractabiliti and logic». Preprint, 1999.

[6] FRICK, M., GROHE, M. «Deciding first-order properties of locally tree-decomposable graphs», *Proceed. of the 26th Int. Coll. on Automata, Languages and Programming, Lect. Notes in Comp. Sci.* Springer-Verlag, 1999.

[7] PLEHN, J., VOIGT, B. «Finding minimally weighted subgraphs». In R. Möhring, editor, *Graph-Theoretic Concepts in Computer Science, WG '90*, *Lect. Notes in Comp. Sci.* 484, 18–29. Springer-Verlag, 1990.

[8] ROBERTSON, N., SEIMOUR, P. D. «Graph minors II. Algorithmic aspects of tree-width», *J. of Algorithms*, 7(1986): 309–322.

[9] SEIMOUR, P. D., THOMAS, R., «Graph searching and a min-max theorem for tree-width», *J. Comb. Th. Ser. B*, 58(1993): 22–33.

MATHEMATISCHES INSTITUT
ABTEI. FÜR MATHEMATISCHE LOGIK
UNIVERSITÄT FREIBURG
ALBERTSTRASSE 23B
FREIBURG, D-70104, ALEMANYA
flum@uni-freiburg.de