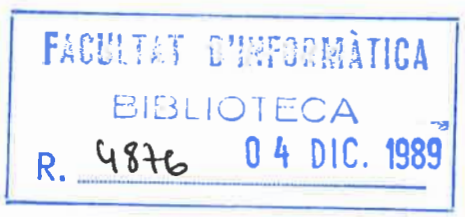
Tractabilitat de NP i altres clases de complexitat per famílies de circuits booleans

R. Gavaldà

Report LSI–88–4



Abstract: We search for nonuniform analogs of the complexity class $NP \cap coNP$. Following mainly the work of Karp and Lipton on these classes, 1) We define two types of polynomial time reducibility, study some of their basic properties and separate them; 2) We show that the corresponding reduction classes of the sparse sets give two versions of the desired nonuniform class; 3) We propose a model of circuits for *one* of the classes obtained.

Resum: En aquest treball busquem versions no uniformes de la classe $NP \cap coNP$. Seguint principalment les tcniques de Karp i Lipton, 1) Definim dos tipus de redubilitat en temps polinmic, n'estudiem algunes propietats bsiques i les separem. 2) Veiem que aplicant aquestes reduibilitats a la classe dels conjunts esparsos obtenim dues versions de la classe no uniforme buscada. 3) Proposem un model de circuits per a una de les classes que obtenim.

TRACTABILITAT DE NP I ALTRES CLASSES DE COMPLEXITAT PER FAMÍLIES DE CIRCUITS BOOLEANS

Tesina presentada per
Ricard Gavaldà Mestre

sota la direcció del doctor
José Luis Balcázar Navarro

a la Facultat d'Informàtica de Barcelona,
Universitat Politècnica de Catalunya

Barcelona, Novembre de 1987

Aquest treball ha estat parcialment suportat amb un ajut a tesina de la U.P.C., resolució del 13 de Maig de 1987.

1. INTRODUCCIÓ

Un dels enfocs dominants de l'estudi de la complexitat dels problemes utilitza únicament models de càlcul seqüencial. Habitualment, un problema es considera tractable si els recursos emprats per a resoldre'l amb algun d'aquests models són suficientment petits. Considerant diferents recursos i amb diferents definicions d'aquest "suficientment petit", la teoria de complexitat ha classificat els problemes en unes classes de complexitat, de les quals les més conegudes són P, NP i PSPACE.

No obstant, són possibles altres definicions de problemes tractables basades en models que explotin el paral·lelisme. D'entre aquests models, el de més tradició, i possiblement el més estudiat, és el circuit booleà.

Quins problemes poden ser tractats per circuits booleans de grandària relativament petita? Per a respondre a aquesta pregunta, Karp i Lipton (1980) defineixen una altra mena de classes de complexitat, que anomenen "no uniformes" per contrast amb les més clàssiques, o "uniformes". Amb l'enfoc de Karp i Lipton, és possible fer correspondre a cada classe uniforme una classe no uniforme.

Així, Pippenger (1979) demostra que la classe no uniforme corresponent a P (anomenada P/poly) és la dels problemes resolubles amb circuits de grandària polinòmica. Yap (1983) i Schöning (1984) demostren que la classe que correspon a NP (NP/poly) és la dels conjunts amb circuits generadors que els ténen com a abast (els que ells anomenen *generadors petits*).

En aquest treball, la nostra intenció és trobar un model no uniforme similar a aquests, és a dir, famílies petites de circuits, per a una altra important classe de complexitat: $NP \cap coNP$.

El model proposat són els anomenats *circuits generadors forts*, que són equivalents a la classe $(NP/poly) \cap (coNP/poly)$. Encara que no és estrictament cap de les classes com les definides per Karp i Lipton, presentem arguments a favor de la seva naturalitat.

Considerem llavors la classe no uniforme que sembla correspondre a $NP \cap coNP$ de manera més directa, que és $(NP \cap coNP)/poly$. Quina és la relació entre aquestes dues classes? Per a veure aquesta relació, en altres casos s'ha utilitzat el mètode consistent

en definir una classe uniforme adequada i relativitzar-la a oracles esparsos. En aquest sentit, Schöning (1984) presenta un resultat molt general que permet fer aquest pas de manera gairebé mecànica per a moltes classes de complexitat.

Comencem per trobar una classe uniforme (és a dir, un model de màquina seqüencial) per a la classe $(NP/poly) \cap (coNP/poly)$ que li correspongui. El model més adequat és el de *màquina forta*, que havia estat proposat i estudiat exhaustivament per Long (1982) a partir d'una idea de Adleman i Manders (1977).

Provem llavors, que aquestes màquines fortes de Long relativitzades a oracles esparsos corresponen exactament a la classe $(NP/poly) \cap (coNP/poly)$, i per tant als generadors forts. Veiem que la tècnica desenvolupada per Schöning no és aplicable a aquest cas. Donem llavors condicions suficients perque es pugui aplicar aquesta tècnica a una classe de complexitat.

Per a intentar determinar quin és, doncs, el significat de $(NP \cap coNP)/poly$, definim un altre tipus de màquina. Aquesta és la *màquina uniformement forta*, més restrictiva que la de Long i molt menys intuitiva. Demostrem, en primer lloc, que les màquines uniformement fortes són essencialment diferents de les màquines simplement fortes, és a dir, que defineixen classes diferents en qualsevol alçada dels conjunts recursius.

Les màquines uniformement fortes que consulten oracles esparsos defineixen, efectivament, la classe $(NP \cap coNP)/poly$. L'obtenció d'aquest resultat és senzilla perque, en aquest cas, la tècnica de Schöning sí que és aplicable.

Caracteritzem, doncs, les dues classes no uniformes estudiades mitjançant relativització de classes uniformes. Fent ús d'aquestes caracteritzacions, discutim les conseqüències que tindria la igualtat i la desigualtat d'aquestes classes. D'altra banda, el fet que una de les caracteritzacions (les màquines uniformement fortes) és molt menys manejable que l'altra (les màquines simplement fortes), sembla indicar de nou que la classe no uniforme que generalitza més naturalment $NP \cap coNP$ és $(NP/poly) \cap (coNP/poly)$ i no $(NP \cap coNP)/poly$.

2. NOTACIÓ I DEFINICIONS BÀSIQUES

2.1 Notació

Considerarem llenguatges, o conjunts de paraules, sobre un alfabet finit, Σ. Una paraula és una seqüència finita d'elements de Σ, o símbols. La longitud d'una paraula x, $|x|$, és el número de símbols que la formen. La paraula de longitud zero serà denotada per λ. Σ^n és el conjunt de totes les paraules de longitud n sobre Σ. Σ^* és el conjunt de totes les paraules sobre Σ.

Una classe de complexitat és el conjunt dels llenguatges que són reconeguts per algun tipus de màquina de Turing en la que s'ha afitat algun recurs. Els recursos que es consideren afitats amb més freqüència són el temps de càlcul i l'espai de treball. Exemples de classes de complexitat són P, NP i PSPACE. Així, les màquines deterministes en les que el temps s'ha afitat per un polinomi de la longitud de l'entrada determinen la classe P. Altres classes a les que farem referència en aquest treball són les de la jerarquia polinòmica de Meyer i Stockmeyer; veure Stockmeyer (1977).

Sigui C una classe de complexitat i A un conjunt. La classe C relativitzada a A, denotada $C(A)$, és el conjunt de llenguatges reconeguts per les màquines que defineixen C quan poden consultar el conjunt A com oracle. Quan considerem les propietats de la classe $C(A)$ que no depenen específicament de l'oracle A, utilitzarem la notació $C()$. En la classe relativitzada PSPACE(), suposarem que la fita polinòmica en l'espai de treball s'aplica també a la cinta d'oracle.

Amb el símbol $B \leq_T^P A$ indicarem que $B \in P(A)$. Amb $B \leq_T^{NP} A$ voldrem dir que $B \in NP(A)$.

Denotarem per $\overline{A} = \Sigma^* - A$ el conjunt complementari d'A. Sigui C una classe de complexitat. Denotarem per co-C el conjunt $\{A : \overline{A} \in C\}$.

Definim la unió marcada ($\oplus$) dels conjunts A i B com

$$A \oplus B = \{1w : w \in A\} \cup \{0w : w \in B\}$$

Denotarem per $\langle .,. \rangle$ una bijecció recursiva de $\Sigma^* \times \Sigma^* \to \Sigma^*$, que suposarem calculable i

invertible en temps polinòmic. L'extensió de $\langle ., . \rangle$ a més de dos arguments és immediata per composició.

Una funció $f : \mathbb{N} \to \Sigma^*$ està afitada polinòmicament si existeix un polinomi $p()$ tal que per a tot n, $|f(n)| \leq p(n)$.

Per a tot conjunt A, la funció $cens_A : \mathbb{N} \to \mathbb{N}$ es defineix per

$$cens_A(n) = \text{ cardinal } (A \cap \Sigma^n)$$

Un conjunt S és espars si $cens_S$ està afitada polinòmicament. Un conjunt T és tally si totes les paraules de T estan formades per un únic símbol.

Els dos resultats següents relacionen la complexitat de conjunts tallys i esparsos. Només es dóna un esboç de la seva demostració.

Proposició 2.1: Tot conjunt tally és espars.

Només cal veure que un conjunt tally no pot tenir més que una paraula de cada longitud. Per tant, el seu cens està afitat pel polinomi $p(n) = 1$.

Teorema 2.2 (Hartmanis 1983): Per a tot conjunt espars S hi ha un tally T tal que $S \in P(T)$.

Per a provar-ho, donat un conjunt espars S considerem el conjunt

$$bits(S) = \{\langle n, k, i, j, b \rangle : k = cens_S(n) \text{ i } b \text{ és l'}i\text{-èssim bit de } y_{n,j}\}$$

on $y_{n,j}$ és la j-èssima paraula de longitud n de S, en l'ordre lexicogràfic. Llavors, es demostra hi ha una màquina determinista que funciona en temps polinòmic i que amb oracle

$$\{0^m : m \in bits(S)\}$$

accepta el llenguatge S.

Si C és un circuit booleà, la grandària de C, escrit $\|C\|$, és el número de portes lògiques de C. Amb $\{C_n\}$ denotarem una família de circuits booleans infinita i enumerable tal que, per a tot n, el circuit C_n té n entrades. Suposarem que existeix una funció afitada polinòmicament en $\|C\|$ que codifica circuits booleans com a paraules de Σ^*.

Ocasionalment farem servir altres notacions o conceptes que no definirem. Els que no estiguin clars pel contexte es poden trobar, per exemple, en Garey i Johnson (1979) o Hopcroft i Ullman (1979).

2.2 Models uniformes de càlcul: conceptes bàsics

El nostre model de càlcul serà una màquina de Turing que podrà ser determinista o indeterminista, i consultar un oracle o no. Ocasionalment, utilitzarem també màquines que produeixen una paraula de sortida, anomenades transductors.

Tractarem també amb un tipus de màquina no determinista que té tres menes d'estats finals: d'acceptació, de rebuig i indefinits. Segons l'estat en que acaben, els càlculs d'una d'aquestes màquines amb una entrada fixa podran classificar-se llavors en :

- càlculs que accepten

- càlculs que refusen

- càlculs indefinits, que indiquen la impossibilitat d'arribar a una conclusió amb aquesta entrada.

Com es desprèn d'aquesta classificació, considerarem només màquines en que tots els possibles càlculs acaben. Assegurarem això afegint a cada màquina un "rellotge" que forci l'acabament del càlcul, normalment en estat indefinit, passat cert número de passos (número que pot dependre de la longitud de l'entrada).

Si no s'indica el contrari, l'alfabet d'entrada d'aquestes màquines serà sempre $\{0,1\}$.

Ocasionalment, serà necessari simular màquines mitjançant altres màquines. Suposarem que una màquina que n'incorpora una altra del mateix tipus (determinista o indeterminista), pot simular-la augmentant el temps de càlcul en un polinomi de la longitud de l'entrada. Si la màquina simulada, M, consulta un oracle, podrem substituir quan convingui l'oracle per una paraula y. Llavors, suposarem que y codifica un conjunt de paraules $\langle y_1, ..., y_k \rangle$ respecte al qual es respondran les consultes.

Sigui M una màquina de Turing indeterminista amb oracle. Denotarem per QUERY, SÍ i NO respectivament l'estat de consulta a l'oracle, i els dos estats en que pot continuar el càlcul a partir de l'estat QUERY. Denotarem per $M^B(x)$ el conjunt dels possibles càlculs de M amb entrada x i consultant l'oracle B.

Una màquina sense oracle N accepta el llenguatge A (escrit $L(N) = A$) quan per a tot x, $N(x)$ conté un càlcul que accepta si i només si $x \in A$. Una màquina M amb oracle B accepta el llenguatge A (escrit $L(M,B) = A$) quan per a tot x, $M^B(x)$ conté un càlcul que accepta si i només si $x \in A$.

Direm que una màquina funciona en temps polinòmic si existeix un polinomi $p()$ tal que per a tota entrada x, el càlcul més llarg de M sobre x té com a molt $p(|x|)$ passes. Si la màquina consulta un oracle, el polinomi $p()$ ha de ser independent de l'oracle.

2.3 Models no uniformes de càlcul: conceptes bàsics

Karp i Lipton (1980) defineixen les classes "no uniformes" i justifiquen la seva importància com a models de càlcul i com a eina per a l'estudi de les classes uniformes corresponents. En aquest apartat exposarem algunes de les definicions i els resultats d'aquesta línia que utilitzarem en el nostre treball.

Les classes no uniformes més estudiades són les de la forma C/poly, definides a continuació, on C és una classe de complexitat clàssica.

Definició 2.3: Per a una classe de complexitat C, C/poly és la classe de conjunts A per als quals hi ha una funció "consellera" $h : \mathbb{N} \to \Sigma^*$ afitada polinòmicament i un conjunt $B \in C$ tal que per a tot $x \in \Sigma^*$,

$$x \in A \iff \langle x, h(|x|)\rangle \in B$$

Intuitivament, C/poly són aquells conjunts tals que, amb una informació addicional o consell relativament petit (polinòmic), són de la mateixa complexitat que els de la classe C. A més, aquest consell és el mateix per a totes les paraules de la mateixa longitud.

És interessant notar la següent propietat:

Proposició 2.4: Per a tota classe de complexitat C, (co-C)/poly = co-(C/poly).

Demostració: Sigui C una classe de complexitat donada, i sigui $A \in (\text{co-}C)/\text{poly}$. Llavors (i només en aquest cas) existeixen $B \in \text{co-}C$, i $h : \mathbb{N} \to \Sigma^*$ afitada polinòmicament tals que

$$\forall x(x \in A \iff \langle x, h(|x|)\rangle \in B)$$

que equival a

$$(1) \qquad \forall x(x \in \overline{A} \iff \langle x, h(|x|)\rangle \in \overline{B})$$

Però $B \in \text{co-}C$ si i només si $\overline{B} \in C$. Llavors, $\overline{B}$ i la mateixa funció $h()$, com que satisfan (1), proven que $\overline{A} \in C/\text{poly}$. Això prova $A \in \text{co-}(C/\text{poly})$.

Com que els recíprocs de tots els raonaments usats també són certs, es poden invertir tots per a provar que co-$(C/\text{poly})\subset(\text{co-}C)/\text{poly}$. □

En aquest treball utilitzarem dues menes de resultats coneguts sobre les classes C/poly: la connexió amb classes relativitzades a oracles esparsos i l'equivalència amb algun tipus de circuit booleà.

Generalitzant resultats anteriors, Schöning (1984) presenta un resultat que permet relacionar de forma gairebé mecànica classes relativitzades a oracles esparsos i les classes no uniformes de Karp i Lipton. A continuació reformulem aquest resultat fent explícits alguns punts relativament informals de la demostració de Schöning.

Comencem per definir les condicions necessàries per a l'aplicació del teorema:

Sigui $C()$ una classe de complexitat relativitzada. Suposarem que hi ha un predicat TIPUS-$C()$ (possiblement no recursiu) tal que, per a tot B, $C(B)$ es pot definir com

$$A \in C \iff \text{hi ha una màquina } N \text{ tal que TIPUS-}C(N) \text{ i } L(N,B) = A$$

Direm que una màquina de Turing M és "de tipus C" si TIPUS-$C(M)$ és cert. Per exemple, les màquines de tipus P són les deterministes que funcionen en temps afitat per un polinomi de l'entrada.

Direm que $C()$ és *bona* si per a tota màquina N de tipus C i tot transductor M de tipus P (amb oracle), la màquina (amb oracle)

llegir(x);
simular $M(x)$, consultant l'oracle;
si $M(x)$ refusa llavors refusar
  sino
    sigui y la sortida de $M(x)$;
    simular $N(y)$ amb oracle $\emptyset$;
  fsi;

també és de tipus C.

Direm que $C()$ és *resistent als oracles* si per a tota màquina N de tipus C, la màquina sense oracle

llegir($\langle x, y\rangle$);

simular $N(x)$ amb oracle y;

també és de tipus C.

Informalment, la condició de "bondat" d'una classe significa que la classe és tancada per la concatenació amb màquines P. Això ens permetrà fer càlculs previs amb l'entrada (modificant-la) i simular després una d'aquestes màquines sense excedir les possibilitats de càlcul de la classe.

La resistència als oracles es pot interpretar dient que les màquines que són de la classe no han de dependre d'un oracle concret per a ser-ne, o sigui, que segueixen comportant-se tal com correspon a la classe sigui quin sigui l'oracle que utilitzin. Això ens permetrà convertir una màquina amb oracle en una altra sense oracle sense sortir-nos de la classe.

Teorema 2.5 (Schöning 1984): Si $C()$ és una classe de complexitat relativitzada, bona, resistent als oracles tal que les màquines de tipus C només consulten el seu oracle amb paraules de longitud afitada polinòmicament per la longitud de l'entrada, llavors

$$C(\emptyset)/poly = \bigcup\{C(S) : S \text{ és espars}\}$$

Demostració: Sigui $A \in C(\emptyset)/poly$, i siguin la funció $h()$ i el conjunt $B \in C(\emptyset)$ els que ho proven. Llavors, per a tot x de Σ^*,

$$x \in A \iff \langle x, h(|x|)\rangle \in B$$

Sigui M la màquina (sense oracle) que prova $B \in C(\emptyset)$. Definim el conjunt espars $S_h = \{\langle 0^n, y\rangle : y \text{ és un prefixe de } h(n)\}$ i la màquina M' amb oracle:

M':

llegir (x);

$n := |x|$;

$y := \lambda$;

mentre $\langle 0^n, y0\rangle \in$ oracle o

$\langle 0^n, y1\rangle \in$ oracle fer

si $\langle 0^n, y0\rangle \in$ oracle

llavors $y := y0$;

sino $y := y1$;

fsi;

fmentre;

simular $M(\langle x, y\rangle)$;

M' és la concatenació de

- un transductor determinista que, amb l'oracle adequat, troba el valor de $h(|x|)$ i té sortida $\langle x, h(|x|)\rangle$

- i una màquina de tipus C que no consulta el seu oracle.

Com que $C()$ és bona, aquesta màquina és també de tipus C. A més, $L(M', S_h) = A$, i per tant $A \in C(S_h)$.

Per a la inclusió contrària, sigui $A = L(M, S)$ per a algun espars S i alguna màquina M de tipus C. Per hipòtesi, hi ha un polinomi $p()$ que afita la longitud de les consultes fetes per M a l'oracle, i un altre $q()$ que afita $cens_S$. Considerem la següent màquina M':

M':

llegir (z);

si z no és de la forma $\langle x, y_1, ..., y_k\rangle, k \leq q(p(|x|))$

llavors refusar

sino simular M amb entrada x i utilitzant $\{y_1, ..., y_k\}$ com a oracle

M' és la concatenació de dues màquines:

- una màquina P

- i una altra que simula una màquina de tipus C substituint l'oracle per l'entrada llegida.

La segona de les màquines és de tipus C perque C és resistent als oracles. A més, C és bona i llavors la seva concatenació amb una màquina P, M', també és de tipus C. Observi's a més que M' no consulta cap oracle. Per tant, si anomenem B el llenguatge $L(M', \emptyset)$, llavors $B \in C(\emptyset)$.

Definim la funció $h_S : \mathbb{N} \to \Sigma^*$:

$$h_S(n) = \langle y_1, ..., y_k\rangle$$

on $\{y_1, ..., y_k\} = S \cap \{x : |x| \leq p(n)\}$. Aleshores, h_S està afitada polinòmicament perque S és espars, i a més

$$\forall x, x \in A \iff \langle x, h_S(|x|)\rangle \in B$$

Això demostra $A \in C(\emptyset)/poly$. □

Noti's que aquesta demostració només és vàlida si la classe $C()$ satisfà les dues condicions de ser bona i ser resistent als oracles, la segona de les quals no apareixia explícitament en Schöning (1984). En l'apartat 5 presentem una classe bona a la qual no és aplicable aquesta demostració perque no és resistent als oracles.

Veiem ara que aquest resultat és aplicable a algunes de les classes de complexitat més importants:

Proposició 2.6: PSPACE() i totes les classes de la jerarquia polinòmica són bones, resistents als oracles i reconegudes per màquines que fan preguntes de longitud polinòmica a l'oracle.

Demostració: Noti's, en primer lloc, que en les condicions imposades a una classe per a ser bona i resistent als oracles no intervé enlloc el tipus de llenguatge acceptat per les màquines.

Recordem breument la definició recursiva de les classes de la jerarquia polinòmica.

$\Sigma_{n+1} = NP(\Sigma_n)$

$\Pi_n = \text{co-}\Sigma_n$

$\Delta_{n+1} = P(\Sigma_n)$,

essent $\Sigma_0 = \Pi_0 = \Delta_0 =$P, és a dir, P($\emptyset$).

Comencem per provar l'enunciat per a les classes Δ_n i les del nivell 0. Per a això, considerem la classe P() relativitzada a qualsevol oracle.

És bona perque la concatenació d'una màquina P() i una màquina P que només consulta l'oracle $\emptyset$ segueix funcionant en temps polinòmic determinista. És resistent als oracles perque el fet que les consultes a l'oracle es resolguin amb un conjunt finit llegit d'entrada, no ha d'afectar el temps de càlcul d'una màquina P() (encara que, evidentment, afecti el llenguatge que accepta).

El mateix raonament és vàlid per a la classe NP() relativitzada a qualsevol oracle, substituint càlculs deterministes per indeterministes en les màquines originals. En efecte,

l'única exigència en aquest cas és que les màquines segueixin funcionant en temps polinòmic indeterminista, condició que no es deixa de satisfer per concatenació amb altres màquines polinòmiques ni substituint l'oracle per una entrada. Això prova que les classes Σ_n també són bones i resistents als oracles.

Tractem ara les classes Π_n. Una màquina de tipus $\Pi_n()$ és la que reconeix un llenguatge complementari d'un de $\Sigma_n()$ en temps polinòmic indeterminista. De nou, l'única condició que cal demanar a les màquines Π_n és seguir funcionant en temps polinòmic quan es concatenen amb màquines P() i quan l'oracle es substitueix per una entrada, que satisfan trivialment.

Per a la classe PSPACE el raonament és similar: tota màquina PSPACE ha d'utilitzar un espai polinòmic en la longitud de l'entrada. Si una màquina P es concatena amb una màquina PSPACE, l'espai utilitzat entre les dues estarà també afitat per un polinomi. A més, com que una màquina PSPACE no utilitza oracle, segueix essent PSPACE si se li dóna un oracle amb l'entrada. Per tant, PSPACE també és bona i resistent als oracles.

Totes les màquines de la jerarquia polinòmica funcionen en temps polinòmic, i per tant només poden utilitzar espai polinòmic. Les màquines PSPACE són les que utilitzen espai polinòmic. Llavors, cap d'elles té espai per a construir preguntes a l'oracle més grans que un polinomi de l'entrada.

Per tant, totes aquestes classes són bones, resistents als oracles i només fan preguntes de longitud polinòmica a l'oracle, que són les condicions requerides per a aplicar el teorema 2.5. □

Un altre interès de les classes no uniformes és que sovint ténen models molt naturals. En especial, Pippenger (1979) i Karp i Lipton (1980) proposen utilitzar famílies de circuits booleans per a representar-les.

Karp i Lipton utilitzen circuits amb una sola sortida, que val 0 ó 1. En aquest sentit, els circuits es consideren com una mena d'autòmats, que accepten o rebutgen la paraula que és present a les portes d'entrada. Llavors, el llenguatge acceptat per un circuit C, escrit $L(C)$ és el conjunt de les paraules que fan aparéixer 1 a la sortida.

Cal notar que el llenguatge acceptat per un circuit només conté paraules de longitud

igual al número de les seves entrades, i que, per tant, és finit. Això suggereix que, donat un conjunt L, es podria definir una família infinita de circuits $\{C_i\}$ tal que cada un d'ells reconeix la "llesca" de L que correspon a paraules d'una mateixa longitud, o sigui,

$$\forall i \geq 0, L(C_i) = L \cap \Sigma^i$$

Llavors, $L = \bigcup_{i \geq 0} L(C_i)$, i direm que L és el conjunt acceptat per la família de circuits $\{C_i\}$. El fet que la representació del llenguatge en una longitud (el circuit) sigui diferent de la que té en altres longituds explica el nom de "classe no uniforme".

És interessant notar que aquesta representació no uniforme és possible per a tot conjunt, fins i tot no recursiu.

Proposició 2.7: Per a tot conjunt L hi ha una família de circuits $\{C_i\}$ que accepta L.

En efecte, per a tota longitud fixa sempre és possible construir un circuit que accepti exactament les paraules de L d'aquesta longitud. No obstant, en general aquest circuit tindrà una grandària exponencial en el número d'entrades. Aquest creixement és massa gran si el que pretenem és utilitzar la representació amb circuits per a resoldre problemes.

Per a assegurar una relativa "tractabilitat" dels conjunts, Pippenger i més tard Karp i Lipton proposen imposar una fita polinòmica al creixement dels circuits. Pippenger demostra llavors que els conjunts que són acceptats per una família de circuits afitada polinòmicament són exactament els de P/poly.

Yap (1983) proposa utilitzar circuits que dónen una sortida, o sigui, que calculen una funció de l'entrada. Per a un circuit de n entrades, el domini d'aquesta funció és igualment Σ^n, però en aquest cas té sentit preguntar-se quin és l'abast del circuit. Yap estudia els conjunts que poden ser abast d'una família infinita de circuits. Imposant de nou una fita polinòmica sobre el creixement de la família, Yap (1983) proposa la següent definició:

Definició 2.8: Un conjunt A té *generadors polinòmics petits* ("Small Generators") si per a algun polinomi $p()$ i per a tot $n \geq 0$ existeix un circuit C_n i un enter e_n tals que:

1) $\|C_n\| \leq p(n)$

2) C_n té e_n entrades

3) C_n té n sortides, més una sortida "indicadora de domini" que val 1 si la sortida és vàlida, i 0 si no

4) per a tot x de Σ^n $x \in A \iff \exists y(|y| = e_n$ i C_n amb entrada y té sortida $x)$.

Observi's que el mateix polinomi $p()$ afita els enters e_n donat que tota entrada ha de correspondre amb al menys una porta lògica. Això garanteix que si una paraula és de l'abast de la funció calculada pel circuit, la paraula del domini que ho prova és, com a màxim, polinòmicament més gran.

L'indicador de domini s'introdueix per dos motius: Primer, permet que funcions amb dominis diferents de Σ^* puguin ser calculades per generadors petits. Segon, permet evitar la incomoditat que tot conjunt amb generadors petits hagi de tenir al menys una paraula de cada longitud.

Els generadors petits van ser proposats com una versió no uniforme de la classe NP. Yap (1983) va demostrar la següent caracterització només en un sentit, i Schöning (1984) va completar-la:

Teorema 2.9: $A \in NP/poly \iff$ A té generadors polinòmics petits.

Demostració (indicació): Sigui $A \in NP/poly$. Llavors hi ha una funció consellera $h()$ afitada polinòmicament i un conjunt $B \in NP$ tal que

$$\forall x, x \in A \iff \langle x, h(|x|)\rangle \in B$$

Per la caracterització per quantificació de NP, és que hi ha una màquina M determinista que funciona en temps polinòmic i un polinomi $p()$ tal que

$$\forall x, x \in A \iff \exists y \text{ tal que } |y| \leq p(|x|) \text{ i } M(\langle x, h(|x|), y\rangle) \text{ accepta}$$

Donades M i $h(n)$, es pot construir un circuit C_n que simuli els càlculs de M sobre una entrada $\langle x, y\rangle$, on $|x| = n$:

$$C_n(\langle x, y\rangle) = \begin{cases} x & \text{si } M(\langle x, h(|x|), y\rangle) \text{ accepta} \\ \text{indefinit} & \text{si no} \end{cases}$$

Com que M funciona en temps polinòmic, $h()$ està afitada polinòmicament i la y necessària també, $\|C_n\|$ pot afitar-se per un polinomi. A més, C_n pot donar sortida x amb alguna entrada si i només si $x \in A$. Per tant, la família $\{C_n\}$ són generadors petits d'A.

Per al contrari, sigui A un conjunt i $\{D_n\}$ una família de generadors petits d'A afitada pel polinomi $p()$. Això vol dir que per a tot x de longitud n,

$$(1) \qquad x \in A \iff \exists y(|y| \leq p(|x|) \text{ i } D_n(y) = x)$$

Definim la funció h_A tal que per a tot n, $h_A(n) = \text{codificació}(D_n)$, i el conjunt B

$$B = \{\langle c, x, y\rangle : c \text{ és un circuit i } c(y) = x\}$$

És possible simular el funcionament d'un circuit amb una entrada en temps determinista polinòmic respecte de la grandària del circuit i aquesta entrada. Per tant, $B \in$ P.

Llavors, (1) equival a dir

$$x \in A \iff \exists y(|y| \leq p(|x|) \text{ i } \langle D_n, x, y\rangle \in B)$$

i es pot invertir el raonament anterior per a obtenir $A \in$ NP/poly. □

Schöning fa notar que aquest resultat és equivalent a la igualtat de dues conegudes caracteritzacions dels conjunts recursivament enumerables: (i) ser la quantificació existencial d'un predicat recursiu i (ii) ser l'abast d'una funció recursiva parcial. En aquesta analogia, la quantificació existencial (afitada polinòmicament) correspon a NP/poly, i l'abast d'una funció parcial (calculable en temps polinòmic) és el que accepta un generador petit.

Aquest comentari ens suggereix un altre model de circuit que és únicament acceptador de paraules, i no calculador de funcions, però en que aquesta quantificació existencial és també molt intuitiva.

Donat un conjunt A que té generadors petits, $\{C_n\}$, considerem els circuits $\{D_n\}$ definits per:

1) Si C_n té n sortides i e_n entrades, D_n té $n + e_n$ entrades i una sortida (més un indicador de domini), i

2) per a tot x de longitud n i tot y de longitud e_n

$$D_n(xy) = \begin{cases} 1 & \text{si } C_n(y) = x \\ \text{indefinit} & \text{si no} \end{cases}$$

D_n pot construir-se a partir de C_n (afegint poques portes lògiques) de manera que també hi hagi un polinomi que afita la seva grandària. Com que C_n genera A en la longitud n, està clar que

$$x \in A \iff \exists y(|y| = e_n \text{ i } D_n(xy) = 1)$$

D'altra banda, una família de circuits d'aquest tipus permet també trobar una família de generadors petits per al mateix conjunt.

En aquests circuits, l'indeterminisme apareix perque per a tota paraula del conjunt que s'introdueixi en les n primeres entrades del circuit, hi ha una manera de completar la resta de les entrades que fa que la paraula sigui acceptada.

També és possible considerar aquestes entrades addicionals com portes indeterministes, que poden valer 0 ó 1 independentment de l'entrada. Llavors, una paraula és acceptada si aquestes portes poden prendre valors (de forma indeterminista) de manera que la sortida del circuit sigui 1.

3. MAQUINES FORTES SENSE ORACLE

En els següents apartats presentem un model de màquina indeterminista definit i estudiat exhaustivament per Long (1982). Comencem per presentar la versió sense oracle i algunes propietats senzilles.

Una màquina de Turing sense oracle M és *forta* si per a tota entrada x:

1. hi ha algun càlcul definit en $M(x)$

2. hi ha algun càlcul de $M(x)$ que accepta si i només si no hi ha cap càlcul de $M(x)$ que refusa.

Apart la seva importància teòrica com a eina per a definir reduïbilitats, les màquines fortes ténen l'atractiu de ser indeterministes sense ser inconsistents. Això és, una màquina forta potser no respongui en tots els càlculs, però sempre dóna resposta en al menys un dels càlculs, i no pot donar resultats contradictoris en dos càlculs diferents.

La següent proposició estableix que, per a tota màquina forta, és possible trobar-ne una altra que accepti el llenguatge complementari.

Proposició 3.1: Si M és una màquina forta, la màquina M' resultat d'intercanviar tots els estats d'acceptació i els de rebuig també és forta i a més $L(M) = \overline{L(M')}$.

Demostració: Sigui x una paraula i considerem cada càlcul de $M'(x)$. Un d'aquests càlculs acaba en estat d'acceptació si i només si hi ha un càlcul idèntic en $M(x)$ que acaba en un estat de rebuig (el mateix, canviant el seu significat). El raonament simètric serveix per als càlculs que refusen. Un càlcul està indefinit si i només si hi ha un altre càlcul idèntic, també indefinit, en $M(x)$.

Per tant, si en $M(x)$ hi ha un càlcul definit, també hi serà en $M'(x)$ (amb resultat contrari). Si en $M(x)$ tots els càlculs definits accepten, tots els càlculs definits de $M'(x)$ refusen, i viceversa. Com que això és cert per a tot x, M' és forta.

A més, per a tot x, $x \in L(M)$ si i només si hi ha un càlcul en $M(x)$ que accepta. Com que M' és forta i accepta el complementari de $L(M)$, això és cert si i només si hi no hi ha cap càlcul en $M'(x)$ que accepta, que significa $x \notin L(M')$.

O sigui, una paraula és a $L(M)$ si i només si no és a $L(M')$, que vol dir que els dos llenguatges són complementaris. □

La següent proposició caracteritza els llenguatges tractables amb màquines fortes. En Long (1982) es pot trobar una demostració per a la versió relativitzada, que també enunciem en 4.5(iv).

Proposició 3.2: A és acceptat per una màquina forta que funciona en temps polinòmic si i només si $A \in NP \cap coNP$.

Demostració: Sigui A fixat i M una màquina forta que funciona en temps polinòmic i accepta A. Pel fet de funcionar en temps polinòmic, $L(M) = A \in$ NP.

Sigui ara M' la màquina resultant d'intercanviar estats d'acceptació i de refús en M. Per la proposició anterior, M' és forta i $L(M') = \overline{L(M)} = \overline{A}$. Com que els càlculs de M i M' només es diferencien en els estats finals, M' també funciona en temps polinòmic. Això prova $A \in$ coNP, i, per tant, $A \in NP \cap coNP$.

Per al contrari, sigui $B \in NP \cap coNP$. Hi han, doncs, dues màquines indeterministes M i M' que funcionen en temps polinòmic i accepten respectivament B i $\overline{B}$. "Superposant" M i M', construim la següent màquina N:

N:

llegir(x);

$z :=$ conjecturar ($x \in$ B);

si $z =$ cert llavors

simular $M(x)$;

si $M(x)$ accepta llavors acceptar;

sino

simular $M'(x)$;

si $M'(x)$ accepta llavors refusar;

fi si;

(Quan N acaba el càlcul sense acceptar ni refusar, ho fa en algun estat especial que indica indefinició).

N és forta perque, per a tota x,

1) una de les dues màquines ha de tenir un càlcul que accepta (serà M o M' segons que $x \in B$ o no), i per tant $N(x)$ té al menys aquest càlcul.

2) no poden haver-hi dos càlculs de $N(x)$ contradictoris, perque això voldria dir que x és acceptada per M i per M', que és fals perque hem suposat que accepten conjunts complementaris.

N accepta B perque, per a tot x, $N(x)$ accepta si i només si $M(x)$ accepta, i per tant, $L(N) = L(M) = B$. És fàcil veure que el temps de funcionament de N no és molt més gran que el màxim dels de M i M', i també es pot afitar per un polinomi. □

Amb aquest resultat queda caracteritzada la potència de les màquines fortes per elles mateixes. En l'apartat següent les utilitzarem per a comparar la complexitat de conjunts.

4. MÀQUINES FORTES AMB ORACLE

És pot donar a una màquina forta la possibilitat de fer consultes a un conjunt oracle. Quines condicions ha de satisfer la màquina resultant per a seguir considerant-la forta? Al menys dues interpretacions diferents són possibles. A continuació les exposarem les dues, definint per a cada una un concepte de reduïbilitat.

Definició 4.1: M és *forta amb oracle* B si per a tota entrada x:

1. hi ha algun càlcul definit en $M^B(x)$;

2. hi ha algun càlcul de $M^B(x)$ que accepta si i només si no hi ha cap càlcul de $M^B(x)$ que refusa.

Definició 4.2: Un conjunt A és *SN-reduïble* a un conjunt B (escrit $A \leq^{SN} B$) si hi ha una màquina M amb oracle tal que

1) funciona en temps polinòmic

2) és forta amb oracle B, i

3) $L(M,B) = A$.

La SN-reduïbilitat és la "Strong Nondeterministic Turing reducibility" de Long (1982), d'on conservem el nom "SN". La notació exacta de Long és $\leq_T^{SN}$ perque també defineix altres reduïbilitats denotades amb SN.

No entrarem en detall en aquestes altres reduïbilitats. Mencionem, per exemple, $\leq_m^{SN}$ i $\leq_{tt}^{SN}$, que es defineixen com les més clàssiques $\leq_m^P$ i $\leq_{tt}^P$ canviant els transductors deterministes per transductors indeterministes forts. Les reduïbilitats així definides ténen una estructura molt similar a la de les reduïbilitats $\leq^P$. Enunciem a continuació alguns resultats que comparen aquestes reduïbilitats entre elles. Veure Long (1982) per a una demostració.

Teorema 4.3: Per a tots els conjunts A i B,

(i) $A \leq_m^{SN} B \Rightarrow A \leq_{tt}^{SN} B$

(ii) $A \leq_{tt}^{SN} B \Rightarrow A \leq^{SN} B$

(iii) cap de les implicacions anteriors pot ser invertida.

Teorema 4.4: Per a tot A recursiu, $A \notin NP \cap coNP$, existeix un B recursiu tal que $A \leq_T^P B$ i $A \not\leq_{tt}^{SN} B$.

Long demostra també que la SN-reduïbilitat és essencialment diferent de les reduïbilitats més clàssiques $\leq_T^P$ i $\leq_T^{NP}$, i que per la seva potència es troba entre aquestes dues.

Teorema 4.5: Per a tots els conjunts A i B,

(i) $A \leq_T^P B \Rightarrow A \leq^{SN} B$.

(ii) $A \leq^{SN} B \Rightarrow A \leq_T^{NP} B$.

(iii) Cap de les implicacions anteriors pot ser invertida.

(iv) $A \leq^{SN} B \iff (A \leq_T^{NP} B$ i $\overline{A} \leq_T^{NP} B)$. En altres paraules,

$$A \leq^{SN} B \iff A \in NP(B) \cap coNP(B)$$

Presentem ara una interpretació diferent de quines màquines es poden considerar fortes amb oracle, i la reduïbilitat associada a aquesta interpretació.

Definició 4.6: M és ***uniformement forta*** si per a tot oracle A, M és forta amb A.

Definició 4.7: Un conjunt A és ***UN-reduïble*** a un conjunt B (escrit $A \leq^{UN} B$) si hi ha una màquina M amb oracle tal que

1) funciona en temps polinòmic,

2) és uniformement forta, i

3) $L(M, B) = A$.

Noti's que l'única diferència entre $\leq^{UN}$ i $\leq^{SN}$ consisteix en el comportament amb qualsevol oracle de la màquina que fa la reducció: Exigim a les màquines uniformement fortes que mantinguin la seva coherència sigui quin sigui l'oracle que consulten. Aquesta diferència de comportament no sembla tenir una analogia natural en el cas de transductors sense oracle. Per aquesta raó, no definirem, com Long, UN-reduïbilitats similars a $\leq_m^P$ i $\leq_{tt}^P$ a partir de transductors.

Les següents proposicions són relacions bàsiques entre $\leq^{UN}$ i altres tipus de reduïbilitat.

Proposició 4.8: Per a tots els conjunts A i B,

(i) $A \leq_T^P B \Rightarrow A \leq^{UN} B$.

(ii) $\leq^{UN} B \Rightarrow A \leq^{SN} B$.

Per a provar (i), només cal veure que una màquina P, amb qualsevol oracle i qualsevol entrada, té un sol càlcul que sempre és definit, i per tant és uniformement forta.

Per a provar (ii), només cal recòrrer a la definició de les màquines que fan ambdues reduccions: una màquina uniformement forta és forta, en particular, amb l'oracle que consulta durant la reducció.

Les següents són algunes propietats importants d'aquestes reduïbilitats. Una demostració més extensa per al cas de $\leq^{SN}$pot trobar-se de nou en Long (1982).

Teorema 4.9: Per a tot parell de conjunts A i B,

(i) $A \leq^{SN} B$ i $B \in NP \cap coNP \Rightarrow A \in NP \cap coNP$.

(ii) $A \leq^{UN} B$ i $B \in NP \cap coNP \Rightarrow A \in NP \cap coNP$.

(iii) $A \in NP \cap coNP \Rightarrow A \leq^{SN} B$.

(iv) $A \in NP \cap coNP \Rightarrow A \leq^{UN} B$.

Demostració: Per a (i), sigui M la màquina forta (amb oracle) que fa $A \leq^{UN} B$ i M' la màquina forta (sense oracle) que prova $B \in NP \cap coNP$, obtinguda per la proposició 3.2. És fàcil veure que la màquina que simula M substituint les preguntes a l'oracle per simulacions de M' és forta i accepta igualment A.

Per a (iv), sigui M la màquina forta sense oracle que garanteix $A \in NP \cap coNP$. Podem considerar M com una màquina amb oracle que ignora les respostes de l'oracle, i que per tant accepta sempre el mateix llenguatge i és uniformement forta. El llenguatge que accepti amb oracle B seguirà essent A, el que prova $A \leq^{UN} B$.

Com que $\leq^{UN} \Rightarrow \leq^{SN}$, (ii) i (iii) es desprenen directament de (i) i (iv). □

Per a una reduïbilitat $\leq_\alpha$, el conjunt $\{A : A \leq_\alpha \emptyset\}$ s'acostuma a anomenar el seu *grau zero*. Notem que $\leq^{UN}$ i $\leq^{SN}$coincideixen en els seus graus zero.

Corol·lari 4.10:

(i) $A \leq^{SN} \emptyset \iff A \in NP \cap coNP$.

(ii) $A \leq^{UN} \emptyset \iff A \in NP \cap coNP$.

Els dos enunciats es dedueixen immediatament prenent $B = \emptyset$ en la proposició anterior.

La igualtat no és manté més enllà del grau zero. De fet, el nostre primer resultat estableix que, per sobre de $NP \cap coNP$, $\leq^{UN}$ és essencialment diferent de totes les

reduïbilitats definides per Long (1982).

Començarem per provar que difereix de $\leq^{SN}$. Intuitivament, la diferència es deguda a que les màquines que no són uniformement fortes poden "confiar" en rebre l'oracle adequat i explotar a fons la seva estructura. En canvi, les màquines uniformement fortes han de mantenir-se coherents sigui quina sigui la informació que proporcioni l'oracle, i, per tant, no poden esperar "a priori" que l'oracle tingui cap propietat particular.

Teorema 4.11: Per a tot conjunt recursiu A, $A \notin NP \cap coNP$, existeix un conjunt recursiu B tal que $A \leq^{SN} B$ i $A \not\leq^{UN} B$.

Demostració: Sigui A un conjunt fixat. Construirem un conjunt B amb la següent propietat:

(#) $\forall x, x \in A \iff \exists y(|y| = |x| \text{ i } \langle x, y, 1\rangle \in B) \iff \forall z(|z| = |x| \Rightarrow \langle x, z, 0\rangle \notin B)$

o sigui, per cada paraula x incloem en B alguna paraula de la forma $\langle x, y, 1\rangle$ si $x \in A$, i alguna paraula de la forma $\langle x, z, 0\rangle$ si $x \notin A$. Aquesta propietat garanteix $A \leq^{SN} B$, amb el següent procediment:

llegir (x);
$z :=$ conjecturar $(x \in A)$;
si $z =$ cert llavors
    conjecturar y tal que $|y| = |x|$;
    si $\langle x, y, 1\rangle \in$ oracle llavors acceptar;
sino
    conjecturar u tal que $|u| = |x|$;
    si $\langle x, u, 0\rangle \in$ oracle llavors refusar;
fsi;
parar sense sortida;

Observem que aquest procediment funciona en temps polinòmic en $|x|$ i que, per la propietat imposada a B, el procediment és fort amb oracle B (ha de tenir exactament un sol càlcul que accepti o que refusi) i el llenguatge que accepta és A. Això garanteix que $A \leq^{SN} B$.

Noti's, però, que procediments similars mai seràn forts si l'oracle que consulten no satisfà la condició #. Aquesta propietat serà la que explotarem a continuació.

Cal ara veure que B pot ser construit de manera que $A \not\leq^{UN} B$. Per a això, construirem B per diagonalització sobre totes les màquines candidates a fer la reducció, i per a cada una d'elles, assegurarem que:

- o no és uniformement forta, és a dir, trobem un conjunt oracle amb el que no és coherent,

- o podem incloure en B un testimoni de que no redueix A a B.

Veurem que aquest testimoni existeix sempre que $A \notin NP \cap coNP$. En qualsevol cas, una màquina que satisfà alguna d'aquestes condicions no pot fer la reducció.

En el que segueix, direm que un parell de conjunts C i C' són extensions de D i D' si $D \subseteq C$ i $D' \subseteq C'$.

Direm que dos conjunts C i C' són complets i consistents amb $\#$ si:

1) $C \cap C' = \emptyset$;

2) per a algun n, $C \cup C'$ conté exactament totes les paraules de longitud $\leq n$;

3) C satisfà $\#$ en un segment inicial d'A, és a dir, existeix un m tal que per a tot x, si $|x| < m$ llavors

$$x \in A \iff \exists y(|y| = |x| \text{ i } \langle x, y, 1\rangle \in C) \iff \forall z(|z| = |x| \Rightarrow \langle x, z, 0\rangle \notin C)$$

Direm que D i D' simplement són consistents amb $\#$ si existeixen C i C' tals que:

1) C i C' són extensions de D i D';

2) C i C' són complets i consistents amb $\#$.

Construirem B i $\overline{B}$ en etapes; a l'etapa n, B_n serà el conjunt de paraules que ja hem decidit incloure a B, i B'_n el de les que ja hem decidit incloure a $\overline{B}$. Sigui $\{M_n\}$ una enumeració efectiva de les màquines indeterministes que funcionen en temps polinòmic. Llavors, B_n garantirà que M_n no és uniformement forta o bé que no fa la reducció $A \leq^{UN} B$.
Etapa 0:

$B_0 := \emptyset$;

$B'_0 := \emptyset$;

Etapa n:

* buscar una paraula x i dos conjunts finits S i T tals que:

(1) S i T són extensions de B_{n-1} i B'_{n-1}.

(2) S i T són consistents amb (#).

(3) per a algun oracle D, M_n^D no és forta, o bé

x demostra que M_n^S no respon correctament a la pregunta "$x \in A$".

* siguin B_n i B'_n extensions de S i T completes i consistents amb #.

Buscarem x, S i T a l'etapa n amb el procediment:

per a tota paraula x fer

simular tots els càlculs de $M_n(x)$ amb oracle B_{n-1};

(i)

si no hi ha cap càlcul definit o

hi ha un càlcul que accepta i un que refusa llavors

{M_n no és forta amb B_{n-1}}

si $x \in A$ llavors

$S := B_{n-1} \cup \{\langle x, x, 1\rangle\}$; $T := B'_{n-1}$

sino

$S := B_{n-1} \cup \{\langle x, x, 0\rangle\}$; $T := B'_{n-1}$

sortir del bucle;

(ii)

sino si $x \in A$ i hi ha un càlcul que refusa llavors

{M_n s'ha equivocat amb x}

sigui R la llista de les paraules consultades

a l'oracle en aquest càlcul amb resposta negativa;

sigui w tal que $|w| = |x|$ i $\langle x, w, 1\rangle \notin R$;

$S := B_{n-1} \cup \{\langle x, w, 1\rangle\}$;

$T := B'_{n-1} \cup R$;

sortir del bucle;

(iii)

sino si $x \notin A$ i hi ha un càlcul que accepta llavors

{M_n s'ha equivocat amb x}

sigui R la llista de les paraules consultades

a l'oracle en aquest càlcul amb resposta negativa;

sigui w tal que $|w| = |x|$ i $\langle x, w, 0\rangle \notin R$;

$S := B_{n-1} \cup \{\langle x, w, 0\rangle\}$;

$T := B'_{n-1} \cup R$;

sortir del bucle;

sino

{ $M_n(x)$ sembla comportar-se correctament }

{ en aquest cas; x no és encara el testimoni buscat };

fper;

Suposem en primer lloc que aquest procediment no acaba, o sigui, que x, S i T mai es troben. Això significaria que, per a tot x, cap dels casos (i) - (iii) és cert. Per exclusió, ha de ser cert per a tot x que

$M_n(x)$ amb oracle B_{n-1} té al menys un càlcul definit i

tots els càlculs definits donen la mateixa resposta i

(($x \in A$ i té un càlcul que accepta) o ($x \notin A$ i té un càlcul que refusa))

Per definició, això vol dir que M_n és forta amb oracle B_{n-1} i que $L(M_n, B_{n-1}) = A$. Peró com que B_{n-1} és finit, hi ha una altra máquina forta M_p (sense oracle) que incorpora B_{n-1} i tal que $L(M_p) = A$. Per la proposició 3.2, això significa $A \in NP \cap coNP$. Com que això és fals per hipòtesi, el procediment ha d'acabar.

Cal veure ara que els x, S i T satisfan les condicions (1)-(3) desitjades. Si x és trobat en el cas (i), M_n no és forta amb oracle B_{n-1} i (3) es satisfà trivialment (encara que M_n passi a ser forta amb les paraules que afegim a B_{n-1}). En els casos (ii) i (iii), x assegura que la reducció no es fa amb M_n, i cal a més assegurar que

l'extensió de S a B_n es pot fer de manera que (3) segueix essent vàlida. Incloent en T de les paraules consultades a l'oracle que no eren a B_{n-1} assegurem que els càlculs (o contraexemples) trobats segueixen existint amb oracle B_n.

En qualsevol cas, la consistència amb (#) de S i T s'assegura incloent només una paraula de la forma $\langle x, w, 1\rangle$ o $\langle x, w, 0\rangle$ segons $x \in A$. Noti's que la paraula x es pot triar sempre suficientment gran perque, en els casos (ii) i (iii), la paraula w existeixi quan la necessitem.

Amb aquesta construcció, B_n estèn B_{n-1} preservant els testimonis anteriors i la consistència amb (#). Sigui B la unió de tots els B_n. Observi's que (1) si A és recursiu, la construcció de B és recursiva, i (2) B és extensió de tot B_n. Per tant, per la condició (3), si M és uniformement forta no pot reconéixer A. □

Veurem ara que $\leq^{UN}$és diferent de $\leq^{SN}_{tt}$.

Teorema 4.12: **Per a tot A recursiu, $A \notin NP \cap coNP$, existeix B recursiu tal que $A \leq^{UN} B$ i $A \not\leq^{SN}_{tt} B$.**

Demostració: **Sigui A fixat. Pel teorema 4.4, hi ha un B recursiu tal que $A \leq^{P}_{T} B$ i $A \not\leq^{SN}_{tt}$.** Com que $\leq^{P}_{T}$ implica $\leq^{UN}$, el mateix conjunt B satisfà $A \leq^{UN} B$ i per tant l'enunciat del teorema. □

D'aquest teorema, i com que $\leq^{SN}_{m}$ implica $\leq^{SN}_{tt}$, es dedueix que $\leq^{SN}_{m}$ i $\leq^{UN}$ són també diferents. Amb això hem provat que $\leq^{UN}$ és diferent de les reduïbilitats SN definides per Long.

5. CLASSES NO UNIFORMES DEFINIDES PER MÀQUINES FORTES

5.1 Conjunts esparsos i màquines uniformement fortes

Hem vist en l'apartat 2 que moltes classes uniformes poden ser relativitzades amb oracles esparsos i són llavors equivalents a classes no uniformes de la forma /poly. En aquest apartat mostrem equivalències similars per a $\leq^{UN}$ i $\leq^{SN}$.

Començarem per establir la classe no uniforme corresponent a la UN-reduïbilitat.

Teorema 5.1: $(NP \cap coNP)/poly = \{A : \exists S(S \text{ espars i } A \leq^{UN} S)\}$.

Demostració: L'enunciat és la conseqüència d'aplicar el teorema 2.5 a la classe definida per les màquines uniformement fortes. Només cal veure doncs que la classe relativitzada a B

$$C(B) = \{A : A \leq^{UN} B\}$$

satisfà les condicions que aquell teorema exigeix.

Per a veure que és bona, sigui M una màquina uniformement forta, i sigui P un transductor determinista que funciona en temps polinòmic i consulta un oracle. La màquina:

llegir(x);
si $P(x)$ refusa llavors refusar
  sino simular $M(P(x))$ amb oracle $\emptyset$;

amb oracle B té un sol càlcul (el que refusa) si $x \notin L(P,B)$. Si $x \in L(P,B)$, es comporta com $M^{\emptyset}(P(x))$, i per tant és forta. Llavors, siguin quins siguin B, P i x, aquesta màquina és uniformement forta.

Per a veure que és resistent als oracles, sigui la màquina

N:
  llegir(x,y);
  simular $M(x)$ amb el conjunt y com oracle;

on M és uniformement forta. Llavors, sigui quin sigui el conjunt y d'entrada, M és forta amb oracle y i per tant N és simplement forta. Tant en aquest cas com en l'anterior, les màquines construides funcionen en temps polinòmic si les originals ho fan.

A més, com que les màquines que fan reduccions $\leq^{UN}$ funcionen en temps polinòmic, només ténen temps de construir paraules de longitud polinòmica per a consultar-les a l'oracle.

Per tant, les condicions requerides per a aplicar el teorema 2.5 són certes, i l'enunciat del teorema és immediat. □

5.2 Conjunts esparsos i màquines fortes amb algun oracle

En aquesta secció pretenem trobar una classe no uniforme equivalent a la classe definida per SN-reduïbilitat a conjunts esparsos, és a dir, a

$$\{A : \exists S(S \text{ espars i } A \leq^{SN} S)\}$$

No serà possible aplicar el teorema 2.5 de Schöning per a trobar una classe de complexitat X tal que X/poly coincideixi amb l'esmentada.

En efecte, és fàcil veure que aquesta classe no necessàriament satisfà la propietat de resistència als oracles necessària per a l'aplicació del teorema. Això és, no podem provar que donada una màquina M, forta amb cert oracle B, hi ha un altre oracle amb que la màquina N (sense oracle):

N:
    llegir(x,y);
    simular $M(x)$ amb oracle y;

sigui forta. En concret, si M no és forta amb un oracle C, serà possible trobar algun segment inicial finit y de C i una paraula x suficientment gran perque $N(x, y)$ consulti exactament y i no sigui forta.

Aquesta "irregularitat" de la SN-reduïbilitat no significa que la seva interpretació intuitiva sigui més difícil. Al contrari, s'ha trobat un model no uniforme, els *circuits generadors forts*, amb una interpretació concreta molt senzilla. En aquest apartat enunciarem dues caracteritzacions prèvies, que serviran per a presentar aquest model en l'apartat següent.

La primera d'elles és la reduïbilitat a conjunts esparsos i, informalment, demostra que el conjunt espars es pot "treure com factor comú" de dues màquines indeterministes que acceptin llenguatges complementaris.

Teorema 5.2 (Caracterització 1):

$$\{A : \exists S(S \text{ espars i } A \leq^{SN} S)\} = \bigcup\{NP(S) : S \text{ espars }\} \cap \bigcup\{coNP(S) : S \text{ espars }\}$$

Demostració: Sigui A tal que $A \leq^{SN} S$ per a algun espars S. La proposició 4.5(iv) assegura que $A \in NP(S)$ i $A \in coNP(S)$.

Per a la inclusió contraria, sigui A tal que per a alguns conjunts esparsos $S_1, S_2, A \in NP(S_1)$ i $A \in coNP(S_2)$. Això equival a dir $A \leq_T^{NP} S_1$ i $\overline{A} \leq_T^{NP} S_2$. Siguin M i M' les màquines que fan aquestes reduccions.

Definim N, màquina amb oracle:

llegir (x);
$z :=$ conjecturar$(x \in A)$;
si $z =$ cert llavors
  simular $M(x)$, substituint cada pregunta a l'oracle per
    sigui y la paraula preguntada;
    preguntar a l'oracle per $1y$ i passar a SÍ o NO de M
    segons la resposta;
  si $M(x)$ accepta llavors acceptar;
sino
  simular $M'(x)$, substituint cada pregunta a l'oracle per
    sigui y la paraula preguntada;
    preguntar a l'oracle per $0y$ i passar a SÍ o NO de M'
    segons la resposta;
  si $M'(x)$ accepta llavors refusar;
fsi;

Sigui $S = S_1 \oplus S_2$. LLavors $N(x)$ amb oracle S simula $M^{S_1}(x)$ o $M'^{S_2}(x)$, o sigui, només permet a cada màquina l'accès a la part de l'oracle amb la qual és forta. Llavors:

- si $x \in A$, $M(x)$ té al menys un càlcul que accepta i cap que refusa, i $M'(x)$ no té cap càlcul que accepta. Per construcció, $N^S(x)$ manté el camí que accepta i no afegeix cap camí que refusa.

- si $x \notin A$, $M'(x)$ té al menys un càlcul que accepta i cap que refusa, i $M(x)$ no té cap càlcul que accepta. Per construcció, $N^S(x)$ refusa amb algun camí i no té cap camí que accepta.

Per tant, amb oracle S, N és forta i accepta el llenguatge A. És fàcil veure que si M i M' funcionen en temps polinòmic, el càlcul addicional que fa N pot ser afitat també per un polinomi. □

La segona caracterització és l'equivalència amb una classe no uniforme, encara que no serà de la forma habitual /poly. Ja s'ha discutit al principi de l'apartat la dificultat d'aplicar les tècniques normals a aquest cas. En canvi, la classe que presentem es dedueix immediatament de la primera caracterització.

Teorema 5.3 (Caracterització 2):

$$\{A : \exists S(S \text{ espars i } A \leq^{SN} S)\} = (NP/poly) \cap (coNP/poly)$$

Demostració: Per la proposició 2.6 tota classe de complexitat C que formi part de la jerarquia polinòmica satisfà

$$C/poly = \{C(S) : S \text{ espars }\}$$

Com que NP i coNP formen part d'aquesta jerarquia,

$$NP/poly = \{NP(S) : S \text{ espars}\}, \text{ i}$$

$$coNP/poly = \{coNP(S) : S \text{ espars}\}$$

i per la caracterització 1, la seva intersecció és

$$\{A : \exists S(S \text{ espars i } A \leq^{SN} S)\}$$

□

És interessant notar que les tres caracteritzacions donades segueixen essent certes si es substitueixen els conjunts esparsos per conjunts tally.

Proposició 5.4: (i) $A \leq^{SN} S$ per a algun espars $S \iff A \leq^{SN} T$ per a algun tally T. (ii) $A \leq^{UN} S$ per a algun espars $S \iff A \leq^{UN} T$ per a algun tally T.

Demostració: La inclusió d'esquerra a dreta és trivial perque tot conjunt tally és també espars.

Per a la contrària, sigui S un conjunt espars. El teorema 2.2 garanteix que hi ha un tally T tal que $S \in P(T)$. Llavors, en la màquina que demostra $A \leq^{SN} S$ ($A \leq^{UN} S$), cada consulta a l'oracle S es pot substituir per un càlcul determinista que fa consultes a l'oracle T. Com que el càlcul addicional en l'estat de consulta a l'oracle triga un temps polinòmic en la longitud de la paraula consultada a l'oracle, i aquesta està afitada polinòmicament en la longitud de l'entrada, la modificació només fa créixer el temps de càlcul de la màquina indeterminista en un polinomi. □

Convé fer algunes comparacions entre les caracteritzacions fetes en aquest apartat. Notem en primer lloc que $(NP \cap coNP)/poly$ està inclós en NP/poly i en coNP/poly, i per tant en la seva intersecció. Recordant les caracteritzacions donades, també es pot deduir aquesta inclusió del fet que la UN-reduïbilitat implica la SN-reduïbilitat. Llavors, la classe dels conjunts UN-reduïbles a esparsos està inclosa en la dels SN-reduïbles a esparsos, i hem demostrat que aquestes són equivalents a les dues classes no uniformes esmentades.

És certa la inclusió contrària? Està clar que si P = NP, llavors les dues classes coincideixen i són iguals que P/poly. Per tant, si la pregunta es respongués negativament tindriem P≠NP. Noti's que la hipòtesi P= $NP \cap coNP$ no és suficient, perquè això només permet deduïr que P/poly = $(NP \cap coNP)$/poly, i no res respecte a l'altra classe en qüestió.

Respondre afirmativament a $(NP \cap coNP)/\text{poly} \stackrel{?}{=} (NP/poly) \cap (coNP/poly)$ equival a dir que $\leq^{SN}$i $\leq^{UN}$són equivalents en la classe dels conjunts esparsos, és a dir, que tot conjunt SN-reduïble a un espars és també UN-reduïble a algun altre espars. Això sembla difícil perque la màquina uniformement forta que fes la reducció hauria de ser forta no sols amb els esparsos sino amb qualsevol altre oracle.

6. CIRCUITS GENERADORS FORTS

A continuació proposem per a la SN-reduïbilitat amb oracles esparsos un model de circuit equivalent similar als generadors petits presentats en l'apartat 2.

Definició 6.1: Un conjunt A té *generadors polinòmics forts* si hi ha un polinomi $p()$ tal que per a tot $n \geq 0$ hi ha un circuit C_n i un enter e_n tals que:

1) $\|C_n\| \leq p(n)$

2) C_n té e_n entrades

3) C_n té $n+1$ sortides, més una sortida "indicadora de domini" que val 1 si la sortida és vàlida, i 0 si no.

4) per a tot x de Σ^n

$$x \in A \iff \exists y(|y| = e_n \text{ i } C_n(y) = 1x)$$

$$x \notin A \iff \exists z(|z| = e_n \text{ i } C_n(z) = 0x)$$

Observi's que d'1) i 2) es dedueix que e_n està afitat també per $p()$.

Aquests circuits recullen bé la idea de "força" present en la SN-reduïbilitat. Intuitivament, si C_n és de la família de generadors forts d'A, tota paraula de longitud n apareix en la sortida de C_n si sabem triar l'entrada adequada, i llavors C_n indica correctament si aquesta paraula és a A o no.

Tots els comentaris fets a la definició 2.8 són aplicables també als generadors forts. Noti's en particular que cal una sortida indicadora de domini per a reconéixer conjunts que no tenen cap paraula en alguna longitud. Igualment, tot conjunt tindria generadors petits si no afitessim el seu creixement per un polinomi, i llavors el model es tornaria trivial.

El següent teorema és una caracterització de la SN-reduïbilitat a esparsos que es desprèn de les dues donades en l'apartat anterior:

Teorema 6.2:

$$A \leq^{SN} S \text{ per a algun S espars} \iff A \text{ té generadors forts}$$

Demostració: Sigui A SN-reduïble a S per a cert espars S. Per la caracterització 2, $A \in (NP/poly) \cap (coNP/poly)$. Llavors, com que (coNP)/poly = co-(NP/poly) (veure proposició 2.4), $A \in$(NP/poly) i $\overline{A} \in$(NP/poly).

El teorema 2.9 assegura que, en aquestes condicions, A i $\overline{A}$ ténen generadors petits $\{C_n\}$ i $\{D_n\}$ respectivament. Construim per a tot $n \geq 0$ el següent circuit E_n:

$$E_n(\langle x, y\rangle) = \begin{cases} 1x & \text{si } C_n(y) = x \\ 0x & \text{si } D_n(y) = x \\ \text{indefinit} & \text{si } C_n(y) \text{ i } D_n(y) \text{ estan indefinits} \end{cases}$$

Observi's que, si C_n i D_n són els generadors petits suposats, hi ha una entrada que produeix sortida $1x$ si i només si $x \in A$, i una entrada que produeix sortida $0x$ si i només si $x \notin A$. És a dir, les dues primeres condicions de la definició són mutuament excloents.

També és fàcil veure que si C_n i D_n ténen grandària polinòmica, la construcció es pot fer de manera que $\|E_n\|$ sigui polinòmica. Per tant, $\{E_n\}$ és una família de generadors forts d'A.

Per al contrari, suposem que A té la família de generadors forts $\{E_n\}$. Construim dues famílies de circuits $\{C_n\}$ i $\{D_n\}$:

$$C_n(y) = \begin{cases} x & \text{si } E_n(y) = 1x \\ \text{indefinit} & \text{altrament} \end{cases}$$

$$D_n(y) = \begin{cases} x & \text{si } E_n(y) = 0x \\ \text{indefinit} & \text{altrament} \end{cases}$$

Si (i només si) $x \in A$, hi ha una entrada amb que E_n produeix $1x$, i per construcció, amb que C_n produeix x. Per tant els $\{C_n\}$ són generadors petits d'A. Pel mateix raonament $\{D_n\}$ són generadors petits d'$\overline{A}$.

A partir d'aquí, podem invertir el raonament fet al principi de la demostració i veure que $A \in NP/poly$ i $\overline{A} \in NP/poly$. Llavors, per la caracterització 2, és que hi ha un espars S tal que $A \leq^{SN} S$. □

Com en el cas dels generadors petits, els generadors forts poden convertir-se en acceptadors amb portes indeterministes. Si A té generadors forts $\{C_n\}$, definim els circuits $\{D_n\}$ per:

1) Si C_n té $n+1$ sortides i e_n entrades, D_n té $n+e_n$ entrades i una sortida (més un indicador de domini), i

2) per a tot x de longitud n i tot y de longitud e_n

$$D_n(xy) = \begin{cases} 1 & \text{si } C_n(y) = 1x \\ 0 & \text{si } C_n(y) = 0x \\ \text{indefinit} & \text{si no} \end{cases}$$

És fàcil veure que els D_n accepten A, en el sentit següent:

$$x \in A \iff \exists y, D_n(xy) = 1 \iff \forall z, D_n(xz) \neq 0$$

És a dir, per a tota paraula, podem completar les entrades indeterministes del circuit de manera que una paraula del conjunt sigui acceptada. Però, a més, també hi ha un testimoni del contrari per a tota paraula que no és del conjunt. La resta dels comentaris fets al final de l'apartat 2 són fàcilment traduïbles a aquest cas.

7. CONCLUSIONS

En aquest apartat resumim els resultats obtinguts i donem algunes indicacions sobre futures línies d'investigació.

En primer lloc, hem considerat dos models de màquines indeterministes i hem vist que són notablement diferents quan es permet que tinguin oracles. En concret, defineixen classes relativitzades diferents a qualsevol alçada dels conjunts recursius. El primer tipus de màquina és la màquina forta, que ja havia estat estudiada per altres autors. El segon tipus, que hem anomenat màquina uniformement forta, apareix com una restricció del primer.

A continuació, hem considerat les classes que defineixen aquestes màquines quan utilitzen oracles esparsos. Això ens permet definir classes no uniformes de la forma /poly. Així, les màquines fortes corresponen a la classe $(NP/poly) \cap (coNP)/poly$ i les uniformement fortes a $(NP \cap coNP)/poly$. Provem també que és possible utilitzar oracles tallys enlloc d'esparsos sense perdre generalitat.

La resta del treball es dedica a la comparació de les classes obtingudes, $(NP \cap coNP)/poly$ i $(NP/poly) \cap (coNP/poly)$. Comprovem que la segona inclou la primera i discutim les condicions que farien possible la igualtat, sense poder respondre-la afirmativament ni negativament.

Proposem un model concret per a $(NP/poly) \cap (coNP/poly)$ que intenta aclarir aquest problema: els circuits generadors forts. Donem algunes definicions alternatives d'aquests circuits que no afecten la seva potència, la qual cosa confirma la naturalitat de la classe.

No sembla fàcil trobar cap model similar per a $(NP \cap coNP)/poly$. D'altra banda, la classe relativitzada amb que es correspon (les màquines uniformement fortes) és menys intuitiva que les màquines fortes. Aquestes dues indicacions suggereixen que caldrà algun altre enfoc per a la interpretació i manipulació d'aquesta classe.

Hi ha alguna raó que expliqui aquesta artificiositat de les màquines uniformement fortes? Notem, per exemple, que no hem provat que $\leq_T^P$ i $\leq^{UN}$ siguin diferents. Més concretament, és possible trobar una màquina que realment exploti l'indeterminisme sense

deixar de ser coherent en les seves respostes, sigui quin sigui el seu oracle? Se sap que una pregunta similar és molt difícil de respondre:

Book, Long i Selman (1985) defineixen les màquines "fortes, confluents i madures" ("Strong, Confluent and Mature"), que són màquines uniformement fortes amb dues restriccions addicionals. Si denotem per NPSCM() la classe definida per aquestes màquines, els autors proven que

$$\exists B(P(B) \neq NPSCM(B)) \iff P \neq NP \cap coNP$$

És a dir, exhibir una màquina que sigui forta, confluent i madura i que no tingui una equivalent determinista és provar $P \neq NP \cap coNP$.

Si aquest resultat es pogués demostrar eliminant les dues condicions addicionals (confluència i maduresa), explicaria perquè no hem trobat cap màquina uniformement forta que no es pugui transformar trivialment en una determinista. No obstant, refer la demostració de Book, Long i Selman sense imposar aquestes condicions no sembla fàcil.

REFERÈNCIES

L. Adleman, K. Manders: "Reducibility, randomness, and intractability", 9th ACM STOC (1977), 151–163.

R. Book, T. Long, A. Selman: "Qualitative relativizations of complexity classes", *J. Comp. Sys. Sci.* 30 (1985), 395–413.

M. Garey, D. Johnson: *Computers and intractability: a guide to the theory of NP-completeness.* Freeman 1979.

J. Hartmanis: "On sparse sets in NP-P", *Inf. Proc. Lett.* 16 (1983), 55–60.

J. Hopcroft, J. Ullman: *Introduction to automata theory, languages, and computation.* Addison-Wesley 1979.

R. Karp, R. Lipton: "Some connections between nonuniform and uniform complexity classes", 12th ACM STOC (1980), 302–309.

T. Long: "Strong nondeterministic polynomial time reducibilities", *Theor. Comp. Sci.* 21 (1982), 1-25.

N. Pippenger: "On simultaneous resource bounds", 20th IEEE FOCS (1979), 307–311.

U. Schöning: "On small generators", *Theor. Comp. Sci.* 34 (1984), 337–341.

L. Stockmeyer: "The polynomial time hierarchy", *Theor. Comp. Sci.* 3 (1977), 1–22.

C. Yap: "Some consequences of nonuniform conditions on uniform classes", *Theor. Comp. Sci.* 27 (1983), 287–300.

ÍNDEX