Estratto da

C. Bernardi (a cura di), *Atti degli incontri di logica matematica* Siena 7-8-9 gennaio 1982, 15-16-17 aprile 1982, 4-5-6 giugno 1982.

Disponibile in rete su http://www.ailalogica.it

COMPLESSITA' DELLE TEORIE

GABRIELE LOLLI

La teoria della complessità, che qui presentiamo come capitolo della logica più che dell'analisi degli algoritmi, mira a delimitare all'interno del decidibile un confine tra intrattabile e praticamente trattabile, analogo a quello tra indecidibile e decidibile.

I primi tentativi in questa direzione sono connessi allo studio delle gerarchie subricorsive. Ricordiamo ad esempio la gerarchia $\{\mathcal{E}^n\}$ di Grczegorcwzyk per le funzioni ricorsive primitive, dove ogni classe contiene il successore ed è chiusa rispetto a trasformazioni esplicite, ed $\mathcal{E}^1$ la somma, $\mathcal{E}^2$ la moltiplicazione, $\mathcal{E}^3$ l'esponente, e in generale $\mathcal{E}^n$ la n-esima funzione della spina della funzione di Ackermann. $\mathcal{E}^3$ è la classe delle funzioni elementari.

Tale, e analoghe, classificazioni sono basate sull'ordine di gradezza delle funzioni; ma nella teoria della complessità astratta si dimostra che esistono funzioni caratteristiche, a avalori 0 e 1, di complessità arbitraria. Le classificazioni devono perciò riferirsi al costo dell'ottenimento dei risultati e non solo alla grandezza degli stessi.

Per $n \geqslant 3$ sono equivalenti:

- $f \in \mathcal{E}^n$
- f è computabile in tempo limitato da una $g \in \mathcal{E}^n$
- f è computabile in spazio limitato da una $g \in \mathcal{E}^n$

ma per $n < 3$ le relazioni sono più intricate.
(Tempo e spazio qui sopra si riferiscono a macchine di Turing, ma più in generale a ogni modello di computazione aritmetizzato, come le MT, in $\mathcal{E}^2$.)

$\mathcal{E}^2$ contiene esattamente le funzioni computabili, mediante MT, con uno spazio funzione lineare della lunghezza dell'input. $\mathcal{E}^3 - \mathcal{E}^2$ è stata gerarchizzata da Ritchie nei livelli delle funzioni "prevedibilmente calcolabili". (Per maggiori dettagli si veda [1].)

$\mathcal{E}^3$ è una classe naturale, ma per vari motivi è emersa come centrale la classe, individuata da Cobham

[3] ,delle funzioni calcolabili in tempo limitato da un polinomio nella lunghezza degli argomenti. Il fatto è che il dominio del praticamente trattabile dovrebbe essere ancora più ristretto,ma occorrono un minimo di proprietà di chiusura che non si ottengono se si traccia il confine poniamo tra il terzo e il quarto grado. Le misure di complessità più naturali associate ai diversi modelli di computabilità,e le simulazioni reciproche tra questi modelli sono ottenibili con trasformazioni polinomiali (nel senso sopra detto),di grado basso.

Gli esempi di funzioni arbitrariamente complesse che si danno nella teoria generale della computabilità non hanno interesse,nonchè pratico,neppure logico in quanto derivano dalle solite tecniche poco "naturali" della aritmetizzazione e diagonalizzazione. I risultati noti per teorie "naturali" si ottengono con approcci diretti. In generale per la valutazione della complessità di un problema,di un metodo di decisione per un problema o una teoria,si cercherà di stabilire confini superiori e confini inferiori,il più possibile vicini. I confini superiori derivano di solito da una analisi della dimostrazione stessa di decidibilità del problema,dalla valutazione degli algoritmi esibiti o dalla costruzione di algoritmi più efficienti. Per ottenere confini inferiori, e cioè l'affermazione che ogni algoritmo per il problema deve avere almeno un certo grado di complessità, si tenta di ripetere,opportunamente modificati,gli argomenti classici delle dimostrazioni di indecidibilità. Poi c'è naturalmente la tecnica delle interpretazioni, che qui devono essere efficienti.

Siccome dato un algoritmo qualunque è possibile definire una misura di complessità rispetto a cui questo algoritmo abbia misura (costo) zero, non si possono ottenere risultati assoluti che valgano per tutte le misure. D'altra parte non è consigliabile fissarsi su di una sola misura connessa a un particolare modello di computabilità (anche se ad esempio lo spazio per MT è grosso modo un confine inferiore per quasi tutte le altre misure più naturali). Bisognerà perciò individuare classi non ristrette di misure naturali.

Nelle tre lezioni previste discuteremo i seguenti argomenti:

1) un metodo abbastanza generale,derivato dal problema della fermata,per ottenere confini inferiori
2) un metodo abbastanza generale,derivato dai giochi di Ehrenfeucht a opera di Ferrante e Rackoff,con cui si ottengono tutti i confini superiori noti
3) il problema P = NP dal punto di vista dei sistemi di prova per il calcolo proposizionale.

Per 1) e 2) rispettivamente seguiremo l'esposizione in [11] e [7], suggerendo come lettura anche [12], [13] e [14] , e ci soffermeremo solo come esempio sull'aritmetica di Presburger. Per 3) una introduzione può essere rappresentata da [8], [4] e [5].

Prima lezione

Definizione. Un sistema di programmazione accettabile è una enumerazione $\{\varphi_i\}$ che include tutte le funzioni ricorsive parziali a un argomento,per cui esiste una funzione universale che è ricorsiva parziale

$$\varphi_{univ}(i,x) \simeq \varphi_i(x)$$

e una funzione di composizione c

$$\varphi_{c(i,j)} = \varphi_i \circ \varphi_j$$

che è ricorsiva totale.

(quando con le usuali codifiche si passerà alle funzioni di più argomenti si supporrà anche tacitamente che c sia tale che

$$\varphi_{c(i,j)}(\bar{x},\bar{y}) \simeq \varphi_i(\varphi_j(\bar{x}),\bar{y})$$

per $\bar{x}$ e $\bar{y}$ n-uple.)

Teorema di Rogers. Per ogni due sistemi accettabili $\{\varphi_i\}$ e $\{\psi_j\}$ esiste una f ricorsiva totale biiettiva tale che $\varphi_i = \psi_{f(i)}$ per ogni i.

Per ogni sistema accettabile vale il teorema s-m-n, il teorema di ricorsione, il teorema del punto fisso.

Definizione. Dato un sistema accettabile $\{\varphi_i\}$, una enumerazione $\{\Phi_i\}$ di funzioni ricorsive parziali

è una misura di complessità computazionale per $\{\varphi_i\}$ se

1) $dom(\varphi_i) = dom(\Phi_i)$ per ogni i

2) $\Phi_i(x) \leq y$ è un predicato ricorsivo di i,x e y.

Esempi. Se gli indici i sono gödeliani di MT, $\Phi_i(x)$ può essere il numero dei passi eseguiti dalla macchina sull'input x (tempo); oppure lo spazio usato nel calcolo, inteso come numero di caselle distinte visitate, se la macchina si ferma, indefinito altrimenti; se gli i sono gödeliani di macchine a registri, la complessità può essere il massimo numero registrato in qualche registro nel corso della computazione; se gli i sono programmi di un linguaggio evoluto, $\Phi_i(x)$ può essere il numero di volte che certe istruzioni cruciali sono applicate. Ci sono poi molti esempi artificiali, alcuni dei quali saranno esclusi dalla successiva definizione.

Tutte le misure di complessità sono ricorsivamente relate, nel senso che se $\{\Phi_i\}$ e $\{\Psi_j\}$ sono due misure rispettivamente su $\{\varphi_i\}$ e $\{\psi_j\}$, allora esiste una funzione ricorsiva totale r tale che per ogni i

$$\Phi_i(x) \leq r(x, \Psi_{f(i)}(x)) \quad \text{e} \quad \Psi_{f(i)}(x) \leq r(x, \Phi_i(x))$$

per quasi tutti gli x (q.t.x), dove f è la funzione dell'isomorfismo di Rogers.
Per singole coppie di misure può essere importante, per il trasporto di risultati, conoscere esplicitamente r, come vedremo in un esempio in seguito.

$|x|$ è la lunghezza della parola che rappresenta x; per i numeri naturali di solito è $x+1$ o $lg_2 x$ a seconda che si usi la base unaria o binaria. Analogamente $|\bar{x}|$ per $\bar{x}$ vettore.

Definizione. Una misura di complessità $\{\Phi_i\}$ su $\{\varphi_i\}$ si dice linearmente limitata se $\Phi_i(\bar{x}) \geq |\bar{x}|$ per ogni i e $\bar{x}$ (pensando alle funzioni φ_i come funzioni a più argomenti), ed esistono funzioni ricorsive totali

s e c e una costante intera k tale che per tutti gli i j e z :

1) $\Phi_{c(i,j)}(\bar{x},\bar{y}) \leq k[\Phi_i(\varphi_j(\bar{x}),\bar{y}) + \Phi_j(\bar{y})]$ q.t.$\bar{x},\bar{y}$

2) $\Phi_{s(i,z)}(\bar{x}) \leq k\,\Phi_i(z,\bar{x})$ q.t.$\bar{x}$.

Nei sistemi accettabili più usuali composizione e funzione s^1_n possono essere ottenute descrivendo semplici manipolazioni di programmi, e non è difficile verificare che certe misure sono linearmente limitate.

Problema della fermata ristretto. Si ipotizza che un problema della fermata ristretto dovrebbe essere un problema di intrinseca difficoltà computazionale, relativamente al tipo di restrizione, e che potrebbe fornire un metodo generale di dimostrazione di confini inferiori, esattamente come al problema della fermata sono riducibili i risultati di indecidibilità.
Se t è una funzione ricorsiva totale , e vogliamo mostrare che l'appartenenza a un insieme ricorsivo S ha sempre complessità almeno $t(|x|)$ per infiniti x, possiamo provare a cercare delle parole R(i,x) per cui

$$R(i,x) \in S \quad \text{sse} \quad \Phi_i(x) > t(|x|) .$$

Infatti allora: se le parole R(i,x) sono ragionevolmente corte, se il costo di produrle in funzione di i e x è ragionevolmente basso, ne segue che la complessità della decisione dell'appartenenza ad S non dovrebbe essere in generale minore di t , perchè se lo fosse ci sarebbe un metodo per decidere se $\Phi_i(x) > t(|x|)$ con complessità minore di t (verificando cioè $R(i,x) \in S$). Ora questo non dovrebbe essere possibile perchè non si vede come si possa in generale decidere $\Phi_i(x) > t(|x|)$ se non facendo girare il programma i e osservando l'uso della risorsa in questione.

Definizione. Sia t una funzione ricorsiva totale, ed S un insieme ricorsivo. Si dice che S esprime il problema della fermata ristretto a t , o t-ristretto, se esiste un programma r tale che per ogni i $\varphi_r(i,x)$ è

1-1 come funzione di x e

1) $\varphi_r(i,x) \in S$ sse $\Phi_i(x) > t(|x|)$ per ogni i,x

2) $|\varphi_r(i,x)| \leq k_i|x|$ q.t. x (k_i dipende da i)

3) per ogni k, $\Phi_r(i,x) \leq t(|\varphi_r(i,x)|/k)$ q.t.x .

(Si dice che S esprime fortemente se le costanti k_i non dipendono da i; 3) significa che il costo per produrre R(i,x) , cioè $\varphi_r(i,x)$, è molto minore di t o ha ordine di infinito minore.)

Definizione. Una funzione t si dice almeno lineare se per ogni x è $t(x) \leq t(x+1)$ e per ogni k

$$kt(x) \leq t(kx) \quad \text{q.t.x} .$$

Teorema. Sia $\{\Phi_i\}$ una misura di complessità linearmente limitata su $\{\varphi_i\}$ e t una funzione ricorsiva totale almeno lineare. Sia S un insieme ricorsivo che esprime il problema della fermata t-ristretto in relazione alla misura $\{\Phi_i\}$ e d una qualunque procedura di decisione parziale per S ($\varphi_d(y)$ definito sse $y \in S$). Allora esiste un i tale che $R(i,x) \in S$ per ogni x e tale che per qualche intero positivo k

$$\Phi_d(R(i,x)) > t(|R(i,x)|/k) \quad \text{q.t. x} .$$

Dimostrazione. Diamo solo un cenno informale. (Si usano procedure parziali, comunque più generali che le procedure di decisione, solo per comodità in certi passaggi.)
Nel caso del problema della fermata si considera l'insieme $T = \{ \langle i,x \rangle : \varphi_i(x)$ indefinito$\}$ e si osserva che se T fosse ricorsivo allora tale sarebbe anche la funzione f per cui

$$\varphi_{f(i)}(x) \simeq \begin{cases} 1 & \text{se } \langle i,x \rangle \in T \\ \text{indef.} & \text{altrimenti} . \end{cases}$$

Sia i un punto fisso per f, allora:
$\langle i,x \rangle \in T$ sse $\varphi_{f(i)}(x)$ defin. sse $\varphi_i(x)$ defin.

sse $\langle i,x \rangle \notin T$. Il programma i grosso modo dice "mi fermo sse non mi fermo". Nell'ipotesi del teorema ora rimpiazziamo in questo ragionamento $\langle i,x \rangle \in T$ con $\varphi_d(R(i,x))$ definito, cioè con $\Phi_i(x) > t(|x|)$; otteniamo come sopra un i per cui

$$\varphi_i(x) \simeq \begin{cases} 1 & \text{se } \varphi_d(R(i,x)) \text{ defin.} \\ \text{indef.} & \text{altrimenti} \end{cases}$$

cioè $\varphi_i(x)$ converge sse $R(i,x) \in S$. Ma $R(i,x) \in S$ per tutti gli x: infatti $R(i,x) \notin S$ implica $\Phi_i(x) \leq t(|x|)$ per definizione di S , in particolare $\varphi_i(x)$ definito; ma per definizione di i $R(i,x) \notin S$ implica $\varphi_i(x)$ indefinito, contraddizione.

Ora R(i,x) si ottiene a basso costo; se il teorema di ricorsione che dà i non aumenta molto la complessità, a partire dai programmi dati, abbiamo che $\Phi_i(x)$ è piccolo sse $\Phi_d(R(i,x))$ è piccolo. Così $\Phi_i(x) < t(|x|)$ se $\Phi_d(R(i,x))$ è molto più piccolo di $t(|x|)$. Ma $R(i,x) \in S$ implica che mai si ha $\Phi_i(x) < t(|x|)$ da cui la conclusione. In questo argomento i dice "mi fermo presto sse non mi fermo".

La dimostrazione rigorosa consiste nel sostituire tutte le affermazioni vaghe di grandezza e piccolezza con le opportune disuguaglianze.

Una osservazione sul teorema di ricorsione. Del teorema di ricorsione esiste non solo una versione effettiva, ma anche una con la valutazione della complessità dei programmi ottenuti:

Teorema. Se $\{\Phi_i\}$ è una misura linearmente limitata su $\{\varphi_i\}$ allora

1) per ogni j si può trovare in modo effettivo un n tale che per ogni x $\varphi_n(x) \simeq \varphi_j(n,x)$ e

2) esiste k tale che n può essere scelto in modo che $\Phi_n(x) < k\,\Phi_j(n,x)$ q.t.x.

Dimostrazione. Basta esaminare la dimostrazione ori-

ginaria di Kleene che usa esplicitamente le funzioni s^1_n e c e non la funzione universale, e applicare la proprietà della misura di essere lineare. Si veda anche 15 per molte osservazioni sul diverso stato di ricorsione e punto fisso nella trattazione assiomatica dei sistemi di programmazione.

Applicazione alla teoria dell'addizione (o aritmetica di Presburger). Mostriamo come si fa a vedere che la teoria dell'addizione, che è decidibile, ha complessità almeno esponenziale rispetto al tempo delle MT, cioè che ogni macchina di Turing che decida la teoria dell'addizione richiede un tempo almeno esponenziale, in funzione dell'input, per infiniti inputs. Di fatto la complessità è doppiamente esponenziale, ma la dimostrazione di questo risultato migliore è troppo complicata, e richiede solo fatti nuovi della teoria dei numeri, non della teoria della computabilità.
Siccome non è facile parlare delle macchine nel linguaggio con la sola addizione, dovendosi passare attraverso l'aritmetizzazione, useremo inizialmente una misura diversa, accennando poi a come trasportare i risultati alla misura del tempo.
Come sistema accettabile prendiamo i μ-programmi, cioè usiamo la definizione di μ-ricorsività considerando ogni definizione come un oggetto sintattico di un linguaggio formale, che chiameremo programma. Se P è un tale programma, definiamo $\Phi_P(\bar{x}) \leqslant y$ see P su $\bar{x}$ converge e in ogni eventuale moltiplicazione eseguita nel corso del calcolo i fattori sono sempre $\leqslant y$.
Misure analoghe sono abbastanza naturali nella analisi della complessità di programmi scritti in linguaggi evoluti. Non dimostriamo (esercizio) che Φ_P così definita è una misura, e linearmente limitata. Rispetto a questa misura la teoria dell'addizione ha complessità doppiamente esponenziale.
Con formule dell'addizione si possono descrivere porzioni finite della tavola della moltiplicazione. Se riusciamo a scrivere formule brevi per tavole grandi, allora nella teoria dell'addizione potremo esprimere un

problema della fermata ristretto, perchè nell'aritmetica si esprime il problema della fermata.
Occorrono due trucchi, di applicazione molto generale, per abbreviare in modo logicamente equivalente formule lunghe:
1) Supponiamo di voler scrivere che

$$\exists u\, \exists v\, \exists w\, \exists x\, [F(u) \wedge F(v) \wedge F(w) \wedge F(x) \wedge G(u,v,w,x,y)]$$

dove F è una formula lunga, qui con quattro ripetizioni. Ci possiamo ricondurre a una sola occorrenza di F scrivendo

$$\exists u\, \exists v\, \exists w\, \exists x\, \forall t\, [((u=t \vee v=t \vee w=t \vee x=t) \rightarrow F(t)) \wedge G(u,v,w,x,y)] .$$

2) Rinomina delle variabili sia libere che vincolate, ma con il massimo possibile di confusione, per usare il minor numero possibile di variabili diverse. Si ricordi che siccome in questo contesto tutti gli alfabeti devono essere finiti, l'i-esima variabile comunque realizzata (ad esempio x seguita da i apici) viene ad avere lunghzza dello stesso ordine di i.
Entrambi i trucchi si usano nella dimostrazione del

Teorema. Per ogni naturale k esiste una formula della addizione $M_k(x_1,x_2,x_3)$ tale che per ogni m,n,p

$$M_k(\underline{m},\underline{n},\underline{p}) \quad \text{sse} \quad mn=p \text{ e } m < 2^{2^{2k}} ;$$

inoltre esiste una costante positiva c tale che

$$|M_k| \leqslant c(k+1)$$

e ogni M_k ha undici variabili vincolate, per $k > 0$.

Ammesso questo teorema, si dà una traduzione dei μ-programmi in formule dell'aritmetica dell'addizione, con la differenza rispetto alla solita rappresentazione che la moltiplicazione non è rappresentata da una formula ma da infinite formule, cioè per ogni k da $M_k(x,y,z) \wedge M_k(y,x,z)$. Allora per ogni k abbiamo traduzioni $\mathrm{Trad}_k(P)$ dei programmi P per cui per ogni $\bar{m}$ e q

$\mathrm{Trad}_k(P)(\underline{\bar{m}},\underline{q})$ è vero sse $P(\bar{m}) = q$ e $\Phi_P(\bar{m}) \leqslant 2^{2^{2k}}$

e inoltre $|\mathrm{Trad}_k(P)| \leq d|M_k|$ per qualche costante d dipendente da P.
La traduzione è eseguibile da una MT in tempo cubico.

Se ora $t(k) = 2^{2^{2k}} - 1$, t è almeno lineare e

Teorema. La teoria dell'addizione esprime il problema della fermata t-ristretto rispetto a $\{\Phi_P\}$.

Dimostrazione. Per ogni programma P a un argomento e ogni k, la parola R(P,k) sia

$$\sim \exists x \exists y(\mathrm{Trad}_k(P)(x,y) \wedge x=\underline{k}) .$$

Si ha subito che R(P,k) sse $\Phi_P(k) > t(|k|)$. Le altre verifiche sono banali salvo quella che R(P,k) può essere prodotta a basso costo rispetto alla nostra misura. Abbiamo detto sopra cosa comporta in termini di tempo. E' ora di accennare alle relazioni tra queste due misure. In generale sull'argomento si veda [11], cap.2. La relazione che lega le due misure è esponenziale, cioè per ogni macchina esiste un programma P tale che Φ_P è minore o uguale a 2 elevato a un polinomio nel tempo della macchina, e P è un programma per la stessa funzione computata dalla macchina.
Allora siccome t è doppiamente esponenziale e la complessità di produrre R è esponenziale ne viene che le condizioni sono soddisfatte, e nello stesso tempo si può anche concludere con la seguente versione

Teorema. Per ogni MT che dia una decisione parziale per l'aritmetica dell'addizione esiste una costante k e infiniti enunciati φ della teoria per cui il tempo di decisione è maggiore di 2 elevato a $|\varphi|/k$.

Tornando alle formule M_k ora, si procede per induzione. Ovviamente M_0 è $(x_1=\underline{0} \wedge x_3=\underline{0}) \vee (x_1=1 \wedge x_2=x_3)$.
Ammesso di avere M_k si osservi che

$$2^{2^{2k+1}} = (2^{2^{2k}})^2$$

e che siccome $(n+1)^2 = n^2+2n$ ogni numero minore di

$(n+1)^2$ si può scrivere, non in modo unico, come somma x^2+y+z con $x,y,z < n+1$, e viceversa. Possiamo allora scrivere una formula per la moltiplicazione $xy=z$,

con $x < 2^{2^{2k+1}}$ ponendo

$$\exists \bar{u} \exists s \exists \bar{p}(M_k(u_1,u_1,s) \wedge x_1=(s+(u_2+u_3) \wedge M_k(u_1,x_2,P_1) \wedge$$
$$\wedge M_k(u_1,P_1,P_2) \wedge M_k(u_2,x_2,P_3) \wedge M_k(u_3,x_2,P_4) \wedge$$
$$\wedge x_3=(P_2+(P_3+P_4))).$$

Ma se questa, o meglio quella che si ottiene iterando due volte la fomazione di questa formula per ottenere la limitazione voluta per k+1 fosse M_{k+1} avremmo $|M_{k+1}| \geq 25|M_k|$ (ci sono 5 ripetizioni di M_k) e una crescita esponenziale della lunghezza.
Si usano allora i due trucchi di sopra e si ottiene il risultato voluto; i dettagli si possono leggere in [11].

Anche la dimostrazione di Meyer che ogni procedimento di decisione per il calcolo dei predicati monadici (anche rispetto a formule con solo 7 quantificatori) richiede tempo esponenziale è basata sulla possibilità di esprimere con una formula corta, di lunghezza $n^2 \lg n$, e precisamente $\forall x(\bigwedge_1^n \exists y D_i(x,y))$, dove

$D_i(x,y)$ è $(P_i(x) \leftrightarrow \sim P_i(y)) \wedge \bigwedge_{j \neq i}(P_j(x) \leftrightarrow P_j(y))$

il fatto che in ogni struttura con n predicati tutte le 2^n combinazioni di predicati sono realizzate, cioè la struttura ha cardinalità 2^n.

Seconda lezione

Dato un linguaggio L e una classe C di strutture del

tipo di L, si cerca di associare a ogni struttura una norma, e pure una norma a ogni elemento di ogni struttura, nella prospettiva che poi la validità in C, o in ogni struttura di C, possa essere ridotta alla validità nelle strutture di norma fissata, possibilmente in numero finito, e la validità in una struttura possa essere verificata facendo variare i quantificatori sugli elementi di una certa norma fissata, di nuovo possibilmente in numero finito.
Le norma saranno in genere numeri naturali, ma in una impostazione generale basta che siano parole su alfabeti finiti Σ, con una relazione di ordine parziale $\preccurlyeq$ (si ammette anche una norma ∞, che per non appesantire le notazioni supporremo in Σ^* anch'essa).
Scriviamo $\bar{a}_k$ per indicare una k-upla $a_1,\dots,a_k$.

Definizione. Sia $\mathcal{A}$ una struttura e $H: \mathbb{N}\times\mathbb{N}\times\Sigma^* \to \Sigma^*$. Si dirà che $\mathcal{A}$ è H-limitata se per ogni $n,k\in\mathbb{N}$ e $m\in\Sigma^*$, per ogni formula $F(x_1,\dots,x_{k+1})$ con "profondità di quantificazione" $\leq n$ e ogni k-upla $\bar{a}_k\in A^k$ con $|a_i| \preccurlyeq m$ per $i=1,\dots,k$, se $\mathcal{A}\models \exists x_{k+1} F[\bar{a}_k]$ allora esiste $a_{k+1}\in A$ con $|a_{k+1}| \preccurlyeq H(n,k,m)$ tale che $\mathcal{A}\models F[\bar{a}_{k+1}]$.
(Nella definizione $\bar{a}_{k+1}$ è ovviamente la sequenza $a_1,\dots,a_k,a_{k+1}$; profondità di quantificazione è il numero di blocchi di quantificatori nel prefisso della formula supposta in forma normale prenessa, o qualcosa di simile; la chiameremo per brevità q-profondità.)

Teorema. Se $\mathcal{A}$ è H-limitata e $Q_1x_1\dots Q_kx_kF$ è un enunciato di q-profondità $\leq n+k$ e $m_0 \preccurlyeq \dots \preccurlyeq m_k \in \Sigma^*$ e $H(n+k-i,i-1,m_{i-1}) \preccurlyeq m_i$, $i=1,\dots,k$, allora $\mathcal{A}\models Q_1x_1\dots Q_kx_kF$ sse $\mathcal{A}\models Q_1x_1 \preccurlyeq m_1 \dots Q_kx_k \preccurlyeq m_k F$ (dove nei prefissi $x_i \preccurlyeq m_i$ vuol dire $|x_i| \preccurlyeq m_i$).
Il teorema, che non dimostriamo come quasi tutto in questa lezione, trattandosi sempre di lunghe dimostrazioni per induzione sulla complessità delle formule, afferma che se $\mathcal{A}$ è H-limitata i quantificatori si possono far variare solo sugli elementi di norma opportunamente limitata.

Definizione. Per una classe C di strutture, $\mathcal{A},\mathcal{B}\in C$, $n,k\in\mathbb{N}$, $\bar{a}_k\in A^k$, $\bar{b}_k\in B^k$, si definisce

$$\langle \mathcal{A},\bar{a}_k\rangle \equiv_{n,k} \langle \mathcal{B},\bar{b}_k\rangle$$

sse per ogni formula $F(x_1,\dots,x_k)$ di q-profondità $\leq n$, $\mathcal{A}\models F[\bar{a}_k]$ sse $\mathcal{B}\models F[\bar{b}_k]$.

Teorema. (Definizione originaria induttiva di Fraissé ed Ehrenfeucht delle relazioni $\equiv_{n,k}$) Nella situazione della definizione, $\langle \mathcal{A},\bar{a}_k\rangle \equiv_{n+1,k} \langle \mathcal{B},\bar{b}_k\rangle$ sse
1) per ogni $a_{k+1}\in A$ esiste $b_{k+1}\in B$ per cui
$\langle \mathcal{A},\bar{a}_{k+1}\rangle \equiv_{n,k+1} \langle \mathcal{B},\bar{b}_{k+1}\rangle$ e
2) viceversa.

Si dimostra in genere la h-limitatezza utilizzando dei raffinamenti delle relazioni $\equiv_{n,k}$.

Teorema. Data H, supponiamo di avere sulle strutture della classe C delle relazioni $E_{n,k}$ che soddisfano
1) $\langle \mathcal{A},\bar{a}_k\rangle E_{0,k} \langle \mathcal{B},\bar{b}_k\rangle$ sse $\langle \mathcal{A},\bar{a}_k\rangle \equiv_{0,k} \langle \mathcal{B},\bar{b}_k\rangle$
2) Se $\langle \mathcal{A},\bar{a}_k\rangle E_{n+1,k} \langle \mathcal{B},\bar{b}_k\rangle$ e $|b_i| \preccurlyeq m$ per $i=1,\dots,k$, allora per ogni $a_{k+1}\in A$ esiste $b_{k+1}\in B$ con $|b_{k+1}| \preccurlyeq H(n,k,m)$ per cui $\langle \mathcal{A},\bar{a}_{k+1}\rangle E_{n,k+1} \langle \mathcal{B},\bar{b}_{k+1}\rangle$.
Allora $E_{n,k} \subseteq \equiv_{n,k}$ per ogni n e k, e inoltre ogni struttura in C è H-limitata.

Applicazione alla teoria dell'addizione. Nella struttura degli interi relativi la norma è data dal valore assoluto. Si possono definire le $E_{n,k}$ in modo che significhino grosso modo che $\bar{a}_k$ e $\bar{b}_k$ soddisfano le

stesse disuguaglianze lineari e hanno certe proprietà di divisibilità in comune, e si ottiene il

Teorema. Esiste c per cui $\mathbb{Z}$ è H-limitata, dove

$$H(n,k,m) = (m+1)2^{2^{c(n+k)}}$$

(in realtà la classe dei modelli di Th($\mathbb{Z}$) è H-limitata).

Per la dimostrazione si introducono gli insiemi

$V_0 = \{-2,-1,0,1,2\}$

$V_i' = \{ \delta/v \cdot v' \mid \delta = \text{m.c.m.}V_i \; ; v,v' \in V_i ; v \neq 0 \}$

$V_{i+1} = V_i \cup \{ a+b \mid a,b \in V_i' \}$

e per $n,k \in \mathbb{N}$ si definisce $E_{n,k}$ su $\mathbb{Z}^k$ così: se $\delta = \text{m.c.m.}V_n$ allora $\bar{a}_k \, E_{n,k} \, \bar{b}_k$ sse per ogni $v_1,\ldots,v_k \in V_n$ e ogni v, con $|v| \leq \delta^2$, si ha

1) $v + \sum_k v_i a_i \leq 0$ sse $v + \sum_k v_i b_i \leq 0$ e

2) $a_i \approx b_i \bmod \delta^2$ per $i=1,\ldots,k$.

Per verificare le proprietà volute, e vedere come saltano fuori certi esponenziali, si osservi che $\text{card}(V_n) \leq 2^{2^{cn}}$ e $\max V_n \leq 2^{2^{2^{cn}}}$ e simili. La verifica richiede alcuni calcoli.
Si deduce poi

Teorema. Th($\mathbb{Z}$) può essere decisa da una MT che lavora con spazio limitato da $2^{2^{cn}}$, n lunghezza dell'input, per qualche c fissato.

Dimostrazione. Ogni formula F si trasforma in una forma prenessa $Q_1x_1 \ldots Q_kx_kG$ in tempo e spazio polinomiali. L'enunciato risulta equivalente a $Q_1x_1 \leq m_1 \ldots Q_kx_k \leq m_kG$ se si prendono m_i che soddisfano $m_i \geq H(k-i,i-1,m_{i-1})$ per $i=1,\ldots,k$, ad esempio per

$m_i = 2^{2^{2^{ck+i}}}$, $i = 0,\ldots,k$. Ora per ogni $i \leq k$ si riservano $2^{2^{2^{ck+i}}}+2$ caselle perchè ivi in binario ogni numero $2^{2^{2^{ck+i}}}$ può essere scritto. Si cicla poi su questi spazi testando G sulle differenti k-uple, e si vede facilmente che per un opportuno c bastano per questo lavoro $2^{2^{ck}}$ caselle.

Con la stessa tecnica si può migliorare, come è fatto in [7], il risultato di Feferman e Vaught sulla decidibilità della teoria della potenza debole di una struttura decidibile, che nella dimostrazione originaria esibiva una procedura non efficiente.
Se $\langle A, R_1, \ldots, R_l, e \rangle$, con $\text{ar-}R_i = t_i$, è una struttura con almeno una costante, la sua potenza debole $\langle A^*, R_1^*, \ldots, R_l^*, e^* \rangle$ è definita prendendo $A^* = \{ f: \mathbb{N} \to A \mid f(i) \neq e \text{ solo per finiti } i \}$ e definendo le relazioni punto per punto.
Se A ha una norma, anche A^* ne ha una naturale, $|f| = \max(\{ i \in \mathbb{N} \mid f(i) \neq e \} \cup \{ |f(i)| \mid i \in \mathbb{N} \})$.
Se A è H-limitata, A^* è H^*-limitata, dove H^* è definita in modo un po' complicato ma controllabile: si indica con M(n,k) il numero delle $\equiv_{n,k}$-classi di equivalenza (per M esistono valutazioni generali $\left\{ 2^{\cdot^{\cdot^{2^{(n+k)}}}} \right\}_{n+1}$, per cui $M(n,k) \leq 2^{\left\{ 2^{\cdot^{\cdot^{2^{(n+k)}}}} \right\}_{n+1}}$, ma in molti casi concreti è esplicitamente calcolabile), poi si introduce $\mu(n,k) = \prod_1^n M(n-i,k+i)$ e infine

$$H^*(n,k,m) = \max \{ H(n,k,m), m + \mu(n+1,k), |e| \} .$$

Siccome $\langle \mathbb{N}^*, +^* \rangle$ è isomorfa a $\langle \mathbb{N}^+, \cdot \rangle$, si trova per questa via anche un confine superiore per la moltiplicazione di interi:

$Th(\langle \mathbb{N}^+, \cdot \rangle)$ è decidibile in spazio $\leq 2^{2^{2^{cn}}}$.

Analogamente si ottengono risultati sulla teoria dei gruppi abeliani finiti. Sia rinvia a [7] per un quadro dei risultati noti.

Terza lezione

Parte integrante dell'inizio di questa lezione è il cap. 7 di [2]. Per un approfondimento si veda poi [8]. (Si noti soltanto una confusione nella presentazione in [2], tra la definizione che precede il teorema 7.1 e quella che precede 7.2; la distinzione è discussa in modo approfondito in [8]. La nozione che si usa nella definizione di NP-completezza è quella di riducibilità polinomiale, la seconda di [2] e non quella di Turing-riducibililità in tempo polinomiale.)
Dati due linguaggi $L_1 \subseteq \Sigma_1^*$ e $L_2 \subseteq \Sigma_2^*$ si dice che L_1 si riduce, o si trasforma polinomialmente a L_2, $L_1 \leq_p L_2$, se esiste una funzione $f: \Sigma_1^* \to \Sigma_2^*$ tale che

1) $x \in L_1$ sse $f(x) \in L_2$ e
2) f è computabile in tempo polinomiale.

Se $L_1 \leq_p L_2$ e $L_2 \in P$ allora $L_1 \in P$.
Un linguaggio L è NP-completo se:

1) $L \in NP$ e
2) per ogni $L' \in NP$, $L' \leq_p L$.

Teorema.(Cook) SAT è NP-completo.

Se un problema (un linguaggio) è in NP e non dimostrabilmente in P lo è perchè la sua soluzione nota consiste in una ricerca su di un insieme di dimensioni esponenziale, in funzione dell'input, dove la verifica di ogni singolo caso è fattibile in tempo polinomiale. Se si pensa che nessun algoritmo possa essere essenzialmente diverso da questa ricerca, allora il complemento del problema non dovrebbe essere neppure in NP, perchè per decidere il complemento occorre esaminare tutte le possibilità. Questo ragionamento è plausibile, però si ha soltanto, dimostrabilmente, che P=NP implica NP=coNP e non viceversa.

Teorema. NP=coNP sse esiste un problema NP-completo il cui complemento è in NP.
In particolare
Teorema. NP=coNP sse TAUT è in NP.
Dimostrazione. TAUT è in coNP perchè SAT è in NP, dunque se NP=coNP anche TAUT è in NP. Viceversa: per il teorema di Cook, per ogni L esiste, se L è in NP, una f polinomiale tale che $x \in L$ sse $f(x)$ non è una tautologia. Ammesso che TAUT sia in NP, un procedimento polinomiale non deterministico per il complemento di L è: per ogni x calcolare $f(x)$, e applicare il procedimento per TAUT.

Possiamo allora trasformare insensibilmente la terminologia computazionale in una logica, introducendo la

Definizione. Per un linguaggio $L \subseteq \Sigma^*$, un sistema di prova per L è una funzione $f: \Sigma_1^* \longrightarrow\!\!\!\!\!\rightarrow L$, per qualche alfabeto Σ_1, calcolabile in tempo polinomiale.

Un sistema di prova per L è polinomialmente limitato, o veloce, se esiste un polinomio p tale che per ogni $y \in L$ c'è un $x \in \Sigma_1^*$ con $f(x)=y$ e $|x| \leq p(|y|)$.

Lemma. L è in NP sse $L = \emptyset$ o L ha un sistema di prova veloce.
Dimostrazione. L'enunciato del lemma corrisponde alla caratterizzazione degli insiemi r.e. come insiemi o vuoti o rango di una funzione ricorsiva. La dimostrazione è solo una riformulazione di altre definizioni: se $L \in NP$ e $L \neq \emptyset$, data M che accetta L in tempo polinomiale non deterministico definiamo f sulle computazioni di M, in modo che $f(x) = y$ se x è una computazione che accetta y, $f(x) = y_o$ L fissato altrimenti. Viceversa se f è un sistema di pro-

va veloce per L , allora un procedimento non deterministico per L consiste nell'indovinare una prova corta x di y e verificare se f(x)=y.

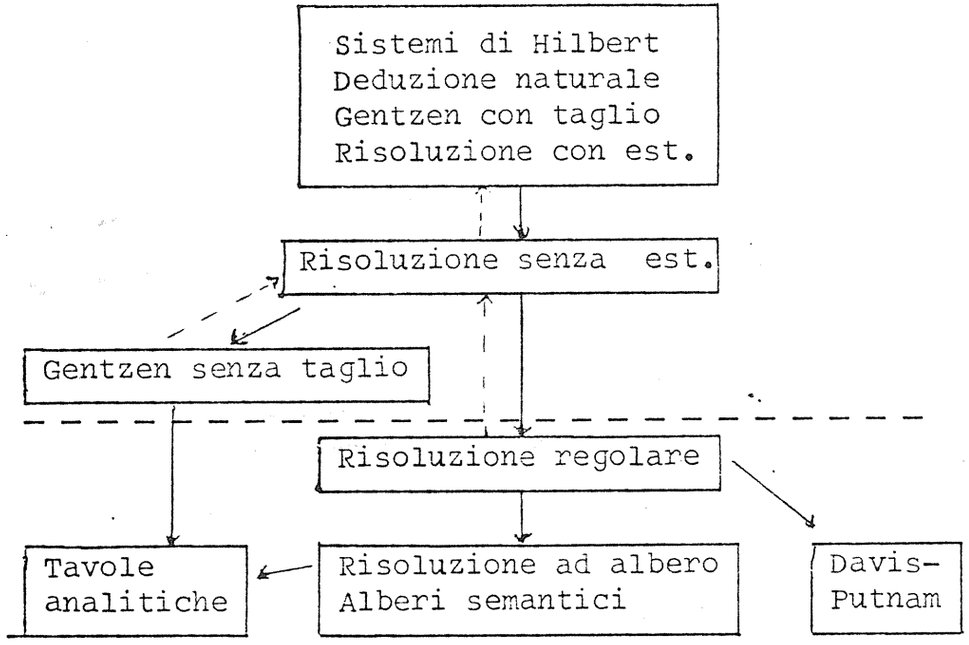
Allora NP=coNP sse TAUT ha un sistema di prova veloce. Esaminiamo i sistemi di prova dati dai calcoli logici classici. Esistono due tipi di risultati:
1) la dimostrazione che certi sistemi di prova non sono veloci
2) rapporti di riduzione, o simulazione, tra un sistema e l'altro.
La nozione di riduzione è formulata qui come simulazione in questo modo.
Definizione. Dati due sistemi di prova $f_1: \Sigma_1^* \twoheadrightarrow L$ e $f_2: \Sigma_2^* \twoheadrightarrow L$ si dice che f_2 simula(polinomialmente) f_1 se esiste $g: \Sigma_1^* \to \Sigma_2^*$ calcolabile in tempo polinomiale che traduce una prova in f_1 in una prova in f_2 dello stesso elemento di L, cioè $f_2(g(x)) = f_1(x)$.

Se f_2 simula f_1 e f_1 è veloce anche f_2 lo è.
Consideriamo sistemi di prova completi, o almeno completi quel tanto che interessa: se ad esempio invece delle tautologie trattiamo le contraddizioni allora abbiamo bisogno di sistemi (di refutazione più che di prova) completi rispetto alla refutazione; è chiaro che è equivalente considerare sistemi di refutazione oppure di prova; la completezza che basta è quella della derivazione di una contraddizione da una teoria inconsistente, non necessariamente la derivazione di ogni cosa.
Tra i sistemi che sono stati studiati ci sono i seguenti.
- Sistemi di Hilbert: sono i soliti sistemi dati da assiomi e regole (schematiche), per cui si può dare una definizione generale. In [5] sono chiamati sistemi di Frege, solo per dover poi riconoscere che il sistema di Frege non era un sistema di Frege, per cui tale proposta terminologica sembra balzana (nel

sistema di Frege c'è la regola di sostituzione di tautologie a lettere di proposizioni dimostrate che non ha carattere esclusivamene sintattico ma dipende dalla nozione stessa di derivabilità.
-Deduzione naturale.
-Sistemi di Gentzen con o senza taglio (le derivazioni qui come nella deduzione naturale vengono meglio definite come grafi invece che come alberi in modo da evitare che una volta generato un sequente debba essere ripetuto; lo stesso non si può fare in modo naturale con il seguente sistema).
-Tavole analitiche, alberi semantici e analoghi.
-Risoluzione: è il metodo di base per la dimostrazione automatica. Si usano formule in forma normale congiuntiva; ogni tale formula S è identificata con l'insieme dei suoi congiunti, detti clausole, ogni clausola essendo una disgiunzione di letterali, cioè una variabile o la negazione di una variabile. La regola di risoluzione è: da $A \vee \mathcal{B}$ e $\bar{A} \vee \mathcal{C}$ derivare $\mathcal{B} \vee \mathcal{C}$ dove A è un letterale e $\bar{A}$ il suo complemento e $\mathcal{B}$ e $\mathcal{C}$ sono digiunzioni di letterali.
La regola è sostanzialmente la cut-rule del sistema di Shoenfield, e vi si aggiunge tacitamente la cancellazione della doppia negazione se si definiscono i letterali come abbiamo fatto (altrimenti i letterali sono o le variabili o le negazioni di letterali).
Il sistema è usato per dimostrare la insoddisfacibilità di un insieme S di clausole, perchè è completo nel senso che S è soddisfacibile sse da S non è derivabile la clausola vuota.
-Vari tipi di risoluzione con aggiunta di regole strutturali o di modalità di ordine di applicazione della risoluzione.
-Risoluzione con estensione. La regola dell'estensione che si può anche aggiungere ai sistemi di Hilbert consiste nell'aggiungere (l'equivalente formale di) $p \leftrightarrow \mathcal{A}$, dove p è una nuova lettera che non compare in $\mathcal{A}$, a una derivazione di $\mathcal{A}$.
-Procedura di Davis-Putnam. Precede ed è (o può essere presentata in modo che sia) una variante della risoluzione: dato un insieme di clausole S scegliere un letterale A ed applicare la risoluzione a tutte

le coppie di clausole per cui si elimina A;se si ottiene la clausola vuota S è inconsistente;cancellare le eventuali tautologie così ottenute; alle restanti così ottenute aggiungere tutte quelle che non contenevano A o $\bar{A}$. Se così si ottengono solo tautologie, l'insieme S è consistente; se no si ricomincia.

Si consiglia di leggere [6] come introduzione ai sistemi possibilmente meno noti di risoluzione; in [6] si trovano anche molte osservazioni banali ma utili per vedere dove stanno effettivamente i problemi;ad esempio la validità di una formula in forma normale congiuntiva può essere verificata facilmente,così come la soddisfacibilità di una in forma normale disgiuntiva; ecco perchè per SAT si considerano forme congiuntive e per TAUT forme disgiuntive.Naturalmente si capisce che il problema si sposta alla trasformazione di una formula in forma disgiuntiva conservando soddisfacibilità, e dualmente; Tseitin ha dimostrato che si può fare in modo veloce con l'aggiunta di variabili. Ancora:le formule grasse,cioè formule lunghe con poche variabili,sono decidibili in tempo polinomiale,perchè già esibiscono le diverse possibilità, che altrimenti vanno generate. Sempre in [6] c'è un esempio di quante tautologie inutili vengano generate dalla risoluzione, e come si possa abbreviare il lavoro con opportune regole aggiuntive.
La situazione nota per quel che riguarda i sistemi di sopra, in assoluto o nelle loro relazioni,è riassunta dal seguente diagramma, che però probabilmente richiede qualche correzione o aggiunta (sembra ad esempio che sia stato appena dimostrato che la risoluzione regolare simula la risoluzione).
Nel disegno all'interno di uno stesso quadro stanno sistemi che mutuamente si simulano; una freccia continua significa che i sistemi nel riquadro superiore simulano quelli nei riquadro inferiore; una freccia tratteggiata significa che non si sa se vale la freccia continua; la mancanza di freccia significa che non si ha simulazione in quella direzione.



La linea tratteggiata orizzontale segna,per adesso, la demarcazione tra i sistemi prvabilmente non veloci, quelli al di sotto della linea, e quelli per cui non si ha ancora la risposta.
Il primo risultato negativo,per risoluzione regolare e alberi semantici, è stato ottenuto da Tseitin nel 1968. Per alcuni esempi si rimanda a [5] e [6].

Si chiamano tautologie difficili (relativamente a un sistema di prova) quelle atutologie che costituiscono una smentita della velocità del sistema, cioè una successione di tautologie tali che nel sistema la lunghezza della più corta derivazione delle stesse cresce in modo esponenziale con la lunghezza delle tautologie stesse (ovviamente non è restrttivo supporre che le tautologie siano tutte di lunghezza diversa). Cook ha osservato che ogni volta che si costruisce una successione di tautologie come controesempio a qualche sistema,occorre simultaneamente dimostrare che appunto di tautologie si tratta, e la dimostrazione in genere,proprio per essere informale e facilmente comprensibile,deve avere un carattere

costruttivo;ciò significa che l'argomento informale potrà probabilmente essere trasformato in una prova formale,veloce,in qualche sistema di prova. Ciò indica una difficoltà intrinseca al reperimento di tautologie intrinsecamente difficili, ovvero alla dimostrazione che TAUT non è in NP.
In [10] questa osservazione è stata intelligentemente sfruttata per costruire una situazione la cui via d'uscita sia comunque significativa:tutti gli esempi di tautologie difficili finora costruiti sono ricavati dalla rappresentazione proposizionale di problemi combinatori; ci si rivolge di nuovo allora al solito teorema di Ramsey,finito, e si codificano istanze del teorema nella logica proposizionale. Si ottengono delle proposizioni tali che la prova che si tratta di tautologie richiede la dimostrazione che un certo numero è un numero di Ramsey. Allo stato attuale in combinatoria non si conoscono metodi generali per provare che un numero è un numero di Ramsey: se tali tautologie risultassero intrinsecamente difficili, allora si avrebbe $P \neq NP$; se si trova per esse qualche sistema veloce di prova, allora si ha un avanzamento nella teoria di Ramsey.
Tra gli altri sviluppi possibili e forse accessibili c'è da approfondire lo studio della regola dell'estensione nei sistemi di Hilbert; oppure la nuova strada indicata in [9] : il brutto risultato negativo relativo alla procedura di Davis e Putnam contrasta in modo evidente con l'esperienza che ne fa uno dei sistemi più usati e utili di dimostrazione automatica; occorre rivolgersi allora forse a una diversa analisi di questi sistemi basata sul comportamento medio invece che su quello del caso peggiore; in [9] si dimostra che la procedura di Davis e Putnam è polinomiale (con un polinomio basso) in media rispetto alla ovvia distribuzione uniforme (probabilità $1/3$ che una lettera sia in una clausola, che vi sia la negazione, che non vi sia nessuna delle due).

LETTURE CONSIGLIATE

[1] Ausiello,Complessità di calcolo delle funzioni, Boringhieri,1975

[2] Baase,Computer Algorithms,Addison-Wesley,1978

[3] Cobham,The intrinsic computational difficulty of functions, in Proc.Intern.Congr.Logic, Method.,Phil?Sci.,1964

[4] Cook-Reckhow,On the length of proofs in the propositional calculus, 6th Symp.ACM Theory of Comp.,1974

[5] Cook-Reckhow,The relative efficiency of propositional proof systems, J.S.L.44(1979)

[6] Dunham-Wang,Towards feasible solutions of the tautology problem,Annals Math.Logic 10(1976)

[7] Ferrante-Rackoff,The computational Complexity of Logical Theories, Springer LNM 718

[8] Garey-Johnson,Computers and Intractability, Freeman,1979

[9] Goldberg,Average Case complexity of the SAT Problem, pross.pubblic.

[10] Krishnamurthy,Examples of hard tautologies in the propositional calculus, 11th Symp.ACM Theory of Comp.,1981

[11] Machtey-Young,Introduction to the Theory of Algorithms,North-Holland,

[12] Meyer,The inherent computational complexity of theories of ordered sets,Atti ICM Vancouver

[13] Meyer,Weak monadic second order theory of succes-

sor is not elementary-recursive, in Logic Colloquium, Springer LNM 453

[14] Rabin, articolo sull'Handbook of Math. Logic

[15] Smith, Applications of classical recursion theory to Computer Science, in Drake-Wainer (eds.), Recursion Theory: its generalizations and applications, Cambridge Univ. Press