Automi e Linguaggi Formali

Dai problemi indecidibili a quelli intrattabili, classi P e NP

Lamberto Ballan
lamberto.ballan@unipd.it

Padova, 6 Giugno 2017

Università degli Studi di Padova

Esercizi

- Definire una TM che un numero scritto in codifica binaria ad un numero scritto in codifica decimale. In input ci aspettiamo una stringa binaria (i.e. la codifica binaria del numero) e in output dovremo produrre la corrispondente codifica in decimale.

Soluzione vista a lezione:

Definiamo una TM che implementa la seguente strategia:

- utilizziamo un contatore decimale e, di volta in volta, incrementiamo il contatore di una unità e decrementiamo il numero binario di una unità

- appena il numero binario “raggiunge” lo zero vuol dire che abbiamo terminato la conversione; cancelliamo tutti i suoi simboli e terminiamo la computazione

Un problema NP-completo: soddisfacibilità

- Un esempio *NP-completo* è il problema della soddisfacibilità, cioè decidere se una espressione *booleana* è soddisfacibile
- Le espressioni *booleane* sono costruite a partire da:
  - variabili a valori booleani, ossia 1 (vero) e 0 (falso)
  - gli operatori binari $\wedge$ (AND logico) e $\vee$ (OR logico)
  - L'operatore unario $\neg$ (NOT logico)
  - parentesi per raggruppare operatori e operandi in modo da modificare, se necessario, l'ordine di precedenza

Un problema NP-completo: soddisfacibilità

- Un *assegnamento di verità* per una espressione booleana E assegna i valori 1 (vero) e 0 (falso) a ogni variabile in E
    - il valore di E rispetto ad un assegnamento di valori di verità T, indicato con $E(T)$, è il risultato della valutazione di E con ogni variabile x sostituita dal valore $T(x)$ (1 o 0) che T assegna a x
- Un assegnamento di valori di verità T alle variabili *soddisfa* l'espressione booleana E, se il valore di E (cioè $E(T)$) è 1
- Un'espressione booleana E si dice *soddisfacibile* se esiste almeno un T tale che $E(T)=1$

Un problema NP-completo: soddisfacibilità

- Esempio: $E = x \wedge \neg(y \vee z)$
  - la "sotto-espressione" $y \vee z$ è vera se almeno una tra le variabili y e z è vera, ed è falsa se lo sono entrambe
  - perciò $\neg(y \vee z)$ è vera solo quando y e z sono entrambe false
  - l'espressione completa (essendo un AND logico) è vera solo quando entrambe le variabili/sotto-espressioni sono vere; quindi sarà vera solo per x vera, y falsa e z falsa
  - l'espressione è soddisfacibile, abbiamo appena visto che l'assegnamento di valori di verità T definito da $T(x)=1$, $T(y)=0$, $T(z)=0$, è l'unico assegnamento che soddisfa E

Un problema NP-completo: soddisfacibilità

- Il *problema della soddisfacibilità (SAT)* è così definito: data un'espressione booleana *E*, è soddisfacibile?

- Enunciato come linguaggio, il problema SAT è quindi l'insieme delle espressioni booleane (codificate) soddisfacibili

Rappresentazione di istanze di SAT

- Un'espressione booleana può contenere un numero infinito di simboli; dobbiamo perciò ideare un codice con alfabeto finito che consenta di rappresentare espressioni booleane
- Codifica: da un problema SAT, ad una stringa su un alfabeto
  - Alfabeto: $\{\wedge, \vee, \neg, (,), 0, 1, x_i\}$
  - la variabile x_i è rappresentata dal simbolo x seguito dalla rappresentazione binaria dell'indice i
- Tutte le istanze di SAT sono stringhe su questo alfabeto finito
  - es. $E = x \wedge \neg(y \vee z)$ viene codificata con $x1\wedge\neg(x10\vee x11)$
- nota: se E ha m variabili, la codifica ha $O(m \log(m))$ simboli

Il teorema di Cook

- Teorema di Cook (10.9): SAT è *NP-completo*

D: La dimostrazione prevede due parti.

Nella prima parte si dimostra che SAT è in NP.
Prendiamo una TM non-deterministica che genera tutti i possibili assegnamenti in parallelo (2^n se abbiamo n variabili).
Per ogni assegnamento, la TM controlla se è soddisfacibile (in tempo polinomiale). Quindi esiste una TM non-deterministica polinomiale che risolve SAT.

Nella seconda parte dobbiamo invece dimostrare che dato un qualunque linguaggio L' in NP, esiste una riduzione polinomiale da L' a SAT.

Una versione ristretta di soddisfacibilità

- Intendiamo dimostrare l'*NP-completezza* di un'ampia gamma di problemi (tra cui il TSP precedentemente introdotto)
- Dovremmo procedere quindi per riduzione polinomiale dal problema SAT al problema in esame
- Esiste però un importante problema "intermedio", detto 3SAT, molto più facile da ridurre ai problemi tipici rispetto a SAT
  - anche 3SAT è un problema di soddisfacibilità di espressioni booleane, così come SAT
  - 3SAT però richiede che le espressioni siano di una forma ben precisa, formate cioè da congiunzione logica di clausole ognuna delle quali è disgiunzione logica di tre variabili (anche negate)

Forme normali di espressioni booleane

- Un *letterale* è una variabile o una variabile negata, ad es. x, $\neg y$
- Una *clausola* è la disgiunzione logica (OR) di uno o più letterali, ad es. x, $x \vee \neg y$
- Un espressione booleana si dice in *forma normale congiuntiva (CNF)*, se è la congiunzione logica (AND) di una o più clausole
  - ad es. $(x \vee y) \wedge (\neg x \vee z)$, e $x \wedge y$ sono in CNF
  - mentre $(x \vee y \vee z) \wedge (\neg y \vee \neg z) \wedge (x \vee y \wedge z)$ non è in CNF
- Un espressione si dice in *forma normale k-congiuntiva (k-CNF)* se è il prodotto di clausole che hanno k letterali distinti
  - ad es. $(x \vee \neg y) \wedge (y \vee \neg z) \wedge (z \vee \neg x)$ è in 2-CNF

Una versione ristretta di soddisfacibilità

- Ogni vincolo sulle espressioni booleane dà luogo ad un problema di soddisfacibilità delle espressioni che lo soddisfano
- CSAT: una data espressione booleana in CNF è soddisfacibile?
- kSAT: una data espressione booleana in k-CNF è soddisfacibile?

- CSAT, 3SAT e kSAT per ogni $k \geq 3$ sono NP-completi
- Esistono invece algoritmi in tempo lineare per 1SAT e 2SAT

Riduzioni

- Possiamo trasformare una espressione booleana “generica” a CNF in tempo polinomiale ⇒ anche CSAT è *NP-completo*

- Possiamo ridurre CSAT a 3SAT in tempo lineare ⇒ anche 3SAT è *NP-completo*

Conversione in CNF di espressioni booleane

- Due espressioni booleane sono *equivalenti* se danno lo stesso risultato per ogni assegnazione di valori di verità alle variabili
    - quindi se sue espressioni sono equivalenti, o sono entrambe soddisfacibili o nessuna delle due lo è
    - perciò un modo per ricavare una riduzione polinomiale da SAT a CSAT consiste nel convertire espressioni arbitrarie in CNF
    - questo però non è semplicissimo, possiamo convertire ogni espressione in CNF ma potrebbe richiedere tempo esponenziale
    - “buona notizia”: non è necessario convertire una espressione in una CNF equivalente; per ridurre SAT a CSAT basterà convertire un’istanza *E* di SAT in un’istanza *F* di CSAT in modo tale che *F* sia soddisfacibile se e solo se *E* lo è

Conversione in CNF di espressioni booleane

- La riduzione di SAT in CSAT si articola in due parti:
    - spostiamo tutti i ¬ verso il fondo dell'albero di espressione (finché i ¬ si applicano a singole variabili); così si genera un'espressione equivalente E formata da ∧ o ∨ di letterali
    - scriviamo poi l'espressione formata da ∧ o ∨ di letterali come prodotto di clausole, cioè in CNF; ricorrendo a nuove variabili la trasformazione richiede un tempo polinomiale (in generale la nuova espressione F non è equivalente ad E)

Conversione in CNF di espressioni booleane

- Esempio: consideriamo $E = \neg((\neg(x \lor y)) \land (\neg x \lor y))$
  - il primo passo consiste nello “spingere” i $\neg$ sotto $\land$ e $\lor$
  - per effettuare la trasformazione ci servono le tre regole
    $\neg(E \lor F) \Rightarrow \neg(E) \land \neg(F)$

    $\neg(E \land F) \Rightarrow \neg(E) \lor \neg(F)$

    $\neg(\neg(E)) \Rightarrow E$
  - E si trasforma quindi seguendo i passaggi:
    $\neg((\neg(x \lor y)) \land \neg(\neg x \lor y))$
    $x \lor y \lor \neg(\neg x \lor y)$
    $x \lor y \lor (\neg(\neg x))\land(\neg y)$
    $x \lor y \lor x\land(\neg y)$

Conversione in CNF di espressioni booleane

- Teorema (10.12): per ogni espressione booleana E esiste un'espressione equivalente F in cui le negazioni compaiono solo nei letterali (cioè si applicano alle variabili)
  - la lunghezza di F è lineare nel numero di simboli di F ed E si costruisce in tempo polinomiale

NP-completezza di 3SAT

- 3SAT: data un'espressione *E*, prodotto di clausole formate dalla congiunzione di 3 letterali distinti, dire se *E* è soddisfacibile
- Teorema (10.15): 3SAT è NP completo