Azzolini Riccardo 2020-12-21

Equivalenza tra la risoluzione di problemi e il riconoscimento di linguaggi

1 Problemi di decisione

Un **problema di decisione** è una domanda che dipende da un insieme finito di parametri e che ammette una risposta sì/no (in generale, una risposta binaria).

Un'**istanza di un problema** viene fornita specificando dei valori per i suoi parametri, quindi un problema di decisione Π può anche essere pensato come una funzione,

$$\Pi : D_\Pi \to \{S\grave{I}, NO\}$$

dove:

- D_Π (il dominio) è l'insieme delle istanze di Π;
- per ogni $x \in D_\Pi$, $\Pi(x)$ è la soluzione del problema relativa all'istanza x.

Un'**istanza positiva** di Π è un'istanza del problema Π che ha risposta $S\grave{I}$; l'insieme delle istanze positive di Π è dunque:

$$Y_\Pi = \{x \in D_\Pi \mid \Pi(x) = S\grave{I}\}$$

2 Linguaggio associato a un problema di decisione

Ci sono dei problemi che non possono essere risolti dalle macchine di Turing (e quindi, per la tesi di Church-Turing, neanche da altri modelli di calcolo). Avendo formalizzato le macchine di Turing come riconoscitori di linguaggi, per studiare i problemi che esse possono risolvere bisogna prima trovare una corrispondenza tra il riconoscimento di linguaggi e la risoluzione di altre classi di problemi, a partire dai problemi di decisione (che sono quelli per cui la corrispondenza è più immediata).

Una **codifica** di un problema Π su un alfabeto Σ è una funzione

$$\# : D_\Pi \to \Sigma^*$$

che associa a ogni istanza del problema una stringa $\#(w) \in \Sigma^*$. Tale funzione deve essere invertibile (cioè deve esistere una funzione $\#^{-1} : \Sigma^* \to D_\Pi$ tale che $\#^{-1}(\#(x)) = x$ per ogni $x \in D_\Pi$, il che rende in pratica possibile la decodifica di una stringa, per riottenere

la corrispondente istanza del problema) e "calcolabile da un programma" (formalmente, dovrebbe essere calcolabile da una macchina di Turing o un altro modello equivalente, ma per semplicità si può sfruttare la tesi di Church-Turing, che permette di considerare funzioni "intuitivamente calcolabili").

Data una funzione di codifica # su Σ per un problema Π, il **linguaggio associato** a Π è l'insieme delle stringhe che codificano le istanze positive del problema:

$$\begin{aligned} L(\Pi) &= \{w \in \Sigma^* \mid w \text{ è la codifica di un'istanza positiva di } \Pi\} \\ &= \{w \in \Sigma^* \mid w = \#(x) \text{ con } x \in Y_\Pi\} \\ &= \{w \in \Sigma^* \mid \Pi(\#^{-1}(w)) = S\grave{I}\} \end{aligned}$$

Una macchina di Turing che riconosce $L(\Pi)$ risolve a tutti gli effetti Π: data la stringa $\#(x)$ che codifica un'istanza x del problema, la macchina accetta $\#(x)$ se e solo se $\Pi(x) = S\grave{I}$, mentre rifiuta $\#(x)$ o non si arresta se e solo se $\Pi(x) = NO$.

3 Esempio: l'ultimo teorema di Fermat

Si consideri il seguente problema di decisione UTF, trattato dal famoso "ultimo teorema di Fermat":

Parametri: Un intero positivo n.

Domanda: Esistono interi positivi x, y, z tali che $x^n + y^n = z^n$?

Un'istanza di UTF viene specificata indicando un valore per n. L'ultimo teorema di Fermat afferma che $x^n + y^n = z^n$ non ammette soluzioni intere positive per $n > 2$, quindi l'insieme delle istanze positive del problema è $Y_{\text{UTF}} = \{1, 2\}$. Allora, fissata una qualche codifica # per i numeri naturali (ad esempio la rappresentazione unaria o quella binaria), il linguaggio associato al problema è

$$L(\text{UTF}) = \{\#(1), \#(2)\}$$

(in particolare, questo è un linguaggio finito, dunque per riconoscerlo è sufficiente un automa a stati finiti).

4 Altre classi di problemi

Ci sono altre classi di problemi ai quali è possibile associare linguaggi, riconducendoli a problemi di decisione equivalenti. Ad esempio, si consideri il seguente problema TS, chiamato *problema del commesso viaggiatore* (in inglese *travelling salesman problem*):

Parametri: Un insieme di città $C = \{c_1, \ldots, c_n\}$, e la distanza $d(c_i, c_j) \in \mathbb{N}$ tra ogni coppia di città.

Domanda: Qual è il *percorso minore* che visita tutte le città una e una sola volta e torna alla città di origine? Formalmente, qual è un ordinamento $\langle c_{\tau(1)}, \ldots, c_{\tau(n)} \rangle$ delle città che *minimizza* la funzione

$$\left(\sum_{i=1}^{n-1} d(c_{\tau(i)}, c_{\tau(i+1)}) \right) + d(c_{\tau(n)}, c_{\tau(1)})$$

(dove $\tau : \{1, \ldots, n\} \to \{1, \ldots, n\}$ è una permutazione degli indici delle città)?

TS è un **problema di ottimizzazione**, la cui soluzione è ottenuta *minimizzando una funzione obiettivo* (in questo caso, la lunghezza del percorso). Tuttavia, lo si può riformulare come un problema di decisione TSD:

Parametri: Un insieme di città $C = \{c_1, \ldots, c_n\}$, la distanza $d(c_i, c_j) \in \mathbb{N}$ tra ogni coppia di città, e un valore $B \in \mathbb{N}$.

Domanda: Esiste un percorso di lunghezza minore di B che visita tutte le città una e una sola volta e torna alla città di origine? Formalmente, esiste un ordinamento $\langle c_{\tau(1)}, \ldots, c_{\tau(n)} \rangle$ delle città tale che

$$\left(\sum_{i=1}^{n-1} d(c_{\tau(i)}, c_{\tau(i+1)}) \right) + d(c_{\tau(n)}, c_{\tau(1)}) \leq B \ ?$$

Se si riesce a risolvere TSD, allora si riesce anche a risolvere TS. Infatti, il percorso che è soluzione di TS arriva in ogni città una e una sola volta, dunque la sua lunghezza è sicuramente minore rispetto alla somma M delle distanze tra ogni coppia di città,

$$M = \sum_{i=1}^{n} \sum_{j=1}^{n} d(c_i, c_j)$$

ovvero M è un *upper bound* alla lunghezza della soluzione di TS. Allora, risolvendo TSD per tutti i valori $B < M$ (o impiegando una strategia più "furba", come ad esempio una ricerca dicotomica), si trova il valore minimo di B per cui esiste ancora un percorso che visiti tutte le città: questa è la lunghezza esatta della soluzione di TS. Infine, il calcolo che risolve un'istanza TSD(B) può farlo solo trovando effettivamente un percorso di lunghezza B, quindi dalla risoluzione corrispondente alla lunghezza del percorso minimo può essere anche estratto il percorso stesso, che costituisce la soluzione di TS.

In generale, la soluzione di ogni problema di ottimizzazione può essere ridotta alla soluzione del corrispondente problema di decisione, e lo stesso vale per un'ampia classe di problemi. Ad esempio, il calcolo di una funzione $y = f(x)$ può essere ridotto al problema di decidere se una coppia (x, y) appartiene al *grafo* della funzione, cioè all'insieme $\{(x, y) \mid y = f(x)\}$.

È importante sottolineare che questa strategia di riduzione a un problema di decisione vale solo dal punto di vista della computabilità, e non da quello della complessità: la complessità di un problema può essere anche molto diversa da quella del corrispondente problema di decisione.

4.1 Linguaggio associato a TSD

Siccome TSD è un problema di decisione, è possibile fissare una codifica per le sue istanze su un alfabeto Σ e definire il linguaggio associato al problema:

$$L(\text{TSD}) = \{w \in \Sigma^* \mid w \text{ è la codifica di un'istanza positiva di TSD}\}$$

Una possibile codifica delle istanze di TSD su un alfabeto $\Sigma = \{C, [,], /, 0, 1, \ldots, 9\}$ è quella mostrata nel seguente esempio:

$$\underbrace{C[1]C[2]C[3]C[4]}_{\text{città}} // \overbrace{10/5/9}^{d(c_1,c_2)/d(c_1,c_3)/d(c_1,c_4)} // \underbrace{6/9}_{d(c_2,c_3)/d(c_2,c_4)} // \overbrace{3}^{d(c_3,c_4)} /// \underbrace{23}_{B}$$

1. Per prima cosa vengono indicate le città.
2. Seguono una serie di blocchi separati da //, ciascuno dei quali indica le distanze, separate da /, di una città c_i dalle altre città c_j. Siccome la distanza è simmetrica, $d(c_i, c_j) = d(c_j, c_i)$, è sufficiente indicare le distanze $d(c_i, c_j)$ per $i < j$, evitando così dati ridondanti.
3. Infine, dopo un ultimo separatore ///, viene indicato il valore del parametro B.

5 Classificazione dei problemi

Data la corrispondenza tra linguaggi e problemi di decisione, la classificazione dei linguaggi in base all'esistenza di MdT che li riconoscano può essere riportata anche sui problemi di decisione. Un problema di decisione P è:

- **ricorsivamente enumerabile** (**r.e.**) se esiste una MdT M che riconosce il linguaggio associato a P, cioè tale che $L(M) = L(P)$;
- **decidibile** se esiste una MdT M che riconosce $L(M) = L(P)$ e si arresta per ogni input;
- **indecidibile** se non è decidibile.

Un problema decidibile è per definizione anche ricorsivamente enumerabile, mentre un problema indecidibile può essere o meno r.e.

Studiare lo **status computazionale** di un problema di decisione significa appunto stabilire se esso sia decidibile, r.e. ma indecidibile, oppure neanche r.e. La teoria che si occupa di ciò è la **teoria della computabilità** (detta anche *teoria della calcolabilità* o *della ricorsività*).