Laboratorio di Algoritmi e Strutture Dati

Docenti: M. Goldwurm, S. Aguzzoli

Appello dell'8 Febbraio 2005

Progetto "Richiami"
Consegna entro il 24 Febbraio 2005

Il problema

Obiettivo del progetto è studiare gli spostamenti di automi puntiformi che si muovono su un terreno accidentato e sono sensibili a segnali che li richiamano verso un certo luogo.

Formalmente, chiamiamo *piano* l'insieme dei punti

$$\{\ (x, y) \in \mathbb{Z} \times \mathbb{Z}\ \}.$$

Ogni *automa* è identificato univocamente da un *nome* η, che è una stringa finita sull'alfabeto $\{0, 1\}$ ($\eta = b_1 b_2 \cdots b_n$ per qualche intero positivo n e $b_i \in \{0, 1\}$ per ogni $i \in \{1, \ldots, n\}$).

La *posizione* $P(\eta)$ di un automa η in un dato momento è specificata dando le *coordinate* $(x_0, y_0) \in \mathbb{Z} \times \mathbb{Z}$ del punto del piano in cui l'automa si trova. Quindi, $P(\eta) = (x_0, y_0)$ significa che η si trova nel punto (x_0, y_0); in un punto può esservi più di un automa.

Nel piano sono presenti degli *ostacoli* che limitano le possibilità di spostamento degli automi. Ogni ostacolo è rappresentato da un rettangolo $R(x_0, y_0, x_1, y_1)$, dove x_0, x_1, y_0, y_1 sono interi e

$$R(x_0, y_0, x_1, y_1) \ = \ \{(x, y) \in \mathbb{Z} \times \mathbb{Z} \ \mid \ x_0 \leq x \leq x_1,\ y_0 \leq y \leq y_1\}.$$

È possibile che gli ostacoli si sovrappongano; nessun automa potrà posizionarsi in un punto appartenente a qualche ostacolo.

Un *passo unitario orizzontale* è un segmento della forma $\{(x, y_0) \mid x_0 \leq x \leq x_0 + 1\}$ per x_0, y_0 interi. Un *passo unitario verticale* è un segmento della forma $\{(x_0, y) \mid y_0 \leq y \leq y_0 + 1\}$ per x_0, y_0 interi.

Un *percorso* P è una successione $P = p_1, p_2, \ldots, p_k$ di k passi unitari orizzontali e/o verticali tali che, per ogni $i \in \{1, \ldots, k-1\}$, l'intersezione $p_i \cap p_{i+1}$ contiene uno e un solo punto. La *lunghezza* di P è k. Un percorso è *libero* se, per ogni passo unitario p del percorso e ogni ostacolo R presente sul piano, $p \cap R = \emptyset$.

La *distanza* tra due punti (x_0, y_0), (x_1, y_1) è data dalla lunghezza minima fra i percorsi che collegano (x_0, y_0) a (x_1, y_1) nel piano sgombro da ostacoli. Dunque:

$$D((x_0, y_0), (x_1, y_1)) = |x_1 - x_0| + |y_1 - y_0|\ .$$

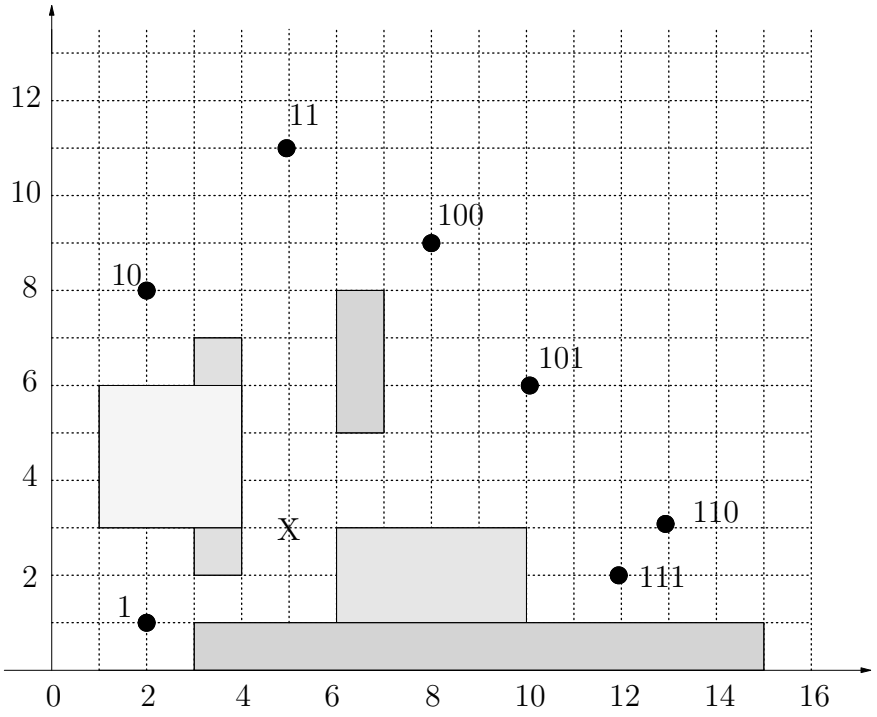
Una sorgente collocata nel punto (x, y) può emettere *segnali di richiamo* rappresentati da stringhe finite α sull'alfabeto $\{0, 1\}$. All'atto dell'emissione del segnale ogni automa determina se tale segnale lo riguarda o meno: l'automa di nome η deve rispondere al richiamo del segnale α se e solo se α è un prefisso di η, vale a dire $\alpha = b_1 b_2 \cdots b_h$ e $\eta = b_1 b_2 \cdots b_n$ per qualche $h \leq n$.

Supponiamo che l'insieme di automi a cui è diretto il richiamo α emesso dal punto (x, y) sia $\{\eta_1, \ldots, \eta_u\}$ (dunque α è prefisso di η_i per ogni $i \in \{1, \ldots, u\}$). Sia $P(\eta_i) = (x_i, y_i)$ la posizione attuale di η_i e sia d_{η_i} la distanza $D((x_i, y_i), (x, y))$. Allora, fra tutti gli automi η_i che possono raggiungere (x, y) con un percorso libero di lunghezza d_{η_i}, si sposteranno effettivamente in (x, y) tutti e soli gli automi a distanza minima da (x, y). Formalmente, sia $A \subseteq \{\eta_1, \ldots, \eta_u\}$ l'insieme degli automi η_i per cui esiste un percorso libero

di lunghezza d_{η_i} e sia $d = \min_{i=1}^{u}\{d_{\eta_i}\}$. Allora, per ogni $\eta_j \in A$ tale che $d_{\eta_j} = d$, si pone $P(\eta_j) = (x, y)$; per ogni altro automa η, $P(\eta)$ rimane immutata.

Esempio

Si supponga che nel piano ci siano gli automi 1, 10, 11, 100, 101, 110 e 111 e cinque ostacoli come nella figura



dove

$$P(1) = (2,1) \qquad P(10) = (2,8) \qquad P(11) = (5,11) \qquad P(100) = (8,9)$$
$$P(101) = (10,6) \qquad P(110) = (13,3) \qquad P(111) = (12,2)$$

e supponiamo che il richiamo 1 sia emesso dal punto $X = (5,3)$. Il segnale è recepito da tutti gli automi del piano.

L'automa 1 ha distanza 5 da X, tuttavia non esiste alcun percorso libero da $P(1)$ a X di lunghezza 5, quindi 1 non può raggiungere X.

L'automa 10 ha distanza $d_{10} = 8$ da X e può raggiungere X; lo stesso vale per gli automi 11 e 101.

L'automa 100 ha distanza $d_{100} = 9$ da X e può raggiungere X.

Gli automi 110 e 111 hanno distanza 8 da X, ma nessuno dei due può raggiungere X.

Quindi, $d = \min\{d_{10},\, d_{11},\, d_{101},\, d_{100}\}$ è 8 e si spostano in X gli automi 10, 11, 101 mentre gli altri stanno fermi.

Dato un percorso $P = p_1, \ldots, p_u$, siano $\delta_1, \ldots, \delta_u$ tali che, per ogni $i \in \{1, \ldots, u\}$, $\delta_i = O$ sse p_i è un passo unitario orizzontale, $\delta_i = V$ sse p_i è un passo unitario verticale. La *tortuosità* di P è data dal numero totale di indici $i \in \{1, \ldots, u-1\}$ tali che ($\delta_i = O$ e $\delta_{i+1} = V$) o ($\delta_i = V$ e $\delta_{i+1} = O$), ossia, dal numero di volte che seguendo il percorso si cambia direzione. Quando un automa si sposta sceglie, fra i percorsi a distanza minima, quello meno tortuoso. Ad esempio, 11 può andare direttamente in X con

un percorso di tortuosità 0, l'unico percorso per 10 ha tortuosità 1 mentre la minima tortuosità di un percorso per 101 è 2.

Si richiede di implementare una struttura dati efficiente che permetta di eseguire le operazioni seguenti (si tenga presente che la minima porzione rettangolare di piano contenente tutti gli automi e tutti gli ostacoli può essere molto grande rispetto al numero di automi e ostacoli presenti, quindi *non è sicuramente efficiente rappresentare il piano mediante una matrice*).

- **crea** ()

  Crea un piano vuoto (eliminando l'eventuale piano già esistente).

- **automa** (x_0, y_0, η)

  Se il punto (x_0, y_0) è contenuto in qualche ostacolo, allora non esegue alcuna operazione. Altrimenti, se non esiste alcun automa di nome η lo crea e lo pone in (x_0, y_0). Se η esiste già, lo riposiziona nel punto (x_0, y_0).

- **ostacolo** (x_0, y_0, x_1, y_1)

  Se i punti nel rettangolo $R(x_0, y_0, x_1, y_1)$ non contengono alcun automa, inserisce l'ostacolo rappresentato da $R(x_0, y_0, x_1, y_1)$, altrimenti non compie alcuna operazione.

- **segnale** (x, y, α)

  Viene emesso il segnale di richiamo α dal punto (x, y) (ovviamente, se il punto (x, y) appartiene a qualche ostacolo, esso non è raggiungibile da alcun automa).

- **posizione** (α)

  Restituisce la lista di tutti gli automi η tali che α è un prefisso di η, specificando l'attuale posizione $P(\eta)$, nel formato definito nelle note della sezione *Specifiche di implementazione*.

- **esistePercorso** (x, y, η)

  Stampa `SI` se esiste almeno un percorso libero da $P(\eta)$ a (x, y) di lunghezza $D(P(\eta), (x, y))$, `NO` in caso contrario (ovviamente, se η non esiste, stampa `NO`).

- **tortuosità** (x, y, η)

  Restituisce la tortuosità minima fra tutti i percorsi liberi da $P(\eta)$ a (x, y) di lunghezza $D(P(\eta), (x, y))$. Restituisce -1 se non esiste alcun siffatto percorso libero (ovviamente, se η non esiste, stampa -1).

Specifiche di implementazione

Il programma deve leggere dallo standard input (`stdin`) una sequenza di linee (separate da `\n`), ciascuna delle quali corrisponde a una linea della prima colonna della Tabella 1, dove a, b, c, d sono numeri interi e α è una stringa finita sull'alfabeto $\{0, 1\}$.

I vari elementi sulla linea sono separati da uno o più spazi. Quando una linea è letta, viene eseguita l'operazione associata; le operazioni di stampa sono effettuate sullo standard output (`stdout`), e ogni operazione deve iniziare su una nuova linea.

Linea di input	Operazione
`c`	**crea** ()
`a` a b α	**automa** (a, b, α)
`o` a b c d	**ostacolo** (a, b, c, d)
`s` a b α	**segnale** (a, b, α)
`p` α	**posizione** (α)
`e` a b α	**esistePercorso** (a, b, α)
`t` a b α	**tortuosità** (a, b, α)
`f`	Termina l'esecuzione del programma

Tabella 1: Specifiche del programma

Note

1. Non devono essere presenti vincoli sulla dimensione del piano, sulla lunghezza delle stringhe e sul numero di elementi nel piano. Non si richiede – anzi si sconsiglia – l'uso di grafica, se non per test personali: in modo particolare, non si usi `conio.h` e neppure `clrscr()`.

2. Per semplicità si suppone che l'input sia sempre conforme alle specifiche di Tabella 1, per cui non è necessario controllare la correttezza dell'input. Per leggere l'input si usino le funzioni standard ANSI C `getchar()` e/o `scanf()`.

3. Siano $\eta_1, \ldots, \eta_h$ tutti i nomi di automi nel piano di cui α è un prefisso, e sia per ogni $i \in \{1, \ldots, h\}$, $P(\eta_i) = (x_i, y_i)$. Allora l'output del comando `p` α deve essere visualizzato nel seguente formato:
   (
   $\eta_1 : x_1, y_1$
   $\eta_2 : x_2, y_2$
   $\vdots$
   $\eta_h : x_h, y_h$
)

   L'ordine in cui appaiono gli automi $\eta_1, \ldots, \eta_h$ non è rilevante.

Esempio

Si supponga che le linee di input siano:

```
c
o -8 -5 2 -2
o -12 2 -7 5
o -8 3 -2 4
o -10 7 -4 9
```

```
o -1 7 1 9
o -2 8 6 11
o 3 2 7 6
o 4 -2 12 0
o 9 2 12 10
a -13 4 1
a -3 6 11
a 7 8 10
a 13 11 0
a -10 -3 00
a 8 -4 01
e 8 10 1
t 8 10 1
e 2 5 1
t 2 5 1
e -10000 6 11
t -10000 6 11
e -14 0 10
t -14 0 10
e 7 1 10
e 7 8 0
t 7 8 0
e -14 -5 0
t -14 -5 0
e -14 2 0
t -14 2 0
e 10 11 00
t 10 11 00
e -12 9 01
t -12 9 01
t 13 11 00
t 12 11 00
o 7 11 8 12
e 12 11 00
t -10 12 01
o -8 11 -6 12
t -10 12 01
t 1000 2000 01
s 2 7 1
s 2 7 0
p 0
p 1
f
```

L'output prodotto dal programma deve essere:

```
SI
4
NO
-1
```

```
SI
0
SI
3
NO
SI
1
SI
2
SI
5
SI
3
SI
5
2
3
NO
6
7
1
(
0: 13, 11
00: -10, -3
01: 2, 7
)
(
1: -13, 4
10: 2, 7
11: 2, 7
)
```

Presentazione del progetto

Il progetto deve essere inviato per posta elettronica all'indirizzo `aguzzoli@dsi.unimi.it` entro il 24 Febbraio 2005 (incluso). La discussione del progetto e l'esame orale si svolgeranno in data e luogo da specificarsi (consultare al riguardo il sito: `http://homes.dsi.unimi.it/~goldwurm/algo`).

Occorre presentare:

1. il codice sorgente (rigorosamente ANSI C, compilabile con **gcc**);
2. una sintetica relazione (formato pdf o rtf) che illustra le strutture dati utilizzate e analizza il costo delle diverse operazioni richieste dalla specifica.

I due o più file (file sorgenti C + relazione) devono essere contenuti in un unico file `.zip` il cui nome dovrà essere `cognome.zip`. La relazione e il codice devono riportare il vostro nome, cognome e matricola.

Una copia cartacea della relazione e del codice deve inoltre essere consegnata al dr. Aguzzoli sempre entro il 24 Febbraio 2005 (lasciandola eventualmente nella sua casella postale presso il dipartimento in via Comelico).

Si ricorda infine di presentarsi alla prova orale con una copia stampata della relazione e del codice.

Per ogni ulteriore chiarimento:
E-mail: `aguzzoli@dsi.unimi.it`
Ricevimento: il mercoledì, ore 15-16, stanza S204.

Avvisi

La versione aggiornata del progetto è pubblicata in .pdf sul sito:
`http://homes.dsi.unimi.it/~aguzzoli/algo.htm`.
Si consiglia di consultare periodicamente questo sito per eventuali correzioni e/o precisazioni relative al testo del progetto.

Si richiede allo studente di effettuare un adeguato collaudo del proprio progetto su numerosi esempi diversi per verificarne la correttezza e valutarne le prestazioni.

La realizzazione del progetto è una prova d'esame da svolgersi **individualmente**. I progetti giudicati frutto di **collaborazioni** saranno **estromessi** d'ufficio dalla valutazione.