Laboratorio di Algoritmi e Strutture Dati

Docenti: M. Torelli, S. Aguzzoli

Appello del 3 luglio 2007

Progetto "Domino 2"
Consegna entro il 22 luglio 2007

Il problema

Il gioco del domino consiste nel disporre una sequenza di tessere, ciascuna delle quali suddivisa in due parti aventi colori distinti, in modo che i lati a contatto di due tessere adiacenti abbiano lo stesso colore.

Formalmente, sia $\mathcal{A}$ l'insieme delle lettere minuscole dell'alfabeto. Una *tessera* t è definita da una tripla (α, β, σ), dove α, β e σ sono parole di lunghezza *arbitraria* sull'alfabeto $\mathcal{A}$; α e β rappresentano i colori con cui è colorata ciascuna delle due metà di t, σ il nome di t. Per ottenere un domino a partire da un insieme di tessere $\mathcal{D}$ detto *deposito*, possiamo immaginare di prelevare le tessere da $\mathcal{D}$ e di disporle in una sequenza. Denotiamo una *disposizione* di una tessera t con la coppia (t, k), con $k \in \{-1, +1\}$. Data una tessera $t = (\alpha, \beta, \sigma)$ e una sua disposizione (t, k), il *colore sinistro* Col_S di (t, k) e il *colore destro* Col_D di (t, k) sono definiti nel modo seguente, a seconda del valore di k:

- $Col_S((t, 1)) = \alpha$ e $Col_D((t, 1)) = \beta$;
- $Col_S((t, -1)) = \beta$ e $Col_D((t, -1)) = \alpha$.

Un *domino* τ costruito su un deposito $\mathcal{D}$ è dato da una sequenza di disposizioni di tessere

$$(t_1, k_1),\ (t_2, k_2),\ \ldots,\ (t_n, k_n)$$

tali che:

- per ogni $1 \leq i \leq n$, la tessera t_i appartiene a $\mathcal{D}$
- per ogni $1 \leq i \leq n$ e $1 \leq j \leq n$, $i \neq j$ implica $t_i \neq t_j$;
- per ogni $1 \leq i \leq n-1$, $Col_D((t_i, k_i)) = Col_S((t_{i+1}, k_{i+1}))$.

A seguito della costruzione del domino τ, le tessere che lo costituiscono vengono tolte dal deposito $\mathcal{D}$; formalmente il deposito aggiornato è l'insieme di tessere $\mathcal{D} \setminus \{t_1, \ldots, t_n\}$.

La *lunghezza* di τ è n (numero di tessere usate in τ). Se $Col_S((t_1, k_1)) = \alpha$ e $Col_D((t_n, k_n)) = \beta$, diciamo che τ è un domino dal colore α al colore β. Un domino *minimo* da α a β su $\mathcal{D}$ è un domino da α a β avente lunghezza minima fra tutti i possibili domini da α a β costruibili sul deposito $\mathcal{D}$.

Esempio

Supponiamo che il deposito $\mathcal{D}$ contenga le seguenti tessere:

$$t_1 = (rosso, verde, pane),\ t_2 = (rosso, blu, cane),\ t_3 = (giallo, blu, casa),\ t_4 = (verde, giallo, cosa)$$

Allora i domini da *rosso* a *giallo* costruibili su $\mathcal{D}$ sono i seguenti:

$$(t_1, 1), (t_4, 1) \quad \text{e} \quad (t_2, 1), (t_3, -1)$$

entrambi di lunghezza minima 2. Nel primo caso il deposito aggiornato risulta essere $\{t_2, t_3\}$, nel secondo $\{t_1, t_4\}$. I domini da *rosso* a *verde* costruibili sul deposito iniziale $\mathcal{D}$ sono:

$$(t_1, 1) \quad \text{e} \quad (t_2, 1), (t_3, -1), (t_4, -1)$$

e il domino $(t_1, 1)$ che ha lunghezza 1, è l'unico domino da *rosso* a *verde* di lunghezza minima. Nel primo caso il deposito aggiornato risulta essere $\{t_2, t_3, t_4\}$, nel secondo $\{t_1\}$.

Dato un domino $\tau = (t_1, k_1), (t_2, k_2), \ldots, (t_n, k_n)$, la sequenza di colori definita da τ è la sequenza di colori $Col_S((t_1, k_1)), Col_S((t_2, k_2)), \ldots, Col_S((t_n, k_n)), Col_D((t_n, k_n))$. Nell'esempio precedente, la sequenza di colori definita dal domino $(t_1, 1), (t_4, 1)$ è $rosso, verde, giallo$, la sequenza di colori definita da $(t_2, 1), (t_3, -1)$ è $rosso, blu, giallo$. Dati due domini τ_1 e τ_2, diciamo che τ_1 e τ_2 sono *simili* se la sequenza di colori definita dal domino τ_1 è uguale a quella definita da τ_2. Diciamo che τ_2 può essere ottenuto da τ_1 se è possibile effettuare a partire da τ_1 una sequenza di operazioni elementari di inserimento o cancellazione di tessere in modo da ottenere un domino simile a τ_2. Più formalmente, una sequenza di tessere σ' è ottenuta con una operazione elementare da una sequenza σ sul deposito $\mathcal{D}$ se si verifica uno dei seguenti casi:

- *Inserimento di una tessera t del deposito in σ*

  Sia t una tessera del deposito e sia σ la sequenza di tessere $t_1, t_2, \ldots, t_h$. Allora, per qualche $1 \leq j \leq h$, σ' è la sequenza $t_1, t_2, \ldots, t_j, t, t_{j+1}, \ldots, t_h$ (nota che se $j = h$ allora t è l'ultima tessera di σ', e naturalmente t_{j+1} in questo caso non esiste). Dopo l'inserimento di t, il deposito $\mathcal{D}$ viene modificato ponendo $\mathcal{D} = \mathcal{D} \setminus \{t\}$.

- *Cancellazione di una tessera t_j da σ*

  Sia σ la sequenza $t_1, t_2, \ldots, t_h$. Allora, per qualche $1 \leq j \leq h$, σ' è la sequenza $t_1, t_2, \ldots, t_{j-1}, t_{j+1}, \ldots, t_h$ (se $j = h$, l'ultima tessera di σ' è t_{h-1}). Il deposito $\mathcal{D}$ viene modificato ponendo $\mathcal{D} = \mathcal{D} \cup \{t_j\}$.

Diciamo che è possibile *trasformare* τ_1 in τ_2 sul deposito $\mathcal{D}$ se esiste una sequenza finita $\sigma_1, \sigma_2, \ldots, \sigma_k$ di sequenze di tessere tali che:

- Se $\tau_1 = (t_1, k_1), \ldots, (t_n, k_n)$, allora σ_1 è la sequenza di tessere $t_1, \ldots, t_n$.
- Per ogni $1 \leq i < k$, la sequenza σ_{i+1} è ottenuta dalla sequenza σ_i con una operazione elementare.
- Sia σ_k la sequenza di tessere $t'_1, \ldots, t'_m$. Allora esistono $k'_1, \ldots, k'_m$ (dove, per ogni $1 \leq j \leq m$, $k'_j \in \{-1, 1\}$) tale che il domino

  $$(t'_1, k'_1), \ldots, (t'_m, k'_m)$$

  è simile a τ_2.

Si osservi che le sequenze intermedie $\sigma_2, \sigma_3, \ldots, \sigma_{k-1}$ possono anche non dare origine a domini. Il numero delle operazioni elementari effettuate per trasformare τ_1 in τ_2 tramite la sequenza $\sigma_1, \sigma_2, \ldots, \sigma_k$ è $k - 1$.

Se è possibile trasformare τ_1 in τ_2, il *costo* del passaggio da τ_1 a τ_2 è dato dal minimo numero di operazioni elementari necessarie per trasformare τ_1 a τ_2. In caso contrario, il costo del passaggio da τ_1 a τ_2 è indefinito.

Nota

Il calcolo del costo del passaggio τ_1 a τ_2 non determina alcuna effettiva modifica dei domini esistenti e del deposito. Quindi, quando il costo è stato calcolato, i domini e il deposito sono quelli che si avevano prima del calcolo.

Esempio

Si supponga di avere costruito i domini

$$\tau_1 \;=\; (t_1, 1),\, (t_2, -1),\, (t_3, 1) \qquad\qquad \tau_2 \;=\; (t_4, 1),\, (t_5, -1),\, (t_6, 1),\, (t_7, -1)$$

dove

$$t_1 \;=\; (a, b, uno), \quad t_2 \;=\; (c, b, due), \quad t_3 \;=\; (c, d, tre)$$
$$t_4 \;=\; (d, c, quattro), \quad t_5 \;=\; (a, c, cinque), \quad t_6 \;=\; (a, b, sei), \quad t_7 \;=\; (c, b, sette)$$

Se il deposito $\mathcal{D}$ è vuoto, il costo di passaggio da τ_1 a τ_2 è indefinito (infatti, non è possibile in alcun modo aggiungere al primo domino una tessera colorata con a e c). Supponiamo ora che il deposito contenga le tessere

$$t_8 \;=\; (a, b, otto), \quad t_9 \;=\; (a, c, nove), \quad t_{10} \;=\; (c, a, dieci)$$

In questo caso il costo di passaggio da τ_1 a τ_2 è 3. Infatti, la sequenza di partenza è

$$\sigma_1 \;=\; t_1,\, t_2,\, t_3$$

Un esempio di sequenza di tre operazioni elementari che permette di trasformare τ_1 in τ_2 è:

1. Eliminazione di t_3 da σ_1. Dopo questa operazione, si ottiene la sequenza

   $$\sigma_2 \;=\; t_1,\, t_2$$

   e $\mathcal{D}$ contiene le tessere $\{t_3,\, t_8,\, t_9,\, t_{10}\}$.

2. Inserimento di t_3 in σ_1 (in testa). Si ottiene:

   $$\sigma_2 \;=\; t_3,\, t_1,\, t_2 \qquad \mathcal{D} \;=\; \{t_8,\, t_9,\, t_{10}\}.$$

3. Inserimento di t_9 in σ_2 (dopo t_3). Si ottiene:

   $$\sigma_3 \;=\; t_3,\, t_9,\, t_1,\, t_2 \qquad \mathcal{D} \;=\; \{t_8,\, t_{10}\}.$$

Si possono ora disporre le tessere della sequenza σ_3 in modo da ottenere il domino

$$(t_3, -1),\, (t_9, -1),\, (t_1, 1),\, (t_2, -1)$$

che è simile a τ_2 (infatti, definiscono la stessa sequenza di colori d, c, a, b, c). È facile verificare che non è possible trasformare τ_1 a τ_2 con meno di 3 operazioni elementari, quindi 3 è il costo di passaggio.

Si supponga ora di voler calcolare il costo di passaggio dal domino τ_1 al domino

$$\tau_3 \;=\; (t_6, -1),\, (t_5, 1),\, (t_4, -1)$$

avendo nel deposito le tessere t_8, t_9 e t_{10}. La sequenza di partenza è t_1, t_2, t_3. Possiamo togliere la tessera t_2 e sostituirla con la tessera t_9, ottenendo la sequenza t_1, t_9, t_3. Da questa sequenza, è possibile costruire il domino

$$(t_1, -1),\, (t_9, 1),\, (t_3, 1)$$

che è simile al domino τ_3, definendo la stessa sequenza di colori b, a, c, d. Poiché non è possibile trasformare τ_1 in τ_3 con meno di due operazioni elementari, il costo di passaggio da τ_1 a τ_3 è 2. Si noti che la prima tessera di τ_1 viene semplicemente ruotata e questo non incide nel computo del calcolo del costo di passaggio.

Si richiede di implementare una struttura dati efficiente che permetta di eseguire le operazioni seguenti:

- **tessera** (α, β, σ)

  Se nel deposito attuale $\mathcal{D}$ esiste già una tessera di nome σ oppure se α è uguale a β, non compie alcuna operazione. Altrimenti, aggiunge a $\mathcal{D}$ la tessera (α, β, σ).

- **elimina**(σ)

  Se non esiste alcune tessera di nome σ nel deposito attuale non compie alcuna operazione. Altrimenti, elimina la tessera di nome σ.

- **domino**(α, β, σ)

  Se esiste già un domino di nome σ non compie alcuna operazione. Altrimenti, crea un domino di lunghezza minima dal colore α al colore β e gli assegna il nome σ, togliendo dal deposito le tessere utilizzate. Se non è possible crearlo, stampa il messaggio:

  `non esiste domino da` α `a` β

- **stampaDomino**(σ)

  Se non esiste alcuna domino di nome σ non compie alcuna operazione. Altrimenti, stampa il domino di nome σ secondo il formato specificato nell'apposita sezione.

- **cancellaDomino**(σ)

  Se non esiste alcun domino di nome σ non compie alcuna operazione. Altrimenti, il domino di nome σ viene cancellato e tutte le tessere che lo compongono vengono aggiunte al deposito attuale.

- **costo**(σ_1, σ_2)

  Se almeno uno tra σ_1 e σ_2 non è un nome di domino, non compie alcuna operazione. Altrimenti stampa il costo del passaggio da σ_1 a σ_2 (stampa `indefinito` se il costo di passaggio è indefinito).

All'inizio del programma il deposito è vuoto. Si noti che le operazioni richieste sono liberamente implementabili; in particolare, non vanno necessariamente intese come prototipi di funzioni.

Specifiche di implementazione

Il programma deve leggere dallo standard input (`stdin`) una sequenza di righe (separate da `\n`), ciascuna delle quali corrisponde a una riga della prima colonna della Tabella 1, dove α, β e σ sono stringhe finite sull'alfabeto $\mathcal{A}$ di lunghezza *arbitraria*. I vari elementi sulla riga sono separati da uno o più spazi. Quando una riga 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 riga.

Note

1. Non devono essere presenti vincoli sul numero di tessere e domini esistenti, e sulla lunghezza dei nomi di colori e tessere (se non quelli determinati dal tipo di dato intero). 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()`.

Riga di input	Operazione
`t` $\alpha\ \beta\ \sigma$	**tessera** (α, β, σ)
`e` σ	**elimina** (σ)
`d` $\alpha\ \beta\ \sigma$	**domino** (α, β, σ)
`s` σ	**stampaDomino** (σ)
`c` σ	**cancellaDomino** (σ)
`C` $\alpha\ \beta$	**costo** (α, β)
`f`	Termina l'esecuzione del programma

Tabella 1: Specifiche del programma

3. **Formato per la stampa di un domino**
   Si supponga che il domino di nome σ da stampare sia

$$(t_1, k_1), (t_2, k_2), \ldots, (t_n, k_n)$$

   e che le tessere che lo costituiscono siano $t_i = (\alpha_i, \beta_i, \sigma_i)$ per ogni $1 \leq i \leq n$. Si ponga

$$\gamma_i = \sigma_i : \alpha_i, \beta_i \qquad \text{se} \qquad k_i = +1\,,$$

$$\gamma_i = \sigma_i : \beta_i, \alpha_i \qquad \text{se} \qquad k_i = -1\,.$$

   L'output da produrre è il seguente:

$(\sigma :$
γ_1
γ_2
$\vdots$
γ_n
$)$

Esempio
Si supponga che le righe di input siano:

```
t rosso blu uno
t rosa grigio due
t grigio rosa tre
t verde giallo quattro
t blu giallo cinque
t nero rosso sei
t nero arancione sette
t arancione verde otto
```

```
t arancione giallo nove
t arancione rosso dieci
d nero rosso primo
s primo
c primo
e sei
e dieci
d nero rosso primo
s primo
t verde azzurro undici
t rosso blu dieci
d rosso azzurro secondo
s secondo
t giallo blu dodici
d rosso azzurro secondo
s secondo
t azzurro verde tredici
t verde giallo quattordici
C primo secondo
c primo
c secondo
e dodici
t blu azzurro sedici
d rosso nero terzo
s terzo
t arancione nero quindici
d rosso nero quarto
s quarto
t azzurro blu diciassette
t verde arancione diciotto
C terzo quarto
e diciassette
C terzo quarto
C terzo terzo
f
```

L'output prodotto dal programma deve essere il seguente (si noti che a fronte di questo input non vi è un unico output corretto: laddove vi sono delle alternative, esse sono specificate fra parentesi quadre: ad esempio, nella settima riga a partire dal fondo, `undici[tredici]` significa che rimpiazzando in questa linea `undici` con `tredici` si ottiene ancora un output corretto):

```
(primo:
sei: nero, rosso
)
(primo:
sette: nero, arancione
nove: arancione, giallo
cinque: giallo, blu
uno: blu, rosso
)
```

```
non esiste domino da rosso a azzurro
(secondo:
dieci: rosso, blu
dodici: blu, giallo
quattro: giallo, verde
undici: verde, azzurro
)
6
(terzo:
uno[dieci]: rosso, blu
cinque: blu, giallo
nove: giallo, arancione
sette: arancione, nero
)
(quarto:
dieci[uno]: rosso, blu
sedici: blu, azzurro
undici[tredici]: azzurro, verde
otto: verde, arancione
quindici: arancione, nero
)
5
indefinito
0
```

Presentazione del progetto

Il progetto deve essere inviato per posta elettronica all'indirizzo `aguzzoli@dsi.unimi.it` entro il 22 luglio 2007 (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/~torelli/algoritmi.html`).

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 entro il 23 luglio 2007 (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.

La discussione del progetto e lesame orale di Algoritmi e Strutture Dati si svolgeranno indicativamente nei giorni 25 e 27 luglio 2007.

Alla consegna del progetto, indicare nel testo della e-mail la data in cui si preferisce sostenere la prova orale; nei limiti del possibile si cercherà di tener conto di tali indicazioni (se non si hanno preferenze, non dare alcuna indicazione).

Il calendario degli esami orali sarà disponibile sulla pagina del corso qualche giorno dopo il termine di consegna del progetto.

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.