Laboratorio di Algoritmi e Strutture Dati

Docenti: M. Goldwurm, S. Aguzzoli

Appello del 4 luglio 2006

Progetto "Ingegneria genetica"
Consegna entro il 23 luglio 2006

Il problema

Lo scopo del progetto è studiare la produzione di proteine tramite una particolare tecnica di ingegneria genetica.

Una *stringa* finita su $\{A, C, G, T\}$ è una sequenza $c_1 c_2 \cdots c_n$, con $n \geq 1$, dove ogni $c_i \in \{A, C, G, T\}$. Il *codice genetico* e i *geni* sono stringhe finite su $\{A, C, G, T\}$. Nel seguito assumiamo che in ogni istante ci sia un unico codice genetico τ, mentre può esserci più di un *gene*. Un gene σ è *presente* in τ se σ è una sottostringa di τ, vale a dire, se esistono stringhe η_1, η_2, eventualmente vuote, tali che $\tau = \eta_1 \cdot \sigma \cdot \eta_2$, dove $\cdot$ è l'operazione di concatenazione fra stringhe. Il gene σ *occorre* nel codice τ in posizione (l, r) $(1 \leq l \leq r)$ se il primo carattere di σ è l'l-esimo di τ e l'ultimo carattere di σ è l'r-esimo di τ. Se σ occorre più di una volta in τ, la *prima* occorrenza di σ in τ è quella in cui l'indice l della posizione ha valore minimo. Due posizioni di geni (l_1, r_1) e (l_2, r_2) sono *sovrapposte* se $l_1 \leq l_2 \leq r_1$ oppure $l_2 \leq l_1 \leq r_2$.

Chiamiamo *genoma* una sequenza $\mathcal{G}$ di geni a due a due distinti; la *lunghezza* di $\mathcal{G}$, denotata da $|\mathcal{G}|$, è data dal numero di geni appartenenti a $\mathcal{G}$. Dato il codice genetico τ e un insieme di geni $\mathcal{S} = \{\sigma_1, \ldots, \sigma_u\}$ chiamato *base*, un genoma $\mathcal{G}$ *compatibile* con τ e con la base $\mathcal{S}$ è una sequenza di geni $\mathcal{G}$ definita nel seguente modo:

- Sia $\mathcal{S}'$ l'insieme dei geni della base $\mathcal{S}$ presenti in τ e, per ogni $\sigma \in \mathcal{S}'$, sia (l_σ, r_σ) la prima occorrenza di σ in τ. Allora:
  1. I geni in $\mathcal{G}$ appartengono a $\mathcal{S}'$;
  2. Pe ogni $\sigma, \sigma' \in \mathcal{G}$, σ precede σ' in $\mathcal{G}$ se e solo se $r_\sigma < l_{\sigma'}$;
  3. Per ogni coppia di geni distinti σ e σ' di $\mathcal{G}$, (l_σ, r_σ) e $(l_{\sigma'}, r_{\sigma'})$ non si sovrappongono;
  4. Per ogni altra sequenza $\mathcal{G}'$ di geni che verifica 1, 2 e 3, deve valere $|\mathcal{G}| \geq |\mathcal{G}'|$.

Si noti che vi possono essere più genomi compatibili con τ e $\mathcal{S}$. Si osservi inoltre che $\mathcal{G}$ può essere la sequenza vuota.

Esempio 1

Sia

$$\tau \;=\; GATTACATTAGAGCGCCCCAAATATAT$$

il codice genetico e sia

$$\mathcal{S} \;=\; \{\, ATTA,\, CCC,\, CCCA,\, CATTA,\, GAG,\, TAC,\, TAG,\, TAT,\, GGG \,\}$$

la base.

La prima occorrenza di $ATTA$ in τ è $(2, 5)$, ma $ATTA$ occorre anche in posizione $(7, 10)$. La prima occorrenza di TAC è $(4, 6)$, dunque le prime occorrenze di $ATTA$ e TAC sono sovrapposte. GGG è

l'unico gene della base $\mathcal{S}$ che non è presente in τ. Più in dettaglio, le prime occorrenze dei geni della base $\mathcal{S}$ che occorrono in τ sono:

$$ATTA : (2,5) \qquad CCC : (16,18) \qquad CCCA : (17,20) \qquad CATTA : (6,10)$$
$$GAG : (11,13) \qquad TAC : (4,6) \qquad TAG : (9,11) \qquad TAT : (23,25)$$

Esempi di sequenze di geni che verificano 1,2 e 3 sono:

$$\mathcal{G}'_1 \;=\; ATTA,\, TAG \qquad \mathcal{G}'_2 \;=\; TAC,\, GAG,\, CCCA$$

La massima lunghezza di una sequenza di geni che verifica 1,2 e 3 è 5. I genomi compatibili con τ e $\mathcal{S}$ sono quindi le sequenze di geni che verificano 1,2 e 3 di lunghezza 5, ossia:

$$\mathcal{G}_1 \;=\; ATTA,\, CATTA,\, GAG,\, CCC,\, TAT$$
$$\mathcal{G}_2 \;=\; ATTA,\, CATTA,\, GAG,\, CCCA,\, TAT$$

Una *proteina* $P(\mathcal{G})$ è prodotta utilizzando un genoma $\mathcal{G} = \sigma_1, \ldots, \sigma_v$ e n istanze distinte $\tau_1, \ldots, \tau_n$ del codice genetico τ. Per definizione di $\mathcal{G}$, ogni gene σ_k di $\mathcal{G}$ occorre almeno una volta in τ. Per $1 \le i \le n$ e $1 \le k \le v$, denotiamo con $\sigma_{i,k}$ la istanza di σ_k in τ_i corrispondente alla prima occorrenza di σ_k in τ_i. Ad ogni $\sigma_{i,k}$ è associato un costo di *attivazione* $a(i,k)$, che rappresenta l'energia necessaria ad utilizzare $\sigma_{i,k}$. Inoltre, se $1 \le k \le v-1$, a $\sigma_{i,k}$ è anche associato un costo fisso di *passaggio* $p(i,k)$, che rappresenta l'energia necessaria per passare da τ_i a un diverso τ_j.

La produzione della proteina $P(\mathcal{G})$ avviene scegliendo, per ogni gene σ_k occorrente in $\mathcal{G}$, una istanza $\sigma_{s(k),k}$ da $\tau_{s(k)}$ per un'opportuno indice $s(k) \in \{1, \ldots, n\}$.

La funzione $s\colon \{1, \ldots, v\} \to \{1, \ldots, n\}$ che, per ogni gene σ_k di $\mathcal{G}$, "sceglie" quale istanza $\sigma_{s(k),k}$ prendere, è determinata in modo da minimizzare il dispendio energetico D necessario a produrre la proteina, calcolato come segue:

$$D \;=\; \sum_{k=1}^{v} a(s(k),k) \;+\; \sum_{k=1}^{v-1} q_k$$

dove $q_k = p(s(k),k)$ se $s(k) \neq s(k+1)$, $q_k = 0$ se invece $s(k) = s(k+1)$ (quindi, i costi di passaggio vengono considerati solo se si scelgono occorrenze di σ_k e σ_{k+1} in due distinte istanze di τ). È possibile che il minimo valore di D sia determinato da più di una funzione s, in tal caso si considera una arbitraria fra queste.

Esempio 2

Si consideri il genoma $\mathcal{G}_1 = (\, \sigma_1 = ATTA, \sigma_2 = CATTA, \sigma_3 = GAG, \sigma_4 = CCC, \sigma_5 = TAT \,)$ dell'Esempio 1, e siano τ_1, τ_2, τ_3 tre distinte istanze di τ, caratterizzate dai seguenti costi:

$$\begin{array}{lllll} a_{1,1}=7 & a_{1,2}=4 & a_{1,3}=3 & a_{1,4}=9 & a_{1,5}=3 \\ a_{2,1}=5 & a_{2,2}=6 & a_{2,3}=6 & a_{2,4}=8 & a_{2,5}=1 \\ a_{3,1}=6 & a_{3,2}=5 & a_{3,3}=9 & a_{3,4}=4 & a_{3,5}=3 \end{array}$$

$$\begin{array}{llll} p_{1,1}=2 & p_{1,2}=2 & p_{1,3}=5 & p_{1,4}=3 \\ p_{2,1}=4 & p_{2,2}=1 & p_{2,3}=4 & p_{2,4}=2 \\ p_{3,1}=3 & p_{3,2}=1 & p_{3,3}=1 & p_{3,4}=1 \end{array}$$

Allora la funzione $s\colon \{1, \ldots, 5\} \to \{1, \ldots, 3\}$ che minimizza il dispendio energetico D è $s(1)=1$, $s(2)=1$, $s(3)=1$, $s(4)=3$, $s(5)=2$, mentre il dispendio è $D = (7+4+3+4+1) + (0+0+5+1) = 25$.

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

- **codice** (α)

  Stabilisce che α è il codice genetico dell'organismo, eliminando l'eventuale codice precedente, e crea una base di geni vuota, eliminando l'eventuale base esistente.

- **introduci_gene** (α)

  Introduce il gene α nella base. Se α è già presente nella base, non compie alcuna azione.

- **elimina** (α)

  Elimina il gene di codice α dalla base. Se α non esiste, non compie alcuna azione.

- **stampa** (α)

  Stampa l'elenco dei geni di prefisso α contenuti nella base, cioè i geni il cui codice β è tale che $\beta = \alpha \cdot \gamma$, per qualche stringa γ eventualmente vuota. La stampa va effettuata secondo il formato specificato nell'apposita sezione.

- **genoma** ()

  Determina un genoma compatibile col codice genetico e la base, e lo stampa secondo il formato specificato nell'apposita sezione.

- **proteina** ($r, c, nome$)

  Sia $\mathcal{G} = \sigma_1, \ldots, \sigma_v$ un genoma compatibile col codice genetico e la base. Il programma stampa il dispendio energetico necessario a produrre la proteina $P(\mathcal{G})$ utilizzando il genoma $\mathcal{G}$ e r istanze del codice genetico. I valori dei costi di attivazione e dei costi di passaggio sono determinati leggendo il file *nome* contenente $2 \cdot r \cdot c$ valori positivi, in accordo con le specifiche nell'apposita sezione.

Si noti che le operazioni richieste sono liberamente implementabili; in particolare, non vanno necessariamente intese come prototipi di funzioni.

Per avere indicazioni sugli algoritmi da utilizzare, si faccia riferimento al libro di testo (T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, *Introduzione agli algoritmi e strutture dati*, McGraw-Hill, II edizione, 2005).

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 è una stringa finita sull'alfabeto $\{A, C, G, T\}$, *nome* è una stringa finita corrispondente al nome di un file, r e c sono interi positivi. 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 e sulla lunghezza dei geni nella base, sulla lunghezza del codice genetico, sul numero di istanze necessarie a produrre una proteina (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
`c` α	**codice** (α)
`i` α	**introduci_gene** (α)
`e` α	**elimina** (α)
`s` α	**stampa** (α)
`g`	**genoma** ()
`p` r c *nome*	**proteina** $(r, c, nome)$
`f`	Termina l'esecuzione del programma

Tabella 1: Specifiche del programma

3. Siano $\sigma_1, \ldots, \sigma_u$ i geni di prefisso α presenti nella base. Allora l'output del comando
   `s` α
   va visualizzato come segue:
   (
   σ_1
   σ_2
   $\vdots$
   σ_u
).
   L'ordine non è rilevante.

4. Sia $\mathcal{G} = \sigma_1, \ldots, \sigma_w$ il genoma determinato dall'operazione **genoma** (), e sia, per ogni $k \in \{1, \ldots, w\}$, (l_k, r_k) la posizione della prima occorrenza di σ_k nel codice genetico τ. Allora l'output del comando
   `g`
   va visualizzato come segue:
   (
   σ_1, l_1, r_1
   σ_2, l_2, r_2
   $\vdots$
   σ_w, l_w, r_w
).

5. Sia dato il comando
   `p` r c *nome*.
   Allora il file *nome* deve contenere $2 \cdot r \cdot c$ numeri interi positivi $n_0, n_1, \ldots, n_{2 \cdot r \cdot c - 1}$. Essi specificano due matrici A e P di dimensioni $r \times c$ di interi positivi, in questo modo:

   $$A(i,j) \;=\; n_{2(i \cdot c + j)} \qquad P(i,j) \;=\; n_{2(i \cdot c + j)+1}$$

   per ogni $0 \leq i < r$ e $0 \leq j < c$.

   Sia $\mathcal{G} = \sigma_1, \ldots, \sigma_v$ il genoma della proteina da produrre. Le istanze del codice genetico τ da considerare sono $\tau_1, \tau_2, \ldots, \tau_r$, e i costi associati sono determinati nel seguente modo:

- per ogni $1 \leq i \leq r$ e $1 \leq k \leq v$, $a(i,k) = A(i-1, (k-1) \,\%\, c)$;
- per ogni $1 \leq i \leq r$ e $1 \leq k \leq v-1$, $p(i,k) = B(i-1, (k-1) \,\%\, c)$

dove $m \,\%\, n$ è il resto della divisione intera fra m e n. Ad esempio, se $r = 2$, $c = 3$, $v = 5$ e il file contiene i numeri 1,2,3,4,...,12 si ha:

$$\begin{array}{lllll} a(1,1) = 1 & a(1,2) = 3 & a(1,3) = 5 & a(1,4) = 1 & a(1,5) = 3 \\ a(2,1) = 7 & a(2,2) = 9 & a(2,3) = 11 & a(2,4) = 7 & a(2,5) = 9 \end{array}$$

$$\begin{array}{llll} p(1,1) = 2 & p(1,2) = 4 & p(1,3) = 6 & p(1,4) = 2 \\ p(2,1) = 8 & p(2,2) = 10 & p(2,3) = 12 & p(2,4) = 8 \end{array}$$

Esempio

Si supponga che le righe di input siano:

```
c AAAACCCCGGGGTTTTAAGT
i AAACCCCGG
i GGG
i AAAA
i GGTT
i CCCCGG
i TAA
i TTTA
i ATAA
s GGT
g
i A
i AAAC
i CG
g
e AAACCCCGG
s AA
p 3 4 f.txt
f
```

dove il file `f.txt` contiene i valori

```
2 2 7 1 4 4 3 1
5 7 1 2 5 1 1 1
3 7 8 3 1 2 7 3
```

L'output prodotto dal programma deve essere:

```
(
GGTT
)
(
AAAA,1,4
CCCCGG,5,10
GGTT,11,14
```

```
TAA,16,18
)
(
A,1,1
AAAC,2,5
CG,8,9
GGTT,11,14
TAA,16,18
)
(
AAAA
AAAC
)
14
```

Presentazione del progetto

Il progetto deve essere inviato per posta elettronica all'indirizzo `aguzzoli@dsi.unimi.it` entro il 23 luglio 2006 (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 entro il 24 luglio 2006 (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.