Mappe di restrizione ed il problema del compendio parziale

Laboratorio di Programmazione II

Corso di Laurea in Bioinformatica

Dipartimento di Informatica - Università di Verona

Sommario

- Mappe di restrizione, compendio parziale (richiami)
- Algoritmo a forza bruta
- Algoritmo di Skiena

Mappe di restrizione ed il problema del compendio parziale

Mappe di restrizione

Analisi del DNA e mappe di restrizione

- Punti di restrizione (siti): punti in cui l'enzima (e.g., HindIII) divide il DNA
- Occorrenze di una particolare sequenza delle basi ATGC (GTGCAC, TTAAC)
- Mappe di restrizione: posizione dei siti di restrizione sul DNA
- La distanza tra siti di restrizione e' misurabile sperimentalmente (gel electrophoresis)
- Problema: date le distanze tra ciascuna coppia di siti di restrizione, costruire la mappa di restrizione

Il problema del compendio parziale

Problema del Compendio Parziale (Partial Digest Problem)

- Date le distanze tra tutte le possibili coppie di punti disposti su una retta calcolare la posizione dei punti sulla retta
- punti: siti di restrizione
- distanze: dati proveniente dalla gel electrophoresis
- disposizione dei punti sulla linea: mappa di restrizione

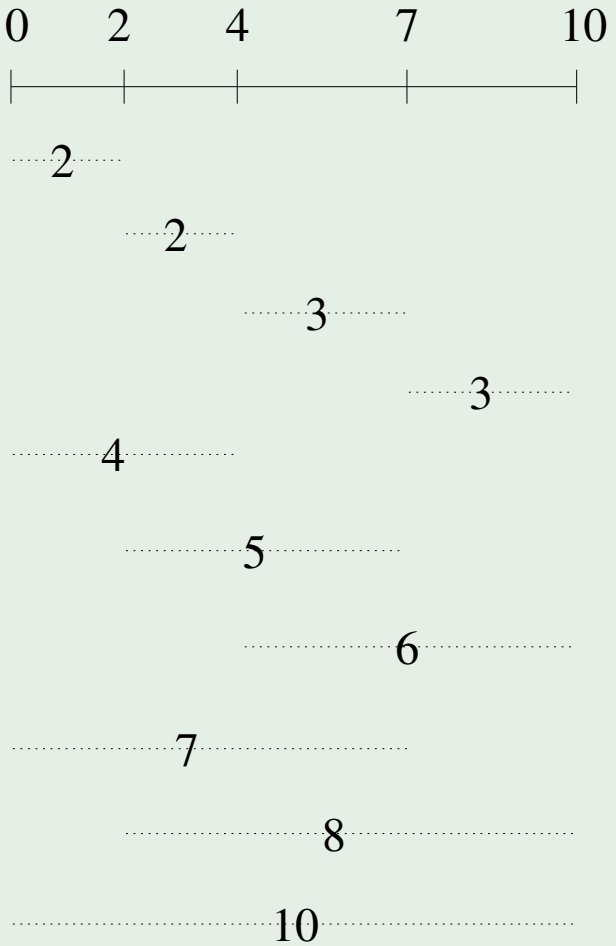
Il problema del compendio parziale

terminologia

- $X = \{x_1 = 0, x_2, \cdots, x_n\}$ insiemi ordinato di punti sulla linea
- $\Delta X = \{x_j - x_i : 1 \leq i < j \leq n\}$ multiinsieme di distanze tra coppie di punti
- La cardinalita' di ΔX e' pari al coefficiente binonmiale $\binom{n}{2}$ ovvero al numero di combinazioni di n punti presi a coppie
- In generale $\binom{n}{k} = \frac{n!}{(n-k)!k!}$, in particolare $\binom{n}{2} = \frac{n(n-1)}{2}$
- Vogliamo ricostruire X dato ΔX

Esempio: Il problema del compendio parziale

Example (Problema Compendio Parziale)



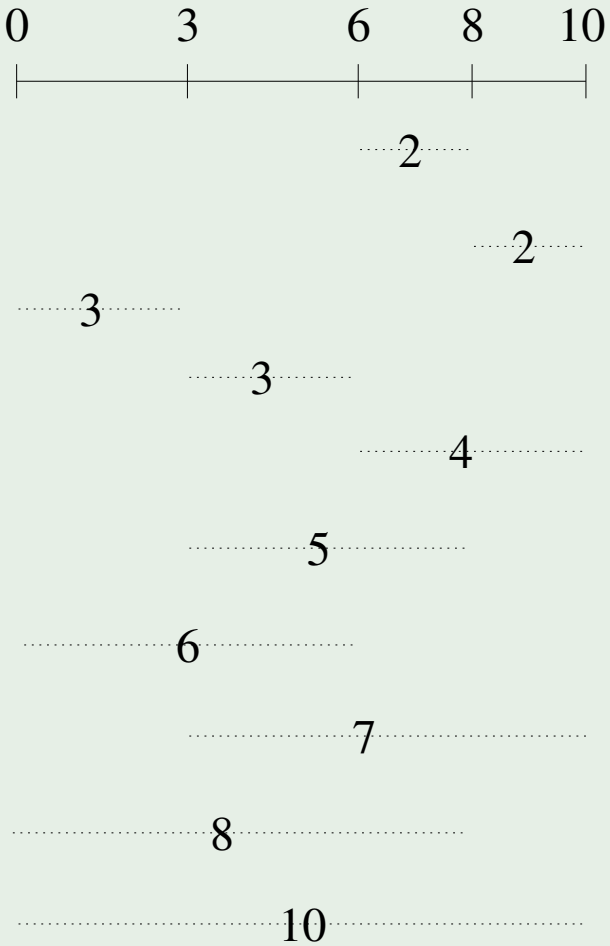
Insiemi omeometrici

Non unicita' di X dato ΔX

- Dato una multiinsieme L di distanze possono esistere piu' X tali che $\Delta X = L$.
- Tali X sono detti omeometrici
- Casi banali: traslare tutti i punti di un dato valore, $\Delta A = \Delta(-A)$
- Esistono anche casi non banali
- In genere in bioniformatica siamo interessati a tutti gli insiemi X omeometrici

Esempio Insiemi omeometrici

Example (Insiemi omeometrici)



Approccio a forza bruta

Approccio a forza bruta

Idee principali

- Calcolo $M =$ il massimo elemento in L
- Calcolo tutti i possibili insiemi di $n-2$ punti X tali che $X = \{0 < x_2, < \cdots < x_{n-1} < M\}$
- Controllo se $\Delta X = L$, se si includo X nel mio insieme di risultati
- Calcolare ΔX da L non e' computazionalmente oneroso
- Considerare tutti i possibili X e' computazionalmente molto oneroso: $\binom{M-2}{n-2}$ possibili insiemi da considerare

Approccio a forza bruta: pseudocodice

pseudocodice

Algorithm 1 bruteForcePDP

Require: L,n
  OUT $\leftarrow \emptyset$
  M $\leftarrow$ Massimo in L
  for all insiemi X di n-2 numeri da 0 a M **do**
    Calcola ΔX a partire da X
    if $\Delta X = L$ **then**
      aggiungi X a OUT
    end if
  end for
  return OUT

Realizzare in Java l'approccio a forza bruta

Realizzazione in java

- Classe che rappresenta multiinsiemi di distanze DistanceList.java
- Classe che implementa l'algoritmo BruteForcePDP.java
- Metodo di BruteForcePDP.java che genera tutti gli insiemi di numeri data una lista di distanze
- Utilizziamo come strutture dati: array, List, Set.

Esercizi

Esercizi su bruteforcePDP

- Implementare il costruttore di DistanceList.java che prende come parametro un insieme di interi
  1. La lista costruita deve essere ordinata (utilizzare il metodo Sort(..) della classe Collections)
  2. Utile utilizzare una lista di appoggio che contiene tutti i punti dell'insieme S, per poter accedere gli elementi in base all posizione.
- Implementare il metodo getMax() della classe DistanceList.java
- Implementare il metodo computePDP secondo lo pseudocodice fornito, utilizzando i metodi messi a disposizione dalla classe DistanceList.java ed il metodo getAllSet(...) della classe BruteForcePDP.java (Nota: scaricare la classe FunzioniRicorsive.java)

Algoritmo di Skiena

Algoritmo di Skiena

Idee principali

- Metodo ricorsivo
- Ad ogni passo estraggo la distanza maggiore da L
- Costruisco X posizionando i punti che possono dare luogo a quella distanza
- Ambiguita': distanza maggiore puo' essere generata dai due estremi dell'insieme X
- Mediamente molto efficiente
- Utilizza ricorsione doppia, caso peggiore complessita' esponenziale
- Caso peggiore: si verifica sempre un caso ambiguo

Algoritmo di Skiena: pseudocodice

pseudocodice

Algorithm 2 PartialDigest(L)

Require: L

width $\leftarrow$ Massimo in L
Elimina width dal L
OUT $\leftarrow \emptyset$
$X \leftarrow \{0, width\}$
Place(L,X,width,OUT)

Algoritmo di Skiena: pseudocodice

pseudocodice per Place

Algorithm 3 Place(L,X,width,OUT)

Require: L,X,width,OUT
  if L vuota **then**
    aggiungi X ad OUT
    return
  end if
  y $\leftarrow$ Massimo in L
  if $\Delta(y, X) \subseteq L$ **then**
    aggiungi y a X, elimina $\Delta(y, X)$ da L
    Place(L,X,width,OUT)
    rimuovi y da X, aggiungi $\Delta(y, X)$ a L
  end if
  if $\Delta(width - y, X) \subseteq L$ **then**
    aggiungi $width - y$ a X, elimina $\Delta(width - y, X)$ da L
    Place(L,X,width,OUT)
    rimuovi $width - y$ da X, aggiungi $\Delta(width - y, X)$ a L
  end if

Realizzare in Java l'approccio di Skiena

Realizzazione in java

- Utilizziamo DistanceList.java per rappresentare la lista di distanze
- Classe SkienaPDP.java realizza l'algoritmo
- Metodo che calcola la lista di distanze di un punto da un insieme di punti ($\Delta(y, X)$)
- Place(...) rappresenta il metodo ricorsivo

Esercizi

Esercizi su SkienaPDP

- Implementare il metodo computeDelta(...) della classe SkienaPDP.java
- Implementare il metodo ricorsivo Place(...) della classe SkienaPDP.java