Lecture 4 (Part 1)

Shortest Superstring Problem

Example set of strings: {001, 100, 101}

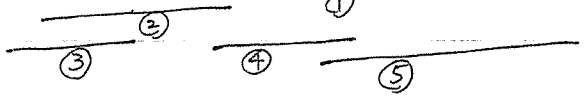
001100101 (Concatenation)

0010100 (Better)

Problem Given a set of strings $\vec{s}_1, \vec{s}_2, \cdots, \vec{s}_n$, find the shortest string $\vec{s}$ such that $\forall\ i = 1, 2, \cdots, n$, $\vec{s}_i$ is a a substring of $\vec{s}$.

Example set of strings {CATGC ①, CTAAGT ②, GCTA ③, TTCA ④, ATGCATC ⑤}.

$\vec{s}$: GCTAAGTTCATGCATC



Greedy Algorithm (把可以接在一起的依次接好，重疊部份愈多愈好。)

Step 1. ① + ⑤ CATGCATC ⑥

Step 2. ② + ③ GCTAAGT ⑦

Step 3. ④ + ⑥ TTCATGCATC ⑧

Step 4 ⑦ + ⑧ GCTAAGTTCATGCATC.

(Answer)

Idea (Prefix graph)

Definition (prefix and overlap)

prefix $(\vec{s}_i, \vec{s}_j)$: First "letters" of $\vec{s}_i$ where overlap $(\vec{s}_i, \vec{s}_j)$ is removed.

overlap $(\vec{s}_i, \vec{s}_j)$: The maximum overlap between $\vec{s}_i$ and $\vec{s}_j$.

$\vec{s}_i$: A C G G C T A T

$\vec{s}_j$: C T A T T A G C

overlap $(\vec{s}_i, \vec{s}_j)$ = C T A T

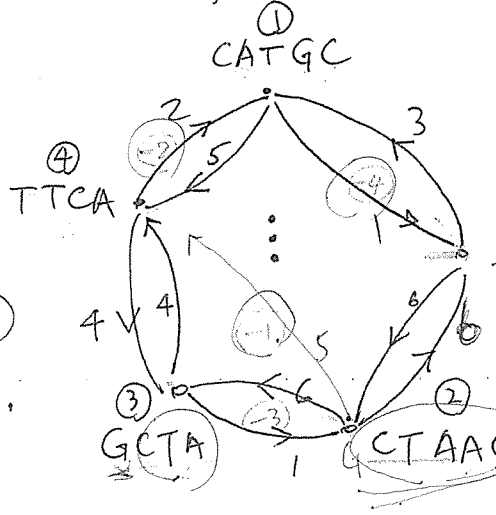
prefix $(\vec{s}_i, \vec{s}_j)$ = A C G G

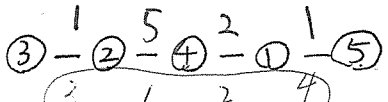
Definition (prefix graph G) Use the idea of Hamiltonian path!

The prefix graph G of a set of strings $\{\vec{s}_1, \vec{s}_2, \cdots, \vec{s}_n\}$ is a double-weighted complete digraph such that $V(G) = \{\vec{s}_1, \vec{s}_2, \cdots, \vec{s}_n\}$ and the weight of $(\vec{s}_i, \vec{s}_j)$ = $|\vec{s}_i| -$ overlap $(\vec{s}_i, \vec{s}_j)$.

(*) Overlapping

Find a Hamilton di-path which maximizes the sum of overlappings.





25 - 10 = 15

G C T A A G T T C A T G C A T C

Sequencing by Hybridization (SBH Problem)
(解定序問題)

Definition (l-mer composition)

For a string $\vec{s}$ of length n, the l-mer composition, or spectrum, of $\vec{s}$, is the multiset of $n-l+1$ l-mers (consecutive l letters, substring of length l) in $\vec{s}$, denoted by $Spectrum(\vec{s}, l)$

(overlap $l-1$ letters)

For example $\vec{s} = ATGCTGCAAG$, $|\vec{s}| = 10$

$l = 3$

$\Rightarrow Spectrum(\vec{s}, l) = \{ATG, TGC, GCT, CTG, TGC, GCA, CAA, AAG\}$

Two l-mers have overlap $l-1$, i.e. $|overlap(\vec{p}, \vec{q})| = l-1$.
$\vec{p}$ and $\vec{q}$ — letters

(*) **SBH Problem**

Let S be a set of l-mers obtained from $\vec{s}$.

Reconstruct $\vec{s}$ from $S = Spectrum(\vec{s}, l)$.

Note: Clearly, we did not know the "order" of elements in S.

Idea: Two consecutive l-mers, $\vec{p}$ and $\vec{q}$ has $overlap(\vec{p}, \vec{q}) = l-1$.

(**) One $\vec{s}$ gives a unique $Spectrum(\vec{s}, l)$ for each $l = 1, 2, \cdots, n-1$.

(***) A given set of l-mers may produce two or more feasible strings, i.e., each of the strings has the same spectrum.

For example: ATGCGTGCA $= \vec{s}_1$

ATGGCGTGCA $= \vec{s}_2$

We may obtain the above two distinct strings from the set Spectrum($\vec{s}$?3) $= \{ATG, TGG, TGC, GTG, GGC, GCA, GCG, CGT\}$.

$v_1 \quad v_2 \quad v_3 \quad v_4 \quad v_5 \quad v_6 \quad v_7 \quad v_8$

$\vec{s}_1$: $v_1 - v_3 - v_7 - v_8 - v_4 - v_2 - v_5 - v_6$.

$\vec{s}_2$: $v_1 - v_2 - v_5 - v_7 - v_8 - v_4 - v_3 - v_6$.

Observation

SBH problem is a particular (special) case of the Shortest Superstring problem, see it? However, in contrast to SSP, there exists a simple linear-time algorithm for the SBH problem.

(*) Construct one $\vec{s}$; not all $\vec{s}$'s.

Approach

① Hamiltonian Path problem (Hard problem) ↓ digraph

Not a linear-time algorithm

Let $V(G) = \{\vec{s}_1, \vec{s}_2, \cdots, \vec{s}_n\}$ and $(\vec{s}_i, \vec{s}_j) \in A(G)$ iff overlap$(\vec{s}_i, \vec{s}_j) = l-1$. Then, a Hamiltonian path (directed) gives an $\vec{s}$.

② Eulerian Path Problem

From the set of ℓ-mers we obtain a set T of $(\ell-1)$-mers $\{\vec{t}_1, \vec{t}_2, \cdots, \vec{t}_m\}$. Then, let $\tilde{G}$ be the graph obtained from letting $V(\tilde{G}) = T$ and $\vec{t}_i$ is incident to $\vec{t}_j$ if and only if overlap $(\vec{t}_i, \vec{t}_j) = \ell - 2$ and $\vec{t}_i \wedge \vec{t}_j$ is an ℓ-mer in S, $\vec{t}_i \wedge \vec{t}_j$ is obtained by combining $\vec{t}_i$ and $\vec{t}_j$ together such that the (suffix of $\vec{t}_i$ and prefix of $\vec{t}_j$) (overlaping portion) occurs exactly once. For example

$A\boxed{TG} \wedge \boxed{TG}G = ATGG.$

ATG
TGG

(*) Then, an eulerian path (directed) of $\tilde{G}$ gives an $\vec{s}$.

(Note) The number of (directed) eulerian circuits in a (connected) balanced digraph (directed eulerian graph) can be enumerated. (BEST Theorem).

Problem (Open) The number of eulerian circuits in an eulerian graph can also be enumerated, but how?

(**) After the short (500-700 bp) DNA reads are sequenced, biologists need to assemble them together to reconstruct the entire genomic DNA sequence. (Fragment Assembly Problem)

$\Leftarrow \ell$?

(Notable Facts)

1. The error rate in DNA reads produced by modern machines varies from 1% to 3%.

2. One never knowns whether a read came from a target strand DNA sequence or from its Watson-Crick complement (DNA is double-stranded).

3. Repeats in DNA cause a lot of trouble in sequencing DNA. For example, roughly 300 nucleotide Alu sequence is repeated more than a million times throughout the genome, with only 1~5% to sequence variation.
(Repeats can occur at several different scales.)

4. Fragment assembly algorithms: (Three steps)
   (1) Overlap: Finding potentially overlapping reads
   (2) Layout: Finding the order of reads along DNA
   (3) Consensus: Deriving the DNA sequence from the layout

Review of Graph Theory

1. Any tournament contains a Hamiltonian path.
   The existence of a superstring is warrant.
2. Any even connected graph contains an eulerian circuit.
   (Euler's Theorem)
3. A graph contains an eulerian path if the graph is connected and has at most two odd vertices.
4. (Kotzig, 1968) Let G be a colored connected graph with (edge) even degree of vertices. Then, there is an alternating eulerian circuit in G if and only if every vertex is balanced, i.e. $\forall$ color c, the number of edges incident to v and colored $d_c(v) \le d(v)/2$. ($d(v)$: the degree of v in G)

Problem

Proof 把每一类的相接边分成 pairs of edges with distinct colors.

Corollary G is bicolored $\Rightarrow d_1(v) = d_2(v)$ for each $v \in V(G)$.

Open problem Find the number of alternating eulerian circuits in a bicolored eulerian graph.

Set Cover Problem

Let $S_1, S_2, \cdots, S_k$ be subsets of S with $wt(S_i) = w_i$, $i = 1, 2, \cdots$, and $\bigcup_{i=1}^{k} S_i \supseteq S$. Find a set of indices $I \subseteq \{1, 2, \cdots, k\}$ such that (1) $\bigcup_{j \in I} S_j \supseteq S$, and (2) $\sum_{j \in I} w_j$ is minimized.

Example $S = [1, 6]$, $S_1 = \{1, 2\}$, $S_2 = \{1, 3\}$, $S_3 = \{2, 3\}$, $S_4 = \{2, 4, 6\}$, $S_5 = \{3, 5, 6\}$, $S_6 = \{4, 5, 6\}$, $S_7 = \{4, 5\}$, $wt(S_i) = |S_i|$.

Natural weight

	S_1	S_2	S_3	S_4	S_5	S_6	S_7
1	1	1	0	0	0	0	0
2	1	0	1	1	0	0	0
3	0	1	1	0	1	0	0
4	0	0	0	1	0	1	1
5	0	0	0	0	1	1	1
6	0	0	0	1	1	1	0

$S_2 \cup S_3 \cup S_6$

(*) In general S_7 is allowed. ($S_7 \subseteq S_6$)

(**) So we are looking for some rows $S_{i_1}, S_{i_2}, \cdots, S_{i_t}$ such that $\bigcup_{j=1}^{t} S_{i_j} \supseteq S$ and $\sum_{j=1}^{t} |S_{i_j}|$ is minimized.

(=)

Natural weight

extra!

(***) $S_1 \cup S_2 \cup S_3 \cup S_6 \supseteq S$ and weight $= 9$. (Can you find a better one? Yes!)

(=)

Example (Real)

IBM finds computer viruses (Wikipedia)

S : 5000 known viruses

Subsets (strings) 9000 substrings of 20 or more consecutive bytes from viruses, not found in "good" code.

(*) A set cover of 180 substrings was found. It suffices to search for these 180 substrings to verify the existence of known computer viruses.

Greedy Algorithm

C : represent the set of elements covered so far

$\alpha_{\tilde{S}}$: average cost per newly covered node $\tilde{S}$

Algorithm

1. $C \leftarrow \emptyset$

2. While $C \neq S$ or $C \subsetneq S$ do

Find the set whose cost effectiveness is smallest, say $\tilde{S}$,

Let $\alpha_{\tilde{S}} = \frac{wt(\tilde{S})}{|\tilde{S} \setminus C|} \geq 1$.

$\tilde{S} \setminus C$ 尚未 cover 的部分

$wt(\tilde{S})$ is given

$|\tilde{S}| / |\tilde{S} \setminus C| = \alpha_{\tilde{S}}$

ee值代表效率 ≥ 1

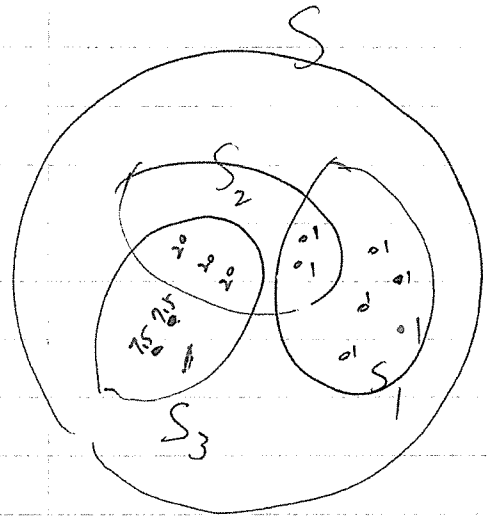
$\forall\ e \in S \setminus C$, set $price(e) = \alpha$

$C \leftarrow C \cup S$

3. Output picked sets

==

Example



$wt(S_1) = 7$
$wt(S_2) = 6$
$wt(S_3) = 15$

==

By "looking", optimal solution is $S_1 \cup S_3$ of weight 22.

By greedy algorithm

1. Choose S_1, $\alpha_{S_1} = \frac{wt(S_1)}{|S_1 \setminus \phi|} = 1$, $C \leftarrow S_1$

2. Choose S_2: $\alpha_{S_2} = \frac{wt(S_2)}{|S_2 \setminus S_1|} = \frac{6}{3} = 2$, $C \leftarrow S_1 \cup S_2$

3. choose S_3: $\alpha_{S_3} = \frac{wt(S_3)}{|S_3 \setminus (S_1 \cup S_2)|} = \frac{15}{2} = 7.5$.

Total: $7 + 6 + 15 = 28$