Exhaustive Search: Restriction Mapping Algorithms

Restriction Mapping

Restriction enzyme. A *restriction enzyme* is a small molecule composed of nucleotides, which has the ability to **cut** (restrict) a large DNA molecule at the place where it binds. An enzyme called `HindII` cuts DNA molecules at all occurrences of nucleotide sequences `GTGCAC` and `GTTAAC`.

An occurrence of the nucleotide sequences `GTGCAC` or `GTTAAC` in a DNA molecule is called a **restriction site**.

Complete Restriction Digest. A biological procedure in which DNA is amplified and then cut into chunks using a *restriction enzyme.* In this procedure, each DNA molecule is cut at each restriction site. The lengths of the chunks obtained from the cutting are measured using *gel electrophoresis.*

Partial Restriction Digest. A biological procedure in which DNA is amplified and the cut into chunks using a *restriction enzyme*, in which every possible cut in a DNA sequence is made in *some*, but *not all* molecules. A partial digest, is the list of lengths of chunks obtained after the procedure using *gel electrophoresis.*

Restriction mapping problem. Given a ***partial restriction digest*** construct a "map" of restriction sites on the DNA molecule.

More formally: let us assume that a partial restriction digest is a *multiset* $D = \{n_1, \ldots, n_k\}$, representing a list of **all** distances between pairs of restriction sites on the DNA molecules. The *restriction mapping problem* is the problem of given D as input, producing as output a sequence of *locations* $p_1 < p_2 < \ldots < p_s$ of the restriction sites of the DNA molecule whose partial restriction digest is D.

Restriction Mapping and Turnpike Problem

Definifion. Given a set $X = \{a_1 \leq a_2 \ldots \leq a_s\}$ numbers, a multiset $\Delta X = \{a_k - a_j | a_k, a_j \in X; k > j\}$ is the set of all pairwise distances between elements of X.

Example. Consider a set $X = \{1, 4, 5, 8\}$. Then the set $\Delta X = \{8-1, 5-1, 8-4, 4-1, 5-4, 8-5\} = \{7, 4, 4, 3, 1, 3\} = \{1, 3, 3, 4, 4, 7\}$.

Turnpike problem. Suppose we know the pairwise distances between all exists on a stretch of a turnpike. Find the locations of all the exits on the turnpike.

Homometry. Two sets A and B are *homometric* if $\Delta A = \Delta B$.

Definition. Let U and V be two sets of numbers. Then,

$$U \oplus V = \{u + v | u \in U, v \in V\}.$$

$$U \ominus V = \{u - v | u \in U, v \in V\}.$$

Theorem. $U \oplus V$ and $U \ominus V$ are homometric for any pair of sets U, V.

Naïve Algorithm

A naïve brute force algorithm for finding restriction mapping works as follows:

ALGORITHM BRUTEFORCEPDP:

1. **Input**: $D = \{n_1, \ldots, n_s\}$
2. Let $M = \max(D)$. The largest element in our output is the distance between M and 0.
3. For each set X of $C_s^2 = k$ numbers $a_1 < a_2 < \ldots < a_k$:
   (a) Compute ΔX.
   (b) If $\Delta X = D$, **output** X.
4. If no solution is found, **output** `"No solution"`.

Analysis. BRUTEFORCEPDP will examine C_{M-1}^{s-2} different sets X in the internal loop, which means it will run in $O(s^{M-1}$ time.

Restriction Mapping with Backtracking.

A more efficient exhaustive search algorithm is due to Skiena et al. [**?**]. The algorithm makes "guesses" on each step, and backtracks if the guesses turn out to be incorrect.

Algorithm outline.

1. **Input:** $D = \{n_1, \ldots, n_s\}$. $X = \emptyset$.
2. **Step 1**: Find $M = \max(n_1, \ldots, n_s)$. Let $X = \{0, M\}$. Remove M from D.
3. **Step** i. If $D = \emptyset$, then **output** X. Otherwise, find $K = \max(D)$ - the largest remaining number in D. Two cases are possible:
   (a) $K \in X$. Do the following:

i. Add K to X.
ii. Compute distances $d_1, \ldots d_i$ from K to all other elements of X.
iii. Remove $d_1, \ldots d_i$ from D.
iv. If some $d_i \notin D$ then proceed to the second case.
v. if all $d_1, \ldots d_i$ successfully removed from D, proceed to step $i+1$ of the procedure.

(b) $M - K \in X$. Do the following:

i. Add $M - K$ to X.
ii. Compute distances $d_1, \ldots d_i$ from $M - K$ to all other elements of X.
iii. Remove $d_1, \ldots d_i$ from D.
iv. If some $d_i \notin D$ then **output FAILURE**.
v. if all $d_1, \ldots d_i$ successfully removed from D, proceed to step $i+1$ of the procedure.

References

[1] S.S. Skiena, W.D. Smith, P. Lemke (1990) Reconstructing Sets From Interpoint Distances. In *Proc. Sixth Annual Symposium on Computational Geometry*, pp. 332-339, Berkeley, CA, June 1990.