Physical Mapping – Restriction Mapping

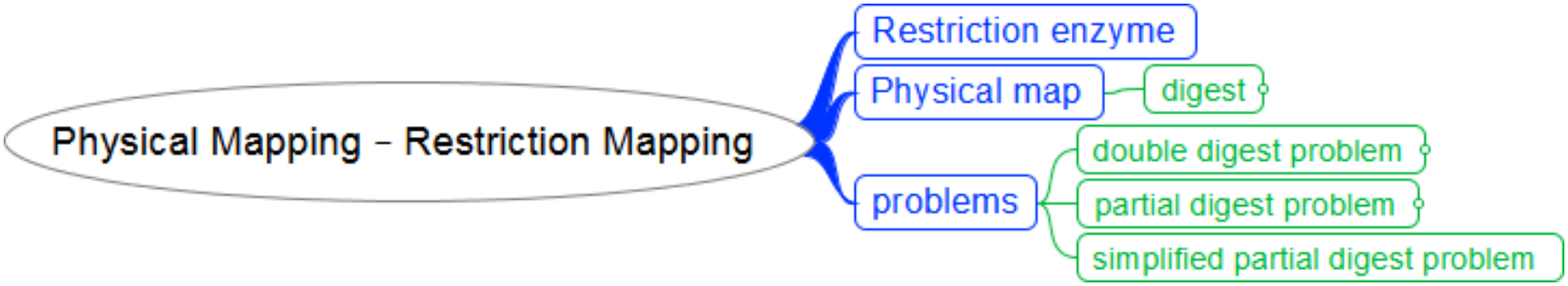
Bioinformatics Algorithms part 2

František Mráz, KSVI

Based on slides from http://bix.ucsd.edu/bioalgorithms/slides.php

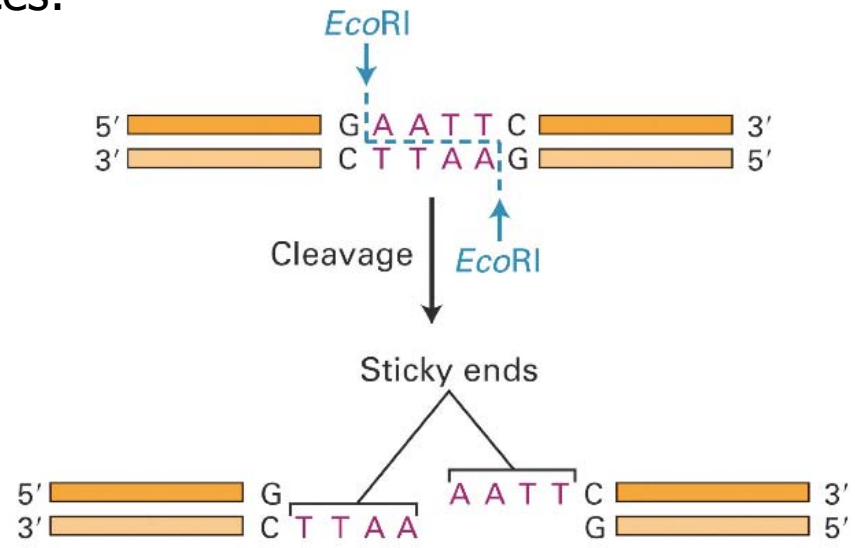
And other sources

Contents


Physical Mapping – Restriction Mapping
Restriction enzyme
Physical map
digest
problems
double digest problem
partial digest problem
simplified partial digest problem

Molecular Scissors – Restriction Enzymes

- *HindII* - first restriction enzyme – was discovered accidentally in 1970 while studying how the bacterium *Haemophilus influenzae* takes up DNA from the virus
- Recognizes and cuts DNA at sequences:
  - GTGCAC
  - GTTAAC



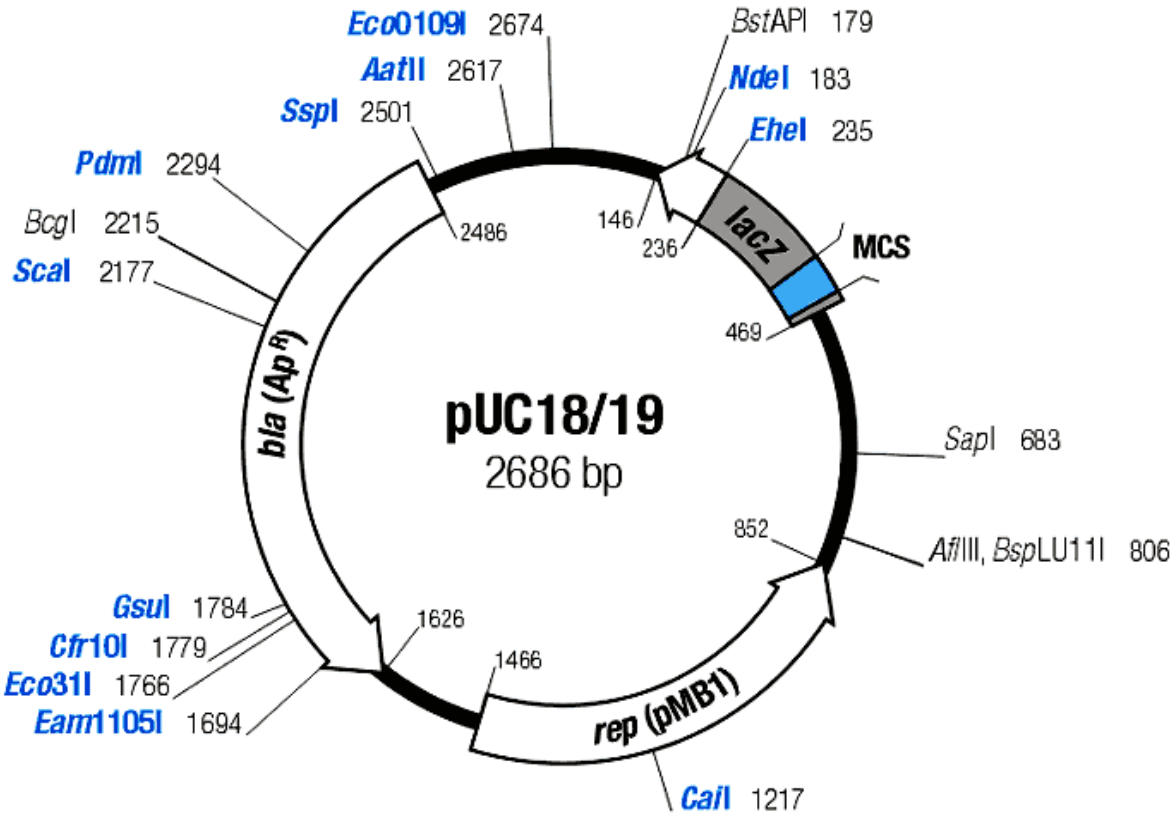
Recognition Sites of Restriction Enzymes

Enzyme	Source	Recognition Sequence	Cut
EcoRI	Escherichia coli	5'GAATTC	5'---G AATTC---3'
		3'CTTAAG	3'---CTTAA G---5'
BamHI	Bacillus amyloliquefaciens	5'GGATCC	5'---G GATCC---3'
		3'CCTAGG	3'---CCTAG G---5'
HindIII	Haemophilus influenzae	5'AAGCTT	5'---A AGCTT---3'
		3'TTCGAA	3'---TTCGA A---5'
MstII	Microcoleus species	5'CCTNAGG	5'---CC TNAGG---3'
		3'GGANTCC	3'---GGANT CC---5'
TaqI	Thermus aquaticus	5'TCGA	5'---T CGA---3'
		3'AGCT	3'---AGC T---5'
NotI	Nocardia otitidis	5'GANTC	5'---GC GGCCGC---3'
		3'CTNAG	3'---CGCCGG CG---5'
HinfI	Haemophilus influenzae	5'GANTC	5'---G ANTC---3'
		3'CTNAG	3'---CTNA G---5'
AluI*	Arthrobacter luteus	5'AGCT	5'---AG CT---3'
		3'TCGA	3'---TC GA---5'

* = blunt ends

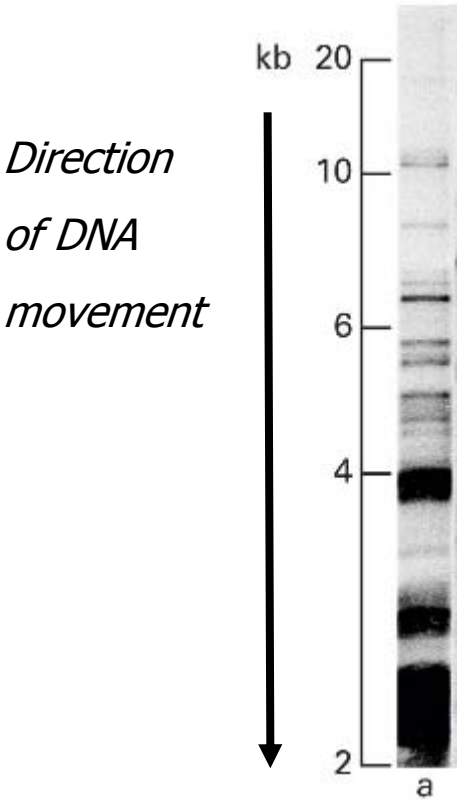
Restriction Maps

- A map showing positions of restriction sites in a DNA sequence
- If DNA sequence is known then construction of restriction map is a trivial exercise
- In early days of molecular biology DNA sequences were often unknown
- Biologists had to solve the problem of constructing restriction maps without knowing DNA sequences



Measuring Length of Restriction Fragments

- Restriction enzymes break DNA into restriction fragments.
- Gel electrophoresis is a process for separating DNA by size and measuring sizes of restriction fragments
- Visualization: autoradiography or fluorescence



Physical Map, Restriction Mapping Problem

- **Definition:** Let S be a DNA sequence. A **physical map** consists of a set M of markers and a function $p : M \rightarrow \mathbf{N}$ that assigns each marker a position of M in S.

  $\mathbf{N}$ denotes the set of nonnegative integers

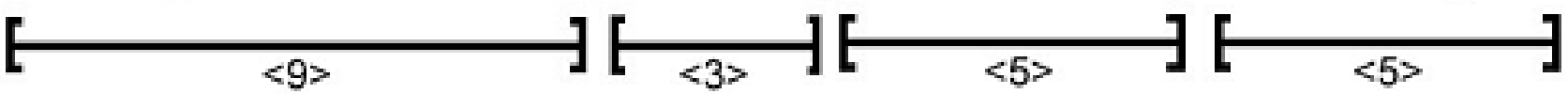
- For a set X of points on the line, let

$$\delta X = \{ | x_1 - x_2 | : x_1, x_2 \in X \}$$

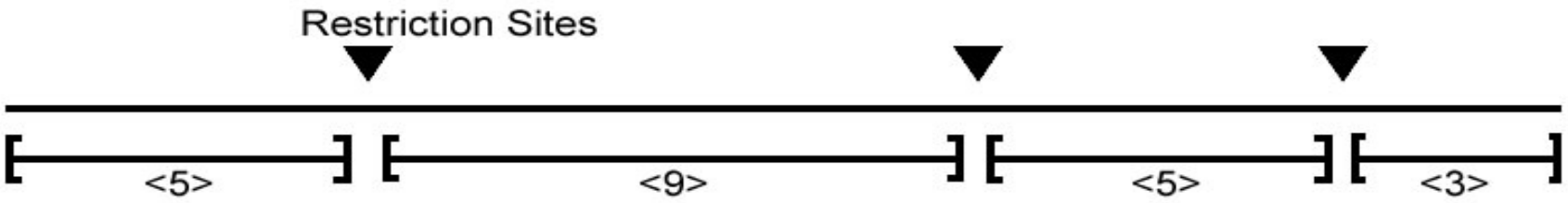
  denote the **multiset** of all pairwise distances between points in X called **partial digest**. In the ***restriction mapping problem***, a ***subset*** $E \subseteq \delta X$ (of experimentally obtained fragment lengths) is given and the task is to reconstruct X from E.

Full Restriction Digest: Multiple Solutions

- Reconstruct the order of the fragments from the sizes of the fragments {3,5,5,9}



- Alternative ordering of restriction fragments:



- Reconstruction from the full restriction digest is impossible.

Three different problems

- One (full) digest is not enough
  - Use 2 restriction enzymes
  - Use 1 restriction enzyme, but differently

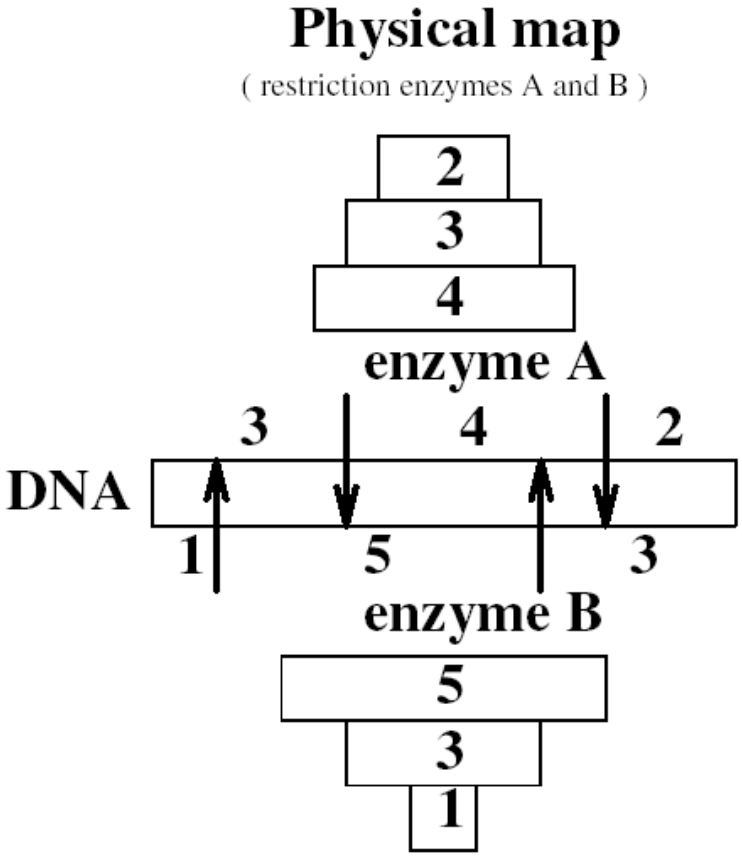
1. The ***double digest*** *problem* – DDP
2. The ***partial digest*** *problem* – PDP
3. The ***simplified partial digest problem*** – SPDP

Double Digest Mapping

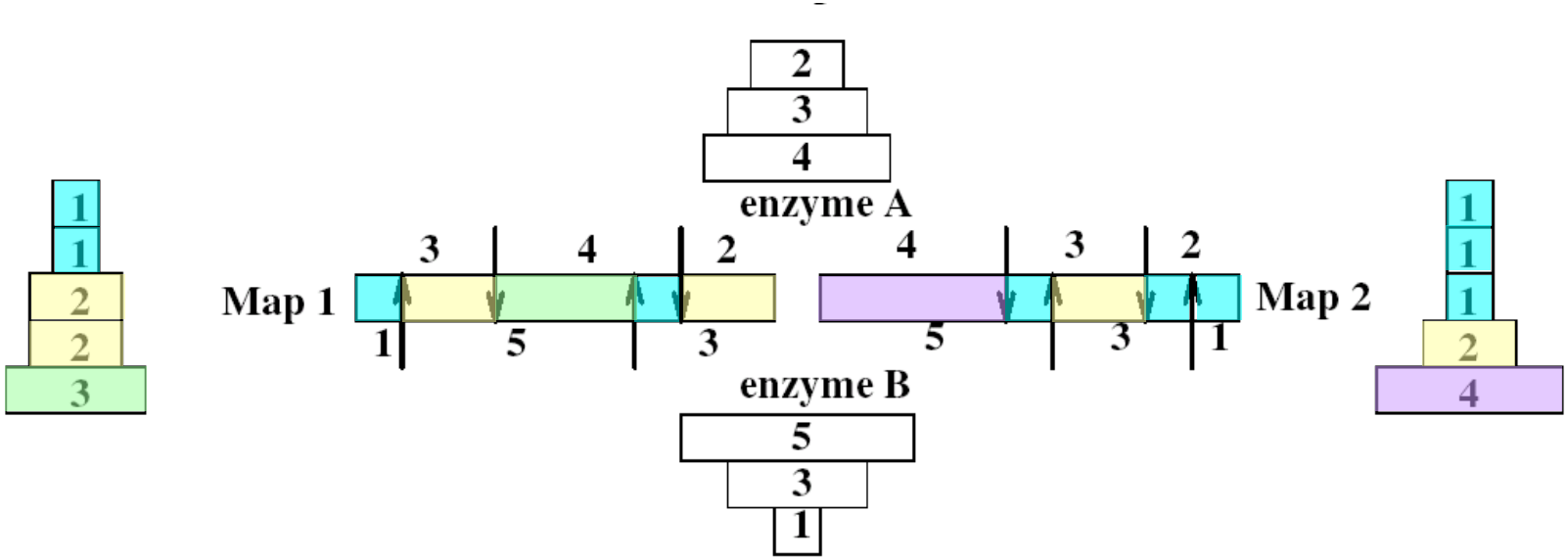
- Use two restriction enzymes; three full digests:
  - ΔA – a complete digest of S using A,
  - ΔB – a complete digest of S using B, and
  - ΔAB – a complete digest of S using both A and B.

- Computationally, Double Digest problem is more complex than Partial Digest problem

Double Digest: Example



Double Digest: Example



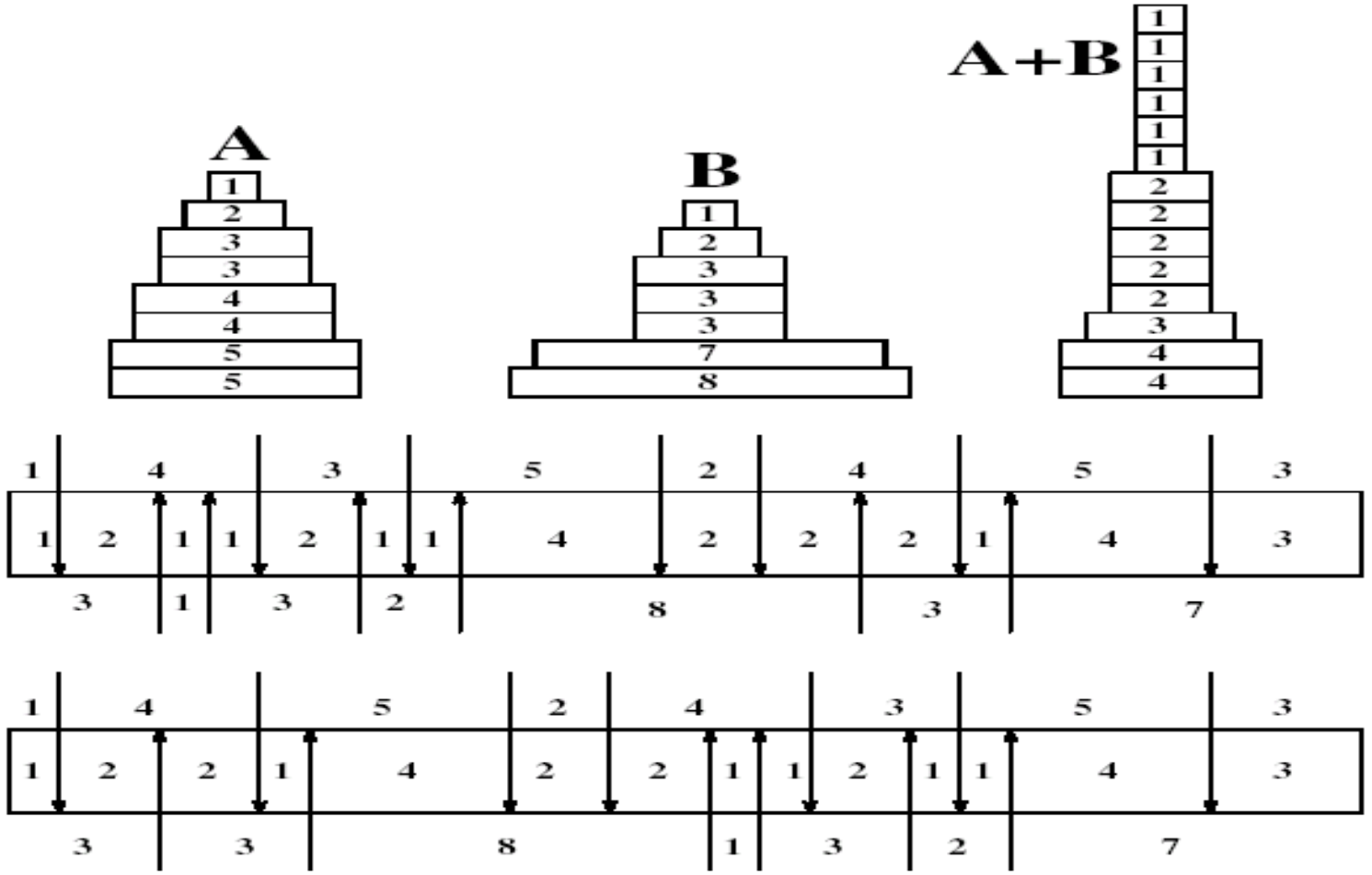
Without the information about X (i.e. ΔAB), it is impossible to solve the double digest problem as this diagram illustrates

Double Digest Problem

Input: ΔA – fragment lengths from the complete digest with enzyme ***A***.
ΔB – fragment lengths from the complete digest with enzyme ***B***.
ΔAB – fragment lengths from the complete digest with *both* ***A*** and ***B***.

Output: ***A*** – location of the cuts in the restriction map for the enzyme ***A***.
B – location of the cuts in the restriction map for the enzyme ***B***.

Double Digest: Multiple Solutions



Double digest

- The decision problem of the DDP is NP-complete.
- All algorithms have problems with more than 10 restriction sites for each enzyme.
- A solution may not be unique and the number of solutions grows exponentially.
- DDP is a favourite mapping method since the experiments are easy to conduct.

DDP is NP-complete

1) DDP is in NP (easy)
2) given a (multi-)set of integers $X = \{x_1, \ldots, x_n\}$. The *Set Partitioning Problem (SPP)* is to determine whether we can partition X into two subsets X_1 and X_2 such that

   This problem is known to be NP-complete.

$$\sum_{x \in X_1} x = \sum_{x \in X_2} x$$

DDP is NP-complete

- Let X be the input of the SPP, assuming that the sum of all elements of X is even. Then set
  - $\Delta A = X$,
  - $\Delta B = \left\{\frac{K}{2}, \frac{K}{2}\right\}$. with $K = \sum_{i=1}^{n} x_i$, and
  - $\Delta AB = \Delta A$.
- then there exists an integer n_0 and indices $\{j_1, j_2, \ldots j_n\}$ with

$$\sum_{i=1}^{n_0} x_{j_i} = \sum_{i=n_0+1}^{n} x_{j_i}$$

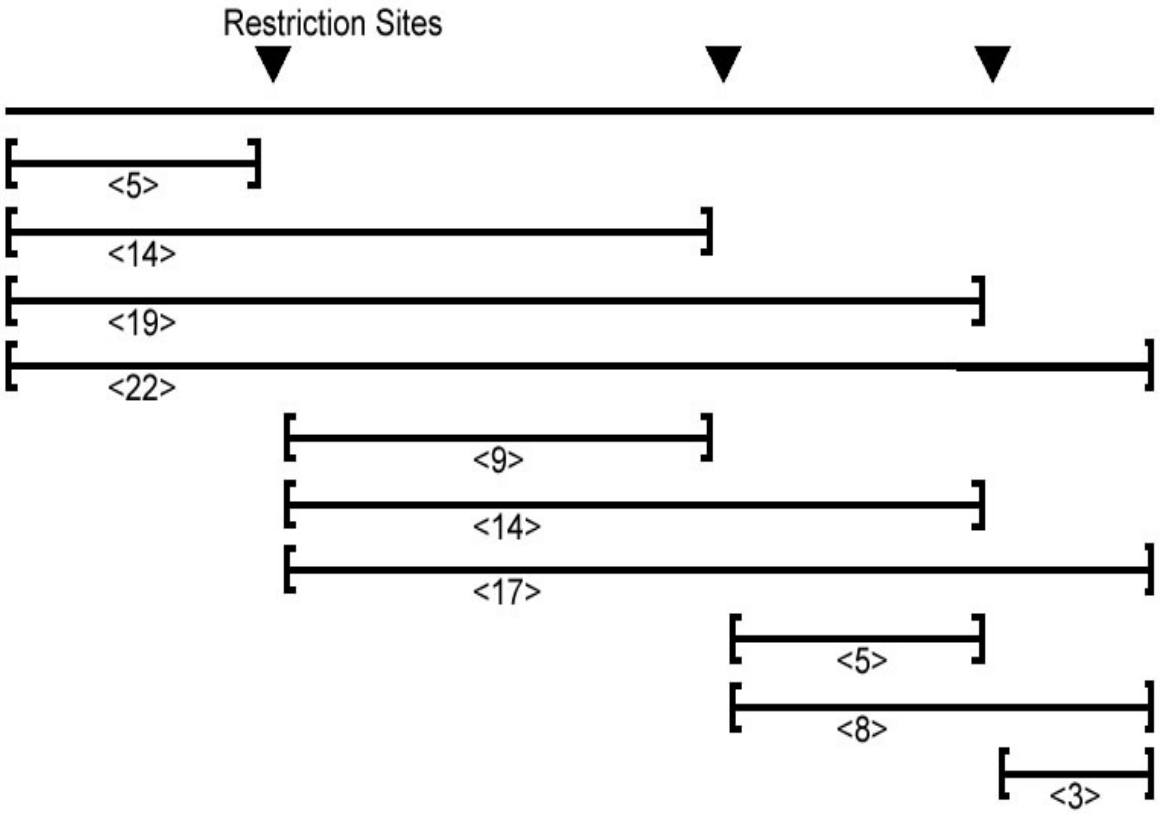
because of the choice of ΔB and ΔAB. Thus a solution for the SPP exists. Thus SPP is a DDP in which one of the two enzymes produced only two fragments of equal length.

Partial Restriction Digest

- The sample of DNA is exposed to the restriction enzyme for only a limited amount of time to prevent it from being cut at all restriction sites.
- This experiment generates the set of all possible restriction fragments between every two (not necessarily consecutive) cuts.
- This set of fragment sizes is used to determine the positions of the restriction sites in the DNA sequence.

Multiset of Restriction Fragments

- We assume that multiplicity of a fragment can be detected, i.e., the number of restriction fragments of the same length can be determined (e.g., by observing twice as much fluorescence intensity for a double fragment than for a single fragment)



Multiset: {3, 5, 5, 8, 9, 14, 14, 17, 19, 22}

Partial Digest Fundamentals

X: the set of ***n*** integers representing the location of all cuts in the restriction map, including the start and end

n: the total number of cuts

δ***X:*** the multiset of integers representing lengths of each of the fragments produced from a partial digest

One More Partial Digest Example

X	0	2	4	7	10
0		2	4	7	10
2			2	5	8
4				3	6
7					3
10					

Representation of $\delta\boldsymbol{X}$ = {2, 2, 3, 3, 4, 5, 6, 7, 8, 10} as a two dimensional table, with elements of

$$\boldsymbol{X} = \{0, 2, 4, 7, 10\}$$

along both the top and left side. The elements at (i, j) in the table is

$$x_j - x_i \text{ for } 1 \leq i < j \leq n.$$

Partial Digest Problem: Formulation

- **Goal***:* Given all pairwise distances between points on a line, reconstruct the positions of those points.

- **Input:** The multiset of pairwise distances L, containing $n(n-1)/2$ integers.

- **Output:** A set X, of n integers, such that $\delta X = L$.

Partial Digest: Multiple Solutions

- It is not always possible to uniquely reconstruct a set X based only on δX.
- For example, the set $X = \{0, 2, 5\}$

  and $(X + 10) = \{10, 12, 15\}$

  both produce $\delta X = \{2, 3, 5\}$ as their partial digest set.
- The sets {0,1,2,5,7,9,12} and {0,1,5,7,8,10,12} present a less trivial example of non-uniqueness. They both digest into:

  {1, 1, 2, 2, 2, 3, 3, 4, 4, 5, 5, 5, 6, 7, 7, 7, 8, 9, 10, 11, 12}

Homometric Sets

	0	1	2	5	7	9	12
0		1	2	5	7	9	12
1			1	4	6	8	11
2				3	5	7	10
5					2	4	7
7						2	5
9							3
12							

	0	1	5	7	8	10	12
0		1	5	7	8	10	12
1			4	6	7	9	11
5				2	3	5	7
7					1	3	5
8						2	4
10							2
12							

Partial Digest: Brute Force

1. Find the restriction fragment of maximum length ***M***. ***M*** is the length of the DNA sequence.
2. For every possible set

   $$\boldsymbol{X} = \{0, x_2, \ldots, x_{n-1}, M\}$$

   compute the corresponding $\delta\boldsymbol{X}$
3. If $\delta\boldsymbol{X}$ is equal to the experimental partial digest ***L***, then ***X*** is the correct restriction map

BruteForcePDP

BruteForcePDP(L, n):

$M \leftarrow$ maximum element in L

for every set of $n - 2$ integers $0 < x_2 < \ldots x_{n-1} < M$

$X \leftarrow \{0, x_2, \ldots, x_{n-1}, M\}$

Form δX from X

if $\delta X = L$

return X

output "no solution"

- BruteForcePDP takes $O(M^{n-2})$ time since it must examine all possible sets of positions.
- One way to improve the algorithm is to limit the values of x_i to only those values which occur in L.

AnotherBruteForcePDP

AnotherBrutForcePDP(*L*, *n*)

M ← maximum element in *L*

for every set of *n* – 2 integers $0 < x_2 < \ldots x_{n-1} < M$ **from *L***

$X \leftarrow \{ 0, x_2, \ldots, x_{n-1}, M \}$

Form δX from *X*

if $\delta X = L$;

return *X*

output "no solution"

- It is more efficient, but still slow
- If ***L*** = {2, 998, 1000} (***n*** = 3, ***M*** = 1000), BruteForcePDP will be extremely slow, but AnotherBruteForcePDP will be quite fast
- Fewer sets are examined, but runtime is still exponential:

$O(n^{2n-4})$

Branch and Bound Algorithm for PDP

1. Begin with $X = \{0\}$
2. Remove the largest element in L and place it in X
3. See if the element *fits* on the right or left side of the restriction map
4. When it fits, find the other lengths it creates and remove those from L
5. Go back to step 2 until L is empty

Branch and Bound Algorithm for PDP

1. Begin with $X = \{0\}$
2. Remove the largest element in L and place it in X
3. See if the element *fits* on the right or left side of the restriction map
4. When it fits, find the other lengths it creates and remove those from L
5. Go back to step 2 until L is empty

WRONG ALGORITHM

Defining D(y, X)

- Before describing PartialDigest, first define

$$D(y, X)$$

as the multiset of all distances between point y and all other points in the set $\boldsymbol{X}$

$$D(y, X) = \{|y - x_1|, |y - x_2|, \ldots, |y - x_n|\}$$

for $X = \{x_1, x_2, \ldots, x_n\}$

PartialDigest Algorithm

- S. Skiena

PartialDigest(*L*):

width ← Maximum element in *L*

DELETE(*width*, *L*)

X ← {0, *width*}

PLACE(*L*, *X*)

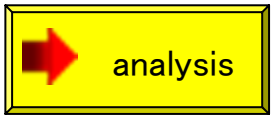
PartialDigest Algorithm (cont' d)

PLACE(L, X):
  if L is empty
    output X
    return
  $y \leftarrow$ maximum element in L
if $D(y, X) \subseteq L$
    Add y to X and remove lengths $D(y, X)$ from L
    PLACE(L, X)
    Remove y from X and add lengths $D(y, X)$ to L
  if $D(width - y, X) \subseteq L$
    Add $(width - y)$ to X and remove lengths $D(width - y, X)$ from L
    PLACE(L, X)
    Remove $(width - y)$ from X and add lengths $D(width - y, X)$ to L
  return

analysis

An Example

L = { 2, 2, 3, 3, 4, 5, 6, 7, 8, 10 }
X = { 0 }



An Example

L = { 2, 2, 3, 3, 4, 5, 6, 7, 8, 10 }
X = { 0 }

Remove 10 from ***L*** and insert it into ***X***. We know this must be the length of the DNA sequence because it is the largest fragment.

An Example

L = { 2, 2, 3, 3, 4, 5, 6, 7, 8, 10 }

X = { 0, 10 }

An Example

L = { 2, 2, 3, 3, 4, 5, 6, 7, 8, 10 }
X = { 0, 10 }

Take 8 from ***L*** and make ***y*** = 2 or 8. But since the two cases are symmetric, we can assume ***y*** = 2.

An Example

L = { 2, 2, 3, 3, 4, 5, 6, 7, 8, 10 }
X = { 0, 10 }

We find that the distances from $y=2$ to other elements in X are D($\boldsymbol{y}$, $\boldsymbol{X}$) = {8, 2}, so we remove {8, 2} from $\boldsymbol{L}$ and add 2 to $\boldsymbol{X}$.

An Example

L = { 2, 2, 3, 3, 4, 5, 6, 7, 8, 10 }

X = { 0, 2, 10 }

An Example

L = { 2, 2, 3, 3, 4, 5, 6, 7, 8, 10 }
X = { 0, 2, 10 }

Take 7 from $\boldsymbol{L}$ and make $y = 7$ or $y = 10 - 7 = 3$. We will explore $\boldsymbol{y} = 7$ first, so D(y, X) = {7, 5, 3}.

An Example

L = { 2, 2, 3, 3, 4, 5, 6, 7, 8, 10 }
X = { 0, 2, 10 }

For y = 7 first, D(y, X) = {7, 5, 3} = {|7 – 0|, |7 – 2|, |7 – 10|}.
Therefore we remove {7, 5 ,3} from ***L*** and add 7 to ***X***.

An Example

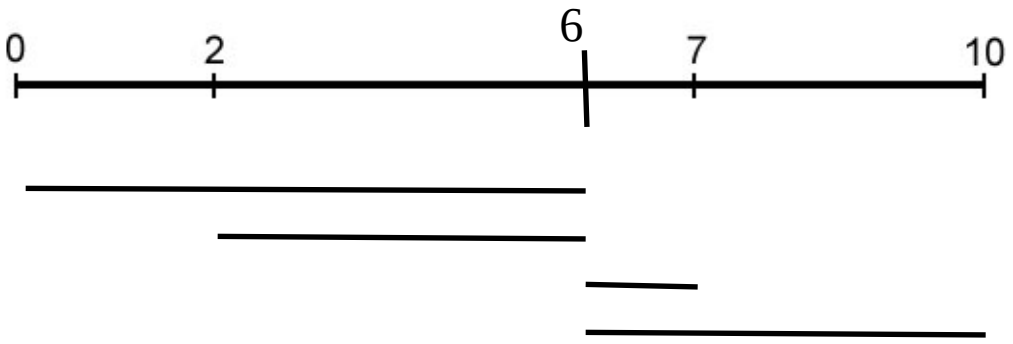
L = { 2, 2, 3, 3, 4, 5, 6, 7, 8, 10 }

X = { 0, 2, 7, 10 }

An Example

L = { 2, 2, 3, 3, 4, 5, 6, 7, 8, 10 }
X = { 0, 2, 7, 10 }

Take 6 from ***L*** and make ***y*** = 6. Unfortunately
D(y, X) = {6, 4, 1 ,4}, which is not a subset of ***L***. Therefore we won't explore this branch.



An Example

L = { 2, 2, 3, 3, 4, 5, 6, 7, 8, 10 }
X = { 0, 2, 7, 10 }

This time make y = 4. D(y, X) = {4, 2, 3 ,6}, which is a subset of ***L*** so we will explore this branch. We remove {4, 2, 3 ,6} from ***L*** and add 4 to ***X***.

An Example

L = { 2, 2, 3, 3, 4, 5, 6, 7, 8, 10 }

X = { 0, 2, 4, 7, 10 }

An Example

L = { 2, 2, 3, 3, 4, 5, 6, 7, 8, 10 }
X = { 0, 2, 4, 7, 10 }

L is now empty, so we have a solution, which is X.

An Example

L = { 2, 2, 3, 3, 4, 5, 6, 7, 8, 10 }
X = { 0, 2, 7, 10 }

To find other solutions, we backtrack.

An Example

L = { 2, 2, 3, 3, 4, 5, 6, 7, 8, 10 }
X = { 0, 2, 10 }

More backtrack.

An Example

L = { 2, 2, 3, 3, 4, 5, 6, 7, 8, 10 }
X = { 0, 2, 10 }

This time we will explore y = 3. D(y, X) = {3, 1, 7}, which is not a subset of L, so we won't explore this branch.

An Example

L = { 2, 2, 3, 3, 4, 5, 6, 7, 8, 10 }
X = { 0, 10 }

We backtracked back to the root. Therefore we have found all the solutions.

Analyzing PartialDigest Algorithm

- Still exponential in worst case, but is very fast on average
- Informally, let *T(**n**)* be time PartialDigest takes to place *n* cuts
  - No branching case: $T(n) < T(n-1) + O(n)$
    - Quadratic
  - Branching case: $T(n) < 2T(n-1) + O(n)$
    - Exponential

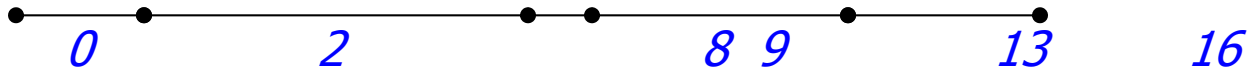
PDP analysis

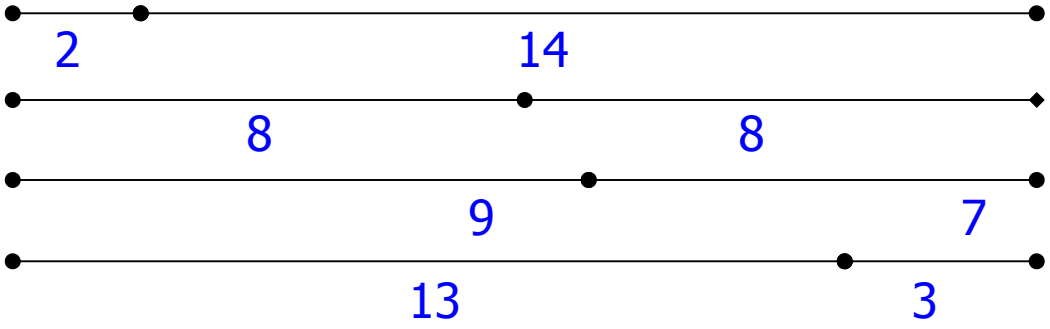
- No polynomial time algorithm is known for PDP. In fact, the complexity of PDP is an open problem.
- *PartialDigest Algorithm* by *S. Skiena* performs well in practice, but may require exponential time.
- This approach is not a popular mapping method, as it is difficult to reliably produce *all* pairwise distances between restriction sites.

Simplified partial digest problem

- Given a target sequence *S* and a single restriction enzyme *A*. Two different experiments are performed
- on two sets of copies of *S*:
  - In the ***short*** experiment, the time span is chosen so that each copy of the target sequence is cut precisely once by the restriction enzyme. Let $\Gamma = \{\gamma_1, \ldots, \gamma_{2N}\}$ be the multi-set of all fragment lengths obtained by the short experiment, where *N* is the number of restriction sites in *S*, and
  - In the ***long*** experiment, a complete digest of *S* by *A* is performed. Let $\Lambda = \{\lambda_1, \ldots, \lambda_{N+1}\}$ be the multi-set of all fragment lengths obtained by the long experiment.

SPDP

- Example: Given these (unknown) restriction sites (in kb):
- 

- We obtain Λ = {2*kb*, 6*kb*, 1*kb*, 4*kb*, 3*kb*} from the long experiment.
- The short experiment yields:
- 

- Γ = {2*kb*, 14*kb*, 8*kb*, 8*kb*, 9*kb*, 7*kb*, 13*kb*, 3*kb*}

SPDP

- In the following we assume that $\Gamma = \{\gamma_1, \ldots, \gamma_{2N}\}$ is sorted in non-decreasing order.
- For each pair of fragment lengths γ_i and γ_{2N-i+1}, we have

  $\gamma_i + \gamma_{2N-i+1} = L$, where L is the length of S.
- Each such pair $\{\gamma_i, \gamma_{2N-i+1}\}$ of *complementary lengths* corresponds to precisely one restriction site in the target sequence S, which is either at position γ_i or at position γ_{2N-i+1}.
- Let $P_i = \langle \gamma_i, \gamma_{2N-i+1} \rangle$ and $P_{2N-i+1} = \langle \gamma_{2N-i+1}, \gamma_i \rangle$ denote the two possible orderings of the pair $\{\gamma_i, \gamma_{2N-i+1}\}$. We call the first component a of any such ordered pair $P = \langle a, b \rangle$ the *prefix* of P.

SPDP

- We obtain a set X of putative restriction site positions as follows: For each complementary pair $\{\gamma_i, \gamma_{2N-i+1}\}$, we choose one of the two possible orderings P_i and P_{2N-i+1}, and then add the corresponding prefix to X.
- Any such ordered choice $X = \langle x_1, \ldots, x_N \rangle$ of putative restriction sites gives rise to a multi-set of integers $R = \{r_1, \ldots, r_{N+1}\}$, with
- $$r_i := \begin{cases} x_i & \text{if } i = 1 \\ x_i - x_{i-1} & \text{if } i = 2, \ldots, N \\ L - x_N & \text{if } i = N + 1. \end{cases}$$

SPDP

- **Simplified Partial Digest Problem (SPDP):** *Given multi-sets* Γ *and* Λ *of fragment lengths, determine a choice of orderings of all complementary fragment lengths in* Γ *such that the arising set* R *equals* Λ.
- **Example:** In the example above we have
- $\Gamma = \{2kb, 3kb, 7kb, 8kb, 8kb, 9kb, 13kb, 14kb\}$
- $\Lambda = \{2kb,,6kb, 1kb, 4kb, 3kb\}$
- We obtain $P_1 = \langle 2, 14 \rangle$, $P_8 = \langle 14, 2 \rangle$,
  $P_2 = \langle 3, 13 \rangle$, $P_7 = \langle 13, 3 \rangle$,
  $P_3 = \langle 7, 9 \rangle$, $P_6 = \langle 9, 7 \rangle$,
  $P_4 = \langle 8, 8 \rangle$, $P_5 = \langle 8, 8 \rangle$.

  Because of the long experiment we obtain $Q = \{P_1, P_7, P_6, P_4\}$ and $X = \{2, 8, 9, 13\}$, from which we get $R = \{2, 6, 1, 4, 3\}$, our restriction site map.

SPDP

- **Simplified Partial Digest Problem (SPDP):** *Given multi-sets* Γ *and* Λ *of fragment lengths, determine a choice of orderings of all complementary fragment lengths in* Γ *such that the arising set* R *equals* Λ.
- **Example:** In the example above we have
- $\Gamma = \{2kb, 3kb, 7kb, 8kb, 8kb, 9kb, 13kb, 14kb\}$
- $\Lambda = \{2kb,,6kb, 1kb, 4kb, 3kb\}$
- We obtain $P_1 = \langle 2, 14 \rangle$, $P_8 = \langle 14, 2 \rangle$,
  $P_2 = \langle 3, 13 \rangle$, $P_7 = \langle 13, 3 \rangle$,
  $P_3 = \langle 7, 9 \rangle$, $P_6 = \langle 9, 7 \rangle$,
  $P_4 = \langle 8, 8 \rangle$, $P_5 = \langle 8, 8 \rangle$.

  Because of the long experiment we obtain $Q = \{P_1, P_7, P_6, P_4\}$ and $X = \{2, 8, 9, 13\}$, from which we get $R = \{2, 6, 1, 4, 3\}$, our restriction site map.

SPDP – algorithm

- the algorithm generates all possible choices of ordered pairs – when called with variable i, it considers both alternatives P_i and P_{2N-i+1}.
- During a call, the current list of restriction sites $X = \langle x_1, \ldots, x_k \rangle$ and the list $R = \langle r_1, \ldots, r_k, r_{k+1} \rangle$ of all fragment lengths are passed as a parameter. Note that $x_1 < x_2 < \ldots < x_k$.
- When processing a new corresponding pair of fragment lengths, the last element r_{k+1} of the list R is replaced by two new fragment lengths that arise because the last fragment is split by the new restriction site.
- Initially, X and R are empty.
- **SPDP** ($X = \langle x_1, \ldots, x_k \rangle$, $R = \langle r_1, \ldots, r_k, r_{k+1} \rangle$, i)

Index of the next pair

Already placed restriction sites

Corresponding fragments

The last fragment can be split by further restrictions sites

SPDP – algorithm

Already placed restriction sites

Corresponding fragments

The last fragment can be split by further restrictions sites

Index of the next pair

Algorithm SPDP ($X = \langle x_1, \ldots, x_k \rangle$, $R = \langle r_1, \ldots, r_k, r_{k+1} \rangle$, i):

if $k = N$ **and** $R = \Lambda$ **then** print X // output putative restriction sites

else if $i \leq 2N$ **then**

Consider $P_i = \langle a, b \rangle$

if $b \notin X$ **then** // the reversed ordering of P_i was

// not used

if $k = 0$ **then**

Set $R' = \langle a, b \rangle$, $X' = \langle a \rangle$

if $a \in \Lambda$ **then** call SPDP($X', R', i+1$)

else

Set $p = a - (L - r_{k+1})$ and $q = L - a$ // new fragment lengths,

// $a - (L - r_{k+1})$ equals $a - x_k$ for $k \geq 1$

if $p \in \Lambda$ **then**

Set $R' = \langle r_1, \ldots, r_k, p, q \rangle$

Set $X' = \langle x_1, \ldots, x_k, a \rangle$ // add a to the set of restriction sites

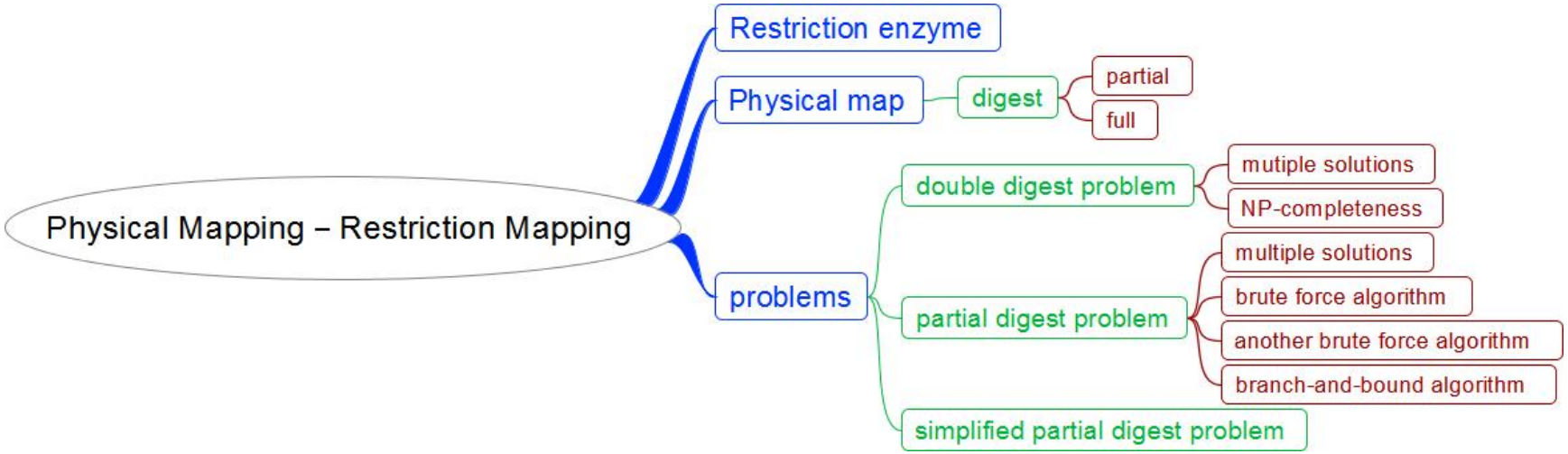
Call SPDP($X', R', i + 1$) // continue using a in this tree's lineage

Call SPDP($X, R, i + 1$) // consider other alternative

SPDP – algorithm

- Clearly, the worst case running time complexity of this algorithm is exponential. However, it seems to work quite well in practice.
- This algorithm is designed for ideal data. In practice there are two problems:
  1. Fragment length determination by gels leads to *imprecise measurements*, down to about 2 – 7% in good experiments. This can be addressed by using interval arithmetic in the above algorithm.
  2. The second problem is *missing fragments*. The SPDP does not suffer from this problem much because both digests are easy to perform. Moreover, the short experiment must give rise to complementary values and any failure to do so can be detected. The long experiment should give rise to precisely $N + 1$ fragments.

Summary


Physical Mapping – Restriction Mapping
Restriction enzyme
Physical map
digest
partial
full
problems
double digest problem
mutiple solutions
NP-completeness
partial digest problem
multiple solutions
brute force algorithm
another brute force algorithm
branch-and-bound algorithm
simplified partial digest problem