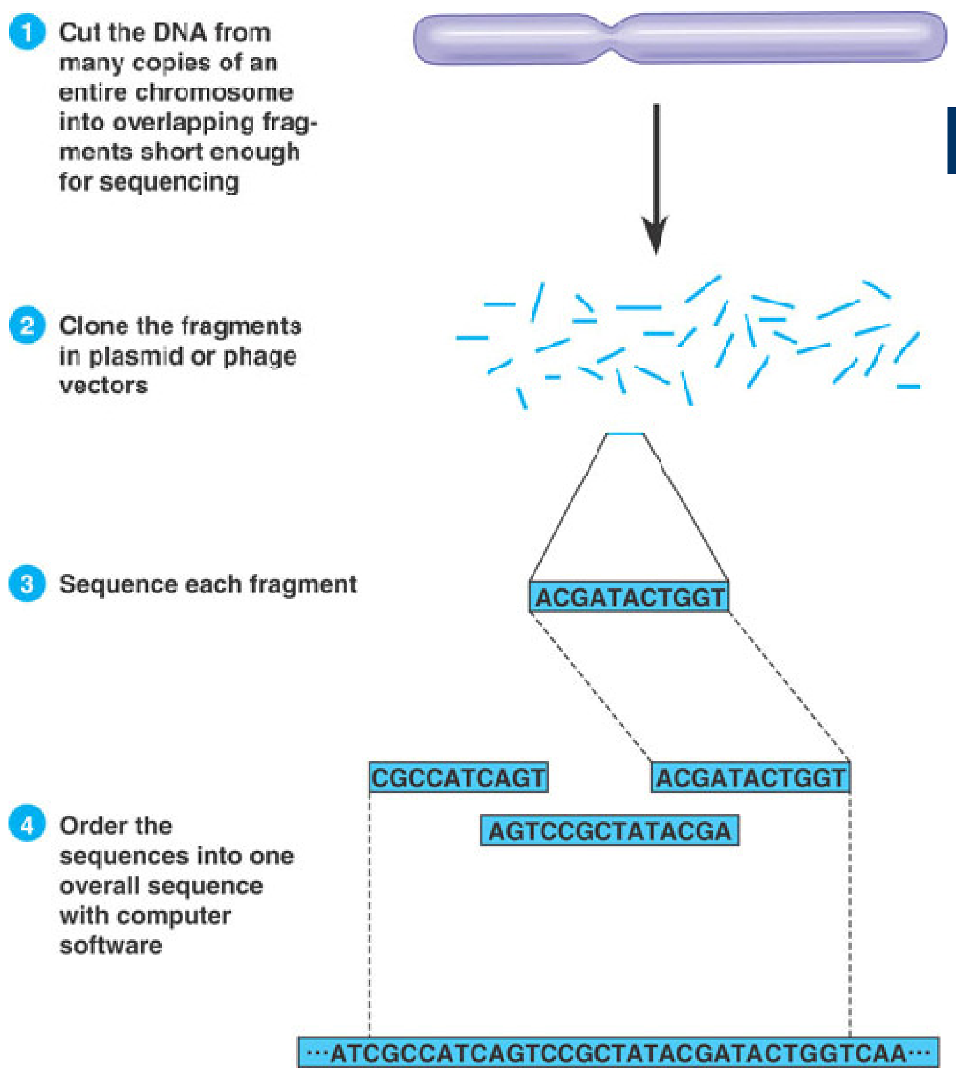
Fragment Assembly

--overview of the genome sequencing process

Introduction

- Q: **What is Fragment Assembly?**
- A: Aligning and merging fragments of a much longer DNA sequence in order to reconstruct the original sequence.

Shotgun Sequencing



1 Cut the DNA from many copies of an entire chromosome into overlapping fragments short enough for sequencing
2 Clone the fragments in plasmid or phage vectors
3 Sequence each fragment
ACGATACTGGT
4 Order the sequences into one overall sequence with computer software
CGCCATCAGT
ACGATACTGGT
AGTCCGCTATACGA
···ATCGCCATCAGTCCGCTATACGATACTGGTCAA···

Introduction

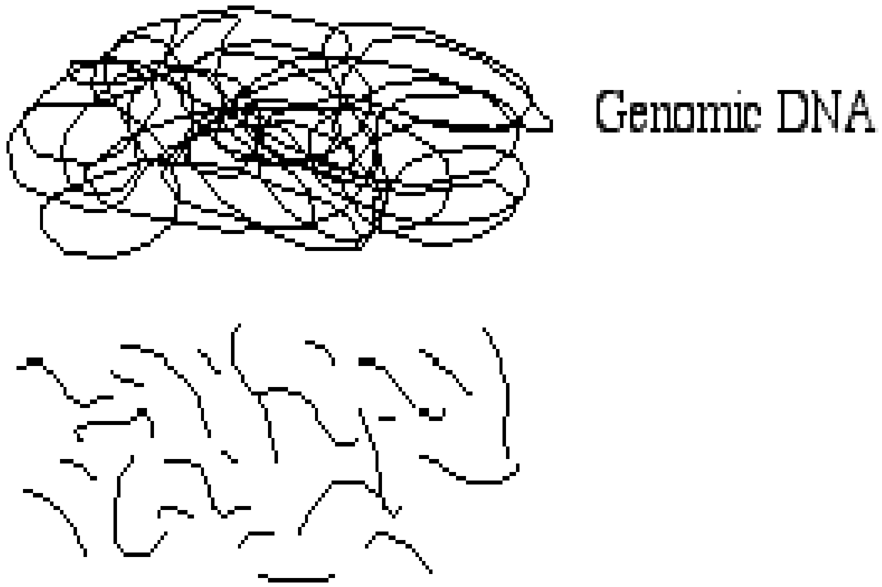
- **Solution**
  - **Break** the DNA into small fragments randomly
  - **Sequence** the readable fragment directly
  - **Assemble** the fragment together to reconstruct the original DNA

Solving a one-dimensional jigsaw puzzle with millions of pieces(without the box) !

1. Break
2. Sequence
3. Assemble
4. Conclusion

Break

❖DNA can be cutten into pieces through mechanical means



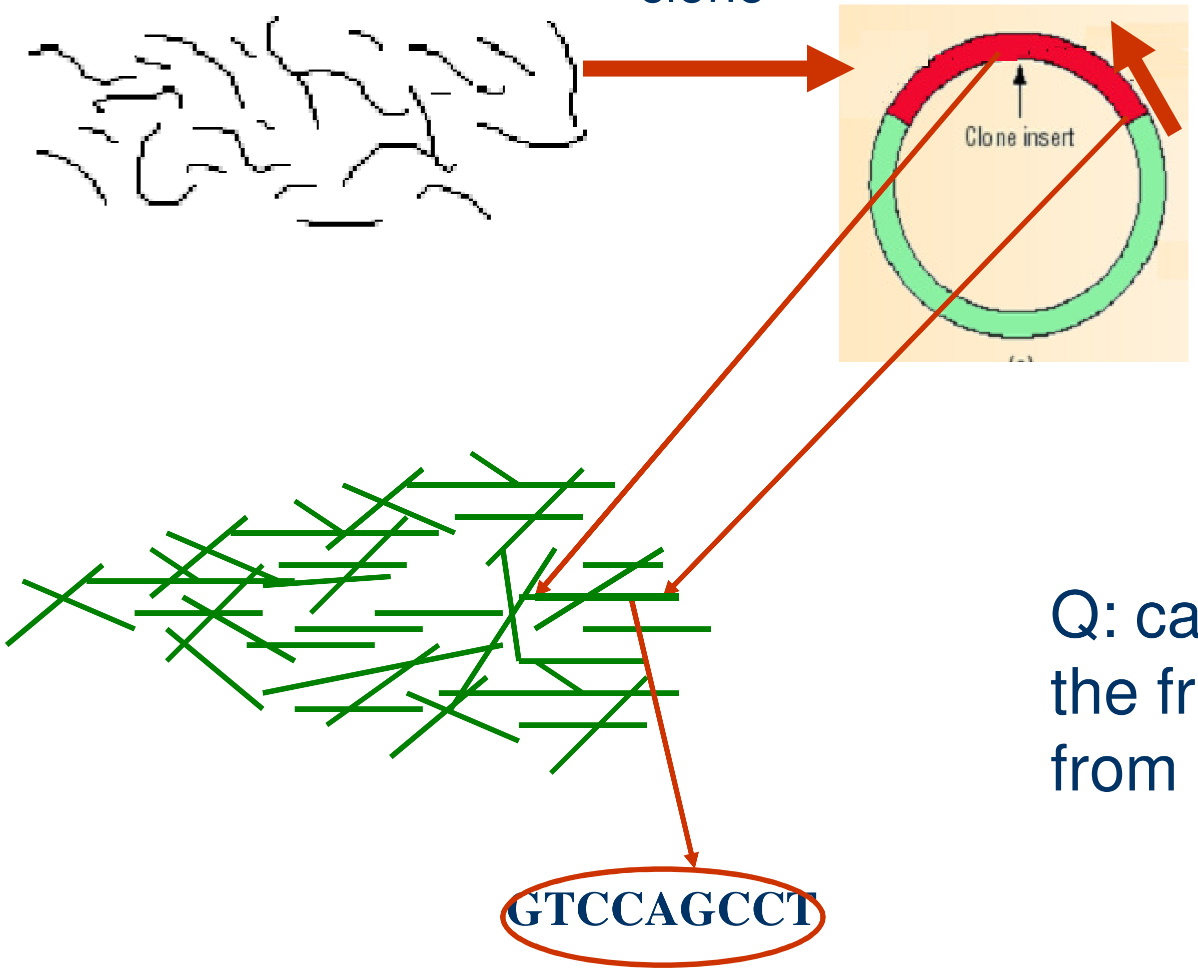
Issues in Break

❖ **How?**

- Coverage: The whole fragments provide an 8X oversampling of the genome
- Random Libraries with pieces sizes of 2,4,6,10, 12 and 40 k bp were produced
- Clone: Obtaining several copies of the original genome and fragments

1. Break
2. Sequence
3. Assemble
4. Conclusion

Sequence



Directed sequencing

(GEL)

Q: can we read the fragment from both end?

1. Break
2. Sequence
3. Assemble
4. Conclusion

3. Assemble

- **A Simple Example**

ACCGT

CGTGC

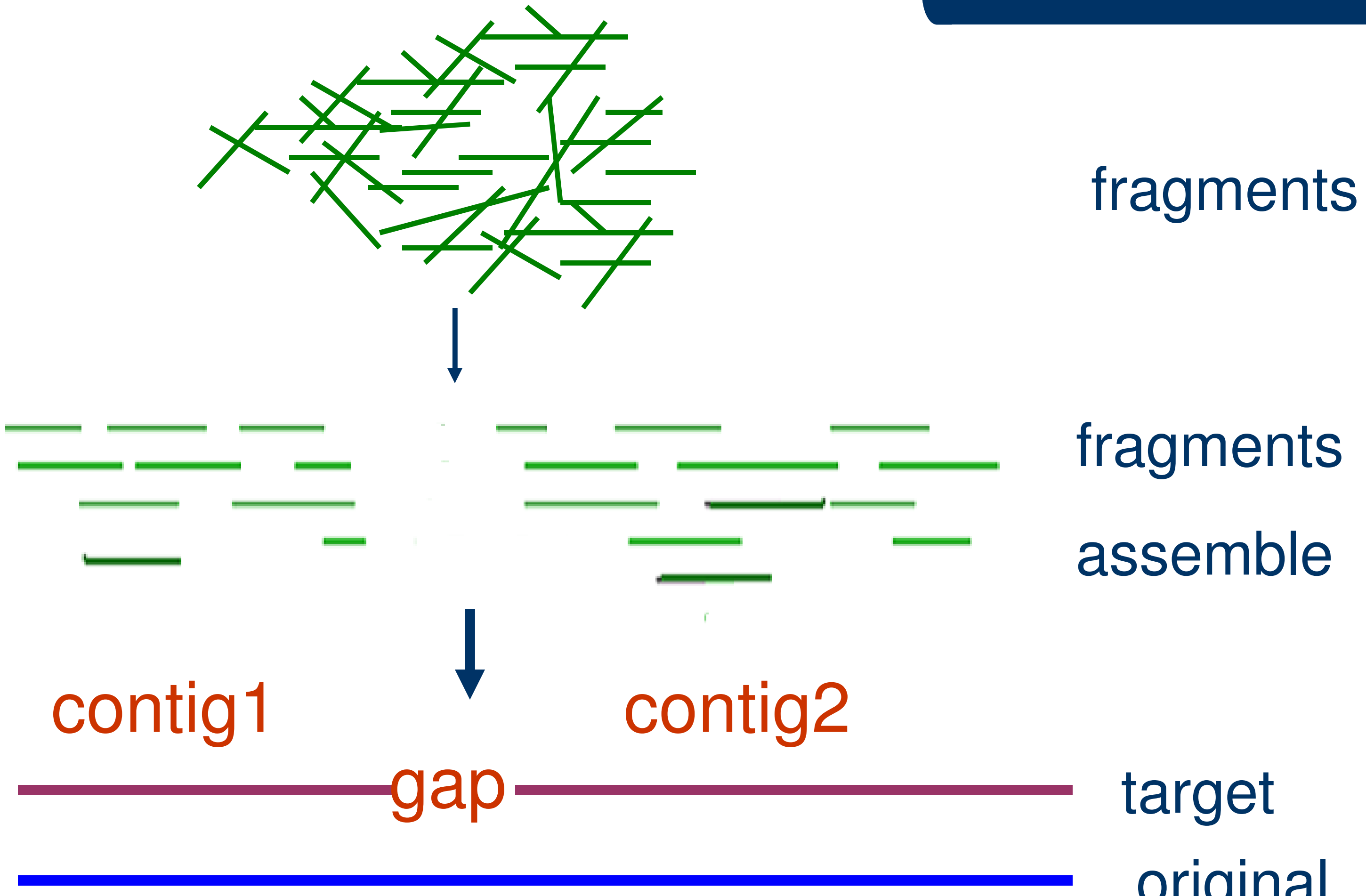
TTAC

```
--ACCGT
----CGTGC
TTAC
---------
TTACCGTGC
```

Overlap: The suffix of a fragment is same as the prefix of another.

Assemble: align multiple fragments into single continuous sequence based on fragment overlap

3. Assemble


fragments
fragments
assemble
contig1
contig2
gap
target
original

A simple model

- The simplest, naive approximation of DNA assemble corresponds to **Shortest Common Superstring Problem (SCS)**: Given a set of string s1, ... , sn, find the shortest string s such that each si appears as a substring of s.

SCS

```
--ACCGT
----CGTGC
TTAC
_________
TTACCGTGC
```

(1) Overlap step

Create an overlap graph in which every node is a fragment and edges indicate an overlap

(2) Layout step

Determine which overlaps will be used in the final assembly, find an optimal spanning forest on the overlap graph

Overlap step

Finding overlap

- Compare each fragment with other fragments to find whether there's overlap on its end part and another's beginning part.

We call 'a overlap b' when a's suffix equal to b's prefix

Overlap step

Overlap graph

- Directed, weighted graph G(V,E,w)
- V: set of fragments
- E : set of directed edge indicates the overlap between two fragments. An edge <a,b,w> means an overlap between a and b with weight w. this equal to suffix(a,w)=prefix(b,w)

Example

W=AGTATTGGCAATC

Z=AATCGATG

U=ATGCAAACCT

X=CCTTTTGG

Y=TTGGCAATCA

S=AATCAGG

$$\begin{tikzcd}
 & S & & Y \arrow[ll, "5"'] & \\
W \arrow[urrr, "9"] \arrow[dr, "4"'] & & & & X \arrow[ul, "4"'] \\
 & Z \arrow[r, "3"] & U \arrow[urr, "3"] & &
\end{tikzcd}$$

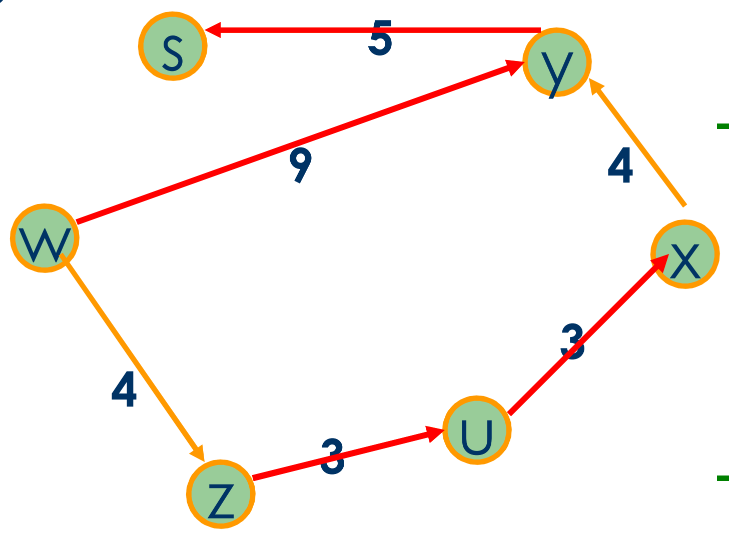
Layout step

- Looking for shortest common superstring is the same as looking for path of maxium weight
- Using greedy algorithm to select a edge with the best weight at every step.
- The selected edge is checked by Rule. If this check is accepted, the edge is accepted, otherwise omit this edge
- Rule: for either node on this edge, indegree and outdegree <=1; Acyclic

- At last the fragments merged together , from the point of graph, it is a forest of hamitonian paths(a path through the graph that contains each node at most once)., each path correspond to a contig

Example

W=AGTATTGGCAATC
Z=AATCGATG
U=ATGCAAACCT
X=CCTTTTGG
Y=TTGGCAATCA
S=AATCAGG



W->Y->S

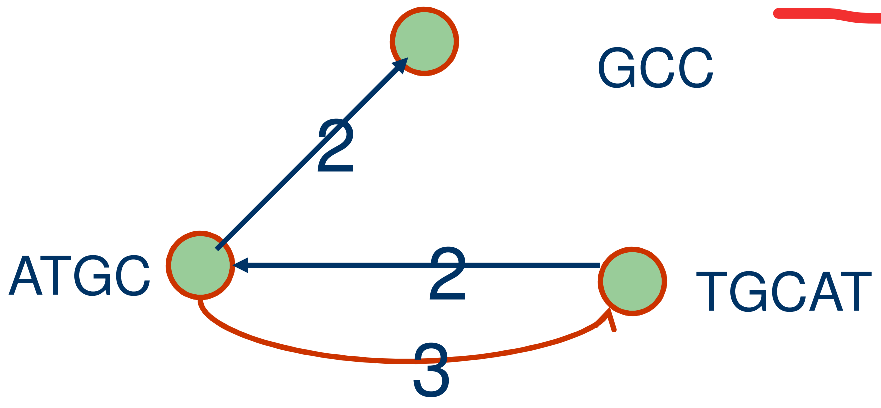
```
AGTATTGGCAATC
      TTGGCAATCA
             AATCAGG
AGTATTGGCAATCAGG
```

Z->U->X

```
AATCGATG
     ATGCAAACCT
              CCTTTTGG
AATCGATGCAAACCT TTTGG
```

[illegible]

- Geedy Algorithm is neither optimal nor complete, and will introduce gap



[illegible]

- Can't correctly model the assembly problem due to complication in the real problem instance

Complication with Assemble

- Sequencing errors. **Most sequencers have around 1% error in the best case.**
- Unknown orientation. **Could have sequenced either strand.**
- Bias in the reads. **Not all regions of the sequence will be covered equally.**
- Repeats. **There is much repetitive sequence, especially in human and higher plants**

Sequenceing Errors

Fragments contains3 kinds of errors: insert, deletion, substitution

Possibility :Substitutions (0.5-2%), insert and deletion occur roughly 10 times less frequently

Problems with the simple model - Errors

x:ACCGT

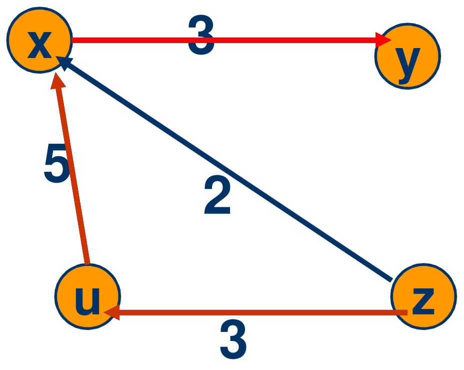
Y:CGTGC

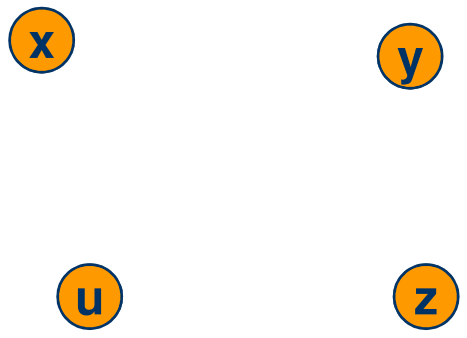
A

Z:TTAC

U:TACCGT

G





```
--ACCGT
----CGTGC
TTAC
-TACCGT
TTACCGTGC
```

Problems with the simple model - Errors

Solution

Allow for bounded number of mismatches between overlapping fragments ----- Approximate overlaps

Criterion: minimum overlap length(40 bps), error rate(less than 6% mismatches)

How?

Using semi-global alignment to find the best match between the suffix of one sequence and the prefix of another.

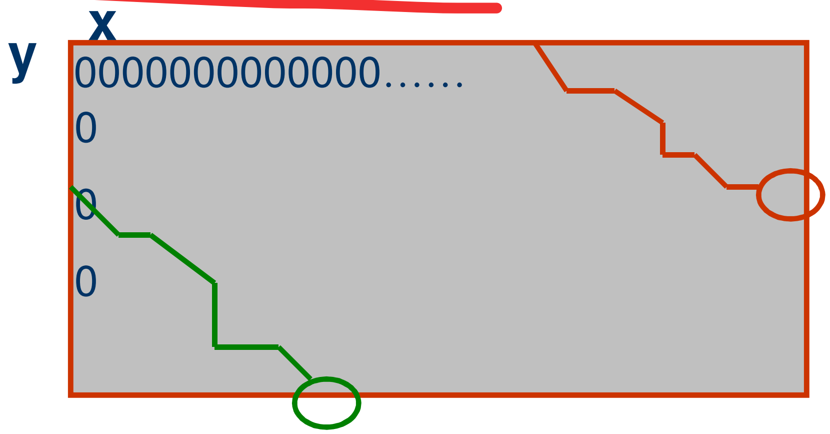
semi-global alignment

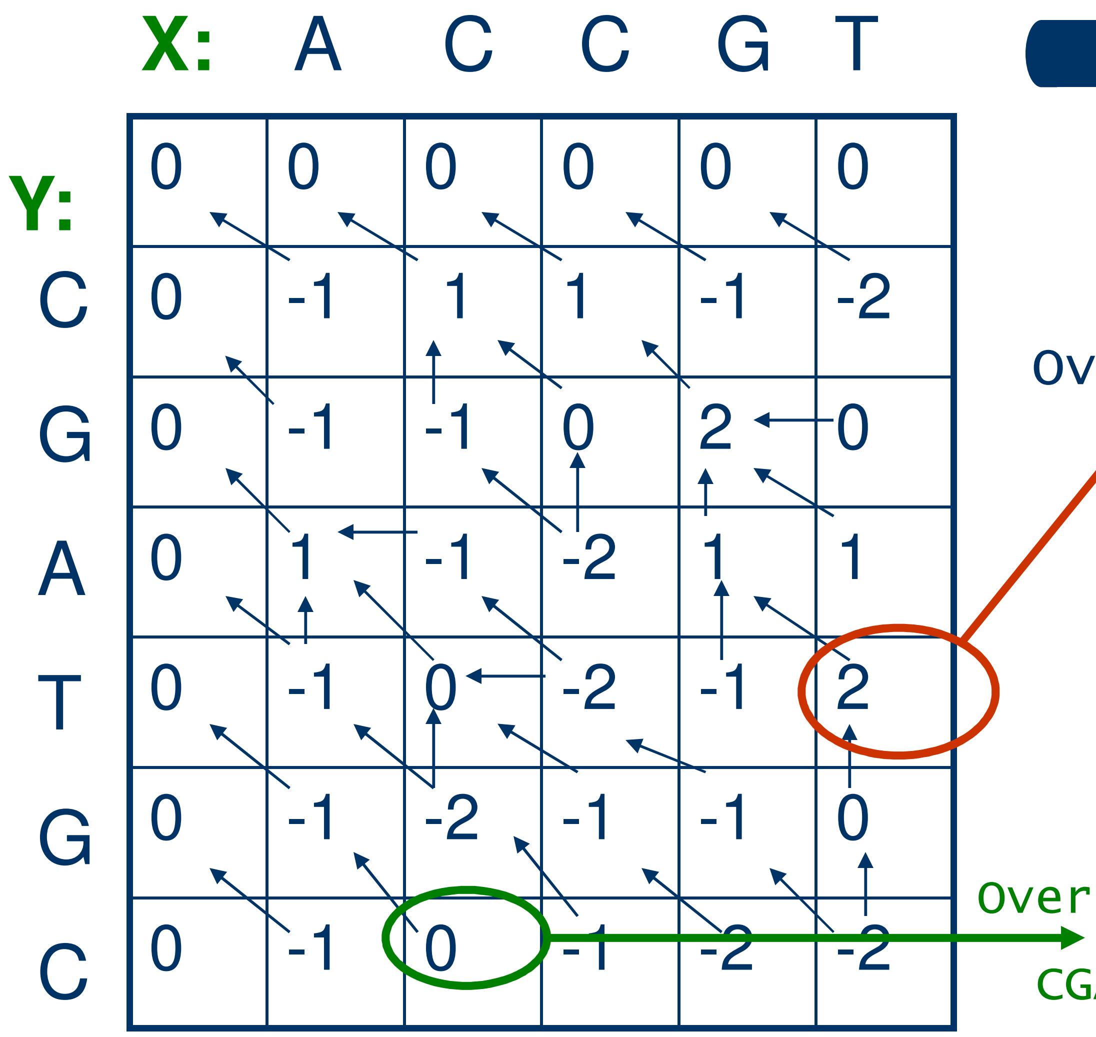
Score system: 1 for matches, -1 for mismatches, -2 for gaps

Initializing the first row and first column of zero, ignore gap in both extremities

Algorithm is same as global comparision

Search last column for higest score and obtain alignment by tracing back to start point (overlap of x over y). overlap of y over x corresponds to the max in the last row





Y: \ X:		A	C	C	G	T
	0	0	0	0	0	0
C	0	-1	1	1	-1	-2
G	0	-1	-1	0	2	0
A	0	1	-1	-2	1	1
T	0	-1	0	-2	-1	2
G	0	-1	-2	-1	-1	0
C	0	-1	0	-1	-2	-2

```
Overlap:x->y
 ACCG-T-
 --CGATGC
```

```
Overlap: y->x
 CGATGC---
 ----ACCGT
```

Problems with the simple model - Errors

x:ACCGT

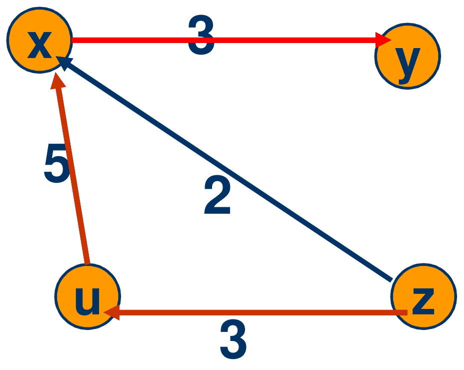
Y:CGTGC

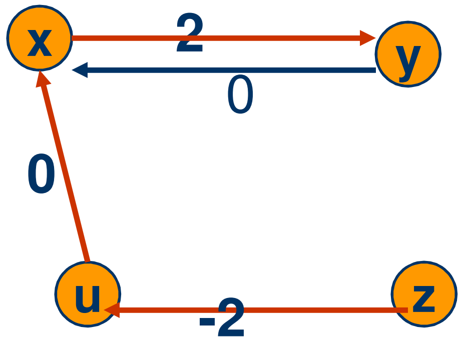
A

Z:TTAC

U:TACCGT

G





Criterion

1.Score>-3

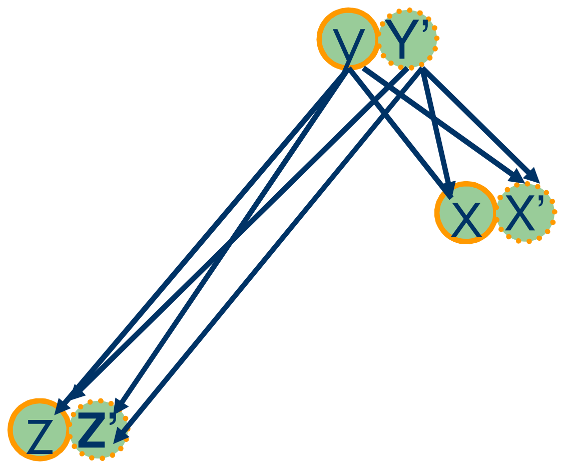
2. Mismatch<2

```
--ACCGT
----CGTGC
TTAC
-TACCGT
_________
TTACCGTGC
```

```
--ACCG-T
----CGATGC
TT-C
-TAGCGT
_________
TTACCGTGC
```

Problems with the simple model - Unkown orientation

Unknowns Orientation:
Fragments can be read from both of the DNA strands.
Solution
Try all possible combination

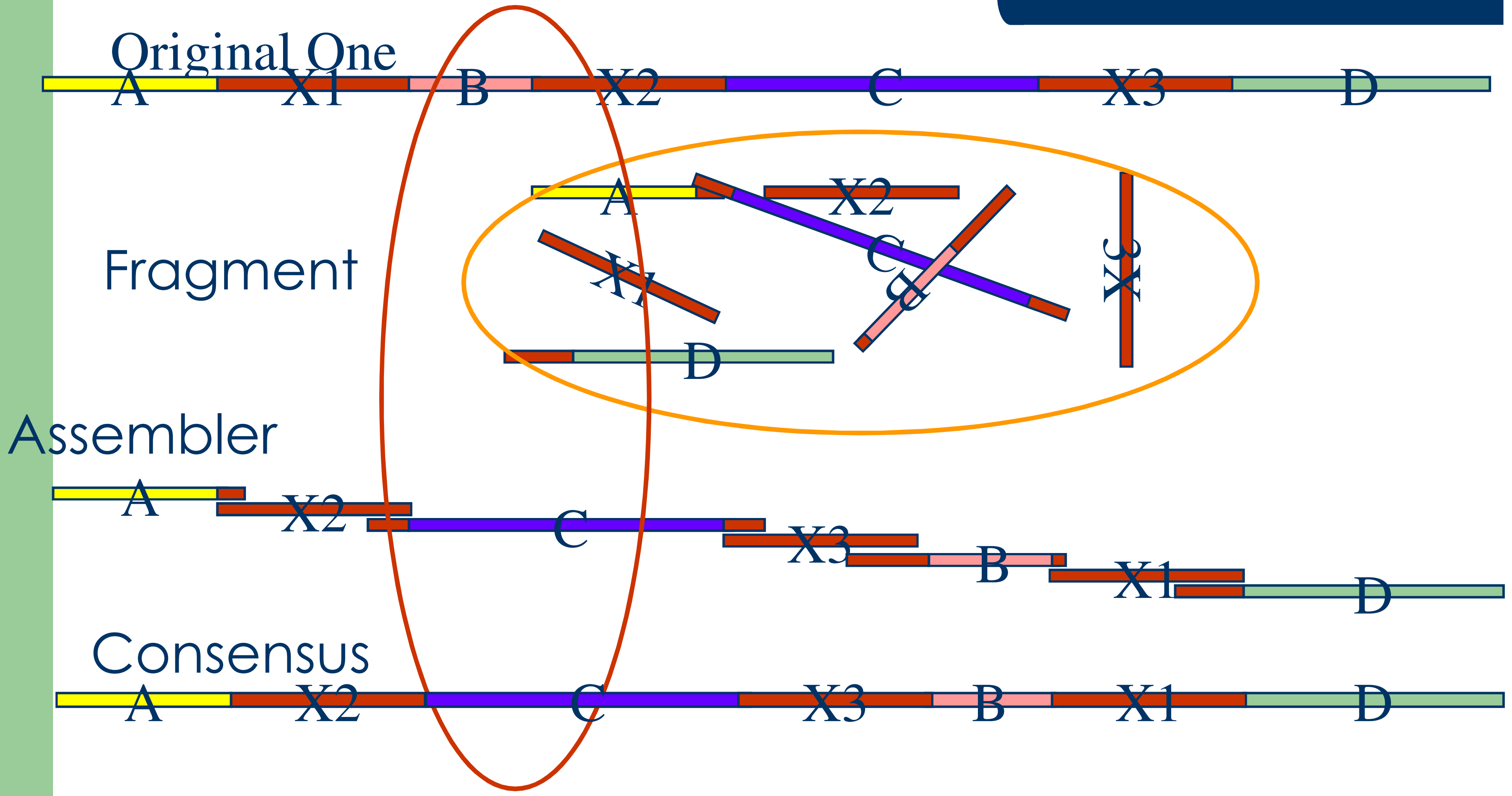


CACGT	→	CACGT	CACGT
ACGT	→	ACGT	-ACGT
ACTACG	←	CGTAGT	--CGTAGT
GTACT	←	AGTAC	-----AGTAC

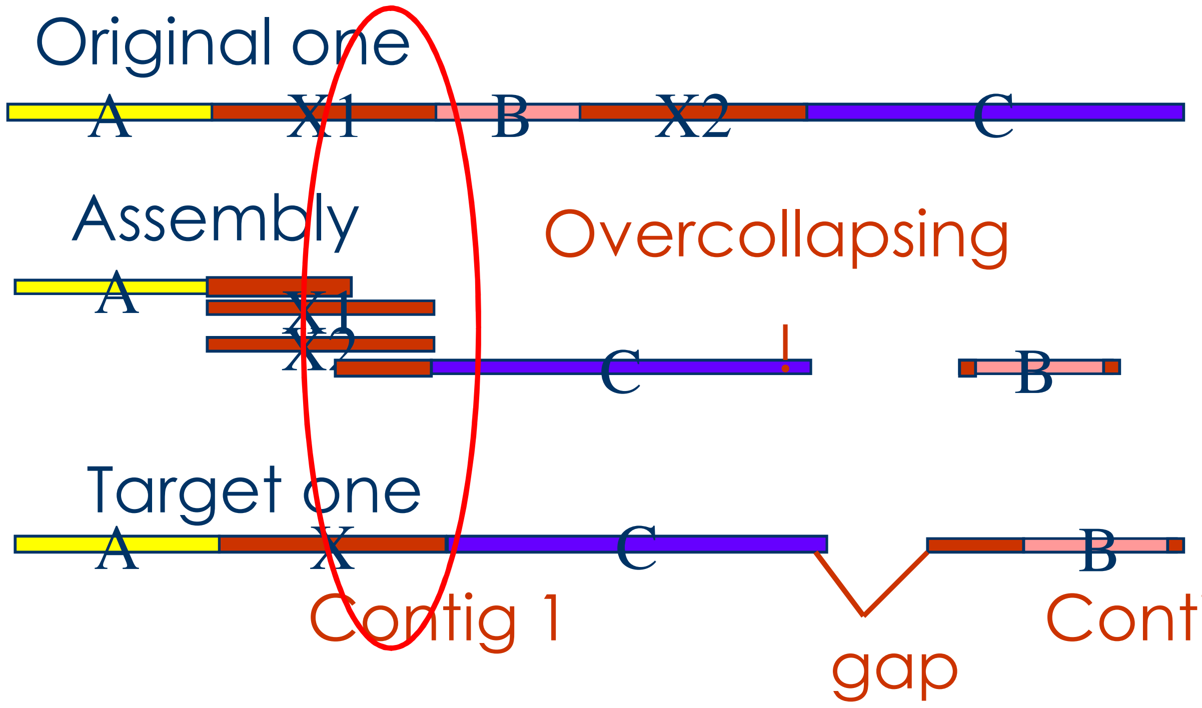
CACGTAGTACTGA

Problems with the simple model - Repeat

Rearrangment



Problems with the simple model - Repeat

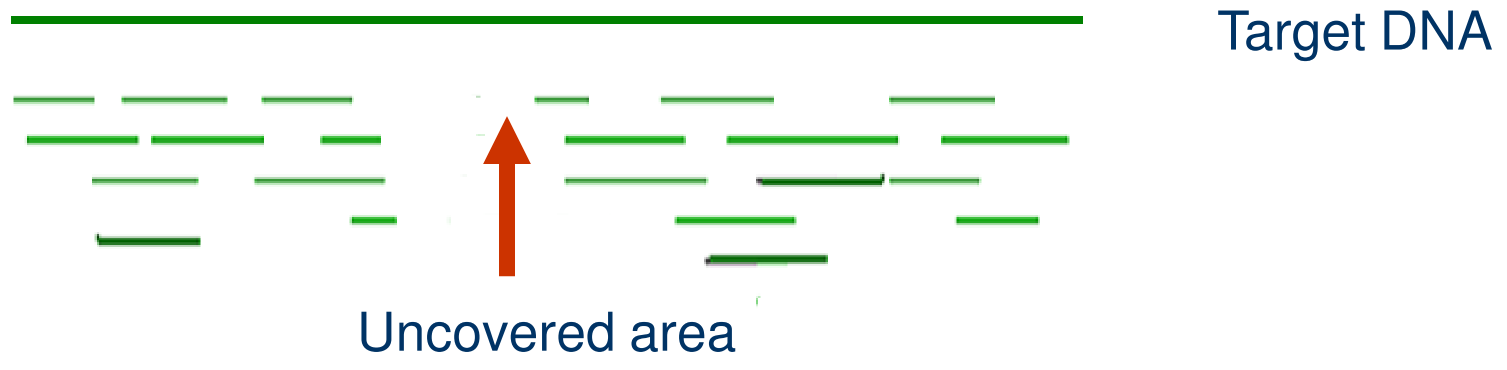


Shortest string is not always the best!

Problems with the simple model -Lack of coverage

Lack of coverage

Not all regions of the sequence will be covered equally



Solution

Do more sampling to increase the coverage level

Using scaffolder technology

1. ~~Break~~
2. ~~Sequence~~
3. ~~Assemble~~
4. ~~Conclusion~~

4. Conclusion

- The whole genome sequencing process

Break-> Sequence -> Assemble

- A Simple Model

Using overlap graph to construct the shortest common string

However, it can’t correctly model the assembly problem