15-853:Algorithms in the Real World

Computational Biology III
- Multiple Sequence Alignment
- Sequencing the Genome

Multiple Alignment

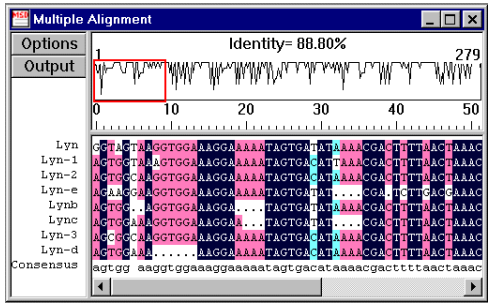
```
A C T _ G T A
A C A C G T T
A G T G _ T A
C C _ G C T A
```

Goal: match the "maximum" number of aligned pairs of symbols.

Applications:
- Assembling multiple noisy reads of fragments of sequences
- Finding a canonical among members of a family and studying how the members differ

The problem is NP-hard

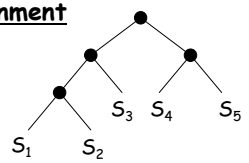
Example Output



Output from typical multiple alignment software DNAMAN (using ClustalW)

Scoring Multiple Alignments

1. **Distance from consensus S_c:** $D = \sum_{S_i \in S} D(S_i, S_c)$
2. **Pairwise distances:** $D = \sum_{S_i \in S} \sum_{S_j \in S / S_i} D(S_i, S_j)$
3. **Evolutionary Tree Alignment**



$$D = D(S_1, S_2) + D(S_4, S_5) + D(S_{12}, S_3) + D(S_{123}, S_{45})$$

Approaches

Dynamic programming: optimal, but takes time that is exponential in *p*

Center Star Method: approximation

Clustering Methods: also called iterative pairwise alignment. Typically an approximation. Many variants, many software packages

Using Dynamic Programming

For *p* sequences of length *n* we can fill in a *p*-dimensional array in n^p time and space.

For example for $p = 3$:

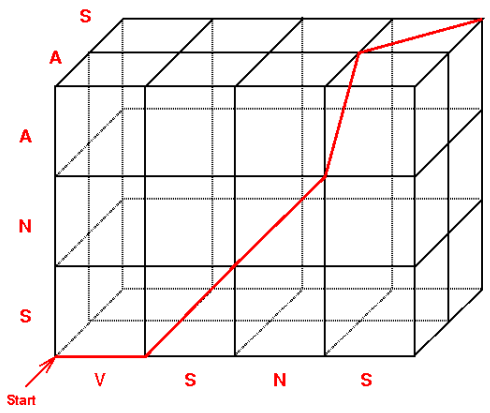
$$D_{ijk} = \min \begin{cases} D_{i-1,j-1,k-1} + d(a_i, b_j, c_k) \\ D_{i-1,j-1,k} + d(a_i, b_j, \_) \\ D_{i-1,j,k} + d(a_i, \_, \_) \\ \ldots \end{cases}$$ 7 cases

where $d(a,b,c) = d(a,b) + d(b,c) + d(a,c)$

assuming the pairwise distance metric.

Takes time exponential in *p*. Perhaps OK for $p = 3$

Example



V S N _ S
_ S N A _
_ _ _ A S

Optimization

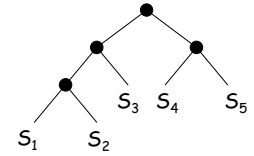
As in the case of pairwise alignment we can view the array as a graph and find shortest paths.

Used in a program called MSA.

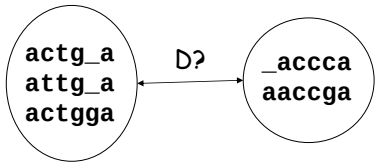
Can align 6 strings consisting of 200 bp each in a "practical" amount of time.

Using Clustering

1. Compute $D(S_i,S_j)$ for all pairs
2. Bottom up cluster
   I. All sequences start as their own cluster
   II. Repeat
      a) find the two "closest" clusters and join them into one
      b) Find best alignment of the two clusters being joined



Distances between Clusters



Could use difference between consensus.

A popular technique is called the "Unweighted Pair-Group Method using arithmetic Averages" (UPGMA).
It takes the average of all distances among the two clusters.

Implemented in Clustal and Pileup

Summary of Matching

Types of matching:
- **Global**: align two sequences A and B
- **Local**: align A with any part of B
- **Multiple**: align k sequences (NP-complete)

Cost models
- **LCS and MED**
- **Scoring matrices:** Blosum, PAM
- **Gap cost:** affine, general

Methods
- **Dynamic programming:** many optimizations
- **"Fingerprinting"**: hashing of small seqs. (approx.)
- **Clustering:** for multiple alignment (approx.)

Sequencing the Genome

One of the great achievements of the 21st century.

Tools of the Trade

Cutting:

Arber, Nathans, and Smith, **Nobel Prize in Medicine** (1978) for "the discovery of restriction enzymes and their application to problems of molecular genetics".

Copying:

Mullis, **Nobel Prize in Chemistry** (1993) for "his invention of the polymerase chain reaction (PCR) method"

Reading: (sequencing)

Gilbert and Sanger, **Nobel Prize in Chemistry** (1980) for "contributions concerning the determination of base sequences in nucleic acids"

Cutting

Cutting:

- Restriction Enzines:
  Cut at particular sites, e.g. ACTTCTAGAT
- Chemical, physical or radiation cuts
  Cut at random locations

Copying

Copying:

Cloning a strand of DNA

- Cosmids: clones sequences up to 40K bps
- BAC, PAC: up to about 200K bps
- YAC (yeast artificial chromosones): up to 1 M

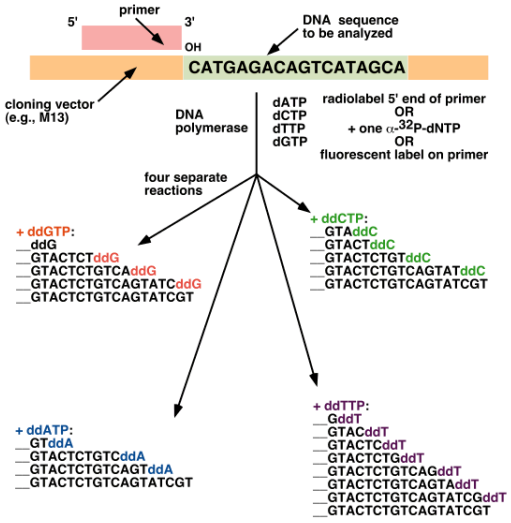
Copying between two specific sites

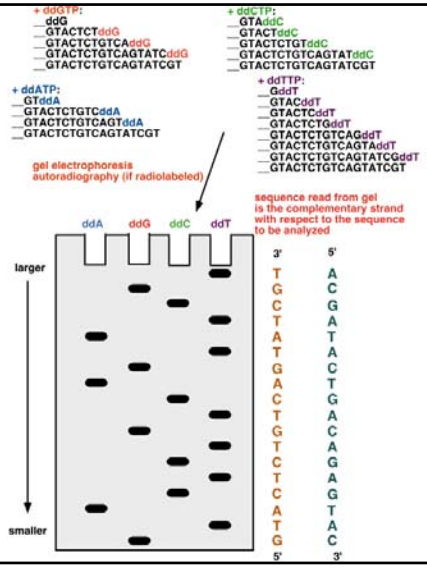
- PCR (polymerase chain reaction): 500 bps

Reading using lengths

Can use special base-pairs that stop growth: DDC, DDA, DDT, DDG. (terminator bases)

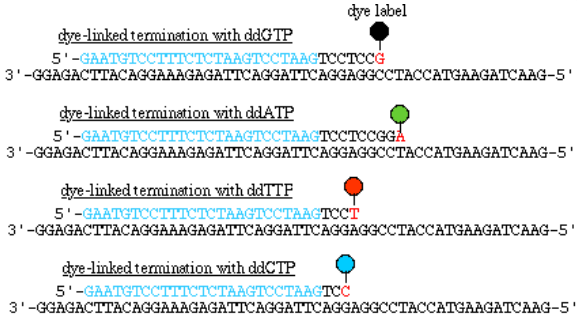
Will generate all prefixes that end in A, T, C or G.

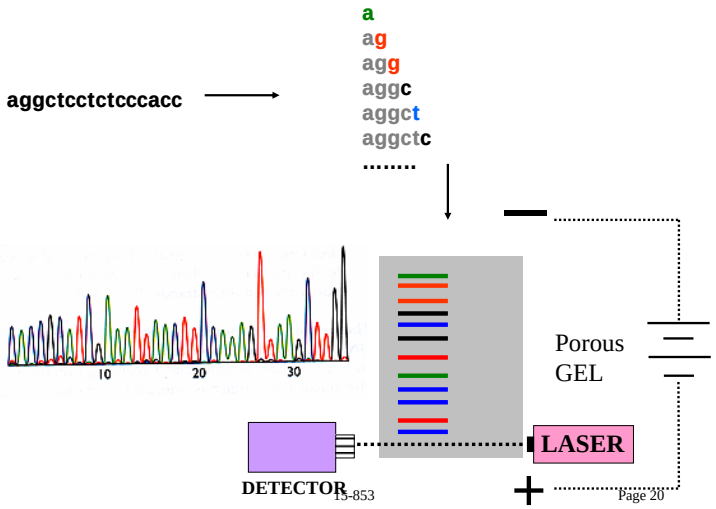




Improvements

Use fluorescent dies on the base pairs and laser to excite the die as it passes a certain point on the gel.

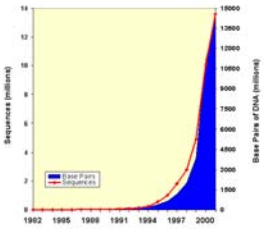




ABI 3700 sequencer

History of Sequencing

1971 Nobel prize for restriction enzymes
1973 First recombinant DNA
1980 Nobel prize for DNA sequencing
1988 Congress establishes Genbank
1995 First genomic sequence
1998 First multicellular organism
2000 Fly genome
2000 First plant genome
2001 Human genome
2003 Mouse genome



22 million sequences

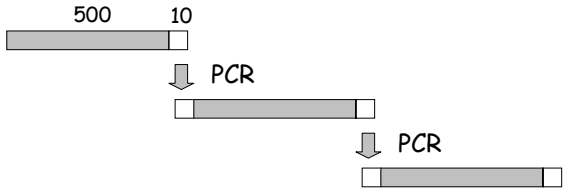
28 billion base pairs

Sequencing the Whole Genome

Problem: we only know how to sequence about 500 bps at a time in the lab.

1. Linear sequencing
2. The shotgun method
3. Hierarchical shotgun method
4. Whole genome and double-barreled shotgun methods

Linear Sequencing



Each step takes too long. Requires "wet" runs.

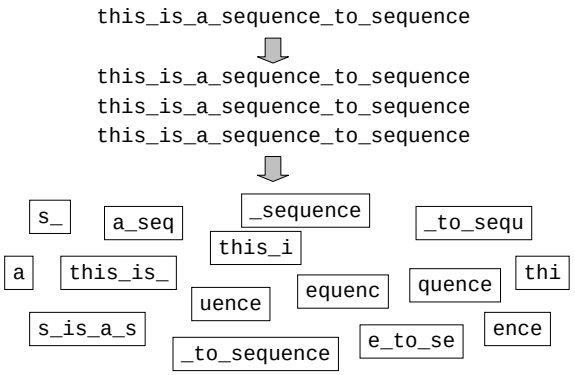
e.g. if each step took 4 hours, sequencing the human genome would take
$4 \pounds 3 \pounds 10^9/500$ hours = 3000 years

Also no interesting Computer Science ☺

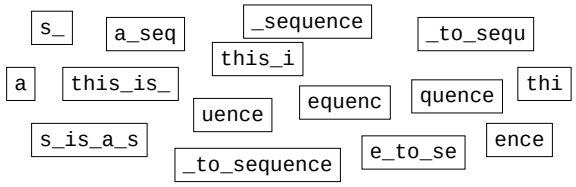
The Shotgun Method

1. Make multiple copies of the sequence.
2. Randomly break sequences into parts (e.g. using radiation or chemicals).
3. Throw away parts that are too small or too large.
4. Read about 500bp from the end of each part
5. Try to put the information together to reconstruct the original sequence

Example

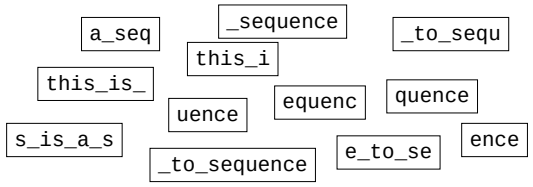


Example



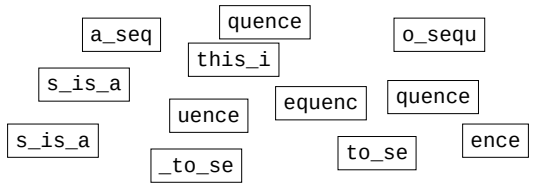
Remove strands that are too short (or too long)

Example



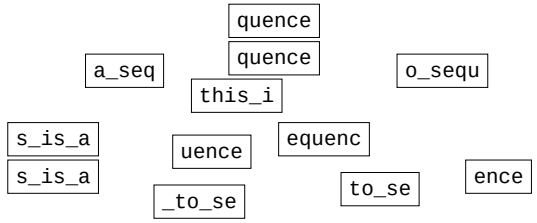
Sequence k characters from each (e.g. 6), from either end.

Example

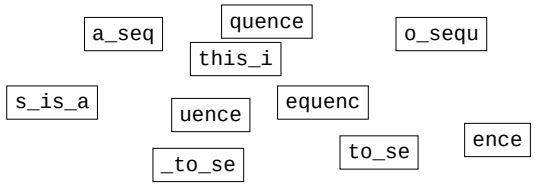


Find overlaps

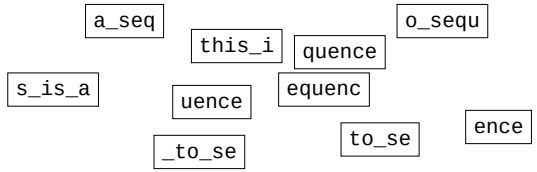
Example



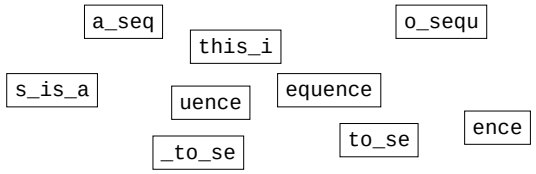
Example



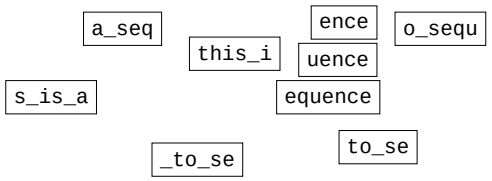
Example



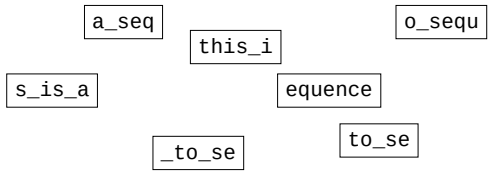
Example



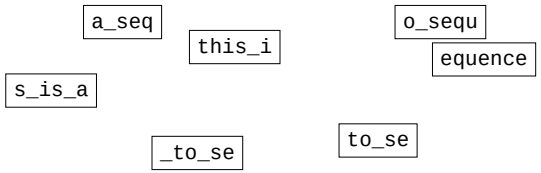
Example



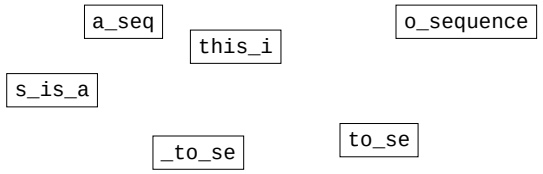
Example



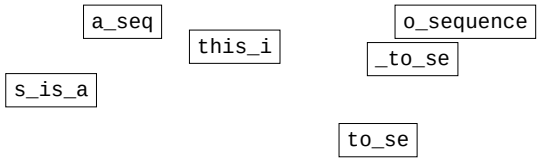
Example



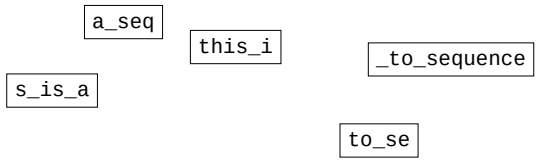
Example



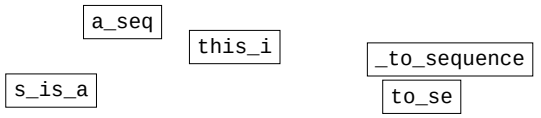
Example



Example



Example



Example

a_seq

this_i

_to_sequence

s_is_a

Example

a_seq

_to_sequence

s_is_a

this_i

Example

a_seq

_to_sequence

this_is_a

Having a single character overlap might not be enough to assume they overlap.

Example

a_seq this_is_a _to_sequence

Example

a_seq this_is_a _to_sequence

We are left with **gaps**, and unsure matches.

Each covered region (e.g. `this_is_a`) is called a **contig**

Is there a systematic way to find or even define a "best solution"?

The SSP: an attempt

The shortest superstring problem: given a set of strings $s_1, s_2, \ldots, s_n$ find the shortest string S that contains all s_i.

NP-Hard, but can be reduced to TSP and solved approximately (nearly optimally in practice).

Even if easy to solve, are we done?

Our example gives:

`this_is_a_seq_to_sequence`

but this is the best we can do given the data.

This problem is caused by repeats.

Other problems?

Problems

In practice the data is noisy.

- Reads have up to a 1% error rate
- Samples could have contaminants
- Fragments can sometimes join up

The reads could be in either direction (front-to-back or back-to-front). Cannot distinguish.

Assembly in Practice

gatcgat_ga

attgactactatg

Score all suffix-prefix pairs

- This can use a variant of the global alignment prob. It is the most expensive step (n^2 scores).

Repeat:

- Select best score and check for consistency
- If score is too low, quit
- If there is a good overlap, merge the two.

Determine consensus:

- We know the ordering among strands, but since matches are approximate, we need to select bps. Can use, e.g., multiple alignment over windows.

Some Programs for Assembly

Phrap
SEQAID
CAP
TIGR
Celera assembler
ARACHNE

After using one of these programs to generate a set of "contigs" with some gaps, one can use the linear method to fill in the gaps (assuming they are small).

atgattagccagtacgtt t tcagcatcccagtacgttatgca c ttagccaga

Suquencing the Whole Genome

Problem: we only know how to sequence about 500 bps at a time in the lab.

1. Linear sequencing
2. The shotgun method
3. Hierarchical shotgun method
4. Whole genome and double-barreled shotgun methods

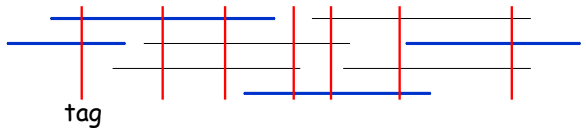
Shotgun on the Whole Genome?

Problems:

- Computationally very expensive
- 50% of genome consist of repeats. Causes major problems.
- Hard to partition work among multiple labs.

Hierarchical Shotgun

1. Generate clone Libraries (100K – 1M per clone)
2. Order the clones by finding "tags" that overlap multiple clones. Use these for ordering.
3. Identify a set of clones that cover the whole length (minimum tiling path)
4. Use shotgun technique on each identified clone
5. Put the results together.



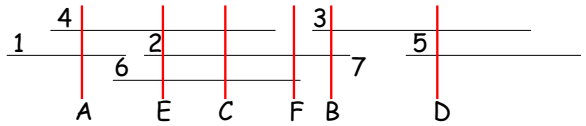
1. Clone Libraries

A "BAC" library will contain sequences of about 200K bps each. These can be cloned using "BAC Vectors" (Bacterial Artificial Chromosome)

A "YAC" library will contain sequences of about 1M bps each. These can be cloned using "YAC Vectors" (Yeast Artificial Chromosome)

These are typically stored at a common site and can be ordered. Many can be purchased from companies.

2. Ordering Clones



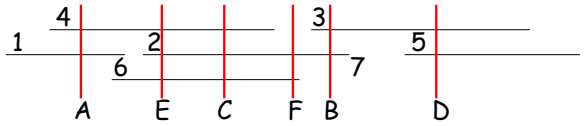
We have the clones, but we don't know their order or how they overlap.

Pick random small sequences that only appear once in one location covered by the library.

These are called STS (Sequence Tagged Sites)

Figure out which clones contain which STSs using PCR (use tag site to start copy…will only copy of the sequence contains the site).

2. Ordering Clones (cont.)



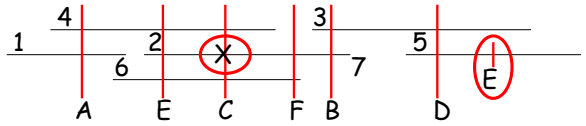
	A	B	C	D	E	F
1	1	0	0	0	0	0
2	0	1	1	0	1	1
3	0	1	0	1	0	0
4	1	0	1	0	1	0
5	0	0	0	1	0	0
6	0	0	1	0	1	1

Goal: Reorder the columns so that all the 1s in each row are contiguous.

Can be done in O(n) time, where n is the number of entries in the array.

But!!!, what about **errrors**?

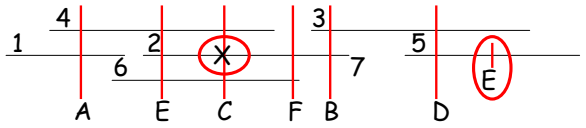
2. Ordering Clones (cont.)



	A	B	C	D	E	F
1	1	0	0	0	0	0
2	0	1	0	0	1	1
3	0	1	0	1	0	0
4	1	0	1	0	1	0
5	0	0	0	1	1	0
6	0	0	1	0	1	1

	A	E	C	F	B	D
1	1	0	0	0	0	0
2	0	1	0	1	1	0
3	0	0	0	0	1	1
4	1	1	1	0	0	0
5	0	1	0	0	0	1
6	0	1	1	1	0	0

2. Ordering Clones (cont.)



Find ordering that minimizes the number of zero-one and one-zero transitions (i.e. errors).

This is NP-hard, but can be posed as a Traveling Salesman Problem (TSP).

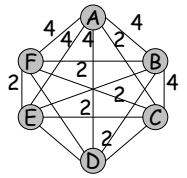
Any ideas?

2. Ordering Clones (cont.)

Create graph with one vertex per STS.

Edge weights = hamming distance (number of bits that differ).

	A	B	C	D	E	F
1	1	0	0	0	0	0
2	0	1	0	0	1	1
3	0	1	0	1	0	0
4	1	0	1	0	1	0
5	0	0	0	1	1	0
6	0	0	1	0	1	1

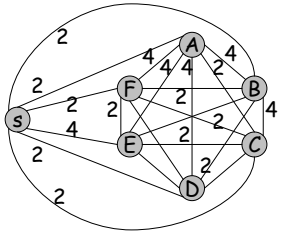


2. Ordering Clones (cont.)

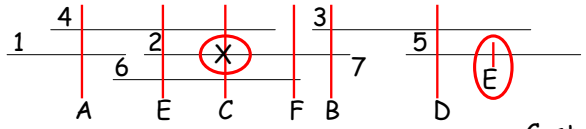
Add in source (s) node with weights equal to number of 1s in each row.

Solve TSP. Answer gives min number of transitions.

	A	B	C	D	E	F
1	1	0	0	0	0	0
2	0	1	0	0	1	1
3	0	1	0	1	0	0
4	1	0	1	0	1	0
5	0	0	0	1	1	0
6	0	0	1	0	1	1

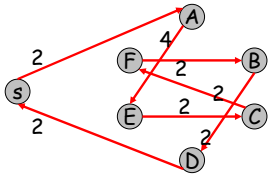


2. Ordering Clones (cont.)

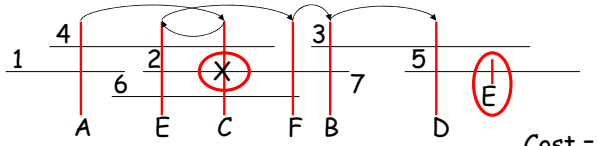


Cost = 16

	A	B	C	D	E	F
1	1	0	0	0	0	0
2	0	1	0	0	1	1
3	0	1	0	1	0	0
4	1	0	1	0	1	0
5	0	0	0	1	1	0
6	0	0	1	0	1	1

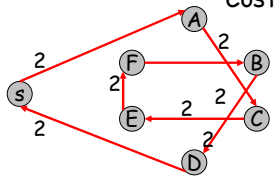


2. Ordering Clones (cont.)



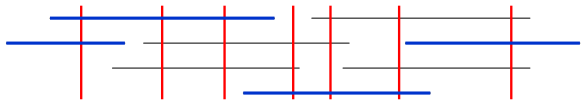
	A	B	C	D	E	F
1	1	0	0	0	0	0
2	0	1	0	0	1	1
3	0	1	0	1	0	0
4	1	0	1	0	1	0
5	0	0	0	1	1	0
6	0	0	1	0	1	1

Cost = 14



The "wrong" answer has smaller cost

3. Find "Minimum Tiling Path"



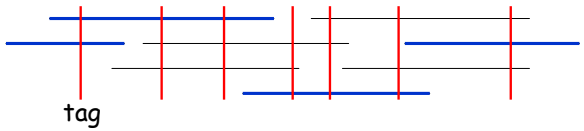
Minimum Tiling Path: Find a set of clones that cover the whole length and for which the total number of bps is minimized.

Can be posed as a shortest path problem.

Any ideas?

Hierarchical Shotgun (revisited)

1. Generate clone Libraries (100K – 1M per clone)
2. Order the clones by finding "tags" that overlap multiple clones. Use these for ordering.
3. Identify a set of clones that cover the whole length (minimum tiling path)
4. Use shotgun technique on each identified clone
5. Put the results together.

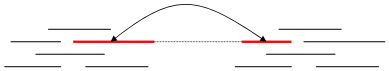


Celera's Method

Whole genome shotgun:

Use shotgun method on whole genome.

Use **double-barreled** approach: some sequences of known length (e.g. 2-5K) are sequenced at both ends. These can be used to bridge across repeats.



In practice they used some mapping (hierarchical) data from the NIST effort, which was freely available. This was needed to deal with long repeats.