Computational Biology: Algorithms

Lecture 4

9 October 2008

1

Overview: First Half of Lecture

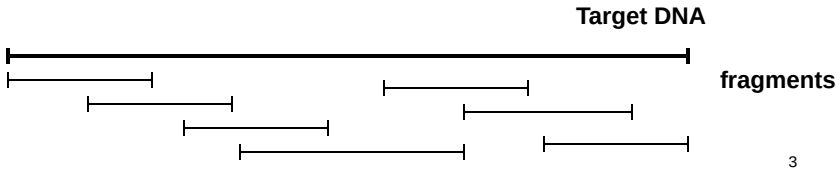
- Clone vectors
- Clone libraries and coverage
- Fingerprints and map building
- Mapping with unique probes

2

Cloning

Can't deal with DNA at chromosomal lengths (for many organisms)

- Need to process it in pieces
- But need to put it together again
- Hence, capture it as collection (library) of overlapping fragments



3

Need to replicate fragments

Insert into some *vector* that a host organism will reproduce

Vector and host determine clone length

- Plasmid 0.1-10 kbp
- λ vector 2-20 kbp
- Cosmid 20-45 kbp
- Bacterial artificial chromosome (BAC) 75-300 kbp
- Yeast artificial chromosome (YAC) 100-1000 kbp

4

Clone vectors differ in several ways

- Length of fragment accommodated
- How readily the host takes up the vector
- Copy number per host
- Stability: how faithfully is DNA fragment reproduced
  - Chimerism: fragment splicing or crossover
  - Deletions: of part of fragment
  - Bacterial transposons: “jumping genes”

5

Library coverage

Have a library of clones – how much of genome is covered?

Rough estimate

- Assume every cloned fragment has length L
- Assume N clones randomly distributed over genome of length G
- [There are boundary considerations for linear genomes] Assumes a circular genome

6

Probability of a base not covered

For particular base b and clone C, probability of base lying in the clone

P(b ∈ C) = L/G $1 - L/G$ $\mathbb{P}(b \notin C)$

Probability that b isn't in any of N clones

$(1 - L/G)^N = (1 - L/G)^{G(N/G)} =$

$((1 - L/G)^G)^{N/G} =$ $(e^{-L})^{N/G} = e^{-LN/G}$ $\left(1 - \frac{a}{d}\right)^d \approx e^{-a}$ gets closer as d grows larger

Number of bases not covered: $G \cdot e^{-LN/G}$

7

Example of coverage

Human genome G = 3 gpb

BAC clones L = 200 kbp

Library size N = 25K

LN/G = (25K)(200K)/3 G = $\frac{25 \cdot 200}{30 \phi \phi}$ $\frac{50}{30} \approx 1.67$

3 gbp•$e^{-1.67}$ = $3\text{ gbp} \cdot .188 = 564\text{ Mbp}$

8

Physical Mapping

Trying to figure out the order and direction of the clones on the genome

- Generally, without fully sequencing them

9

Use “Fingerprints” of Clones

Fingerprint: some property that can be determined for each clone

- Overlapping clones $\rightarrow$ similar fingerprints
- (Hopefully, vice versa)
  - Restriction maps: E. coli, 1987
  - Restriction fragment sizes: Yeast, 1986
  - Hybridization data: Human, 1992

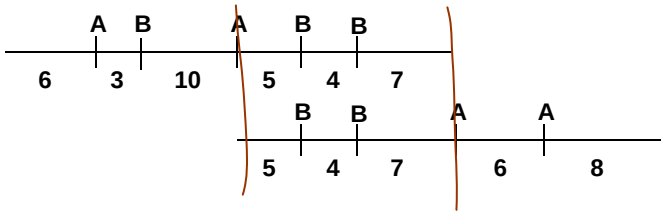
10

Restriction Map

Derive restriction maps of all clones

Infer overlap if end of one map matches beginning of another (consider reversals)

Time consuming, lengths not precise



11

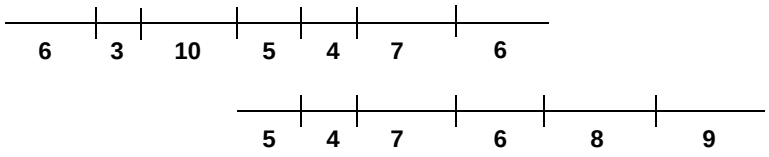
Restriction Fragment Sizes

Generally based on a threshold – certain number or percentage of fragment lengths in common

Faster, more false positives

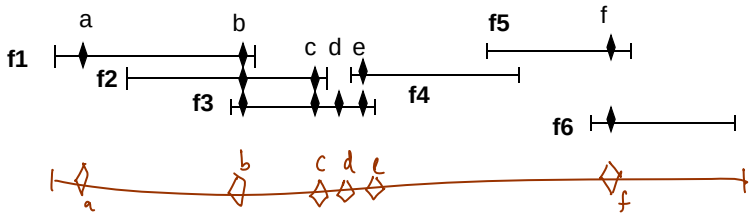
3, 4, 5, 6, 7, 10

4, 5, 6, 7, 8, 9



12

Hybridization Probes



Which probes bind to which fragments

- Unique probes: Sequence Tagged Sites (STS)
- Non-unique probes: short random sequences

13

Map Assembly

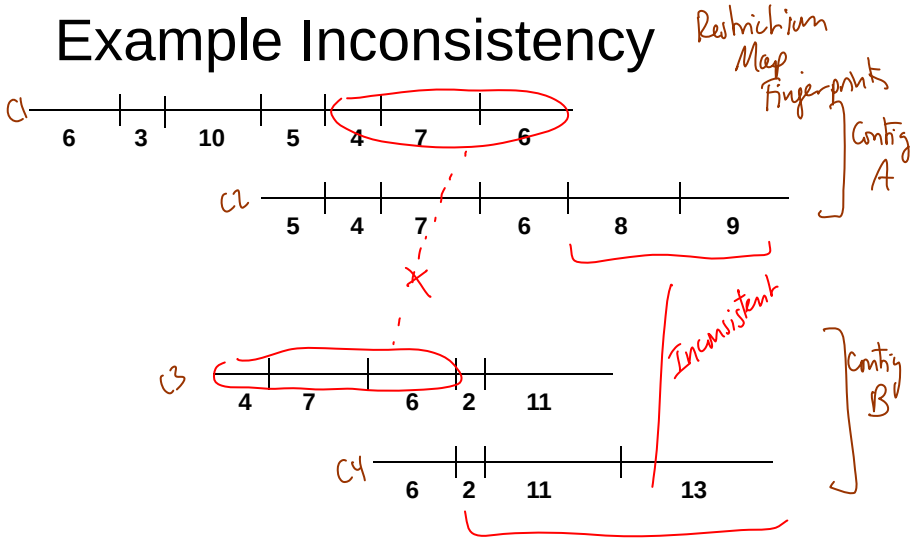
Combine clones into *contigs*: groups of consistently overlapping clones

Often based on greedy algorithm

- Compute all pairwise overlaps with fingerprinting "strength" of overlap
- Initialize each contig to a single clone
- Repeat: "Greedy"
  - Combine contigs with highest remaining overlap
  - [Make sure consistent with other clones in contig]

14

Example Inconsistency



15

Mapping with Unique Probes

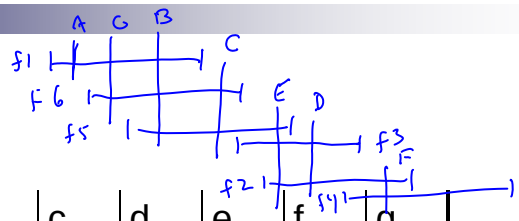
Given a matrix of which probes are found on which clones:

Can you find an ordering such that in each column, the 1's are consecutive?

	a	b	c	d	e	f	g
f1	1	1					1
f2				1	1	1	
f3				1	1		
f4						1	
f5		1	1		1		
f6		1	1				1

16

A Solution



	a	b	c	d	e	f	g
f1	1	1					1
f6		1	1				1
f5		1	1		1		
f3				1	1		
f2				1	1	1	
f4						1	

17

Another Approach

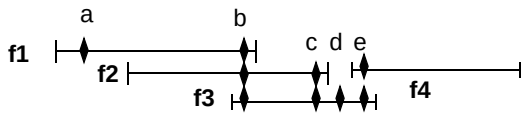
- Can find consecutive ones in linear time if such an ordering exists
  - Lueker and Booth,1976
  - Not very tolerant of errors
    - Chimerism
    - False negatives
- Are more fault-tolerant algorithms
  - Alizadeh, Karp, Weisser, Zweig, “Physical Mapping of Chromosomes using Unique Probes”
  - In ACM Portal

18

Ordering of Probes

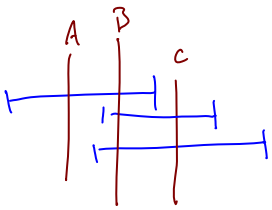
Can tell something about probe order from clone sets

Ca = {f1}
Cb = {f1, f2, f3}
Cc = {f2, f3}
Cd = {f3}
Ce = {f3, f4}



19

Ordering Information



# of elements (cardinality)

If b is between a and c, then

$|Ca \cap Cb| \geq |Ca \cap Cc|$

$|\{f1\} \cap \{f1,f2,f3\}| \geq |\{f1\} \cap \{f2,f3\}|$

$|\{f1\}| \geq |\emptyset|$

$1 \geq 0$

Isn't always strict. Consider c, d and e

Is d between c and e

$|C_c \cap C_d| \geq |C_c \cap C_e|$

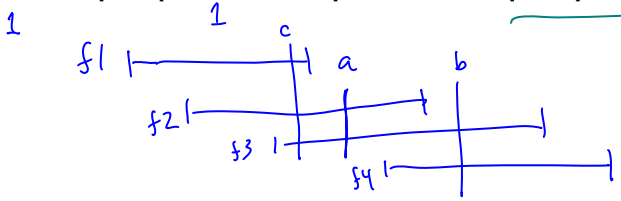
$|\{f3\}| \geq |\{f3\}|$

$=$

20

"Closeness" ordering

"Probe a is closer than c to b" if
$|Ca \cap Cb| > |Cc \cap Cb|$
or
$|Ca \cap Cb| = |Cc \cap Cb| \neq \emptyset$ and $|Ca| < |Cc|$



21

Neighbors of Probe p

N(p) = set of probes that share some clone with p (excludes p itself)

N(a) = {b, g}	Ca = {f1}
N(b) = {a, c, e, g}	Cb = {f1, f5, f6}
N(c) = {b, e, g}	Cc = {f5, f6}
N(d) = {e, f}	Cd = {f2, f3}
N(e) = {b, c, d, f}	Ce = {f2, f3, f5}
N(f) = {d, e}	Cf = {f2, f4}
N(g) = {a, b, c}	Cg = {f1, f6}

22

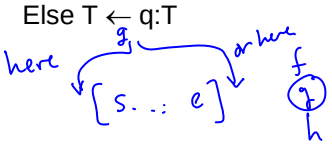
Alizadeh, et al. Algorithm

Build up a probe sequence from a single probe

1. Pick a starting probe p: T = [p]
2. Let T = [s, …, e]. Repeat:
   a. If all of N(e) is in T, T ← reverse(T)
   b. Else, pick a probe q in N(e) not already in T that is closest to e.

      If e is closer than s to q, T ← T:q

      Else T ← q:T



23

T = [c]

$C_c \cap C_b = \{f5, f6\}$ 2

$C_c \cap C_e = \{f5\}$ 1

$C_c \cap C_g = \{f6\}$ 1

N(a) = {b, g}	Ca = {f1}
N(b) = {a, c, e, g}	Cb = {f1, f5, f6}
N(c) = {b, e, g}	Cc = {f5, f6}
N(d) = {e, f}	Cd = {f2, f3}
N(e) = {b, c, d, f}	Ce = {f2, f3, f5}
N(f) = {d, e}	Cf = {f2, f4}
N(g) = {a, b, c}	Cg = {f1, f6}

T = [c b]

{a, e, g}

$C_b \cap C_a = \{f1\}$ 1

$C_b \cap C_e = \{f5\}$ 1

$C_b \cap C_g = \{f1, f6\}$ 2

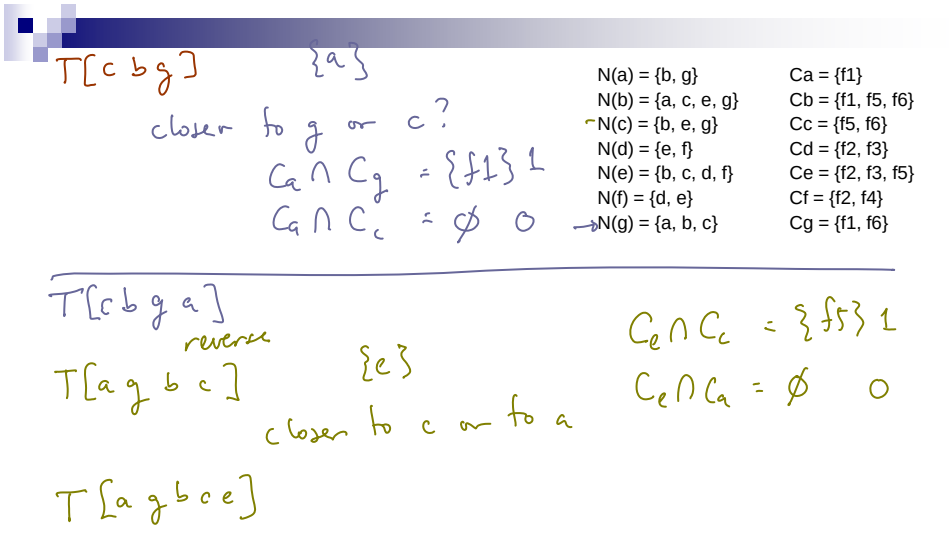
g go at beginning or end?

$C_g \cap C_b = \{f1, f6\}$

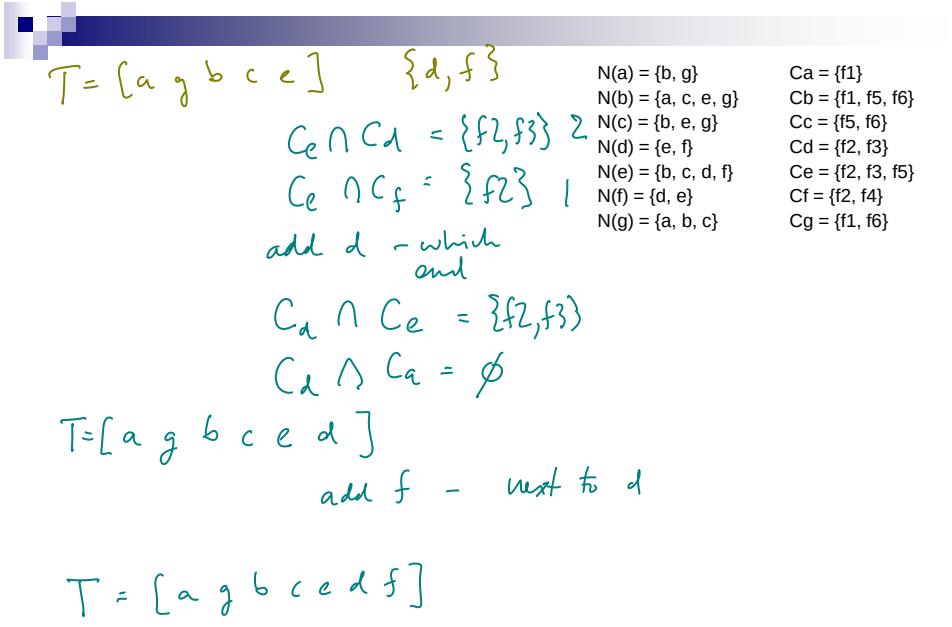
$C_g \cap C_c = \{f6\}$

T = [c b g]

24


T[c b g]
{a}
closer to g or c?
Ca ∩ Cg = {f1} 1
Ca ∩ Cc = ∅ 0
N(a) = {b, g}
N(b) = {a, c, e, g}
N(c) = {b, e, g}
N(d) = {e, f}
N(e) = {b, c, d, f}
N(f) = {d, e}
N(g) = {a, b, c}
Ca = {f1}
Cb = {f1, f5, f6}
Cc = {f5, f6}
Cd = {f2, f3}
Ce = {f2, f3, f5}
Cf = {f2, f4}
Cg = {f1, f6}
T[c b g a]
reverse
T[a g b c]
{e}
closer to c or to a
Ce ∩ Cc = {f5} 1
Ce ∩ Ca = ∅ 0
T[a g b c e]

25

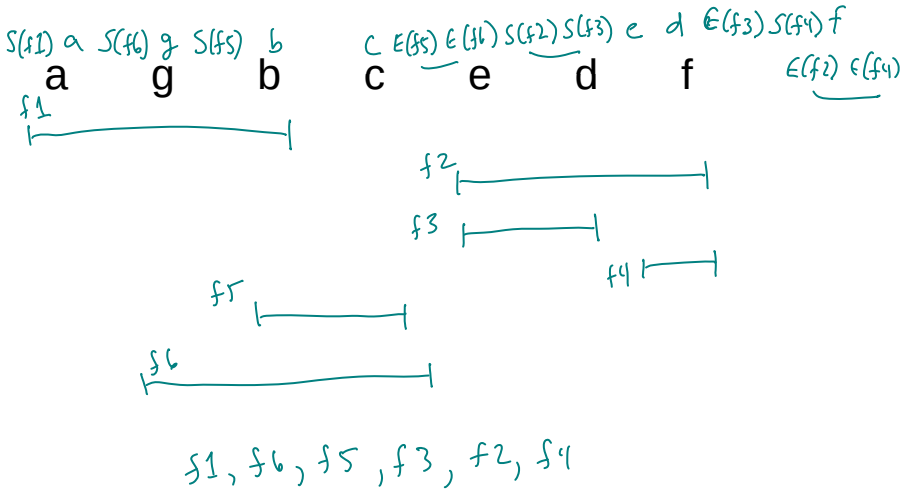

T=[a g b c e]
{d, f}
Ce ∩ Cd = {f2, f3} 2
Ce ∩ Cf = {f2} 1
add d - which and
Cd ∩ Ce = {f2, f3}
Cd ∩ Ca = ∅
T=[a g b c e d]
add f - next to d
T = [a g b c e d f]
N(a) = {b, g}
N(b) = {a, c, e, g}
N(c) = {b, e, g}
N(d) = {e, f}
N(e) = {b, c, d, f}
N(f) = {d, e}
N(g) = {a, b, c}
Ca = {f1}
Cb = {f1, f5, f6}
Cc = {f5, f6}
Cd = {f2, f3}
Ce = {f2, f3, f5}
Cf = {f2, f4}
Cg = {f1, f6}

26

N(a) = {b, g}	Ca = {f1}
N(b) = {a, c, e, g}	Cb = {f1, f5, f6}
N(c) = {b, e, g}	Cc = {f5, f6}
N(d) = {e, f}	Cd = {f2, f3}
N(e) = {b, c, d, f}	Ce = {f2, f3, f5}
N(f) = {d, e}	Cf = {f2, f4}
N(g) = {a, b, c}	Cg = {f1, f6}

27

From Probe Order to Map



28