Algorithms in Comp. Bio

Partial Digest Problem (cont.)

Profiles + Motif Finding

Recap:

- Essay due Thursday
- New HW, due next Thursday: will cover 1st part of Chapter 4

Recall:
- use whatever, but cite source + put in your own words
- groups are fine: rule of thumb is to write up late w/out notes, + only use them if you need to

- A couple of you had questions + suggestions about future topics - please email me a reminder!

Now - Ch 4, on Exhaustive Search...

Last time: Notation

Dfn: A multiset:

ex: $\{2,2,2,3,3,4,5\}$

$\{2_3, 3_2, 4, 5\}$

Dfn: If X is a set of n points on a line segment,

$$\Delta X = \{x_i - x_j : 1 \leq i < j \leq n\}$$

Aside: How big is ΔX?

$$\binom{n}{2} = \frac{n(n-1)}{2} = \frac{n!}{2!(n-2)!}$$

Ex: Let $X = \{0, 2, 4, 7, 10\}$.

$$\Delta X = \{2, 2, 3, 3, 4, 5, 6, 7, 8, 10\}$$

$$= \{2_2, 3_2, 4, 5, 6, 7, 8, 10\}$$

Last time (cont):

Partial Digest Problem:
Given all pairwise distances between points on a line, reconstruct the positions of those points.

Input: The multiset of pairwise distances L, containing $\binom{n}{2}$ integers.

Output: A set X, of n integers, such that $\Delta X = L$

Why?

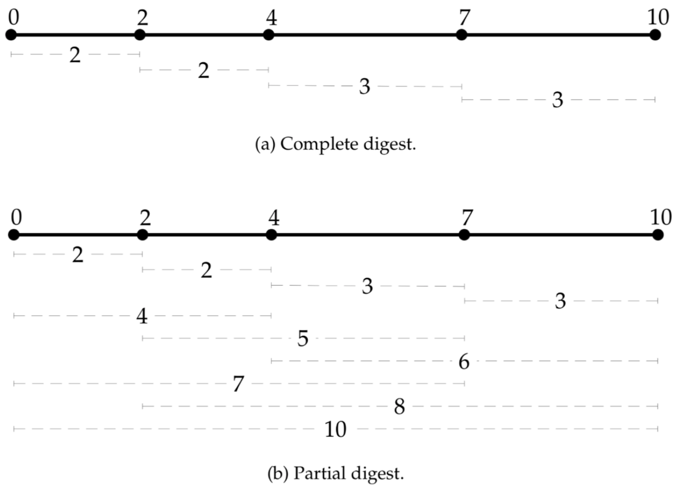


Figure 4.1 Different methods of digesting a DNA molecule. A complete digest produces only fragments between consecutive restriction sites, while a partial digest yields fragments between any two restriction sites. Each of the dots represents a restriction site.

Two (slow) solutions:

```
BRUTEFORCEPDP($L, n$)
1  $M \leftarrow$ maximum element in $L$
2  for every set of $n-2$ integers $0 < x_2 < \cdots < x_{n-1} < M$
3      $X \leftarrow \{0, x_2, \ldots, x_{n-1}, M\}$
4      Form $\Delta X$ from $X$
5      if $\Delta X = L$
6          return $X$
7  output "No Solution"
```

$\sim O(M^{n-2})$

* observe: do we really need to try every value $< M$?

Since 0 is in X, if some $x \notin L$, then $x \notin X$.

So:

```
ANOTHERBRUTEFORCEPDP($L, n$)
1  $M \leftarrow$ maximum element in $L$
2  for every set of $n-2$ integers $0 < x_2 < \cdots < x_{n-1} < M$ from $L$
3      $X \leftarrow \{0, x_2, \ldots, x_{n-1}, M\}$
4      Form $\Delta X$ from $X$
5      if $\Delta X = L$
6          return $X$
7  output "No Solution"
```

$\sim O(L^{n-2})$

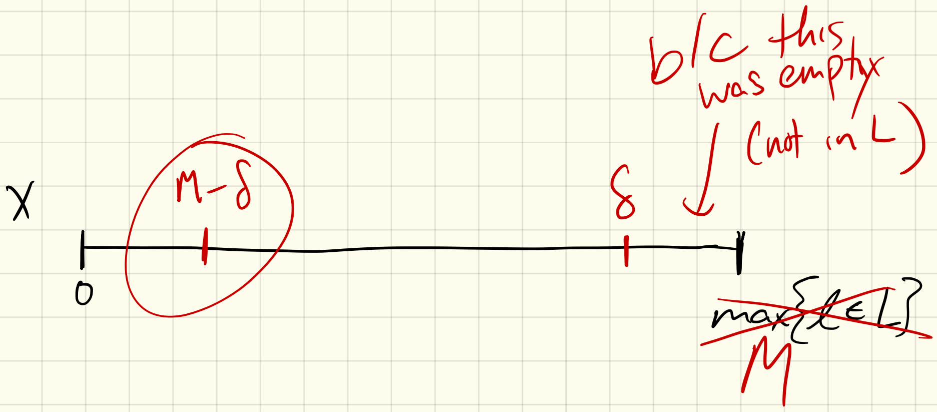
A better way [Skiena 1990]:

Include 0 & largest value in L. "M"

remove M from L

Consider the next largest value, called δ.

Where could δ be?



Then what?

remove δ, check $(0, M-\delta) \in L$

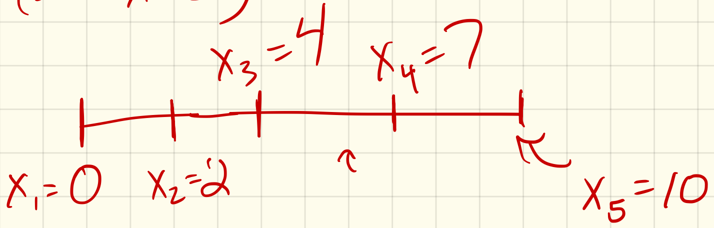
Ex: $\cancel{\Delta X} = L = \{\cancel{2}, \cancel{2}, \cancel{3}, \cancel{3}, \cancel{4}, \cancel{5}, \cancel{6}, \cancel{7}, \cancel{8}, \cancel{10}\}$

$$|L| = 10 = \binom{n}{2} = \frac{n(n-1)}{2} = \frac{n^2}{2} - \frac{n}{2}$$

solve...

$n = 5$

Build X:
(s.t. $\Delta X = L$)



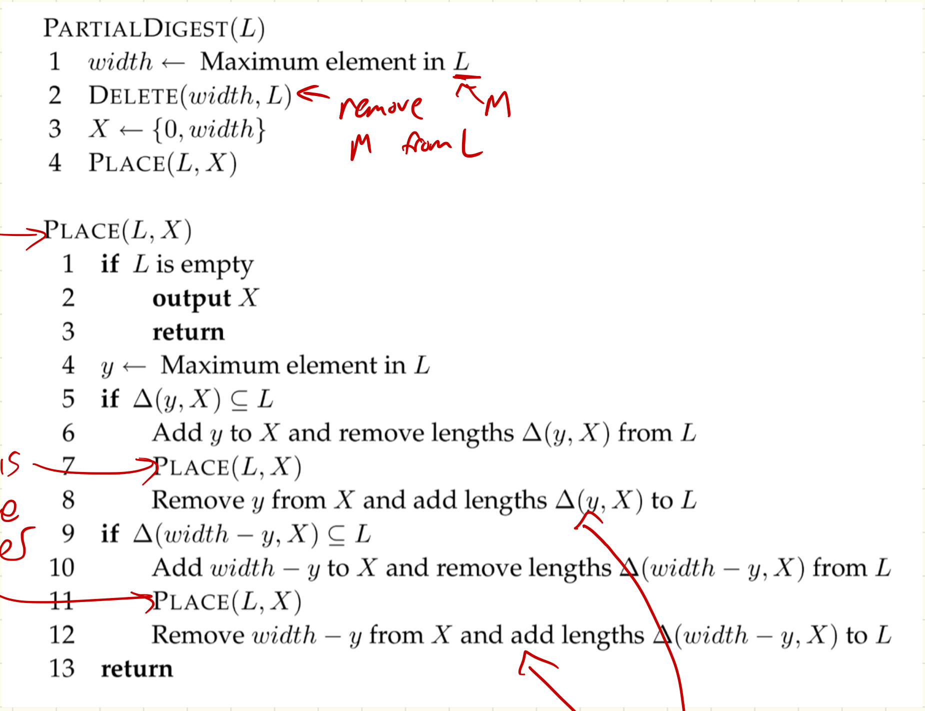
$\delta_1 = 8$, so $x_2 = 2$

now $\delta_2 = 7$

now $\delta_3 = 6$

$\vdots$

Formal algorithm:



```
PARTIALDIGEST($L$)
1  $width \leftarrow$ Maximum element in $L$        ↖ M
2  DELETE($width, L$)  ← remove M from L
3  $X \leftarrow \{0, width\}$
4  PLACE($L, X$)

→ PLACE($L, X$)
 1  if $L$ is empty
 2      output $X$
 3      return
 4  $y \leftarrow$ Maximum element in $L$
 5  if $\Delta(y, X) \subseteq L$
 6      Add $y$ to $X$ and remove lengths $\Delta(y, X)$ from $L$
 7      PLACE($L, X$)      ← |X| is one bigger
 8      Remove $y$ from $X$ and add lengths $\Delta(y, X)$ to $L$
 9  if $\Delta(width - y, X) \subseteq L$
10      Add $width - y$ to $X$ and remove lengths $\Delta(width - y, X)$ from $L$
11      PLACE($L, X$)      ← |X| is one bigger
12      Remove $width - y$ from $X$ and add lengths $\Delta(width - y, X)$ to $L$
13  return
```

Note: - Recursive!
- Undoes mistakes along the way
- Lists all sets X s.t. $\Delta X = L$!

Runtime:

Worst case:

2 recursive calls

check all D's in L, remove

$$T(n) = \underbrace{2T(n-1)}_{\text{2 recursive calls}} + \underbrace{O(n)}_{\text{check all D's in L, remove}}$$

(Towers of Hanoi, but worse [was $O(1)$])

$$O(n \cdot 2^n)$$

If only one viable alternative, then considerably faster $\leftarrow O(n^2)$
(But both can be viable!)

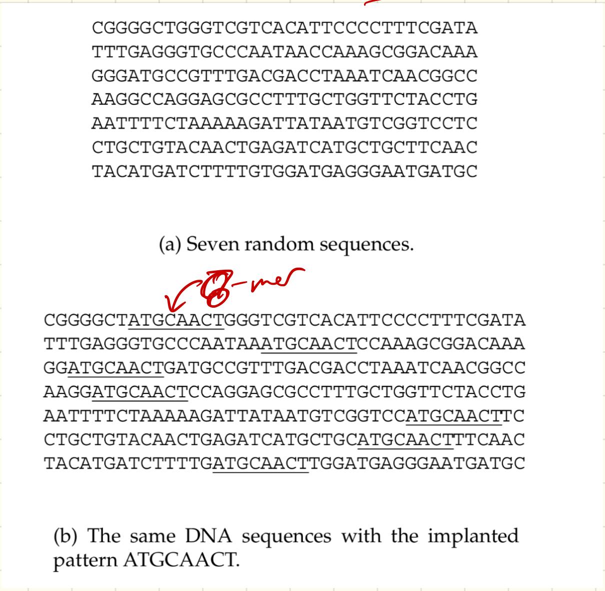
This works much faster in practice, but polynomial time algorithms didn't come until 2002.

Shifting to another problem:

DNA profiles + motifs : Idea

Frequent (or rare) substrings may correspond to regulatory motifs in DNA. ℓ-mer

Picture:



```
CGGGGCTGGGTCGTCACATTCCCCTTTCGATA
TTTGAGGGTGCCCAATAACCAAAGCGGACAAA
GGGATGCCGTTTGACGACCTAAATCAACGGCC
AAGGCCAGGAGCGCCTTTGCTGGTTCTACCTG
AATTTTCTAAAAAGATTATAATGTCGGTCCTC
CTGCTGTACAACTGAGATCATGCTGCTTCAAC
TACATGATCTTTTGTGGATGAGGGAATGATGC
```

(a) Seven random sequences.

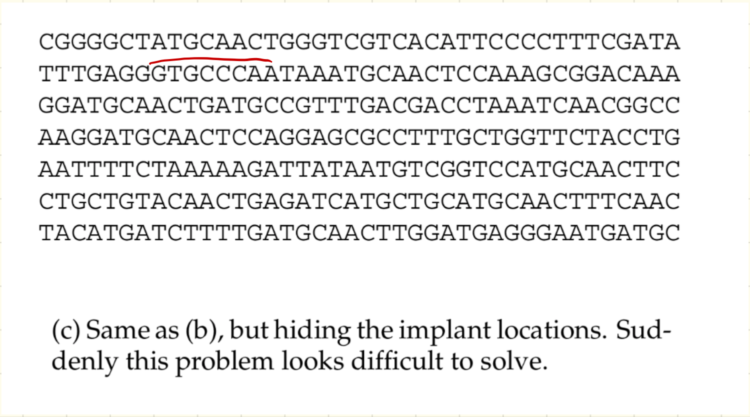
8-mer

```
CGGGGCTATGCAACTGGGTCGTCACATTCCCCTTTCGATA
TTTGAGGGTGCCCAATAAATGCAACTCCAAAGCGGACAAA
GGATGCAACTGATGCCGTTTGACGACCTAAATCAACGGCC
AAGGATGCAACTCCAGGAGCGCCTTTGCTGGTTCTACCTG
AATTTTCTAAAAAGATTATAATGTCGGTCCATGCAACTTC
CTGCTGTACAACTGAGATCATGCTGCATGCAACTTTCAAC
TACATGATCTTTTGATGCAACTTGGATGAGGGAATGATGC
```

(b) The same DNA sequences with the implanted pattern ATGCAACT.

Brute force?

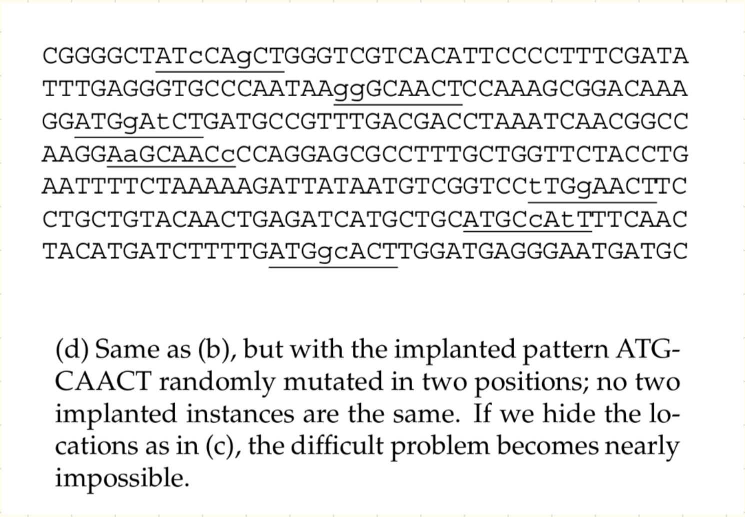
However, hard to spot (when not underlined)!



```
CGGGGCTATGCAACTGGGTCGTCACATTCCCCTTTCGATA
TTTGAGGGTGCCCAATAAATGCAACTCCAAAGCGGACAAA
GGATGCAACTGATGCCGTTTGACGACCTAAATCAACGGCC
AAGGATGCAACTCCAGGAGCGCCTTTGCTGGTTCTACCTG
AATTTTCTAAAAAGATTATAATGTCGGTCCATGCAACTTC
CTGCTGTACAACTGAGATCATGCTGCATGCAACTTTCAAC
TACATGATCTTTTGATGCAACTTGGATGAGGGAATGATGC
```

(c) Same as (b), but hiding the implant locations. Suddenly this problem looks difficult to solve.

Even worse: DNA mutates!



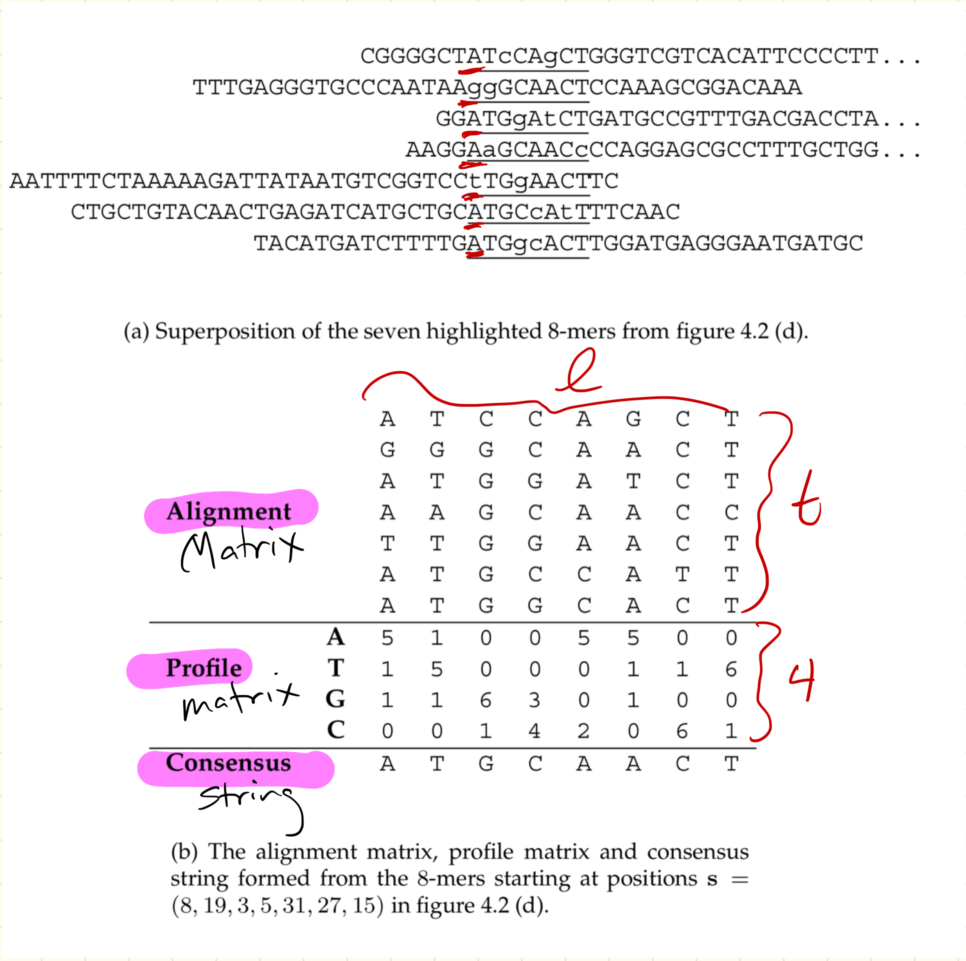
```
CGGGGCTATcCAgCTGGGTCGTCACATTCCCCTTTCGATA
TTTGAGGGTGCCCAATAAggGCAACTCCAAAGCGGACAAA
GGATGgAtCTGATGCCGTTTGACGACCTAAATCAACGGCC
AAGGAaGCAACcCCAGGAGCGCCTTTGCTGGTTCTACCTG
AATTTTCTAAAAAGATTATAATGTCGGTCCtTGgAACTTC
CTGCTGTACAACTGAGATCATGCTGCATGCcAtTTTCAAC
TACATGATCTTTTGATGgcACTTGGATGAGGGAATGATGC
```

(d) Same as (b), but with the implanted pattern ATGCAACT randomly mutated in two positions; no two implanted instances are the same. If we hide the locations as in (c), the difficult problem becomes nearly impossible.

Formalize: • t DNA sequences, l-mers, n nucleotides each

• Select a position in each: $\underline{(s_1, s_2, \ldots, s_t)}$

$+ \; 1 \le s_i \le n-l+1$



```
                       CGGGGCTATcCAgCTGGGTCGTCACATTCCCCTT...
            TTTGAGGGTGCCCAATAAggGCAACTCCAAAGCGGACAAA
                            GGATGGAtCTGATGCCGTTTGACGACCTA...
                          AAGGAaGCAACcCCAGGAGCGCCTTTGCTGG...
AATTTTCTAAAAAGATTATAATGTCGGTCCtTGgAACTTC
    CTGCTGTACAACTGAGATCATGCTGCATGCcAtTTTCAAC
                TACATGATCTTTTGATGgcACTTGGATGAGGGAATGATGC
```

(a) Superposition of the seven highlighted 8-mers from figure 4.2 (d).

l

		A	T	C	C	A	G	C	T
		G	G	G	C	A	A	C	T
		A	T	G	G	A	T	C	T
Alignment Matrix		A	A	G	C	A	A	C	C
		T	T	G	G	A	A	C	T
		A	T	G	C	C	A	T	T
		A	T	G	G	C	A	C	T
	A	5	1	0	0	5	5	0	0
Profile matrix	**T**	1	5	0	0	0	1	1	6
	G	1	1	6	3	0	1	0	0
	C	0	0	1	4	2	0	6	1
Consensus string		A	T	G	C	A	A	C	T

t

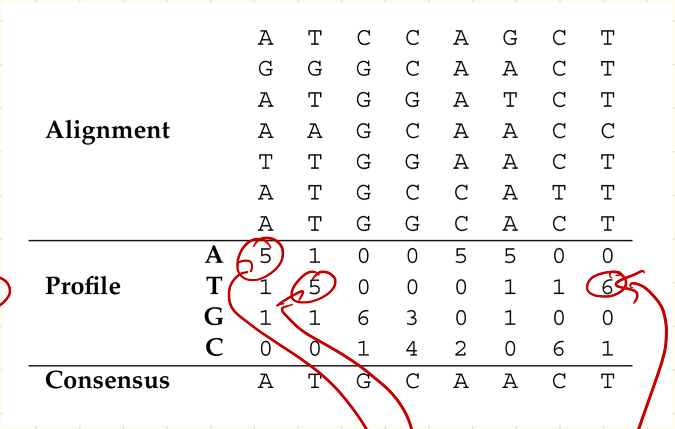
4

(b) The alignment matrix, profile matrix and consensus string formed from the 8-mers starting at positions $\mathbf{s} = (8, 19, 3, 5, 31, 27, 15)$ in figure 4.2 (d).

Notation :

$P(\vec{s})$:= profile matrix wrt starting position vector $\vec{s}$

$M_{P(\vec{s})}(j)$:= largest count in column j of $P(s)$



Alignment		A	T	C	C	A	G	C	T
		G	G	G	C	A	A	C	T
		A	T	G	G	A	T	C	T
		A	A	G	C	A	A	C	C
		T	T	G	G	A	A	C	T
		A	T	G	C	C	A	T	T
		A	T	G	G	C	A	C	T
Profile	**A**	5	1	0	0	5	5	0	0
$P(\vec{s}) \rightarrow$	**T**	1	5	0	0	0	1	1	6
	G	1	1	6	3	0	1	0	0
	C	0	0	1	4	2	0	6	1
Consensus		A	T	G	C	A	A	C	T

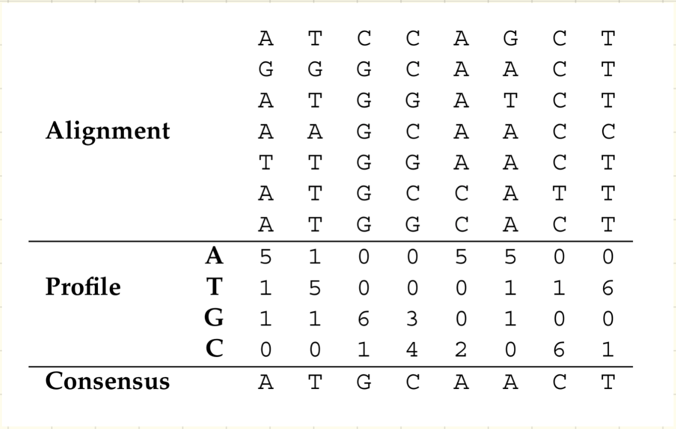
$M_{P(s)}(1) = 5$

$M_{P(s)}(2) = 5$

$M_{P(s)}(8) = 6$

Consensus score:

$$\text{Score}(\bar{s}, DNA) = \sum_{j=1}^{\ell} M_{P(s)}(j)$$



Alignment		A	T	C	C	A	G	C	T
		G	G	G	C	A	A	C	T
		A	T	G	G	A	T	C	T
		A	A	G	C	A	A	C	C
		T	T	G	G	A	A	C	T
		A	T	G	C	C	A	T	T
		A	T	G	G	C	A	C	T
Profile	**A**	5	1	0	0	5	5	0	0
	T	1	5	0	0	0	1	1	6
	G	1	1	6	3	0	1	0	0
	C	0	0	1	4	2	0	6	1
Consensus		A	T	G	C	A	A	C	T

Here,

$$\text{Score}(\bar{s}, DNA) = 5+5+6+4+5+5+6+6$$

Why? Strength of a profile:

$\ell \cdot t$ means best possible alignment - same letter in each spot

$\frac{\ell t}{4}$: worst alignment - equal mix of nucleotides per spot

Motif Finding Problem:
Given a set of DNA sequences, find a set of l-mers, one from each sequence, that maximizes the consensus score.

Input: A $t \times n$ matrix of DNA, and l, the length of the pattern to find.

Output: An array of t starting positions $\mathbf{s} = (s_1, s_2, \ldots, s_t)$ maximizing $Score(\mathbf{s}, DNA)$.

Note: In reality, often use entropy:

Let $p_{i,j}$ be (i,j)th entry in profile.

$$\text{Entropy} = \sum_{j=1}^{l} \sum_{i=1}^{4} \left[\frac{p_{i,j}}{t} \log \frac{p_{i,j}}{t} \right]$$

where $t = \#$ sequences

This is more statistically robust measure

(but algorithm is essentially unchange)

Reframing:

Sift through large # of alternatives to find best one

$(n-\ell+1)^t$ starting positions!

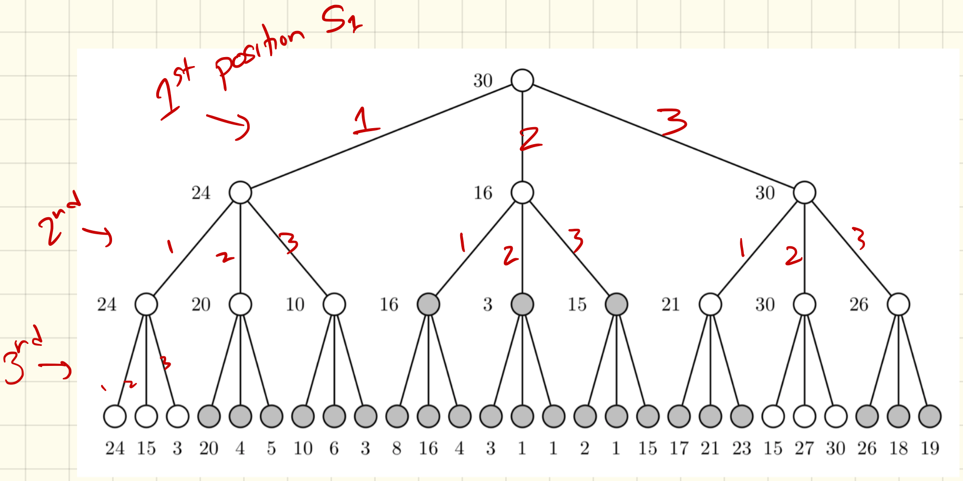
possible $\vec{S}$'s:

$$
\begin{array}{cccccccc}
(& \underline{1}, & 1, & \ldots, & 1, & 1 &) \\
(& 1, & 1, & \ldots, & 1, & 2 &) \\
(& 1, & 1, & \ldots, & 1, & 3 &) \\
 & & & \vdots & & & \\
(& 1, & 1, & \ldots, & 1, & n-l+1 &) \\
(& 1, & 1, & \ldots, & 2, & 1 &) \\
(& 1, & 1, & \ldots, & 2, & 2 &) \\
(& 1, & 1, & \ldots, & 2, & 3 &) \\
 & & & \vdots & & & \\
(& 1, & 1, & \ldots, & 2, & n-l+1 &) \\
 & 2 \rightarrow & & \vdots & & & \\
(& n-l+1, & n-l+1, & \ldots, & n-l+1, & 1 &) \\
(& n-l+1, & n-l+1, & \ldots, & n-l+1, & 2 &) \\
(& n-l+1, & n-l+1, & \ldots, & n-l+1, & 3 &) \\
 & & & \vdots & & & \\
(& n-l+1, & n-l+1, & \ldots, & n-l+1, & n-l+1 &)
\end{array}
$$

(counting in base $n-\ell+1$?)

Branch + bound intuition:

What if we can go partway, but rule out entire subtree?



(More details next time, plus connection to medians)

(through mid 4.6)
rest of Ch 4 on Thursday