Seeding Methods in Homology Search

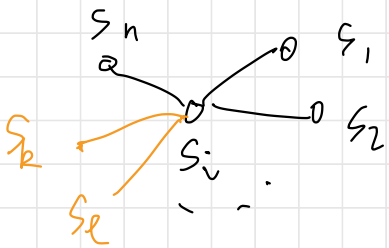
Review:

① multiple alignment (exact) algorithm.

② Relative entropy score for each column.

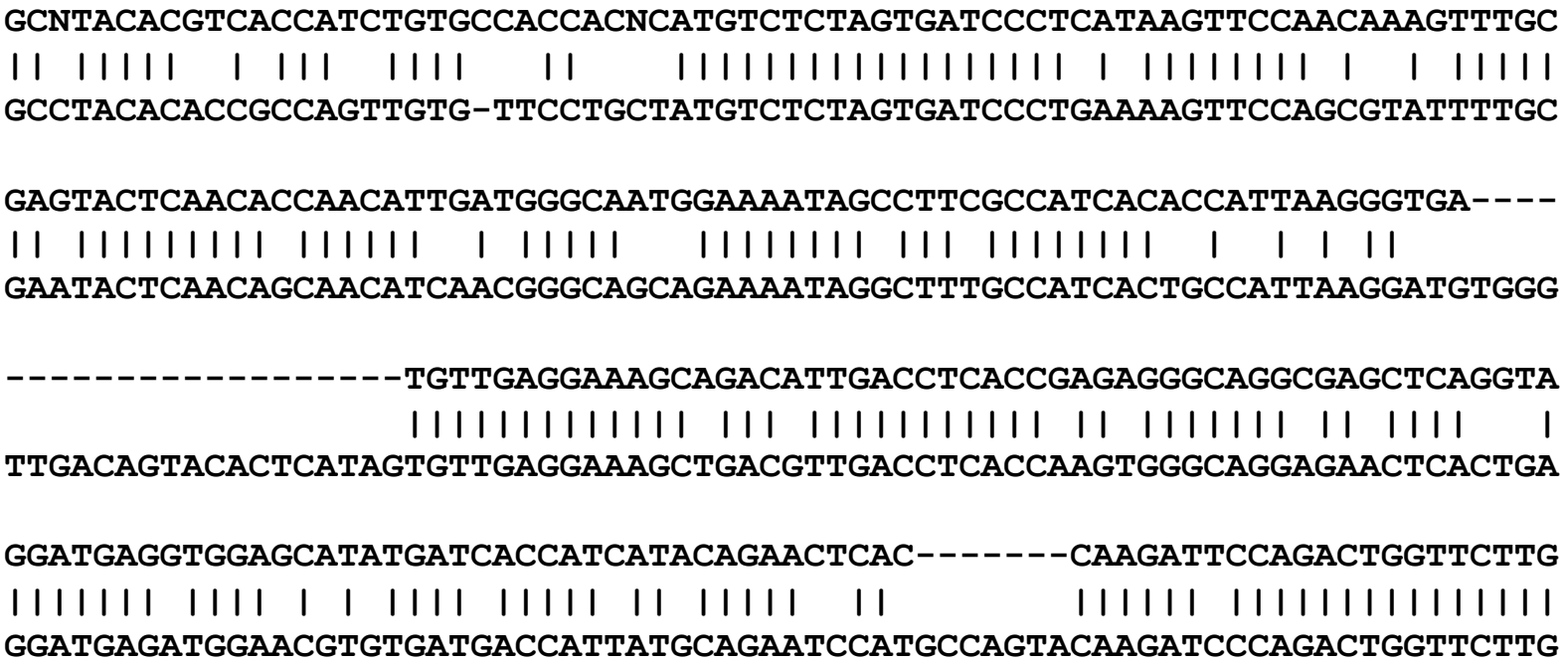
$$\sum_{a \in \Sigma} n(a) \cdot \log \frac{n(a)/n}{q(a)}$$

③ Approximation algorithm for SP-score with Triangular Inequality.



$$d_i'(S_k, S_\ell) \le d(S_i, S_k) + d(S_i, S_\ell)$$
$$\le d^*(S_i, S_k) + d^*(S_i, S_\ell)$$

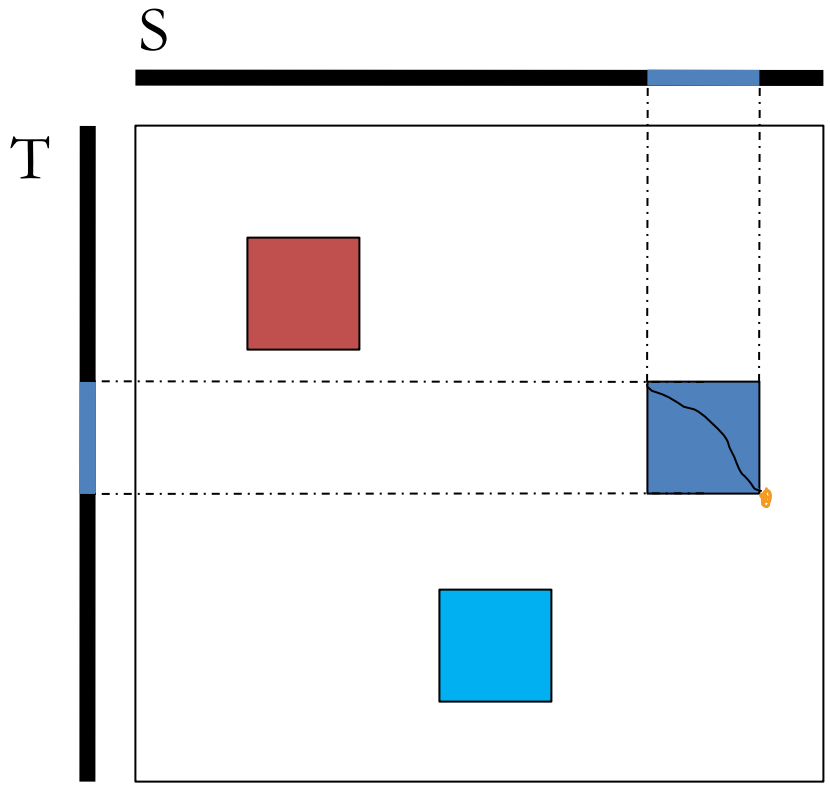
A similarity between mouse and human genomes



Smith-Waterman is the most accurate method.

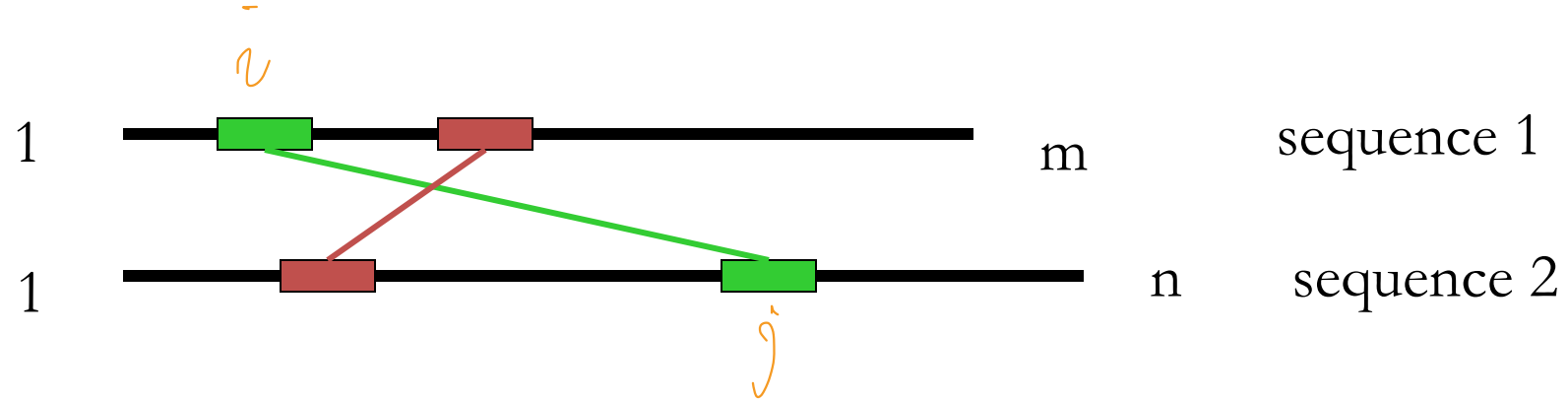
Time complexity： O(mn).

Smith-Waterman Algorithm



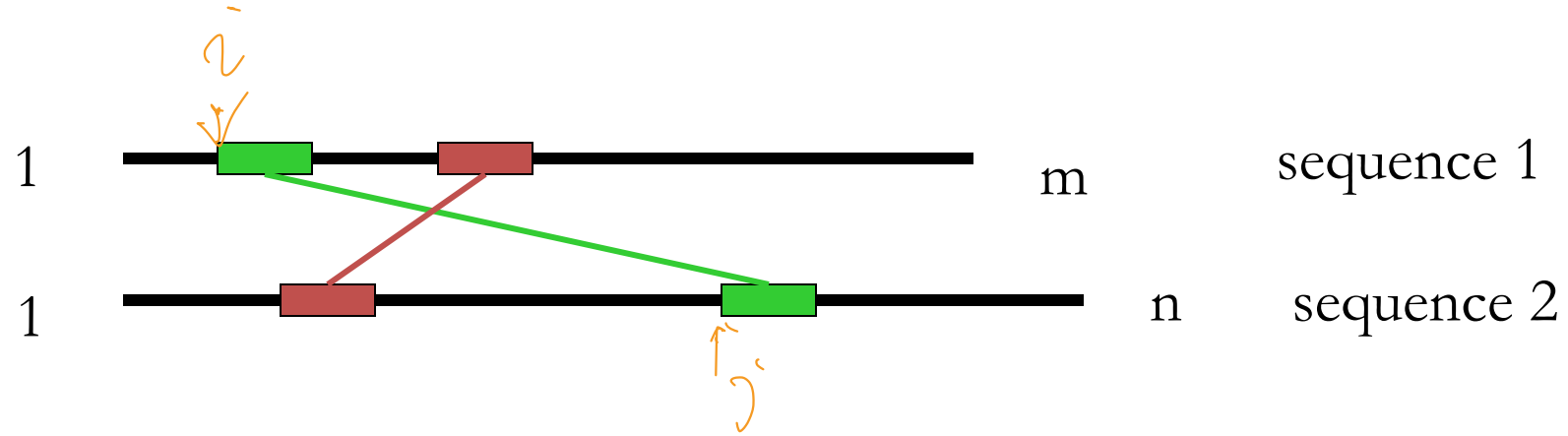
- The old algorithm requires O(mn) and is too slow.
- Human v.s. mouse: $3x10^9x3x10^9=9x10^{18}$

Similarity Search



- Most similarities (local alignments) are very short relative to the genomes.

Similarity Search



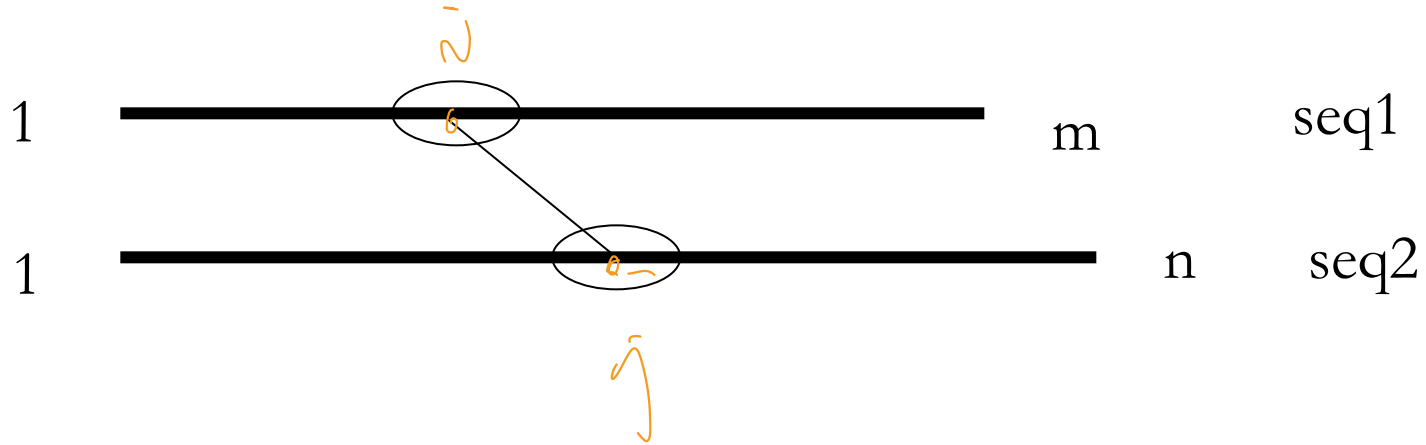
- For every pairs of (i, j), build a local alignment around it.
  - O(mnT)
  - Not better than Smith-Waterman.
- But this leads to an important idea…

Main Idea

- Most pairs of (i, j) are useless. We only want to try local alignments on the "promising" pairs of (i, j).
- In the context of sequence similarity search in bioinformatics, these "promising" pairs are called "seeds" or "hits".
- We need
  - a proper definition of hits.
  - some efficient way to enumerate the hits faster than trying every pair of (i, j).

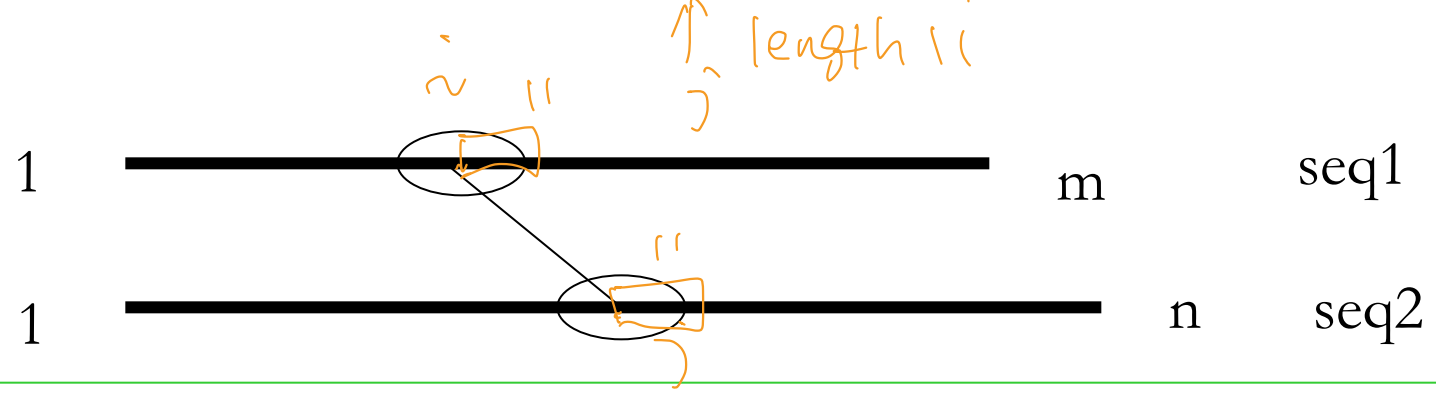
BLAST Uses Short Consecutive Match as Hits

```
GCNTACACGTCACCATCTGTGCCACCACNCATGTCTCTAGTGATCCCTCATAAGTTCCAACAAAGTTTGC
||  |||||   |  |||   ||||    ||     ||||||||||||||||||  |  ||||||||  |   |  |||||
GCCTACACACCGCCAGTTGTG-TTCCTGCTATGTCTCTAGTGATCCCTGAAAAGTTCCAGCGTATTTTGC
```



BLAST Uses Short Consecutive Match as Hits

```
GCNTACACGTCACCATCTGTGCCACCACNCATGTCTCTAGTGATCCCTCATAAGTTCCAACAAAGTTTGC
||  |||||   |  |||   ||||   ||    ||||||||||||||||||  |  ||||||||  |   |  |||||
GCCTACACACCGCCAGTTGTG-TTCCTGCTATGTCTCTAGTGATCCCTGAAAAGTTCCAGCGTATTTTGC
```



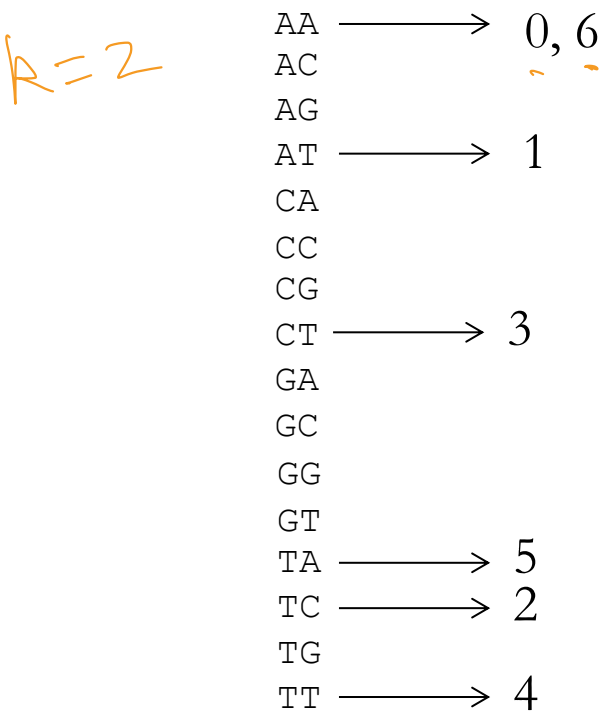
- Majority of (i,j) are random and probability of generating a random hit is small.
- For length-k seed, time complexity becomes $O(4^{-k}mnT)$
- By default, BLAST used k=11.
- What's the speed up factor for k=11?

The Idea behind Seeding

- A true similarity has a high chance of being hit. (Hitting at any position in the similarity will do.)
- A random pair (i, j) has low chance of being hit.
- Thus, if we use hit to filter (i, j), we will
  - Detect most true similarities.
  - Not wasting time on random pairs of (i, j).

The Data Structure for Finding Hit?

- for each *k*-mer, index table to remember all its occurrences in S.
- for each *k*-mer of T, find its hits in the index table.
- The index table can be a trie or a hash table.

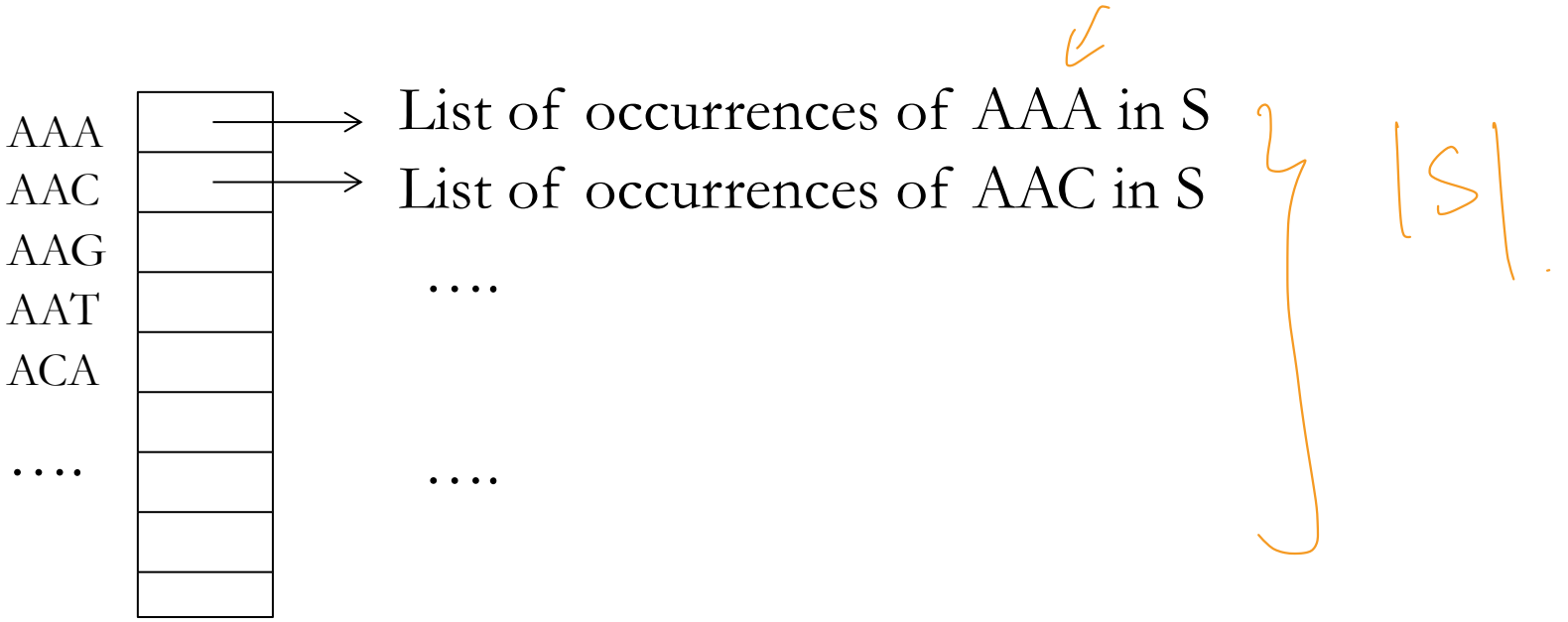


S: AATCTTAA
01234567

T: GAACTTA

10

The Data Structure for Finding Hit?



Space complexity?

4^k $4^{11} \approx 4M$.

$k = 11$

Overall runtime

- Build the index using S: O(n) time.
- Find matches between the index and sequence T: O(m) time to scan T, plus we need to examine all of the N hits found. Let t be the examination time. Then O(m+Nt).
- Overall runtime: O(n+m+Nt).
- The term Nt is the most expensive part. Indexing overhead is small.
- In practice, most of the hits encountered are random hits.

Filtration can have multiple rounds

```
GCNTACACGTCACCATCTGTGCCACCAGCCATGTCTCTAGTGATCCCTCATGGTGGCCAACAAAGTTTGC
          |  |   |      || || |||||||||| |||||| |      ||| |  | |||||
TGCCTACACACCGCCAGTTGTGTTCCTGCTATGTCTCTAGTTATCCCTGAAAAGTTCCAGCGTATTTTGC
```

- After finding a hit, instead of trying to build a local alignment directly, BLAST uses another round of filtration to determine if a hit is a "good" or "bad" hit.
- Quick search in both directions; if most symbols match, it's a good hit. Otherwise it's bad.
  - More precisely, use ungapped extension to find HSPs.
- If an HSP is above a certain score threshold, build a local alignment around it.

HSP extension

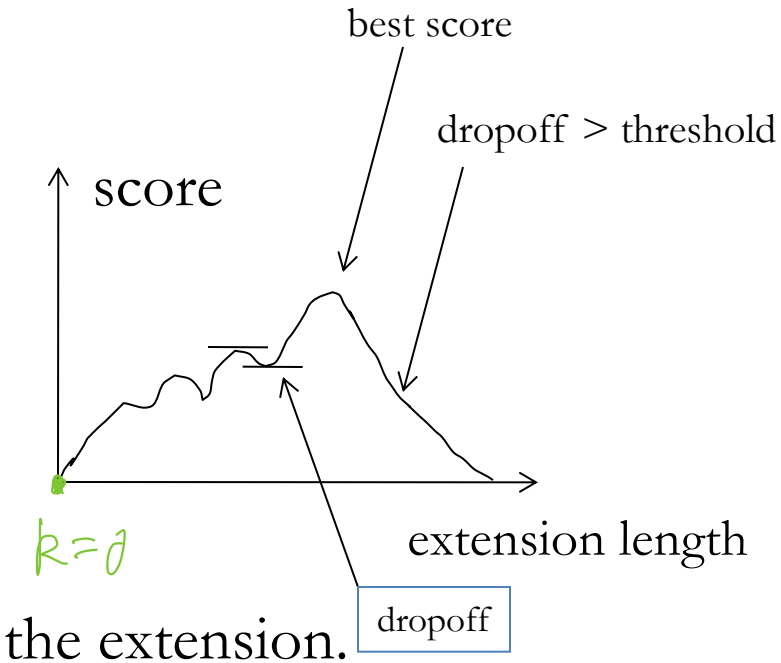
```
GCNTACACGTCACCATCTGTGCCACCAGCCATGTCTCTAGTGATCCCTCATGGTGGCCAACAAAGTTTGC
          |  |   |      || || |||||||||| |||||| |      ||| |  | |||||
TGCCTACACACCGCCAGTTGTGTTCCTGCTATGTCTCTAGTTATCCCTGAAAAGTTCCAGCGTATTTTGC
```

for k from 0 to …

    score += sc(S[i+k],T[j+k])

for k from 1 to …

    score += sc(S[i-k],T[j-k])



- But when to stop?
- Score will increase and decrease during the extension.
- Extension stops when drop off greater than threshold.

HSP Extension

- How long will the extension continue after reaching best score?
- Assumptions:
  - After reaching best score, sequence becomes random.
  - match=1 and mismatch=-1
- Expected score on each additional base is -0.5.
- If dropoff=k, then after 2k bases, the expected dropoff will reach k.
- Conclusion: Not too long.

prob	score
$\frac{1}{4}$	1
$\frac{3}{4}$	-1

Example of missing a target

- Fail:

```
GAGTACTCAACACCAACATTAGTGGGCAATGGAAAAT
||  |||||||||  ||||||  |  ||||||    ||||||
GAATACTCAACAGCAACATCAATGGGCAGCAGAAAAT
```

- Dilemma
  - **Sensitivity** – needs shorter seeds
    - the success rate of finding a homology
  - **Speed** – needs longer seeds
    - Mega-BLAST uses seeds of length 28.

PatternHunter uses "spaced seeds"

- 111*1**1*1**11*111 (called a spaced seed)
  - Eleven required matches (**weight**=11, **length** = 18)
  - Seven "don't care" positions

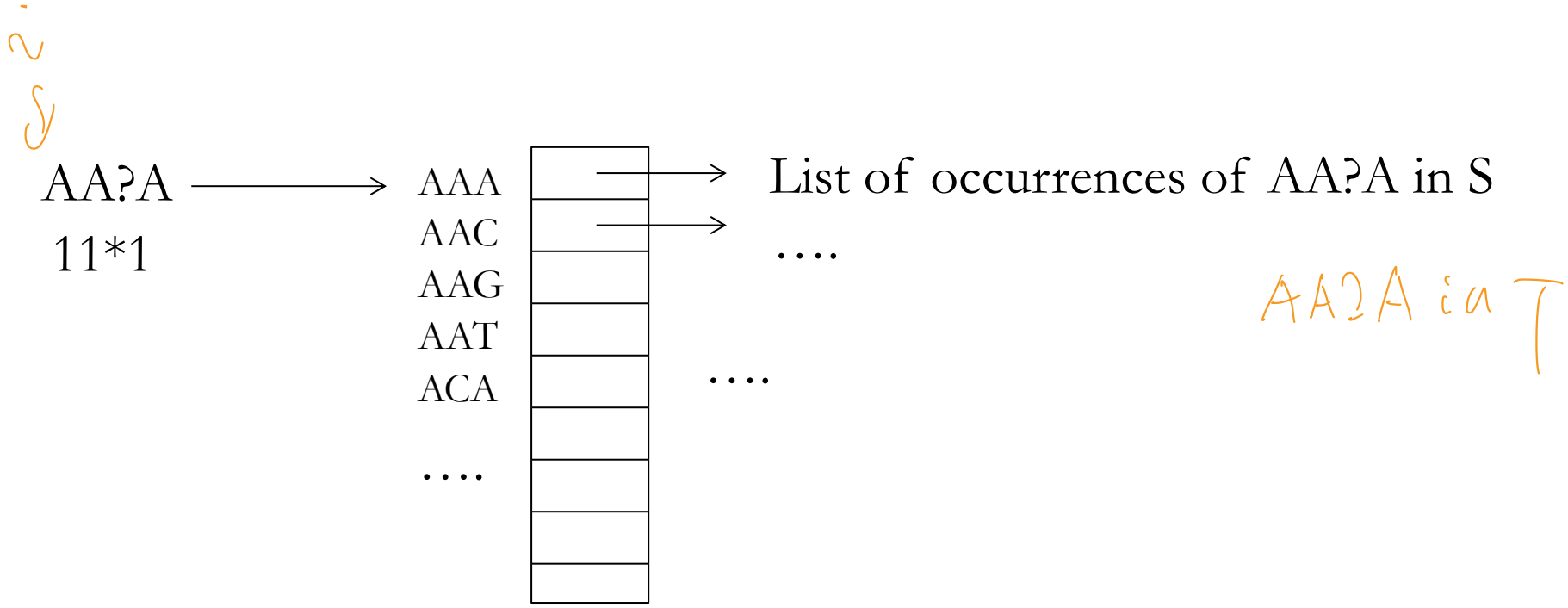
- Hit = all the required matches are satisfied.
- BLAST's seed = 11111111111

Notes about the notation

- A homology/similarity region's actual sequences do not matter, the match/mismatch matters.
- Therefore, a region is often denoted by a binary 0-1 sequence, **`110111110011101110111`**
- A hit is then as follows:
- **`110111110011101110111`**
- **`  111*1**1*1**11*111`**

The Data Structure for Finding Hit

- The same as consecutive seed. Except that now we have a length *l*, weight *w* seed. E.g. 11*1.
  - Each *l*-mer, take the *w* letters out and put in index table.
- The index table can be a hash table.



Time Complexity Comparison

- Lemma: for random sequence S and T with lengths m and n, the expected number of hits for weight w, length l seed is

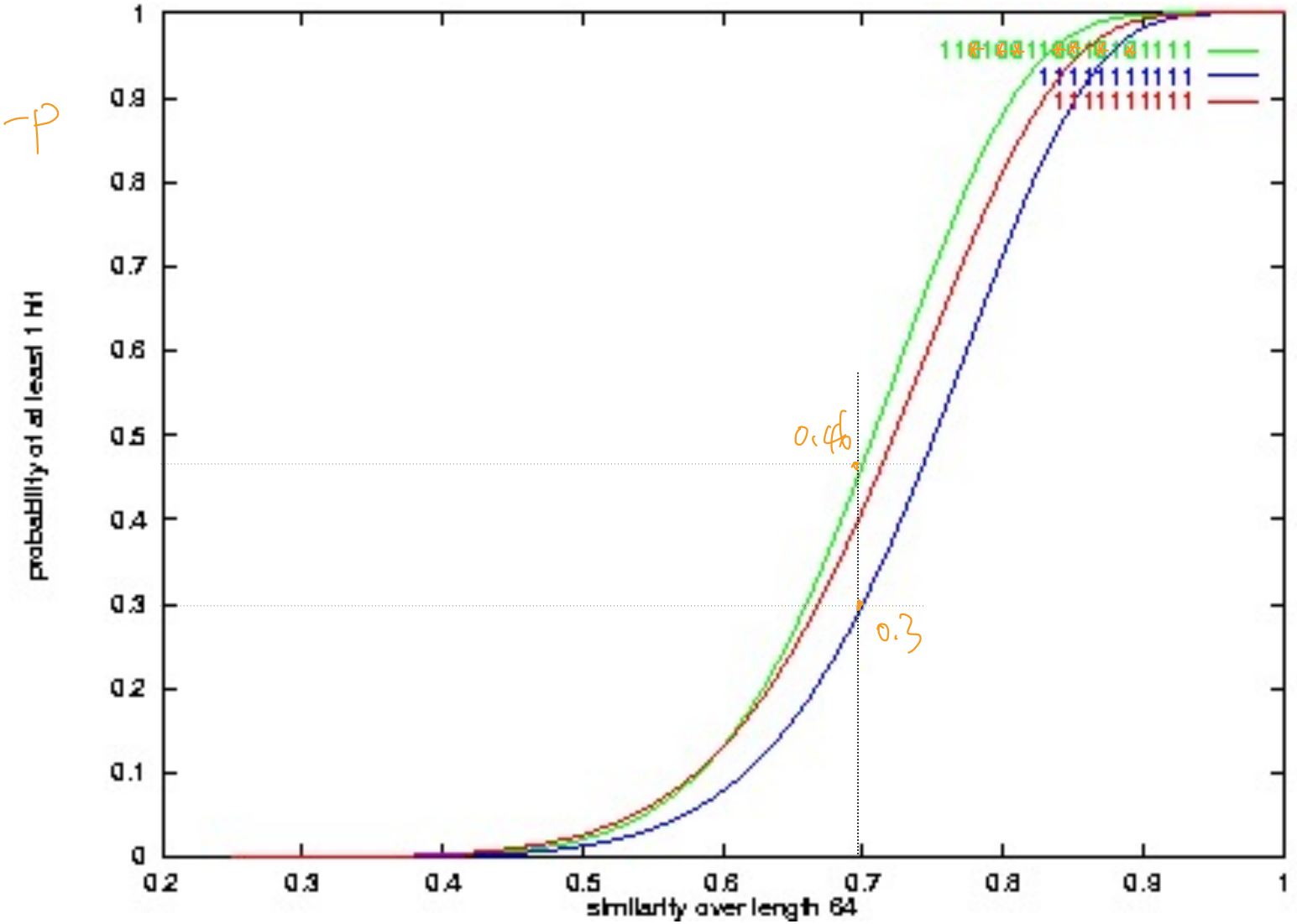
$$(m-l+1)(n-l+1)4^{-w}$$

- Because usually l is much shorter than S and T, this is approximately $4^{-w}mn$
- That is, the expected number of hits in random regions only depends on the weight, but not the shape of the seed. So does the running time.
- So, speed-wise, spaced seed is similar to consecutive seed.
- What about the sensitivity?

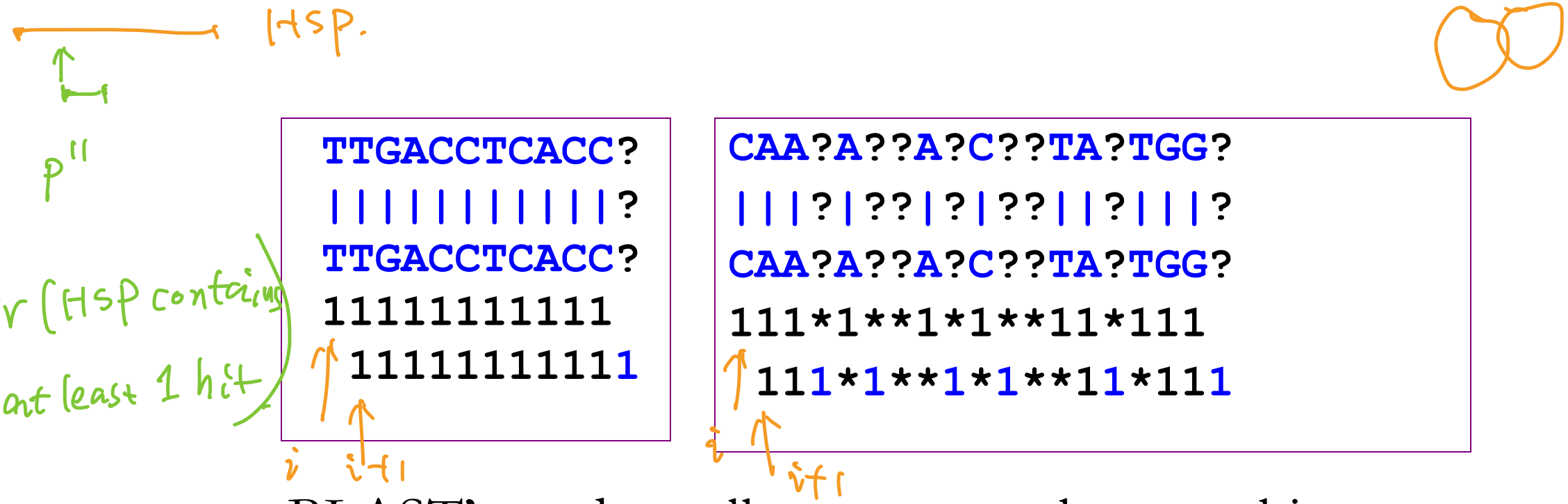
Simulated sensitivity curves

prob(1) = p

prob(0) = 1 − p



Why spaced seeds are better?



- BLAST's seed usually uses more than one hits to detect one homology (redundant)
- Spaced seeds uses fewer hits to detect one homology (efficient)