CRAST Document

Heartsh (email: heartsh@heartsh.io)

Homologous-ncRNA search in genomic scale

- Homologous-ncRNA search considering the secondary structure requires more than $O(n^4)$ time/$O(n^3)$ space complexity.

- For example, search of several human ncRNAs in all (about 18,000) house mouse ones is impossible unless you use a supercomputer.

- The factor of the heavy complexity = simultaneously solving sequence alignment (with $O(n^2)$) & RNA folding (with $O(n^2)$ ≦).

- **If the secondary structure were NOT explicitly but implicitly (probabilistically) considered, BLAST-like search could be enabled?**

Alignment with folding

- **The Sankoff algorithm: an algorithm for the simultaneous alignment with folding (with the O(n^6) time/O(n^4) space complexity).**

- The strict algorithm is impractical even these days.

- Even heuristics such as Foldalign (with pruning) & banded Sankoff alg. require more than O(n^4) time/ O(n^3) space complexity.

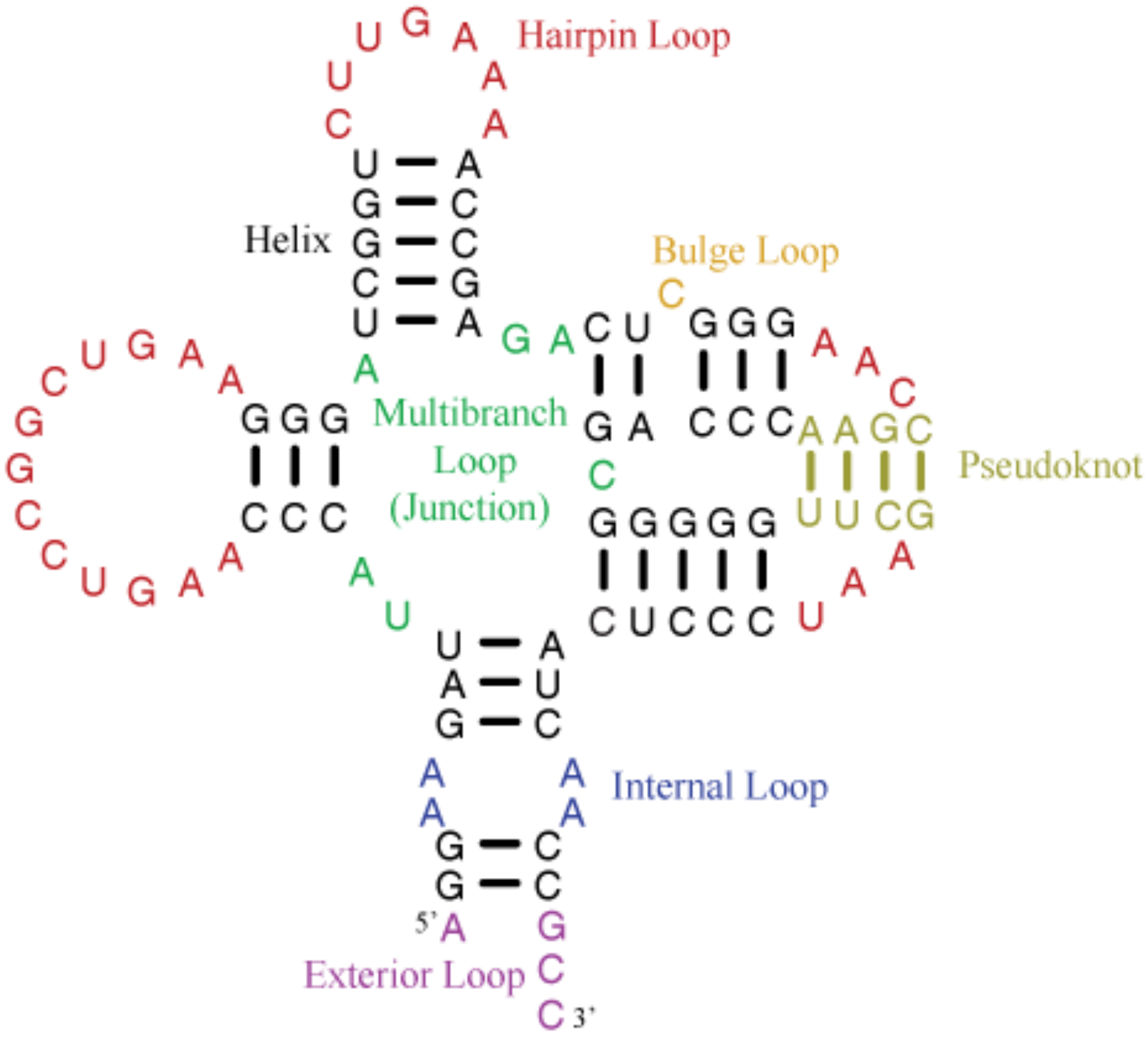
NcRNA context probability distribution

- Complexity for ncRNA alignment wouldn't be less than the heuristics as long as we explicitly considered the secondary structure by the folding?

- CapR & RNAplfold estimate a probability distribution of secondary structure motifs formed in each base in O(w^2n; w: a maximum span between a base pair).

- We consider the parameter w as a constant then the complexity can be regarded as O(n).

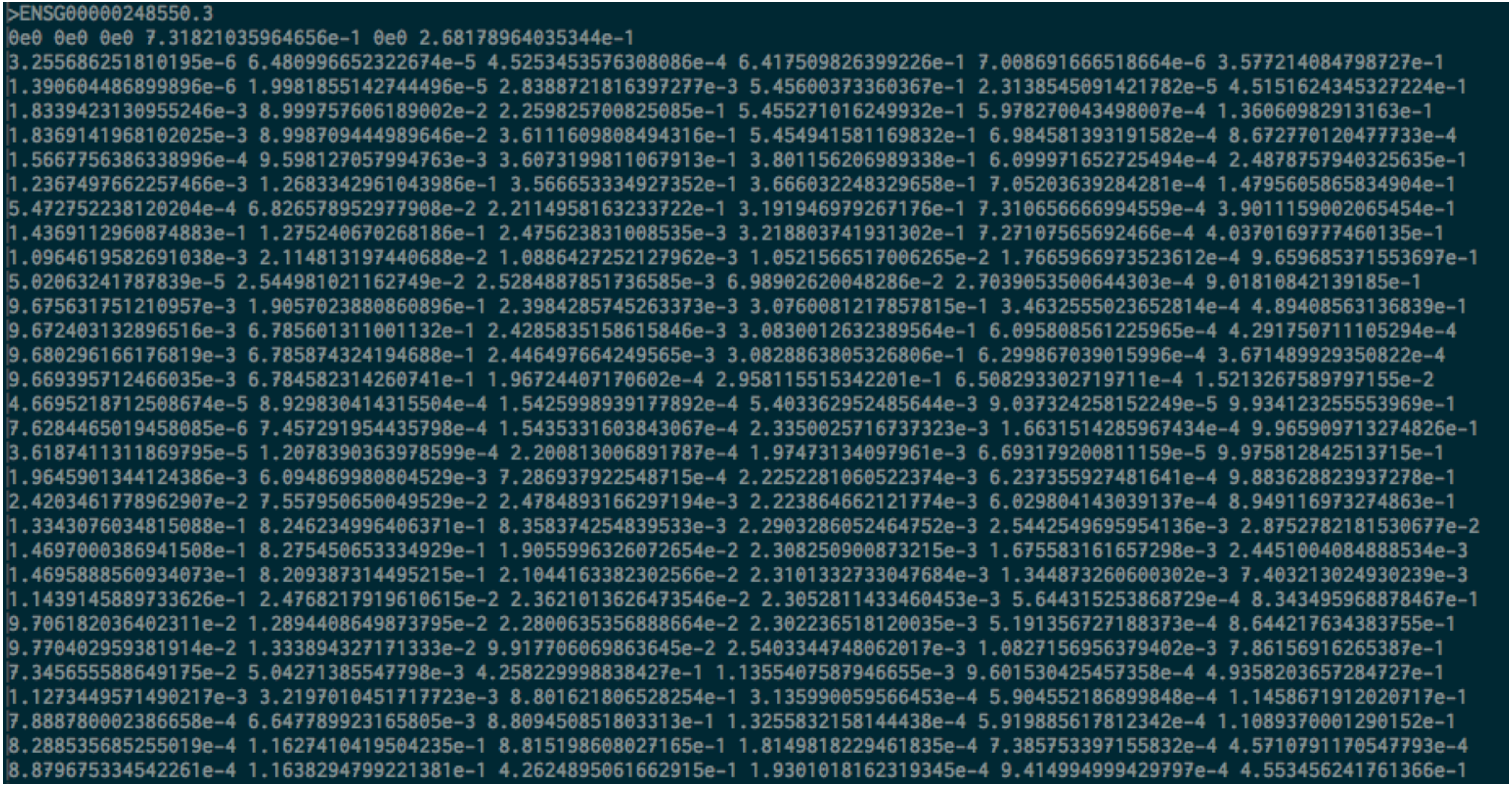
- **If the secondary structure were converted into a line of the probability distributions, the complexity would be reduced into less than the heuristics?**



RNA motif

Pseudo-knot is usually not taken into account since it is nested (in the figure, interposed between the 2 helices).

AGCAAATGTGCTGAAGCTCTT...



```
>ENSG00000248550.3
0e0 0e0 0e0 7.31821035964656e-1 0e0 2.68178964035344e-1
3.255686251810195e-6 6.480996652322674e-5 4.5253453576308086e-4 6.417509826399226e-1 7.008691666518664e-6 3.577214084798727e-1
1.390604486899896e-6 1.9981855142744496e-5 2.838872181639727e-3 5.45600373360367e-1 2.3138545091421782e-5 4.5151624345327224e-1
1.8339423130955246e-3 8.999757606189002e-2 2.259825700825085e-1 5.455271016249932e-1 5.978270043498007e-4 1.36060982913163e-1
1.8369141968102025e-3 8.998709444989646e-2 3.6111609808494316e-1 5.454941581169832e-1 6.984581393191582e-4 8.672770120477733e-4
1.5667756386338996e-4 9.598127057994763e-3 3.6073199811067913e-1 3.801156206989338e-1 6.099971652725494e-4 2.4878757940325635e-1
1.2367497662257466e-3 1.2683342961043986e-1 3.566653334927352e-1 3.666032248329658e-1 7.05203639284281e-4 1.4795605865834904e-1
5.472752238120204e-4 6.826578952977908e-2 2.2114958163233722e-1 3.191946979267176e-1 7.310656666994559e-4 3.9011159002065454e-1
1.4369112960874883e-1 1.275240670268186e-1 2.475623831008535e-3 3.218803741931302e-1 7.2710756565692466e-4 4.0370169777460135e-1
1.0964619582691038e-3 2.114813197440688e-2 1.0886427252127962e-3 1.0521566517006265e-2 1.7665966973523612e-4 9.659685371553697e-1
5.02063241787839e-5 2.544981021162749e-2 2.5284887851736585e-3 6.98902620048286e-2 2.7039053500644303e-4 9.01810842139185e-1
9.675631751210957e-3 1.9057023880860896e-1 2.3984285745263373e-3 3.0760081217857815e-1 3.4632555023652814e-4 4.89408563136839e-1
9.672403132896516e-3 6.785601311001132e-1 2.4285835158615846e-3 3.0830012632389564e-1 6.095808561225965e-4 4.291750711105294e-4
9.680296166176819e-3 6.785874324194688e-1 2.446497664249565e-3 3.0828863805326806e-1 6.299867039015996e-4 3.671489929350822e-4
9.669395712466035e-3 6.784582314260741e-1 1.96724407170602e-4 2.958115515342201e-1 6.508293302719711e-4 1.5213267589797155e-2
4.6695218712508674e-5 8.929830414315504e-4 1.5425998939177892e-4 5.403362952485644e-3 9.037324258152249e-5 9.934123255553969e-1
7.6284465019458085e-6 7.457291954435798e-4 1.5435331603843067e-4 2.3350025716737323e-3 1.6631514285967434e-4 9.965909713274826e-1
3.6187411311869795e-5 1.2078390363978599e-4 2.200813006891787e-4 1.97473134097961e-3 6.693179200811159e-5 9.975812842513715e-1
1.9645901344124386e-3 6.094869980804529e-3 7.286937922548715e-4 2.2252281060522374e-3 6.237355927481641e-4 9.883628823937278e-1
2.4203461778962907e-2 7.557950650049529e-2 2.4784893166297194e-3 2.223864662121774e-3 6.029804143039137e-4 8.949116973274863e-1
1.3343076034815088e-1 8.246234996406371e-1 8.358374254839533e-3 2.2903286052464752e-3 2.5442549695954136e-3 2.8752782181530677e-2
1.4697000386941508e-1 8.275450653334929e-1 1.9055996326072654e-2 2.308250900873215e-3 1.675583161657298e-3 2.4451004084888534e-3
1.4695888560934073e-1 8.209387314495215e-1 2.1044163382302566e-2 2.3101332733047684e-3 1.344873260600302e-3 7.403213024930239e-3
1.1439145889733626e-1 2.4768217919610615e-2 2.3621013626473546e-2 2.3052811433460453e-3 5.644315253868729e-4 8.343495968878467e-1
9.706182036402311e-2 1.2894408649873795e-2 2.2800635356888664e-2 2.302236518120035e-3 5.191356727188373e-4 8.644217634383755e-1
9.770402959381914e-2 1.333894327171333e-2 9.917706069863645e-2 2.5403344748062017e-3 1.0827156956379402e-3 7.86156916265387e-1
7.345655588649175e-2 5.04271385547798e-3 4.258229998838427e-1 1.1355407587946655e-3 9.601530425457358e-4 4.9358203657284727e-1
1.1273449571490217e-3 3.2197010451717723e-3 8.801621806528254e-1 3.135990059566453e-4 5.904552186899848e-4 1.1458671912020717e-1
7.888780002386658e-4 6.647789923165805e-3 8.809450851803313e-1 1.3255832158144438e-4 5.919885617812342e-4 1.1089370001290152e-1
8.288535685255019e-4 1.1627410419504235e-1 8.815198608027165e-1 1.8149818229461835e-4 7.385753397155832e-4 4.5710791170547793e-4
8.879675334542261e-4 1.1638294799221381e-1 4.2624895061662915e-1 1.9301018162319345e-4 9.414994999429797e-4 4.553456241761366e-1
```

Example line of ncRNA context prob. dists

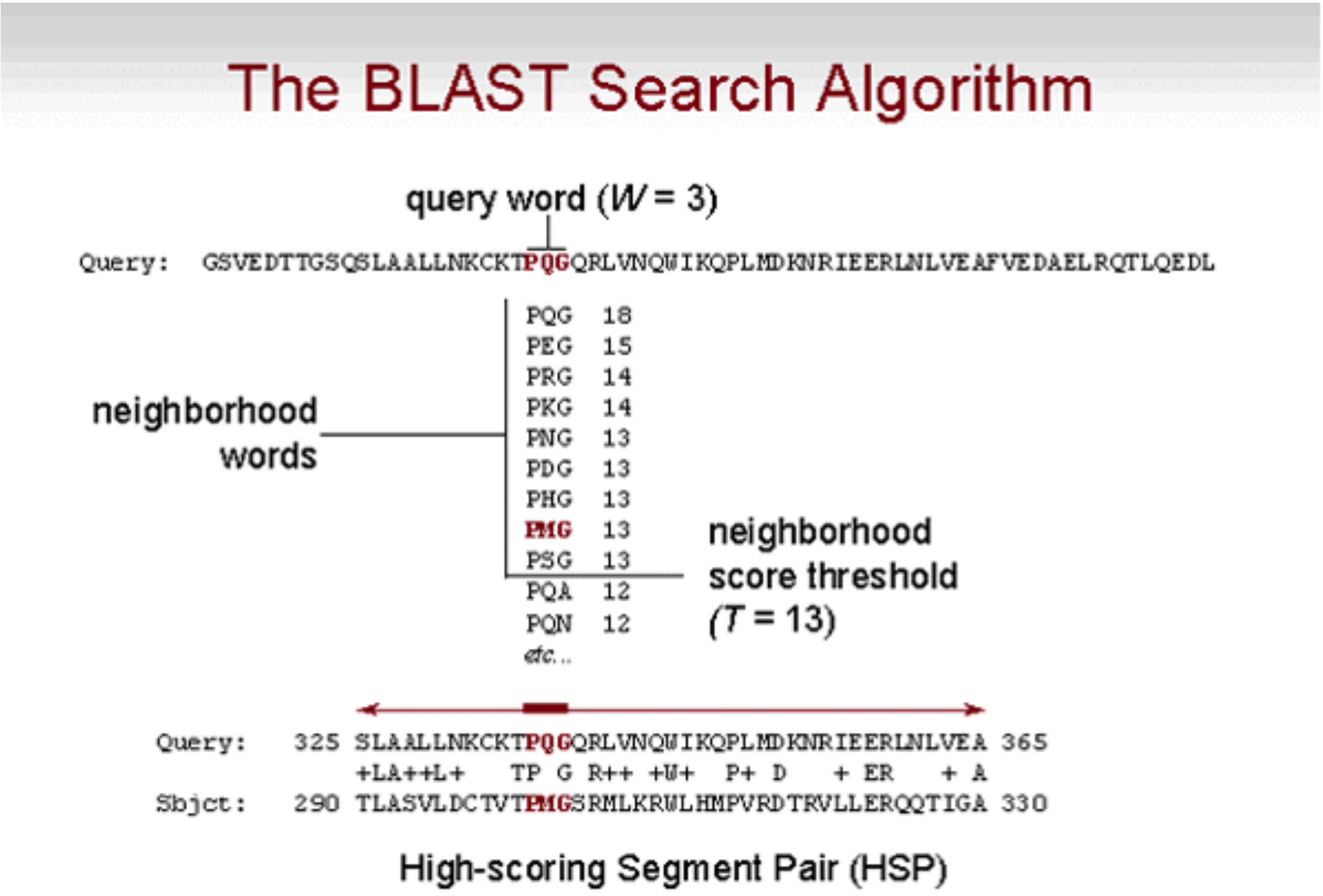
From left, probabilities of bulge/internal/hairpin/exterior/multi-branch loop/helix.
e-x = 10^{-x} and the figure is for a certain human lncRNA processed by CapR.

BLAST

- BLAST: a heuristic of the Smith-Waterman algorithm (with O(mn)) for pairwise sequence alignment.

- **BLAST solves final gapped alignment only when seeds & ungapped/gapped ones satisfying certain conditions are obtained between any 2 sequences.**

- For the seed search, binary search in suffix array (with O(log(n))) & hash-map (with O(1)) are used.



The BLAST Search Algorithm

query word ($W = 3$)
neighborhood words
neighborhood score threshold ($T = 13$)
High-scoring Segment Pair (HSP)

```
Query:   GSVEDTTGSQSLAALLNKCKTPQGQRLVNQWIKQPLMDKNRIEERLNLVEAFVEDAELRQTLQEDL
```

```
PQG  18
PEG  15
PRG  14
PKG  14
PNG  13
PDG  13
PHG  13
PMG  13
PSG  13
PQA  12
PQN  12
etc...
```

```
Query:   325 SLAALLNKCKTPQGQRLVNQWIKQPLMDKNRIEERLNLVEA 365
             +LA++L+    TP G R++ +W+  P+ D   + ER   + A
Sbjct:   290 TLASVLDCTVTPMGSRMLKRWLHMPVRDTRVLLERQQTIGA 330
```

BLAST1 algo. overview

In BLAST2, the gapped one using dynamic programming (DP) is performed after ungapped/gapped one (to generate only 1 alignment).

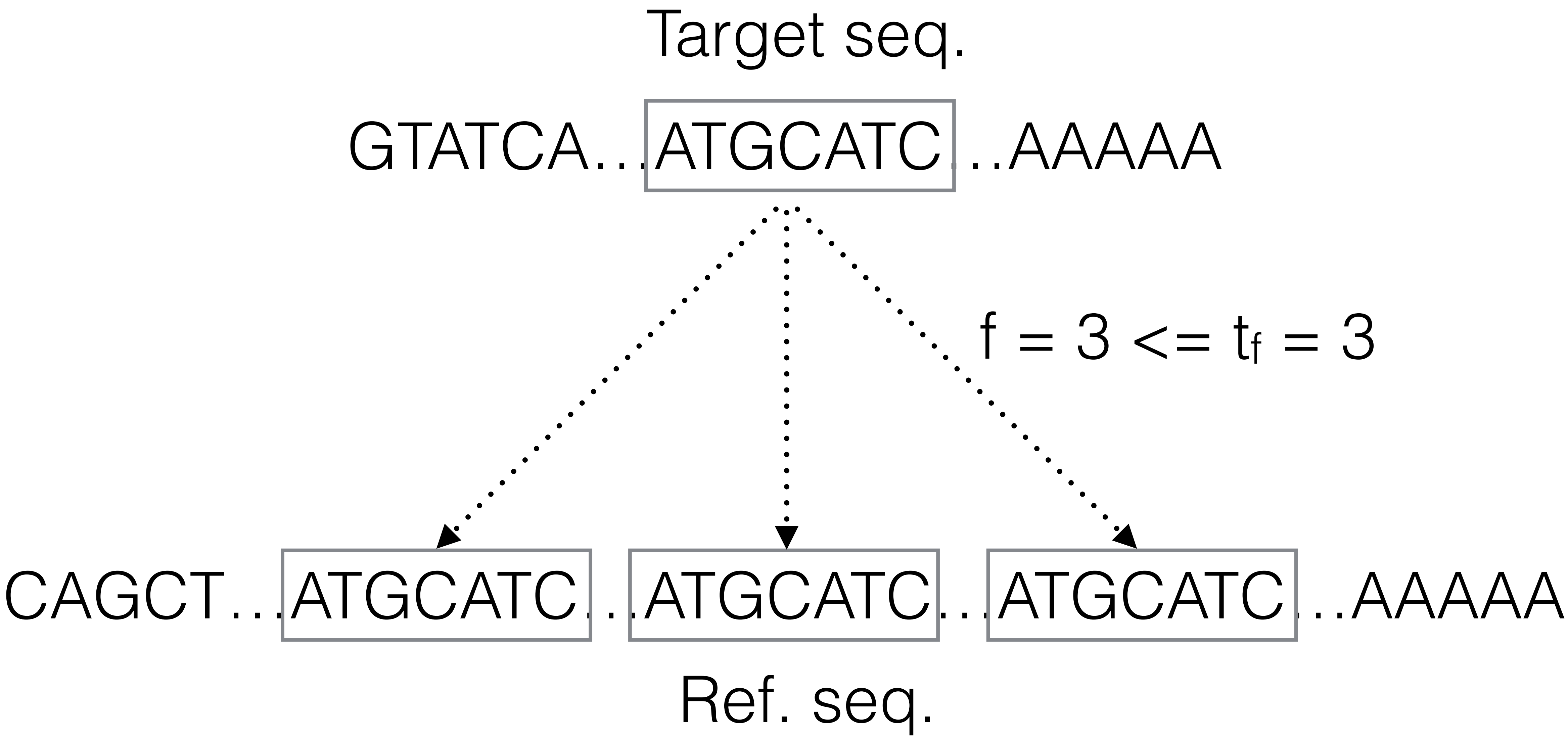
LAST

- LAST: another heuristic with fixed-length (short) seed of BLAST replaced with short seed in equal to/less than a certain frequency.

- **A number of the seeds linearly increases, then the complexity is O(n). (BLAST quadratically does.)**

- A number of seeds actually observed in BLAST is rather large. (The sensitivity becomes low.)

- The reason of the increment = a base distribution on a biological sequence in reality differs from a uniform one.



Target seq.

GTATCA…ATGCATC…AAAAA

$f = 3 <= t_f = 3$

CAGCT…ATGCATC…ATGCATC…ATGCATC…AAAAA

Ref. seq.

LAST seed abstract

Any short rare sequence in a reference one = a seed in LAST.
Even if we use the same threshold, its length differs between genome & script.

Context RNA Alignment Search Tool

- We’ve created ncRNA alignment tool based on LAST called CRAST.
- **We’ve confirmed it increased TPs with 2 & reduced FPs into less than 1/3 when having experimented using human 34 lncRNAs having homologs in house mouse and all (18,185) house mouse ncRNAs.**
- **It reduces the seeds by adding a condition of similarity of a pair of lines of the probability distributions to LAST seed one.**
- Measuring similarity of probability distribution = doing distance of one.
- Using <u>the Jensen-Shannon distance</u> as one of pairs of the distributions p, q, we score $d(p, q) < d_t$ as a match with +1 otherwise as a mismatch with -1.
- The added condition = a threshold of an expected number of seeds based on a binomial distribution and the score.

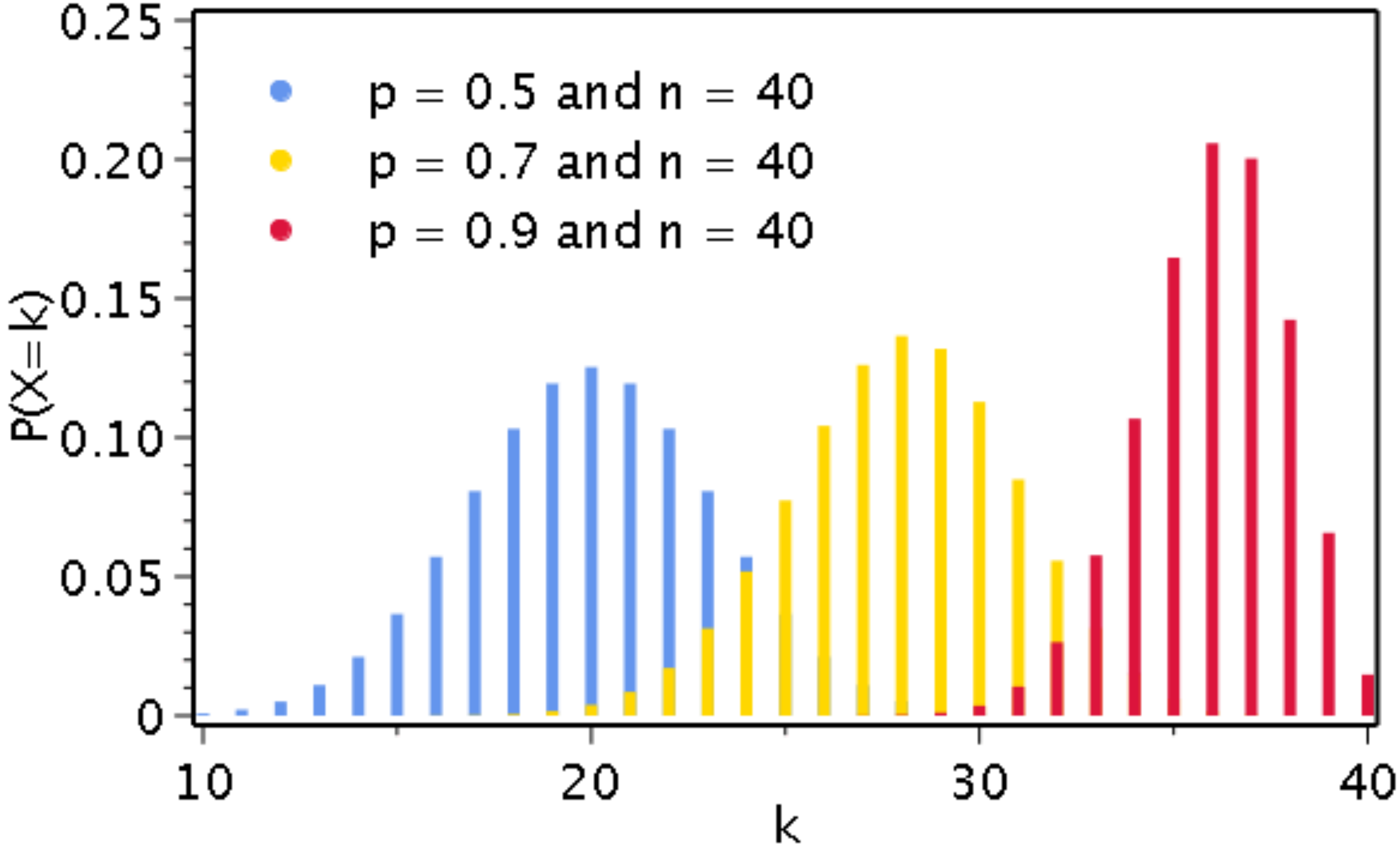
Binomial distribution

- A binomial distribution: a probability one to model an observation number of 2 conditions such as the front & back of a coin within a certain number of trials. (e.g., 3 times of the front of a biased coin with a probability p = 0.25 s.t. the front is observed.)

- The match & mismatch obeys this distribution. (The match probability 0 < p (= d_t) < 0.5.)

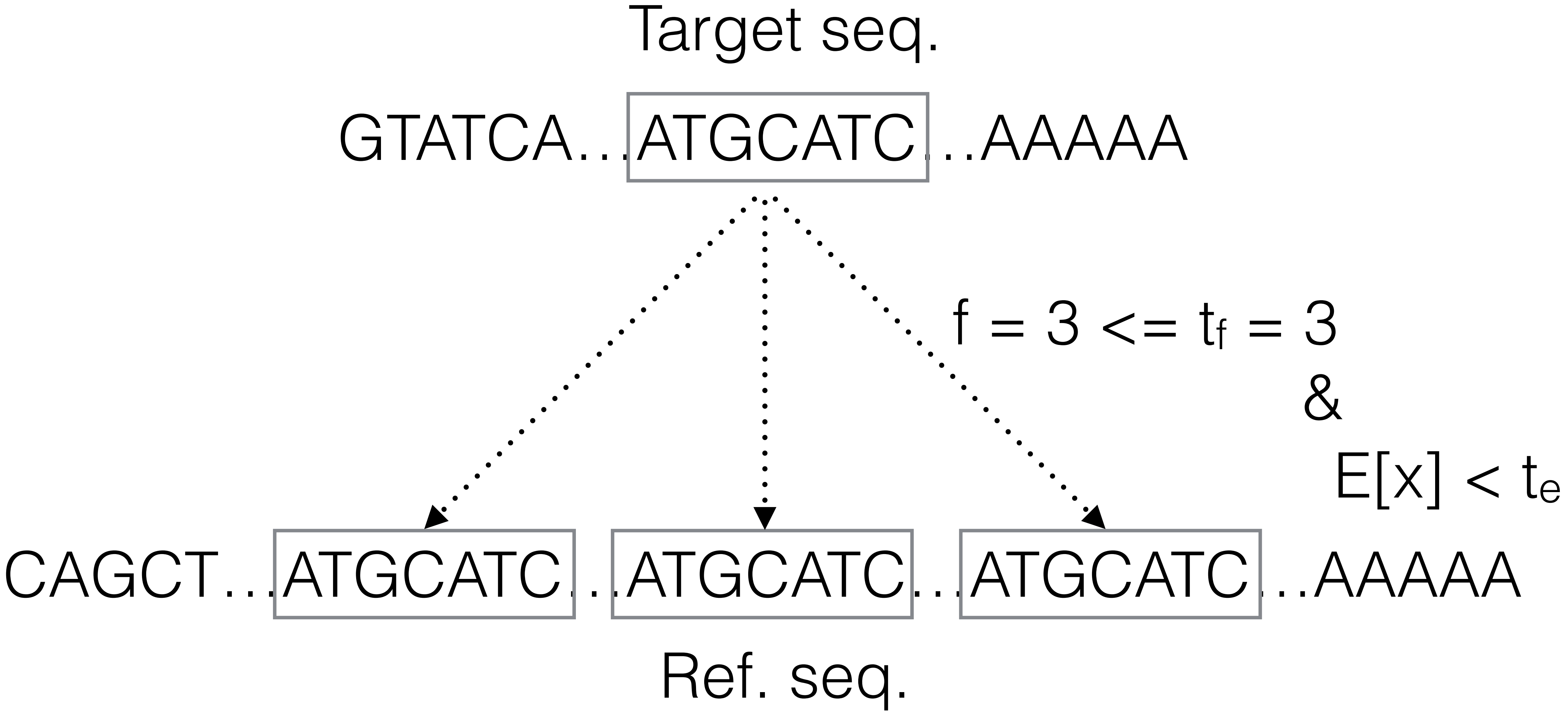
- **An expected number x of seeds of a length N s.t. times the distributions matches is equal to/greater than n E[xl n, N] = (target seq. len. - N + 1) * (1 - P(x ≦ n)).**

- p < 0.5 is to let the expected score less than 0 & enable to recognize significant alignments against the others. (If the expectation is equal to/ greater than 0, alignment is possible even if random sequences are used.)



Binomial distributions

The more p differs from 0.5, the more the mean moves & the variance gets small.



Target seq.

GTATCA…ATGCATC…AAAAA

$f = 3 <= t_f = 3$

&

$E[x] < t_e$

CAGCT…ATGCATC…ATGCATC…ATGCATC…AAAAA

Ref. seq.

CRAST seed overview

A condition for an expectation based on similarity of the distribution lines is added to the LAST seed condition.

Scoring System

- The match/mismatch score of a pair of bases = +1/-1, the gap open/extension penalty = -7/-1. (To compare it with LAST, let them be the same as LAST.)

- **We let the alignment score as a combination of the base/distribution score.**

- $s = rs_b + (1 - r)s_c$, $0 \leqq r \leqq 1$: a contribution ratio of base to the score, s_b: the base score, s_c: the distribution score.

Greedy (fast) alignment

- **The X-drop algorithm: one for greedy ungapped/gapped alignment. Lengthening one edge of a seed, it records a maximum score m. If a score is equal to/below m - x (x is a beforehand determined value), the lengthening is stopped & reverts to one when the latest m is observed.**

- If we set a larger value to x, we can include regions with high scores even if ones with low scores are interposed between them.

- Large x prones redundant searches against short sequences.

- After alignment, it calculates expectations for the alignments as well as the seeds then discards ones above certain thresholds. (However, expectations for the matches of bases/the distributions are separately calculated.)

Alignment expectation

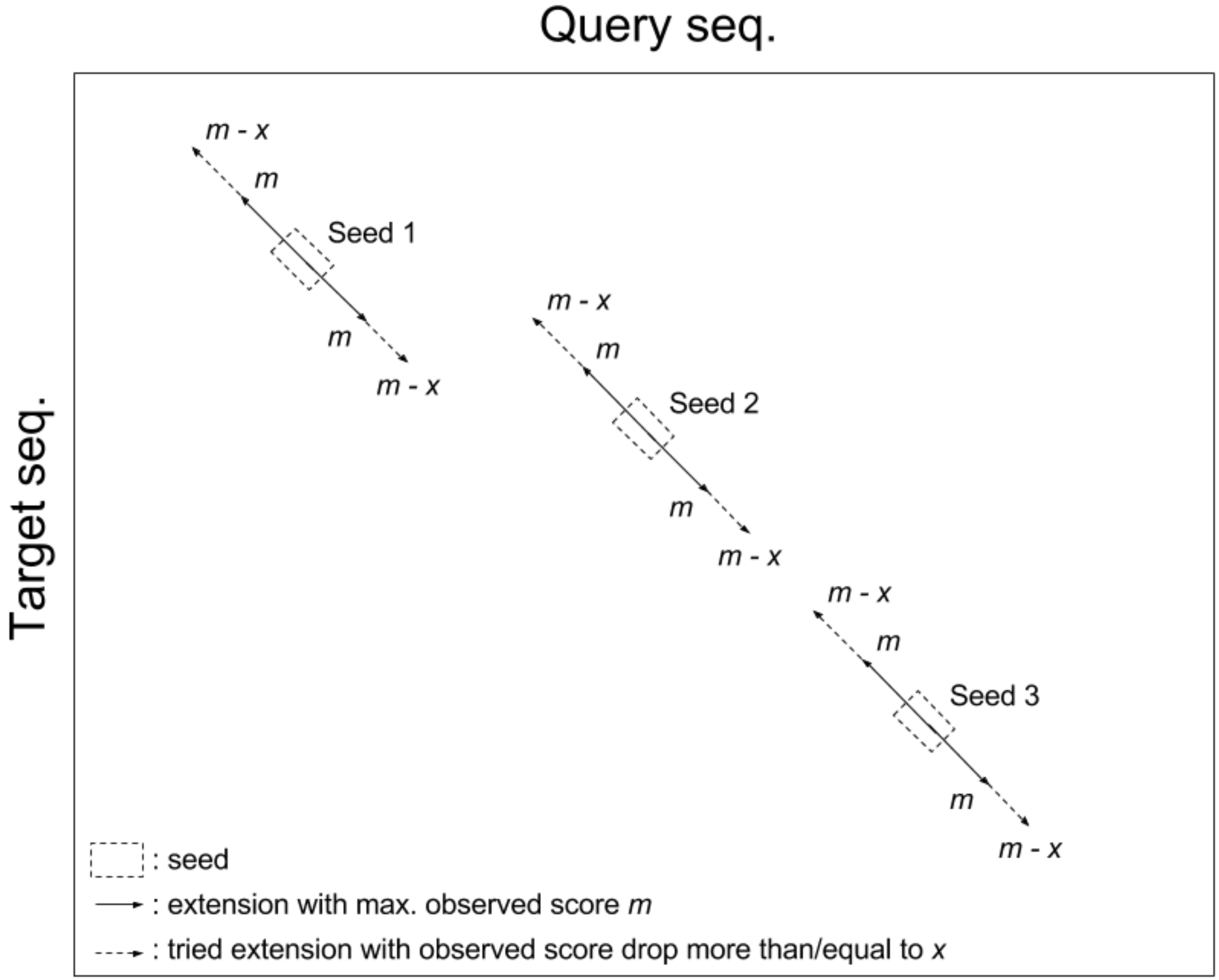
- **We can’t calculate the expectations considering gaps as well as the ungapped one because positions where gaps are observed = ones where indels are done.**

- In this case, considering a distribution of alignment scores is usual; the distribution is the Gumbel one.

- However, the Gumbel one has 2 parameters. To estimate them, alignment of random sequences is required.

- If we estimated these parameters, ranges of available CRAST parameters are limited & the estimation isn’t necessarily possible.

- Hence we let gaps given (because of its uncertainty), then calculate the expectations as well as the ungapped one.


Query seq.
Target seq.
m - x
m
Seed 1
m
m - x
m - x
m
Seed 2
m
m - x
m - x
m
Seed 3
m
m - x
: seed
: extension with max. observed score m
: tried extension with observed score drop more than/equal to x

X-drop algorithm overview

If the score is x less than the maximum score, the lengthening is finished.

Final gapped alignment

- **The constrained Smith Waterman algorithm: one for solving a DP table within ranges constrained by gapped ones.**

- The more gapped ones are found, the smaller areas of the table to solve becomes from mn.

- In case when a pair of alignments overlap with each other, the removal of an alignment with lower score is needed.

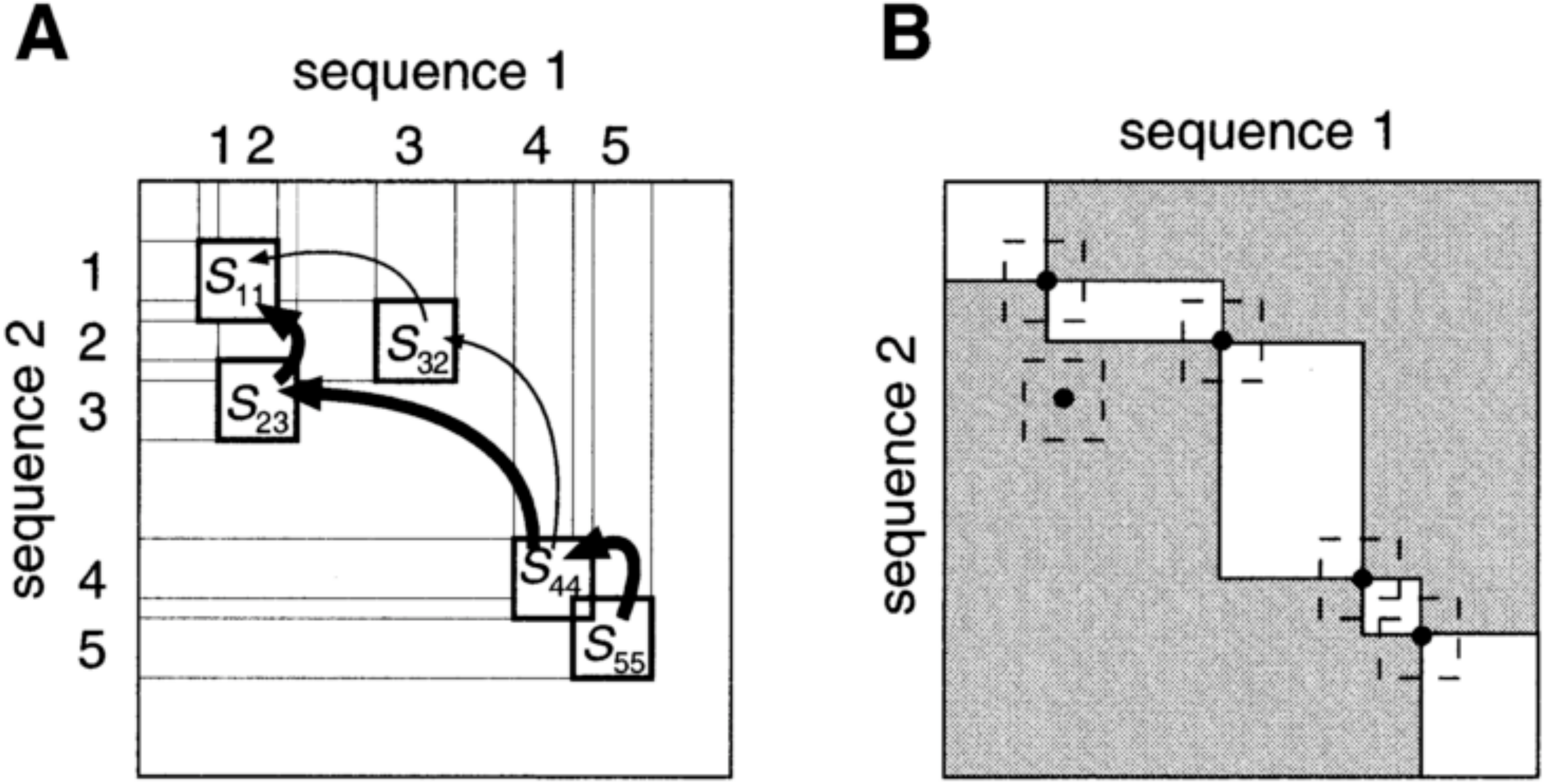


Figure 2. (**A**) An example of the segment-level DP; (**B**) Reducing the area for DP on a homology matrix.

Constrained SW algorithm overview

The strategy by MAFFT. (The bold arrow = if S_{23} > S_{32}, S_{32} is discarded.)
The shadowed portions aren’t solved. CRAST discards one of them as well.

Comparison with other BLAST-like tools

- Sequences to use for the comparison are 34 human lncRNAs having homologs in house mouse (e.g., HOTAIR & Xist) & all 18,185 house mouse ncRNAs.

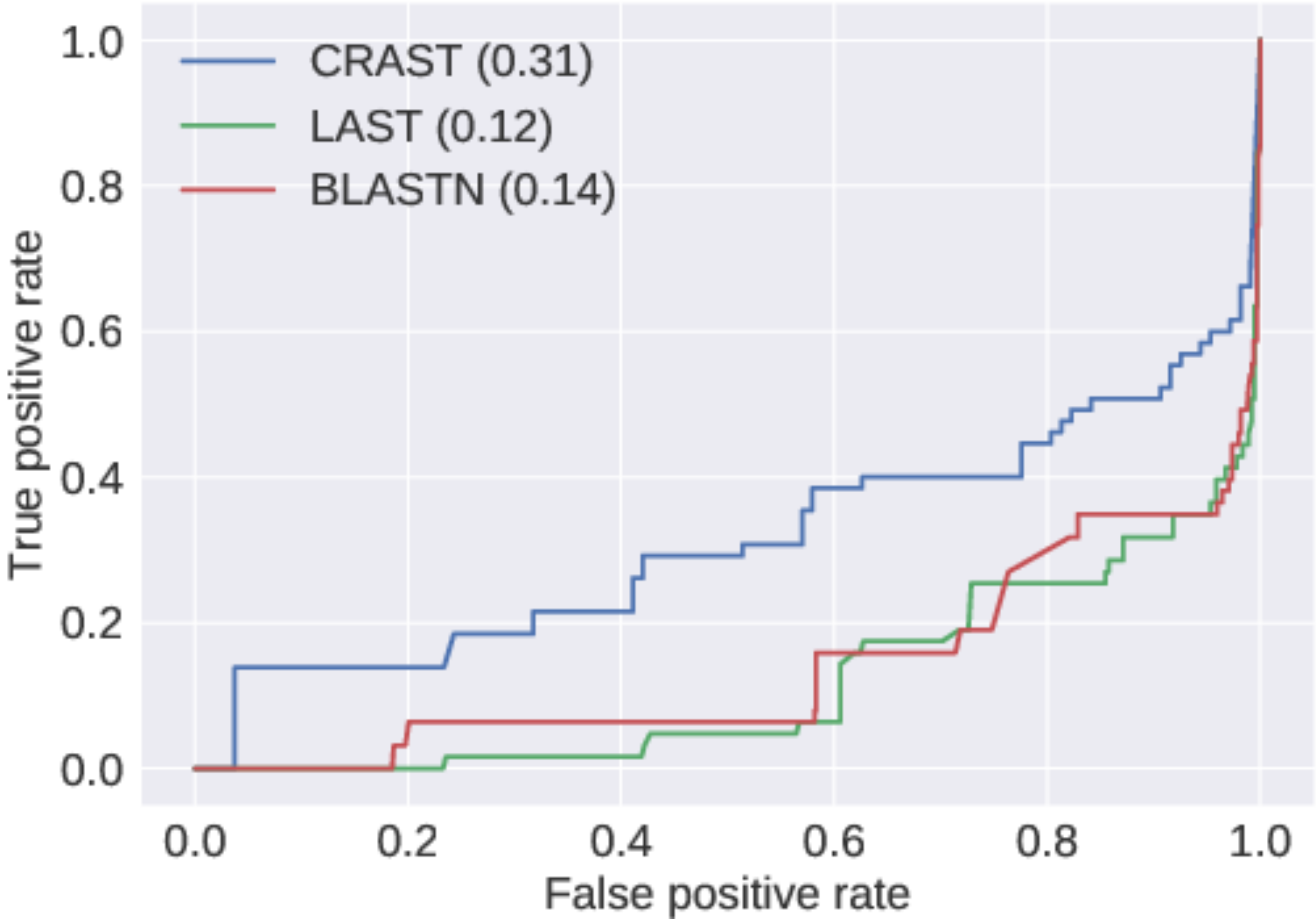
- We've set the human lncRNAs dinucleotide-shuffled with UShuffle to a negative dataset. (Dinucleotide-shuffle = shuffling a sequence preserving its 2-mer frequencies.)

- The TP = a map of any of the lncRNAs to any corresponding homolog, the map to the other = the FP, the TN = a map of any of the shuffled ones to others than the homologs, the FN = the map to the homolog.

Comparison with other BLAST-like tools

	TPs	FPs	TNs	FNs	F-meas.	DB time	Align. time
CRAST	65	107	0	0	0.548	189.5[m]	34.60[s]
LAST	63	365	0	0	0.256	7.246[s]	0.195[s]
BLASTN	63	623	20	0	0.168	1.646[s]	1.007[s]

Comparison with other BLAST-like tools



Comparison with other BLAST-like tools

- The factor why the DB generation of all house mouse ncRNAs in CRAST is slow = **$O(w^2n)$ of CapR dominates a whole of the time complexity**.

- The alignment of CRAST is relatively slow in spite of the reduced seeds due to:

  - in a range of frequently-used frequencies of the seeds, **the seed candidates are NOT filtered in by the binary-search in suffix arrays of database sequences** (if we did it, we didn't need the binary-search anymore, then performance measurement of the algorithm isn't established)

  - **the Jensen-Shannon distance is slow due to the involved logarithms** (as the rescue, we reduces it by its approximation).

Supplements

- The relationships between CRAST parameters & its homolog detectability are noted in the thesis.

- We've also noted a comparison between CRAST & Foldalign in it, however, **the Foldalign detectability is less than any of the BLAST-like tools**. (The TPs are few (32) & the FPs are so many (1,923). The consideration & verification are in it.)

- We implement it in Rust, not C/C++. The reasons are:

  - the thread-safety (guarantee of no data races)

  - the zero-cost abstraction (minimum required amounts of runtime & memory to add/use a language function, e.g., not using garbage-collection for heap management; this abstraction is achieved by C/C++ and Rust)

  - the data is basically immutable & the compiler is strict on type checking, then runtime errors troublesome for human can be blocked/reduced.

Conclusion

- Ask me via email if you have anything you can’t understand in this document/the thesis.
- If you had any bug/lack of function, the remedies are:
  - Issuing a pull-request on Github (fork the repository -> generate a branch for edit -> add a change you’d like in it -> send the request)
  - reporting it at “issues” in Github.