CRAST Document

Heartsh (email: heartsh@heartsh.io)

Homologous-ncRNA search in genomic scale

- 二次構造を考慮してncRNAのホモログを探索するのは, 時間$O(n^4)$以上, メモリ$O(n^3)$以上の計算量を要する.

- 例えば, ヒトのncRNA数本をマウスの全ncRNA(約18,000)に対して探索するのはスパコンを使わない限り無理.

- 重い計算量の原因はアラインメント$O(n^2)$とRNAの折りたたみ(=folding)$O(n^2)$ ≦を同時に解いていること.

- **2次構造を明示的に考慮せず, 確率的(暗示的)に考慮すれば, BLAST-likeに探索が行えるのでは?**

アラインメント with folding

- **Sankoff algorithm: ncRNAのfoldingとアラインメントを同時に解くアルゴリズム. 時間計算量O(n^6), 空間計算量O(n^4).**

- Sankoff alg.の厳密計算は, 現代のPCでは非実用的.

- Foldalign(枝切り)やbanded Sankoff alg.といったヒューリスティックスがあるが, それでも時間O(n^4), メモリO(n^3)以上.

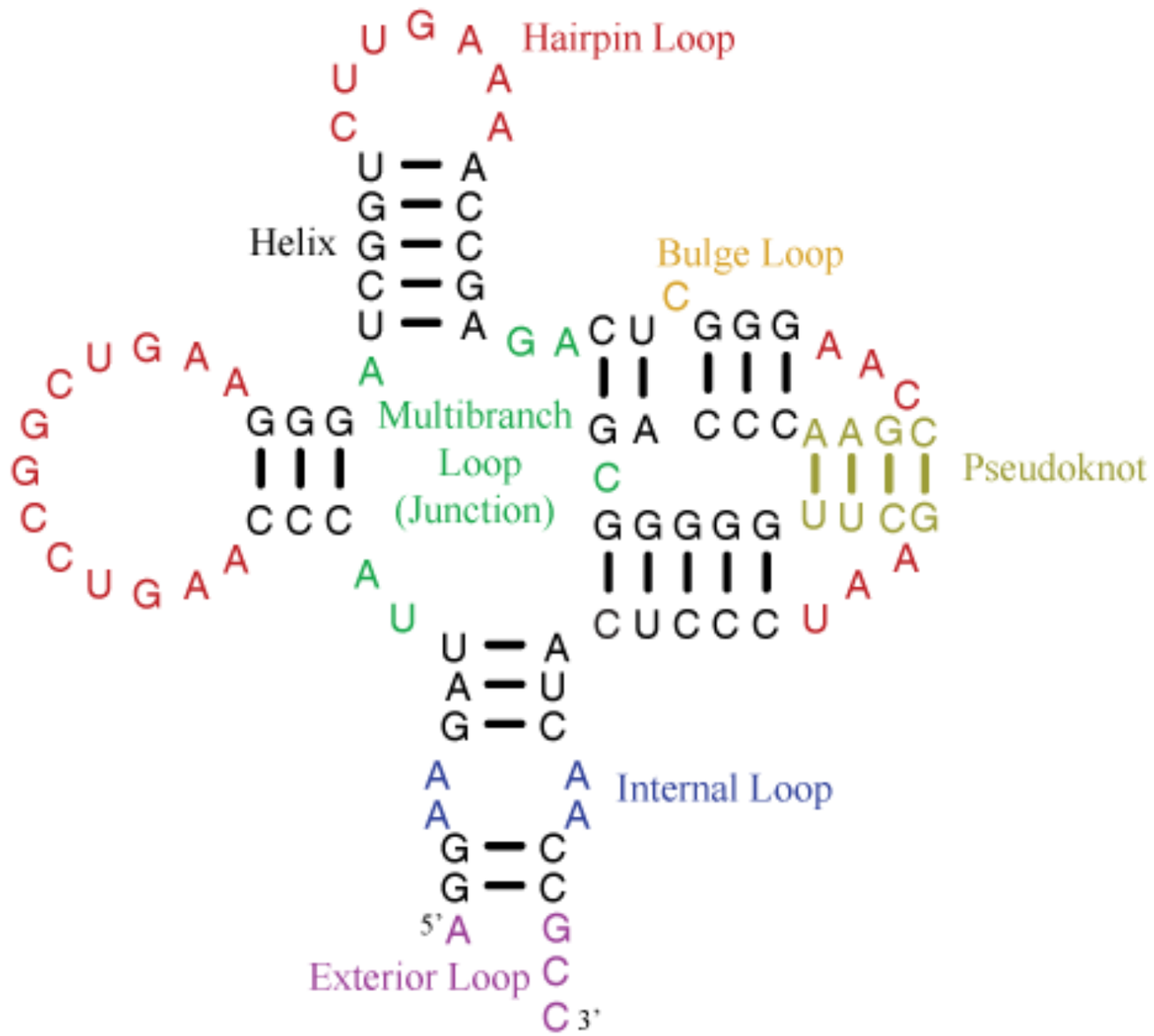
NcRNA context確率分布

- foldingでRNA2次構造を明示的に考慮する限り, 計算量はヒューリスティックス以下にならないのでは?

- CapRやRNAplfoldは, ncRNAの各塩基が2次構造モチーフ(e.g., stem)を取る確率の分布(context確率分布)を$O(w^2n$, w: 最大塩基対幅)で計算.

- wを定数(e.g. w = 200)と考えれば, O(n)とみなせる.

- **ncRNAの2次構造を確率分布列に変換し, BLAST-likeなスコアリングシステムに組み込めば, ヒューリスティックス以下になるのでは?**



RNA motif

Pseudoknotは他のモチーフと異なり入れ子になった(図だと2つのhelixに挟まれてる)モチーフなので, 通常計算対象にはならない.

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
1.3906044868999896e-6 1.9981855142744496e-5 2.8388721816397277e-3 5.45600373360367e-1 2.3138545091421782e-5 4.5151624345327224e-1
1.8339423130955246e-3 8.999757606189002e-2 2.259825700825085e-1 5.4552710162499332e-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 3.5666533349273520e-1 3.666032248329658e-1 7.052036392842810e-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 1.2752406702681860e-1 2.4756238310085350e-3 3.2188037419313020e-1 7.271075656924660e-4 4.0370169777460135e-1
1.0964619582691038e-3 2.1148131974406880e-2 1.0886427252127962e-3 1.0521566517006265e-2 1.7665966973523612e-4 9.659685371553697e-1
5.020632417878390e-5 2.5449810211627490e-2 2.5284887851736585e-3 6.9890262004828600e-2 2.7039053500644303e-4 9.0181084213918500e-1
9.6756317512109570e-3 1.9057023880860896e-1 2.3984285745263373e-3 3.0760081217857815e-1 3.4632555023652814e-4 4.8940856313683900e-1
9.6724031328965160e-3 6.7856013110011320e-1 2.4285835158615846e-3 3.0830012632389564e-1 6.0958085612259650e-4 4.2917507111052940e-4
9.6802961661768190e-3 6.7858743241946880e-1 2.4464976642495650e-3 3.0828638053268060e-1 6.2998670390159960e-4 3.6714899293508220e-4
9.6693957124660350e-3 6.7845823142607410e-1 1.9672440717060200e-4 2.9581155153422010e-1 6.5082933027197110e-4 1.5213267589797155e-2
4.6695218712508674e-5 8.9298304143155040e-4 1.5425998939177892e-4 5.4033629524856440e-3 9.0373242581522490e-5 9.9341232555539690e-1
7.6284465019458085e-6 7.4572919544357980e-4 1.5435331603843067e-4 2.3350025716737323e-3 1.6631514285967434e-4 9.9659097132748260e-1
3.6187411311869795e-5 1.2078390363978599e-4 2.2008130068917870e-4 1.9747313409796100e-3 6.6931792008111590e-5 9.9758128425137150e-1
1.9645901344124386e-3 6.0948699808045290e-3 7.2869379225487150e-4 2.2252281060522374e-3 6.2373559274816410e-4 9.8836288239372780e-1
2.4203461778962907e-2 7.5579506500495290e-2 2.4784893166297194e-3 2.2238646621217740e-3 6.0298041430391370e-4 8.9491169732748630e-1
1.3343076034815088e-1 8.2462349964063710e-1 8.3583742548395330e-3 2.2903286052464752e-3 2.5442549695954136e-3 2.8752782181530677e-2
1.4697000386941508e-1 8.2754506533349290e-1 1.9055996326072654e-2 2.3082509008732150e-3 1.6755831616572980e-3 2.4451004084888534e-3
1.4695888560934073e-1 8.2093873144952150e-1 2.1044163382302566e-2 2.3101332733047684e-3 1.3448732606003020e-3 7.4032130249302390e-3
1.1439145889733626e-1 2.4768217919610615e-2 2.3621013626473546e-2 2.3052811433460453e-3 5.6443152538687290e-4 8.3434959688784670e-1
9.7061820364023110e-2 1.2894408649873795e-2 2.2800635356888664e-2 2.3022365181200350e-3 5.1913567271883730e-4 8.6442176343837550e-1
9.7704029593819140e-2 1.3338943271713330e-2 9.9177060698636450e-2 2.5403344748062017e-3 1.0827156956379402e-3 7.8615691626538700e-1
7.3456555886491750e-2 5.0427138554779800e-3 4.2582299988384270e-1 1.1355407587946655e-3 9.6015304254573580e-4 4.9358203657284727e-1
1.2734495714902170e-3 3.2197010451717723e-3 8.8016218065282540e-1 3.1359900595664530e-4 5.9045521868998480e-4 1.1458671912020717e-1
7.8887800023866580e-4 6.6477899231658050e-3 8.8094508518033130e-1 1.3255832158144438e-4 5.9198856178123420e-4 1.1089370001290152e-1
8.2885356852550190e-4 1.1627410419504235e-1 8.8151986080271650e-1 1.8149818229461835e-4 7.3857533971558320e-4 4.5710791170547793e-4
8.8796753345422610e-4 1.1638294799221381e-1 4.2624895061662915e-1 1.9301018162319345e-4 9.4149949994297970e-4 4.5534562417613660e-1
```

NcRNA context確率分布列の例

左からbulge, internal, hairpin, exterior, multi-branch loop, helixの確率.

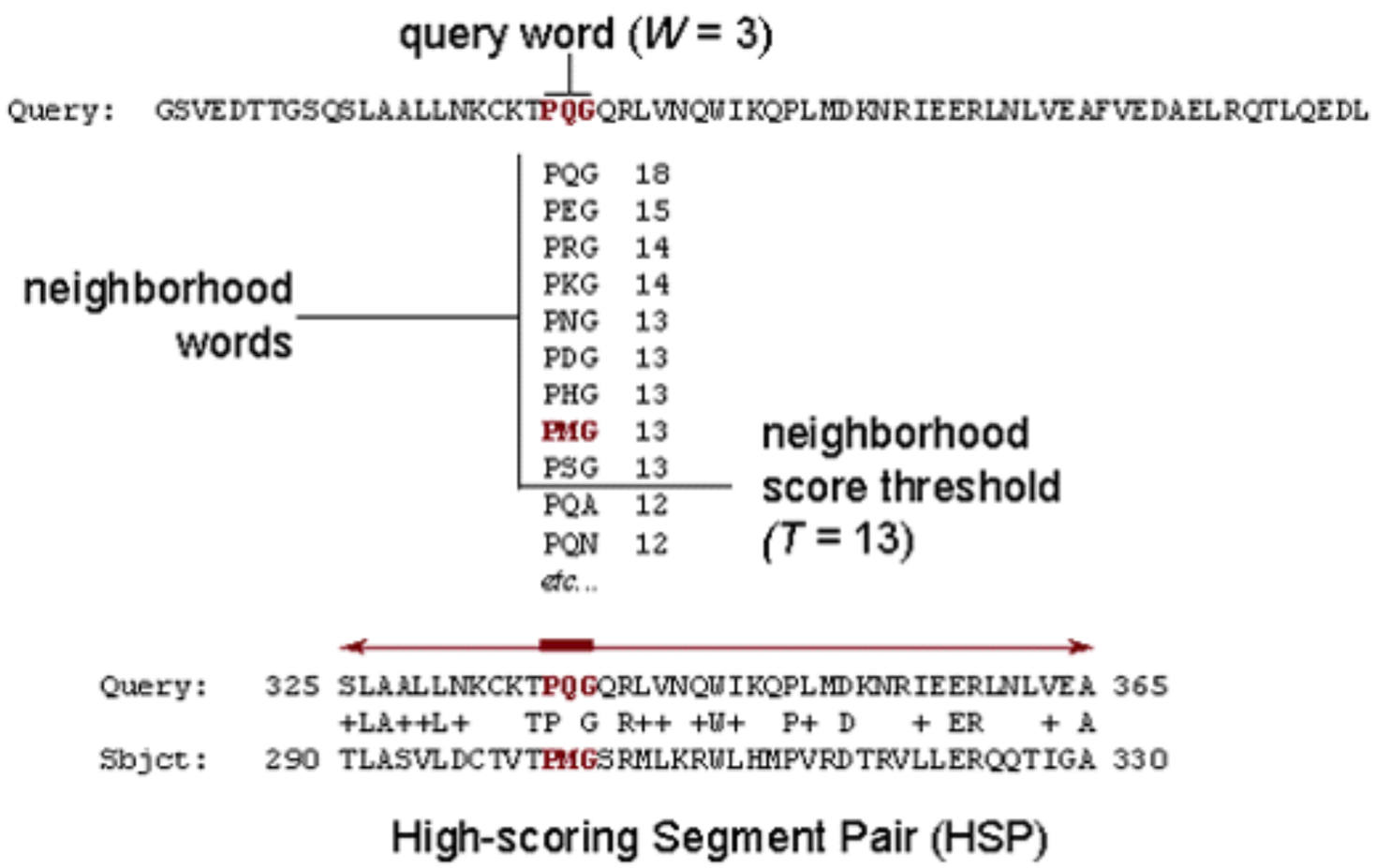
e-x = 10の-x乗で, 画像はヒトlncRNAをCapRにかけたもの.

BLAST

- BLAST: 2配列間のアラインメントをO(mn)で計算する Smith Waterman alg.のヒューリスティックス.

- **BLASTは, 条件をクリアするseed(完全一致する短い配列)とギャップ無し/有りアラインメントを持つ2配列間のみでO(mn)のギャップありアラインメントを解く.**

- seed発見には, 時間計算量O(log(n))のsuffix arrayの二分探索とO(1)のハッシュマップを用いる.



BLAST1アルゴリズムの概観

BLAST2ではギャップ無し/有りアラインメントの後に, 動的計画法(DP)を用いて
ギャップ有りアラインメントを行う. (アラインメントを1つだけにするため.)

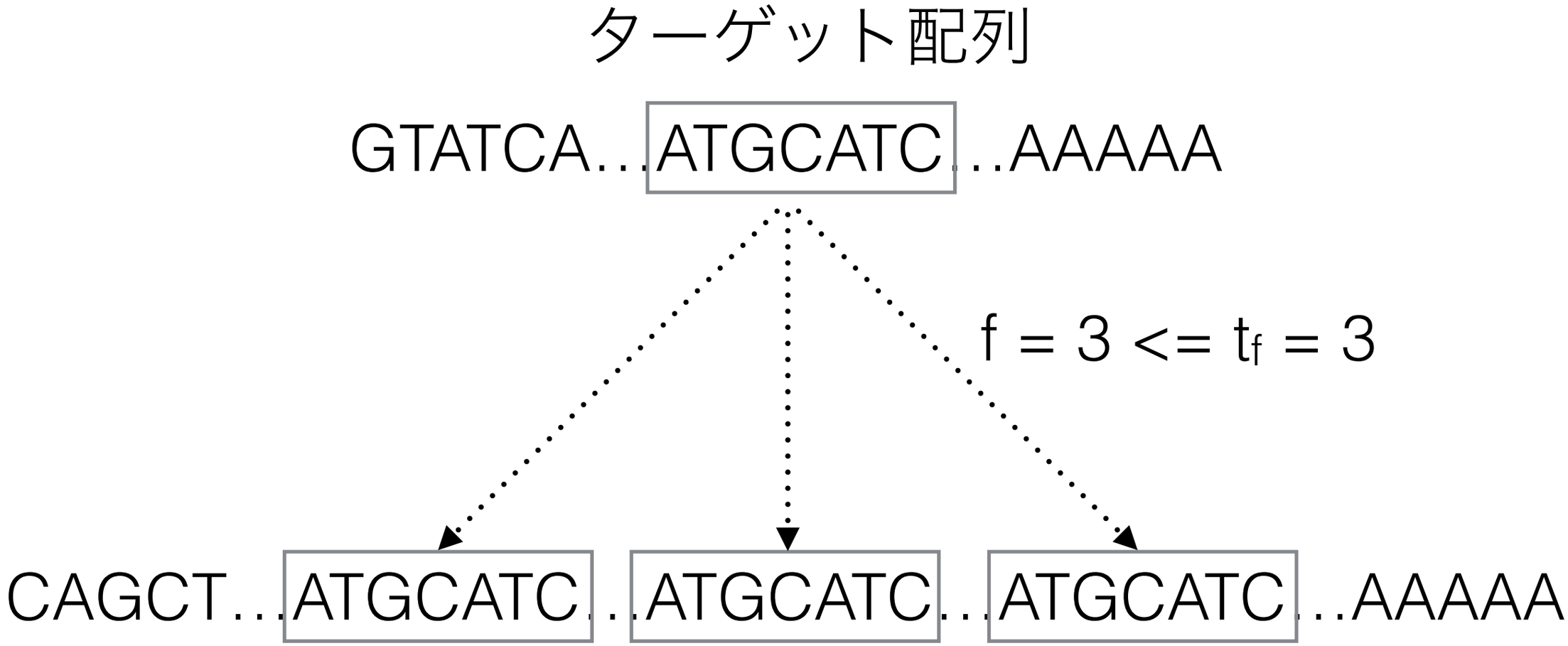
LAST

- LAST: BLASTの短い固定長の配列によるseed発見を短い配列のref. seq.内での頻度(e.g., 頻度10以下)に置き換えたもの.

- **seedがref. seq.の長さに対して線形に増える(BLASTは2乗)ため, O(n)になる.**

- BLASTで実際に観測されるseedの数は, seed長から期待されるものよりかなり大きくなる. (sensitivityが下がる.)

- BLASTのseedの増大は, 現実の生物配列内での塩基分布が一様分布(e.g., A:T:C:G = 1:1:1:1)から離れているため.



LASTのseed発見の概観

reference配列内でレアな短い配列をseedにする.

同じ閾値を用いても, ゲノムとスクリプトではseedの長さは異なる.

Context RNA Alignment Search Tool

- CRASTというLASTベースのアラインメントツールを作成.

- **マウスにホモログを持つヒトlncRNA34本とマウスのncRNA全て(18,185本)を用いて実験し, TPを2つ増やし, FPを1/3以下に減らすことを確認した.**

- **LASTのseed発見の条件に, seed間のcontext確率分布列の類似度の条件を加え, 更にseedを減らした.**

- 確率分布の類似度を測る = 確率分布の距離を測る.

- 確率分布p, qのJensen-Shannon distanceを距離に使い, $d(p, q) < d_t$となる場合をマッチとしてスコア = +1, それ以外をミスマッチとしてスコア = -1.

- 加えた条件は2項分布とスコアを元にしたseed数の期待値の閾値.

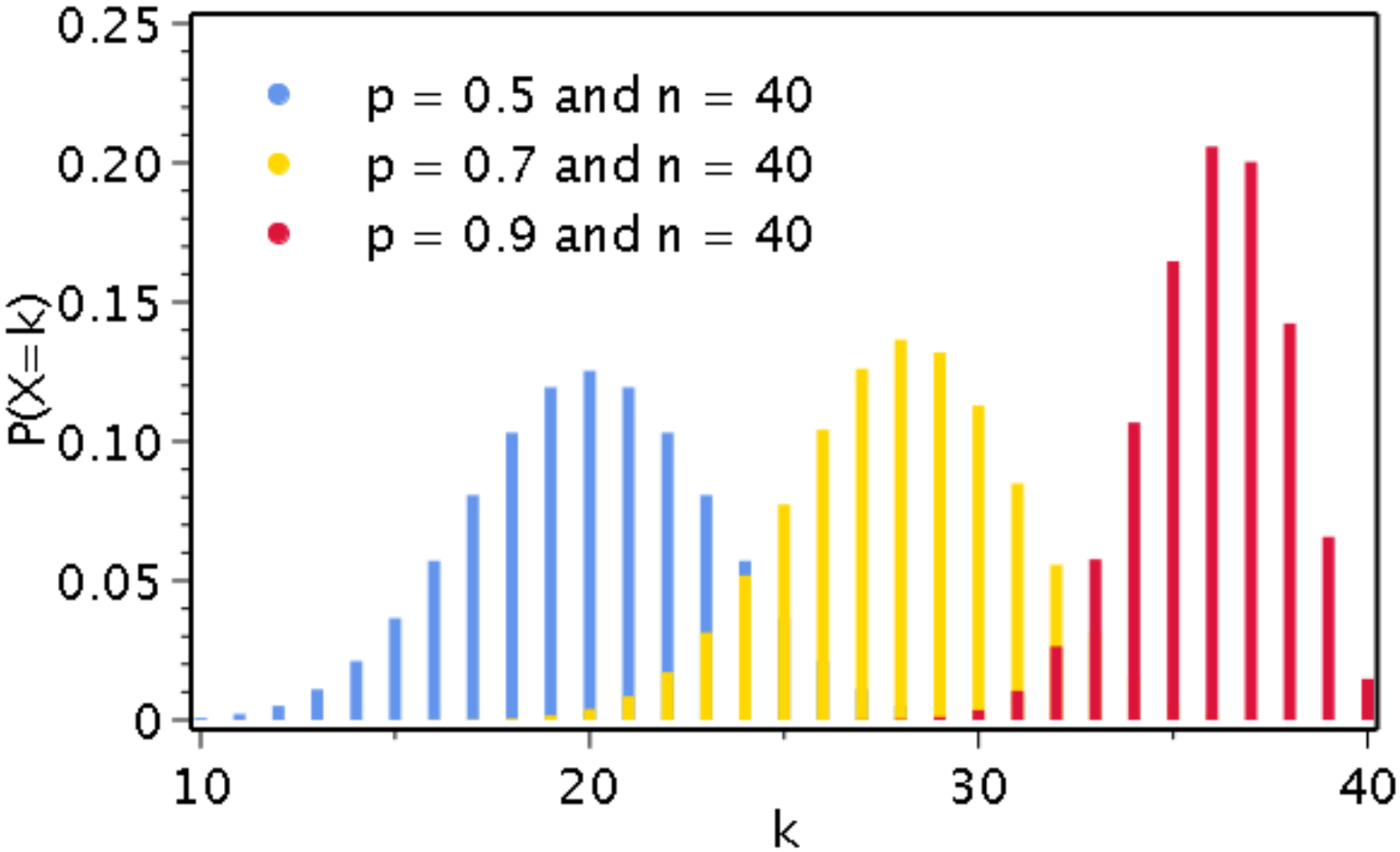
2項分布

- 2項分布: コインの表/裏のような2つの状態を, ある試行回数内で観測する回数(e.g., 表が出る確率p = 0.25のコインを5回投げて3回表)をモデル化する確率分布.

- context確率分布列のマッチ/ミスマッチもこれに従う. (分布のマッチ確率0 < p (= d_t) < 0.5.)

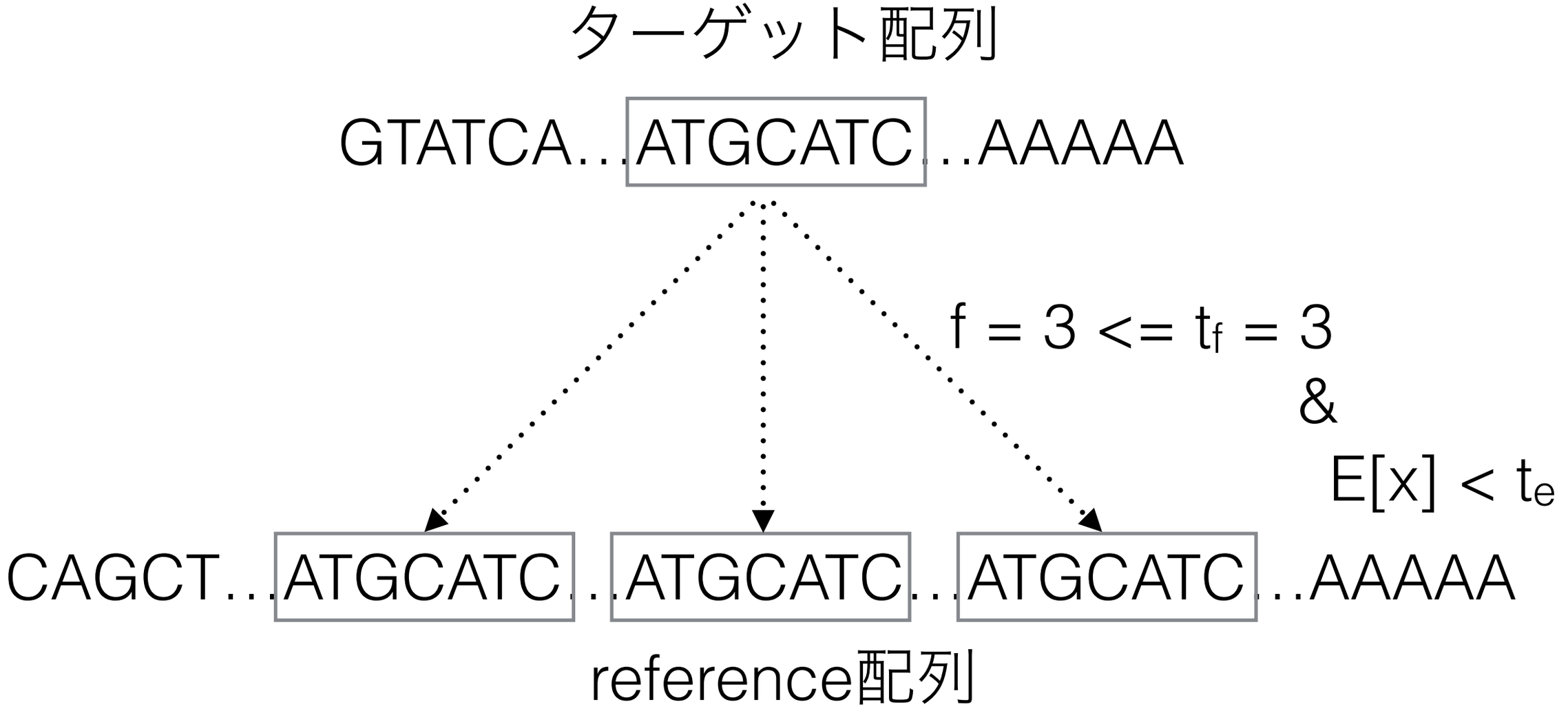
- **分布のマッチ回数がn以上となる長さNのseed数xの期待値は, E[xl n, N] = (target seq. len. - N + 1) * (1 - P(x <= n)).**

- p < 0.5 としているのは, スコアの期待値を0より小さくし, significantなアラインメントとそれ以外を区別できるようにするため. (期待値0以上の場合, ランダム(相関なし)の配列でもアラインメントを作れる.)



2項分布の例

pが0.5から離れるほど, 平均はシフトし, 分散は小さくなる.



CRASTのseed発見の概観

LASTのseed発見の条件に, context確率分布列の相似度を元にした期待値の条件t_eを追加.

スコアリングシステム

- 塩基のマッチ/ミスマッチは, スコア+1/-1, ギャップオープン/エクステンションは, コスト-7/-1. (比較のためLASTのデフォルトと同じ.)

- **塩基とcontext確率分布のスコアを組み合わせたスコアをアラインメントに用いるスコアとする.**

- $s = rs_b + (1 - r)s_c$, $0 \leqq r \leqq 1$: 塩基の寄与率, s_b: 塩基のスコア, s_c: 分布のスコア.

貪欲な(速い)アラインメント

- **X-drop alg.: 貪欲的にギャップ無し/ありアラインメントを行うアルゴリズム. seedの一端を伸長しながら最大スコアを記録, スコアが最大スコアよりx下回れば伸長停止し, 最大スコアとなった伸長に戻る.**

- xの値を大きく取れば, スコアの悪い領域が高い領域に挟まれていても高い領域を含むことができる.

- 短い配列に対しては, 大きいxは無駄な探索を行いやすい.

- アラインメント後にseed同様に期待値を計算, 閾値以上のものを捨てる. (但し, 塩基とcontext確率分布の一致それぞれ別々に期待値を計算.)

アラインメントの期待値

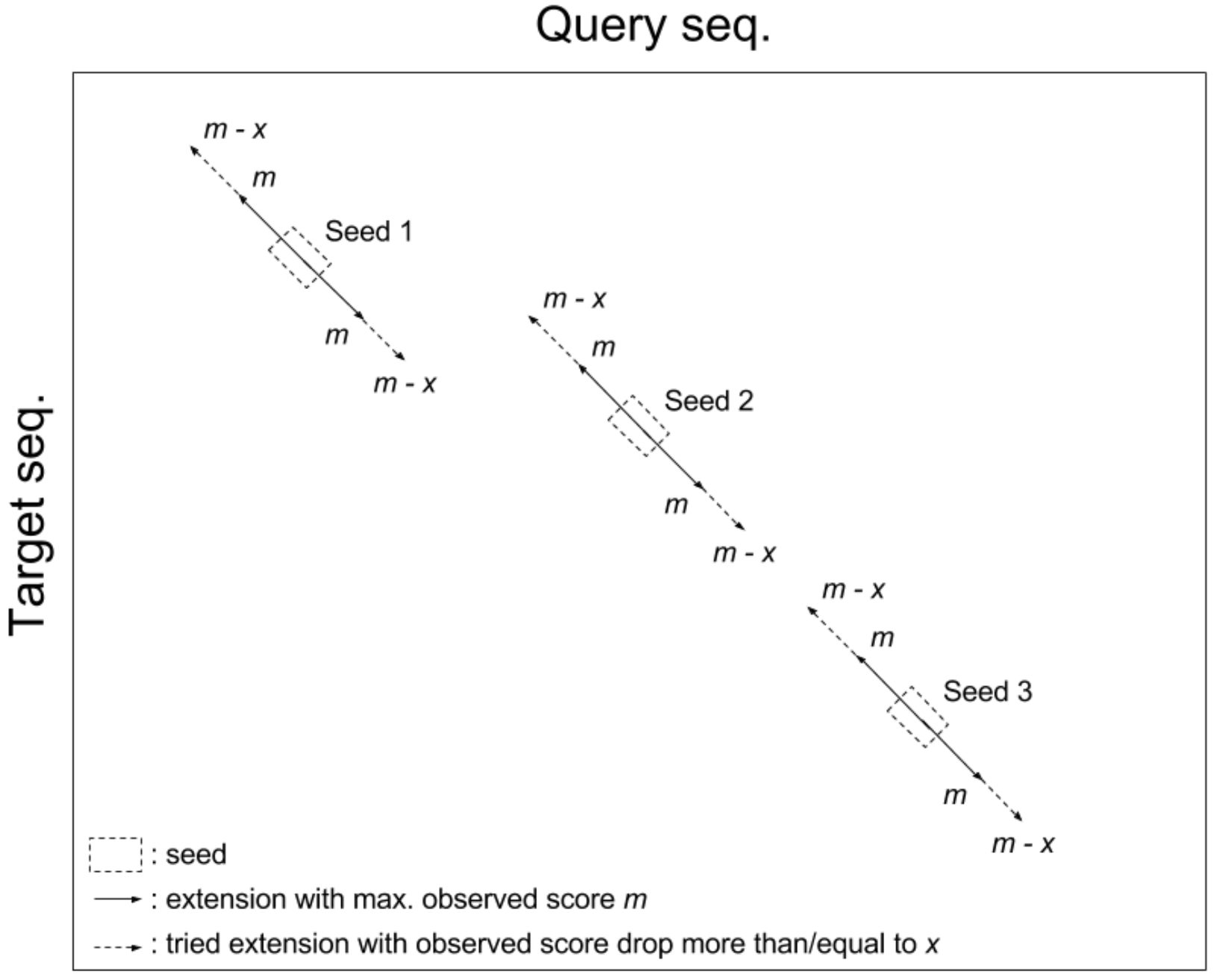
- **ギャップ部分は挿入/欠失が起こった部分であるため, ギャップを考慮すると**
  ギャップ無しアラインメントと同じく二項分布で期待値が計算できない.

- この場合, スコアの分布を考えるのが一般的で, それに用いる分布がGumbel分布.

- 但し, Gumbel分布にはパラメータが2つあり, これを推定するためにランダム配列(e.g., シャッフル配列)を用いてアラインメントを行う必要がある.

- このパラメータ推定をしてしまうと, ユーザが使えるパラメータの範囲(e.g., 塩基の寄与率rは0.0, 0.1, …, 1.0のみ)が限られ, 推定できるとも限らない.

- よって, ギャップ部分はgivenである(未知だから無視する)とし, ギャップ無しアラインメントと同様に期待値を求める.



X-drop alg.の概観

最大スコアよりX下回れば伸長を終了, 最大スコアの伸長に戻る.

スコアの期待値が0以上だと無相関でも伸ばせることが分かる.

ギャップありアラインメント

- **Constrained Smith Waterman alg.: DPのテーブルをギャップありアラインメントで制限された範囲で解く.**

- ギャップありアラインメントがたくさん見つかれば, その分テーブルの解く範囲はmnから小さくなる.

- ギャップありアラインメント同士がオーバラップする場合, スコアが低いアラインメントの除去を行う必要がある.

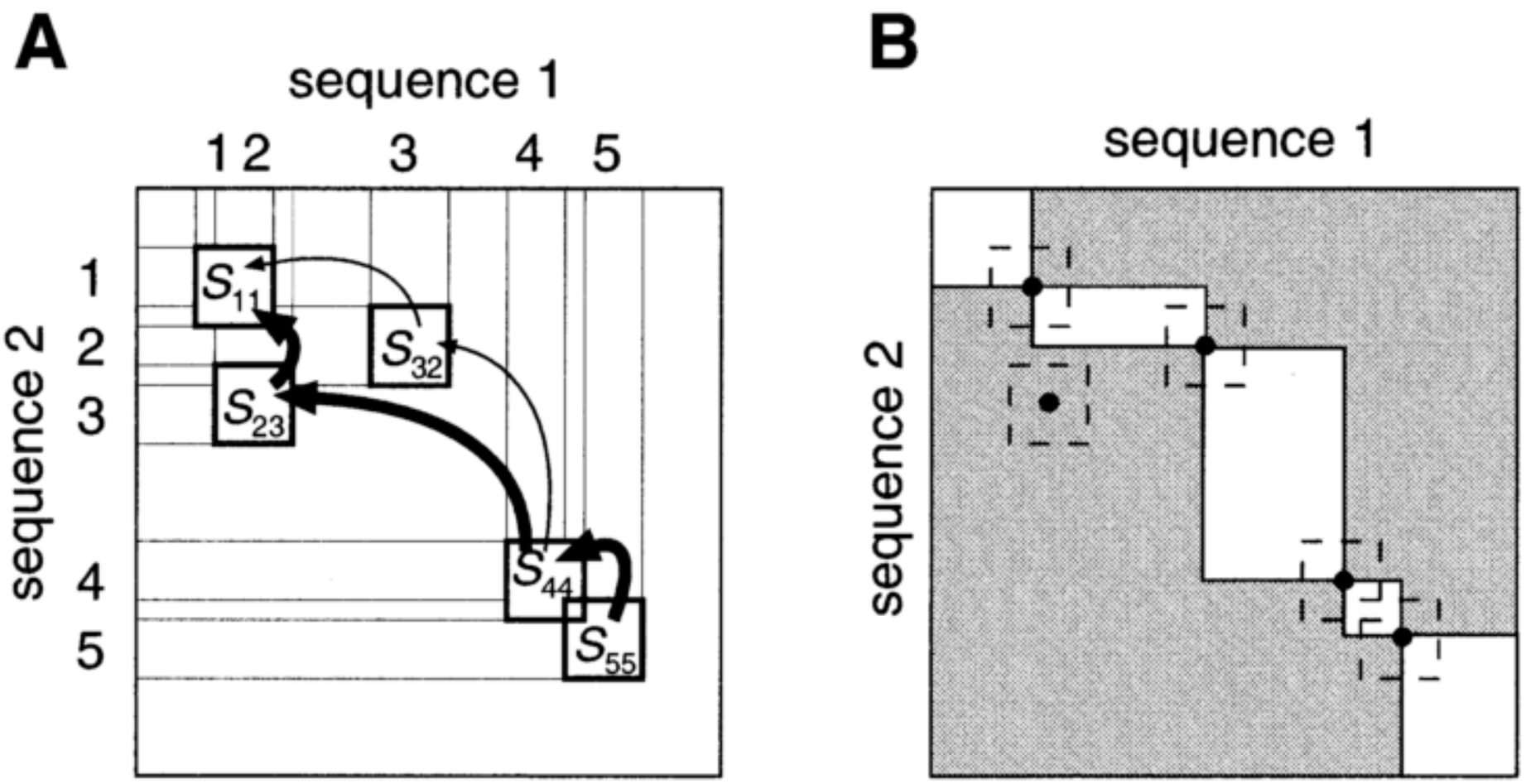


Figure 2. (**A**) An example of the segment-level DP; (**B**) Reducing the area for DP on a homology matrix.

Constrained SW alg.の概観

MAFFTのDPテーブルの解き方. (太矢印は$S_{23} > S_{32}$の場合, S_{32}を捨てるということ.)
陰影部分はDPを解かない. CRASTも同じようにオーバラップするどちらかを捨てる.

BLAST系ツールとの比較実験

- 実験に用いるのは, マウスにホモログを持つヒトlncRNA34本(e.g., MALAT-1やXist)とマウスのncRNA全て.

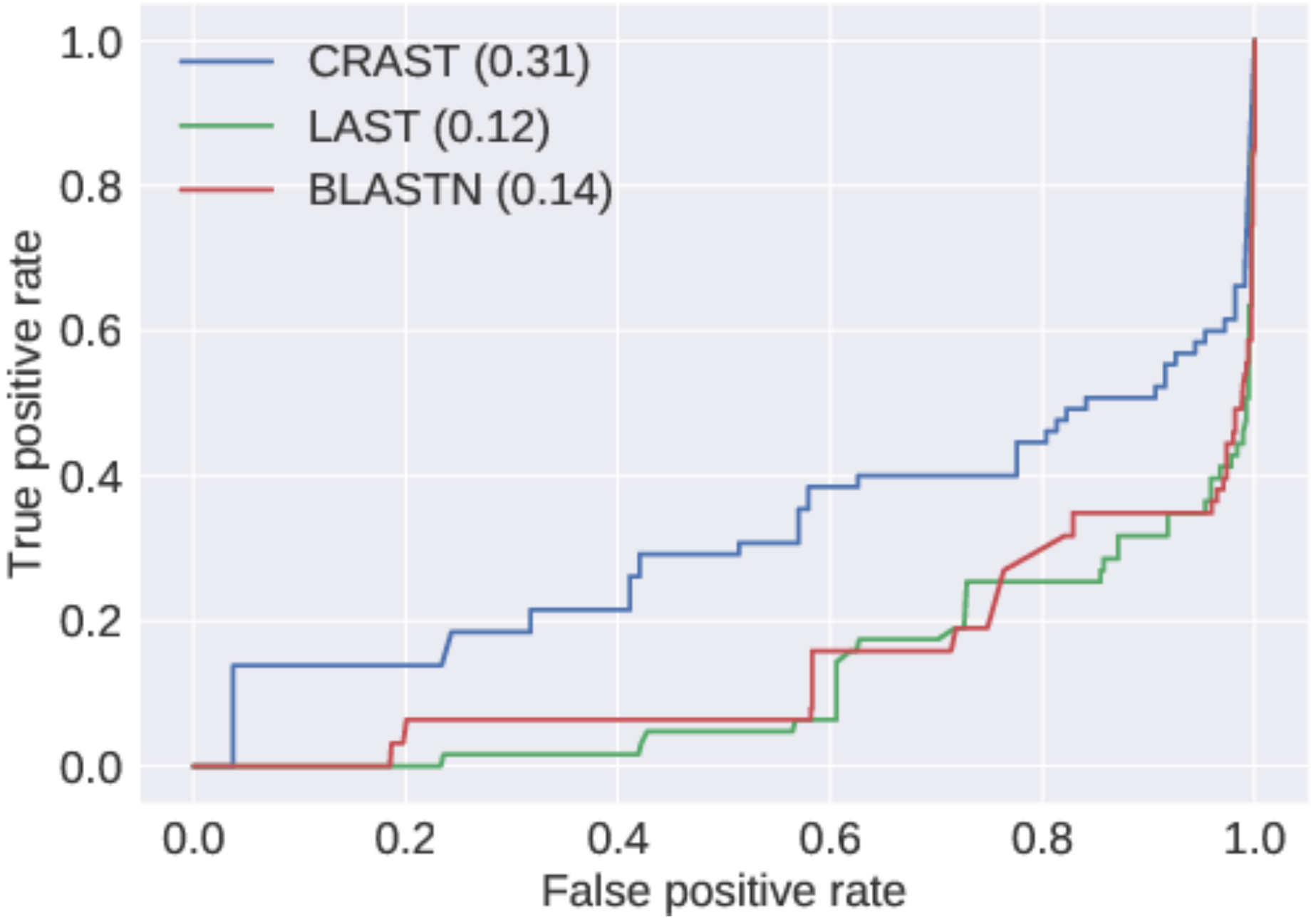
- ヒトのlncRNAをuShuffleでdinucleotide shuffleしたものをネガティブデータとした. (dinucleotide shuffle = 2-merの頻度を保ったまま配列をシャッフルすること.)

- lncRNAが対応するマウスのホモログにマップされればTP, それ以外にされればFP, シャッフルしたlncRNAがホモログ以外にマップされればTN, ホモログにマップされればFN.

BLAST系ツールとの比較実験

	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]

BLAST系ツールとの比較実験



BLAST系ツールとの比較実験

- CRASTでマウスのncRNA全てのDBを作るのが遅いのは, **CapRのO(w^2n)が時間計算量全体を支配してる**から.

- CRASTのアラインメントがseedを減らしているのにも関わらず遅いのは:
  - seedの頻度の大体使われるであろう範囲(e.g., 1~10)で**事前にDB配列のsuffix arrayを2分探索することで, seedの候補を絞ってないこと**(これをすると遅い2分探索をDBに対してしなくてよくなり, アルゴリズムのパフォーマンスの計測にならない)

- **Jensen-Shannon distanceがlogの計算を含んでいて遅い**こと(近似logを使って軽減はしている).

補足

- CRASTのパラメータとホモログ検出能の関係は論文に書いてある.

- 論文にCRASTとFoldalignの比較も載せたが, 予想に反して**Foldalignのホモログ検出能はどのBLAST系ツールより低かった**. (TPが32と少なく, FPが1,923とかなり多い, 考察と検証は論文に.)

- C/C++ではなくRustで実装, Rustを選んだ理由(メリット)は:

    - スレッドセーフ(並列実行してもデータ競合が無いことを保証すること)

    - ゼロコストアブストラクション(言語機能を追加/使用するのに要する実行時間やメモリの大きさが最小なこと, ヒープ管理にガベコレを使わないとか)

    - 基本的にデータがimmutableで, コンパイラの型チェックが厳しいので, 人間では発見するのが難しいランタイムエラー(セグフォ, 副作用が元になった状態依存のバグ)を弾ける/減らせる事.

最後に

- 論文とこのドキュメントを読んでも分からない所はメールで聞いて下さい.

- バグ/機能の不足があれば改善方法は2通り:
  - Github上でプルリクエスト(リポジトリをフォーク -> 編集用のブランチを生成 -> そのブランチで変更を入れる -> リクエストを送る)
  - Github上の”issues”で報告.