Blastn's seed length

- Recall: blastn's seed match is of length $w = 11, 12$
  - *exact* match
  - $w > 10$ is compatible with the packing speedup
  - a seed match is extended to a gapless alignment
- What is the significance of w?

w controls the sensitivity

- The *sensitivity* of the seed is the precentage or "real alignments" discovered
  - The real alignments/similarities can come from a db of alignments
  - or from a model
- We shall assume that the gapless extension never fails so w essentially determines the sensitivity
- As w decreases the sensitivity increases
  - as it is more likely that an aligned pair of sequences would contain a perfect match of length w

w effects the search speed

- Assuming an aggressive search (high sensitivity) the search speed is largely determined by the number of random seed matches
  - with each one triggering an extension attempt
- Let $A_{ij} = A_{ij}(w)$ be the event: a match of length w starts at position i of the first sequence and j of the second
- The expected number of random seed hits is:

$$E_0 \sum_{ij} 1_{A_{ij}} = \sum_{ij} E_0(1_{A_{ij}}) = \sum_{ij} P_0(A_{ij}) \approx mnP_0(A_{ij}) = mn\rho^w \approx \frac{mn}{4^w}$$

- One can prove that $\rho \geq 4$
- Thus, lowering w from 11 to 10 increases the expected number of random matches by a factor of 4 (at least)

PatternHunter - Ma, Tromp, Li (02)

- Human-mouse analysis (Waterstone et al., Nature 2002)
- Ma, Tromp and Li: a seed is a pattern of w matches
- Spaced seeds seem better:
  - for the same weight w the sensitivity can increase
- For example, $\pi = 111\text{-}1$ designed to detect
  - `...ACC?T...`
    `...ACC?T...`

  is "typically" more sensitive than $\pi_c = 1111$ which detects
  - `...ACCT...`
    `...ACCT...`

Why are spaced seeds better?

- Related to a problem studied by John Conway: which word are you more likely to see first in a random text
  - ABC or AAA?
- In any given position what is the probability of seeing ABC?
  - $1/26^3$
  - What about AAA?
- The expected number of letters between consecutive occurrences of ABC is 26^3 (renewal theory)
  - Same for AAA
- Given this symmetry which word would you expect to see first ABC or AAA?
- The correct answer is ABC

Advantage spaced seeds

- Given w the expected number of random seed matches is identical for all seeds of weight w
  - therefore the running time is about the same
- A spaced seed would typically yield better sensitivity than blastn's contiguous w-mer
- Conversely, by choosing an optimal spaced seed of weight $w+1$ we can reduce the random hits (FP) by a factor of 4
  - and attain a sensitivity $\geq$ sensitivity(π_c^w) (blastn's contiguous w-mer)
- Using db of real alignments, Buhler, K and Sun verified that an optimally selected seed of weight 11 is more sensitive than π_c^{10}
- NCBI's BLAST server has over 10^5 queries/day

Evaluating a seed

- A seed's quality: weight vs. sensitivity
- Determine the sensitivity:
  - experimentally: learn the sensitivity from a database of real alignments
    - computationally intensive
  - parametrically: using a model
    - can yield some insight on what makes some seeds better
    - can lead to designing seeds rather than choosing ones

Model of a similarity region

- Our similarity region models a gapless subsection of the alignment:
  - no gaps
  - fixed length, l, shorter than typical alignment region (64)
- Key step: translate the gapless alignment to a single "mismatch string":
  - binary string S, where $S(i) = \begin{cases} 1 & x_i = y_i \\ 0 & x_i \neq y_i \end{cases}$
    - For example,

      `TcgAaTCGtTACt`

      `TatAcTCGgTACa`

      `1001011101110`
- We model S as k-th order Markov chain $(k = 0, 1, \ldots, 6)$
  - for coding region use a 3-periodic transition probabilities

Seed's sensitivity

- A seed is a pattern of 1s, corresponding to positions of identical letters in the matched pair of words
  - for example, $\pi = 111\text{-}1$
- π *detects* S if its patterns of 1s occurs in S
  - For example, the similarity
    - `TcgAaTCGtTACt`
      `TatAcTCGgTACa`
      `1001011101110`
    - is detected by $\pi = 111\text{-}1$ but not by $\pi_c = 1111$
- *Sensitivity*: $P\{\pi \text{ detects } S\}$

Computing the seed's sensitivity

- Simplified case: S is a sequence of iid Bernoulli random variables
  - $p = P(S[i] = 1)$
- Given $l = |S|$ and a seed π compute $P(E)$ where $E = \{\pi \text{ detects } S\}$
- Let $s(\pi)$ be the span of the seed: $w + \#$ don't care positions
  - for $\pi =$ `111-1`, $s(\pi) = 5$
- Let $H_n = H_n(\pi) = \{\pi \text{ occurs at } S[n : n + s - 1]\}$
- Then, $P(E) = P(\cup_{n=1}^{l-s+1} H_n)$
  - Clearly, $P(H_i) = p^w$
  - But the occurrences overlap
    - H_n are not independent

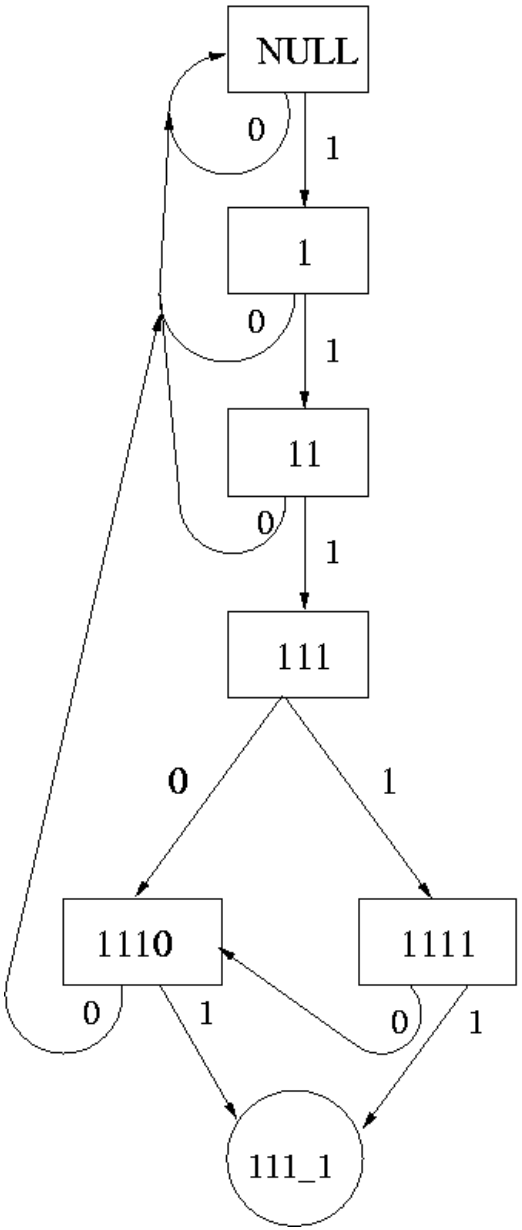
- Inclusion-Exclusion:

$$P(E) = \sum_{n=1}^{l-s+1} P(H_n) - \sum_{i<j} P(H_i \cap H_j) + \ldots$$

Better techniques

- The combinatorics of the inclusion-exclusion formula are quite messy
- Use Guibas-Odlyzko overlap polynomials (1981): $O(ls2^{3(s-w)})$
- Nícodeme, Salvy, and Flajolet (1999): $O(lw2^{s-w})$
  - Construct an automaton that accepts the strings that end with the unique occurrence of π
    - The states are prefixes of π
    - Upon input x transition from state α to β: the longest suffix of αx that is a prefix of π

NSF's automaton for $\pi = 111\text{-}1$



Adding probability to the automaton

- The automaton is ignorant of the probability space
- A naturally associated Markov chain, X, can be defined on the states of the automaton:

$$P_m(\alpha, \beta) = \begin{cases} P_S(x) & \text{there is an edge labeled } x \text{ from } \alpha \text{ to } \beta \\ 0 & \text{otherwise} \end{cases}$$

- By construction the probability of any automaton path starting from $\emptyset$ is the same as the probability of the corresponding substring

Computing the sensitivity from the automaton

- Let T be the accepting state (absorbing, no transitions out)
- Claim: $P_S(E) = P_m(X_l = T | X_1 = \emptyset)$
- Proof.
  - $E = \cup_i E_i$ where the event

$$E_i = \{S : \text{1st occurrence of } \pi \text{ ends with } S(i)\}$$

  - Partition each E_i to equivalence classes of strings according to their prefix of length i
    - each class corresponds to a distinct path of length i from $\emptyset$ to T
    - the probability of the class is identical to that of the path
  - Summing the probabilities of all classes completes the proof

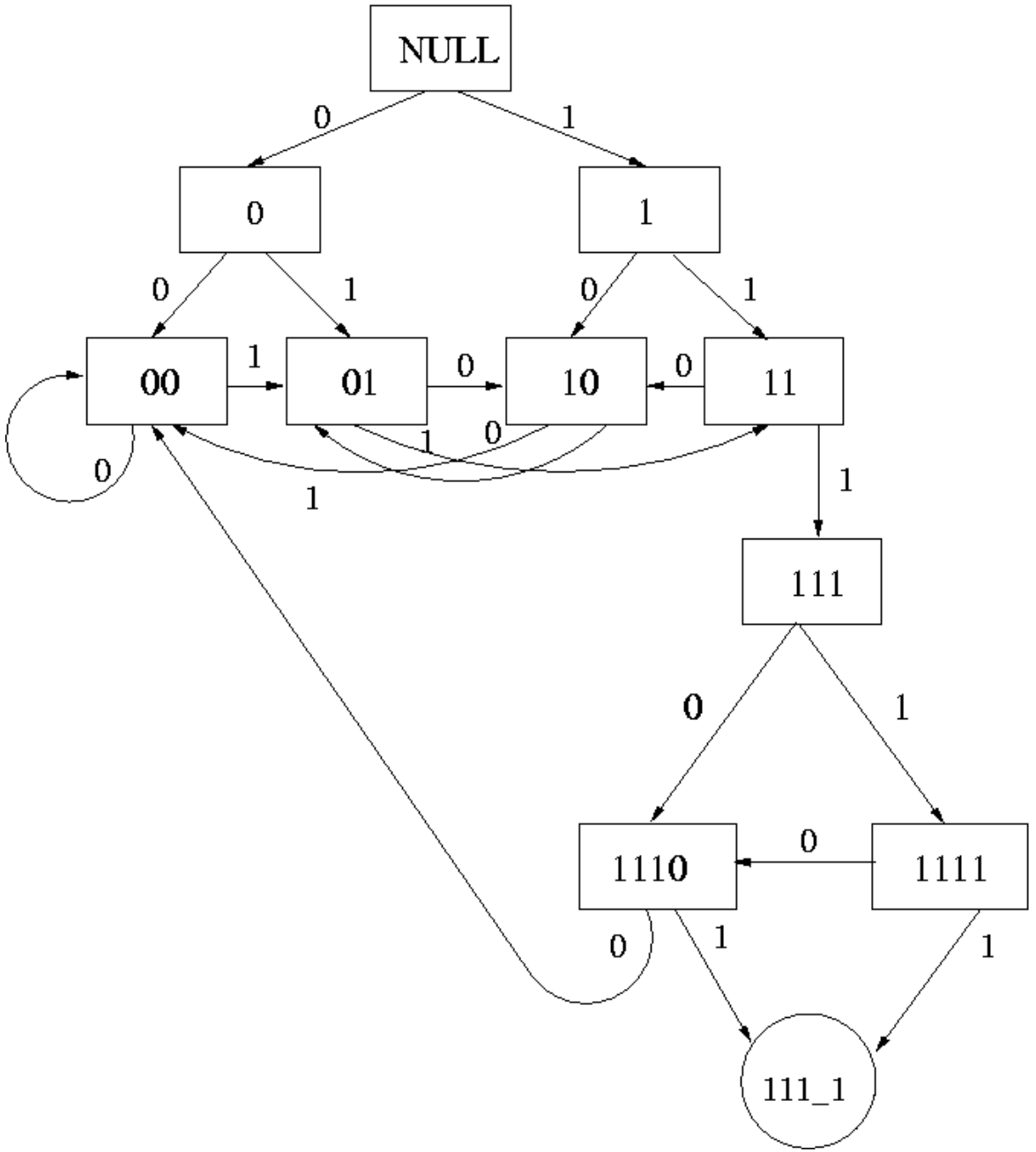
Computing the chain's probability

- $P_m(X_l = T | X_1 = \emptyset) = P_m^l(\emptyset, T)$
- Let $N =$ number of automaton/chain states
  - $N = O(w2^{s-w})$
- For a general transition matrix P, computing P^2 generally requires $O(N^3)$ steps
- We only need $P^l(a, b)$ for a particular a which generally requires $O(lN^2)$
- However, P_m is a sparse transition matrix:
  - there are two transitions out of every state
  - there are at most $2N$ non-vanishing entries in P_m
- Thus, we can compute $P_m^l(\emptyset, T)$ in $O(lN)$ steps

What about Markov strings?

- So far we assumed a Bernoulli mismatch string
- Will this scheme work for a Markov mismatch string?
- Key: probabilities of string and corresponding path should agree
- Suppose S is generated by a 2nd order Markov chain
  - If we are at state "111" what is the probability of moving to state "1110"?
    - $P_S(0|11)$
  - If we are at state "$\emptyset$" what is the probability of moving to state "1"?
    - depends on how we got to $\emptyset$
- The states at depth $\geq$ order of chain have sufficient memory
- We need to add memory to the "leaner" states

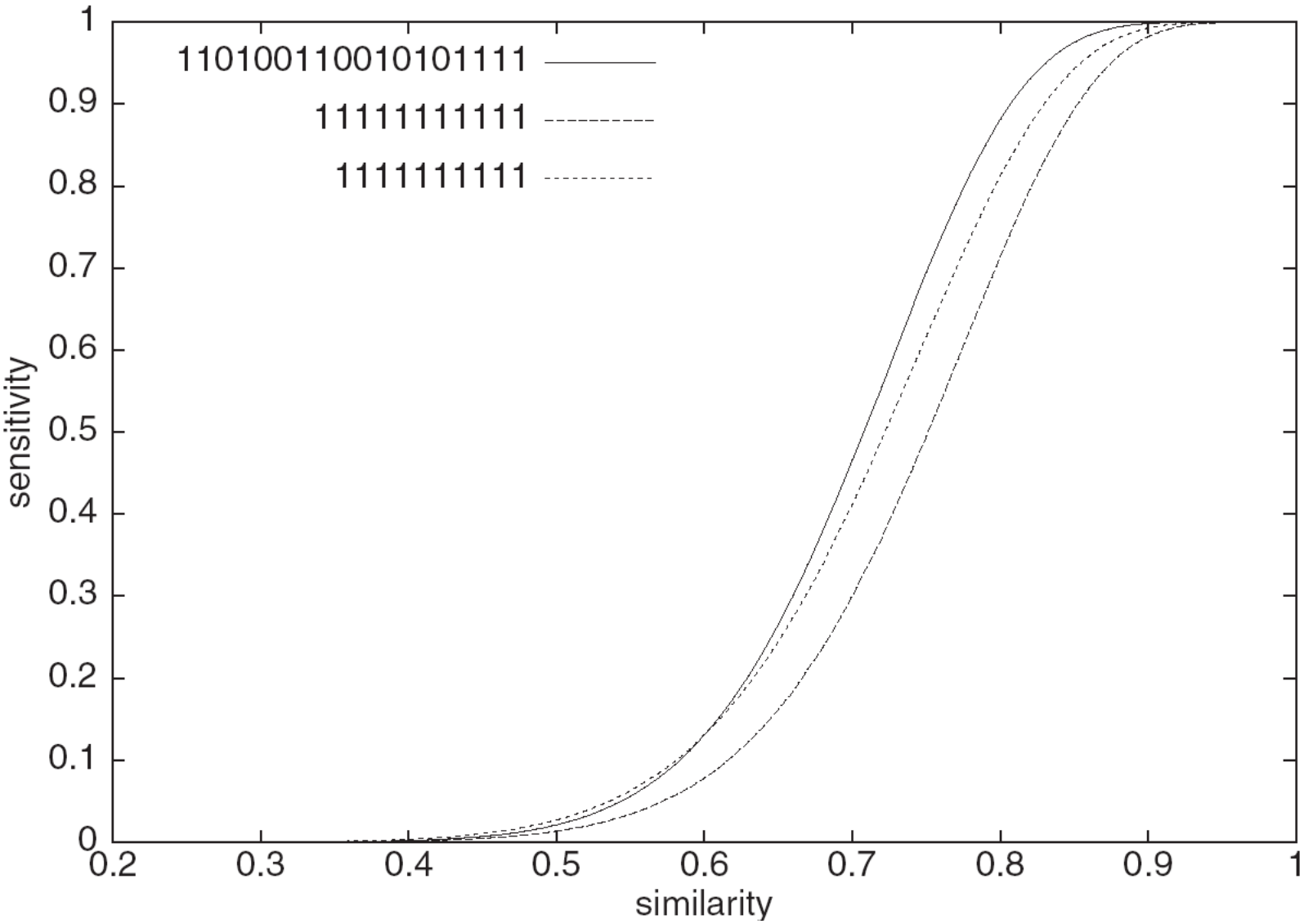
Extension to Markov strings



Finding Optimal Seeds

- Given a black box which computes the sensitivity find an optimal seed for a given mismatch model and w
- Short sighted approach: local search strategy
  - hill climbing
- Brute force approach: exhaustive enumeration for all $s \leq s_{max}$
  - not feasible for the empirical sensitivity
- For example, finding the optimal seed with $w = 11$ and $s \leq 22$ for a Bernoulli model with $l = 64$, $p = 0.7$
  - takes about 1 hour for exhaustive search on a 2.5GHz P4
  - a local search yields approximate results in seconds
- By design: identify the salient features of good seeds

Bernoulli sensitivity of optimal seed

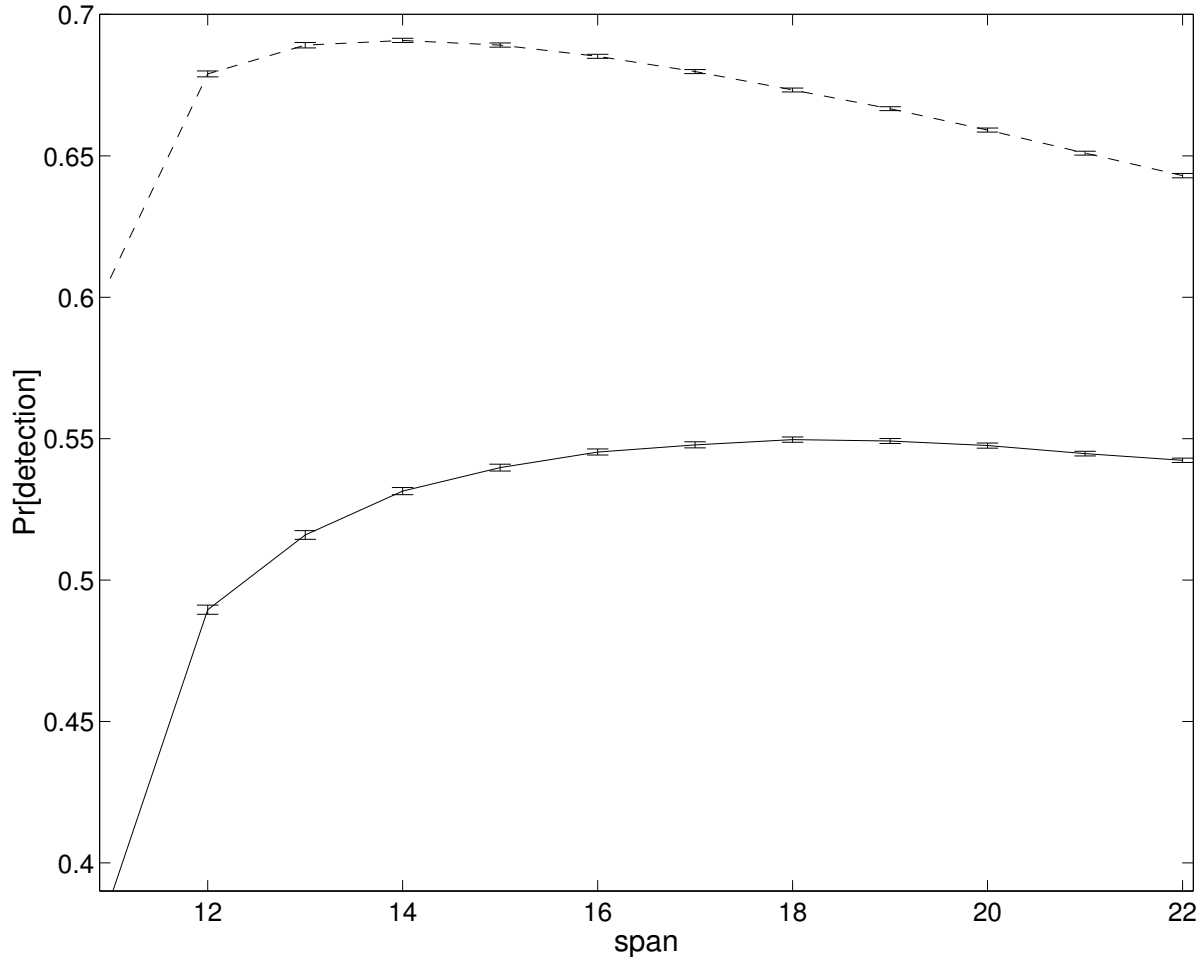


Mandala's optimal seeds: non-coding

Seed	Pattern	$P_5(E)$	Found $\times 10^3$	Time (mins)
π_c	$\{0,1,2,3,4,5,6,7,8,9,10\}$	0.607	220	382
π_{c10}	$\{0,1,2,3,4,5,6,7,8,9\}$	0.712	246	502
π_{ph}	$\{0,1,2,4,7,9,12,13,15,16,17\}$	0.689	252	417
π_{N_0}	$\{0,1,2,5,7,10,11,14,16,17,18\}$	0.680	252	417
π_{N_1}	$\{01,2,3,5,8,9,12,13,14,15\}$	0.699	252	423
π_{N_2}	$\{0,1,2,3,6,8,9,10,12,13,14\}$	0.707	253	424
π_{N_3}	$\{0,1,2,3,5,6,9,11,12,13,14\}$	0.704	252	422
π_{N_4}	$\{0,1,2,4,5,6,8,11,12,13,14\}$	0.707	253	425
π_{N_5}	$\{0,1,2,3,5,6,7,10,12,13,14\}$	0.709	253	424

Gapped alignments found and running times are on 500 megabases of homologous noncoding regions from human and mouse.

5-th vs. 0-th order Markov sensitivity



Average detection probabilities of 1000 random seeds given by 0-th (solid) and 5-order (dashed) Markov models. Error bars are 95% CI.

Data generation

- Human-Mouse genomes from UCSC Genome Browser
- Extracted 1262 pairs ($\approx 2.65 \times 10^9$) annotated as syntenic regions
  - orthologous regions with no major internal rearrangements
- Pairs were masked for repeats and low-complexity
- Divide into coding and non-coding regions (Twinscan predictions)
- Estimate 0-5th order non-coding Markov transition probabilities
  - by Sampling $\approx 1.4 \times 10^6$ ungapped alignments of $l = 64$ and 70-75% identity from non-coding pairs
    - higher identity rate: harder to distinguish seeds
    - sampling stratgey is important
- Tested on 449 pairs of syntenic fragments ($\approx 500 \times 10^6$ unmasked)
  - seed hits followed by BLAST's ungapped followed by banded SW

The Contiguous Seed π_c

- What's wrong with it?
- Going back to Conway's problem: why should we wait longer for `AAA` than for `ABC`?
- The *average* interarrival times (letters between occurrences) is the same for `AAA` and `ABC`
- Occurrences of `AAA` have certain tendency to cluster
- Occurrences of `ABC` cannot cluster
- Therefore interarrival times between clusters of `AAA` are typically longer
- More likely to see `ABC` before `AAA`

Analogy

- Arriving at a random time to a train station, which train line are we more likely to see departing first:
  - one that has 5 trains departing one per minute for the first 5 minutes after the hour
  - or one that has 5 trains departing at 12-minute intervals?

Shall we rule out π_c?

- What happens if $l = w$?
  - Due to its compactness, in some (pathological) cases π_c is the optimal seed
- Another example is when p is sufficiently small
  - Proof: inclusion-exclusion

  Moreover, there are seeds that will always be worse

  Claim 1. If π is an arithmetic progression with $d > 1$, e.g. $\pi = \{0, 2, 4 \dots\}$, then $P(\pi \in S(1 : l)) < P(\pi_c \in S(1 : l))$
- However, if we "level the playing field" then

  Claim 2.

$$P(\pi \in S(1 : l + s - w)) > P(\pi_c \in S(1 : l))$$

Asymptotic sensitivity

- The last claim somewhat goes out on a limb but
- There exists $\lambda = \lambda(\pi) \in [0, 1]$ and $\beta = \beta(\pi) > 0$ s.t.

$$P(\pi \notin S(1 : l)) \sim \beta\lambda^l$$

  - λ is the maximal eigenvalue of the automaton transition matrix
- Corollary: $\lambda(\pi_c) \geq \lambda(\pi)$
- Proof:

$$1 < \frac{P(\pi_c \notin S(1 : l))}{P(\pi \notin S(1 : l + s - w))} \sim \frac{\beta(\pi_c)\lambda(\pi_c)^l}{\beta(\pi)\lambda(\pi)^{l+s-w}}$$

$$\Longrightarrow \qquad \frac{\beta(\pi_c)}{\beta(\pi)\lambda(\pi)^{s-w}} \lim_{l\to\infty} \left[\frac{\lambda(\pi_c)}{\lambda(\pi)}\right]^l \geq 1,$$

which proves the claim

Asymptotic sensitivity - cont.

- Even one space can lead to better asymptotical result
- Let $\pi =$`111...1-1` and π_c be of weight $w \geq 2$

  Claim 3. $\lambda(\pi_c) > \lambda(\pi)$

  - Example: if $w \geq 4$ then for $l = w + 3$ and $p > 1/2$,

$$P(\pi \in S(1:l)) > P(\pi_c \in S(1:l))$$

- Conjecture: all non-periodic spaced seeds satisfy $\lambda < \lambda(\pi_c)$

Asymptotically optimal seeds

- Studying asymptotically optimal seeds elucidates structure
- The following seeds *seem* to be asymptotically optimal
  - $w = 3$: $\{0, 1, 3\}$
  - $w = 4$: $\{0, 1, 4, 6\}$
  - $w = 5$: $\{0, 3, 4, 9, 11\}$
  - $w = 6$: $\{0, 1, 8, 11, 13, 17\}$
  - $w = 7$: $\{0, 2, 3, 10, 16, 21, 25\}$
    - ▷ last one took a month of CPU time
- What's the rule?

Golomb rulers

- Every positive difference appears exactly once
- Minimal span with this property
- From James Shearer's home page (IBM) a minimal Golomb ruler of $w = 11$ (marks):
  - $\{0, 1, 4, 13, 28, 33, 47, 54, 64, 70, 72\}$
  - Demonstratively more sensitive for long sequences than the previously known optimal seed
  - How was this determined?
    - 2^{s-w} is too big: can't build automaton
    - Take large l (700)
    - Draw random mismatch strings of length l
    - Check in how many of those does the seed occurs
    - Obtain a high confidence interval for $P(\pi_G \in S(1:l))$

Golomb rulers and optimal seeds

- The contiguous seed is in some sense the worst
  - it suffers from heavy dependencies between adjacent occurrences
- Hypothetically independent occurrences provide optimal sensitivity
  - more precisely, yields an upper bound on sensitivity
- Golomb rulers represent minimal possible overlap (at most 1 in each shift)
  - best approximtion of independence given that you cannot avoid the overlap
- Open questions:
  - Can this be proved (*independently of* p)?
  - If there are multiple optimal Golomb rulers which one is the asymptotically optimal seed?