What next?

- Blastn: sensitivity/specificity is controlled by seed length
- Setting specificity by the seed weight w we optimize sensitivity using spaced seeds
- Possible further improvements:
  - generalize the notion of a seed match
    - for example allow a small number of mismatches
  - use multiple seeds

Multiple seeds

- Suppose π_w is an optimal seed of weight w
- To further improve our sensitivity we can look for τ_{w-1}, an optimal seed of weight $w-1$
- Alternatively, we can consider two seeds of weight w and define a seed hit as a match of any of them
- look for $\Omega = \{\rho_w, \xi_w\}$ that maximizes the sensitivity
- There is an overhead associated with using multiple seeds
  - two dictionaries
  - two scans of sequences

  The temporal overhead can be significantly reduced by
  - parallel machines/special hardware
  - indexing the DB for Q-DB search

Are two better than one?

- Specificity:
  - the FP rate of $\Omega = \{\rho_w, \xi_w\}$ is $\approx 2 \cdot 4^{-w}$
  - as opposed to $4 \cdot 4^{-w}$ for τ_{w-1}
- Sensitivity:

w	n	# alignments found	% improvement	total seed matches
11	1	251941	–	1.57×10^9
10	1	273831	8.7	5.88×10^9
9	1	293670	16.6	1.72×10^{10}
11	2	279902	11.1	3.10×10^9
11	3	292093	15.9	4.56×10^9
11	4	298968	18.7	6.05×10^9
11	5	303197	20.3	7.61×10^9

Finding optimal seeds - Sun & Buhler 04

- Mandala used an automaton to find the seed set sensitivity
- Local search was used to optimize the sensitivity
- Limited in practice to two seeds
  - slow convergence
  - sensitivity is recomputed from scratch at every step
- New ideas:
  - Add seeds according to a greedy strategy
  - beam search was also tried
    - ▷ at each stage keep b best seeds
    - ▷ keep N extensions of each of the previously b best
  - Compute $P(E_\pi | E_\Pi^c)$
    - ▷ maximize over π reusing a significant chunk of the computation for a given Π