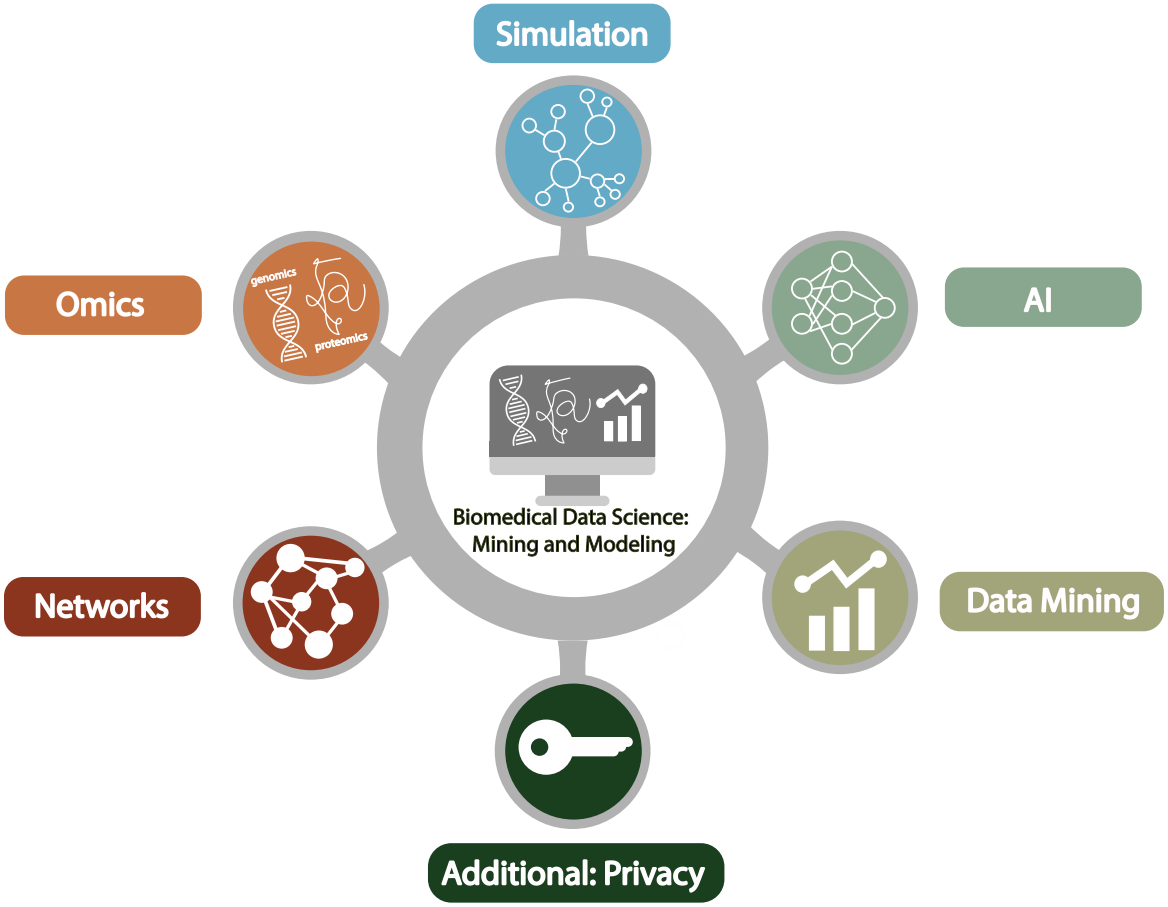
Biomedical Data Science (GersteinLab.org/courses/452)
Fast Alignment (25m5)



Mark Gerstein
Yale U.

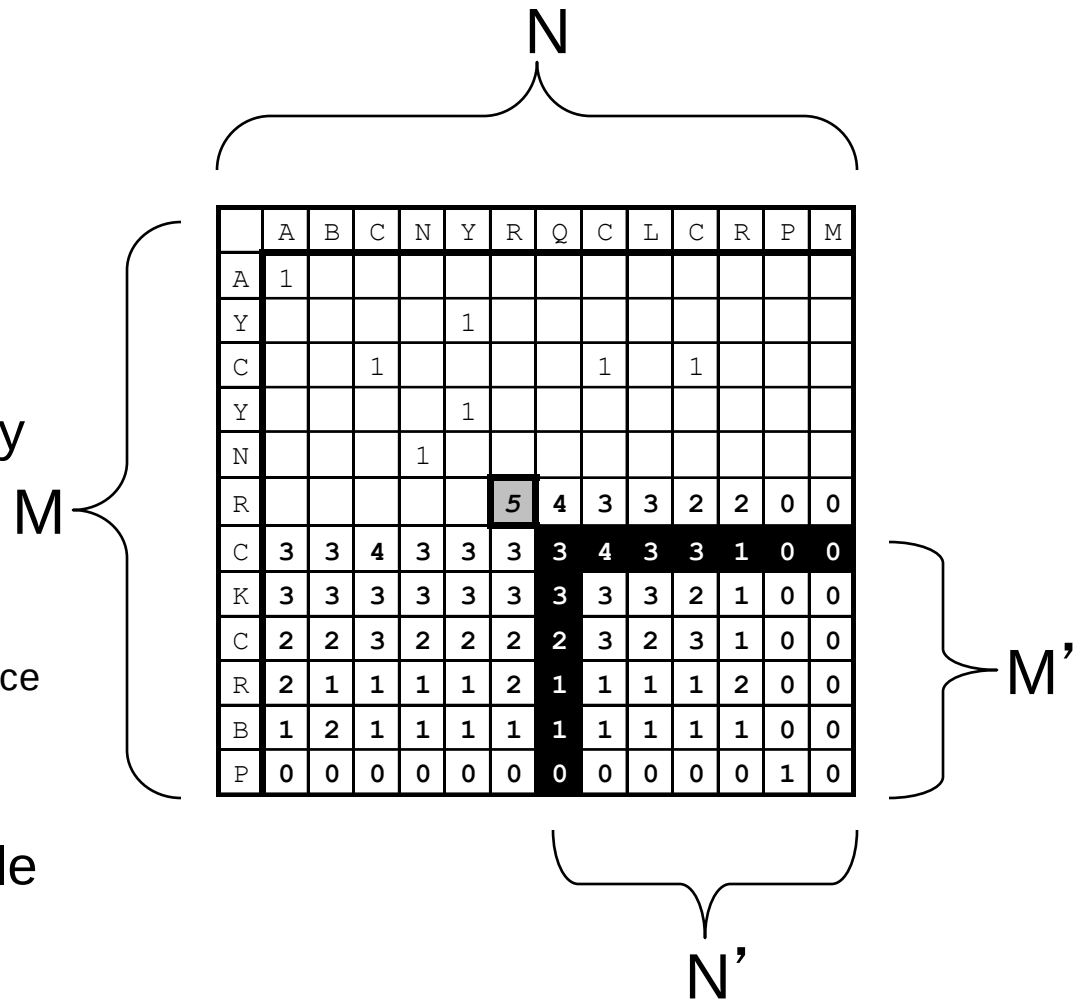
Last edit in spring ’25. Essentially unchanged from 22m5 & 2021’s M5 [which has a video].

Computational Complexity

- The dynamic programming alignment algorithm is
  $O(n\ m) \sim O(n^2)$
  in speed and memory

$O(n^2)$ in speed and memory is not good enough for important applications
  - ◊ database search
  - ◊ short read alignment to reference genome

- Note how this would scale to 3, 4, 5 sequences



Fast sequence alignment

- Alignment via dynamic programming (NW/SW)
  - ◊ useful for aligning the small numbers of protein, DNA sequences available in the 1980s
- 1990s hundreds of thousands of protein sequences
- Today thousands of genome sequences

- => need for faster, more coarse-grained alignment methods
  - ◊ first application: find your favorite protein in a sequence database
  - ◊ next-gen seq application: align millions of short reads to a reference database

Computational Complexity

- Designing algorithms involves a trade-off between calculation time and memory usage & sensitivity

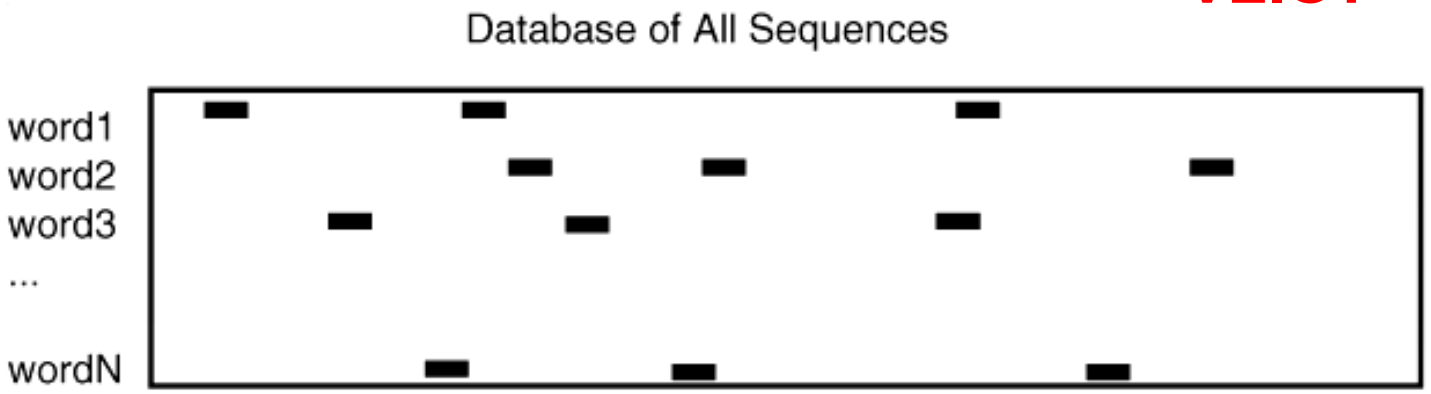
- Steps that can be pre-calculated and stored efficiently in memory speed up the algorithm

- FASTA (hashing the query)
- BLAST (more efficient query hashing)
- BLAT (hashing the DB)
- BWA / Bowtie (BW transform of the DB)

FASTA

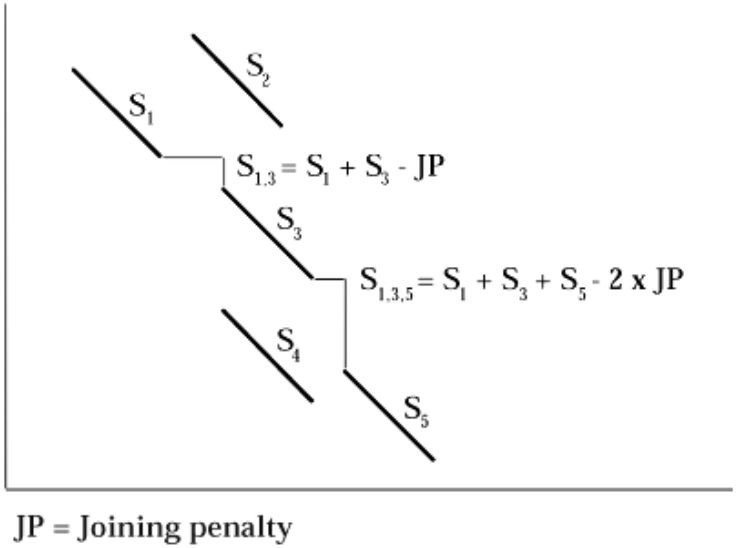
- Hash table of short words in the query sequence
- Go through DB and look for matches in the query hash (linear in size of DB with const. time hash)
- perl: $where{“ACT”} = 1,45,67,23....
- K-tuple determines word size (k-tup 1 is single aa)
- by Bill Pearson

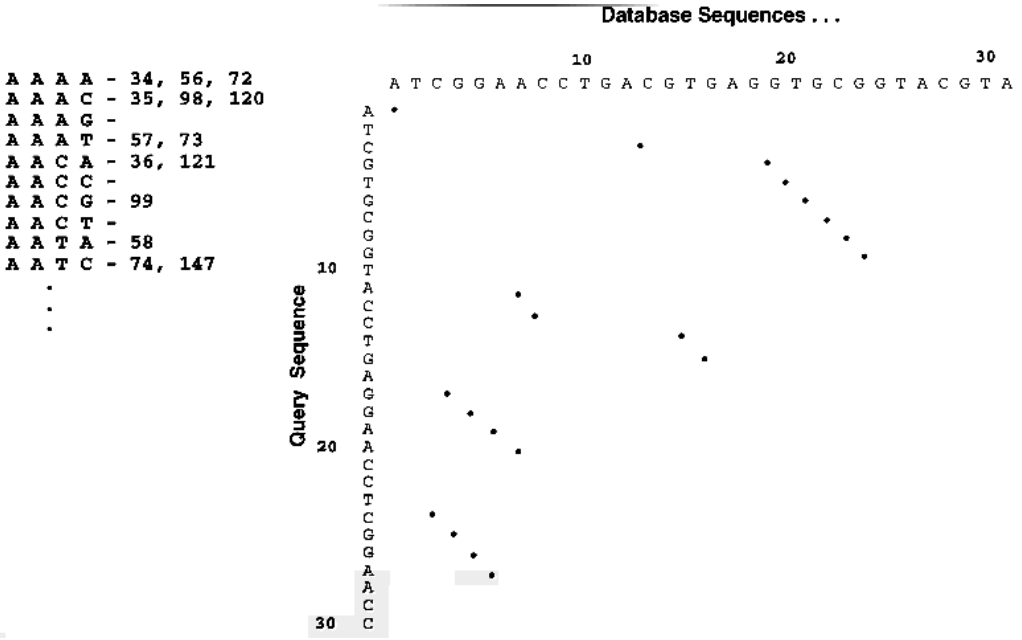
VLICT = —

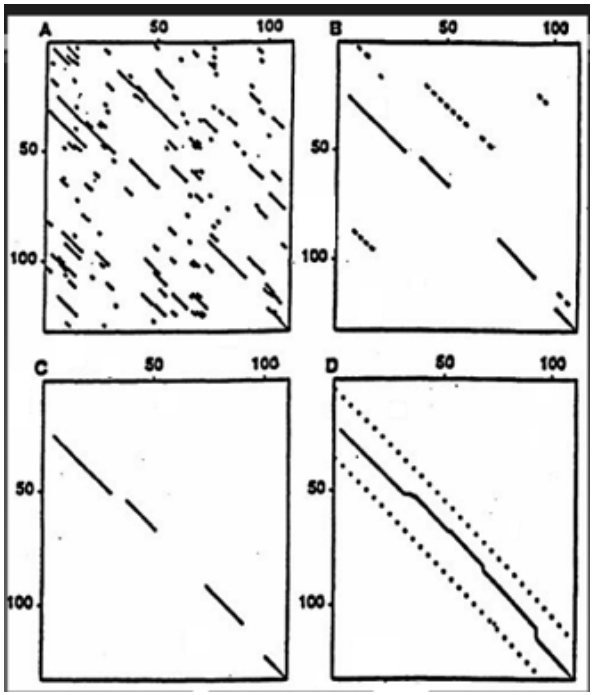


VLICTAVLM**VLICT**AAA**VLICT**MSDFFD

Join together query lookups into diagonals and then do a full alignment





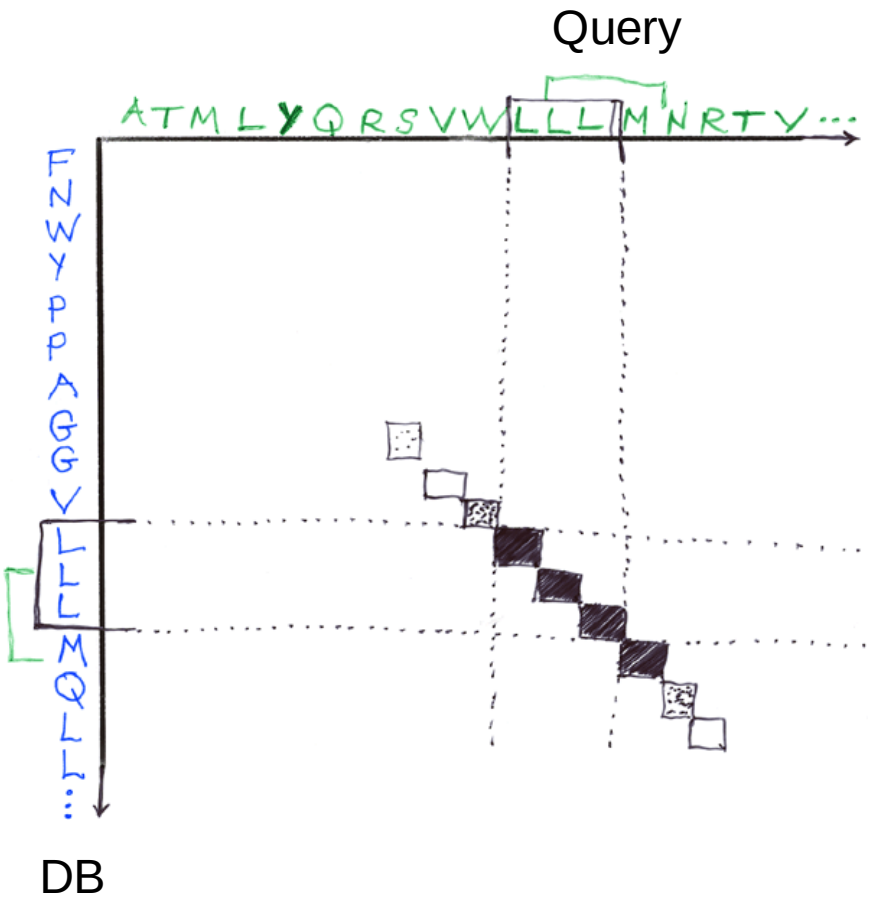


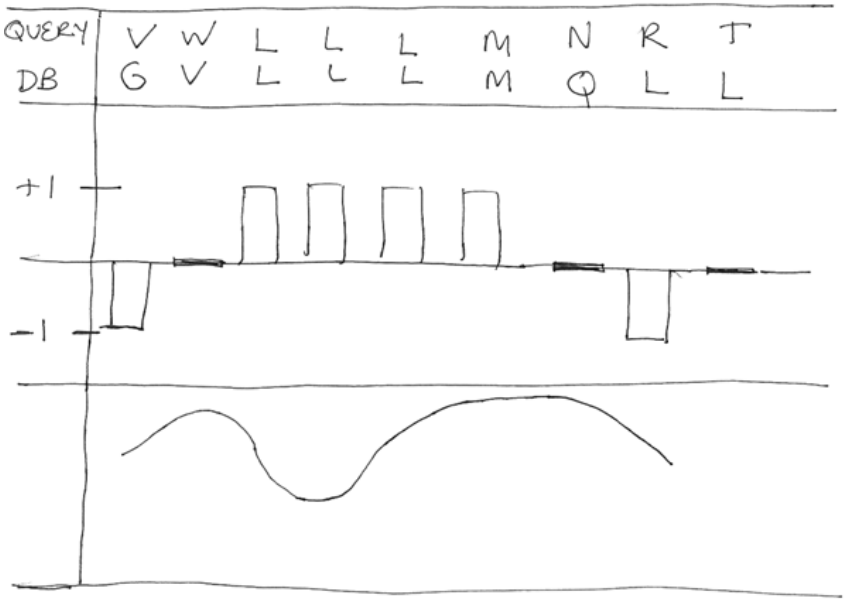
(Adapted from D Brutlag)

Basic Blast

- Altschul et al. *J. Mol. Biol.* **215**, 403-410
- Indexes query
- Starts with all overlapping words from query
- Calculates “neighborhood” of each word using PAM matrix and probability threshold matrix and probability threshold
- Looks up all words and neighbors from query in database index
- Extends High Scoring Pairs (HSPs) left and right to maximal length
- Finds Maximal Segment Pairs (MSPs) between query and database
- Blast 1 does not permit gaps in alignments

BLAST: Basic Local Alignment Search Tool





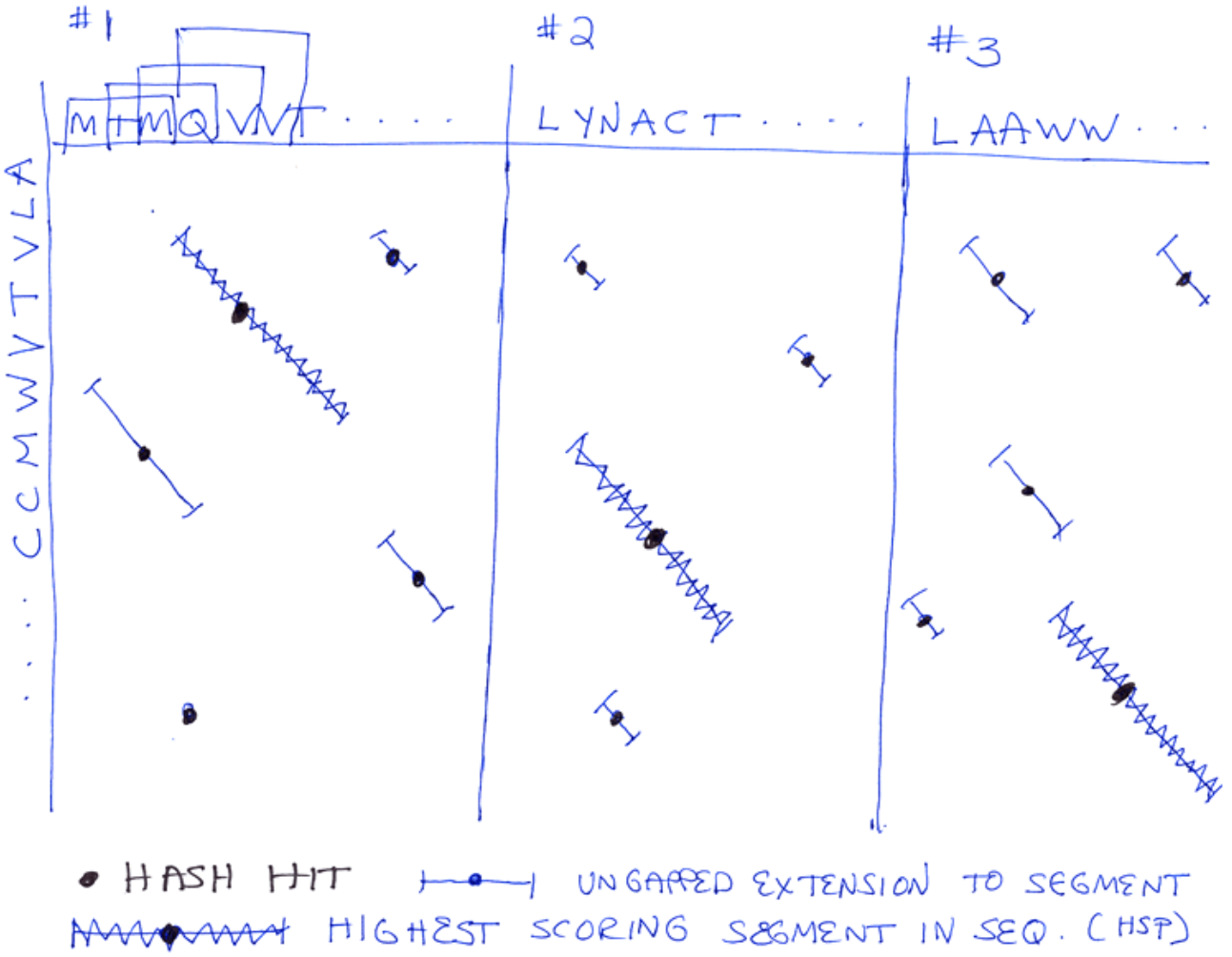
- Extend hash hits into High Scoring Segment Pairs (HSPs)
- Stop extension when total score doesn't increase
- Extension is O(N). This takes most of the time in BLAST

BLAST: Basic Local Alignment Search Tool

- In simple BLAST algorithm, find best scoring segment in each DB sequence
- Statistics of these scores determine significance

Number of hash hits
~ O(N*M*D)

where
N is query size
M is average size of seq in DB
D is DB size



Short read alignment to a reference genome

- BLAT
- Burrows-Wheeler transform

BLAT

- "BLAST-like alignment tool"
- created by Jim Kent (UCSC) during assembly of the human genome

- Where BLAST builds an index of the query sequence, BLAT builds an index of the database.
  - ◊ Obviously, this will scan more quickly through the DB at the expense of building a huge hash table of the DB initially
  - ◊ DB index non-overlapping, potentially sacrificing some sensitivity for decreased memory usage

BLAT – Genome Res. (2002) 12: 656

Burrows Wheeler Transform

- What's next: more sophisticated ways of organizing the genome pre-search to speed things up beyond building the DB hash table as in BLAT

- High Level
  - ◊ Build a BWT of the genome (cyclically permuting, then sorting, then compressing)
  - ◊ Then build a prefix tree of this
  - ◊ Take each read and search along the prefix tree in linear time
  - ◊ Reverse the transform to find the location of the read in the genome from its position in the prefix tree.

Burrows and Wheeler, A Block-sorting Lossless Data Compression Algorithm, Digital Systems Research Center Report 124 (1994)

Burrows Wheeler Transform

- BWT is a reversible permutation of the characters in a string X

1. build matrix of cyclic rotations of X
2. sort matrix alphabetically

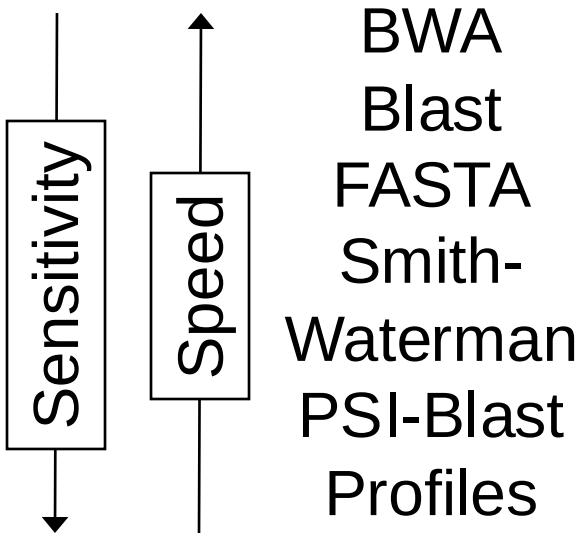
```
example:   X = acaacg

0 acaacg$          6 $acaacg
1 caacg$a          2 aacg$ac
2 aacg$ac          0 acaacg$
3 acg$aca   =>     3 acg$aca
4 cg$acaa          1 caacg$a
5 g$acaac          4 cg$acaa
6 $acaacg          5 g$acaac
```

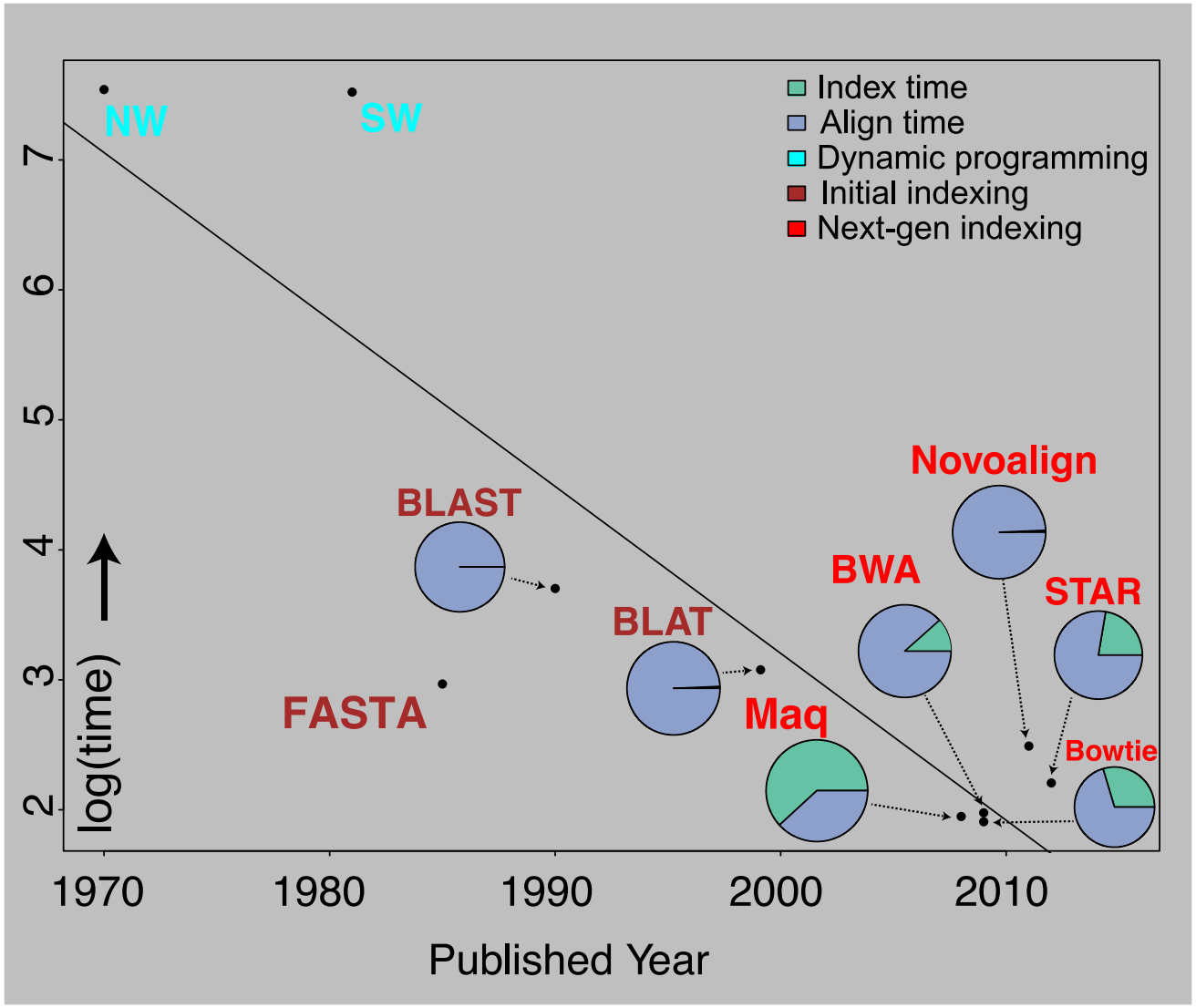

Bowtie – Genome Biol. (2009) 10: R25

Speed v Sensitivity Tradeoff

PSI Blast as a form of Semi-supervised learning



Alignment algorithms scaling to keep pace with data generation



[Sboner et al. (‘11), Muir et al. (‘15) Genome Biology]

What sequence alignment algorithms need to be designed next?

A couple of important problems:

- rapidly align a personal genome to a reference population of human genomes
  - ◊ with clinical turn-around time; with privacy => encryption?
- 3rd generation sequencers: long, error-prone reads
  - ◊ useful as scaffolds mixed with more accurate, cheaper short reads

References

- (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 11, No. 6, 2020
  A Categorization of Relevant Sequence Alignment Algorithms with Respect to Data Structures
  https://pdfs.semanticscholar.org/ce61/04863eedcfaecad98ea2bd31d4c9435d2b9b.pdf
  (Most Important)

- Altschul, S. F., Gish, W., Miller, W., Myers, E. W., & Lipman, D. J. (**1990**). Journal of Molecular Biology, 215(3), 403–410.
  Basic local alignment search tool.
  https://doi.org/10.1016/s0022-2836(05)80360-2
  (http://www.gersteinlab.org/courses/452/10-spring/pdf/Altschul.pdf)
  (Just Methods Section)