Master Course
Sequence Alignment

Lecture 7
Database searching (1)

Sequence searching - challenges
- Exponential growth of databases

Bioinformatics justification
- "Mind the Gap"
- There are far more sequence data than structural/functional data
- We need to fill this gap by analysis and prediction pipelines

Sequence searching - definition
- Task:
  - Query: short, new sequence (~1000b)
  - Database (searching space): very many sequences
  - Goal: find seqs related to query
- We want:
  - fast tool
  - primarily a filter: most sequences will be unrelated to the query
  - fine-tune the alignment later

Heuristic Alignment Motivation
- the dynamic programming algorithm has complexity $O(mn)$, which is too slow for large databases with high query traffic
  - MPtree [Sharrock & Collins, MPtree version 1.3 (1993) – Massively parallel DP]
  - heuristic methods do fast approximation to dynamic programming
    - FASTA [Pearson & Lipman, 1988]
    - BLAST [Altschul et al., 1990]

Heuristic Alignment Motivation
- consider the task of searching SWISS-PROT against a query sequence:
  - say our query sequence is 362 amino-acids long
  - SWISS-PROT release 38 contains 29,085,265 amino acids
  - finding local alignments via dynamic programming would entail $O(10^{15})$ matrix operations
  - many servers handle thousands of such queries a day (NCBI > 50,000)
  - Using the DP algorithm for this is clearly prohibitive
  - Note: each database search can be sped up by ‘trivial parallelisation’
Heuristic Alignment

- Today: BLAST is discussed to show you a few of the tricks people have come up with to make alignment and database searching fast, while not losing too much quality.

What is BLAST
- Basic Local Alignment Search Tool
- Bad news: it is only a heuristic
- Heuristics: A rule of thumb that often helps in solving a certain class of problems, but makes no guarantees.
- Also see http://en.wikipedia.org/wiki/Heuristic
- Basic idea:
  - Discard putatively unrelated sequences fast
  - High scoring segments have well conserved (almost identical) part
  - As well conserved parts are identified, extend these to the real alignment

What means well conserved for BLAST?
- BLAST works with k-words (words of length k)
  - k is a parameter
  - different for DNA (>10) and proteins (2..4), default k values are 11 and 3, resp.
  - word w_i is T-similar to w_j if the sum of pair scores is at least T (e.g. T=12)

BLAST algorithm 3 basic steps
1) Preprocess the query sequence: extract all the k-words
2) Scan for T-similar matches in database
3) Extend them to alignments

BLAST, Step 1: Preprocess the query
- Take the query (e.g. LVNKRPVVP)
- Chop it into overlapping k-words (k=3 in this case)
  - Query: LVNKRPVVP
  - Word1: LVN
  - Word2: VNR
  - Word3: NRK
- For each word find all similar words (scoring at least 7)
  - E.g. for RKP: the following 3-words are similar:
    - QKP, KKP, RKP, RQP, REP, RSP, RKP

Step 2: Scanning the Database with DFA (Deterministic Finite-state Automaton)
- search database for all occurrences of query words
  - can be a massive task
  - approach:
    - build a DFA (deterministic finite-state automaton) that recognizes all query words
    - run DB sequences through DFA
    - remember hits
**DFA**
Finite state machine
- abstract machine
- constant amount of memory (states)
- used in computation and languages
- recognizes regular expressions
- cp dmt*.pdf /home/john

**BLAST, Step 2:**
Find “exact” matches with scanning
- Use all the T-similar k-words to build the Finite State Machine
- Scan for exact matches

**Scanning the Database - DFA**

**Example (next 2 slides):**
- consider a DFA to recognize the query words: QL, QM, ZL
- all that a DFA does is read strings, and output "accept" or "reject."
- use Mealy paradigm (accept on transitions) to save space and time

---

**a DFA to recognize the query words: QL, QM, ZL in a fast way**

**BLAST, Step 3:**
Extending “exact” matches
- Having the list of matches (hits) we extend alignment in both directions

**Query:**
L V N R R P V P

**T-similar:**
R R P

**Subject:**
G V C R R R L K C

**Score:**
-3 4 3 5 2 7 1 2 -3

- ...till the sum of scores drops below some level X from the best known

---

**Step 3: Extending Hits**
- extend hits in both directions (without allowing gaps)
- terminate extension in one direction when score falls certain distance below best score for shorter extensions

**return segment pairs scoring at least S**
More Recent BLAST Extensions

- the two-hit method
- gapped BLAST
- hashing the database
- PSI-BLAST

All are aimed at increasing sensitivity while keeping run-times minimal.

Altshul et al., Nucleic Acids Research 1997

The Two-Hit Method

- extension step typically accounts for 90% of BLAST’s execution time
- key idea: do extension only when there are two non-overlapping hits on the same diagonal within distance A of each other
- to maintain sensitivity, lower T parameter
- more single hits found
- but only small fraction have associated 2nd hit

Gapped BLAST

- trigger gapped alignment if two-hit extension has a sufficiently high score
- find length-11 segment with highest score; use central pair in this segment as seed
- run DP process both forward & backward from seed
- prune cells when local alignment score falls a certain distance below best score yet

Gapped BLAST

Combining the two-hit method and Gapped BLAST

- Before:
  - relatively high T threshold for 3-letter word (hashed) lists
  - two-way hit extension (see earlier slides)
- Current BLAST:
  - Lower T, many more hits (more 3-letter words accepted as match)
  - Relatively few hits (diagonal elements) will be on same matrix diagonal within A given distance A
  - Perform 2-way local Dynamic Programming (gapped BLAST) only on ‘two-hits’ (preceding bullet)

The new way is a bit faster on average and gives better (gapped) alignments and better alignment scores!
Making things even faster-indexing the complete database (or genome sequence)

- SSAHA – Sequence Search and Alignment by Hashing Algorithms (Ning et al., 2001)
- BLAT – BLAST-like Alignment Tool (Kent, 2002)
- PatternHunter (Ma et al., 2002)
- BLASTZ – alignment of genomic sequences (Schwartz et al., 2003)

Hashing - examples

- T9 Predictive Text in mobile phones
  - "hello": 4, 4, 3, 3, 5, 5, 5, (pause) 5, 5, 5, 6, 6
  - "hello" in T9: 4, 3, 5, 5, 6
  - Collisions: 4, 6; "in", "go"

Hashing - examples (cont..)

- Other easier hash function: let a=1, b=2, c=3, etc.
  - "hello" now gets hash address 8+5+12+12+15 = 52
  - "olleh" will get same address (collision)
  - Each word encountered gets a hash address immediately and can be indexed.
  - How good is this hash function?

BLAT - BLAST-Like Alignment Tool

- Analyzing vertebrate genomes requires rapid mRNA/DNA and cross-species protein alignments. BLAT (the BLAST-like alignment tool) was developed by Jim Kent from UCSC. It is more accurate and 500 times faster than popular existing tools such as BLAST for mRNA/DNA alignments and 50 times faster for protein alignments at sensitivity settings typically used when comparing vertebrate sequences (e.g. BLAST).
- BLAT's speed stems from an index of all nonoverlapping k-mers in the genome. This index fits inside the RAM of inexpensive computers, and need only be computed once for each genome assembly. BLAT has several major stages. It uses the index to find regions in the genome likely to be homologous to the query sequence. It performs an alignment between homologous regions. It stitches together these aligned regions (often existing larger alignments) typically graces. Finally, BLAT revisits small internal exons possibly missed at the first stage and adjusts large gap boundaries that have canonical splice sites where feasible.
- From Wikipedia, the free encyclopedia

Hashing - associative arrays

- Indexing with the object, the
- Hash function:
  - hash: small
  - large
  - set of possible objects
- Objects should be "well spread"

Indexing the database: Find "exact" matches with hashing

- Preprocess the database
- Hash the database with k-words
- For each k-word store in which sequences it appears

1/5/2008
Indexing the database:
Find “exact” matches with hashing

- The database is preprocessed only once! (independent from the query)
- In a constant time we can get the sequences with a certain k-word

BLAST flavours

- blastp: protein query, protein db
- blastn: DNA query, DNA db
- blastx: DNA query, protein db
  - in all reading frames. Used to find potential translation products of an unknown nucleotide sequence.
- tblastn: protein query, DNA db
  - database dynamically translated in all reading frames.
- tblastx: DNA query, DNA db
  - all translations of query against all translations of db (compare at protein level)