CMSC423: Bioinformatic Algorithms, Databases and Tools
Lecture 16

Software testing

Dynamic programming

Lessons from project 1

- How do you test your code?
- Mechanics:
  - diagnostic output: printfs (send to STDERR), -debug option, log files
  - invariants (assert function in C)
- Semantics:
  - small artificial data-sets that can be checked manually
  - compare to other programs
  - artificial data-sets that could confuse your program

Specific for Project 1

- Swap text and query – result should be the same
- Short sequence that matches: exactly, with one substitution; with one gap; with two gaps AGACCTAG, AGAGCTAG, AGACGCTAG, AGACGGCTAG – easy to compute score by hand
- Sequence that matches exactly end-to-end
- Sequence that matches with errors at the end only
- Sequence that doesn't match at all
- Sequence that matches many places
- Low-complexity sequence (e.g. AAAAAA)
- Compare to Jaligner, Blast, FASTA, etc.

Dynamic programming

- Key elements
  - Problem has to be decomposable (solution of an instance of the problem can be computed from smaller instances)
  - Recurrence relationship – how to combine solution to smaller instances to compute a larger instance
  - Initial conditions – set of trivially solved instances of the problem
  - location of answer in the matrix
  - trace-back
- Sequence alignment
  - decomposable: alignment of string of length k can be computed from alignment of string of length k – 1.
  - recurrence: V[i,j] = V[i, j – 1] + cost of gap, etc...
  - initial conditions: V[i,0] = i * cost of gap, etc...

Some examples

- 1-D chaining
- All-pairs shortest path