Tree-based Parallel Smith-Waterman Gene Sequencing

Avrilia Floratou
University of Wisconsin

Paradyn / Dynist Week
April 27-28, 2009

The problem

- Determine similar regions between 2 DNA sequences (local alignment).
- Useful in bioinformatics
  a) Identify mutations.
  b) Produce phylogenetic trees .
- This study is interesting as an intense evaluation of the TBON as a vehicle to support large-scale application development.

Why is the problem difficult to solve?

- Existing algorithms perform well on small sequences.
- As the sequences become lengthier we need
  a) much more computing power
  b) much more memory

Length of the human genome =~3.2 billion base pairs

Existing approaches

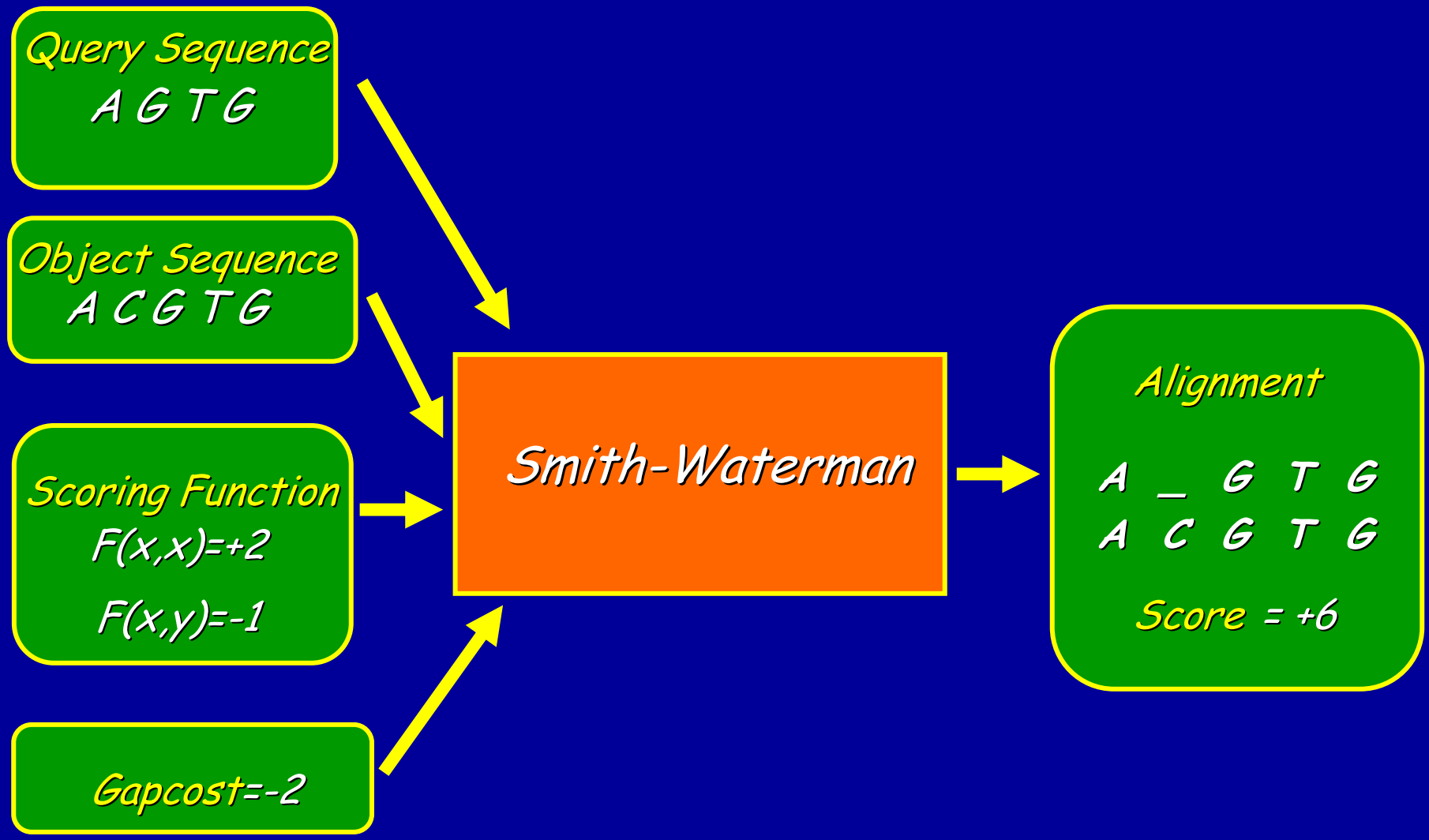
Smith-Waterman	*BLAST*
• *100% accurate* • *Slow*	• *Uses heuristics* • *Fast*

Need for parallelism

a) Parallel Smith-Waterman.

b) Parallel BLAST.

Smith-Waterman algorithm



Basic Operations: Replacement

Score

+5

Q:	A	C	T	G
O:	A	G	T	G

F(x, x) = +2

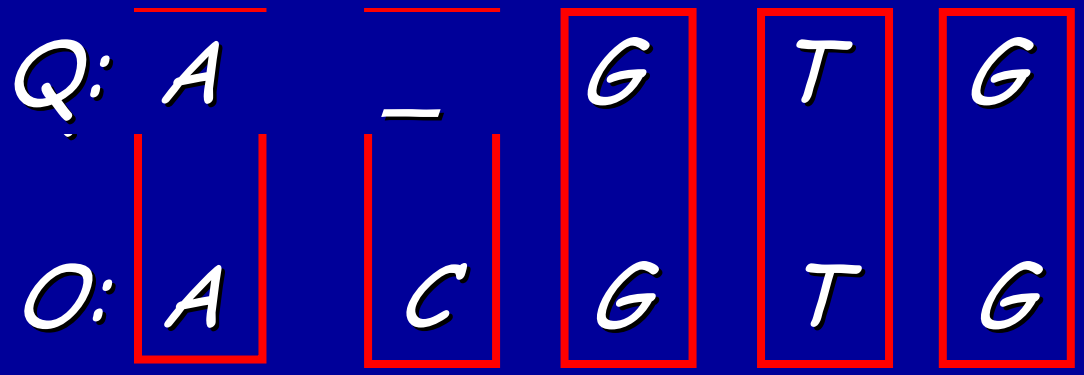
F(x, y) = -1

Gapcost = -2

Basic Operations: Insertion

Score

+6



F(x, x) = +2

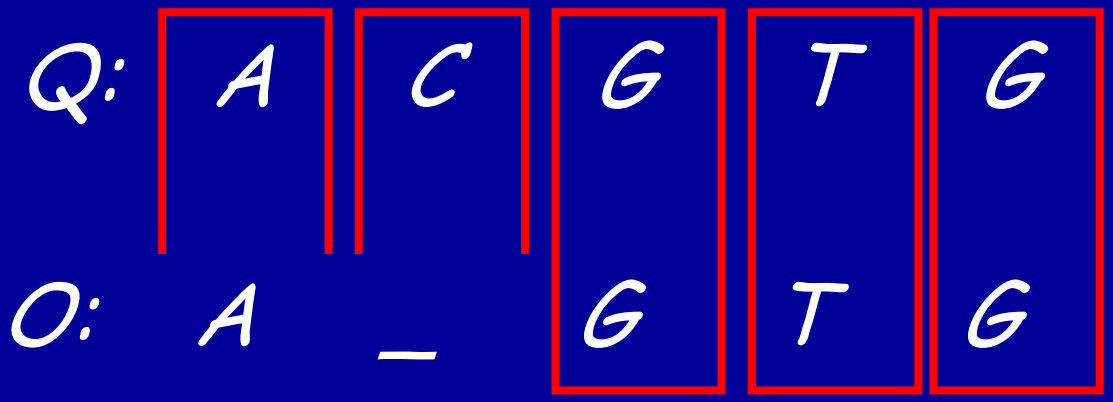
F(x, y) = -1

Gapcost = -2

Basic Operations: Deletion

Score

+6

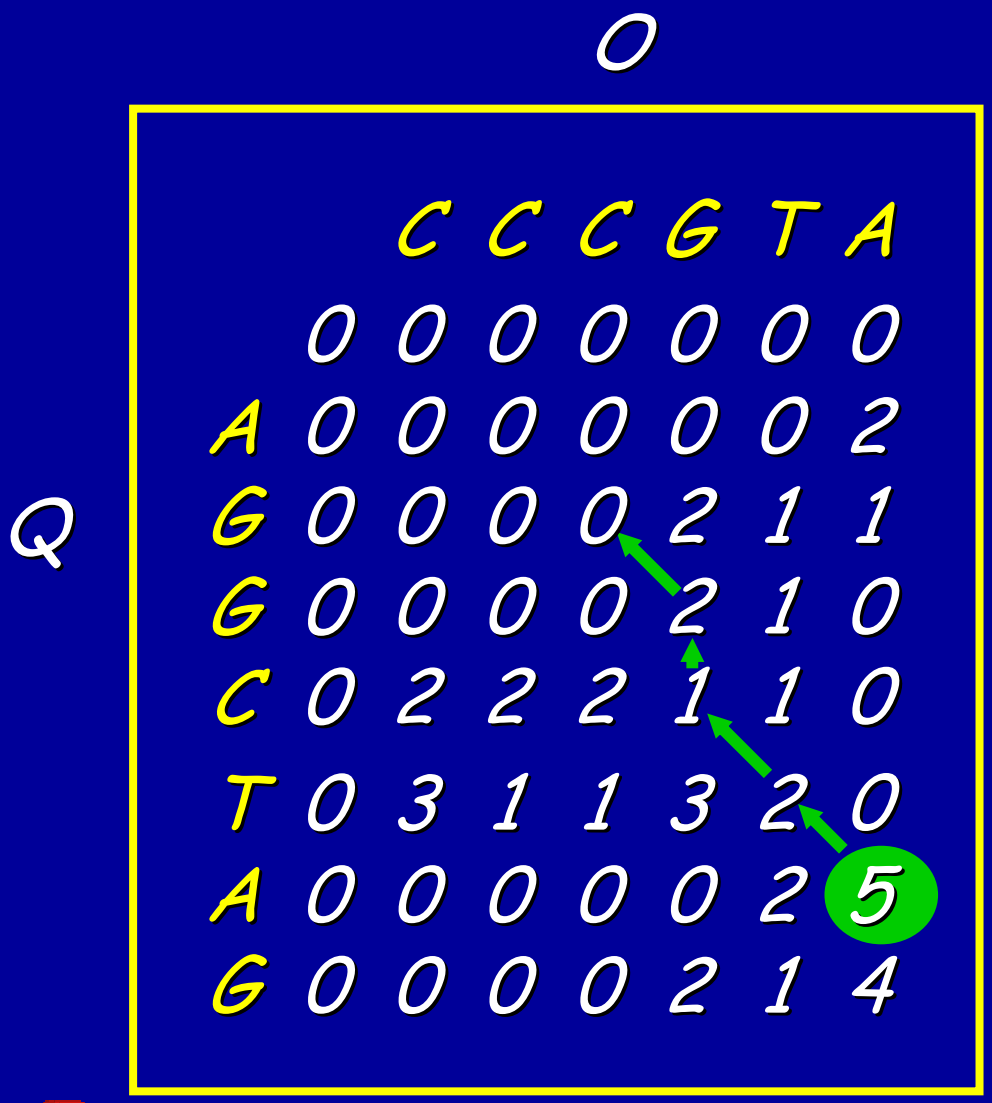


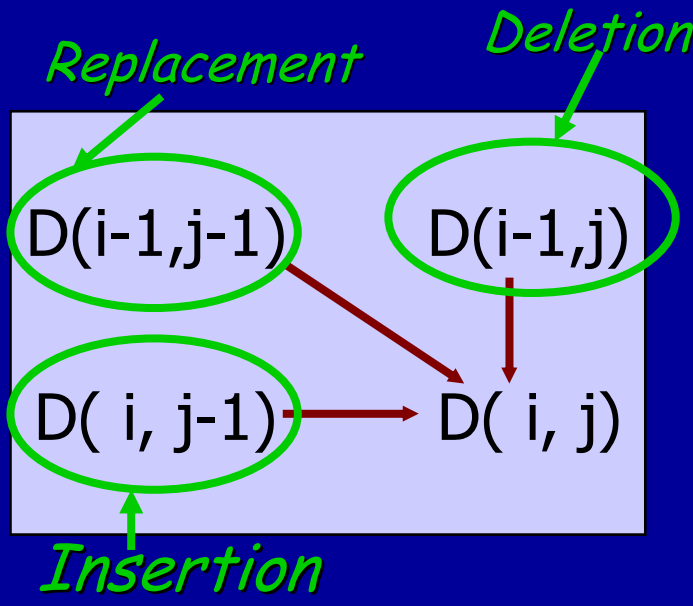
F(x, x) = +2

F(x, y) = -1

Gapcost = -2

Smith-Waterman algorithm





Alignment

G C T A

C _ T A

Recurrence Relation

- $D(i, j) = \max \begin{cases} 0 \\ D(i-1, j-1) + f(s_i, t_j) \\ D(i-1, j) - \text{gapcost} \\ D(i, j-1) - \text{gapcost} \end{cases}$

Replacement

Deletion

Insertion

Problems of this approach

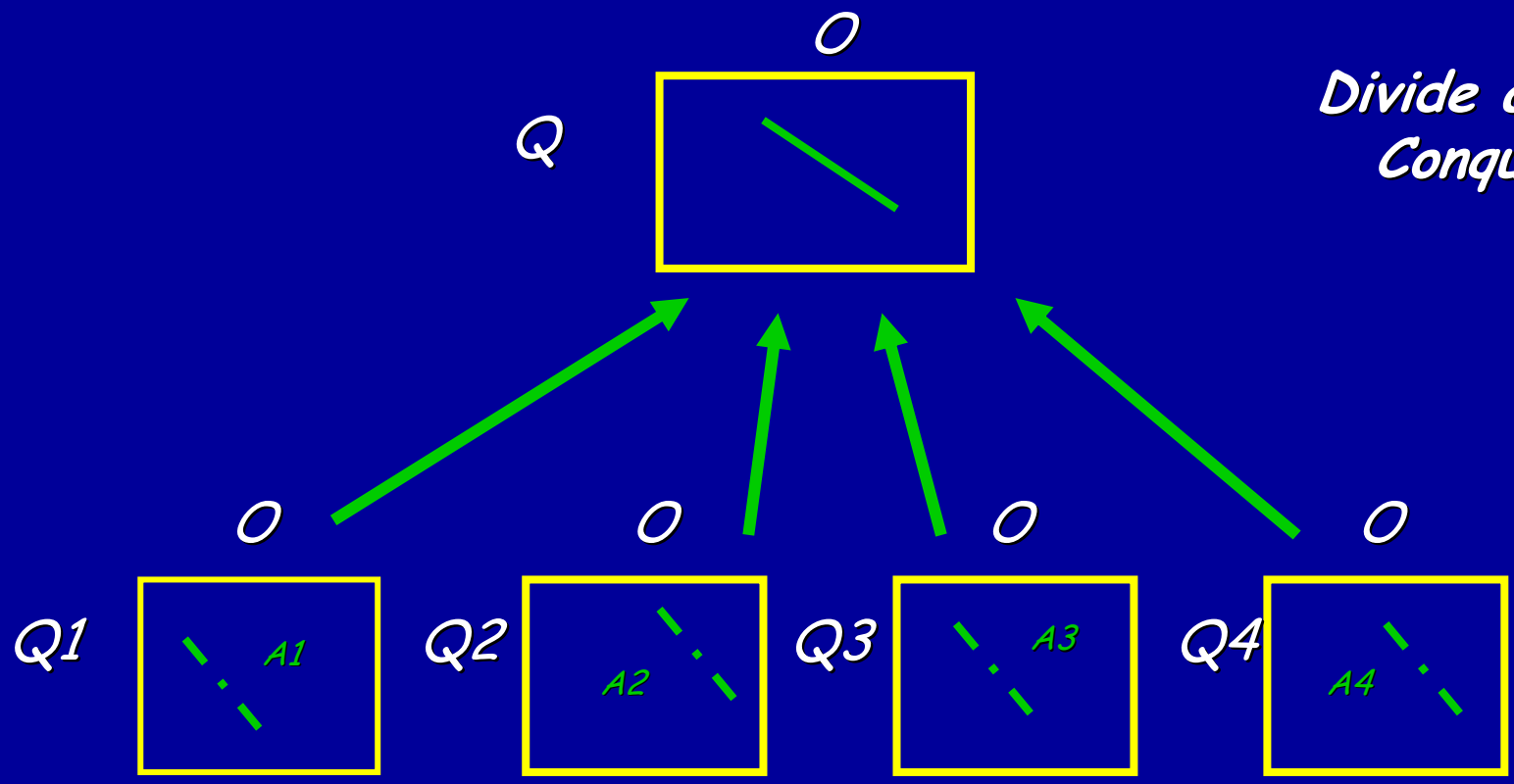
- Memory requirements

  For sequences of length N the edit matrix has size N^2.

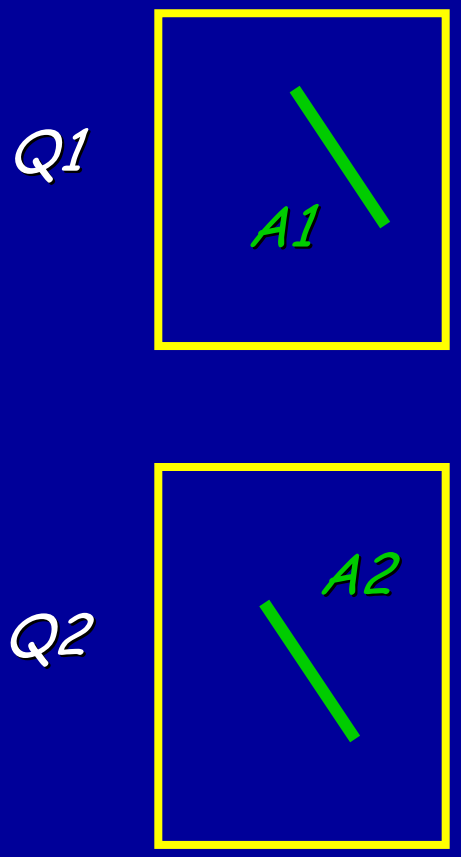
- Inefficiency in performance.

  Dynamic programming is time-consuming.

PSW-DC : A parallel Smith- Waterman algorithm

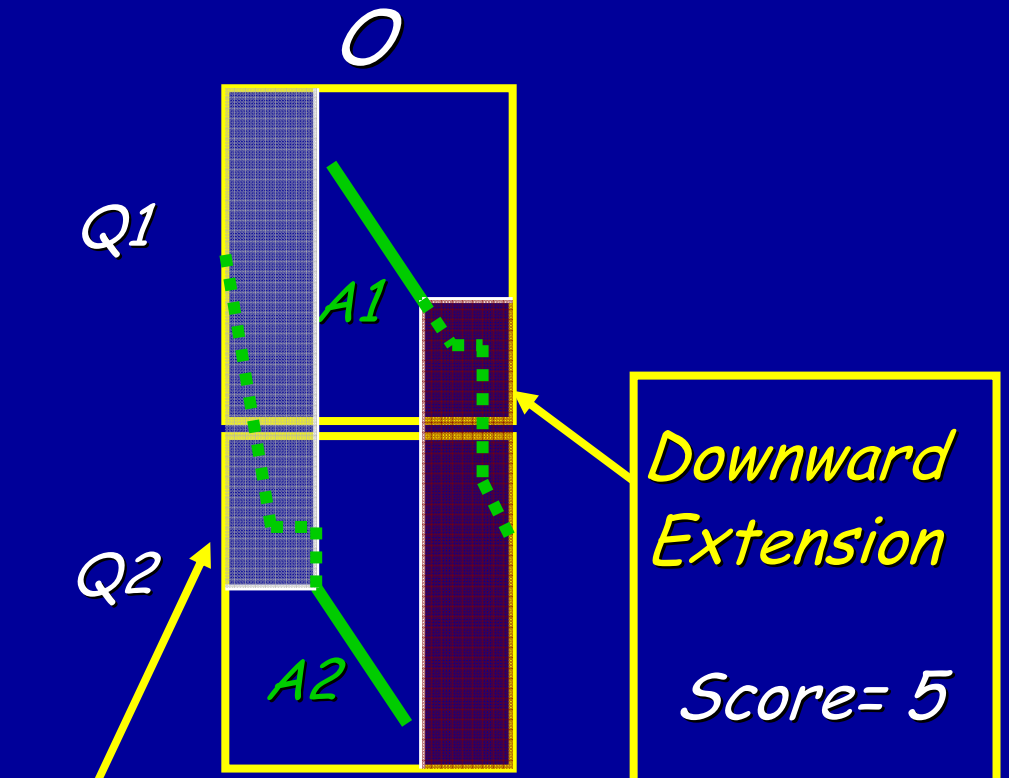


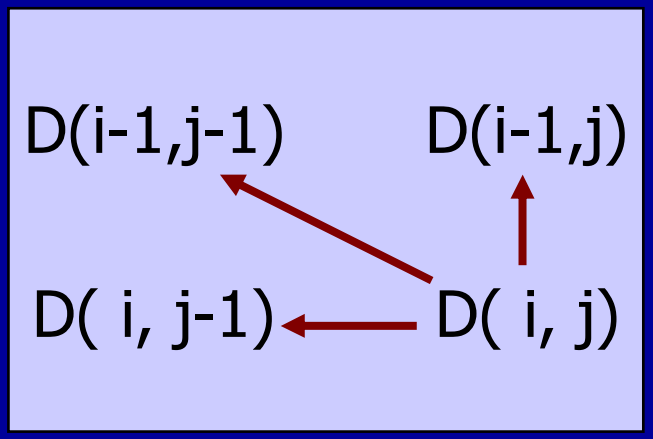
How do we combine the local alignments?



- The goal is to create a new alignment by joining A1 and A2.
- Connecting the 2 alignments is not always possible. Instead we may need to ***extend*** the alignments upwards or downwards according to their position in the local matrices.

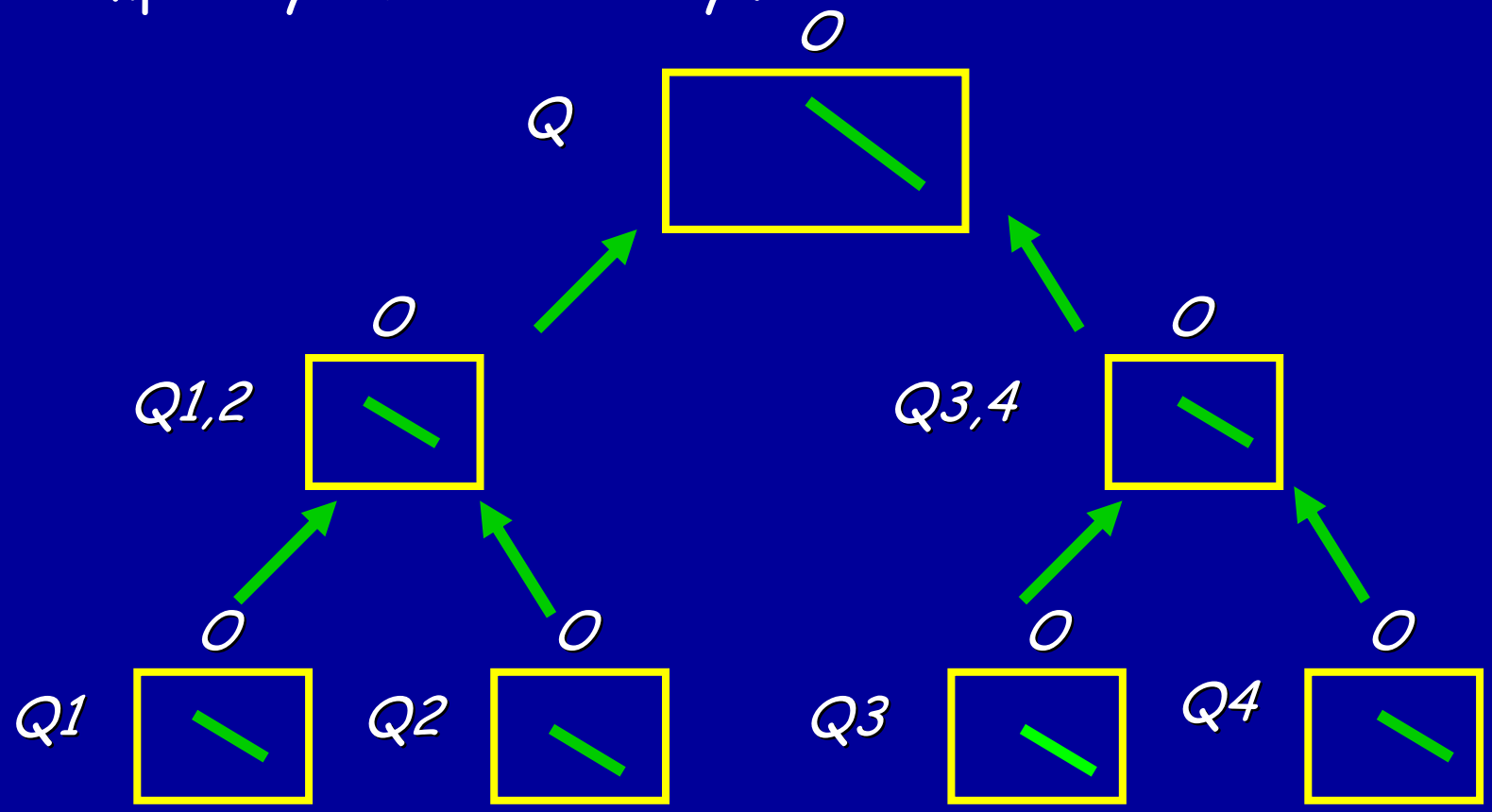
Combine and Extend





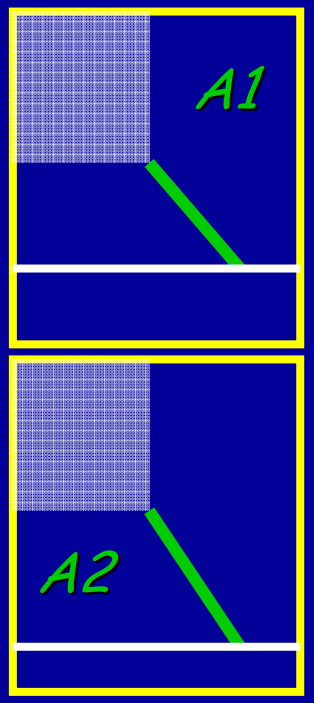
Tree-based Parallel Smith-Waterman

- The tree-based computation is an attractive structure for its simplicity and scalability .



Combine and Extend

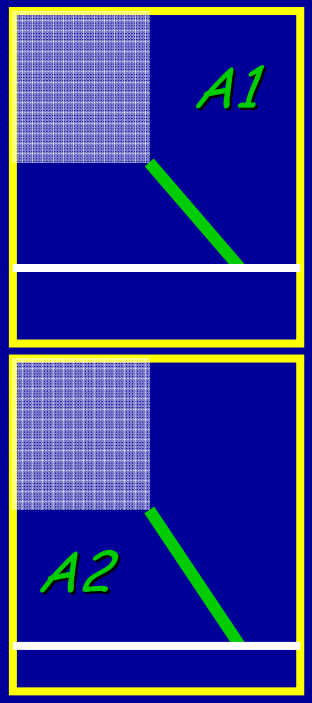
- We don't need the whole matrix to make an extension.
- We can't compute a row if we don't know the previous row.



Store only necessary information in memory .

Parallel Smith-Waterman

- Creating smaller local matrices and then combining them results in loss of sensitivity - accuracy.
- The dependency between the last row of a matrix and the first row of the next one is lost.



Tree-based SW is 100% accurate in a downward extension.

Initial Experiments

- We used 269 sequences of human mitochodrian DNA.
  (Avg. length = ~16000 base pairs).

- The topology was a 1-level deep tree with 4 nodes.

- 5 machines used
  CPU speed : 3GHz
  Memory : 1GB

Initial Experiments

- We performed 269 tests and measured the average sensitivity and runtime of our approach.
- One sequence of our dataset was used as the object sequence and all the others were the query sequences.
- We compared our results with those of the serial Smith-Waterman algorithm which is 100% accurate.
- Sensitivity was measured by comparing the scores produced by the two algorithms.

Results

- Sensitivity : 99.3%
  More testing is needed to verify if this is going to scale.

- Runtime: ~7 sec
  Speedup: 2.43

  We expect this to scale.
  Different tree topologies will result to different speedups.

Future Work

- Applying tree-based SW to longer sequences *e.g a whole chromosome.*
- Consider different types of sequences (more polymorphic).
- Experiment with various topologies.

References

- 1. Fa Zhang, Xiang-Zhen Qiao, Zhi-Yong Liu. A parallel Smith-Waterman Algorithm Based on Divide and Conquer. Engineering in Medicine and Biology . Society, 2004. IEMBS , 26th Annual International Conference of the IEEE.

- 2. Smith TF, Waterman MS (1981). Identification of common molecular subsequences . Journal of Molecular Biology.