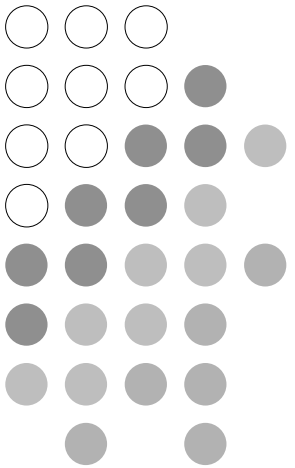
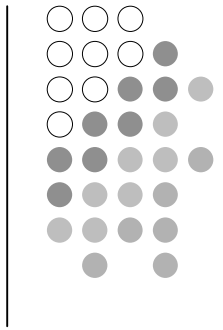
Using a DSM Application to Locally Align DNA Sequences

R. Batista, D. N. Silva, A. Melo, L. Weigang
University of Brasilia (UnB), Brazil



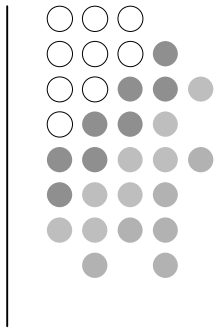
WSDSM 2004 - CCGrid 2004 - April 21st - Chicago, US

Outline



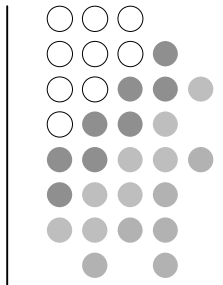
- Introduction
- Algorithm for Local Sequence Alignment
- Scope-consistent DSM systems
- Parallel Algorithm to Compare Biological Sequences
- Experimental Results
- Conclusions and Future Work

Introduction



- The comparison of two genomic sequences is one of the most basic operations in Computational Biology.
- Its goal is to define how similar two sequences are.
  - Local alignment: similarity between two portions of two sequences
  - Global alignment: similarity between the whole sequences.

Introduction



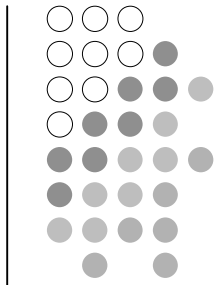
- Smith-Waterman´s Algorithm: one of the most widely used algorithm to compute local alignments.

  ➔Quadratic time and space complexity

- For long sequences, it is a very compute intesive task.

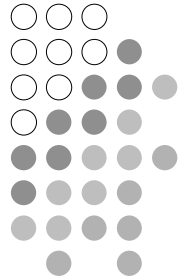
  ➔Parallel processing

Introduction



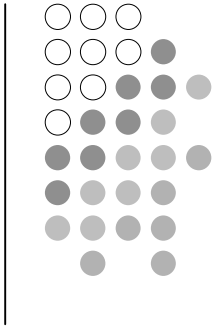
- Distributed Shared Memory (DSM) allows the use of the shared memory programming paradigm in distributed architectures.
- To overcome the coherence overhead of DSM systems, relaxed memory models were proposed.
- JIAJIA is a DSM system that implements the scope consistency memory model (relaxed model)

Goal of the present work



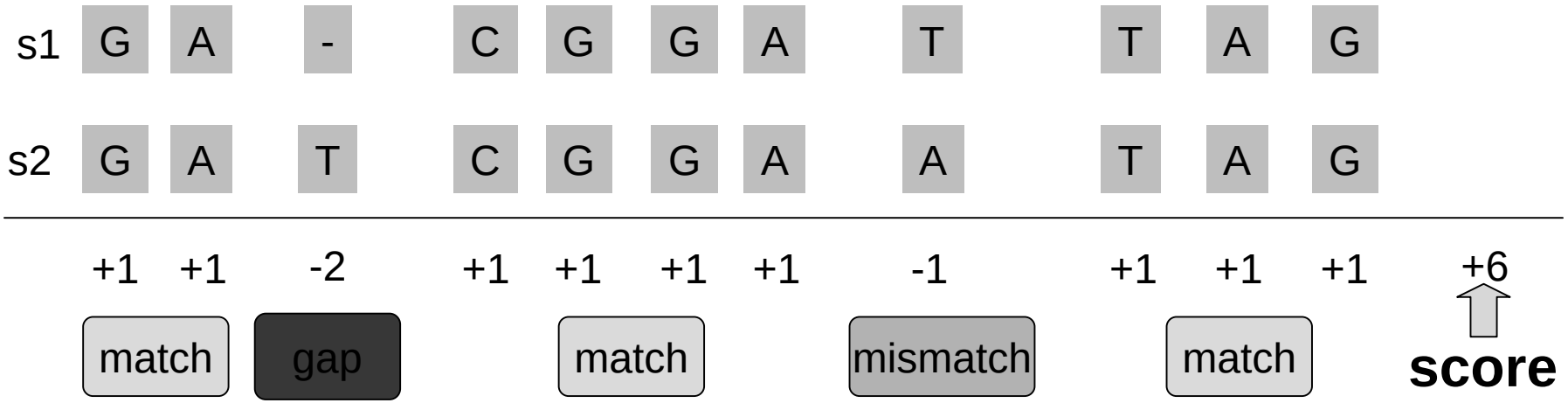
- To propose and evaluate a parallelisation strategy to implement the Smith-Waterman algorithm in a scope-consistent DSM System.
- To compare the results obtained with DSM and MPI implementations
- The space complexity was reduced by the use of an heuristic.

Biological Sequence Alignment

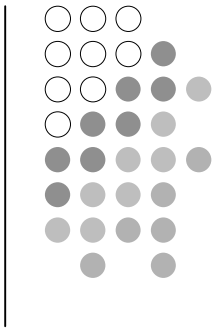


- To find the best alignment between two sequences, one sequence is placed above the other and a score is computed.

G A C G G A T T A G } s1 G A T C G G A A T A G } s2



Smith-Waterman´s Algorithm Local Sequence Alignment



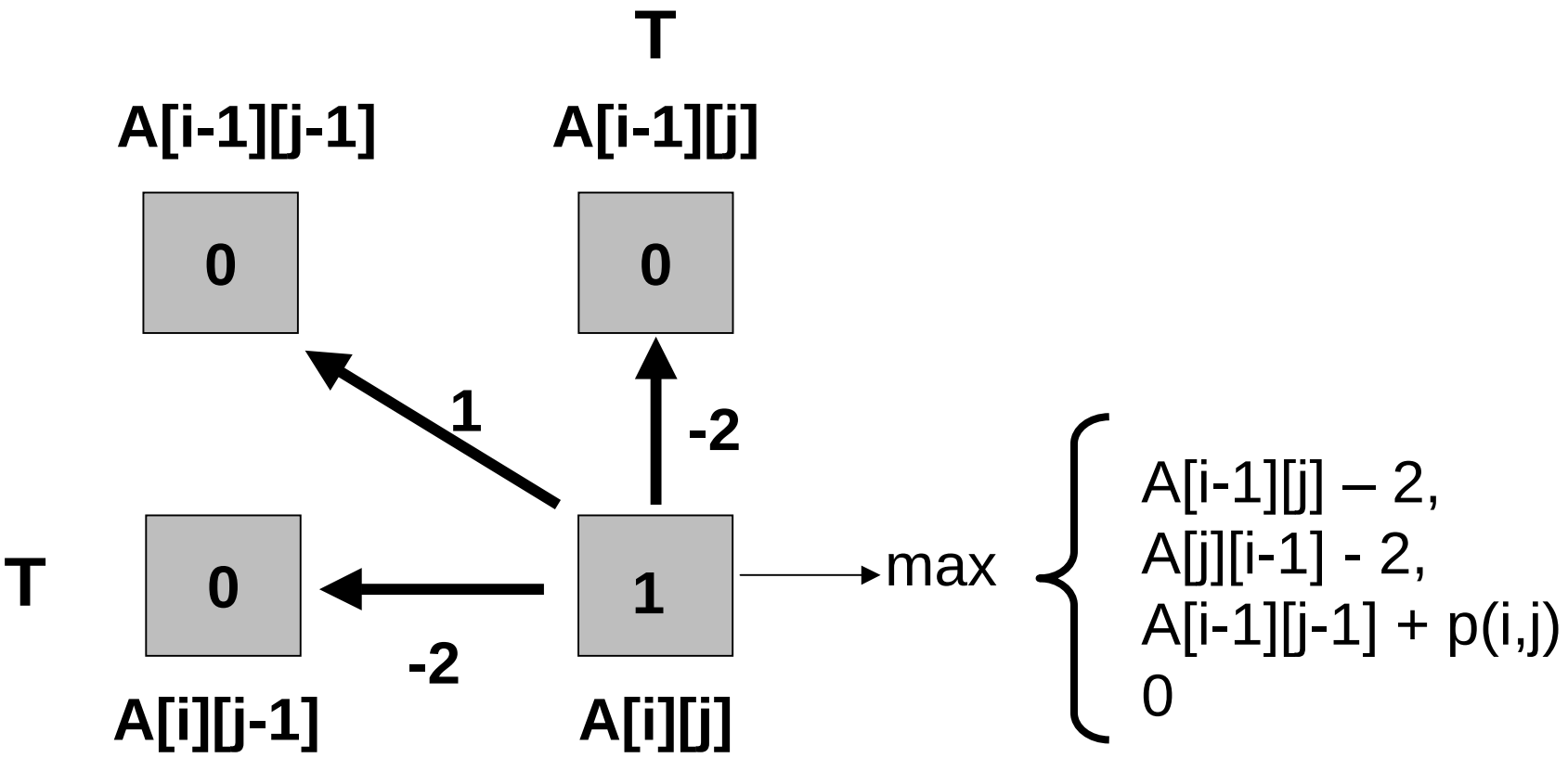
- Based on dynamic programming with quadratic time and space complexity.
- Given two sequences *s* and *t*, where |*s*|=*m* and |*t*|=*n*, an array $A_{m+1,n+1}$ is built using the following equation:

$$sim(s[1..i],t[1..j]) = \max \begin{cases} sim(s[1..i],t[1..j-1]) - 2 \\ sim(s[1..i-1],t[1..j-1]) + p(i,j) \\ sim(s[1..i-1],t[1..j]) - 2 \\ 0. \end{cases}$$

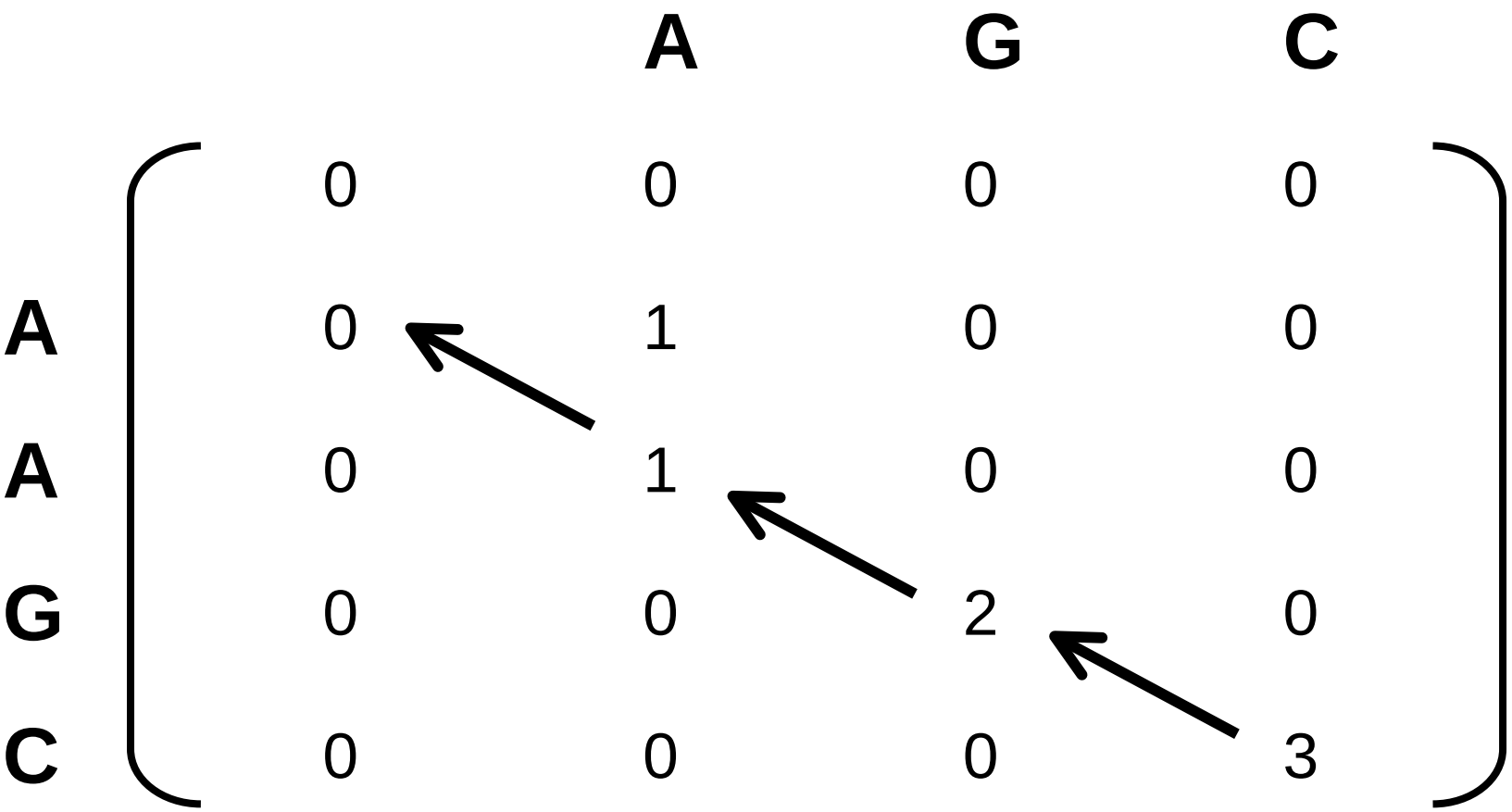
p(i,j) = 1, if s[i] = t[j]
-1, otherwise

To compute each value A[i][j], we need to access A[i-1][j], A[i-1][j-1] and A[i][j-1]

Smith-Waterman´s Algorithm Local Sequence Alignment



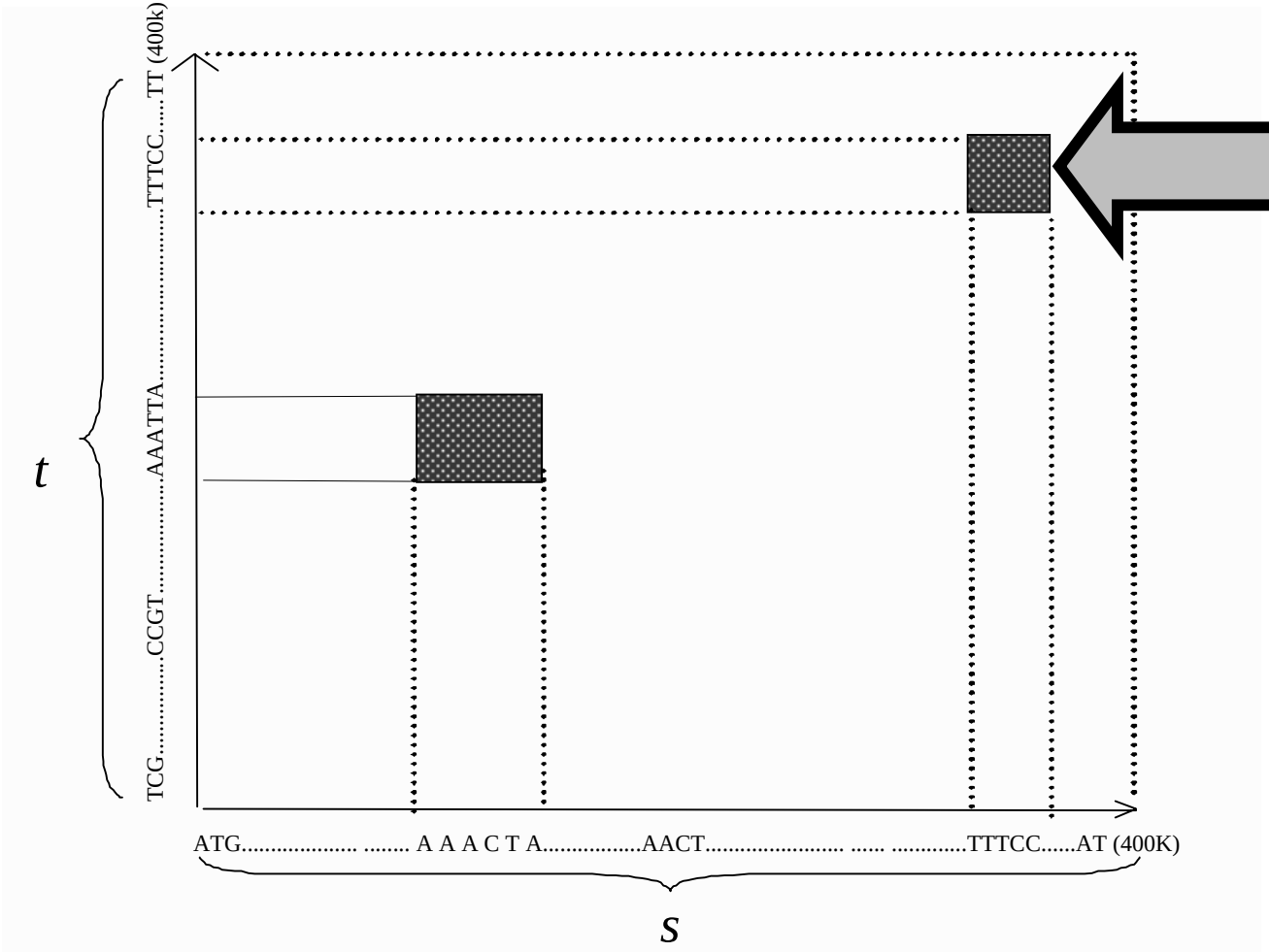
Smith-Waterman´s Algorithm Example



local alignment

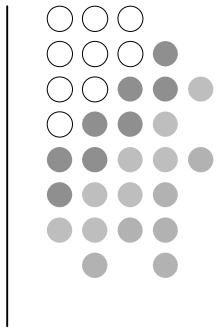
A G C
A G C

Local Sequence Alignment x Global Sequence Alignment



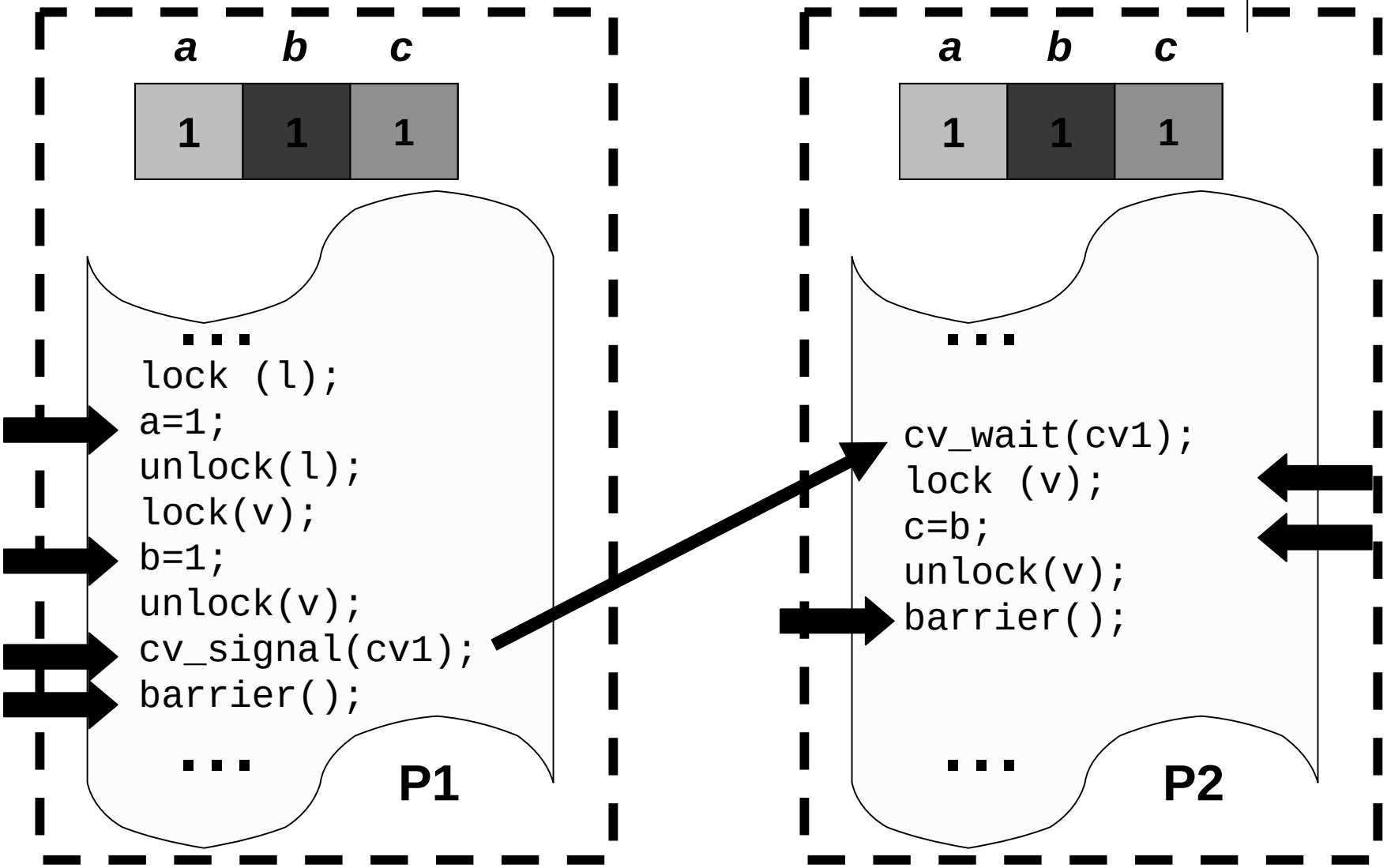
similarity region

Scope Consistent DSM Systems

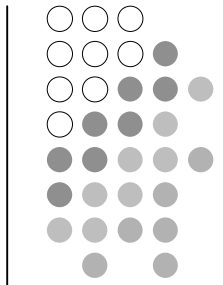


- Scope consistency is a relaxed memory model proposed by Iftode in 1996.
- Synchronisation mechanisms are locks, barriers and condition variables.
- The execution is divided in scopes, which are created in a per-lock basis.
  - Only data inside scope *i* are kept consistent for processors that access that scope.
- Barriers define a global synchronisation point.

Scope Consistent DSM Systems (Example)

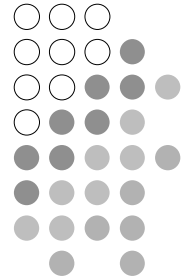


Scope Consistent DSM Systems - JIAJIA



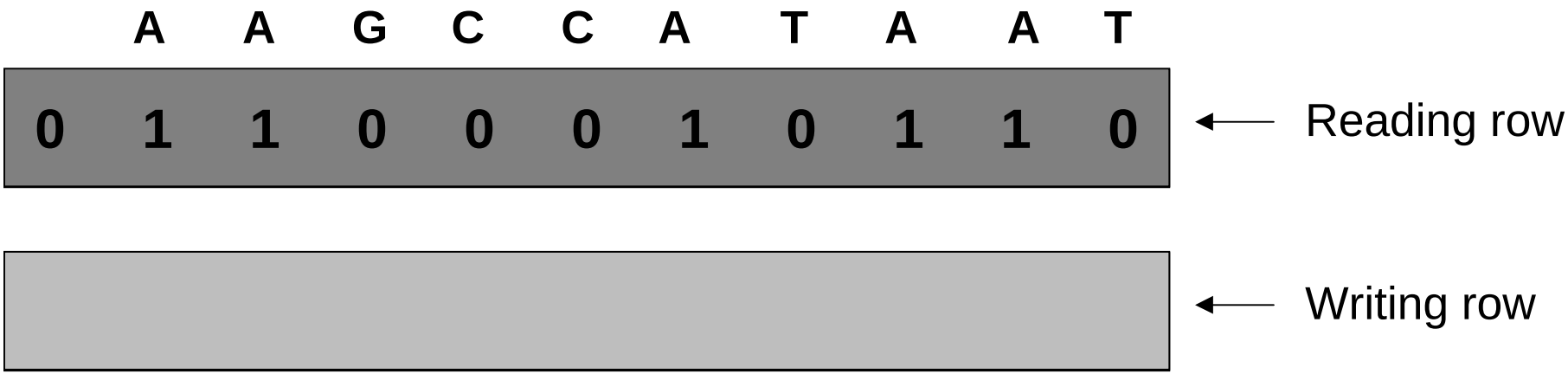
- JIAJIA implements scope consistency with an optimised coherence protocol.
- Lock primitives: jia_lock, jia_unlock
- Condition variable primitives: jia_setcv, jia_waitcv
- Barrier primitive: jia_barrier

Parallel Algorithm to Compare sequences



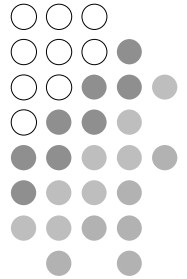
- As our goal is to compare long sequences in clusters, quadratic space complexity is prohibitive.
  - For instance, to compare two 400KB sequences, we would need 1.6 TB.
- For this reason, we used an heuristic that works only with two rows, reducing space complexity to O(n).

Parallel Algorithm to Compare sequences



- A data structure is used to keep the following information:
  - Current score,
  - Initial and final alignment coordinates,
  - Maximal and minimal score,
  - Gaps, matches and mismatches counters
- Thus, information of A[i-1][j], A[i-1][j-1] and A[i][j-1] is passed to A[i][j]

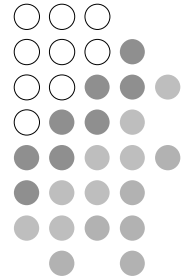
Parallel Algorithm to Compare DNA sequences



- The access pattern presented by the algorithm leads to a non-uniform amount of parallelism.

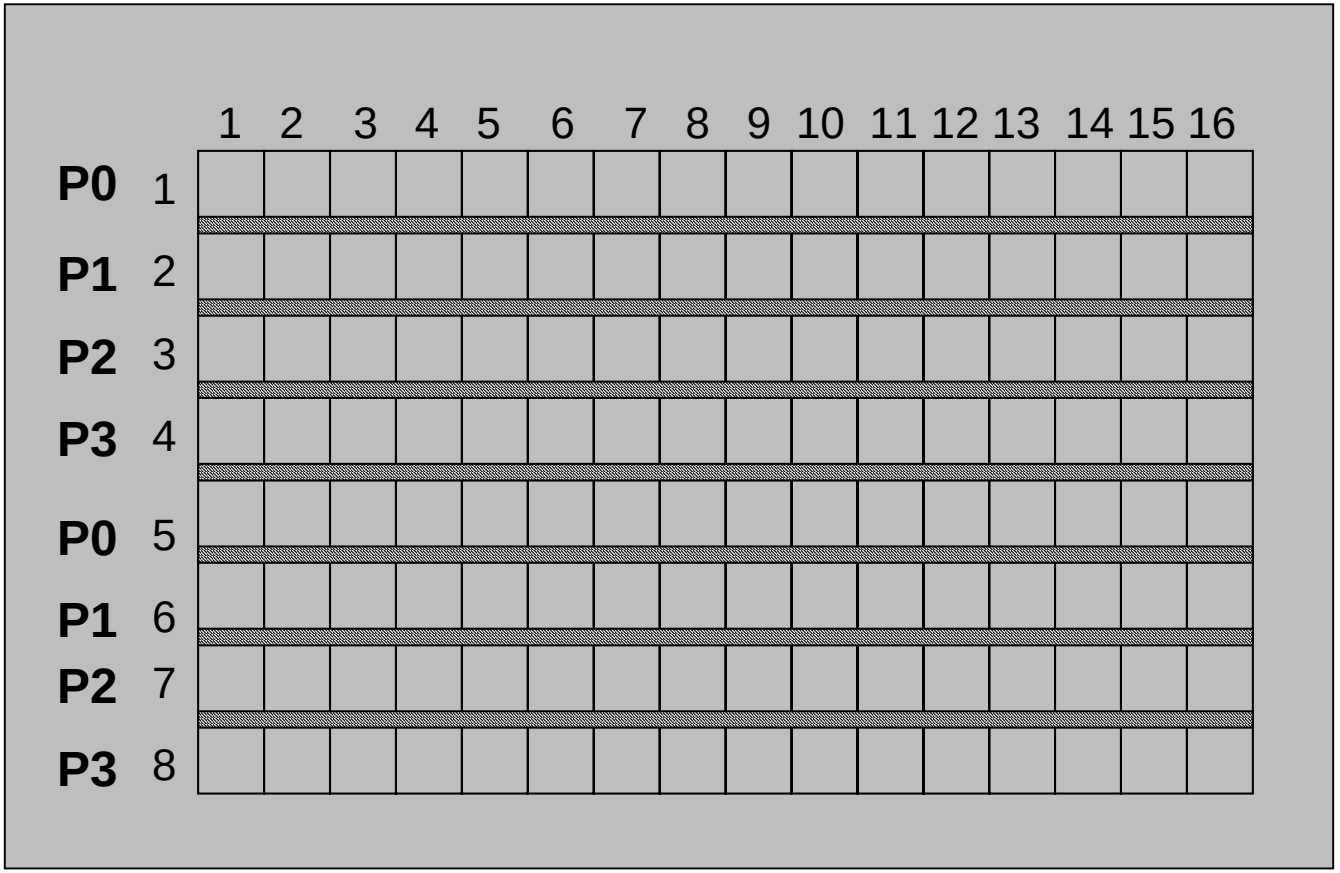
		A	G	C
	0	0	0	0
A	0	1	0	0
A	0	1	0	0
G	0	0	2	0
C	0	0	0	3

Parallel Algorithm to Compare sequences



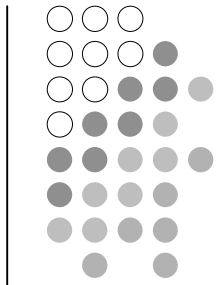
- Each processor *p* acts on two rows, a reading row and a writing row
- Work is assigned in a column basis.
- Synchronisation is achieved by locks and condition variables
- Barriers are only used at the beginning and at the end of computation.

Parallel Algorithm to Obtain Local Alignments



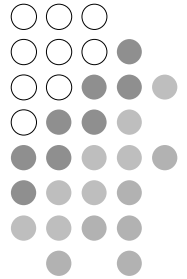
16x8 blocking factor for 4 processors

Experimental Results



- The algorithm was implemented in C, using JIAJIA v2.1 and mpich 1.2.4 on top of Debian Linux 2.1.
- Experiments were run on a dedicated cluster of 8 Pentium II 350MHz, 160MB RAM connected by a 100Mbps Ethernet switch.
- We used real DNA sequences obtained from *www.ncbi.nlm.nih.gov/PMGifs/Genomes*.

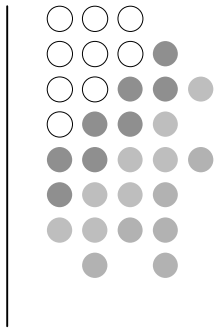
Experimental Results
Defining the block factor



- Execution times to align 50K sequences with 8 processors, with JIAJIA

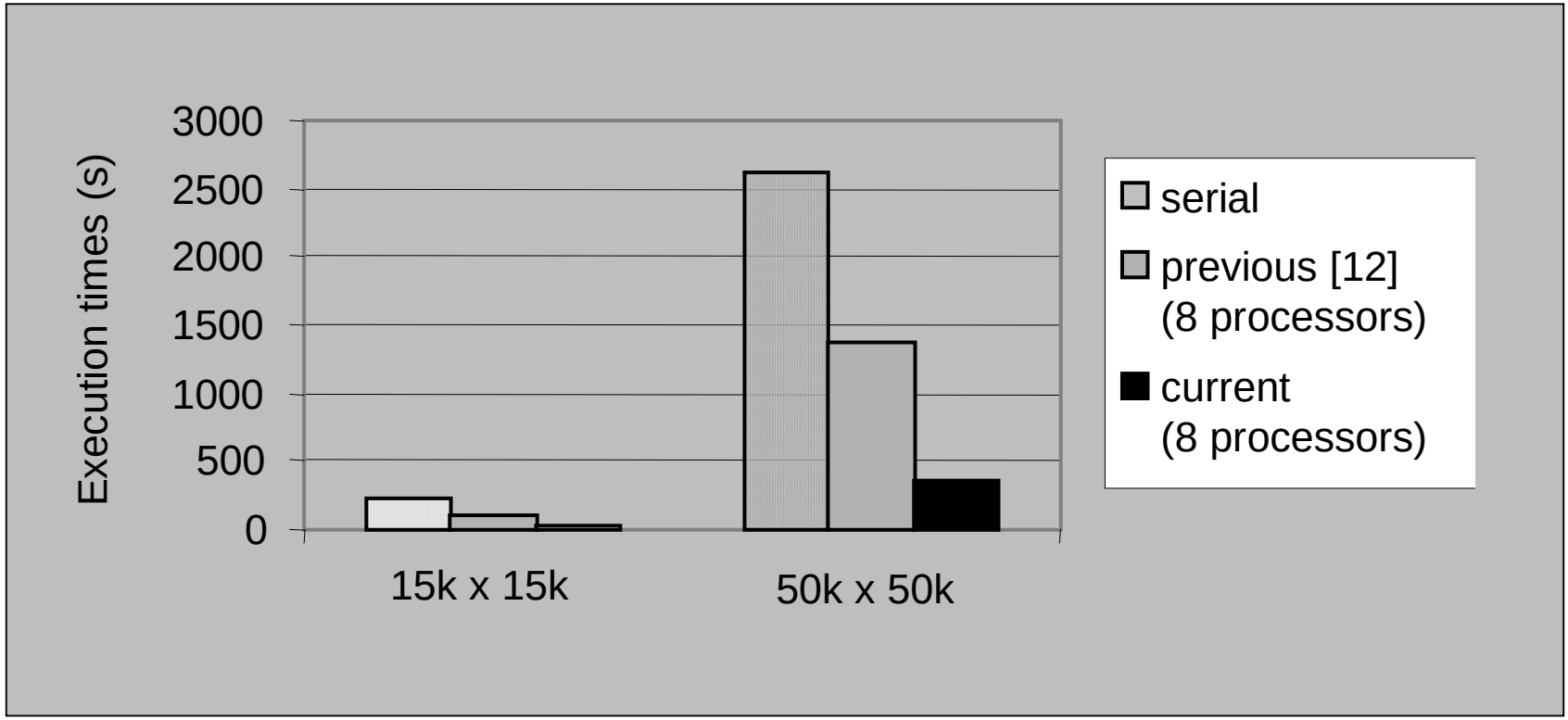
Blocking	Time(s)	Performance gain (relative to 1x1)
1 x 1	732.79	0%
2 x 2	459.80	59%
3 x 3	394.59	85%
4 x 4	368.15	99%
5 x 5	363.13	101%

Experimental Results Execution Times(s) / Speedups (40bandsx40blocks)

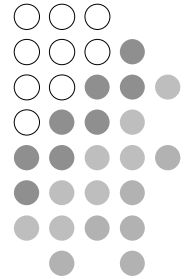


Size	Serial Exec	2 proc Exec /Speedup	4 proc Exec /Speedup	8 proc Exec /Speedup
8K x 8K	57.18	38.59/1.48	21.18/2.69	12.55/4.55
15K x 15K	226.51	130.22/1.73	67.42/3.35	36.51/6.20
50K x 50K	2620.64	1352.76/1.93	701.95/3.73	363.13/7.28

Execution times (8 processors) - current and previous

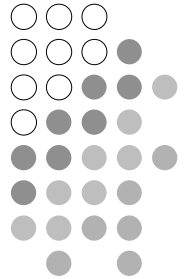


JIAJIA and MPICH (5x5 blocking factor)



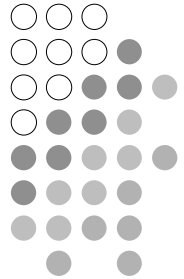
Processors	15Kx15K		8Kx8K	
	JIAJIA Time(s)/speedup	MPICH Time(s)/speedup	JIAJIA Time(s)/speedup	MPICH Time(s)/speedup
1	238.58/1	239.10/1	58.23/1	58.01/1
2	120.57/1.97	119.01/2.00	32.06/1.81	30.62/1.89
4	62.67/3.80	64.54/3.70	18.16/3.20	18.78/3.08
8	36.51/6.53	36.26/6.59	12.56/4.63	8.64/6.71

Conclusions



- We proposed and evaluated a DSM implementation of a variant of the Smith-Waterman algorithm for biological sequence alignment.
- The results obtained in an 8-machine cluster presented very good speedups.
- For 15K DNA sequences, results obtained with JIAJIA and mpich were very similar

Future work



- We intend to evaluate na alternative approach that uses variable block size.
- The mpi strategy will be ported to mpich-g in order to compare very long DNA sequences (3M sequences) in a grid system.