Multiparty Finite Computations

T. Jurdziński (Wrocław University)

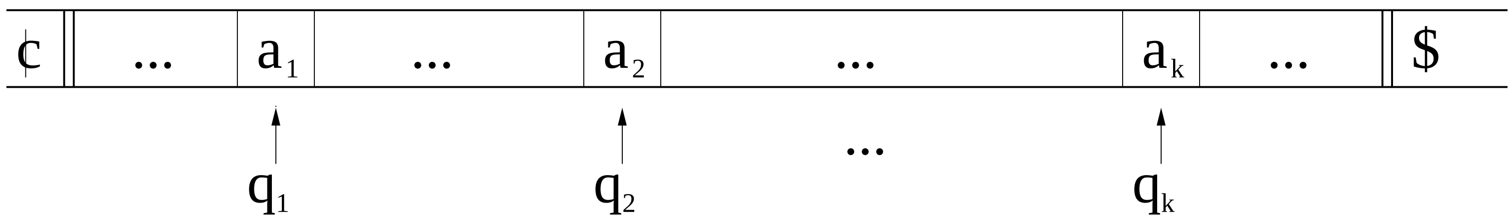
M. Kutyłowski (Wrocław University)

K. Loryś (Wrocław University)

some results in cooperation

with P. Ďuriš (Comenius University, Bratislava)

Multihead Finite Automata: *cooperation*

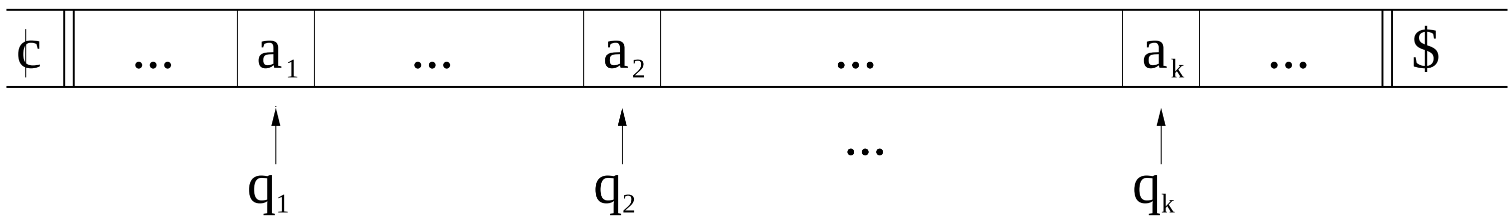


- each automaton may send a message after each step
- transition function of a single automaton depends on the input symbol currently seen and messages sent by the other automata

$$(q_i', r) = \delta(q_i, a, m_1, \ldots, m_{i-1}, m_{i+1}, \ldots, m_k)$$

where m_j is the message sent by the jth automaton after previous step.

Multiprocessor Finite Automata: *common master*



- Each processor is a **genuine** finite automaton, it does not receive any messages

- **The master**: after each step inspects the states of all processors and determines which processors are **frozen** and **active**:

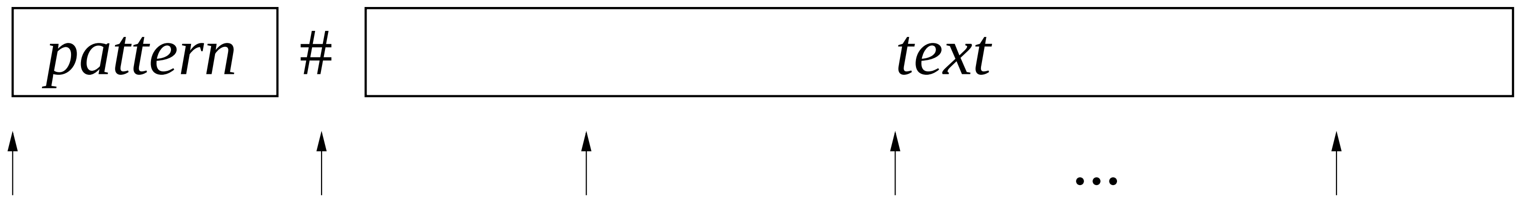
$$h(i, q_1, \ldots, q_k) \in \{\text{ACTIVE}, \text{FROZEN}\}$$

  where i is the processor number and $q_1, \ldots, q_k$ – the states of all processors

Classical problems

Sensing sensing vs *non-sensing* heads (Duris, Galil – 1995)

Pattern matching by one-way, deterministic *sensing* automata (Jiang, Li – 1993)



Hardware the number of automata vs computational power (Yao, Rivest – 1976; Monien – 1980)

New problems

Means of cooperation :

Cooperation means vs computational power.

Computational power of *multiprocessor* vs *multihead* systems

Communication :

Communication volume vs computational power

Message complexity classes

$MESSAGE_l(k(n))$
– the languages that may be recognized by systems of l automata where at most $k(n)$ messages are sent for every input word of length n

$MESSAGE(k) = \bigcup_l MESSAGE_l(k)$

Motivations

1. "Communication" Complexity
   - Does the weaker mode of communication between devices decrease their computational power?
   - How the limitation of communication size influence the power of devices?
   - Communication complexity of problems in the case, when other resources are bounded.
2. Computation on sequential devices vs computation on devices with random access to memory (example: complexity of matrix transposition)

Obvious properties

1. k heads are at least as powerful as k processors (coordinated by a master)

2. $\{a^n b^n : n \in \mathbb{N}\}$ can be recognized with 2 one-way processors, by 2 one-way heads with a single message,
   (not a regular language!)

3. $\{a^n b^n c^n : n \in \mathbb{N}\}$ can be recognized by 2 one-way heads
   (not a context free language!)

4. $\bigcup_{k=1}^{\infty} \mathbf{2}\text{-}Y\mathbf{H}(k) = Y\mathbf{LOGSPACE}$ for $Y \in \{\mathbf{N}, \mathbf{D}\}$

Results: Heads vs Processors

1. **Simulations**
   - X-$\mathbf{NH}(k) = X$-$\mathbf{NP}(k)$ for every $k \in \mathbb{N}$ and $X \in \{1,2\}$
   - X-$\mathbf{DH}(k) \subseteq X$-$\mathbf{DP}(k+2)$ for every $k \in \mathbb{N}$ and $X \in \{1,2\}$
2. **Separation**
   - 1-$\mathbf{DP}(2) \subsetneq$ 1-$\mathbf{DH}(2)$
3. **Closure properties**
   - $\bigcup_{k=1}^{\infty}$ **2-NP**(k) and $\bigcup_{k=1}^{\infty}$ **2-DP**(k) are closed under complement
   - $\bigcup_{k=1}^{\infty}$ 1-**NP**(k) is **not** closed under complement
   - 1-**DP**(k) is closed under complement for every $k > 2$

Results: Size of Communication

1. The hierarchy for the communication of constant size:
   (a) $MESSAGE(k-1) \subsetneq MESSAGE(k)$ for *one-to-one* model
   (b) $MESSAGE(k-1) \subsetneq MESSAGE(k+l-1)$ for *on-bus* model
2. probably a gap between $MESSAGE(O(1))$ and $MESSAGE(\Theta(\log n))$.
3. The languages which have the complexity of communication between $\log n$ and n form quite a dense hierarchy.
4. There are languages in $MESSAGE(\Theta(n))$.
5. Matrix multiplication requires $n^{3/2}/\log^{2+\varepsilon} n$ messages.
   (easy proof, also follows from two-party communication of branching programs)

Separation between *heads* and *processors*

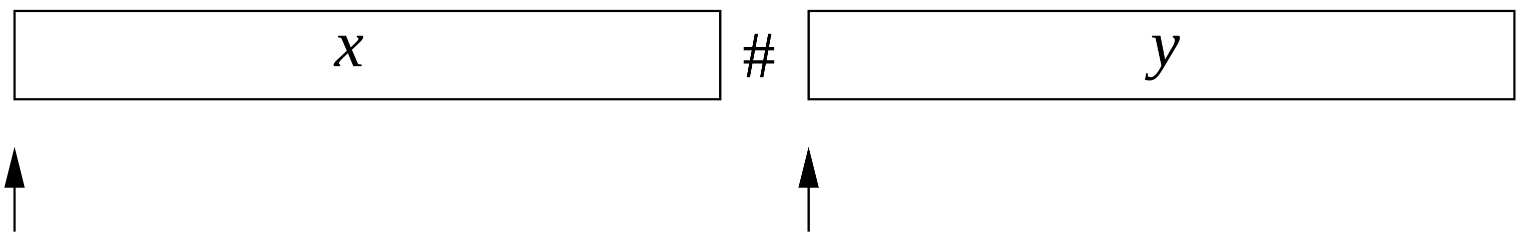
Theorem For $|\Sigma| \geq 3$ the language

$$L_P = \{x\#y : \ x,y \in \Sigma^*, \ |x| = |y|, \ p(x,y) = 1\}$$

separates classes 1-$\mathbf{DP}(2)$ and 1-$\mathbf{DH}(2)$, where $p(x,y)$ is the parity of the number of positions on which x and y differ.

How to recognize L_P by two-head one-way deterministic automaton:

- heads scan x and y simultaneously and synchronously
- one of the heads changes its states between ODD and EVEN on every position in which x and y differ



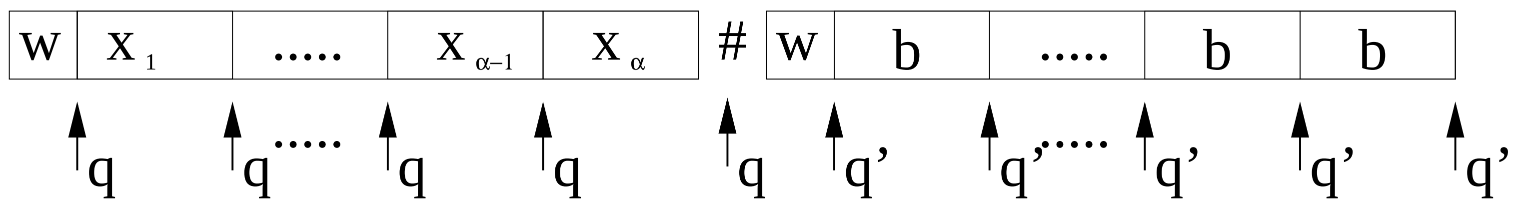
Separation: ***a sketch***

Why two processors cannot recognize L_P (intuition):

1. Processors have to compare x and y simultaneously.
2. The only possibility to recognize current parity of the number of differences is to "desynchronize" processors on x and y
3. For appropriately constructed input words it is necessary to increase or decrease (monotonically) difference between positions of processors on x and y
4. The difference between positions of processors comparing appropriately constructed words x and y will grow too much contradicting 1.

Separation: *details*

The set of words W on which we cheat the automaton:

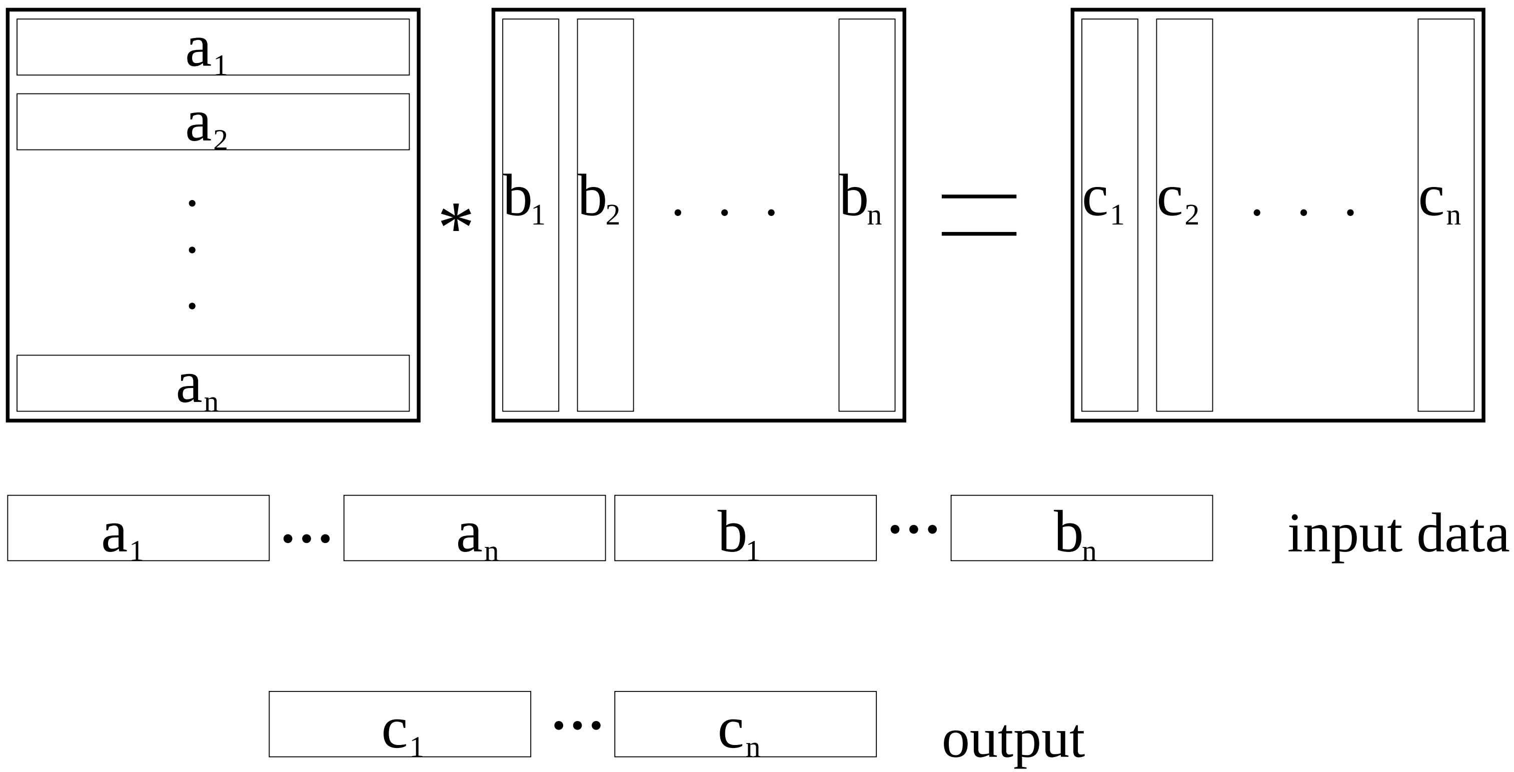


- $x_i = a_1$ or $x_i = a_2$ for every i
- $|a_1| = |a_2| = |b| = n$
- $p(a_1, b) \neq p(a_2, b) = 0$
- $K(a_i|b) \geq n - c\log n, \quad K(b|a_i) \geq n - c\log n$
- both processors start and finish computation on every x_i and on b in the same state

Systems of finite automata which compute functions

1. The read-only tape includes the arguments of the function
2. The result of the function is written on one-way write-only tape with one head
3. Example: The representation of matrix multiplication problem: arguments are stored as a sequence of rows (the first matrix) and a sequence of columns (the second matrix)

Matrix multiplication problem



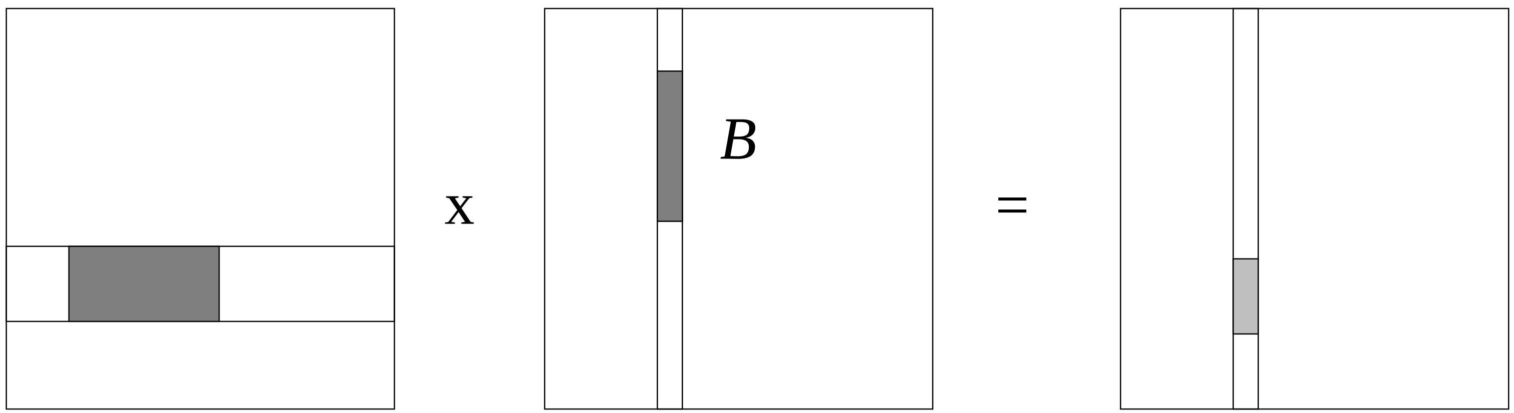
The lower bound for the matrix multiplication

Theorem 1 *The matrix multiplication function requires* $\Omega(N^{3/2}/\log^{2+\varepsilon} N)$ *messages for every* $\varepsilon > 0$.
($O(N^{3/2})$ *suffice by the straightforward algorithm.)*

Sketch of the proof

1. We consider matrices with large Kolmogorov complexity
2. We divide the computation on the stages. One stage consists of the part of computation in which $s(n)$ bits of the result are written
3. The amount of communication in one stage is “almost” $(n/s(n))$ when $s(n) = \omega(\log n)$

Matrix multiplication - one stage



Matrix multiplication - one stage

1. During the stage we multiply the submatrix A by the vector B
2. Let B' be the longest subword of B on which no message was sent
3. Let A' be the submatrix of A corresponding to the block b'
4. A' has a big non-singular submatrix A_N,
   the product with an appropriate subvector b' of b is uniquely determined without knowledge of b'
5. we encode b' as one of the vectors giving this result while multiplied with A_N

Conclusions and open problems

Conclusion: multiprocessor automata have **similar** computational power to multihead automata, but in some cases heads are **more powerful** than processors

Conclusion: The amount of communication substantially influence the power of systems of finite automata.

Problem: $1\text{-}\mathbf{DP}(k) \overset{?}{\subsetneq} 1\text{-}\mathbf{DH}(k)$ for $k > 2$, $2\text{-}\mathbf{DP}(k) \overset{?}{\subsetneq} 2\text{-}\mathbf{DH}(k)$ for $k > 1$

Problem: Are there any languages which require a non-constant, sub-logarithmic number of messages?

Problem: a superlinear lower bound on the communication size for a decision problem.