Deterministic k-Local Tree Automata and their Work-optimal Parallel Run

Štěpán Plachý Jan Janoušek

Faculty of Information Technology
Czech Technical University in Prague

London Stringology Days & London Algorithmic Workshop
8. February 2019

Overview

Synchronizing word

- Property of deterministic finite string automata
- Reading such word from any configuration puts the automaton in a well defined state

input		a	b	**a**	**b**	**c**	c	a	**a**	**b**	**c**	a	c	b	**a**	**b**	**c**	b	a	c	**a**	**b**	**c**
state	0	1	2	1	2	**3**	0	1	1	2	**3**	1	0	0	1	2	**3**	0	1	0	1	2	**3**

k-locality

- Stronger property
- All words of length at least k are synchronizing
- Property of pattern matching automata
  - k is the length of the pattern

k-local DFA run parallelization

Option 1

- Divide input to processors
- Start run k letters in advance of each block from an arbitrary state

k-local DFA run parallelization

Option 2

- Divide input into blocks of length k
- On each block run twice starting from an arbitrary state

Finite tree automata (bottom-up)

- Formalism for accepting regular tree languages
  - Rooted, ordered and labeled trees
  - Degree of a node is determined by its symbol
- Run is a bottom-up computation on trees with constant bounded degree

DFTA run

$A = (Q, \mathcal{F}, Q_f, \Delta)$

$Q = \{0, 1, 2, 3, 4\}$ (States)

$\mathcal{F} = \{a_2, b_1, c_0\}$ (Alphabet)

$Q_f = \{4\}$ (Final states)

Δ: (Transition function)

$c \rightarrow 1$

$b\,(0) \rightarrow 2$

$b\,(1) \rightarrow 3$

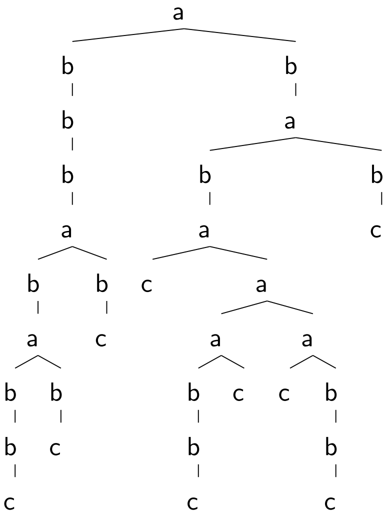
$b\,(2) \rightarrow 2$

$b\,(3) \rightarrow 2$

$a\,(2, 3) \rightarrow 4$

$a\,(3, 3) \rightarrow 4$

$a\,(q_1, q_2) \rightarrow 0$ otherwise



DFTA run

$A = (Q, \mathcal{F}, Q_f, \Delta)$

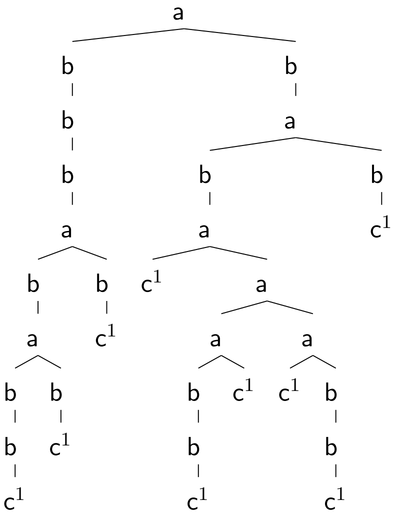
$Q = \{0, 1, 2, 3, 4\}$ (States)

$\mathcal{F} = \{a_2, b_1, c_0\}$ (Alphabet)

$Q_f = \{4\}$ (Final states)

Δ: (Transition function)

c	$\rightarrow 1$
$b\,(0)$	$\rightarrow 2$
$b\,(1)$	$\rightarrow 3$
$b\,(2)$	$\rightarrow 2$
$b\,(3)$	$\rightarrow 2$
$a\,(2,3)$	$\rightarrow 4$
$a\,(3,3)$	$\rightarrow 4$
$a\,(q_1, q_2)$	$\rightarrow 0$ otherwise



DFTA run

$A = (Q, \mathcal{F}, Q_f, \Delta)$

$Q = \{0, 1, 2, 3, 4\}$ (States)

$\mathcal{F} = \{a_2, b_1, c_0\}$ (Alphabet)

$Q_f = \{4\}$ (Final states)

Δ: (Transition function)

$c \rightarrow 1$

$b\,(0) \rightarrow 2$

$b\,(1) \rightarrow 3$

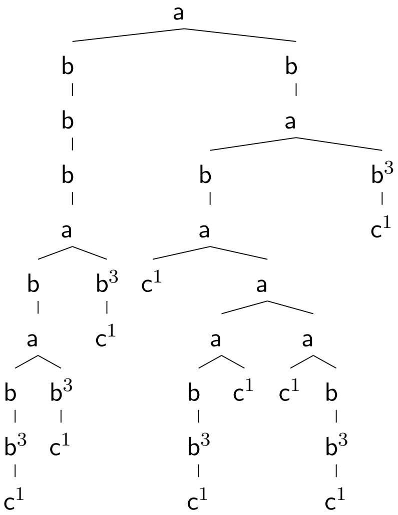
$b\,(2) \rightarrow 2$

$b\,(3) \rightarrow 2$

$a\,(2,3) \rightarrow 4$

$a\,(3,3) \rightarrow 4$

$a\,(q_1, q_2) \rightarrow 0$ otherwise



DFTA run

$A = (Q, \mathcal{F}, Q_f, \Delta)$

$Q = \{0, 1, 2, 3, 4\}$ (States)

$\mathcal{F} = \{a_2, b_1, c_0\}$ (Alphabet)

$Q_f = \{4\}$ (Final states)

Δ: (Transition function)

$c \to 1$

$b\,(0) \to 2$

$b\,(1) \to 3$

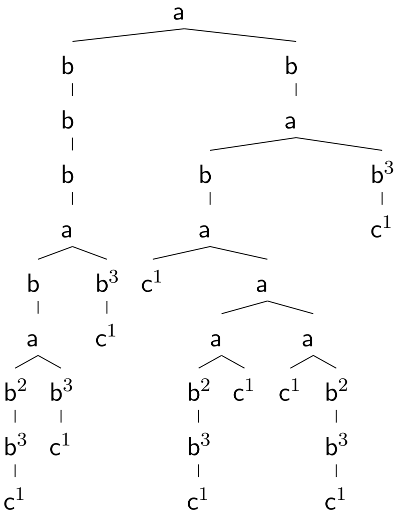
$b\,(2) \to 2$

$b\,(3) \to 2$

$a\,(2,3) \to 4$

$a\,(3,3) \to 4$

$a\,(q_1, q_2) \to 0$ otherwise



DFTA run

$A = (Q, \mathcal{F}, Q_f, \Delta)$

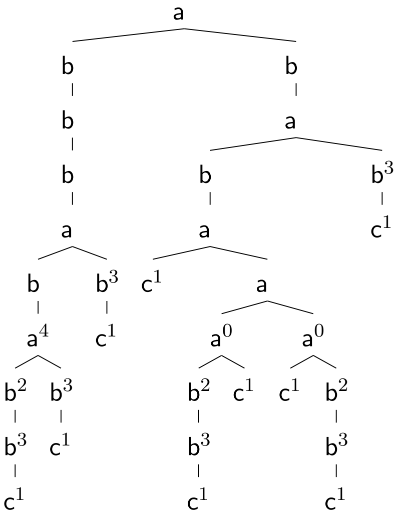
$Q = \{0, 1, 2, 3, 4\}$ (States)

$\mathcal{F} = \{a_2, b_1, c_0\}$ (Alphabet)

$Q_f = \{4\}$ (Final states)

Δ: (Transition function)

c	$\rightarrow 1$
$b\,(0)$	$\rightarrow 2$
$b\,(1)$	$\rightarrow 3$
$b\,(2)$	$\rightarrow 2$
$b\,(3)$	$\rightarrow 2$
$a\,(2,3)$	$\rightarrow 4$
$a\,(3,3)$	$\rightarrow 4$
$a\,(q_1, q_2)$	$\rightarrow 0$ otherwise



DFTA run

$A = (Q, \mathcal{F}, Q_f, \Delta)$

$Q = \{0, 1, 2, 3, 4\}$ (States)

$\mathcal{F} = \{a_2, b_1, c_0\}$ (Alphabet)

$Q_f = \{4\}$ (Final states)

Δ: (Transition function)

$c \rightarrow 1$

$b\,(0) \rightarrow 2$

$b\,(1) \rightarrow 3$

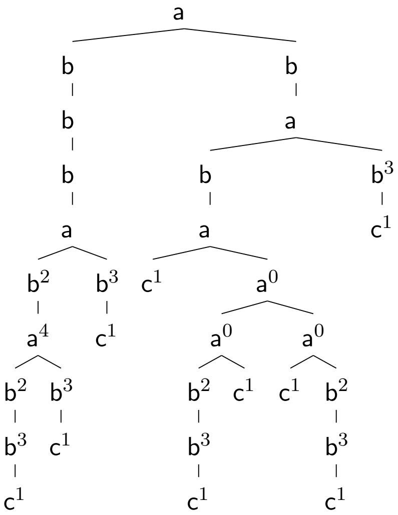
$b\,(2) \rightarrow 2$

$b\,(3) \rightarrow 2$

$a\,(2, 3) \rightarrow 4$

$a\,(3, 3) \rightarrow 4$

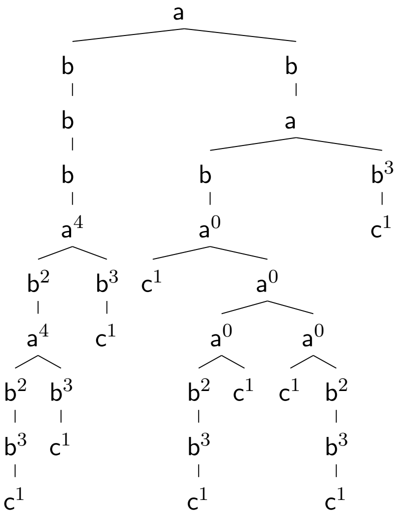
$a\,(q_1, q_2) \rightarrow 0$ otherwise



DFTA run

$A = (Q, \mathcal{F}, Q_f, \Delta)$
$Q = \{0, 1, 2, 3, 4\}$ (States)
$\mathcal{F} = \{a_2, b_1, c_0\}$ (Alphabet)
$Q_f = \{4\}$ (Final states)
Δ: (Transition function)

$c \rightarrow 1$
$b\,(0) \rightarrow 2$
$b\,(1) \rightarrow 3$
$b\,(2) \rightarrow 2$
$b\,(3) \rightarrow 2$
$a\,(2, 3) \rightarrow 4$
$a\,(3, 3) \rightarrow 4$
$a\,(q_1, q_2) \rightarrow 0$ otherwise



DFTA run

$A = (Q, \mathcal{F}, Q_f, \Delta)$

$Q = \{0, 1, 2, 3, 4\}$ (States)

$\mathcal{F} = \{a_2, b_1, c_0\}$ (Alphabet)

$Q_f = \{4\}$ (Final states)

Δ: (Transition function)

$c \rightarrow 1$

$b\,(0) \rightarrow 2$

$b\,(1) \rightarrow 3$

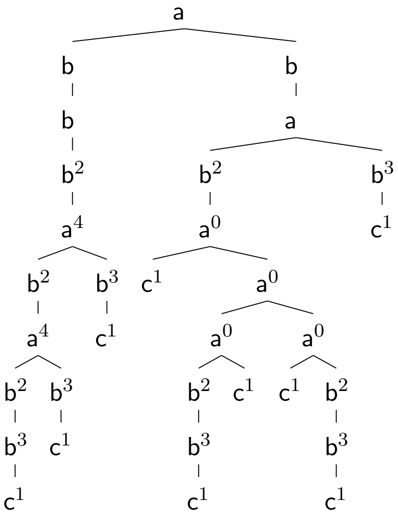
$b\,(2) \rightarrow 2$

$b\,(3) \rightarrow 2$

$a\,(2,3) \rightarrow 4$

$a\,(3,3) \rightarrow 4$

$a\,(q_1, q_2) \rightarrow 0$ otherwise



DFTA run

$A = (Q, \mathcal{F}, Q_f, \Delta)$

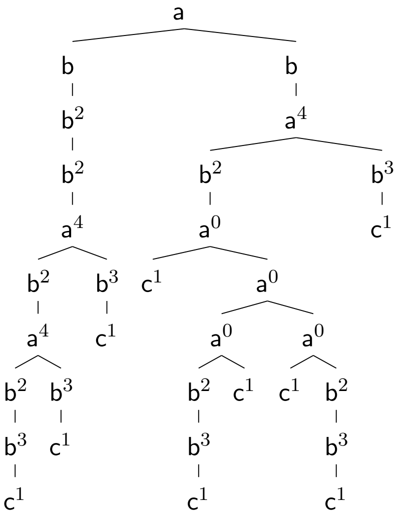
$Q = \{0, 1, 2, 3, 4\}$ (States)

$\mathcal{F} = \{a_2, b_1, c_0\}$ (Alphabet)

$Q_f = \{4\}$ (Final states)

Δ: (Transition function)

$$
\begin{aligned}
c &\rightarrow 1\\
b\,(0) &\rightarrow 2\\
b\,(1) &\rightarrow 3\\
b\,(2) &\rightarrow 2\\
b\,(3) &\rightarrow 2\\
a\,(2,3) &\rightarrow 4\\
a\,(3,3) &\rightarrow 4\\
a\,(q_1, q_2) &\rightarrow 0 \text{ otherwise}
\end{aligned}
$$



DFTA run

$A = (Q, \mathcal{F}, Q_f, \Delta)$

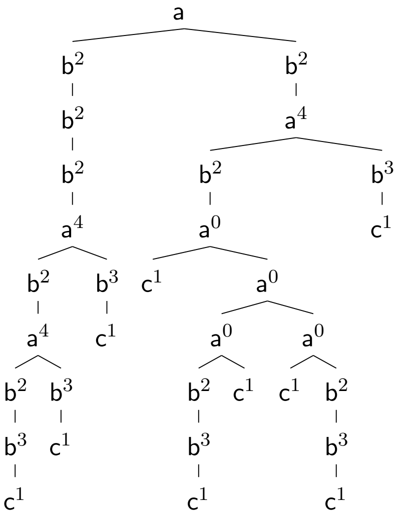
$Q = \{0, 1, 2, 3, 4\}$ (States)

$\mathcal{F} = \{a_2, b_1, c_0\}$ (Alphabet)

$Q_f = \{4\}$ (Final states)

Δ: (Transition function)

$$
\begin{array}{ll}
c & \to 1 \\
b\,(0) & \to 2 \\
b\,(1) & \to 3 \\
b\,(2) & \to 2 \\
b\,(3) & \to 2 \\
a\,(2,3) & \to 4 \\
a\,(3,3) & \to 4 \\
a\,(q_1, q_2) & \to 0 \text{ otherwise}
\end{array}
$$



DFTA run

$A = (Q, \mathcal{F}, Q_f, \Delta)$

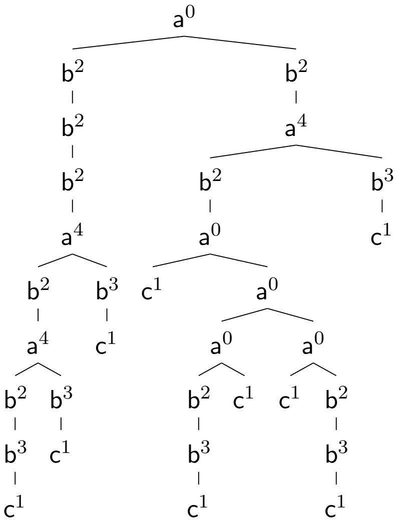
$Q = \{0, 1, 2, 3, 4\}$ (States)

$\mathcal{F} = \{a_2, b_1, c_0\}$ (Alphabet)

$Q_f = \{4\}$ (Final states)

Δ: (Transition function)

$$
\begin{aligned}
c &\rightarrow 1 \\
b\,(0) &\rightarrow 2 \\
b\,(1) &\rightarrow 3 \\
b\,(2) &\rightarrow 2 \\
b\,(3) &\rightarrow 2 \\
a\,(2,3) &\rightarrow 4 \\
a\,(3,3) &\rightarrow 4 \\
a\,(q_1, q_2) &\rightarrow 0 \text{ otherwise}
\end{aligned}
$$



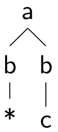
Synchronizing tree

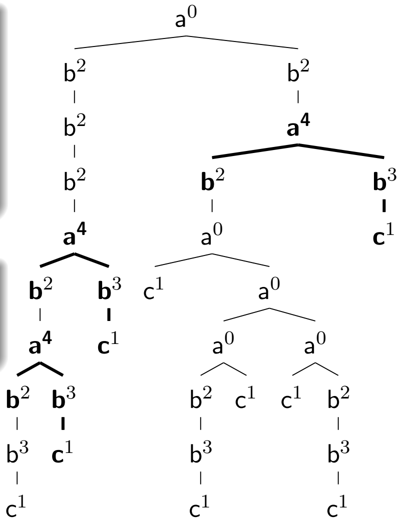
Definition

A tree pattern is *synchronizing* for a DFTA if run on the pattern with variables substituted by any tree ends in the same state.

Remark

All tree patterns without variables are synchronizing.





k-locality

Definition

A DFTA is called k-*local* if all trees with variables in depth at least k are synchronizing

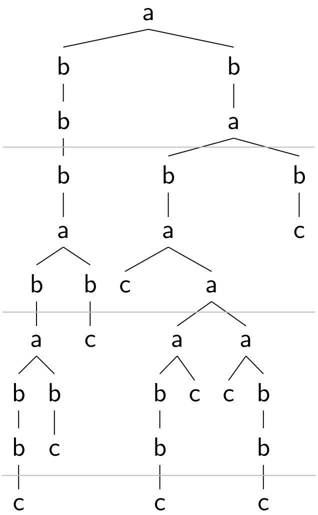
Theorem

All tree pattern matching automata are k-local, where k is the height of the pattern.

Work optimal parallel run

Basic idea

- Divide input tree into segments of k levels
- Assign each node an arbitrary state
- Perform bottom-up computation twice over each segment



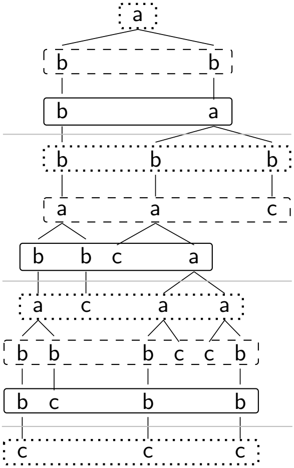
Work optimal parallel run

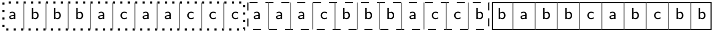
Algorithm overview

- Basic steps:
  1. Computation of a linear order of nodes
  2. Parallel traversal of ordered nodes
- Runs in $\mathcal{O}(\log n)$ time with $n/\log n$ processors on EREW PRAM
- Uses known parallel algorithms:
  - Euler tour technique
  - Parentheses matching
  - List ranking
  - Segmented prefix sum

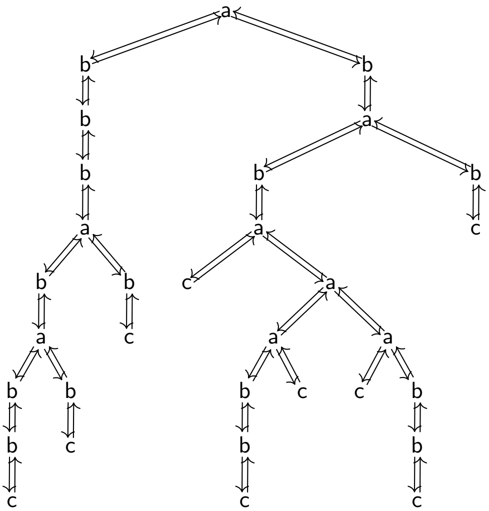
Ordering the input tree

- All nodes with the same relative depth against their segment can be computed at the same time
- Order nodes by their relative depth
- Modified algorithm for Breath-first ordering

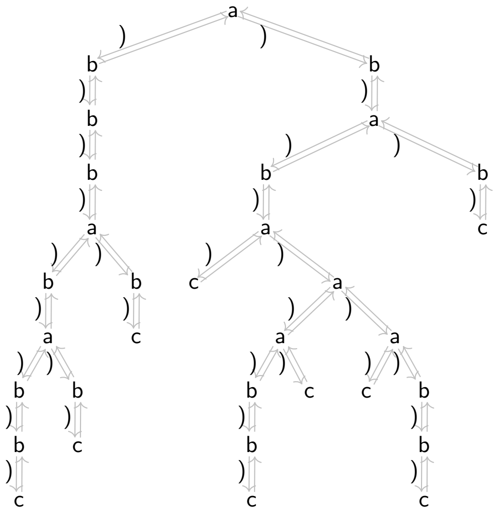




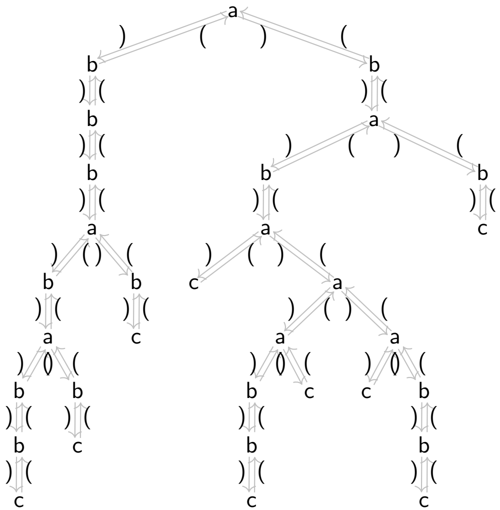
Ordering the input tree



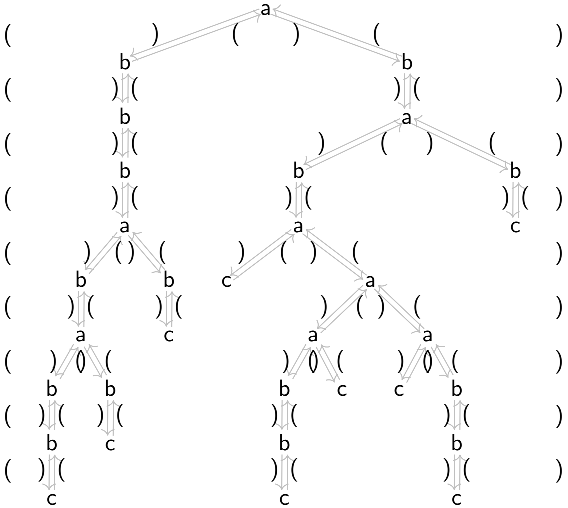
Ordering the input tree



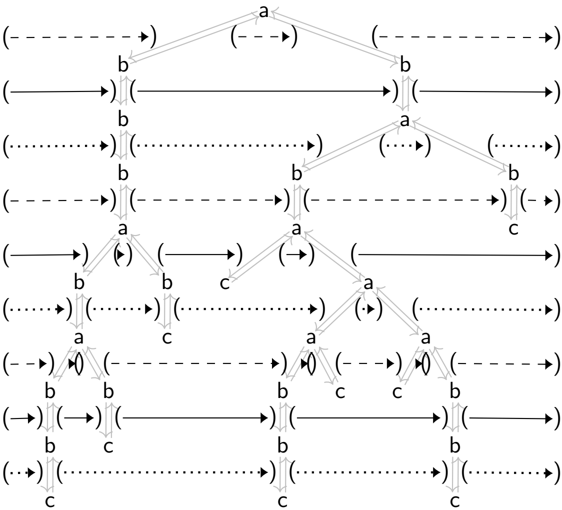
Ordering the input tree



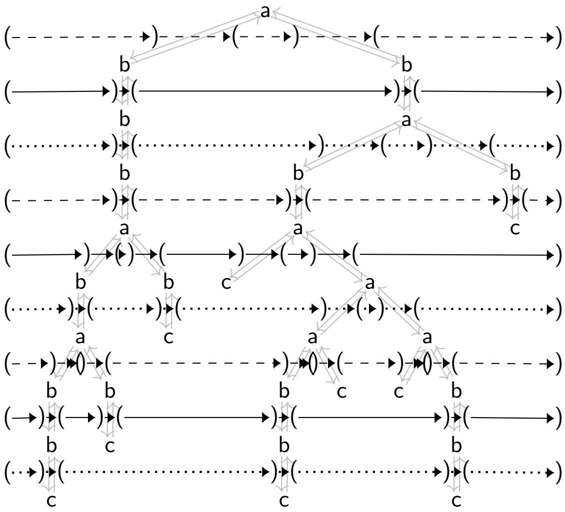
Ordering the input tree



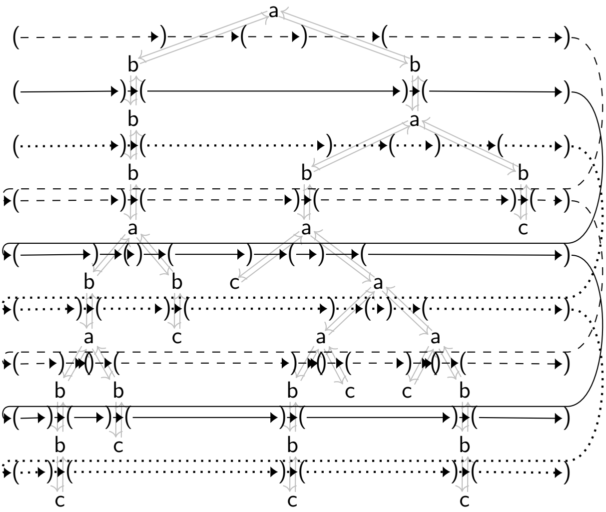
Ordering the input tree



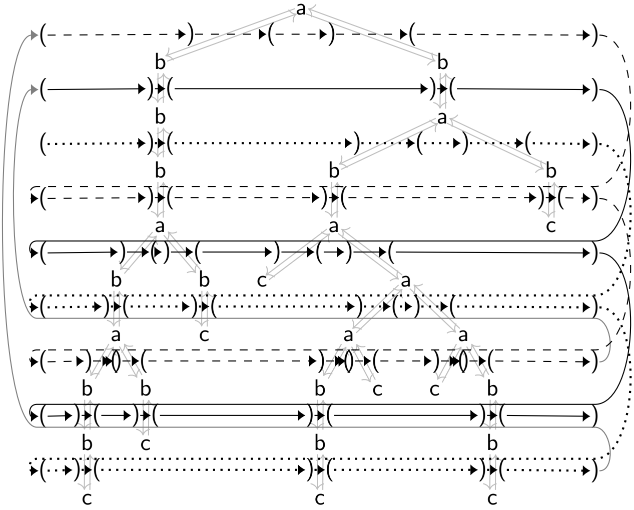
Ordering the input tree



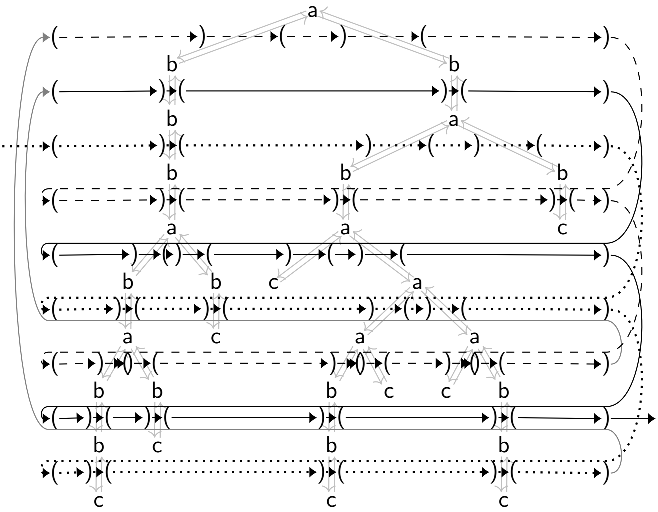
Ordering the input tree



Ordering the input tree



Ordering the input tree



Ordered tree traversal

- Start parallel traversal from the end of the array
- All nodes from a group must be computed before moving to the next
- On the boundary some processors might stall

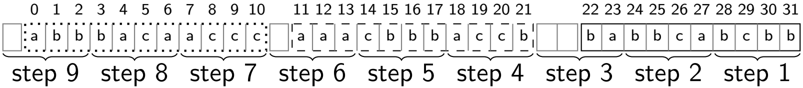


Figure: Example traversal with 4 processors

Summary

- Introduced concepts for DFTA of:
  - Synchronizing tree
  - k-locality
- Proved that tree pattern matching automata are k-local with respect to the height of the pattern
- Shown work optimal parallel algorithm for run of k-local DFTA in $\mathcal{O}(\log n)$ time with $n/\log n$ processors on EREW PRAM
  - for $k = \mathcal{O}(\log n)$
  - for general k in $\mathcal{O}(\max(k, \log n))$ time with $n/\max(k, \log n)$ processors

Thank you!