Lempel-Ziv Computation In Compressed Space (LZ-CICS)

Dominik Köppl[1] Kunihiko Sadakane[2]

[1]TU Dortmund, Germany
[2]University of Tokyo, Japan

March 10, 2016

LZ parsing

a a a b a a b a a a b a a $

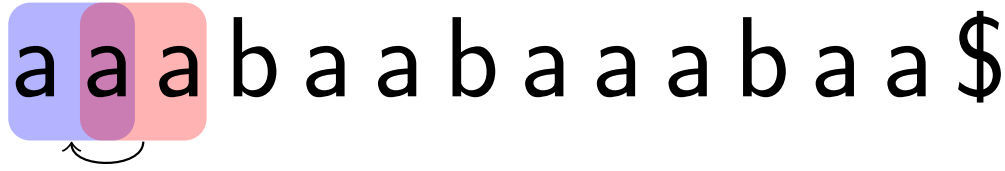
a (1,2) b (2,5) (3,4) $

LZ parsing

a a a b a a b a a a b a a $

a (1,2) b (2,5) (3,4) $

LZ parsing



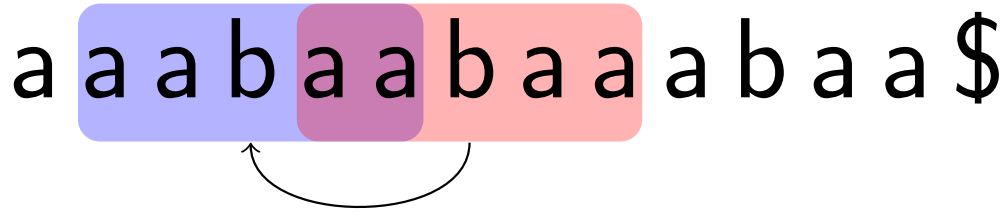
a (1,2) b (2,5) (3,4) $

LZ parsing

a a a b a a b a a a b a a $

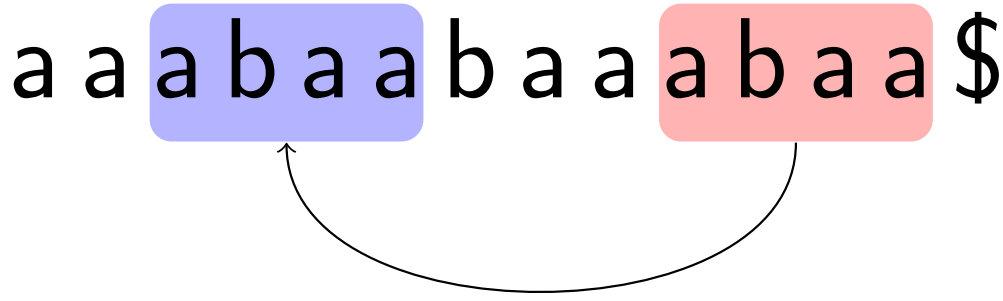
a (1,2) b (2,5) (3,4) $

LZ parsing



a (1,2) b (2,5) (3,4) $

LZ parsing



a (1,2) b (2,5) (3,4) $

LZ parsing

a a a b a a b a a a b a a $

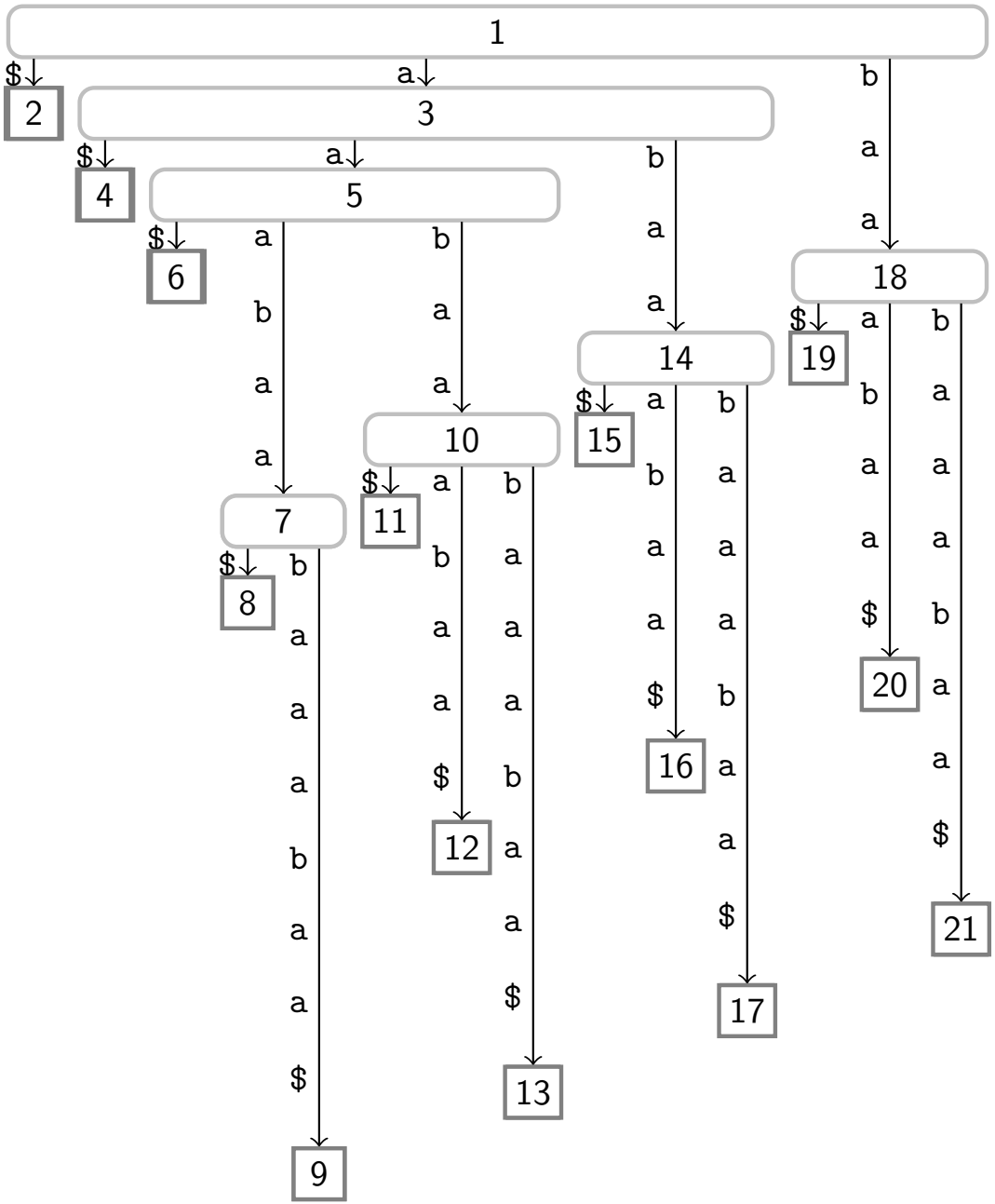
a (1,2) b (2,5) (3,4) $

related work

time	bits of working space	authors
$\mathcal{O}(n)$	$3n \lg n$	Goto and Bannai'13
$\mathcal{O}(n)$	$2n \lg n$	Kärkkäinen et al.'13
$\mathcal{O}(n)$	$n \lg n + \mathcal{O}(\sigma \lg n)$	Goto and Bannai'14
$\mathcal{O}(n \lg \lg \sigma)$	$\mathcal{O}(n \lg \sigma)$	Belazzougui et al.'16
$\mathcal{O}(n)$	$(1+\varepsilon)n \lg n + \mathcal{O}(n)$	CPM'15
$\mathcal{O}(n \lg \lg \sigma)$	$\mathcal{O}(n \lg \sigma)$	DCC'16

n: text size $\quad$ σ: alphabet size $\quad$ $0 < \varepsilon \leq 1$ constant

suffix tree

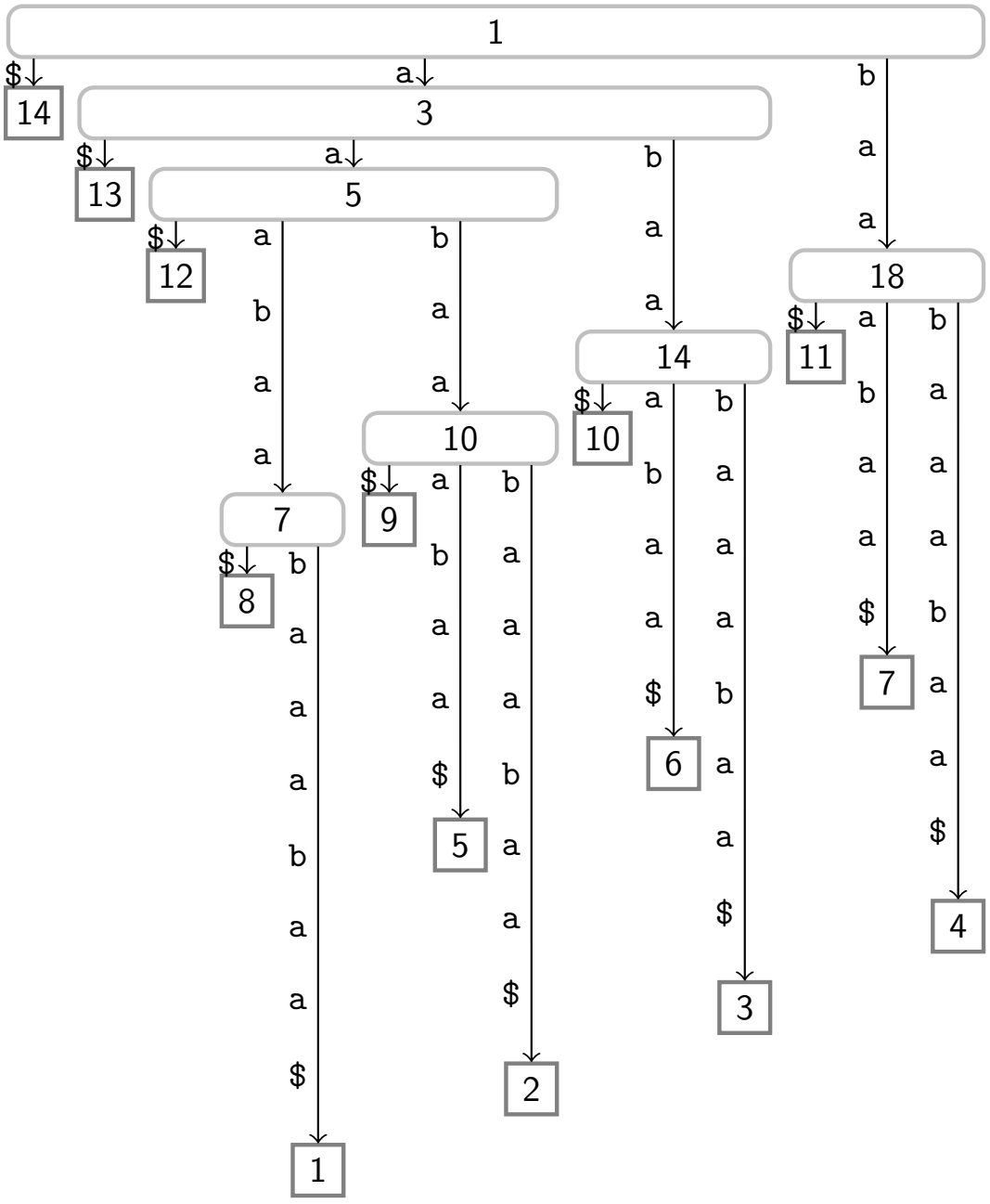


ST of aaabaabaaabaa

labels

nodes:
pre-order number

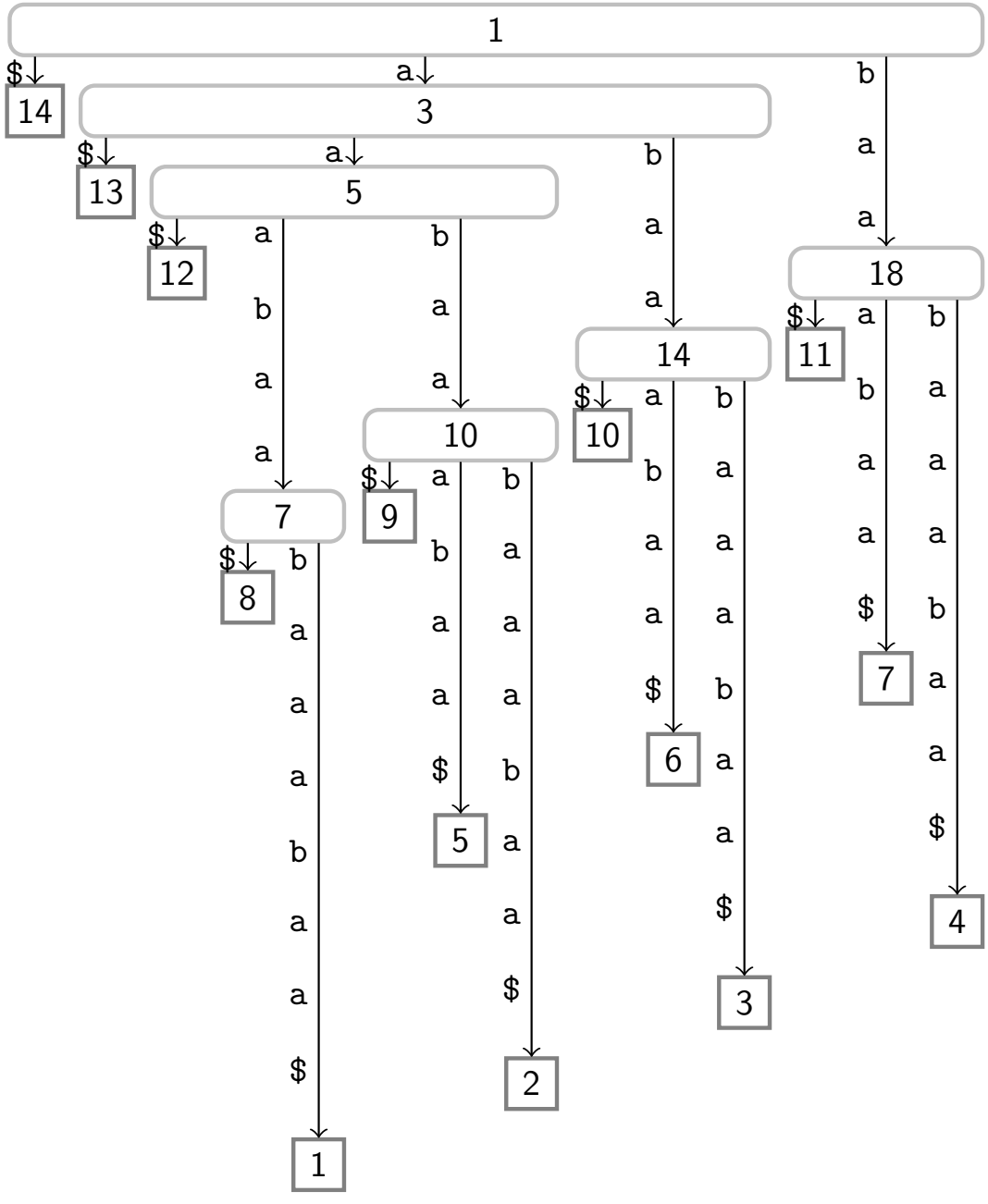
suffix tree



ST of aaabaabaaabaa

labels

- internal nodes:
  pre-order number
- leaves:
  text pos. of suffix

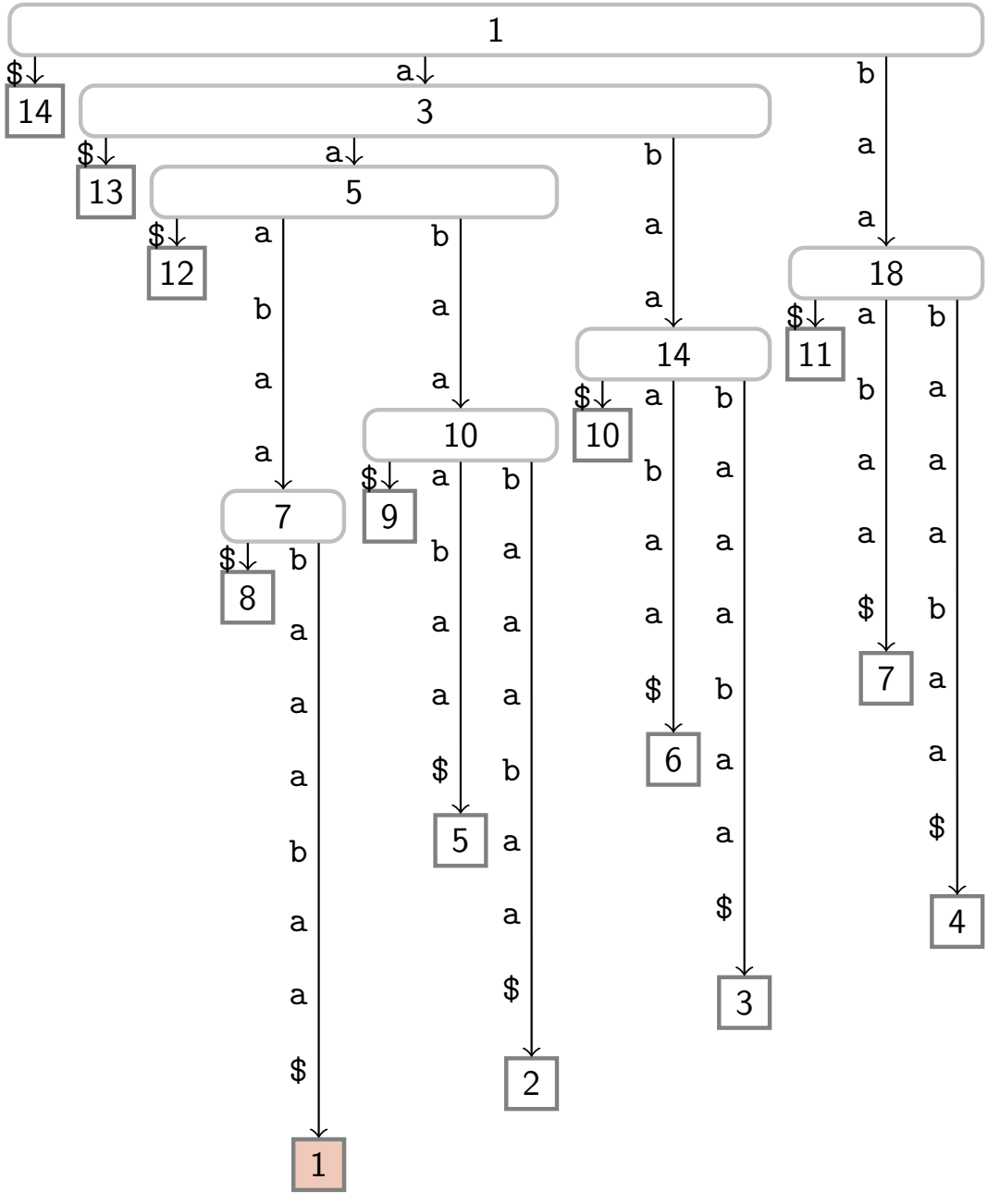


LZ77 parsing of
a aa b aabaa abaa $

witnesses:
5, 10, 14

Definition

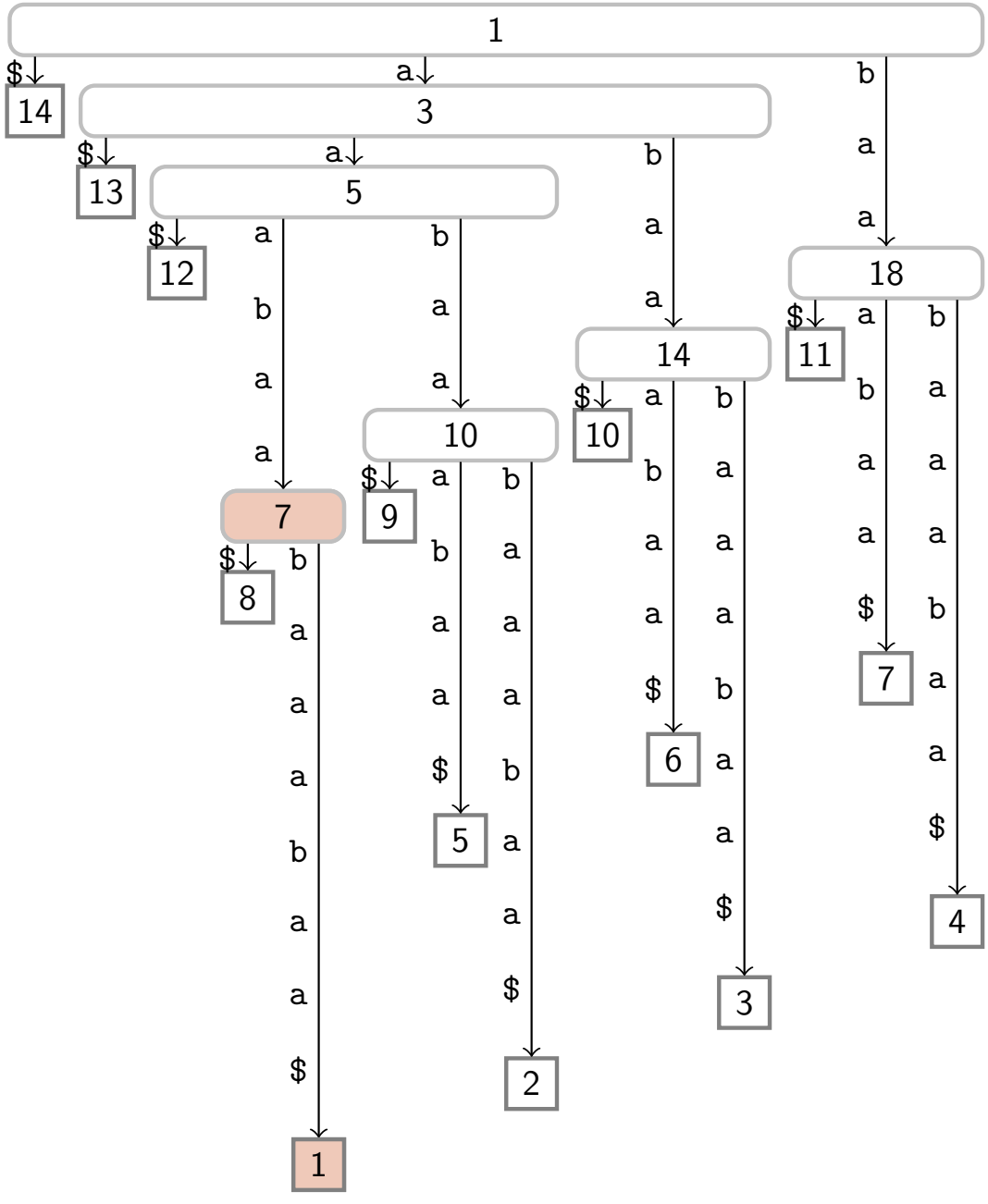
witness: already visited
node accessed by LZ leaf.



LZ77 parsing of
a aa b aabaa abaa $

witnesses:
5, 10, 14

Definition
witness: already visited node accessed by LZ leaf.



LZ77 parsing of

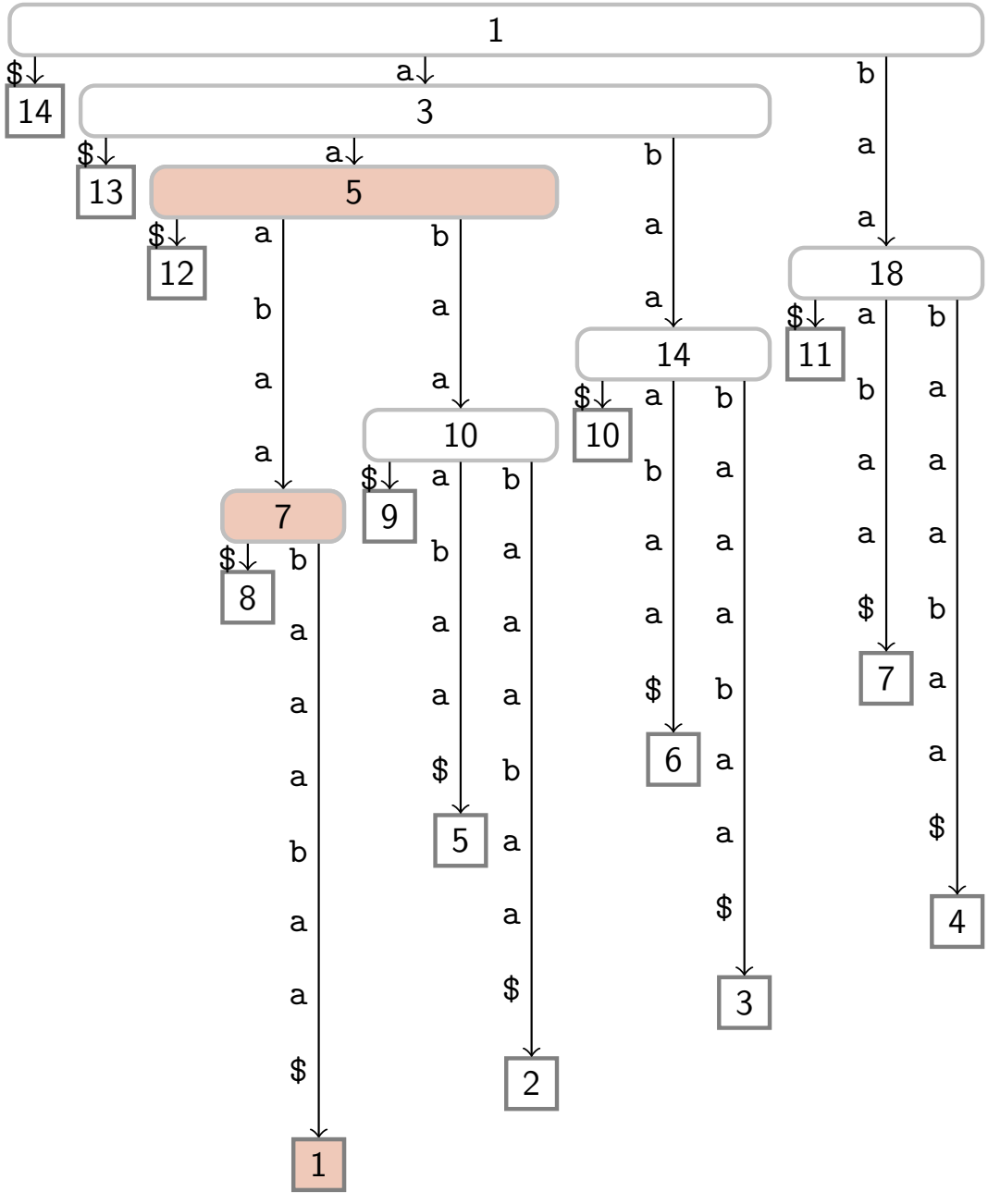
a aa b aabaa abaa $

witnesses:

5, 10, 14

Definition

witness: already visited node accessed by LZ leaf.

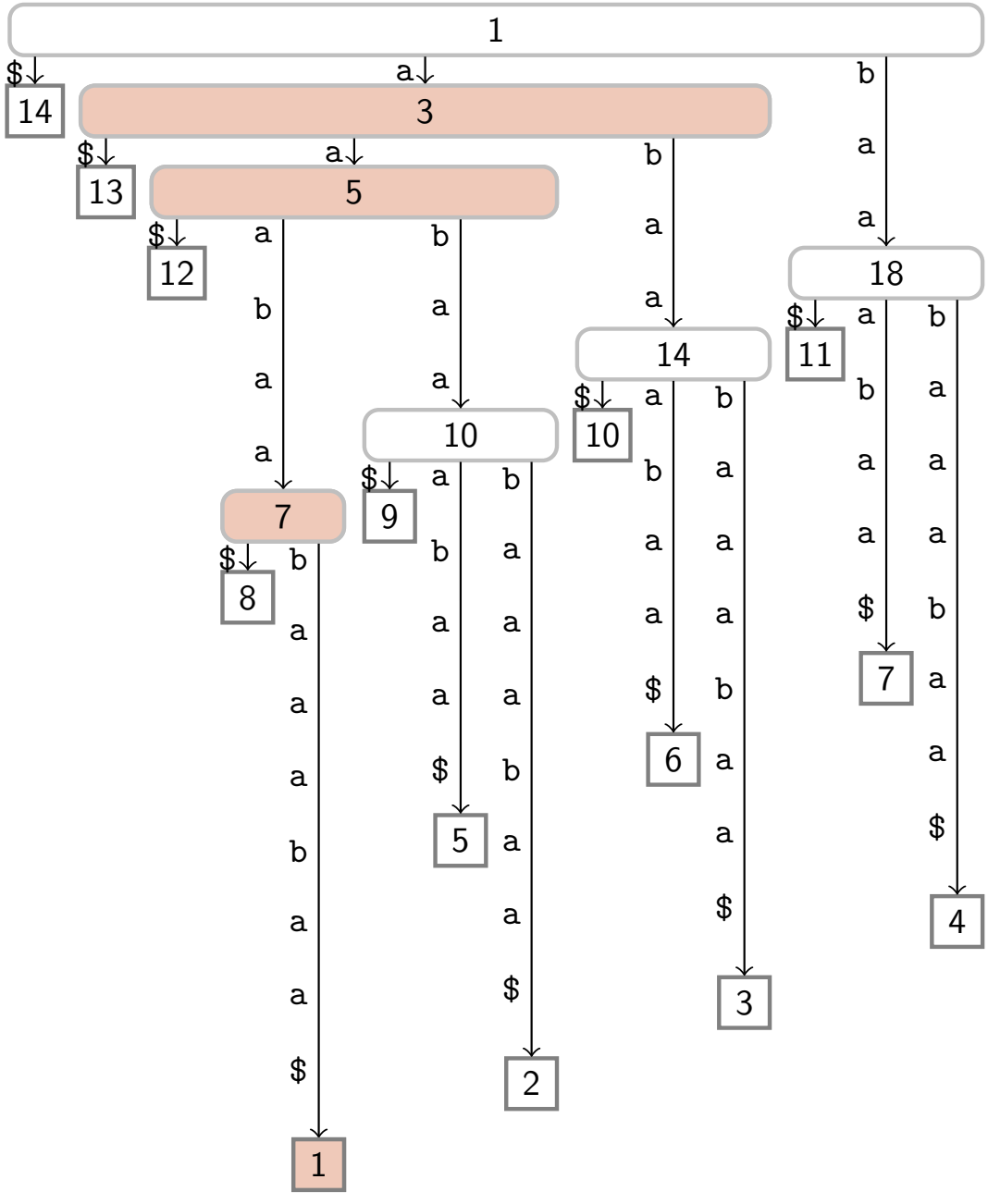


LZ77 parsing of
a aa b aabaa abaa $

witnesses:
5, 10, 14

Definition

witness: already visited node accessed by LZ leaf.

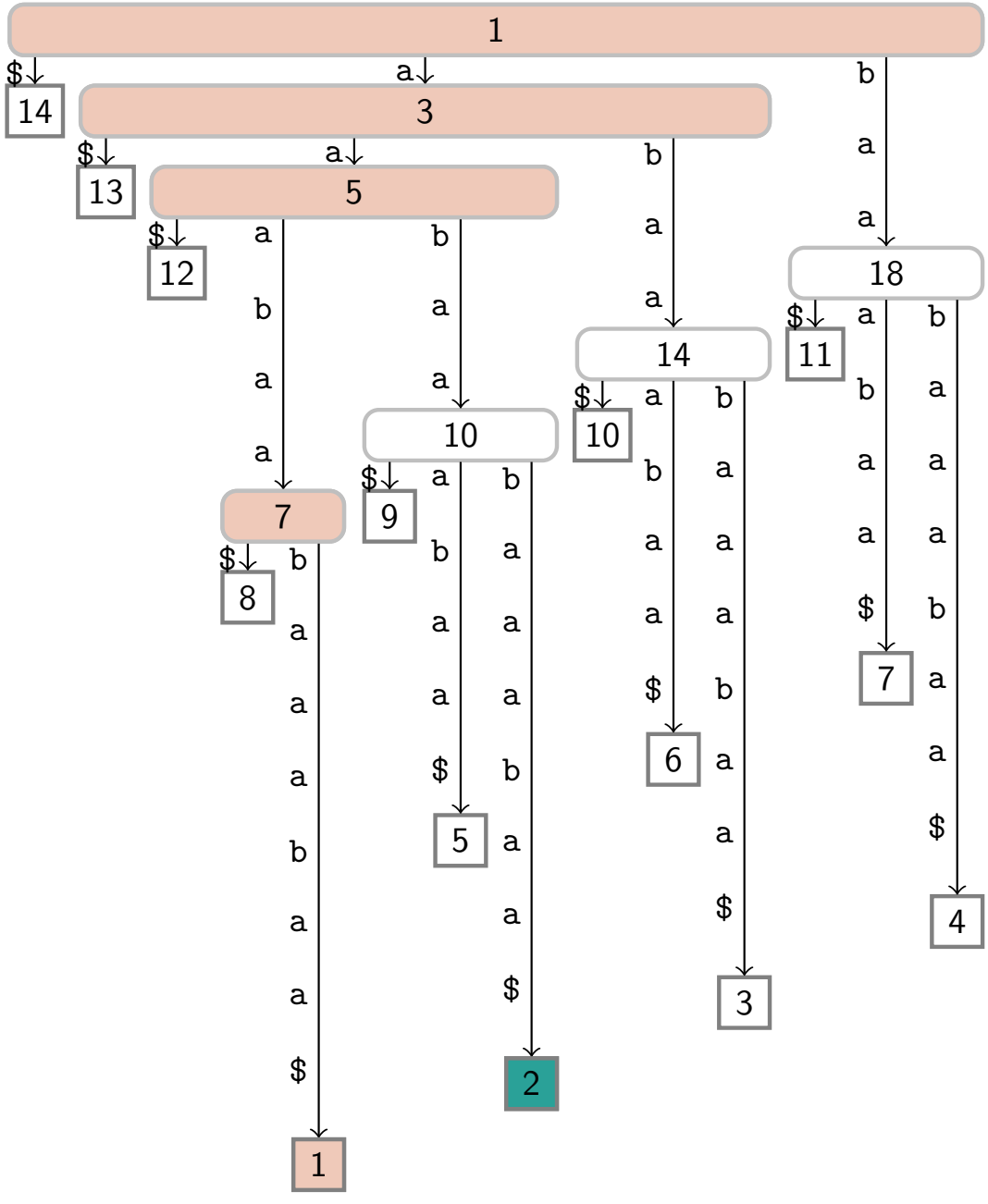


LZ77 parsing of
a aa b aabaa abaa $

witnesses:
5, 10, 14

Definition

witness: already visited
node accessed by LZ leaf.



LZ77 parsing of

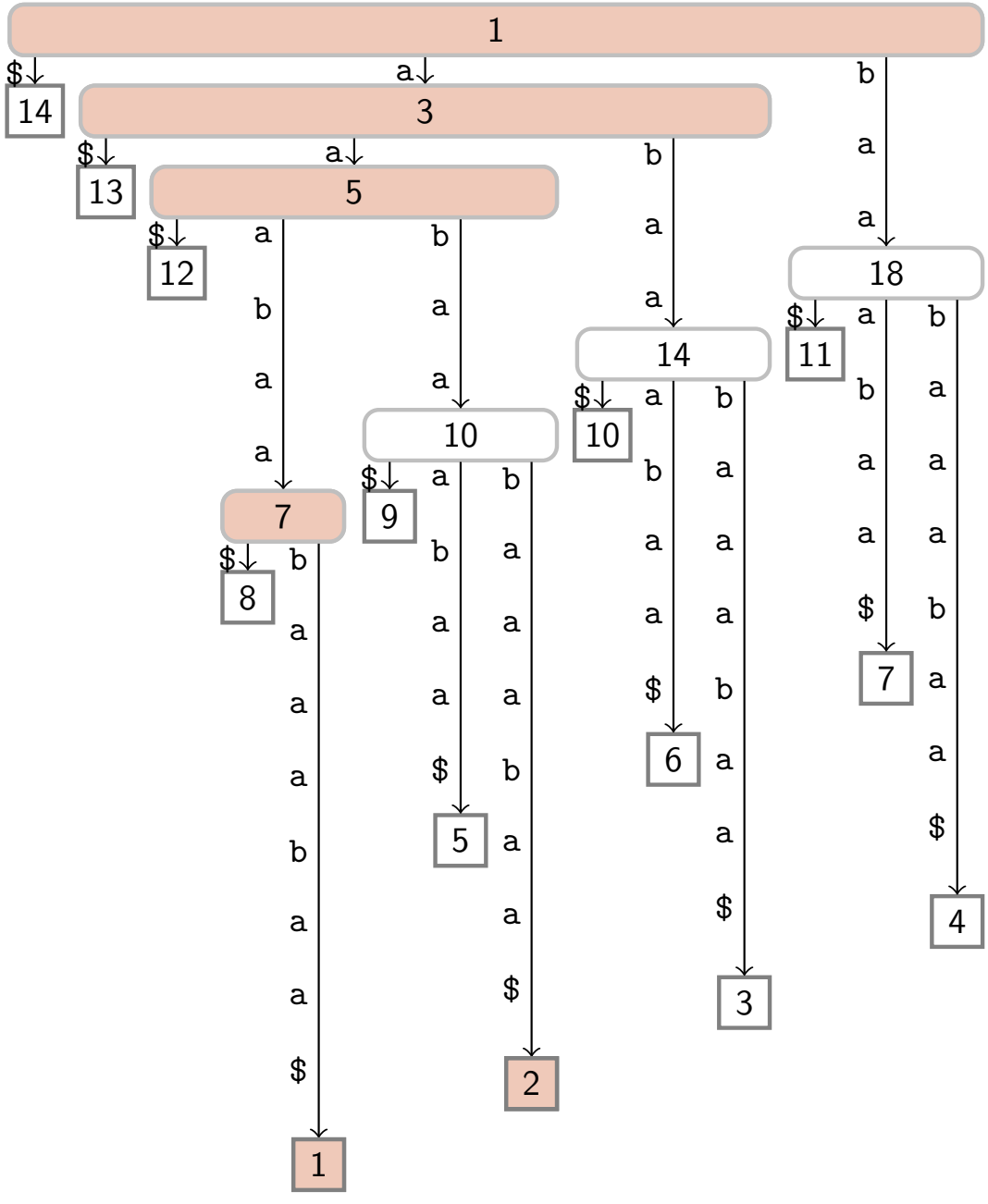
a aa b aabaa abaa $

witnesses:

5, 10, 14

Definition

witness: already visited node accessed by LZ leaf.



LZ77 parsing of

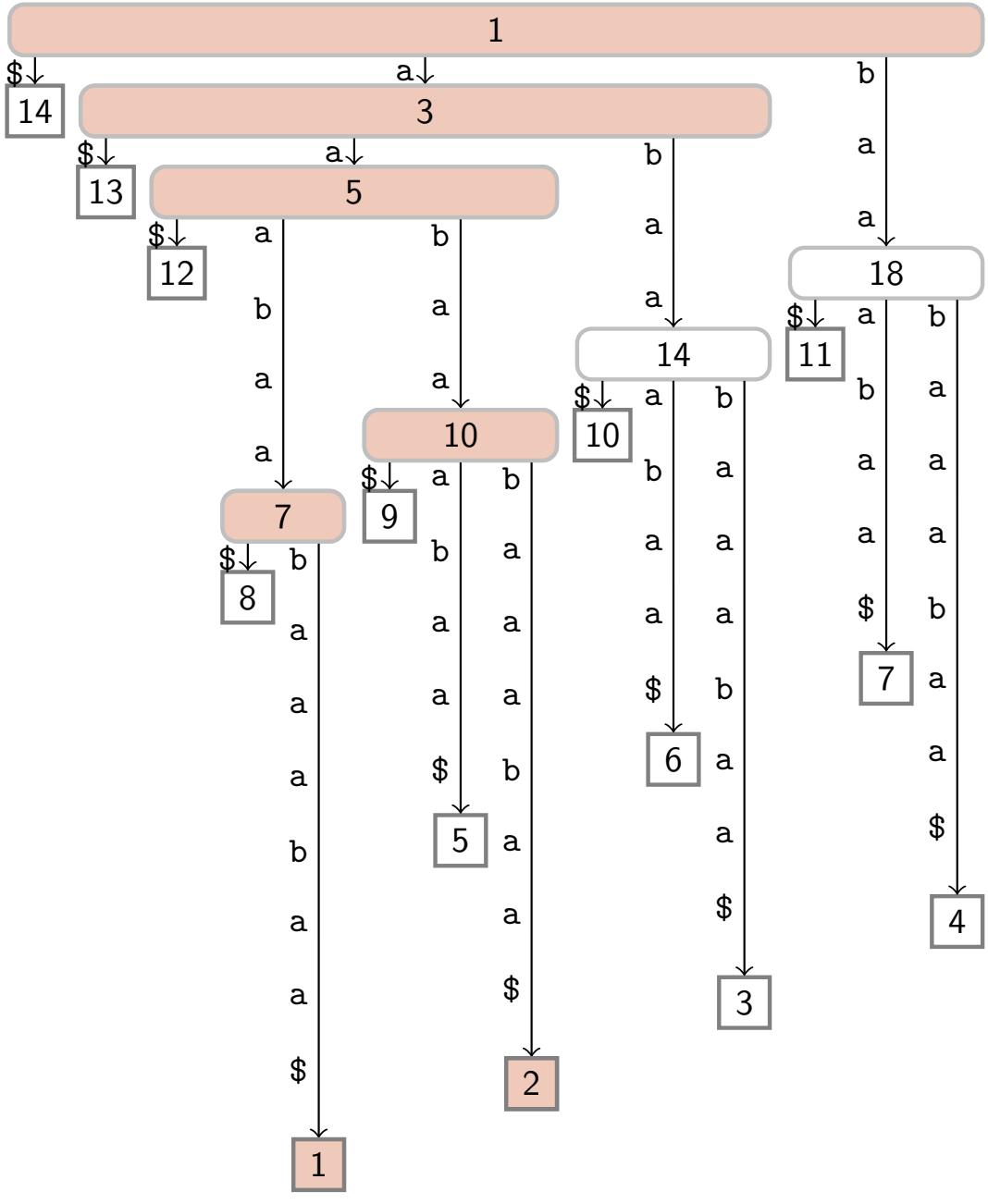
a aa b aabaa abaa $

witnesses:

5, 10, 14

Definition

witness: already visited node accessed by LZ leaf.



LZ77 parsing of

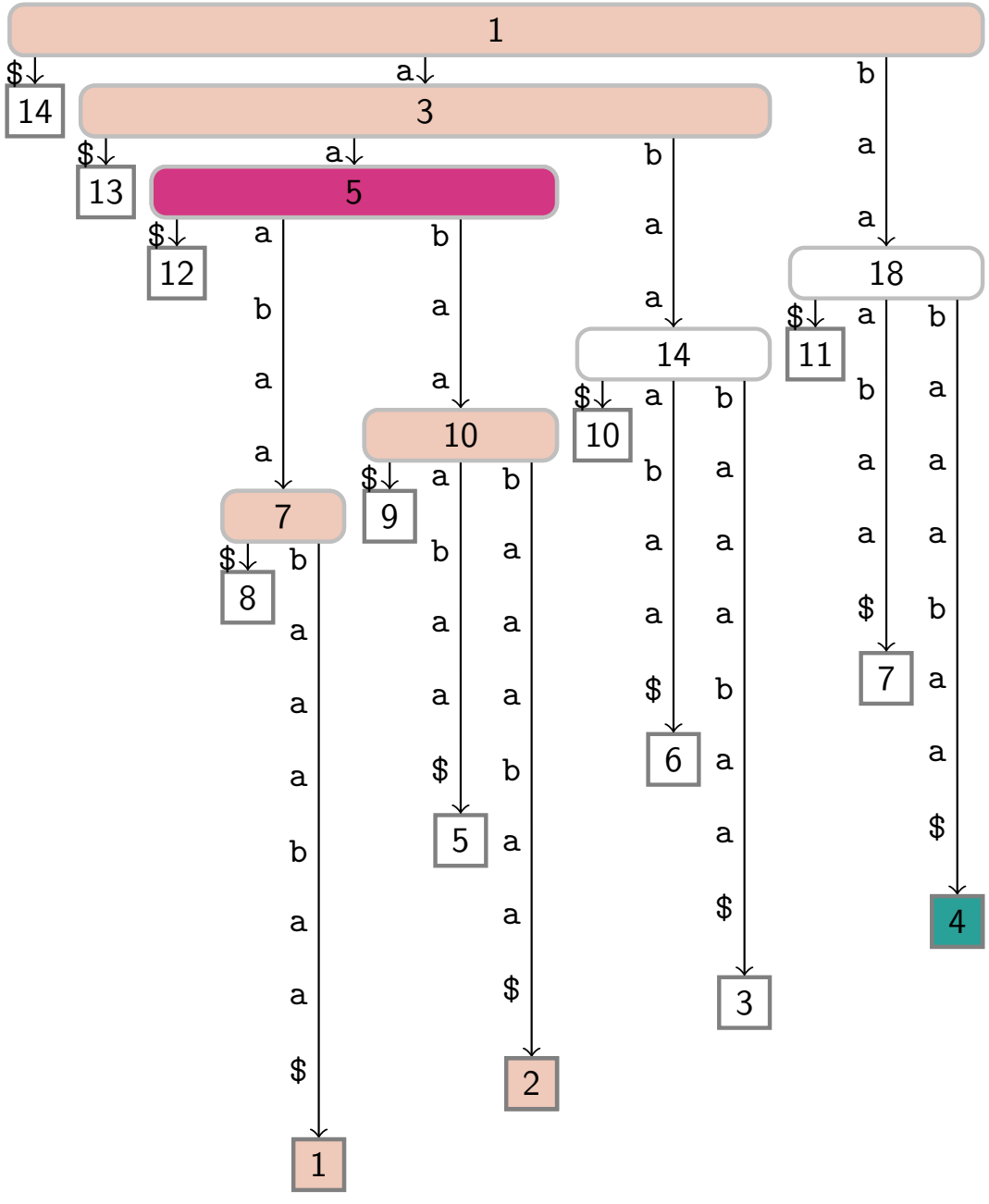
a aa b aabaa abaa $

witnesses:

5, 10, 14

Definition

witness: already visited node accessed by LZ leaf.

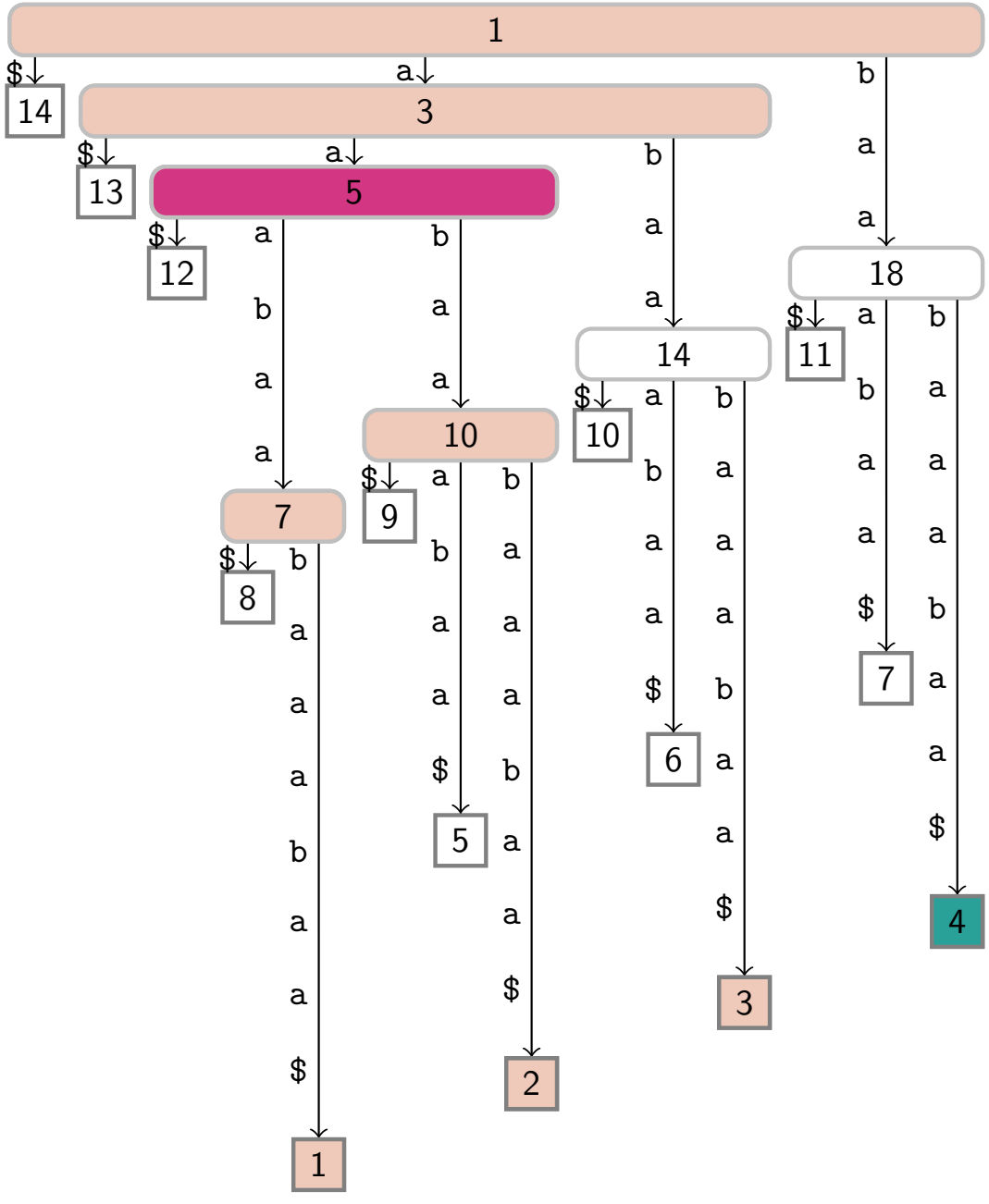


LZ77 parsing of
a|aa b aabaa abaa $

witnesses:
5, 10, 14

Definition

witness: already visited node accessed by LZ leaf.

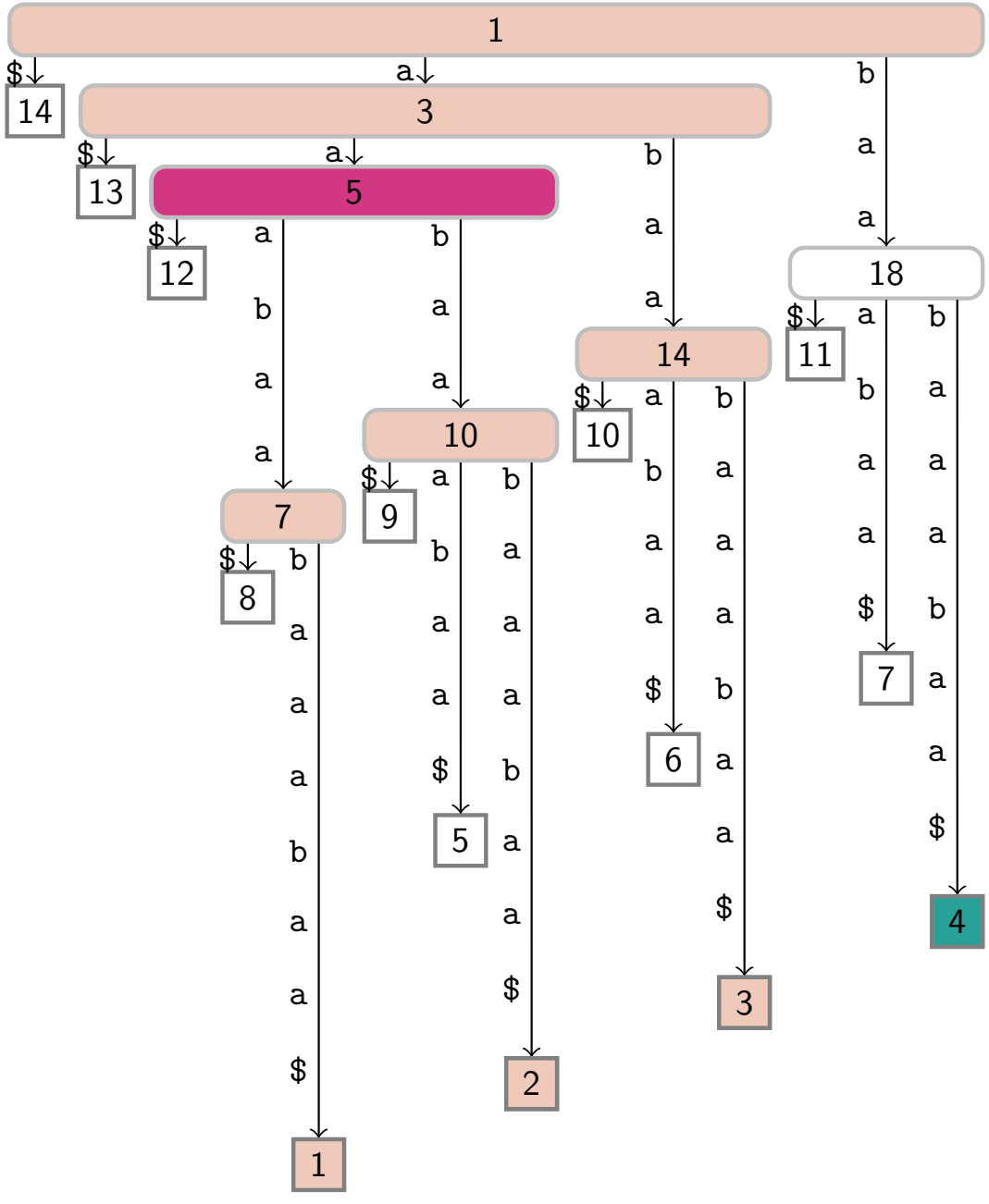


LZ77 parsing of
a|aa b aabaa abaa $

witnesses:
5, 10, 14

Definition

witness: already visited node accessed by LZ leaf.

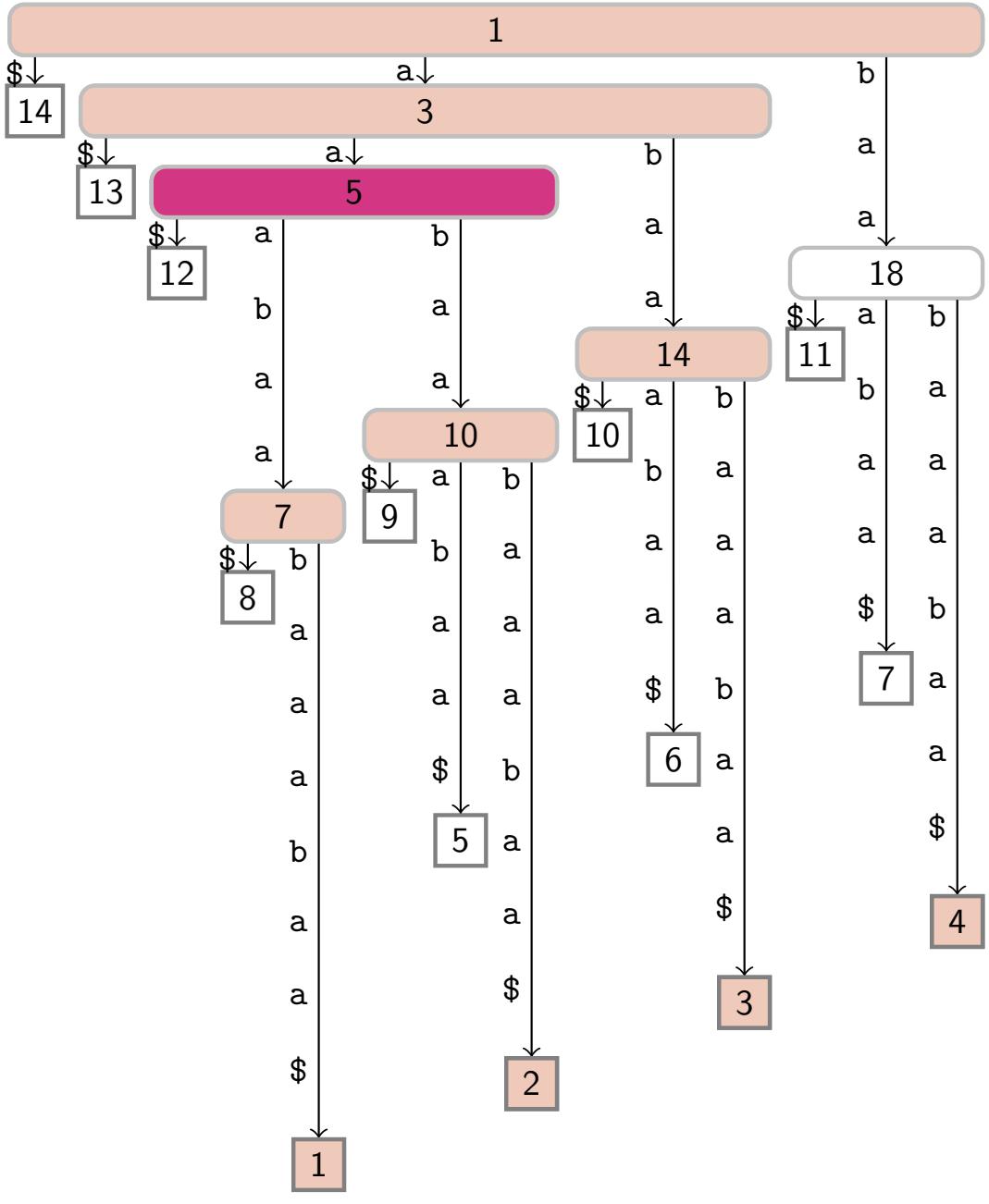


LZ77 parsing of
a|aa b aabaa abaa $

witnesses:
5, 10, 14

Definition

witness: already visited node accessed by LZ leaf.

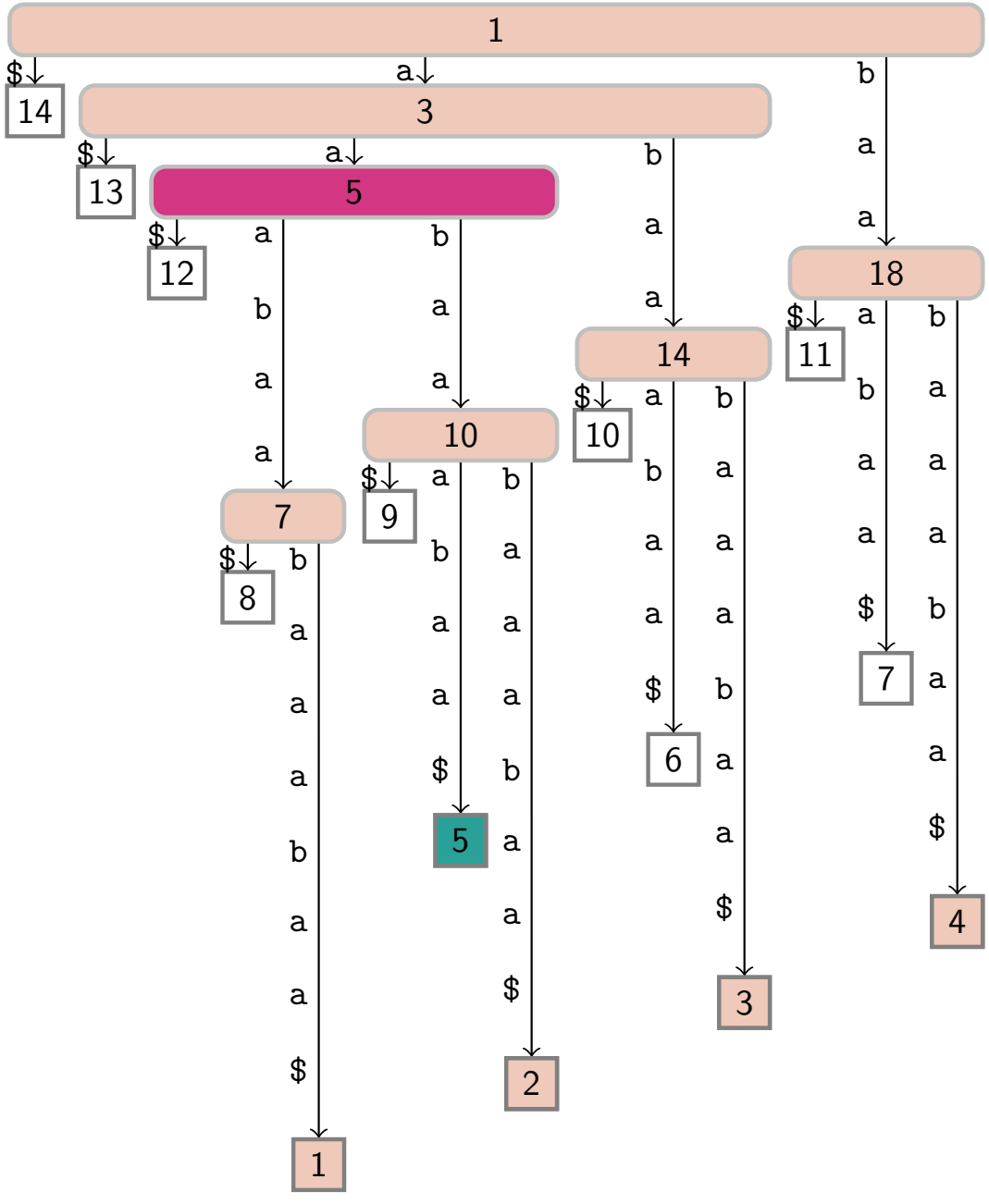


LZ77 parsing of
a|aa b aabaa abaa $

witnesses:
5, 10, 14

Definition

witness: already visited node accessed by LZ leaf.

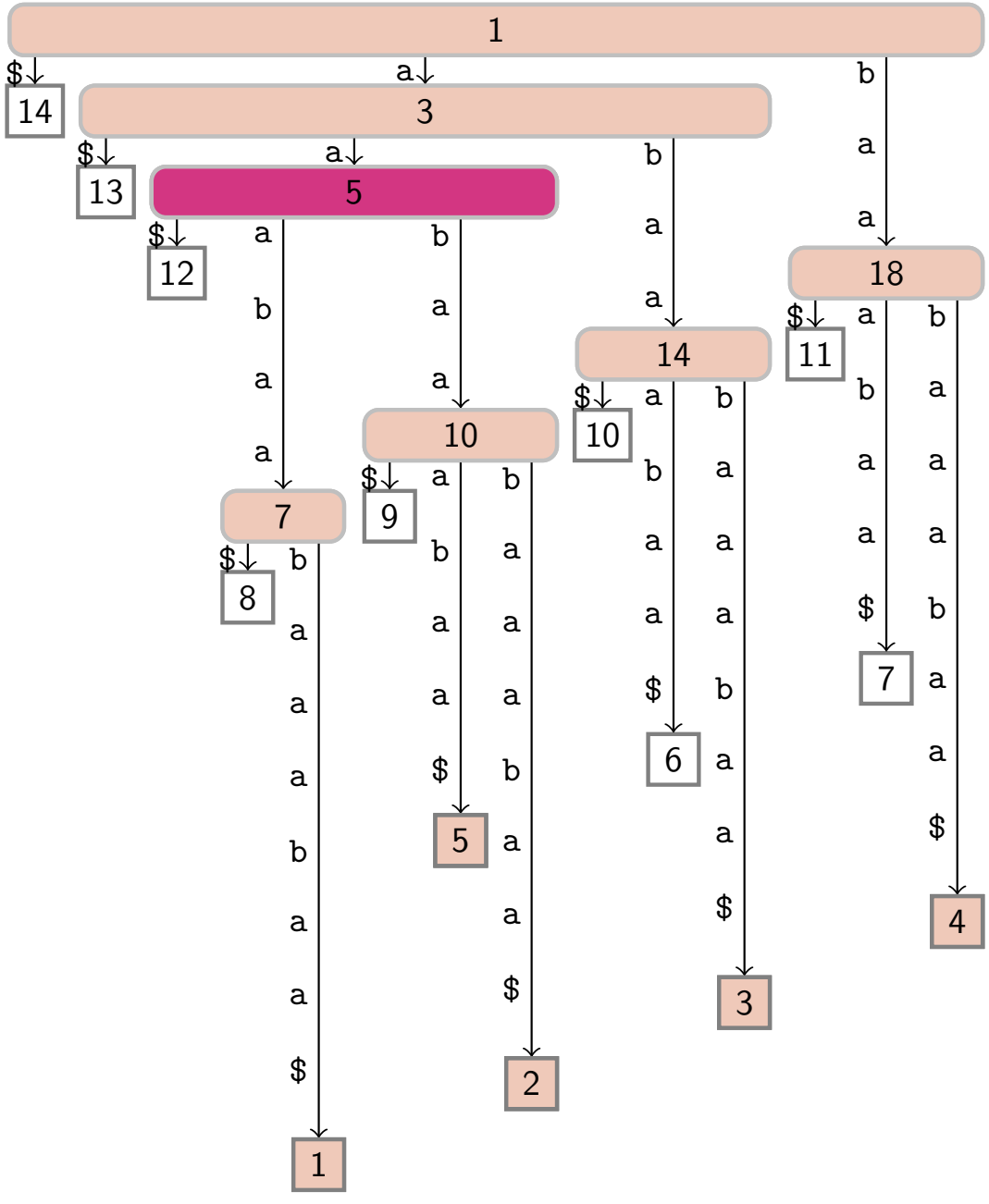


LZ77 parsing of
a|aa|b aabaa abaa $

witnesses:
5, 10, 14

Definition

witness: already visited node accessed by LZ leaf.

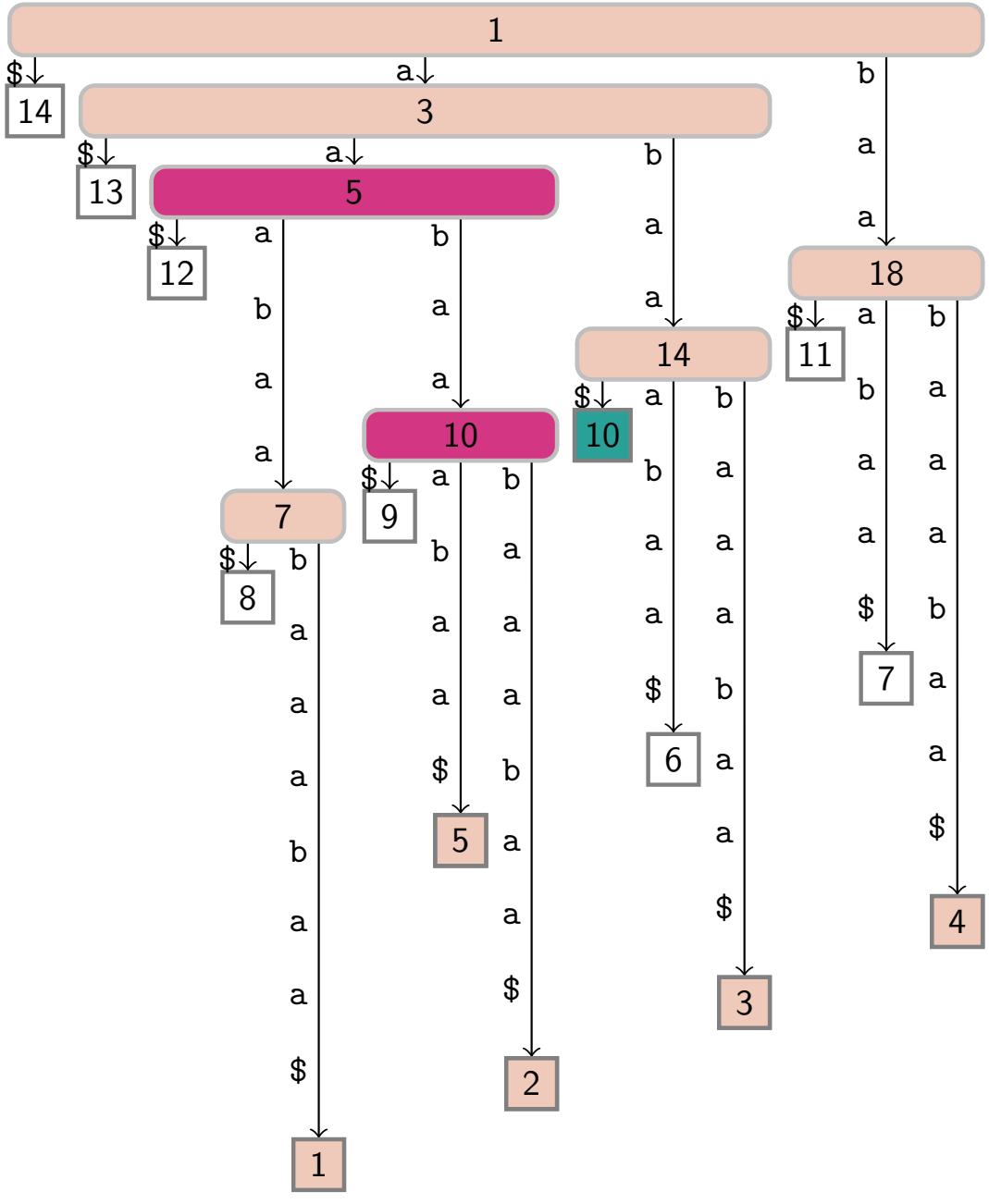


LZ77 parsing of
a|aa|b aabaa abaa $

witnesses:
5, 10, 14

Definition

witness: already visited node accessed by LZ leaf.

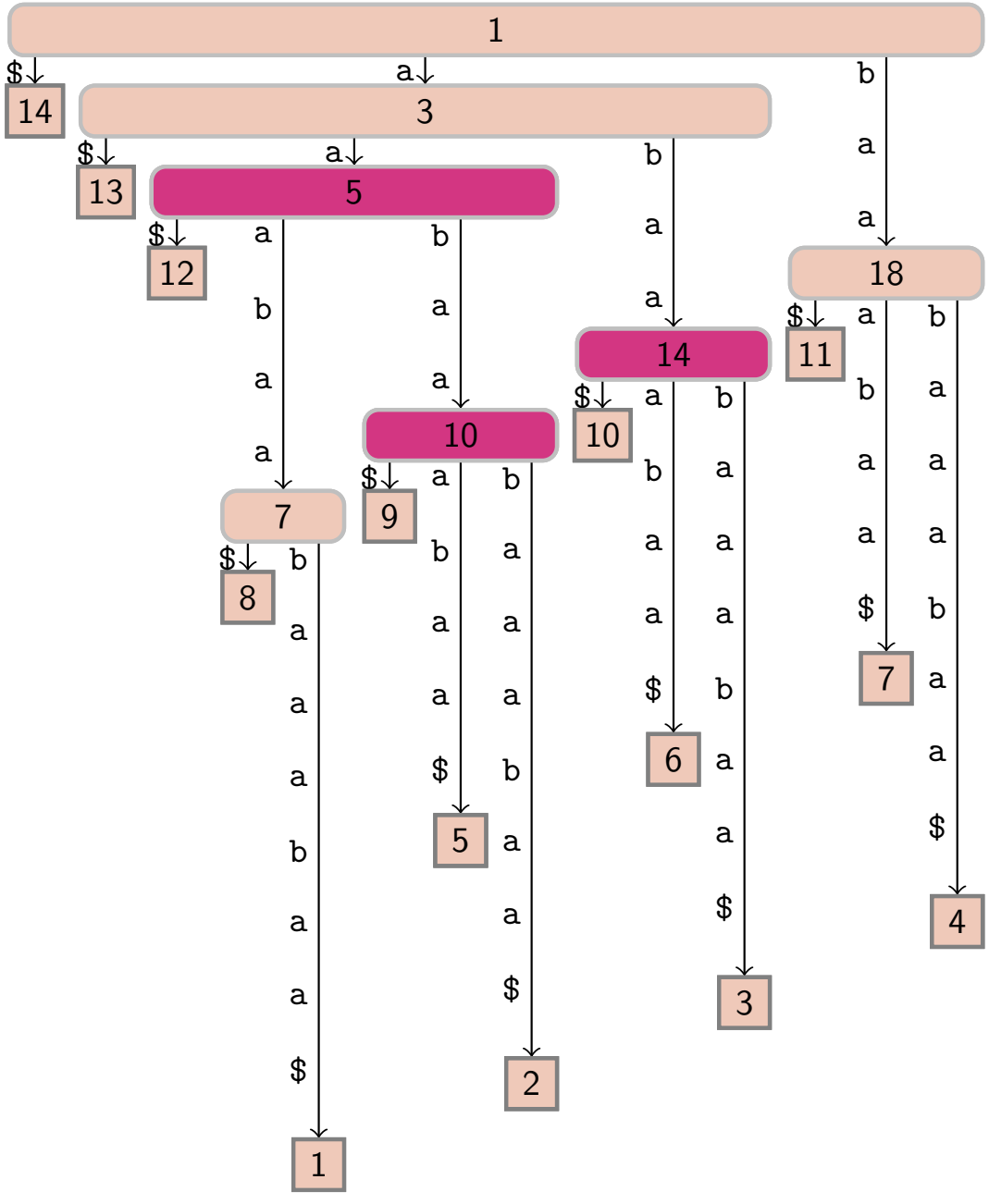


LZ77 parsing of
a|aa|b|aabaa abaa $

witnesses:
5, 10, 14

Definition

witness: already visited node accessed by LZ leaf.

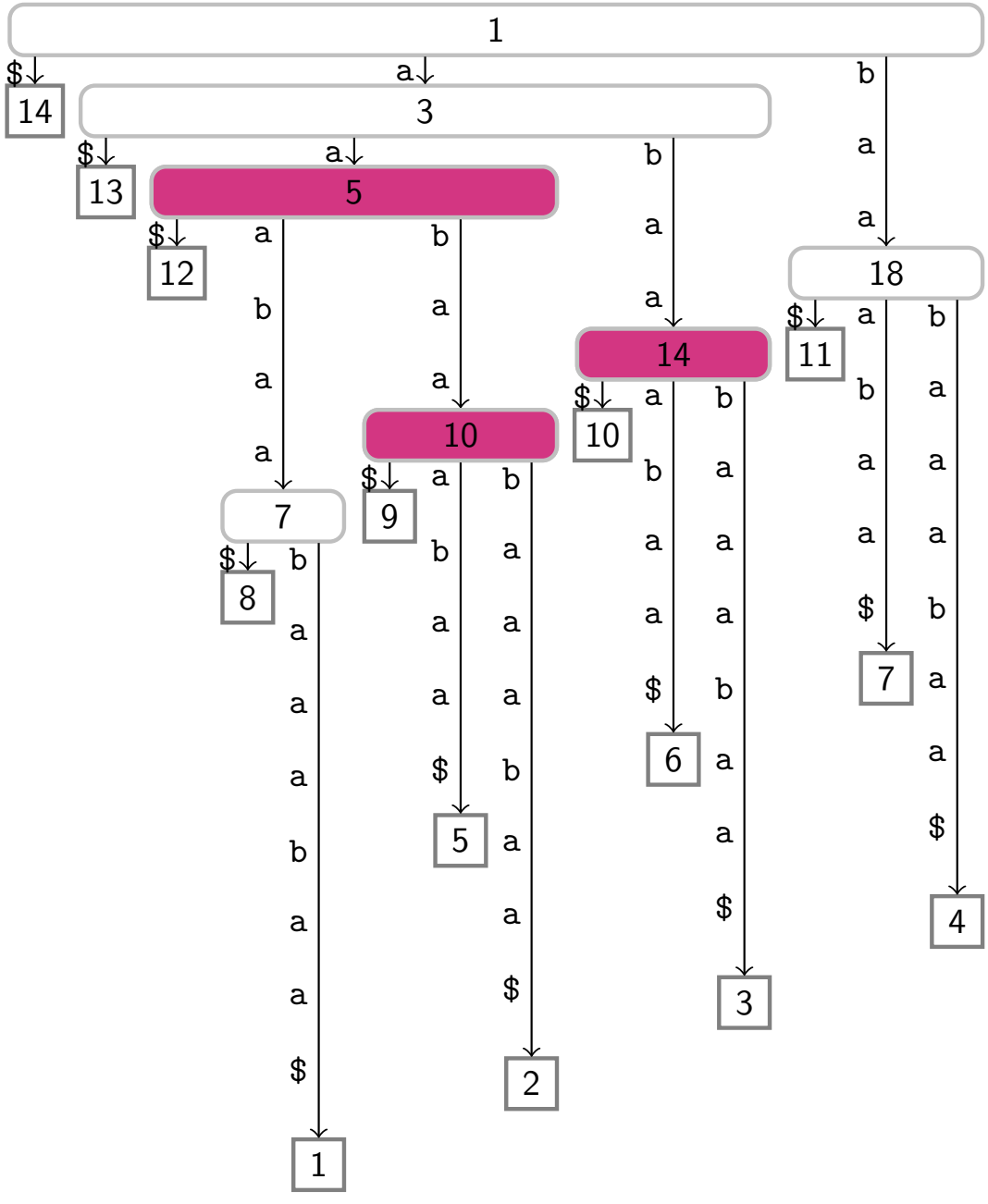


LZ77 parsing of
a|aa|b|aabaa|abaa|$

witnesses:
5, 10, 14

Definition

witness: already visited node accessed by LZ leaf.



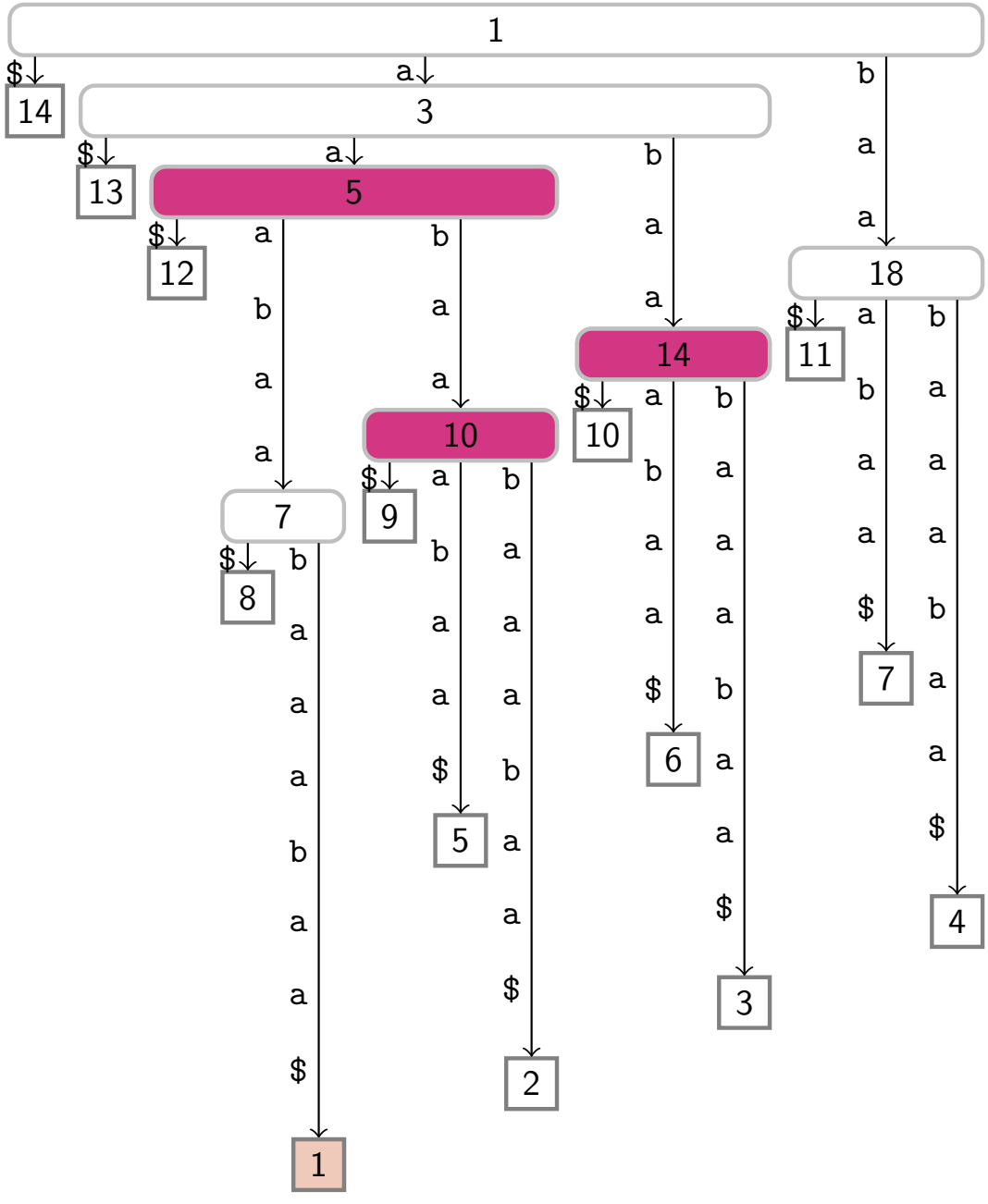
LZ77 parsing of

T = aaabaabaaabaa$:

a	aa	b	aabaa	abaa	$
-	1	-	2	3	-

Mapping W

5	10	14
1	2	3



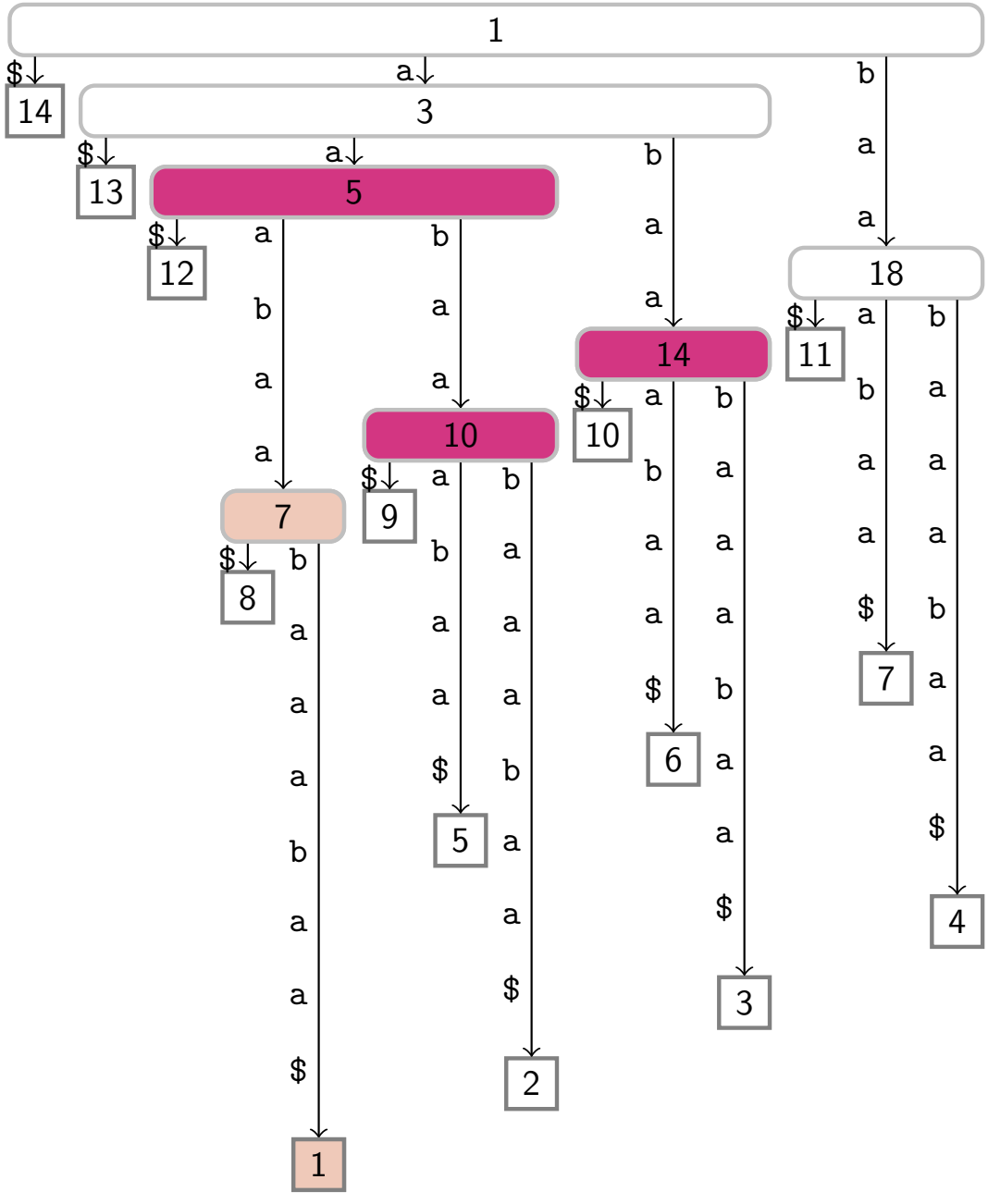
LZ77 parsing of

T = aaabaabaaabaa$:

a	aa	b	aabaa	abaa	$
-	1	-	2	3	-

Mapping W

5	10	14
1	2	3



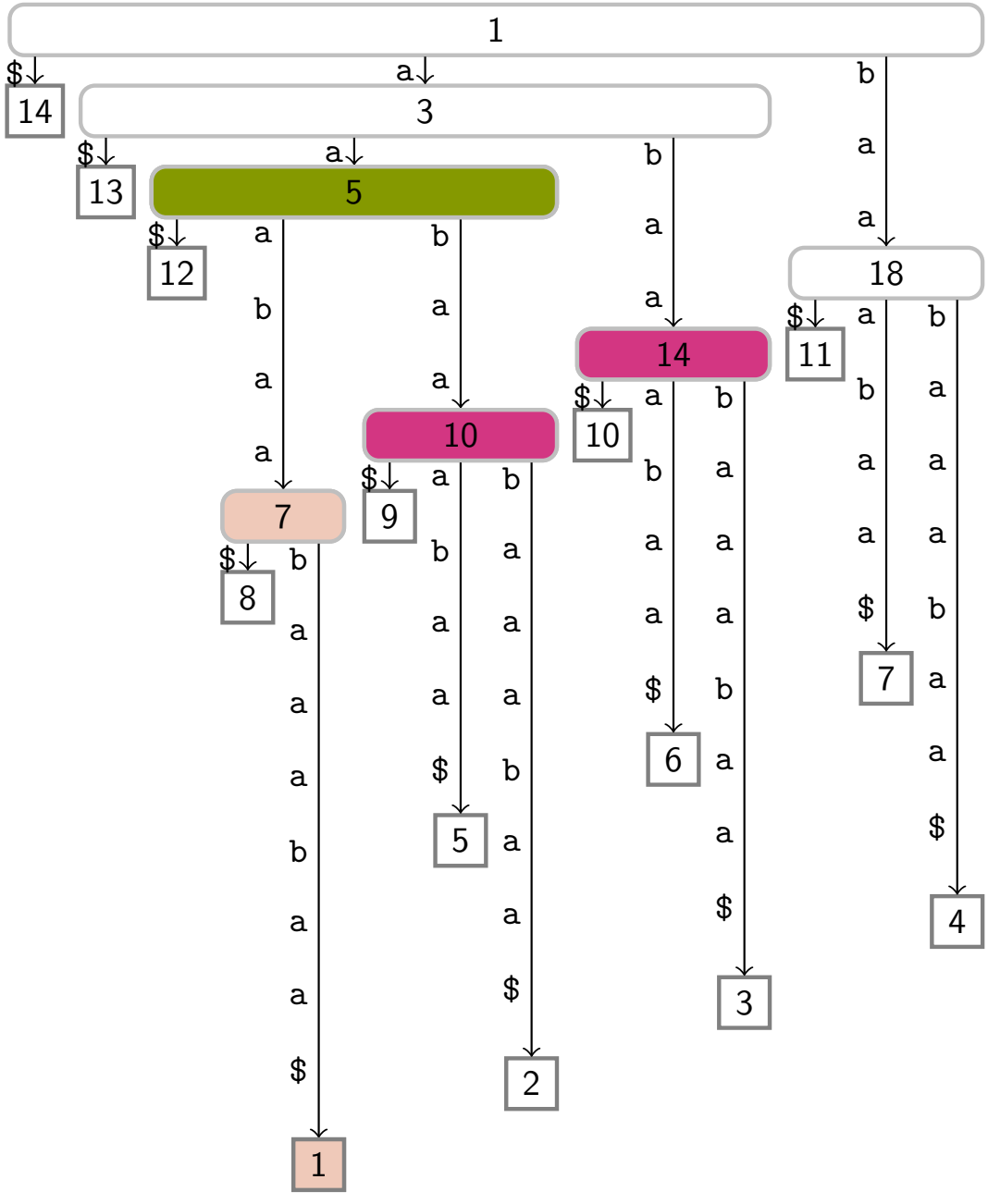
LZ77 parsing of

T = aaabaabaaabaa$:

a	aa	b	aabaa	abaa	$
-	1	-	2	3	-

Mapping W

5	10	14
1	2	3



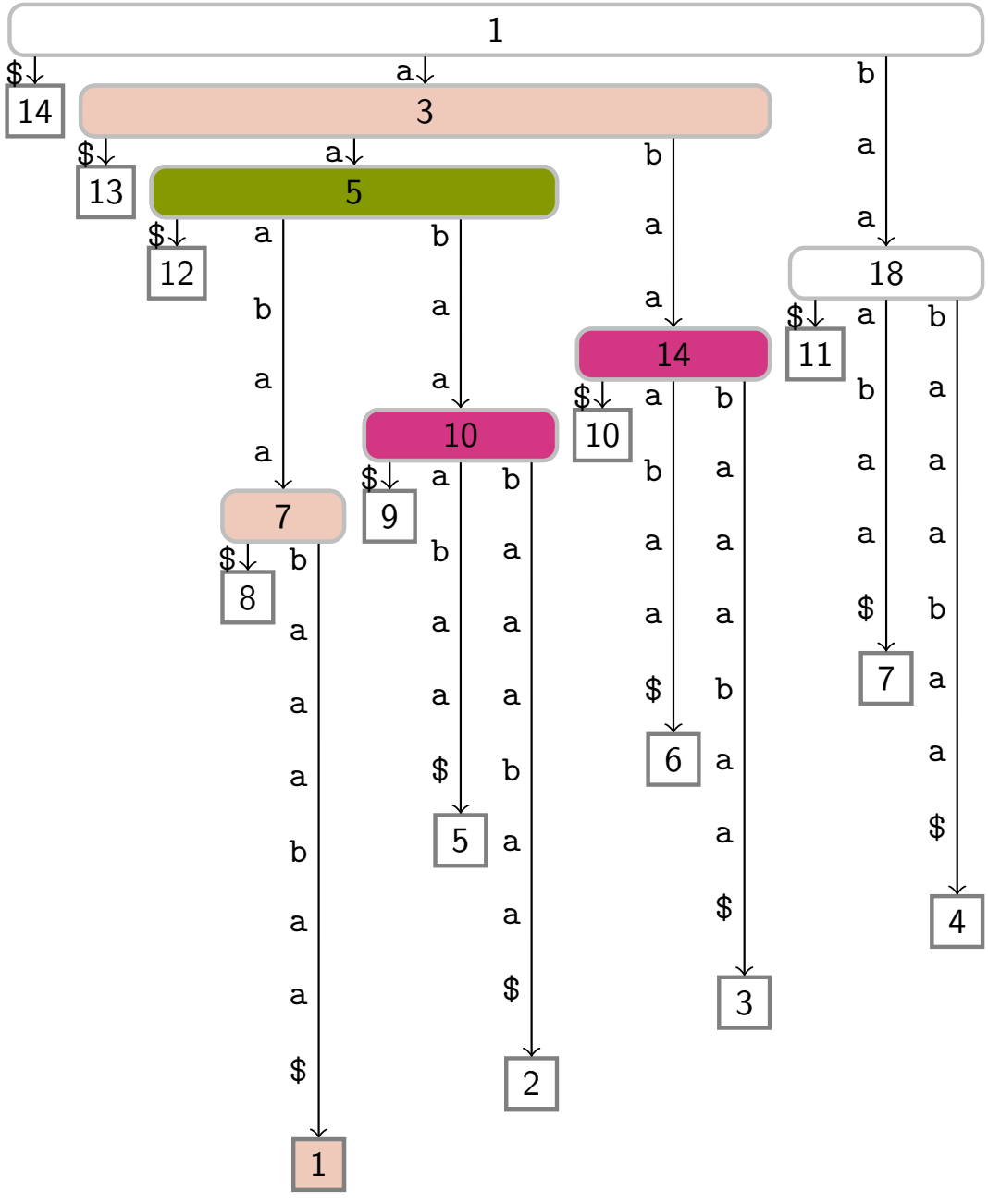
LZ77 parsing of

T = aaabaabaaabaa$:

a	aa	b	aabaa	abaa	$
-	1	-	2	3	-

Mapping W

5	10	14
1	2	3



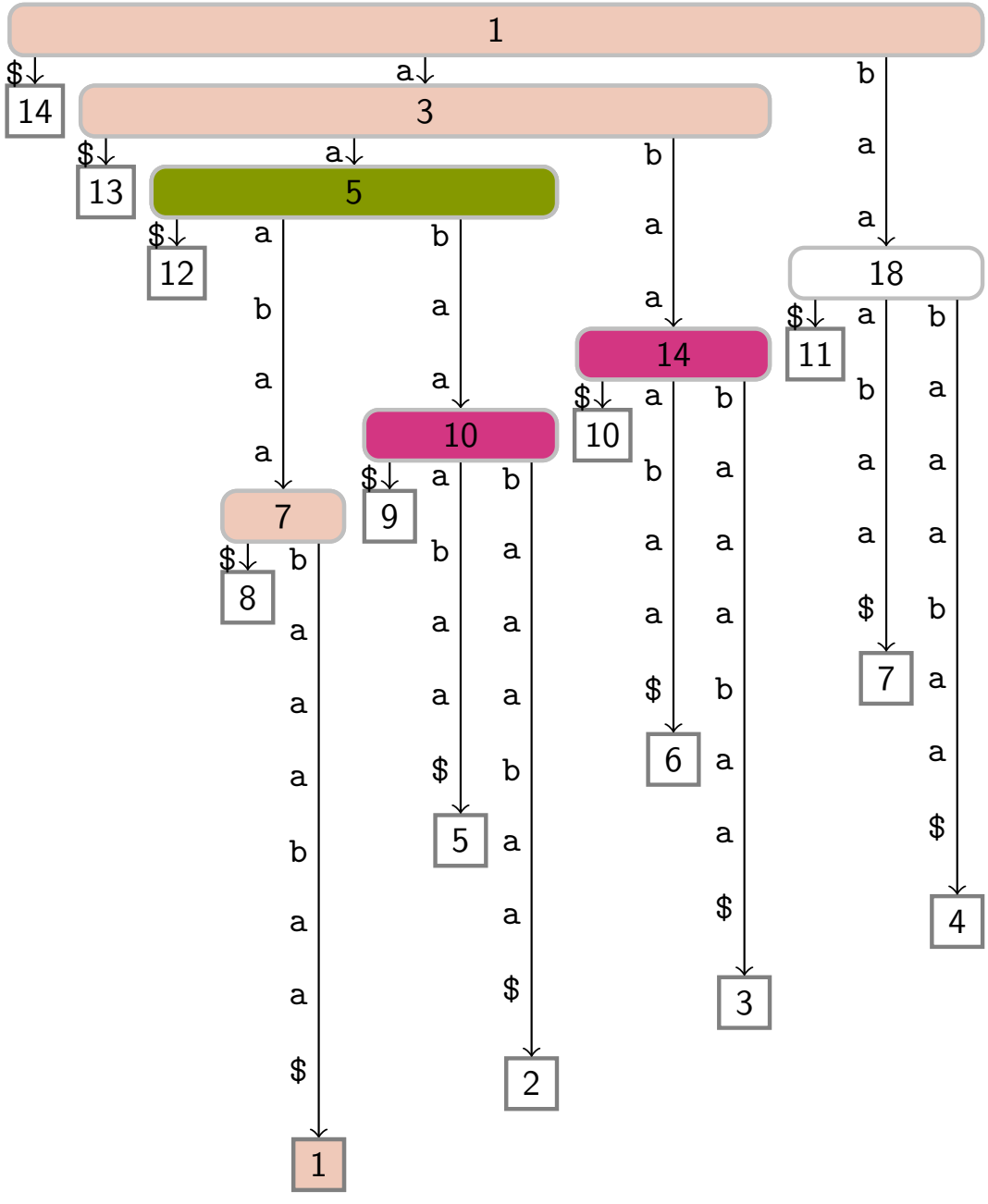
LZ77 parsing of

T = aaabaabaaabaa$:

a	aa	b	aabaa	abaa	$
-	1	-	2	3	-

Mapping W

5	10	14
1	2	3



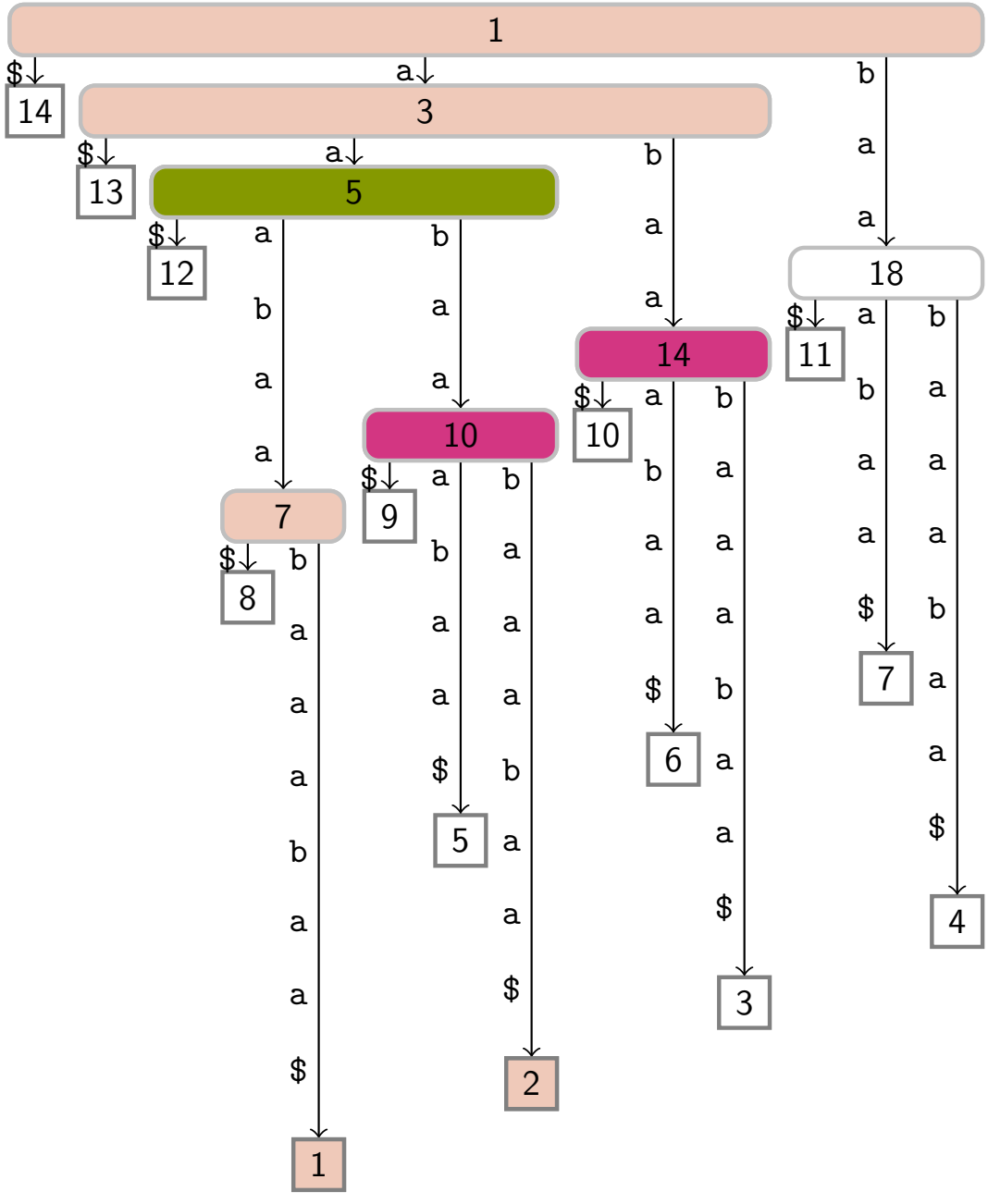
LZ77 parsing of

T = aaabaabaaabaa$:

a	aa	b	aabaa	abaa	$
-	1	-	2	3	-

Mapping W

5	10	14
1	2	3



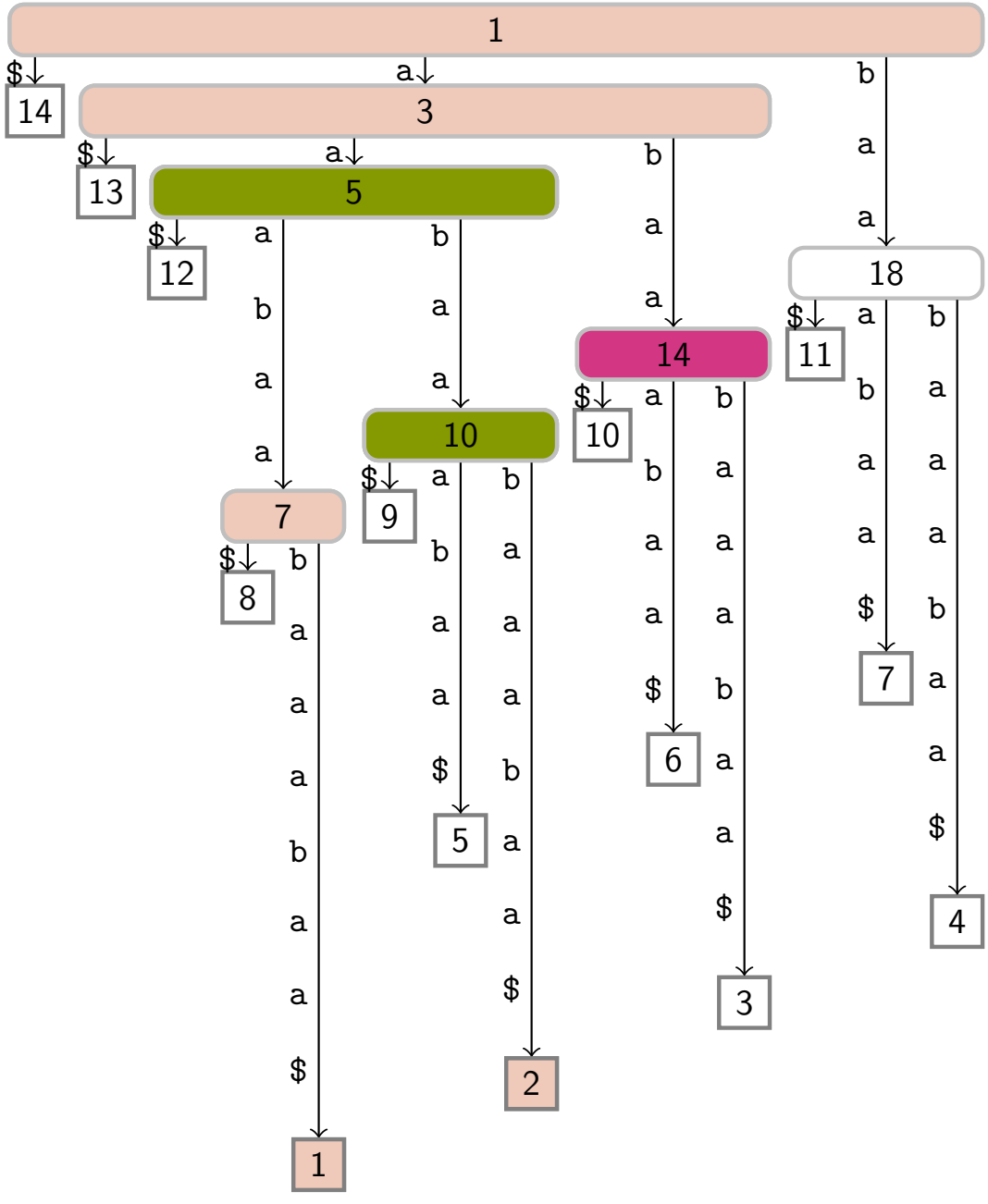
LZ77 parsing of

T = aaabaabaaabaa$:

a	aa	b	aabaa	abaa	$
-	1	-	2	3	-

Mapping W

5	10	14
1	2	3



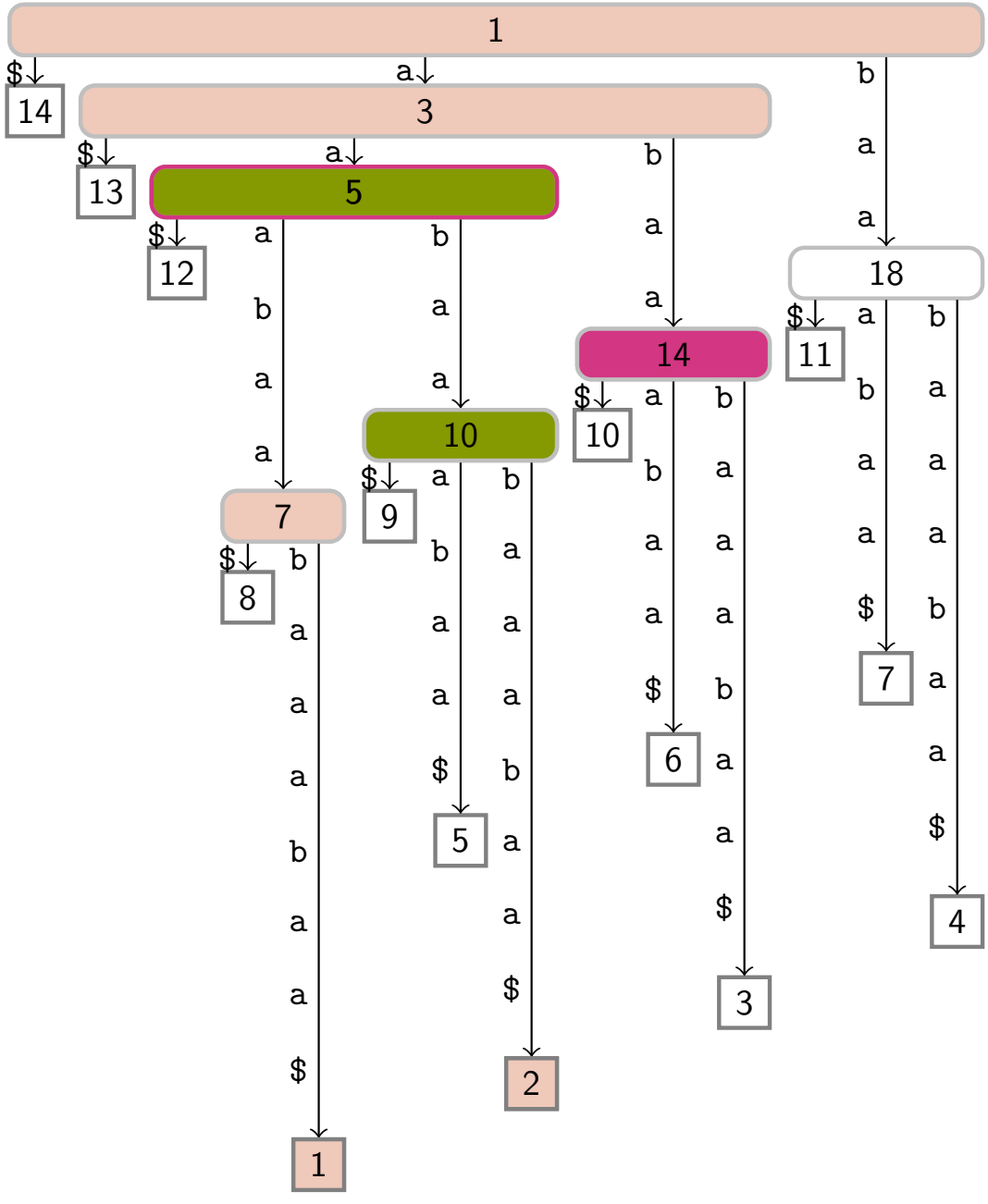
LZ77 parsing of

T = aaabaabaaabaa$:

a	aa	b	aabaa	abaa	$
-	1	-	2	3	-

Mapping W

5	10	14
1	2	3



LZ77 parsing of

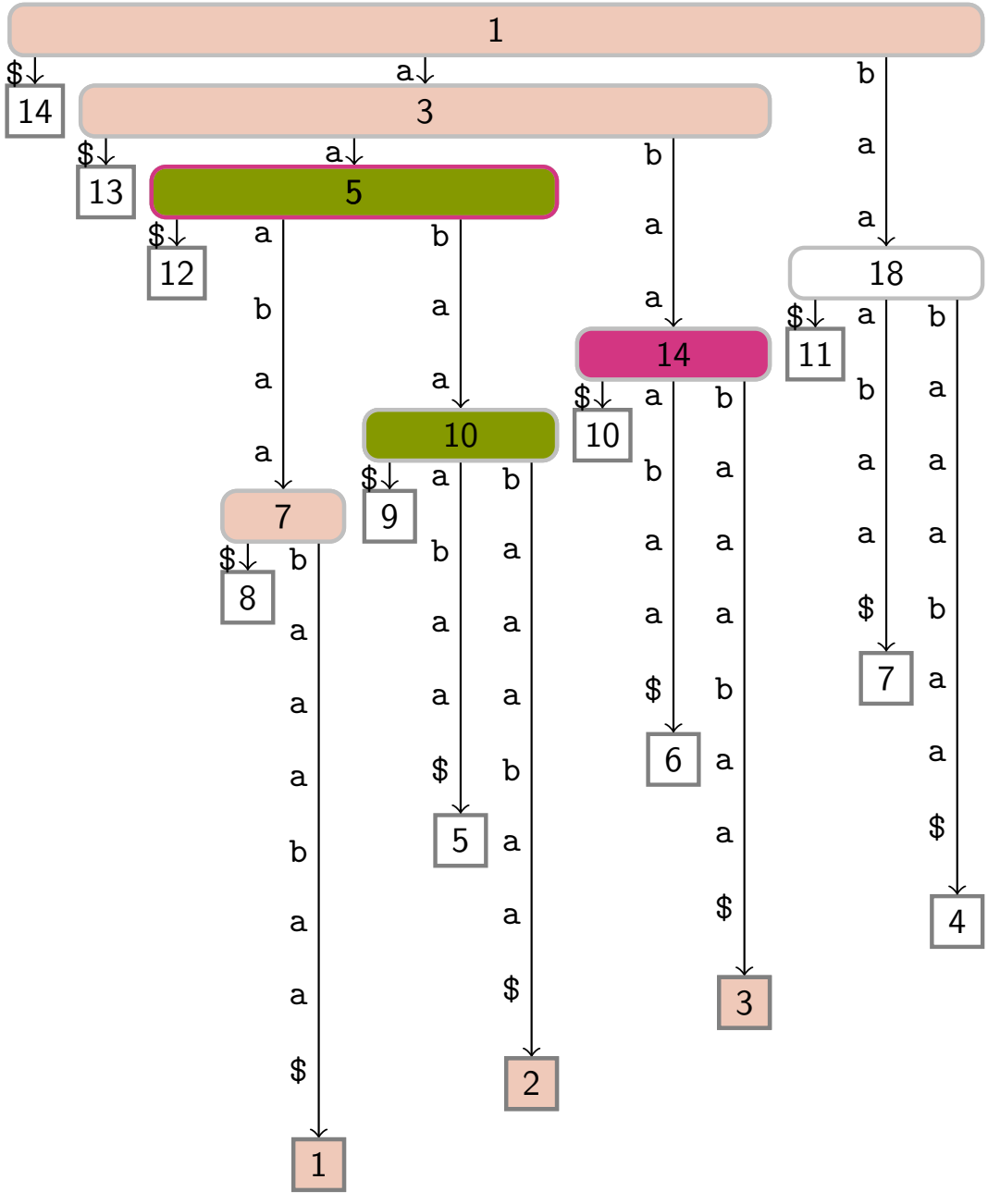
T = aaabaabaaabaa$:

a	aa	b	aabaa	abaa	$
-	1	-	2	3	-

Mapping W

	5	10	14
	1	2	3

factor 2 refers to 1



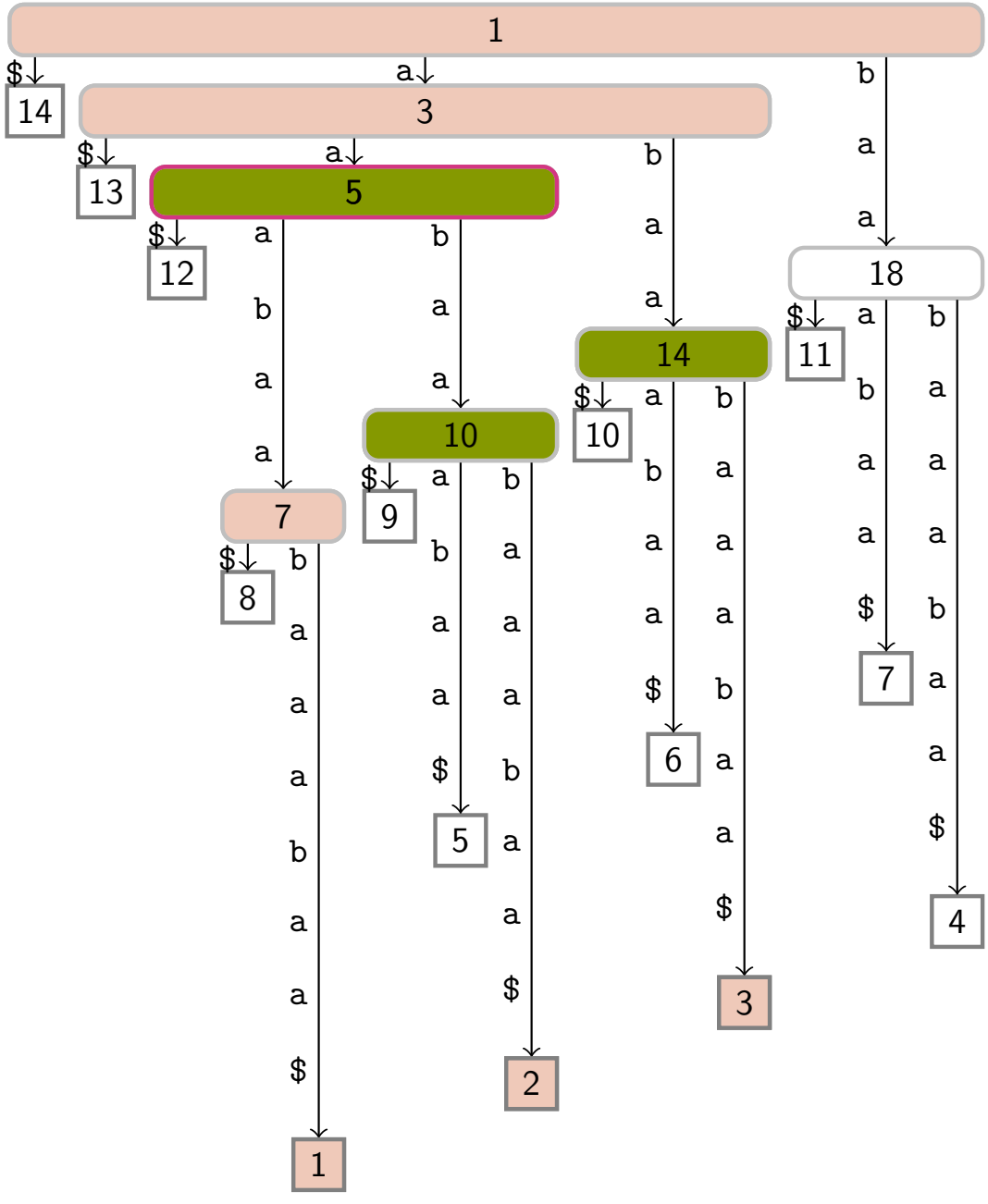
LZ77 parsing of

T = aaabaabaaabaa$:

a	aa	b	aabaa	abaa	$
-	1	-	2	3	-

Mapping W

5	10	14
1	2	3



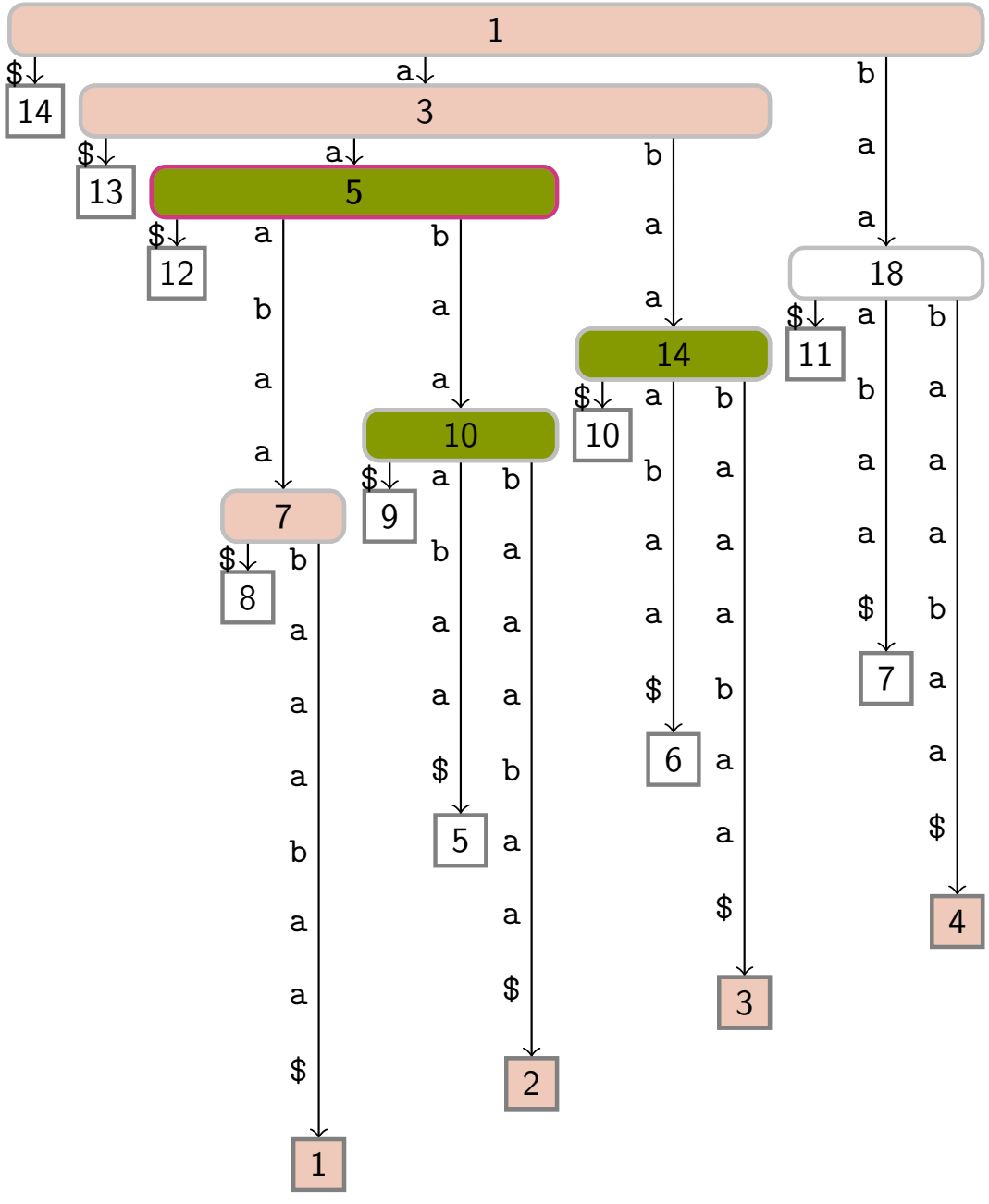
LZ77 parsing of

T = aaabaabaaabaa$:

a	aa	b	aabaa	abaa	$
-	1	-	2	3	-

Mapping W

5	10	14
1	2	3



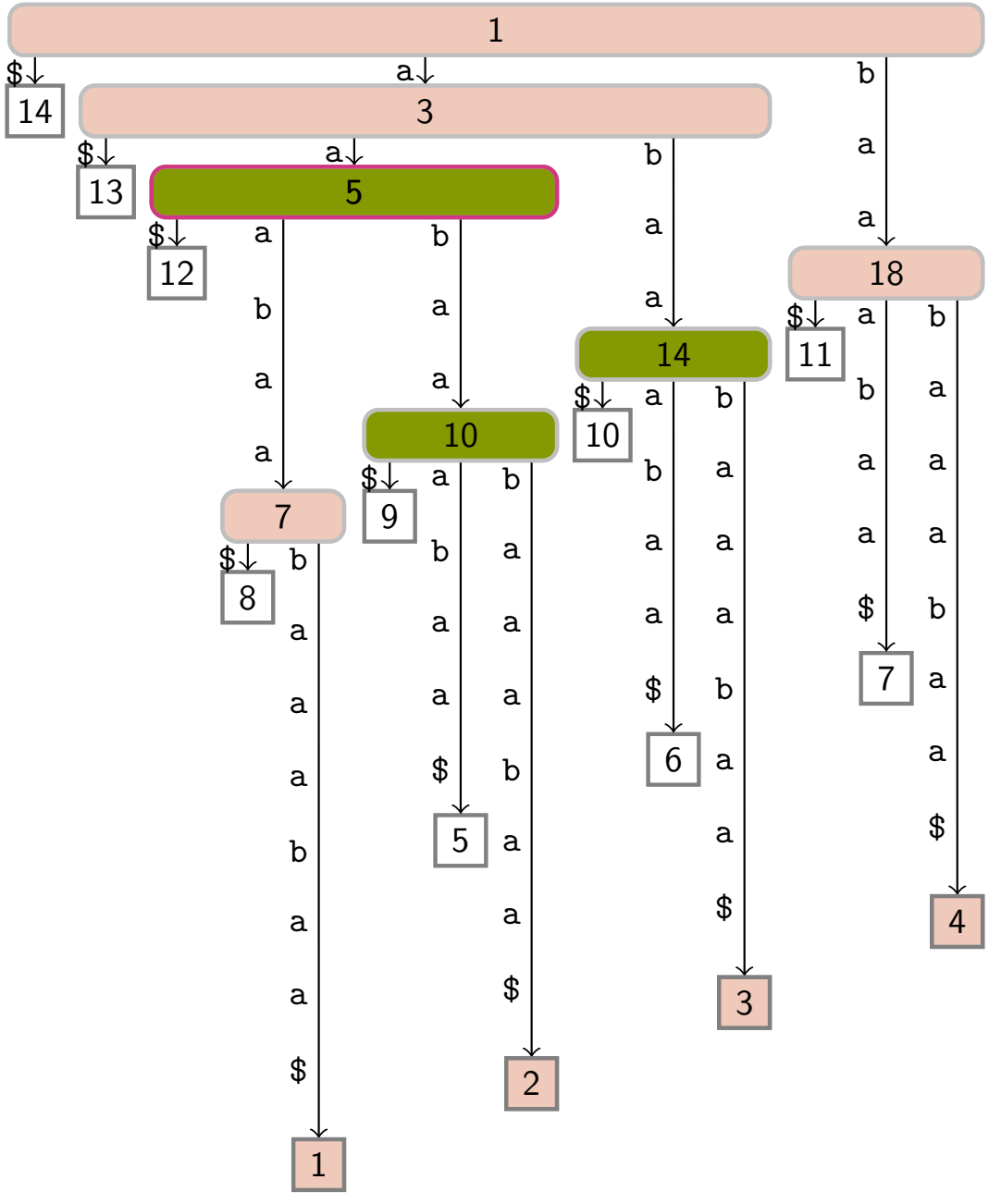
LZ77 parsing of

T = aaabaabaaabaa$:

a	aa	b	aabaa	abaa	$
-	1	-	2	3	-

Mapping W

5	10	14
1	2	3



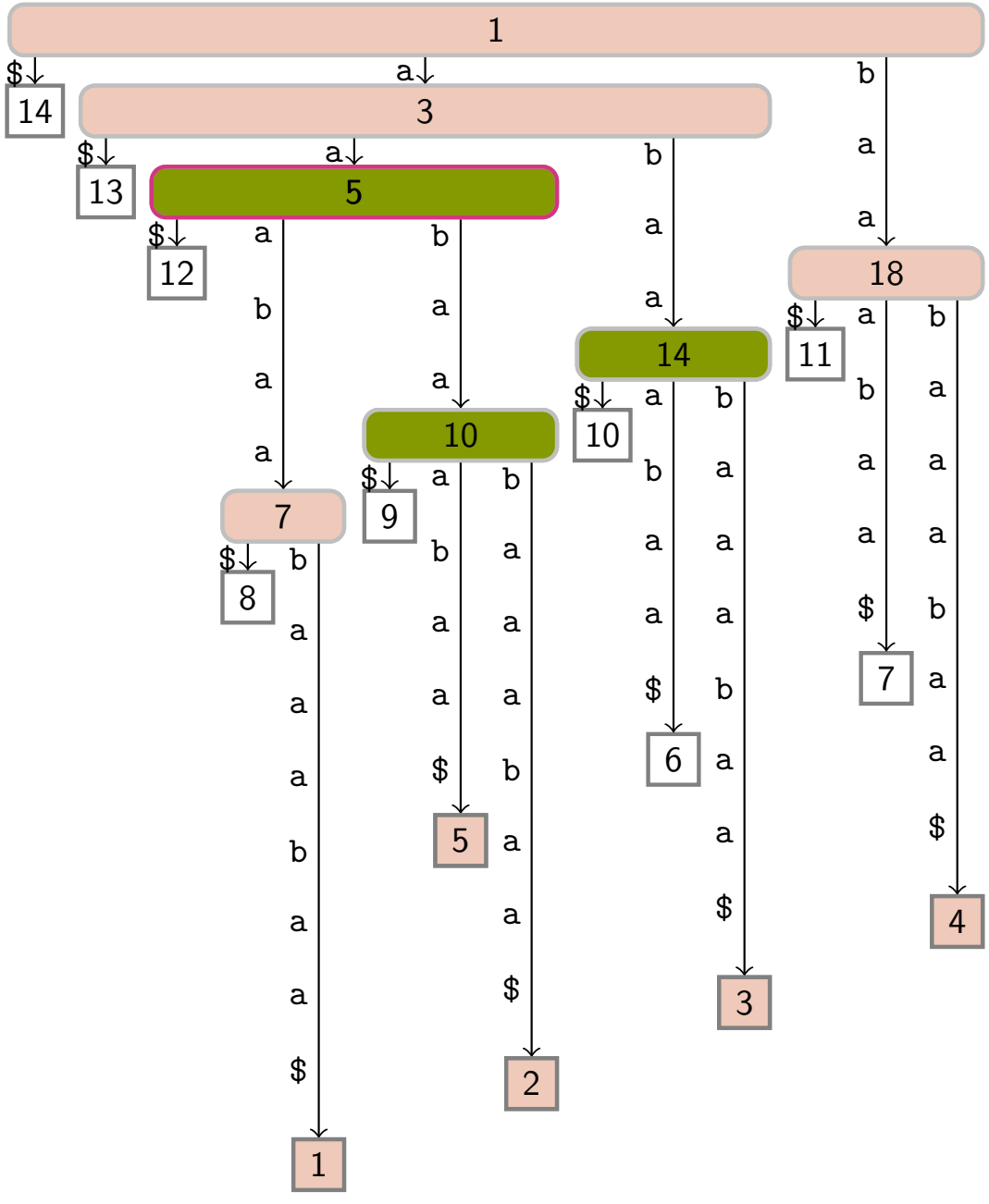
LZ77 parsing of

T = aaabaabaaabaa$:

a	aa	b	aabaa	abaa	$
-	1	-	2	3	-

Mapping W

5	10	14
1	2	3



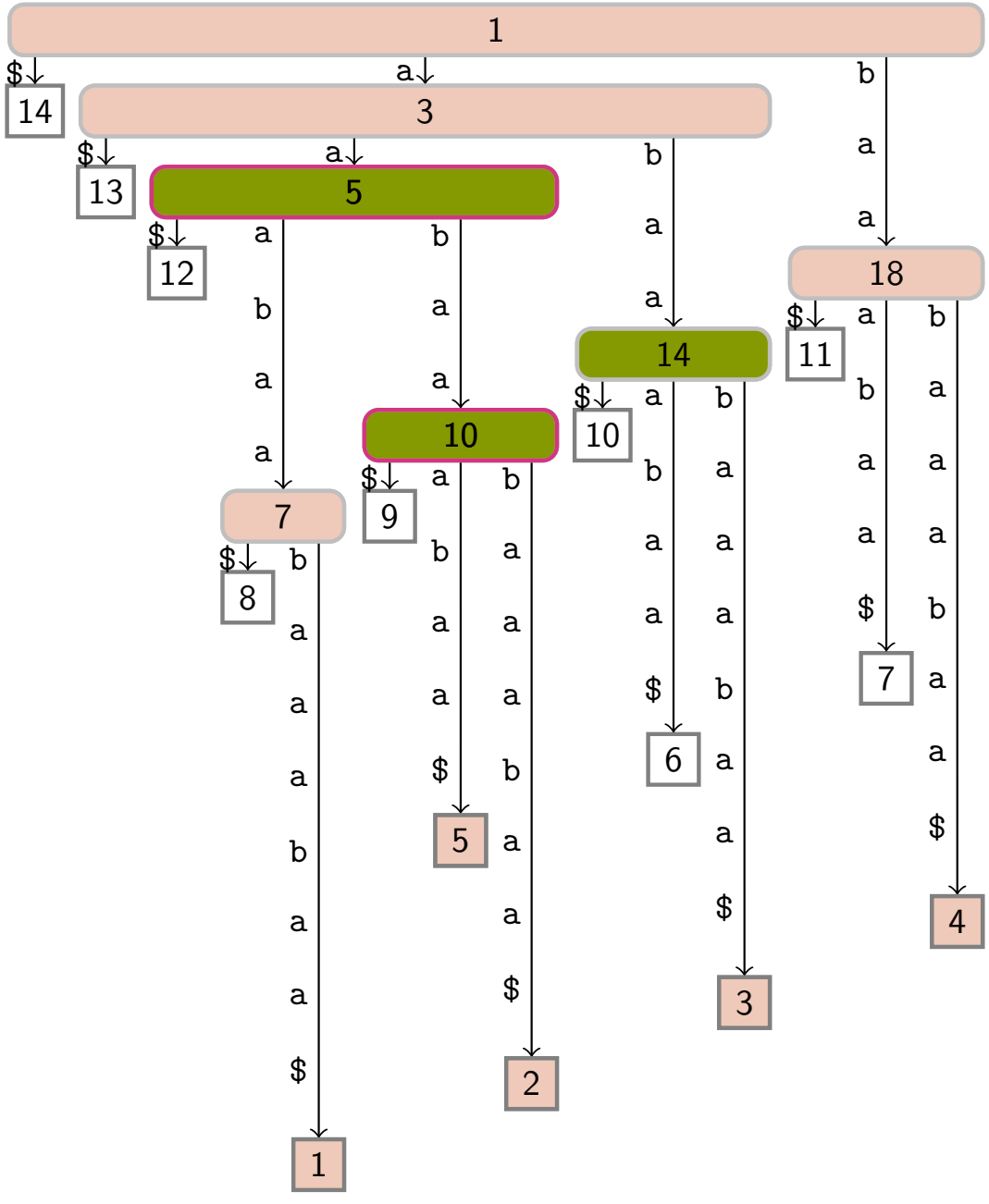
LZ77 parsing of

T = aaabaabaaabaa$:

a	aa	b	aabaa	abaa	$
-	1	-	2	3	-

Mapping W

5	10	14
1	2	3



LZ77 parsing of

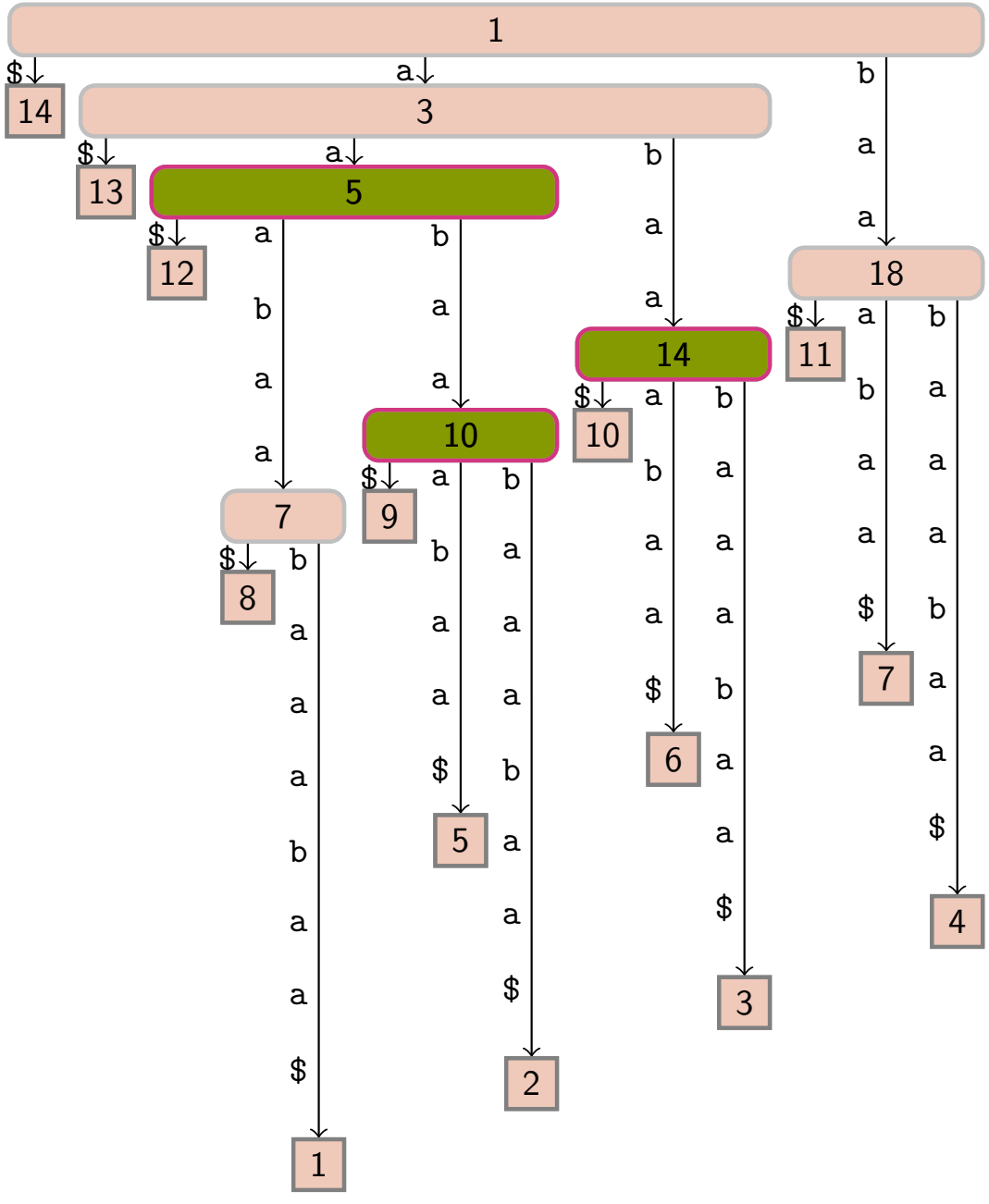
T = aaabaabaaabaa$:

a	aa	b	aabaa	abaa	$
-	1	-	2	3	-

Mapping W

5	10	14
1	2	3

factor 4 refers to 2



LZ77 parsing of

T = aaabaabaaabaa$:

a	aa	b	aabaa	abaa	$
-	1	-	2	3	-

Mapping W

5	10	14
1	2	3

LZ77

two passes over ST

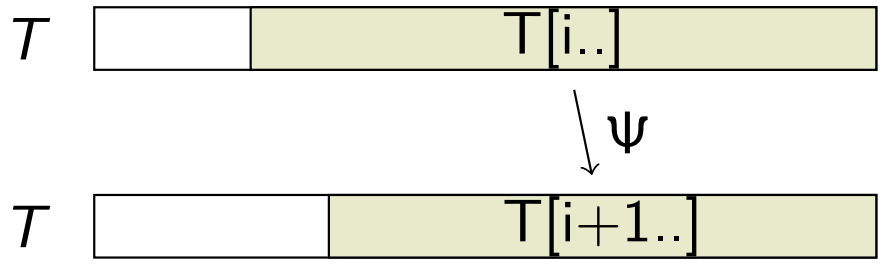
1. pass
   - traverse from *every* leaf to the root
   - mark visited nodes
   - already marked nodes $\equiv$ some reference
   - these nodes witness references
2. pass
   - same procedure
   - we know witnesses already!
   - which leaf discovers a witness first?

tricks & techniques

compressed suffix tree (CST)

lightweight ST

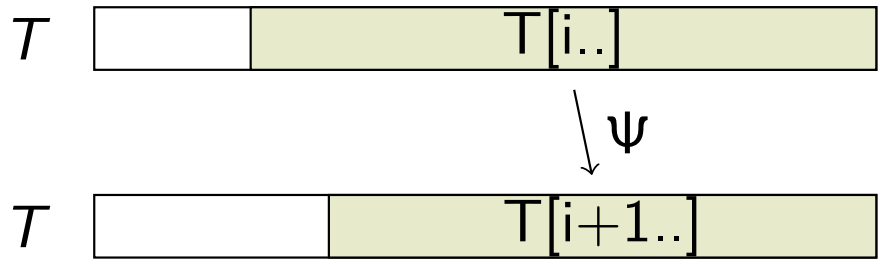
- Ψ array
- BP tree topology



compressed suffix tree (CST)

lightweight ST

- Ψ array
- BP tree topology



Theorem (Belazzougui'15)

can construct CST in

- *time*
  - $\mathcal{O}(n)$ *randomized or*
  - $\mathcal{O}(n \lg \lg \sigma)$ *deterministic*
- $\mathcal{O}(n \lg \sigma)$ *space*

traversal simulation

Problem

ad-hoc not efficient in CST

- *select leaf with given label*
- *read string from root to given node (string depth)*

leaf access

- leaf $`1' = \Psi[1]$
- nextleaf : (leaf $`k'$) $\mapsto$ (leaf $`k+1'$)
  - Ψ
  - $\text{rank}_{()}$ on BP
  - $\text{select}_{()}$ on BP

traversal simulation

Problem

ad-hoc not efficient in CST

- *select leaf with given label*
- *read string from root to given node (string depth)*

string depth

- *head*(ℓ) : first character on the path from root to leaf ℓ
  - make alphabet effective
  - answer: *child_rank* on highest ancestor $\neq$ root
- *str_depth*(v)
  - Ψ
  - head
  - time linear to string depth
  - used for factor length computation $\Rightarrow \mathcal{O}(n)$ time overall

summary

Theorem

given text T of length n, LZ77 of T can be computed with

- *time*
  - $\mathcal{O}(n)$ *randomized*
  - $\mathcal{O}(n \lg \lg \sigma)$ *deterministic*
- $\mathcal{O}(n \lg \sigma)$ *bits space*

same holds for LZ78! (see paper)

techniques:

- compressed suffix tree
- simulating suffix array with Ψ
- indirect matching (witnesses).

Thank you for listening. Any questions are welcome!

summary

Theorem

given text T of length n, LZ77 of T can be computed with

- *time*
  - $\mathcal{O}(n)$ *randomized*
  - $\mathcal{O}(n \lg \lg \sigma)$ *deterministic*
- $\mathcal{O}(n \lg \sigma)$ *bits space*

same holds for LZ78! (see paper)

techniques:

- compressed suffix tree
- simulating suffix array with Ψ
- indirect matching (witnesses).

Thank you for listening. Any questions are welcome!