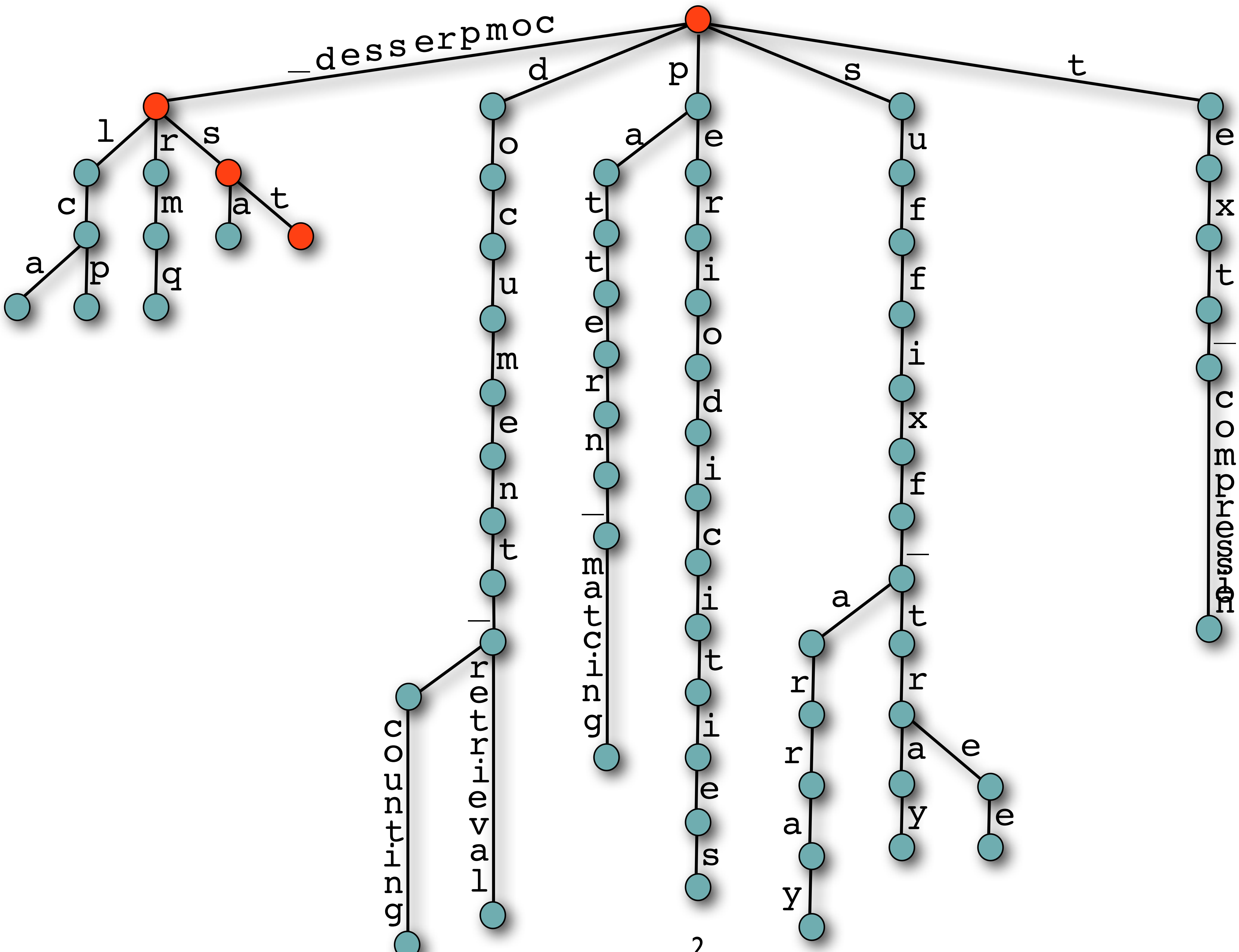
Lecture 10: Compressed Suffix Trees

Johannes Fischer

Topics



_desserpmoc
d
p
s
t
l
r
s
c
m
a
t
a
p
q
o
c
u
m
e
n
t
_
counting
retrieval
a
e
t
t
e
r
n
_
matching
r
i
o
d
i
c
i
t
i
e
s
u
f
f
i
x
f
_
a
r
r
a
y
t
r
a
y
e
e
e
x
t
_
compression

What we already have...

- Text + BWT + WT and backwards search

$\Rightarrow$ O(m lg σ) counting queries for $P[1,m]$

▸ space O(n lg σ) **bits** (text size!)

- + sampled suffix array values

$\Rightarrow$ O(k lg n) for enumerating k occurrences

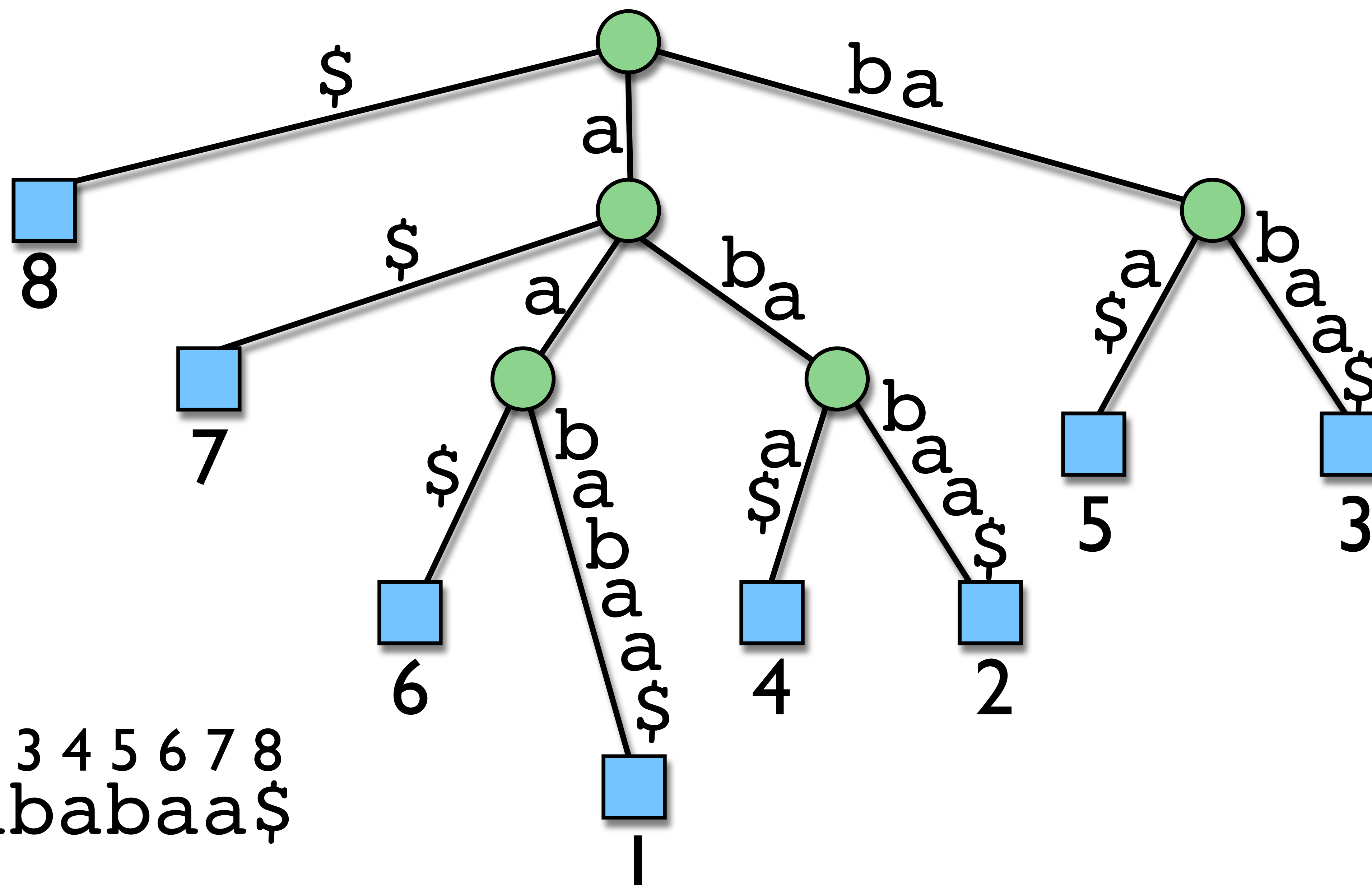
▸ can be improved to O(k $\lg^{\varepsilon}$ n) for $\varepsilon<1$

Suffix Tree Functionality

- often, **more functionality** is desired
  - repeat recognition (e.g. $T=\alpha\boldsymbol{\rho}\beta\boldsymbol{\rho}\gamma$)
  - tandem repeats (e.g. $T=\alpha\boldsymbol{\rho}\boldsymbol{\rho}\beta$)
  - longest common substrings
  - matching statistics
  - suffix-prefix matches
  - etc.
- want **suffix tree** functionality!

compress?

Suffix Tree



$$T = \overset{1}{\texttt{a}}\overset{2}{\texttt{a}}\overset{3}{\texttt{b}}\overset{4}{\texttt{a}}\overset{5}{\texttt{b}}\overset{6}{\texttt{a}}\overset{7}{\texttt{a}}\overset{8}{\texttt{\$}}$$

A = 8,7,6,1,4,2,5,3

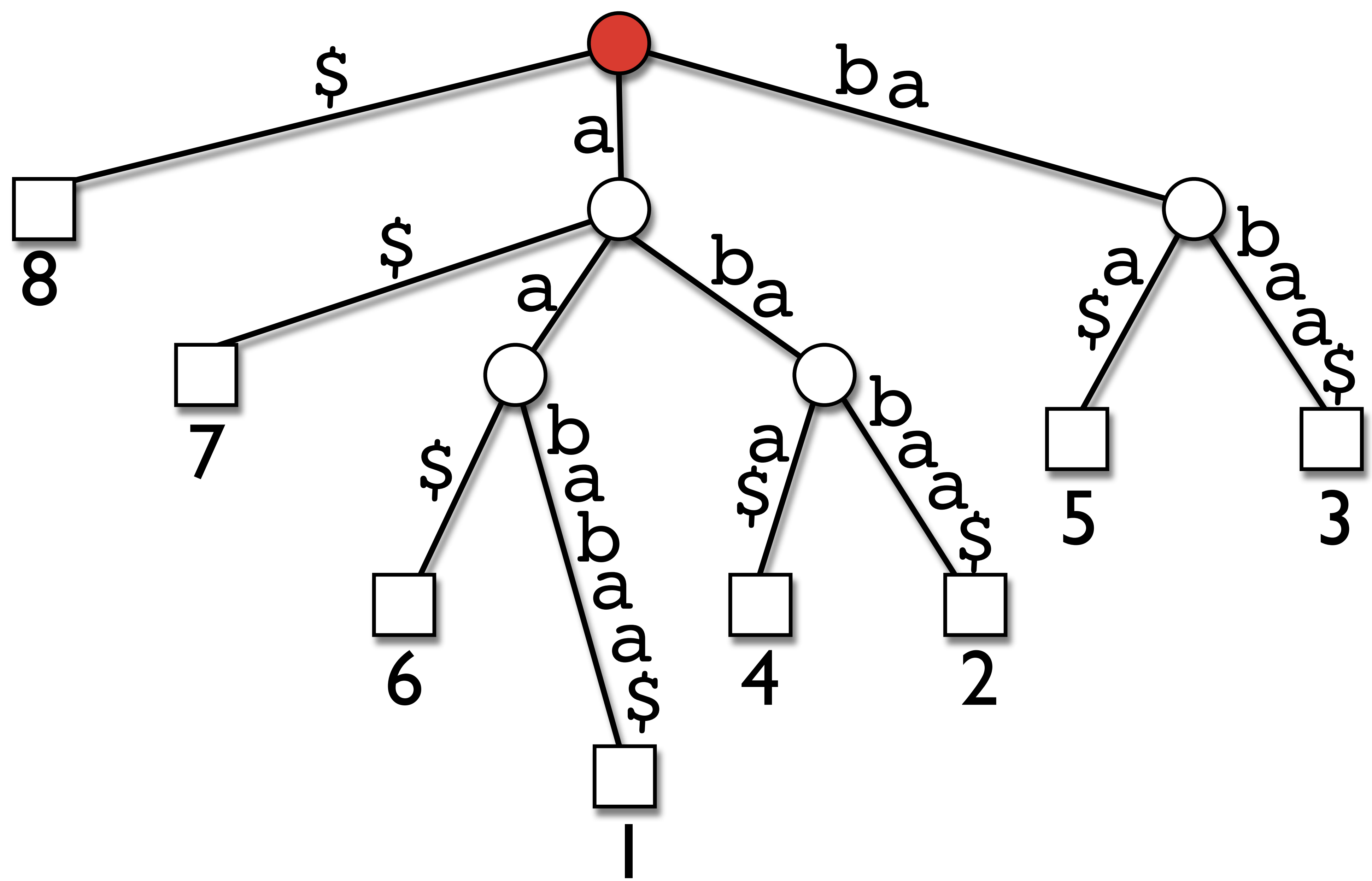
Some real numbers

- **guess**: how much bigger than text is ST?
  A: <10 B: 10-20 C: >20
- Suffix Tree
  - **20-40** times text size !!!
- Text+BWT+WT (incl. rank/select):
  - $\approx$**3** times text size
- **goal**: drop suffix tree and simulate operations using suffix- and LCP array

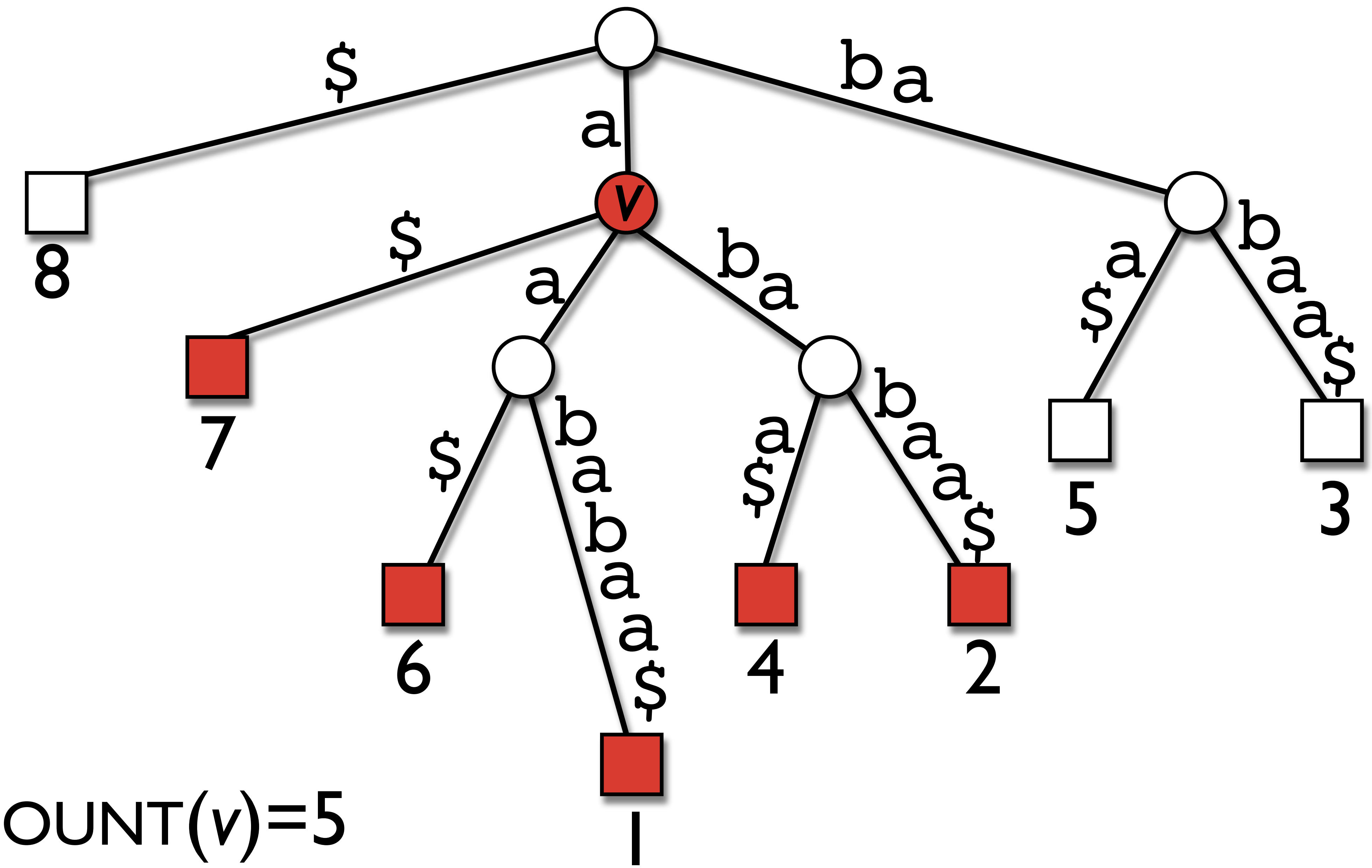
ST Operations

Operation	Description
ROOT()	return root
COUNT(*v*)	count leaves below *v*
ISANCESTOR(*v*,*w*)	true if *v* is an ancestor of *w*
ISLEAF (*v*)	true if *v* is a leaf
LEAFLABEL(*v*)	suffix number represented by leaf *v*
SDEPTH(*v*)	string depth of *v*
PARENT(*v*)	parent node of *v*
FIRSTCHILD(*v*)	first (alphabetically smallest) child of *v*
NEXTSIBLING(*v*)	next sibling of *v*
EDGELABEL(*v*,*i*)	*i*'th letter on the edge leading to *v*
LCA(*v*,*w*)	lowest common ancestor of *v* and *w*

Root()

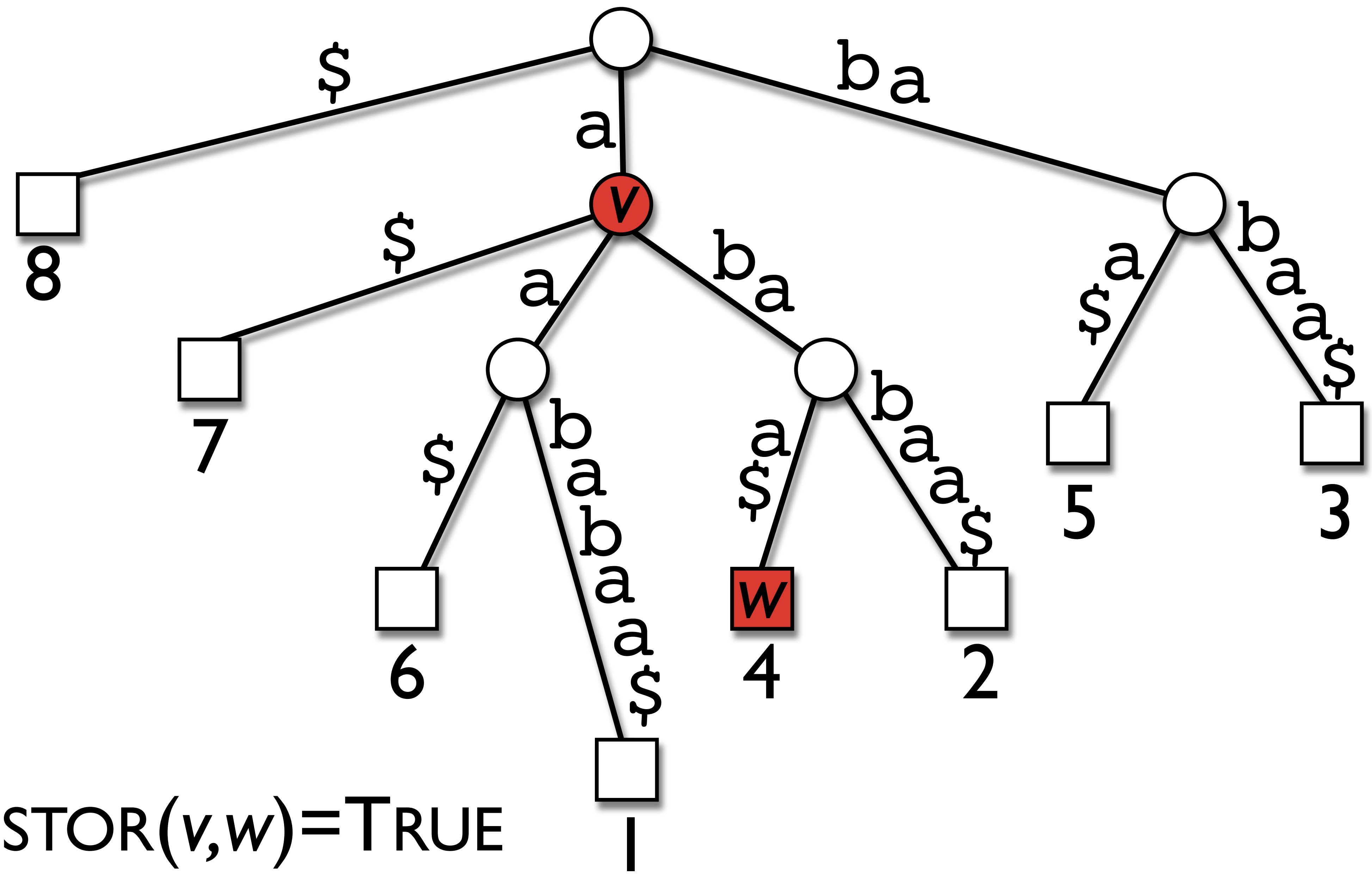


COUNT(v)



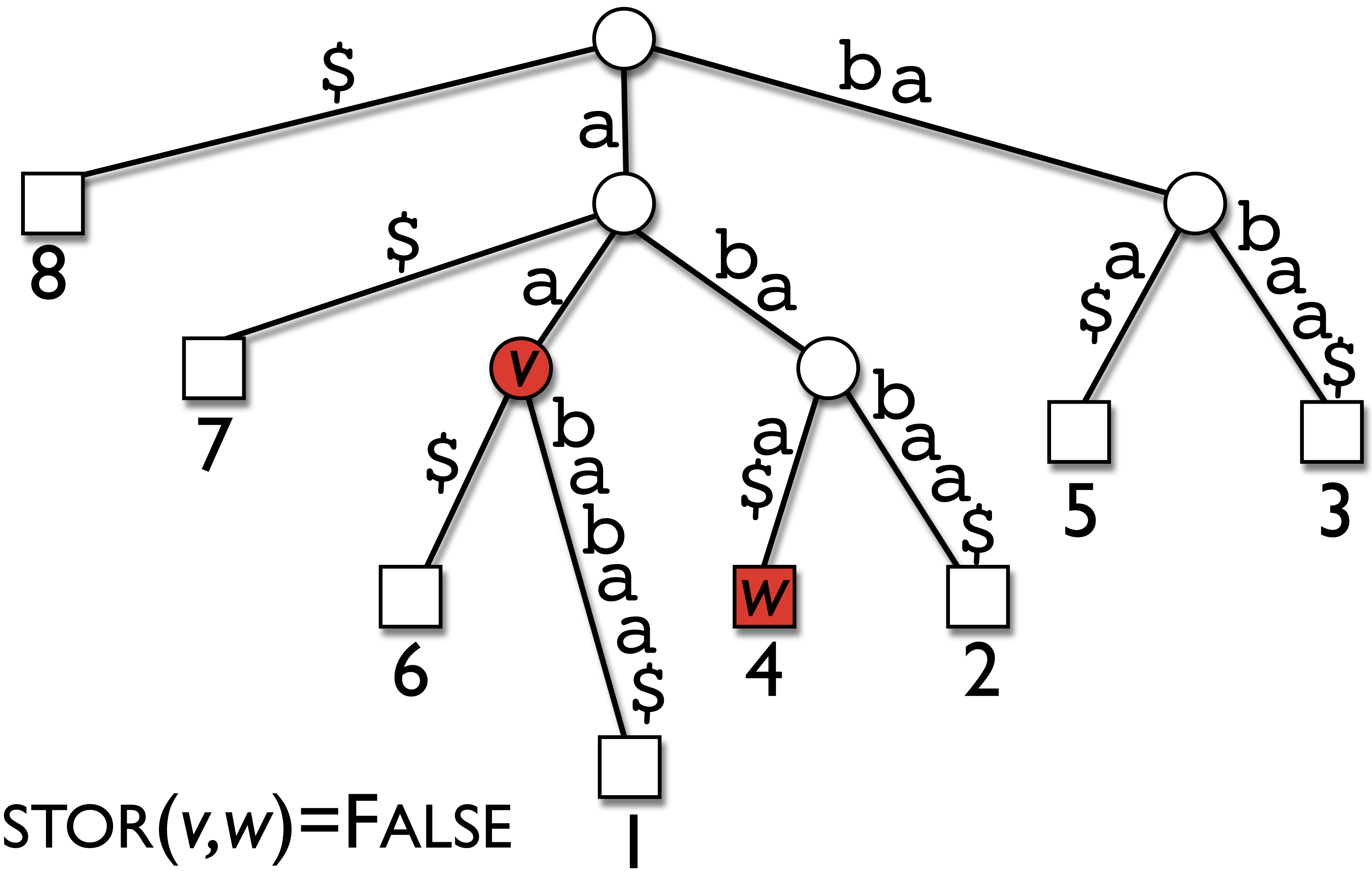
COUNT(v)=5

ISANCESTOR(*v*,*w*)



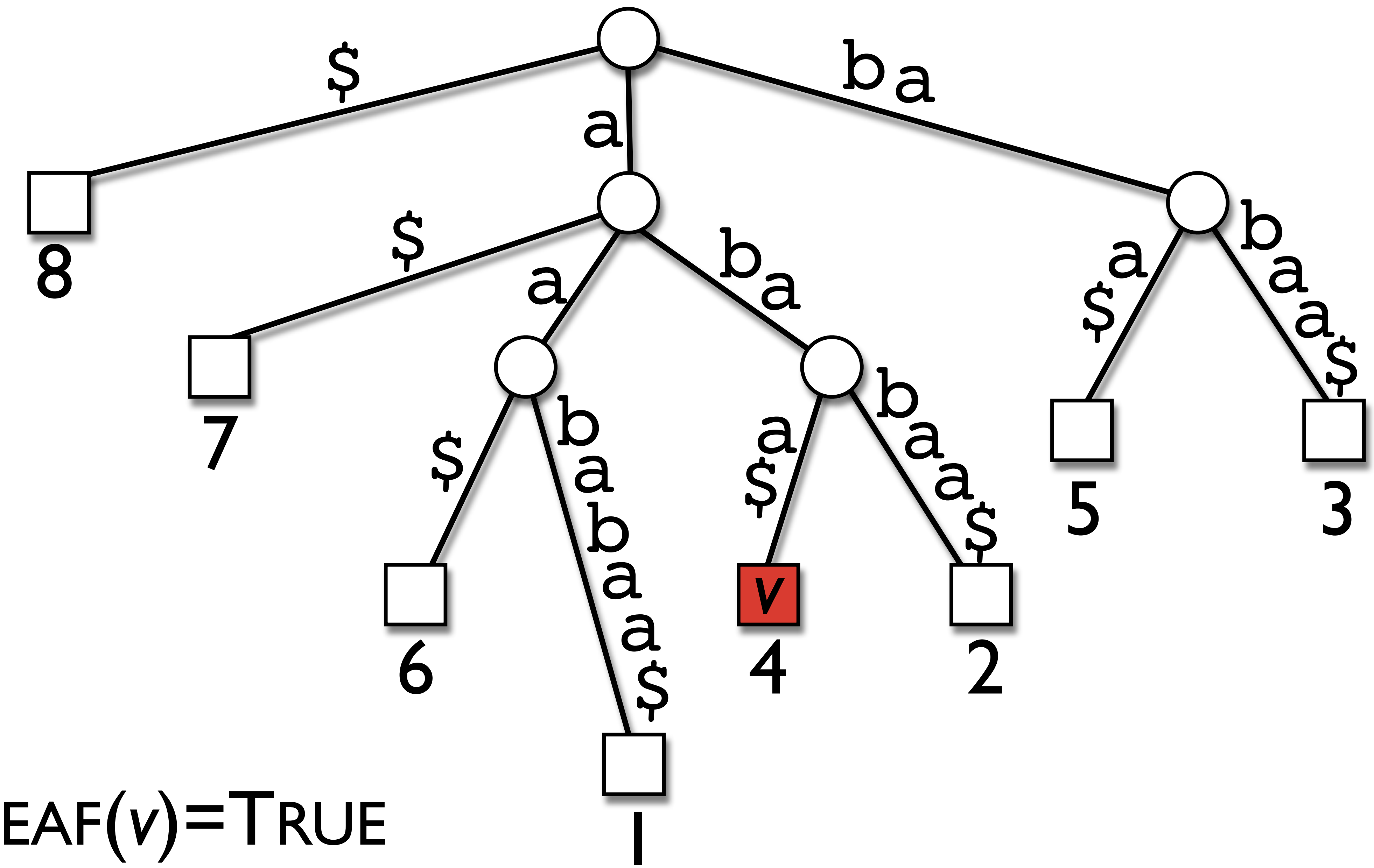
ISANCESTOR(*v*,*w*)=TRUE

IsAncestor(v,w)



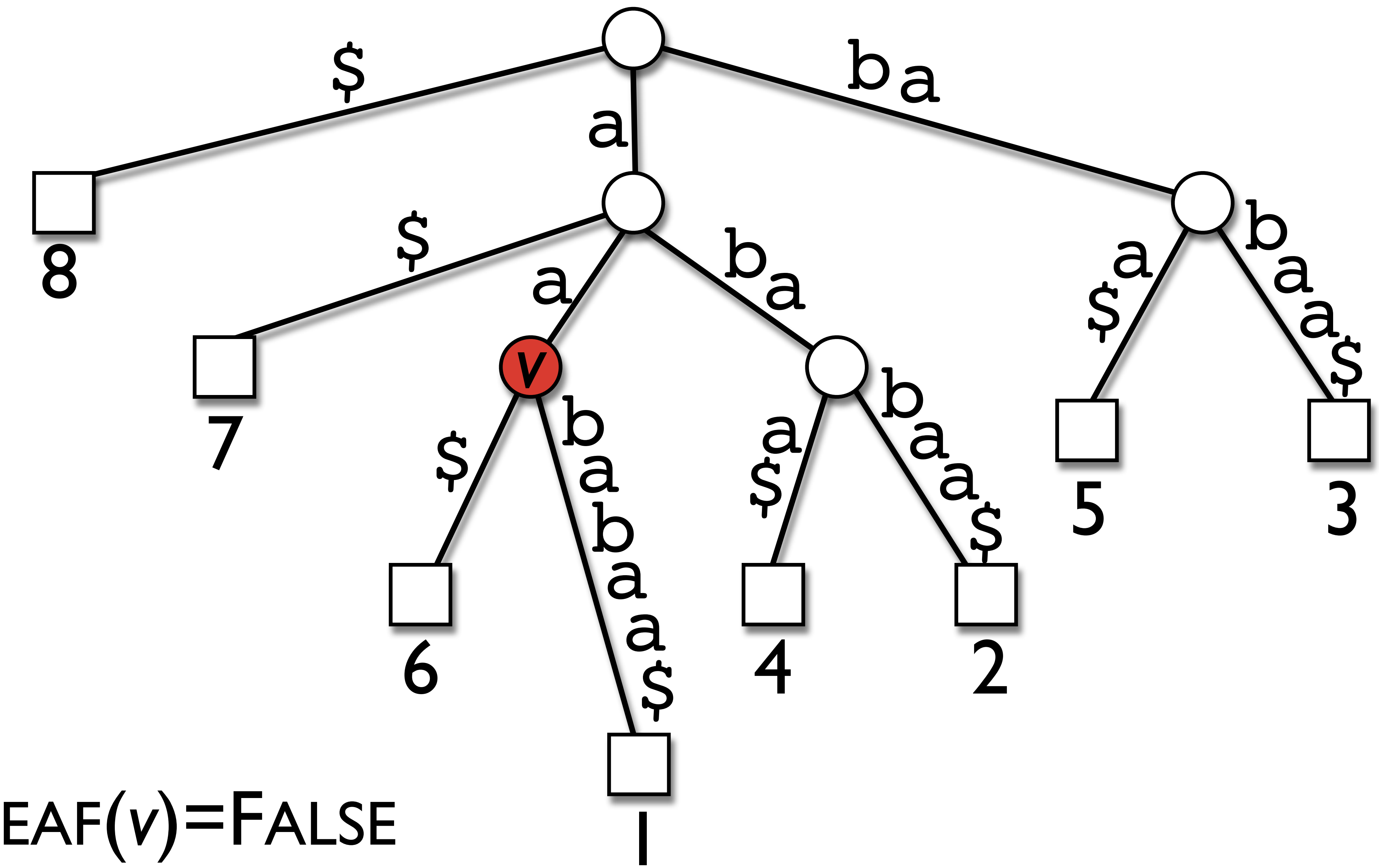
IsAncestor(v,w)=False

IsLeaf(v)



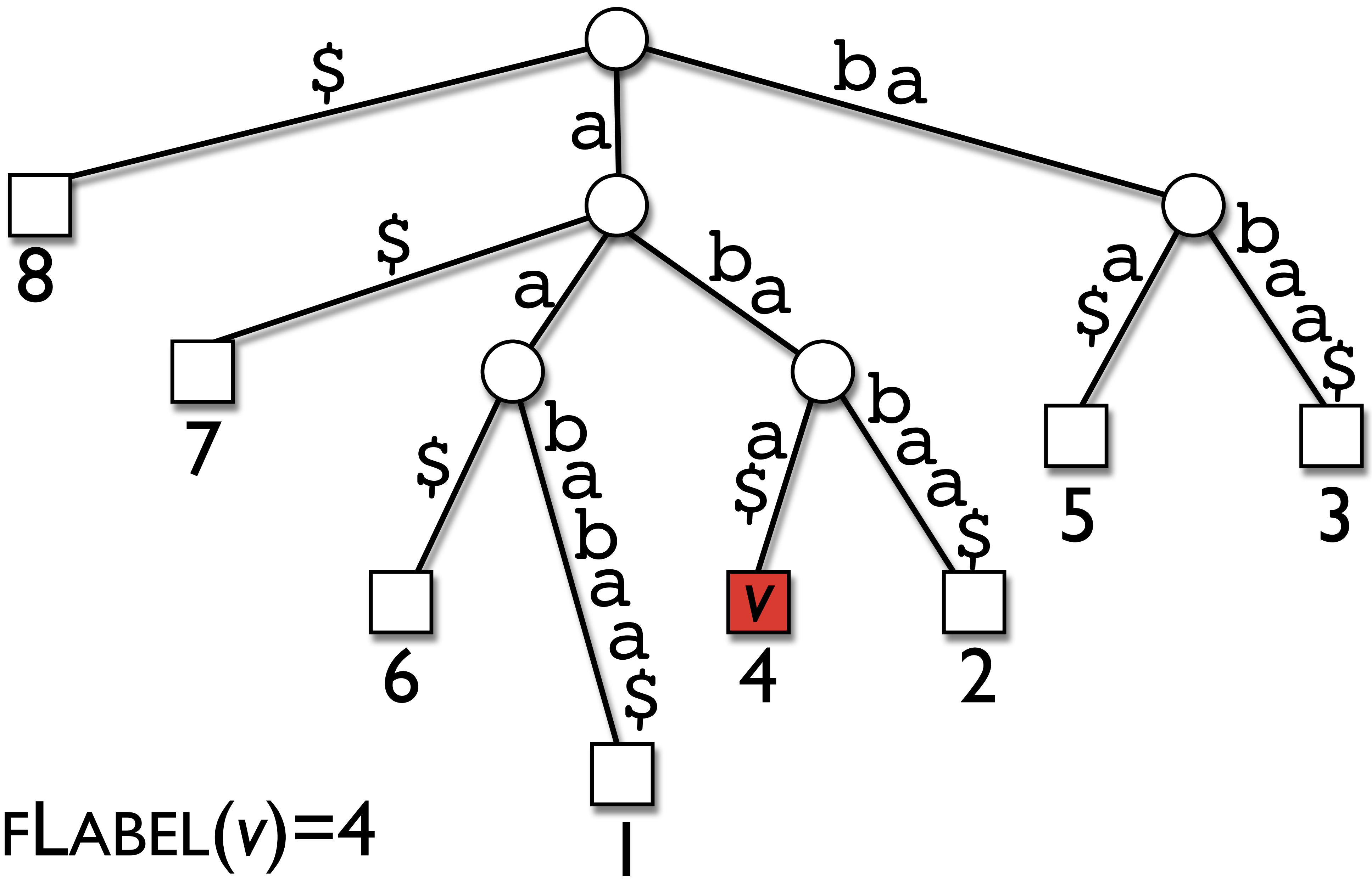
IsLeaf(v)=True

ISLEAF(*v*)



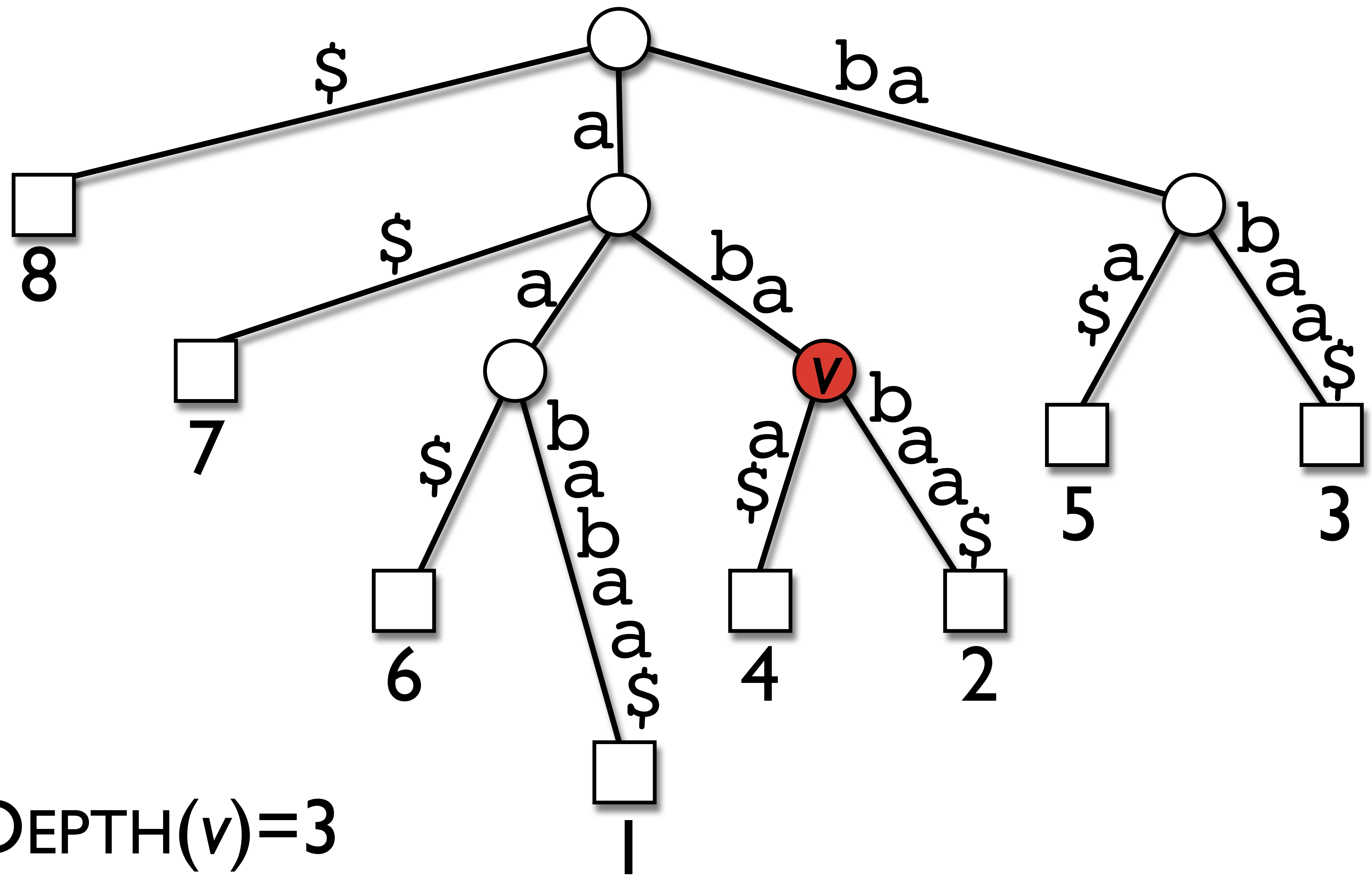
ISLEAF(*v*)=FALSE

LEAFLABEL(*v*)



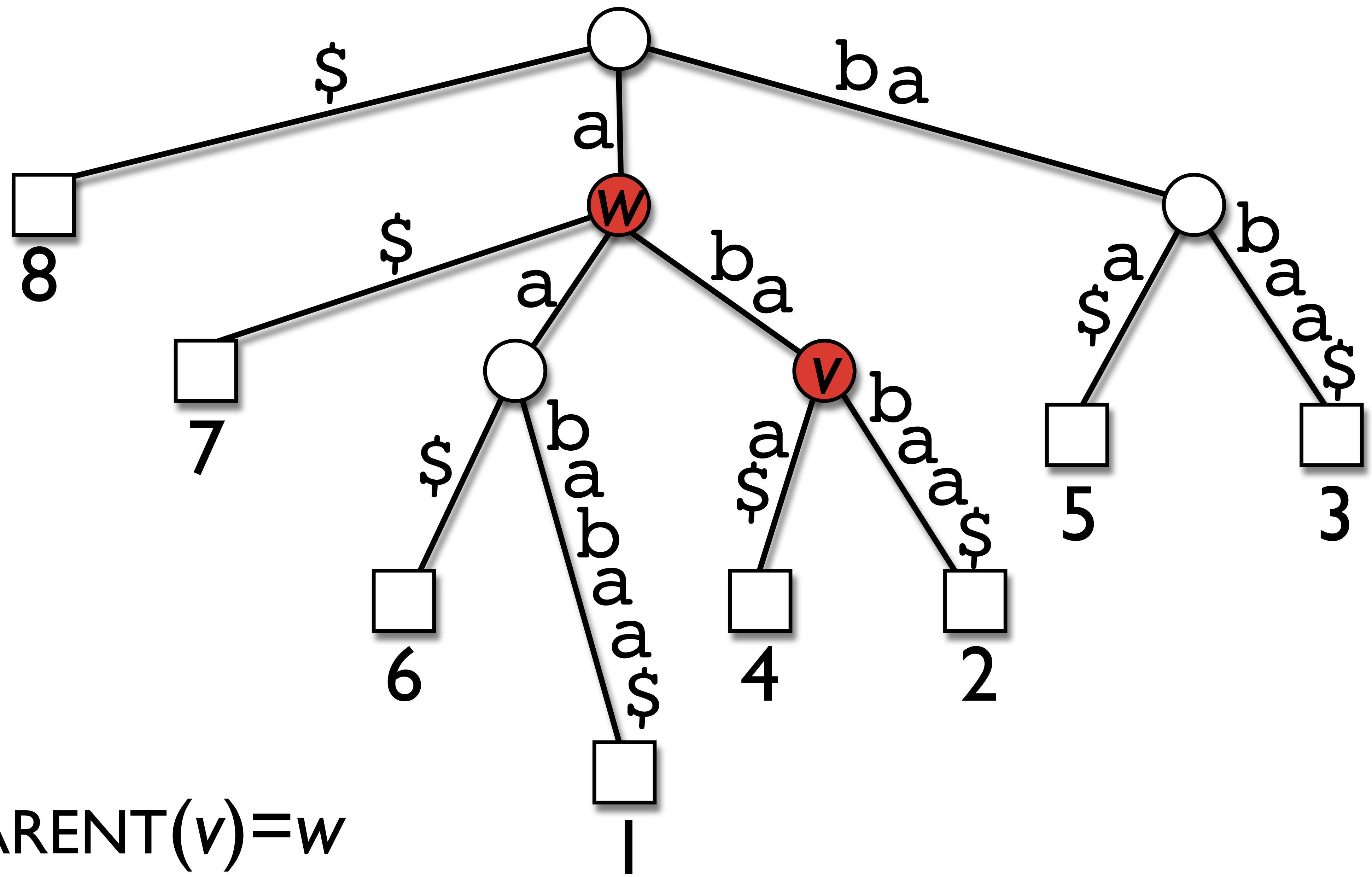
LEAFLABEL(*v*)=4

SDEPTH(v)



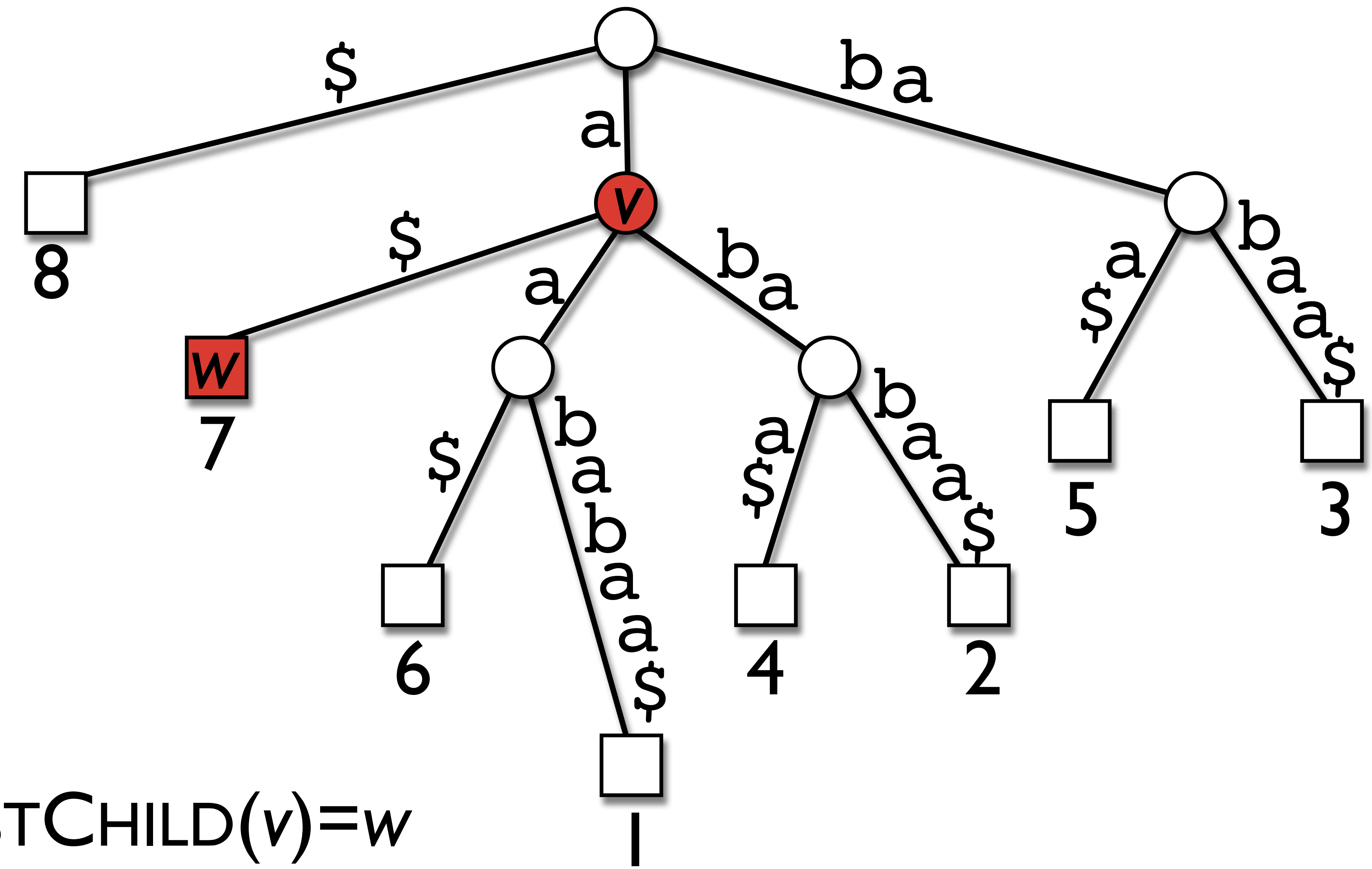
SDEPTH(v)=3

PARENT(v)



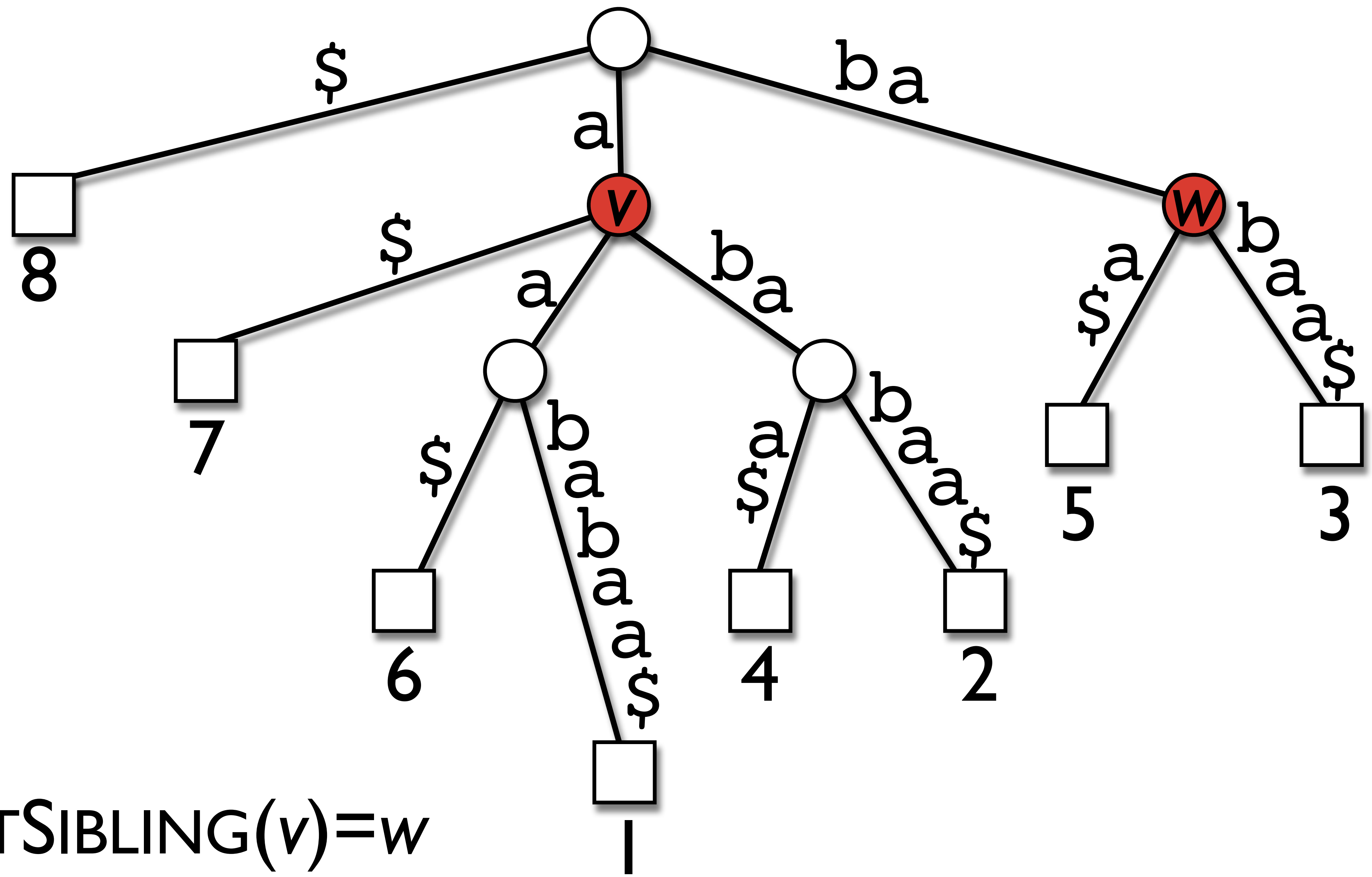
PARENT(v)=w

FIRSTCHILD(v)



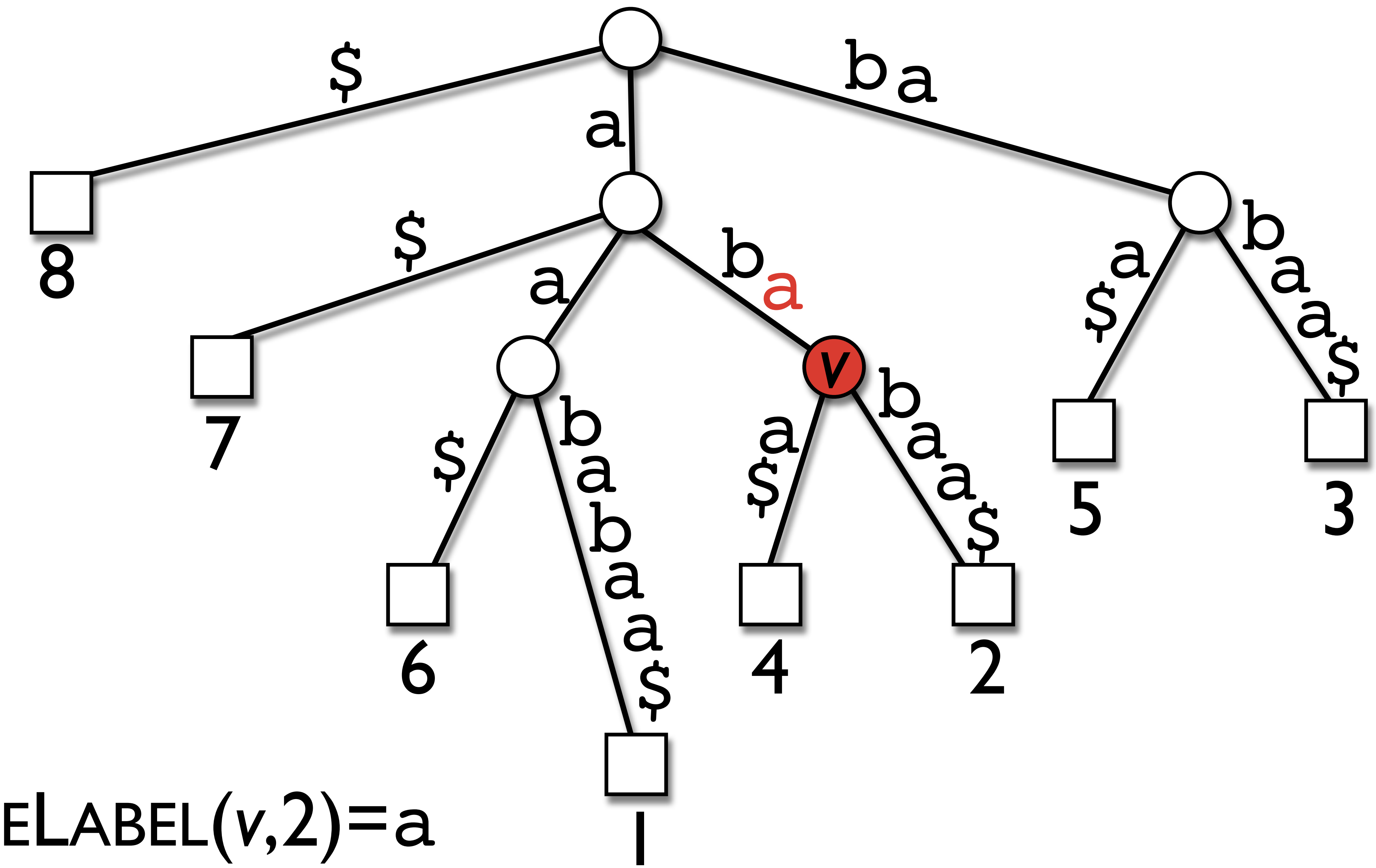
FIRSTCHILD(v)=w

NextSibling(v)



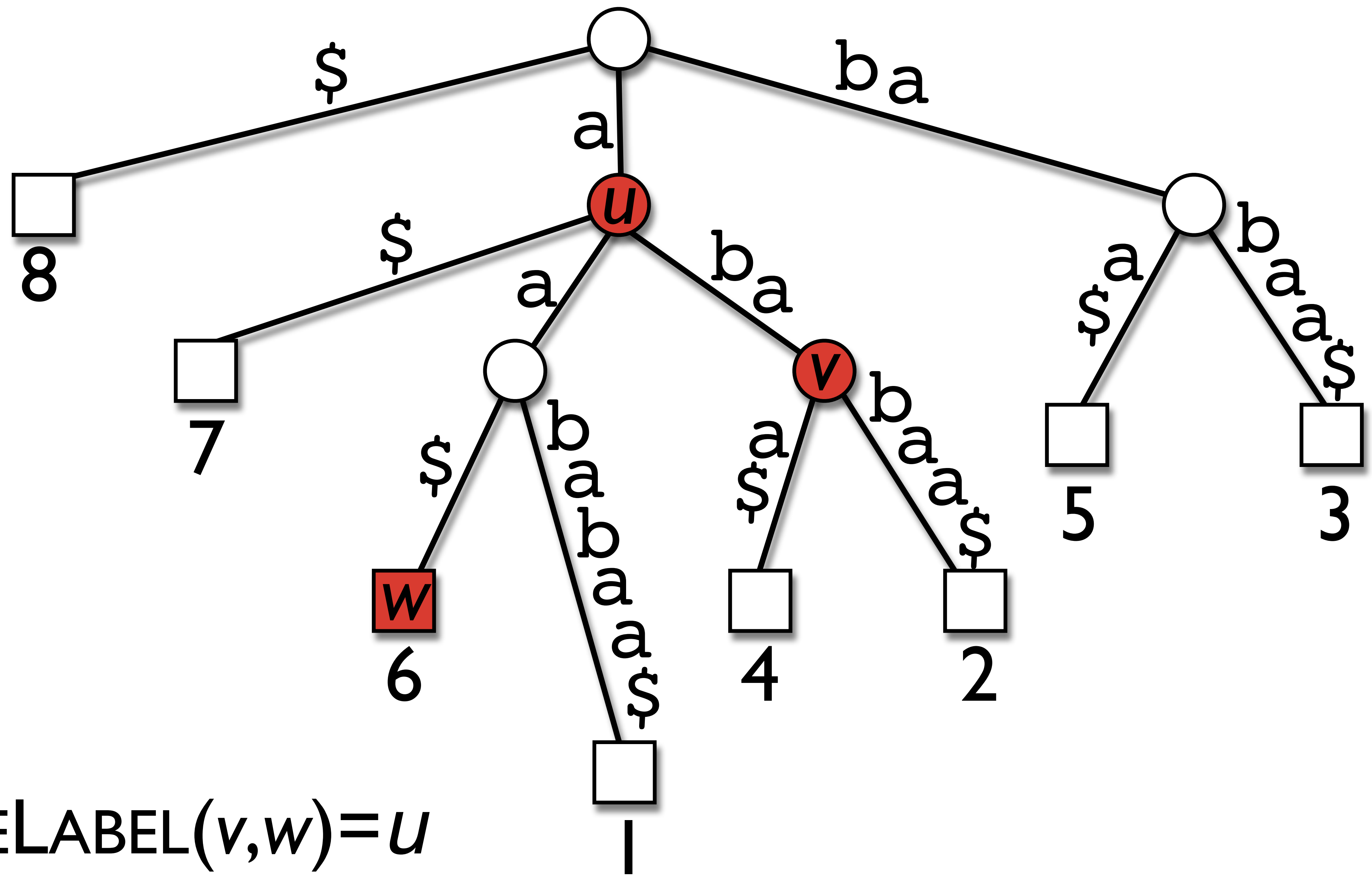
NextSibling(v)=w

EDGELABEL(*v*,*i*)



EDGELABEL(*v*,2)=a

LCA(v,w)



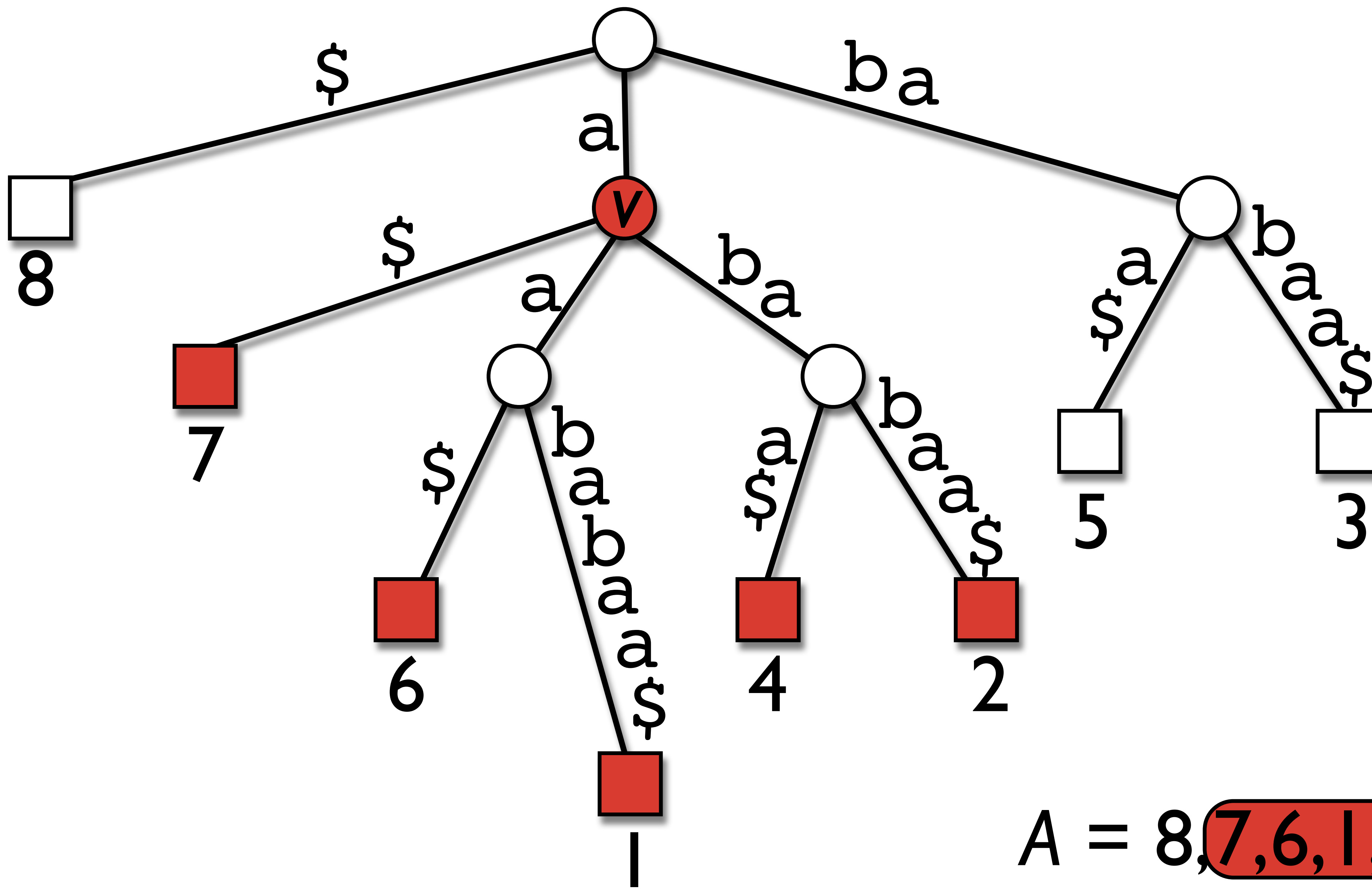
EDGELABEL(v,w)=u

Goal

drop suffix tree and **simulate** operations

using suffix- and LCP array

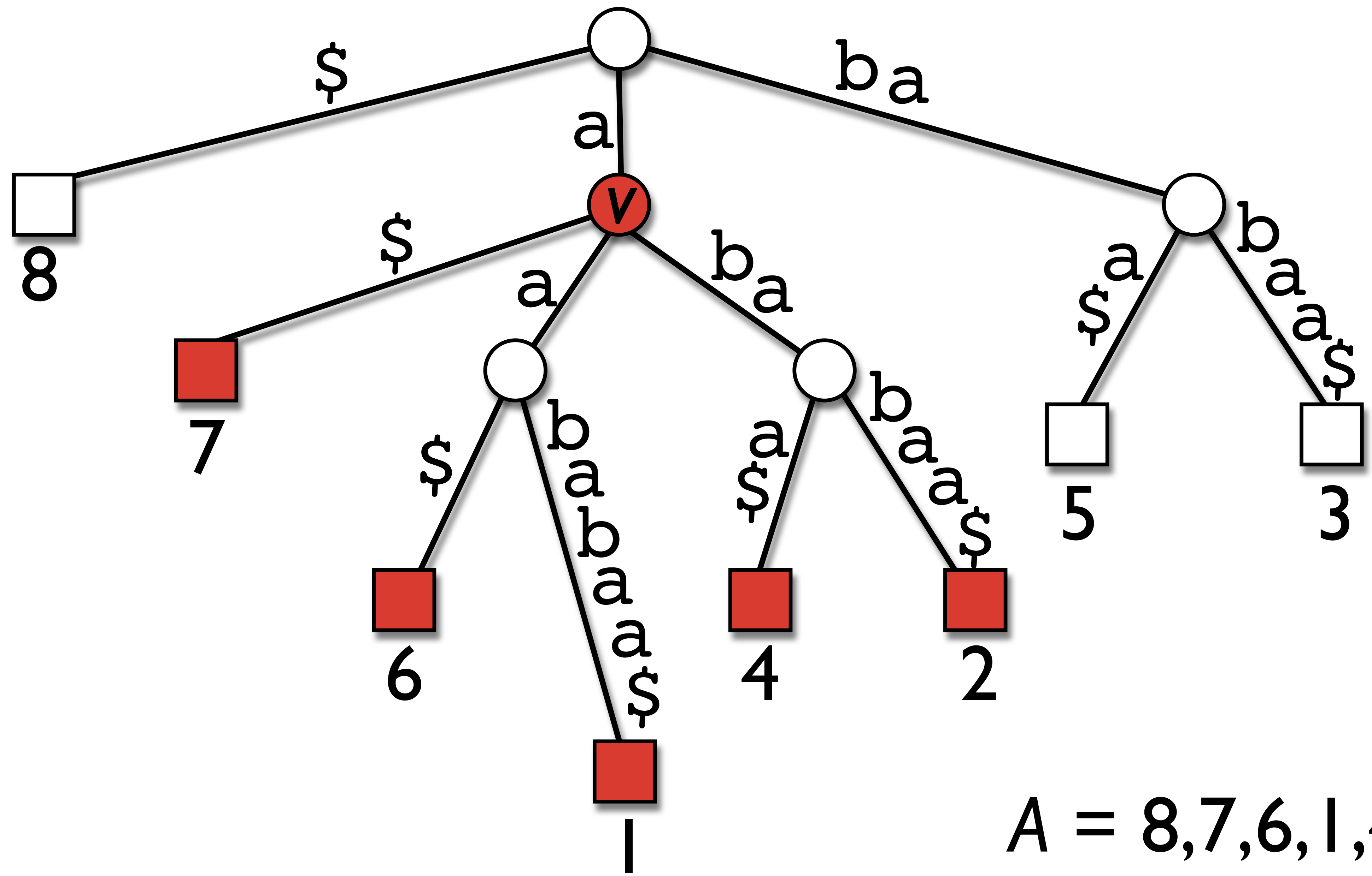
Represent Nodes by Intervals



A = 8,7,6,1,4,2,5,3

H = -1,0,1,2,1,3,0,2

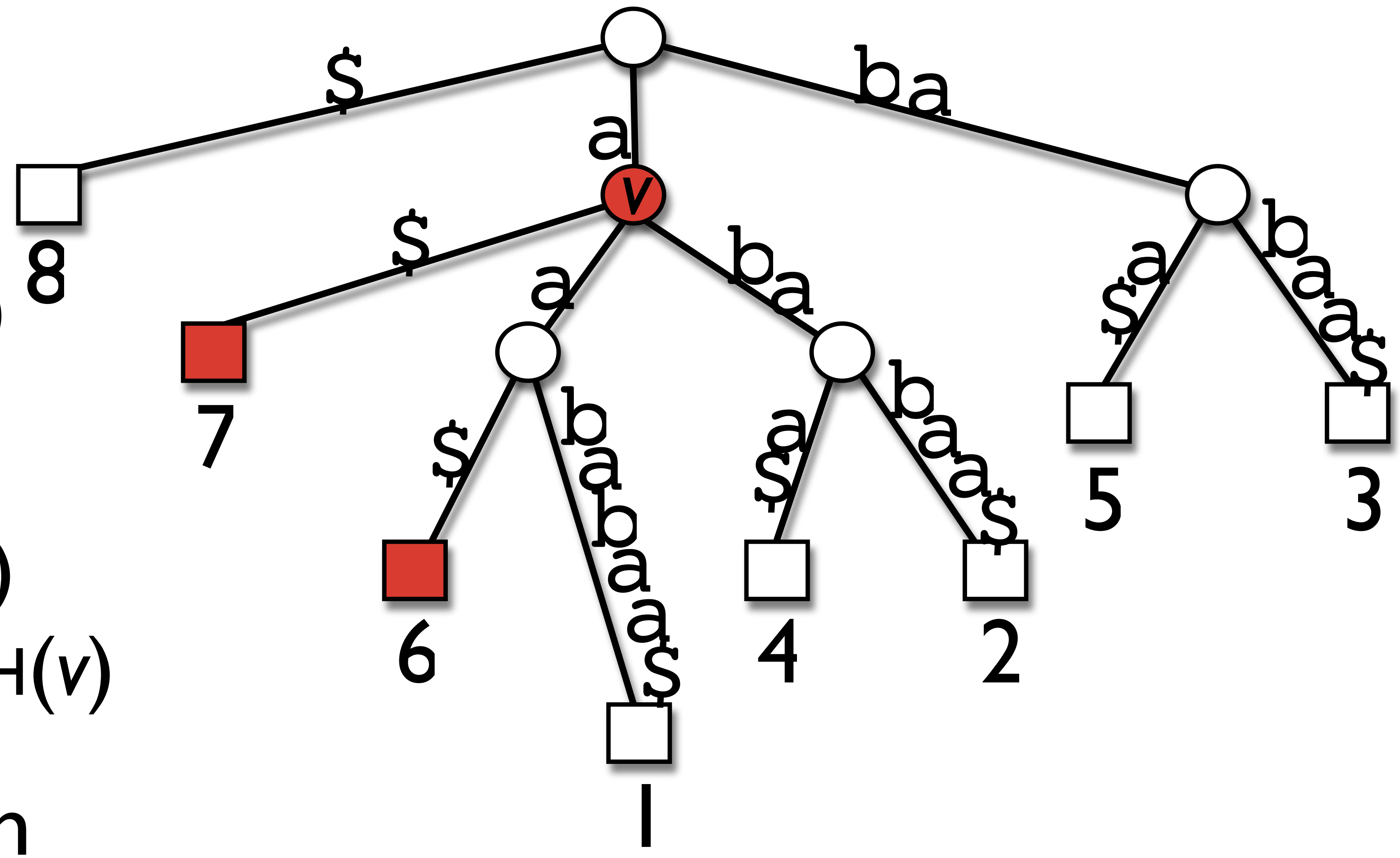
Represent Nodes by Intervals



A = 8,7,6,1,4,2,5,3

H = -1,0,1,2,1,3,0,2

Intervals $[v_l, v_r]$ in H



1. $H[i] \geq \text{SDEPTH}(v)$
   $\forall\ v_l < i \leq v_r$

2. $H[v_l] < \text{SDEPTH}(v)$
   $H[v_r + 1] < \text{SDEPTH}(v)$

3. $\exists\ v_l < i \leq v_r$ with
   $H[i] = \text{SDEPTH}(v)$

A = 8,7,6,1,4,2,5,3

v_l v_r

H = -1,0,1,2,1,3,0,2

Consequences

1. $H[i] \geq \text{SDEPTH}(v)$
   $\forall\ v_l < i \leq v_r$
2. $H[v_l] < \text{SDEPTH}(v)$
   $H[v_r+1] < \text{SDEPTH}(v)$
3. $\exists\ v_l < i \leq v_r$ with
   $H[i] = \text{SDEPTH}(v)$

(1) given v_l & v_r: compute i
by $i \leftarrow \text{RMQ}_H(v_l+1, v_r)$

A = 8,7,6,1,4,2,5,3
v_l v_r
H = -1,0,1,2,1,3,0,2
i
RMQ

Consequences

1. $H[i] \geq \text{SDEPTH}(v)$
   $\forall\ v_l < i \leq v_r$
2. $H[v_l] < \text{SDEPTH}(v)$
   $H[v_r+1] < \text{SDEPTH}(v)$
3. $\exists\ v_l < i \leq v_r$ with
   $H[i] = \text{SDEPTH}(v)$

(1) given v_l & v_r: compute i
by $i \leftarrow \text{RMQ}_H(v_l+1, v_r)$

(2) given i: compute
$v_l \leftarrow \text{PSV}_H(i)$
$v_r \leftarrow \text{NSV}_H(i)\text{-}1$

$A = 8,7,6,1,4,2,5,3$
$v_l \quad v_r$
$H = \text{-}1,0,1,2,1,3,0,2$
i
PSV NSV

3 Components of CST

A: compressed (sampled) suffix array

H: compressed LCP-array

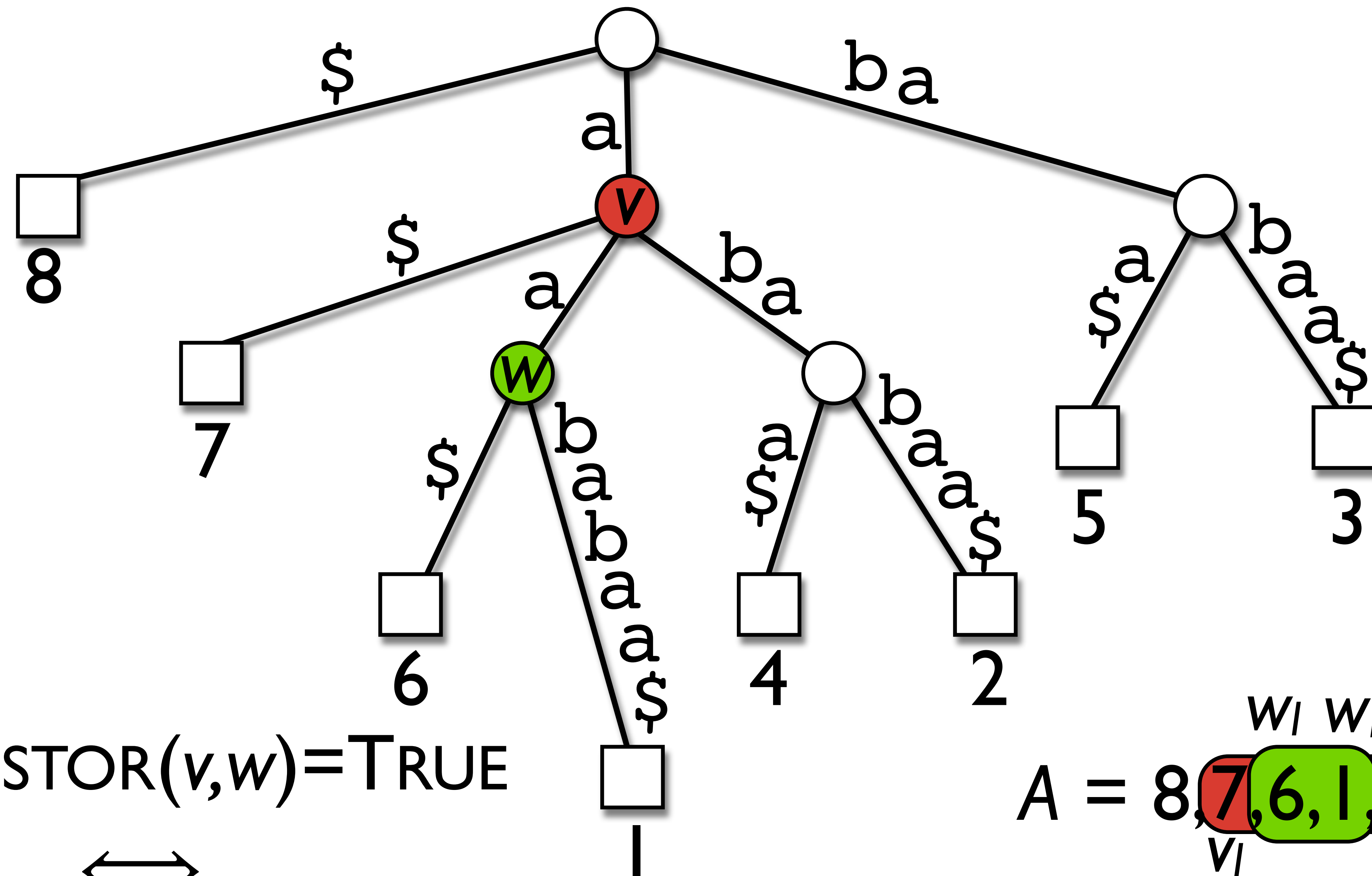
next week

compressed RMQ & PSV/NSV on LCP

CST

node v represented by interval $[v_l, v_r]$ in H (or A)

IsAncestor(v,w)



IsAncestor(v,w)=True

$\Longleftrightarrow$

$v_l \leq w_r \leq v_r$

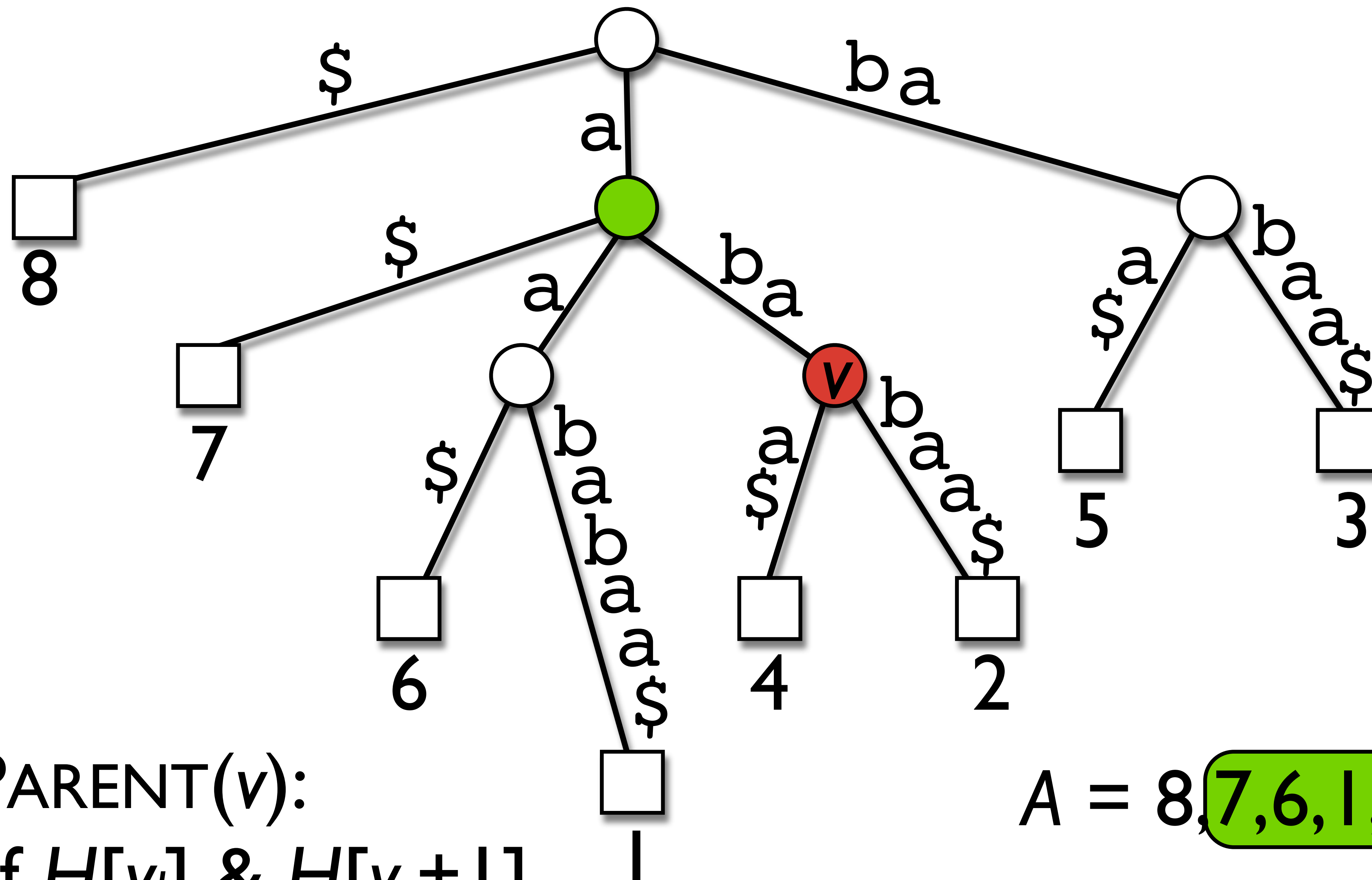
$w_l \; w_r$

$A = 8,7,6,1,4,2,5,3$

$v_l \; v_r$

$H = -1,0,1,2,1,3,0,2$

PARENT(v)



PARENT(v):
larger of $H[v_l]$ & $H[v_r+1]$
is SDEPTH of parent!

A = 8,7,6,1,4,2,5,3

v_l v_r

H = -1,0,1,2,1,3,0,2

PSV NSV

ST Operations

Operation	Description
ROOT()	return root
COUNT(v)	count leaves below v
ISANCESTOR(v,w)	**true if v is an ancestor of w** ✓
ISLEAF (v)	true if v is a leaf
LEAFLABEL(v)	suffix number represented by leaf v
SDEPTH(v)	string depth of v
PARENT(v)	**parent node of v** ✓
FIRSTCHILD(v)	first (alphabetically smallest) child of v
NEXTSIBLING(v)	next sibling of v
EDGELABEL(v,i)	i'th letter on the edge leading to v
LCA(v,w)	lowest common ancestor of v and w

Summary

- Represent ST nodes by **intervals**
- Simulate operations by **RMQ**s and **PSV**s/**NSV**s on LCP-array

$\Longrightarrow$ suffix and LCP-array **replace** suffix tree

- for a completely compressed ST:
  - compress LCP-array
  - small-space solutions for RMQ/PSV/NSV