Compressed Suffix Trees

Johannes Fischer

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

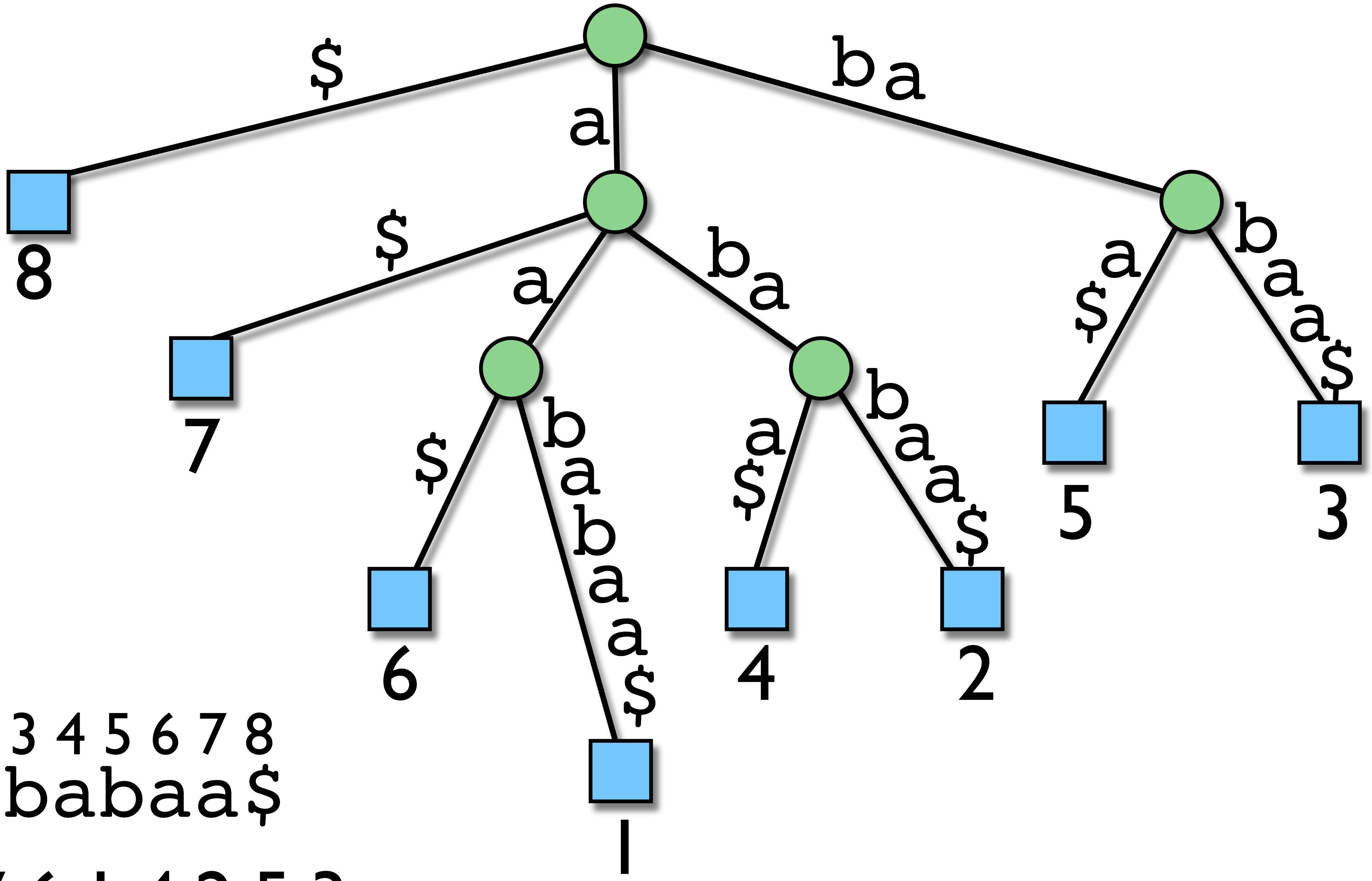
  ▸ can be improved to O(k $\lg^{\varepsilon} n$) for $\varepsilon<1$

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



$$\begin{array}{l} \quad\;\; 1\,2\,3\,4\,5\,6\,7\,8 \\ T = \texttt{aababaa\$} \end{array}$$

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

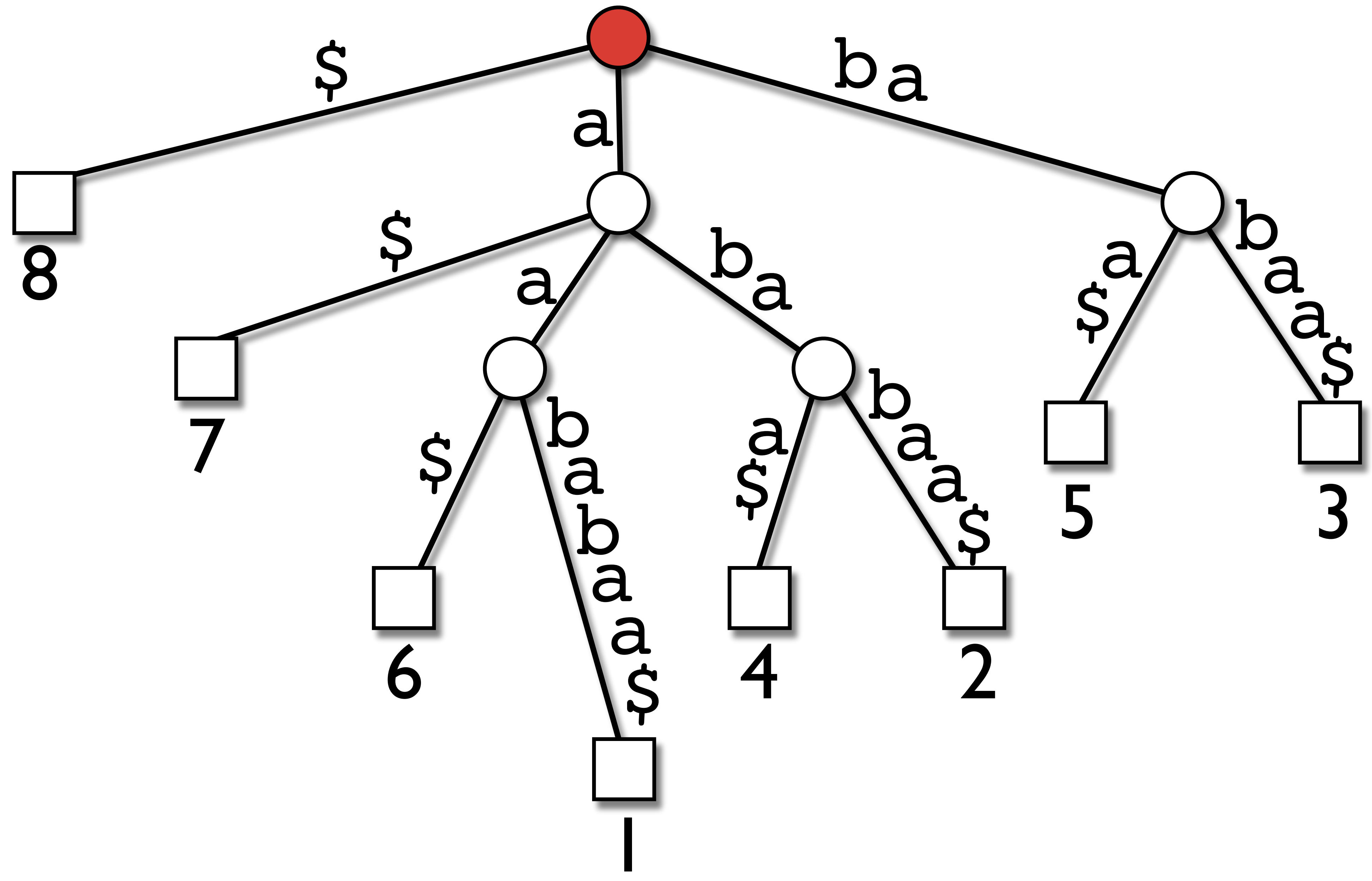
Some real numbers

- **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):
  - ≈**3** times text size
- **goal**: drop suffix tree and simulate operations using suffix- and LCP array

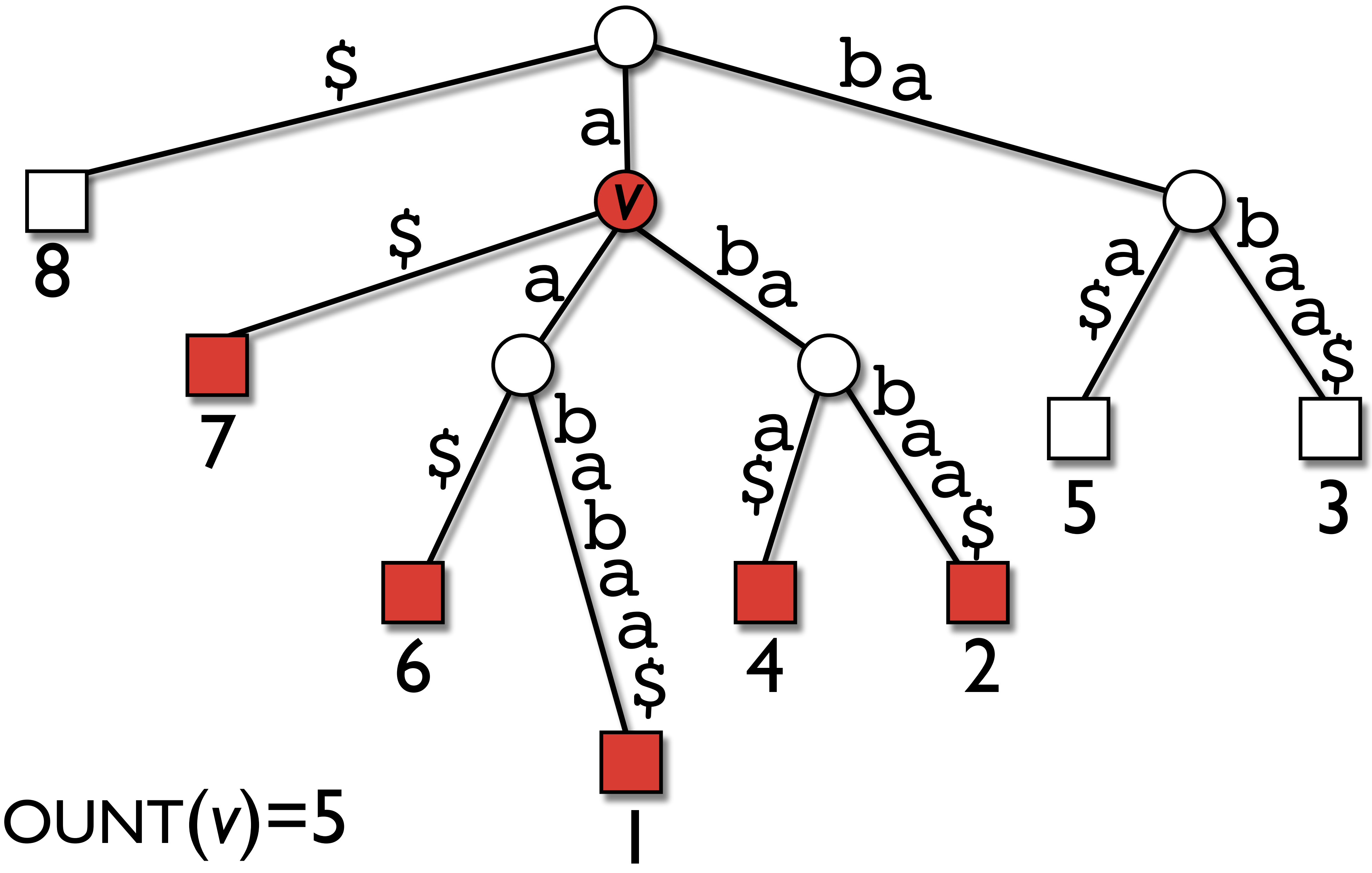
ST Operations

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()

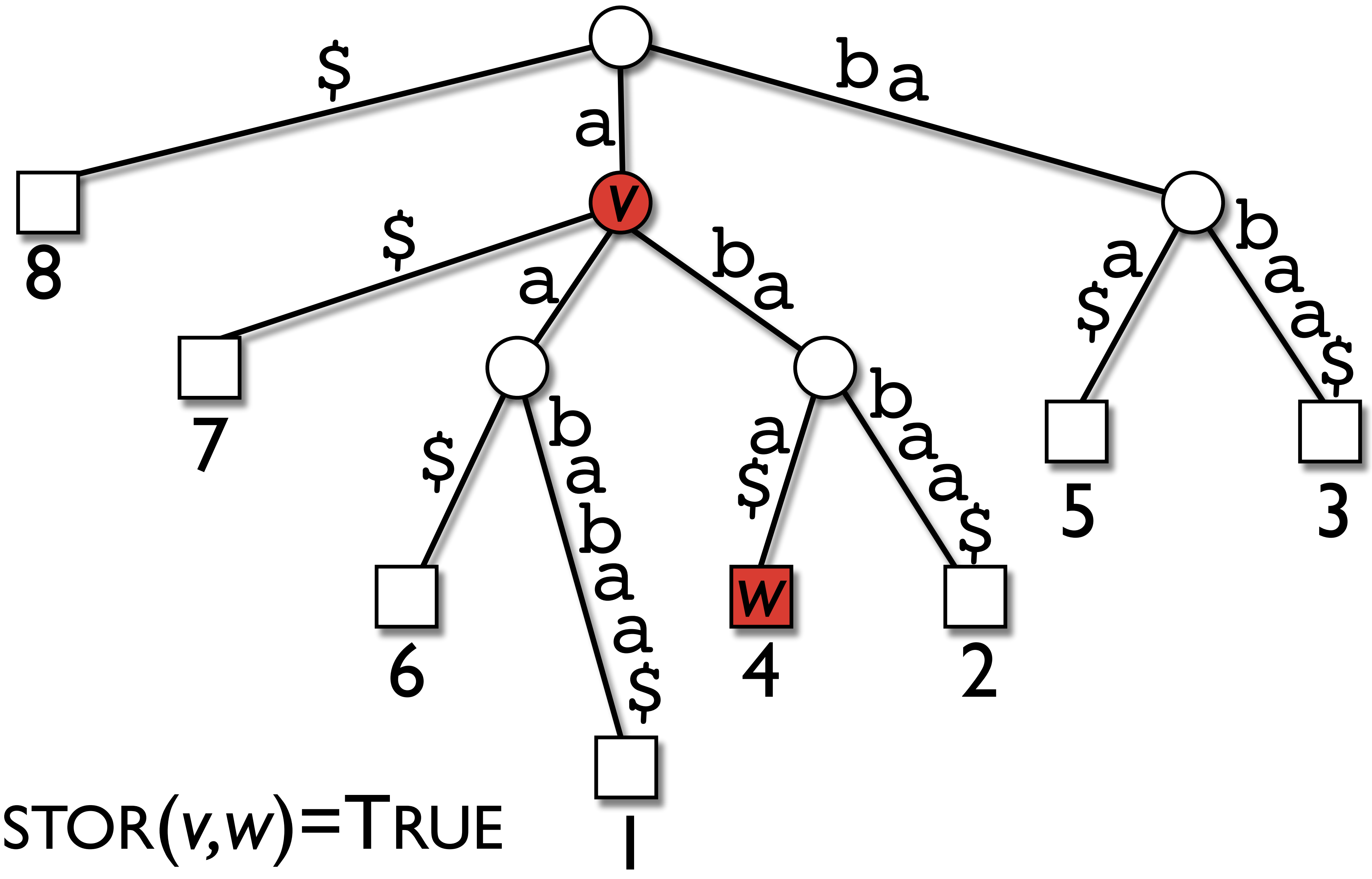


COUNT(v)



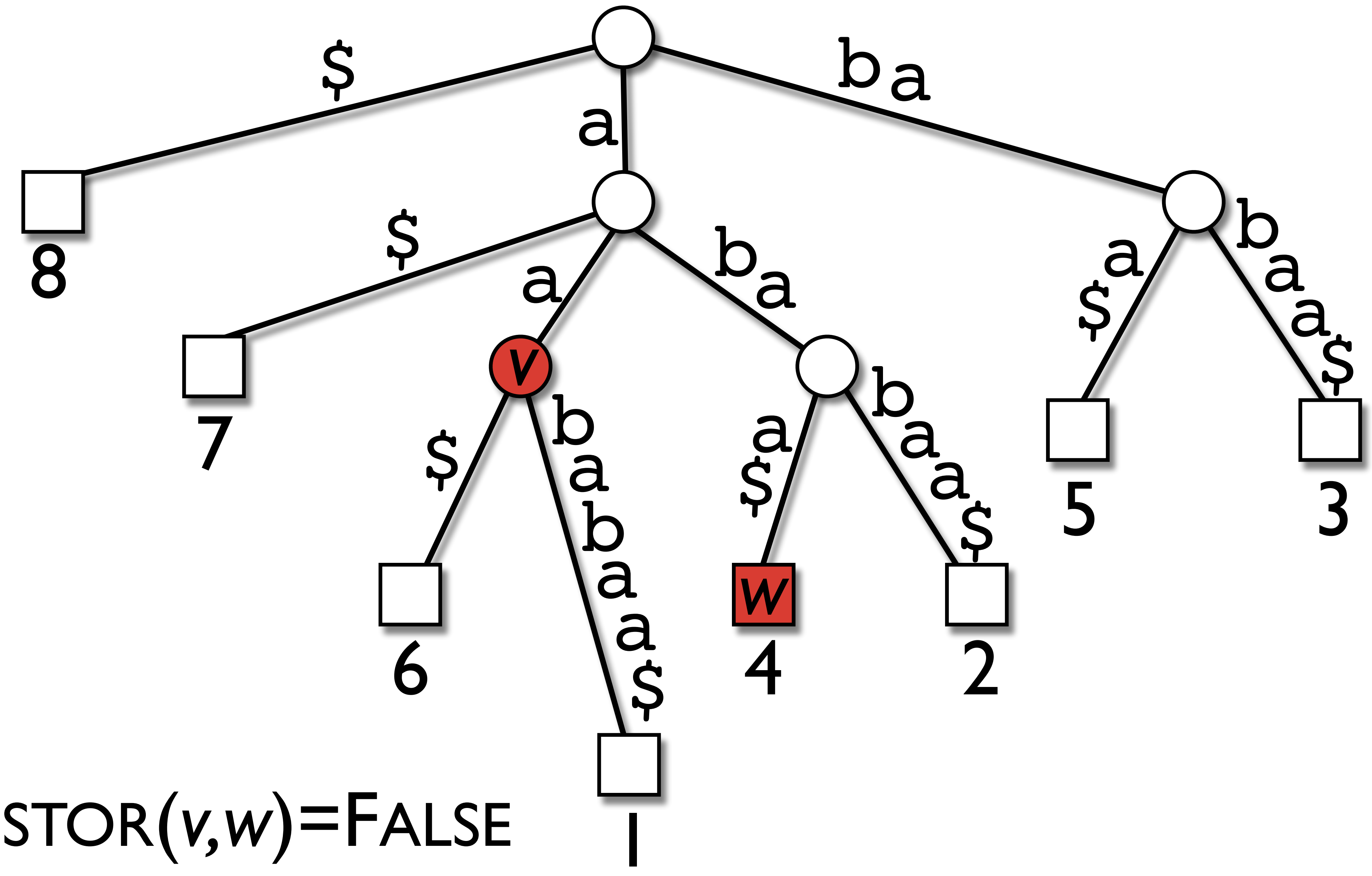
COUNT(v)=5

ISANCESTOR(*v*,*w*)



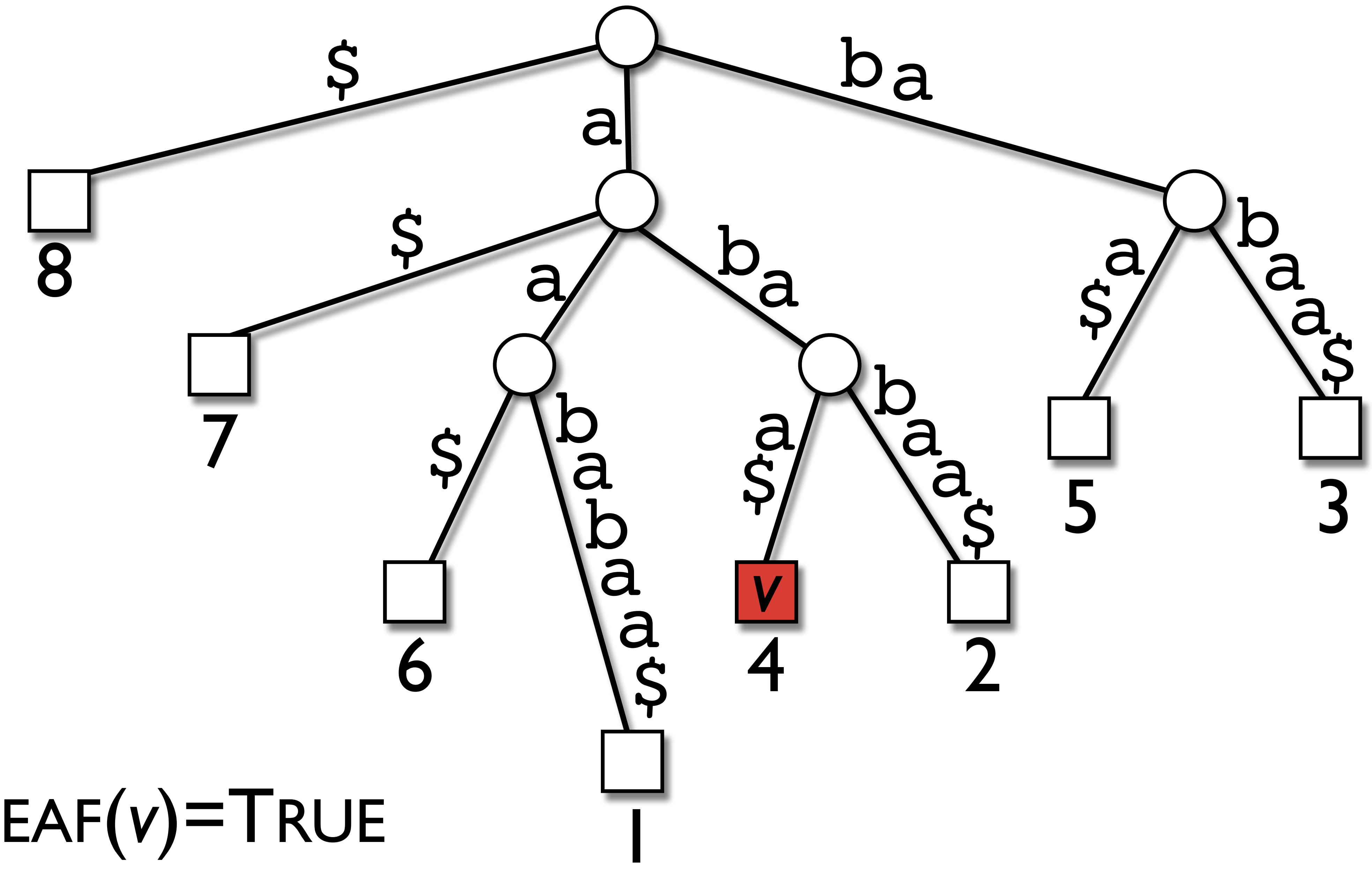
ISANCESTOR(*v*,*w*)=TRUE

IsAncestor(v,w)



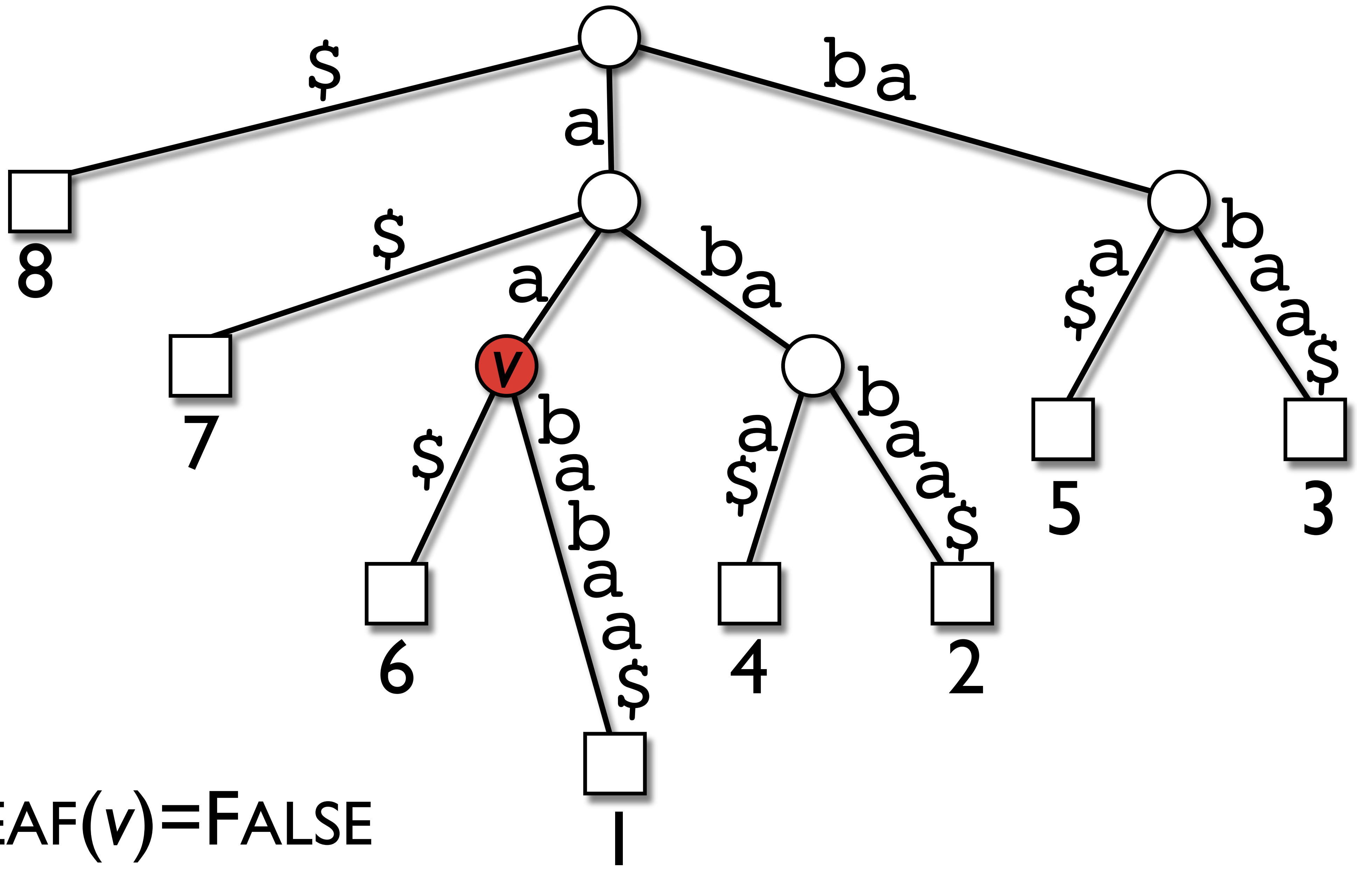
IsAncestor(v,w)=False

IsLeaf(*v*)



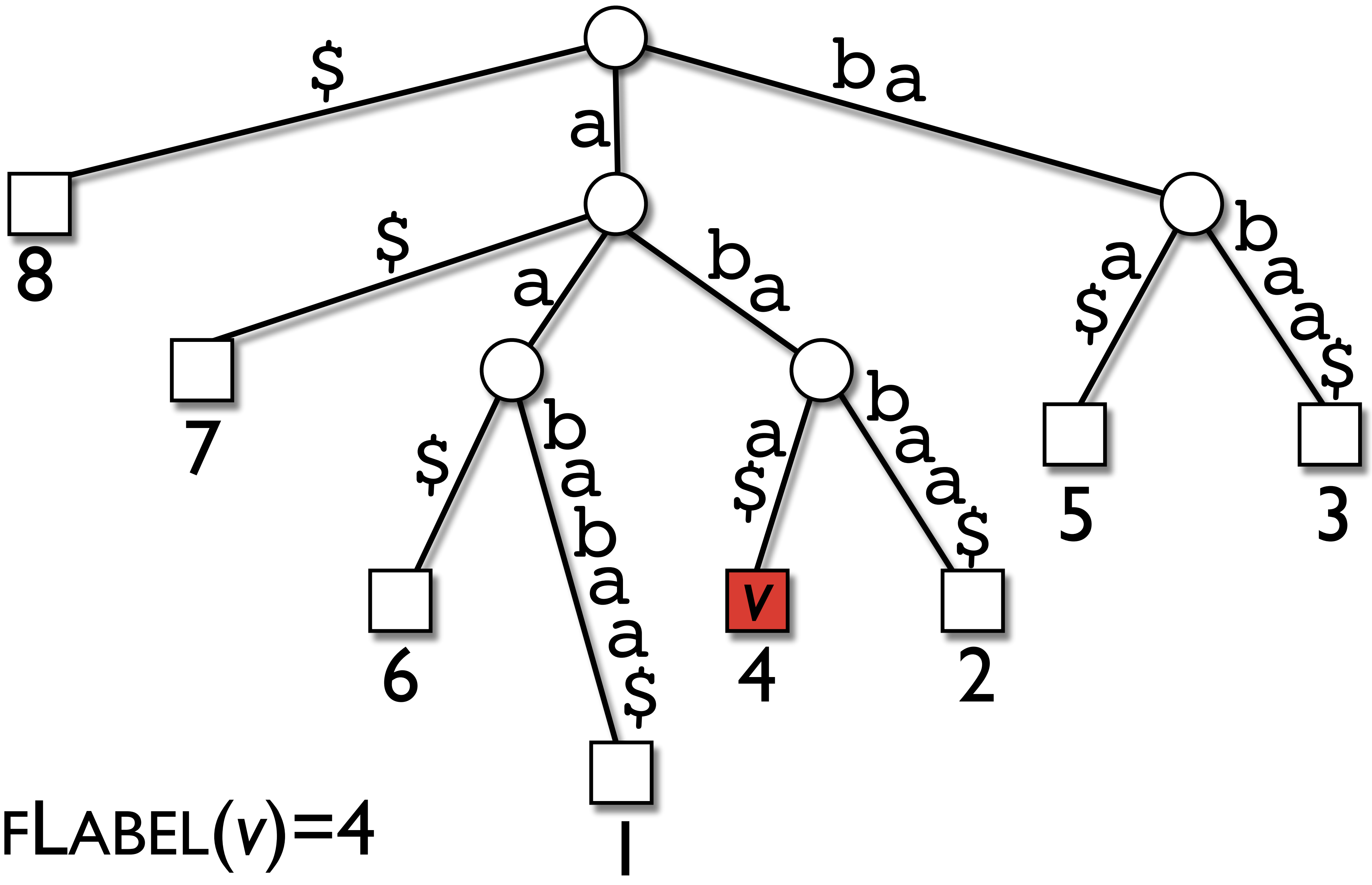
IsLeaf(*v*)=True

ISLEAF(v)



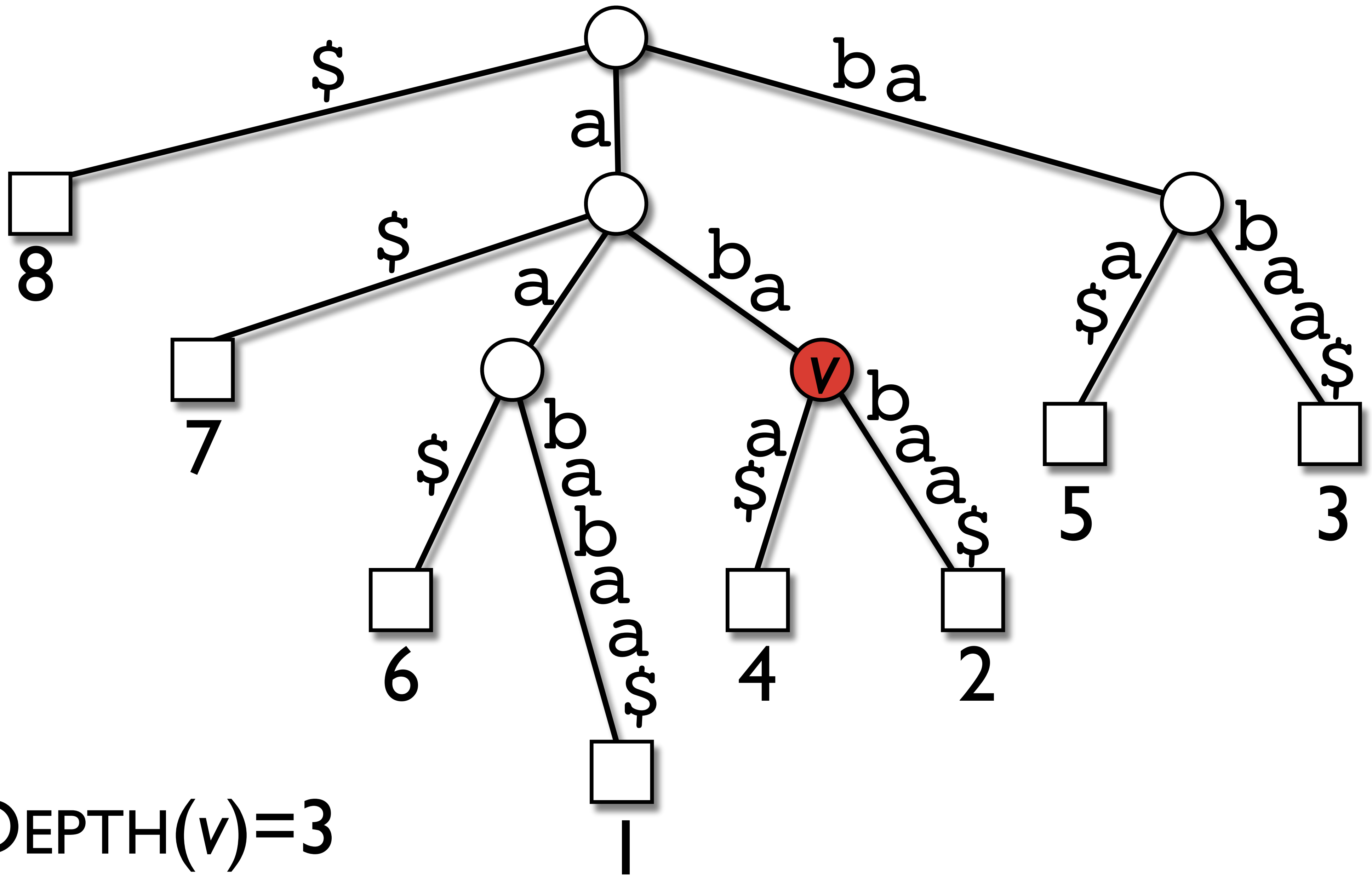
ISLEAF(v)=FALSE

LEAFLABEL(*v*)



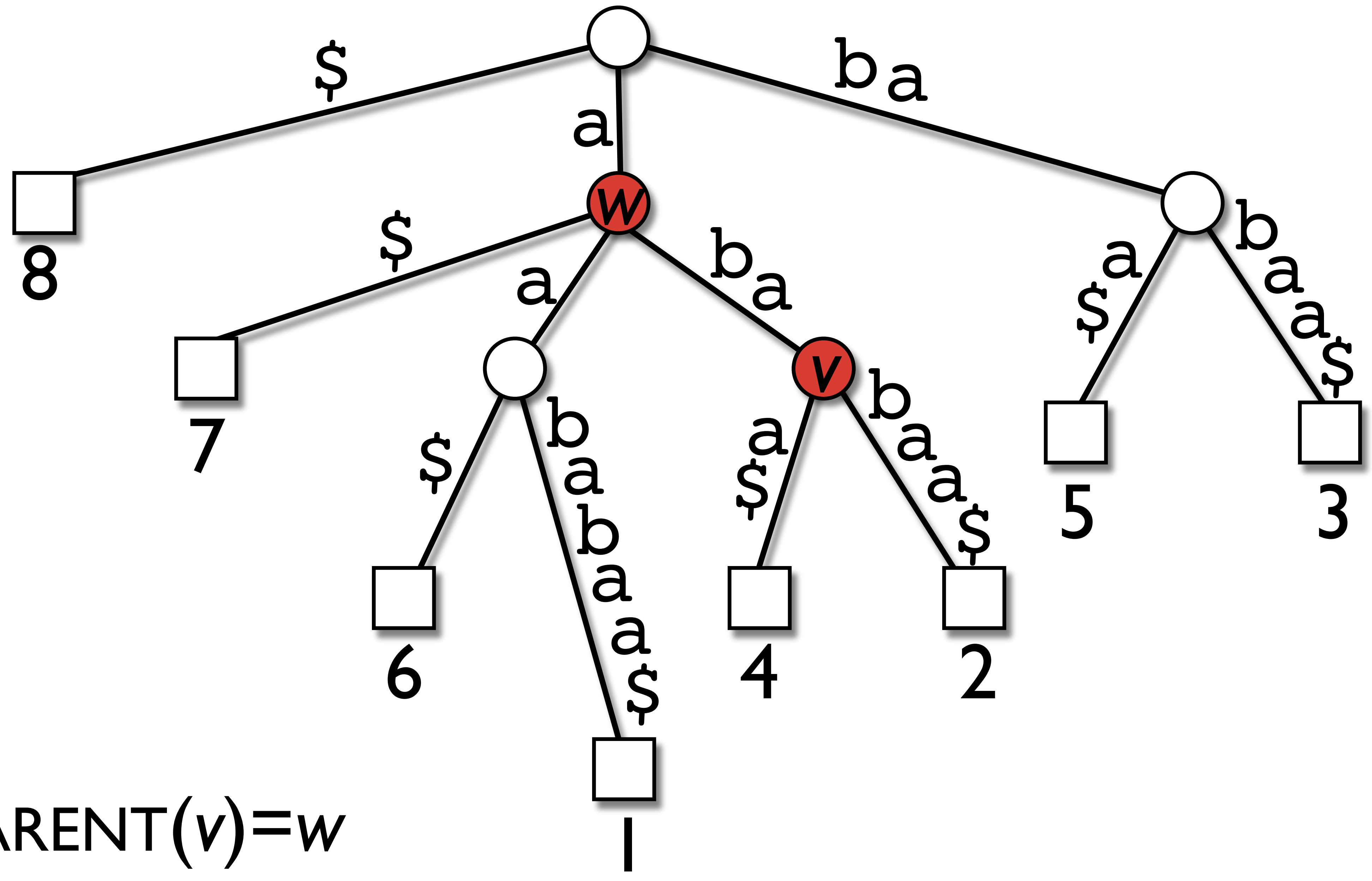
LEAFLABEL(*v*)=4

SDEPTH(v)



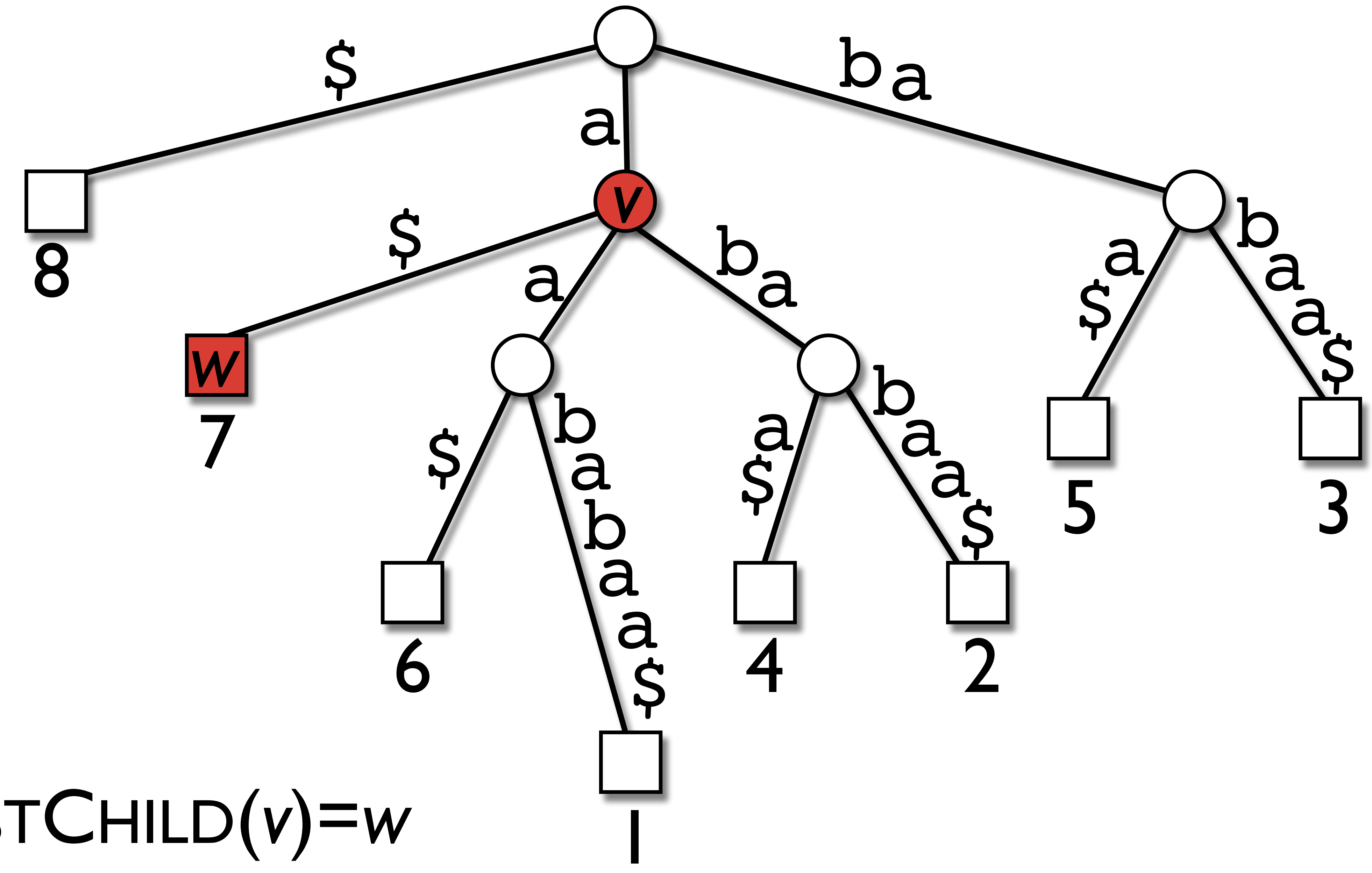
SDEPTH(v)=3

PARENT(*v*)



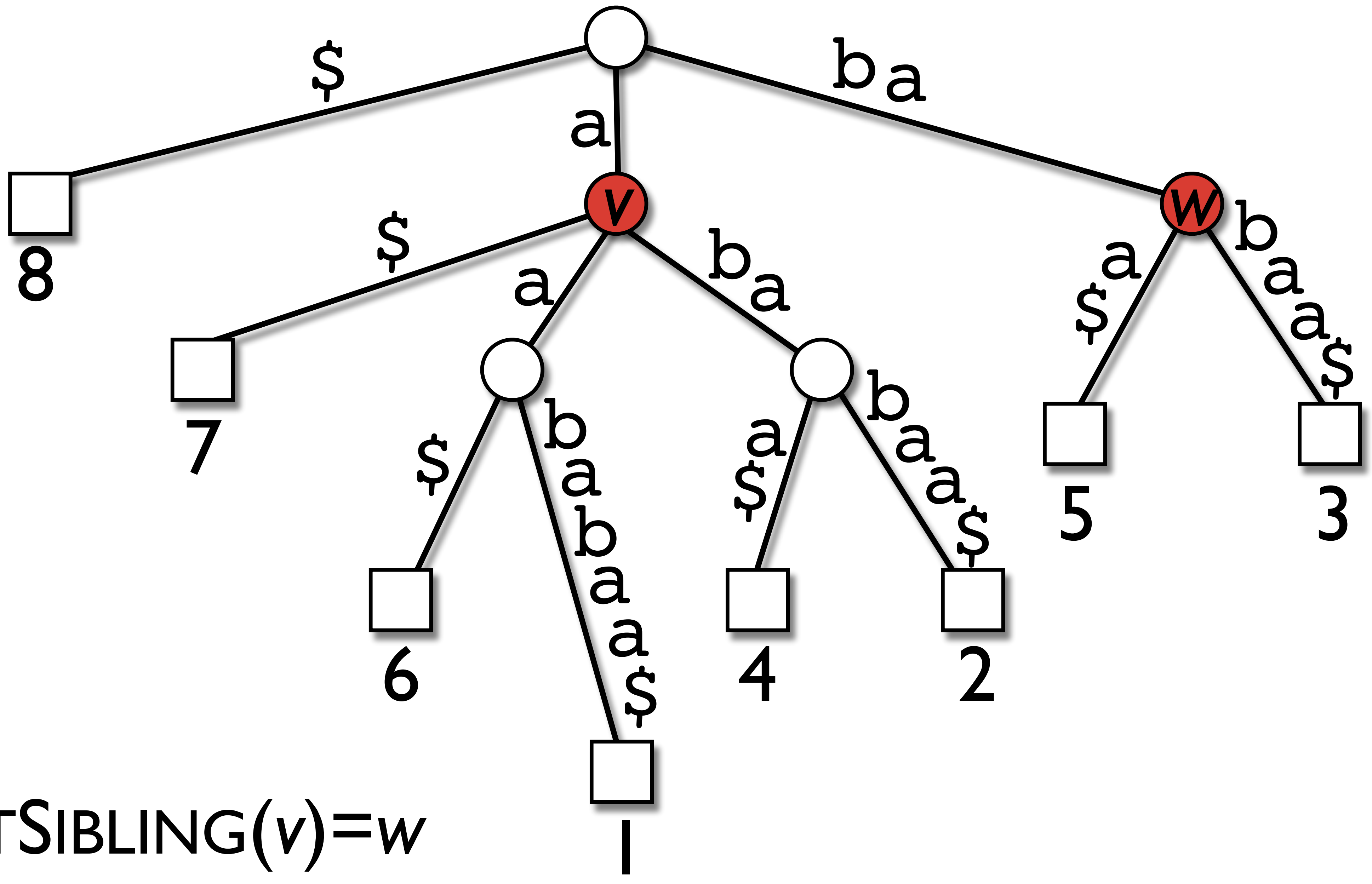
PARENT(*v*)=*w*

FIRSTCHILD(v)



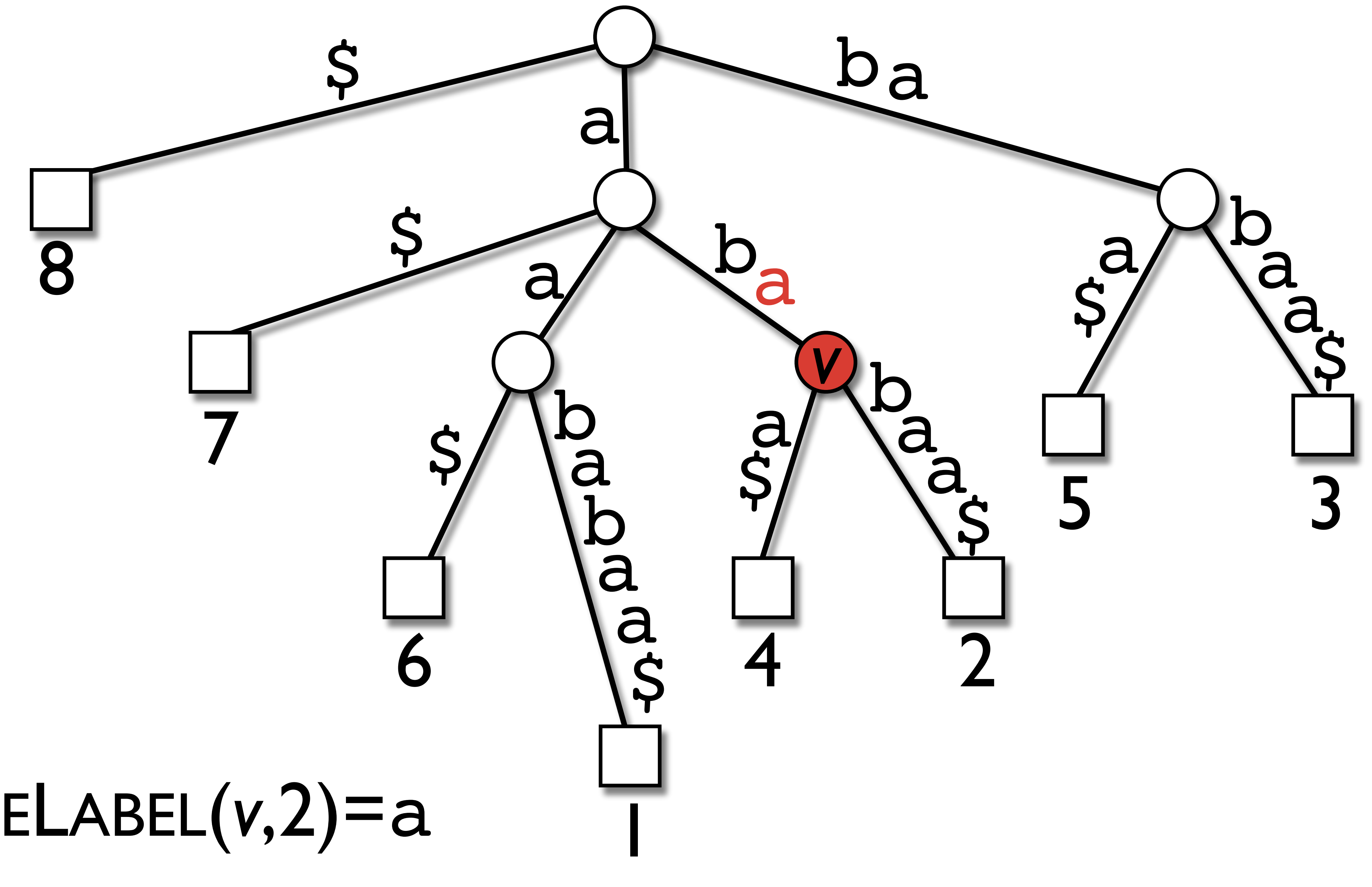
FIRSTCHILD(v)=w

NEXTSIBLING(v)



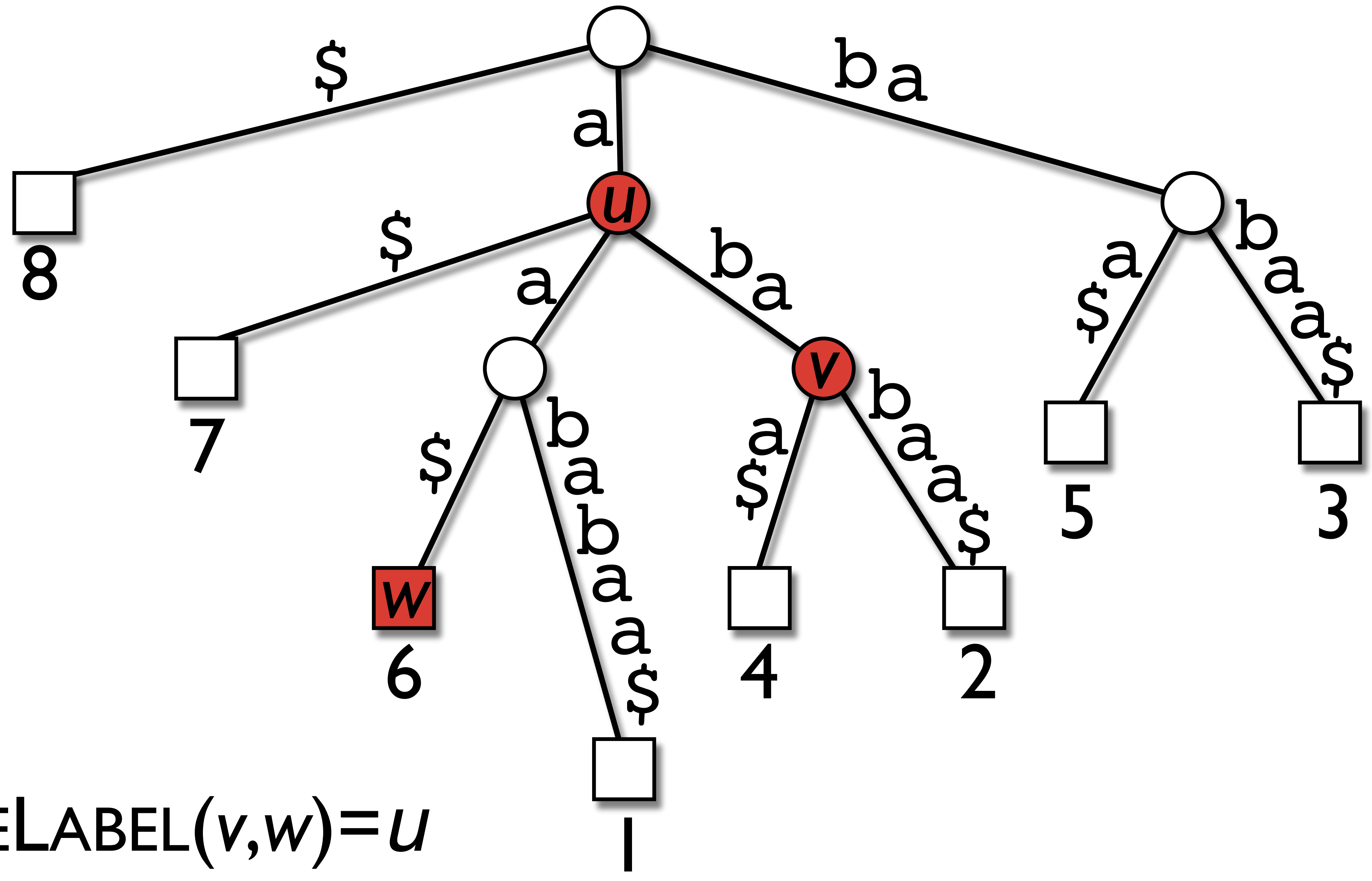
NEXTSIBLING(v)=w

EDGELABEL(v,i)



EDGELABEL(v,2)=a

LCA(*v*,*w*)

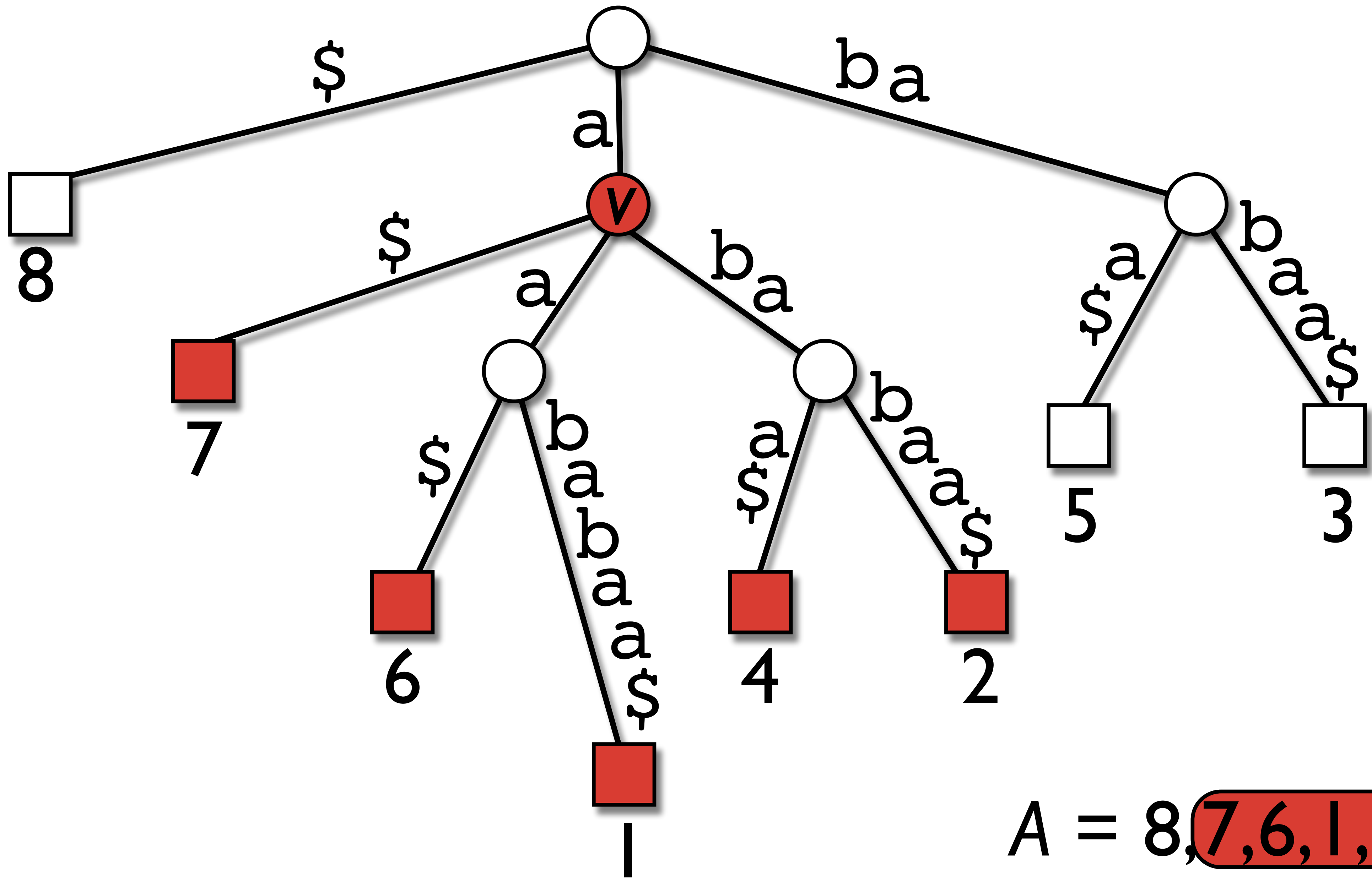


EDGELABEL(*v*,*w*)=*u*

Goal

drop suffix tree and **simulate** operations using suffix- and LCP array

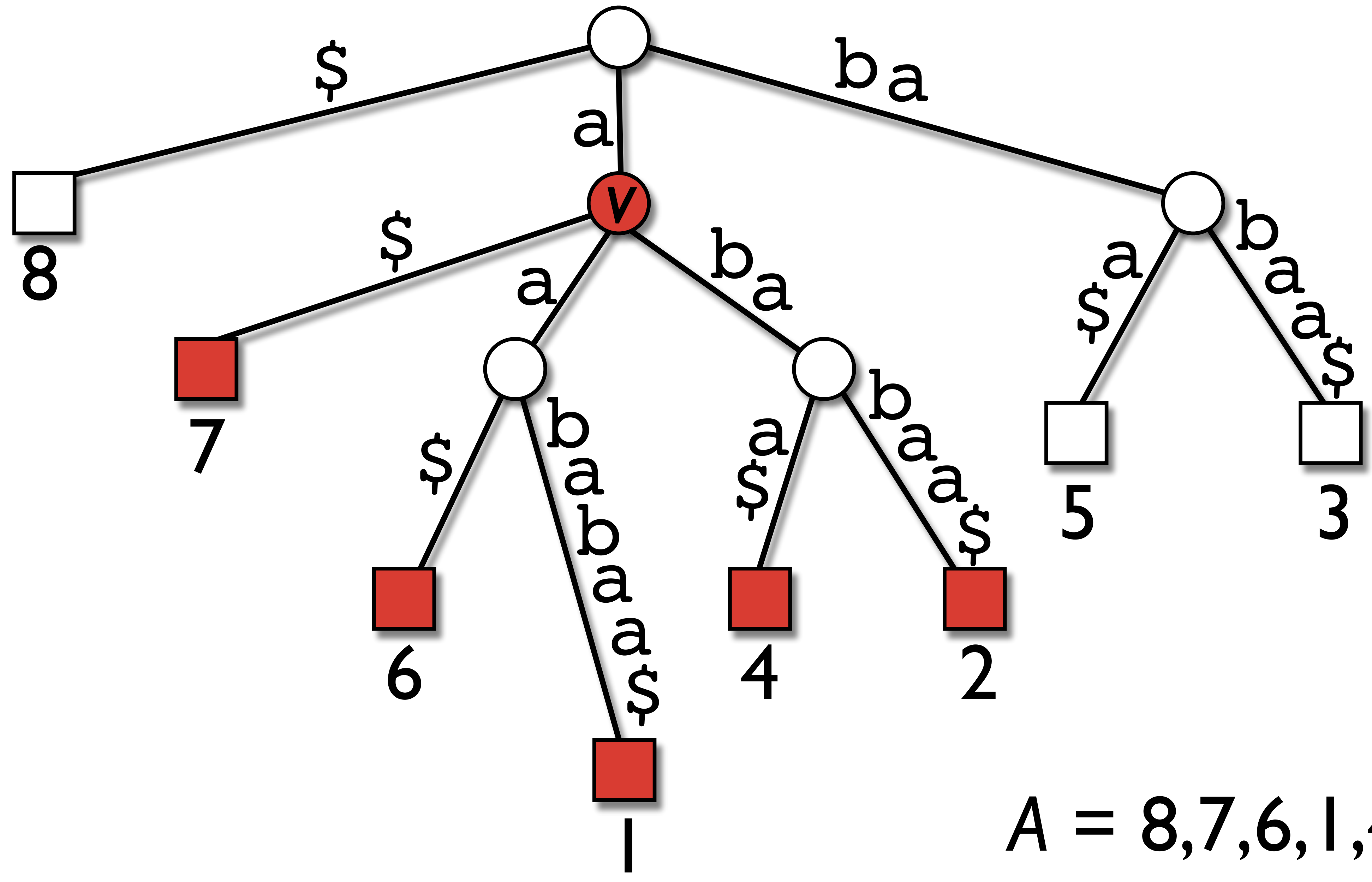
Represent Nodes by Intervals



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

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

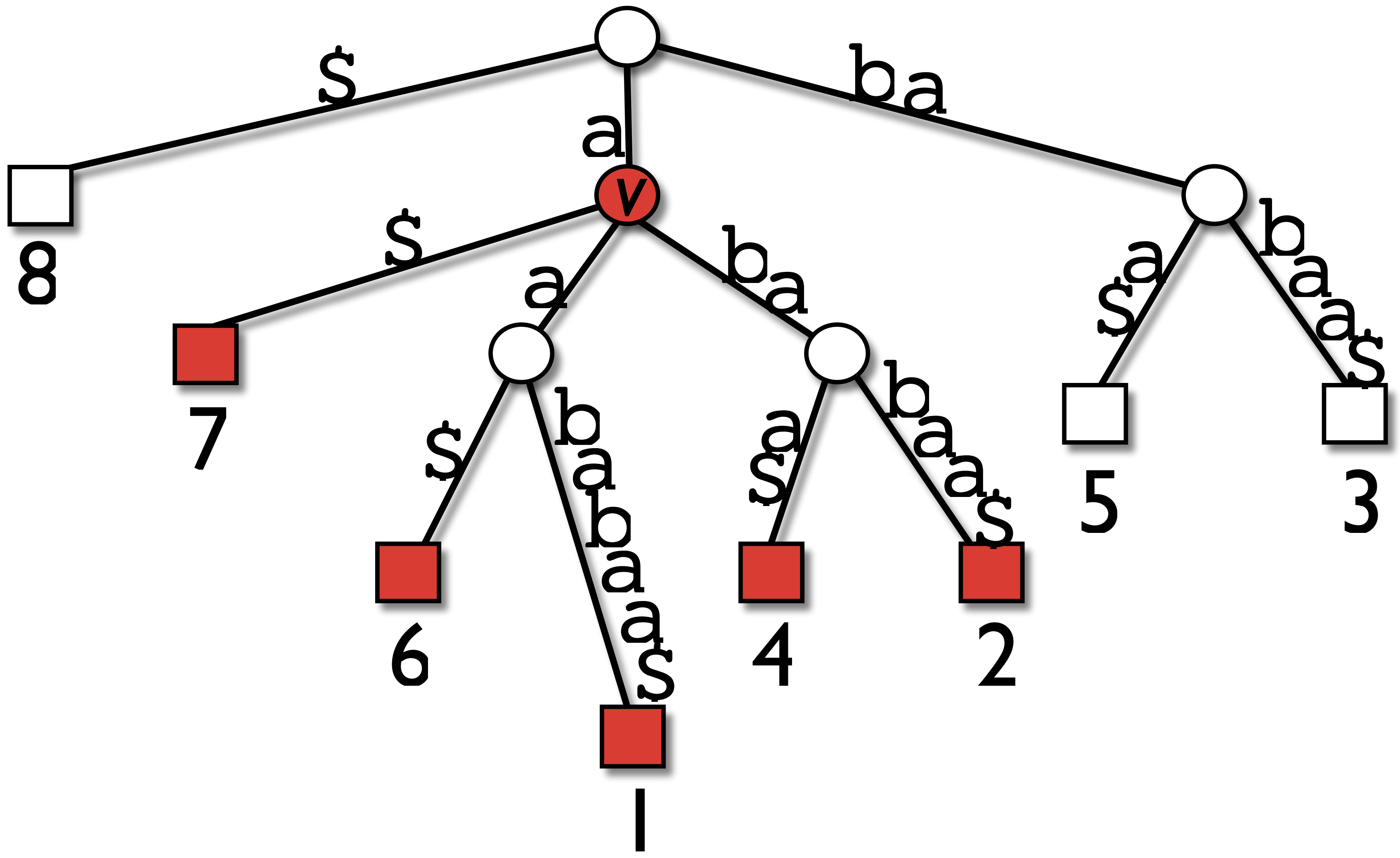
Represent Nodes by Intervals



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

H = -1,$\boxed{0,1,2,1,3}$,0,2

Intervals $[v_l, v_r]$ in H

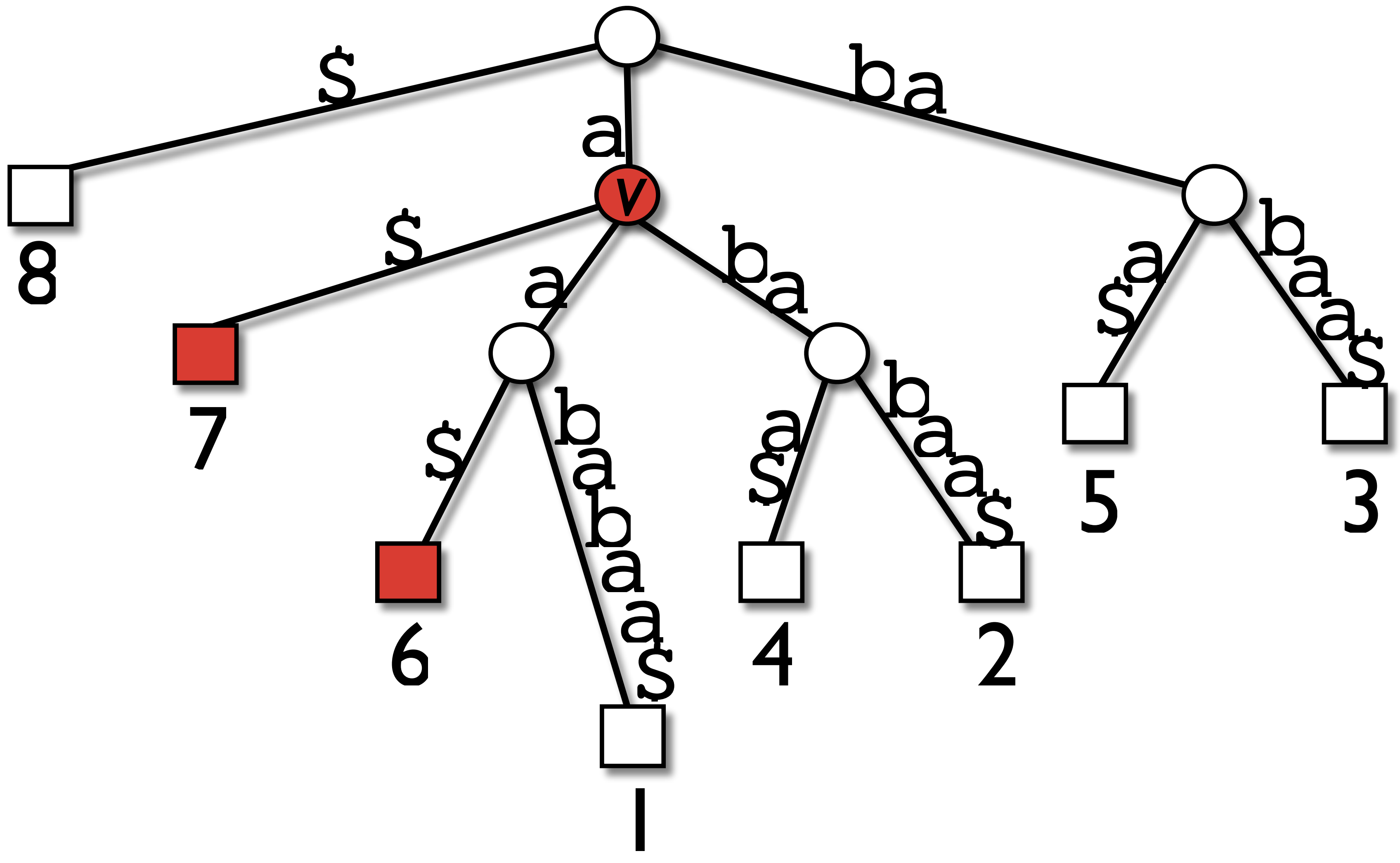


1. $H[i] \geq \text{SDEPTH}(v)$
   $\forall\ v_l < i \leq v_r$

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

$$H = -1,\underset{v_l}{0},1,2,1,\underset{v_r}{3},0,2$$

Intervals $[v_l, v_r]$ in H



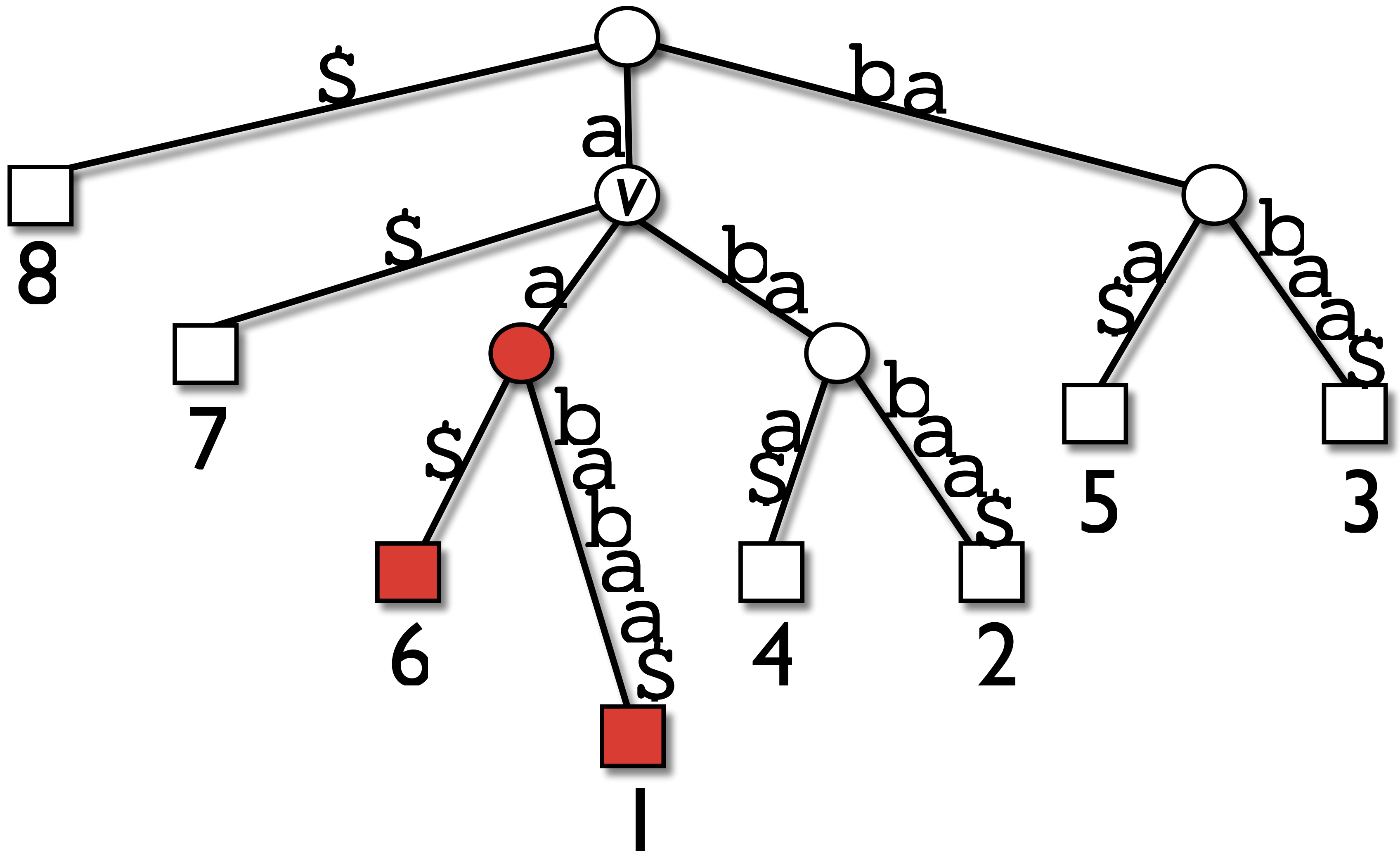
1. $H[i] \geq \text{SDEPTH}(v)$
   $\forall\ v_l < i \leq v_r$

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

v_l v_r

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

Intervals $[v_l, v_r]$ in H



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

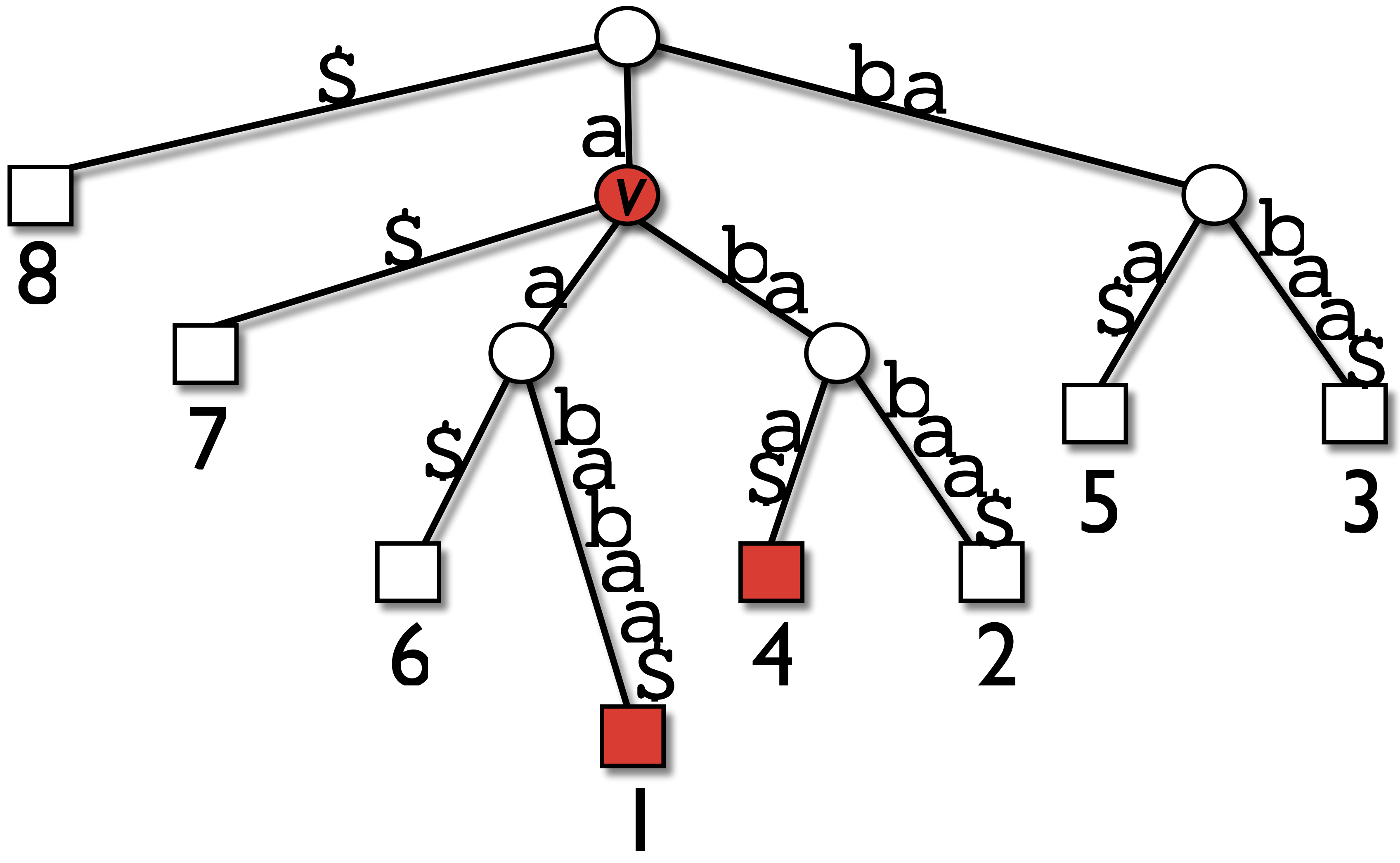
$A = 8, 7, 6, 1, 4, 2, 5, 3$

$v_l \qquad v_r$

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

Intervals $[v_l, v_r]$ in H

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

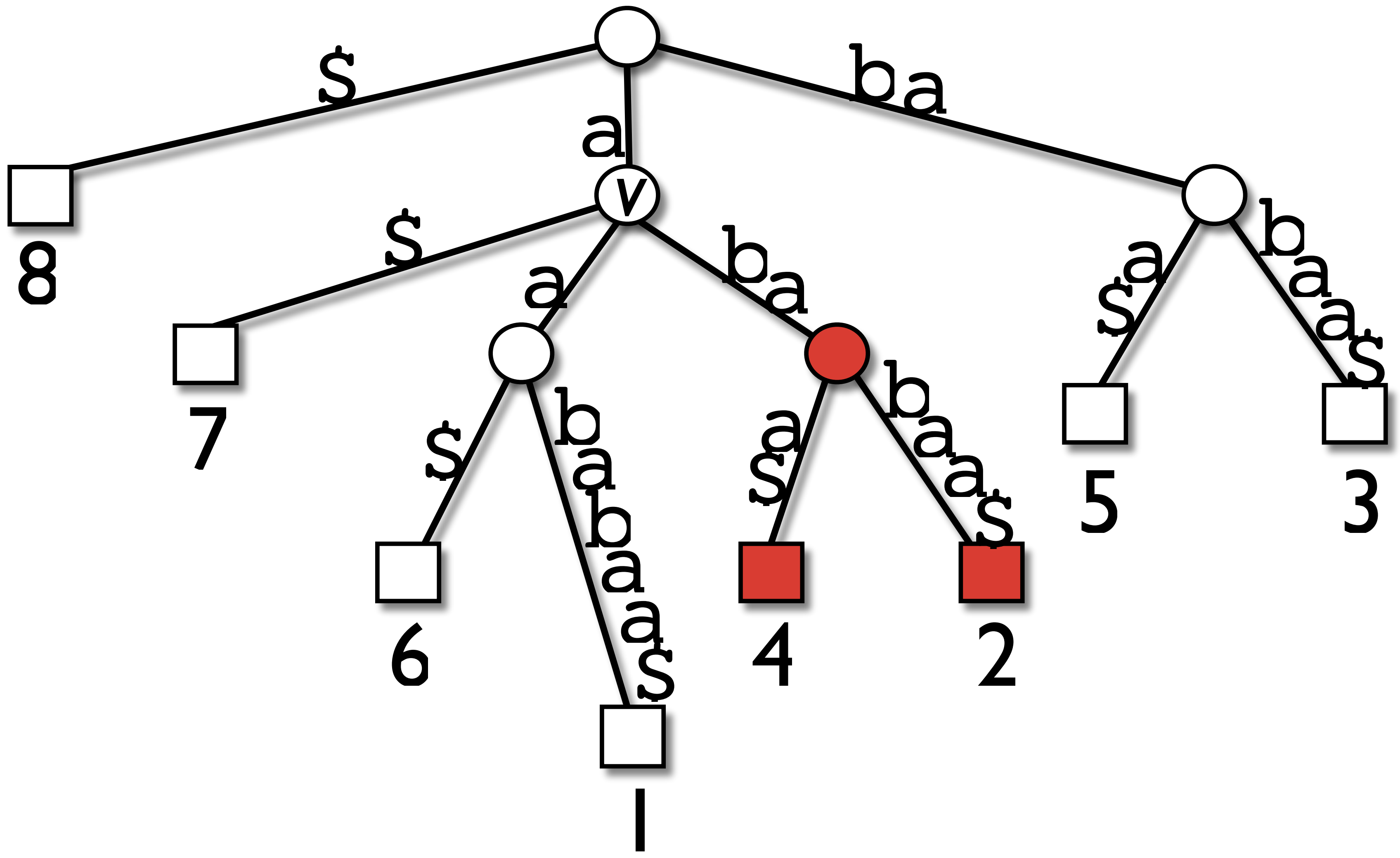


$A = 8,\underbrace{7,6,1,4,2}_{v_l \quad\quad v_r},5,3$

$H = \text{-}1,0,1,2,\boxed{1},3,0,2$

Intervals $[v_l, v_r]$ in H

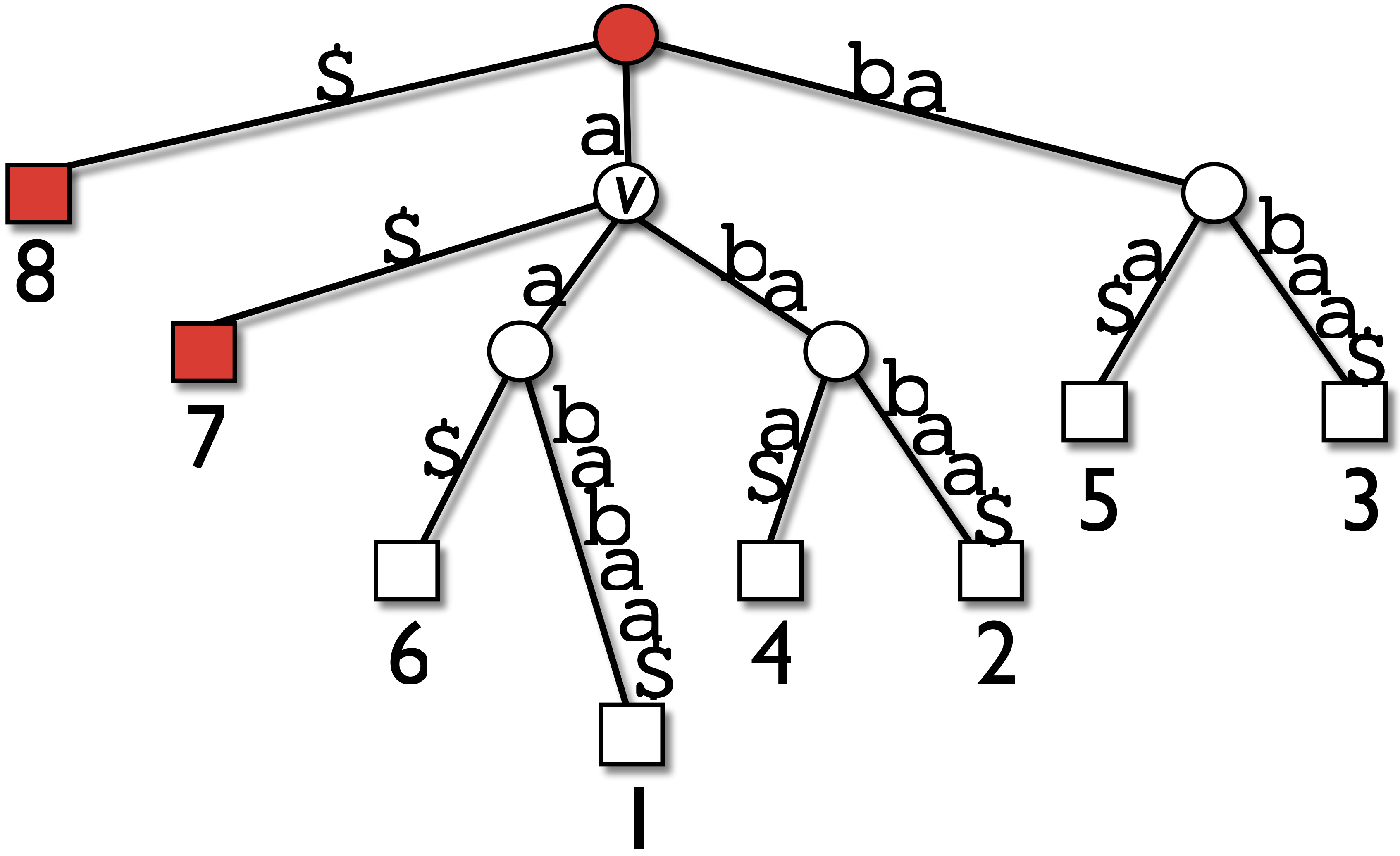
1. $H[i] \geq \text{SDEPTH}(v)$
   $\forall\ v_l < i \leq v_r$



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

Intervals $[v_l, v_r]$ in H

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

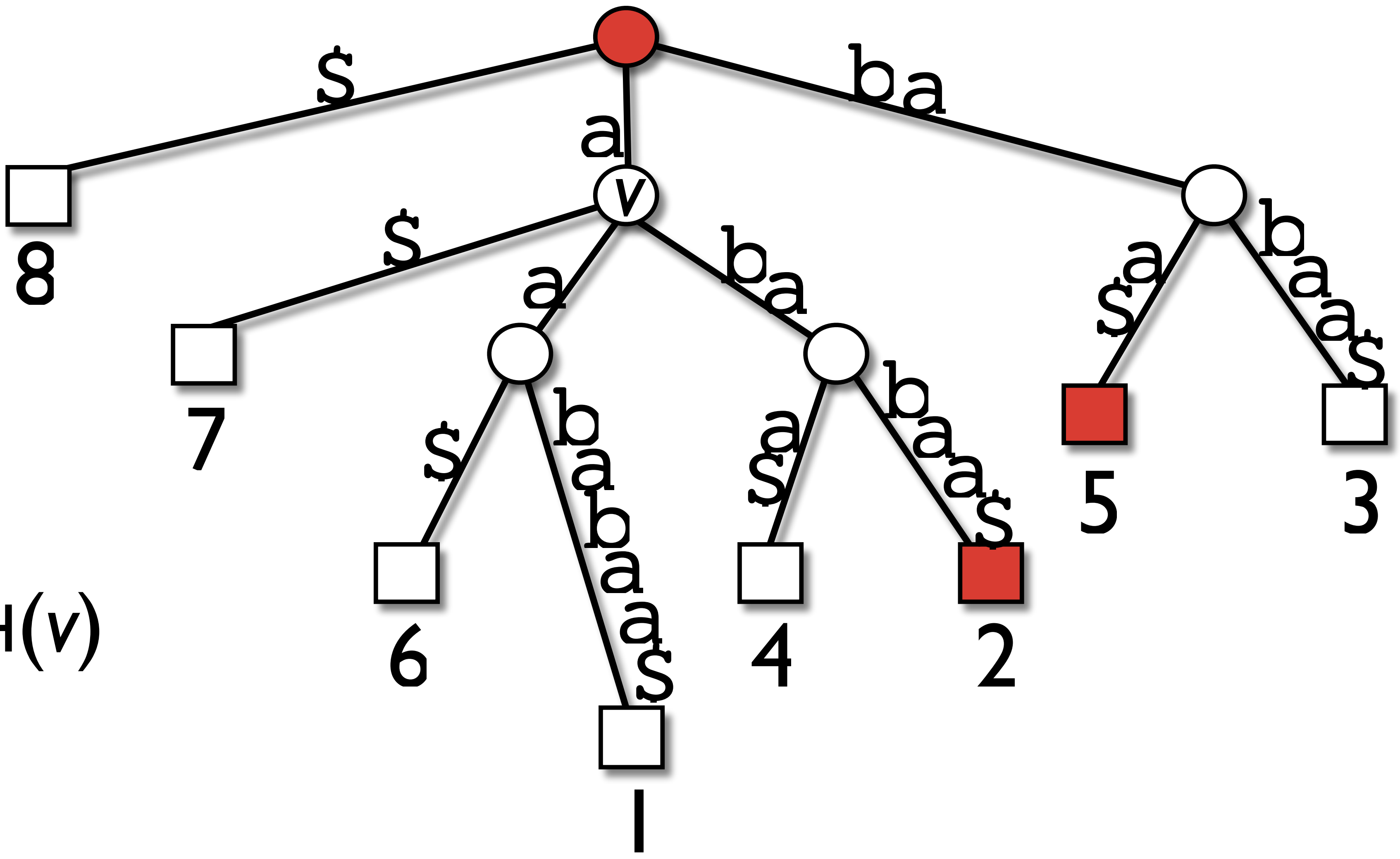


$A = 8, \boxed{7,6,1,4,2}, 5, 3$

$\quad\quad\quad v_l \quad\quad\quad v_r$

$H = -1, \boxed{0}, 1, 2, 1, 3, 0, 2$

Intervals $[v_l, v_r]$ in H

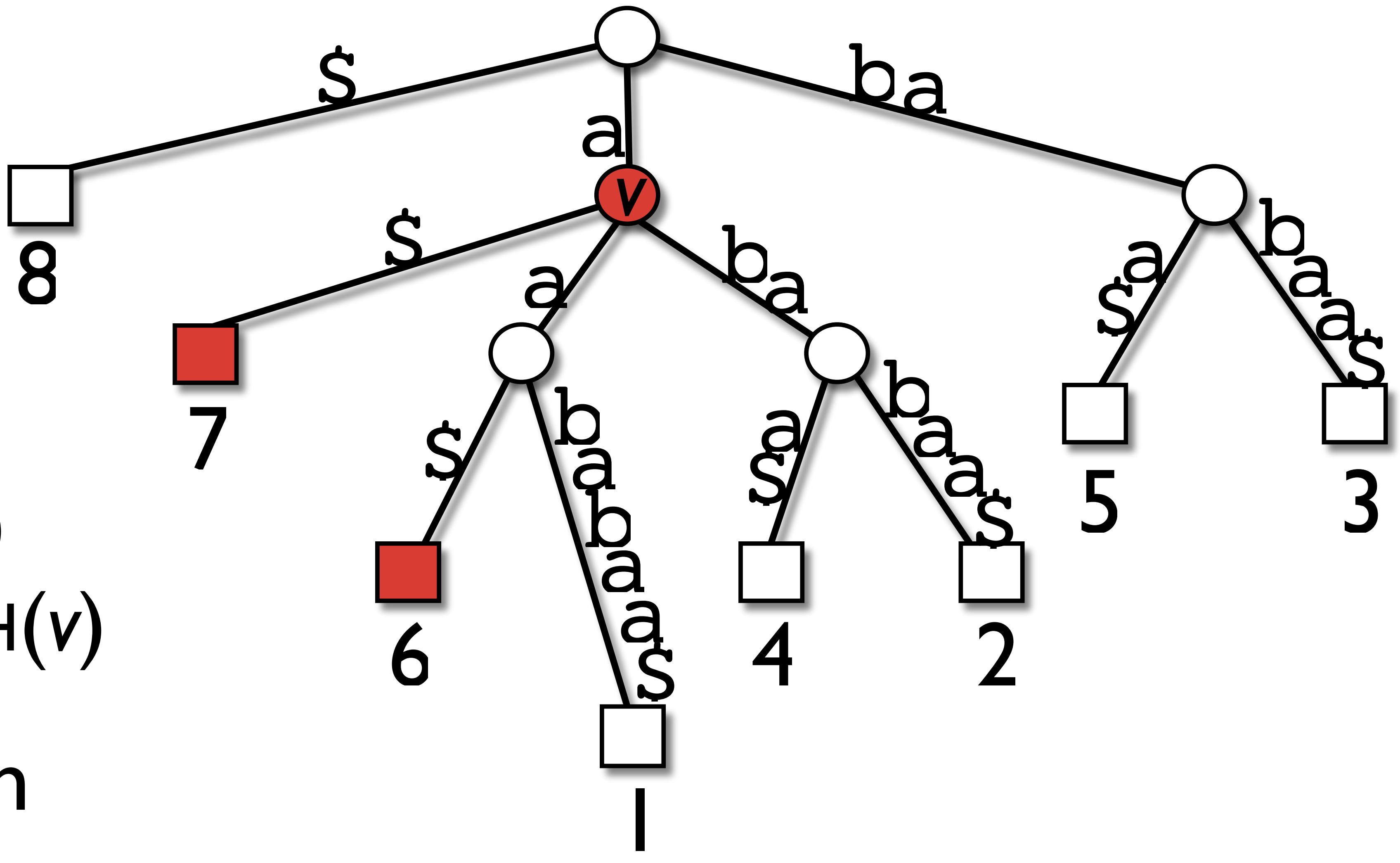


1. $H[i] \geq \text{SD}\small{\text{EPTH}}(v)$
   $\forall\ v_l < i \leq v_r$

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

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

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

$v_l \qquad\qquad v_r$

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

Consequences

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

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

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

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

$$A = 8,\boxed{\underset{v_l}{7},6,1,4,\underset{v_r}{2}},5,3$$

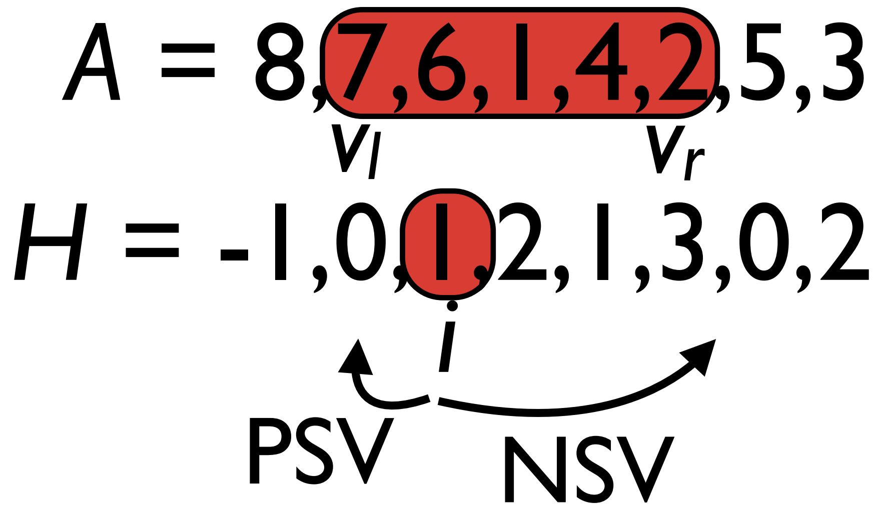
$$H = -1,0,\underbrace{\underset{i}{\boxed{1}},2,1,3}_{\text{RMQ}},0,2$$

Consequences

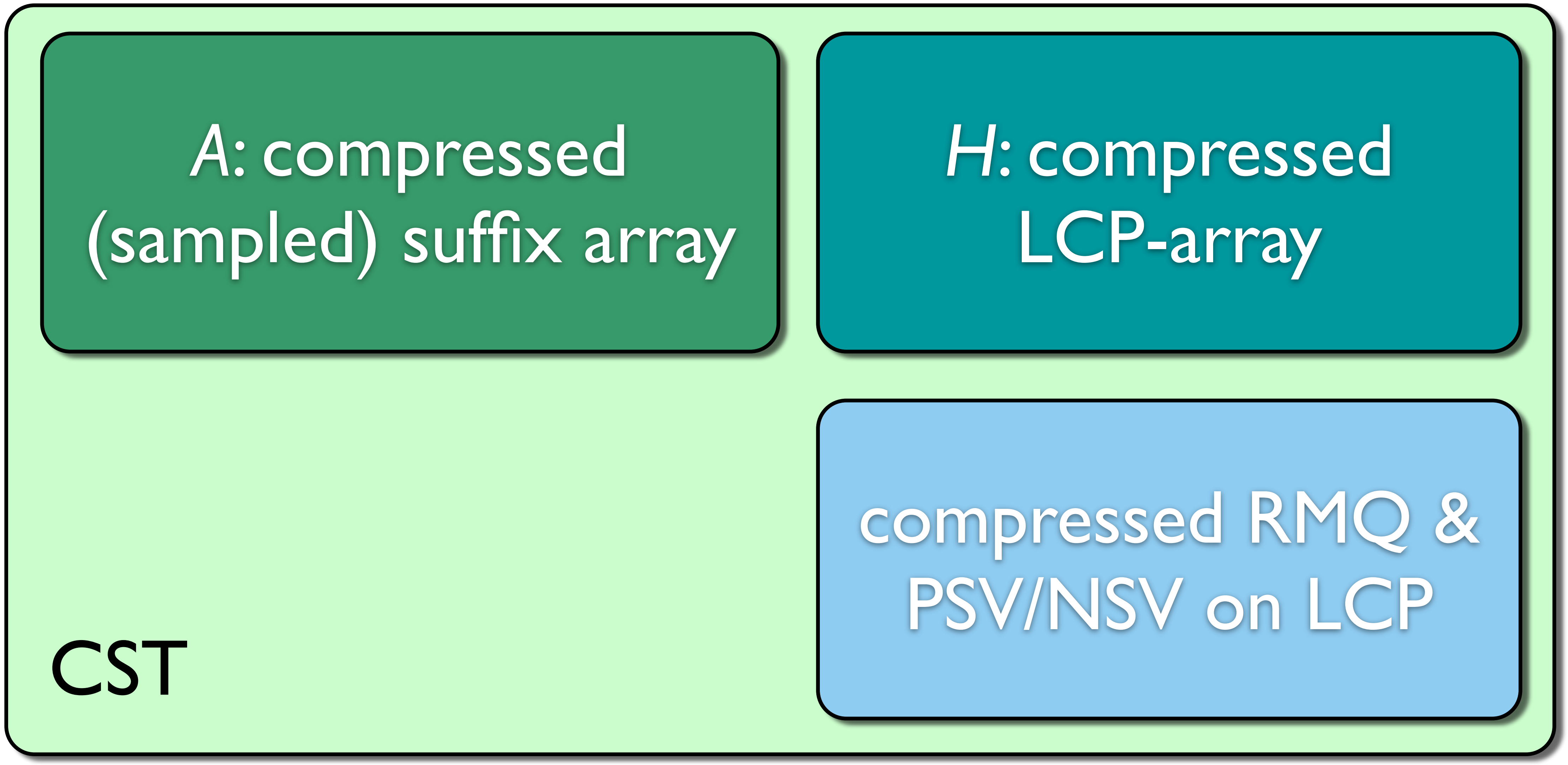
1. $H[i] \geq \mathrm{SDEPTH}(v)$
   $\forall\ v_l < i \leq v_r$
2. $H[v_l] < \mathrm{SDEPTH}(v)$
   $H[v_r+1] < \mathrm{SDEPTH}(v)$
3. $\exists\ v_l < i \leq v_r$ with
   $H[i] = \mathrm{SDEPTH}(v)$

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

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

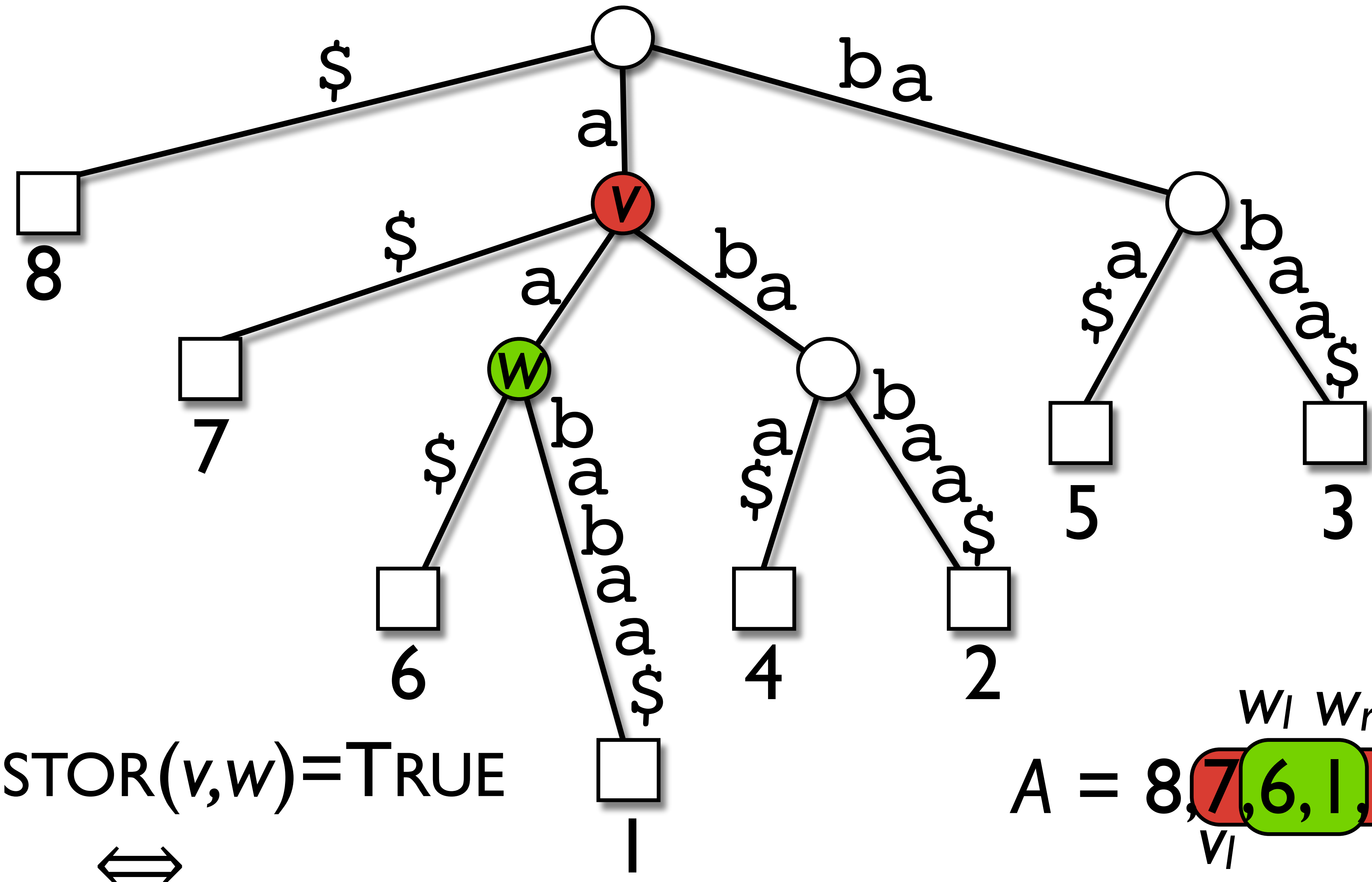


3 Components of CST


A: compressed (sampled) suffix array
H: compressed LCP-array
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

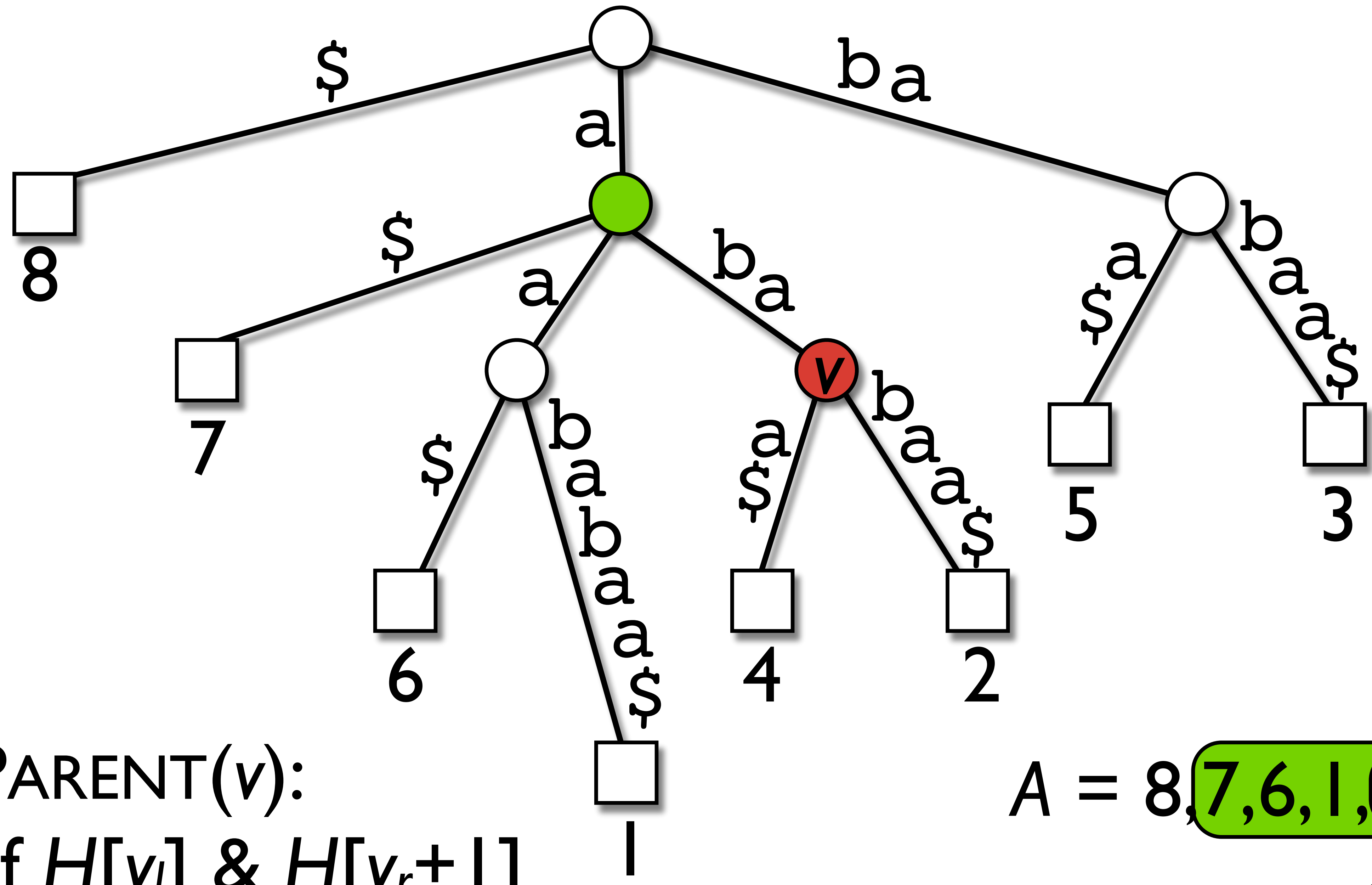
$\Leftrightarrow$

$v_l \leq w_r \leq v_r$

$$A = 8,\underset{v_l}{7},\overset{w_l}{6},\overset{w_r}{1},4,\underset{v_r}{2},5,3$$

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

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