

Wheeler graphs, part 1

Ben Langmead



JOHNS HOPKINS

WHITING SCHOOL
of ENGINEERING

Department of Computer Science



Please sign guestbook (www.langmead-lab.org/teaching-materials) to tell me briefly how you are using the slides. For original Keynote files, email me (ben.langmead@gmail.com).

BWT: matching

P : g a t

\$	g	a	t	t	a	c	a	t
a	c	a	t	\$	g	a	t	t
a	t	\$	g	a	t	t	a	c
a	t	t	a	c	a	t	\$	g
c	a	t	\$	g	a	t	t	a
g	a	t	t	a	c	a	t	\$
t	\$	g	a	t	t	a	c	a
t	a	c	a	t	\$	g	a	t
t	t	a	c	a	t	\$	g	a

BWT: matching

P : g a t

\$	g	a	t	t	a	c	a	t
a	c	a	t	\$	g	a	t	t
a	t	\$	g	a	t	t	a	c
a	t	t	a	c	a	t	\$	g
c	a	t	\$	g	a	t	t	a
g	a	t	t	a	c	a	t	\$
t	\$	g	a	t	t	a	c	a
t	a	c	a	t	\$	g	a	t
t	t	a	c	a	t	\$	g	a

BWT: matching

P : g a t

\$	g	a	t	t	a	c	a	t
a	c	a	t	\$	g	a	t	t
a	t	\$	g	a	t	t	a	c
a	t	t	a	c	a	t	\$	g
c	a	t	\$	g	a	t	t	a
g	a	t	t	a	c	a	t	\$
t	\$	g	a	t	t	a	c	a
t	a	c	a	t	\$	g	a	t
t	t	a	c	a	t	\$	g	a

BWT: matching

P : g a t

\$	g	a	t	t	a	c	a	t
a	c	a	t	\$	g	a	t	t
a	t	\$	g	a	t	t	a	c
a	t	t	a	c	a	t	\$	g
c	a	t	\$	g	a	t	t	a
g	a	t	t	a	c	a	t	\$
t	\$	g	a	t	t	a	c	a
t	a	c	a	t	\$	g	a	t
t	t	a	c	a	t	\$	g	a

BWT: matching

T: g a t t a c a t \$

P: g a t

\$	g	a	t	t	a	c	a	t
a	c	a	t	\$	g	a	t	t
a	t	\$	g	a	t	t	a	c
a	t	t	a	c	a	t	\$	g
c	a	t	\$	g	a	t	t	a
g	a	t	t	a	c	a	t	\$
t	\$	g	a	t	t	a	c	a
t	a	c	a	t	\$	g	a	t
t	t	a	c	a	t	\$	g	a

BWT: matching

T : g a t t a c a t \$

P : g a t

\$	g	a	t	t	a	c	a	t
a	c	a	t	\$	g	a	t	t
a	t	\$	g	a	t	t	a	c
a	t	t	a	c	a	t	\$	g
c	a	t	\$	g	a	t	t	a
g	a	t	t	a	c	a	t	\$
t	\$	g	a	t	t	a	c	a
t	a	c	a	t	\$	g	a	t
t	t	a	c	a	t	\$	g	a

BWT: matching

T : g a t t a c a t \$

P : g a t

\$	g	a	t	t	a	c	a	t
a	c	a	t	\$	g	a	t	t
a	t	\$	g	a	t	a	c	
a	t	t	a	c	a	t	\$	g
c	a	t	\$	g	a	t	t	a
g	a	t	t	a	c	a	t	\$
t	\$	g	a	t	t	a	c	a
t	a	c	a	t	\$	g	a	t
t	t	a	c	a	t	\$	g	a

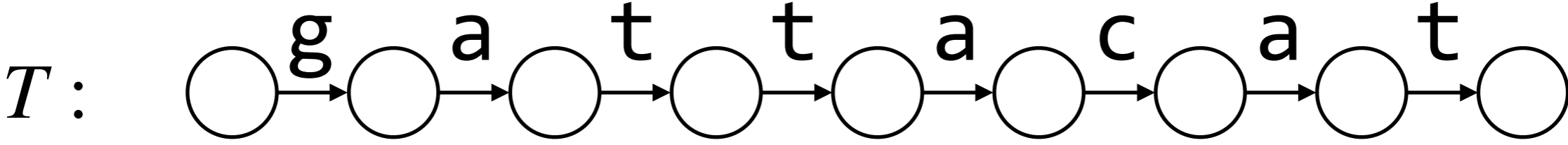
BWT: matching

T: g a t t a c a t \$

P: g a t

\$	g	a	t	t	a	c	a	t
a	c	a	t	\$	g	a	t	t
a	t	\$	g	a	t	t	a	c
a	t	t	a	c	a	t	\$	g
c	a	t	\$	g	a	t	t	a
g	a	t	t	a	c	a	t	\$
t	\$	g	a	t	t	a	c	a
t	a	c	a	t	\$	g	a	t
t	t	a	c	a	t	\$	g	a

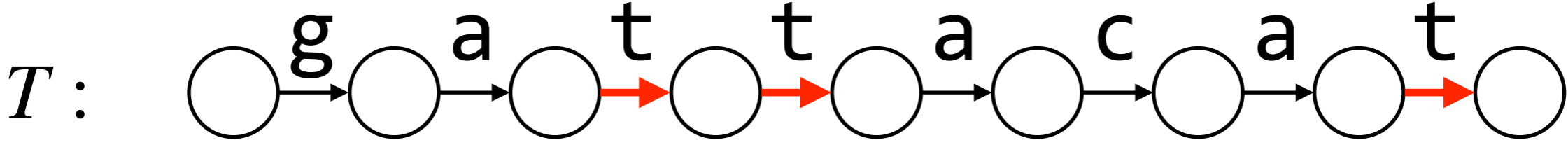
BWT: matching



P: g a t

\$	g	a	t	t	a	c	a	t
a	c	a	t	\$	g	a	t	t
a	t	\$	g	a	t	t	a	c
a	t	t	a	c	a	t	\$	g
c	a	t	\$	g	a	t	t	a
g	a	t	t	a	c	a	t	\$
t	\$	g	a	t	t	a	c	a
t	a	c	a	t	\$	g	a	t
t	t	a	c	a	t	\$	g	a

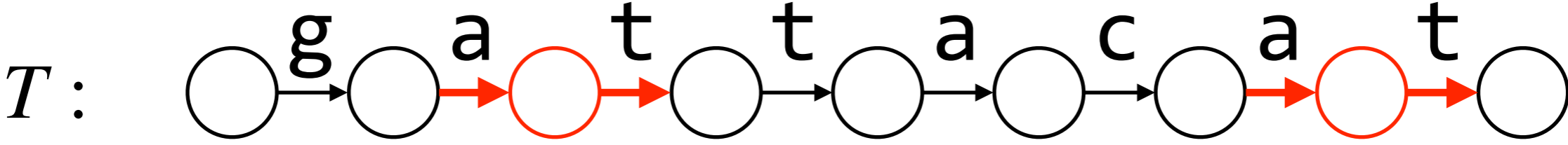
BWT: matching



P: g a t

\$	g	a	t	t	a	c	a	t
a	c	a	t	\$	g	a	t	t
a	t	\$	g	a	t	t	a	c
a	t	t	a	c	a	t	\$	g
c	a	t	\$	g	a	t	t	a
g	a	t	t	a	c	a	t	\$
t	\$	g	a	t	t	a	c	a
t	a	c	a	t	\$	g	a	t
t	t	a	c	a	t	\$	g	a

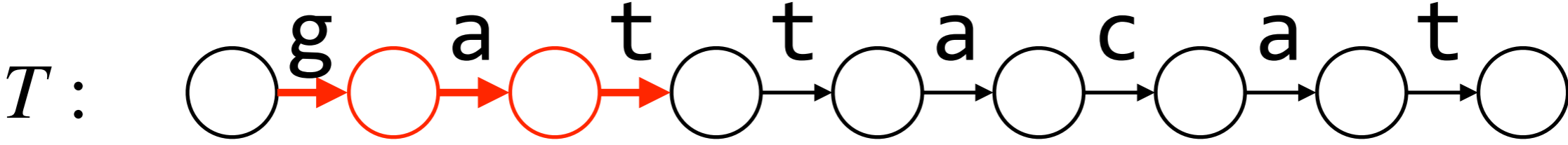
BWT: matching



P: g a t

\$	g	a	t	t	a	c	a	t
a	c	a	t	\$	g	a	t	t
a	t	\$	g	a	t	t	a	c
a	t	t	a	c	a	t	\$	g
c	a	t	\$	g	a	t	t	a
g	a	t	t	a	c	a	t	\$
t	\$	g	a	t	t	a	c	a
t	a	c	a	t	\$	g	a	t
t	t	a	c	a	t	\$	g	a

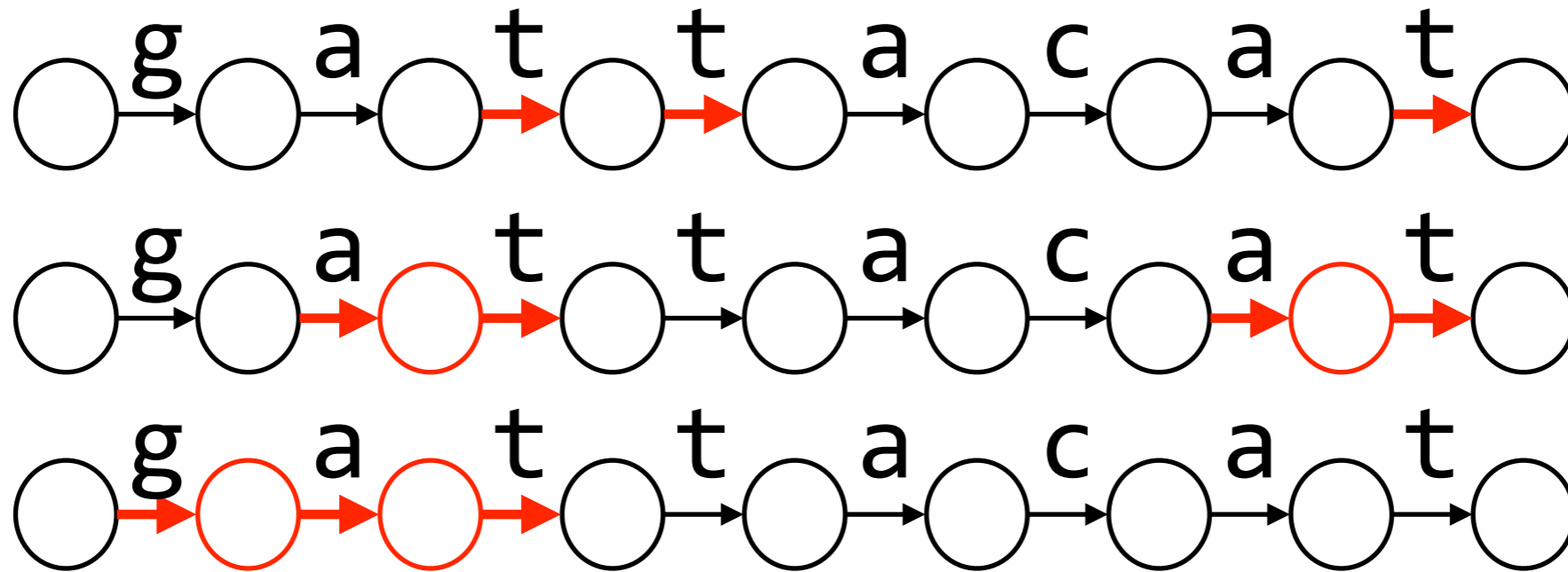
BWT: matching



P: **g a t**

\$	g	a	t	t	a	c	a	t
a	c	a	t	\$	g	a	t	t
a	t	\$	g	a	t	t	a	c
a	t	t	a	c	a	t	\$	g
c	a	t	\$	g	a	t	t	a
g	a	t	t	a	c	a	t	\$
t	\$	g	a	t	t	a	c	a
t	a	c	a	t	\$	g	a	t
t	t	a	c	a	t	\$	g	a

BWT: matching



g a **t t** a c a t

g **a t t** a c **a t**

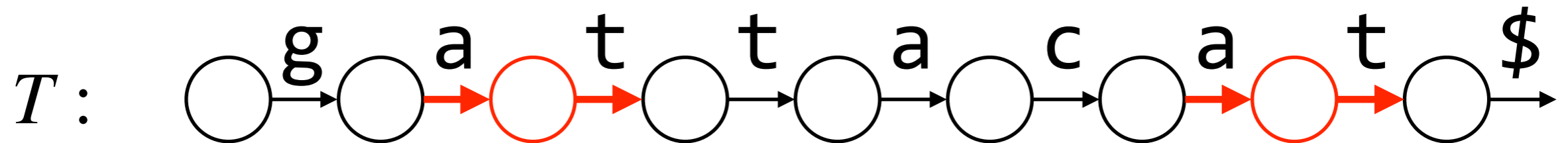
g a t t a c a t

Two interpretations:

Matching a *substring* in a *string*

Matching a *path* in a *graph*

BWT: matching

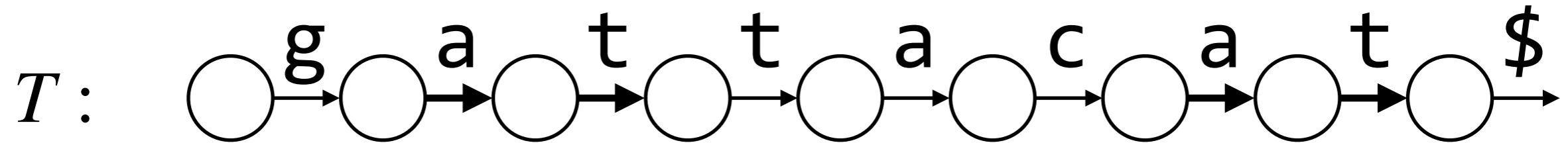


\$	g	a	t	t	a	c	a	t
a	c	a	t	\$	g	a	t	t
a	t	\$	g	a	t	t	a	c
a	t	t	a	c	a	t	\$	g
c	a	t	\$	g	a	t	t	a
g	a	t	t	a	c	a	t	\$
t	\$	g	a	t	t	a	c	a
t	a	c	a	t	\$	g	a	t
t	t	a	c	a	t	\$	g	a

Consecutivity. In BW order, rows with same prefix are consecutive.

Is this visible in the graph? Let's label nodes with **BW order**...

BWT: matching

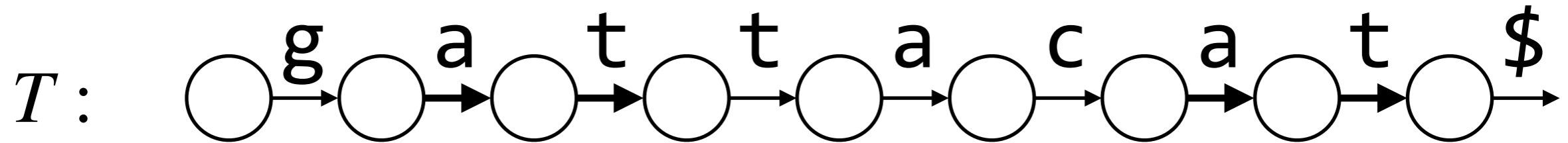


\$ g a t t a c a t
a c a t \$ g a t t
a t \$ g a t t a c
a t t a c a t \$ g
c a t \$ g a t t a
g a t t a c a t \$
t \$ g a t t a c a
t a c a t \$ g a t
t t a c a t \$ g a

Consecutivity. In BW order, rows with same prefix are consecutive.

Is this visible in the graph? Let's label nodes with **BW order**...

BWT: matching

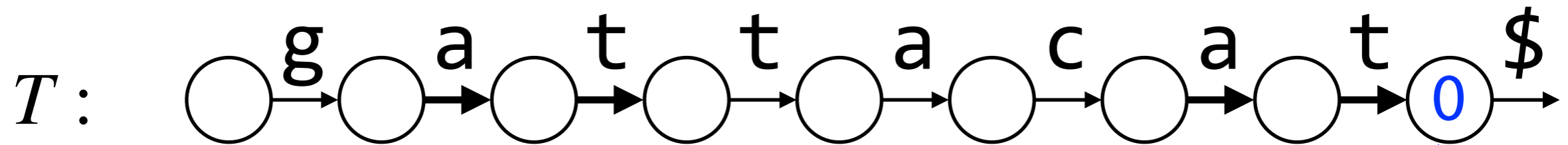


0	\$	g	a	t	t	a	c	a	t
1	a	c	a	t	\$	g	a	t	t
2	a	t	\$	g	a	t	t	a	c
3	a	t	t	a	c	a	t	\$	g
4	c	a	t	\$	g	a	t	t	a
5	g	a	t	t	a	c	a	t	\$
6	t	\$	g	a	t	t	a	c	a
7	t	a	c	a	t	\$	g	a	t
8	t	t	a	c	a	t	\$	g	a

Consecutivity. In BW order, rows with same prefix are consecutive.

Is this visible in the graph? Let's label nodes with **BW order**...

BWT: matching

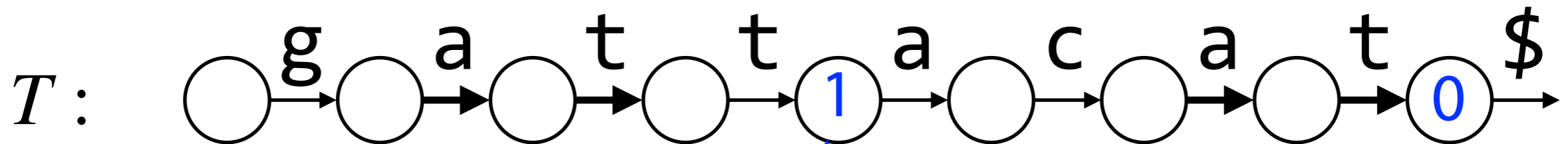


0	\$	g	a	t	t	a	c	a	t
1	a	c	a	t	\$	g	a	t	t
2	a	t	\$	g	a	t	t	a	c
3	a	t	t	a	c	a	t	\$	g
4	c	a	t	\$	g	a	t	t	a
5	g	a	t	t	a	c	a	t	\$
6	t	\$	g	a	t	t	a	c	a
7	t	a	c	a	t	\$	g	a	t
8	t	t	a	c	a	t	\$	g	a

Consecutivity. In BW order, rows with same prefix are consecutive.

Is this visible in the graph? Let's label nodes with **BW order**...

BWT: matching

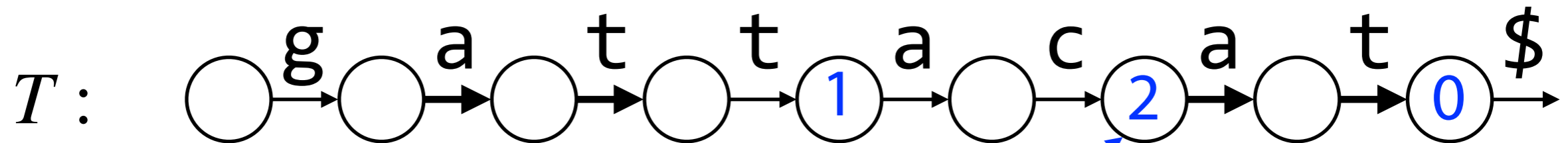


0	\$	g	a	t	t	a	c	a	t
1	a	c	a	t	\$	g	a	t	t
2	a	t	\$	g	a	t	t	a	c
3	a	t	t	a	c	a	t	\$	g
4	c	a	t	\$	g	a	t	t	a
5	g	a	t	t	a	c	a	t	\$
6	t	\$	g	a	t	t	a	c	a
7	t	a	c	a	t	\$	g	a	t
8	t	t	a	c	a	t	\$	g	a

Consecutivity. In BW order, rows with same prefix are consecutive.

Is this visible in the graph? Let's label nodes with **BW order**...

BWT: matching

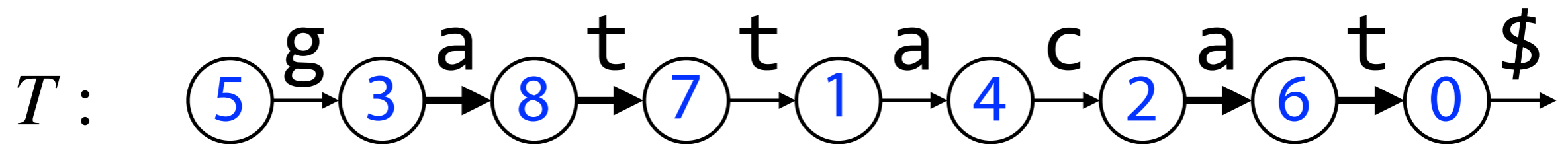


0	\$	g	a	t	t	a	c	a	t
1	a	c	a	t	\$	g	a	t	t
2	a	t	\$	g	a	t	t	a	c
3	a	t	t	a	c	a	t	\$	g
4	c	a	t	\$	g	a	t	t	a
5	g	a	t	t	a	c	a	t	\$
6	t	\$	g	a	t	t	a	c	a
7	t	a	c	a	t	\$	g	a	t
8	t	t	a	c	a	t	\$	g	a

Consecutivity. In BW order, rows with same prefix are consecutive.

Is this visible in the graph? Let's label nodes with **BW order**...

BWT: matching

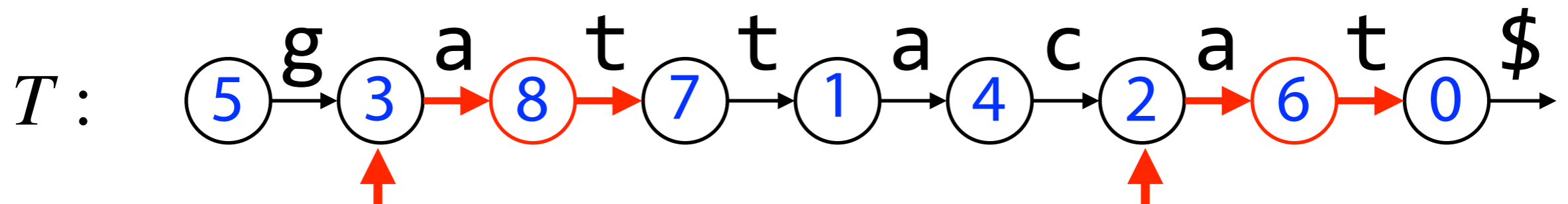


0	\$	g	a	t	t	a	c	a	t
1	a	c	a	t	\$	g	a	t	t
2	a	t	\$	g	a	t	t	a	c
3	a	t	t	a	c	a	t	\$	g
4	c	a	t	\$	g	a	t	t	a
5	g	a	t	t	a	c	a	t	\$
6	t	\$	g	a	t	t	a	c	a
7	t	a	c	a	t	\$	g	a	t
8	t	t	a	c	a	t	\$	g	a

Consecutivity. In BW order, rows with same prefix are consecutive.

Is this visible in the graph? Let's label nodes with **BW order**...

BWT: matching

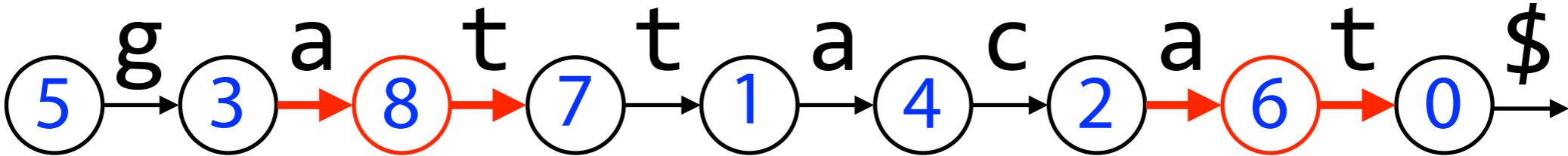


0	\$	g	a	t	t	a	c	a	t
1	a	c	a	t	\$	g	a	t	t
2	a	t	\$	g	a	t	t	a	c
3	a	t	t	a	c	a	t	\$	g
4	c	a	t	\$	g	a	t	t	a
5	g	a	t	t	a	c	a	t	\$
6	t	\$	g	a	t	t	a	c	a
7	t	a	c	a	t	\$	g	a	t
8	t	t	a	c	a	t	\$	g	a

Consecutivity. In BW order, rows with same prefix are consecutive.

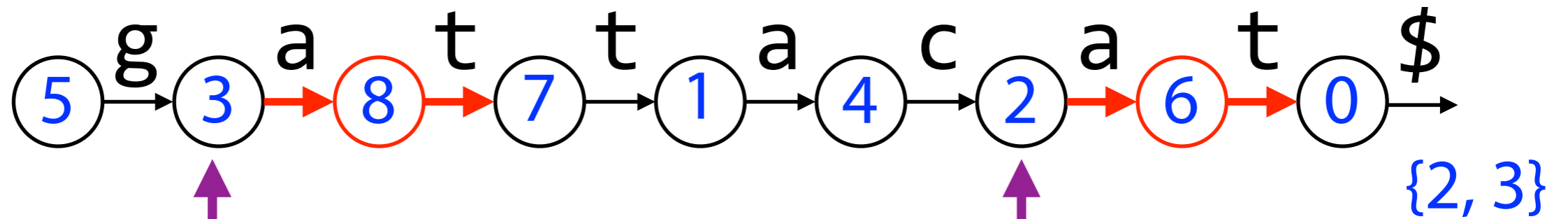
Is this visible in the graph? Let's label nodes with **BW order**...

BWT: matching



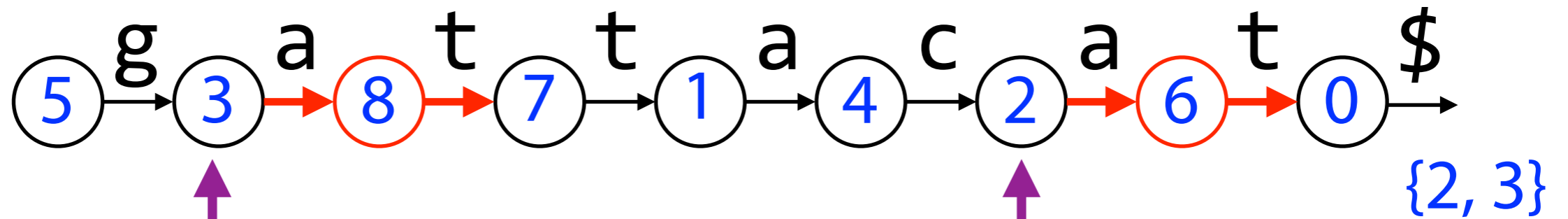
0	\$	g	a	t	t	a	c	a	t
1	a	c	a	t	\$	g	a	t	t
2	a	t	\$	g	a	t	t	a	c
3	a	t	t	a	c	a	t	\$	g
4	c	a	t	\$	g	a	t	t	a
5	g	a	t	t	a	c	a	t	\$
6	t	\$	g	a	t	t	a	c	a
7	t	a	c	a	t	\$	g	a	t
8	t	t	a	c	a	t	\$	g	a

BWT: matching



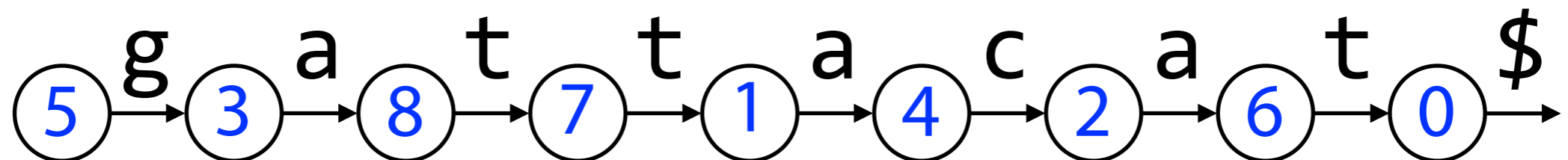
0	\$	g	a	t	t	a	c	a	t
1	a	c	a	t	\$	g	a	t	t
→ 2	a	t	\$	g	a	t	a	c	
→ 3	a	t	t	a	c	a	t	\$	g
4	c	a	t	\$	g	a	t	t	a
5	g	a	t	t	a	c	a	t	\$
6	t	\$	g	a	t	t	a	c	a
7	t	a	c	a	t	\$	g	a	t
8	t	t	a	c	a	t	\$	g	a

BWT: matching

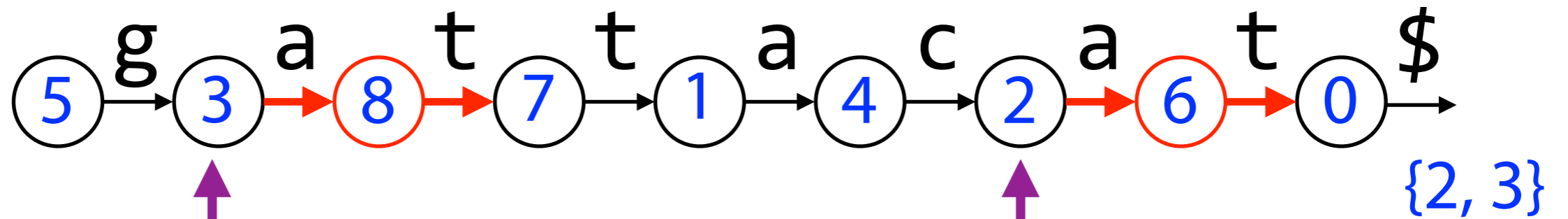


0	\$	g	a	t	t	a	c	a	t
1	a	c	a	t	\$	g	a	t	t
2	a	t	\$	g	a	t	a	c	
3	a	t	t	a	c	a	t	\$	g
4	c	a	t	\$	g	a	t	t	a
5	g	a	t	t	a	c	a	t	\$
6	t	\$	g	a	t	t	a	c	a
7	t	a	c	a	t	\$	g	a	t
8	t	t	a	c	a	t	\$	g	a

Node labels in BW range are consecutive

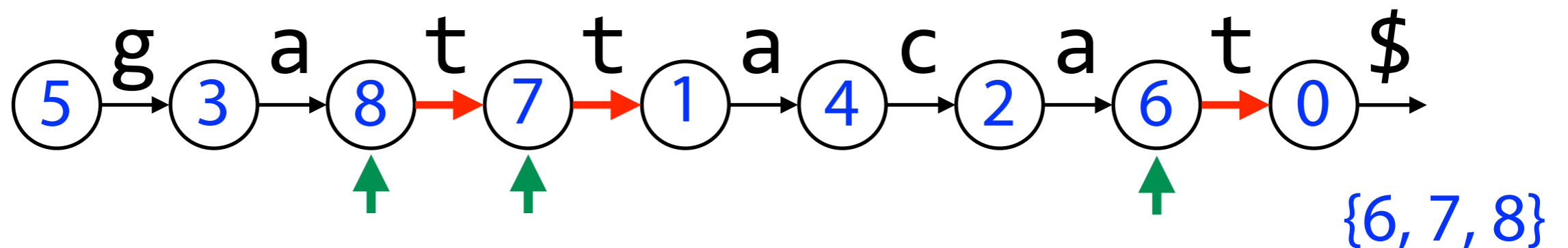


BWT: matching

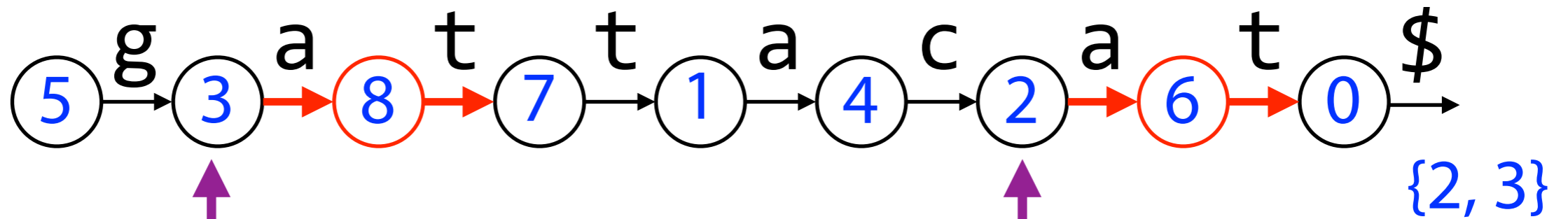


0	\$	g	a	t	t	a	c	a	t
1	a	c	a	t	\$	g	a	t	t
2	a	t	\$	g	a	t	a	c	
3	a	t	t	a	c	a	t	\$	g
4	c	a	t	\$	g	a	t	t	a
5	g	a	t	t	a	c	a	t	\$
6	t	\$	g	a	t	t	a	c	a
7	t	a	c	a	t	\$	g	a	t
8	t	t	a	c	a	t	\$	g	a

Node labels in BW range are consecutive



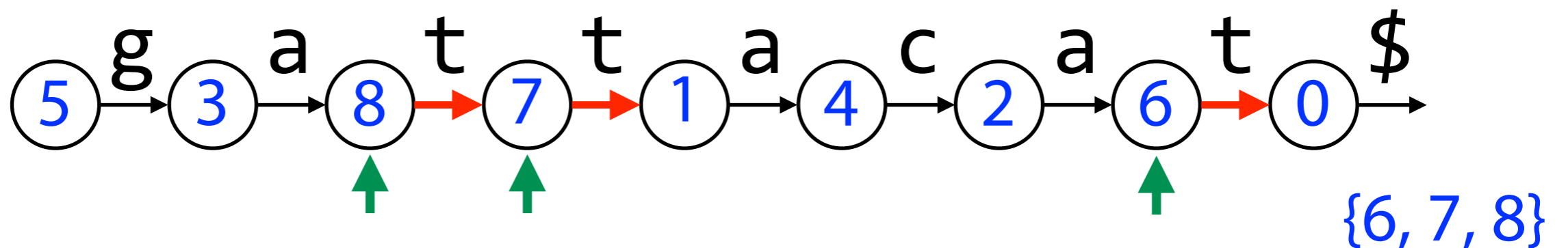
BWT: matching



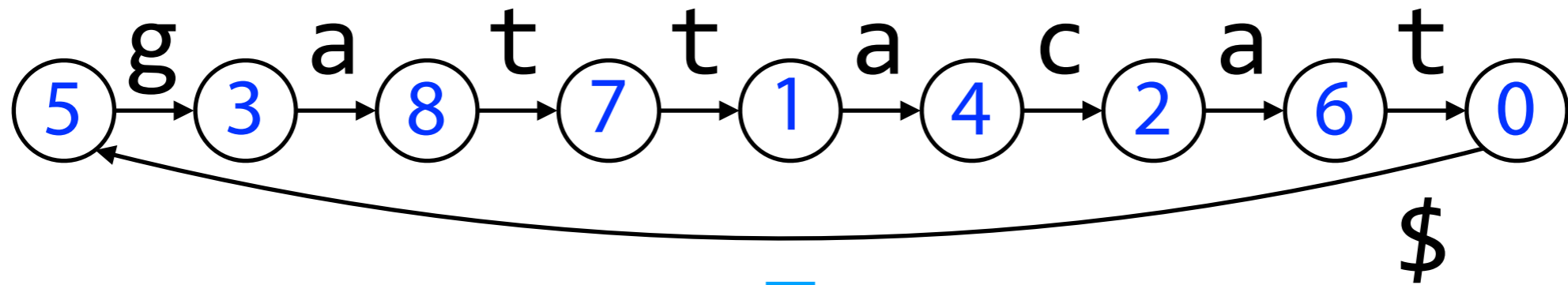
0	\$	g	a	t	t	a	c	a	t
1	a	c	a	t	\$	g	a	t	t
2	a	t	\$	g	a	t	a	c	
3	a	t	t	a	c	a	t	\$	g
4	c	a	t	\$	g	a	t	t	a
5	g	a	t	t	a	c	a	t	\$
6	t	\$	g	a	t	t	a	c	a
7	t	a	c	a	t	\$	g	a	t
8	t	t	a	c	a	t	\$	g	a

Node labels in BW range are consecutive

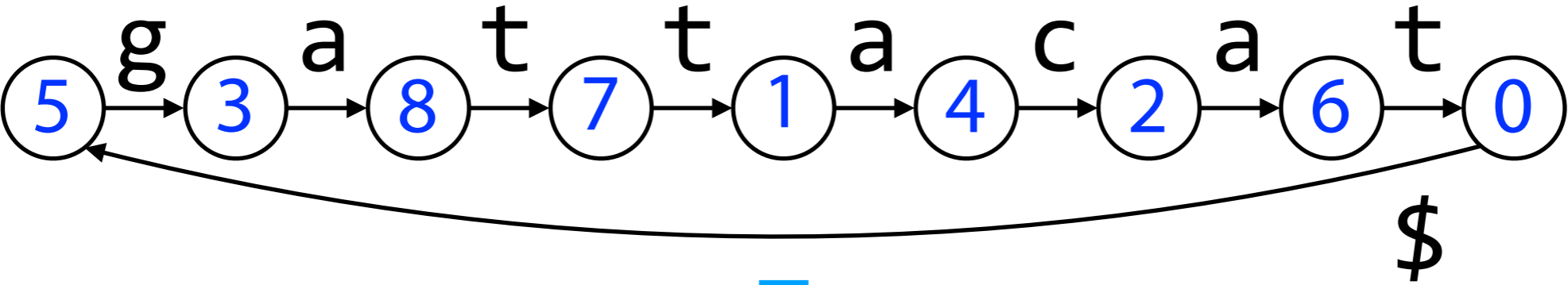
Clearer if we redrew graph in BWT(T) order



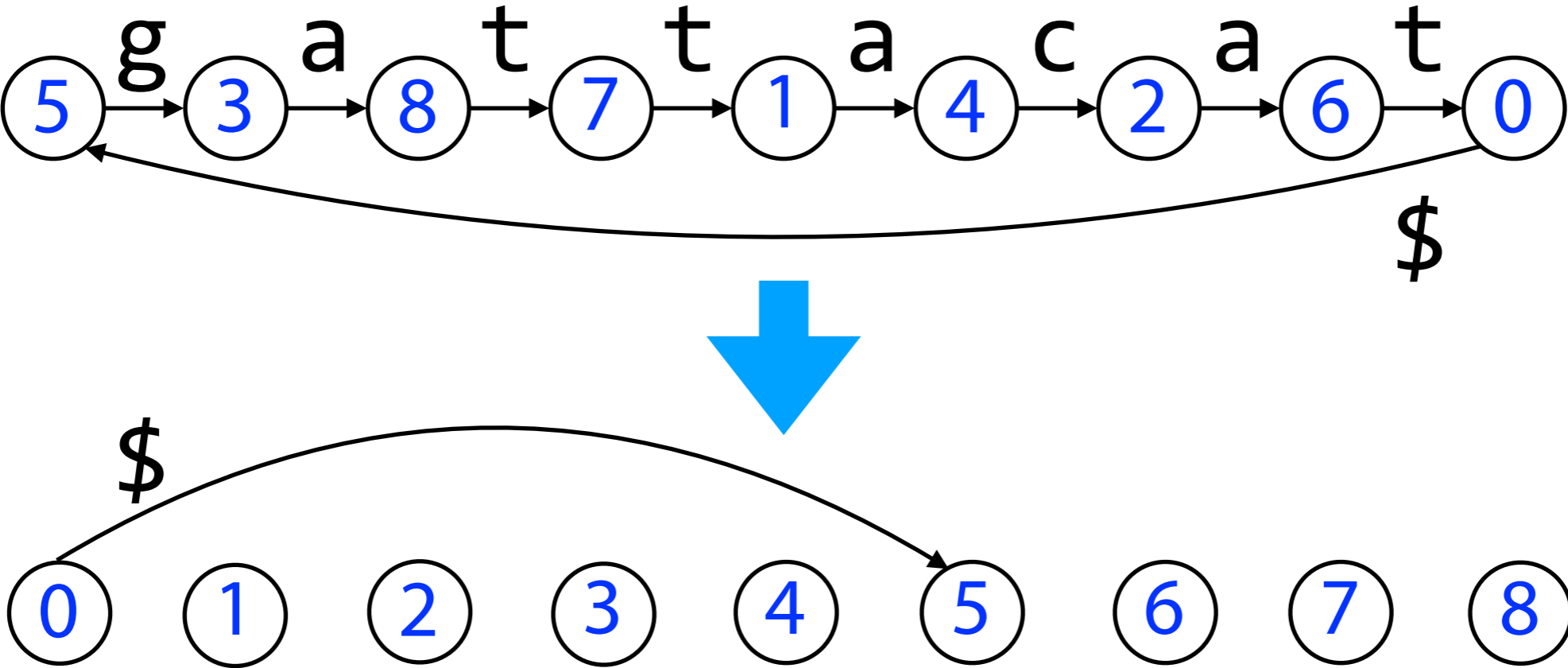
BWT: matching



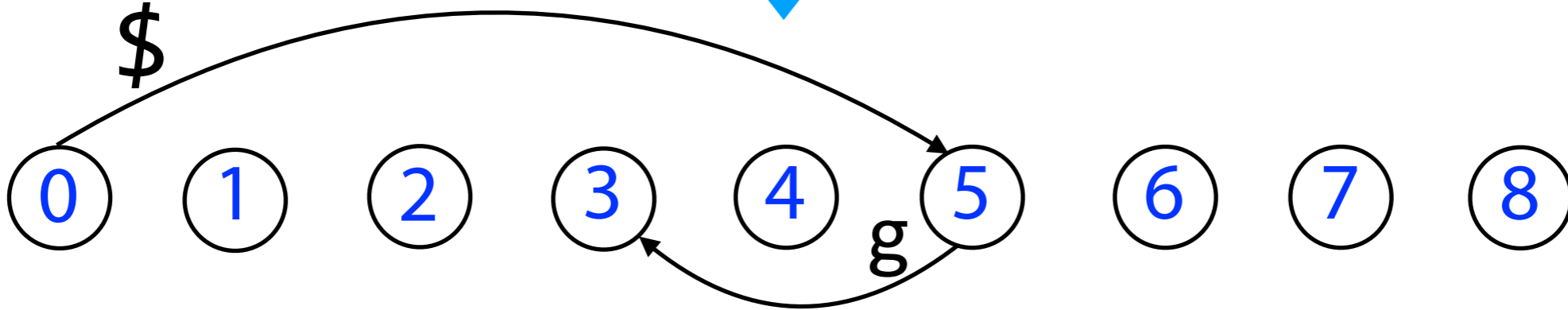
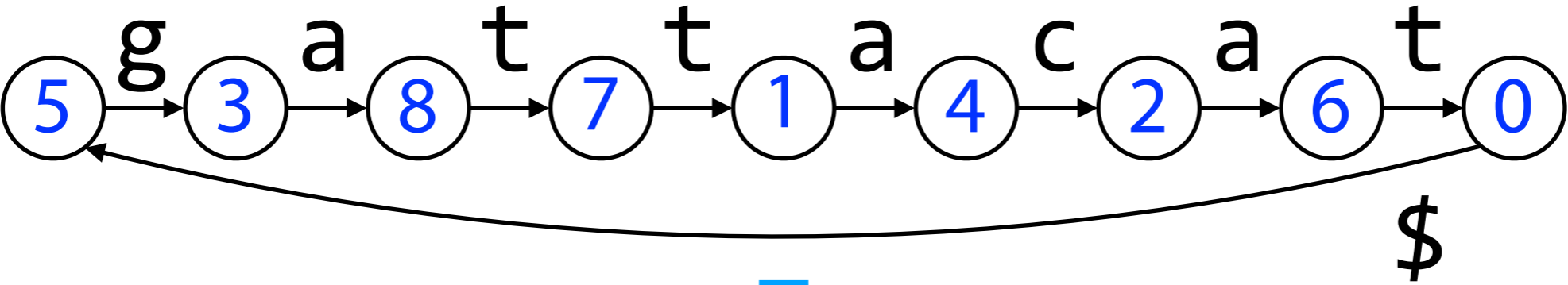
BWT: matching



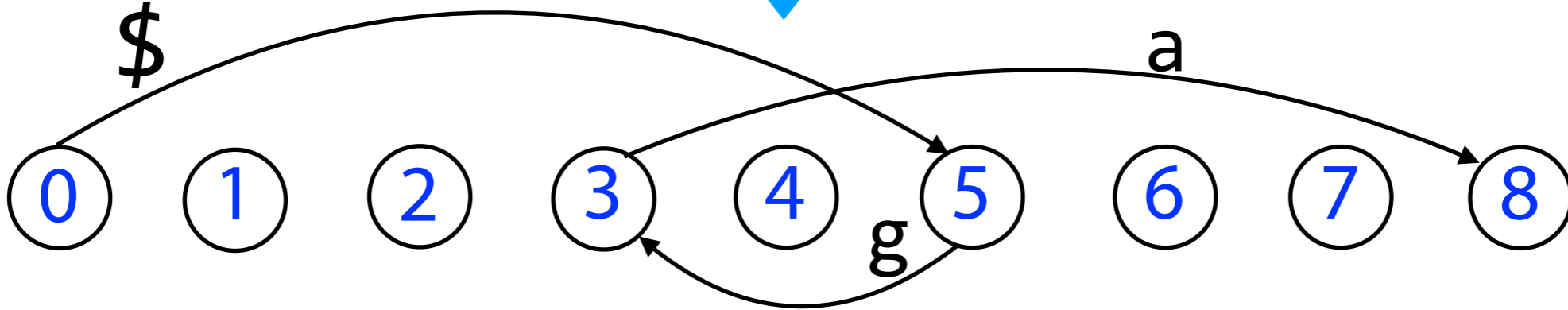
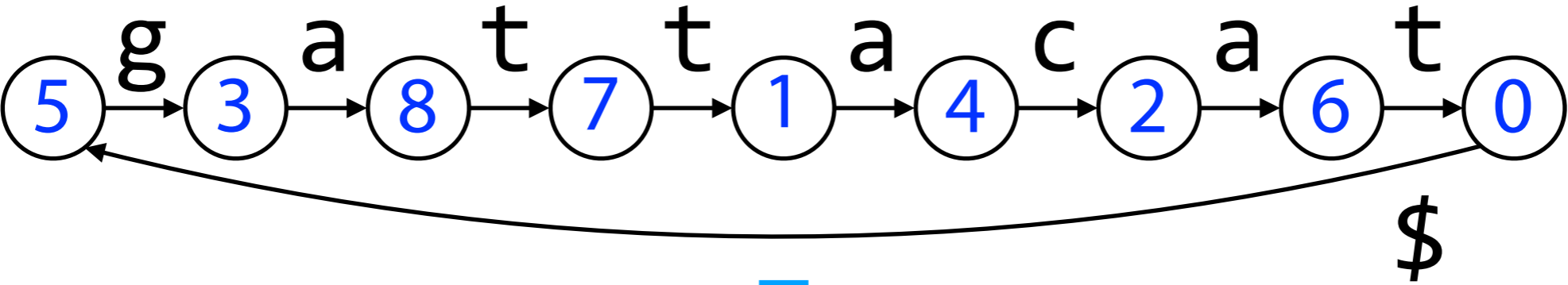
BWT: matching



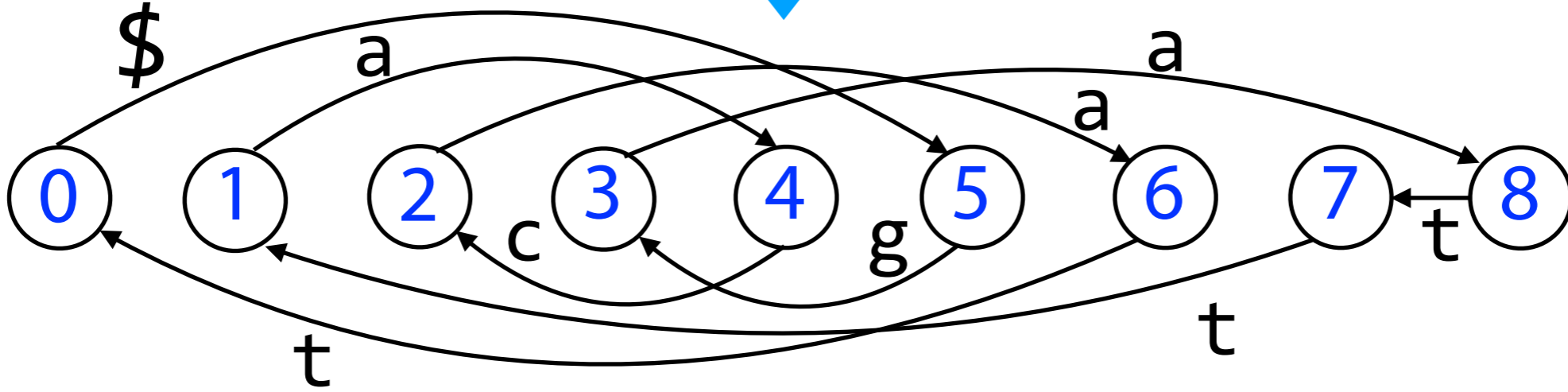
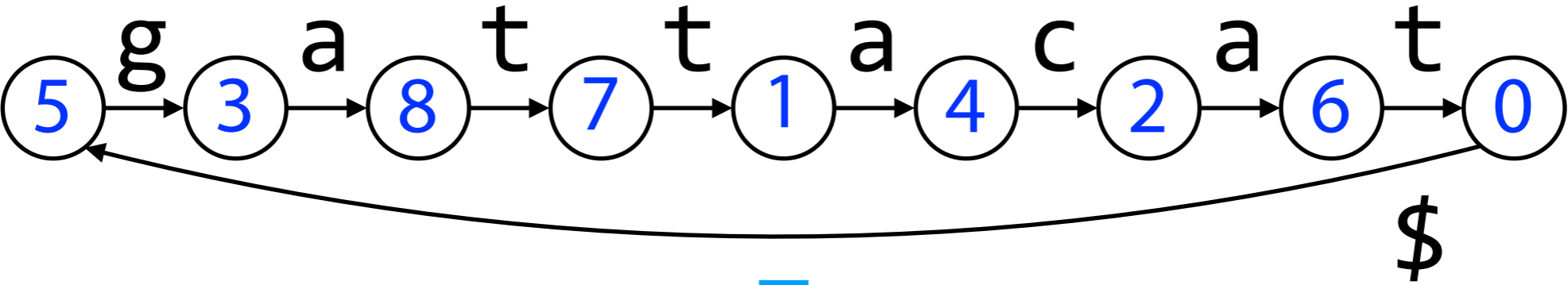
BWT: matching



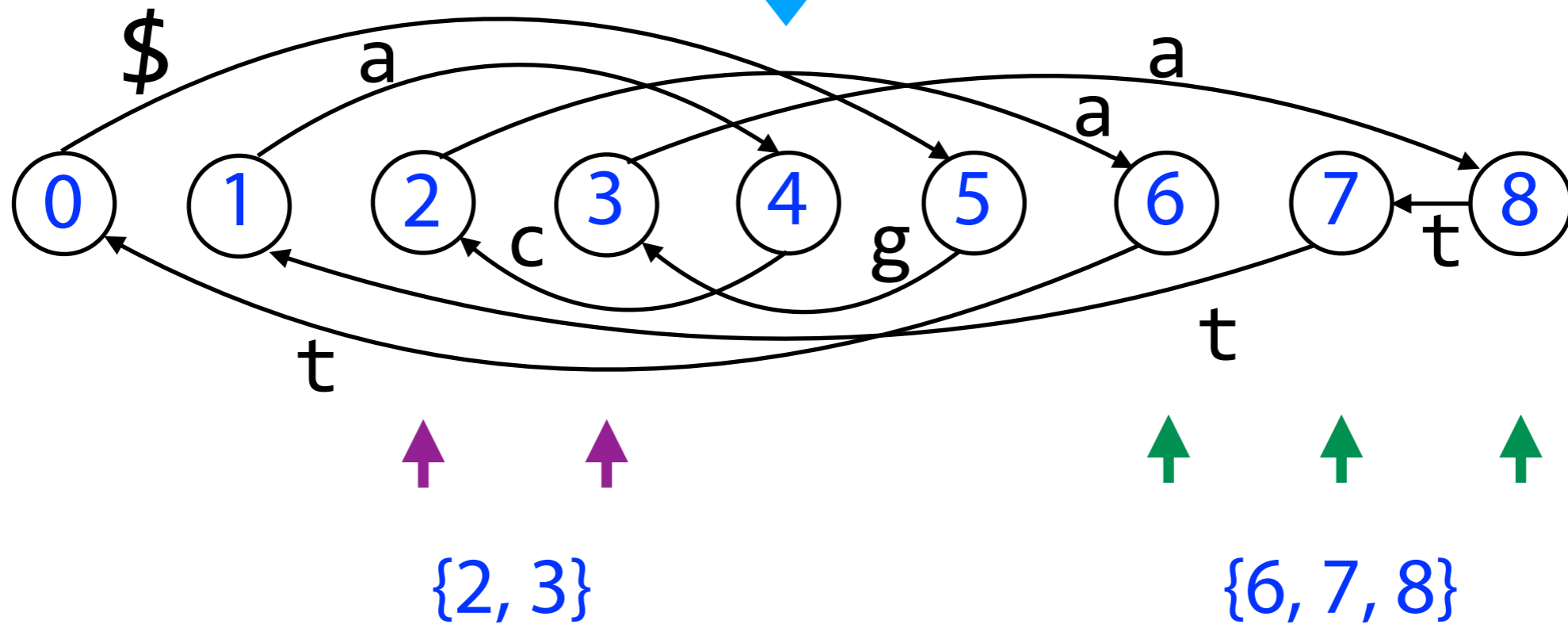
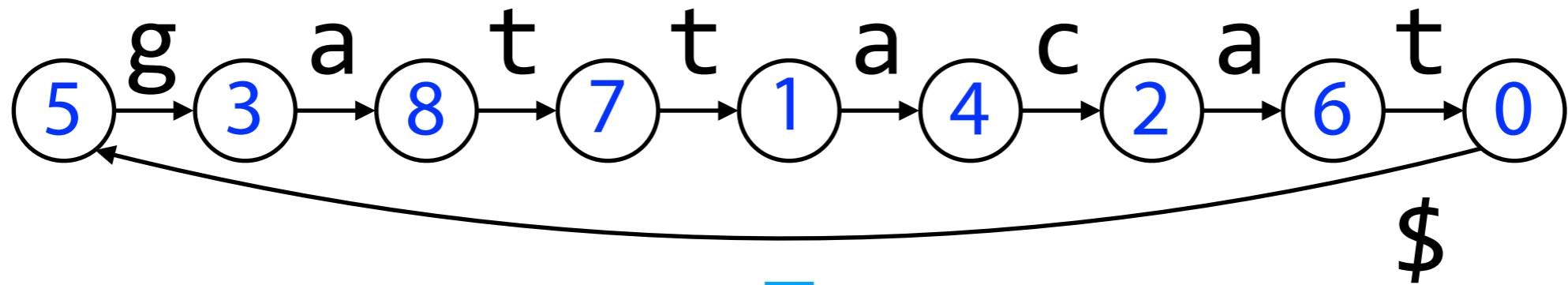
BWT: matching



BWT: matching

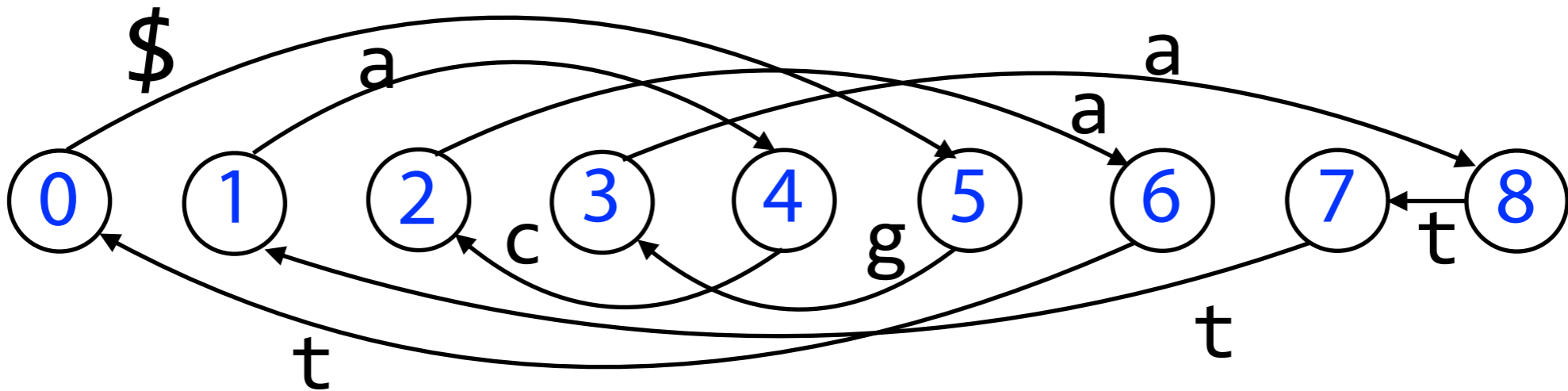


BWT: matching



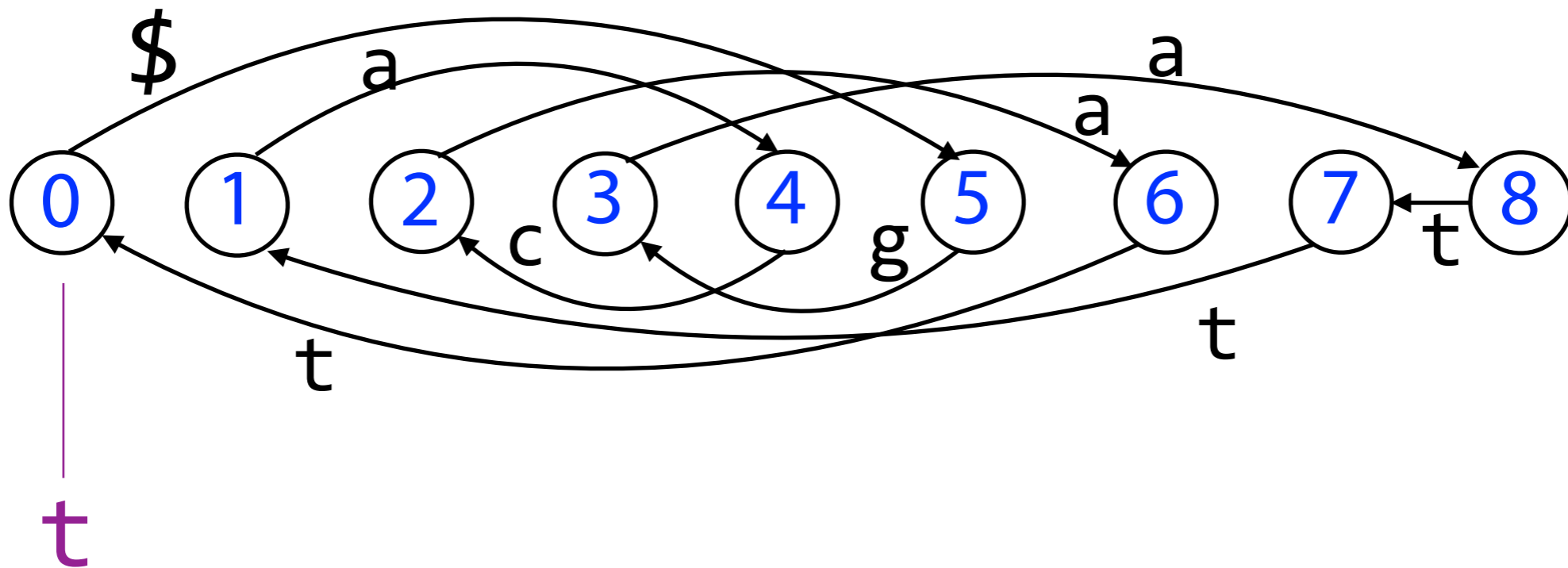
BWT: matching

Nodes can be thought of according to what comes **after** (outgoing edges) and or **just before** (incoming)



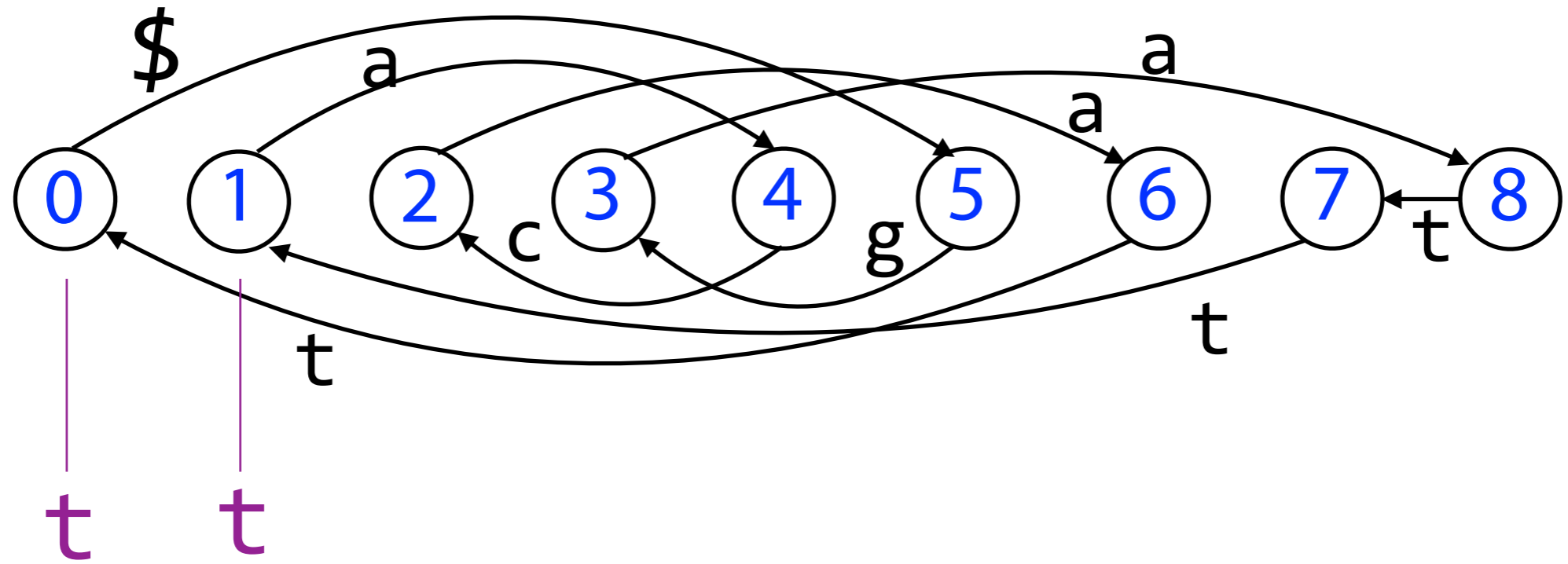
BWT: matching

Nodes can be thought of according to what comes **after** (outgoing edges) and or **just before** (incoming)



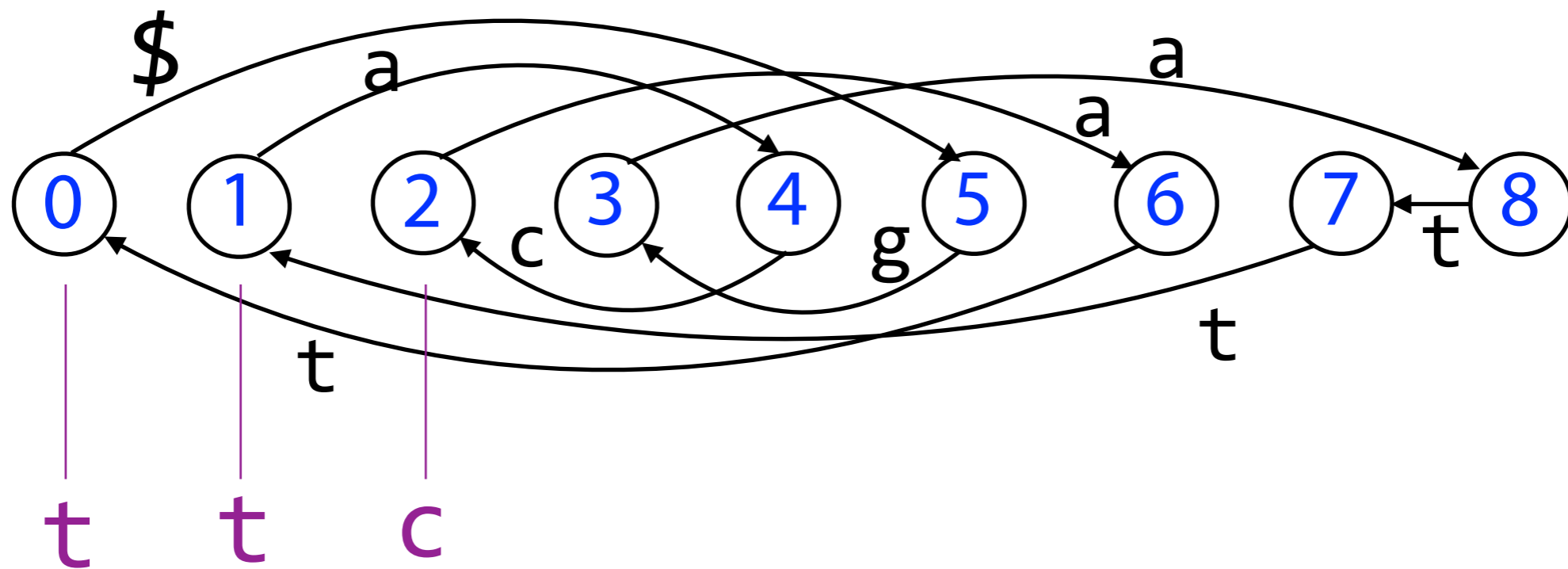
BWT: matching

Nodes can be thought of according to what comes **after** (outgoing edges) and or **just before** (incoming)



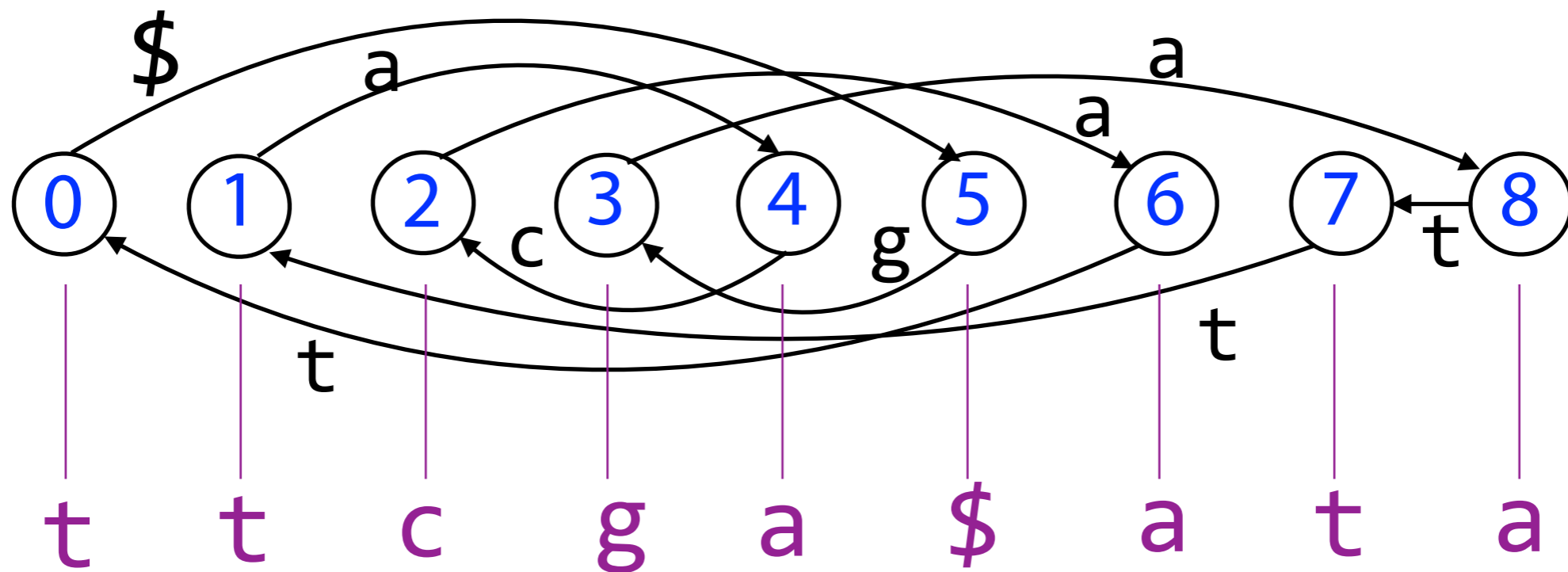
BWT: matching

Nodes can be thought of according to what comes **after** (outgoing edges) and or **just before** (incoming)



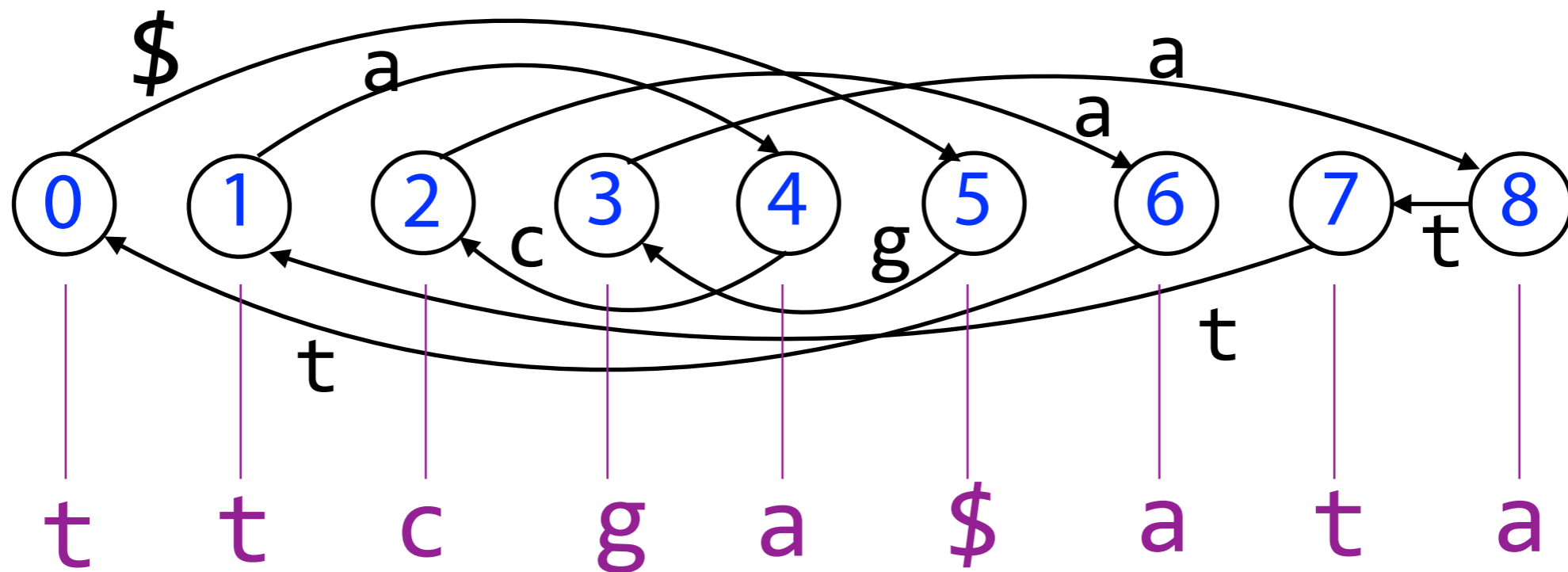
BWT: matching

Nodes can be thought of according to what comes **after** (outgoing edges) and or **just before** (incoming)



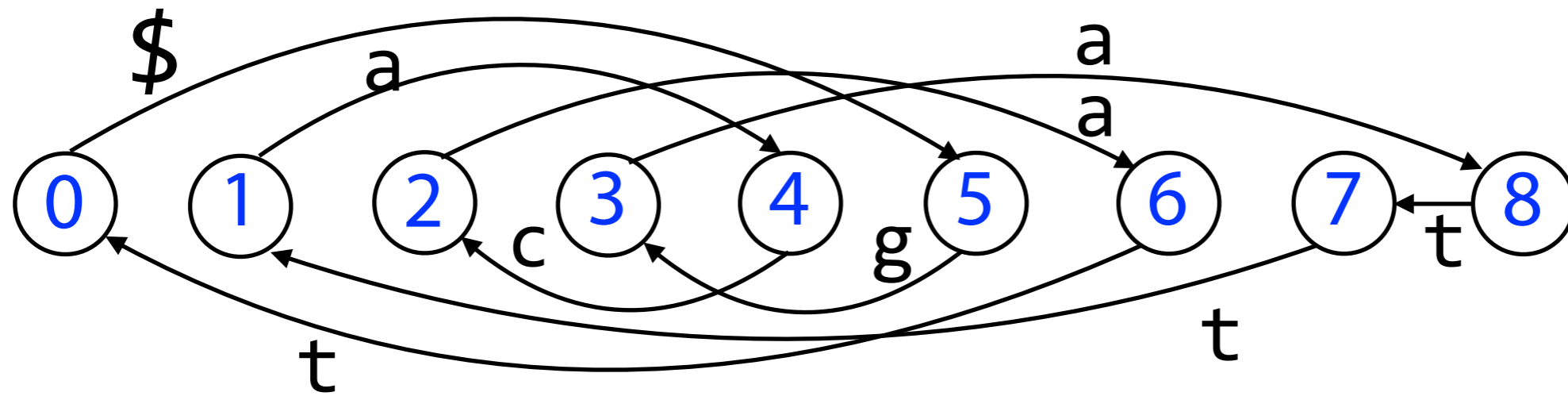
BWT: matching

Nodes can be thought of according to what comes **after** (outgoing edges) and or **just before** (incoming)

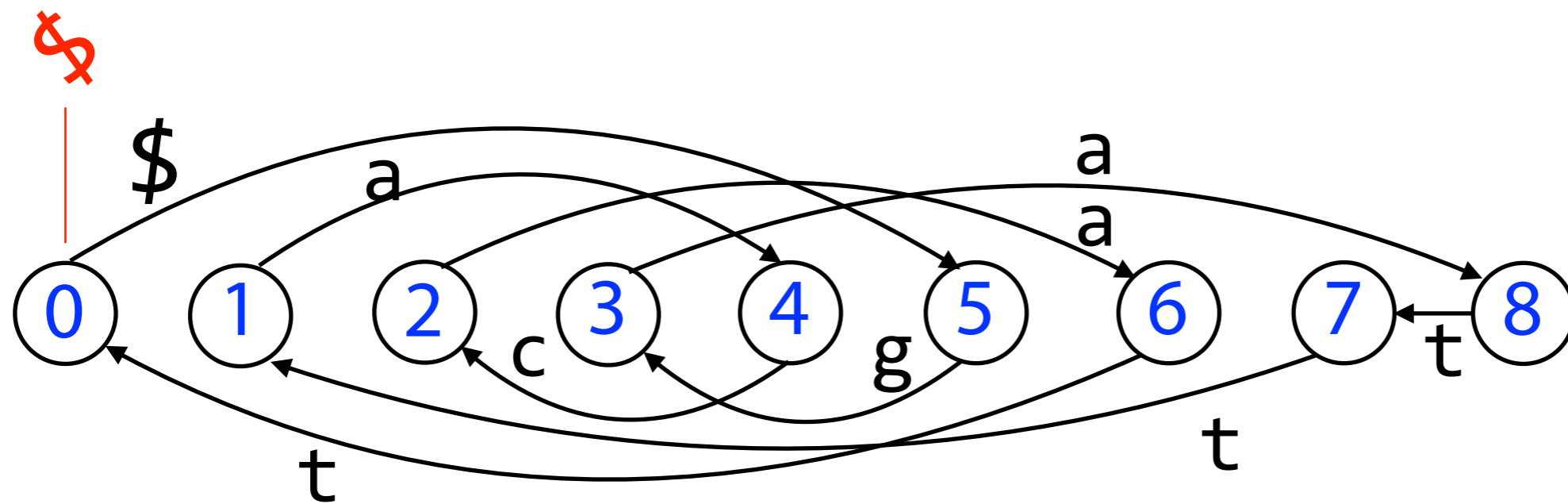


Incoming edges spell out BWT

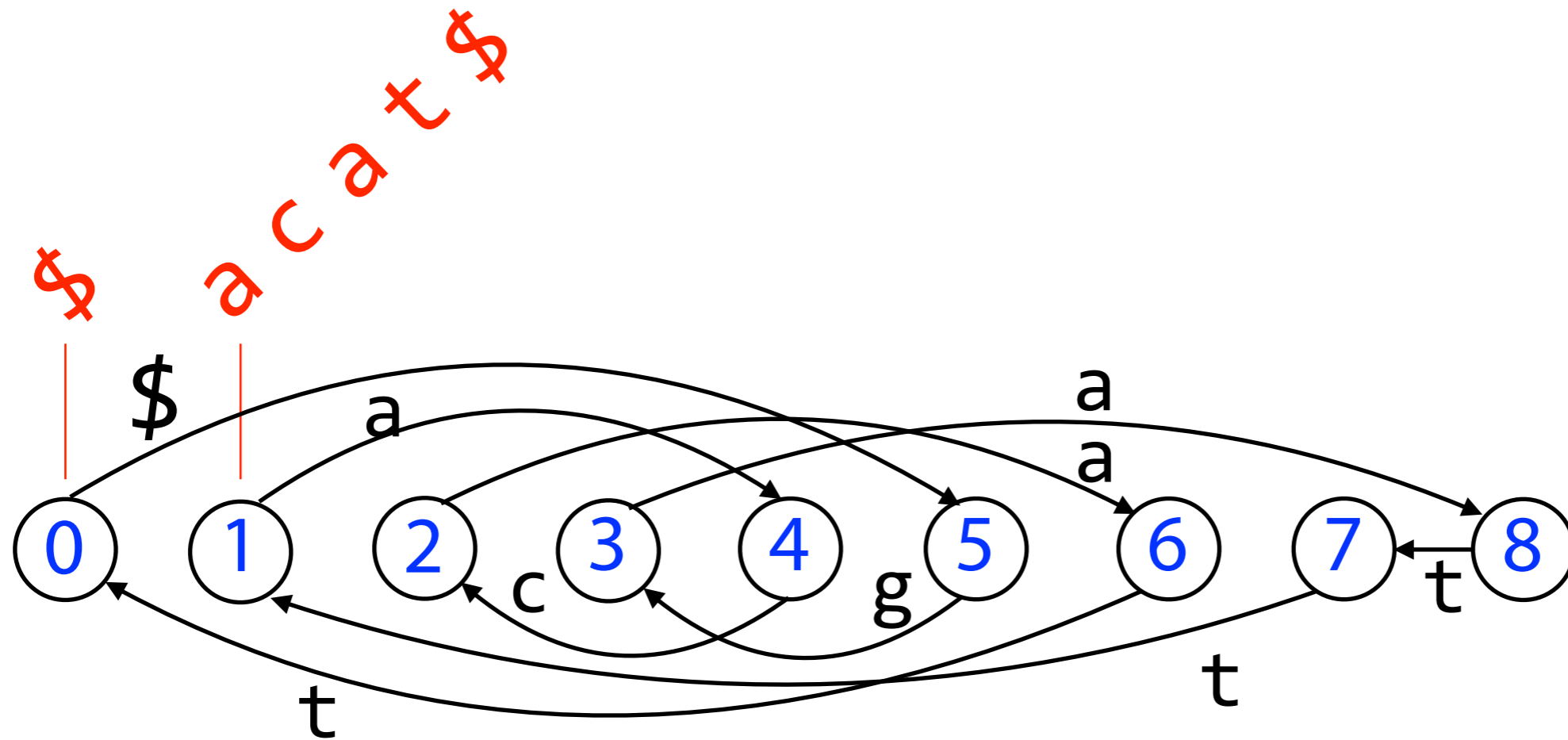
BWT: matching



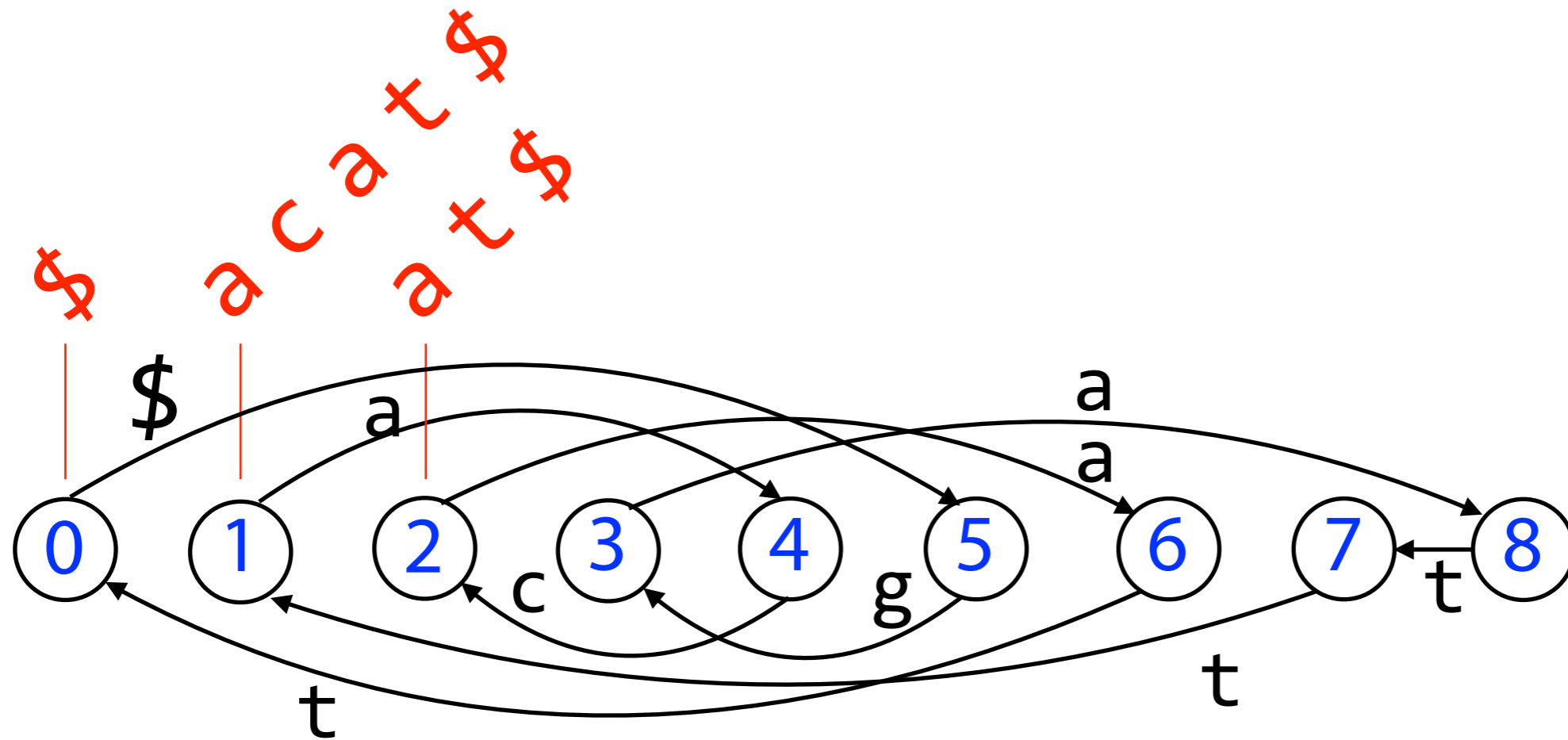
BWT: matching



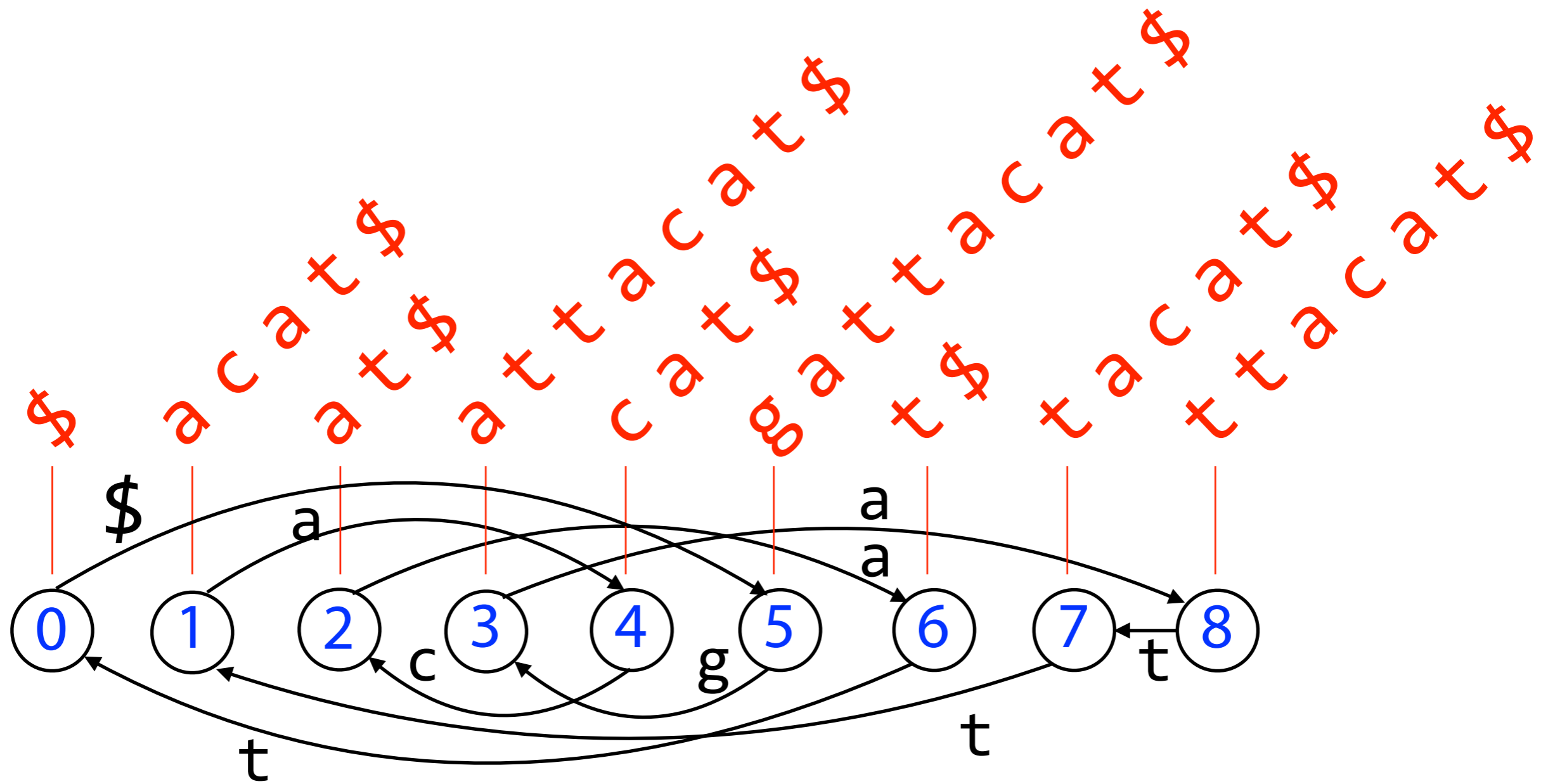
BWT: matching



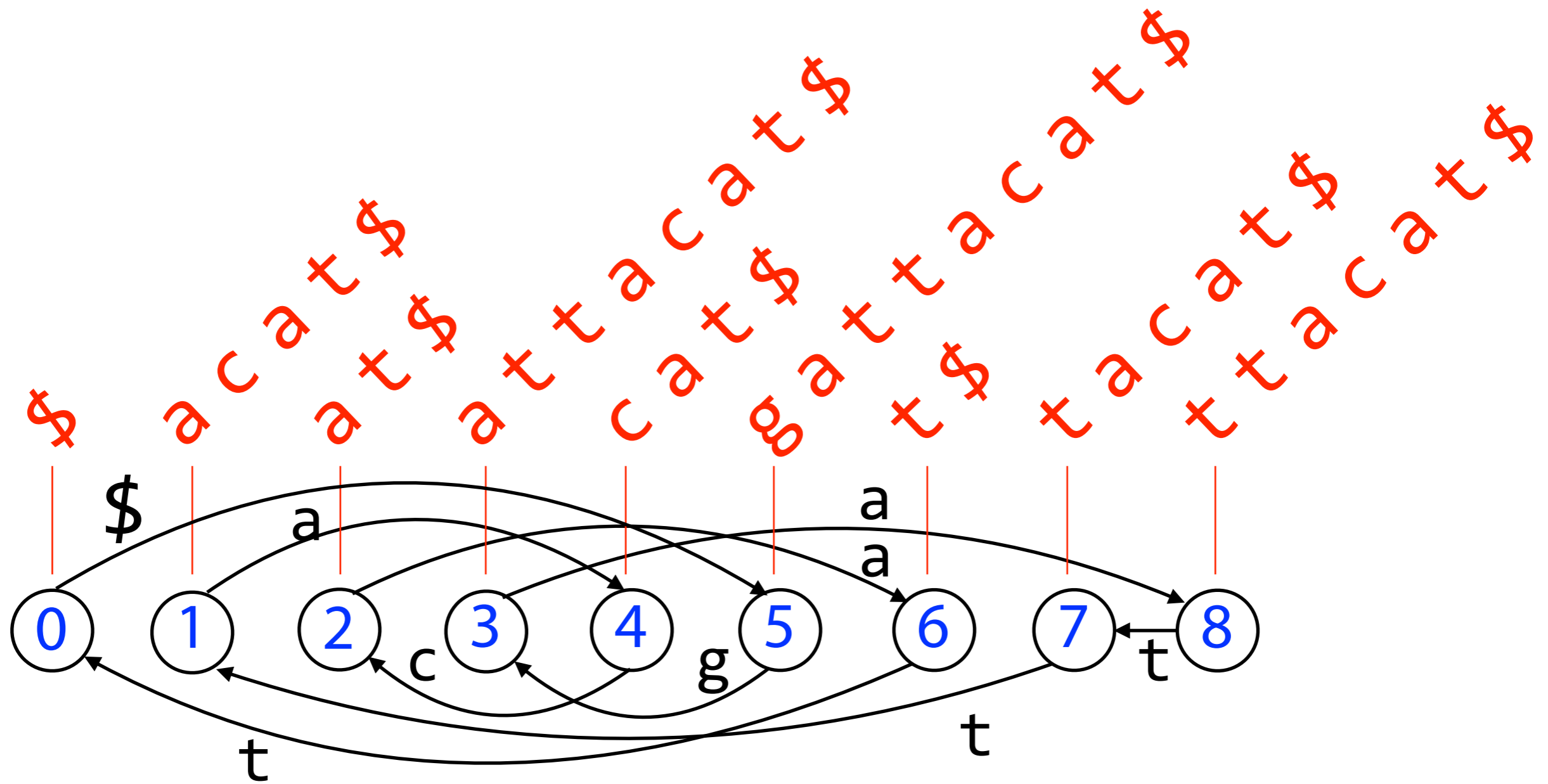
BWT: matching



BWT: matching



BWT: matching



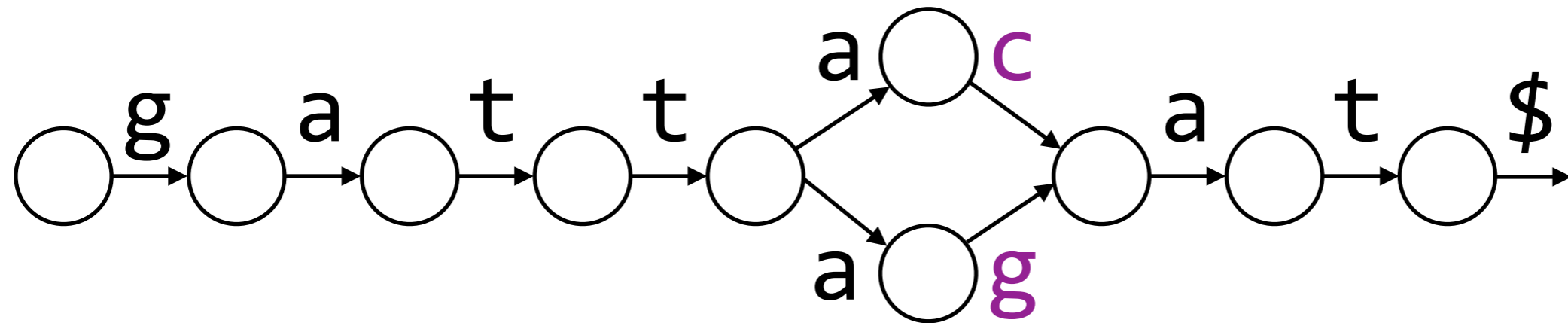
Outgoing paths spell out suffixes/rotations

BWT: matching

Can we go beyond straight-line graphs?

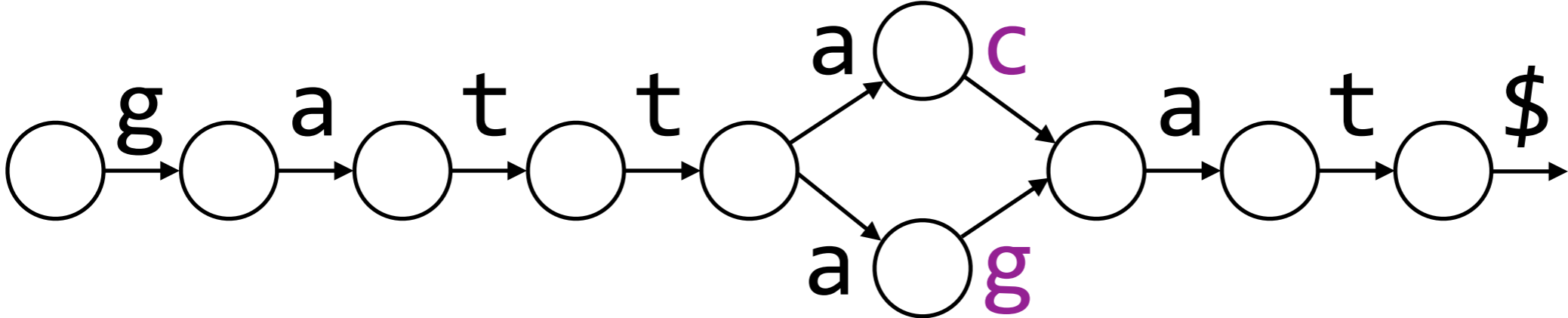
BWT: matching

What does this mean?



BWT: matching

What does this mean?



g a t t a c a t \$

or

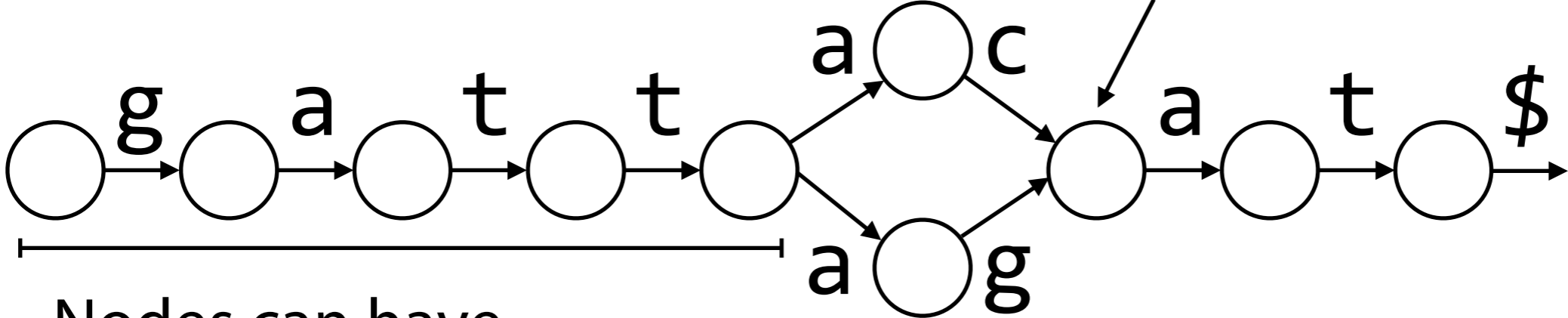
g a t t a g a t \$

BWT: matching

Does our way of thinking about nodes still hold?

No:

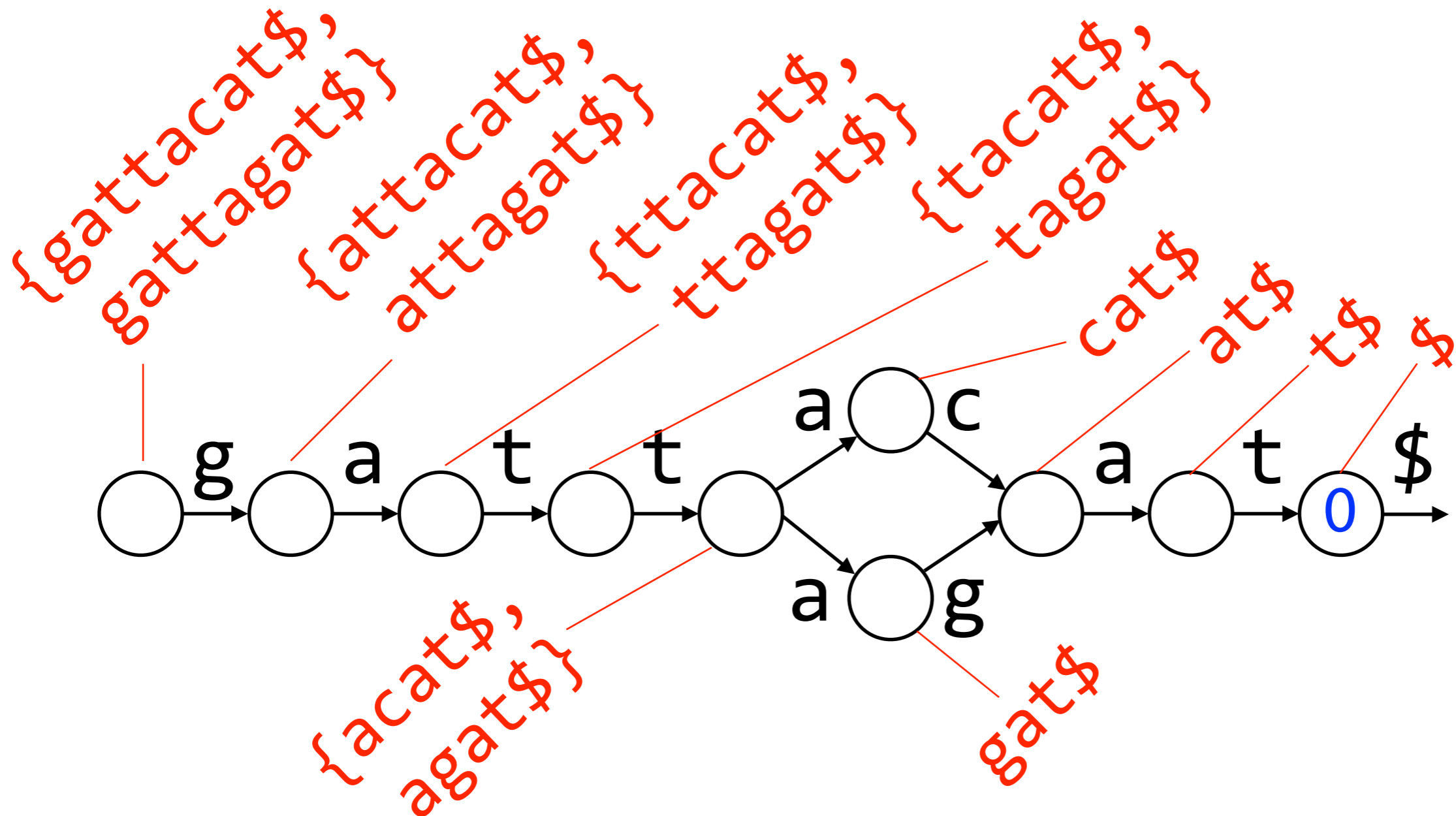
Nodes can have multiple predecessors



Nodes can have multiple suffixes leading out from them

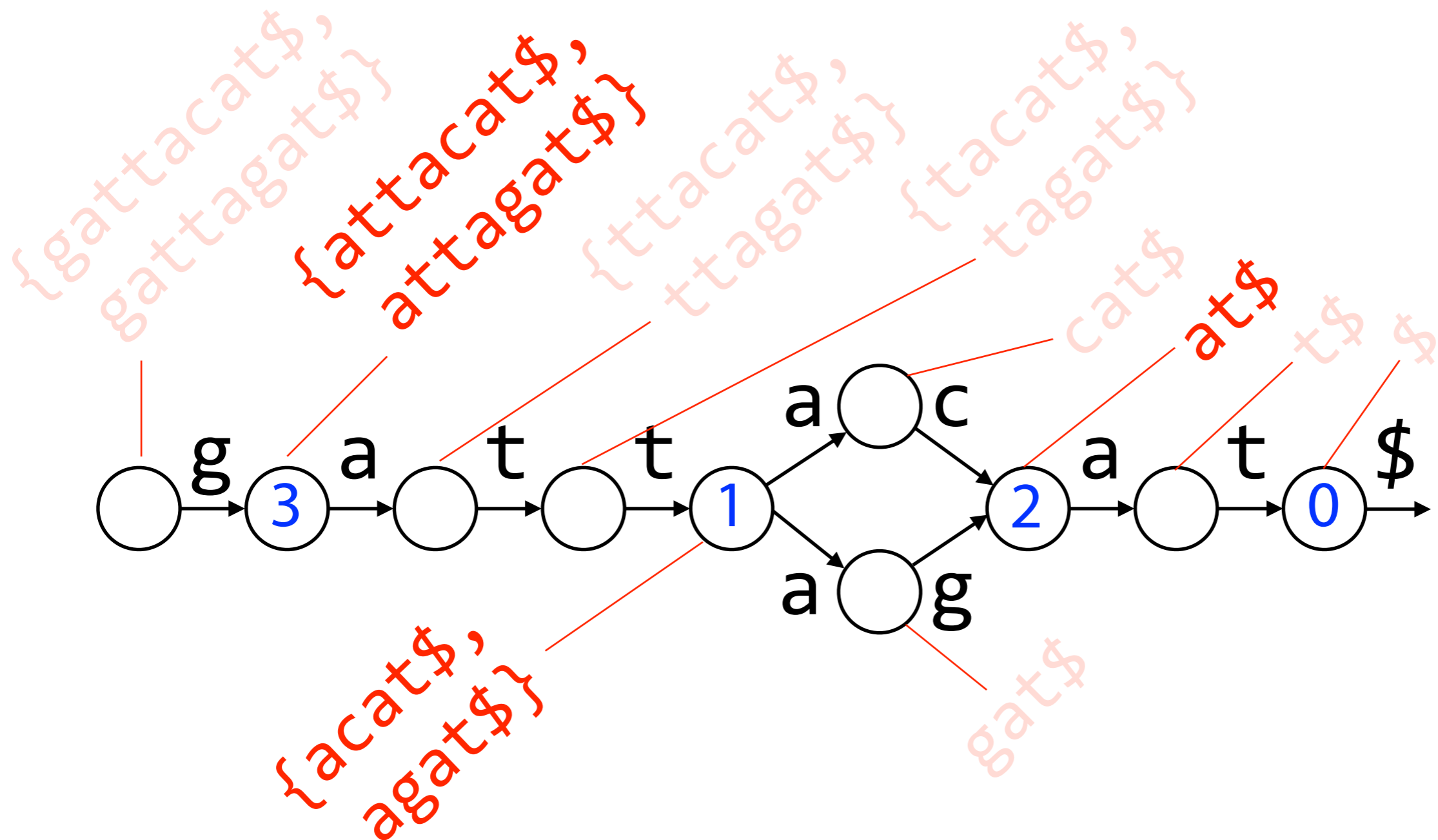
BWT: matching

Can we preserve a total order over outgoing suffixes, even when there's >1 per node?



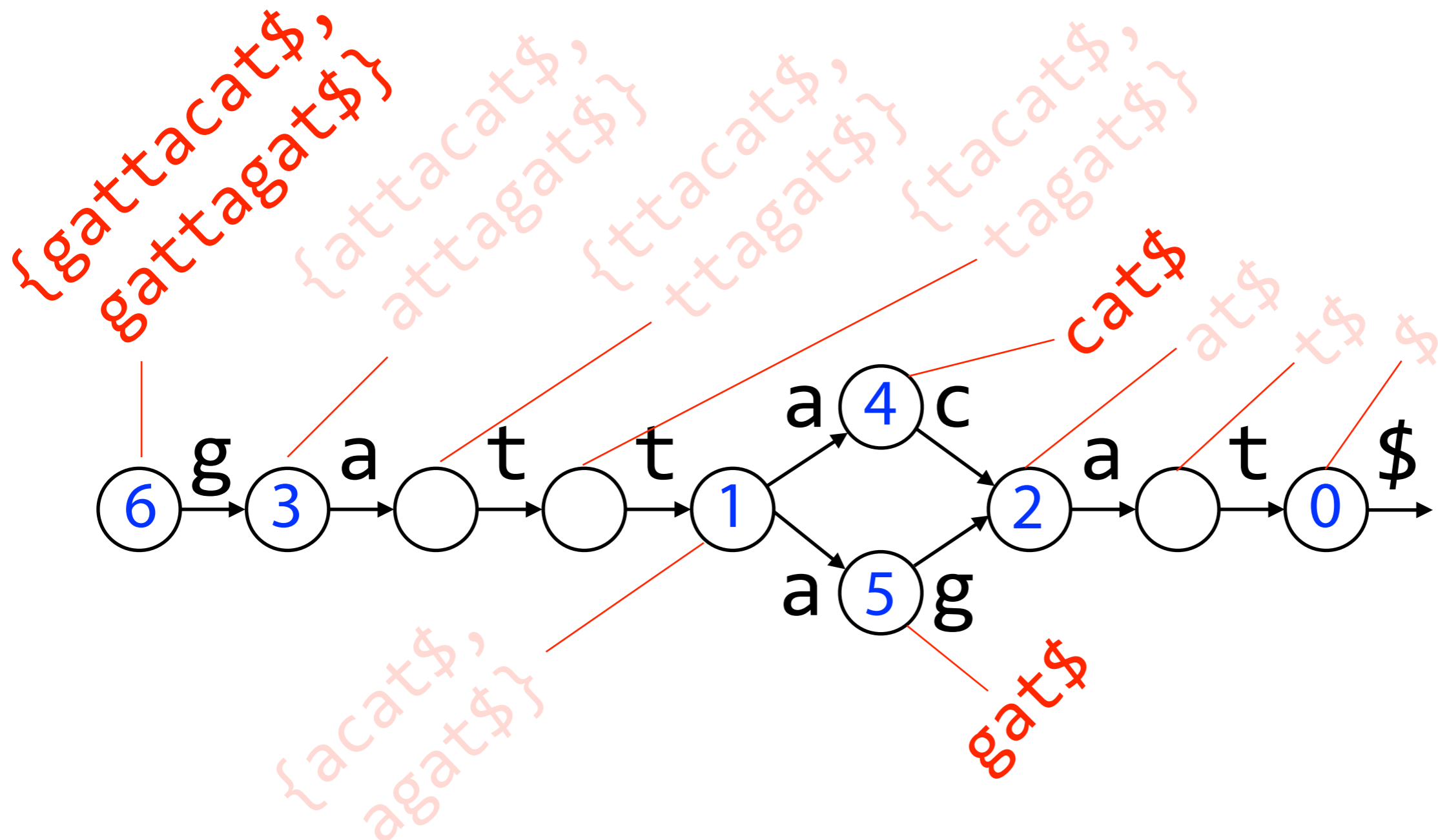
BWT: matching

Can we preserve a total order over outgoing suffixes, even when there's >1 per node?



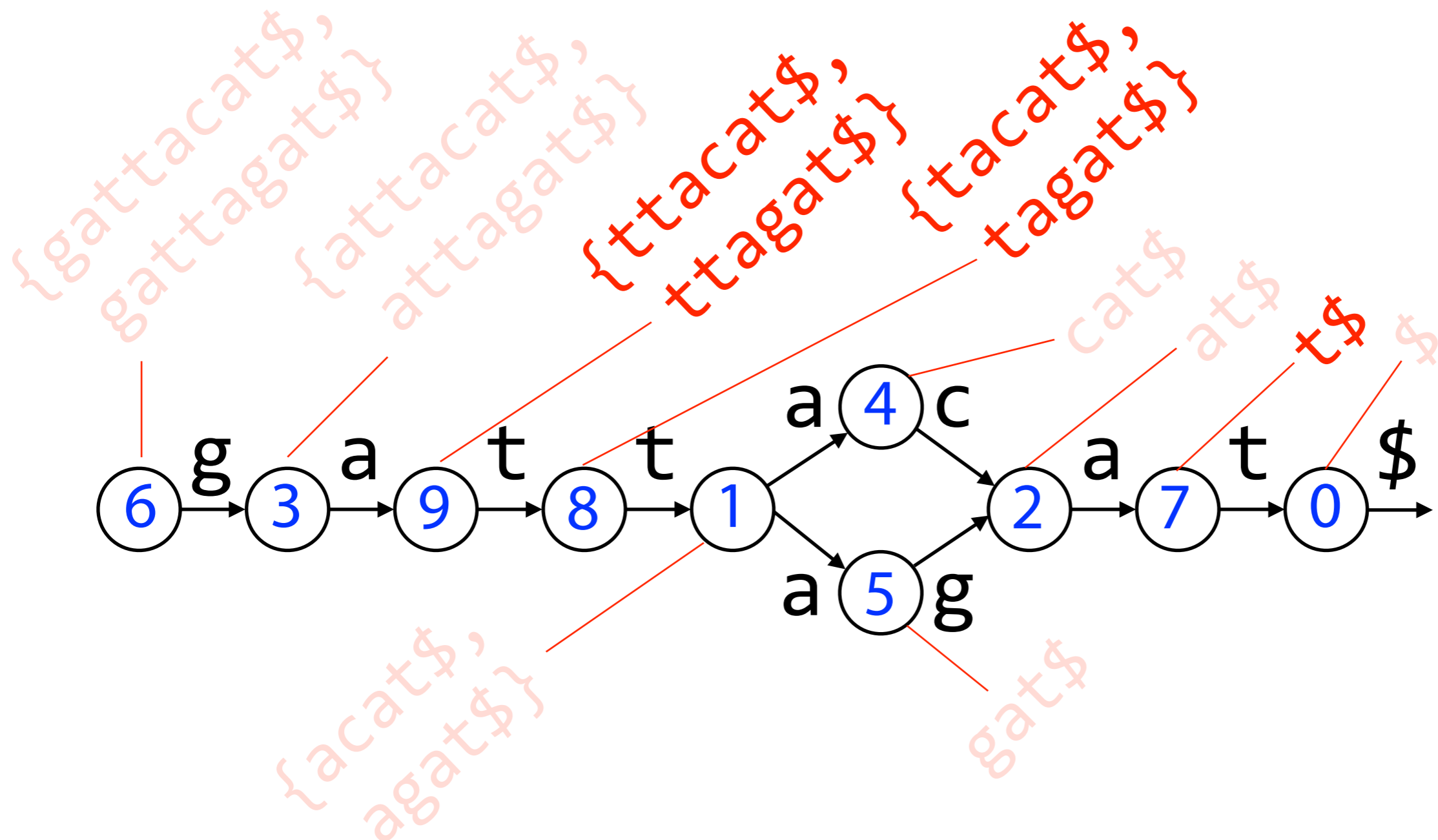
BWT: matching

Can we preserve a total order over outgoing suffixes, even when there's >1 per node?



BWT: matching

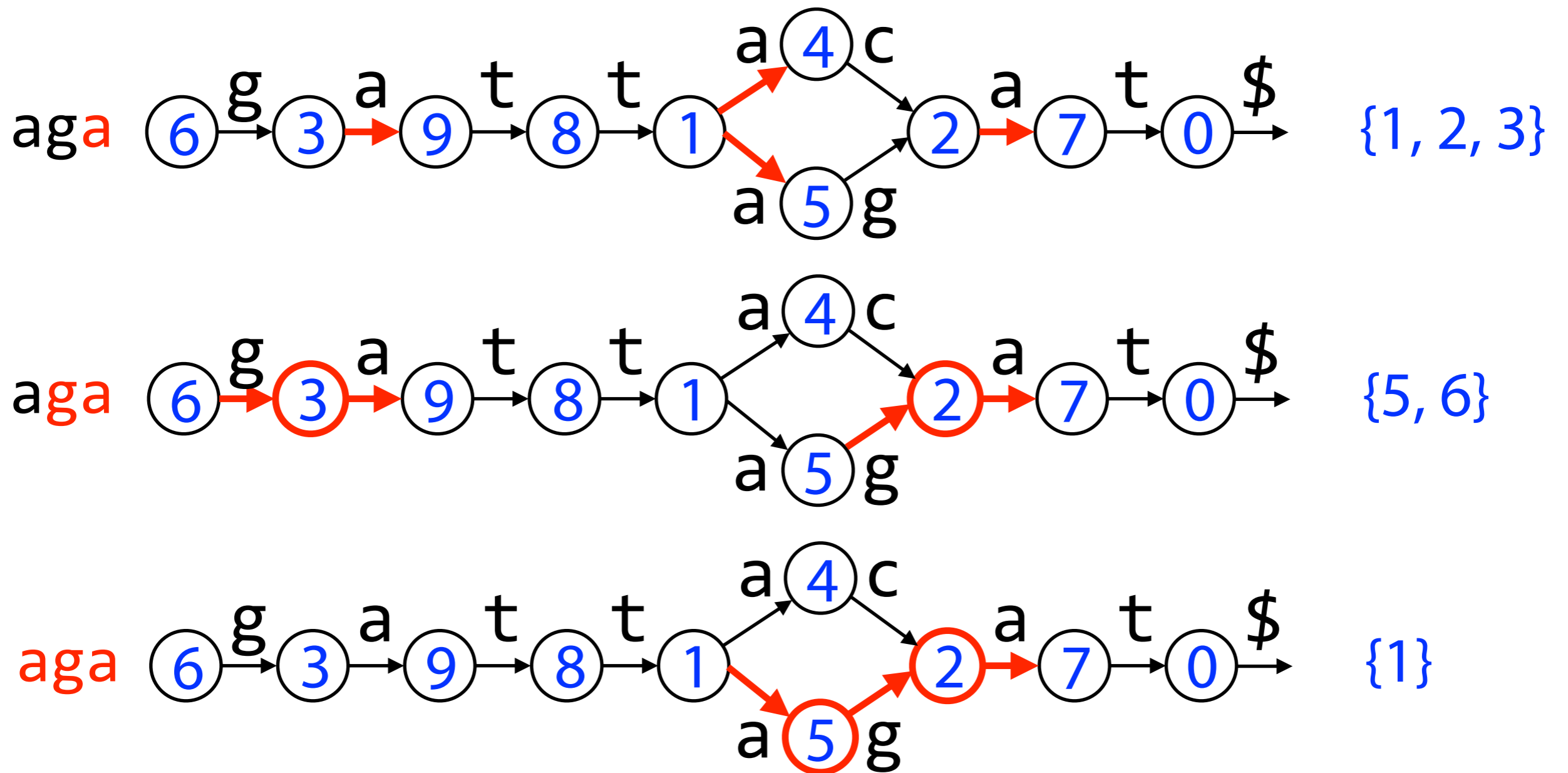
Can we preserve a total order over outgoing suffixes, even when there's >1 per node?



BWT: matching

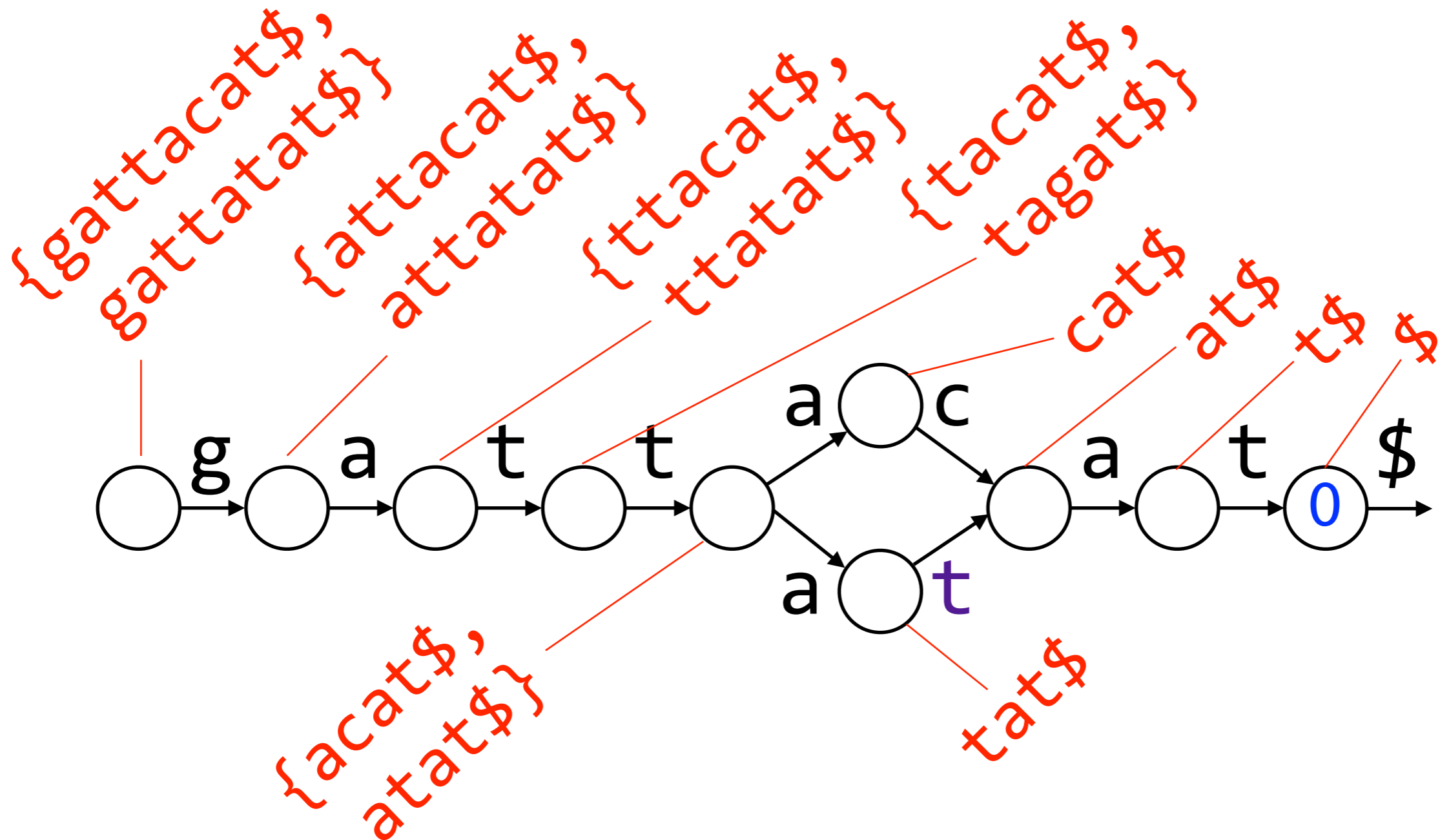
Graph has something like a BW order!

Matching aga, we still have consecutivity.



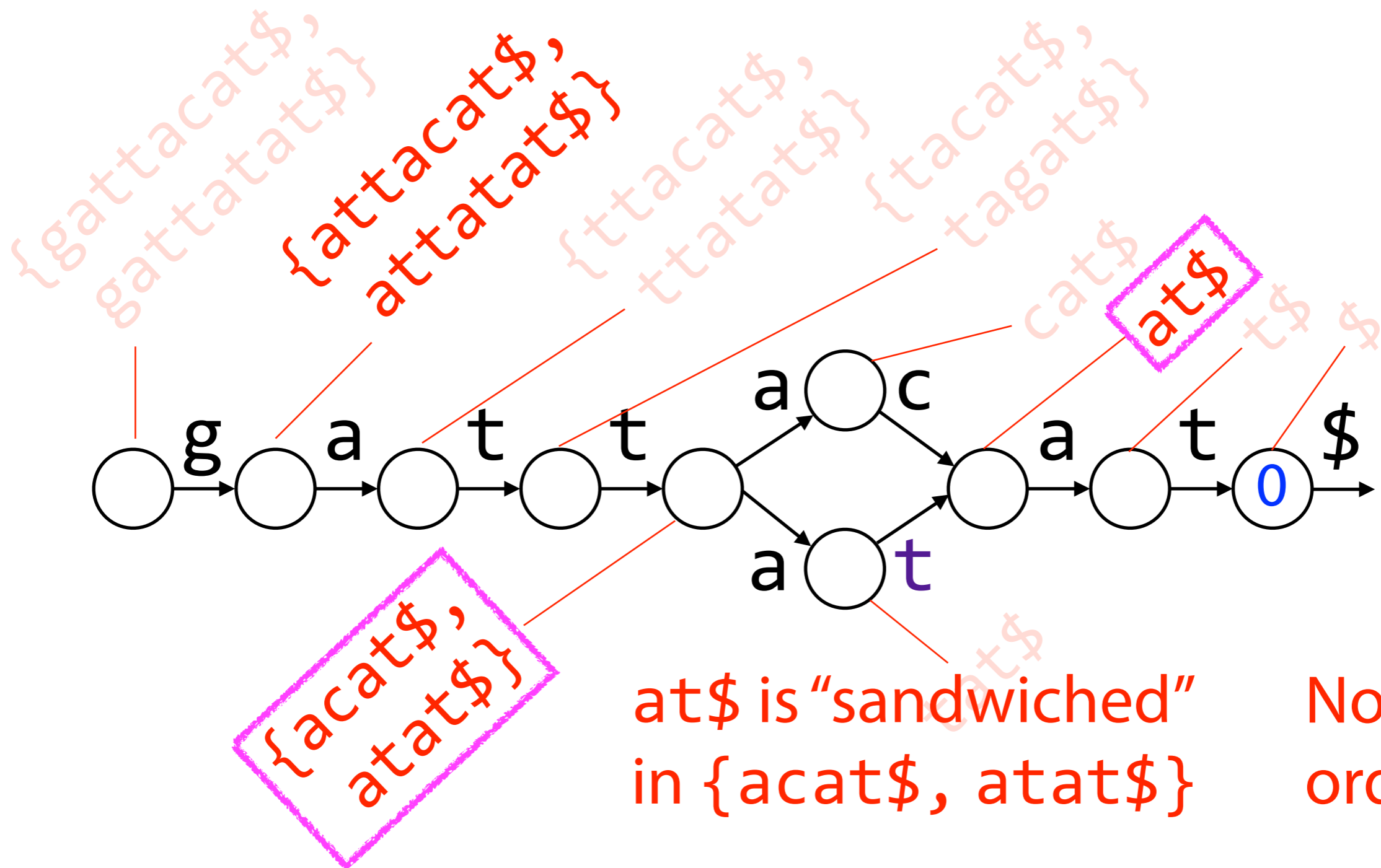
BWT: matching

Does it work for every graph?



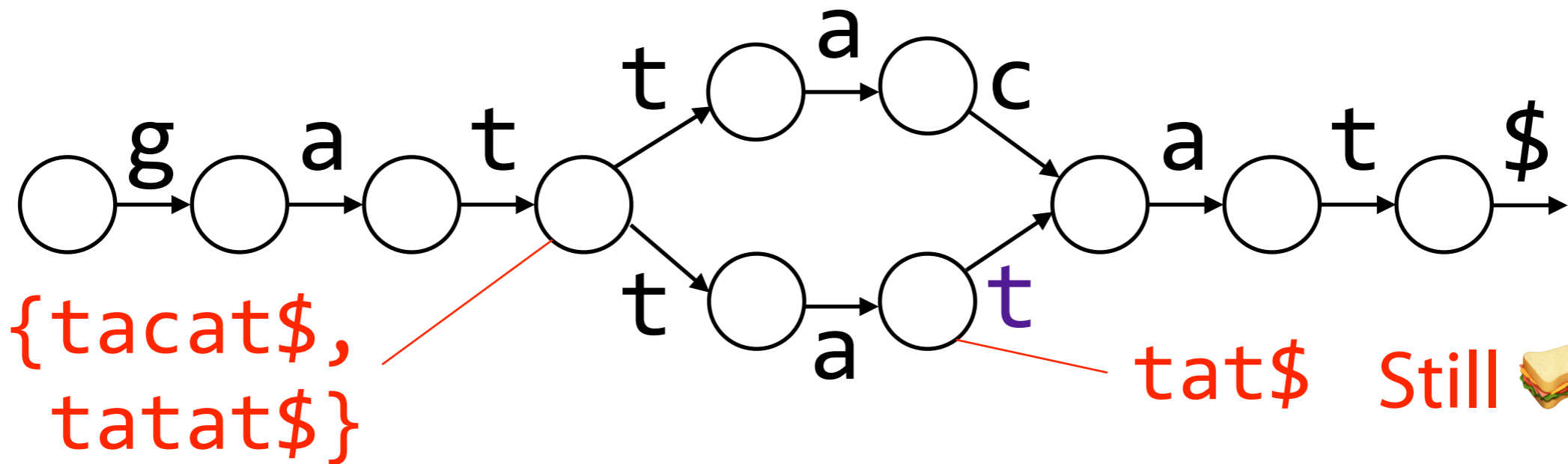
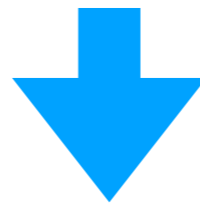
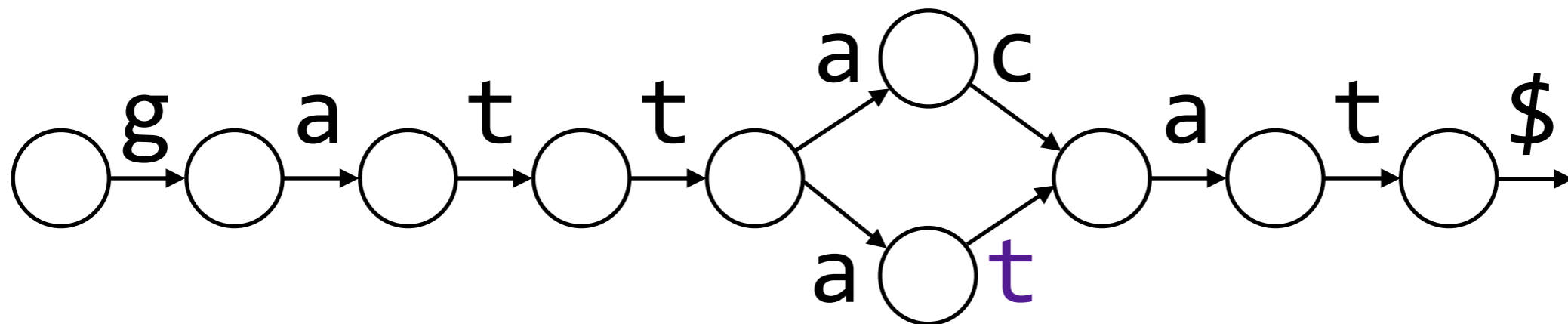
BWT: matching

Does it work for every graph?



BWT: matching

Can I fix it?

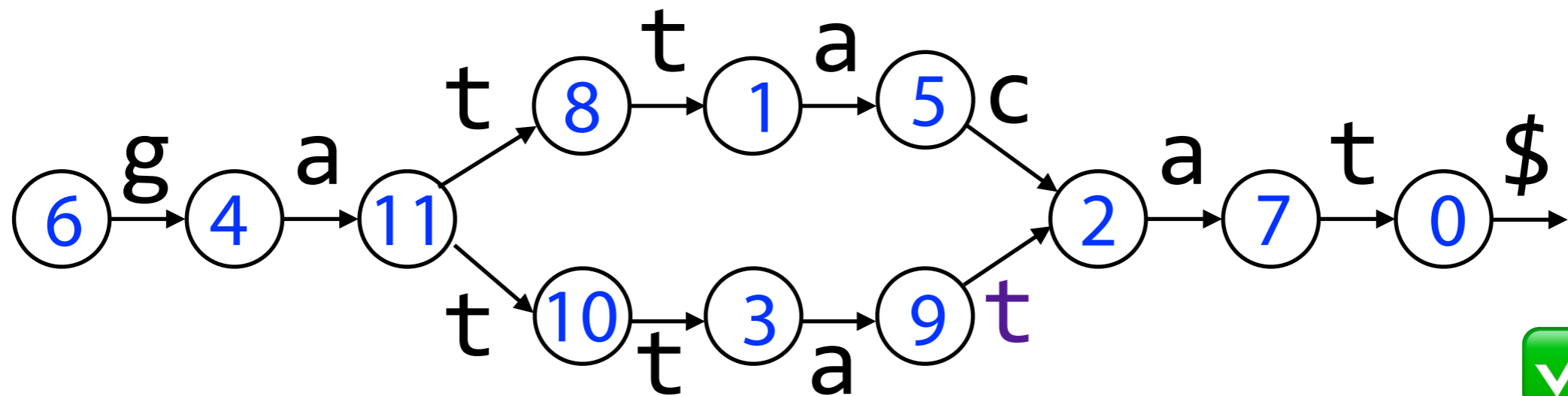
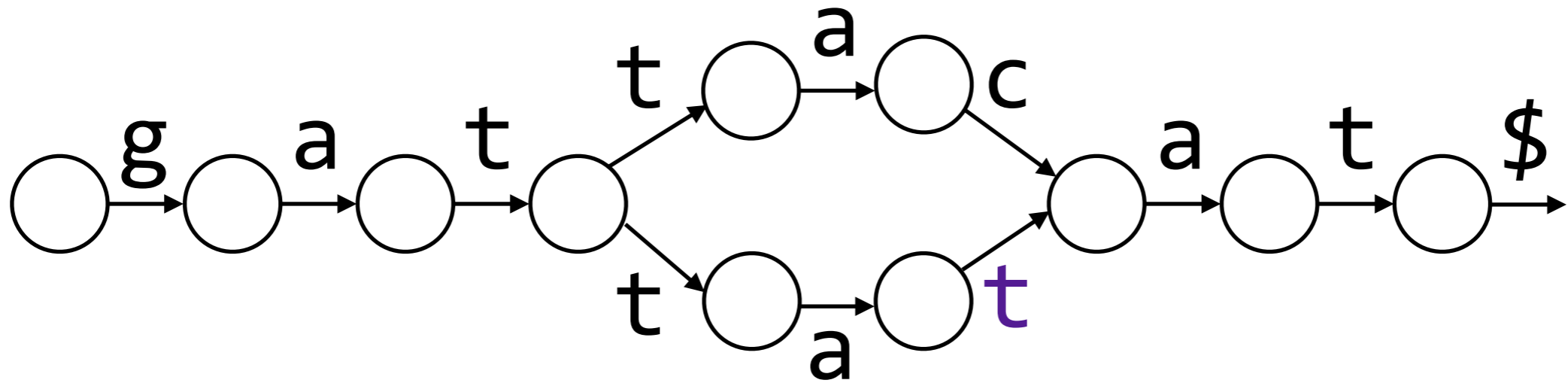


{tacat\$,
tatat\$}

tat\$ Still 🍔ed

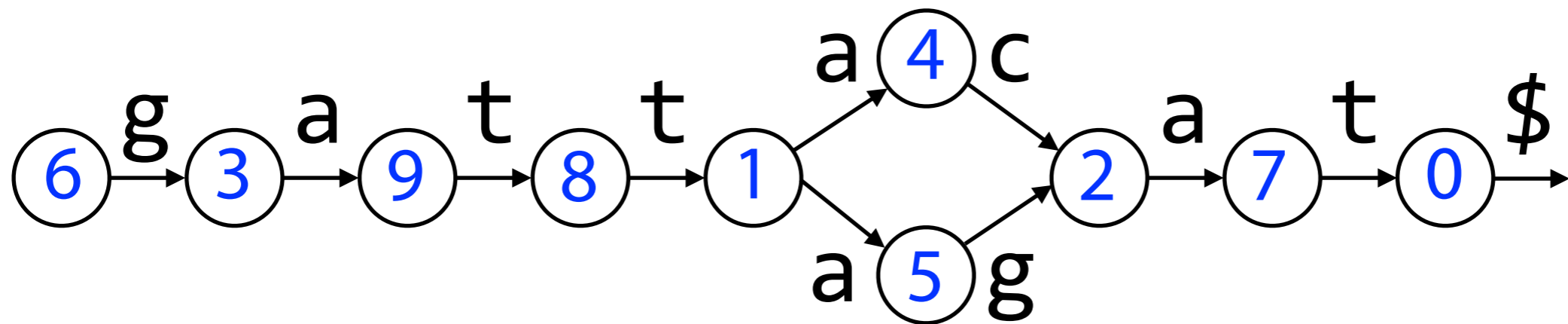
BWT: matching

Can I fix it?

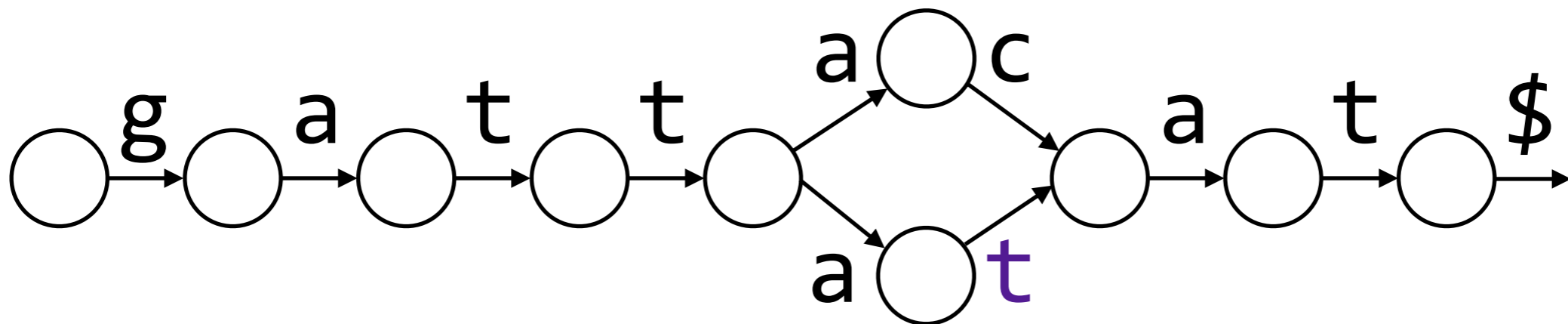


BWT: matching

For some graphs, total order exists



For others, not (but we can "fix" them sometimes)



BWT: matching

Questions:

Which graphs does it work for?

Do these graphs provably have the desired consecutivity property, so we can do matching?

How do we represent and query the graph?

Wheeler graphs



Contents lists available at [ScienceDirect](#)

Theoretical Computer Science

www.elsevier.com/locate/tcs



Wheeler graphs: A framework for BWT-based data structures [☆]

Travis Gagie ^a, Giovanni Manzini ^{b,c,*}, Jouni Sirén ^d

^a *Diego Portales University and CEBIB, Santiago, Chile*

^b *University of Eastern Piedmont, Alessandria, Italy*

^c *IIT-CNR, Pisa, Italy*

^d *Wellcome Trust Sanger Institute, Cambridge, UK*

<http://dx.doi.org/10.1016/j.tcs.2017.06.016>



June 10 – 14 , 2019, Dagstuhl Seminar 19241

25 Years of the Burrows-Wheeler Transform

Organizers

Travis Gagie (Universidad Diego Portales, CL)

Giovanni Manzini (University of Eastern Piedmont – Alessandria, IT)

Gonzalo Navarro (University of Chile – Santiago de Chile, CL)

Jens Stoye (Universität Bielefeld, DE)



Wheeler graphs

An edge-labeled directed multigraph G is a **Wheeler Graph** if nodes can be ordered such that:

1. 0 in-degree nodes come before others
2. For all pairs of edges $e = (u, v)$, $e' = (u', v')$ labeled a, a' respectively, we have:

$$a < a' \implies v < v',$$

$$(a = a') \wedge (u < u') \implies v \leq v'.$$

$<$ alphabetical, $<$ total order for node labels

Wheeler graphs

An edge-labeled directed multigraph G is a **Wheeler Graph** if nodes can be ordered such that:

1. 0 in-degree nodes come before others
2. For all pairs of edges $e = (u, v)$, $e' = (u', v')$ labeled a, a' respectively, we have:

$$a < a' \implies v < v',$$

$$(a = a') \wedge (u < u') \implies v \leq v'.$$

To prove: (a) produce the ordering and (b) show the properties hold for all pairs of edges

Wheeler graphs

For each pair of edges:

If edges have different labels, the destination of the edge with the smaller label must come before the destination of the edge with the larger label

$$a < a' \implies v < v',$$

$$(a = a') \wedge (u < u') \implies v \leq v'.$$

Consequence:
node cannot have
2 incoming edges
with different
labels

If edges have same label but different sources, destination of the lower-source edge must not come after the destination of the higher-source edge

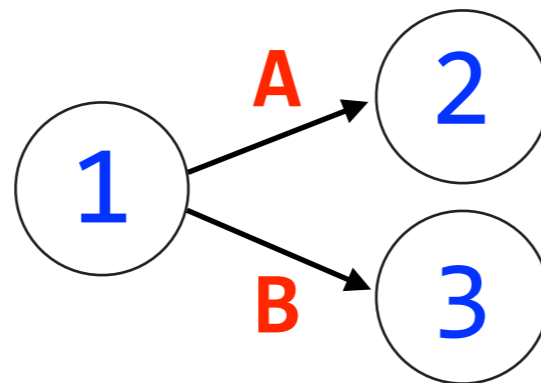
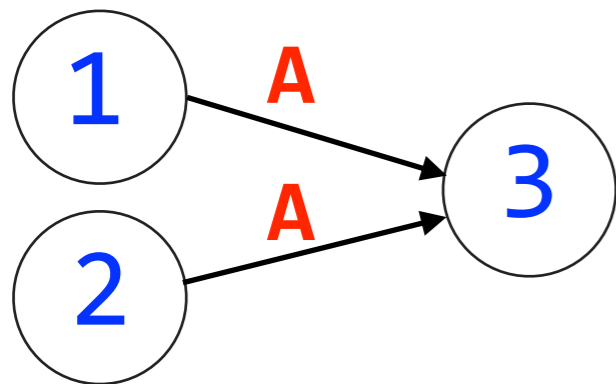
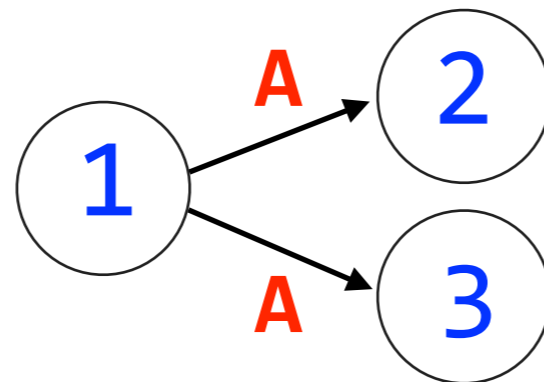
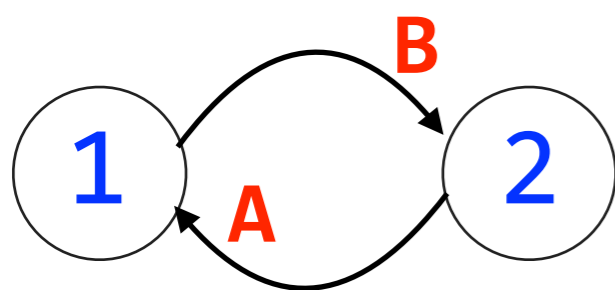
0 in-degree nodes come before others (1)

For all pairs
of edges $\left[\begin{array}{l} a < a' \implies v < v' \quad (2) \\ (a = a') \wedge (u < u') \implies v \leq v' \quad (3) \end{array} \right.$

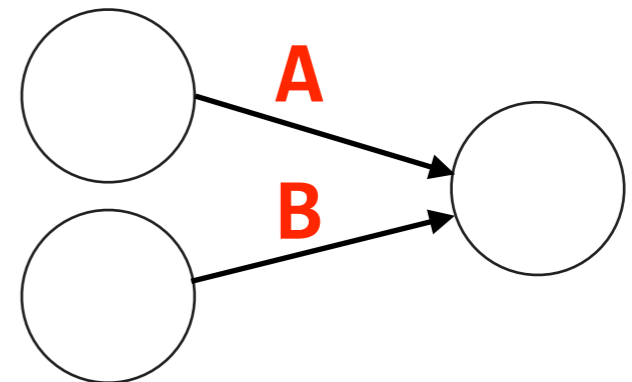
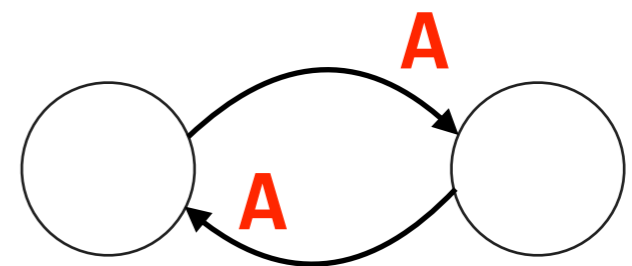
0 in-degree nodes come before others (1)

For all pairs of edges $\left[\begin{array}{l} a < a' \implies v < v' \quad (2) \\ (a = a') \wedge (u < u') \implies v \leq v' \quad (3) \end{array} \right.$

Wheeler



Not Wheeler

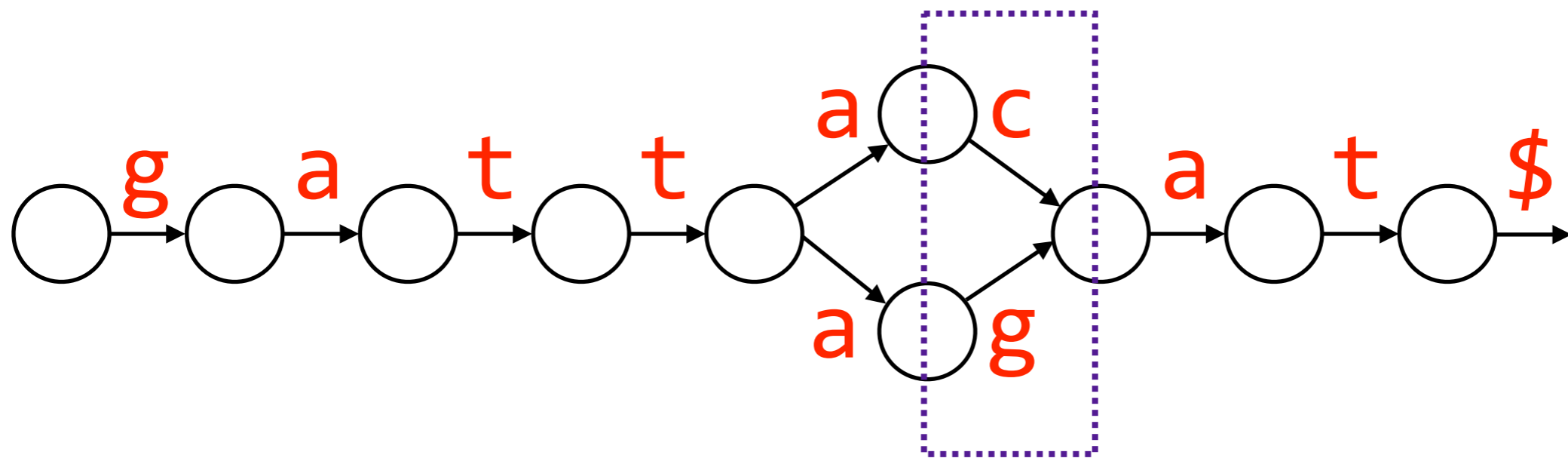


Wheeler graphs

0 in-degree nodes come before others (1)

For all pairs of edges $\left[\begin{array}{l} a < a' \implies v < v' \quad (2) \\ (a = a') \wedge (u < u') \implies v \leq v' \quad (3) \end{array} \right.$

Is this a Wheeler Graph? **No**



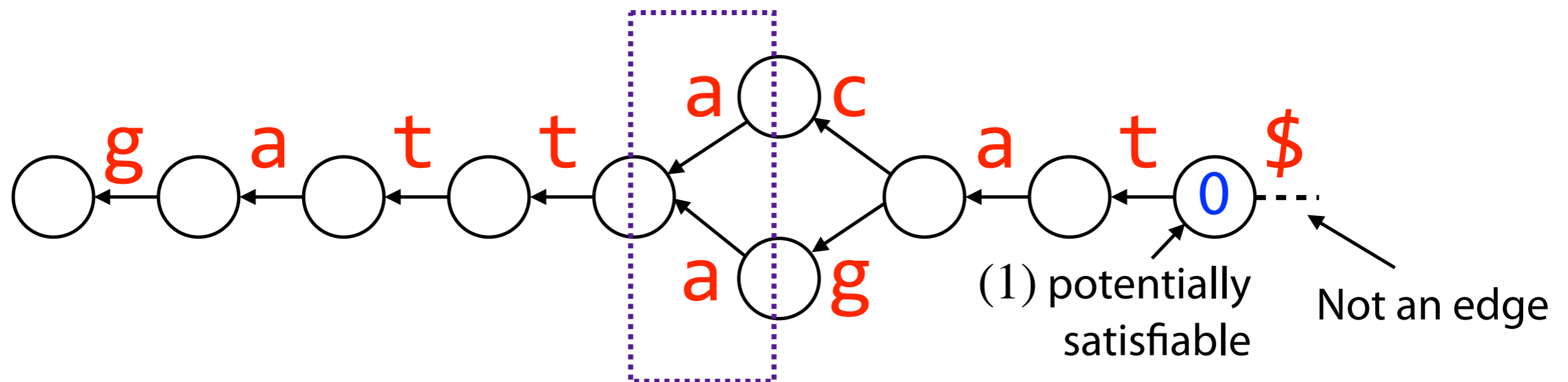
$a < a'$ but $v = v'$ (2) cannot hold

Wheeler graphs

0 in-degree nodes come before others (1)

For all pairs of edges $\left[\begin{array}{l} a < a' \implies v < v' \quad (2) \\ (a = a') \wedge (u < u') \implies v \leq v' \quad (3) \end{array} \right.$

What if we flip edges to follow the direction of matching?



$a = a'$ and $v = v'$, so (3) is satisfied whether or not $u < u'$

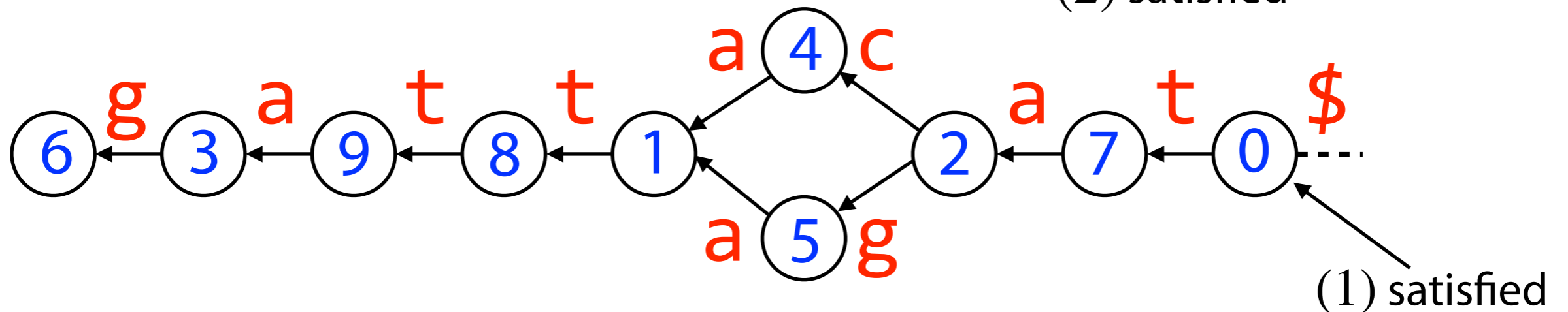
Wheeler graphs

0 in-degree nodes come before others (1)

For all pairs of edges $\left[\begin{array}{l} a < a' \implies v < v' \quad (2) \\ (a = a') \wedge (u < u') \implies v \leq v' \quad (3) \end{array} \right.$

Successors of edges labeled: $a : \{1, 2, 3\}$ $g : \{5, 6\}$
 $c : \{4\}$ $t : \{7, 8, 9\}$

(2) satisfied



Exercise: prove (3) is satisfied for all pairs of edges