



Joint Institute  
of Engineering

SUN YAT-SEN UNIVERSITY

Carnegie Mellon University

# Design and Implementation of Speech Recognition Systems

*Fall 2014*

*Ming Li*

Class 11: Backpointer tables

Nov 6th

*Thanks to Professor Bhiksha Raj for the contribution of the slides*

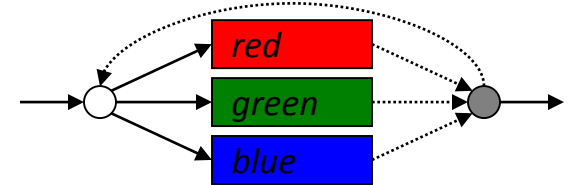
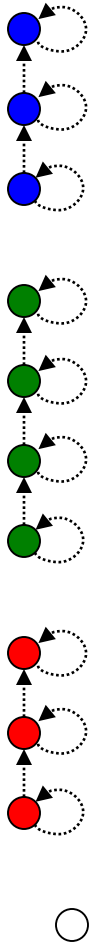
# Backpointers in Viterbi search

---

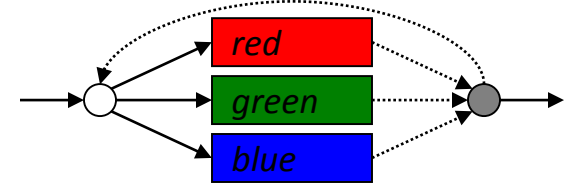
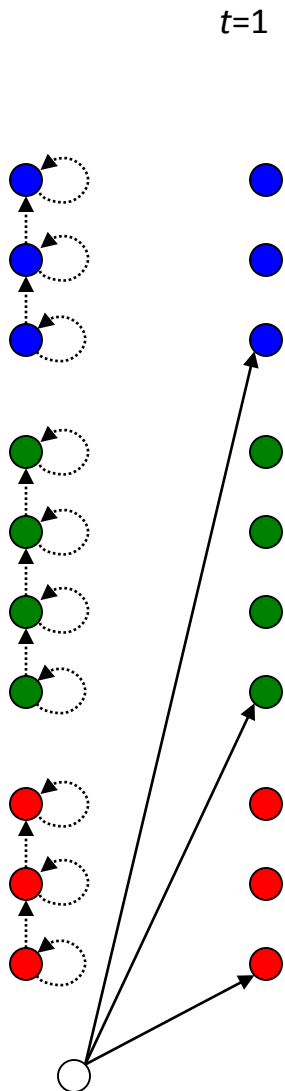
- In Viterbi search, we retain a pointer to the best previous state at all trellis nodes
- This is performed even when we are recognizing continuous speech
  - From Grammars etc.

# Trellis with Complete Set of Backpointers

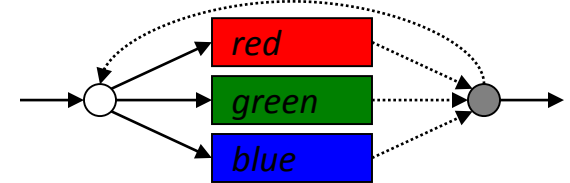
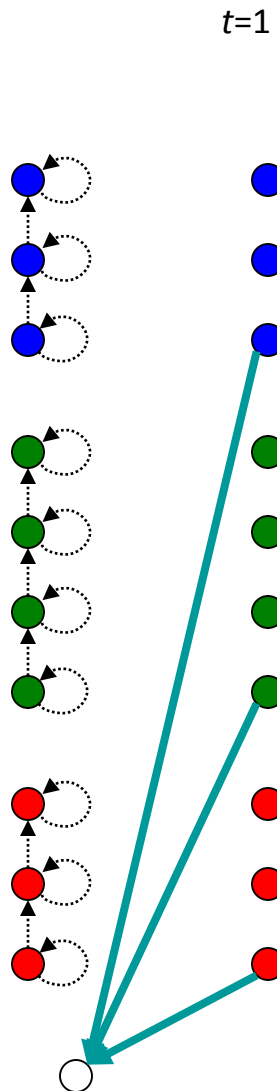
---



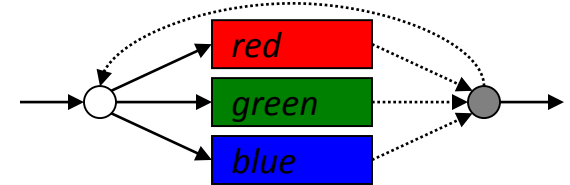
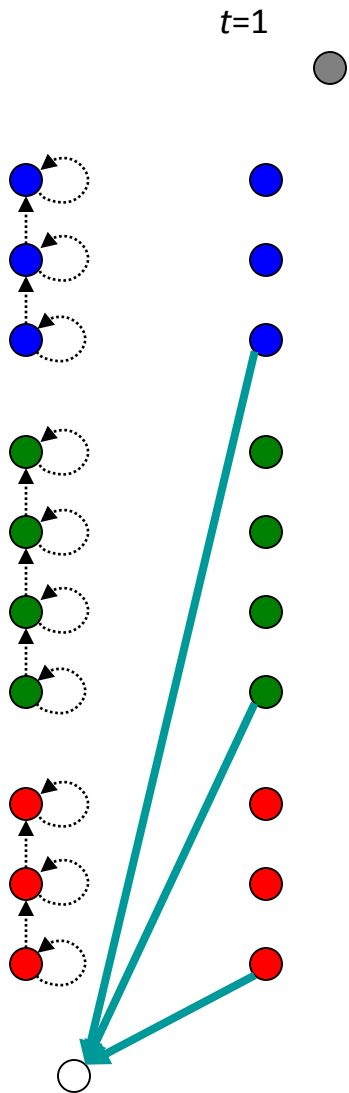
# Trellis with Complete Set of Backpointers



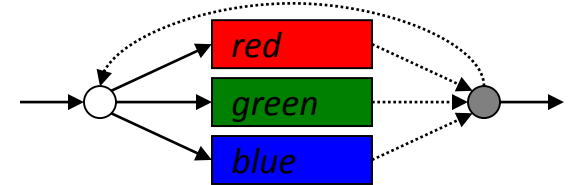
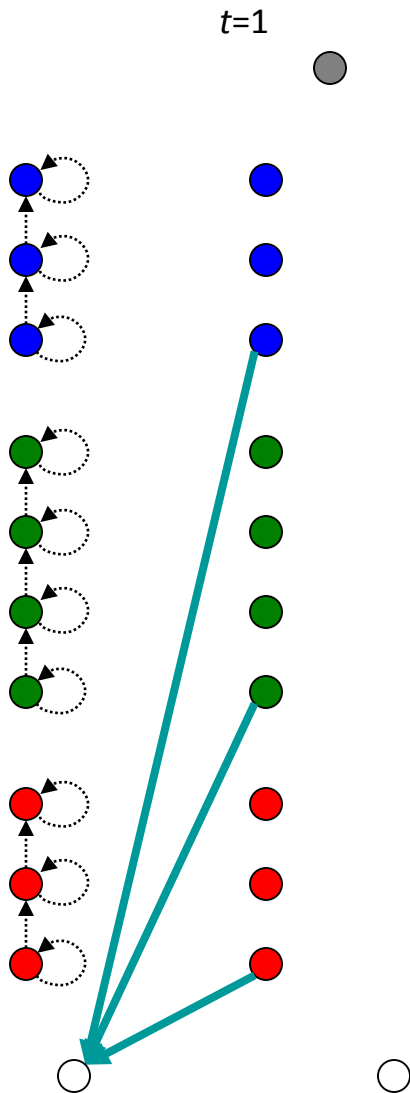
# Trellis with Complete Set of Backpointers



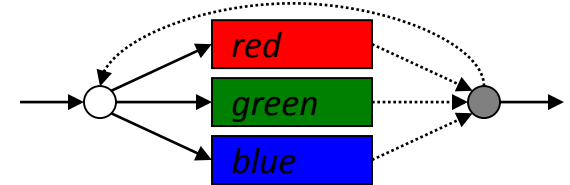
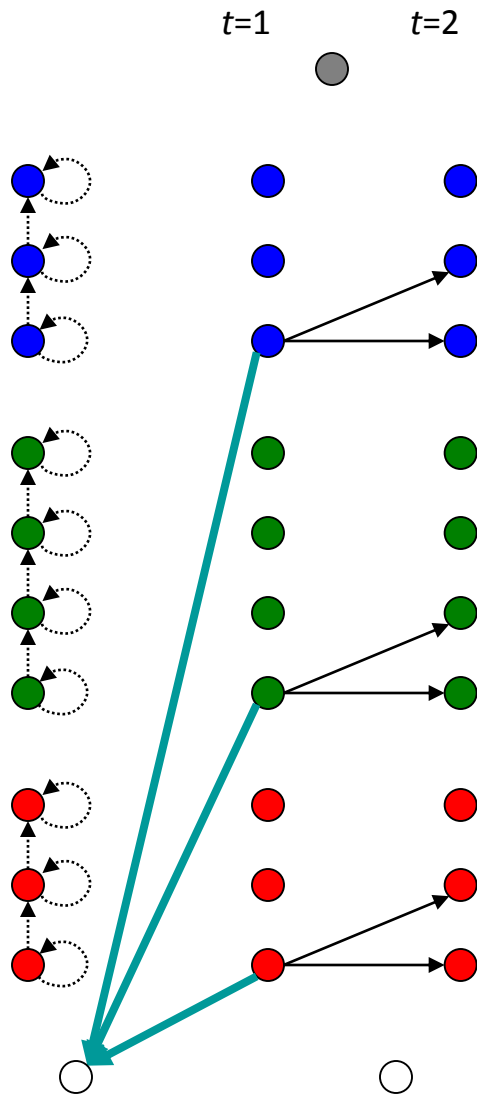
# Trellis with Complete Set of Backpointers



# Trellis with Complete Set of Backpointers

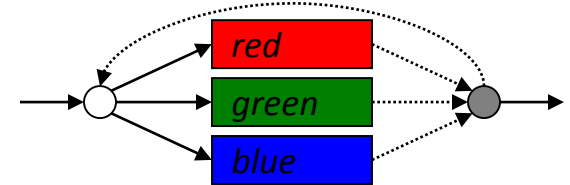
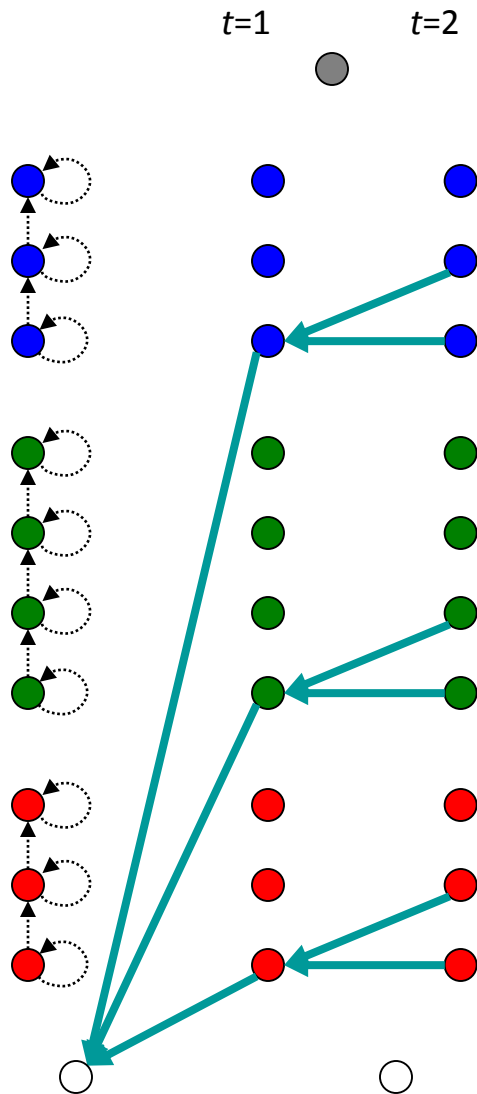


# Trellis with Complete Set of Backpointers

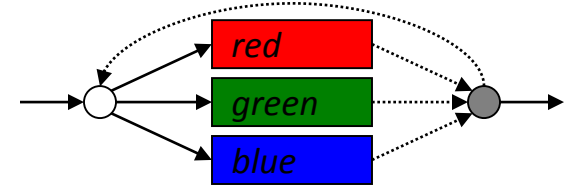
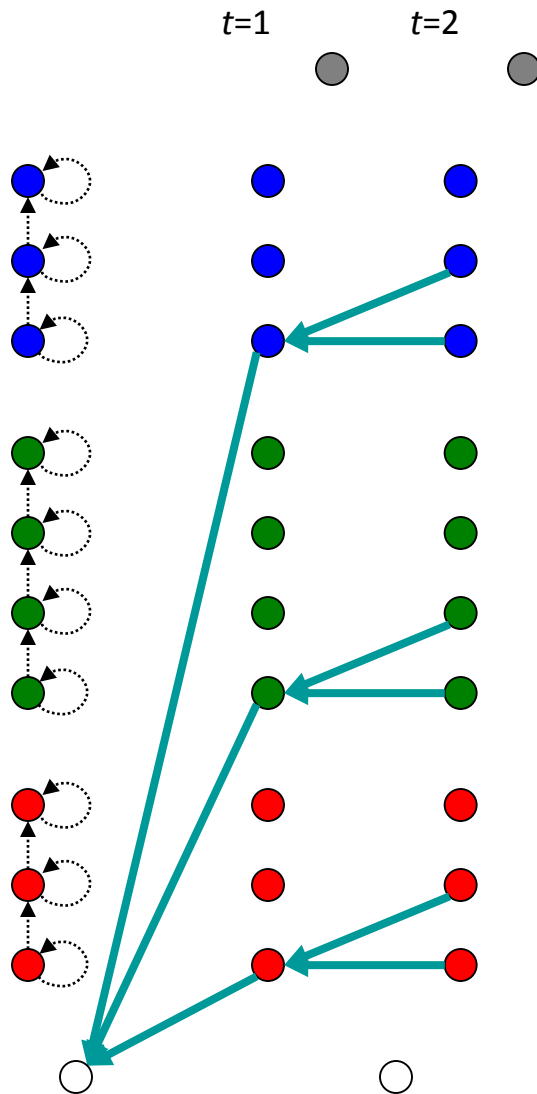




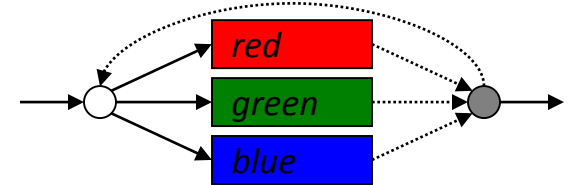
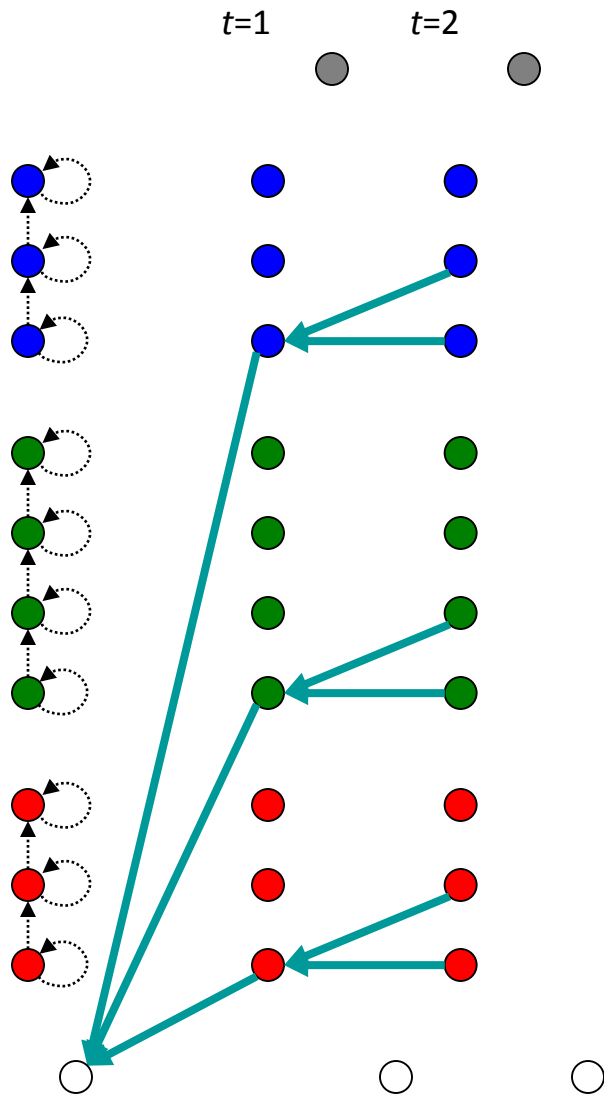
# Trellis with Complete Set of Backpointers



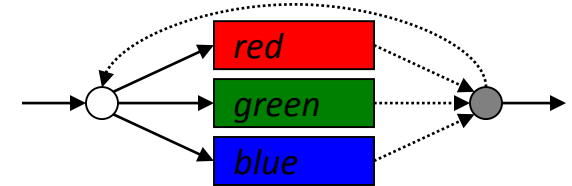
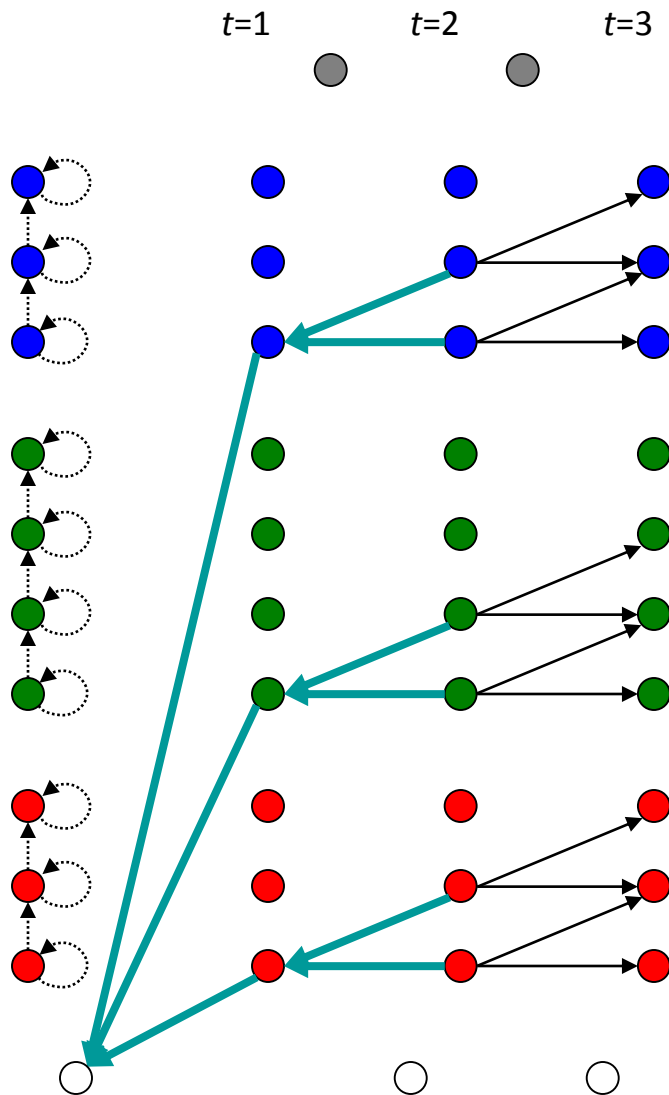
# Trellis with Complete Set of Backpointers



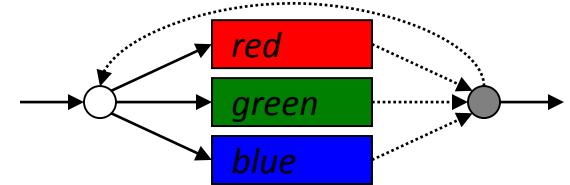
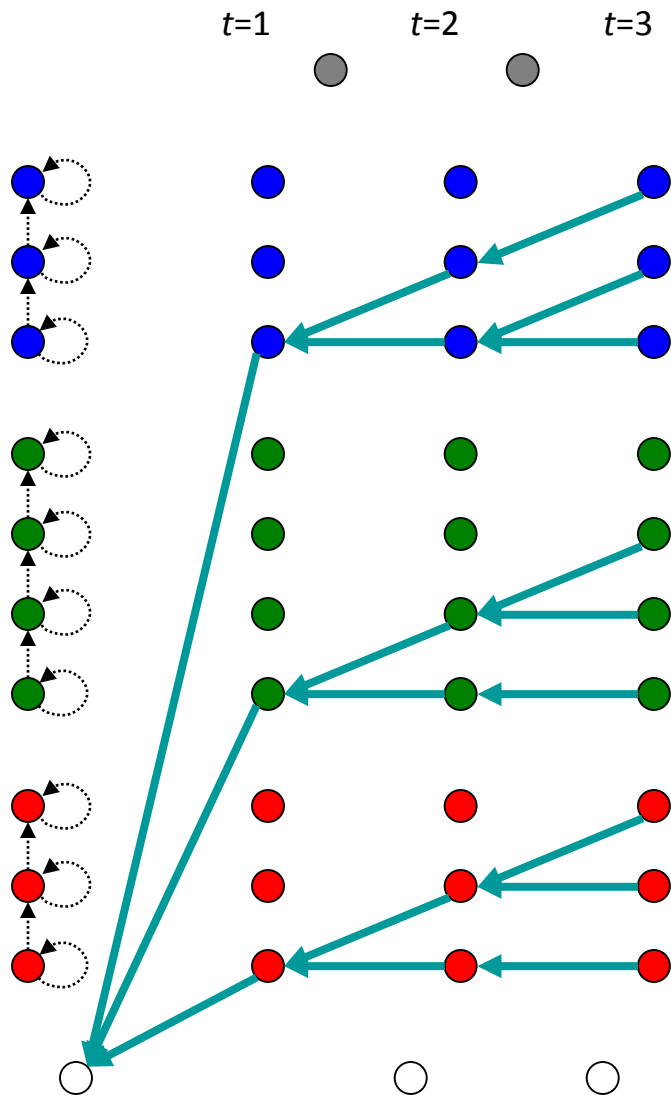
# Trellis with Complete Set of Backpointers



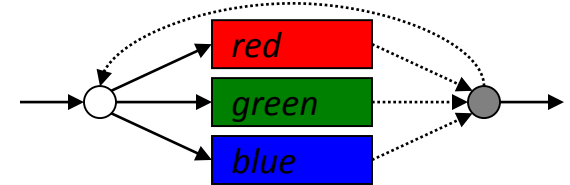
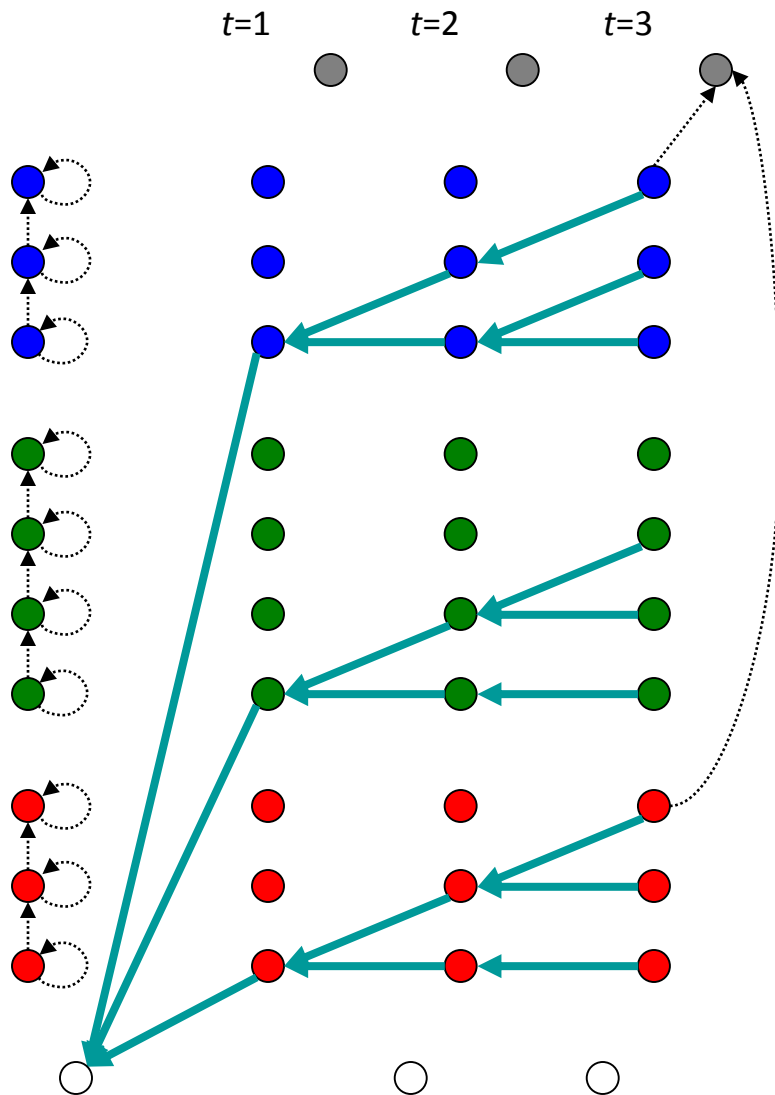
# Trellis with Complete Set of Backpointers



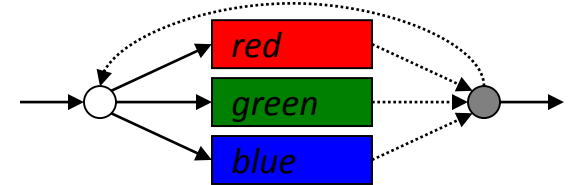
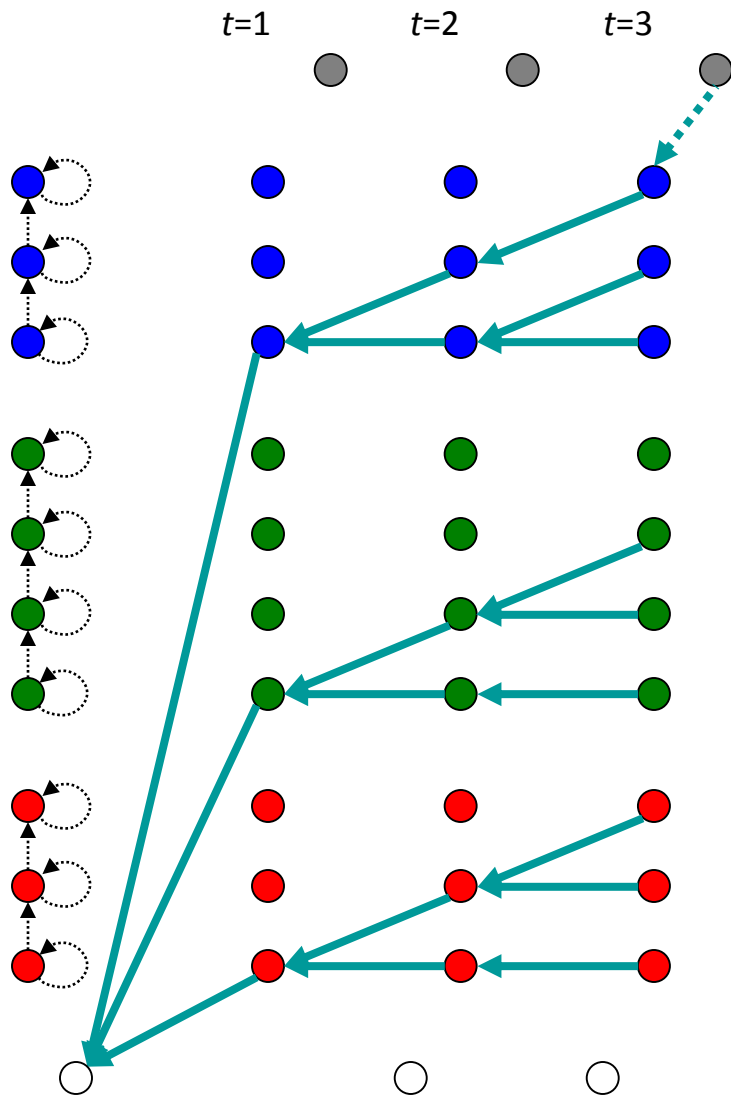
# Trellis with Complete Set of Backpointers



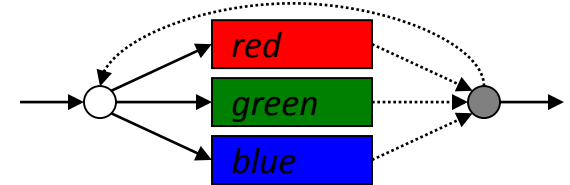
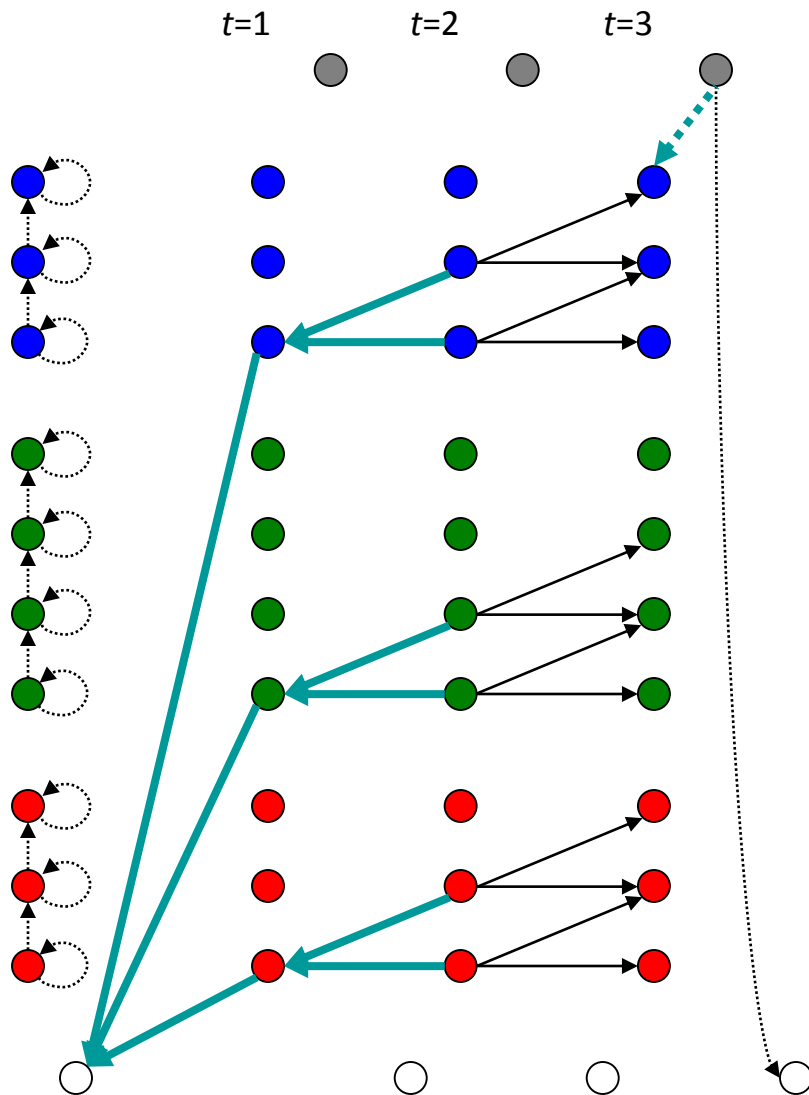
# Trellis with Complete Set of Backpointers



# Trellis with Complete Set of Backpointers

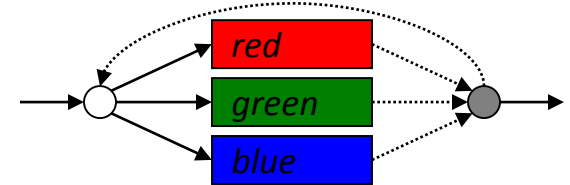
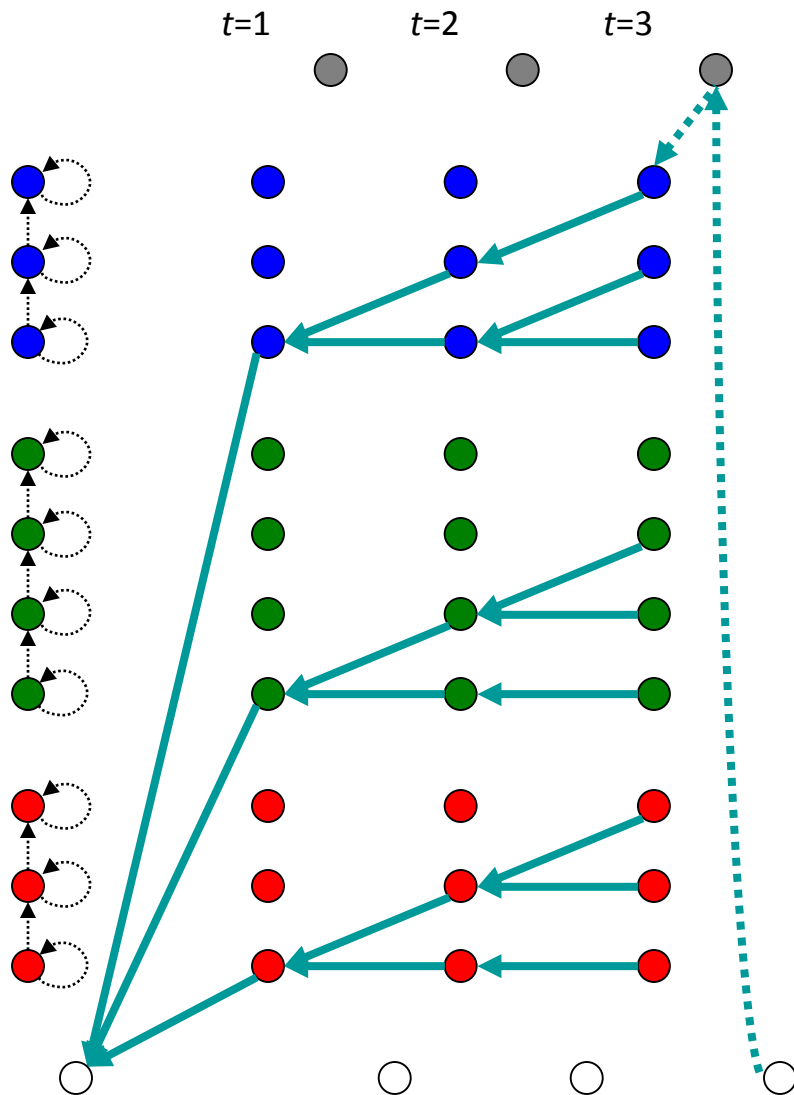


# Trellis with Complete Set of Backpointers

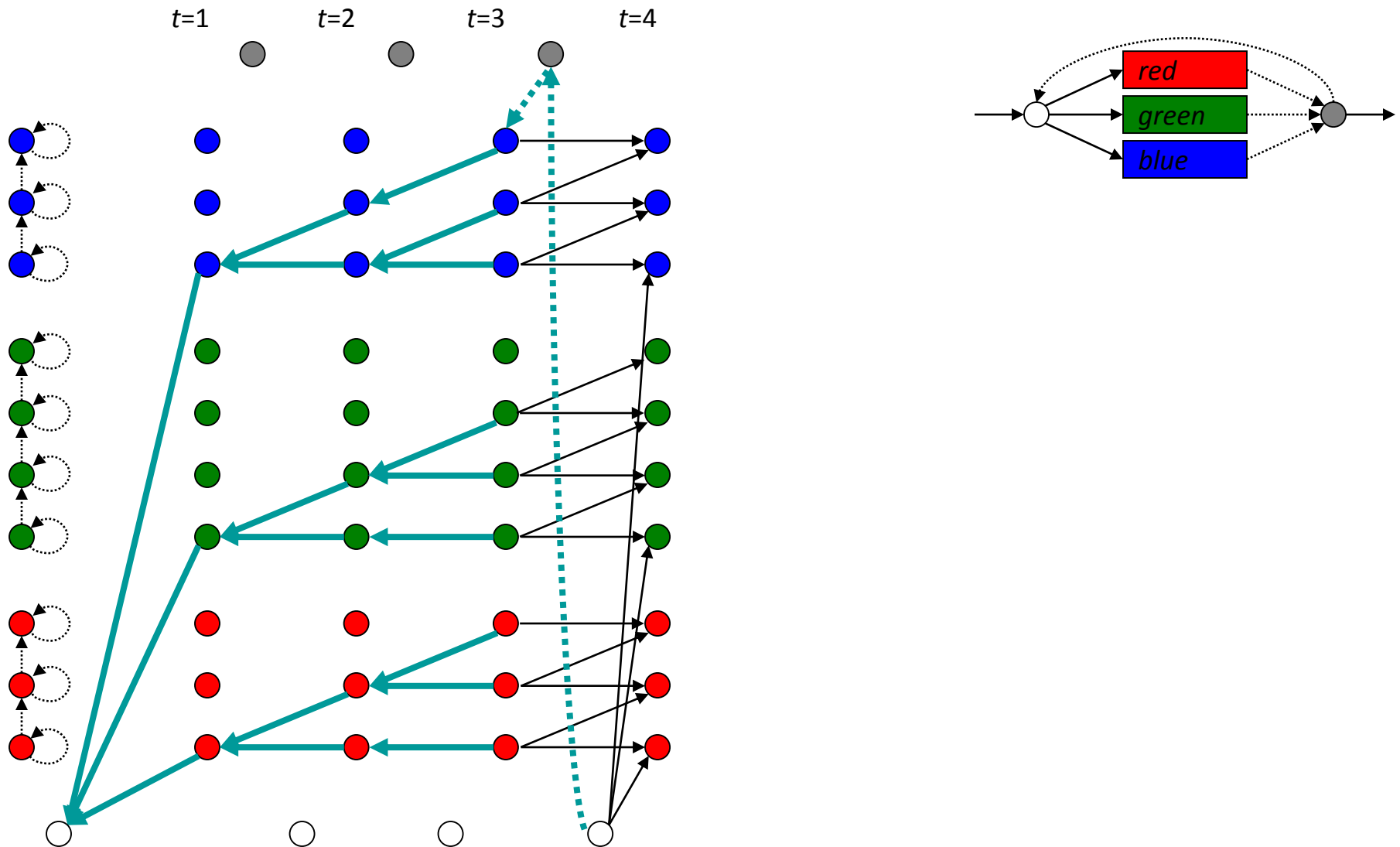




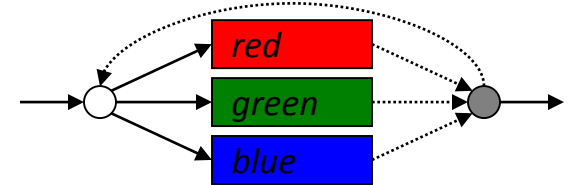
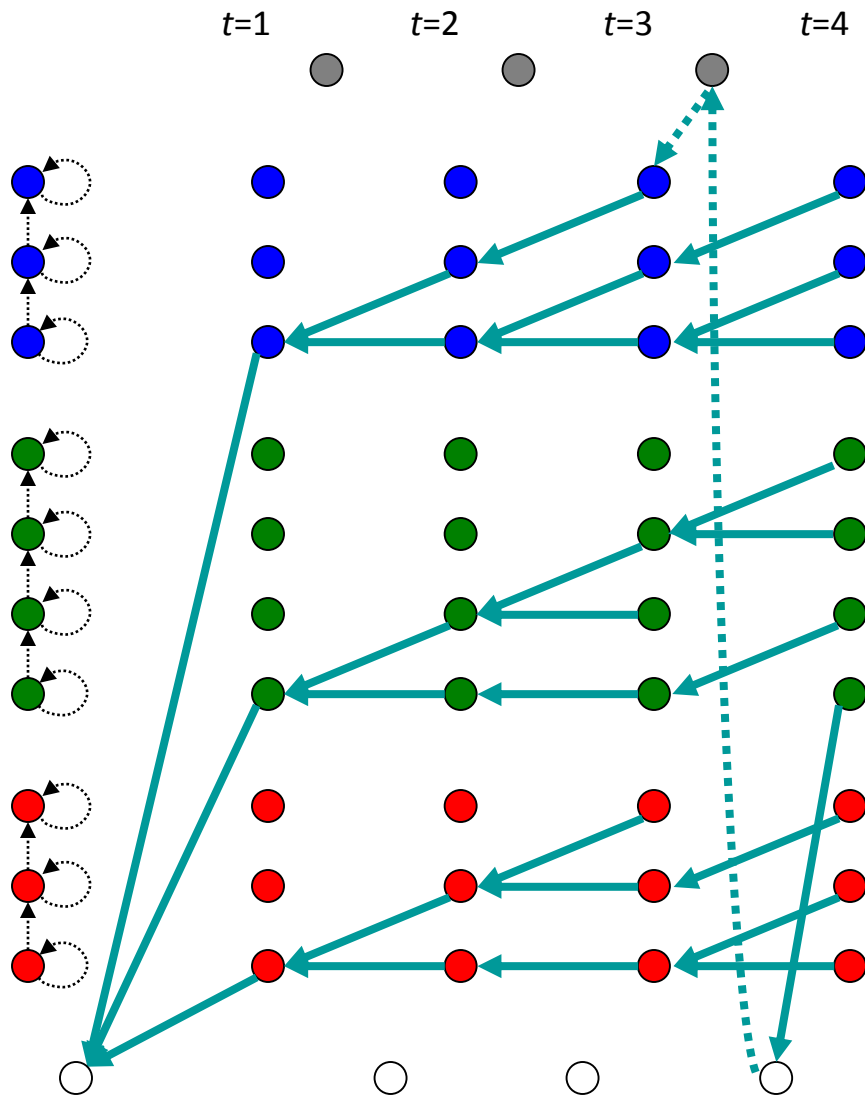
# Trellis with Complete Set of Backpointers



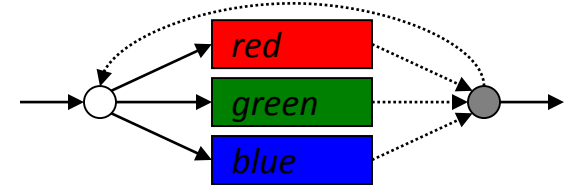
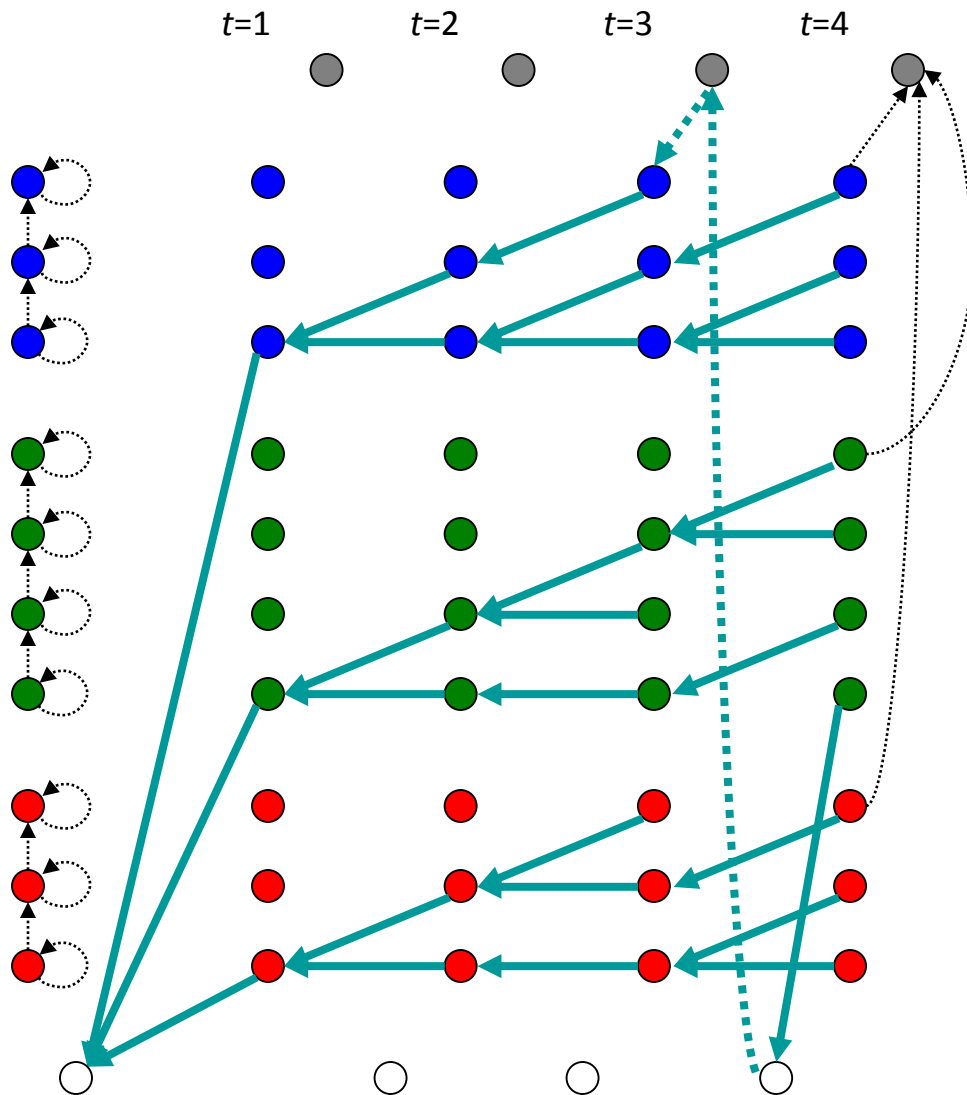
# Trellis with Complete Set of Backpointers



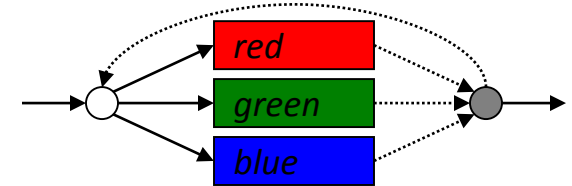
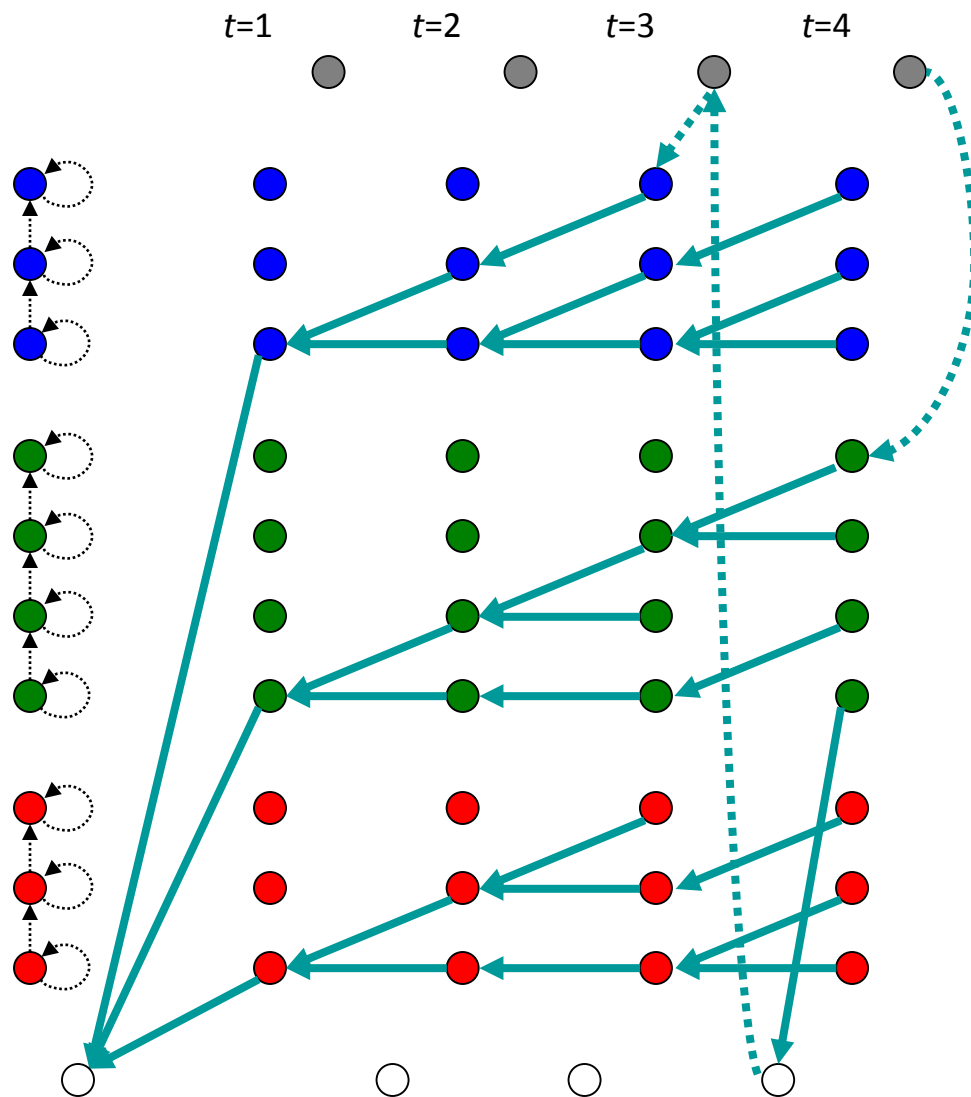
# Trellis with Complete Set of Backpointers



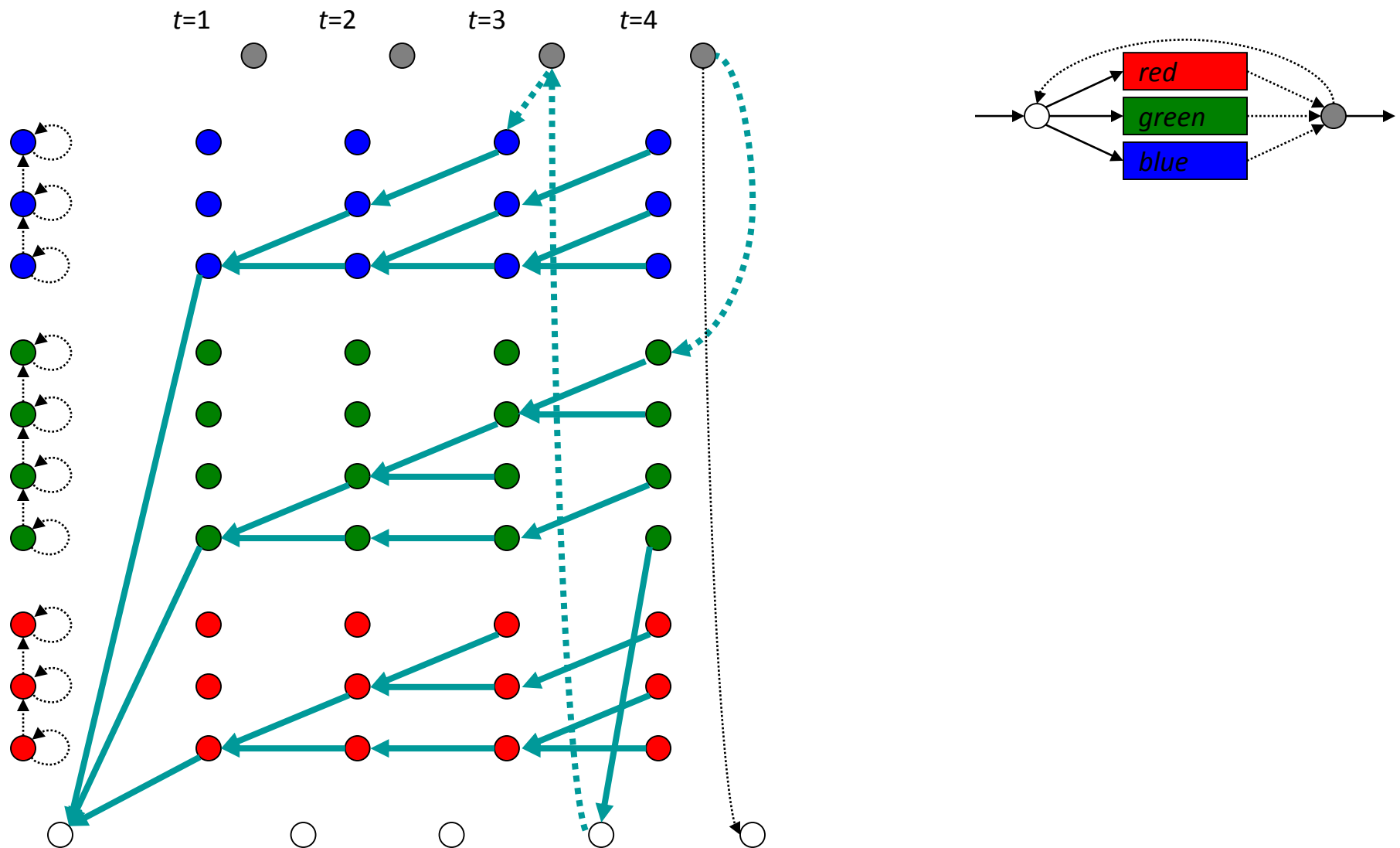
# Trellis with Complete Set of Backpointers



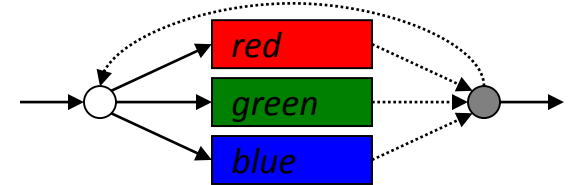
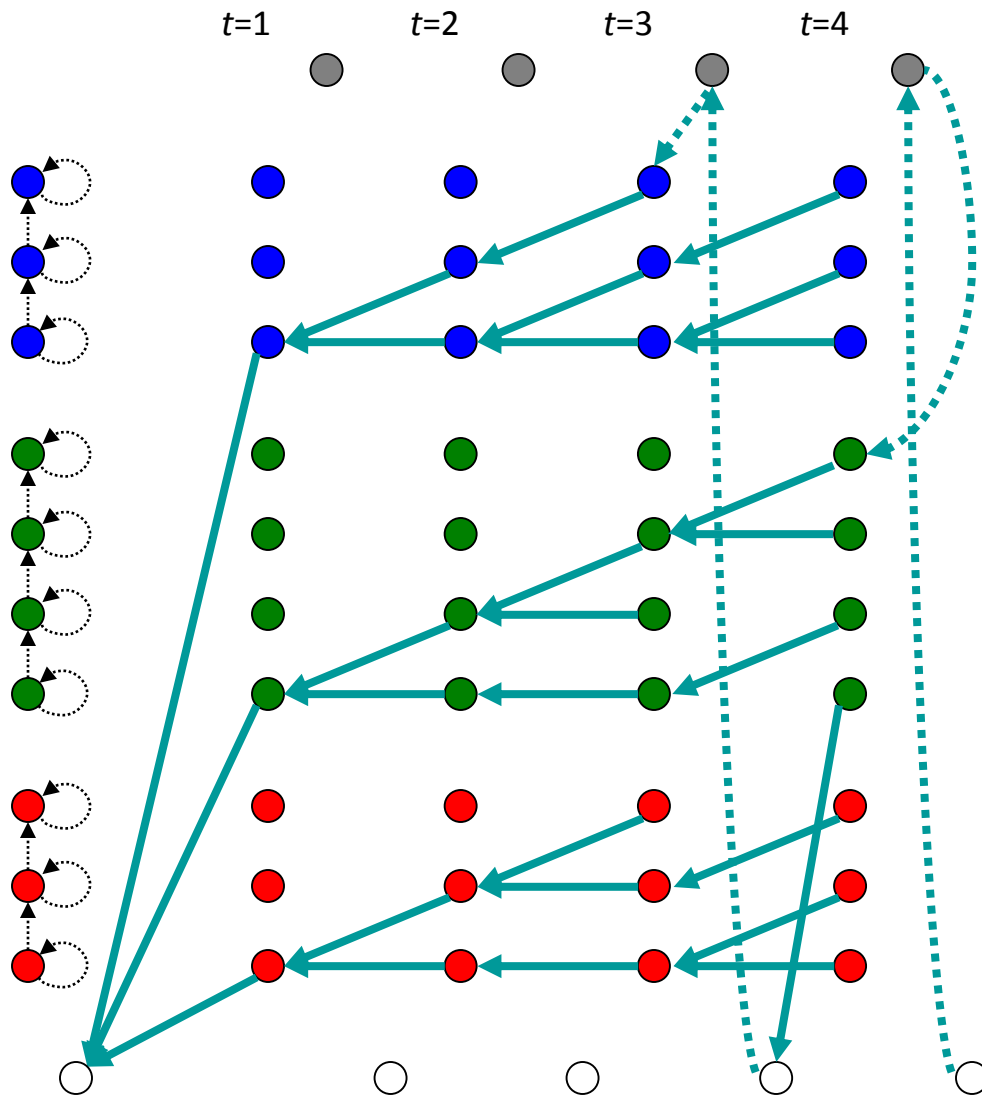
# Trellis with Complete Set of Backpointers



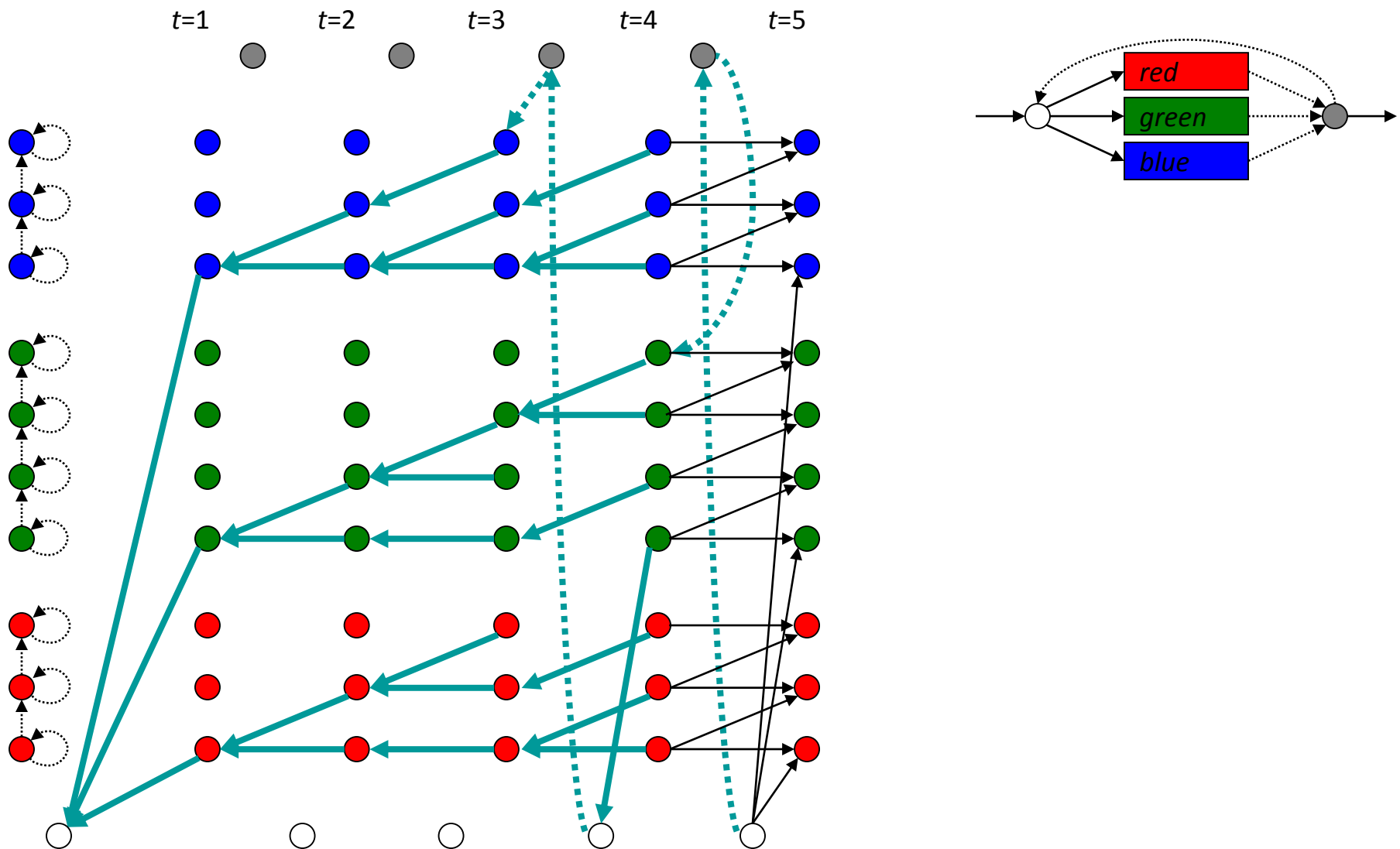
# Trellis with Complete Set of Backpointers



# Trellis with Complete Set of Backpointers

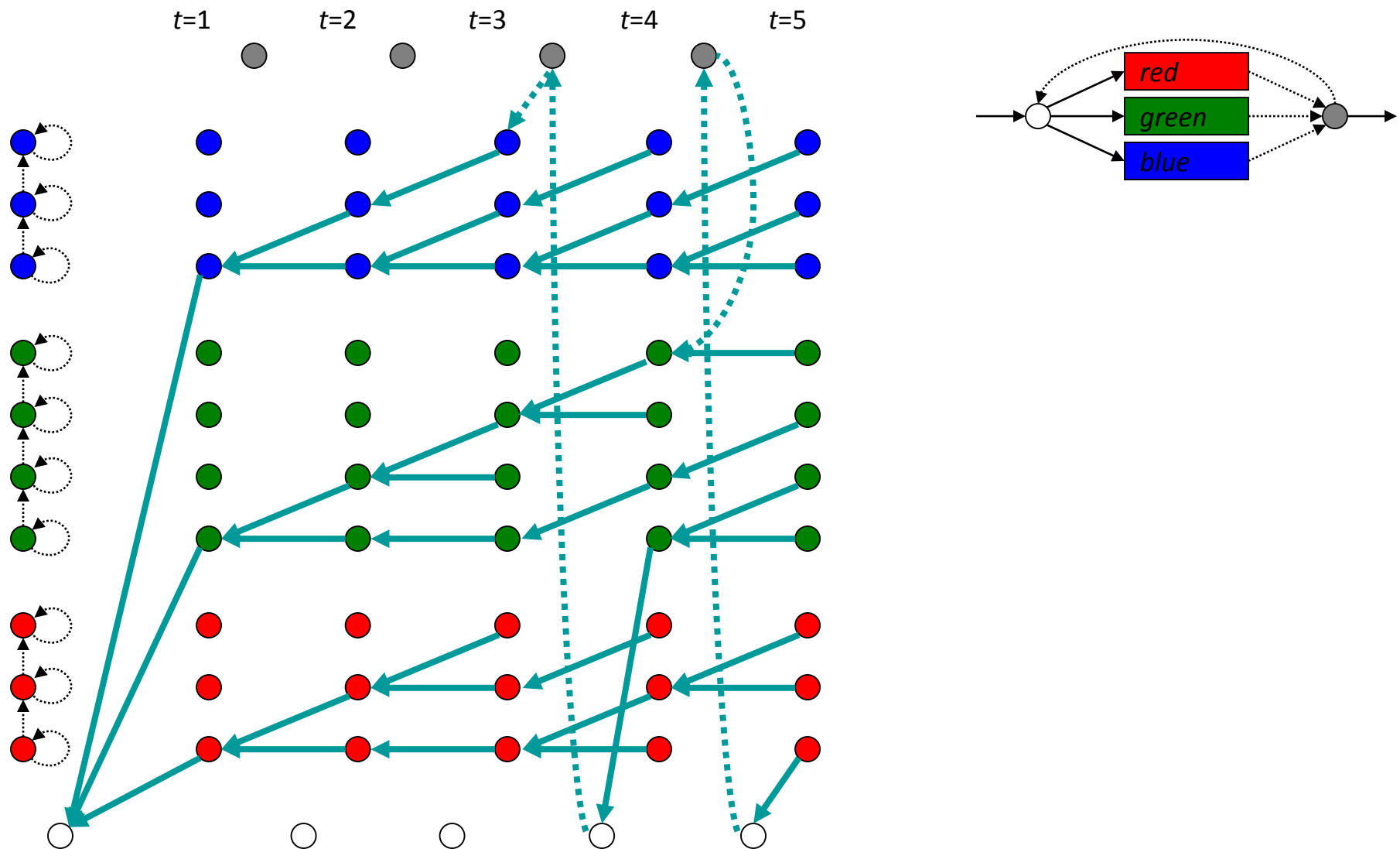


# Trellis with Complete Set of Backpointers

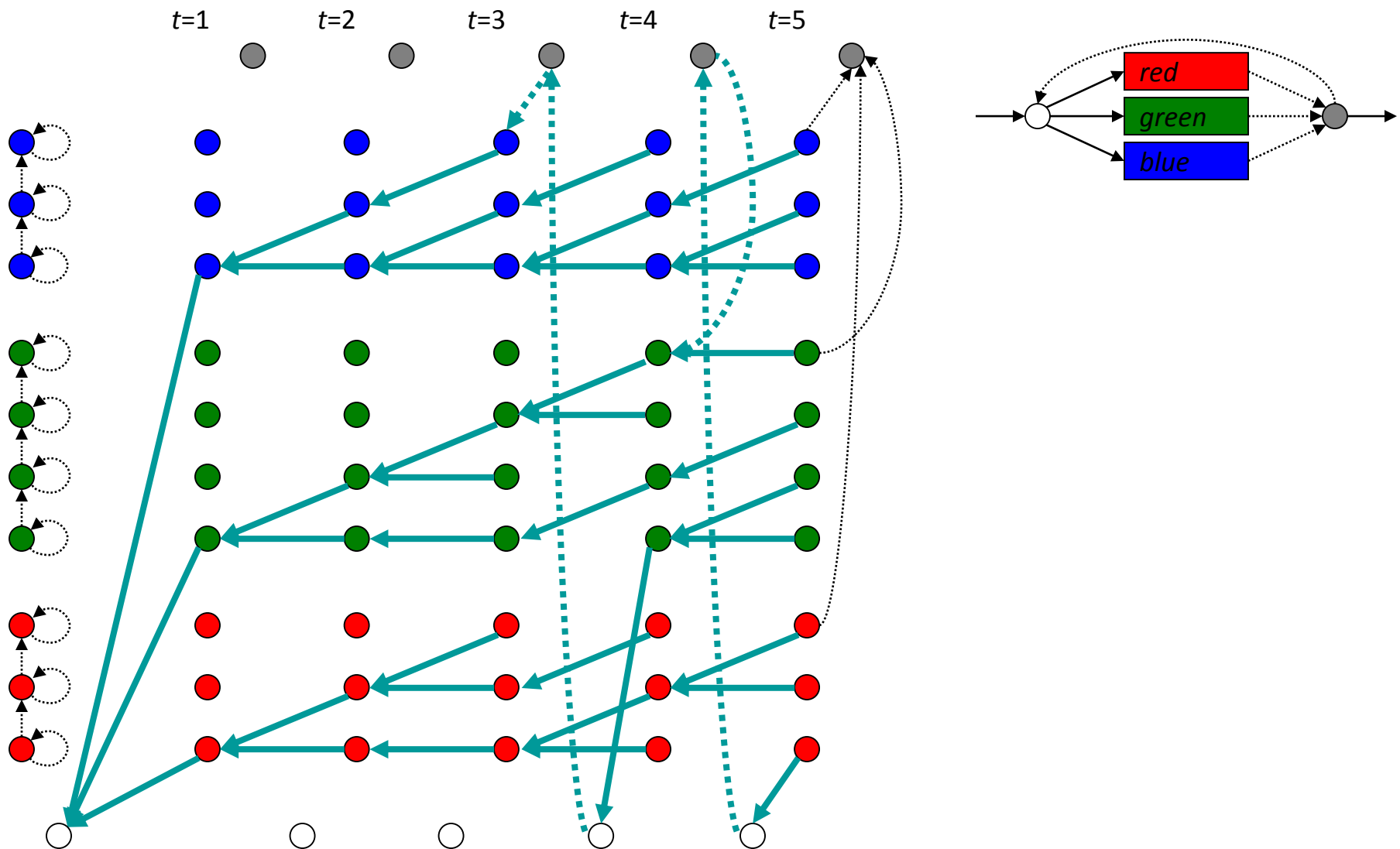




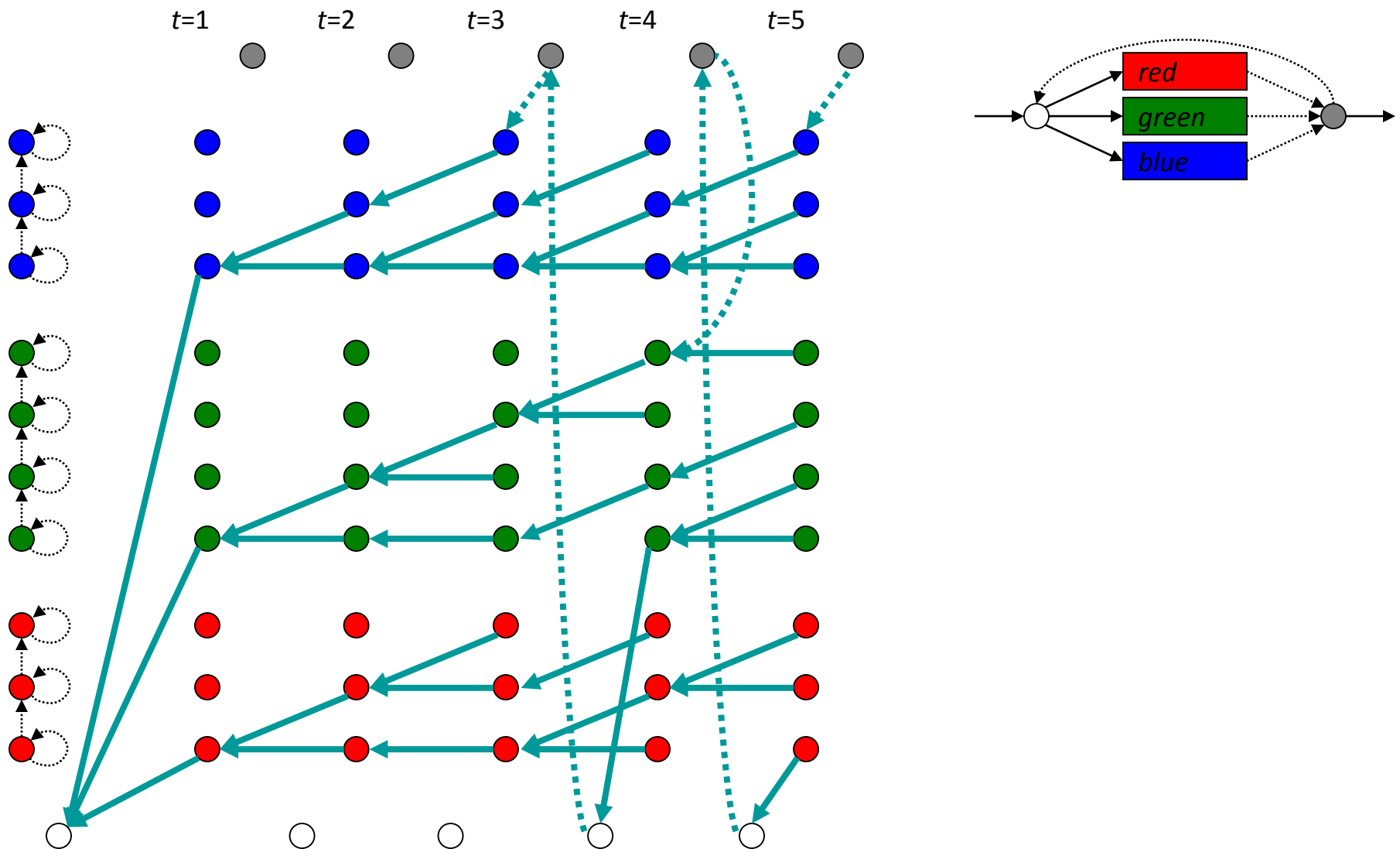
# Trellis with Complete Set of Backpointers



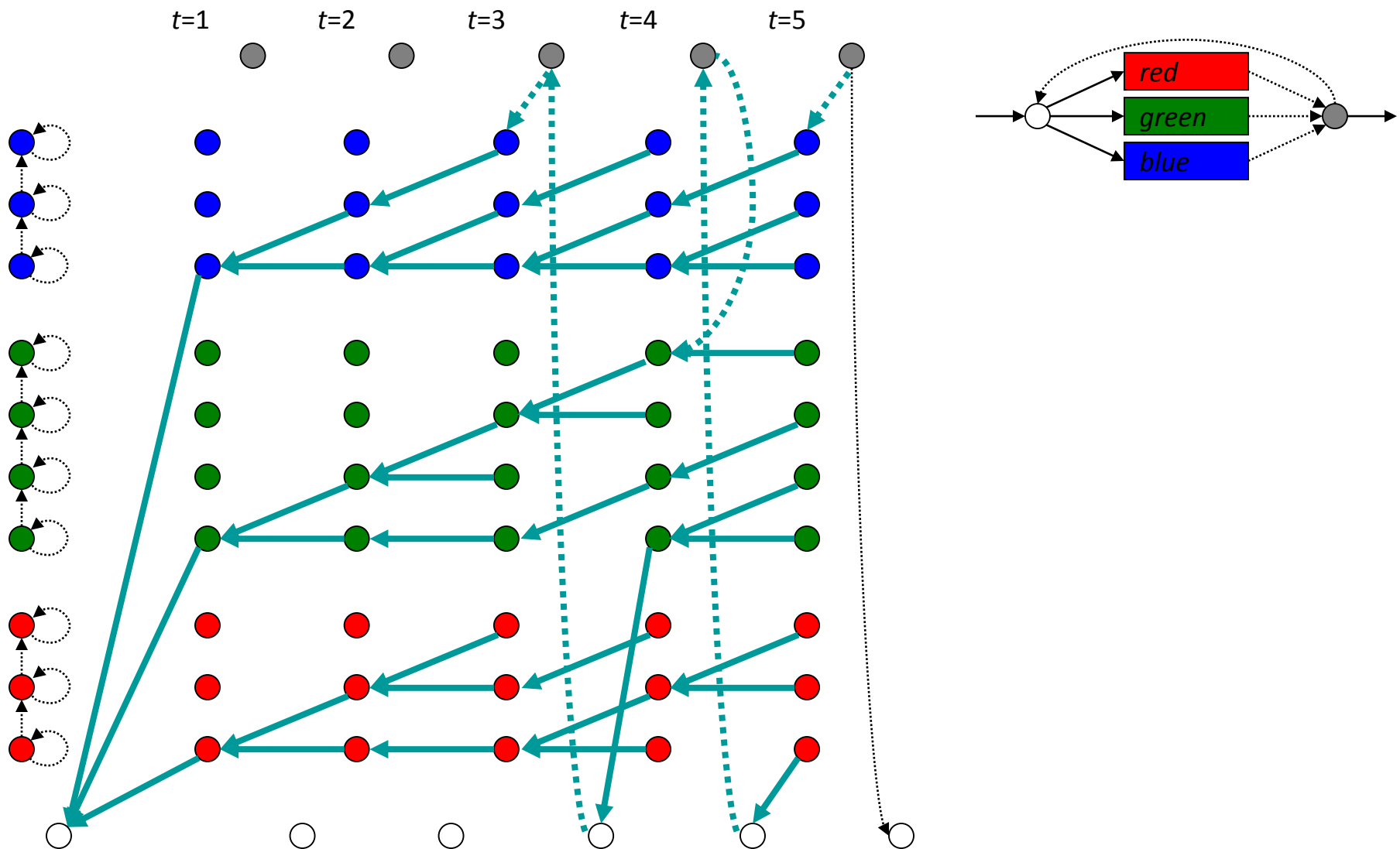
# Trellis with Complete Set of Backpointers



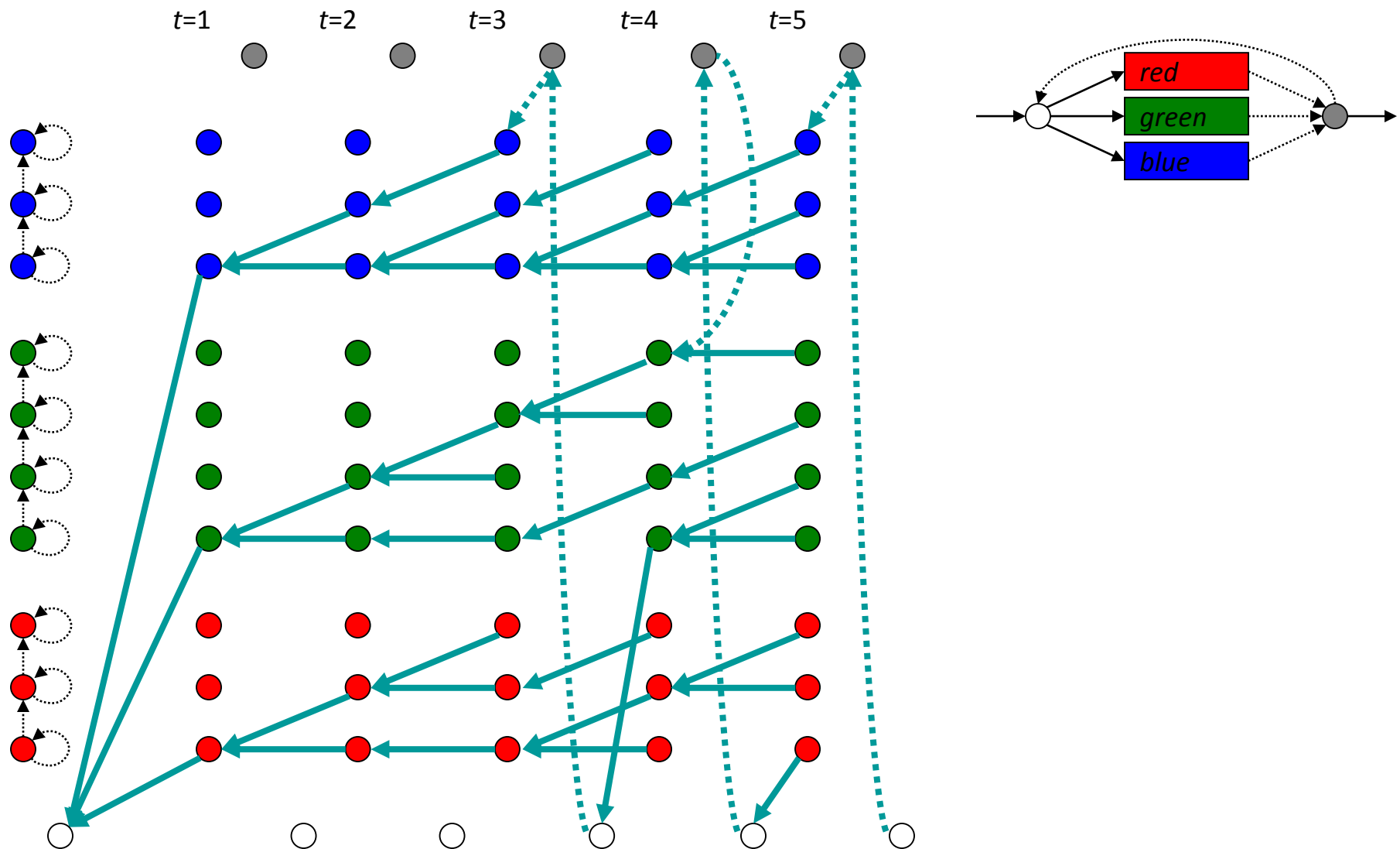
# Trellis with Complete Set of Backpointers



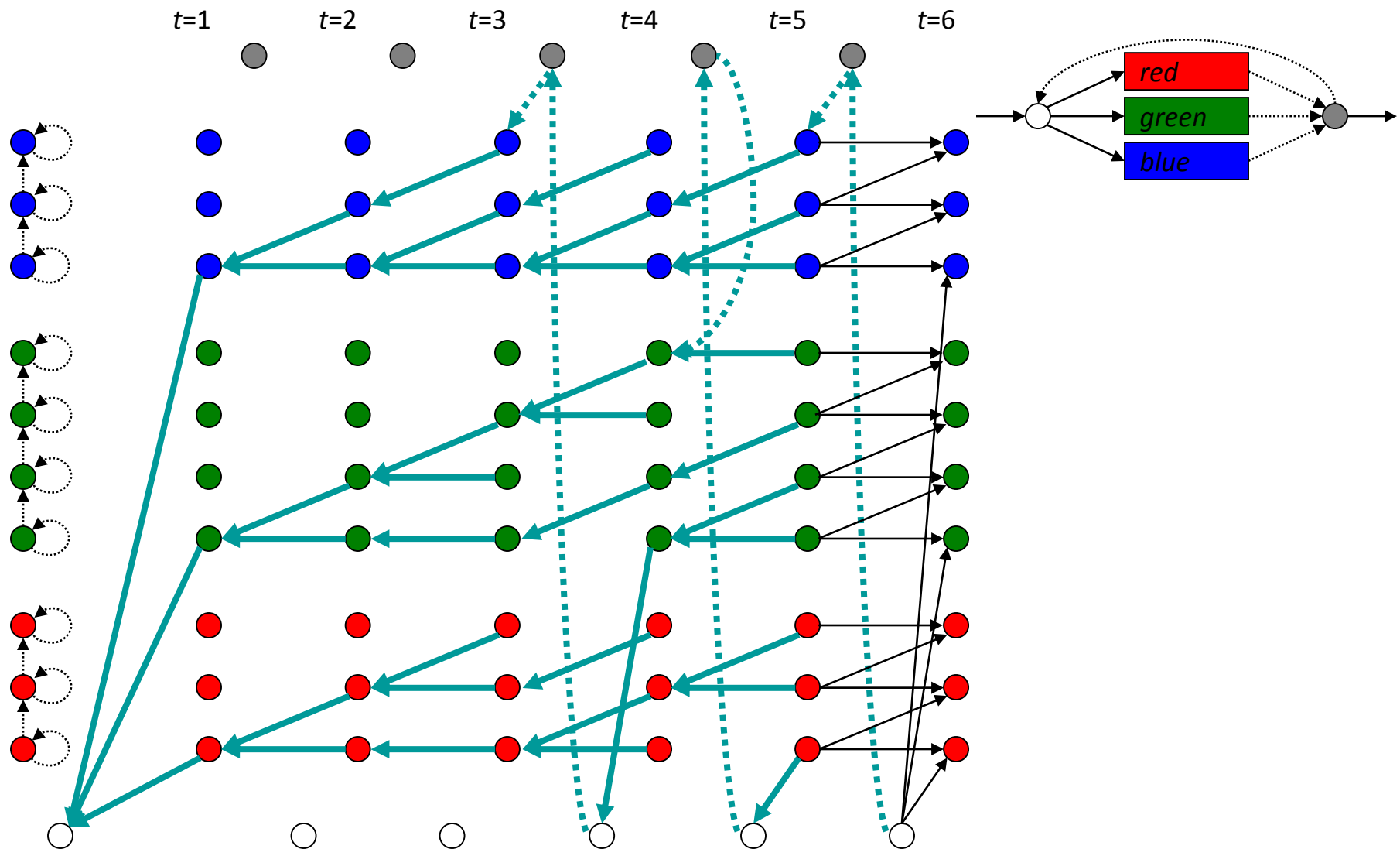
# Trellis with Complete Set of Backpointers



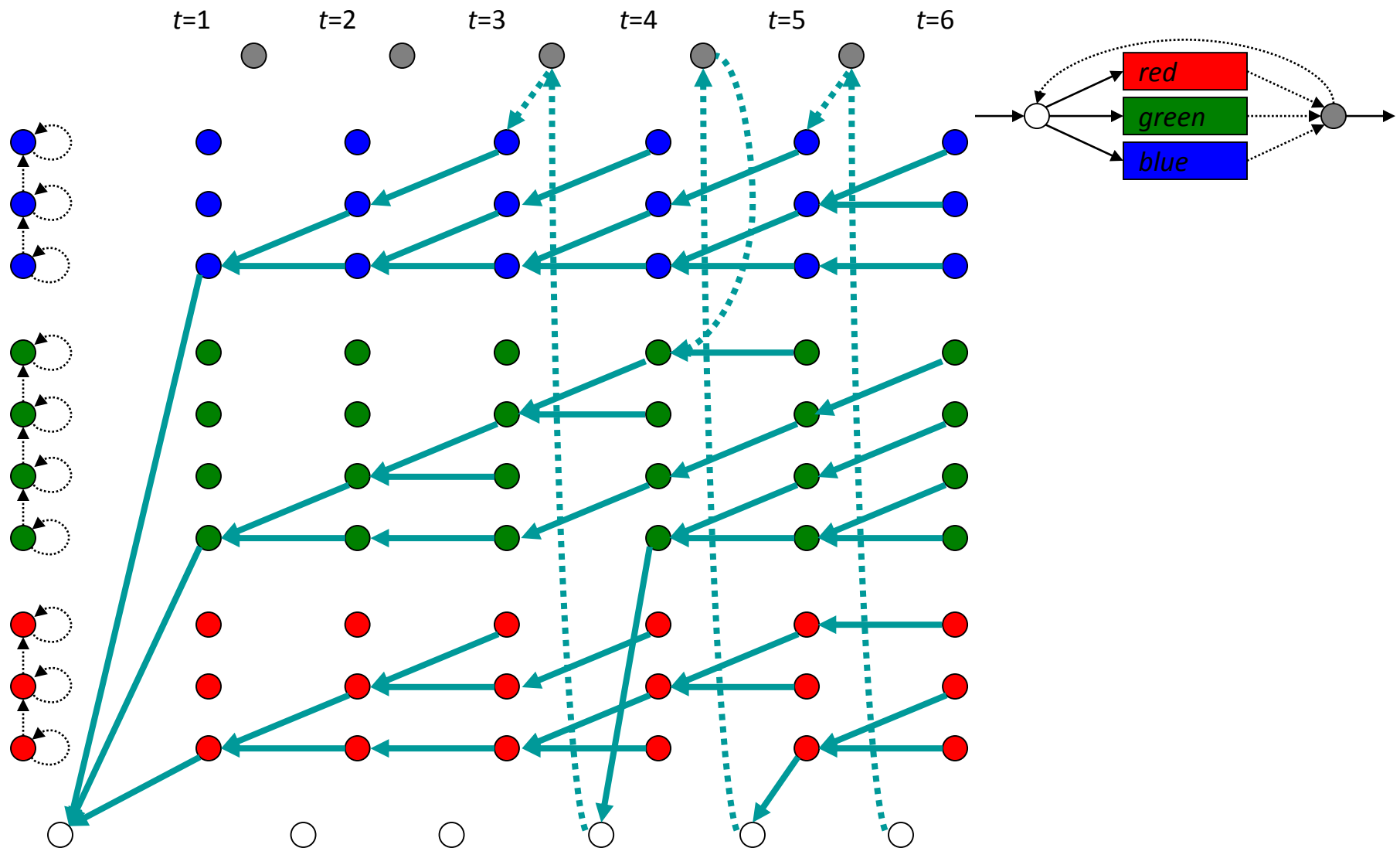
# Trellis with Complete Set of Backpointers



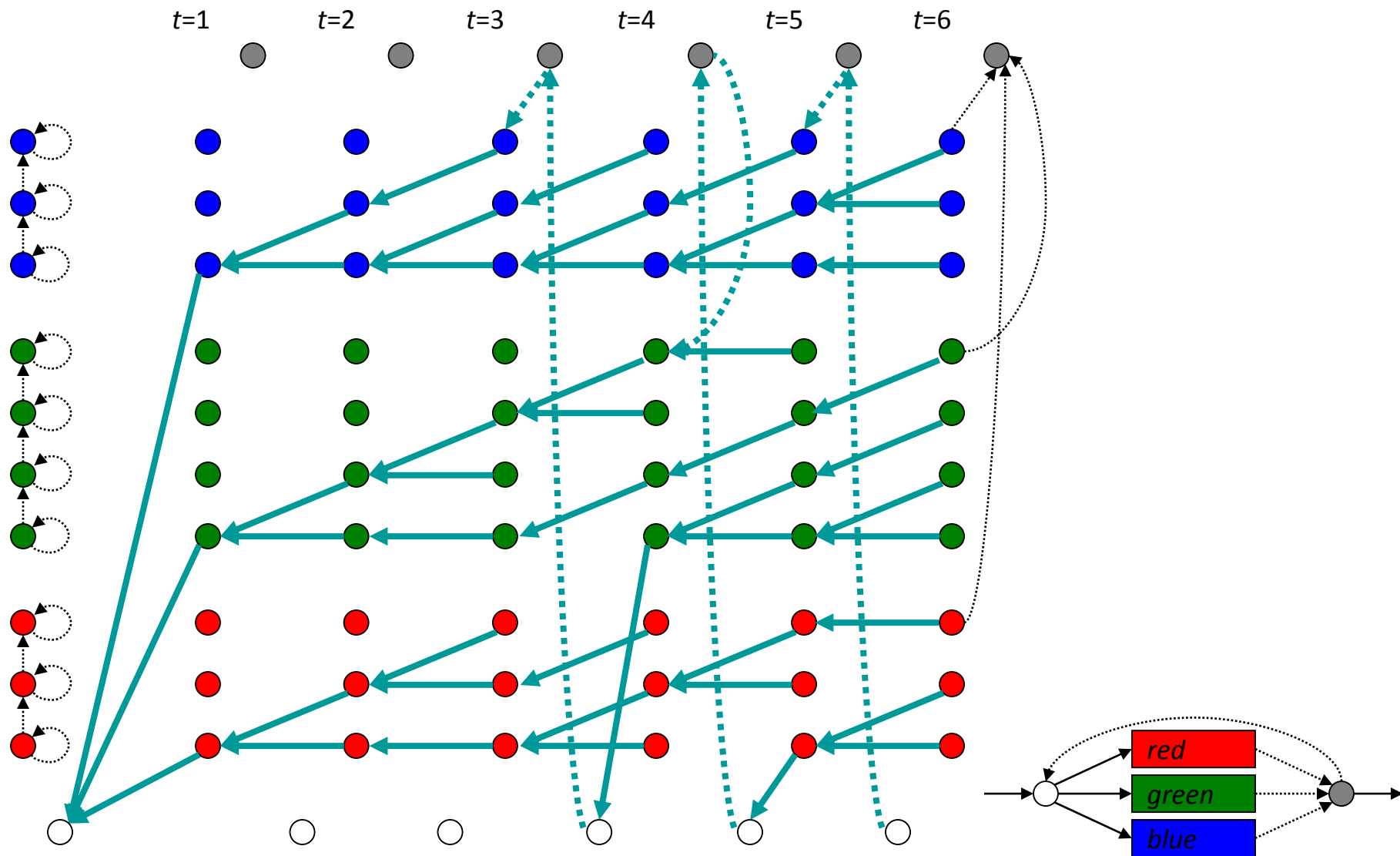
# Trellis with Complete Set of Backpointers



# Trellis with Complete Set of Backpointers

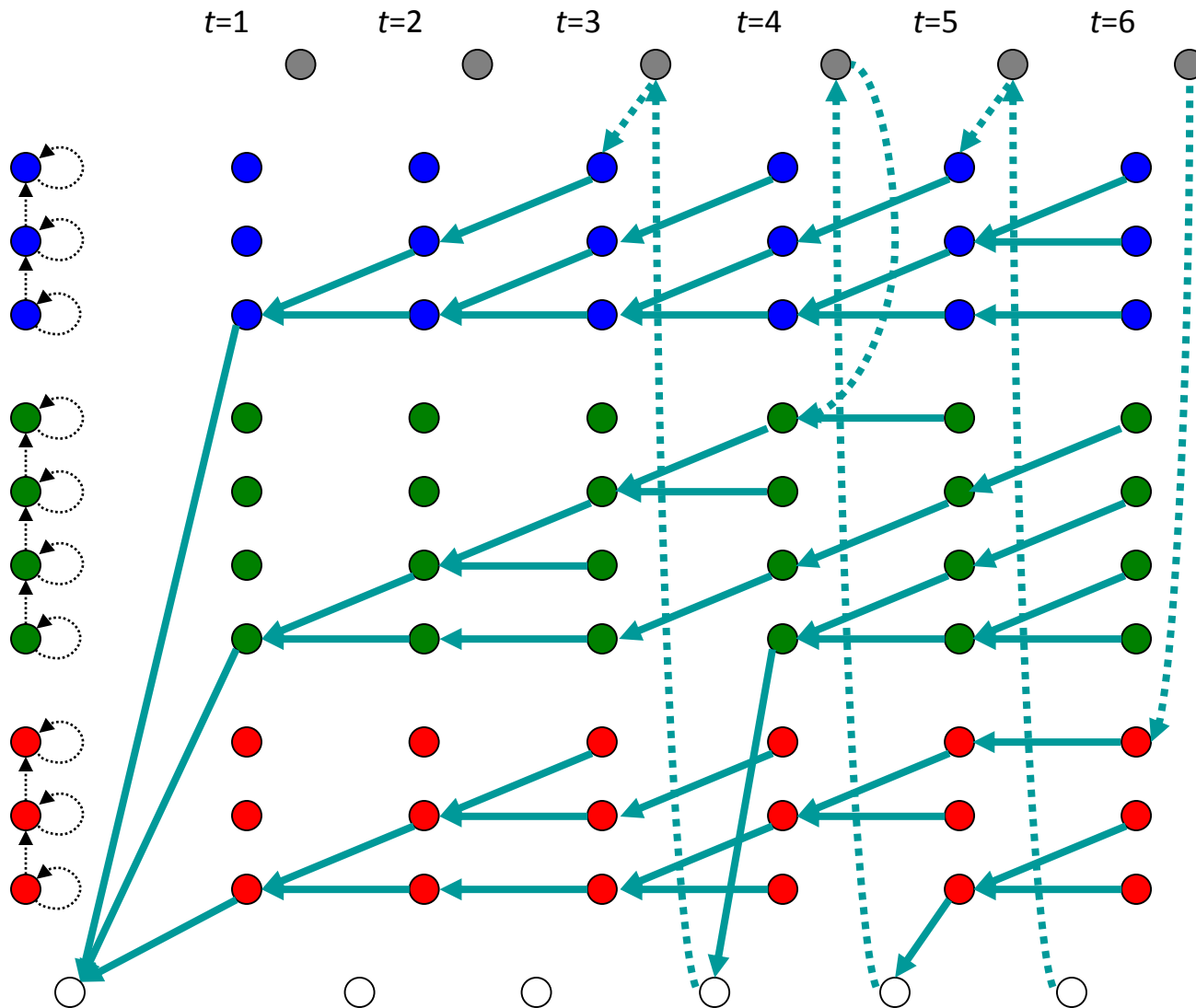


# Trellis with Complete Set of Backpointers

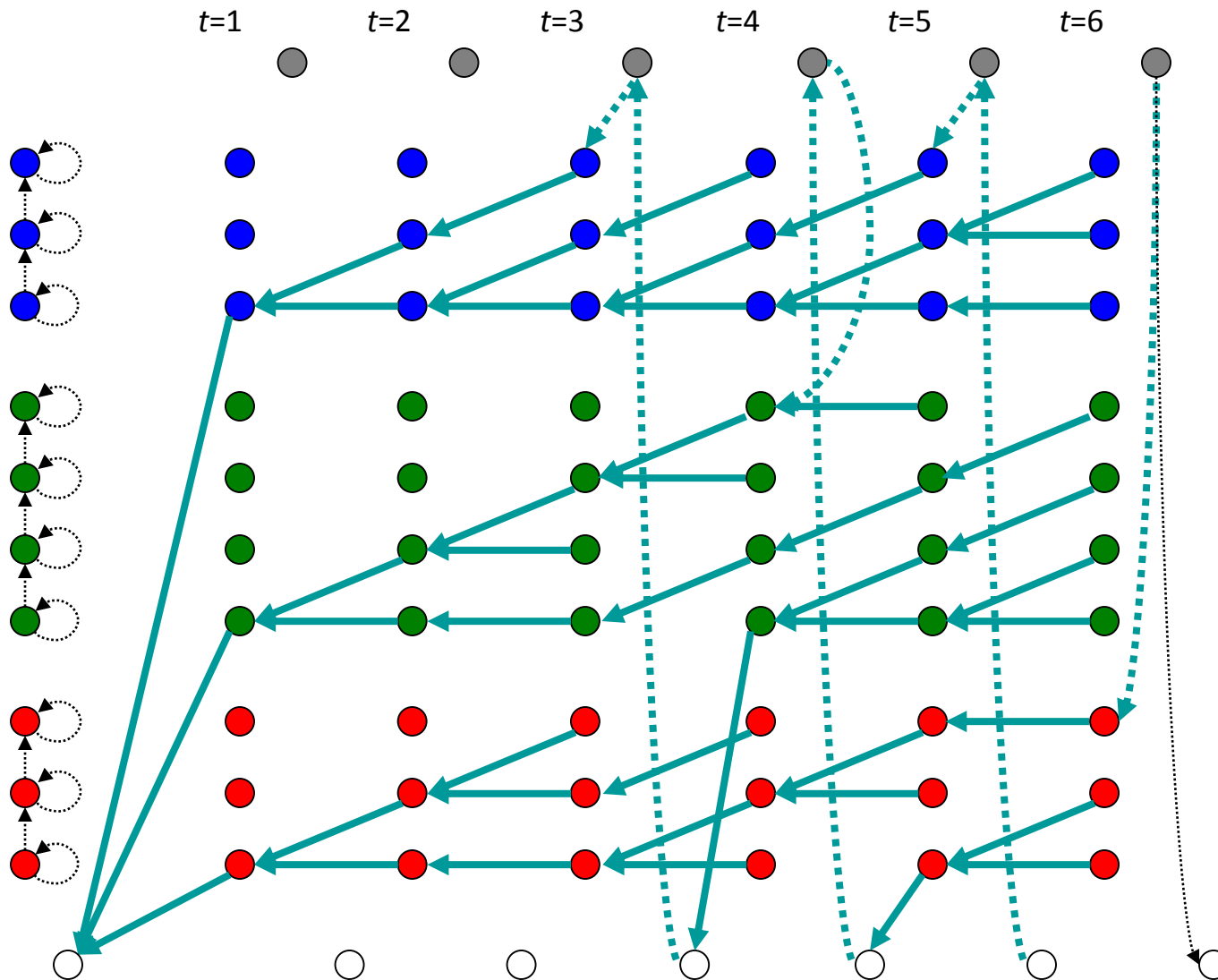




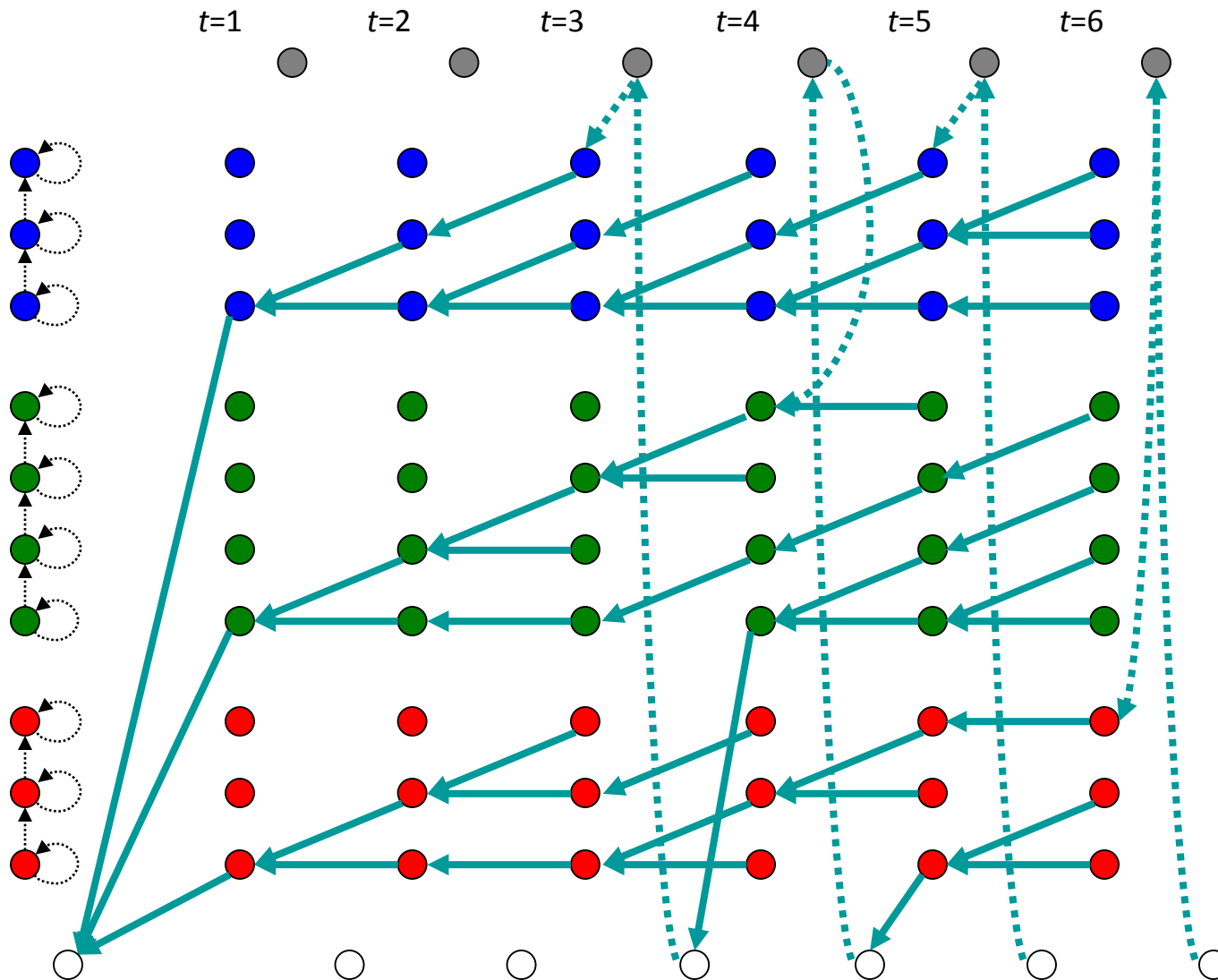
# Trellis with Complete Set of Backpointers



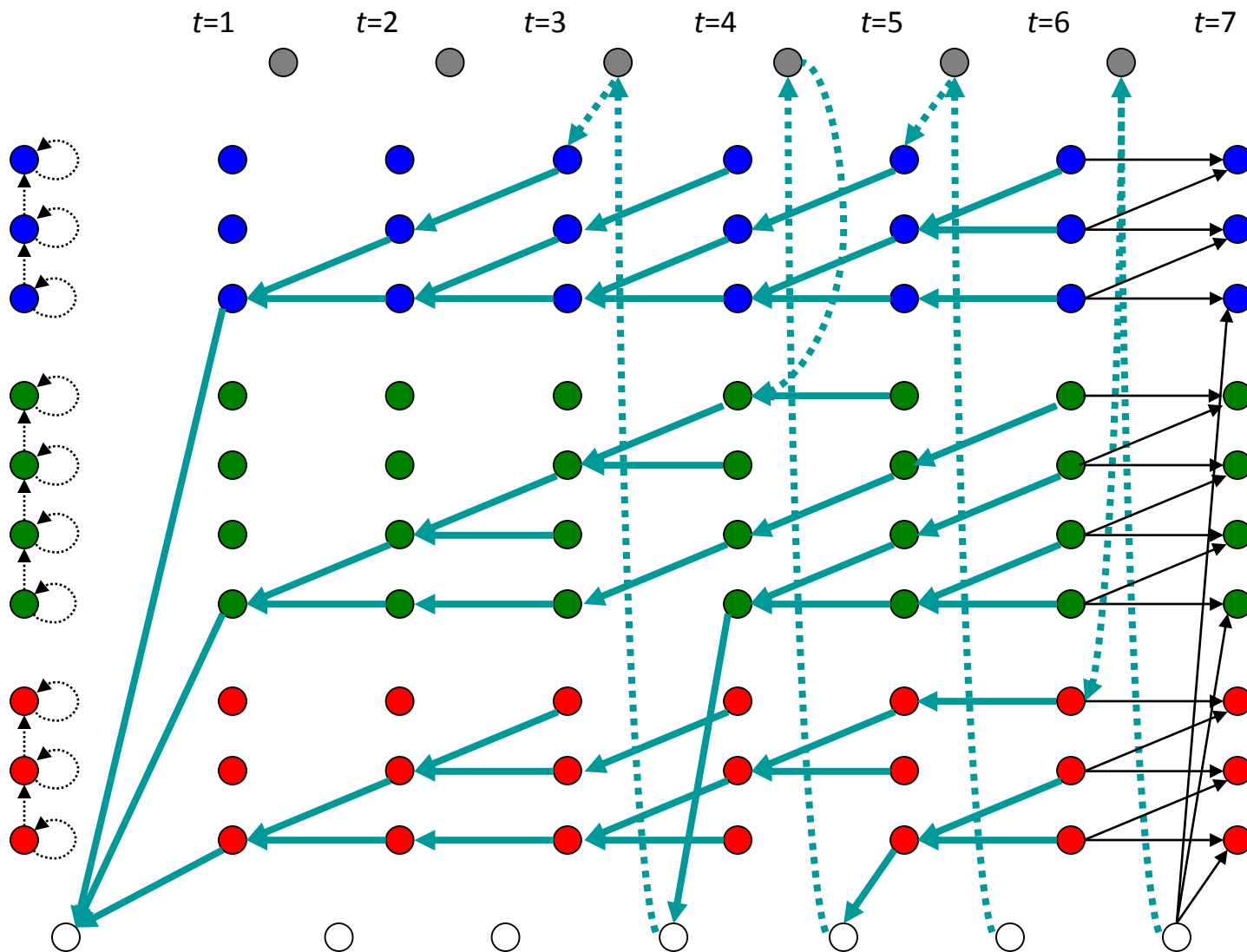
# Trellis with Complete Set of Backpointers



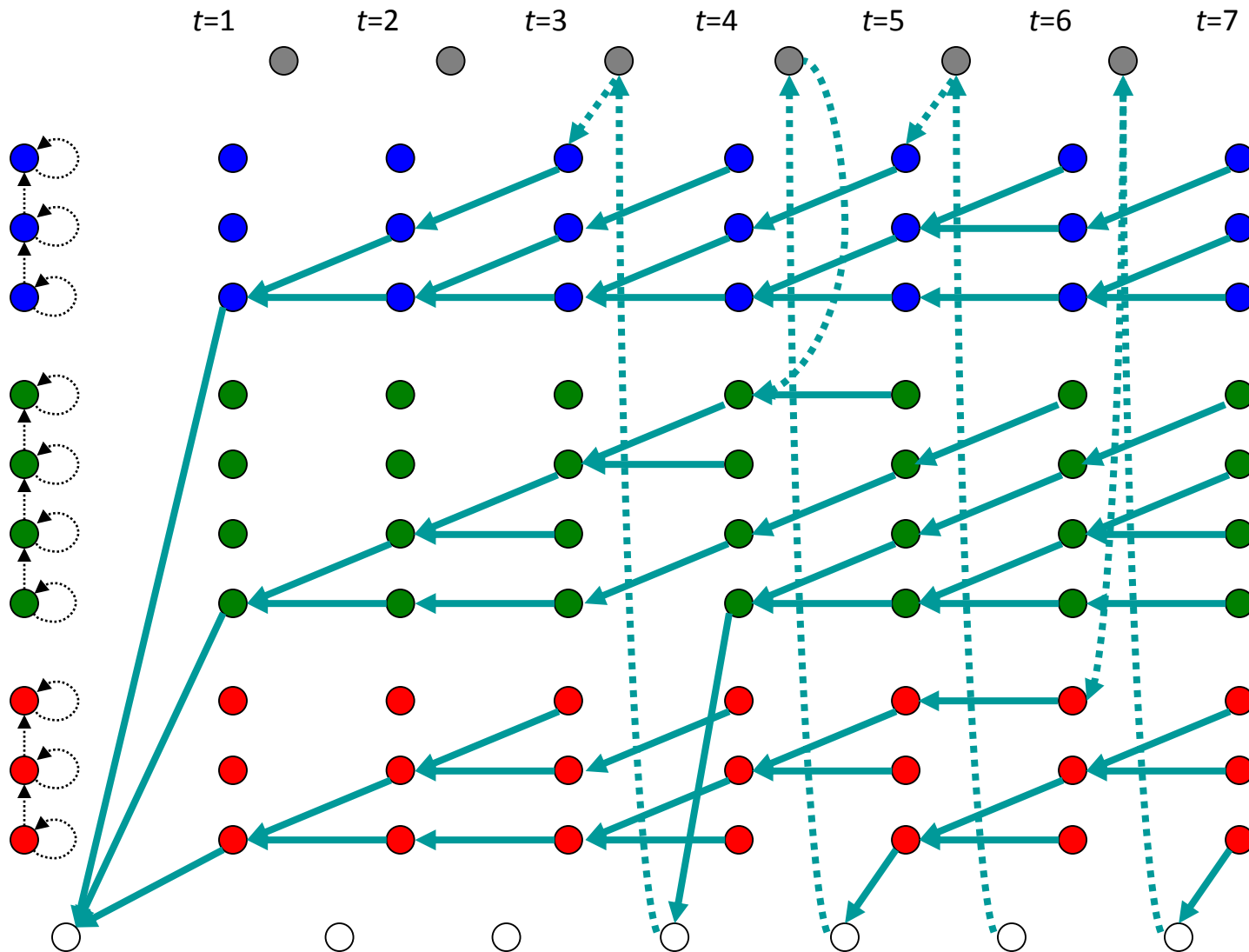
# Trellis with Complete Set of Backpointers



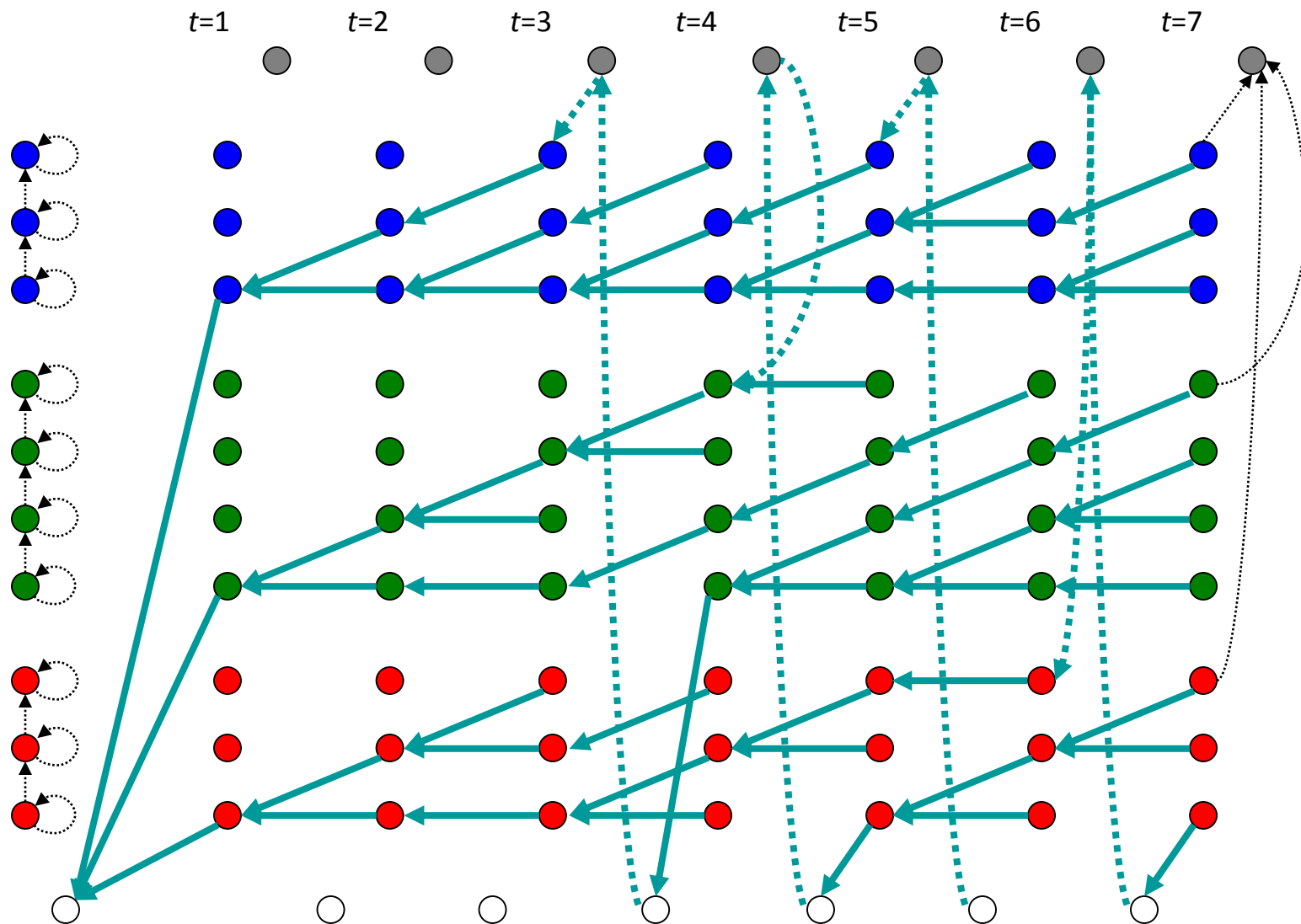
# Trellis with Complete Set of Backpointers



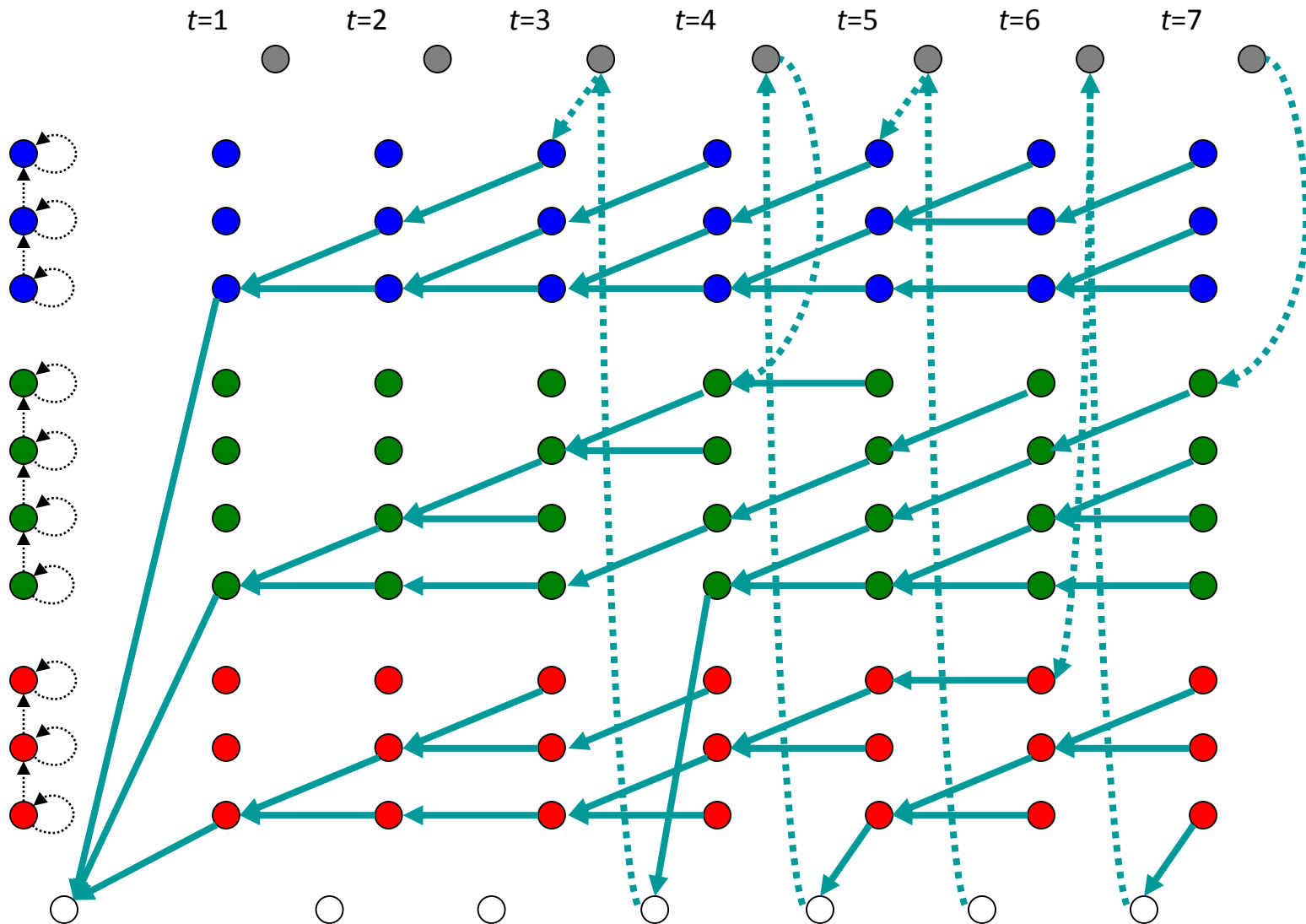
# Trellis with Complete Set of Backpointers



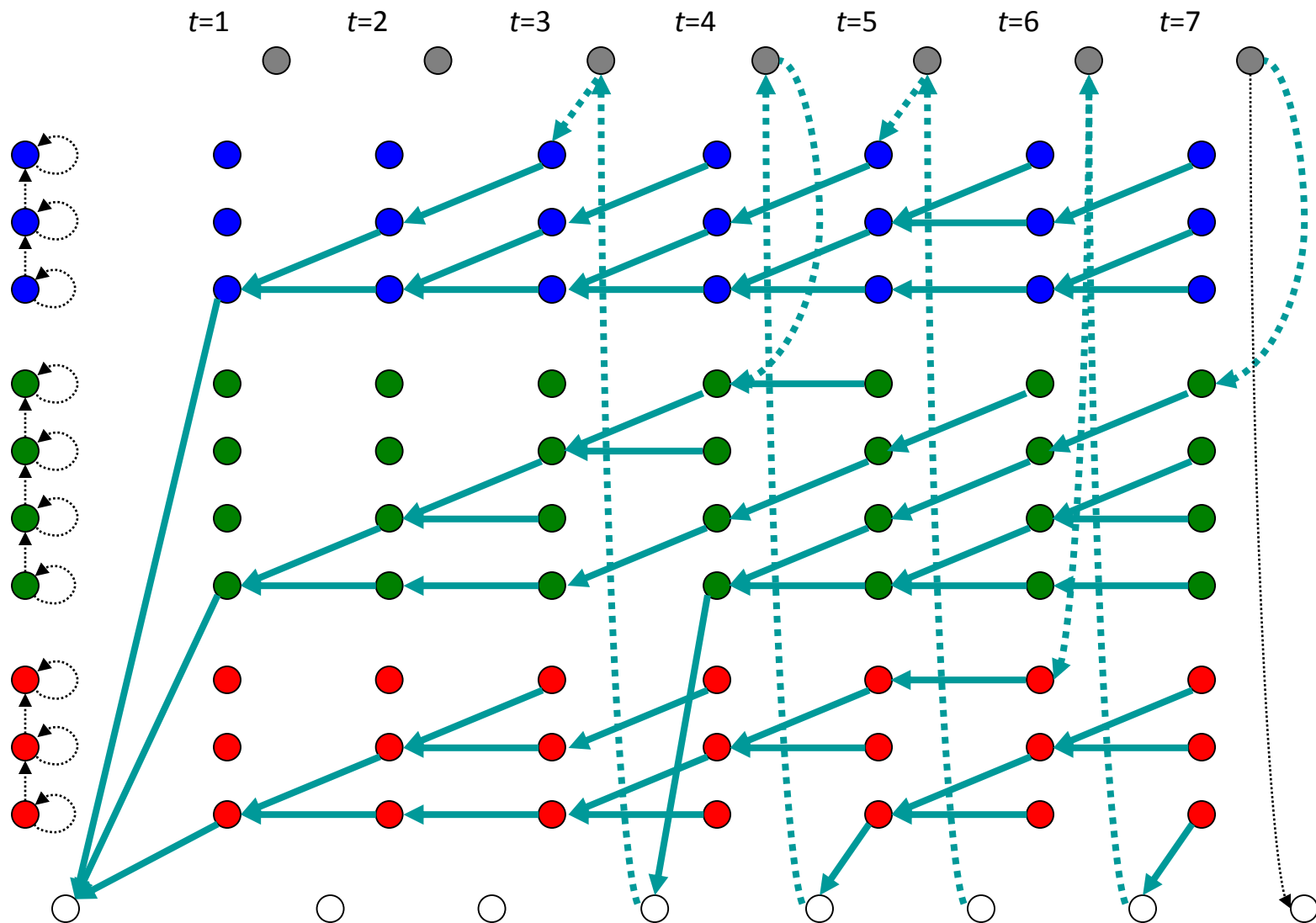
# Trellis with Complete Set of Backpointers



# Trellis with Complete Set of Backpointers

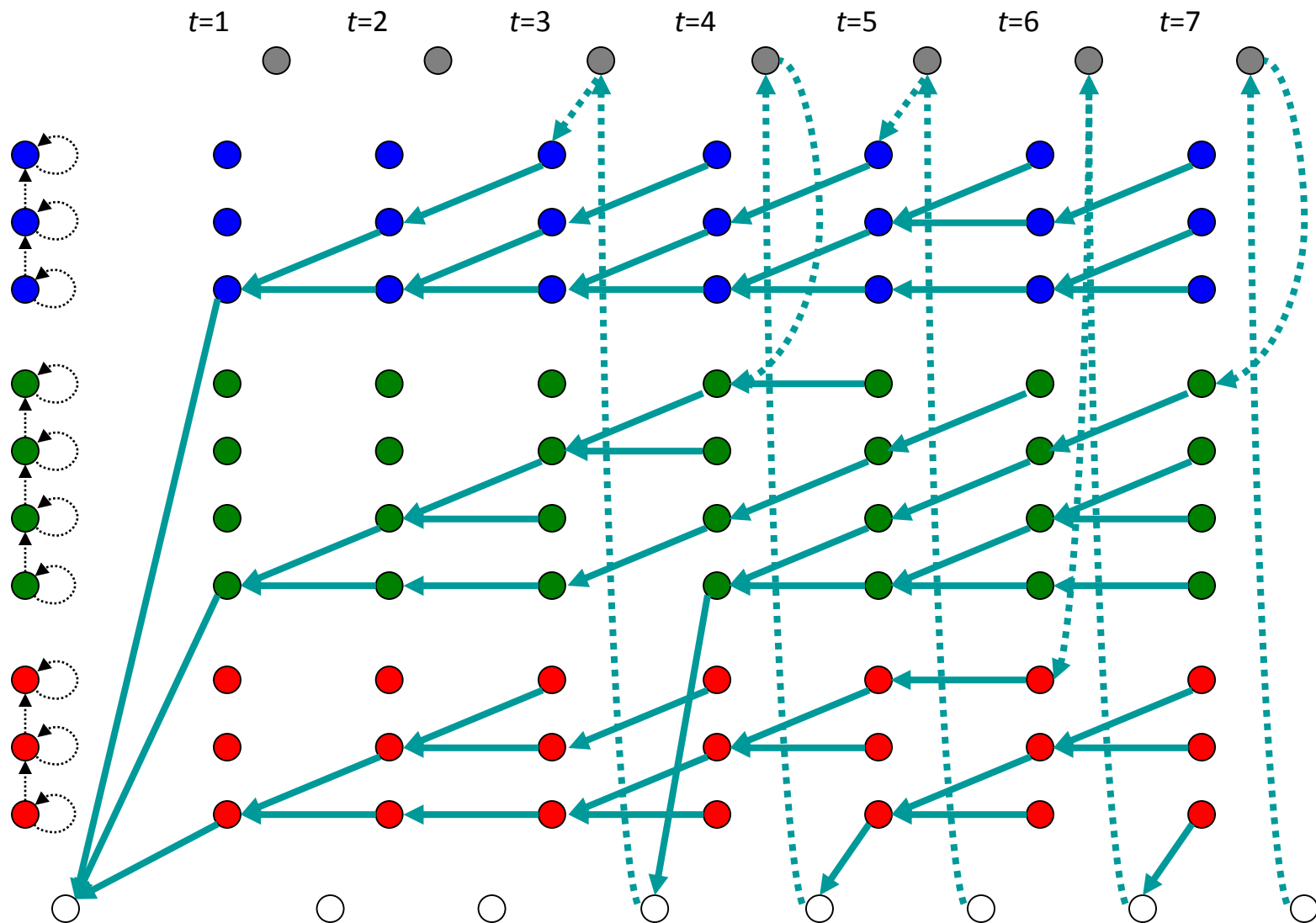


# Trellis with Complete Set of Backpointers

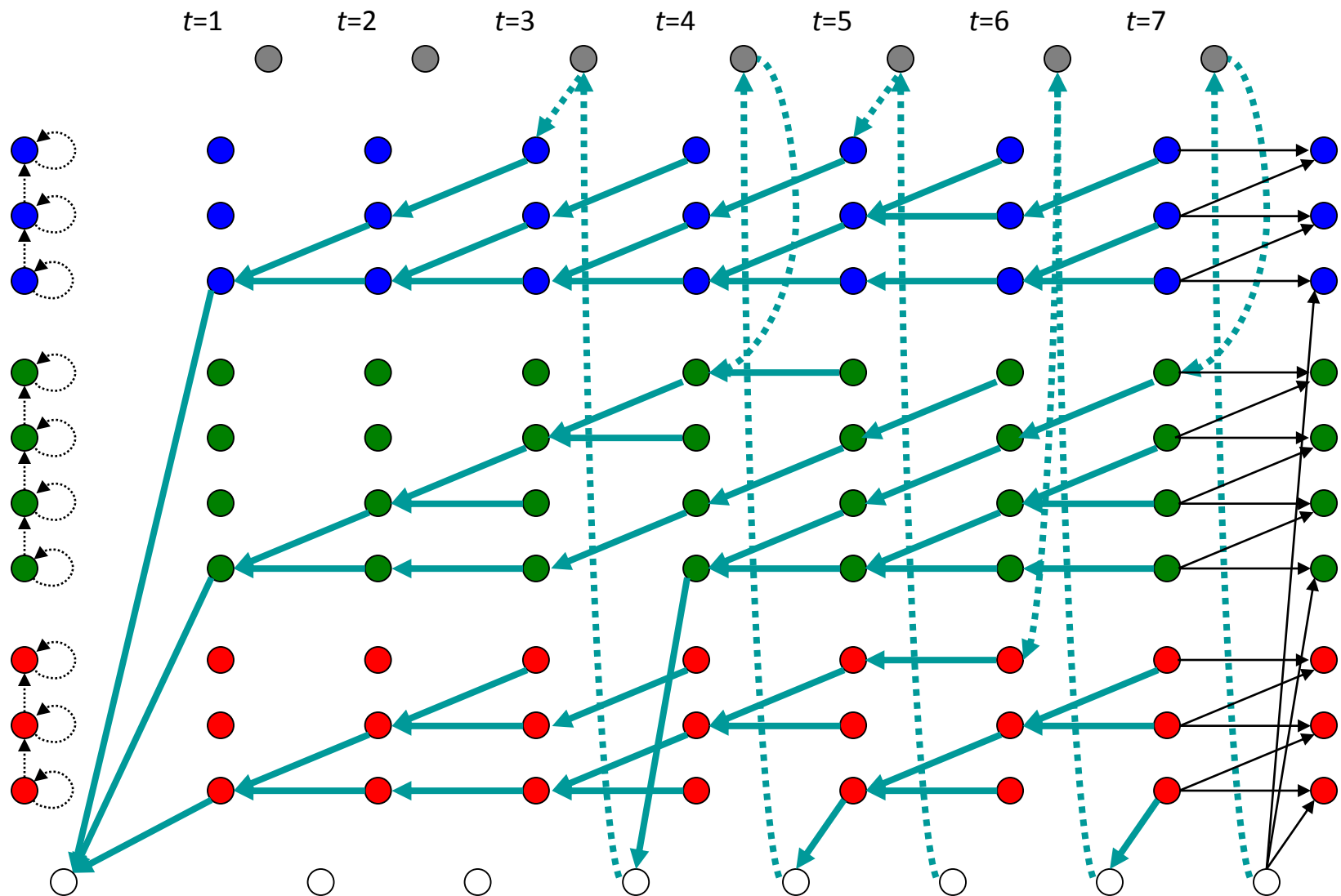




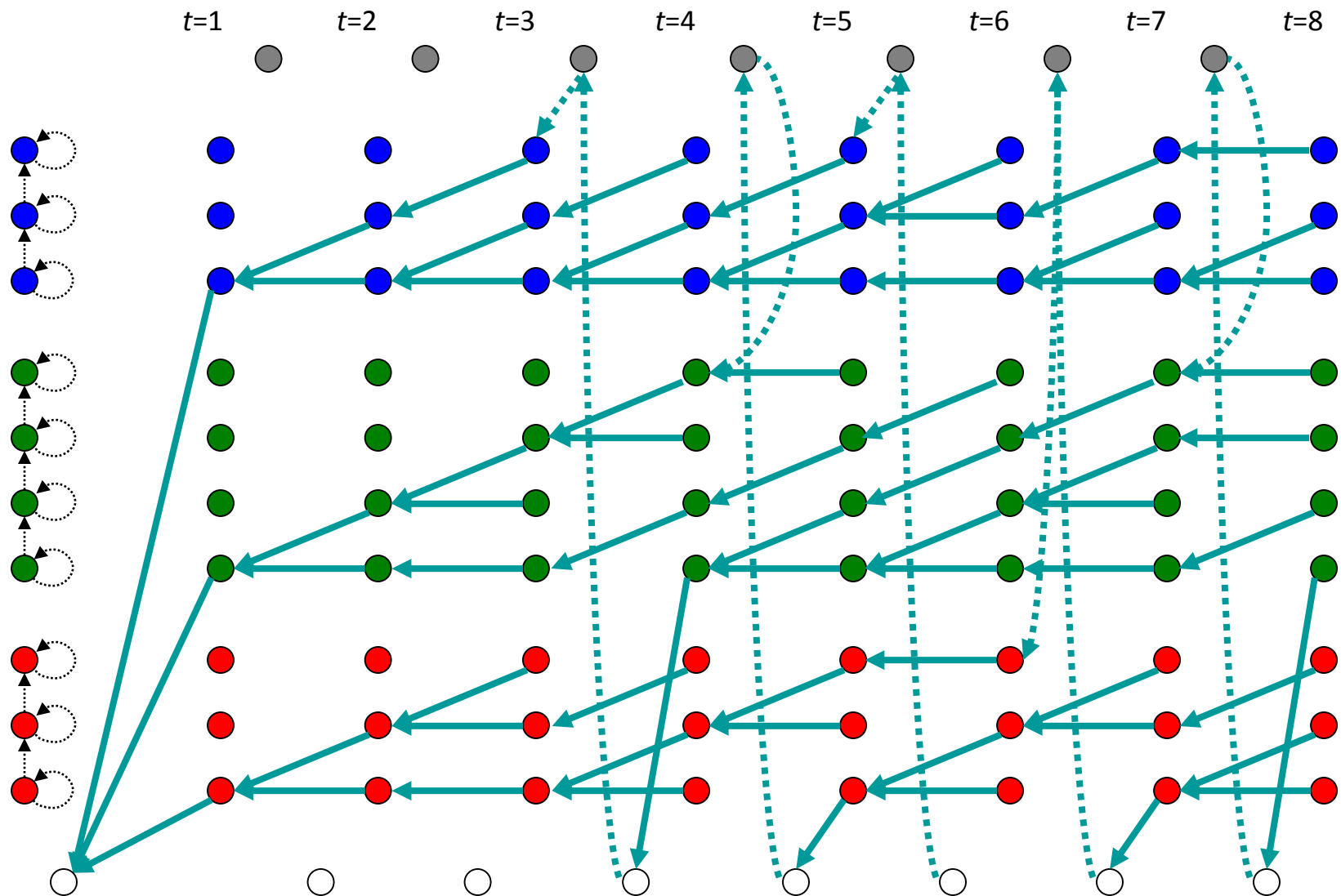
# Trellis with Complete Set of Backpointers



# Trellis with Complete Set of Backpointers



# Trellis with Complete Set of Backpointers



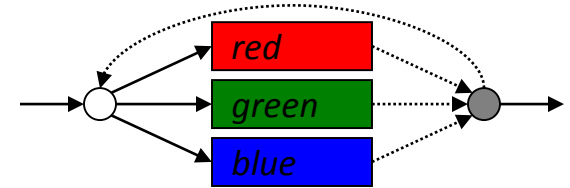
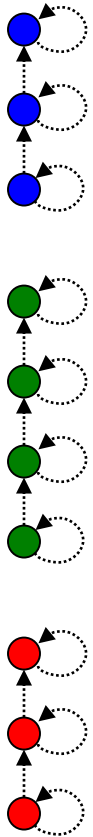
# Using Backpointers

---

- Retaining the complete set of back pointers can be very expensive
  - In terms of memory
- Solution: Only retain back pointers to the entry into words
  - Which can be stored separately as a “backpointer table”

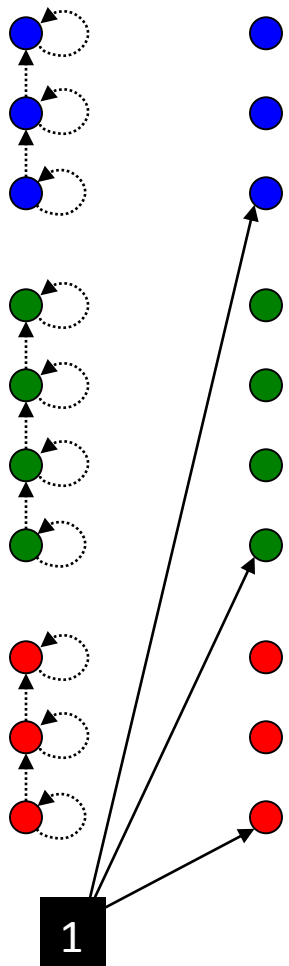
# Trellis with Complete Set of Backpointers

1, t=0, scr1,p=0,...

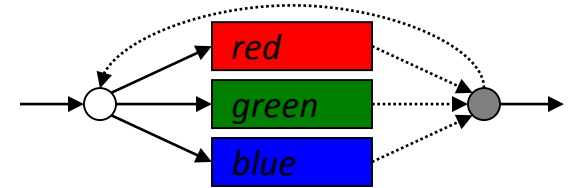


# Trellis with Complete Set of Backpointers

$t=1$

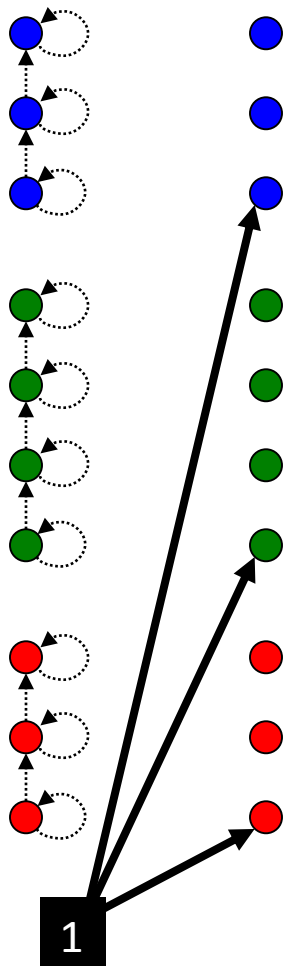


1,  $t=0$ , scr1,  $p=0$ ,...

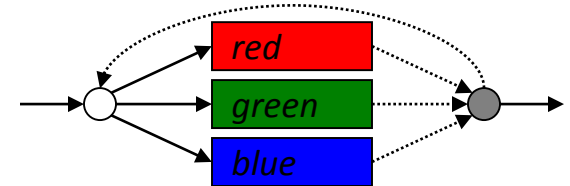


# Trellis with Complete Set of Backpointers

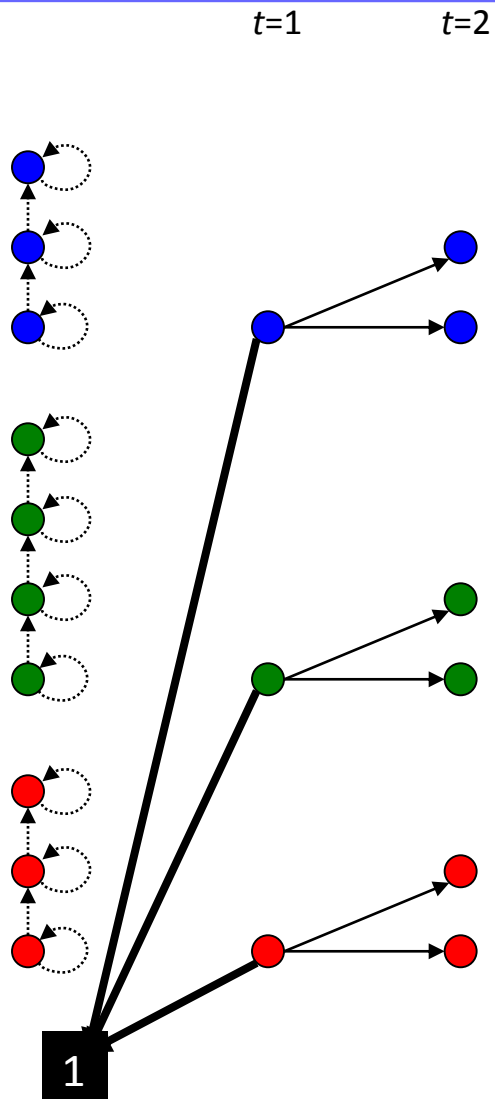
$t=1$



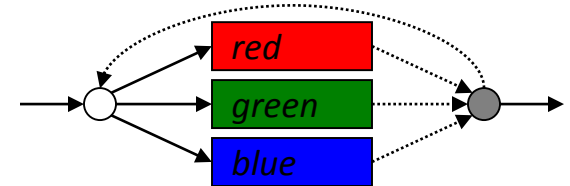
1,  $t=0$ , scr1,  $p=0$ , ...



# Trellis with Complete Set of Backpointers

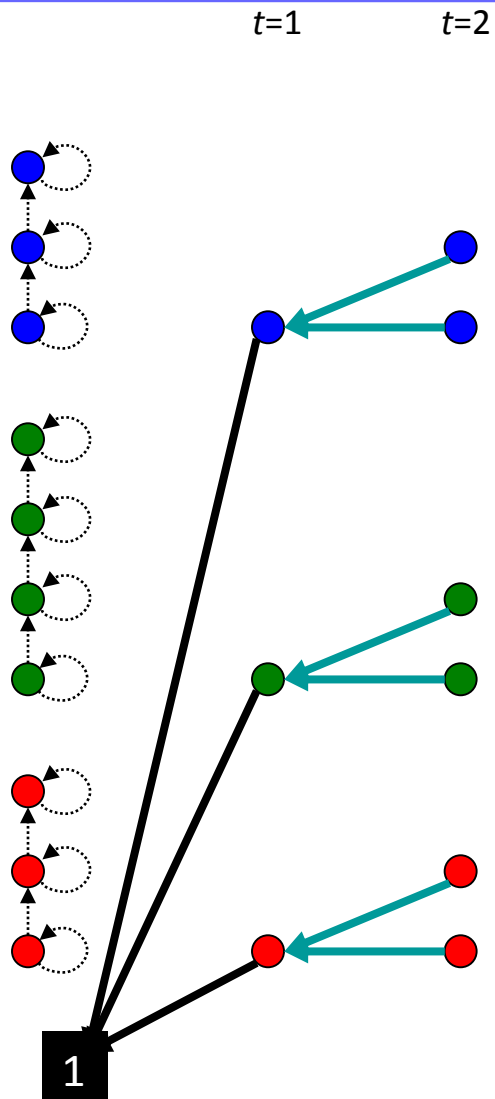


1,  $t=0$ , scr1,  $p=0$ ,...

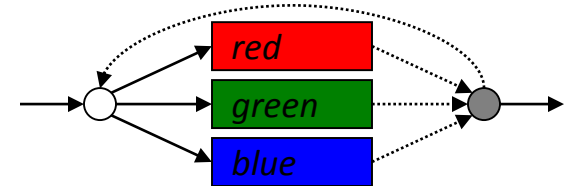




# Trellis with Complete Set of Backpointers



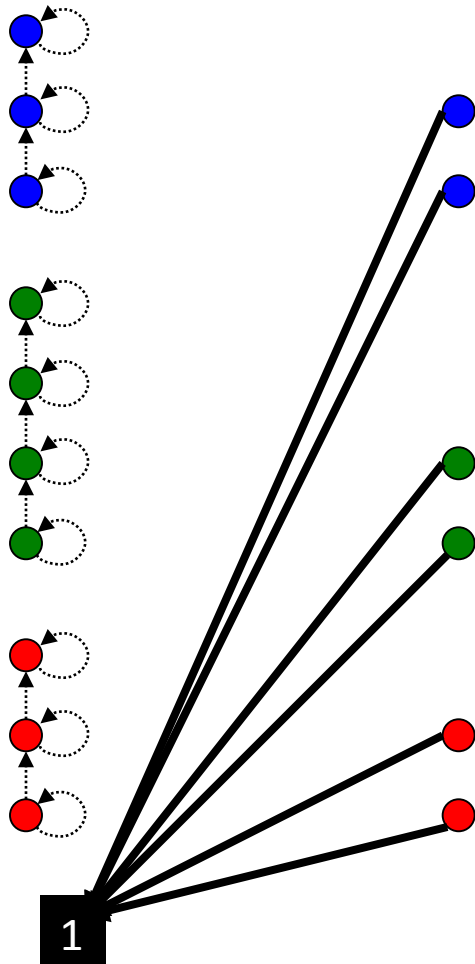
1,  $t=0$ , scr1,  $p=0$ ,...



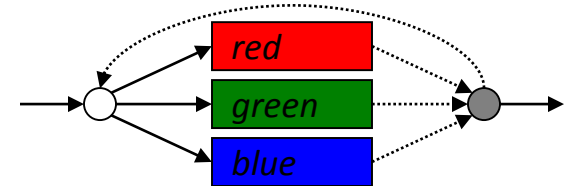
# Trellis with Complete Set of Backpointers

$t=1$

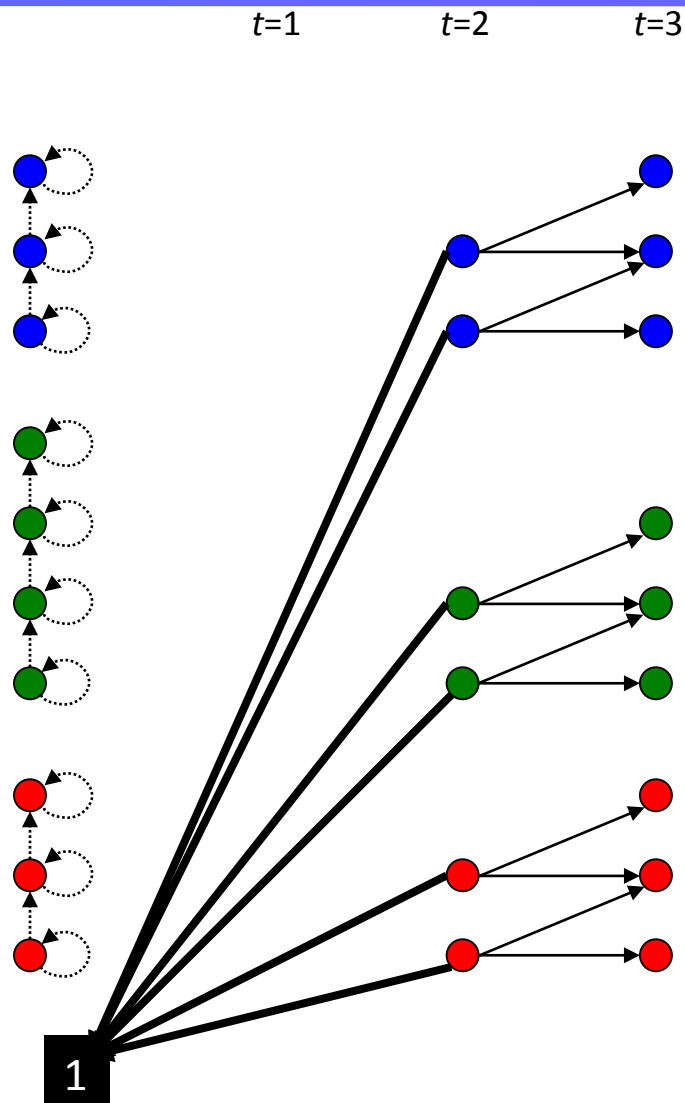
$t=2$



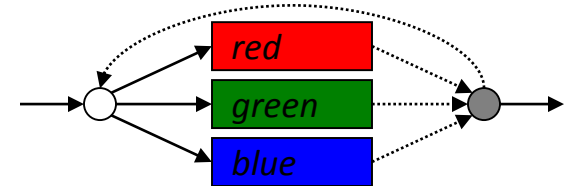
1,  $t=0$ , scr1,  $p=0, \dots$



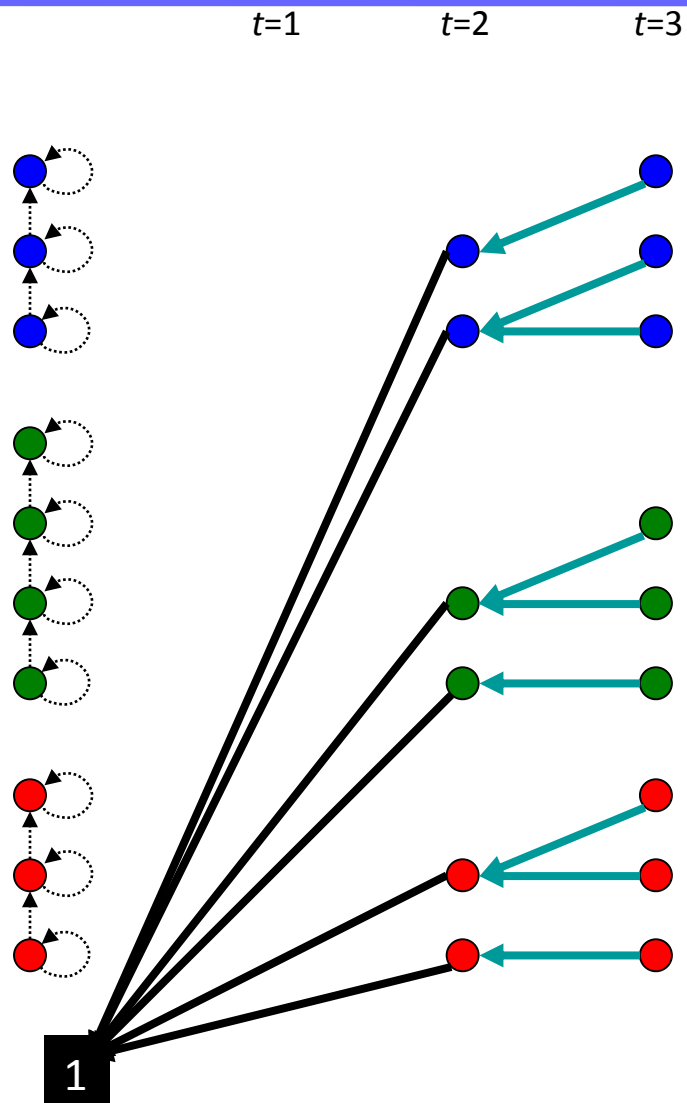
# Trellis with Complete Set of Backpointers



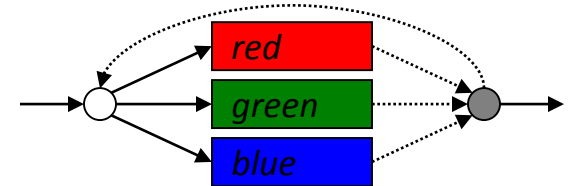
1,  $t=0$ , scr1,  $p=0, \dots$



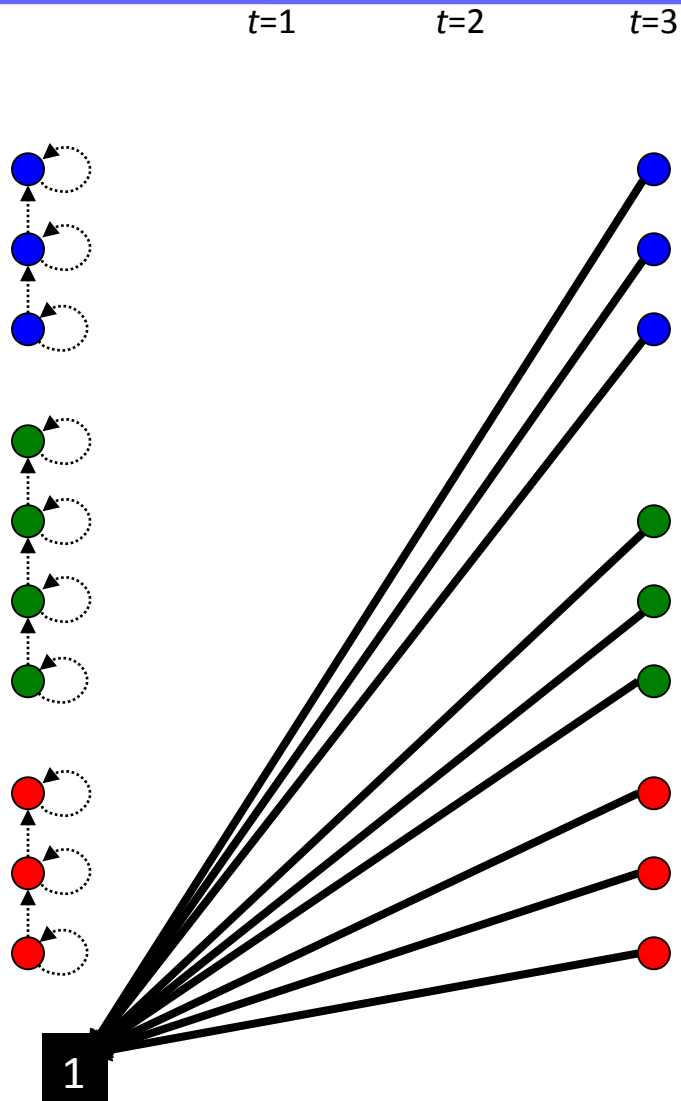
# Trellis with Complete Set of Backpointers



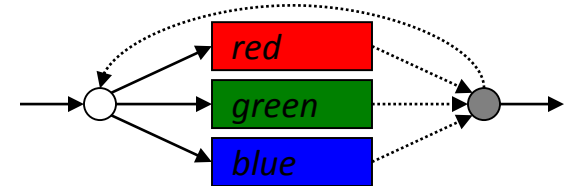
1,  $t=0$ , scr1,  $p=0$ ,...



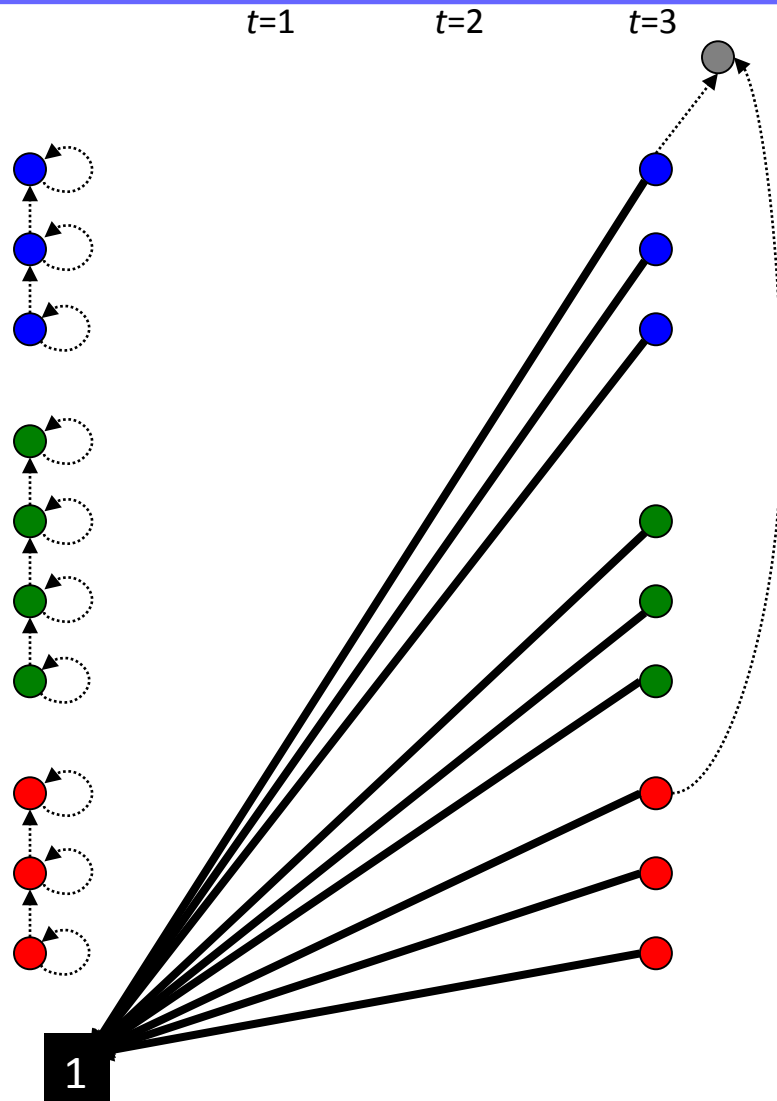
# Trellis with Complete Set of Backpointers



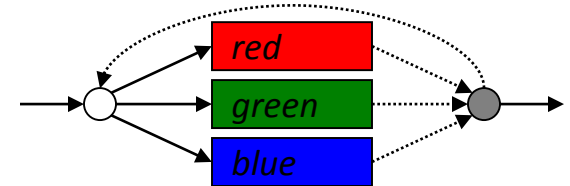
1,  $t=0$ , scr1,  $p=0, \dots$



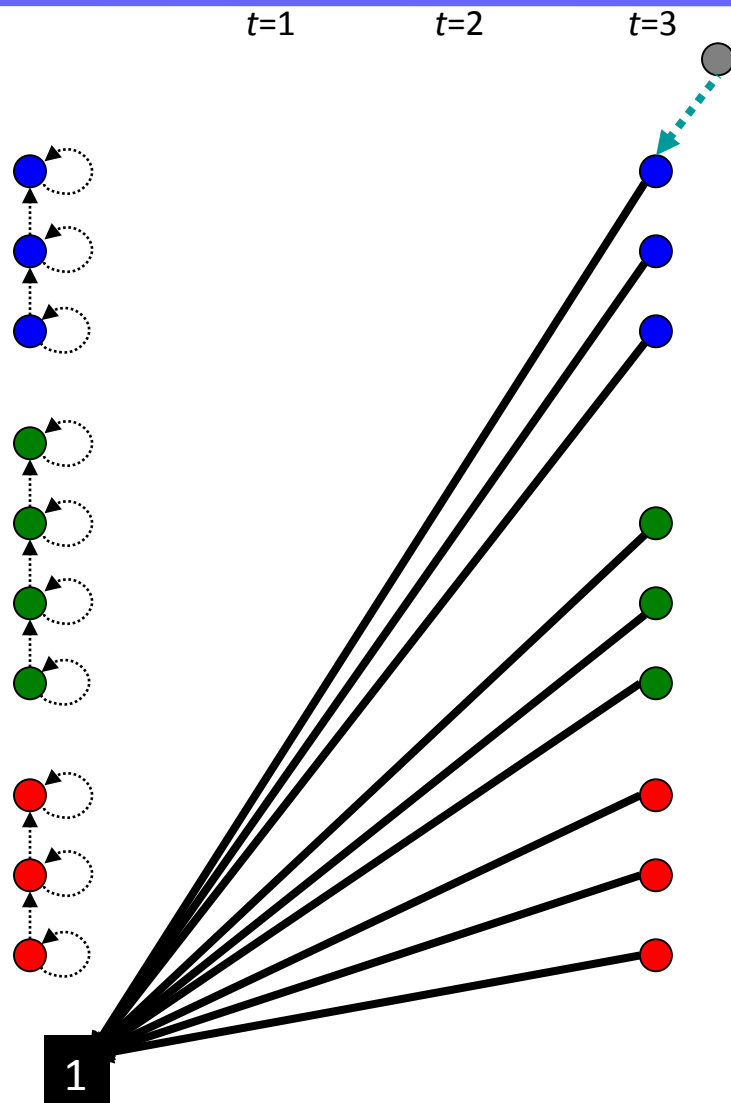
# Trellis with Complete Set of Backpointers



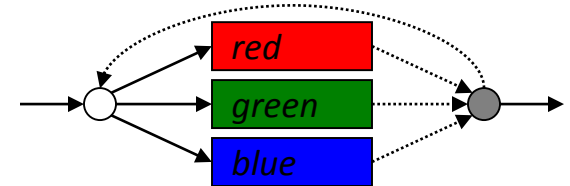
1,  $t=0$ , scr1,  $p=0, \dots$



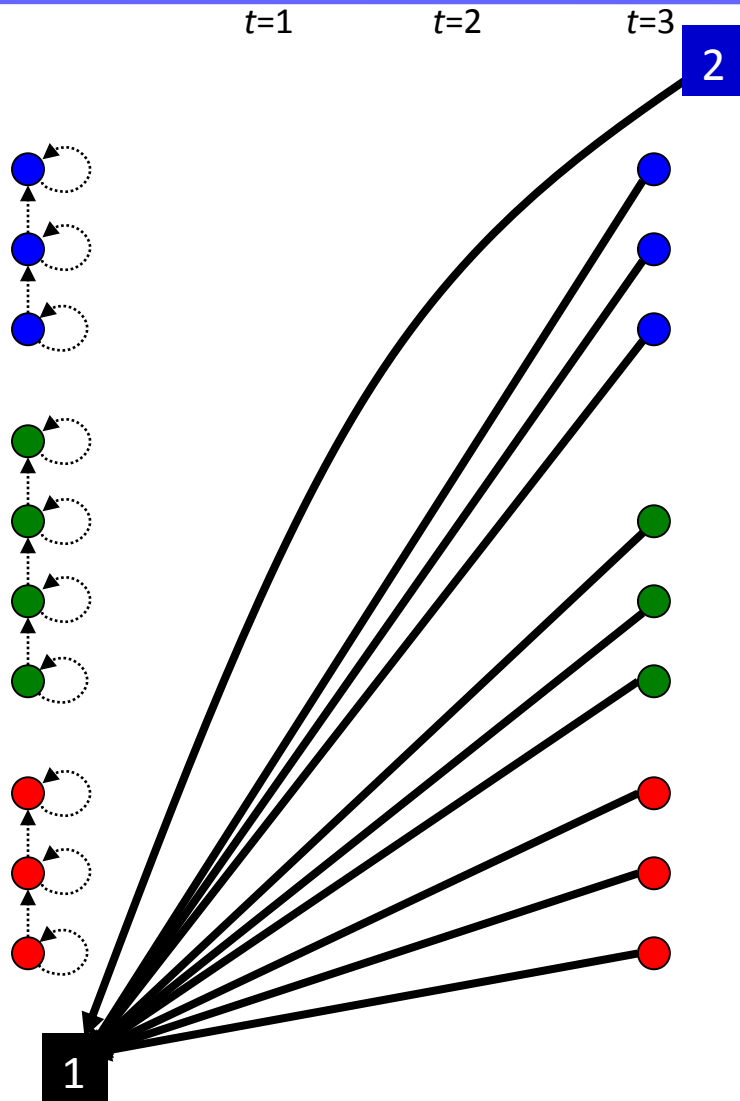
# Trellis with Complete Set of Backpointers



1,  $t=0$ , scr1,  $p=0, \dots$



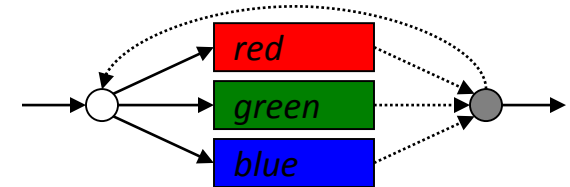
# Trellis with Complete Set of Backpointers



Retain backpointers (and add the to the table)  
if deleting them will result in loss of word  
history

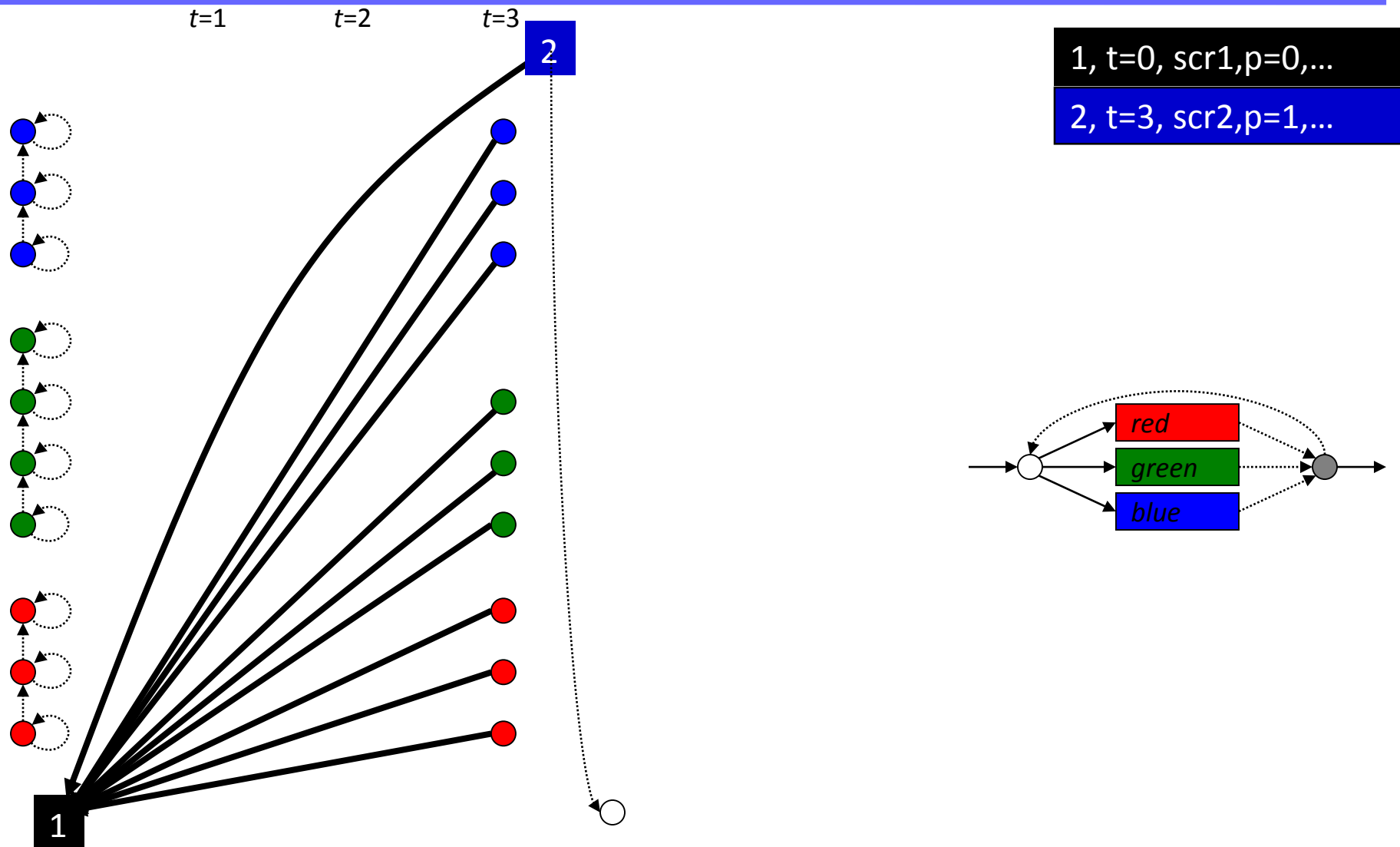
1,  $t=0$ , scr1,  $p=0, \dots$

2,  $t=3$ , scr2,  $p=1, \dots$

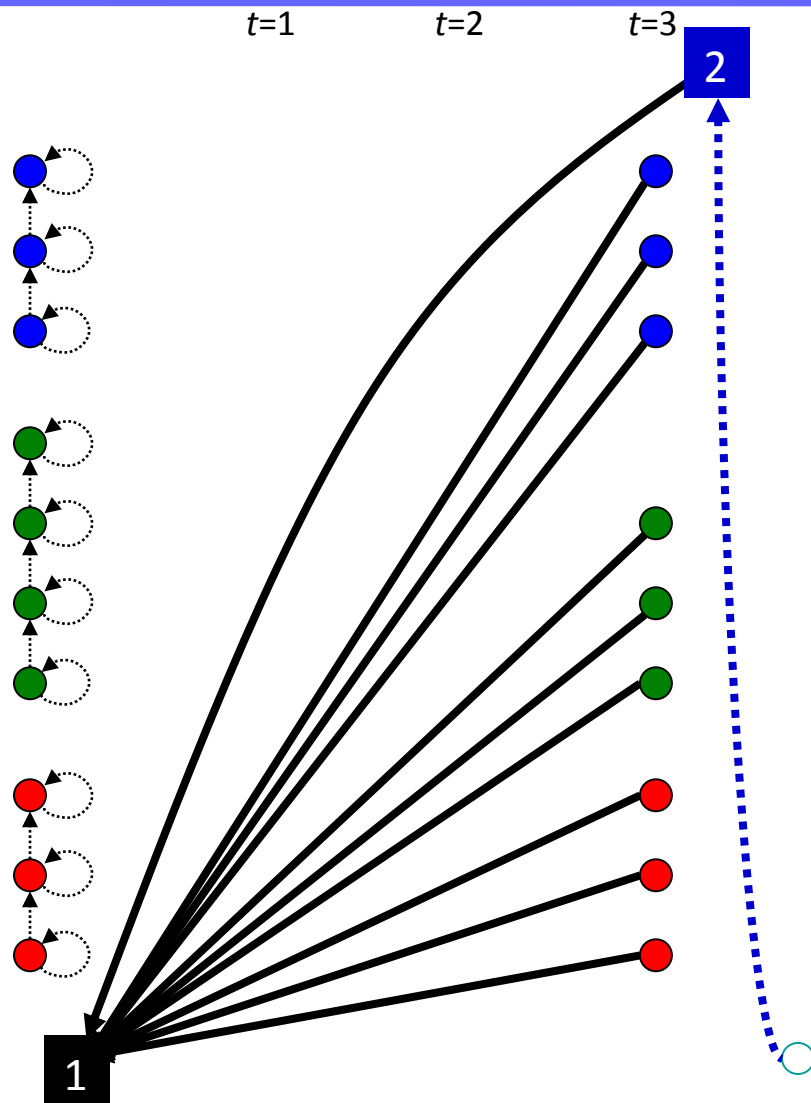




# Trellis with Complete Set of Backpointers

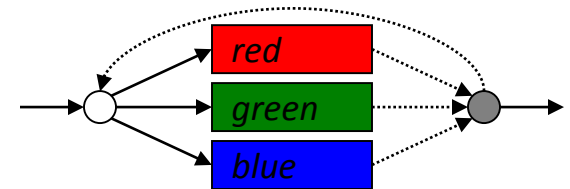


# Trellis with Complete Set of Backpointers

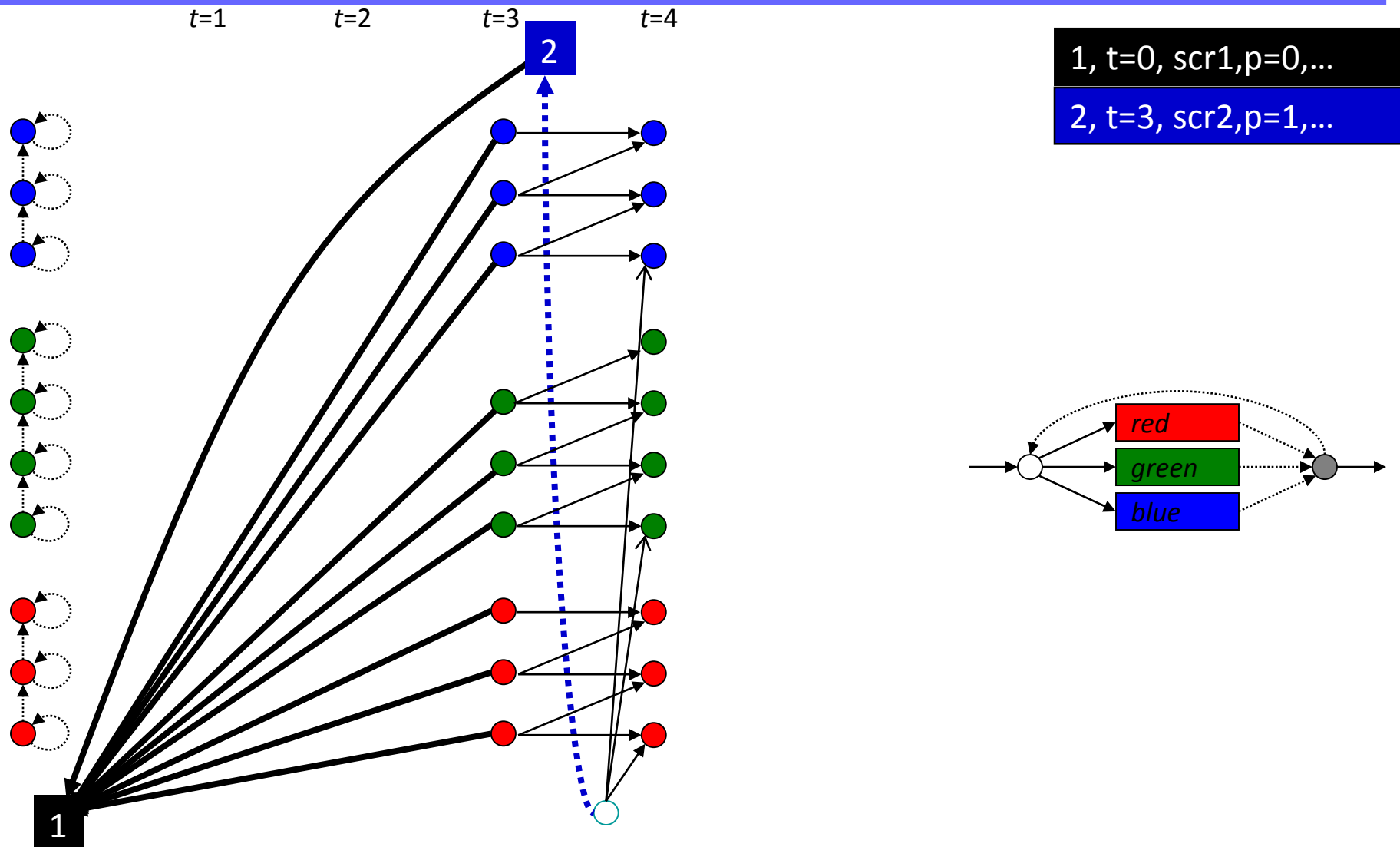


1,  $t=0$ , scr1,  $p=0, \dots$

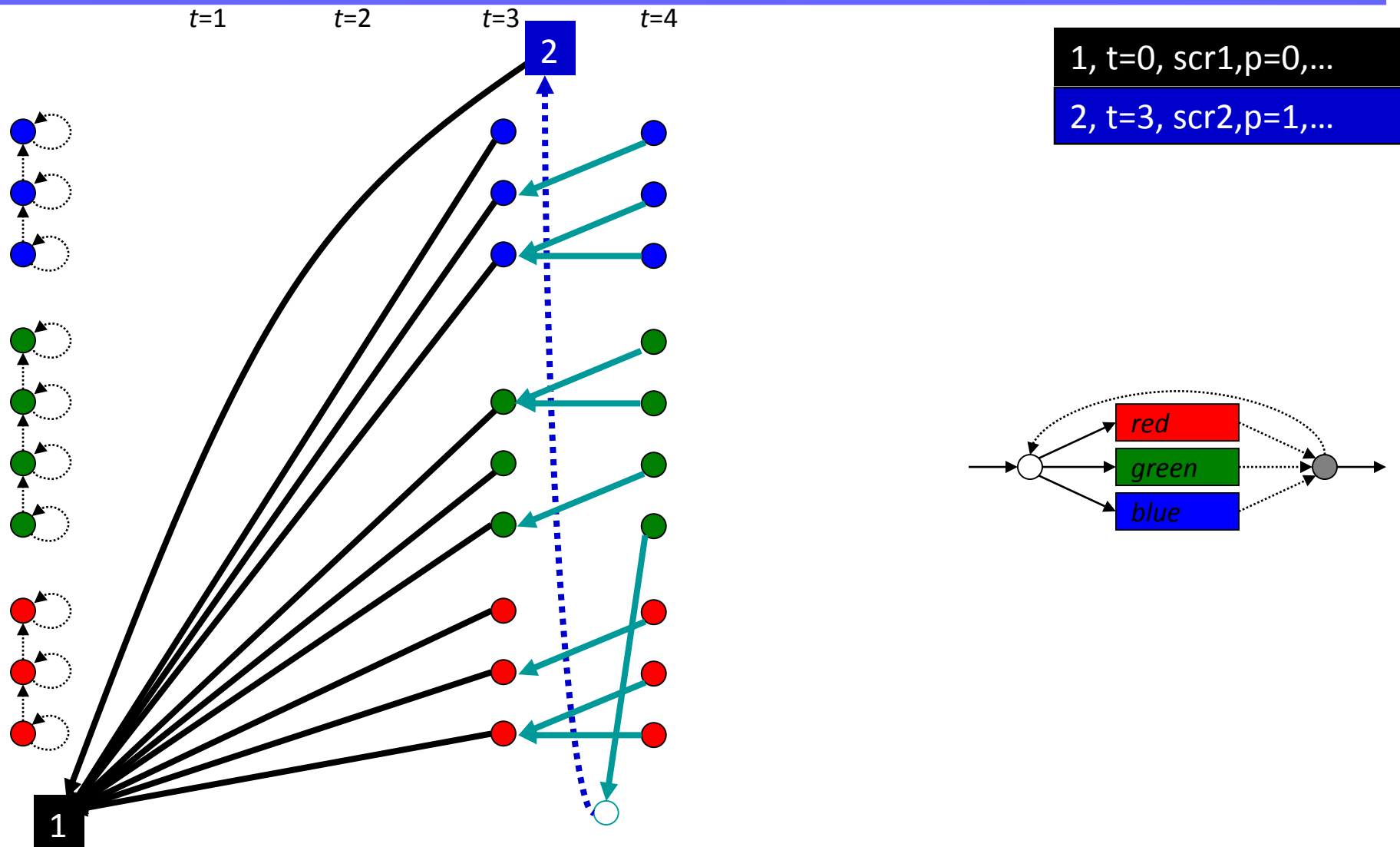
2,  $t=3$ , scr2,  $p=1, \dots$



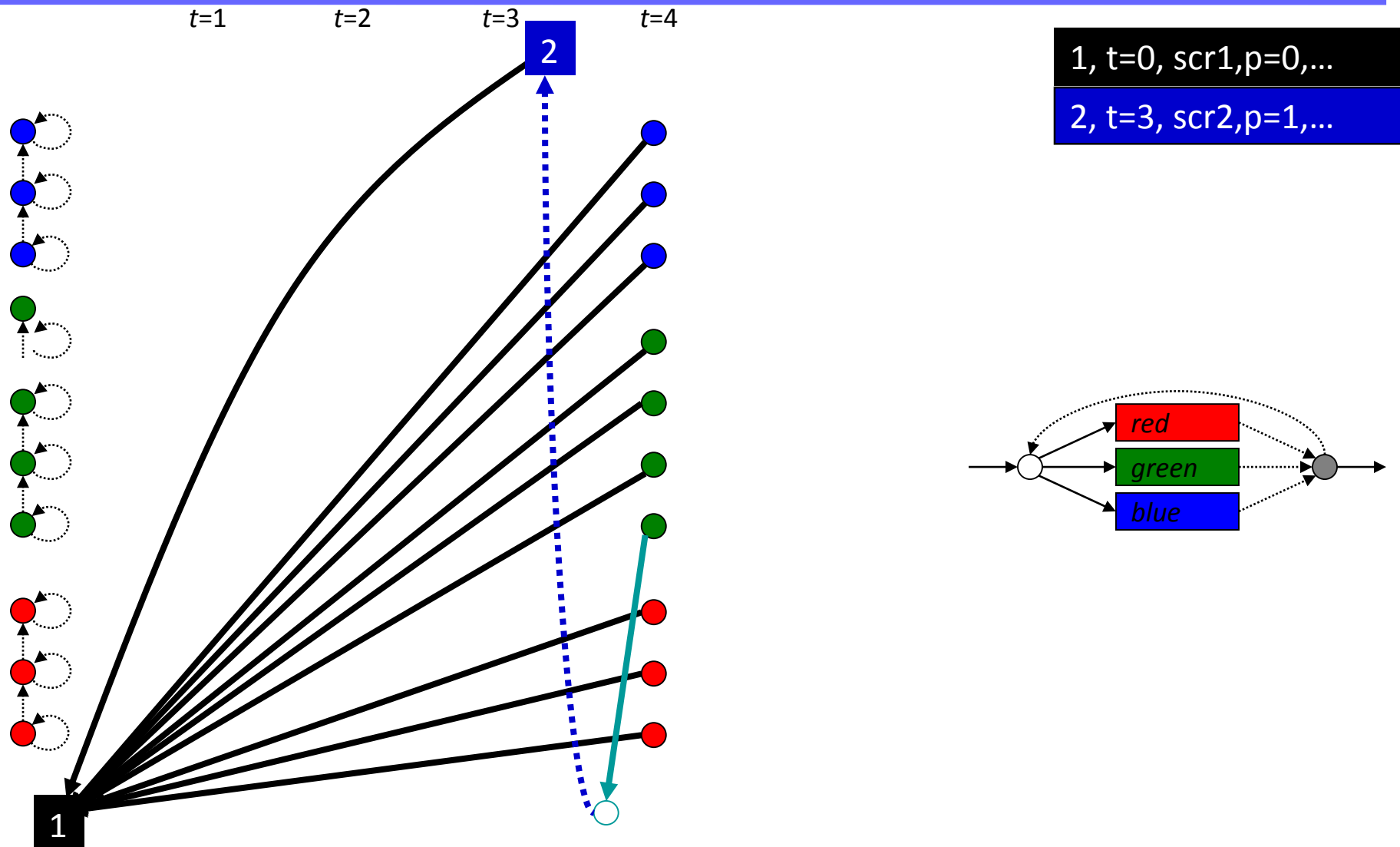
# Trellis with Complete Set of Backpointers



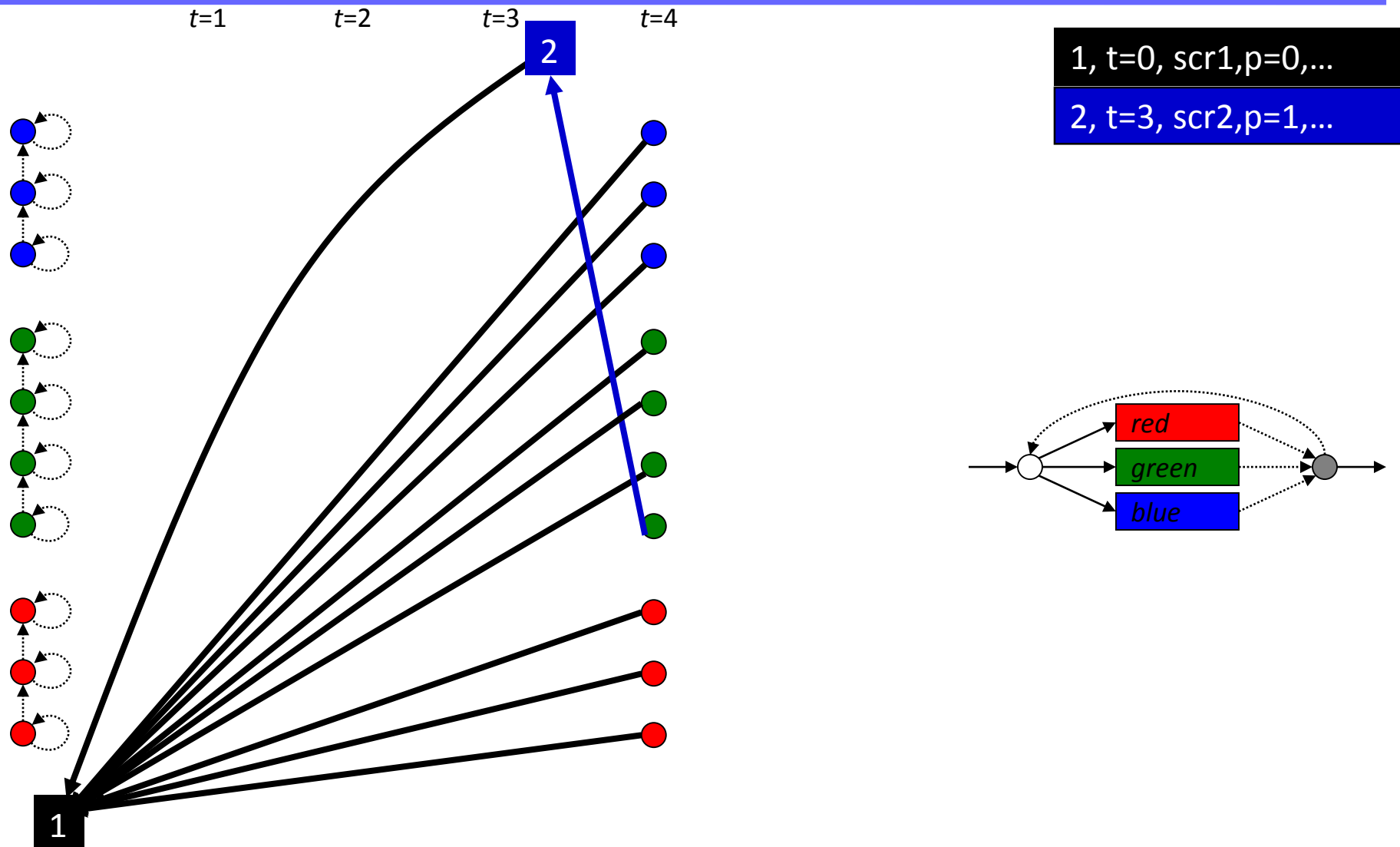
# Trellis with Complete Set of Backpointers



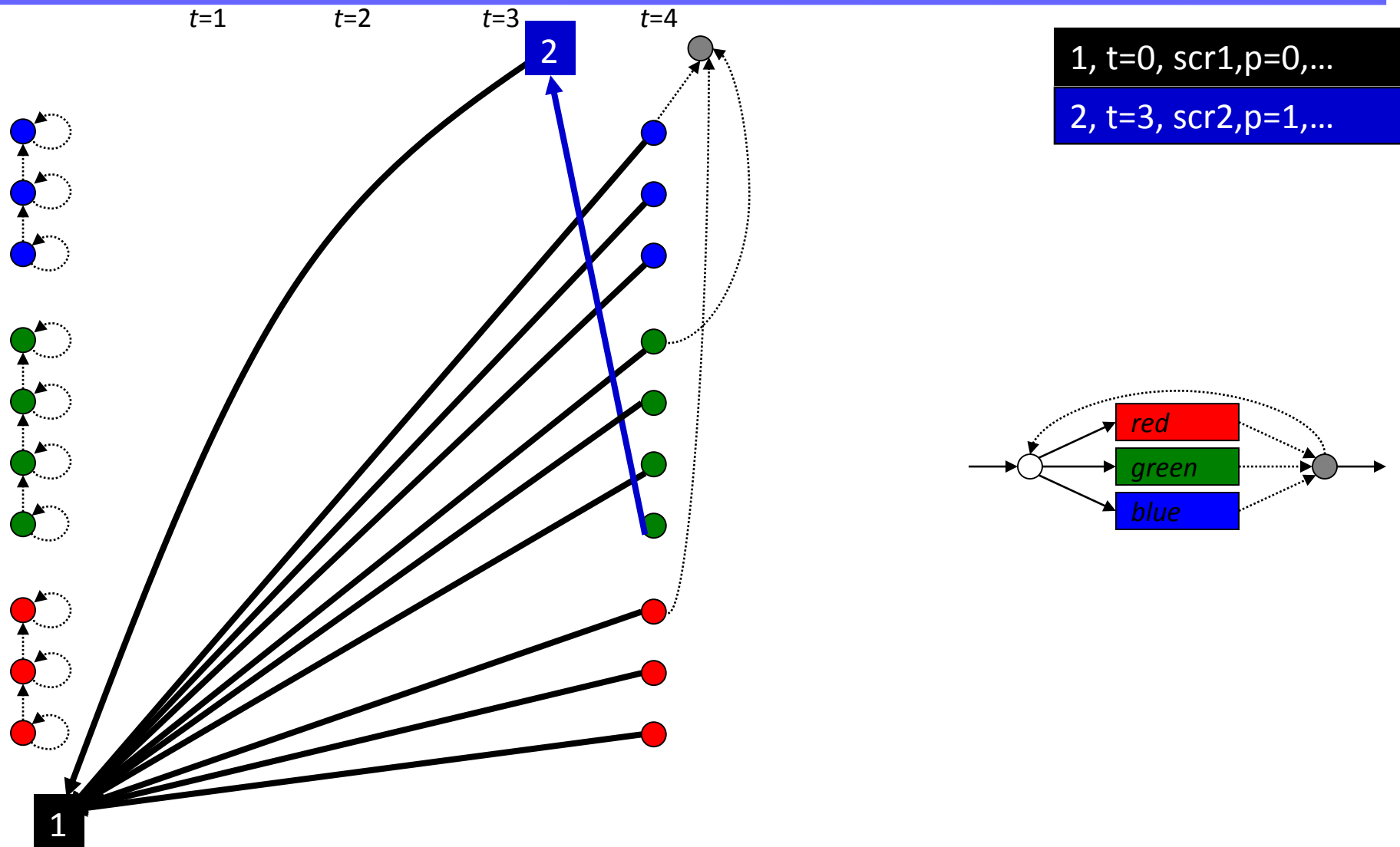
# Trellis with Complete Set of Backpointers



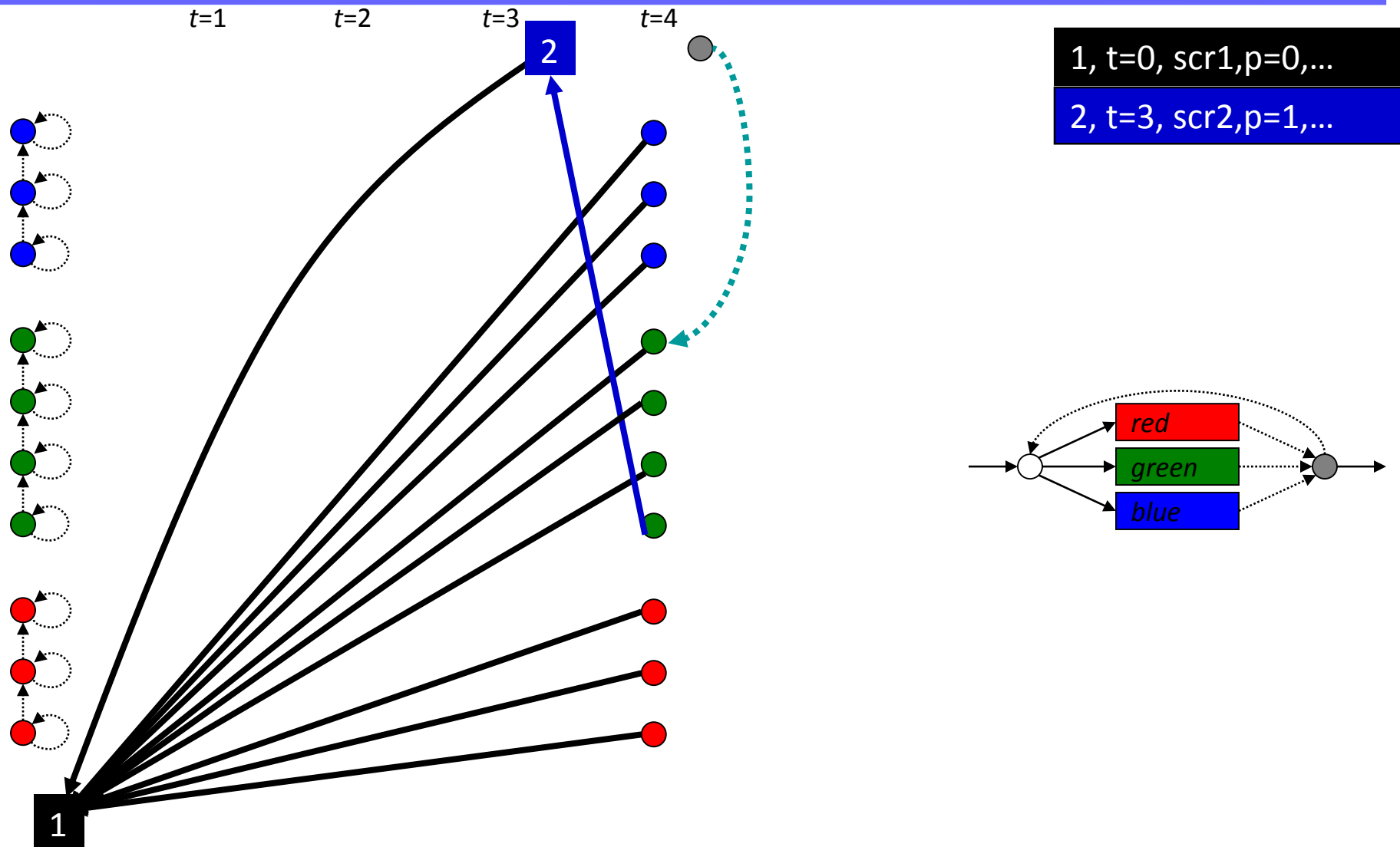
# Trellis with Complete Set of Backpointers



# Trellis with Complete Set of Backpointers

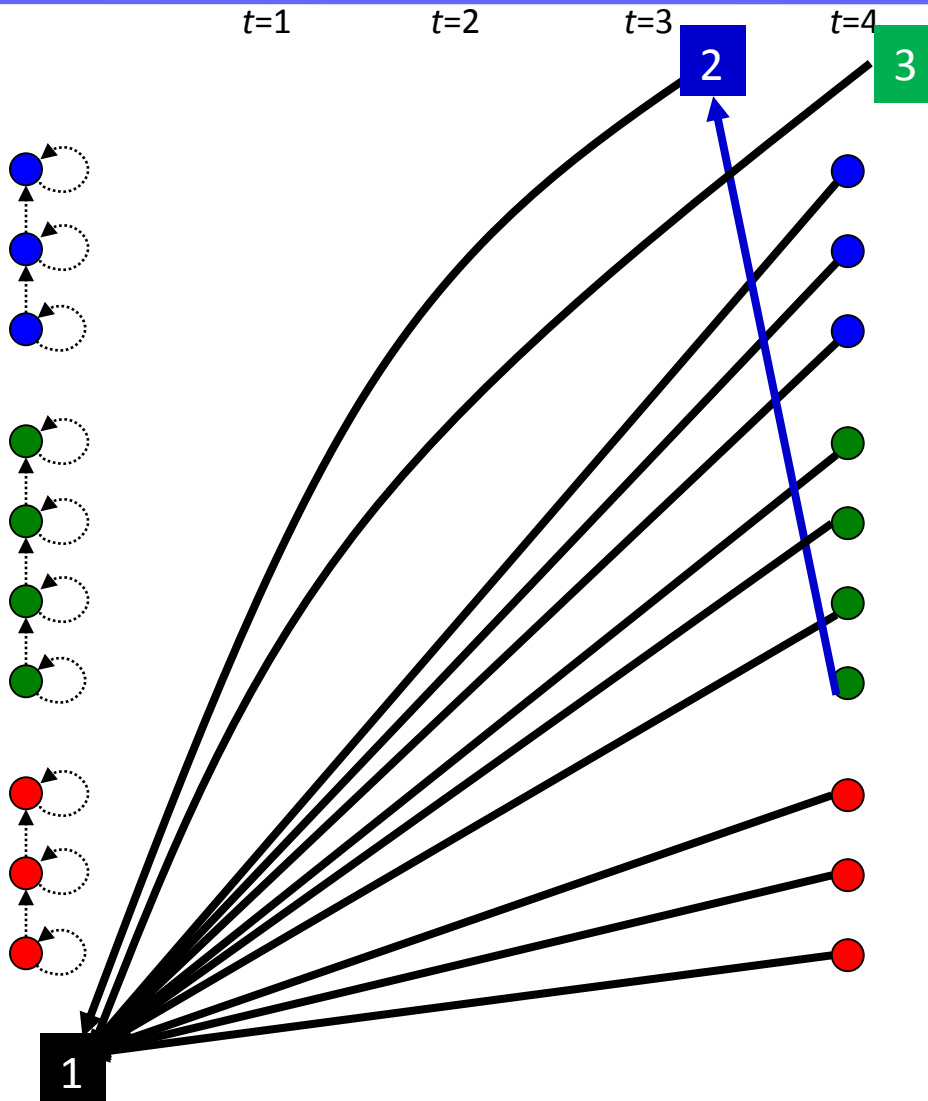


# Trellis with Complete Set of Backpointers





# Trellis with Complete Set of Backpointers

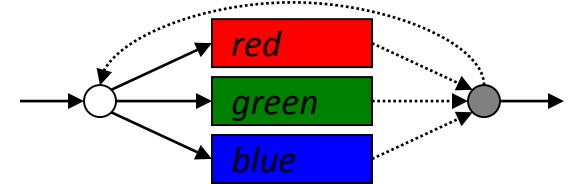


1,  $t=0$ , scr1,  $p=0$ ,...

2,  $t=3$ , scr2,  $p=1$ ,...

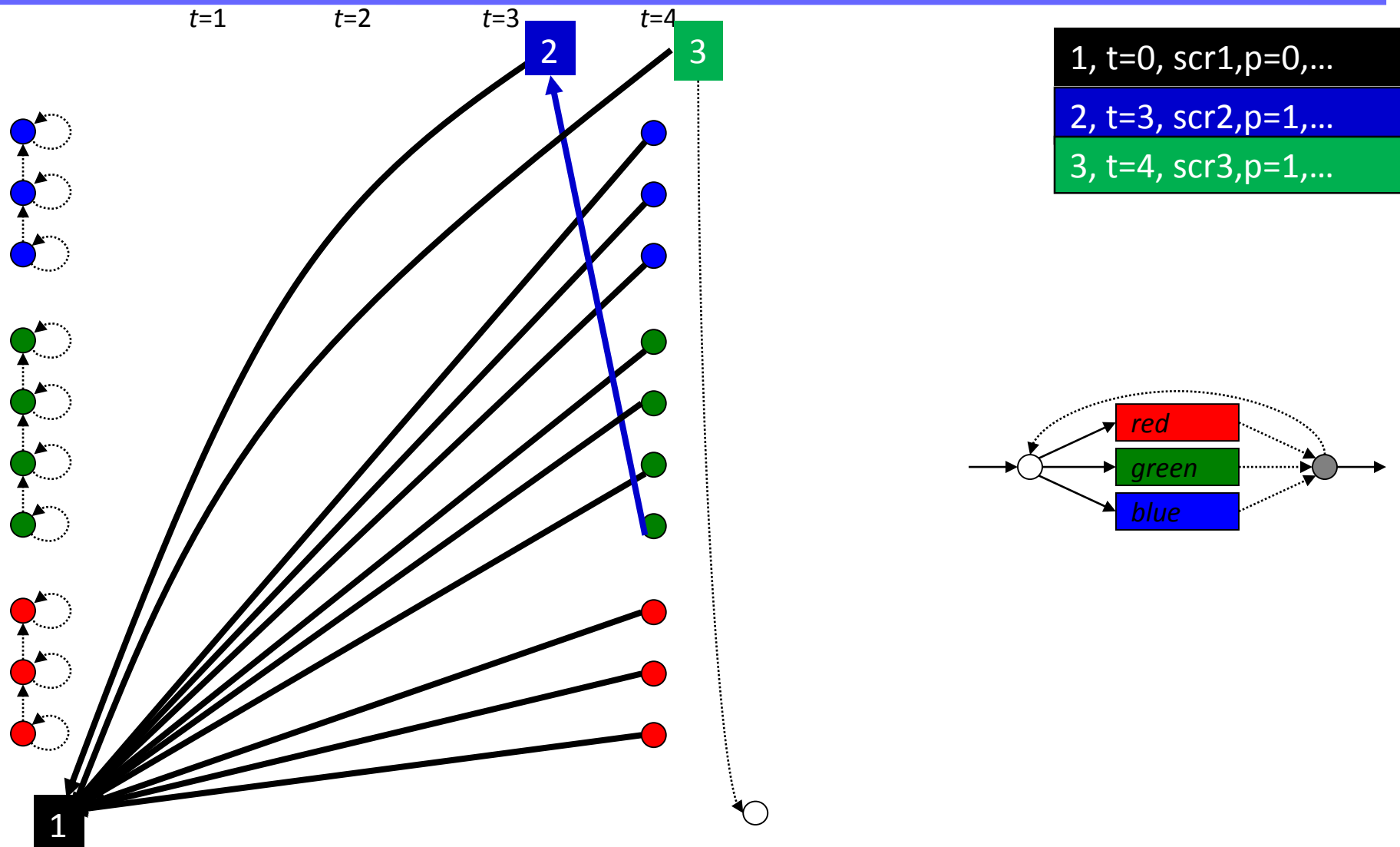
3,  $t=4$ , scr3,  $p=1$ ,...

Retain backpointers (and add the to the table) if deleting them will result in loss of word history

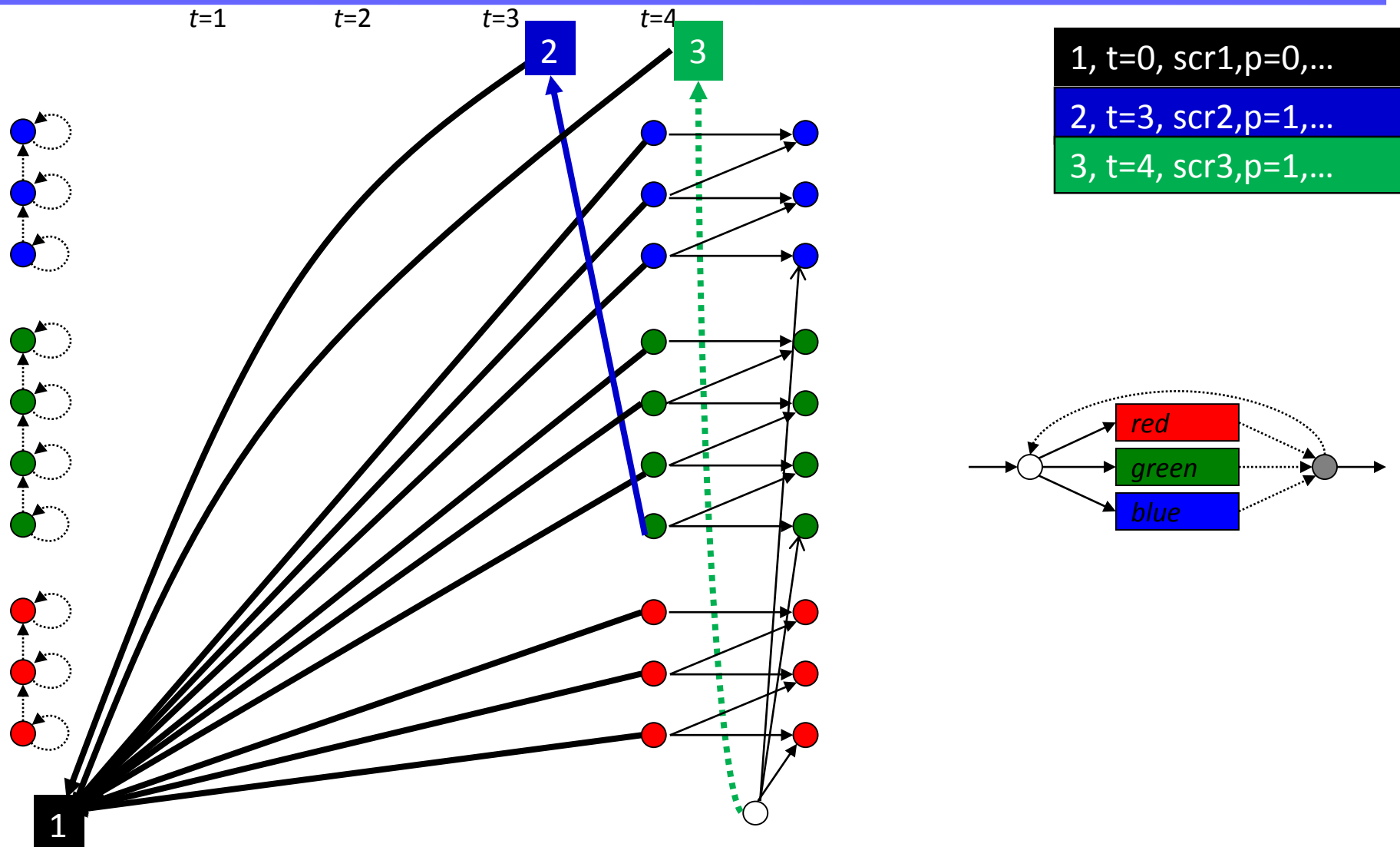


Backpointer table entries also have information about word identity (indicated by color in the figure)

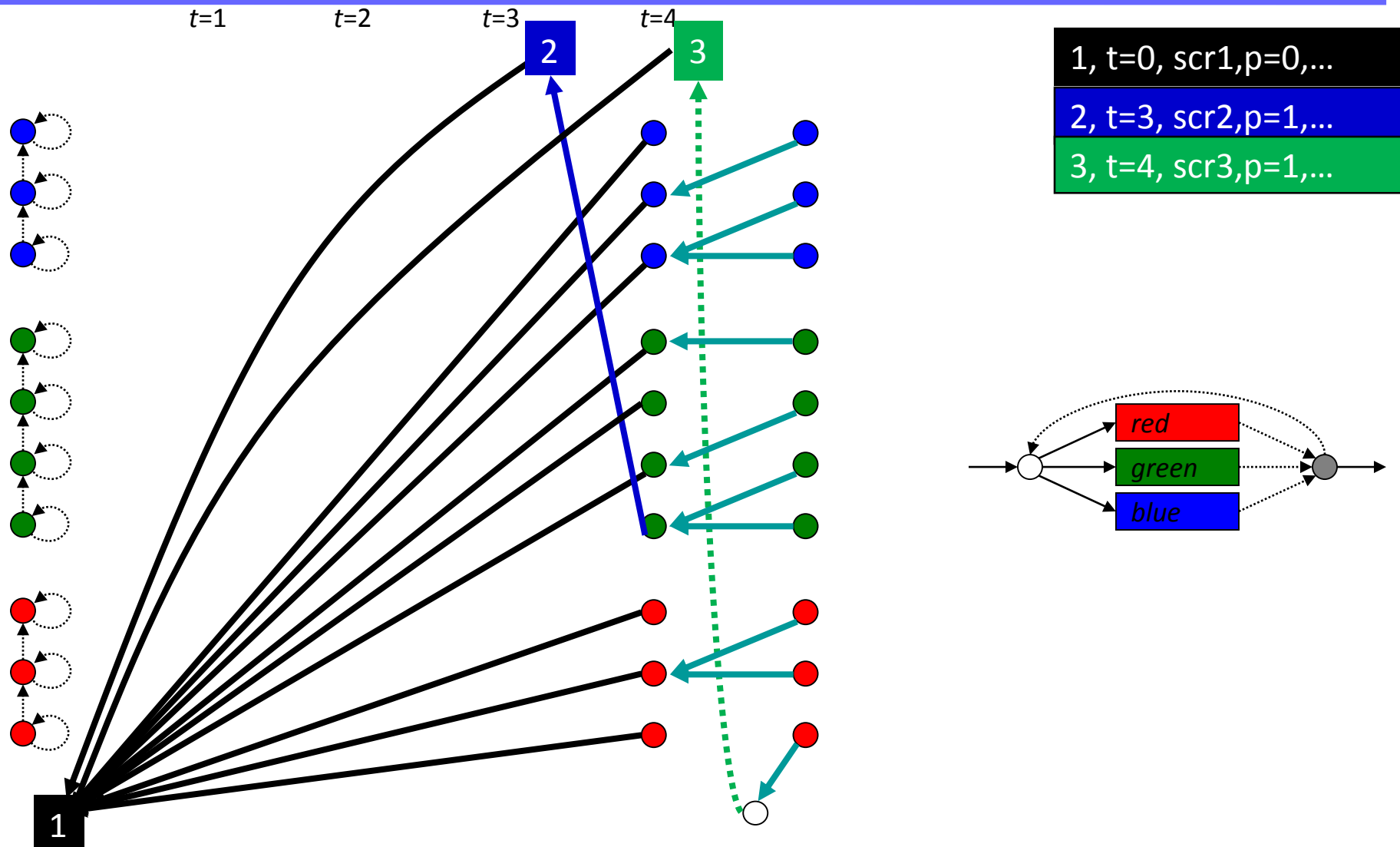
# Trellis with Complete Set of Backpointers



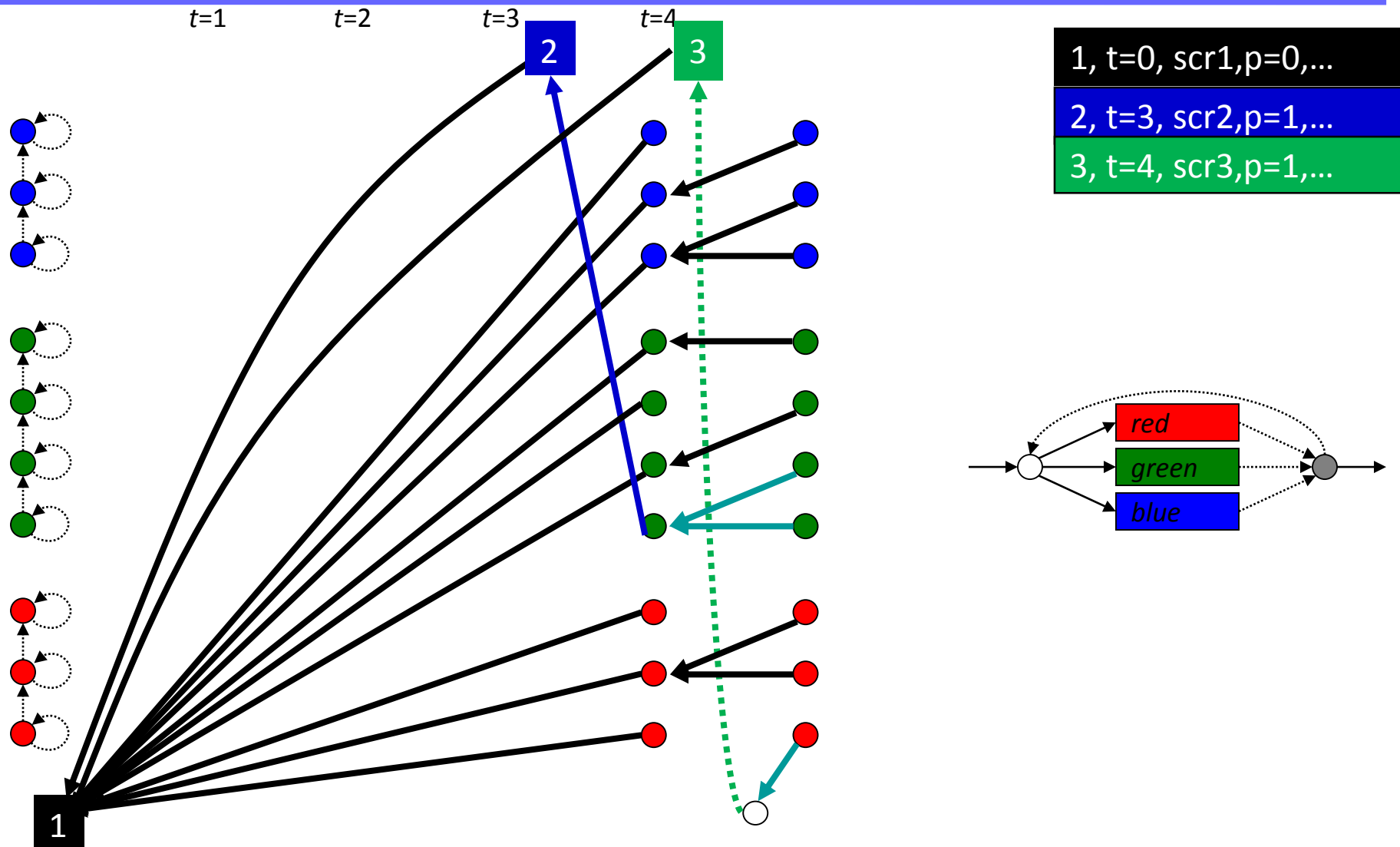
# Trellis with Complete Set of Backpointers



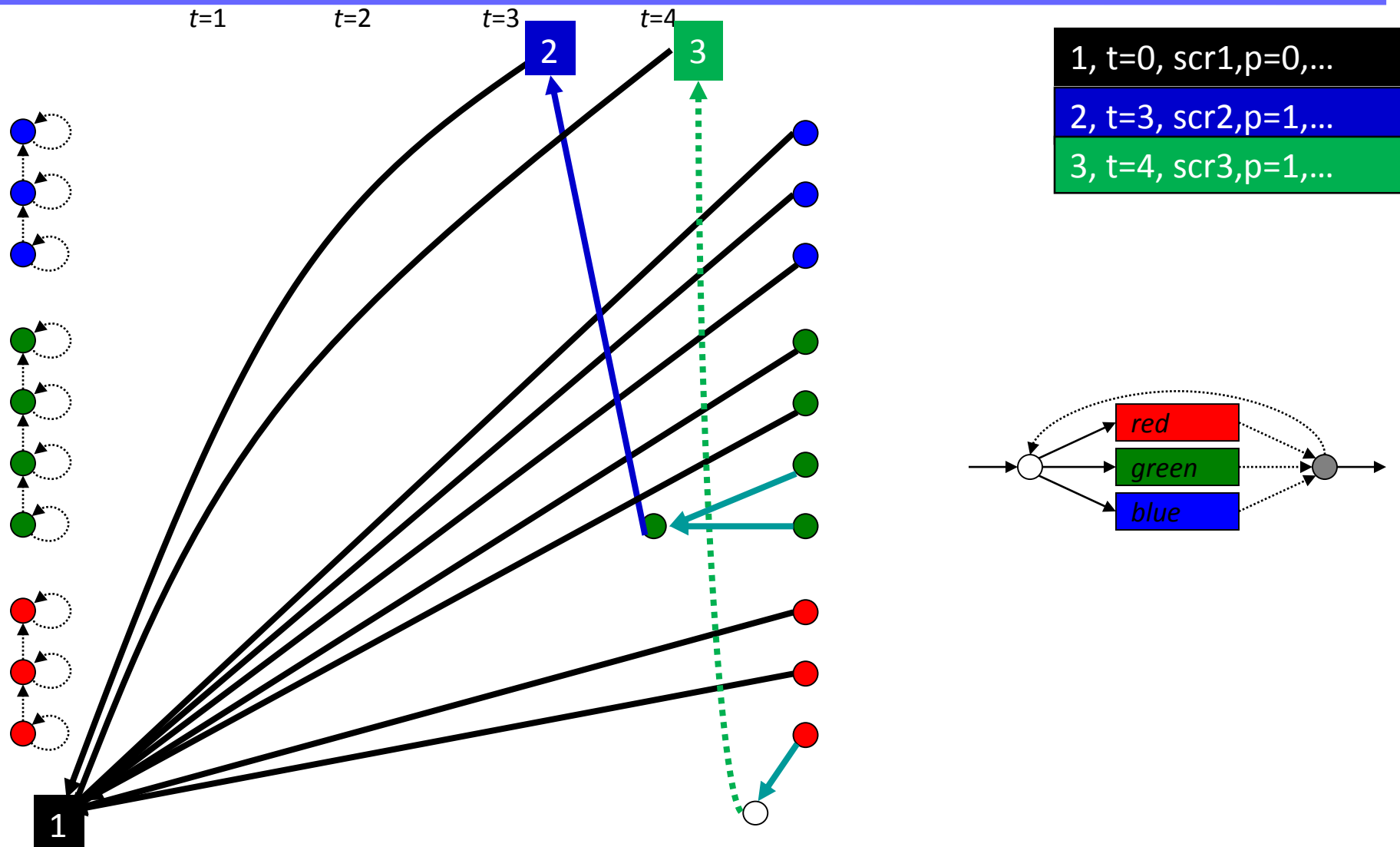
# Trellis with Complete Set of Backpointers



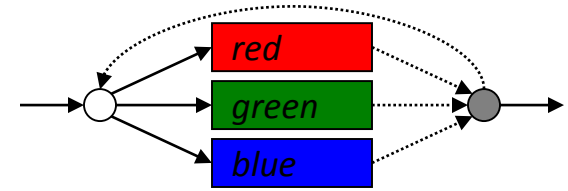
# Trellis with Complete Set of Backpointers



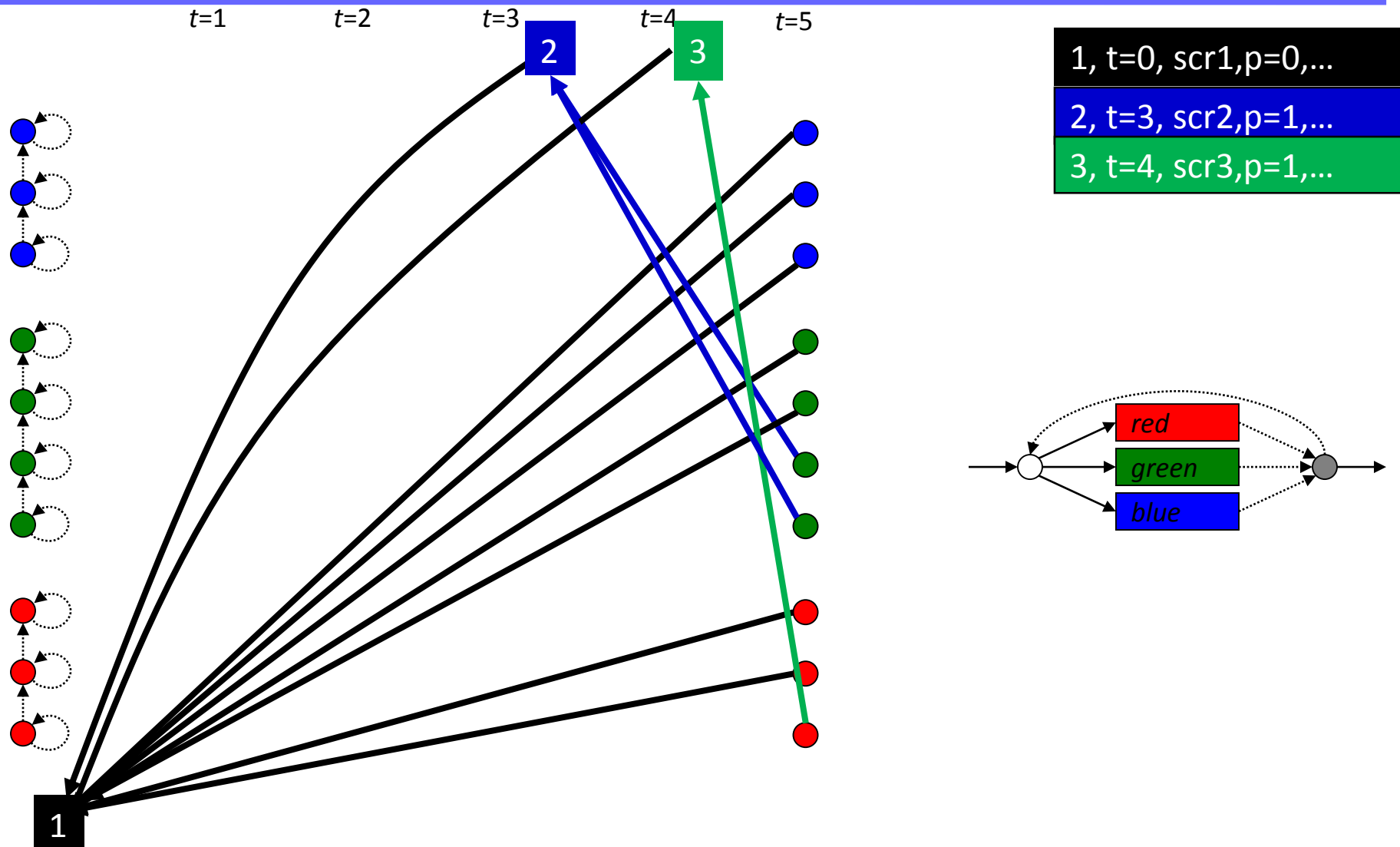
# Trellis with Complete Set of Backpointers



The diagram illustrates a multi-stage process with nodes and transitions. On the left, a vertical column of nodes is shown, grouped by color: blue (top), green (middle), and red (bottom). Each group has a self-loop arrow. A large black node labeled '1' is at the bottom left. Arrows originate from node '1' and point to nodes in the subsequent stages. The stages are labeled  $t=1$ ,  $t=2$ ,  $t=3$ , and  $t=4$  at the top. At  $t=3$ , there is a blue node labeled '2'. At  $t=4$ , there is a green node labeled '3'. A blue arrow points from node '2' to a node in the red group at  $t=4$ . A green dashed arrow points from node '3' to a white node at the bottom right. A legend on the right shows three colored boxes: black (1,  $t=0$ , scr1,  $p=0, \dots$ ), blue (2,  $t=3$ , scr2,  $p=1, \dots$ ), and green (3,  $t=4$ , scr3,  $p=1, \dots$ ). Below the legend, a small diagram shows a white node branching into three colored boxes (red, green, blue), which then converge into a gray node.

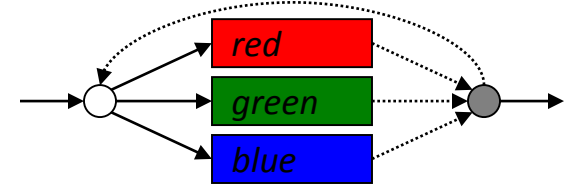
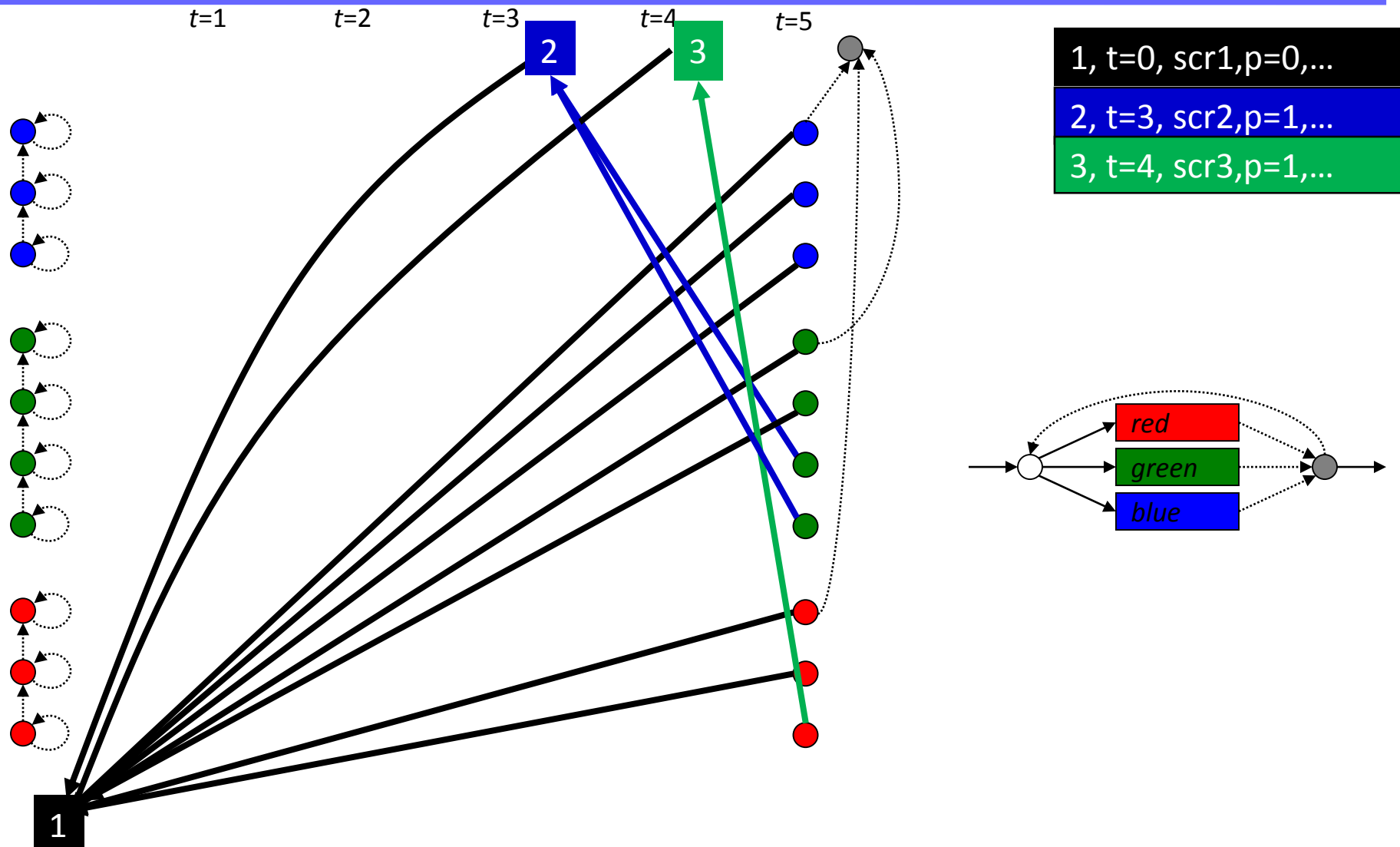


# Trellis with Complete Set of Backpointers

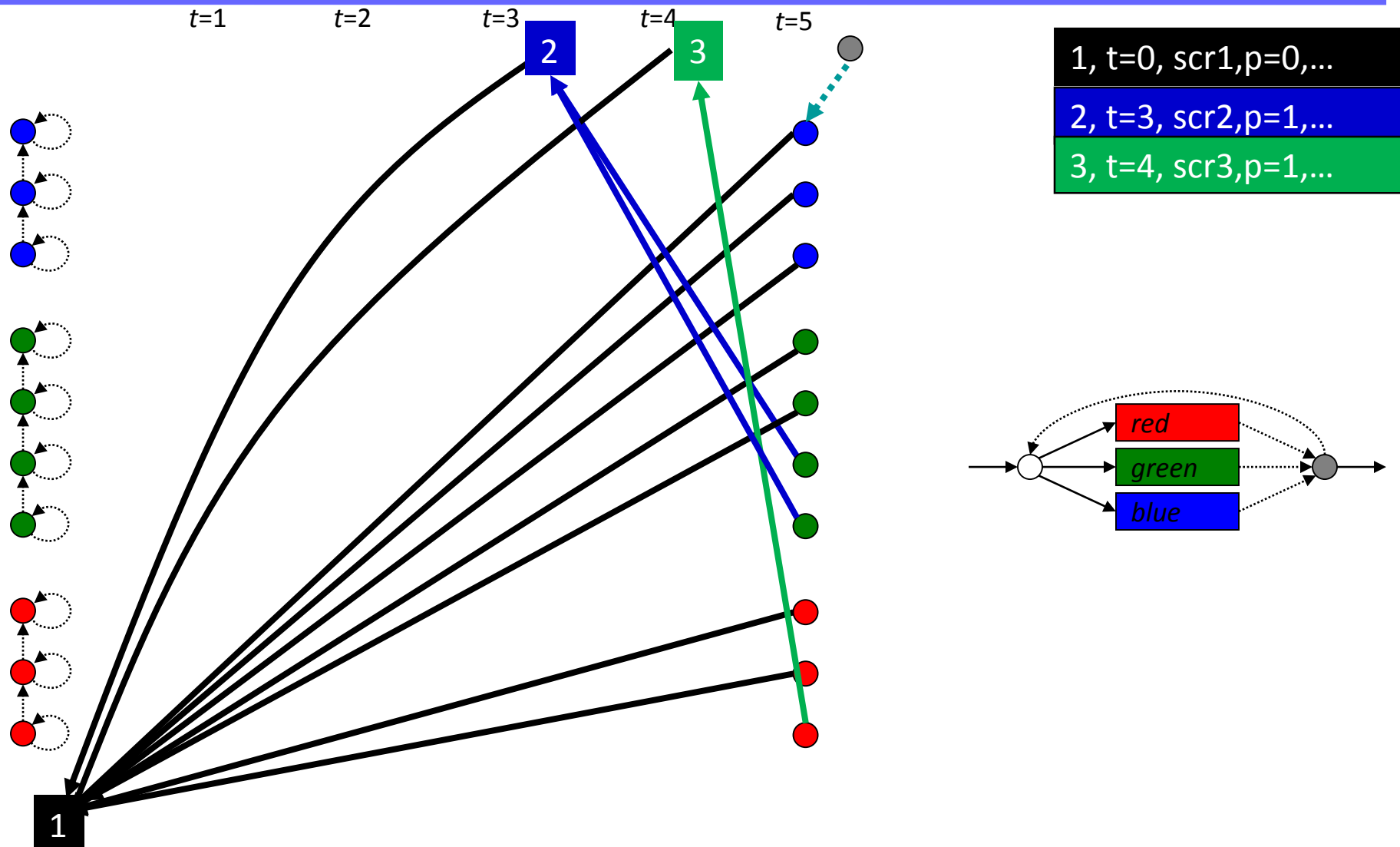




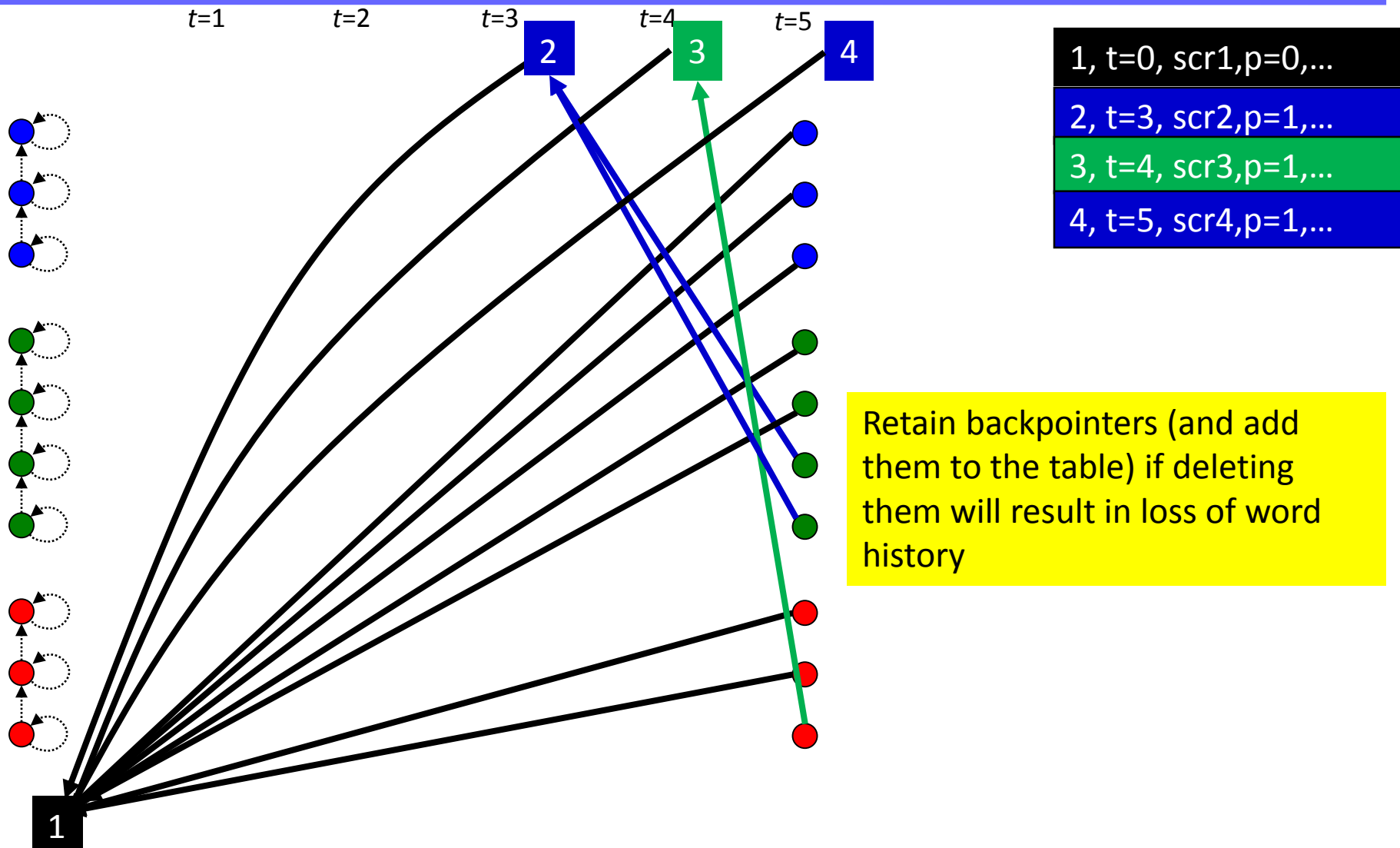
# Trellis with Complete Set of Backpointers



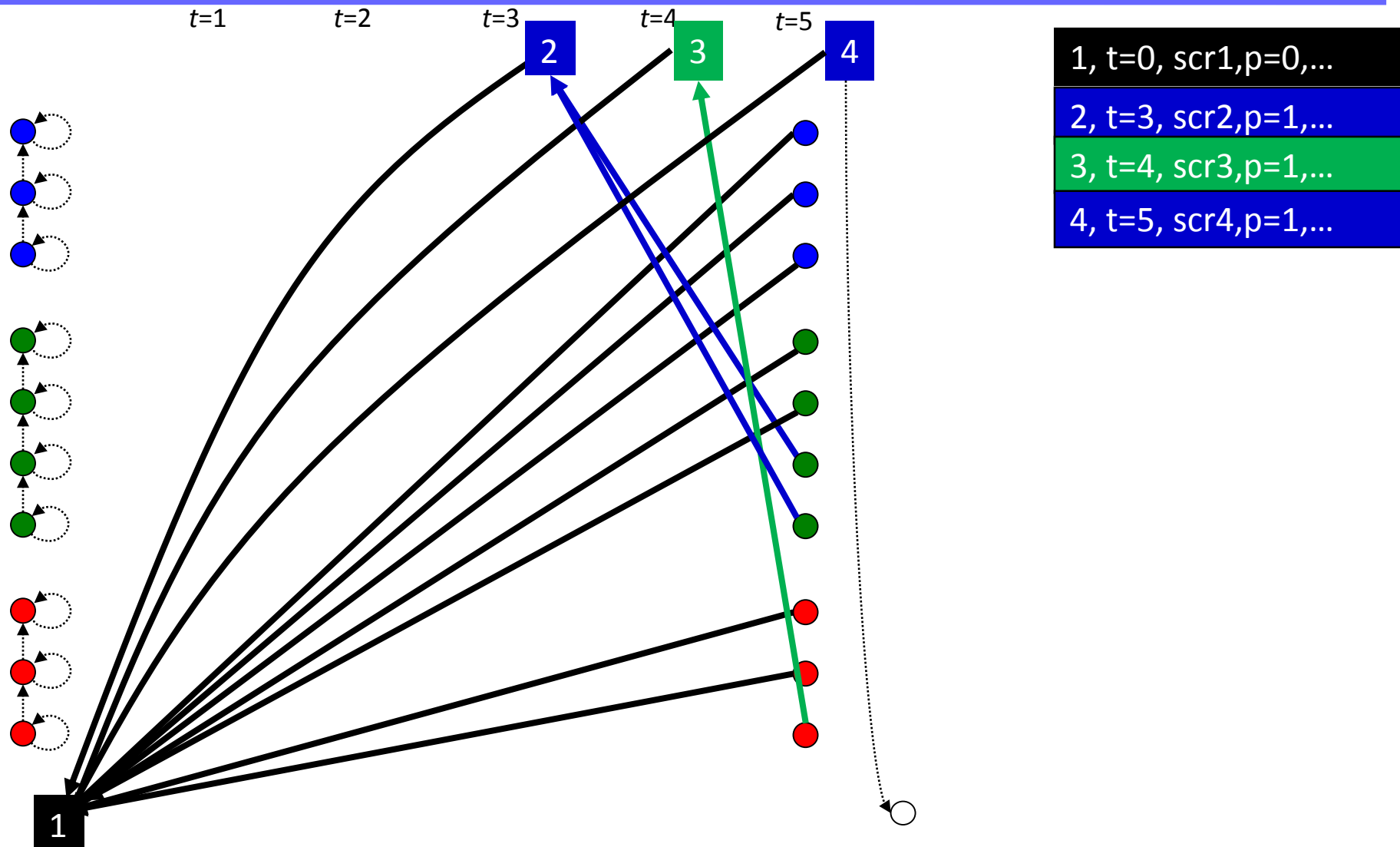
# Trellis with Complete Set of Backpointers



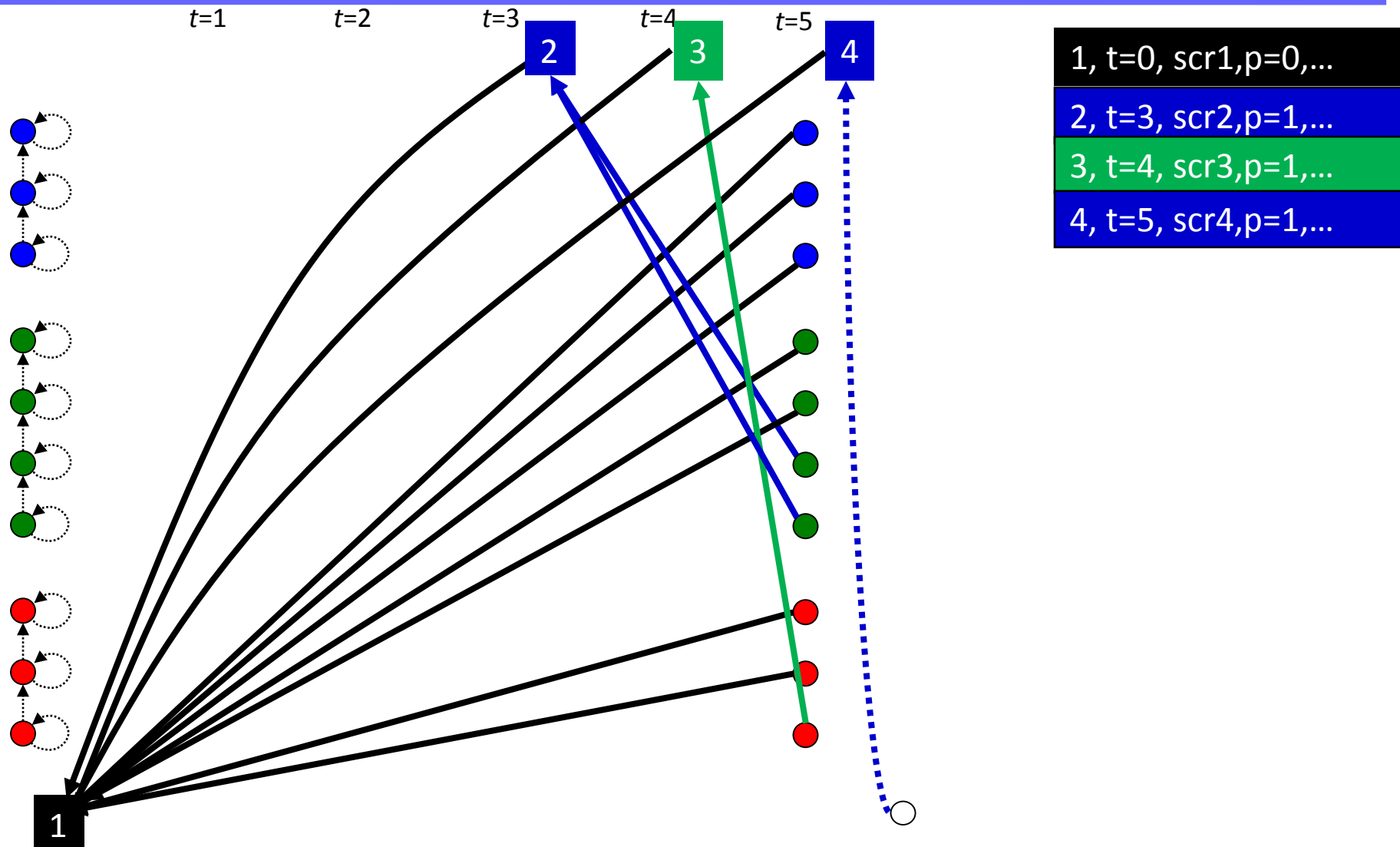
# Trellis with Complete Set of Backpointers



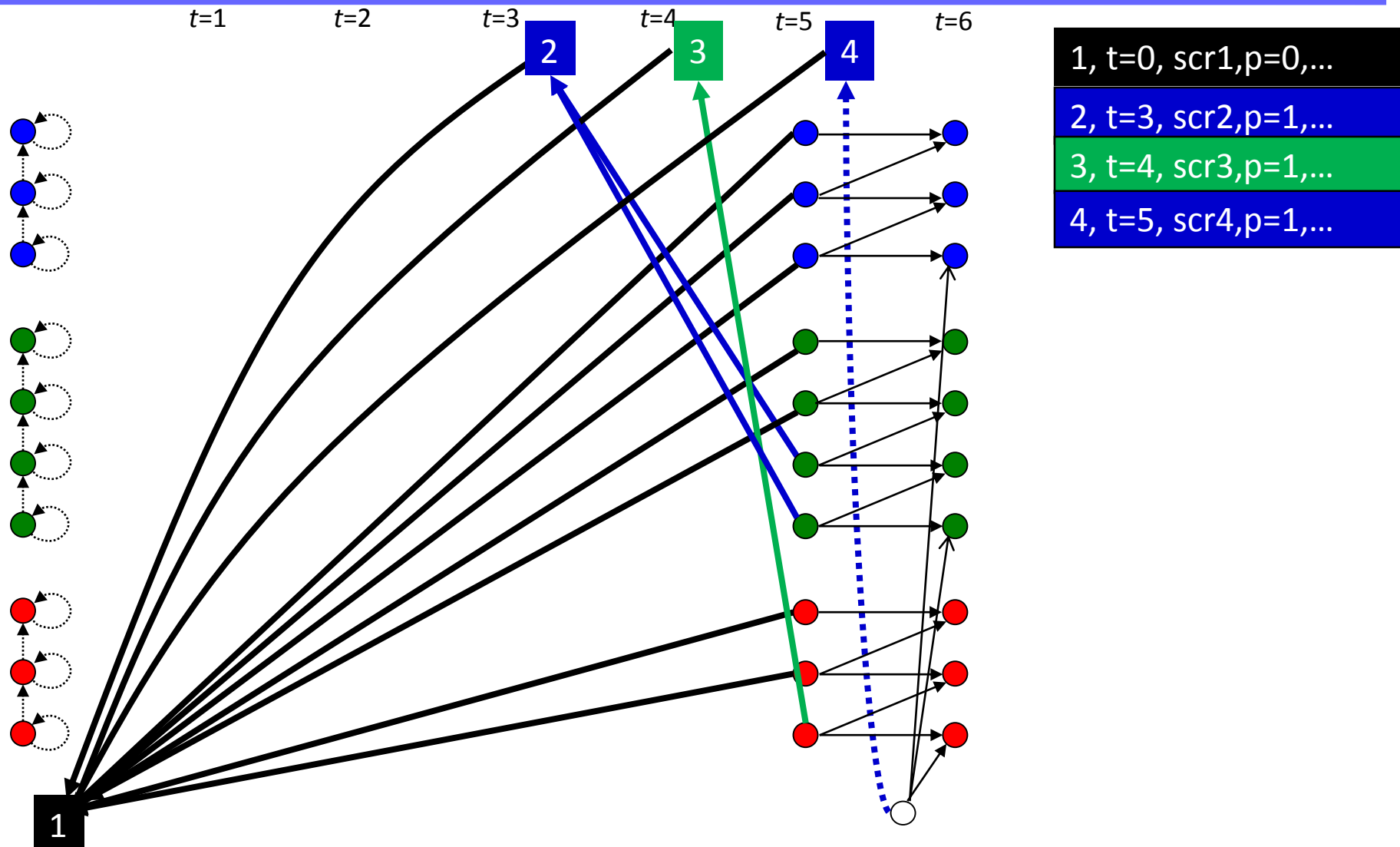
# Trellis with Complete Set of Backpointers



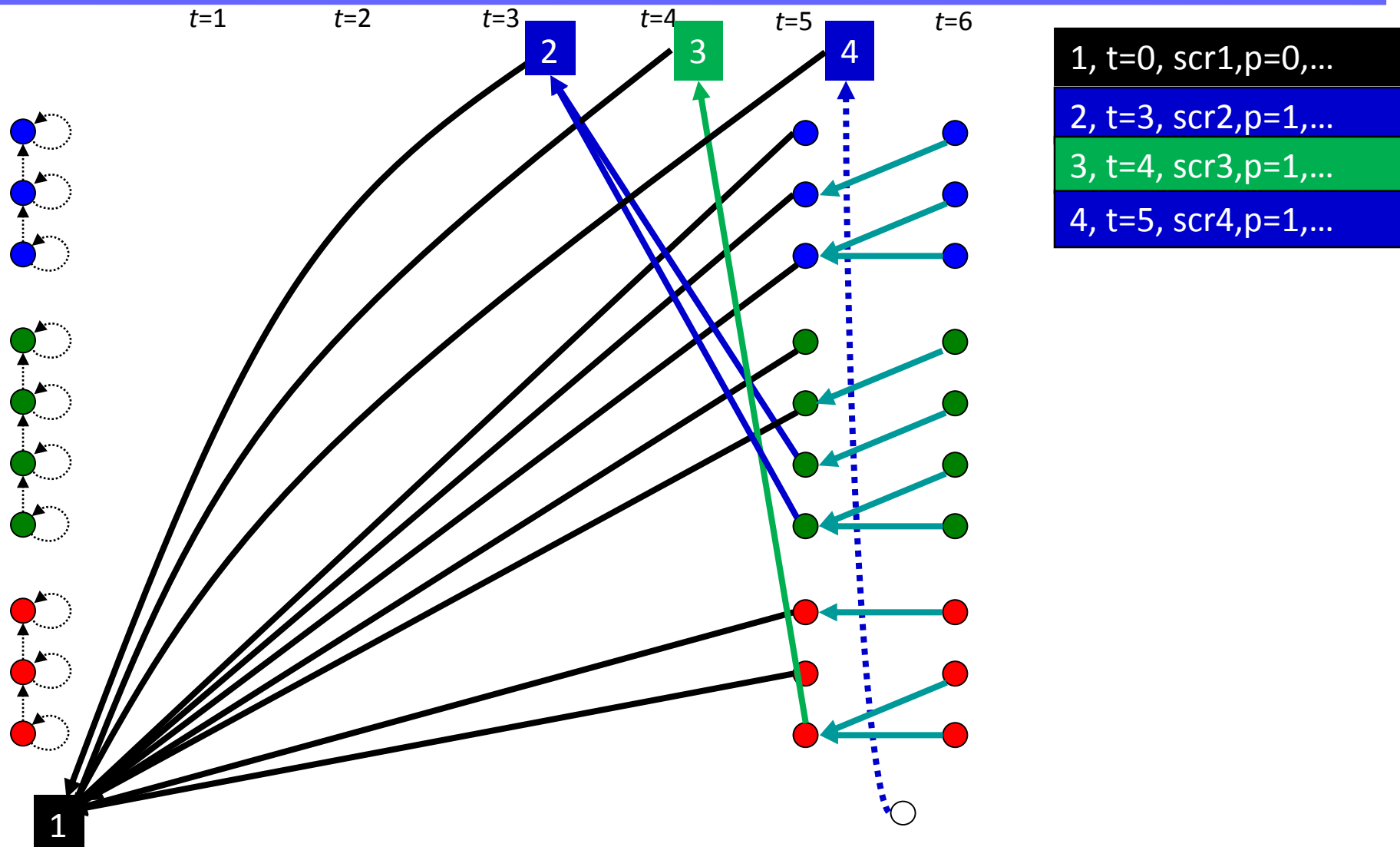
# Trellis with Complete Set of Backpointers



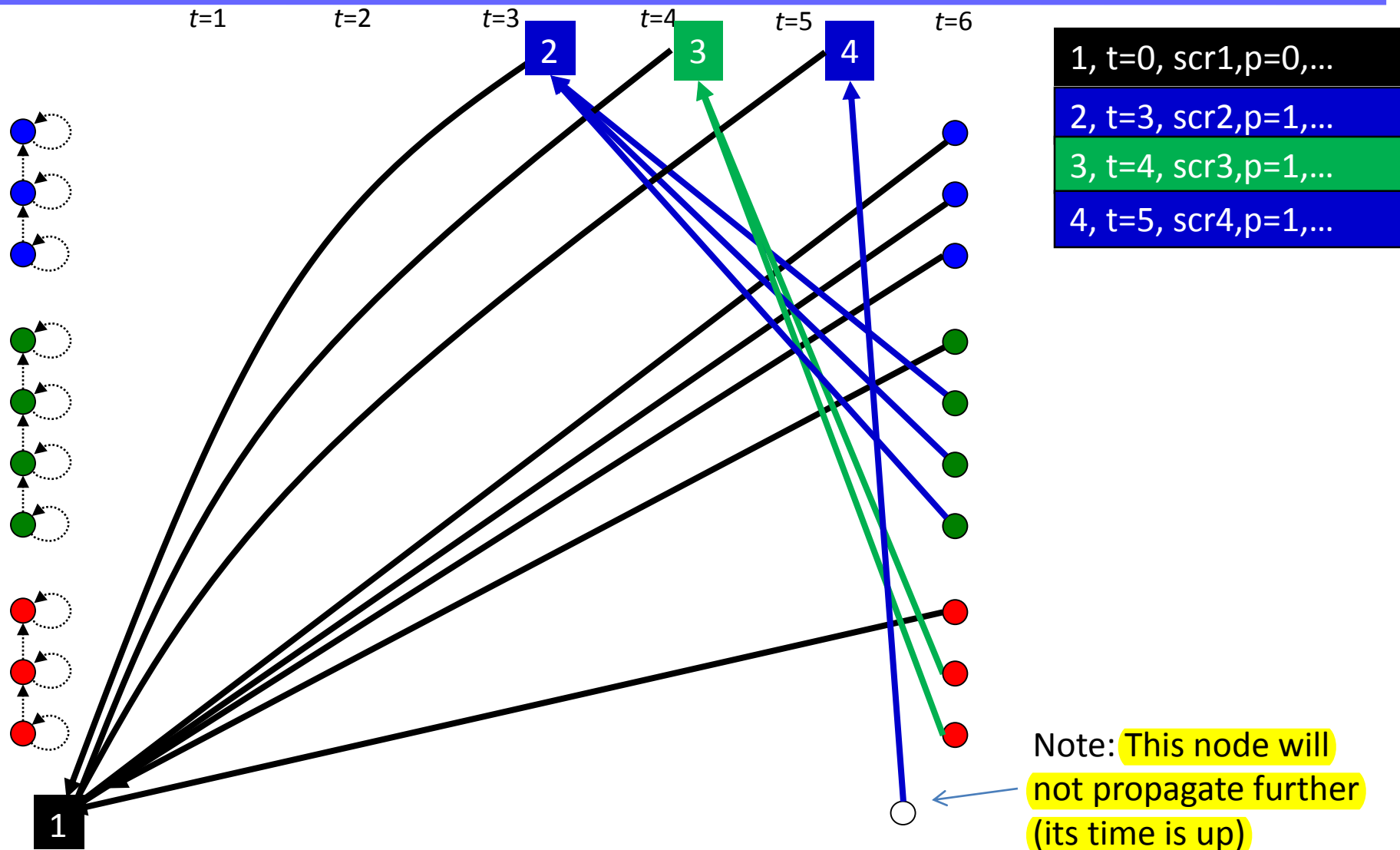
# Trellis with Complete Set of Backpointers



# Trellis with Complete Set of Backpointers

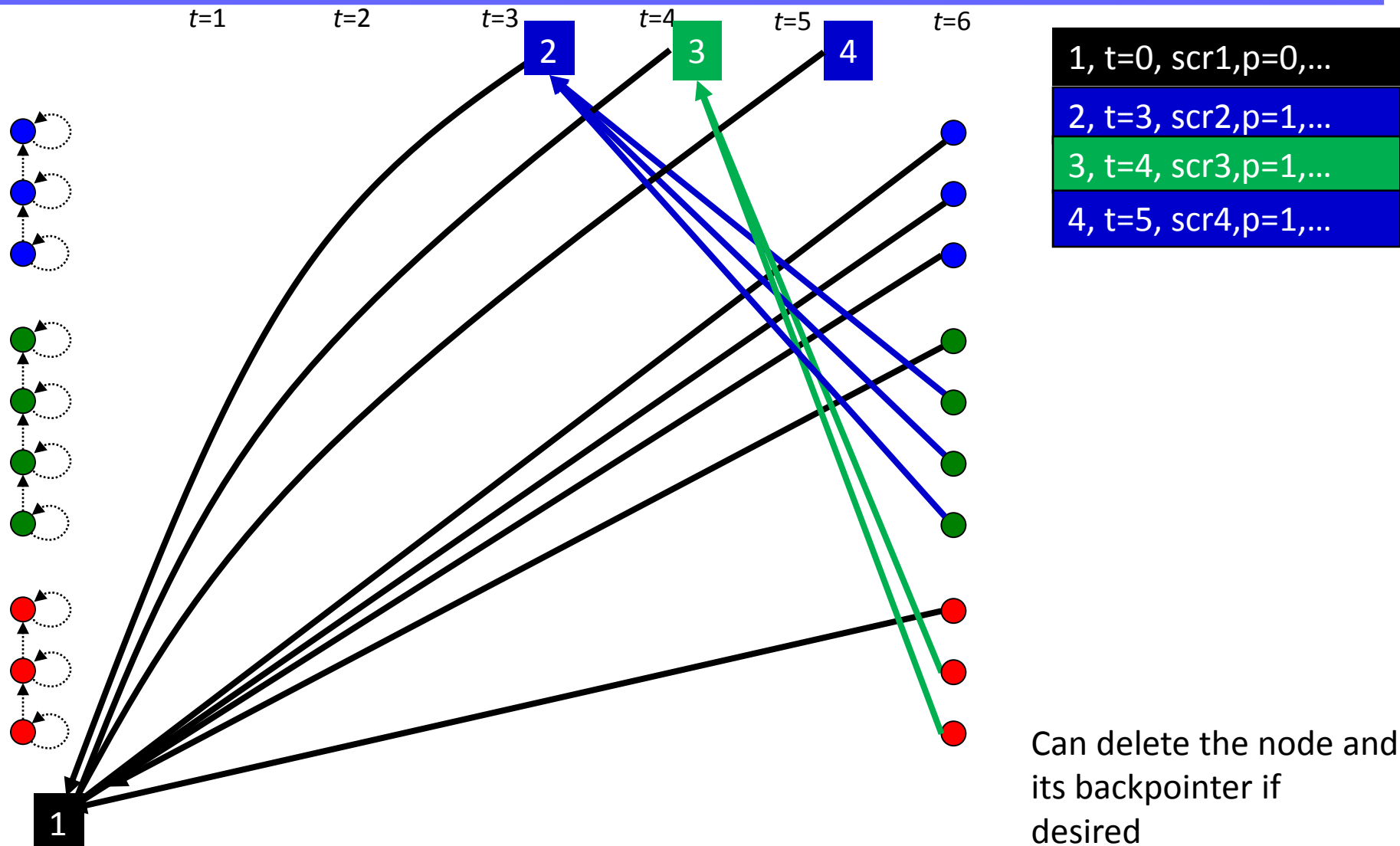


# Trellis with Complete Set of Backpointers

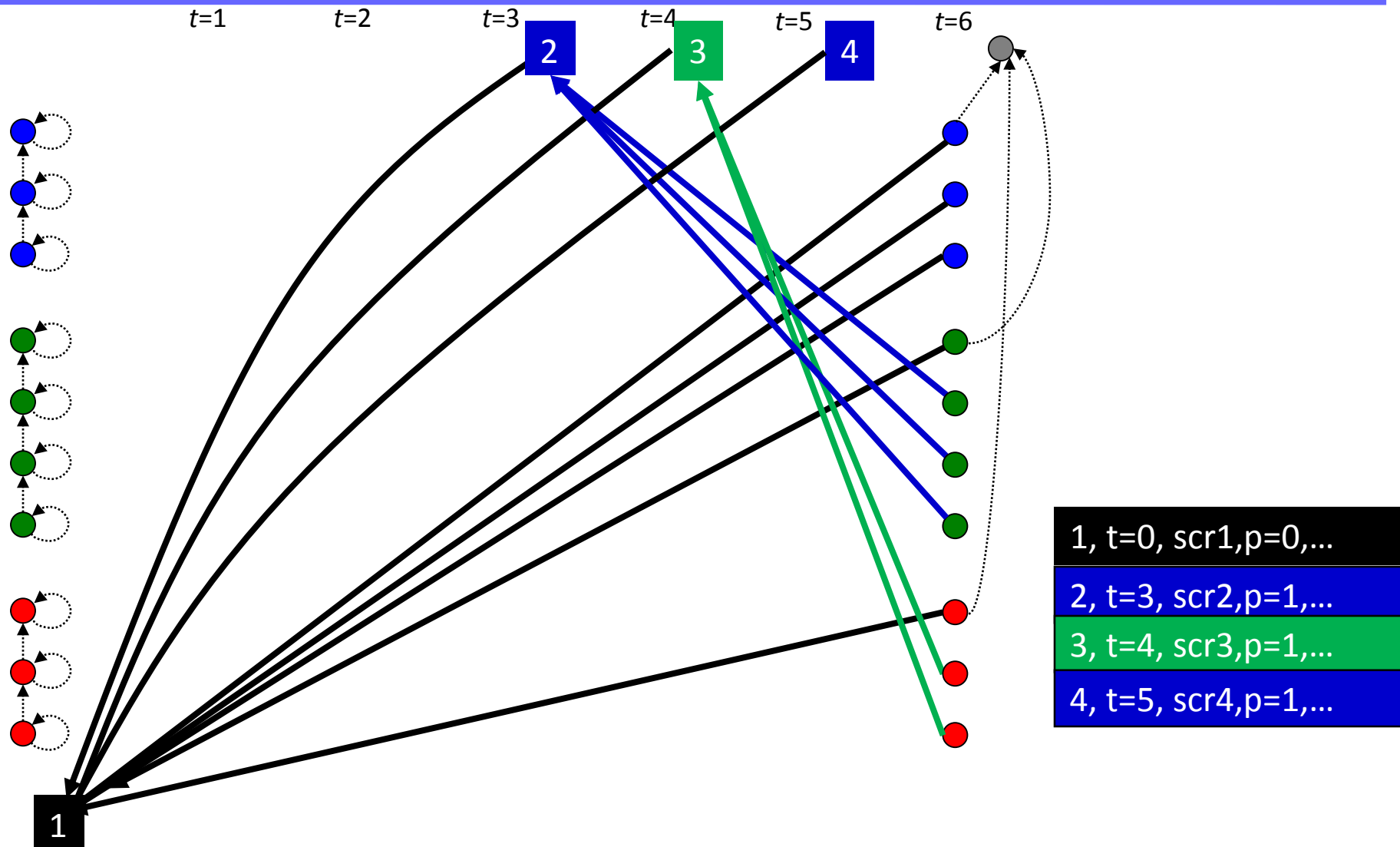




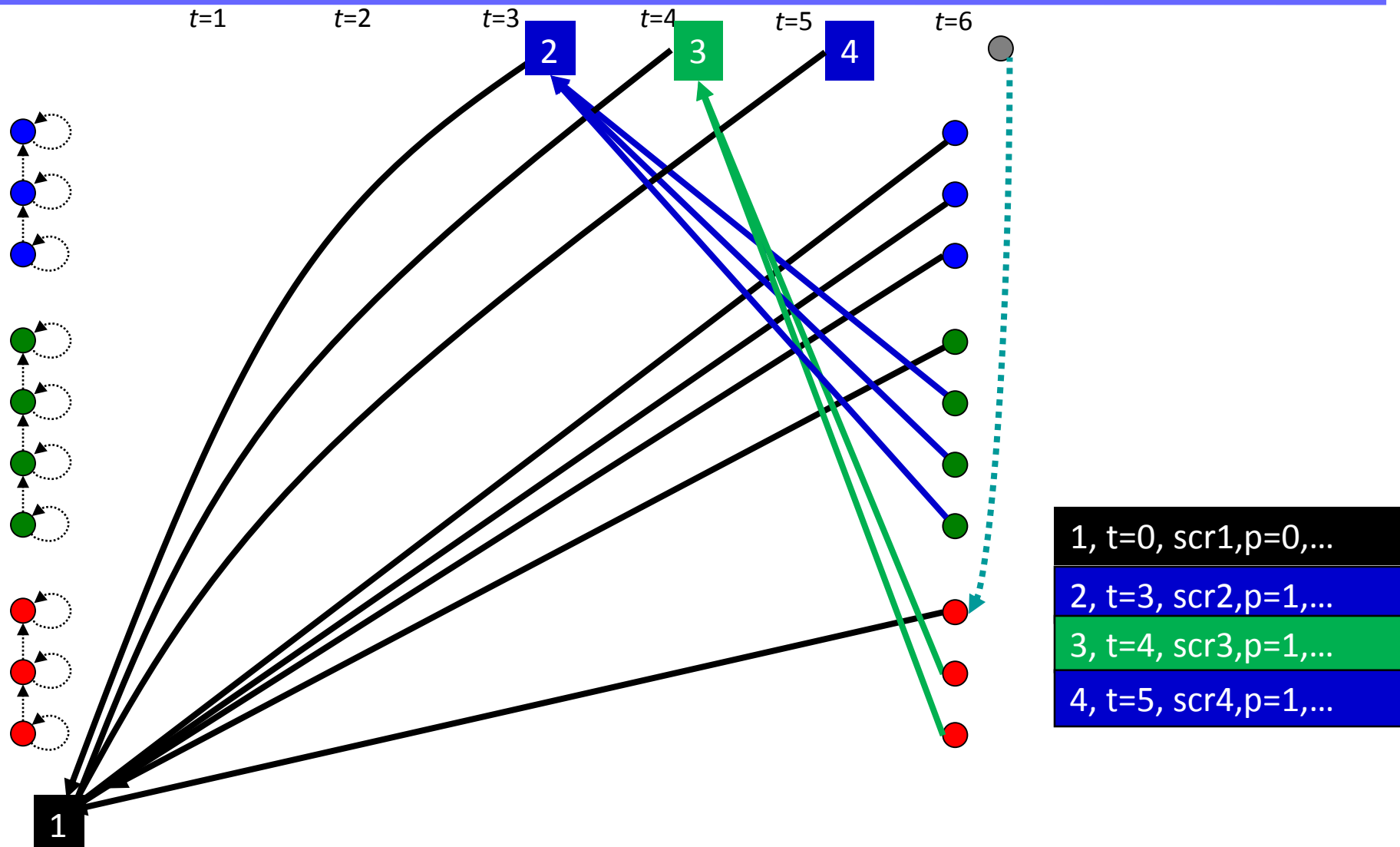
# Trellis with Complete Set of Backpointers



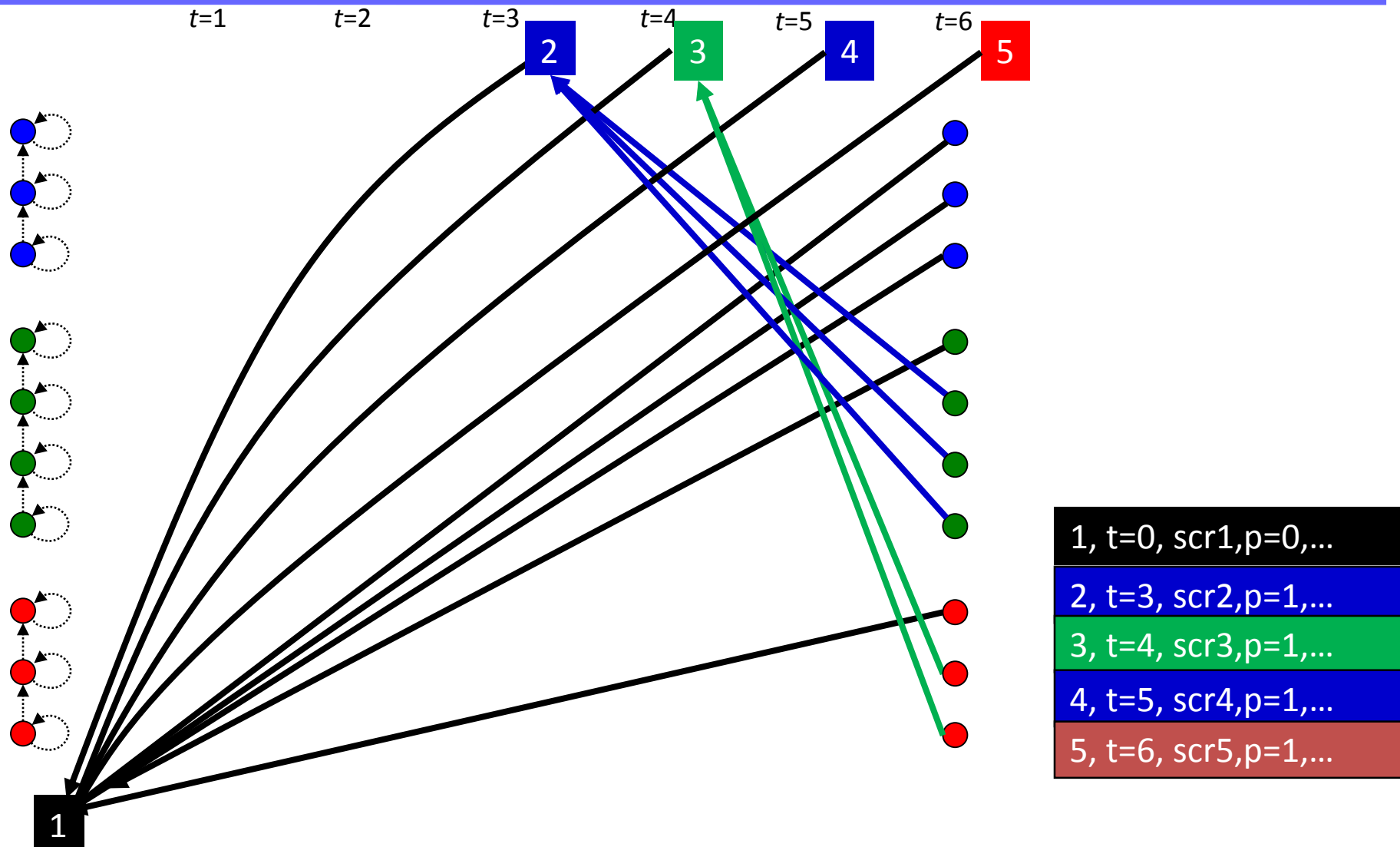
# Trellis with Complete Set of Backpointers



# Trellis with Complete Set of Backpointers



# Trellis with Complete Set of Backpointers

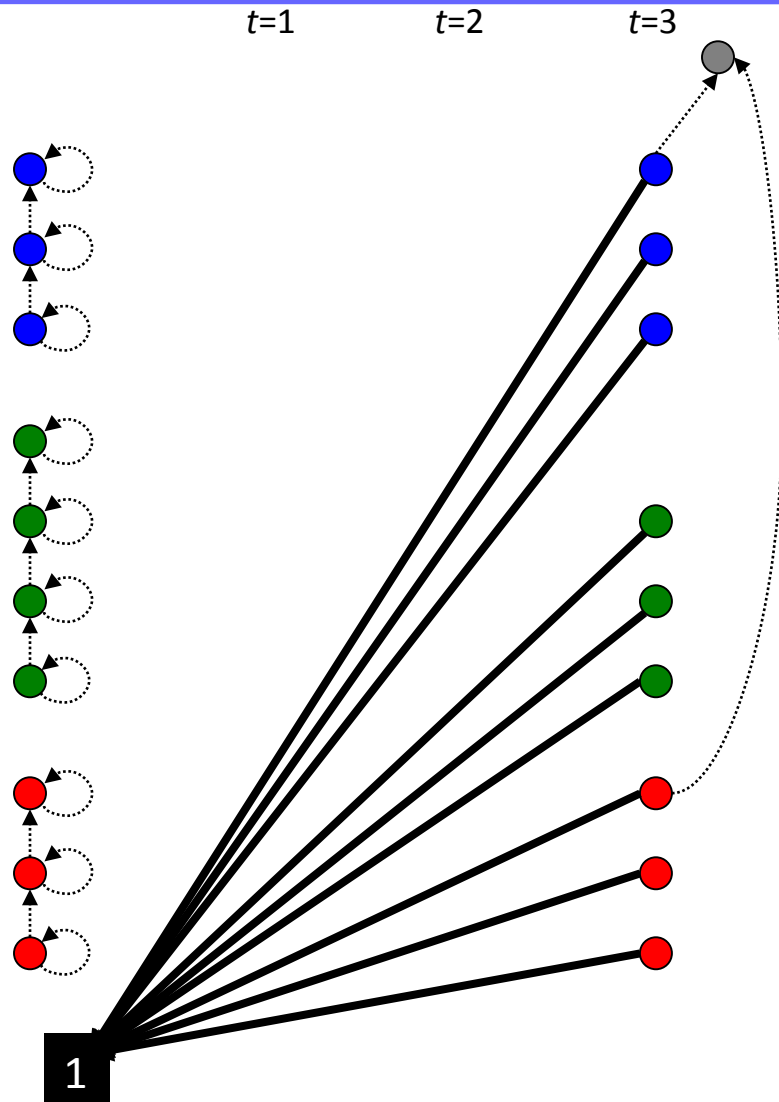


# Using Backpointers

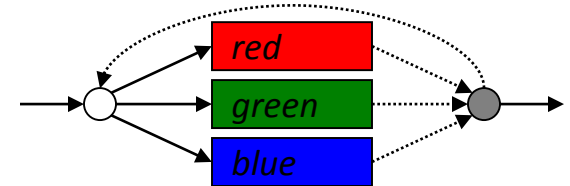
---

- The Backpointer table in the previous figure only retains sufficient history to obtain the best hypothesis
- Sometimes we would like to retain additional information in the backpointer table that tells us what other words were considered (not pruned out) during recognition
  - e.g. when we want to create lattices for finding N-best hypotheses or to compute confidence
- In this case the backpointer table is expanded to include *all* trellis nodes at the final states of words
  - Additionally, all trellis nodes corresponding to non-emitting nodes may also be stored

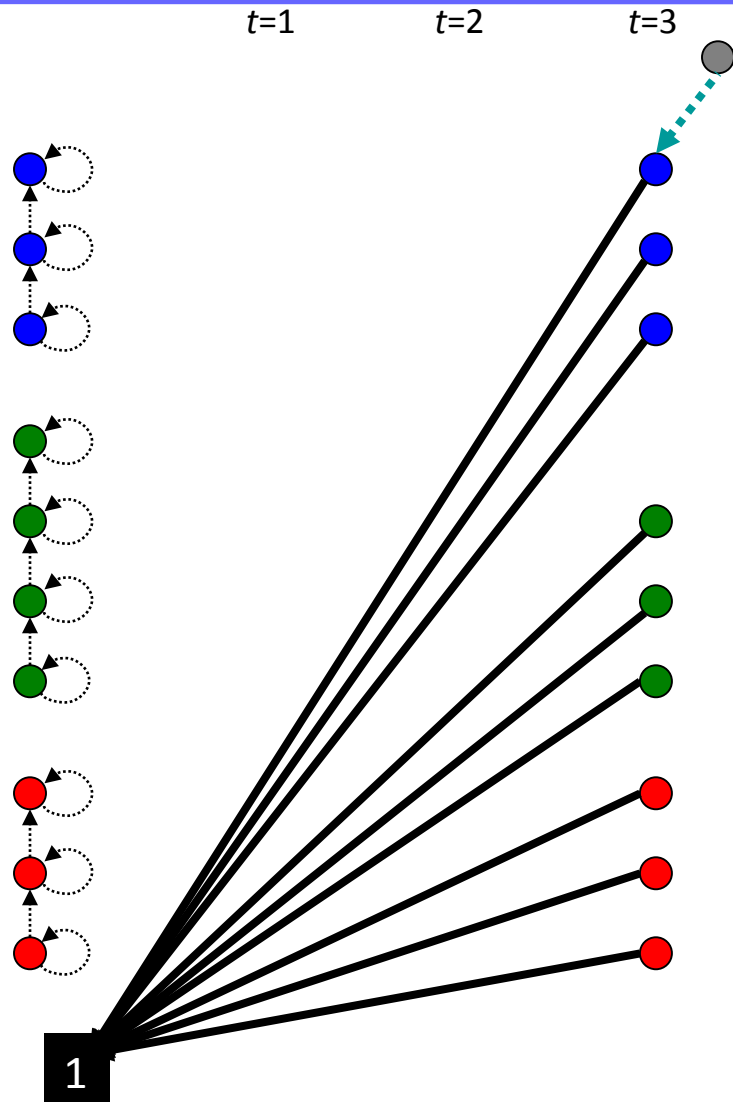
# Regular BP Table



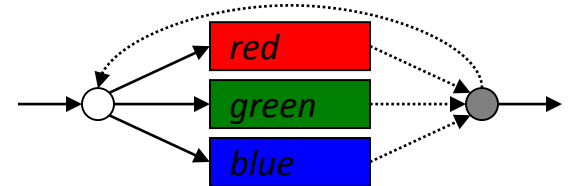
1,  $t=0$ , scr1,  $p=0, \dots$



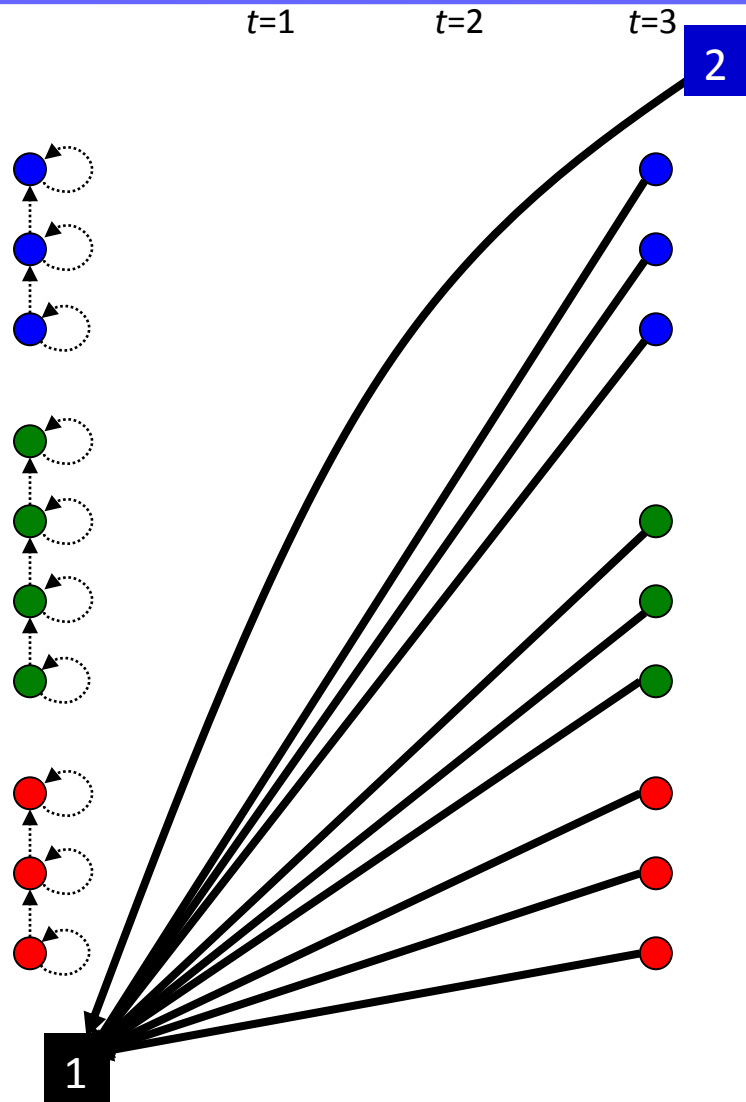
# Regular Trellis



1,  $t=0$ ,  $scr1$ ,  $p=0, \dots$

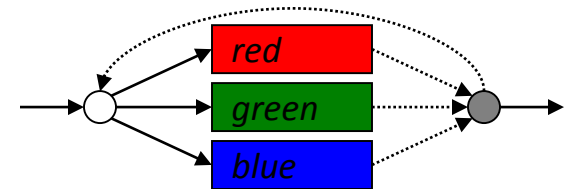


# Regular Trellis



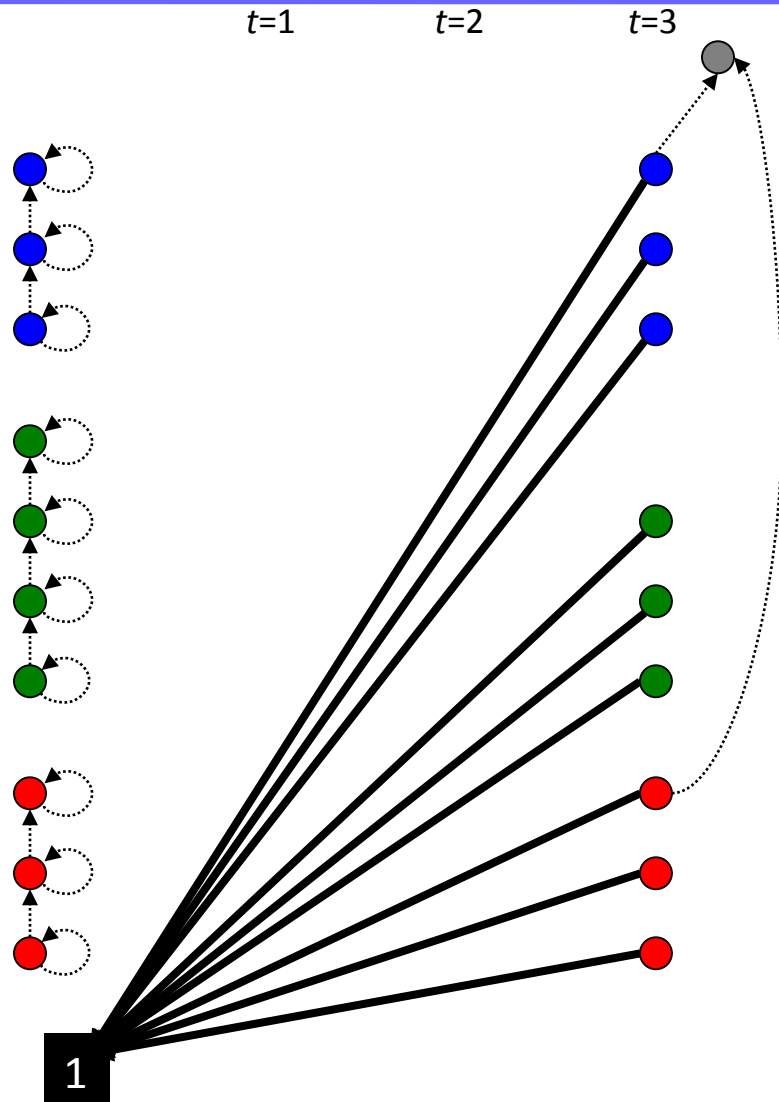
1,  $t=0$ , scr1,  $p=0, \dots$

2,  $t=3$ , scr2,  $p=1, \dots$

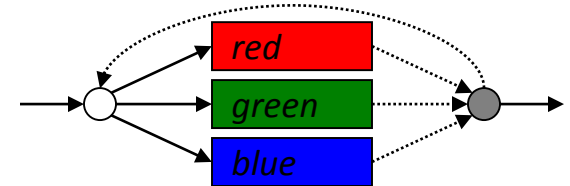




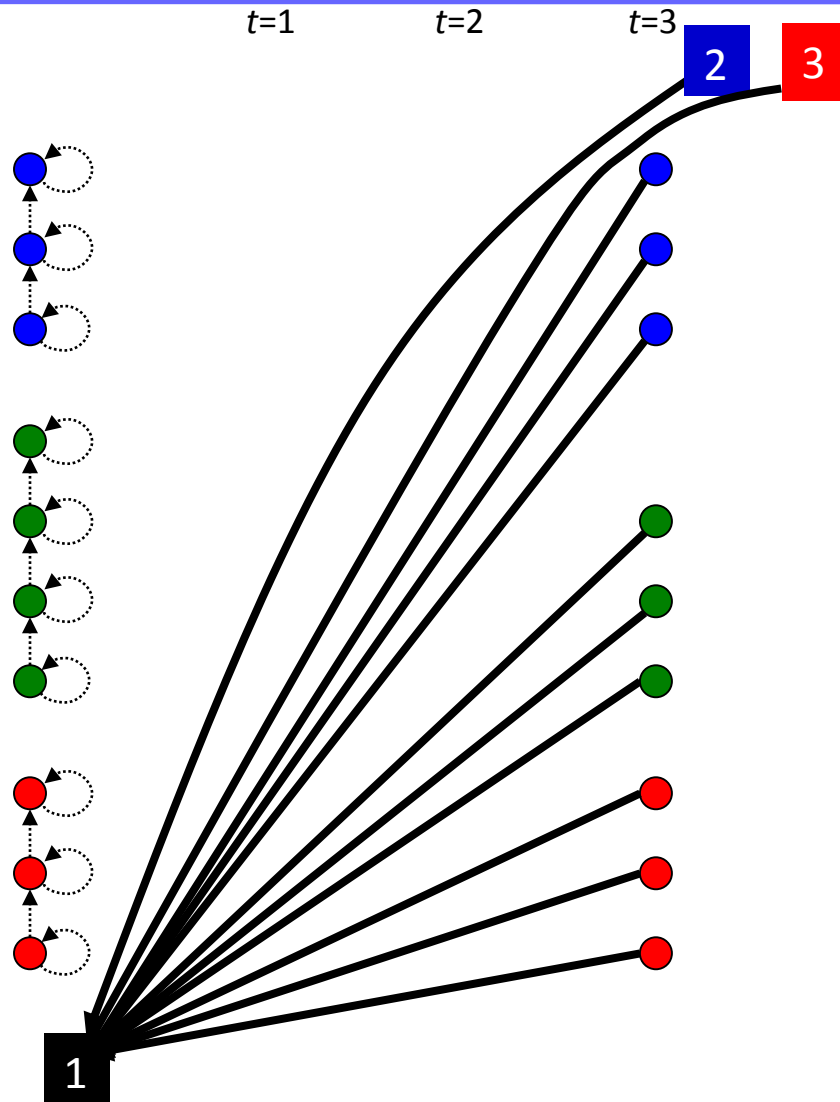
# EXTENDED TRELLIS



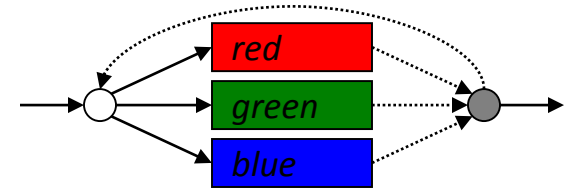
1,  $t=0$ , scr1,  $p=0, \dots$



# EXTENDED TRELLIS

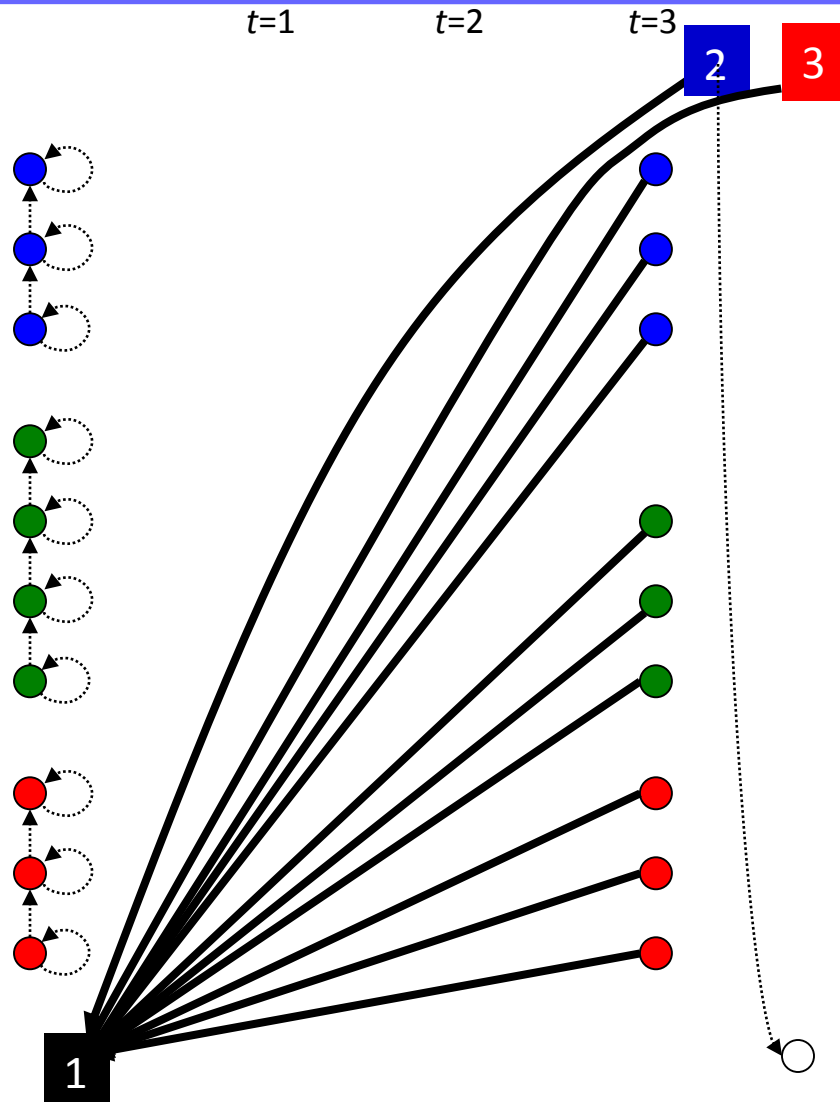


1, $t=0$ , scr1, $p=0, \dots$
2, $t=3$ , scr2, $p=1, \dots$
3, $t=3$ , scr3, $p=1, \dots$



*FAILED* cross-word transitions too enter the backpointer table

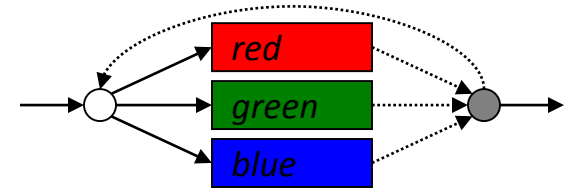
# EXTENDED TRELLIS



1,  $t=0$ , scr1,  $p=0, \dots$

2,  $t=3$ , scr2,  $p=1, \dots$

3,  $t=3$ , scr3,  $p=1, \dots$



*FAILED* cross-word transitions too enter the backpointer table  
*BUT DO NOT PROPAGATE FURTHER*

# Using Backpointers

---

- Even with extended backpointer sets (including word ending and null-state entries), back pointer tables can be much smaller than retaining a full backpointer matrix, one for each trellis node
- Backpointers need not be stored as tables. They can, in fact be stored explicitly in tree format
- The entries in the backpointer table may also contain information about the *forward* transition
  - E.g. word-ending lattice-node backpointer table entries may additionally point forward to the subsequent null-state
  - This information may be used to derive lattices directly from BP tables

# Decoding – finesse

---

- The final computational and memory expense of decoding depends on the *size* of the language HMM
- When composed from HMMs of words (or phonemes, as we will see later), it can get quite large
  - Particularly from redundant non-emitting states
- The HMM must be *minimized*
  - Topic for another day..