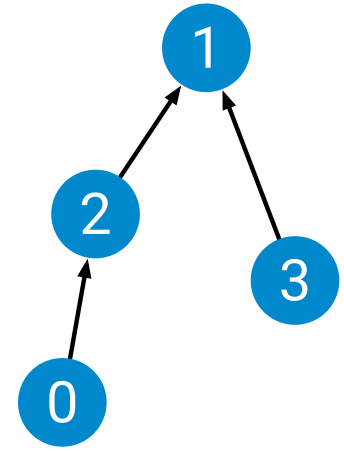
Programming Parallel Computers

Jukka Suomela · Aalto University · ppc.cs.aalto.fi

Part 6C: Pointer jumping

Pointer jumping: setting

x:	0	1	2	3
p[x]:	2	end	1	1

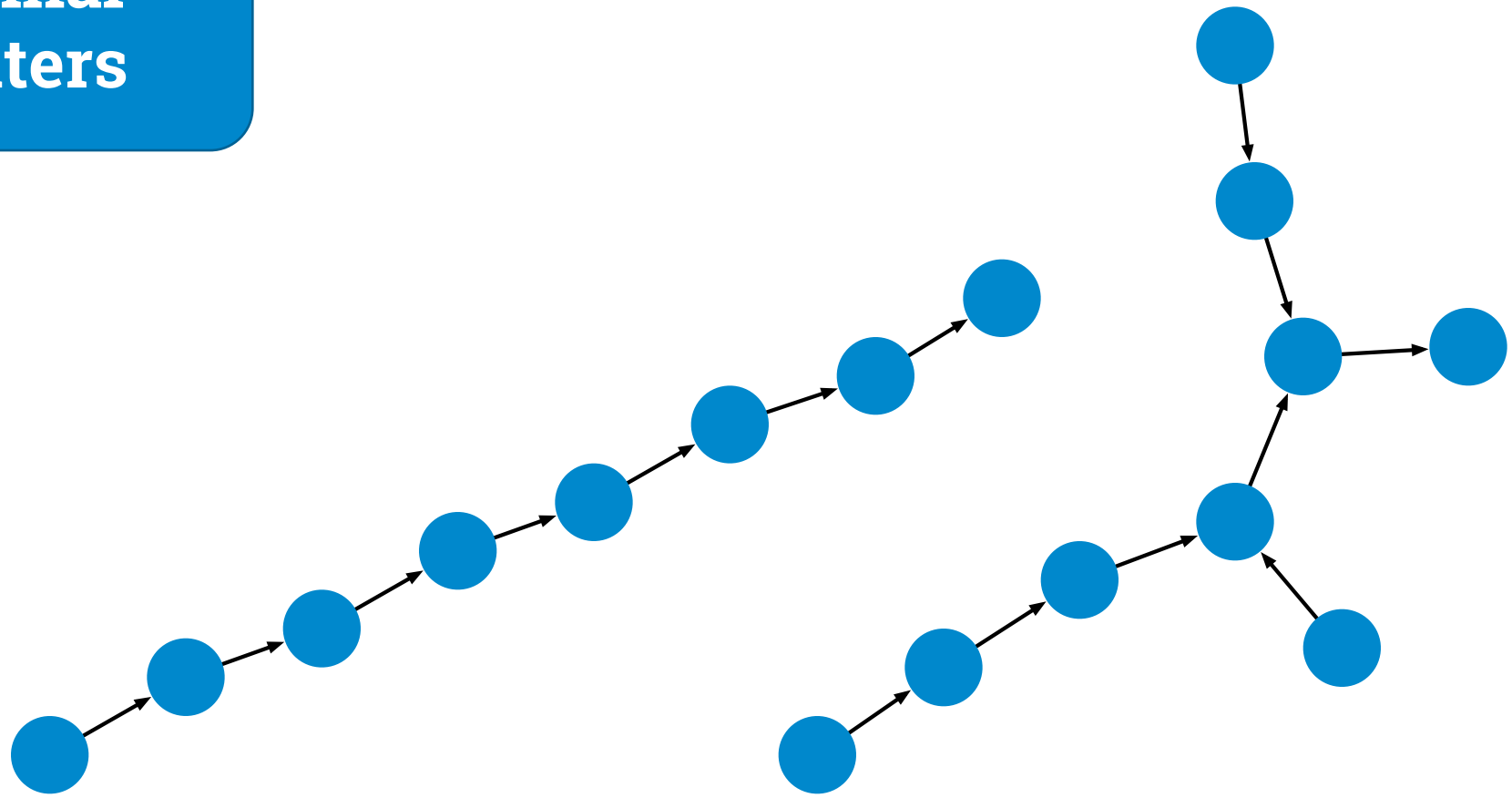


- Linked data structure
  - array *p* with *n* elements
  - *p*[*x*] = ***successor*** of element *x* or special label “end”
  - can represent a **linked list**
  - can represent a **rooted tree** (successor = parent)
- You would like to follow links efficiently
  - example: ***how far is element x from the end?***
- Trivial sequential algorithm:
  - repeatedly set *x* = *p*[*x*] until we reach the end

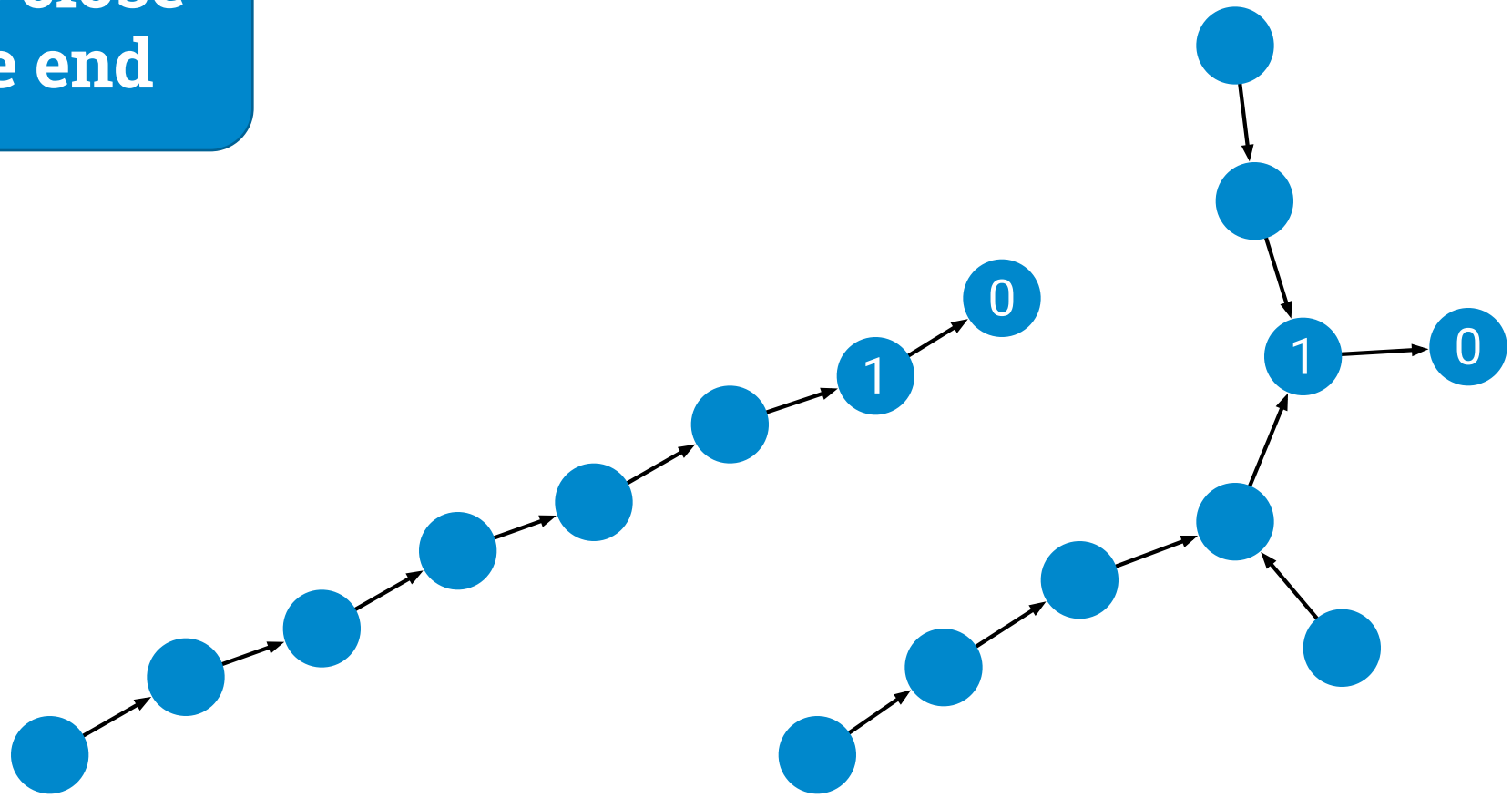
Pointer jumping: idea

- Simple and efficient technique for handling linked data
- Basic idea:
  - **input:** $p[x]$ = successor of element x
  - **in parallel:** set $q[x] = p[p[x]]$ for all x
  - **output:** $q[x]$ = which element is 2 hops from x?
  - array q represents length-2 shortcuts: **"1 hop of q" = "2 hops of p"**
- Repeat:
  - shortcuts of length 4
  - shortcuts of length 8 …

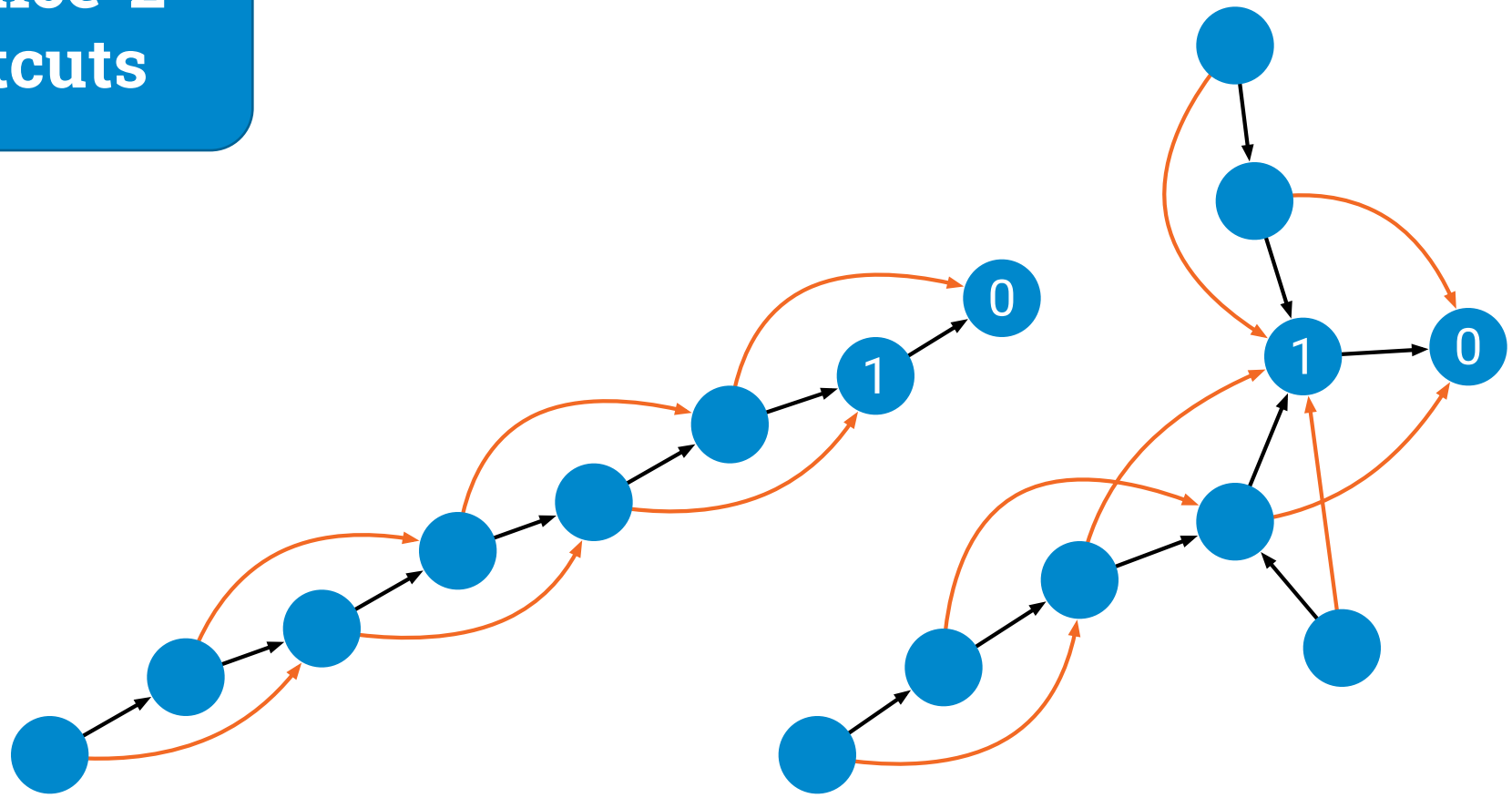
Original pointers



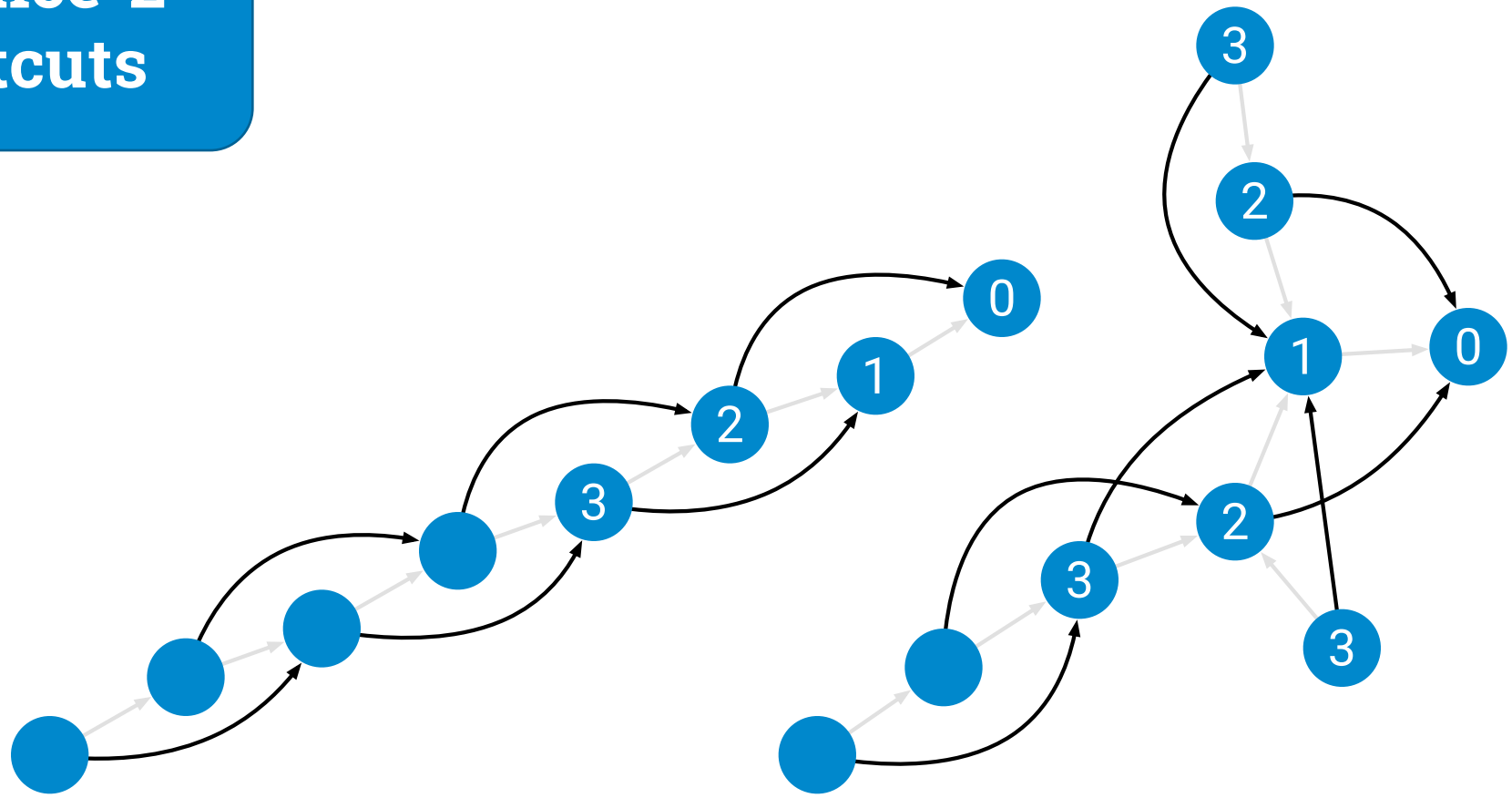
Nodes close to the end



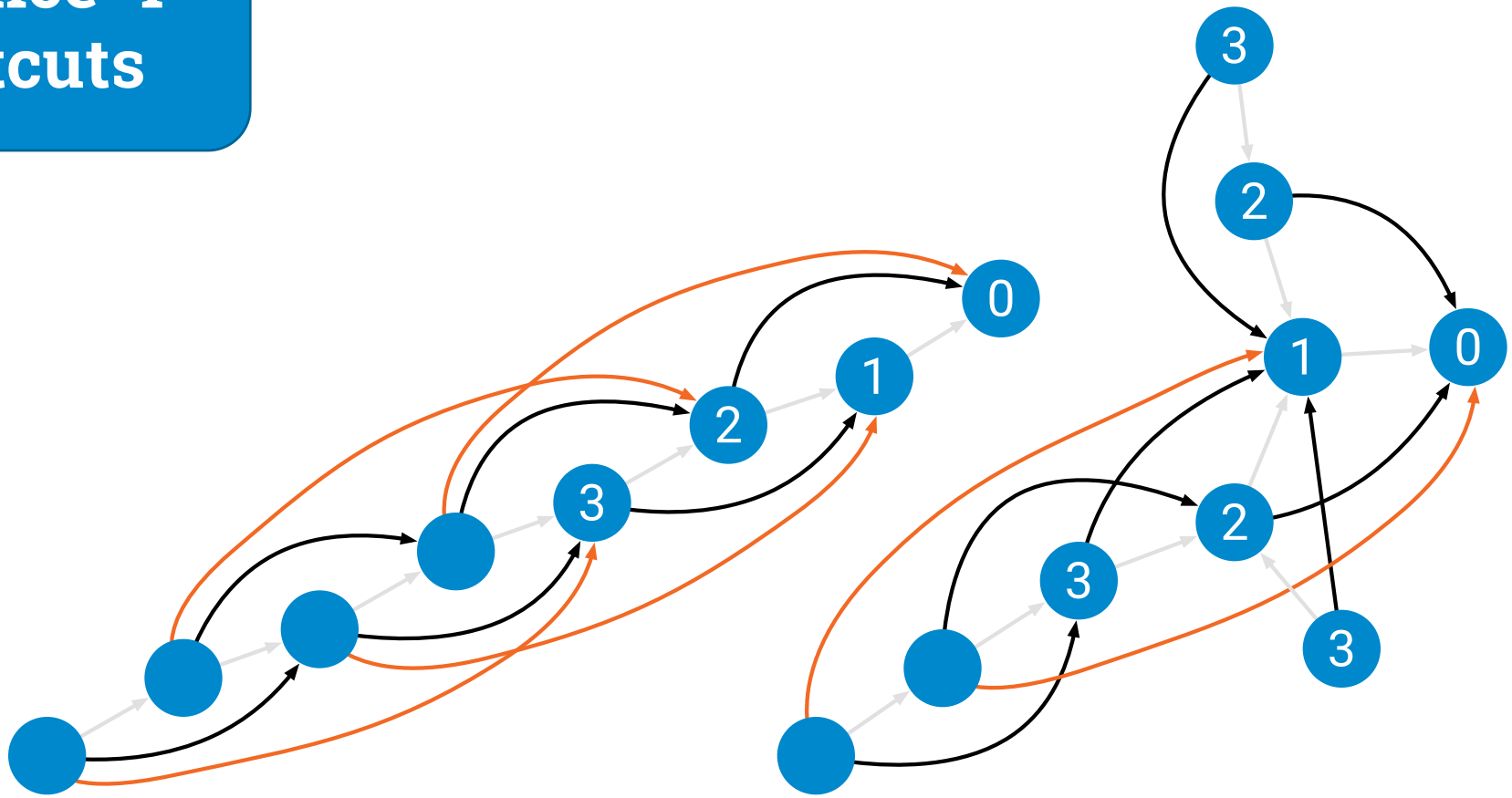
Distance-2 shortcuts



Distance-2 shortcuts



Distance-4 shortcuts



Distance-4 shortcuts