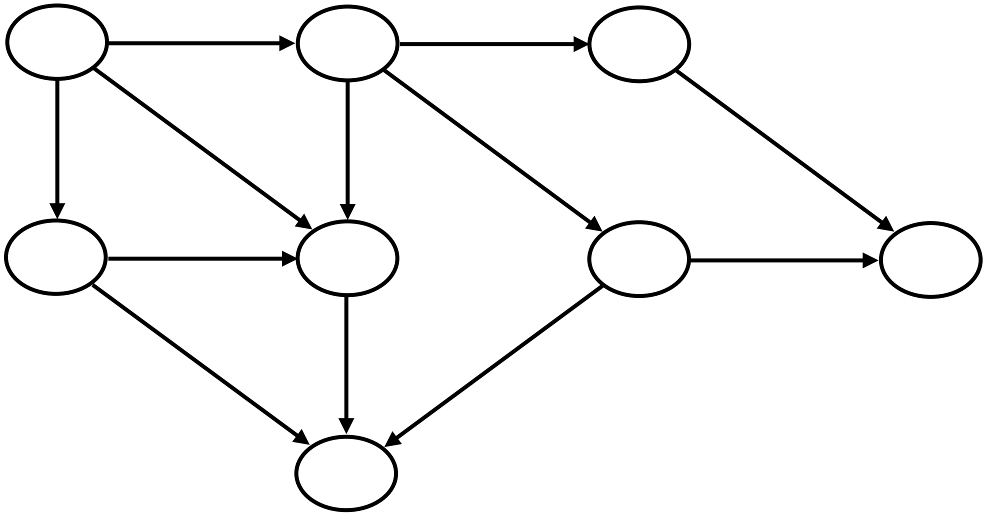
YO-YO

Election in arbitrary graphs:
simple but not optimal

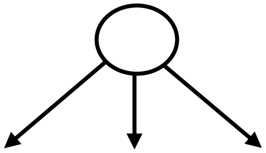
Paola Flocchini

Yo-Yo

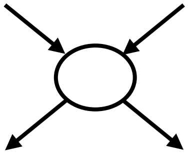
DAG =
Directed Acyclic Graph



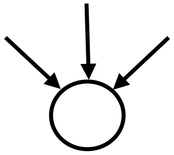
SOURCE:



INTERNAL NODE:

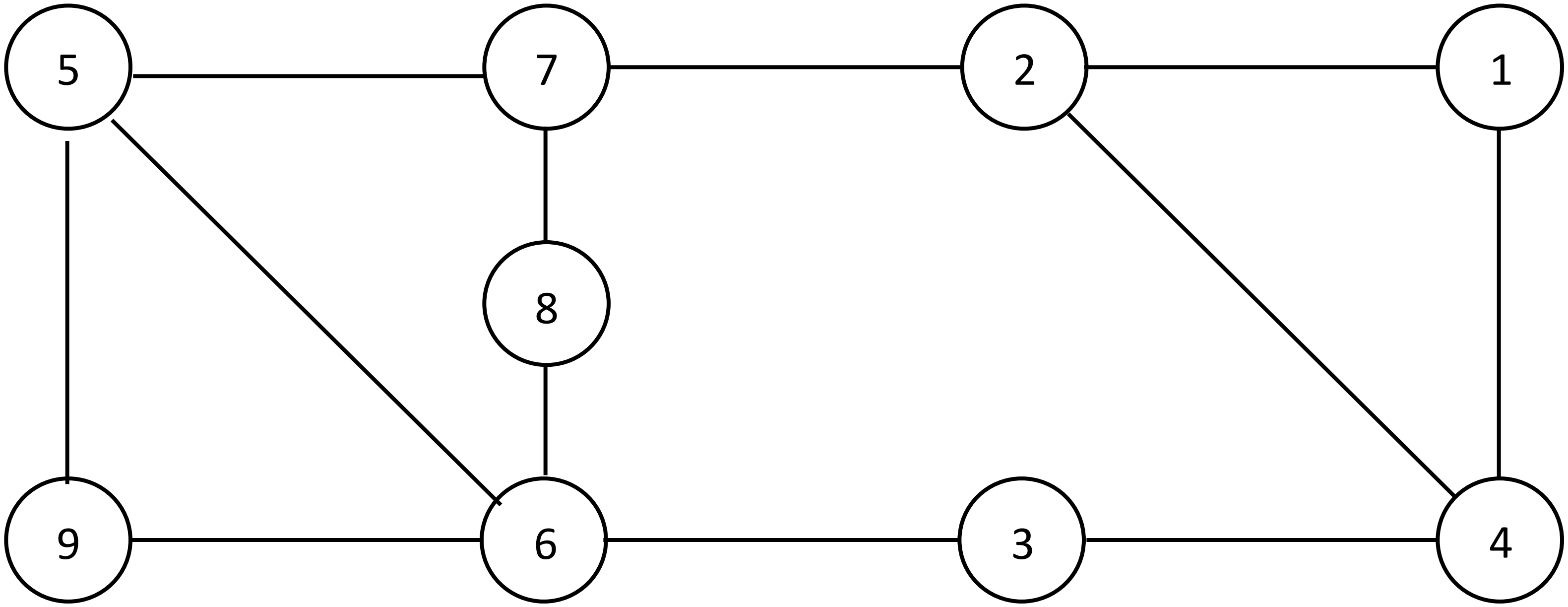


SINK:



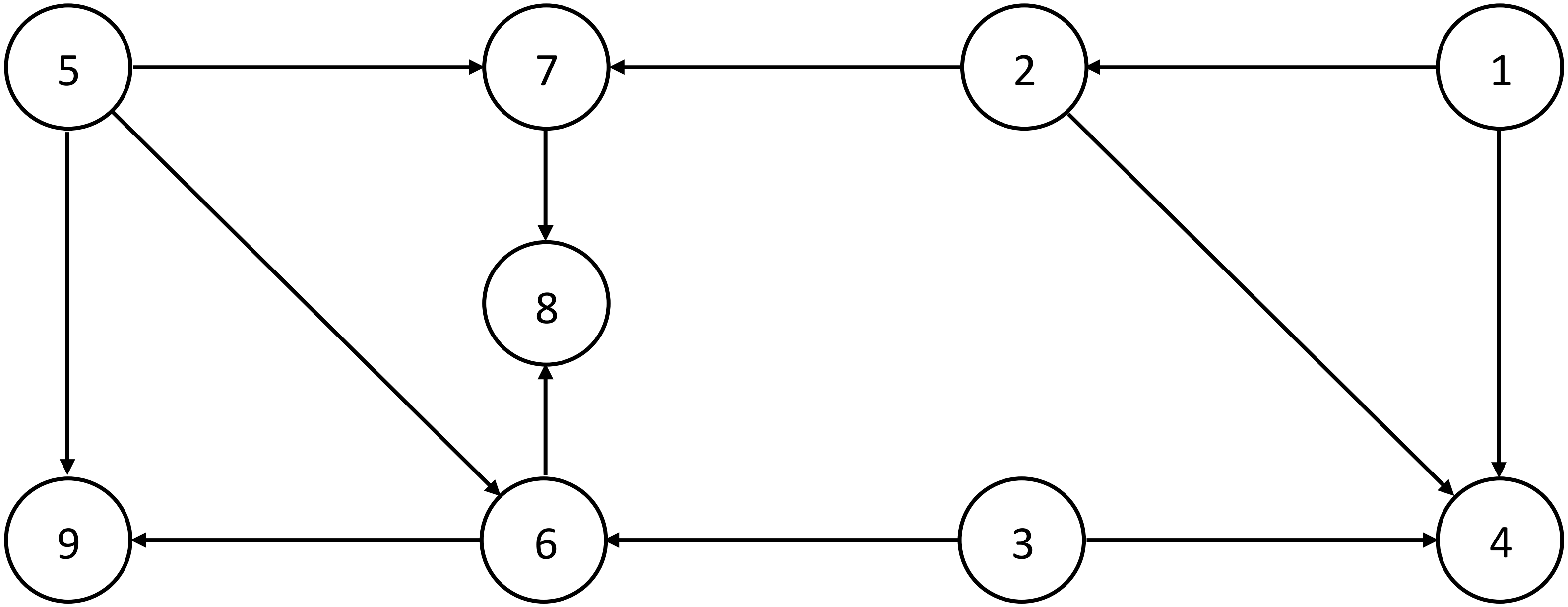
Initialization phase

Given an arbitrary undirected graph



Initialization phase

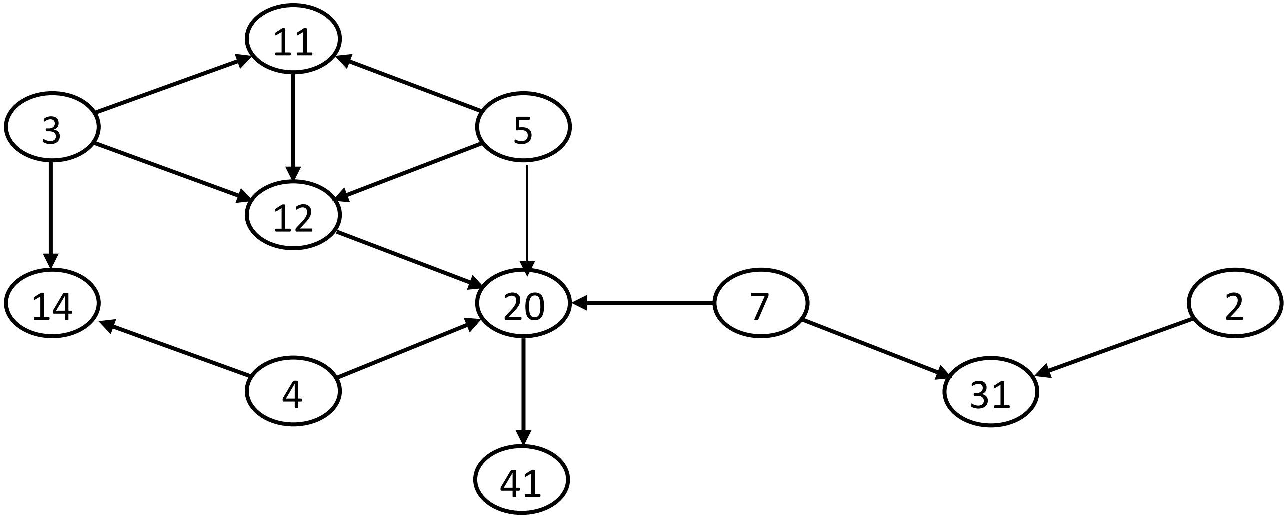
Given an arbitrary undirected graph, smaller entities are directed towards bigger (by exchanging messages with neighbours)

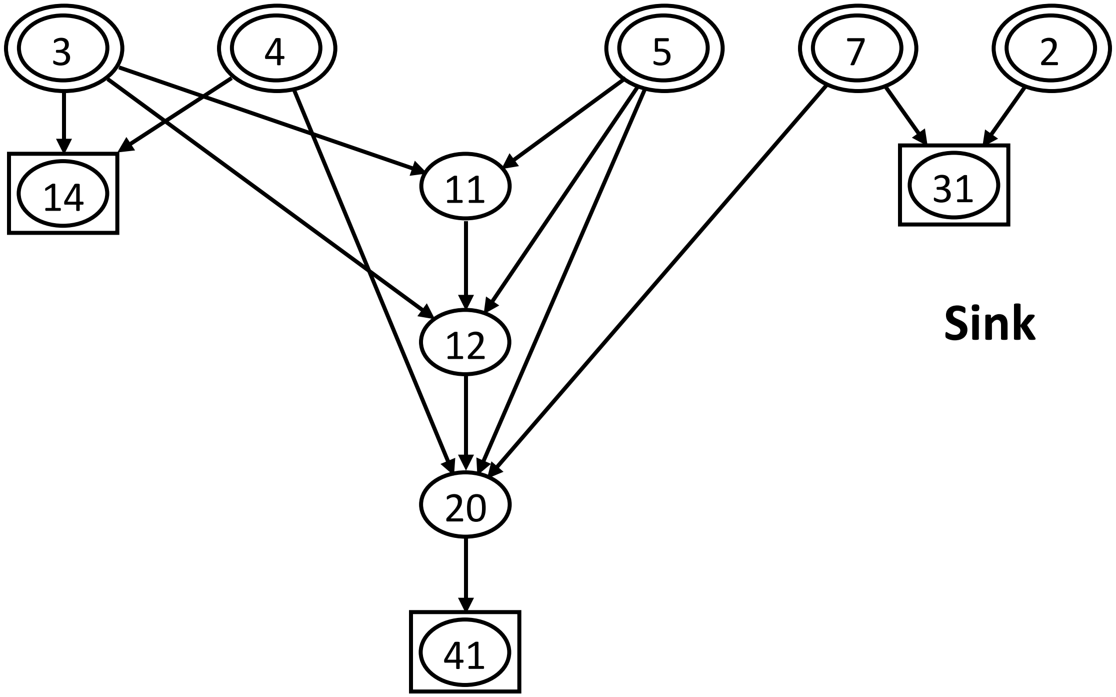


⇨ DAG

O(m) messages

DAG

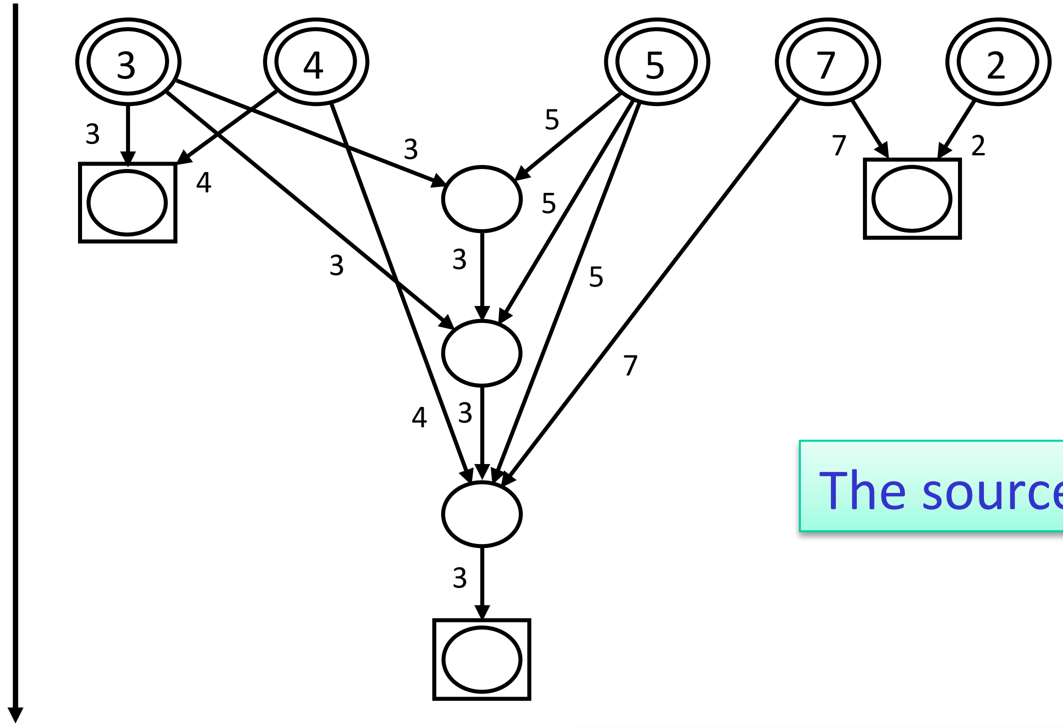




Source

Sink

The protocol

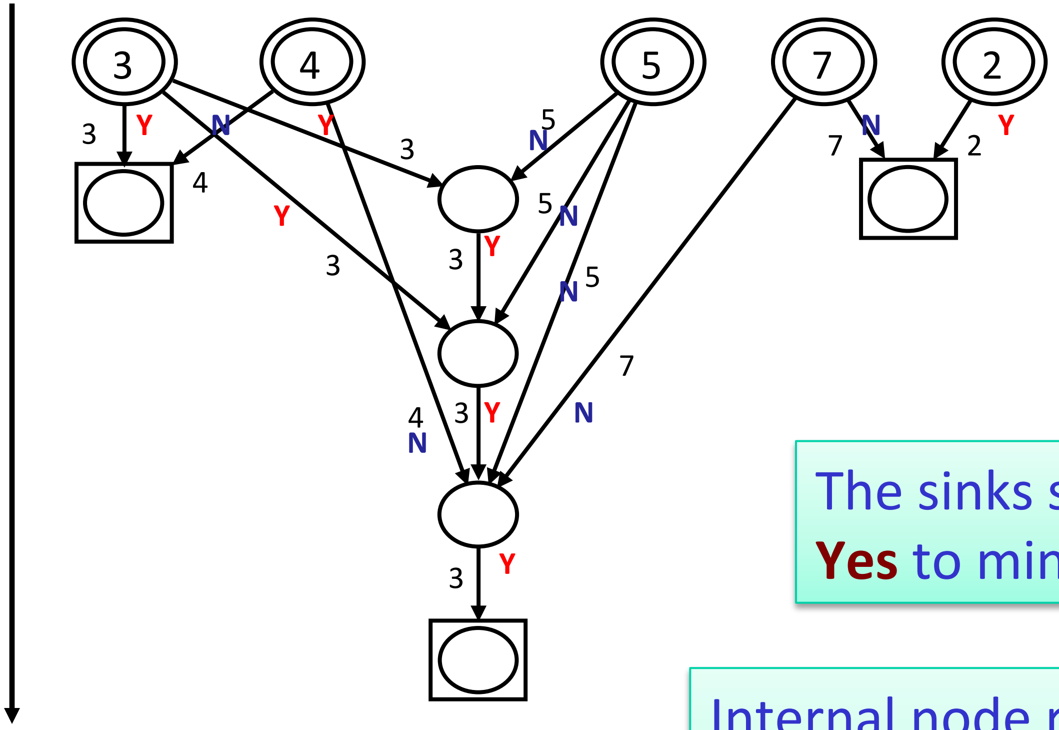


YO: send down

The sources send down their value

Internal node receiving from all in-neighbours
Send min received to all out-neighbours

The protocol



YO: send up

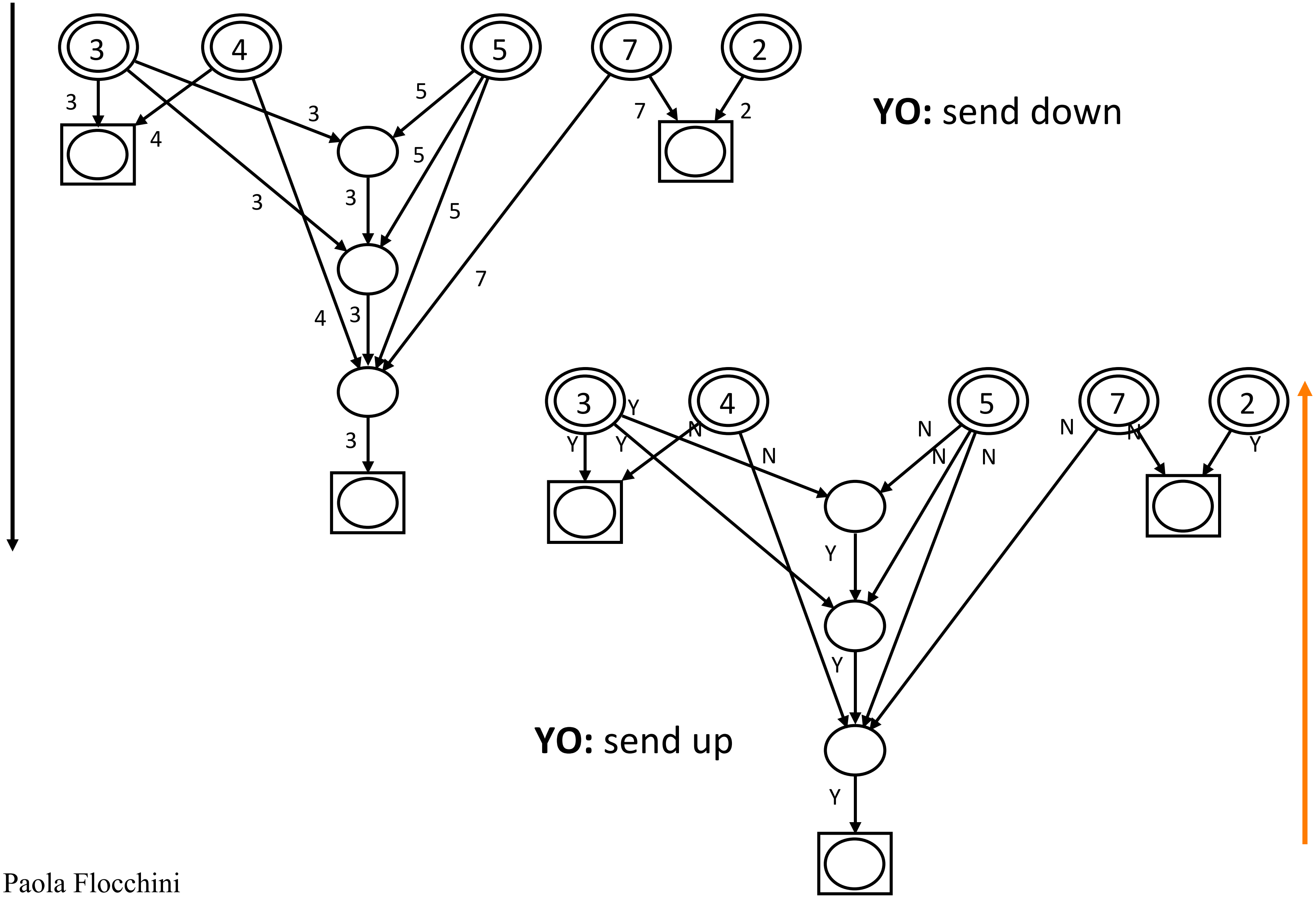
The sinks send:
Yes to min **No** to others

Internal node receiving **only YES** from out-neighbors:
Send **Yes** to min **No** to other in-neighbours

Internal node receiving **at least a NO**:
Send **No** to all in-neighbours

Paola Flocchini

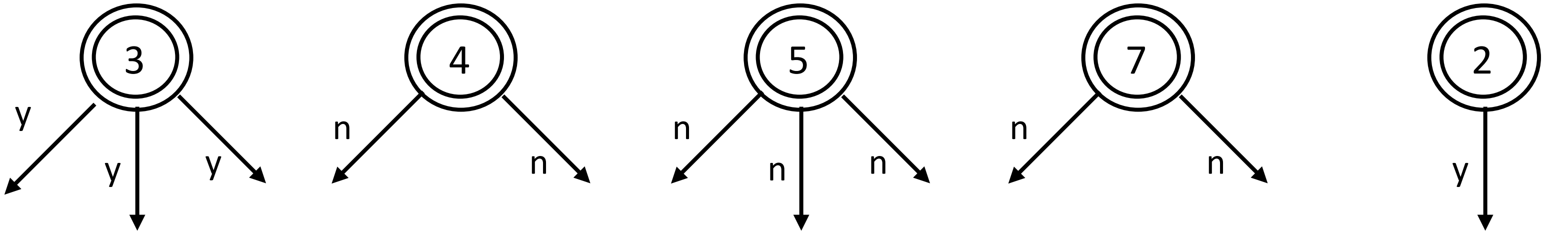
The protocol



YO: send down

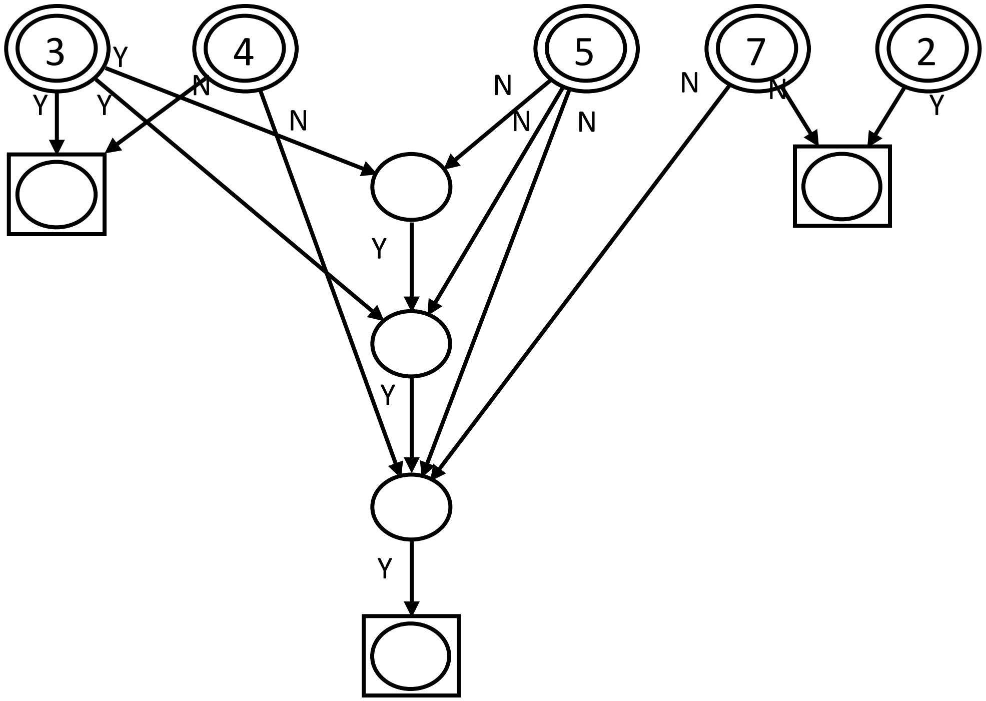
YO: send up

The protocol



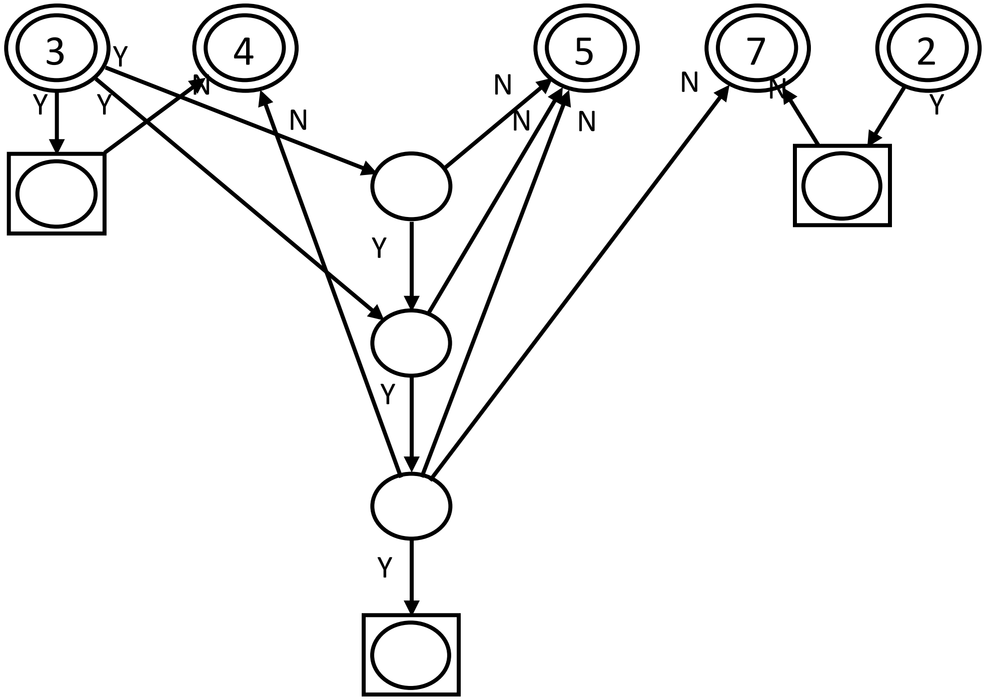
The protocol

The direction of the links with “NO” will be flipped



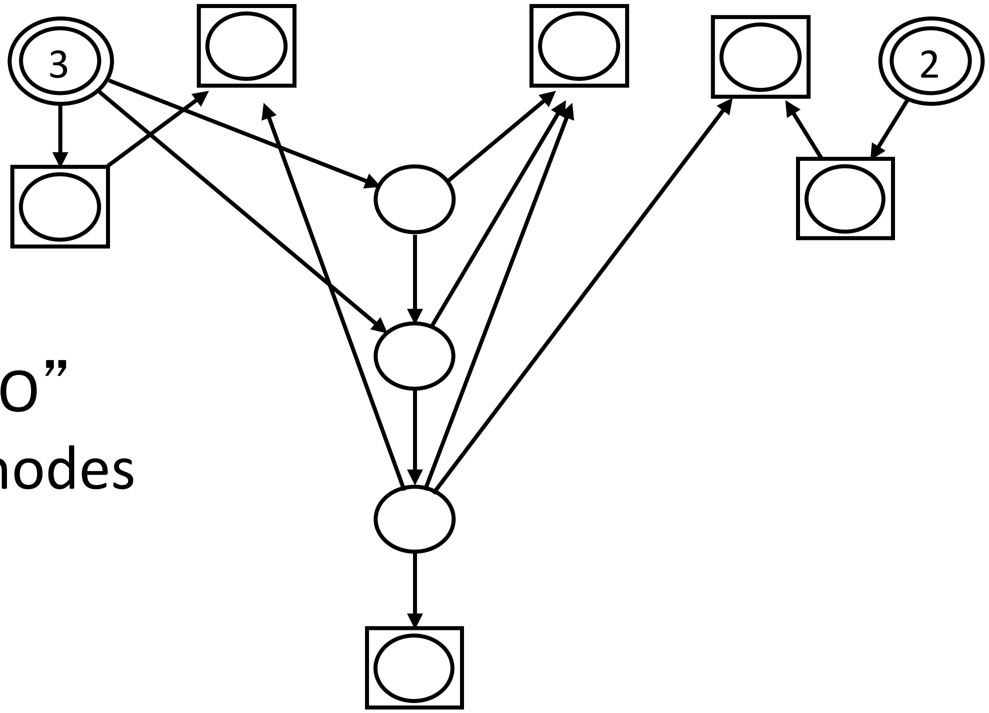
The protocol

The direction of the links with “NO” will be flipped



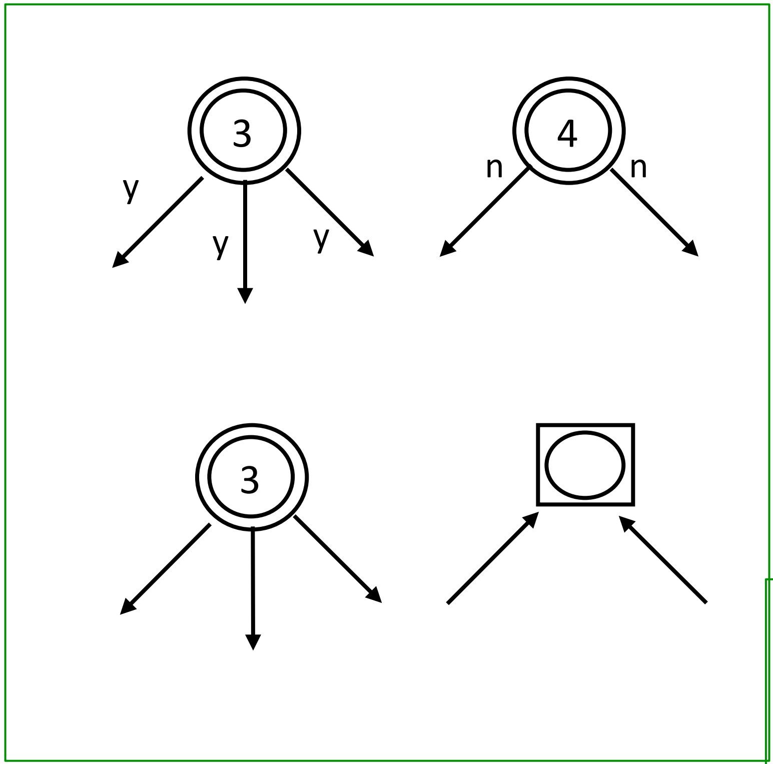
The protocol

The direction of the links with “NO” will be flipped



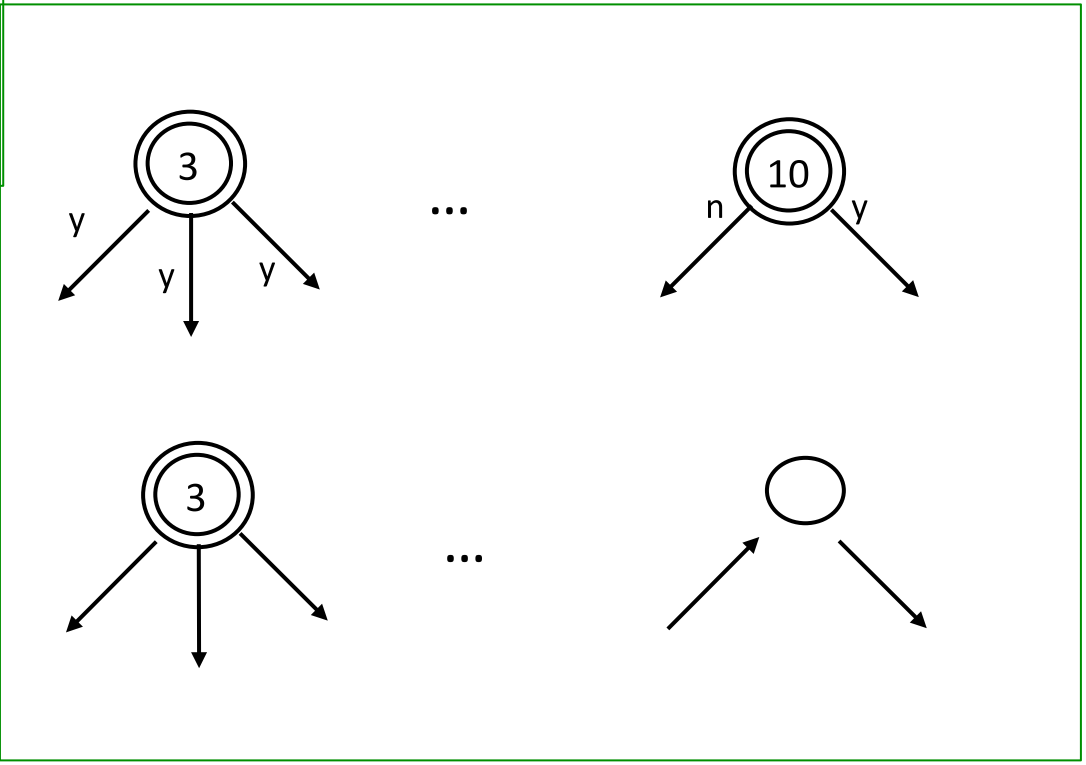
The sources with at least a “NO” will become sinks or internal nodes

In our example a source becomes a sink

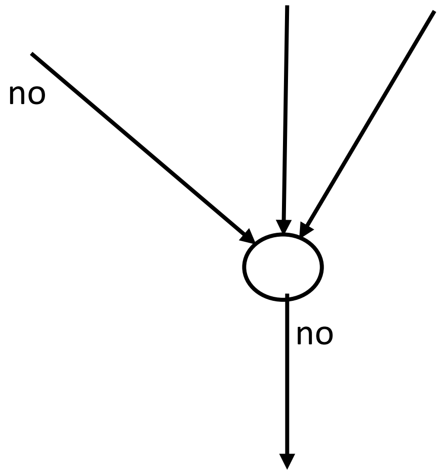


1. At each step at least a source becomes either a sink, or internal

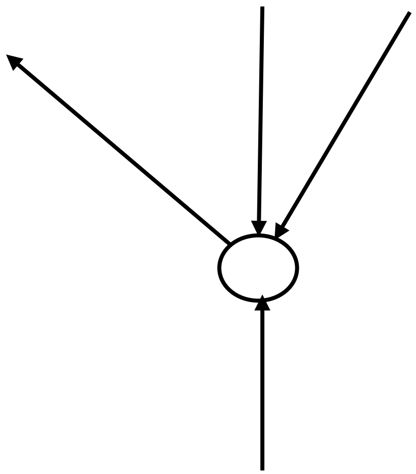
A source becoming internal……



2. An internal node cannot become source after the flipping

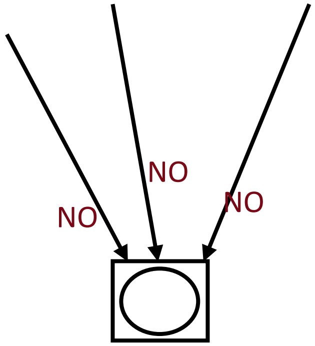


To change state there must be some flipping: so at least a “no” must be received and propagated



But then the node cannot have only outgoing links …

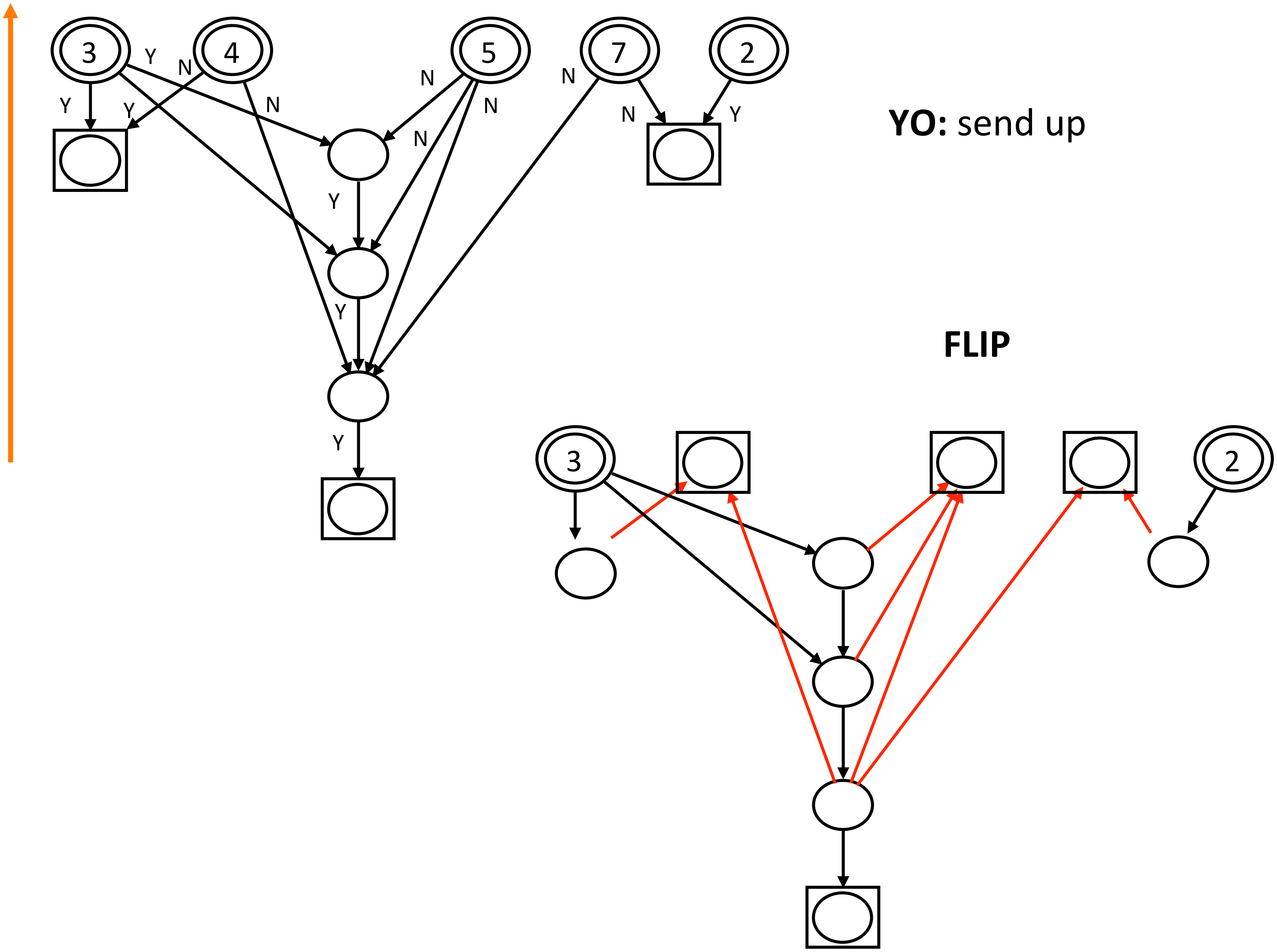
3. A sink cannot become source after the flipping



To become source, it would have to flip ALL its links, but this would mean sending NO to all nodes, which is impossible

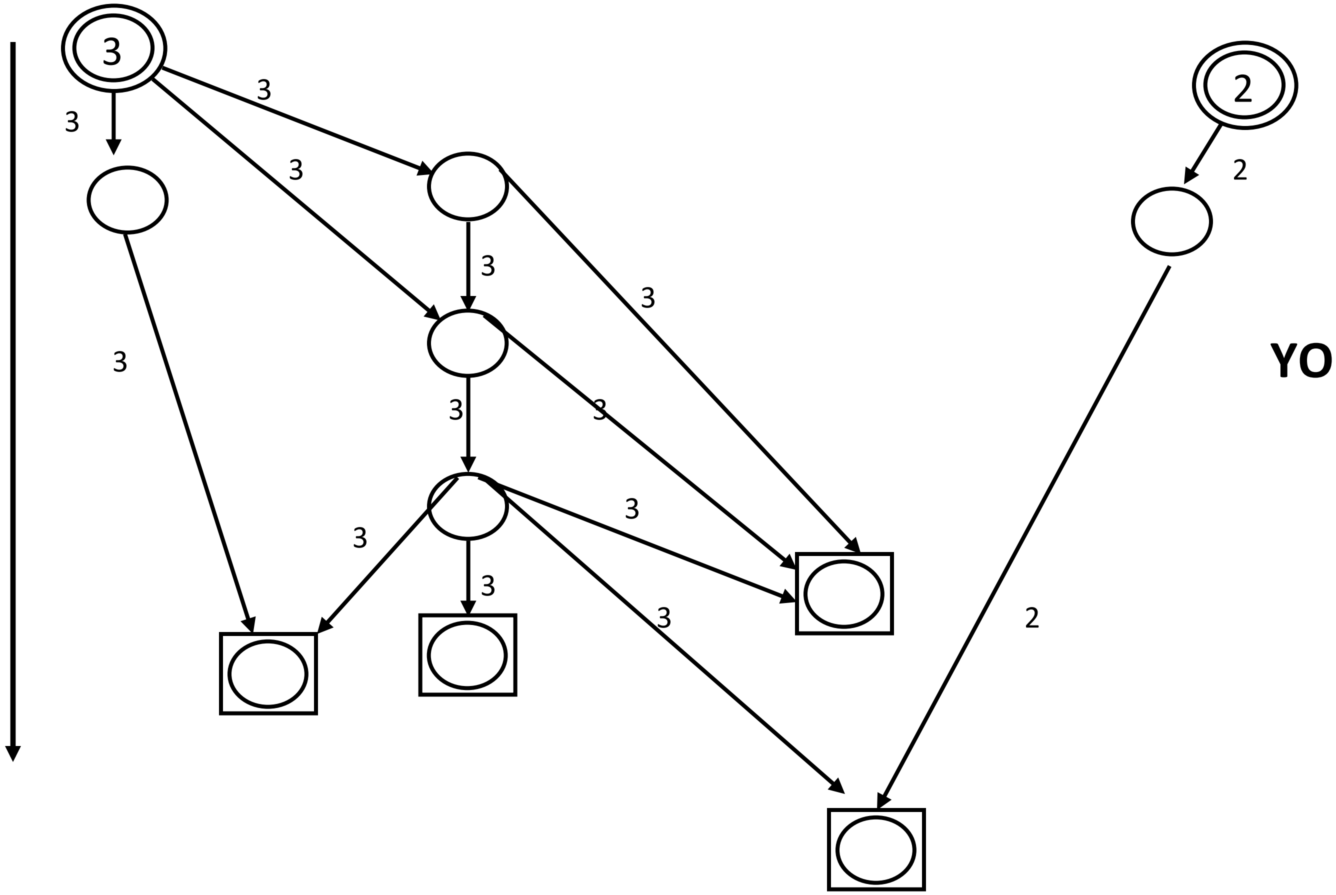
From 1. 2. and 3.:

The number of sources is reduced monotonically



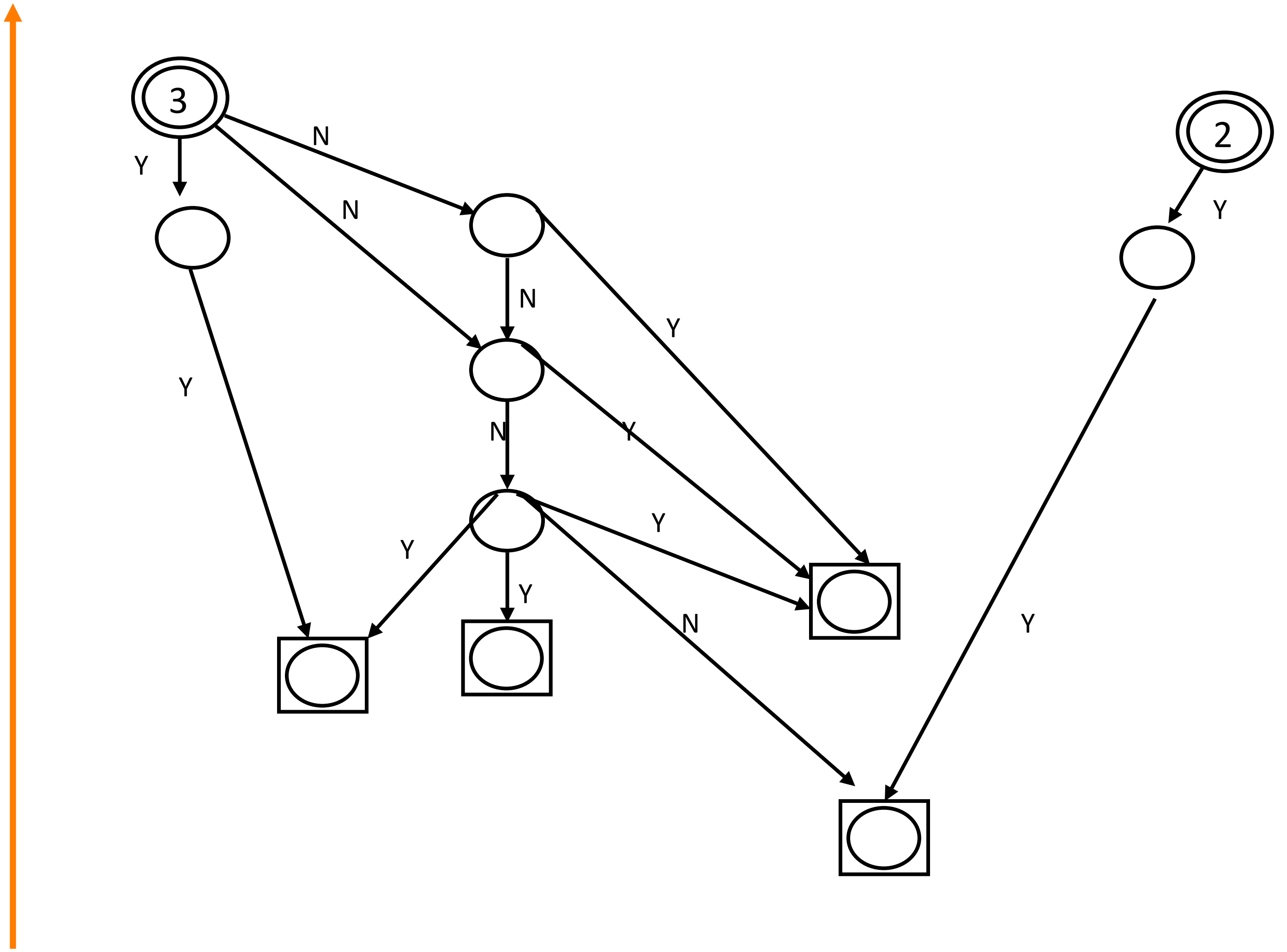
YO: send up

FLIP

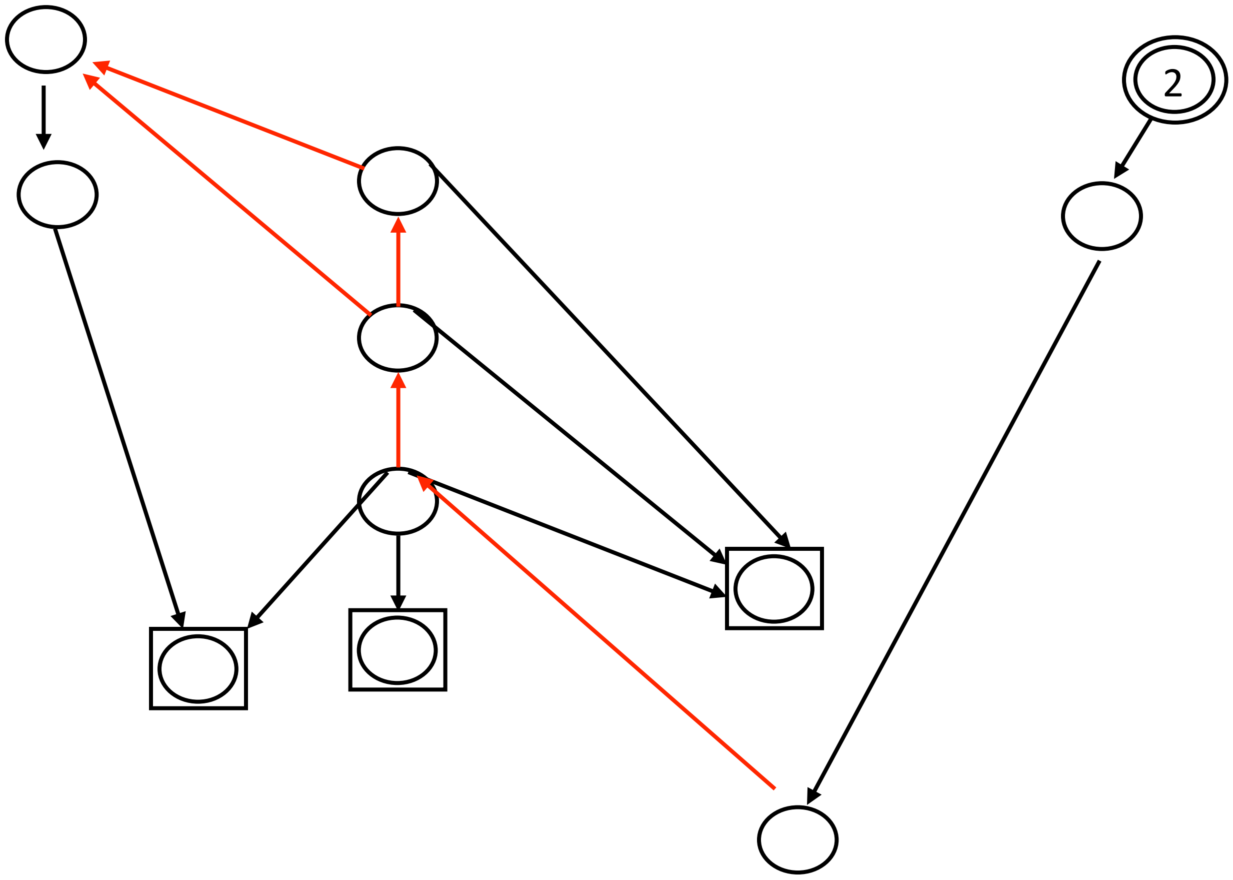


YO: send down

Paola Flocchini

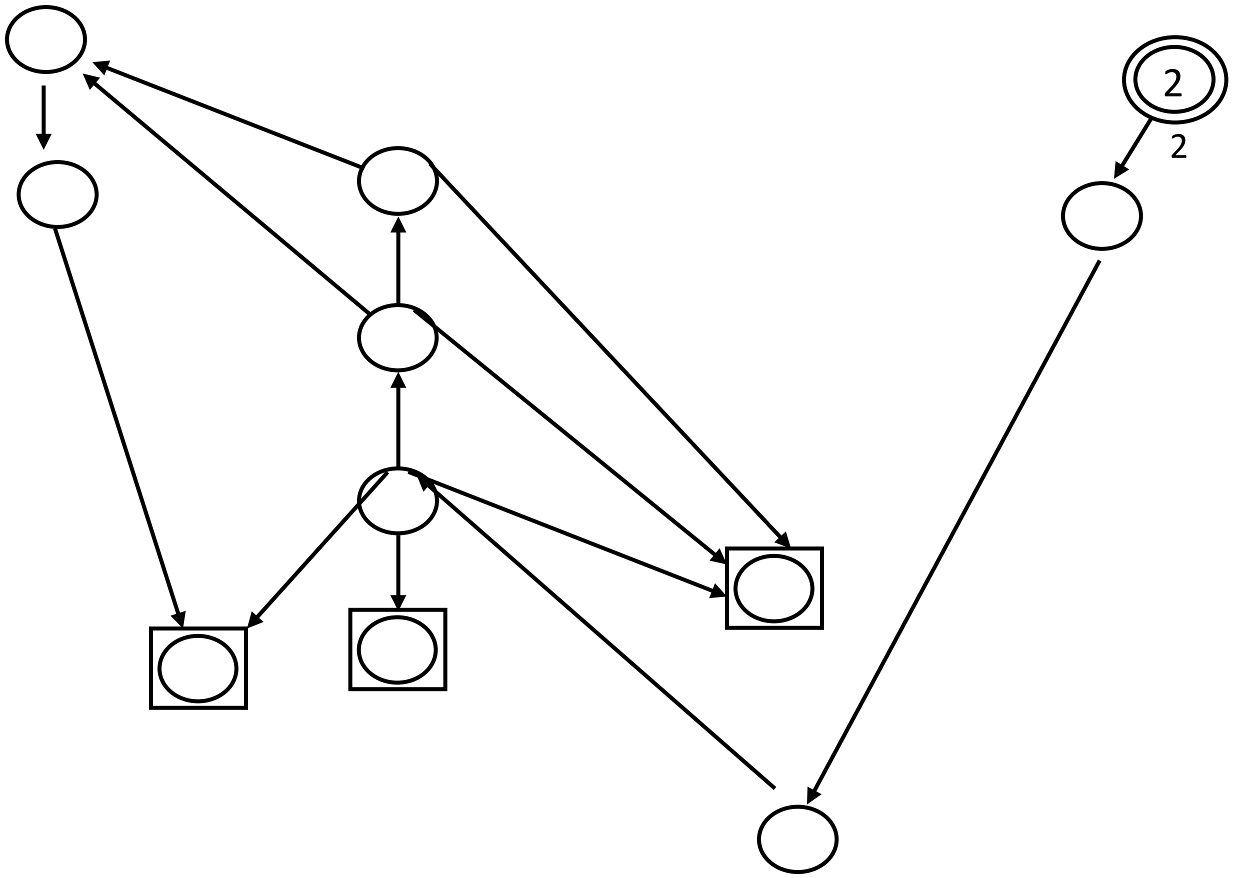


YO: send up



FLIP

Paola Flocchini



FLIP

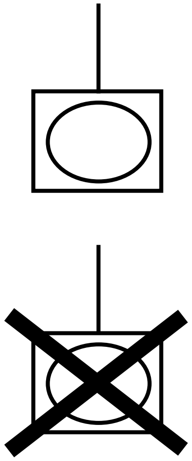
Termination

?

PRUNING

Pruning Rules

1)

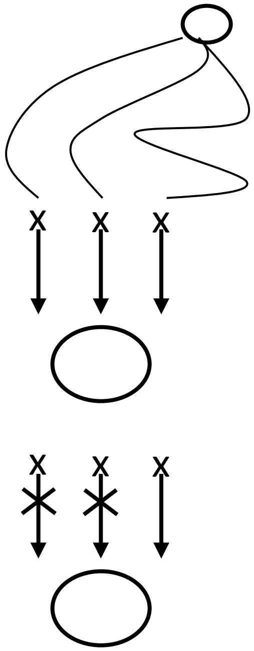


The sinks with one incoming link can be eliminated

(The decision of a sink is the same as the one of the parent)

Pruning Rules

2)



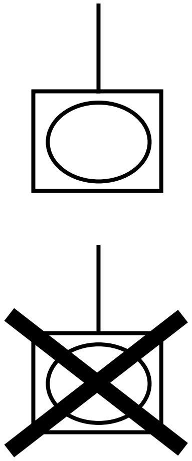
Nodes that receive several identical minimal values:

All the links with the same value can be eliminated except one
(They come from the same source - the decision is the same)

redundancy

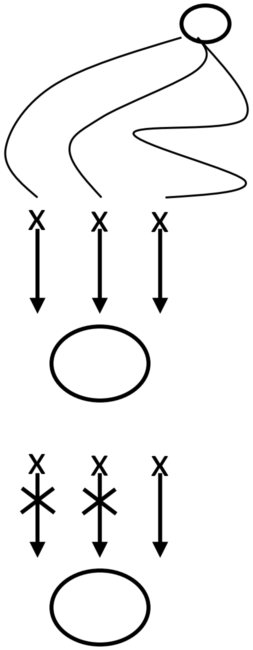
Pruning Rules: in the Yo-up

1)



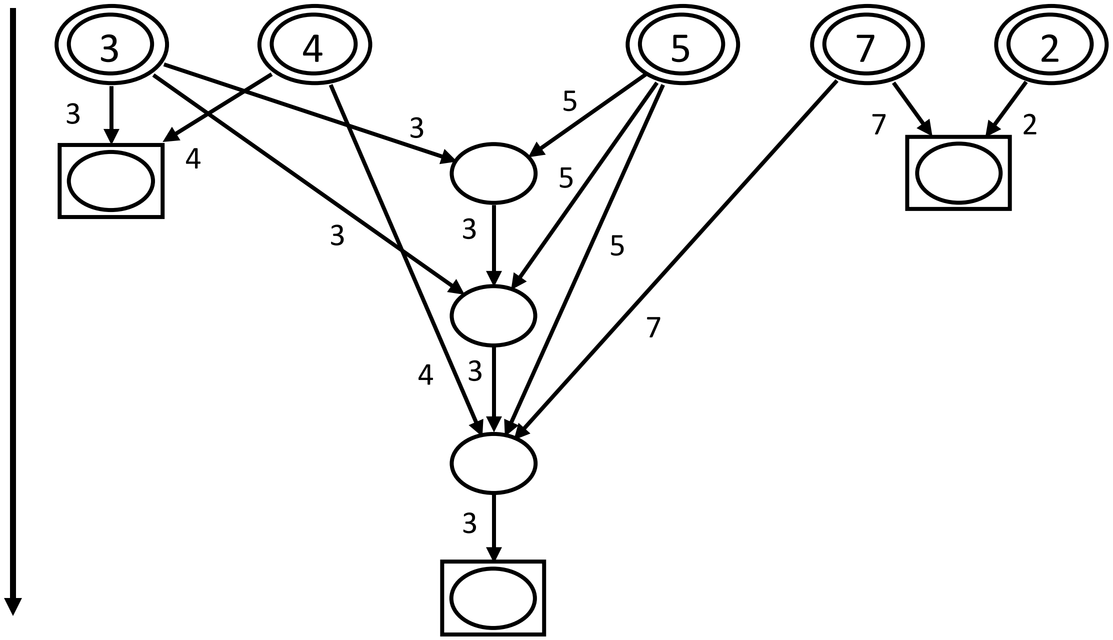
(prune me)

2)

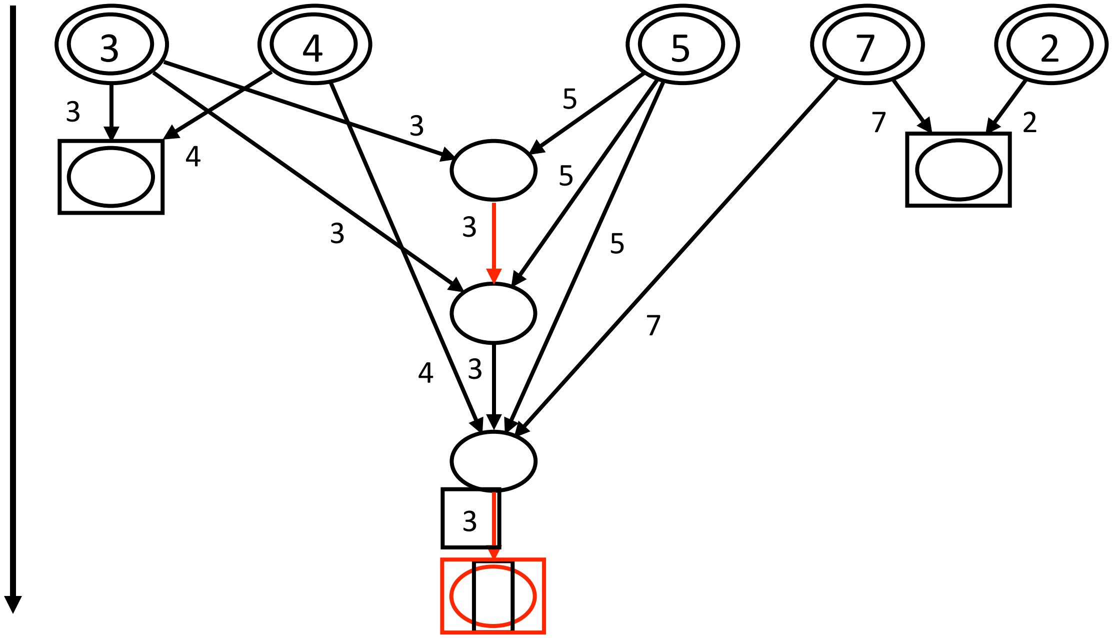


Prune only the redundant YES.
Do not prune the redundant NO
(careful, the book contains a mistake)

Yes (prune me)

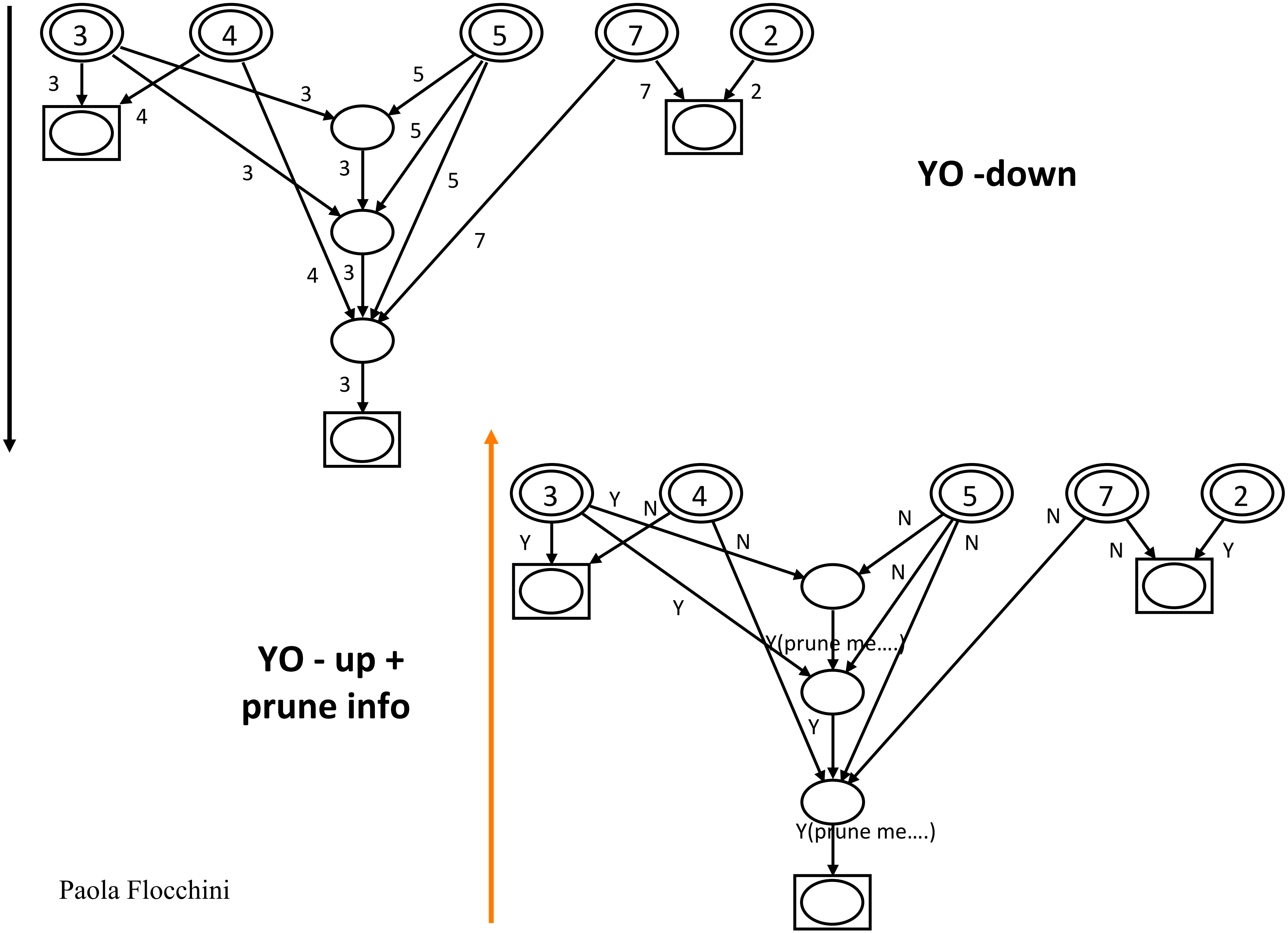


YO -down



YO -down

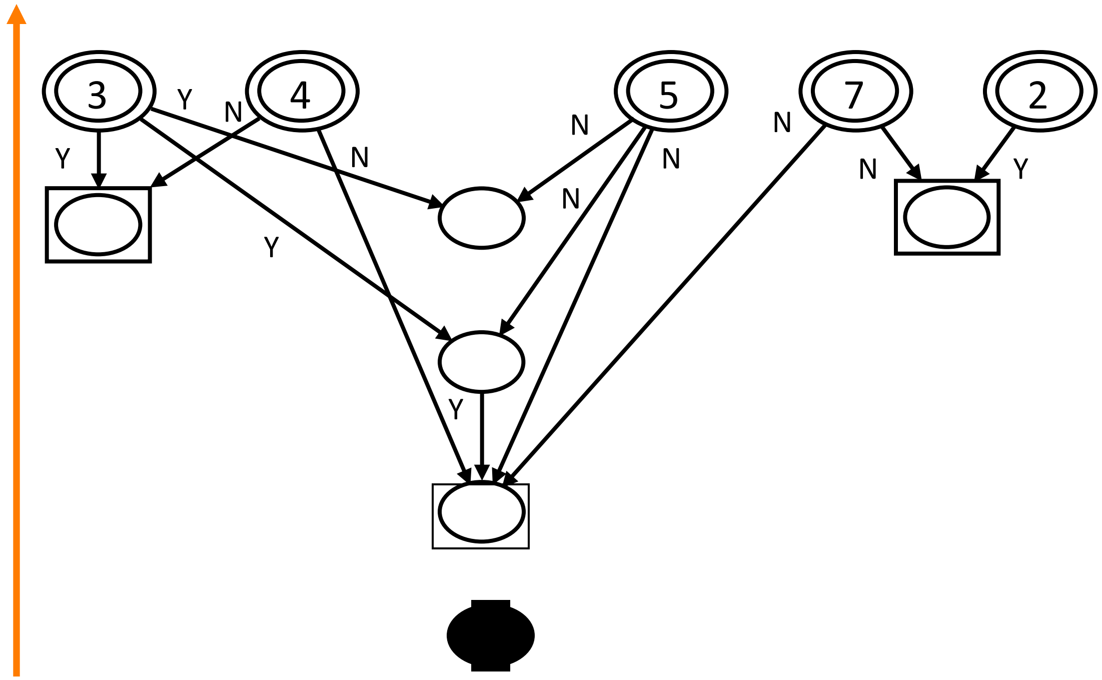
YO - up +
prune info

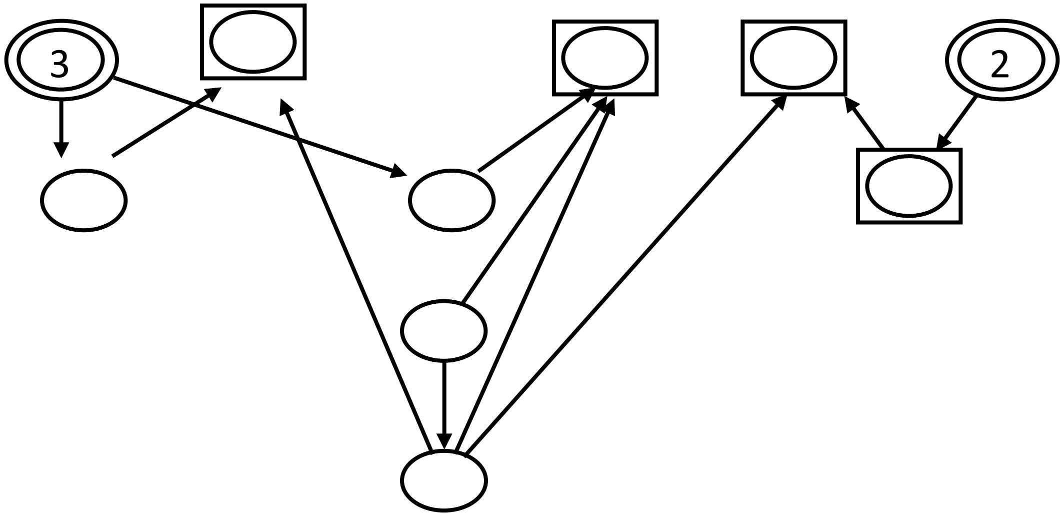


YO -down

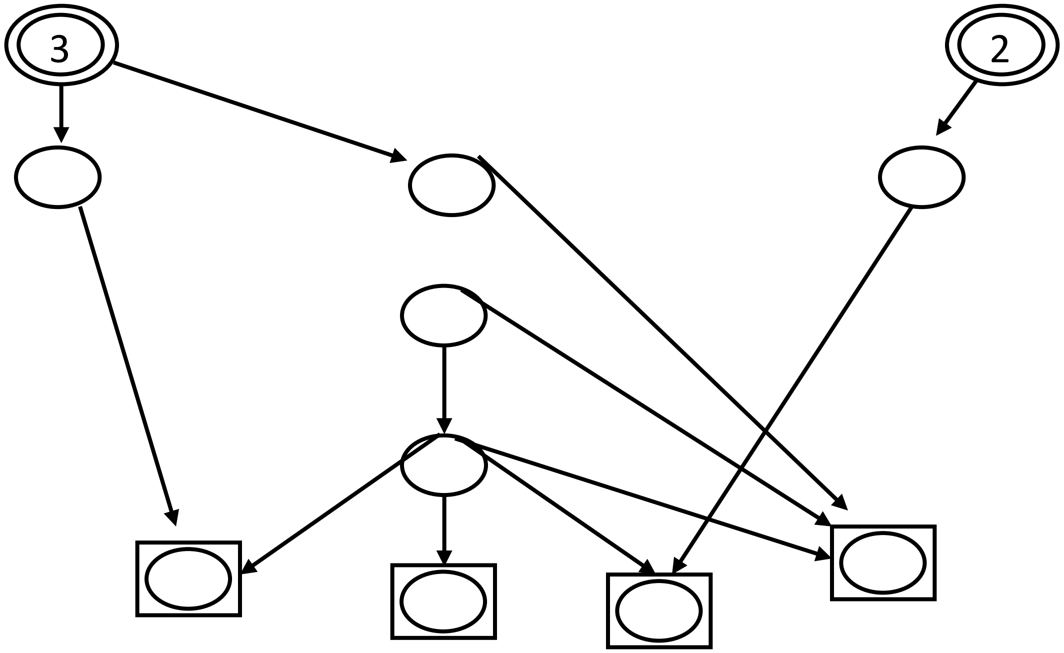
YO - up + prune info

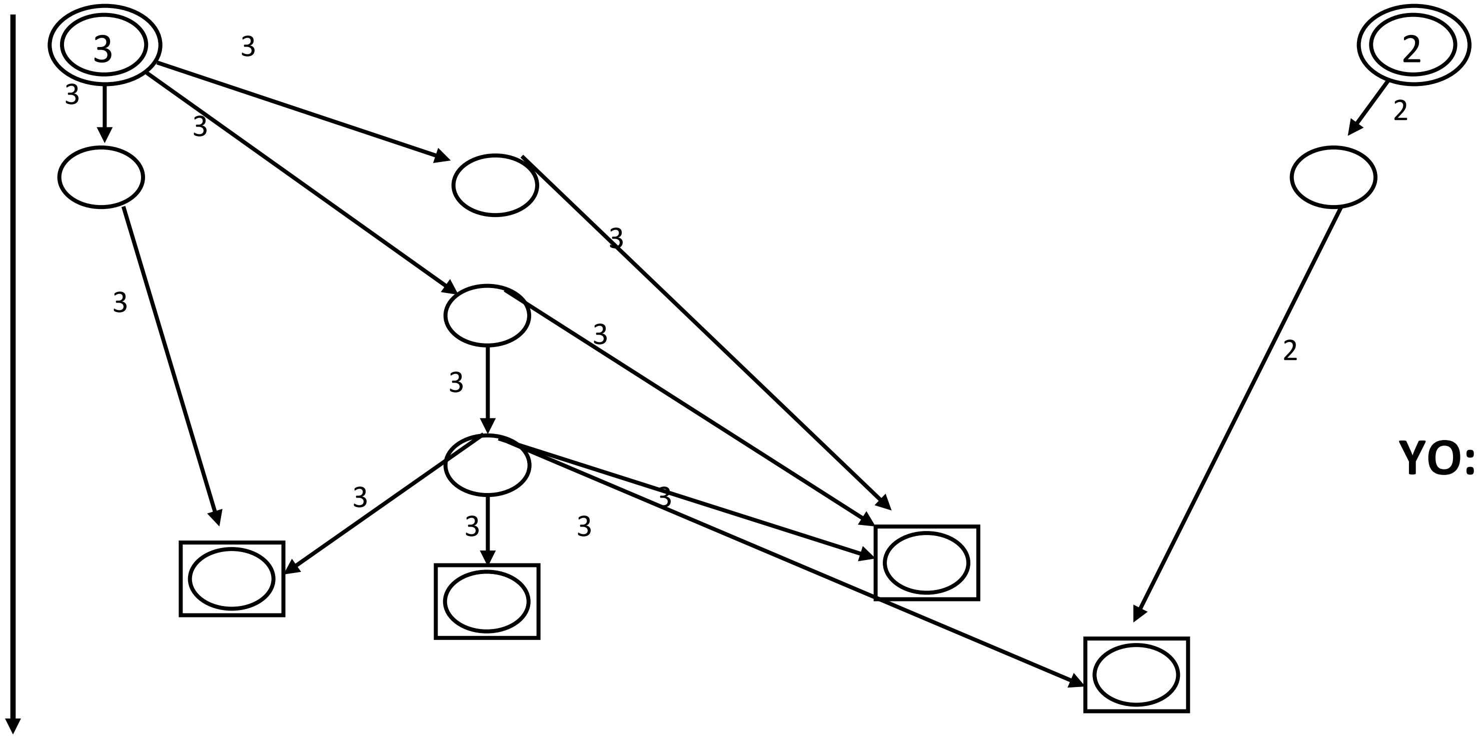
At the same time …. prune



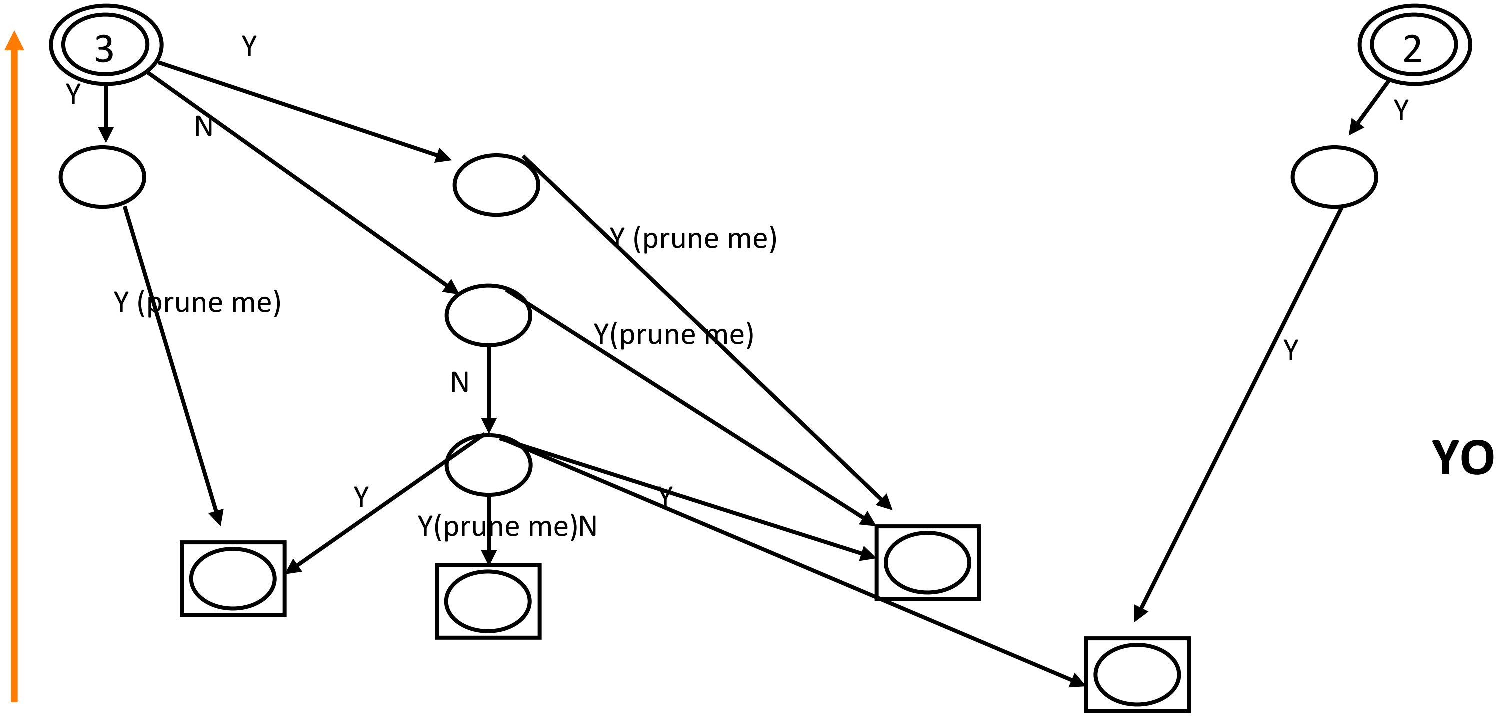


FLIP



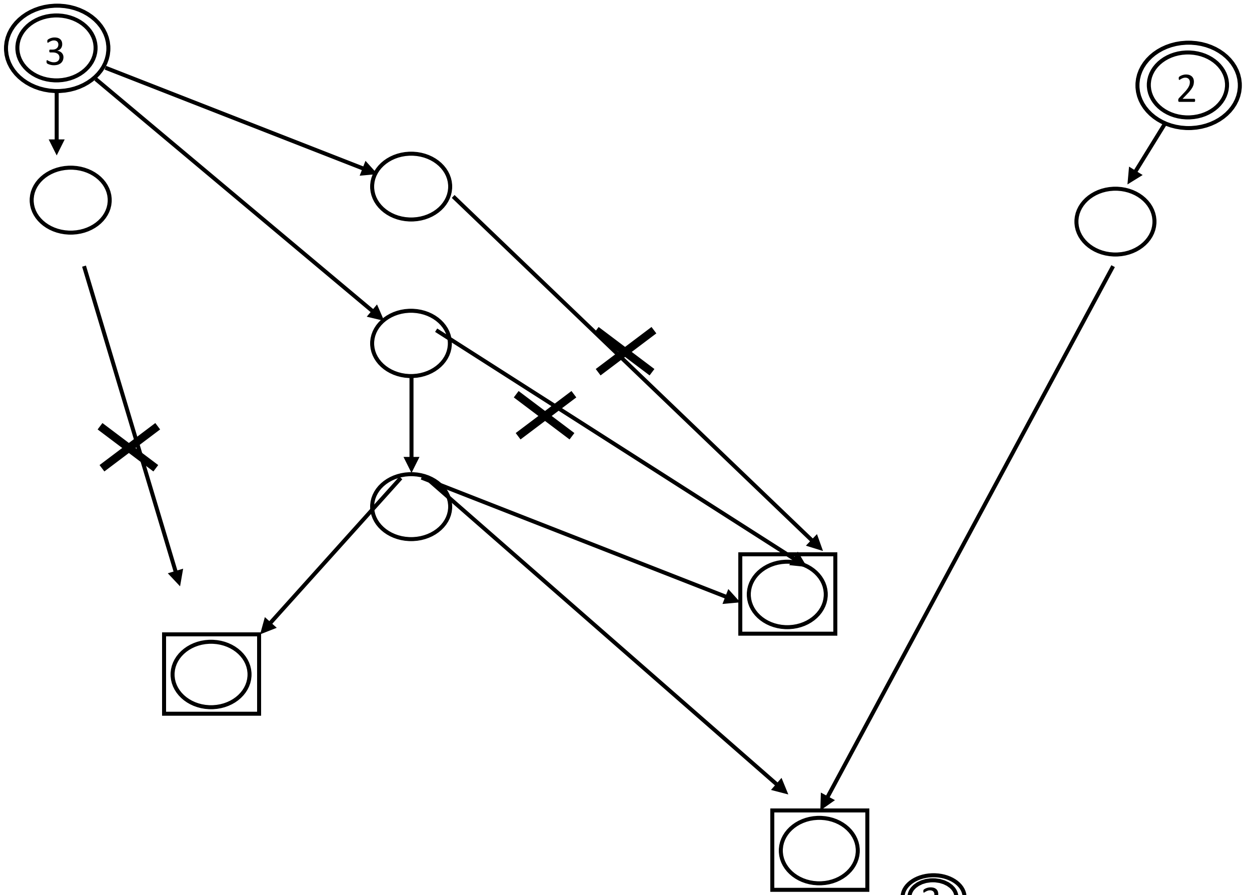


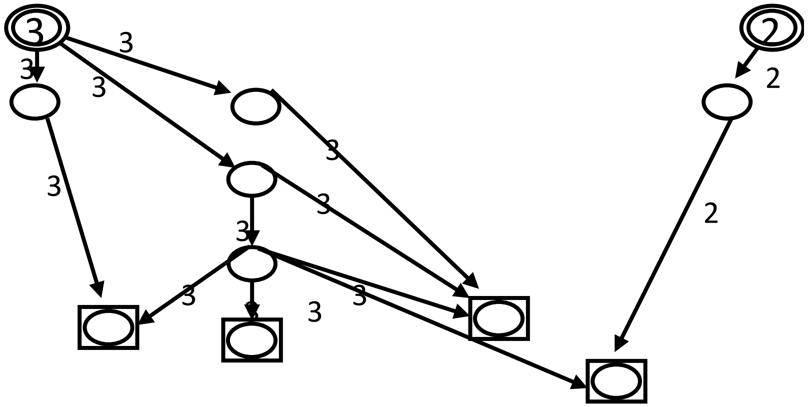
YO: send down



YO: send up

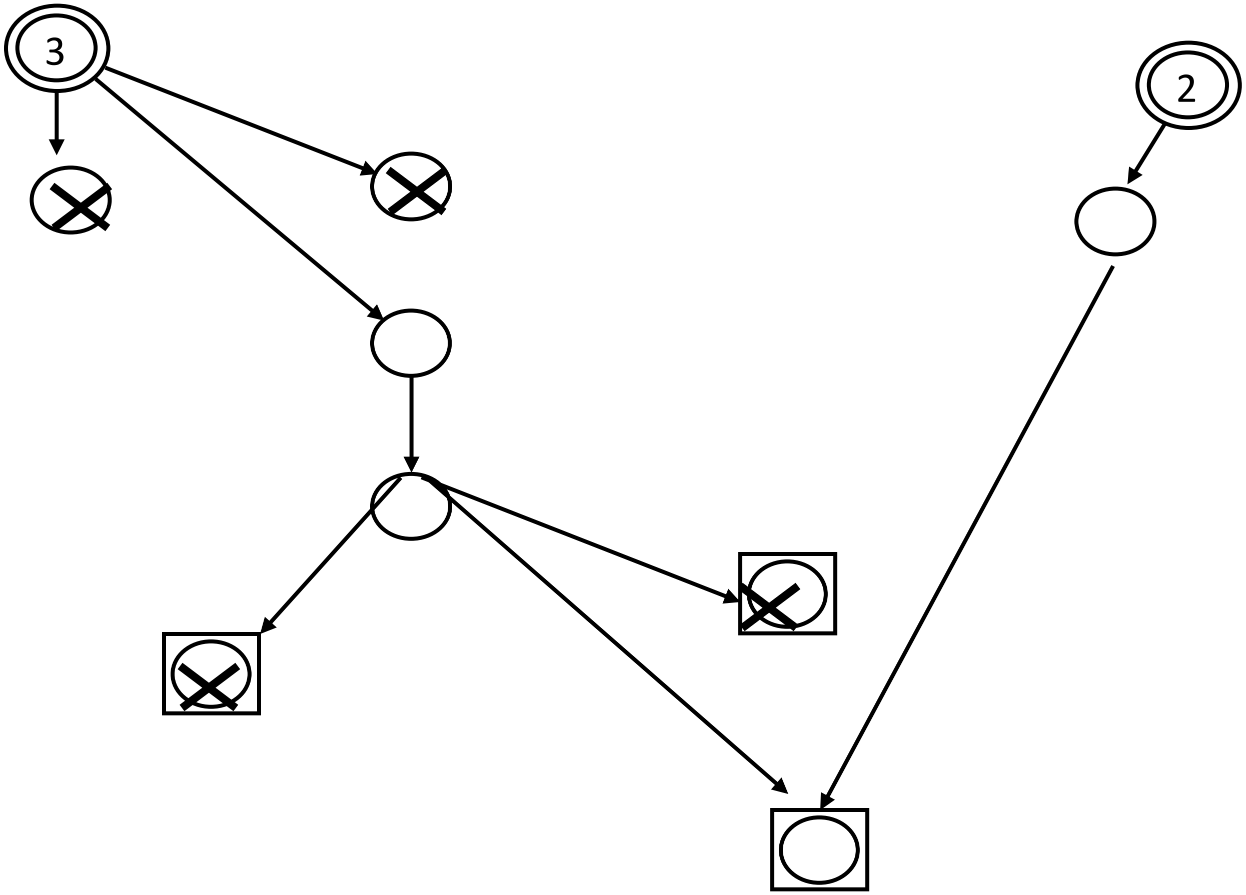
Paola Flocchini



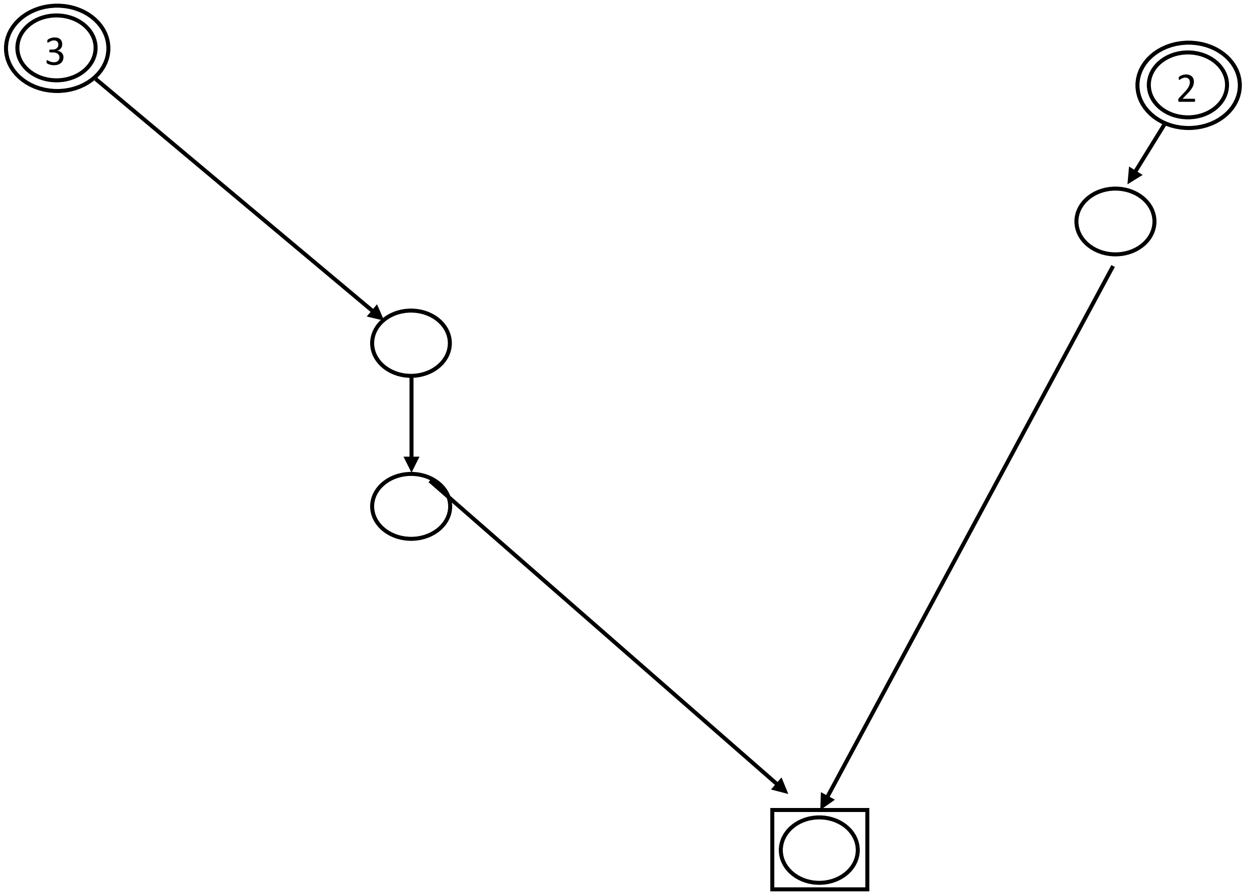


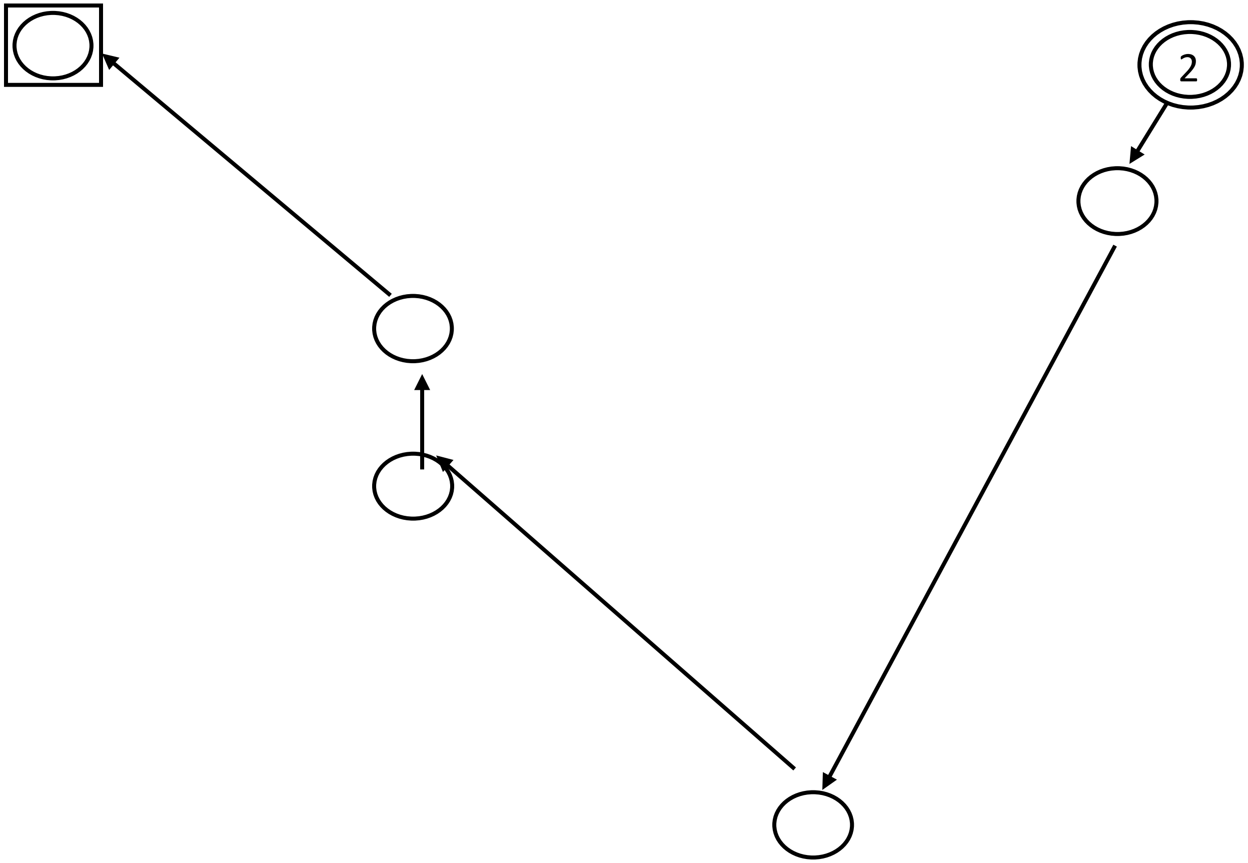
At the same time …. prune

Paola Flocchini

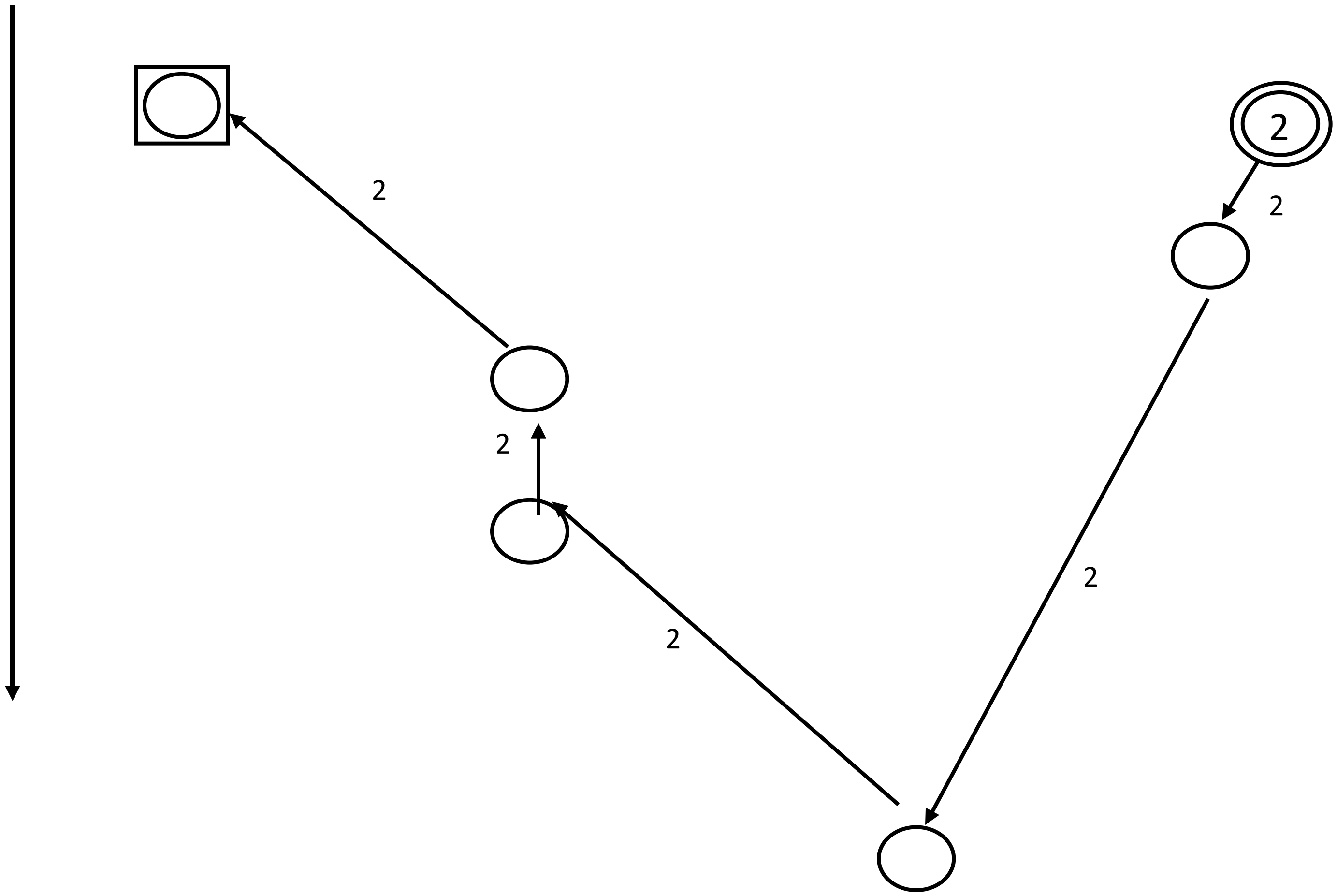


… More pruning

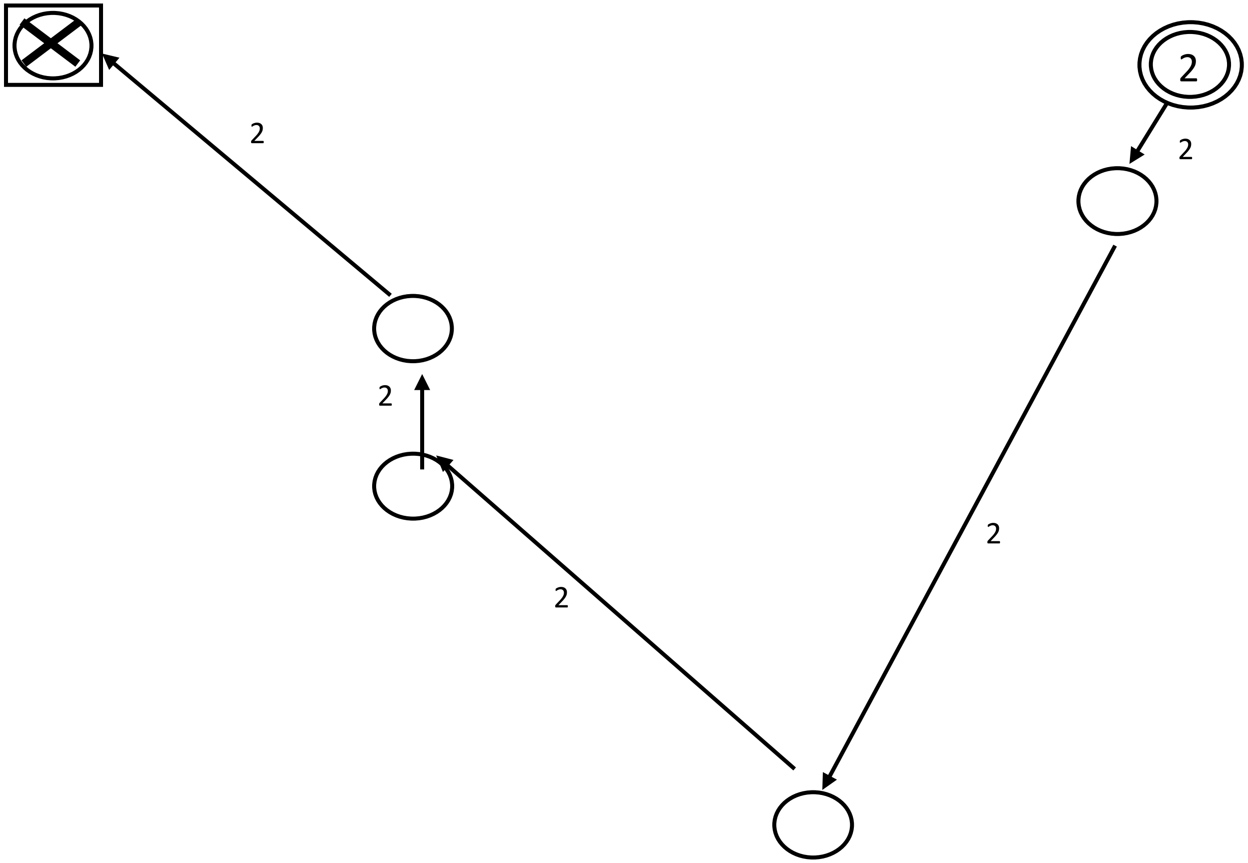

3
2



FLIP

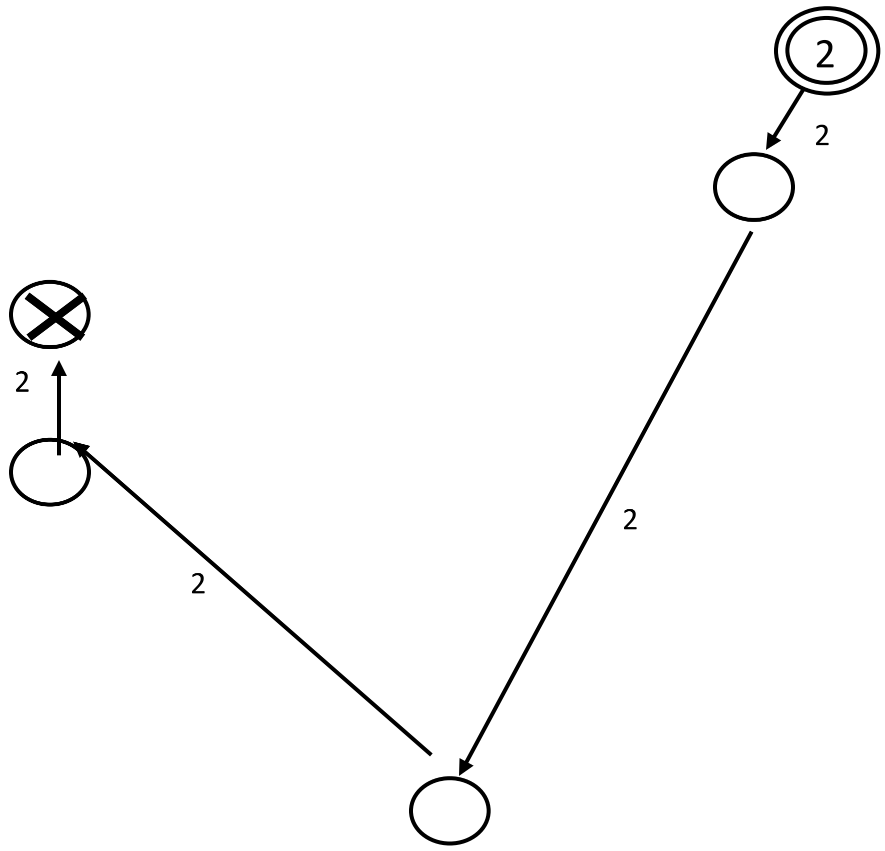


YO: send down



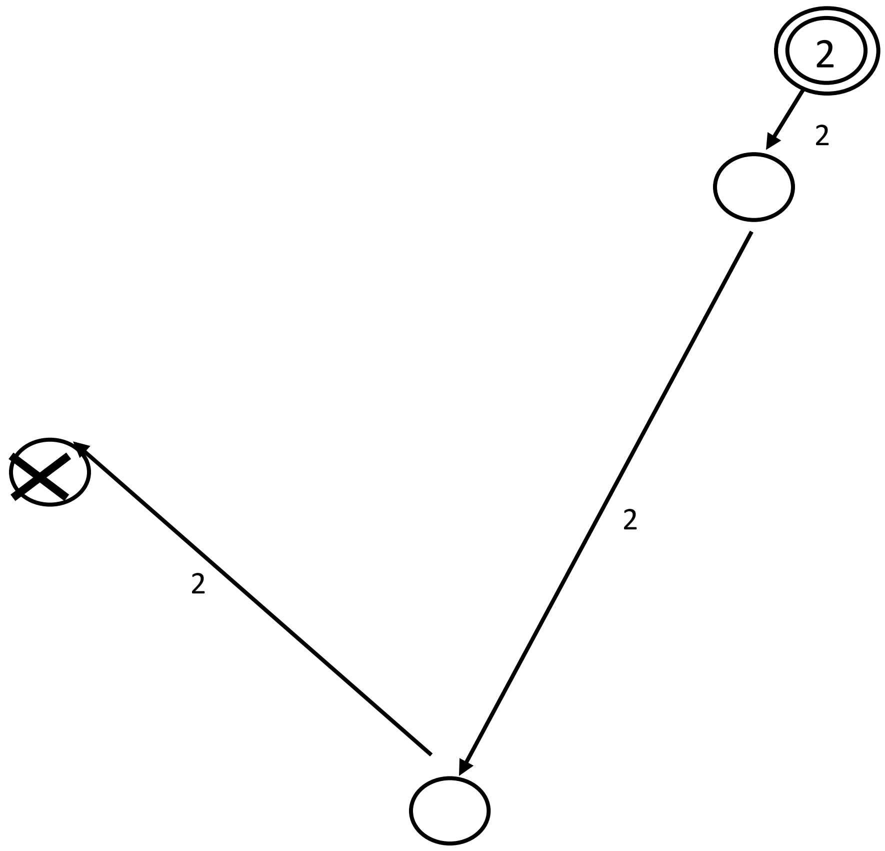
PRUNE

Paola Flocchini



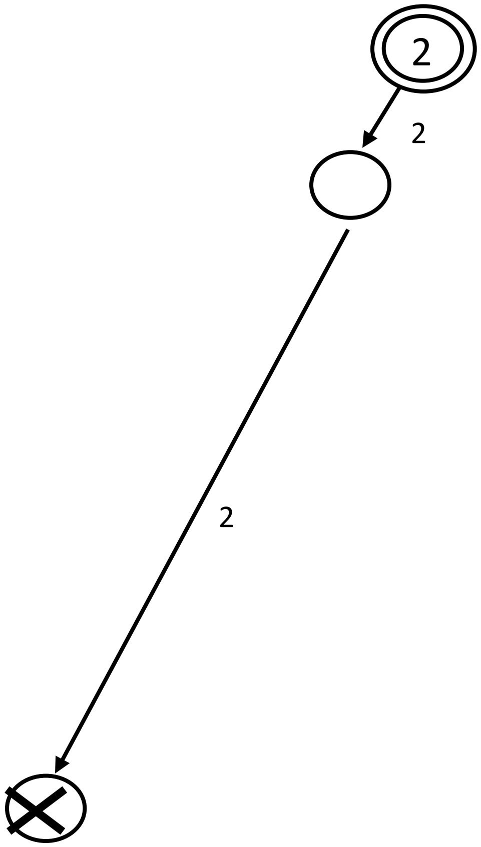
PRUNE

Paola Flocchini

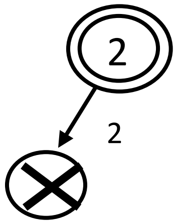


PRUNE

Paola Flocchini



PRUNE



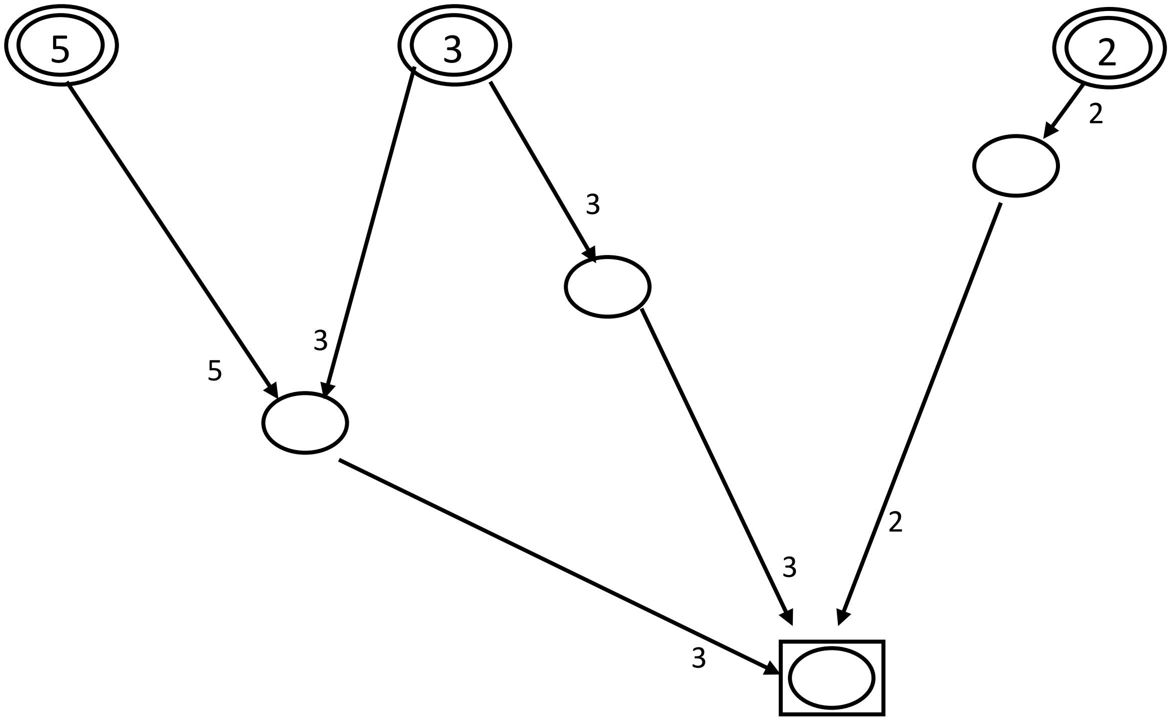
PRUNE

2

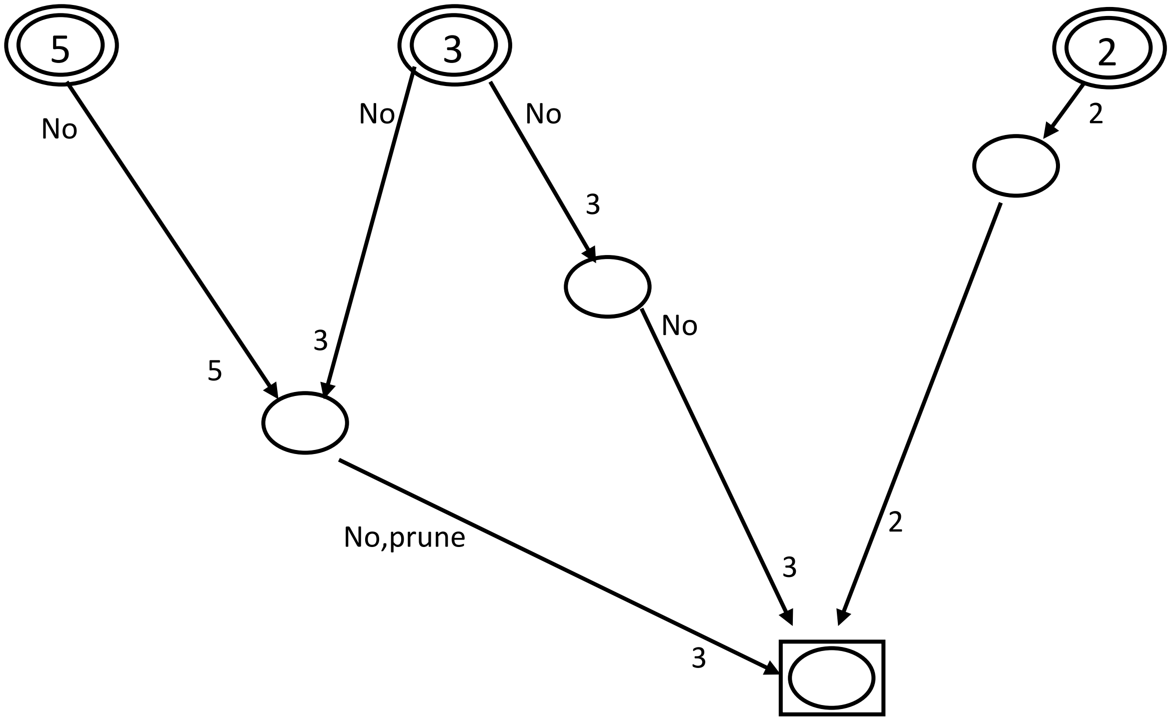
PRUNE

Paola Flocchini

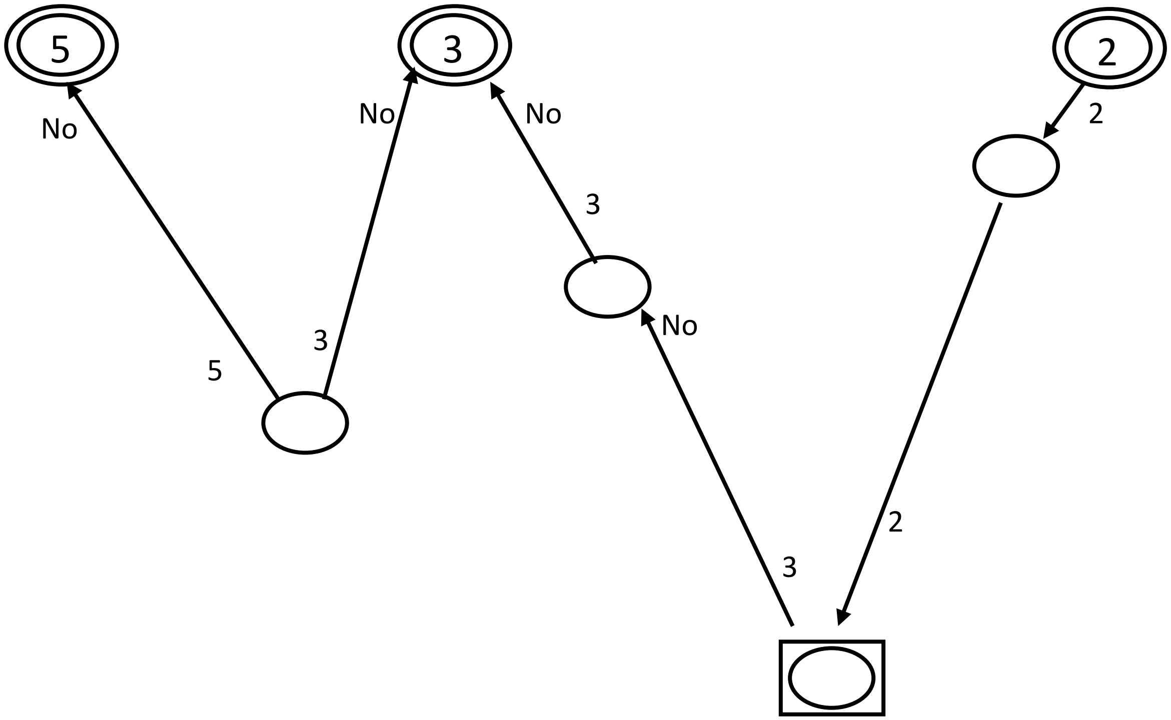
Why not pruning redundant « no »



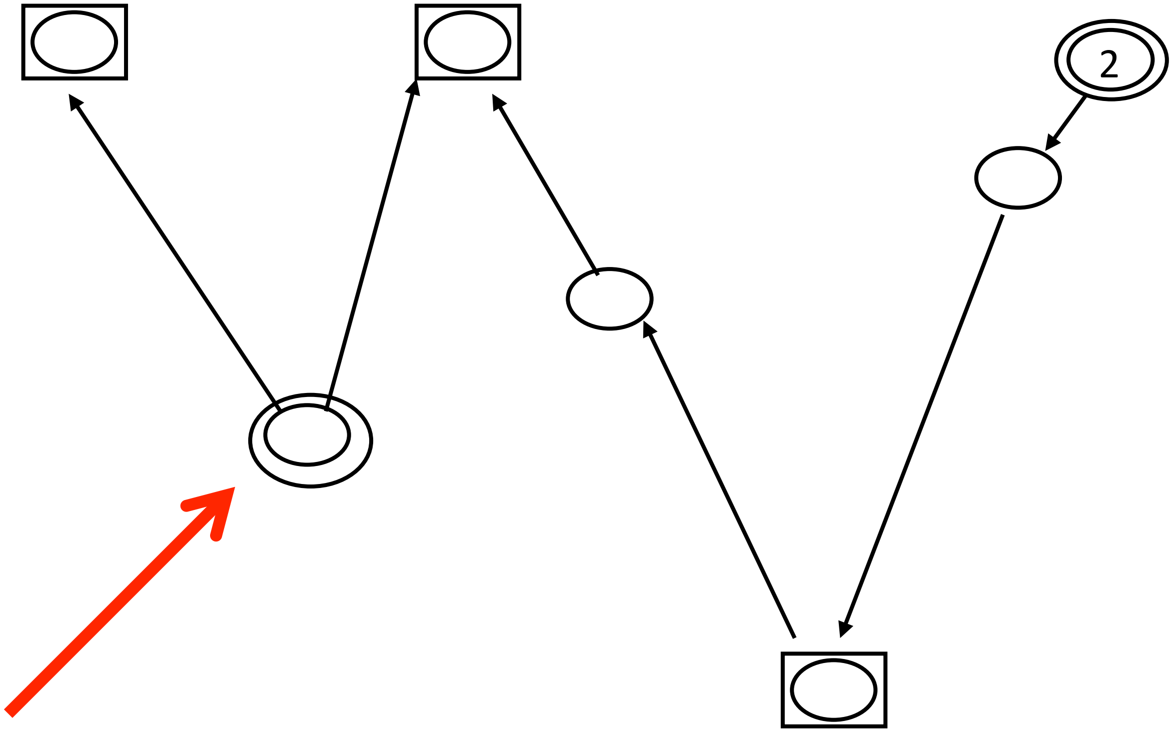
Why not pruning redundant « no »



Why not pruning redundant « no »



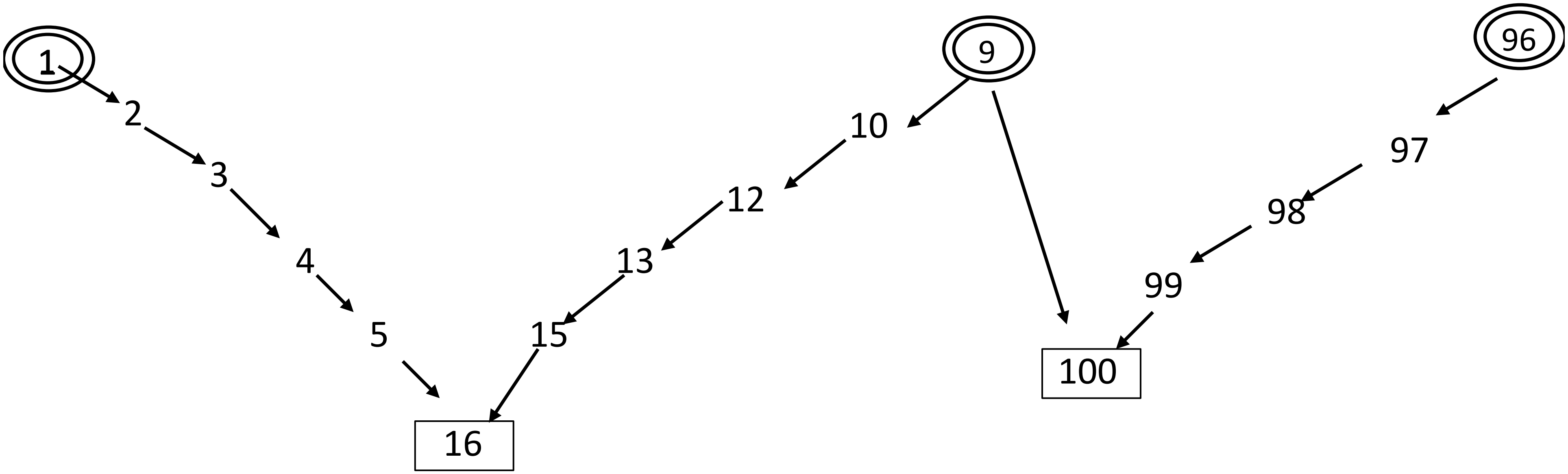
Why not pruning redundant « no »



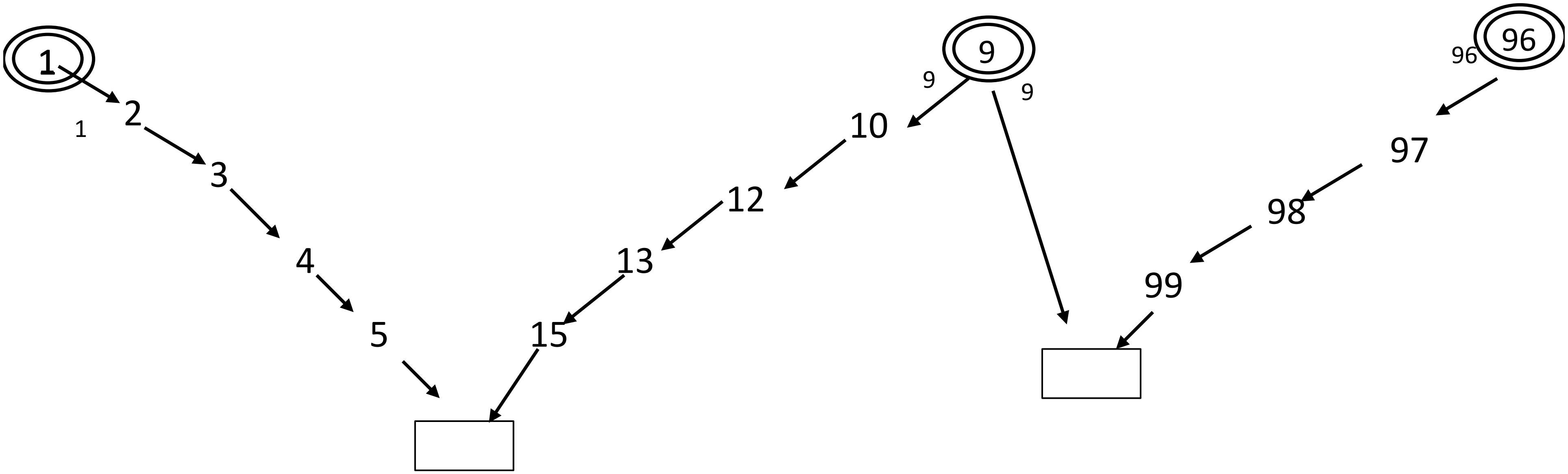
Examples: in a line

1 2 3 4 5 16 15 13 12 10 9 100 99 98 97 96

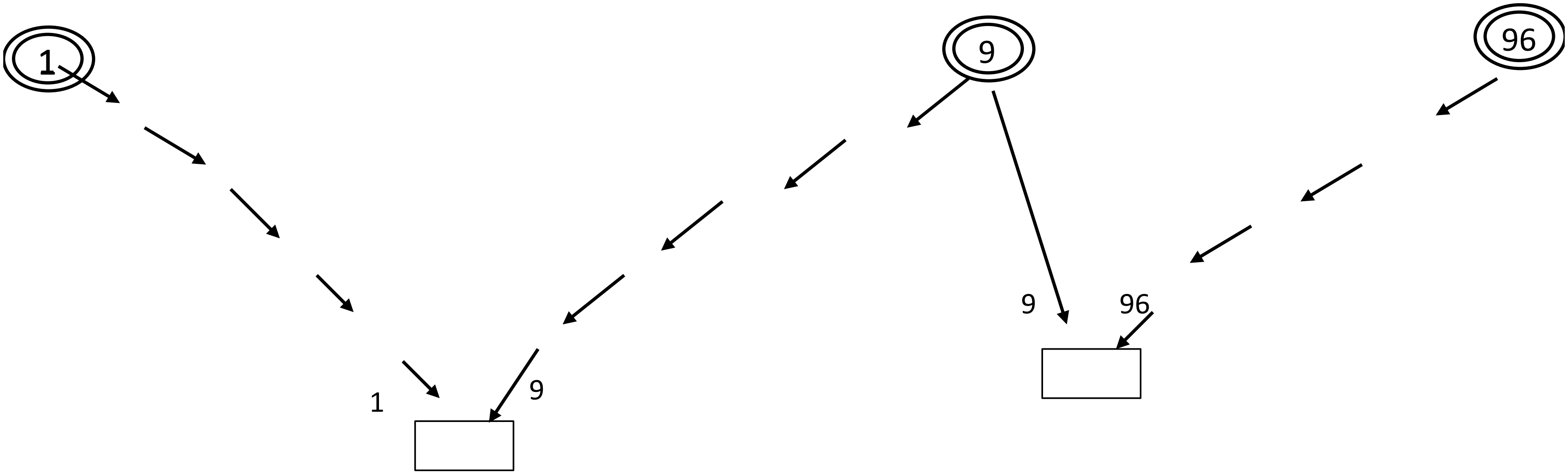
Examples: in a line



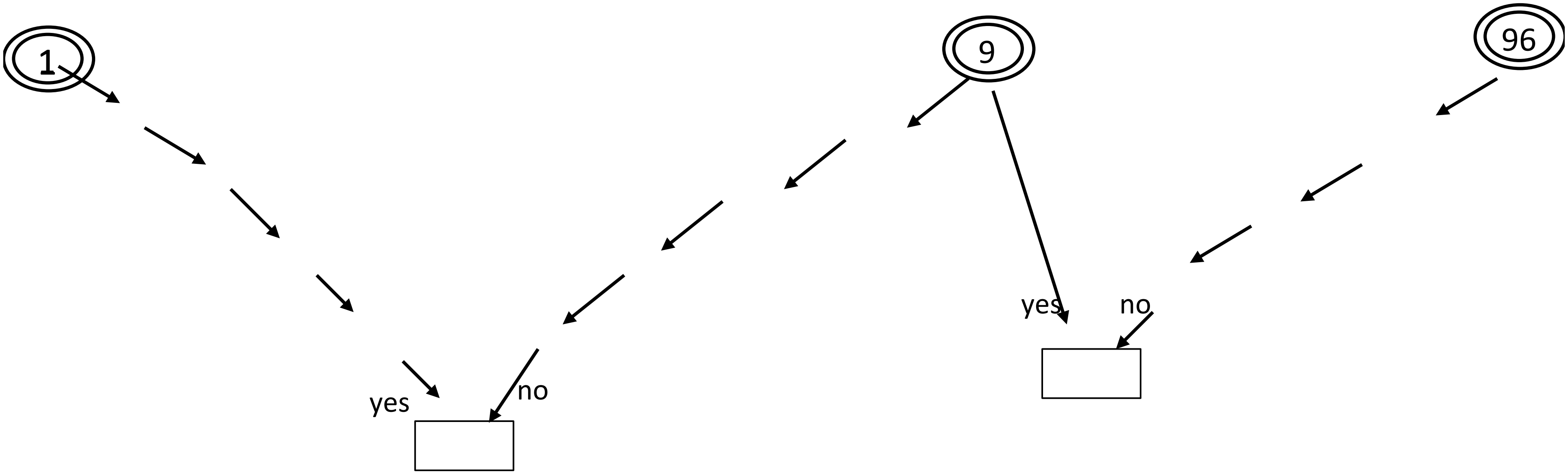
Examples: in a line



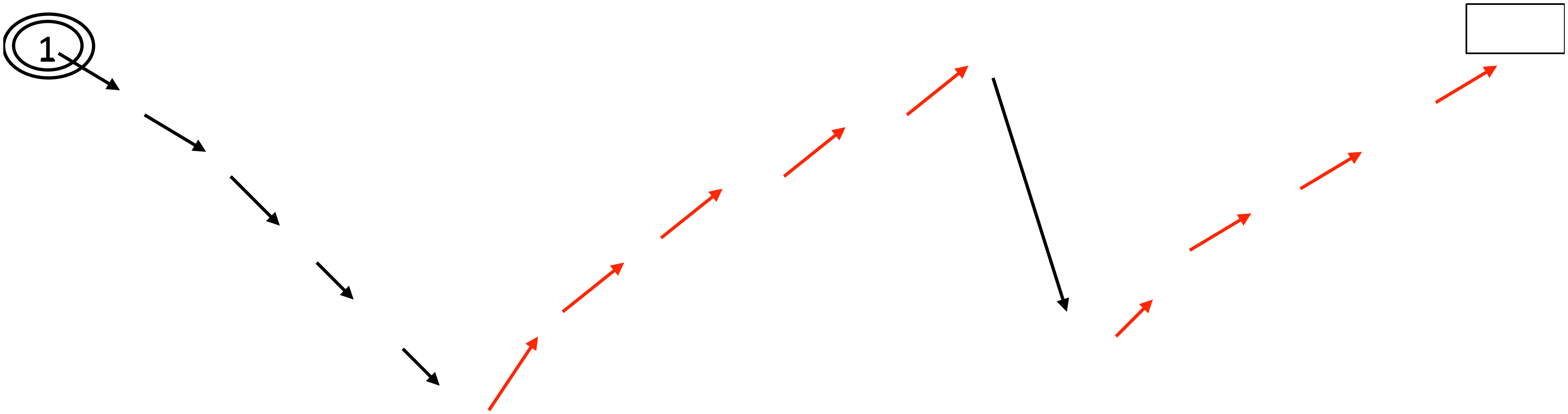
Examples: in a line



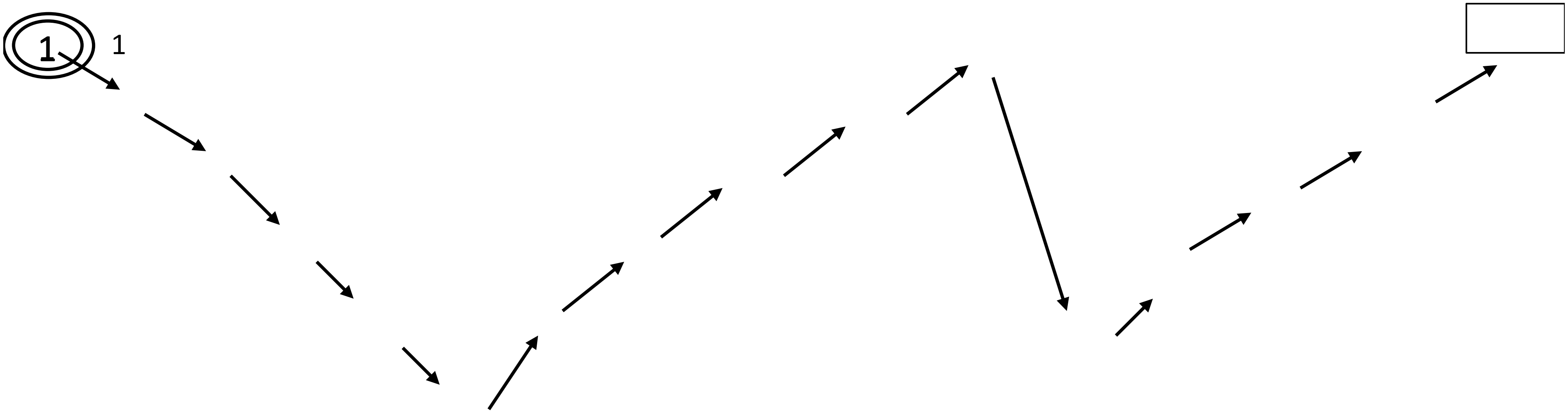
Examples: in a line



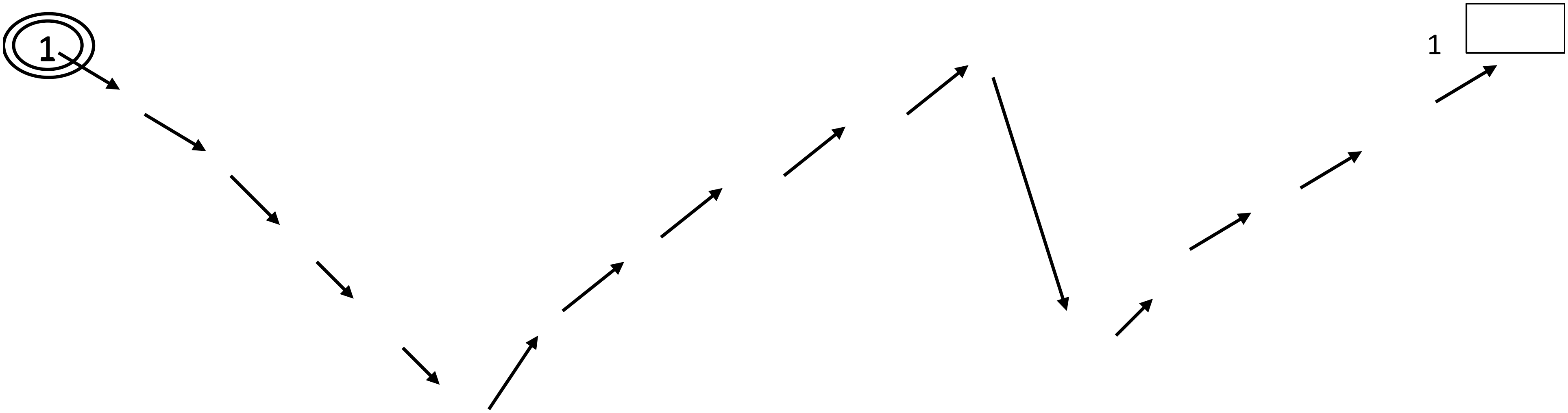
Examples: in a line



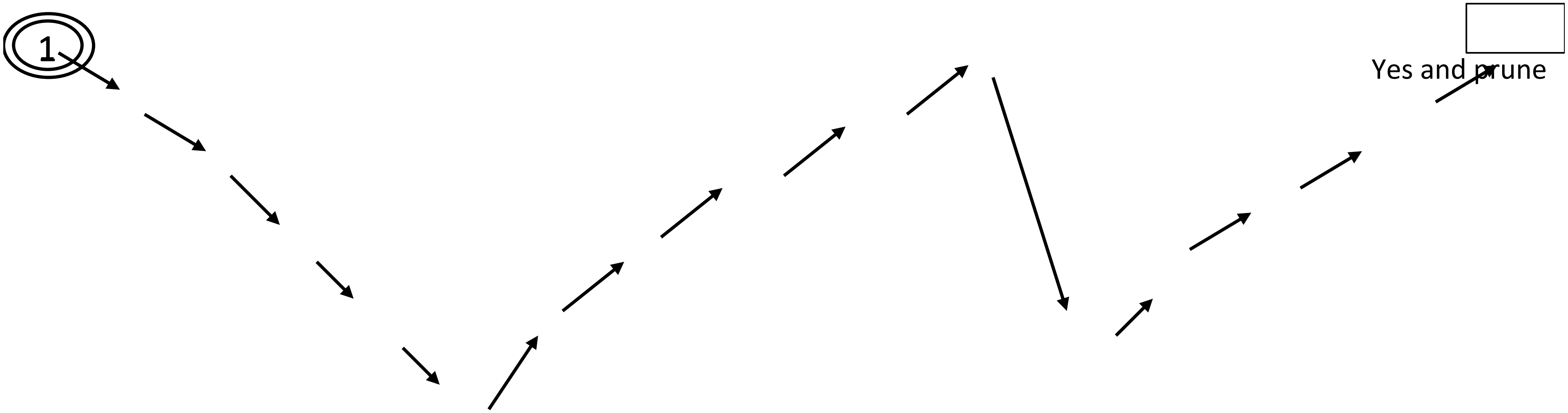
Examples: in a line



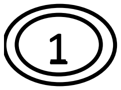
Examples: in a line



Examples: in a line



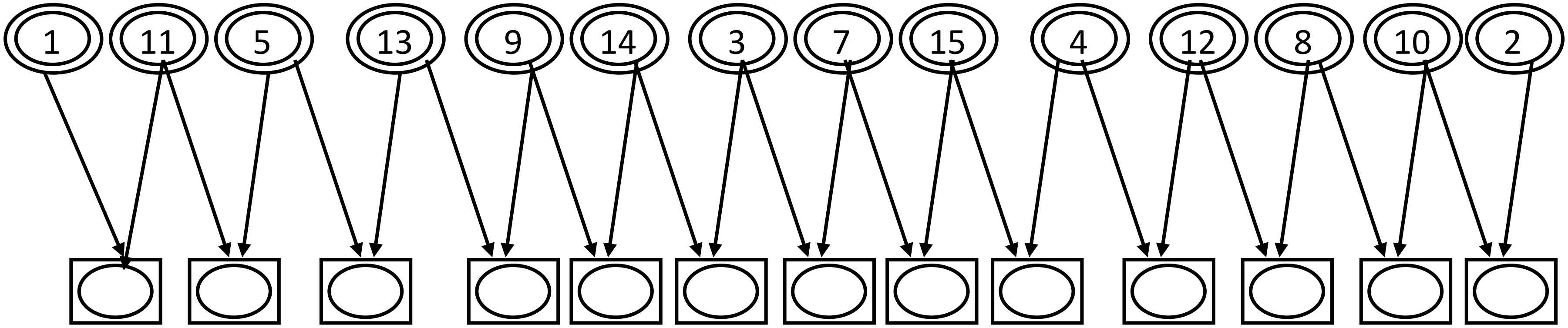
Examples: in a line



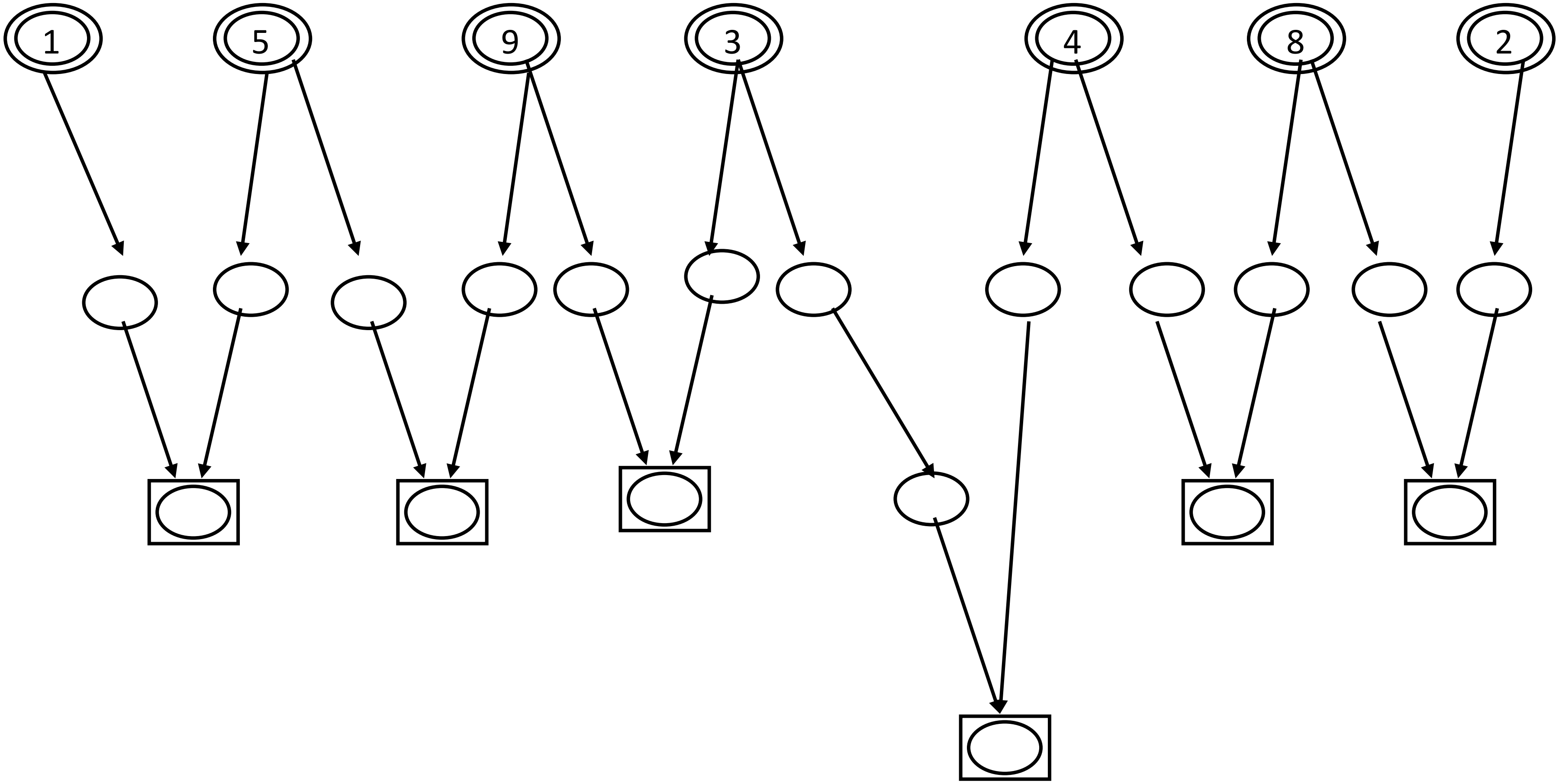
Examples: in a line

1 12 8 15 4 11 6 13 3 10 7 14 5 9 16 2

Worst case ?



Worst case ?



Complexity

Without Considering Pruning

There are 2 messages for each link at each phase

- The number of phases is: log(# sources)

(s = # sources)

TOT: 2m + 2m log s

For initialization

With Pruning: ?