Distributed Computing
PWr, WIT, 2021

Prof. Mirosław Kutyłowski

2- BFS and spanning tree

Broadcasting to the whole network

Message complexity:

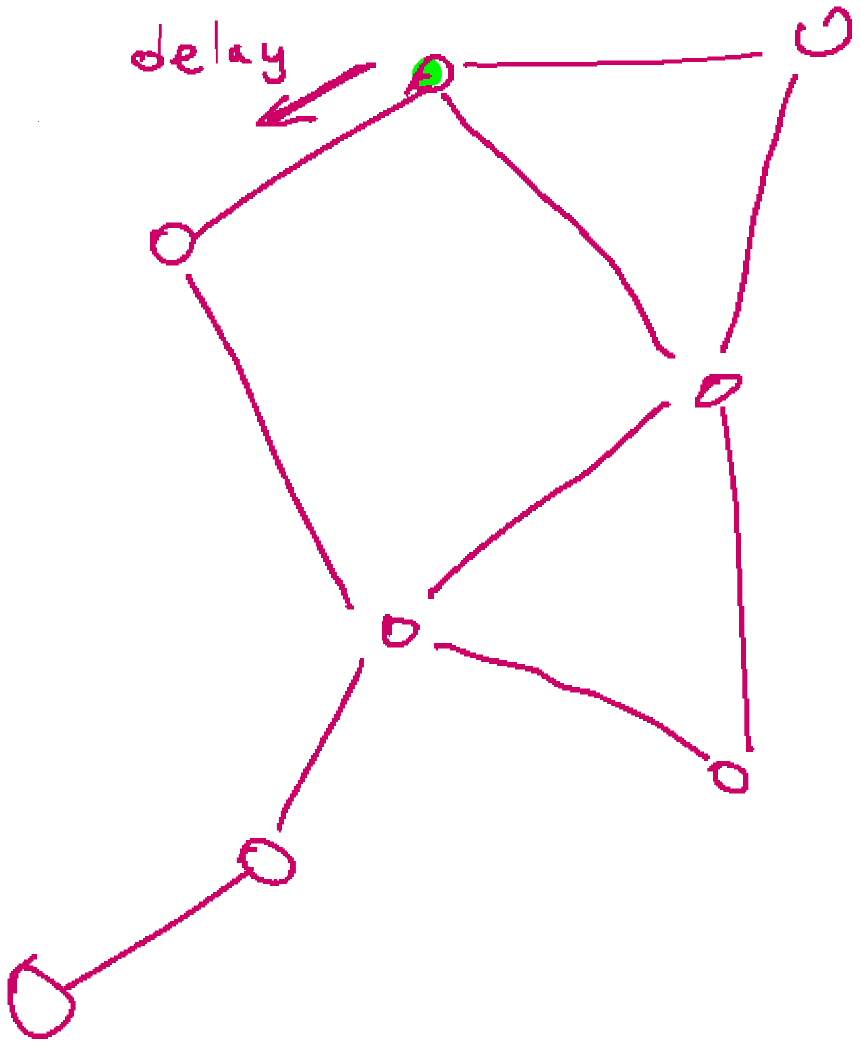
The total number of messages sent

Time complexity

time needed so that ~~all nodes~~ get the message

Assumptions:

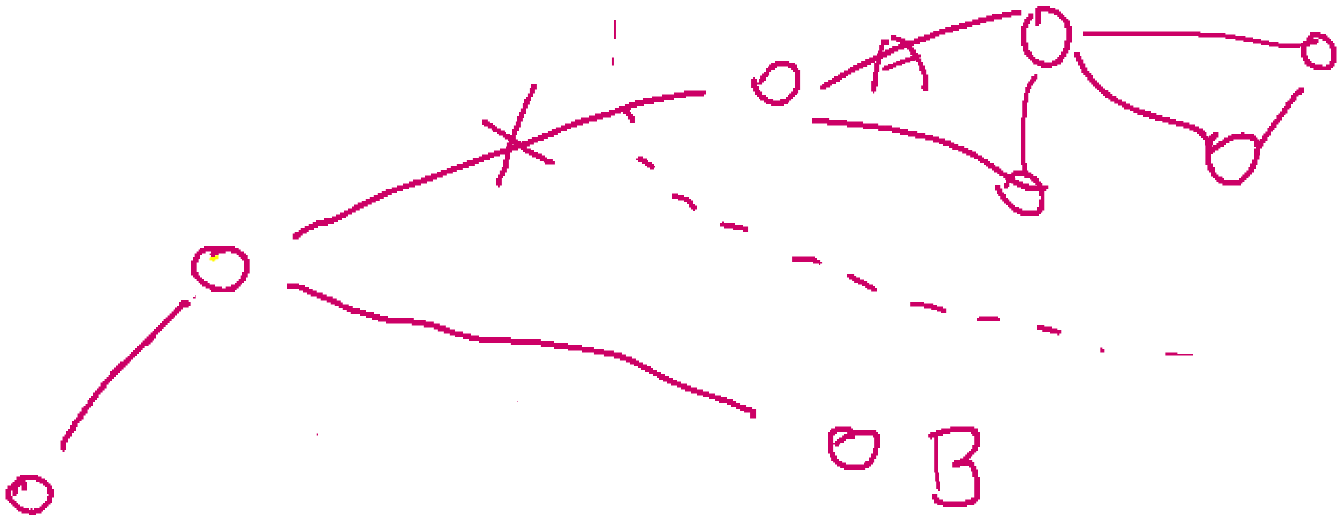
- asynchronous , no global clock
- architecture unknown



Broadcasting by flooding

Algorithm 2.9 Flooding

1: The source (root) sends the message to all neighbors.
2: **Each other node** v upon receiving the message the first time forwards the message to all (other) neighbors.
3: Upon later receiving the message again (over other edges), a node can discard the message.

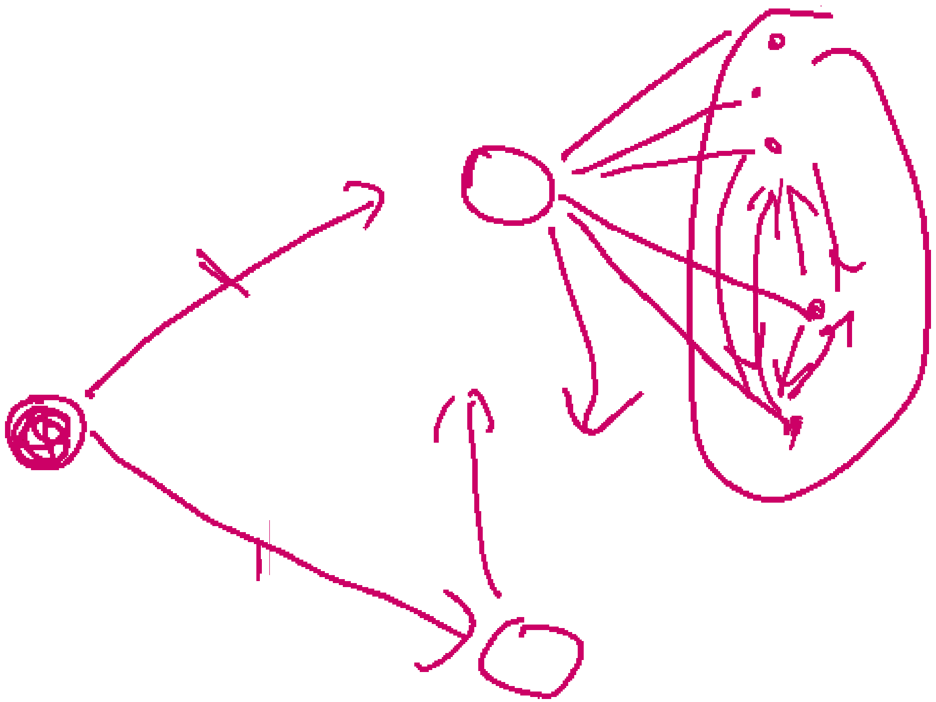


Flooding

Algorithm 2.9 Flooding

1: The source (root) sends the message to all neighbors.
2: **Each other node** v upon receiving the message the first time forwards the message to all (other) neighbors.
3: Upon later receiving the message again (over other edges), a node can discard the message.

~~Problem: message complexity~~



Flooding

Algorithm 2.9 Flooding

1: The source (root) sends the message to all neighbors.
2: **Each other node** v upon receiving the message the first time forwards the message to all (other) neighbors.
3: Upon later receiving the message again (over other edges), a node can discard the message.

Problem: message complexity

każda krawędź użyta

krawędzi

$\Rightarrow$ złożoność = # ~~wierzchołków~~

Flooding

Algorithm 2.9 Flooding

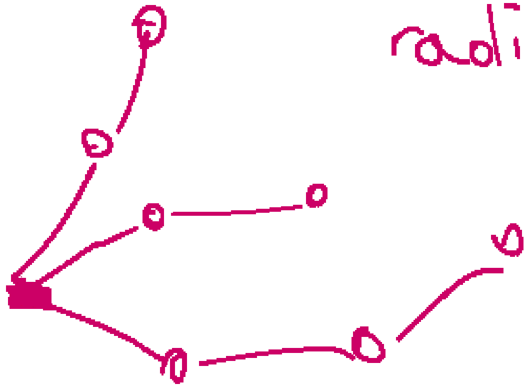
1: The source (root) sends the message to all neighbors.
2: **Each other node** v upon receiving the message the first time forwards the message to all (other) neighbors.
3: Upon later receiving the message again (over other edges), a node can discard the message.

time

Problem: ~~message complexity~~

czas = średnica grafu

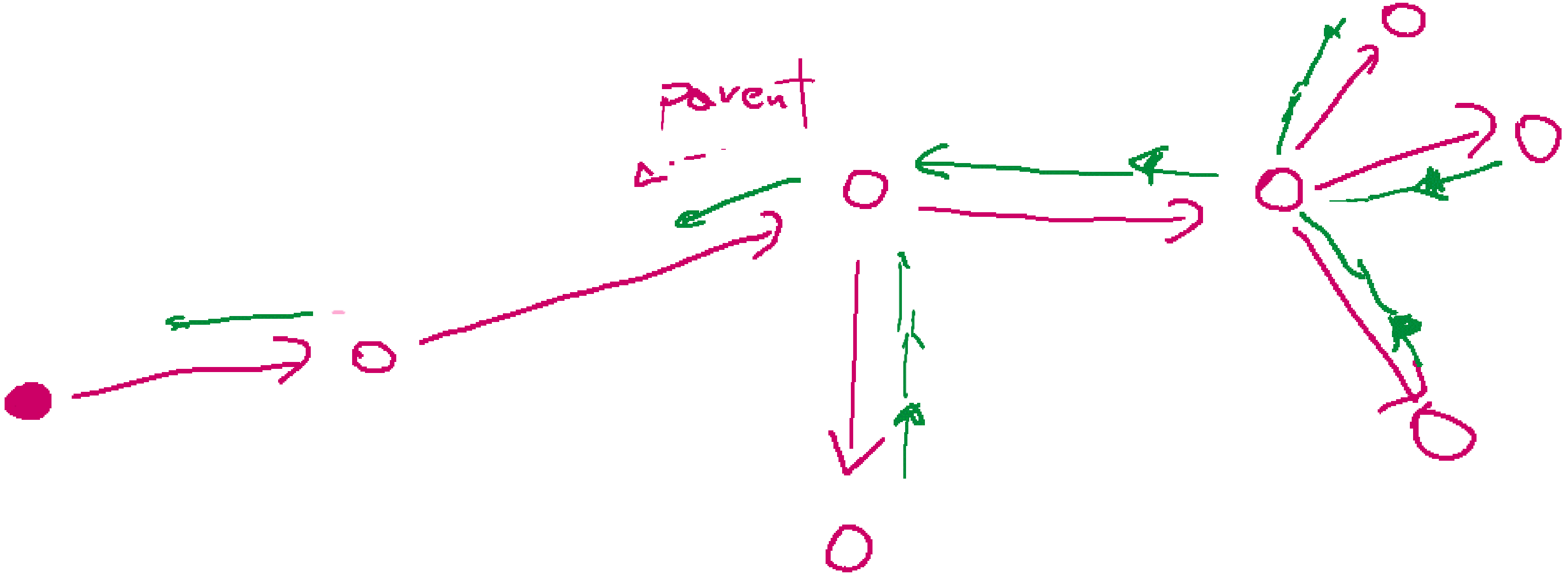
radius



Echo

Algorithm 2.10 Echo

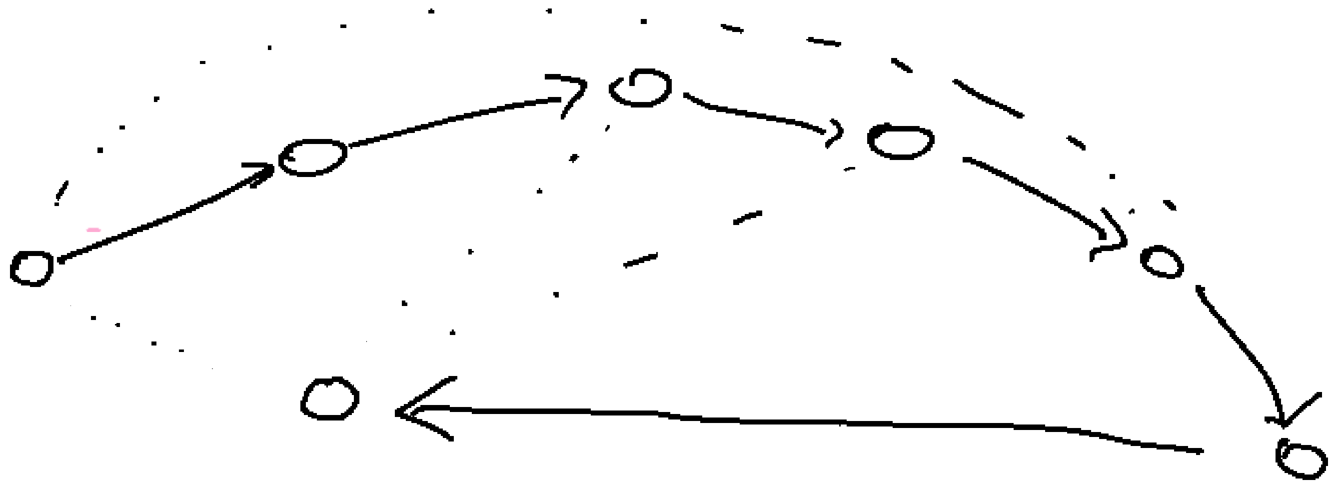
1: A leave sends a message to its parent.
2: If an inner node has received a message from each child, it sends a message to the parent.



Echo

Algorithm 2.10 Echo

1: A leave sends a message to its parent.
2: If an inner node has received a message from each child, it sends a message to the parent.



Bellman-Ford BFS (broad first search)

Find the shortests paths to the root

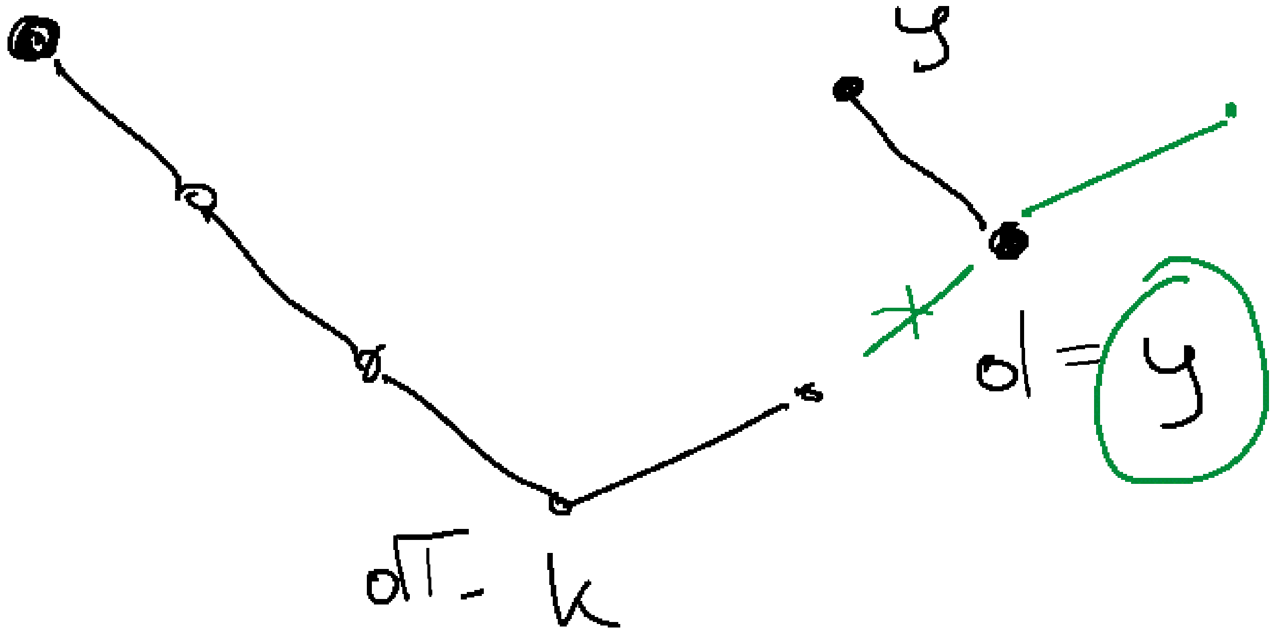
Algorithm 2.13 Bellman-Ford BFS

Each node u stores an integer d_u which corresponds to the distance from u to the root. Initially $d_{\text{root}} = 0$, and $d_u = \infty$ for every other node u.
The root starts the algorithm by sending "1" to all neighbors.
if a node u receives a message "y" with $y < d_u$ from a neighbor v **then**
    node u sets $d_u := y$
    node u sends "$y + 1$" to all neighbors (except v)
end if

Bellman-Ford

Algorithm 2.13 Bellman-Ford BFS

Each node u stores an integer d_u which corresponds to the distance from u to the root. Initially $d_{\text{root}} = 0$, and $d_u = \infty$ for every other node u.
The root starts the algorithm by sending "1" to all neighbors.
if a node u receives a message "y" with $y < d_u$ from a neighbor v **then**
    node u sets $d_u := y$
    node u sends "$y + 1$" to all neighbors (except v)
end if



$y < k$

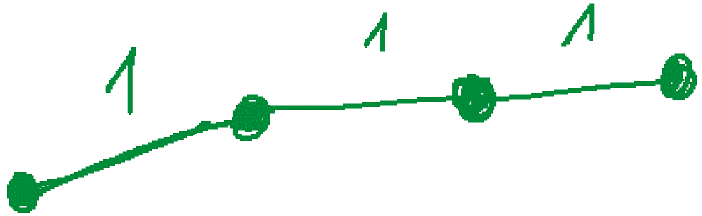
Bellman-Ford

Algorithm 2.13 Bellman-Ford BFS

Each node u stores an integer d_u which corresponds to the distance from u to the root. Initially $d_{\text{root}} = 0$, and $d_u = \infty$ for every other node u.
The root starts the algorithm by sending "1" to all neighbors.
if a node u receives a message "y" with $y < d_u$ from a neighbor v **then**
    node u sets $d_u := y$
    node u sends "$y + 1$" to all neighbors (except v)
end if

For a graph with diameter D, the number of edges m, and n nodes:
Time complexity O(D)
Message complexity O(mn)

time for 1 message: $\in [0, 1]$

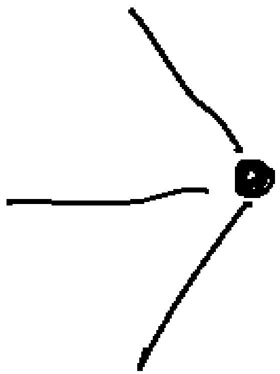


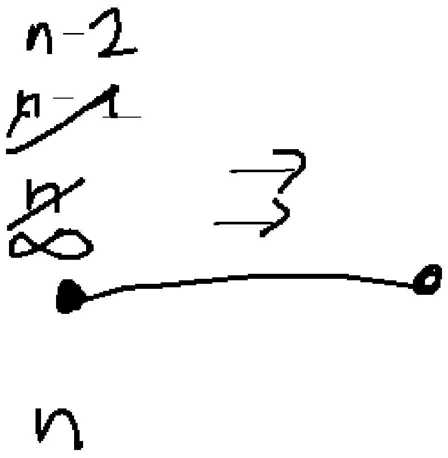
Bellman-Ford

Algorithm 2.13 Bellman-Ford BFS

Each node u stores an integer d_u which corresponds to the distance from u to the root. Initially $d_{\text{root}} = 0$, and $d_u = \infty$ for every other node u.
The root starts the algorithm by sending "1" to all neighbors.
if a node u receives a message "y" with $y < d_u$ from a neighbor v **then**
    node u sets $d_u := y$
    node u sends "$y + 1$" to all neighbors (except v)
end if

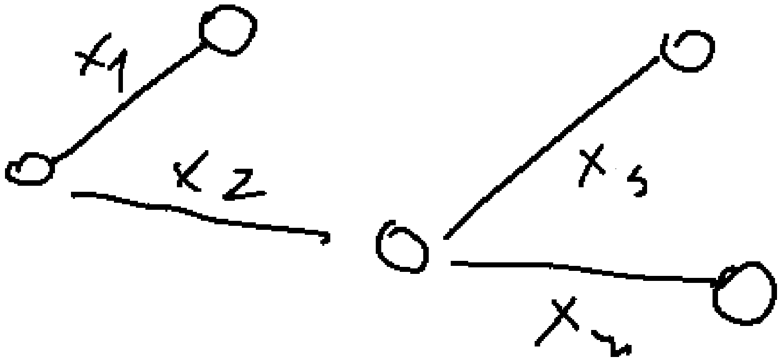
For a graph with diameter D, the number of edges m, and n nodes:
Time complexity O(D)
Message complexity O(mn)





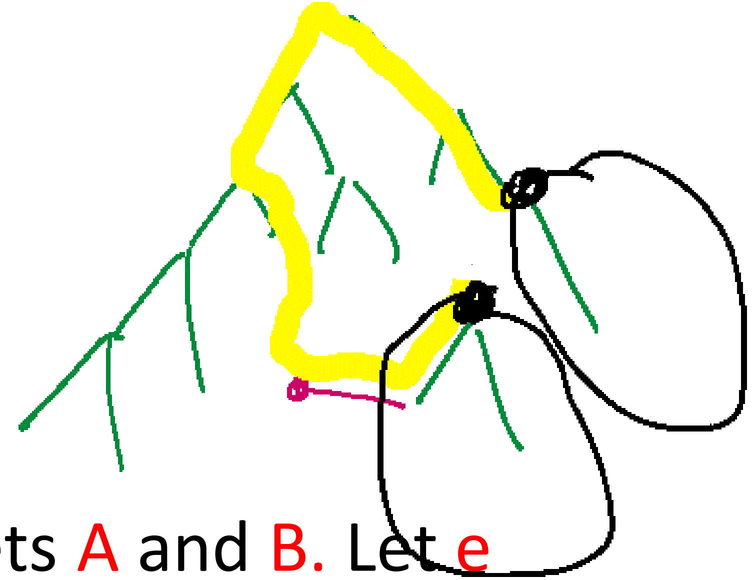
Minimum Spanning Tree

- A tree containing each node of the graph
- Each edges has a weight
- The sum of weigths in the tree should be minimum possible



$$\sum x_i = \text{minimalna}$$

MST- key observation

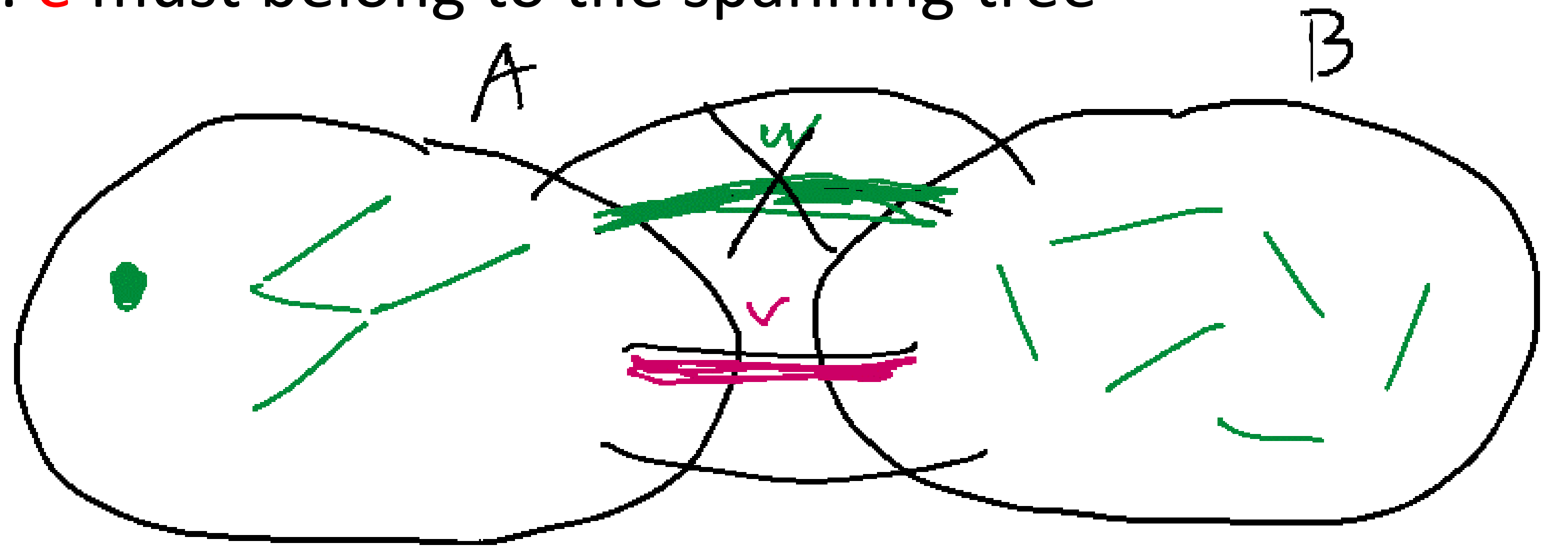


Theorem

Assume: the weights are different

Partition the nodes of the graph into disjoint subsets A and B. Let e be the lightest edge connecting A and B

then: e must belong to the spanning tree



$v < w$

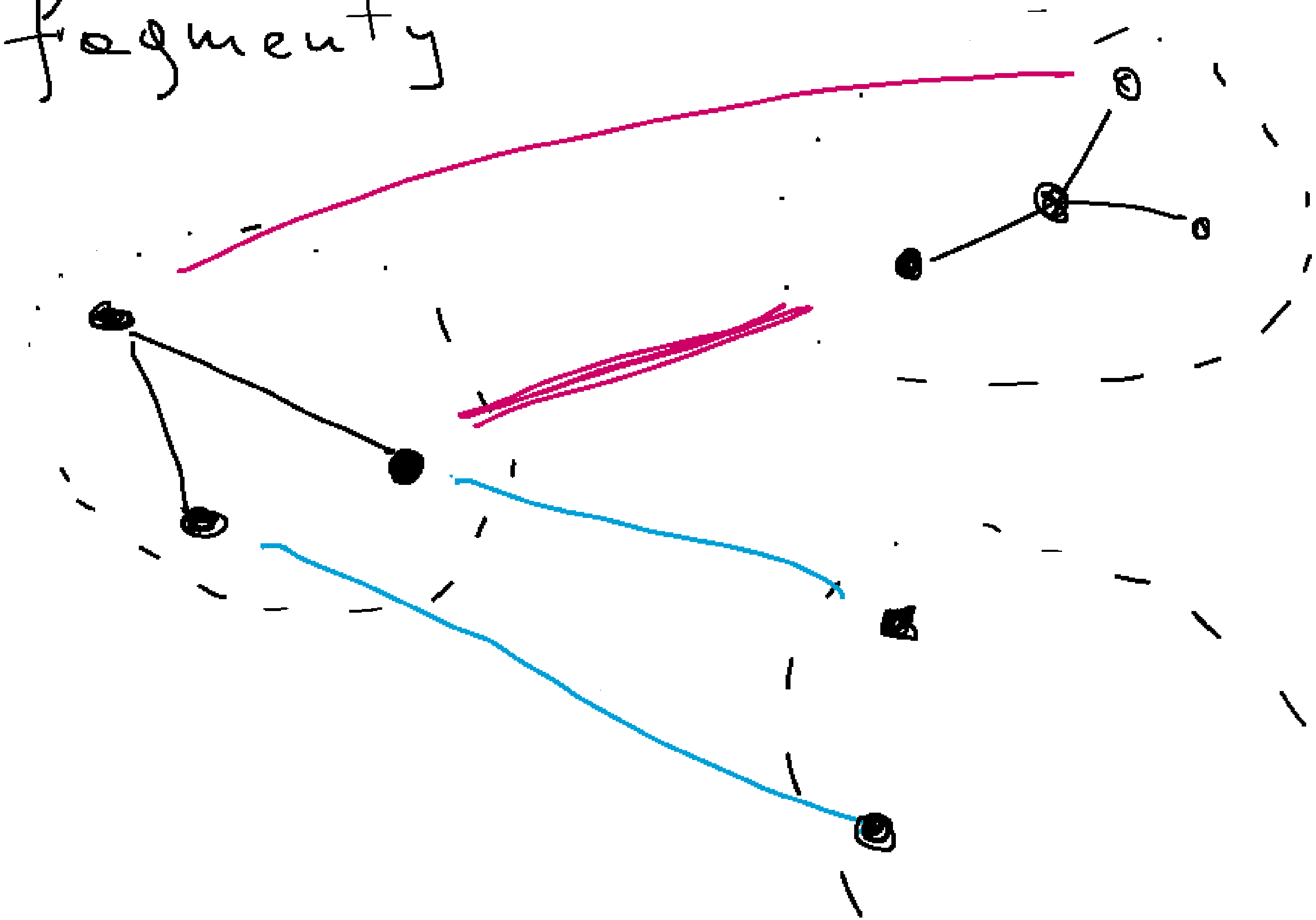
$\Rightarrow$ sprzeczność

MST

Algorithm 2.18 GHS (Gallager–Humblet–Spira)

Initially each node is the root of its own fragment. We proceed in phases:
repeat
  All nodes learn the fragment IDs of their neighbors.
  The root of each fragment uses flooding/echo in its fragment to determine the blue edge $b = (u, v)$ of the fragment.
  The root sends a message to node u; while forwarding the message on the path from the root to node u all parent-child relations are inverted {such that u is the new temporary root of the fragment}
  node u sends a merge request over the blue edge $b = (u, v)$.
  if node v also sent a merge request over the same blue edge $b = (v, u)$ **then**
    either u or v (whichever has the smaller ID) is the new fragment root
    the blue edge b is directed accordingly
  else
    node v is the new parent of node u
  end if
  the newly elected root node informs all nodes in its fragment (again using flooding/echo) about its identity
until all nodes are in the same fragment (i.e., there is no outgoing edge)

fragmenty

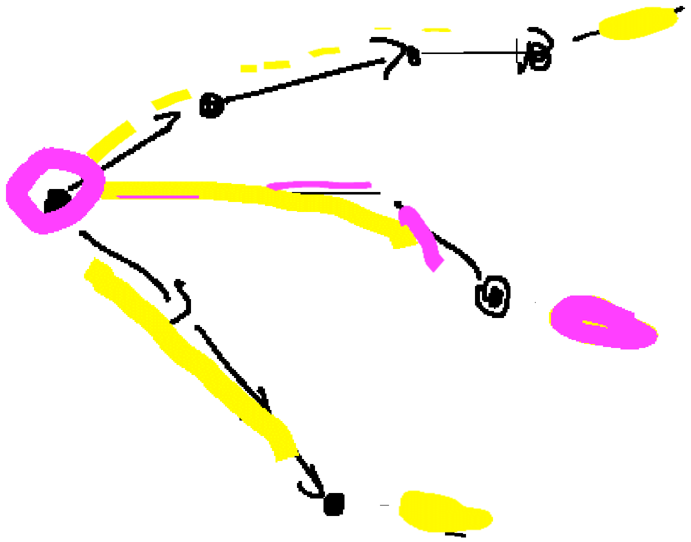


MST

3: All nodes learn the fragment IDs of their neighbors.

4: The root of each fragment uses flooding/echo in its fragment to determine the blue edge $b = (u, v)$ of the fragment.

5: The root sends a message to node u; while forwarding the message on the path from the root to node u all parent-child relations are inverted {such that u is the new temporary root of the fragment}

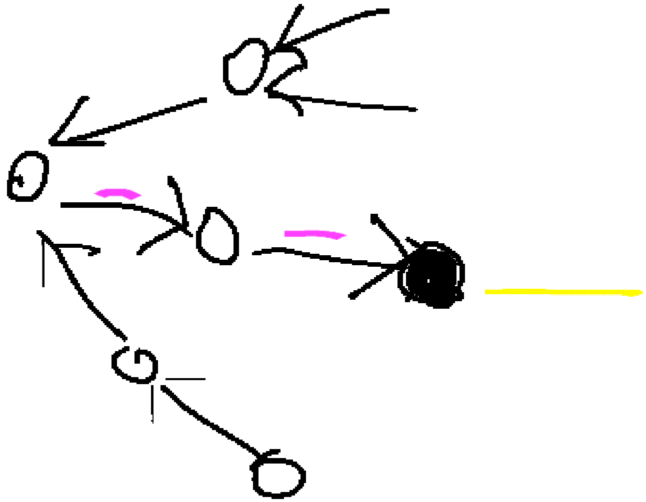


MST

3: All nodes learn the fragment IDs of their neighbors.

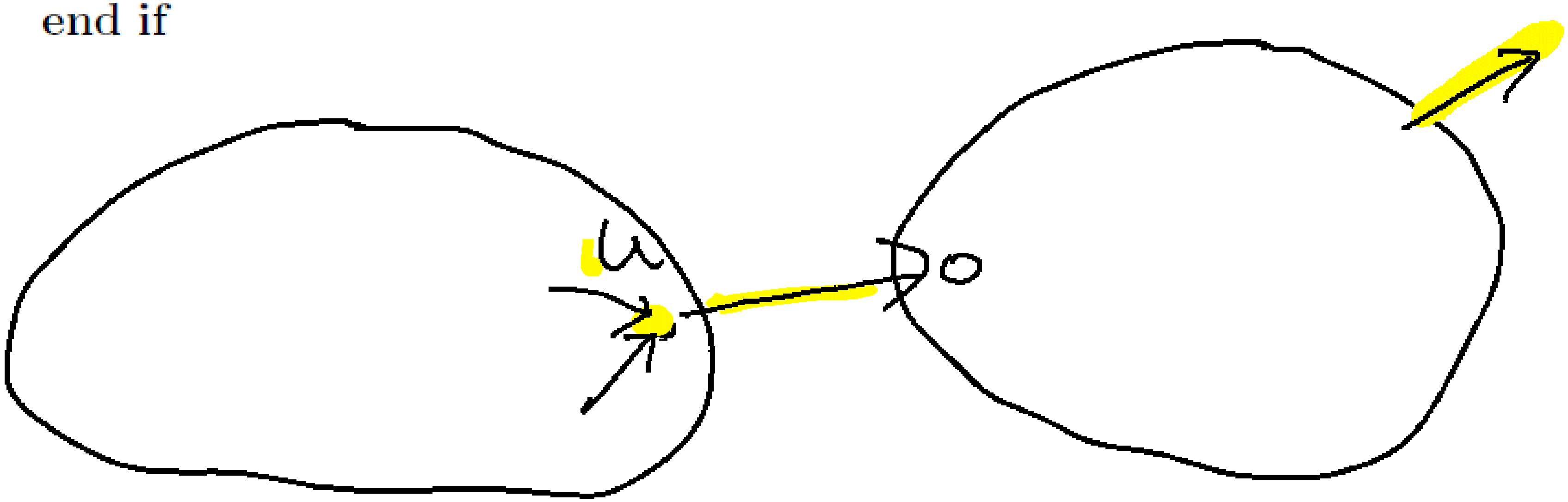
4: The root of each fragment uses flooding/echo in its fragment to determine the blue edge $b = (u, v)$ of the fragment.

5: The root sends a message to node u; while forwarding the message on the path from the root to node u all parent-child relations are inverted {such that u is the new temporary root of the fragment}



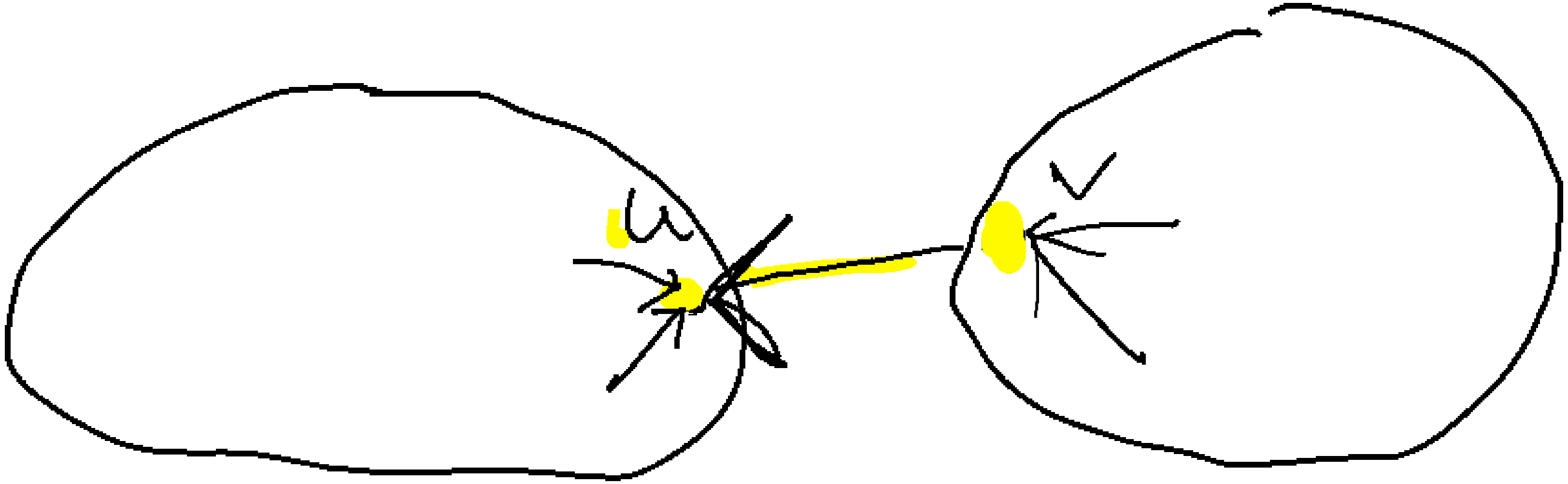
MST

node u sends a merge request over the blue edge $b = (u, v)$.
if node v also sent a merge request over the same blue edge $b = (v, u)$ **then**
  either u or v (whichever has the smaller ID) is the new fragment root
  the blue edge b is directed accordingly
else
  node v is the new parent of node u
end if



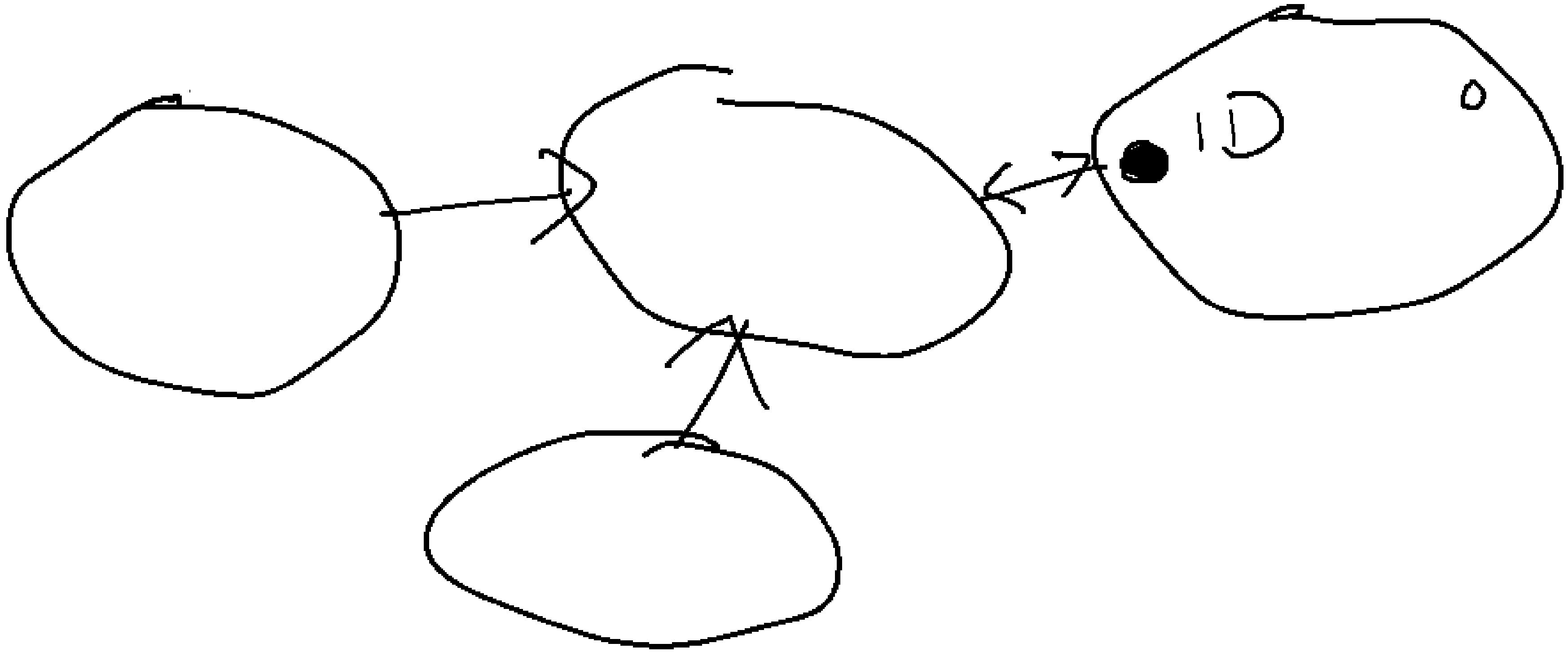
MST

node u sends a merge request over the blue edge $b = (u, v)$.
if node v also sent a merge request over the same blue edge $b = (v, u)$ **then**
  either u or v (whichever has the smaller ID) is the new fragment root
  the blue edge b is directed accordingly
else
  node v is the new parent of node u
end if



MST

13: the newly elected root node informs all nodes in its fragment (again using flooding/echo) about its identity



MST

Algorithm 2.18 GHS (Gallager–Humblet–Spira)

Initially each node is the root of its own fragment. We proceed in phases:
repeat
    All nodes learn the fragment IDs of their neighbors.
    The root of each fragment uses flooding/echo in its fragment to determine the blue edge $b = (u, v)$ of the fragment.
    The root sends a message to node u; while forwarding the message on the path from the root to node u all parent-child relations are inverted {such that u is the new temporary root of the fragment}
    node u sends a merge request over the blue edge $b = (u, v)$.
    if node v also sent a merge request over the same blue edge $b = (v, u)$ **then**
        either u or v (whichever has the smaller ID) is the new fragment root
        the blue edge b is directed accordingly
    else
        node v is the new parent of node u
    end if
    the newly elected root node informs all nodes in its fragment (again using flooding/echo) about its identity
until all nodes are in the same fragment (i.e., there is no outgoing edge)

złożoność czasowa

1 runda: time = O(średnica fragmentu)

$\leq D$

rund ?:

k fragmentów $\xrightarrow{\text{runda}}$ $\leq \frac{k}{2}$ fragmentów

rund $\in \log(n)$