

# Local MST computation with short advice

Pierre Fraigniaud<sup>1</sup>   Amos Korman<sup>2</sup>   Emmanuelle Lebar<sup>1</sup>

<sup>1</sup>CNRS and Univ. Paris

<sup>2</sup>Technion, Israel

ACM Symposium on Parallel Algorithms and Architectures  
(SPAA), 2007

# Outline

- 1 Introduction
  - MST Introduction
  - The Advising Scheme Concept
- 2 Advising Schemes for MST
  - Upper Bound on the Average Size
  - An  $(O(1), O(\log N))$ -Advising Scheme for MST
  - Lower Bound on the Average Size

# Outline

- 1 Introduction
  - MST Introduction
  - The Advising Scheme Concept
- 2 Advising Schemes for MST
  - Upper Bound on the Average Size
  - An  $(O(1), O(\log N))$ -Advising Scheme for MST
  - Lower Bound on the Average Size

# MST Introduction

- Borůvka's algorithm (1926) -  $O(m \log n)$
- GHS Gallager Humblet Spira (1983) -  $O(n \log n)$
- Kutten, S. and Peleg, D. (1995) - Upper bound  $O(\sqrt{n} \log^* n + D)$
- Peleg, D. Rubinfeld, R (1999) - Lower Bound  $\Omega(\sqrt{n}/B)$
- Lotker, Z. Patt-Shamir, B. and Peleg D (2001) - Lower Bound for Constant Diameter Graphs  $\Omega(n^{1/4}), \Omega(n^{1/3})$  for  $D = 3, 4$  respectively

# MST Introduction

- Borůvka's algorithm (1926) -  $O(m \log n)$
- GHS Gallager Humblet Spira (1983) -  $O(n \log n)$
- Kutten, S. and Peleg, D. (1995) - Upper bound  $O(\sqrt{n} \log^* n + D)$
- Peleg, D. Rubinfeld, R (1999) - Lower Bound  $\Omega(\sqrt{n}/B)$
- Lotker, Z. Patt-Shamir, B. and Peleg D (2001) - Lower Bound for Constant Diameter Graphs  $\Omega(n^{1/4}), \Omega(n^{1/3})$  for  $D = 3, 4$  respectively

# MST Introduction

- Borůvka's algorithm (1926) -  $O(m \log n)$
- GHS Gallager Humblet Spira (1983) -  $O(n \log n)$
- Kutten, S. and Peleg, D. (1995) - Upper bound  $O(\sqrt{n} \log^* n + D)$
- Peleg, D. Rubinfeld, R (1999) - Lower Bound  $\Omega(\sqrt{n}/B)$
- Lotker, Z. Patt-Shamir, B. and Peleg D (2001) - Lower Bound for Constant Diameter Graphs  $\Omega(n^{1/4}), \Omega(n^{1/3})$  for  $D = 3, 4$  respectively

# MST Introduction

- Borůvka's algorithm (1926) -  $O(m \log n)$
- GHS Gallager Humblet Spira (1983) -  $O(n \log n)$
- Kutten, S. and Peleg, D. (1995) - Upper bound  $O(\sqrt{n} \log^* n + D)$
- Peleg, D. Rubinfeld, R (1999) - Lower Bound  $\Omega(\sqrt{n}/B)$
- Lotker, Z. Patt-Shamir, B. and Peleg D (2001) - Lower Bound for Constant Diameter Graphs  $\Omega(n^{1/4}), \Omega(n^{1/3})$  for  $D = 3, 4$  respectively

# MST Introduction

- Borůvka's algorithm (1926) -  $O(m \log n)$
- GHS Gallager Humblet Spira (1983) -  $O(n \log n)$
- Kutten, S. and Peleg, D. (1995) - Upper bound  $O(\sqrt{n} \log^* n + D)$
- Peleg, D. Rubinfeld, R (1999) - Lower Bound  $\Omega(\sqrt{n}/B)$
- Lotker, Z. Patt-Shamir, B. and Peleg D (2001) - Lower Bound for Constant Diameter Graphs  $\Omega(n^{1/4}), \Omega(n^{1/3})$  for  $D = 3, 4$  respectively

# MST Introduction

- Borůvka's algorithm (1926) -  $O(m \log n)$
- GHS Gallager Humblet Spira (1983) -  $O(n \log n)$
- Kutten, S. and Peleg, D. (1995) - Upper bound  $O(\sqrt{n} \log^* n + D)$
- Peleg, D. Rubinfeld, R (1999) - Lower Bound  $\Omega(\sqrt{n}/B)$
- Lotker, Z. Patt-Shamir, B. and Peleg D (2001) - Lower Bound for Constant Diameter Graphs  $\Omega(n^{1/4}), \Omega(n^{1/3})$  for  $D = 3, 4$  respectively

# Outline

- 1 Introduction
  - MST Introduction
  - The Advising Scheme Concept
- 2 Advising Schemes for MST
  - Upper Bound on the Average Size
  - An  $(O(1), O(\log N))$ -Advising Scheme for MST
  - Lower Bound on the Average Size

# The Advising Scheme Concept

## Definition

- An  $(m, t)$ -advising scheme for a distributed problem  $P$  is a pair  $(O, A)$  where  $O$  is an oracle, and  $A$  is an algorithm solving  $P$  using advices of  $O$
- Maximal size of advice for a node is  $m$
- $P$  can be solved distributively in at most  $t$  rounds

# The Advising Scheme Concept

## Definition

- An  $(m, t)$ -advising scheme for a distributed problem  $P$  is a pair  $(O, A)$  where  $O$  is an oracle, and  $A$  is an algorithm solving  $P$  using advices of  $O$
- Maximal size of advice for a node is  $m$
- $P$  can be solved distributively in at most  $t$  rounds

# The Advising Scheme Concept

## Definition

- An  $(m, t)$ -advising scheme for a distributed problem  $P$  is a pair  $(O, A)$  where  $O$  is an oracle, and  $A$  is an algorithm solving  $P$  using advices of  $O$
- Maximal size of advice for a node is  $m$
- $P$  can be solved distributively in at most  $t$  rounds

# Advising Scheme Examples

## Examples

- $(\lceil \log n \rceil, 0)$  - Advising Scheme for MST  
advice: port number of MST parent
- $(0, \sqrt{n} + D)$  - Advising Scheme for MST in **CONGEST** model  
no advice, execute Kutten and Peleg Algorithm
- $(0, D + 1)$  - Advising Scheme for MST in **LOCAL** model  
no advice, flooding algorithm

# Advising Scheme Examples

## Examples

- $(\lceil \log n \rceil, 0)$  - Advising Scheme for MST  
advice: port number of MST parent
- $(0, \sqrt{n} + D)$  - Advising Scheme for MST in **CONGEST** model  
no advice, execute Kutten and Peleg Algorithm
- $(0, D + 1)$  - Advising Scheme for MST in **LOCAL** model  
no advice, flooding algorithm

# Advising Scheme Examples

## Examples

- $(\lceil \log n \rceil, 0)$  - Advising Scheme for MST  
advice: port number of MST parent
- $(0, \sqrt{n} + D)$  - Advising Scheme for MST in **CONGEST** model  
no advice, execute Kutten and Peleg Algorithm
- $(0, D + 1)$  - Advising Scheme for MST in **LOCAL** model  
no advice, flooding algorithm

# Advising Scheme Examples

## Examples

- $(\lceil \log n \rceil, 0)$  - Advising Scheme for MST  
advice: port number of MST parent
- $(0, \sqrt{n} + D)$  - Advising Scheme for MST in **CONGEST** model  
no advice, execute Kutten and Peleg Algorithm
- $(0, D + 1)$  - Advising Scheme for MST in **LOCAL** model  
no advice, flooding algorithm

# Advising Scheme Examples

## Examples

- $(\lceil \log n \rceil, 0)$  - Advising Scheme for MST  
advice: port number of MST parent
- $(0, \sqrt{n} + D)$  - Advising Scheme for MST in **CONGEST** model  
no advice, execute Kutten and Peleg Algorithm
- $(0, D + 1)$  - Advising Scheme for MST in **LOCAL** model  
no advice, flooding algorithm

# Advising Scheme Examples

## Examples

- $(\lceil \log n \rceil, 0)$  - Advising Scheme for MST  
advice: port number of MST parent
- $(0, \sqrt{n} + D)$  - Advising Scheme for MST in **CONGEST** model  
no advice, execute Kutten and Peleg Algorithm
- $(0, D + 1)$  - Advising Scheme for MST in **LOCAL** model  
no advice, flooding algorithm

# Outline

- 1 Introduction
  - MST Introduction
  - The Advising Scheme Concept
- 2 Advising Schemes for MST
  - Upper Bound on the Average Size
  - An  $(O(1), O(\log N))$ -Advising Scheme for MST
  - Lower Bound on the Average Size

## Upper Bound on the Average Size

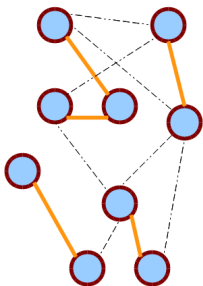
- Algorithm for  $(O(\log^2 n), 1)$ -advising scheme with CONSTANT average advice size
- **CONGEST** model

# Preliminary: Boruvka's MST algorithm

- Converging fragments
- $\lceil \log n \rceil$  phases
- At each phase  $i$  : participate fragments  $F$  satisfying  $|F| < 2^i$
- Every fragment  $F$  that is active at phase  $i$  selects an incident edge  $e$  leading out of  $F$ , and of minimum weight (ties broken using port number, then arbitrarily).

## Preliminary: Boruvka's MST algorithm

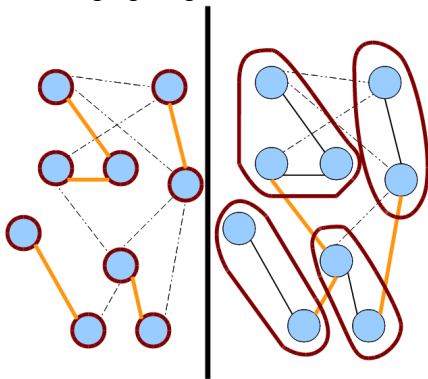
- Converging fragments



- $\lceil \log n \rceil$  phases
- At each phase  $i$ : participate fragments  $F$  satisfying  $|F| < 2^i$
- Every fragment  $F$  that is active at phase  $i$  selects an incident

# Preliminary: Boruvka's MST algorithm

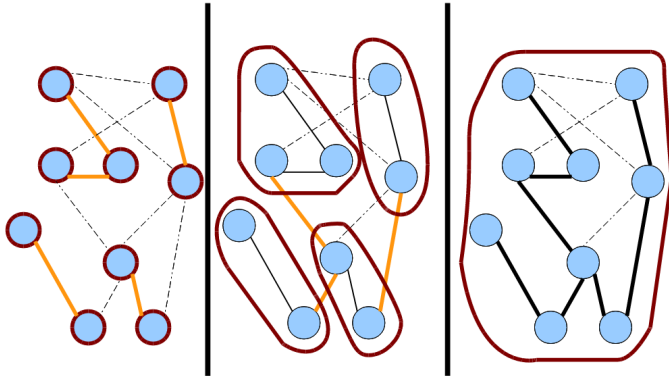
- Converging fragments



- $\lceil \log n \rceil$  phases
- At each phase  $i$ : participate fragments  $F$  satisfying  $|F| < 2^i$
- Every fragment  $F$  that is active at phase  $i$  selects an incident

# Preliminary: Boruvka's MST algorithm

- Converging fragments



- $\lceil \log n \rceil$  phases
- At each phase  $i$  : participate fragments  $F$  satisfying  $|F| < 2^i$
- Every fragment  $F$  that is active at phase  $i$  selects an incident

# Preliminary: Boruvka's MST algorithm

- Converging fragments
- $\lceil \log n \rceil$  phases
- At each phase  $i$  : participate fragments  $F$  satisfying  $|F| < 2^i$
- Every fragment  $F$  that is active at phase  $i$  selects an incident edge  $e$  leading out of  $F$ , and of minimum weight (ties broken using port number, then arbitrarily).

# Preliminary: Boruvka's MST algorithm

- Converging fragments
- $\lceil \log n \rceil$  phases
- At each phase  $i$  : participate fragments  $F$  satisfying  $|F| < 2^i$
- Every fragment  $F$  that is active at phase  $i$  selects an incident edge  $e$  leading out of  $F$ , and of minimum weight (ties broken using port number, then arbitrarily).

# Preliminary: Boruvka's MST algorithm

- Converging fragments
- $\lceil \log n \rceil$  phases
- At each phase  $i$  : participate fragments  $F$  satisfying  $|F| < 2^i$
- Every fragment  $F$  that is active at phase  $i$  selects an incident edge  $e$  leading out of  $F$ , and of minimum weight (ties broken using port number, then arbitrarily).

# How many fragments participating in phase $i$ ?

## Corollary

*After phase  $i$ , the size of every fragments is at least  $2^i$ . Thus a fragment  $F$  that is active at phase  $i$  satisfies  $2^{i-1} \leq |F| \leq 2^i$ . Hence there are at most  $n/2^{i-1}$  active fragments at phase  $i$ .*

# How many fragments participating in phase $i$ ?

## Corollary

*After phase  $i$ , the size of every fragments is at least  $2^i$ . Thus a fragment  $F$  that is active at phase  $i$  satisfies  $2^{i-1} \leq |F| \leq 2^i$ . Hence there are at most  $n/2^{i-1}$  active fragments at phase  $i$ .*

# Boruvka's Algorithm Properties

## Definition

$index_u(e) = (x_u(e), y_u(e))$  where:

- $x_u(e)$  is the rank of the weight  $w(e)$  of  $e$  among all the weights of the edges incident to  $u$ .
- $y_u(e)$  is the rank of the port number of edge  $e$  among all the edges of weight  $w(e)$  incident to  $u$ .

# How many bits are needed to determine the selected edge?

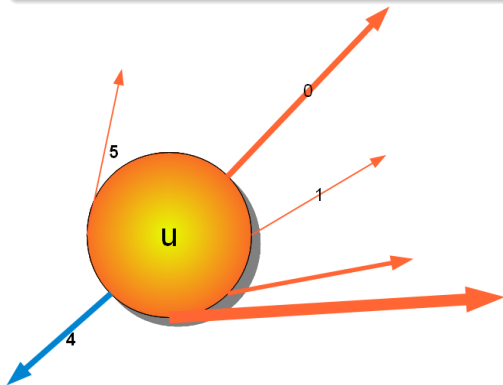
## Corollary

*If  $e$  has been selected by fragment  $F$  and  $u$  is the choosing node of  $e$ , then  $\text{index}_u(e) = (x_u(e), y_u(e))$  satisfies  $x_u(e) + y_u(e) \leq |F|$*

# How many bits are needed to determine the selected edge?

## Corollary

If  $e$  has been selected by fragment  $F$  and  $u$  is the choosing node of  $e$ , then  $\text{index}_u(e) = (x_u(e), y_u(e))$  satisfies  $x_u(e) + y_u(e) \leq |F|$



$(O(\log^2 n), 1)$  - advising scheme, with average size  $O(1)$

## ORACLE

- Run Boruvka's algorithm returning MST  $T$ , with an arbitrary root  $r$
- Remember the following advice for each node  $u$  and phase  $i \in [1, \log n]$ :
  - $index_u(e)$  where  $e$  is the selected edge at phase  $i$  corresponding to  $u$
  - a Boolean  $b$  stating whether  $e$  is up i.e. leading to  $r$  on tree  $T$ .

## ALGORITHM

- if  $b$ : return edge defined by  $index_u(e)$
- else: send message along the edge (upon accepting message: return the edge)

$(O(\log^2 n), 1)$  - advising scheme, with average size  $O(1)$

## ORACLE

- Run Boruvka's algorithm returning MST  $T$ , with an arbitrary root  $r$
- Remember the following advice for each node  $u$  and phase  $i \in [1, \log n]$ :
  - $index_u(e)$  where  $e$  is the selected edge at phase  $i$  corresponding to  $u$
  - a Boolean  $b$  stating whether  $e$  is up i.e. leading to  $r$  on tree  $T$ .

## ALGORITHM

- if  $b$ : return edge defined by  $index_u(e)$
- else: send message along the edge (upon accepting message: return the edge)

$(O(\log^2 n), 1)$  - advising scheme, with average size  $O(1)$

## ORACLE

- Run Boruvka's algorithm returning MST  $T$ , with an arbitrary root  $r$
- Remember the following advice for each node  $u$  and phase  $i \in [1, \log n]$ :
  - $index_u(e)$  where  $e$  is the selected edge at phase  $i$  corresponding to  $u$
  - a Boolean  $b$  stating whether  $e$  is up i.e. leading to  $r$  on tree  $T$ .

## ALGORITHM

- if  $b$ : return edge defined by  $index_u(e)$
- else: send message along the edge (upon accepting message: return the edge)

$(O(\log^2 n), 1)$  - advising scheme, with average size  $O(1)$

## ORACLE

- Run Boruvka's algorithm returning MST  $T$ , with an arbitrary root  $r$
- Remember the following advice for each node  $u$  and phase  $i \in [1, \log n]$ :
  - $index_u(e)$  where  $e$  is the selected edge at phase  $i$  corresponding to  $u$
  - a Boolean  $b$  stating whether  $e$  is up i.e. leading to  $r$  on tree  $T$ .

## ALGORITHM

- if  $b$ : return edge defined by  $index_u(e)$
- else: send message along the edge (upon accepting message: return the edge)

$(O(\log^2 n), 1)$  - advising scheme, with average size  $O(1)$

## ORACLE

- Run Boruvka's algorithm returning MST  $T$ , with an arbitrary root  $r$
- Remember the following advice for each node  $u$  and phase  $i \in [1, \log n]$ :
  - $index_u(e)$  where  $e$  is the selected edge at phase  $i$  corresponding to  $u$
  - a Boolean  $b$  stating whether  $e$  is up i.e. leading to  $r$  on tree  $T$ .

## ALGORITHM

- if  $b$ : return edge defined by  $index_u(e)$
- else: send message along the edge (upon accepting message: return the edge)

# Analysis

- This advice is of size  $i + 1$  bits at most (limit on fragment size).
- At phase  $i$ : #Fragments is  $n/2^{i-1}$ . Each has 1 choosing node.
- Scheme uses at most  $2 \sum_{i=1}^{\log n} (i + 1) n/2^{i-1} = O(n)$  bits

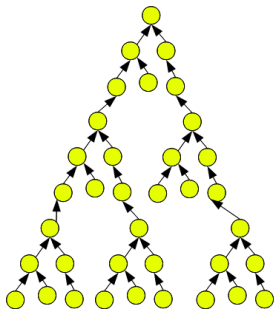
# Outline

- 1 Introduction
  - MST Introduction
  - The Advising Scheme Concept
- 2 Advising Schemes for MST
  - Upper Bound on the Average Size
  - An  $(O(1), O(\log N))$ -Advising Scheme for MST
  - Lower Bound on the Average Size

# An $(O(1), O(\log N))$ -Advising Scheme for MST

Run Boruvka's algorithm to find MST  $T$  with root 'r'

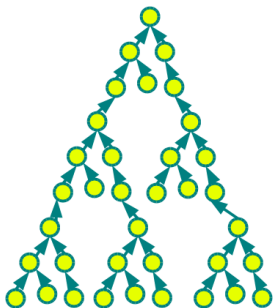
- each fragment  $F$  induces a subtree  $T_F$  of the  $T$ . Such fragment is marked as a single node  $x_F$
- The fragments are connected by edges of  $T$ , at the end of the phase and create a tree of fragments  $T_i$ .



# An $(O(1), O(\log N))$ -Advising Scheme for MST

and again for  $\lceil \log \log n \rceil + 1$  phases

- each fragment  $F$  induces a subtree  $T_F$  of the  $T$ . Such fragment is marked as a single node  $x_F$
- The fragments are connected by edges of  $T$ , at the end of the phase and create a tree of fragments  $T_i$ .

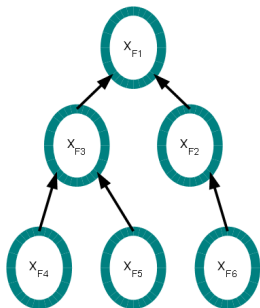




# An $(O(1), O(\log N))$ -Advising Scheme for MST

and again for  $\lceil \log \log n \rceil + 1$  phases

- each fragment  $F$  induces a subtree  $T_F$  of the  $T$ . Such fragment is marked as a single node  $x_F$
- The fragments are connected by edges of  $T$ , at the end of the phase and create a tree of fragments  $T_i$ .



# An $(O(1), O(\log N))$ -Advising Scheme for MST

For any phase  $i \in [1, \log \log n]$  three items of advice are encoded in each active fragment  $F$ :

- 1 choosing node of  $F$
- 2 orientation (up or down)
- 3 0 level of  $x_F$  on  $T_i$  is even, 1 otherwise

# Encoding

- $j$  : index of choosing node of  $F$  in a BFS from  $r_F$ .  $j < 2^i$
- Advice  $A(F) = b_{up} | b_{level} | bin(j) \in \{0, 1\}^{i+2}$
- In BFS order fill up to 11 bits of advice in each node of  $F$  (node may already have all or part of the bits taken in prev. phase).

# An $(O(1), O(\log N))$ -Advising Scheme for MST

At phase  $\lceil \log \log n \rceil + 1$  the scheme encodes inside each fragment  $F$  (even if passive) the index of the edge  $e$  of  $T$  leading from  $x_F$  to its parent in  $T_i$ .

- index is at most  $n-1$  ( $\lceil \log n \rceil$  bits)
- $|F| \geq 2^{\lceil \log \log n \rceil} \geq \lceil \log n \rceil$
- encode in BFS order 1 bit per node in  $F$

## Decoding Process

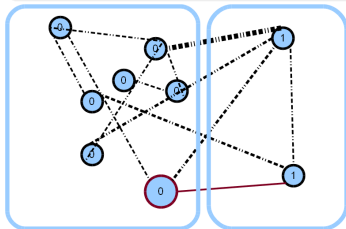
- Input: The distributed network  $G$ , with each node  $v$  provided with  $O(G, v) = \text{advice}(v)$
- Output: Each node  $v$  computes the port number of the edge leading to its parent in the MST  $T$ .

## MST Algorithm Outline

- Repeat Boruvka  $\lceil \log \log n \rceil$  phases replacing search for minimum-edge with the extraction of advices.
- convergecast to  $r_F$  and broadcast the advice in  $T_F$ . (this determines the choosing node/edge, which also determines next phase  $r_F$ )
- the advice at  $i := \lceil \log \log n \rceil + 1$  determines the outgoing edge from  $r_F$

## MST Algorithm Outline

- Repeat Boruvka  $\lceil \log \log n \rceil$  phases replacing search for minimum-edge with the extraction of advices.
- convergeicast to  $r_F$  and broadcast the advice in  $T_F$ . (this determines the choosing node/edge, which also determines next phase  $r_F$ )
- the advice at  $i := \lceil \log \log n \rceil + 1$  determines the outgoing edge from  $r_F$



## MST Algorithm Outline

- Repeat Boruvka  $\lceil \log \log n \rceil$  phases replacing search for minimum-edge with the extraction of advices.
- convergecast to  $r_F$  and broadcast the advice in  $T_F$ . (this determines the choosing node/edge, which also determines next phase  $r_F$ )
- the advice at  $i := \lceil \log \log n \rceil + 1$  determines the outgoing edge from  $r_F$

# Analysis

## Theorem

*Algorithm terminates after  $O(\lceil \log n \rceil)$  rounds.*

## Proof.

- Decoding of Active-Fragment is at most  $|F| \leq 2^{i+1}$
- Decoding of last phase advice is  $O(\lceil \log n \rceil)$
- $\lceil \log n \rceil + \sum_{i=1}^{\lceil \log \log n \rceil} 2^{i+1} \leq 9 \lceil \log n \rceil$



# Analysis

## Theorem

*Algorithm terminates after  $O(\lceil \log n \rceil)$  rounds.*

## Proof.

- Decoding of Active-Fragment is at most  $|F| \leq 2^{i+1}$
- Decoding of last phase advice is  $O(\lceil \log n \rceil)$
- $\lceil \log n \rceil + \sum_{i=1}^{\lceil \log \log n \rceil} 2^{i+1} \leq 9 \lceil \log n \rceil$



# Analysis

## Theorem

*Algorithm terminates after  $O(\lceil \log n \rceil)$  rounds.*

## Proof.

- Decoding of Active-Fragment is at most  $|F| \leq 2^{i+1}$
- Decoding of last phase advice is  $O(\lceil \log n \rceil)$
- $\lceil \log n \rceil + \sum_{i=1}^{\lceil \log \log n \rceil} 2^{i+1} \leq 9 \lceil \log n \rceil$



# Analysis

## Theorem

*Algorithm terminates after  $O(\lceil \log n \rceil)$  rounds.*

## Proof.

- Decoding of Active-Fragment is at most  $|F| \leq 2^{i+1}$
- Decoding of last phase advice is  $O(\lceil \log n \rceil)$
- $\lceil \log n \rceil + \sum_{i=1}^{\lceil \log \log n \rceil} 2^{i+1} \leq 9 \lceil \log n \rceil$



# Analysis

## Theorem

*Algorithm terminates after  $O(\lceil \log n \rceil)$  rounds.*

## Proof.

- Decoding of Active-Fragment is at most  $|F| \leq 2^{i+1}$
- Decoding of last phase advice is  $O(\lceil \log n \rceil)$
- $\lceil \log n \rceil + \sum_{i=1}^{\lceil \log \log n \rceil} 2^{i+1} \leq 9 \lceil \log n \rceil$



# Outline

- 1 Introduction
  - MST Introduction
  - The Advising Scheme Concept
- 2 Advising Schemes for MST
  - Upper Bound on the Average Size
  - An  $(O(1), O(\log N))$ -Advising Scheme for MST
  - Lower Bound on the Average Size

# Lower Bound on the Average Size

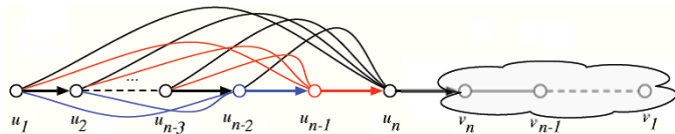
- $(\lceil \log n \rceil, 0)$ - advising scheme is optimal
- **LOCAL** model

## Theorem

*For any  $n$ -node graph  $G$ , and any  $m \geq 0$ , any  $(m, 0)$ -advising scheme for computing an MST of  $G$  gives advices of average size  $\Omega(\log n)$ . This result holds even if all edge-weights are pairwise distinct and even if nodes are given distinct but arbitrary IDs.*

# Proof

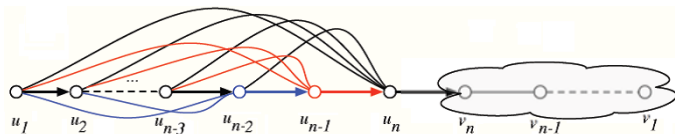
## Building a weighted graph




- $w(\{u_1, v_1\}) = 0$
- Disjoint intervals:  $I_1..I_n$
- $w(u_i, u_{i-1})$  and  $w(v_i, v_{i-1})$  can be any integer in  $I_i \in [a_i, b_i]$ .
- $w(u_i, u_j)$  and  $w(v_i, v_j)$  ( $j \in [i+2, n]$ ) can be any integer in  $I_i$ .

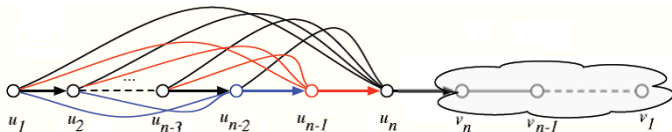
# Proof

## Building a weighted graph



- $w(\{u_1, v_1\}) = 0$
- Disjoint intervals:  $I_1 \dots I_n$   

- $w(u_i, u_{i-1})$  and  $w(v_i, v_{i-1})$  can be any integer in  $I_i \in [a_i, b_i]$ .
- $w(u_i, u_j)$  and  $w(v_i, v_j)$  ( $j \in [i+2, n]$ ) can be any integer in  $I_i$ .

## Proof

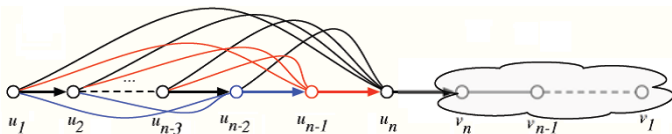


## Corollary

*MST is the path  $(u_n, \dots, u_1, v_1, \dots, v_n)$*

- *$u_n, v_n$  has to belong to any spanning tree of  $G_n$*
- *Every node has its lightest edge in MST :  $u_i, u_{i-1}$  is the lightest edge of  $u_i$*
- *$u_n, u_{n-1}$  is the lightest node to maintain connectivity.*

## Proof



## Proof.

- Let  $I_i \in [a_i, b_i]$  contain  $i$  values.
- Let  $i$  edges be incident to  $u_i$  with  $i$  different weights
- $\log i$  bits needed to determine a port:  $\Omega(\log n)$  bits on average.



# Summary

- Advising Scheme is a method to trade run-time for memory.
- While computing MST in time 0 requires a  $\Omega(\log n)$  advice, only  $O(1)$  average advice size is required to do so in time 1
- an  $(O(1), O(\log N))$ -Advising Scheme was shown
  
- Outlook
  - $(O(1), O(1))$ -Advising Scheme is not proved or disproved.