Definition of a Broadcasts and Accumulation

**Definition of Broadcast:**

Given are $G = (V, E)$ and $v \in V$.
- $v$ has information $I(v)$
- no node from $V \setminus \{v\}$ knows $I(v)$.
- Each node of $V \setminus \{v\}$ has to receive information $I(v)$.

**Definition of Accumulation:**

Given are $G = (V, E)$ and $v \in V$.
- Each node of $w \in V$ has information $I(w)$
- no node from $V \setminus \{w\}$ knows $I(w)$.
- Node $v$ should receive the information $\bigcup_{w \in V} I(w)$.
Definition of a Broadcasts and Accumulation

**Definition of Broadcast:**

Given are \( G = (V, E) \) and \( v \in V \).

- \( v \) has information \( I(v) \) and
- no node from \( V \setminus \{v\} \) knows \( I(v) \).
- Each node of \( V \setminus \{v\} \) has to receive information \( I(v) \).

**Definition of Accumulation:**

Given are \( G = (V, E) \) and \( v \in V \).

- Each node of \( w \in V \) has information \( I(w) \)
- no node from \( V \setminus \{w\} \) knows \( I(w) \).
- Node \( v \) should receive the information \( \bigcup_{w \in V} I(w) \).
Definition of a Broadcasts and Accumulation

**Definition of Broadcast:**

Given are $G = (V, E)$ and $v \in V$.

- $v$ has information $I(v)$ and
- no node from $V \setminus \{v\}$ knows $I(v)$.
- Each node of $V \setminus \{v\}$ has to receive information $I(v)$.

**Definition of Accumulation:**

Given are $G = (V, E)$ and $v \in V$.

- Each node of $w \in V$ has information $I(w)$
- no node from $V \setminus \{w\}$ knows $I(w)$.
- Node $v$ should receive the information $\bigcup_{w \in V} I(w)$. 
Definition of a Broadcasts and Accumulation

**Definition of Broadcast:**
Given are $G = (V, E)$ and $v \in V$.
- $v$ has information $I(v)$ and
- no node from $V \setminus \{v\}$ knows $I(v)$.
- Each node of $V \setminus \{v\}$ has to receive information $I(v)$.

**Definition of Accumulation:**
Given are $G = (V, E)$ and $v \in V$.
- Each node of $w \in V$ has information $I(w)$
- no node from $V \setminus \{w\}$ knows $I(w)$.
- Node $v$ should receive the information $\bigcup_{w \in V} I(w)$. 
Definition of a Broadcasts and Accumulation

**Definition of Broadcast:**

Given are $G = (V, E)$ and $v \in V$.

- $v$ has information $I(v)$ and
- no node from $V \setminus \{v\}$ knows $I(v)$.
- Each node of $V \setminus \{v\}$ has to receive information $I(v)$.

**Definition of Accumulation:**

Given are $G = (V, E)$ and $v \in V$.

- Each node of $w \in V$ has information $I(w)$
- no node from $V \setminus \{w\}$ knows $I(w)$.
- Node $v$ should receive the information $\bigcup_{w \in V} I(w)$. 
Definition of a Broadcasts and Accumulation

**Definition of Broadcast:**

Given are \( G = (V, E) \) and \( v \in V \).

- \( v \) has information \( I(v) \) and
- no node from \( V \setminus \{v\} \) knows \( I(v) \).
- Each node of \( V \setminus \{v\} \) has to receive information \( I(v) \).

**Definition of Accumulation:**

Given are \( G = (V, E) \) and \( v \in V \).

- Each node of \( w \in V \) has information \( I(w) \) and
- no node from \( V \setminus \{w\} \) knows \( I(w) \).
- Node \( v \) should receive the information \( \bigcup_{w \in V} I(w) \).
Definition of a Broadcasts and Accumulation

Definition of Broadcast:
Given are \( G = (V, E) \) and \( v \in V \).
- \( v \) has information \( I(v) \) and
- no node from \( V \setminus \{v\} \) knows \( I(v) \).
- Each node of \( V \setminus \{v\} \) has to receive information \( I(v) \).

Definition of Accumulation:
Given are \( G = (V, E) \) and \( v \in V \).
- Each node of \( w \in V \) has information \( I(w) \) and
- no node from \( V \setminus \{w\} \) knows \( I(w) \).
- Node \( v \) should receive the information \( \bigcup_{w \in V} I(w) \).
Definition of a Broadcasts and Accumulation

**Definition of Broadcast:**
Given are $G = (V, E)$ and $v \in V$.

- $v$ has information $I(v)$ and
- no node from $V \setminus \{v\}$ knows $I(v)$.
- Each node of $V \setminus \{v\}$ has to receive information $I(v)$.

**Definition of Accumulation:**
Given are $G = (V, E)$ and $v \in V$.

- Each node of $w \in V$ has information $I(w)$ and
- no node from $V \setminus \{w\}$ knows $I(w)$.
- Node $v$ should receive the information $\bigcup_{w \in V} I(w)$. 
Definition of a Gossip

**Definition of Accumulation:**
Given are $G = (V, E)$ and $v \in V$.

- Each node of $w \in V$ has information $I(w)$.
- No node from $V \setminus \{w\}$ knows $I(w)$.
- Node $v$ should receive the information $\bigcup_{w \in V} I(w)$.

**Definition (Gossip):**
Given is $G = (V, E)$.

- Each node of $w \in V$ has information $I(w)$.
- No node from $V \setminus \{w\}$ knows $I(w)$.
- Each node of $v \in V$ should receive the information $\bigcup_{w \in V} I(w)$.
Definition of a Gossip

**Definition of Accumulation:**

Given are $G = (V, E)$ and $v \in V$.

- Each node of $w \in V$ has information $I(w)$
- No node from $V \setminus \{w\}$ knows $I(w)$.
- Node $v$ should receive the information $\bigcup_{w \in V} I(w)$.

**Definition (Gossip):**

Given is $G = (V, E)$.

- Each node of $w \in V$ has information $I(w)$
- No node from $V \setminus \{w\}$ knows $I(w)$.
- Each node of $v \in V$ should receive the information $\bigcup_{w \in V} I(w)$.
Definition of a Gossip

**Definition of Accumulation:**

Given are $G = (V, E)$ and $v \in V$.

- Each node of $w \in V$ has information $I(w)$
- no node from $V \setminus \{w\}$ knows $I(w)$.
- Node $v$ should receive the information $\bigcup_{w \in V} I(w)$.

**Definition (Gossip):**

Given is $G = (V, E)$.

- Each node of $w \in V$ has information $I(w)$
- no node from $V \setminus \{w\}$ knows $I(w)$.
- Each node of $v \in V$ should receive the information $\bigcup_{w \in V} I(w)$.
Definition of a Gossip

Definition of Accumulation:

Given are $G = (V, E)$ and $v \in V$.

- Each node of $w \in V$ has information $I(w)$
- No node from $V \setminus \{w\}$ knows $I(w)$.
- Node $v$ should receive the information $\bigcup_{w \in V} I(w)$.

Definition (Gossip):

Given is $G = (V, E)$.

- Each node of $w \in V$ has information $I(w)$
- No node from $V \setminus \{w\}$ knows $I(w)$.
- Each node of $v \in V$ should receive the information $\bigcup_{w \in V} I(w)$. 
Definition of a Gossip

**Definition of Accumulation:**

- Given are $G = (V, E)$ and $v \in V$.
  - Each node of $w \in V$ has information $I(w)$.
  - No node from $V \setminus \{w\}$ knows $I(w)$.
  - Node $v$ should receive the information $\bigcup_{w \in V} I(w)$.

**Definition (Gossip):**

- Given is $G = (V, E)$.
  - Each node of $w \in V$ has information $I(w)$.
  - No node from $V \setminus \{w\}$ knows $I(w)$.
  - Each node of $v \in V$ should receive the information $\bigcup_{w \in V} I(w)$.
Definition of a Gossip

Definition of Accumulation:

Given are $G = (V, E)$ and $v \in V$.

- Each node of $w \in V$ has information $I(w)$ and
- no node from $V \setminus \{w\}$ knows $I(w)$.
- Node $v$ should receive the information $\bigcup_{w \in V} I(w)$.

Definition (Gossip):

Given is $G = (V, E)$.

- Each node of $w \in V$ has information $I(w)$ and
- no node from $V \setminus \{w\}$ knows $I(w)$.
- Each node of $v \in V$ should receive the information $\bigcup_{w \in V} I(w)$.
Definition of a Gossip

**Definition of Accumulation:**

Given are \( G = (V, E) \) and \( v \in V \).
- Each node of \( w \in V \) has information \( I(w) \) and
- no node from \( V \setminus \{w\} \) knows \( I(w) \).
- Node \( v \) should receive the information \( \bigcup_{w \in V} I(w) \).

**Definition (Gossip):**

Given is \( G = (V, E) \).
- Each node of \( w \in V \) has information \( I(w) \) and
- no node from \( V \setminus \{w\} \) knows \( I(w) \).
- Each node of \( v \in V \) should receive the information \( \bigcup_{w \in V} I(w) \).
Definition of a Gossip

**Definition of Accumulation:**

Given are $G = (V, E)$ and $v \in V$.

- Each node of $w \in V$ has information $I(w)$ and
- no node from $V \setminus \{w\}$ knows $I(w)$.
- Node $v$ should receive the information $\bigcup_{w \in V} I(w)$.

**Definition (Gossip):**

Given is $G = (V, E)$.

- Each node of $w \in V$ has information $I(w)$ and
- no node from $V \setminus \{w\}$ knows $I(w)$.
- Each node of $v \in V$ should receive the information $\bigcup_{w \in V} I(w)$. 
Types of Communication

- **Telegraph-Mode**: Communication is directed.
  - Is also called one-way communication.

- **Telephone-Mode**: Information is exchanged.
  - Is also called two-way communication.
  - Communication only between neighbours.
  - Communication is done in rounds.
  - In each round the active edges are a matching.
  - Each round uses one time-unit.
Types of Communication

- **Telegraph-Mode:** Communication is directed.
  - Is also called **one-way communication.**

- **Telephone-Mode:** Information is exchanged.
  - Is also called **two-way communication.**

  - Communication only between neighbours.
  - Communication is done in rounds.
  - In each round the active edges are a matching.
  - Each round uses one time-unit.
Types of Communication

- **Telegraph-Mode**: Communication is directed.
  - Is also called one-way communication.

- **Telephone-Mode**: Information is exchanged.
  - Is also called two-way communication.
  - Communication only between neighbours.
  - Communication is done in rounds.
  - In each round the active edges are a matching.
  - Each round uses one time-unit.
Types of Communication

- **Telegraph-Mode**: Communication is directed.
  - Is also called one-way communication.
- **Telephone-Mode**: Information is exchanged.
  - Is also called two-way communication.
- Communication only between neighbours.
- Communication is done in rounds.
- In each round the active edges are a matching.
- Each round uses one time-unit.
Types of Communication

- Telegraph-Mode: Communication is directed.
  - Is also called one-way communication.
- Telephone-Mode: Information is exchanged.
  - Is also called two-way communication.
- Communication only between neighbours.
- Communication is done in rounds.
- In each round the active edges are a matching.
- Each round uses one time-unit.
Types of Communication

- **Telegraph-Mode**: Communication is directed.
  - Is also called one-way communication.

- **Telephone-Mode**: Information is exchanged.
  - Is also called two-way communication.
  - Communication only between neighbours.
  - Communication is done in rounds.
  - In each round the active edges are a matching.
  - Each round uses one time-unit.
Types of Communication

- Telegraph-Mode: Communication is directed.
- Is also called one-way communication.
- Telephone-Mode: Information is exchanged.
- Is also called two-way communication.
- Communication only between neighbours.
- Communication is done in rounds.
- In each round the active edges are a matching.
- Each round uses one time-unit.
Types of Communication

- **Telegraph-Mode**: Communication is directed.
  - Is also called one-way communication.
- **Telephone-Mode**: Information is exchanged.
  - Is also called two-way communication.
- Communication only between neighbours.
- Communication is done in rounds.
- In each round the active edges are a matching.
- **Each round uses one time-unit.**
Types of Communication

- **Telegraph-Mode**: Communication is directed.
  - Is also called one-way communication.
- **Telephone-Mode**: Information is exchanged.
  - Is also called two-way communication.
- Communication only between neighbours.
- Communication is done in rounds.
- In each round the active edges are a matching.
- Each round uses one time-unit.
Types of Communication

- In the broadcast-problem the information of one node is transferred to all others.
- The accumulation-problem is a “inverse” broadcast.
- A gossip distributes the sum of all informations to all nodes.
- In each round the communication is done by a matching.
- The communication on an edge may be one-way or two-way, depending on the mode.
- The size of send date is ignored.
In the broadcast-problem the information of one node is transferred to all others.

The accumulation-problem is a “inverse” broadcast.

A gossip distributes the sum of all informations to all nodes.

In each round the communication is done by a matching.

The communication on an edge may be one-way or two-way, depending on the mode.

The size of send date is ignored.
In the broadcast-problem the information of one node is transferred to all others.

The accumulation-problem is a “inverse” broadcast.

A gossip distributes the sum of all informations to all nodes.

In each round the communication is done by a matching.

The communication on an edge may be one-way or two-way, depending on the mode.

The size of send date is ignored.
Types of Communication

- In the broadcast-problem the information of one node is transferred to all others.
- The accumulation-problem is a “inverse” broadcast.
- A gossip distributes the sum of all informations to all nodes.
- In each round the communication is done by a matching.
- The communication on an edge may be one-way or two-way, depending on the mode.
- The size of send date is ignored.
Types of Communication

- In the broadcast-problem the information of one node is transferred to all others.
- The accumulation-problem is a “inverse” broadcast.
- A gossip distributes the sum of all informations to all nodes.
- In each round the communication is done by a matching.
- The communication on an edge may be one-way or two-way, depending on the mode.
- The size of send date is ignored.
In the broadcast-problem the information of one node is transferred to all others.

The accumulation-problem is a “inverse” broadcast.

A gossip distributes the sum of all informations to all nodes.

In each round the communication is done by a matching.

The communication on an edge may be one-way or two-way, depending on the mode.

The size of send date is ignored.
Types of Communication

- In the broadcast-problem the information of one node is transfered to all others.
- The accumulation-problem is a “inverse” broadcast.
- A gossip distributes the sum of all informations to all nodes.
- In each round the communication is done by a matching.
- The communication on an edge may be one-way or two-way, depending on the mode.
- The size of send date is ignored.
Definition

- By $\text{comm}(A)$ we denote the complexity (number of rounds) of a communication-algorithm.

- $r(G) = \min \{ \text{comm}(A) \mid A \text{ is a one-way algorithm for the gossip-problem on } G \}$

- $r_2(G) = \min \{ \text{comm}(A) \mid A \text{ is a two-way algorithm for the gossip-problem on } G \}$

- $b(v, G) = \min \{ \text{comm}(A) \mid A \text{ is a one-way algorithm for the broadcast-problem on } G \text{ and } v \}$

- $b_2(v, G) = \min \{ \text{comm}(A) \mid A \text{ is a two-way algorithm for the broadcast-problem on } G \text{ and } v \}$

- $a(v, G) = \min \{ \text{comm}(A) \mid A \text{ is a one-way algorithm for the accumulations-problem on } G \text{ and } v \}$

- $a_2(v, G) = \min \{ \text{comm}(A) \mid A \text{ is a two-way algorithm for the accumulations-problem on } G \text{ and } v \}$
Definition

- By \( \text{comm}(A) \) we denote the complexity (number of rounds) of a communication-algorithm.

- \( r(G) = \min \{ \text{comm}(A) \mid A \text{ is a one-way algorithm for the gossip-problem on } G \} \)

- \( r_2(G) = \min \{ \text{comm}(A) \mid A \text{ is a two-way algorithm for the gossip-problem on } G \} \)

- \( b(v, G) = \min \{ \text{comm}(A) \mid A \text{ is a one-way algorithm for the broadcast-problem on } G \text{ and } v \} \)

- \( b_2(v, G) = \min \{ \text{comm}(A) \mid A \text{ is a two-way algorithm for the broadcast-problem on } G \text{ and } v \} \)

- \( a(v, G) = \min \{ \text{comm}(A) \mid A \text{ is a one-way algorithm for the accumulations-problem on } G \text{ and } v \} \)

- \( a_2(v, G) = \min \{ \text{comm}(A) \mid A \text{ is a two-way algorithm for the accumulations-problem on } G \text{ and } v \} \)
Definition

- By $\text{comm}(A)$ we denote the complexity (number of rounds) of a communication-algorithm.
- $r(G) = \min\{\text{comm}(A) \mid A \text{ is a one-way algorithm for the gossip-problem on } G\}$
- $r_2(G) = \min\{\text{comm}(A) \mid A \text{ is a two-way algorithm for the gossip-problem on } G\}$
- $b(v, G) = \min\{\text{comm}(A) \mid A \text{ is a one-way algorithm for the broadcast-problem on } G \text{ and } v\}$
- $b_2(v, G) = \min\{\text{comm}(A) \mid A \text{ is a two-way algorithm for the broadcast-problem on } G \text{ and } v\}$
- $a(v, G) = \min\{\text{comm}(A) \mid A \text{ is a one-way algorithm for the accumulations-problem on } G \text{ and } v\}$
- $a_2(v, G) = \min\{\text{comm}(A) \mid A \text{ is a two-way algorithm for the accumulations-problem on } G \text{ and } v\}$
Definition

- By $\text{comm}(A)$ we denote the complexity (number of rounds) of a communication-algorithm.

- $r(G) = \min\{\text{comm}(A) \mid A$ is a one-way algorithm for the gossip-problem on $G\}$

- $r_2(G) = \min\{\text{comm}(A) \mid A$ is a two-way algorithm for the gossip-problem on $G\}$

- $b(v, G) = \min\{\text{comm}(A) \mid A$ is a one-way algorithm for the broadcast-problem on $G$ and $v\}$

- $b_2(v, G) = \min\{\text{comm}(A) \mid A$ is a two-way algorithm for the broadcast-problem on $G$ and $v\}$

- $a(v, G) = \min\{\text{comm}(A) \mid A$ is a one-way algorithm for the accumulations-problem on $G$ and $v\}$

- $a_2(v, G) = \min\{\text{comm}(A) \mid A$ is a two-way algorithm for the accumulations-problem on $G$ and $v\}$
Definition

- By \( \text{comm}(A) \) we denote the complexity (number of rounds) of a communication-algorithm.

- \( r(G) = \min \{ \text{comm}(A) \mid A \text{ is a one-way algorithm for the gossip-problem on } G \} \)

- \( r_2(G) = \min \{ \text{comm}(A) \mid A \text{ is a two-way algorithm for the gossip-problem on } G \} \)

- \( b(v, G) = \min \{ \text{comm}(A) \mid A \text{ is a one-way algorithm for the broadcast-problem on } G \text{ and } v \} \)

- \( b_2(v, G) = \min \{ \text{comm}(A) \mid A \text{ is a two-way algorithm for the broadcast-problem on } G \text{ and } v \} \)

- \( a(v, G) = \min \{ \text{comm}(A) \mid A \text{ is a one-way algorithm for the accumulations-problem on } G \text{ and } v \} \)

- \( a_2(v, G) = \min \{ \text{comm}(A) \mid A \text{ is a two-way algorithm for the accumulations-problem on } G \text{ and } v \} \)
Definition

- By $\text{comm}(A)$ we denote the complexity (number of rounds) of a communication-algorithm.

  \[
  r(G) = \min \{ \text{comm}(A) \mid A \text{ is a one-way algorithm for the gossip-problem on } G \} 
  \]

- $r_2(G) = \min \{ \text{comm}(A) \mid A \text{ is a two-way algorithm for the gossip-problem on } G \}$

- $b(v, G) = \min \{ \text{comm}(A) \mid A \text{ is a one-way algorithm for the broadcast-problem on } G \text{ and } v \}$

- $b_2(v, G) = \min \{ \text{comm}(A) \mid A \text{ is a two-way algorithm for the broadcast-problem on } G \text{ and } v \}$

- $a(v, G) = \min \{ \text{comm}(A) \mid A \text{ is a one-way algorithm for the accumulations-problem on } G \text{ and } v \}$

- $a_2(v, G) = \min \{ \text{comm}(A) \mid A \text{ is a two-way algorithm for the accumulations-problem on } G \text{ and } v \}$
Definition

- By $comm(A)$ we denote the complexity (number of rounds) of a communication-algorithm.

- $r(G) = \min\{comm(A) \mid A \text{ is a one-way algorithm for the gossip-problem on } G\}$

- $r_2(G) = \min\{comm(A) \mid A \text{ is a two-way algorithm for the gossip-problem on } G\}$

- $b(v, G) = \min\{comm(A) \mid A \text{ is a one-way algorithm for the broadcast-problem on } G \text{ and } v\}$

- $b_2(v, G) = \min\{comm(A) \mid A \text{ is a two-way algorithm for the broadcast-problem on } G \text{ and } v\}$

- $a(v, G) = \min\{comm(A) \mid A \text{ is a one-way algorithm for the accumulations-problem on } G \text{ and } v\}$

- $a_2(v, G) = \min\{comm(A) \mid A \text{ is a two-way algorithm for the accumulations-problem on } G \text{ and } v\}$


**Definition**

- By $\text{comm}(A)$ we denote the complexity (number of rounds) of a communication-algorithm.

- $r(G) = \min \{ \text{comm}(A) \mid \text{A is a one-way algorithm for the gossip-problem on } G \}$

- $r_2(G) = \min \{ \text{comm}(A) \mid \text{A is a two-way algorithm for the gossip-problem on } G \}$

- $b(v, G) = \min \{ \text{comm}(A) \mid \text{A is a one-way algorithm for the broadcast-problem on } G \text{ and } v \}$

- $b_2(v, G) = \min \{ \text{comm}(A) \mid \text{A is a two-way algorithm for the broadcast-problem on } G \text{ and } v \}$

- $a(v, G) = \min \{ \text{comm}(A) \mid \text{A is a one-way algorithm for the accumulations-problem on } G \text{ and } v \}$

- $a_2(v, G) = \min \{ \text{comm}(A) \mid \text{A is a two-way algorithm for the accumulations-problem on } G \text{ and } v \}$
Definition

- $b(G) = \max \{ b(v, G) \mid v \in V \}$
- $b_2(G) = \max \{ b_2(v, G) \mid v \in V \}$
- $a(G) = \max \{ a(v, G) \mid v \in V \}$
- $a_2(G) = \max \{ a_2(v, G) \mid v \in V \}$
- $\min b(G) = \min \{ b(v, G) \mid v \in V \}$
- $\min a(G) = \min \{ a(v, G) \mid v \in V \}$
Definition

- $b(G) = \max\{b(v, G) \mid v \in V\}$
- $b_2(G) = \max\{b_2(v, G) \mid v \in V\}$
- $a(G) = \max\{a(v, G) \mid v \in V\}$
- $a_2(G) = \max\{a_2(v, G) \mid v \in V\}$
- $\min b(G) = \min\{b(v, G) \mid v \in V\}$
- $\min a(G) = \min\{a(v, G) \mid v \in V\}$
Definition

- $b(G) = \max\{b(v, G) \mid v \in V\}$
- $b_2(G) = \max\{b_2(v, G) \mid v \in V\}$
- $a(G) = \max\{a(v, G) \mid v \in V\}$
- $a_2(G) = \max\{a_2(v, G) \mid v \in V\}$
- $\min b(G) = \min\{b(v, G) \mid v \in V\}$
- $\min a(G) = \min\{a(v, G) \mid v \in V\}$
Definition

- $b(G) = \max\{b(v, G) \mid v \in V\}$
- $b_2(G) = \max\{b_2(v, G) \mid v \in V\}$
- $a(G) = \max\{a(v, G) \mid v \in V\}$
- $a_2(G) = \max\{a_2(v, G) \mid v \in V\}$
- $\min b(G) = \min\{b(v, G) \mid v \in V\}$
- $\min a(G) = \min\{a(v, G) \mid v \in V\}$
**Definition**

- \( b(G) = \max\{b(v, G) \mid v \in V\} \)
- \( b_2(G) = \max\{b_2(v, G) \mid v \in V\} \)
- \( a(G) = \max\{a(v, G) \mid v \in V\} \)
- \( a_2(G) = \max\{a_2(v, G) \mid v \in V\} \)
- \( \text{min}_b(G) = \min\{b(v, G) \mid v \in V\} \)
- \( \text{min}_a(G) = \min\{a(v, G) \mid v \in V\} \)
**Definition**

- \( b(G) = \max\{b(v, G) \mid v \in V\} \)
- \( b_2(G) = \max\{b_2(v, G) \mid v \in V\} \)
- \( a(G) = \max\{a(v, G) \mid v \in V\} \)
- \( a_2(G) = \max\{a_2(v, G) \mid v \in V\} \)
- \( \min b(G) = \min\{b(v, G) \mid v \in V\} \)
- \( \min a(G) = \min\{a(v, G) \mid v \in V\} \)
Definition

- \( b(G) = \max \{ b(v, G) \mid v \in V \} \)
- \( b_2(G) = \max \{ b_2(v, G) \mid v \in V \} \)
- \( a(G) = \max \{ a(v, G) \mid v \in V \} \)
- \( a_2(G) = \max \{ a_2(v, G) \mid v \in V \} \)
- \( \min b(G) = \min \{ b(v, G) \mid v \in V \} \)
- \( \min a(G) = \min \{ a(v, G) \mid v \in V \} \)
Some Results

- For each graph $G$ and $v \in V$ we have:
  - $a_2(v, G) = b_2(v, G)$
  - $a(v, G) = b(v, G)$
  - $a(G) = b(G)$
  - $\text{mina}(G) = \text{minb}(G)$
  - $b(v, G) = b_2(v, G)$
  - $b(G) = b_2(G)$

- Note: reverse broadcast is accumulation.

- There exists a graph $G$ with: $r(G) = 2 \cdot r_2(G)$.

- Note: 2-clique or cycle of length four.

- The following holds: $\text{minb}(G) \leq b(G) \leq r_2(G) \leq r(G) \leq 2 \cdot r_2(G)$.

- The inequalities result from the definitions.

- $\text{minb}(L(n)) = \lceil n/2 \rceil$

- Optimal broadcast on a line start in the center of the line.

- $b(L(n)) = n - 1$

- A message from the left has to traverse all edges.
Some Results

- For each graph $G$ and $v \in V$ we have:
  - $a_2(v, G) = b_2(v, G)$
  - $a(v, G) = b(v, G)$
  - $a(G) = b(G)$
  - $\text{mina}(G) = \text{minb}(G)$
  - $b(v, G) = b_2(v, G)$
  - $b(G) = b_2(G)$

  **Note:** reverse broadcast is accumulation.

- There exists a graph $G$ with: $r(G) = 2 \cdot r_2(G)$.

  **Note:** 2-clique or cycle of length four.

- The following holds: $\text{minb}(G) \leq b(G) \leq r_2(G) \leq r(G) \leq 2 \cdot r_2(G)$.

- The inequalities result from the definitions.

- $\text{minb}(L(n)) = \lceil n/2 \rceil$

- Optimal broadcast on a line start in the center of the line.

- $b(L(n)) = n - 1$

- A message from the left has to traverse all edges.
Some Results

- For each graph $G$ and $v \in V$ we have:
  - $a_2(v, G) = b_2(v, G)$
  - $a(v, G) = b(v, G)$
  - $a(G) = b(G)$
  - $\text{mina}(G) = \text{minb}(G)$
  - $b(v, G) = b_2(v, G)$
  - $b(G) = b_2(G)$

- Note: reverse broadcast is accumulation.
- There exists a graph $G$ with: $r(G) = 2 \cdot r_2(G)$.
- Note: 2-clique or cycle of length four.
- The following holds: $\text{minb}(G) \leq b(G) \leq r_2(G) \leq r(G) \leq 2 \cdot r_2(G)$.
- The inequalities result from the definitions.
- $\text{minb}(L(n)) = \lceil n/2 \rceil$
- Optimal broadcast on a line start in the center of the line.
- $b(L(n)) = n - 1$
- A message from the left has to traverse all edges.
Some Results

- For each graph $G$ and $v \in V$ we have:
  - $a_2(v, G) = b_2(v, G)$
  - $a(v, G) = b(v, G)$
  - $a(G) = b(G)$
  - $\text{mina}(G) = \text{minb}(G)$
  - $b(v, G) = b_2(v, G)$
  - $b(G) = b_2(G)$

- Note: reverse broadcast is accumulation.

- There exists a graph $G$ with: $r(G) = 2 \cdot r_2(G)$.

- Note: 2-clique or cycle of length four.

- The following holds: $\text{minb}(G) \leq b(G) \leq r_2(G) \leq r(G) \leq 2 \cdot r_2(G)$.

- The inequalities result from the definitions.

- $\text{minb}(L(n)) = \lceil n/2 \rceil$

- Optimal broadcast on a line start in the center of the line.

- $b(L(n)) = n - 1$

- A message from the left has to traverse all edges.
For each graph $G$ and $v \in V$ we have:

- $a_2(v, G) = b_2(v, G)$
- $a(v, G) = b(v, G)$
- $a(G) = b(G)$
- $\text{mina}(G) = \text{minb}(G)$
- $b(v, G) = b_2(v, G)$
- $b(G) = b_2(G)$

Note: reverse broadcast is accumulation.

There exists a graph $G$ with: $r(G) = 2 \cdot r_2(G)$.

Note: 2-clique or cycle of length four.

The following holds: $\text{minb}(G) \leq b(G) \leq r_2(G) \leq r(G) \leq 2 \cdot r_2(G)$.

The inequalities result from the definitions.

$\text{minb}(L(n)) = \lceil n/2 \rceil$

Optimal broadcast on a line start in the center of the line.

$b(L(n)) = n - 1$

A message from the left has to traverse all edges.
Some Results

- For each graph $G$ and $v \in V$ we have:
  - $a_2(v, G) = b_2(v, G)$
  - $a(v, G) = b(v, G)$
  - $a(G) = b(G)$
  - $\text{mina}(G) = \text{minb}(G)$
  - $b(v, G) = b_2(v, G)$
  - $b(G) = b_2(G)$

- Note: reverse broadcast is accumulation.
- There exists a graph $G$ with: $r(G) = 2 \cdot r_2(G)$.
- Note: 2-clique or cycle of length four.
- The following holds: $\text{minb}(G) \leq b(G) \leq r_2(G) \leq r(G) \leq 2 \cdot r_2(G)$.
- The inequalities result from the definitions.
- $\text{minb}(L(n)) = \lceil n/2 \rceil$
- Optimal broadcast on a line start in the center of the line.
- $b(L(n)) = n - 1$
- A message from the left has to traverse all edges.
Some Results

- For each graph \( G \) and \( v \in V \) we have:
  - \( a_2(v, G) = b_2(v, G) \)
  - \( a(v, G) = b(v, G) \)
  - \( a(G) = b(G) \)
  - \( \text{mina}(G) = \text{minb}(G) \)
  - \( b(v, G) = b_2(v, G) \)
  - \( b(G) = b_2(G) \)

- Note: reverse broadcast is accumulation.
- There exists a graph \( G \) with: \( r(G) = 2 \cdot r_2(G) \).
- Note: 2-clique or cycle of length four.
- The following holds: \( \text{minb}(G) \leq b(G) \leq r_2(G) \leq r(G) \leq 2 \cdot r_2(G) \).
- The inequalities result from the definitions.
- \( \text{minb}(L(n)) = \lceil n/2 \rceil \)
- Optimal broadcast on a line start in the center of the line.
- \( b(L(n)) = n - 1 \)
- A message from the left has to traverse all edges.
Some Results

For each graph $G$ and $v \in V$ we have:

- $a_2(v, G) = b_2(v, G)$
- $a(v, G) = b(v, G)$
- $a(G) = b(G)$
- $\min a(G) = \min b(G)$
- $b(v, G) = b_2(v, G)$
- $b(G) = b_2(G)$

Note: reverse broadcast is accumulation.

There exists a graph $G$ with: $r(G) = 2 \cdot r_2(G)$.

Note: 2-clique or cycle of length four.

The following holds: $\min b(G) \leq b(G) \leq r_2(G) \leq r(G) \leq 2 \cdot r_2(G)$.

The inequalities result from the definitions.

$\min b(L(n)) = \lceil n/2 \rceil$

Optimal broadcast on a line start in the center of the line.

$b(L(n)) = n - 1$

A message from the left has to traverse all edges.
Some Results

- For each graph $G$ and $v \in V$ we have:
  - $a_2(v, G) = b_2(v, G)$
  - $a(v, G) = b(v, G)$
  - $a(G) = b(G)$
  - $\text{mina}(G) = \text{minb}(G)$
  - $b(v, G) = b_2(v, G)$
  - $b(G) = b_2(G)$

- **Note**: reverse broadcast is accumulation.
- **There exists a graph $G$ with**: $r(G) = 2 \cdot r_2(G)$.
- **Note**: 2-clique or cycle of length four.
- The following holds: $\text{minb}(G) \leq b(G) \leq r_2(G) \leq r(G) \leq 2 \cdot r_2(G)$.
- The inequalities result from the definitions.
- $\text{minb}(L(n)) = \lceil n/2 \rceil$
- Optimal broadcast on a line start in the center of the line.
- $b(L(n)) = n - 1$
- A message from the left has to traverse all edges.
Some Results

- For each graph $G$ and $v \in V$ we have:
  - $a_2(v, G) = b_2(v, G)$
  - $a(v, G) = b(v, G)$
  - $a(G) = b(G)$
  - $\text{mina}(G) = \text{minb}(G)$
  - $b(v, G) = b_2(v, G)$
  - $b(G) = b_2(G)$

- Note: reverse broadcast is accumulation.

- There exists a graph $G$ with: $r(G) = 2 \cdot r_2(G)$.

- Note: 2-clique or cycle of length four.

- The following holds: $\text{minb}(G) \leq b(G) \leq r_2(G) \leq r(G) \leq 2 \cdot r_2(G)$.

- The inequalities result from the definitions.

- $\text{minb}(L(n)) = \lceil n/2 \rceil$

- Optimal broadcast on a line start in the center of the line.

- $b(L(n)) = n - 1$

- A message from the left has to traverse all edges.
Some Results

- For each graph $G$ and $v \in V$ we have:
  - $a_2(v, G) = b_2(v, G)$
  - $a(v, G) = b(v, G)$
  - $a(G) = b(G)$
  - $\min a(G) = \min b(G)$
  - $b(v, G) = b_2(v, G)$
  - $b(G) = b_2(G)$

- Note: reverse broadcast is accumulation.
- There exists a graph $G$ with: $r(G) = 2 \cdot r_2(G)$.
- Note: 2-clique or cycle of length four.
- The following holds: $\min b(G) \leq b(G) \leq r_2(G) \leq r(G) \leq 2 \cdot r_2(G)$.
- The inequalities result from the definitions.
- $\min b(L(n)) = \lceil n/2 \rceil$
- Optimal broadcast on a line start in the center of the line.
- $b(L(n)) = n - 1$
- A message from the left has to traverse all edges.
Some Results

- For each graph $G$ and $v \in V$ we have:
  - $a_2(v, G) = b_2(v, G)$
  - $a(v, G) = b(v, G)$
  - $a(G) = b(G)$
  - $\text{min}_a(G) = \text{min}_b(G)$
  - $b(v, G) = b_2(v, G)$
  - $b(G) = b_2(G)$

- Note: reverse broadcast is accumulation.

- There exists a graph $G$ with: $r(G) = 2 \cdot r_2(G)$.

- Note: 2-clique or cycle of length four.

- The following holds: $\text{min}_b(G) \leq b(G) \leq r_2(G) \leq r(G) \leq 2 \cdot r_2(G)$.

- The inequalities result from the definitions.

- $\text{min}_b(L(n)) = \lceil n/2 \rceil$

- Optimal broadcast on a line start in the center of the line.

- $b(L(n)) = n - 1$

- A message from the left has to traverse all edges.
Some Results

For each graph $G$ and $v \in V$ we have:

- $a_2(v, G) = b_2(v, G)$
- $a(v, G) = b(v, G)$
- $a(G) = b(G)$
- $\text{mina}(G) = \text{minb}(G)$
- $b(v, G) = b_2(v, G)$
- $b(G) = b_2(G)$

Note: reverse broadcast is accumulation.

There exists a graph $G$ with: $r(G) = 2 \cdot r_2(G)$.

Note: 2-clique or cycle of length four.

The following holds: $\text{minb}(G) \leq b(G) \leq r_2(G) \leq r(G) \leq 2 \cdot r_2(G)$.

The inequalities result from the definitions.

$\text{minb}(L(n)) = \lceil n/2 \rceil$

Optimal broadcast on a line start in the center of the line.

$b(L(n)) = n - 1$

A message from the left has to traverse all edges.
Some Results

For each graph $G$ and $v \in V$ we have:

- $a_2(v, G) = b_2(v, G)$
- $a(v, G) = b(v, G)$
- $a(G) = b(G)$
- $\text{mina}(G) = \text{minb}(G)$
- $b(v, G) = b_2(v, G)$
- $b(G) = b_2(G)$

Note: reverse broadcast is accumulation.

There exists a graph $G$ with: $r(G) = 2 \cdot r_2(G)$.

Note: 2-clique or cycle of length four.

The following holds: $\text{minb}(G) \leq b(G) \leq r_2(G) \leq r(G) \leq 2 \cdot r_2(G)$.

The inequalities result from the definitions.

$\text{minb}(L(n)) = \lceil n/2 \rceil$

Optimal broadcast on a line start in the center of the line.

$b(L(n)) = n - 1$

A message from the left has to traverse all edges.
Some Results

For each graph $G$ and $v \in V$ we have:

- $a_2(v, G) = b_2(v, G)$
- $a(v, G) = b(v, G)$
- $a(G) = b(G)$
- $\min(a(G)) = \min(b(G))$
- $b(v, G) = b_2(v, G)$
- $b(G) = b_2(G)$

Note: reverse broadcast is accumulation.

There exists a graph $G$ with: $r(G) = 2 \cdot r_2(G)$.

Note: 2-clique or cycle of length four.

The following holds: $\min(b(G)) \leq b(G) \leq r_2(G) \leq r(G) \leq 2 \cdot r_2(G)$.

The inequalities result from the definitions.

$\min(b(L(n))) = \lceil n/2 \rceil$

Optimal broadcast on a line start in the center of the line.

$b(L(n)) = n - 1$

A message from the left has to traverse all edges.
Some Results

- For each graph $G$ and $v \in V$ we have:
  - $a_2(v, G) = b_2(v, G)$
  - $a(v, G) = b(v, G)$
  - $a(G) = b(G)$
  - $\operatorname{mina}(G) = \operatorname{minb}(G)$
  - $b(v, G) = b_2(v, G)$
  - $b(G) = b_2(G)$

- Note: reverse broadcast is accumulation.

- There exists a graph $G$ with: $r(G) = 2 \cdot r_2(G)$.
- Note: 2-clique or cycle of length four.

- The following holds: $\operatorname{minb}(G) \leq b(G) \leq r_2(G) \leq r(G) \leq 2 \cdot r_2(G)$.

- The inequalities result from the definitions.

- $\operatorname{minb}(L(n)) = \lceil n/2 \rceil$

- Optimal broadcast on a line start in the center of the line.

- $b(L(n)) = n - 1$

- A message from the left has to traverse all edges.
Some Results

- For each graph $G$ and $v \in V$ we have:
  - $a_2(v, G) = b_2(v, G)$
  - $a(v, G) = b(v, G)$
  - $a(G) = b(G)$
  - $\text{mina}(G) = \text{minb}(G)$
  - $b(v, G) = b_2(v, G)$
  - $b(G) = b_2(G)$
- Note: reverse broadcast is accumulation.
- There exists a graph $G$ with: $r(G) = 2 \cdot r_2(G)$.
- Note: 2-clique or cycle of length four.
- The following holds: $\text{minb}(G) \leq b(G) \leq r_2(G) \leq r(G) \leq 2 \cdot r_2(G)$.
- The inequalities result from the definitions.
- $\text{minb}(L(n)) = \lceil n/2 \rceil$
- Optimal broadcast on a line start in the center of the line.
- $b(L(n)) = n - 1$
- A message from the left has to traverse all edges.
Lemma:

For each graph $G$ with $|V| \geq 2$ we have:

- $b(G) \leq r(G) \leq 2 \cdot \min b(G)$
- $b(G) \leq r_2(G) \leq 2 \cdot \min b(G) - 1$

Proof: Consider the following steps.

- Let $v \in V$ with $b(v, G) = \min b(G) = \min a(G) = z$.
- Let $A = E_1, E_2, \cdots E_z$ be the corresponding one-way broadcast-algorithm.
- Let $B = F_1, F_2, \cdots F_z$ be the corresponding one-way accumulation-algorithm.
- Then is $F_1, F_2, \cdots F_z, E_1, E_2, \cdots E_z$ one-way gossip-algorithm.
- Note: in the two-way case holds: $F_z = E_1$.
- Note: For $L(2 \cdot n)$ we have equality.
First Results II

Lemma:

For each graph $G$ with $|V| \geq 2$ we have:

1. $b(G) \leq r(G) \leq 2 \cdot \min b(G)$
2. $b(G) \leq r_2(G) \leq 2 \cdot \min b(G) - 1$

Proof: Consider the following steps.

1. Let $v \in V$ with $b(v, G) = \min b(G) = \min a(G) = z$.
2. Let $A = E_1, E_2, \ldots E_z$ be the corresponding one-way broadcast-algorithm.
3. Let $B = F_1, F_2, \ldots F_z$ be the corresponding one-way accumulation-algorithm.
4. Then is $F_1, F_2, \ldots F_z, E_1, E_2, \ldots E_z$ one-way gossip-algorithm.
5. Note: in the two-way case holds: $F_z = E_1$.
6. Note: For $L(2 \cdot n)$ we have equality.
Lemma:

For each graph $G$ with $|V| \geq 2$ we have:

- $b(G) \leq r(G) \leq 2 \cdot \min b(G)$
- $b(G) \leq r_2(G) \leq 2 \cdot \min b(G) - 1$

Proof: Consider the following steps.

- Let $v \in V$ with $b(v, G) = \min b(G) = \min a(G) = z$.
- Let $A = E_1, E_2, \ldots, E_z$ be the corresponding one-way broadcast-algorithm.
- Let $B = F_1, F_2, \ldots, F_z$ be the corresponding one-way accumulation-algorithm.
- Then is $F_1, F_2, \ldots, F_z, E_1, E_2, \ldots, E_z$ one-way gossip-algorithm.
- Note: in the two-way case holds: $F_z = E_1$.
- Note: For $L(2 \cdot n)$ we have equality.
Lemma:

For each graph $G$ with $|V| \geq 2$ we have:

- $b(G) \leq r(G) \leq 2 \cdot \min b(G)$
- $b(G) \leq r_2(G) \leq 2 \cdot \min b(G) - 1$

Proof: Consider the following steps.

- Let $v \in V$ with $b(v, G) = \min b(G) = \min a(G) = z$.
- Let $A = E_1, E_2, \ldots, E_z$ be the corresponding one-way broadcast-algorithm.
- Let $B = F_1, F_2, \ldots, F_z$ be the corresponding one-way accumulation-algorithm.
- Then is $F_1, F_2, \ldots, F_z, E_1, E_2, \ldots, E_z$ one-way gossip-algorithm.
- Note: in the two-way case holds: $F_z = E_1$.
- Note: For $L(2 \cdot n)$ we have equality.
Lemma:

For each graph $G$ with $|V| \geq 2$ we have:

- $b(G) \leq r(G) \leq 2 \cdot \min b(G)$
- $b(G) \leq r_2(G) \leq 2 \cdot \min b(G) - 1$

Proof: Consider the following steps.

- Let $v \in V$ with $b(v, G) = \min b(G) = \min a(G) = z$.
- Let $A = E_1, E_2, \cdots E_z$ be the corresponding one-way broadcast-algorithm.
- Let $B = F_1, F_2, \cdots F_z$ be the corresponding one-way accumulation-algorithm.
- Then is $F_1, F_2, \cdots F_z, E_1, E_2, \cdots E_z$ one-way gossip-algorithm.
- Note: in the two-way case holds: $F_z = E_1$.
- Note: For $L(2 \cdot n)$ we have equality.
First Results II

Lemma:

For each graph $G$ with $|V| \geq 2$ we have:

- $b(G) \leq r(G) \leq 2 \cdot \min b(G)$
- $b(G) \leq r_2(G) \leq 2 \cdot \min b(G) - 1$

Proof: Consider the following steps.

- Let $v \in V$ with $b(v, G) = \min b(G) = \min a(G) = z$.
- Let $A = E_1, E_2, \cdots, E_z$ be the corresponding one-way broadcast-algorithm.
- Let $B = F_1, F_2, \cdots, F_z$ be the corresponding one-way accumulation-algorithm.
- Then is $F_1, F_2, \cdots, F_z, E_1, E_2, \cdots, E_z$ one-way gossip-algorithm.
- Note: in the two-way case holds: $F_z = E_1$.
- Note: For $L(2 \cdot n)$ we have equality.
First Results II

Lemma:

For each graph \( G \) with \(|V| \geq 2\) we have:

- \( b(G) \leq r(G) \leq 2 \cdot \min b(G) \)
- \( b(G) \leq r_2(G) \leq 2 \cdot \min b(G) - 1 \)

Proof: Consider the following steps.

- Let \( v \in V \) with \( b(v, G) = \min b(G) = \min a(G) = z \).
- Let \( A = E_1, E_2, \cdots, E_z \) be the corresponding one-way broadcast-algorithm.
- Let \( B = F_1, F_2, \cdots, F_z \) be the corresponding one-way accumulation-algorithm.
  
  Then is \( F_1, F_2, \cdots, F_z, E_1, E_2, \cdots, E_z \) one-way gossip-algorithm.

Note: in the two-way case holds: \( F_z = E_1 \).

Note: For \( L(2 \cdot n) \) we have equality.
Lemma:

For each graph $G$ with $|V| \geq 2$ we have:

- $b(G) \leq r(G) \leq 2 \cdot \text{minb}(G)$
- $b(G) \leq r_2(G) \leq 2 \cdot \text{minb}(G) - 1$

Proof: Consider the following steps.

- Let $v \in V$ with $b(v, G) = \text{minb}(G) = \text{mina}(G) = z$.
- Let $A = E_1, E_2, \cdots E_z$ be the corresponding one-way broadcast-algorithm.
- Let $B = F_1, F_2, \cdots F_z$ be the corresponding one-way accumulation-algorithm.
- Then is $F_1, F_2, \cdots F_z, E_1, E_2, \cdots E_z$ one-way gossip-algorithm.
- Note: in the two-way case holds: $F_z = E_1$.
- Note: For $L(2 \cdot n)$ we have equality.
First Results II

Lemma:
For each graph $G$ with $|V| \geq 2$ we have:

- $b(G) \leq r(G) \leq 2 \cdot \min b(G)$
- $b(G) \leq r_2(G) \leq 2 \cdot \min b(G) - 1$

Proof: Consider the following steps.

- Let $v \in V$ with $b(v, G) = \min b(G) = \min a(G) = z$.
- Let $A = E_1, E_2, \cdots E_z$ be the corresponding one-way broadcast-algorithm.
- Let $B = F_1, F_2, \cdots F_z$ be the corresponding one-way accumulation-algorithm.
- Then is $F_1, F_2, \cdots F_z, E_1, E_2, \cdots E_z$ one-way gossip-algorithm.
- Note: in the two-way case holds: $F_z = E_1$.
- Note: For $L(2 \cdot n)$ we have equality.
First Results II

Lemma:
For each graph $G$ with $|V| \geq 2$ we have:

- $b(G) \leq r(G) \leq 2 \cdot \min b(G)$
- $b(G) \leq r_2(G) \leq 2 \cdot \min b(G) - 1$

Proof: Consider the following steps.

- Let $v \in V$ with $b(v, G) = \min b(G) = \min a(G) = z$.
- Let $A = E_1, E_2, \cdots E_z$ be the corresponding one-way broadcast-algorithm.
- Let $B = F_1, F_2, \cdots F_z$ be the corresponding one-way accumulation-algorithm.
- Then is $F_1, F_2, \cdots F_z, E_1, E_2, \cdots E_z$ one-way gossip-algorithm.
- Note: in the two-way case holds: $F_z = E_1$.
- Note: For $L(2 \cdot n)$ we have equality.
Lemma:
For each even \( n \) with \( n \geq 8 \) exists a Graph \( G \) with \( n \) nodes and 
\[ b(G) = r(G) \]

Proof (for \( n = 8 \)):
Lemma:

For each even \( n \) with \( n \geq 8 \) exists a Graph \( G \) with \( n \) nodes and

\[ b(G) = r(G) \]

Proof (for \( n = 8 \)):
First Results III

Lemma:

For each even \( n \) with \( n \geq 8 \) exists a Graph \( G \) with \( n \) nodes and

\[
b(G) = r(G)
\]

Proof (for \( n = 8 \)):
First Results III

Lemma:
For each even $n$ with $n \geq 8$ exists a Graph $G$ with $n$ nodes and

- $b(G) = r(G)$

Proof (for $n = 8$):

Both broadcasts together are a gossip-algorithm.
Lemma:

For each even $n$ with $n \geq 8$ exists a Graph $G$ with $n$ nodes and

\[ b(G) = r(G) \]

Proof (for $n = 8$):

Both broadcasts together are a gossip-algorithm.
First Results III

Lemma:

For each even \( n \) with \( n \geq 8 \) exists a Graph \( G \) with \( n \) nodes and
\[
b(G) = r(G)
\]

Proof (for \( n = 8 \)):
Lemma:

For each even $n$ with $n \geq 8$ exists a Graph $G$ with $n$ nodes and $b(G) = r(G)$

Proof (for $n = 8$):
Lemma:
For each even $n$ with $n \geq 8$ exists a Graph $G$ with $n$ nodes and $b(G) = r(G)$

Proof (for $n = 8$):

Both broadcasts together are a gossip-algorithm.
Lemma:
For each even $n$ with $n \geq 8$ exists a Graph $G$ with $n$ nodes and $b(G) = r(G)$

Proof (for $n = 8$):

Both broadcasts together are a gossip-algorithm.
Lemma:
For each even $n$ with $n \geq 8$ exists a Graph $G$ with $n$ nodes and

$$b(G) = r(G)$$

Proof (for $n = 8$):

Both broadcasts together are a gossip-algorithm.
Lemma:
For each even $n$ with $n \geq 8$ exists a Graph $G$ with $n$ nodes and $b(G) = r(G)$

Proof (for $n = 8$):
First Results III

Lemma:
For each even $n$ with $n \geq 8$ exists a Graph $G$ with $n$ nodes and $b(G) = r(G)$

Proof (for $n = 8$):

Both broadcasts together are a gossip-algorithm.
Lemma:

For each even $n$ with $n \geq 8$ exists a Graph $G$ with $n$ nodes and

$$b(G) = r(G)$$

Proof (for $n = 8$):
Lemma:

For each even $n$ with $n \geq 8$ exists a Graph $G$ with $n$ nodes and

$$b(G) = r(G)$$

Proof (for $n = 8$):

Both broadcasts together are a gossip-algorithm.
Lemma:

For each even $n$ with $n \geq 8$ exists a Graph $G$ with $n$ nodes and

$$b(G) = r(G)$$

Proof (for $n = 8$):

Both broadcasts together are a gossip-algorithm.
Lemma:

For each even $n$ with $n \geq 8$ exists a Graph $G$ with $n$ nodes and

\[ b(G) = r(G) \]

Proof (for $n = 8$):

Both broadcasts together are a gossip-algorithm.
First Results IV

- \( \text{rad}(G) \leq \text{minb}(G) \).
- \( \text{rad}(G) \leq \text{diam}(G) \leq b(G) \).

Let \( G = (V, E) \) and \( H = (V, F) \) with \( F \subseteq E \). Then we have:

- \( b(G) \leq b(H) \).
- \( \text{minb}(G) \leq \text{minb}(H) \).
- \( r(G) \leq r(H) \).
- \( r_2(G) \leq r_2(H) \).

- \( \text{minb}(G) \leq (\text{deg}(G) - 1) \cdot \text{rad}(G) + 1 \).
- \( b(G) \leq (\text{deg}(G) - 1) \cdot \text{diam}(G) + 1 \).
- \( b(G) \leq \text{deg}(G) \cdot \text{rad}(G) \).
- \( r(G) \leq 2(\text{deg}(G) - 1) \cdot \text{rad}(G) + 2 \)
- \( r_2(G) \leq 2(\text{deg}(G) - 1) \cdot \text{rad}(G) + 1 \)

\[
\begin{align*}
diam(G) &= \max \{ \text{dist}(u, v) \mid u, v \in V \} \\
\text{rad}(v, G) &= \max \{ \text{dist}(v, x) \mid x \in V \} \\
\text{rad}(G) &= \min \{ \text{rad}(v, G) \mid v \in V \}
\end{align*}
\]
First Results IV

- \( \text{rad}(G) \leq \text{minb}(G) \).
- \( \text{rad}(G) \leq \text{diam}(G) \leq b(G) \).
- Let \( G = (V, E) \) and \( H = (V, F) \) with \( F \subset E \). Then we have:
  - \( b(G) \leq b(H) \).
  - \( \text{minb}(G) \leq \text{minb}(H) \).
  - \( r(G) \leq r(H) \).
  - \( r_2(G) \leq r_2(H) \).

\[
\begin{align*}
\text{diam}(G) &= \max \{ \text{dist}(u, v) \mid u, v \in V \} \\
\text{rad}(v, G) &= \max \{ \text{dist}(v, x) \mid x \in V \} \\
\text{rad}(G) &= \min \{ \text{rad}(v, G) \mid v \in V \}
\end{align*}
\]
First Results IV

- \( \text{rad}(G) \leq \text{minb}(G) \).
- \( \text{rad}(G) \leq \text{diam}(G) \leq b(G) \).
- Let \( G = (V, E) \) and \( H = (V, F) \) with \( F \subset E \). Then we have:
  - \( b(G) \leq b(H) \).
  - \( \text{minb}(G) \leq \text{minb}(H) \).
  - \( r(G) \leq r(H) \).
  - \( r_2(G) \leq r_2(H) \).

- \( \text{minb}(G) \leq (\text{deg}(G) - 1) \cdot \text{rad}(G) + 1 \).
- \( b(G) \leq (\text{deg}(G) - 1) \cdot \text{diam}(G) + 1 \).
- \( b(G) \leq \text{deg}(G) \cdot \text{rad}(G) \).
- \( r(G) \leq 2(\text{deg}(G) - 1) \cdot \text{rad}(G) + 2 \).
- \( r_2(G) \leq 2(\text{deg}(G) - 1) \cdot \text{rad}(G) + 1 \).
First Results IV

- \( \text{rad}(G) \leq \min b(G) \).
- \( \text{rad}(G) \leq \text{diam}(G) \leq b(G) \).
- Let \( G = (V, E) \) and \( H = (V, F) \) with \( F \subseteq E \). Then we have:
  - \( b(G) \leq b(H) \).
  - \( \min b(G) \leq \min b(H) \).
  - \( r(G) \leq r(H) \).
  - \( r_2(G) \leq r_2(H) \).

\[
\begin{align*}
\text{diam}(G) &= \max \{ \text{dist}(u, v) \mid u, v \in V \} \\
\text{rad}(v, G) &= \max \{ \text{dist}(v, x) \mid x \in V \} \\
\text{rad}(G) &= \min \{ \text{rad}(v, G) \mid v \in V \}
\end{align*}
\]

- \( \min b(G) \leq (\deg(G) - 1) \cdot \text{rad}(G) + 1 \).
- \( b(G) \leq (\deg(G) - 1) \cdot \text{diam}(G) + 1 \).
- \( b(G) \leq \deg(G) \cdot \text{rad}(G) \).
- \( r(G) \leq 2(\deg(G) - 1) \cdot \text{rad}(G) + 2 \)
- \( r_2(G) \leq 2(\deg(G) - 1) \cdot \text{rad}(G) + 1 \)
First Results IV

- $\text{rad}(G) \leq \min b(G)$.
- $\text{rad}(G) \leq \text{diam}(G) \leq b(G)$.
- Let $G = (V, E)$ and $H = (V, F)$ with $F \subseteq E$. Then we have:
  - $b(G) \leq b(H)$.
  - $\min b(G) \leq \min b(H)$.
  - $r(G) \leq r(H)$.
  - $r_2(G) \leq r_2(H)$.
- $\min b(G) \leq (\text{deg}(G) - 1) \cdot \text{rad}(G) + 1$.
- $b(G) \leq (\text{deg}(G) - 1) \cdot \text{diam}(G) + 1$.
- $b(G) \leq \text{deg}(G) \cdot \text{rad}(G)$.
- $r(G) \leq 2(\text{deg}(G) - 1) \cdot \text{rad}(G) + 2$.
- $r_2(G) \leq 2(\text{deg}(G) - 1) \cdot \text{rad}(G) + 1$.

\begin{align*}
\text{diam}(G) &= \max \{\text{dist}(u, v) \mid u, v \in V\} \\
\text{rad}(v, G) &= \max \{\text{dist}(v, x) \mid x \in V\} \\
\text{rad}(G) &= \min \{\text{rad}(v, G) \mid v \in V\} \\
\end{align*}
First Results IV

- \( \text{rad}(G) \leq \text{minb}(G) \).
- \( \text{rad}(G) \leq \text{diam}(G) \leq b(G) \).

Let \( G = (V, E) \) and \( H = (V, F) \) with \( F \subseteq E \). Then we have:

- \( b(G) \leq b(H) \).
- \( \text{minb}(G) \leq \text{minb}(H) \).
- \( r(G) \leq r(H) \).
- \( r_2(G) \leq r_2(H) \).

- \( \text{minb}(G) \leq (\text{deg}(G) - 1) \cdot \text{rad}(G) + 1 \).
- \( b(G) \leq (\text{deg}(G) - 1) \cdot \text{diam}(G) + 1 \).
- \( b(G) \leq \text{deg}(G) \cdot \text{rad}(G) \).
- \( r(G) \leq 2(\text{deg}(G) - 1) \cdot \text{rad}(G) + 2 \).
- \( r_2(G) \leq 2(\text{deg}(G) - 1) \cdot \text{rad}(G) + 1 \).
First Results IV

- \( \text{rad}(G) \leq \text{minb}(G) \).
- \( \text{rad}(G) \leq \text{diam}(G) \leq b(G) \).

Let \( G = (V, E) \) and \( H = (V, F) \) with \( F \subseteq E \). Then we have:
  - \( b(G) \leq b(H) \).
  - \( \text{minb}(G) \leq \text{minb}(H) \).
  - \( r(G) \leq r(H) \).
  - \( r_2(G) \leq r_2(H) \).

- \( \text{minb}(G) \leq (\text{deg}(G) - 1) \cdot \text{rad}(G) + 1 \).
- \( b(G) \leq (\text{deg}(G) - 1) \cdot \text{diam}(G) + 1 \).
- \( b(G) \leq \text{deg}(G) \cdot \text{rad}(G) \).
- \( r(G) \leq 2(\text{deg}(G) - 1) \cdot \text{rad}(G) + 2 \)
- \( r_2(G) \leq 2(\text{deg}(G) - 1) \cdot \text{rad}(G) + 1 \)
First Results IV

\( \text{rad}(G) \leq \minb(G). \)

\( \text{rad}(G) \leq \text{diam}(G) \leq b(G). \)

Let \( G = (V, E) \) and \( H = (V, F) \) with \( F \subset E \). Then we have:

\( b(G) \leq b(H). \)

\( \minb(G) \leq \minb(H). \)

\( r(G) \leq r(H). \)

\( r_2(G) \leq r_2(H). \)

\( \minb(G) \leq (\deg(G) - 1) \cdot \text{rad}(G) + 1. \)

\( b(G) \leq (\deg(G) - 1) \cdot \text{diam}(G) + 1. \)

\( b(G) \leq \deg(G) \cdot \text{rad}(G). \)

\( r(G) \leq 2(\deg(G) - 1) \cdot \text{rad}(G) + 2 \)

\( r_2(G) \leq 2(\deg(G) - 1) \cdot \text{rad}(G) + 1 \)
First Results IV

\[
\begin{align*}
\text{rad}(G) & \leq \text{minb}(G). \\
\text{rad}(G) & \leq \text{diam}(G) \leq b(G). \\
\text{Let } G = (V, E) \text{ and } H = (V, F) \text{ with } F \subset E. \text{ Then we have:} \\
\quad & b(G) \leq b(H). \\
\quad & \text{minb}(G) \leq \text{minb}(H). \\
\quad & r(G) \leq r(H). \\
\quad & r_2(G) \leq r_2(H). \\
\quad & \text{minb}(G) \leq (\text{deg}(G) - 1) \cdot \text{rad}(G) + 1. \\
\quad & b(G) \leq (\text{deg}(G) - 1) \cdot \text{diam}(G) + 1. \\
\quad & b(G) \leq \text{deg}(G) \cdot \text{rad}(G). \\
\quad & r(G) \leq 2(\text{deg}(G) - 1) \cdot \text{rad}(G) + 2 \\
\quad & r_2(G) \leq 2(\text{deg}(G) - 1) \cdot \text{rad}(G) + 1 \\
\end{align*}
\]
First Results IV

- \( \text{rad}(G) \leq \text{minb}(G) \).
- \( \text{rad}(G) \leq \text{diam}(G) \leq b(G) \).

Let \( G = (V, E) \) and \( H = (V, F) \) with \( F \subseteq E \). Then we have:
  - \( b(G) \leq b(H) \).
  - \( \text{minb}(G) \leq \text{minb}(H) \).
  - \( r(G) \leq r(H) \).
  - \( r_2(G) \leq r_2(H) \).

- \( \text{minb}(G) \leq (\text{deg}(G) - 1) \cdot \text{rad}(G) + 1 \).
- \( b(G) \leq (\text{deg}(G) - 1) \cdot \text{diam}(G) + 1 \).
- \( b(G) \leq \text{deg}(G) \cdot \text{rad}(G) \).
- \( r(G) \leq 2(\text{deg}(G) - 1) \cdot \text{rad}(G) + 2 \)
- \( r_2(G) \leq 2(\text{deg}(G) - 1) \cdot \text{rad}(G) + 1 \)
First Results IV

- \( \text{rad}(G) \leq \text{minb}(G) \).
- \( \text{rad}(G) \leq \text{diam}(G) \leq b(G) \).
- Let \( G = (V, E) \) and \( H = (V, F) \) with \( F \subseteq E \). Then we have:
  - \( b(G) \leq b(H) \).
  - \( \text{minb}(G) \leq \text{minb}(H) \).
  - \( r(G) \leq r(H) \).
  - \( r_2(G) \leq r_2(H) \).
- \( \text{minb}(G) \leq (\text{deg}(G) - 1) \cdot \text{rad}(G) + 1 \).
- \( b(G) \leq (\text{deg}(G) - 1) \cdot \text{diam}(G) + 1 \).
- \( b(G) \leq \text{deg}(G) \cdot \text{rad}(G) \).
- \( r(G) \leq 2(\text{deg}(G) - 1) \cdot \text{rad}(G) + 2 \).
- \( r_2(G) \leq 2(\text{deg}(G) - 1) \cdot \text{rad}(G) + 1 \).
First Results IV

- \( \text{rad}(G) \leq \min b(G) \).  
- \( \text{rad}(G) \leq \text{diam}(G) \leq b(G) \).

Let \( G = (V, E) \) and \( H = (V, F) \) with \( F \subset E \). Then we have:

- \( b(G) \leq b(H) \).
- \( \min b(G) \leq \min b(H) \).
- \( r(G) \leq r(H) \).
- \( r_2(G) \leq r_2(H) \).

- \( \min b(G) \leq (\deg(G) - 1) \cdot \text{rad}(G) + 1 \).
- \( b(G) \leq (\deg(G) - 1) \cdot \text{diam}(G) + 1 \).
- \( b(G) \leq \deg(G) \cdot \text{rad}(G) \).
- \( r(G) \leq 2(\deg(G) - 1) \cdot \text{rad}(G) + 2 \).
- \( r_2(G) \leq 2(\deg(G) - 1) \cdot \text{rad}(G) + 1 \).

\[
\begin{align*}
\text{diam}(G) &= \max \{ \text{dist}(u, v) \mid u, v \in V \} \\
\text{rad}(v, G) &= \max \{ \text{dist}(v, x) \mid x \in V \} \\
\text{rad}(G) &= \min \{ \text{rad}(v, G) \mid v \in V \}
\end{align*}
\]
First Results IV

- \( \text{rad}(G) \leq \text{minb}(G) \).
- \( \text{rad}(G) \leq \text{diam}(G) \leq b(G) \).
- Let \( G = (V, E) \) and \( H = (V, F) \) with \( F \subseteq E \). Then we have:
  - \( b(G) \leq b(H) \).
  - \( \text{minb}(G) \leq \text{minb}(H) \).
  - \( r(G) \leq r(H) \).
  - \( r_2(G) \leq r_2(H) \).

- \( \text{minb}(G) \leq (\text{deg}(G) - 1) \cdot \text{rad}(G) + 1 \).
- \( b(G) \leq (\text{deg}(G) - 1) \cdot \text{diam}(G) + 1 \).
- \( b(G) \leq \text{deg}(G) \cdot \text{rad}(G) \).
- \( r(G) \leq 2(\text{deg}(G) - 1) \cdot \text{rad}(G) + 2 \).
- \( r_2(G) \leq 2(\text{deg}(G) - 1) \cdot \text{rad}(G) + 1 \).

\[
\begin{align*}
\text{diam}(G) & = \max \{ \text{dist}(u, v) \mid u, v \in V \} \\
\text{rad}(v, G) & = \max \{ \text{dist}(v, x) \mid x \in V \} \\
\text{rad}(G) & = \min \{ \text{rad}(v, G) \mid v \in V \}
\end{align*}
\]
Lemma

Let $G = (V, E)$ be a graph with $n$ nodes. Then we have:

- $b(G) \geq \min b(G) \geq \lceil \log n \rceil$

Proof:

- Let $A(t)$ be the number of informed nodes after $t$ rounds.
- $A(0) = 1$
- $A(t + 1) \leq 2 \cdot A(t)$
- $A(t) \leq 2^t$
- At the end $2^t \geq n$ must hold.
Lower Bound

Lemma

Let $G = (V, E)$ be a graph with $n$ nodes. Then we have:

- $b(G) \geq \min b(G) \geq \lceil \log n \rceil$

Proof:

- Let $A(t)$ be the number of informed nodes after $t$ rounds.
- $A(0) = 1$
- $A(t + 1) \leq 2 \cdot A(t)$
- $A(t) \leq 2^t$
- At the end $2^t \geq n$ must hold.
Lemma

Let $G = (V, E)$ be a graph with $n$ nodes. Then we have:

- $b(G) \geq \min b(G) \geq \lceil \log n \rceil$

Proof:

- Let $A(t)$ be the number of informed nodes after $t$ rounds.
- $A(0) = 1$
- $A(t + 1) \leq 2 \cdot A(t)$
- $A(t) \leq 2^t$
- At the end $2^t \geq n$ must hold.
**Lemma**

Let $G = (V, E)$ be a graph with $n$ nodes. Then we have:

- $b(G) \geq \min b(G) \geq \lceil \log n \rceil$

**Proof:**

- Let $A(t)$ be the number of informed nodes after $t$ rounds.
- $A(0) = 1$
- $A(t + 1) \leq 2 \cdot A(t)$
- $A(t) \leq 2^t$
- At the end $2^t \geq n$ must hold.
Lower Bound

Lemma

Let $G = (V, E)$ be a graph with $n$ nodes. Then we have:

- $b(G) \geq \min b(G) \geq \lceil \log n \rceil$

Proof:

- Let $A(t)$ be the number of informed nodes after $t$ rounds.
- $A(0) = 1$
- $A(t + 1) \leq 2 \cdot A(t)$
- $A(t) \leq 2^t$
- At the end $2^t \geq n$ must hold.
Lower Bound

Lemma

Let $G = (V, E)$ be a graph with $n$ nodes. Then we have:

- $b(G) \geq \min b(G) \geq \lceil \log n \rceil$

Proof:

- Let $A(t)$ be the number of informed nodes after $t$ rounds.
- $A(0) = 1$
- $A(t + 1) \leq 2 \cdot A(t)$
- $A(t) \leq 2^t$
- At the end $2^t \geq n$ must hold.
Lemma

Let \( G = (V, E) \) be a graph with \( n \) nodes. Then we have:

\[
\begin{align*}
\text{Lemma:} & \quad b(G) \geq \min b(G) \geq \lceil \log n \rceil \\
\text{Proof:} & \quad \text{Let } A(t) \text{ be the number of informed nodes after } t \text{ rounds.} \\
& \quad A(0) = 1 \text{ } \quad \text{\textbullet} \\
& \quad A(t + 1) \leq 2 \cdot A(t) \text{ } \quad \text{\textbullet} \\
& \quad A(t) \leq 2^t \text{ } \quad \text{\textbullet} \\
& \quad \text{At the end } 2^t \geq n \text{ must hold.}
\end{align*}
\]
Optimal Broadcast-Tree

Each informed node has to send in each round the information to a non-informed node:

A tree $T_i$ is a broadcast-tree, iff

- the root of $T_i$ has $i$ successors $v_0, v_1, \cdots, v_{i-1}$ and
- $v_j$ is the root of a $T_j$. 
Optimal Broadcast-Tree

Each informed node has to send in each round the information to a non-informed node:

A tree $T_i$ is a broadcast-tree, iff
- the root of $T_i$ has $i$ successors $v_0, v_1, \cdots, v_{i-1}$ and
- $v_j$ is the root of a $T_j$. 
Optimal Broadcast-Tree

Each informed node has to send in each round the information to a non-informed node:

A tree $T_i$ is a broadcast-tree, iff
- the root of $T_i$ has $i$ successors $v_0, v_1, \cdots, v_{i-1}$ and
- $v_j$ is the root of a $T_j$. 
Optimal Broadcast-Tree

Each informed node has to send in each round the information to a non-informed node:

A tree $T_i$ is a broadcast-tree, iff

- the root of $T_i$ has $i$ successors $v_0, v_1, \cdots, v_{i-1}$ and
- $v_j$ is the root of a $T_j$. 
**Optimal Broadcast-Tree**

Each informed node has to send in each round the information to a non-informed node:

A tree $T_i$ is a broadcast-tree, iff

- the root of $T_i$ has $i$ successors $v_0, v_1, \cdots, v_{i-1}$ and
- $v_j$ is the root of a $T_j$. 
Optimal Broadcast-Tree

Each informed node has to send in each round the information to a non-informed node:

A tree $T_i$ is a broadcast-tree, iff

- the root of $T_i$ has $i$ successors $v_0, v_1, \cdots, v_{i-1}$ and
- $v_j$ is the root of a $T_j$. 
Optimal Broadcast-Tree

Each informed node has to send in each round the information to a non-informed node:

A tree $T_i$ is a broadcast-tree, iff
- the root of $T_i$ has $i$ successors $v_0, v_1, \cdots, v_{i-1}$ and
- $v_j$ is the root of a $T_j$. 
Optimal Broadcast-Tree

Each informed node has to send in each round the information to a non-informed node:

A tree $T_i$ is a broadcast-tree, iff

- the root of $T_i$ has $i$ successors $v_0, v_1, \ldots, v_{i-1}$ and
- $v_j$ is the root of a $T_j$.  


Optimal Broadcast-Tree

Each informed node has to send in each round the information to a non-informed node:

A tree $T_i$ is a broadcast-tree, iff
- the root of $T_i$ has $i$ successors $v_0, v_1, \cdots, v_{i-1}$ and
- $v_j$ is the root of a $T_j$. 
Optimal Broadcast-Tree

Each informed node has to send in each round the information to a non-informed node:

A tree $T_i$ is a broadcast-tree, iff

- the root of $T_i$ has $i$ successors $v_0, v_1, \ldots, v_{i-1}$ and
- $v_j$ is the root of a $T_j$. 
Optimal Broadcast-Tree

Each informed node has to send in each round the information to a non-informed node:

- A tree $T_i$ is a broadcast-tree, iff
  - the root of $T_i$ has $i$ successors $v_0, v_1, \ldots, v_{i-1}$ and
  - $v_j$ is the root of a $T_j$. 
Optimal Broadcast-Tree

Each informed node has to send in each round the information to a non-informed node:

A tree $T_i$ is a broadcast-tree, iff

- the root of $T_i$ has $i$ successors $v_0, v_1, \cdots, v_{i-1}$ and
- $v_j$ is the root of a $T_j$.
Optimal Broadcast-Tree

Each informed node has to send in each round the information to a non-informed node:

A tree $T_i$ is a broadcast-tree, iff

- the root of $T_i$ has $i$ successors $v_0, v_1, \cdots, v_{i-1}$ and
- $v_j$ is the root of a $T_j$. 
First Results

Lemma

We have:

1. $\min b(K(n)) = b(K(n)) = \lceil \log n \rceil$ and
2. $\min b(HQ(m)) = b(HQ(m)) = m$.

Proof ($K(n)$):

for $t = 1$ to $\lceil \log n \rceil$ do
    for all $i \in \{0, 1, \cdots, 2^{t-1} - 1\}$ do in parallel
        if $i + 2^{t-1} \leq n$ then
            $i$ sends to $i + 2^{t-1}$

Proof ($HQ(m)$):

for $i = 1$ to $m$ do
    for all $a_1, a_2, \cdots, a_{i-1} \in \{0, 1\}$ do in parallel
        $a_1 a_2 \cdots a_{i-1} 0 0 \cdots 0$ sends to $a_1 a_2 \cdots a_{i-1} 1 0 \cdots 0$
First Results

Lemma

We have:

- $\min b(K(n)) = b(K(n)) = \lceil \log n \rceil$ and
- $\min b(HQ(m)) = b(HQ(m)) = m$.

Proof ($K(n)$):

for $t = 1$ to $\lceil \log n \rceil$ do

for all $i \in \{0, 1, \ldots, 2^{t-1} - 1\}$ do in parallel

if $i + 2^{t-1} \leq n$ then

$i$ sends to $i + 2^{t-1}$

Proof ($HQ(m)$):

for $i = 1$ to $m$ do

for all $a_1, a_2, \ldots, a_{i-1} \in \{0, 1\}$ do in parallel

$a_1a_2\cdots a_{i-1}00\cdots0$ sends to $a_1a_2\cdots a_{i-1}10\cdots0$
First Results

Lemma

We have:

- \( \min b(K(n)) = b(K(n)) = \lceil \log n \rceil \) and
- \( \min b(HQ(m)) = b(HQ(m)) = m \).

Proof \((K(n))\):

for \( t = 1 \) to \( \lceil \log n \rceil \) do

for all \( i \in \{0, 1, \cdots, 2^{t-1} - 1\} \) do in parallel

if \( i + 2^{t-1} \leq n \) then

\( i \) sends to \( i + 2^{t-1} \)

Proof \((HQ(m))\):

for \( i = 1 \) to \( m \) do

for all \( a_1, a_2, \cdots, a_{i-1} \in \{0, 1\} \) do in parallel

\( a_1a_2\cdots a_{i-1}00\cdots 0 \) sends to \( a_1a_2\cdots a_{i-1}10\cdots 0 \)
The special Broadcast-Problem is:

- Given: $G = (V, E)$, $v \in V$ and $k \in \mathbb{N}$.
- Question: Does $b(v, G) \leq k$ hold?

The Broadcast-Problem is:

- Given: $G = (V, E)$ and $k \in \mathbb{N}$.
- Question: Does $b(G) \leq k$ hold?
Complexity

Definition:
The special Broadcast-Problem is:
- Given: $G = (V, E)$, $v \in V$ and $k \in \mathbb{N}$.
- Question: Does $b(v, G) \leq k$ hold?

Definition:
The Broadcast-Problem is:
- Given: $G = (V, E)$ and $k \in \mathbb{N}$.
- Question: Does $b(G) \leq k$ hold?
Theorem:
The special Broadcast-Problem on trees is in $P$.

- The algorithm computes recursively the broadcast-time from a node (which we consider as root) in its subtree.
- For the leafs is this time 0.
- When all broadcast-times are computed for all successors of the root, we sort these times.
- After this we may compute the order of subtrees of the root in which we forward the information from the root.
- Example: 5 subtrees have broadcast-times 10, 10, 9, 9, 7. Then we inform these subtrees in the same order. The total broadcast-time from the root is $\max(10 + 1, 10 + 2, 9 + 3, 9 + 4, 7 + 5) = 13$.

Theorem:
The Broadcast-Problem on trees is in $P$. 
Complexity

**Theorem:**
The special Broadcast-Problem on trees is in $P$.

- The algorithm computes recursively the broadcast-time from a node (which we consider as root) in its subtree.
- For the leafs is this time 0.
- When all broadcast-times are computed for all successors of the root, we sort these times.
- After this we may compute the order of subtrees of the root in which we forward the information from the root.
- Example: 5 subtrees have broadcast-times 10, 10, 9, 9, 7. Then we inform these subtrees in the same order. The total broadcast-time from the root is $\max(10 + 1, 10 + 2, 9 + 3, 9 + 4, 7 + 5) = 13$.

**Theorem:**
The Broadcast-Problem on trees is in $P$.
Theorem:
The special Broadcast-Problem on trees is in $\mathcal{P}$.

- The algorithm computes recursively the broadcast-time from a node (which we consider as root) in its subtree.
- For the leaves is this time 0.
- When all broadcast-times are computed for all successors of the root, we sort these times.
- After this we may compute the order of subtrees of the root in which we forward the information from the root.
- Example: 5 subtrees have broadcast-times 10, 10, 9, 9, 7. Then we inform these subtrees in the same order. The total broadcast-time from the root is $\max(10 + 1, 10 + 2, 9 + 3, 9 + 4, 7 + 5) = 13$.

Theorem:
The Broadcast-Problem on trees is in $\mathcal{P}$.
Theorem:
The special Broadcast-Problem on trees is in $\mathcal{P}$.

- The algorithm computes recursively the broadcast-time from a node (which we consider as root) in its subtree.
- For the leaves is this time 0.
- When all broadcast-times are computed for all successors of the root, we sort these times.
- After this we may compute the order of subtrees of the root in which we forward the information from the root.
- Example: 5 subtrees have broadcast-times $10, 10, 9, 9, 7$. Then we inform these subtrees in the same order. The total broadcast-time from the root is $\max(10 + 1, 10 + 2, 9 + 3, 9 + 4, 7 + 5) = 13$.

Theorem:
The Broadcast-Problem on trees is in $\mathcal{P}$.
Complexity

Theorem:
The special Broadcast-Problem on trees is in $\mathcal{P}$.

- The algorithm computes recursively the broadcast-time from a node (which we consider as root) in its subtree.
- For the leafs is this time 0.
- When all broadcast-times are computed for all successors of the root, we sort these times.
- After this we may compute the order of subtrees of the root in which we forward the information from the root.

Example: 5 subtrees have broadcast-times 10, 10, 9, 9, 7. Then we inform these subtrees in the same order. The total broadcast-time from the root is $\max(10 + 1, 10 + 2, 9 + 3, 9 + 4, 7 + 5) = 13$.

Theorem:
The Broadcast-Problem on trees is in $\mathcal{P}$.
The special Broadcast-Problem on trees is in \( \mathcal{P} \).

- The algorithm computes recursively the broadcast-time from a node (which we consider as root) in its subtree.
- For the leafs is this time 0.
- When all broadcast-times are computed for all successors of the root, we sort these times.
- After this we may compute the order of subtrees of the root in which we forward the information from the root.
- Example: 5 subtrees have broadcast-times 10, 10, 9, 9, 7. Then we inform these subtrees in the same order. The total broadcast-time from the root is \( \max(10 + 1, 10 + 2, 9 + 3, 9 + 4, 7 + 5) = 13 \).
Theorem:
The special Broadcast-Problem on trees is in \( P \).

- The algorithm computes recursively the broadcast-time from a node (which we consider as root) in its subtree.
- For the leaves is this time 0.
- When all broadcast-times are computed for all successors of the root, we sort these times.
- After this we may compute the order of subtrees of the root in which we forward the information from the root.
- Example: 5 subtrees have broadcast-times 10, 10, 9, 9, 7. Then we inform these subtrees in the same order. The total broadcast-time from the root is \( \max(10 + 1, 10 + 2, 9 + 3, 9 + 4, 7 + 5) = 13 \).

Theorem:
The Broadcast-Problem on trees is in \( P \).
The special Broadcast-Problem on trees is in $\mathcal{P}$.

- The algorithm computes recursively the broadcast-time from a node (which we consider as root) in its subtree.
- For the leafs is this time 0.
- When all broadcast-times are computed for all successors of the root, we sort these times.
- After this we may compute the order of subtrees of the root in which we forward the information from the root.
- Example: 5 subtrees have broadcast-times 10, 10, 9, 9, 7. Then we inform these subtrees in the same order. The total broadcast-time from the root is $\max(10 + 1, 10 + 2, 9 + 3, 9 + 4, 7 + 5) = 13$.

The Broadcast-Problem on trees is in $\mathcal{P}$.
The special Broadcast-Problem is in $NP\overline{C}$.

Proof: simple exercise.

- IF a message from node $v$ has to be send to node $w$ and the remaining time is the same as the distance between $v$ and $w$, then we call this message critical.
- I.e. the messages has to be forwarded towards $w$ without any delay.
- Is the shortest path between $v$ and $w$ unique, then we know precisely the way (times and places) the messages has to traverse towards $w$.
- If there exists an other node $w'$ with: $\text{dist}(v, w) = \text{dist}(v, w') + 1$ and the shortest path towards $w'$ splits from the path from $v$ to $w$, then is the message also critical on this path.
**Theorem:**

The special Broadcast-Problem is in $\mathcal{NP}$.

Proof: simple exercise.

- **IF** a message from node $v$ has to be send to node $w$ and the remaining time is the same as the distance between $v$ and $w$, then we call this message critical.

- **I.e.** the messages has to be forwarded towards $w$ without any delay.

- If the shortest path between $v$ and $w$ unique, then we know precisely the way (times and places) the messages has to traverse towards $w$.

- **If** there exists an other node $w'$ with: $\text{dist}(v, w) = \text{dist}(v, w') + 1$ and the shortest path towards $w'$ splits from the path from $v$ to $w$, then is the message also critical on this path.
The special Broadcast-Problem is in \( \mathcal{NP} \).

**Proof:** simple exercise (if we have the idea).

- **IF** a message from node \( v \) has to be send to node \( w \) and the remaining time is the same as the distance between \( v \) and \( w \), then we call this message critical.

- **I.e.** the messages has to be forwarded towards \( w \) without any delay.

- Is the shortest path between \( v \) and \( w \) unique, then we know precisely the way (times and places) the messages has to traverse towards \( w \).

- If there exists an other node \( w' \) with: \( \text{dist}(v, w) = \text{dist}(v, w') + 1 \) and the shortest path towards \( w' \) splits from the path from \( v \) to \( w \), then is the message also critical on this path.
The special Broadcast-Problem is in $\mathcal{NP}$.

**Proof:** simple exercise.

- **IF** a message from node $v$ has to be send to node $w$ and the remaining time is the same as the distance between $v$ and $w$, then we call this message critical.

- **I.e.** the messages has to be forwarded towards $w$ without any delay.

- **Is** the shortest path between $v$ and $w$ unique, then we know precisely the way (times and places) the messages has to traverse towards $w$.

- **If** there exists an other node $w'$ with: $\text{dist}(v, w) = \text{dist}(v, w') + 1$ and the shortest path towards $w'$ splits from the path from $v$ to $w$, then is the message also critical on this path.
The special Broadcast-Problem is in \( \mathcal{NPC} \).

Proof: simple exercise.

- IF a message from node \( v \) has to be send to node \( w \) and the remaining time is the same as the distance between \( v \) and \( w \), then we call this message critical.

- I.e. the messages has to be forwarded towards \( w \) without any delay.

- Is the shortest path between \( v \) and \( w \) unique, then we know precisely the way (times and places) the messages has to traverse towards \( w \).

- If there exists an other node \( w' \) with: \( \text{dist}(v, w) = \text{dist}(v, w') + 1 \) and the shortest path towards \( w' \) splits from the path from \( v \) to \( w \), then is the message also critical on this path.
Complexity

**Theorem:**

The special Broadcast-Problem is in $NP\overline{C}$.

Proof: simple exercise .

- IF a message from node $v$ has to be send to node $w$ and the remaining time is the same as the distance between $v$ and $w$, then we call this message critical.

- I.e. the messages has to be forwarded towards $w$ without any delay.

- Is the shortest path between $v$ and $w$ unique, then we know precisely the way (times and places) the messages has to traverse towards $w$.

- If there exists an other node $w'$ with: $\text{dist}(v, w) = \text{dist}(v, w') + 1$ and the shortest path towards $w'$ splits from the path from $v$ to $w$, then is the message also critical on this path.
The special broadcast-problem on graphs of degree 3 is in \( \mathcal{NPC} \).

Proof: it is easy to build the above construction with nodes of degree \( \leq 3 \).

The special broadcast-problem on planar graphs of degree 3 is in \( \mathcal{NPC} \).

Idea of proof: The planar 3-SAT is in \( \mathcal{NPC} \). That is the dependency graph between clauses and variables is planar.

Definition:
Let \( \mathcal{F} \) be a boolean formula in \( \text{KNF} \). Let \( V \) be the variables and \( C \) be the clauses. The dependency graph is:

\[
G_{\mathcal{F}} = (V, C, \{\{v, c\} \mid v \text{ is in } c\})
\]
**Theorem:**

The special broadcast-problem on graphs of degree 3 is in \( \mathcal{NPC} \).

Proof: it is easy to build the above construction with nodes of degree \( \leq 3 \).

**Theorem:**

The special broadcast-problem on planar graphs of degree 3 is in \( \mathcal{NPC} \).

Idea of proof: The planar 3-SAT is in \( \mathcal{NPC} \). That is the dependency graph between clauses and variables is planar.

**Definition:**

Let \( \mathcal{F} \) be a boolean formula in KNF. Let \( V \) be the variables and \( C \) be the clauses. The dependency graph is:

\[
G_{\mathcal{F}} = (V, C, \{\{v, c\} \mid v \text{ is in } c\})
\]
Complexity

**Theorem:**
The special broadcast-problem on graphs of degree 3 is in $\mathcal{NP}$.

Proof: it is easy to build the above construction with nodes of degree $\leq 3$.

**Theorem:**
The special broadcast-problem on planar graphs of degree 3 is in $\mathcal{NP}$.

Idea of proof: The planar 3-SAT is in $\mathcal{NP}$. That is the dependency graph between clauses and variables is planar.

**Definition:**
Let $\mathcal{F}$ be a boolean formula in KNF. Let $V$ be the variables and $C$ be the clauses. The dependency graph is:

$$G_{\mathcal{F}} = (V, C, \{\{v, c\} \mid v \text{ is in } c\})$$
Theorem:
The special broadcast-problem on graphs of degree 3 is in NPC.

Proof: it is easy to build the above construction with nodes of degree $\leq 3$.

Theorem:
The special broadcast-problem on planar graphs of degree 3 is in NPC.

Idea of proof: The planar 3-SAT is in NPC. That is the dependency graph between clauses and variables is planar.

Definition:
Let $\mathcal{F}$ be a boolean formula in KNF. Let $V$ be the variables and $C$ be the clauses. The dependency graph is:

$$G_{\mathcal{F}} = (V, C, \{\{v, c\} \mid v \text{ is in } c\})$$
**Theorem:**

The broadcast-problem on planar graphs of degree 3 is in \( \mathcal{NP} \).

**Proof:**

- Extend the above construction, such that there is a unique “hardest” node.
- Add to the above construction a very long path.
- Thus the broadcast from the start node of the long path is the hardest.
Theorem:
The broadcast-problem on planar graphs of degree 3 is in \( \mathcal{NP} \).

Proof:

- Extend the above construction, such that there is a unique “hardest” node.
- Add to the above construction a very long path.
- Thus the broadcast from the start node of the long path is the hardest.
Complexity

**Theorem:**

The broadcast-problem on planar graphs of degree 3 is in $\mathcal{NP}$.

**Proof:**

- Extend the above construction, such that there is a unique “hardest” node.
- Add to the above construction a very long path.
- Thus the broadcast from the start node of the long path is the hardest.
Theorem:
The broadcast-problem on planar graphs of degree 3 is in \( \mathcal{NP} \).

Proof:

- Extend the above construction, such that there is a unique “hardest” node.
- Add to the above construction a very long path.
- Thus the broadcast from the start node of the long path is the hardest.
Complexity

Theorem:
The broadcast-problem on planar graphs of degree 3 is in $\mathcal{NP}$.

Proof:

- Extend the above construction, such that there is a unique “hardest” node.
- Add to the above construction a very long path.
- Thus the broadcast from the start node of the long path is the hardest.
The gossip-problem is:

- Given: $G = (V, E)$ and $k \in \mathbb{N}$.
- Question: Does $r_2(G) \leq k$ hold?

Theorem:
The gossip-problem is in $\mathcal{NP}$.

Proof: Extend the above construction, such that there is a unique “hardest” node.
Definition:
The gossip-problem is:
- Given: $G = (V, E)$ and $k \in \mathbb{N}$.
- Question: Does $r_2(G) \leq k$ hold?

Theorem:
The gossip-problem is in $\mathcal{NP}$.

Proof: Extend the above construction, such that there is a unique “hardest” node.
Definition:

The gossip-problem is:

- Given: $G = (V, E)$ and $k \in \mathbb{N}$.
- Question: Does $r_2(G) \leq k$ hold?

Theorem:

The gossip-problem is in $\mathcal{NP}$.

Proof: Extend the above construction, such that there is a unique “hardest” node.
**Definition:**

The one-way gossip-problem is:

- Given: $G = (V, E)$ and $k \in \mathbb{N}$.
- Question: Does $r(G) \leq k$ hold?

**Theorem:**

The one-way gossip-problem is in $\mathcal{NPC}$.

Proof: Extend the above construction, such that there is a unique “hardest” node.
And prevent the blocking of critical messages.
Definition:

The one-way gossip-problem is:

- Given: \( G = (V, E) \) and \( k \in \mathbb{N} \).
- Question: Does \( r(G) \leq k \) hold?

Theorem:

The one-way gossip-problem is in \( NP \).

Proof: Extend the above construction, such that there is a unique “hardest” node.
And prevent the blocking of critical messages.
Definition:
The one-way gossip-problem is:
- Given: $G = (V, E)$ and $k \in \mathbb{N}$.
- Question: Does $r(G) \leq k$ hold?

Theorem:
The one-way gossip-problem is in $\mathcal{NP}$.

Proof: Extend the above construction, such that there is a unique “hardest” node.
And prevent the blocking of critical messages.
Definition:
The one-way gossip-problem is:

- Given: \( G = (V, E) \) and \( k \in \mathbb{N} \).
- Question: Does \( r(G) \leq k \) hold?

Theorem:
The one-way gossip-problem is in \( \mathcal{NPC} \).

Proof: Extend the above construction, such that there is a unique “hardest” node.
And prevent the blocking of critical messages.
Complexity

**Definition:**

The two-way gossip-problem is:

- Given: $G = (V, E)$ and $k \in \mathbb{N}$.
- Question: Does $r_2(G) \leq k$ hold.

**Definition:**

The one-way gossip-problem is:

- Given: $G = (V, E)$ and $k \in \mathbb{N}$.
- Question: Does $r(G) \leq k$ hold.
Definition:
The two-way gossip-problem is:
- Given: $G = (V, E)$ and $k \in \mathbb{N}$.
- Question: Does $r_2(G) \leq k$ hold.

Definition:
The one-way gossip-problem is:
- Given: $G = (V, E)$ and $k \in \mathbb{N}$.
- Question: Does $r(G) \leq k$ hold.
Complexity

Theorem:

The two-way and one-way gossip-problem on trees is in $\mathcal{P}$

Proof: simple exercise.

Theorem:

The two-way and one-way gossip-problem is in $\mathcal{NP}$

Proof: Same way as the for the broadcast-problem.
Complexity

**Theorem:**

The two-way and one-way gossip-problem on trees is in $\mathcal{P}$

Proof: simple exercise.

**Theorem:**

The two-way and one-way gossip-problem is in $\mathcal{NPC}$

Proof: Same way as the for the broadcast-problem.
Complexity

Theorem:
The two-way and one-way gossip-problem on trees is in \( \mathcal{P} \)

Proof: simple exercise.

Theorem:
The two-way and one-way gossip-problem is in \( \mathcal{NPC} \)

Proof: Same way as the for the broadcast-problem.
Theorem:
The two-way and one-way gossip-problem on trees is in $\mathcal{P}$

Proof: simple exercise.

Theorem:
The two-way and one-way gossip-problem is in $\text{NPC}$

Proof: Same way as the for the broadcast-problem.
Theorem:
Let $n \geq 5$ and $G = (V, E)$ be a graph with $n$ nodes:

- If $\Delta(G) = 3$ holds, we have: $b(G) \geq \min b(G) \geq 1.4404 \log(n) - 3$.
- If $\Delta(G) = 4$ holds, we have: $b(G) \geq \min b(G) \geq 1.1374 \log(n) - 2$.

Proof:

- Let $A$ be a broadcast-algorithm.
- Let $\text{Broad}_i^A(v_0)$ be the set of nodes, which are informed from $v_0$ by $A$ in $i$ rounds.
- Let $\text{Rec}_i^A(v_0) = \text{Broad}_i^A(v_0) \setminus \text{Broad}_{i-1}^A(v_0)$.
- Let $\text{Rec}_0^A(v_0) = \{v_0\}$.
- We have: $|\text{Broad}_i^A(v_0)| = \sum_{s=0}^{i} |\text{Rec}_s^A(v_0)|$. 
Degree of the Nodes

**Theorem:**

Let $n \geq 5$ and $G = (V, E)$ be a graph with $n$ nodes:

- If $\Delta(G) = 3$ holds, we have: $b(G) \geq \min b(G) \geq 1.4404 \log(n) - 3$.
- IF $\Delta(G) = 4$ holds, we have: $b(G) \geq \min b(G) \geq 1.1374 \log(n) - 2$.

**Proof:**

- Let $A$ be a broadcast-algorithm.
- Let $\text{Broad}_i^A(v_0)$ be the set of nodes, which are informed from $v_0$ by $A$ in $i$ rounds.
- Let $\text{Rec}_i^A(v_0) = \text{Broad}_i^A(v_0) \setminus \text{Broad}_{i-1}^A(v_0)$.
- Let $\text{Rec}_0^A(v_0) = \{v_0\}$.
- We have: $|\text{Broad}_i^A(v_0)| = \sum_{s=0}^{i} |\text{Rec}_s^A(v_0)|$. 


Theorem:

Let $n \geq 5$ and $G = (V, E)$ be a graph with $n$ nodes:

- If $\Delta(G) = 3$ holds, we have: $b(G) \geq \min b(G) \geq 1.4404 \log(n) - 3$.
- If $\Delta(G) = 4$ holds, we have: $b(G) \geq \min b(G) \geq 1.1374 \log(n) - 2$.

Proof:

- Let $A$ be a broadcast-algorithm.
- Let $\text{Broad}_i^A(v_0)$ be the set of nodes, which are informed from $v_0$ by $A$ in $i$ rounds.
- Let $\text{Rec}_i^A(v_0) = \text{Broad}_i^A(v_0) \setminus \text{Broad}_{i-1}^A(v_0)$.
- Let $\text{Rec}_0^A(v_0) = \{v_0\}$.
- We have: $|\text{Broad}_i^A(v_0)| = \sum_{s=0}^{i} |\text{Rec}_s^A(v_0)|$. 
Degree of the Nodes

Theorem:
Let $n \geq 5$ and $G = (V,E)$ be a graph with $n$ nodes:
- If $\Delta(G) = 3$ holds, we have: $b(G) \geq \min b(G) \geq 1.4404 \log(n) - 3$.
- If $\Delta(G) = 4$ holds, we have: $b(G) \geq \min b(G) \geq 1.1374 \log(n) - 2$.

Proof:
- Let $A$ be a broadcast-algorithm.
- Let $\text{Broad}_i^A(v_0)$ be the set of nodes, which are informed from $v_0$ by $A$ in $i$ rounds.
- Let $\text{Rec}_i^A(v_0) = \text{Broad}_i^A(v_0) \setminus \text{Broad}_{i-1}^A(v_0)$.
- Let $\text{Rec}_0^A(v_0) = \{v_0\}$.
- We have: $|\text{Broad}_i^A(v_0)| = \sum_{s=0}^{i} |\text{Rec}_s^A(v_0)|$. 
Degree of the Nodes

Theorem:

Let \( n \geq 5 \) and \( G = (V, E) \) be a graph with \( n \) nodes:

- If \( \Delta(G) = 3 \) holds, we have: \( b(G) \geq \min b(G) \geq 1.4404 \log(n) - 3. \)
- If \( \Delta(G) = 4 \) holds, we have: \( b(G) \geq \min b(G) \geq 1.1374 \log(n) - 2. \)

Proof:

- Let \( A \) be a broadcast-algorithm.
- Let \( \text{Broad}_i^A(v_0) \) be the set of nodes, which are informed from \( v_0 \) by \( A \) in \( i \) rounds.
- Let \( \text{Rec}_i^A(v_0) = \text{Broad}_i^A(v_0) \setminus \text{Broad}_{i-1}^A(v_0) \).
- Let \( \text{Rec}_0^A(v_0) = \{v_0\} \).
- We have: \( |\text{Broad}_i^A(v_0)| = \sum_{s=0}^{i} |\text{Rec}_s^A(v_0)|. \)
Degree of the Nodes

**Theorem:**

Let $n \geq 5$ and $G = (V, E)$ be a graph with $n$ nodes:

- If $\Delta(G) = 3$ holds, we have: $b(G) \geq \min b(G) \geq 1.4404 \log(n) - 3$.
- If $\Delta(G) = 4$ holds, we have: $b(G) \geq \min b(G) \geq 1.1374 \log(n) - 2$.

**Proof:**

- Let $A$ be a broadcast-algorithm.
- Let $\text{Broad}^A_i(v_0)$ be the set of nodes, which are informed from $v_0$ by $A$ in $i$ rounds.
- Let $\text{Rec}^A_i(v_0) = \text{Broad}^A_i(v_0) \setminus \text{Broad}^A_{i-1}(v_0)$.
- Let $\text{Rec}^A_0(v_0) = \{v_0\}$.
- We have: $|\text{Broad}^A_i(v_0)| = \sum_{s=0}^{i} |\text{Rec}^A_s(v_0)|$. 
Degree of the Nodes

Theorem:

Let \( n \geq 5 \) and \( G = (V, E) \) be a graph with \( n \) nodes:

- If \( \Delta(G) = 3 \) holds, we have: \( b(G) \geq \min b(G) \geq 1.4404 \log(n) - 3 \).
- If \( \Delta(G) = 4 \) holds, we have: \( b(G) \geq \min b(G) \geq 1.1374 \log(n) - 2 \).

Proof:

- Let \( A \) be a broadcast-algorithm.
- Let \( \text{Broad}_i^A(v_0) \) be the set of nodes, which are informed from \( v_0 \) by \( A \) in \( i \) rounds.
- Let \( \text{Rec}_i^A(v_0) = \text{Broad}_i^A(v_0) \setminus \text{Broad}_{i-1}^A(v_0) \).
- Let \( \text{Rec}_0^A(v_0) = \{v_0\} \).
- We have: \( |\text{Broad}_i^A(v_0)| = \sum_{s=0}^{i} |\text{Rec}_s^A(v_0)| \).
Degree of the Nodes

Theorem:

Let $n \geq 5$ and $G = (V, E)$ be a graph with $n$ nodes:

- If $\Delta(G) = 3$ holds, we have: $b(G) \geq \min b(G) \geq 1.4404 \log(n) - 3$.
- If $\Delta(G) = 4$ holds, we have: $b(G) \geq \min b(G) \geq 1.1374 \log(n) - 2$.

Proof:

- Let $A$ be a broadcast-algorithm.
- Let $\text{Broad}_i^A(v_0)$ be the set of nodes, which are informed from $v_0$ by $A$ in $i$ rounds.
- Let $\text{Rec}_i^A(v_0) = \text{Broad}_i^A(v_0) \setminus \text{Broad}_{i-1}^A(v_0)$.
- Let $\text{Rec}_0^A(v_0) = \{v_0\}$.
- We have: $|\text{Broad}_i^A(v_0)| = \sum_{s=0}^{i} |\text{Rec}_s^A(v_0)|$.
Theorem:

Let $n \geq 5$ and $G = (V, E)$ be a graph with $n$ nodes:

- If $\Delta(G) = 3$ holds, we have: $b(G) \geq \min b(G) \geq 1.4404 \log(n) - 3$.
- If $\Delta(G) = 4$ holds, we have: $b(G) \geq \min b(G) \geq 1.1374 \log(n) - 2$.

Proof:

- Let $A$ be a broadcast-algorithm.
- Let $\text{Broad}_i^A(v_0)$ be the set of nodes, which are informed from $v_0$ by $A$ in $i$ rounds.
- Let $\text{Rec}_i^A(v_0) = \text{Broad}_i^A(v_0) \setminus \text{Broad}_{i-1}^A(v_0)$.
- Let $\text{Rec}_0^A(v_0) = \{v_0\}$.
- We have: $|\text{Broad}_i^A(v_0)| = \sum_{s=0}^{i} |\text{Rec}_s^A(v_0)|$. 
Building the Idea

We consider here only the case $\Delta(G) = 3$. The case $\Delta(G) = 4$ is similar.

- The initial node may send at most three times.
- The initial node sends only in rounds 1, 2, 3.
- Any other nodes will be informed at time $t$ via an edge $e$.
- No further node may be informed via $e$.
- Thus any other node may send at most two times.
- If a node $v$ is informed in round $t$ by $w$, then did $w$ receive the information at round $t - 1$ or $t - 2$.
- Thus the number of newly informed nodes in round $t > 3$, is at most the number of nodes which got informed in rounds $t - 1$ and $t - 2$. 
Building the Idea

We consider here only the case $\Delta(G) = 3$. The case $\Delta(G) = 4$ is similar.

- The initial node may send at most three times.
- The initial node sends only in rounds 1, 2, 3.
- Any other nodes will be informed at time $t$ via an edge $e$.
- No further node may be informed via $e$.
- Thus any other node may send at most two times.
- If a node $v$ is informed in round $t$ by $w$, then did $w$ receive the information at round $t - 1$ or $t - 2$.
- Thus the number of newly informed nodes in round $t > 3$, is at most the number of nodes which got informed in rounds $t - 1$ and $t - 2$. 

We consider here only the case $\Delta(G) = 3$. The case $\Delta(G) = 4$ is similar.

- The initial node may send at most three times.
- The initial node sends only in rounds 1, 2, 3.
- Any other nodes will be informed at time $t$ via an edge $e$.
- No further node may be informed via $e$.
- Thus any other node may send at most two times.
- If a node $v$ is informed in round $t$ by $w$, then did $w$ receive the information at round $t - 1$ or $t - 2$.
- Thus the number of newly informed nodes in round $t > 3$, is at most the number of nodes which got informed in rounds $t - 1$ and $t - 2$. 

Building the Idea
Building the Idea

We consider here only the case $\Delta(G) = 3$. The case $\Delta(G) = 4$ is similar.

- The initial node may send at most three times.
- The initial node sends only in rounds 1, 2, 3.
- Any other nodes will be informed at time $t$ via an edge $e$.
- No further node may be informed via $e$.

- Thus any other node may send at most two times.
- If a node $v$ is informed in round $t$ by $w$, then did $w$ receive the information at round $t - 1$ or $t - 2$.
- Thus the number of newly informed nodes in round $t > 3$, is at most the number of nodes which got informed in rounds $t - 1$ and $t - 2$. 
Building the Idea

We consider here only the case $\Delta(G) = 3$. The case $\Delta(G) = 4$ is similar.

- The initial node may send at most three times.
- The initial node sends only in rounds 1, 2, 3.
- Any other nodes will be informed at time $t$ via an edge $e$.
- No further node may be informed via $e$.

Thus any other node may send at most two times.

- If a node $v$ is informed in round $t$ by $w$, then did $w$ receive the information at round $t - 1$ or $t - 2$.

Thus the number of newly informed nodes in round $t > 3$, is at most the number of nodes which got informed in rounds $t - 1$ and $t - 2$. 
We consider here only the case $\Delta(G) = 3$. The case $\Delta(G) = 4$ is similar.

- The initial node may send at most three times.
- The initial node sends only in rounds 1, 2, 3.
- Any other nodes will be informed at time $t$ via an edge $e$.
- No further node may be informed via $e$.
- Thus any other node may send at most two times.
- If a node $v$ is informed in round $t$ by $w$, then did $w$ receive the information at round $t - 1$ or $t - 2$.
- Thus the number of newly informed nodes in round $t > 3$, is at most the number of nodes which got informed in rounds $t - 1$ and $t - 2$. 
Building the Idea

We consider here only the case $\Delta(G) = 3$. The case $\Delta(G) = 4$ is similar.

- The initial node may send at most three times.
- The initial node sends only in rounds 1, 2, 3.
- Any other nodes will be informed at time $t$ via an edge $e$.
- No further node may be informed via $e$.
- Thus any other node may send at most two times.
- If a node $v$ is informed in round $t$ by $w$, then did $w$ receive the information at round $t - 1$ or $t - 2$.

Thus the number of newly informed nodes in round $t > 3$, is at most the number of nodes which got informed in rounds $t - 1$ and $t - 2$. 
Building the Idea

We consider here only the case $\Delta(G) = 3$. The case $\Delta(G) = 4$ is similar.

- The initial node may send at most three times.
- The initial node sends only in rounds 1, 2, 3.
- Any other nodes will be informed at time $t$ via an edge $e$.
- No further node may be informed via $e$.
- Thus any other node may send at most two times.
- If a node $v$ is informed in round $t$ by $w$, then did $w$ receive the information at round $t - 1$ or $t - 2$.
- Thus the number of newly informed nodes in round $t > 3$, is at most the number of nodes which got informed in rounds $t - 1$ and $t - 2$. 

Proof

- Let $A(i) = |\text{Rec}_i^A(v_0)|$.
- $A(0) = 1$
- $A(1) = 1$
- $A(2) = 2$
- $A(3) = 4$
- $A(i) = A(i - 1) + A(i - 2)$ für $i \geq 4$.
- Show by induction: $A(i) \leq 1.61804^i$ for $i \geq 0$. 
Proof

- Let $A(i) = |Rec_i^A(v_0)|$.
- $A(0) = 1$
- $A(1) = 1$
- $A(2) = 2$
- $A(3) = 4$
- $A(i) = A(i - 1) + A(i - 2)$ für $i \geq 4$.
- Show by induction: $A(i) \leq 1.61804^i$ for $i \geq 0$. 
Proof

- Let $A(i) = |Rec_i^A(v_0)|$.
- $A(0) = 1$
- $A(1) = 1$
- $A(2) = 2$
- $A(3) = 4$
- $A(i) = A(i - 1) + A(i - 2)$ für $i \geq 4$.
- Show by induction: $A(i) \leq 1.61804^i$ for $i \geq 0$. 
Proof

- Let $A(i) = |Rec_i^A(v_0)|$.
- $A(0) = 1$
- $A(1) = 1$
- $A(2) = 2$
- $A(3) = 4$
- $A(i) = A(i - 1) + A(i - 2)$ für $i \geq 4$.
- Show by induction: $A(i) \leq 1.61804^i$ for $i \geq 0$. 
Proof

- Let $A(i) = |\text{Rec}_i^A(v_0)|$.
- $A(0) = 1$
- $A(1) = 1$
- $A(2) = 2$
- $A(3) = 4$
- $A(i) = A(i - 1) + A(i - 2)$ für $i \geq 4$.
- Show by induction: $A(i) \leq 1.61804^i$ for $i \geq 0$. 
Proof

- Let $A(i) = |Rec^A_i(v_0)|$.

- $A(0) = 1$

- $A(1) = 1$

- $A(2) = 2$

- $A(3) = 4$

- $A(i) = A(i - 1) + A(i - 2)$ für $i \geq 4$.

- Show by induction: $A(i) \leq 1.61804^i$ for $i \geq 0$. 
Proof

- Let $A(i) = |Rec_i^A(v_0)|$.
- $A(0) = 1$
- $A(1) = 1$
- $A(2) = 2$
- $A(3) = 4$
- $A(i) = A(i - 1) + A(i - 2)$ für $i \geq 4$.
- Show by induction: $A(i) \leq 1.61804^i$ for $i \geq 0$. 
Proof

- Let $A(i) = |Rec_i^A(v_0)|$.
- $A(0) = 1$
- $A(1) = 1$
- $A(2) = 2$
- $A(3) = 4$
- $A(i) = A(i - 1) + A(i - 2)$ für $i \geq 4$.
- Show by induction: $A(i) \leq 1.61804^i$ for $i \geq 0$. 
Proof

- \( A(0) = 1 \leq 1 = 1.61804^0 \)
- \( A(1) = 1 \leq 1.61804 = 1.61804^1 \)
- \( A(2) = 2 \leq 2.61805 = 1.61804^2 \)
- \( A(3) = 4 \leq 4.23612 = 1.61804^3 \)

Induction step \((i \geq 4):\)
- We have: \( A(j) \leq 1.61804^j \) for any \( j \leq i - 1 \).
- \( A(i) = A(i - 1) + A(i - 2) \leq 1.61804^{i-1} + 1.61804^{i-2} \leq 1.61804^i \)
- Note for this: \( 1.61804 + 1 \leq 1.61804^2 \).

Thus we have: \( n \leq |\text{Broadcast}_t^A(v_0)| = \sum_{i=0}^t |\text{Rec}_i^A(v_0)| \leq \sum_{i=0}^t A(i) \leq \sum_{i=0}^t 1.61804^i = \frac{1.61804^{t+1} - 1}{1.61804 - 1} \leq 3 \cdot 1.61804^t \)
- \( t \geq 1.4404 \cdot \log_2 n - 3. \)
- Proof of the second statement my be done in the same way.
**Proof**

- \( A(0) = 1 \leq 1 = 1.61804^0 \)
- \( A(1) = 1 \leq 1.61804 = 1.61804^1 \)
- \( A(2) = 2 \leq 2.61805 = 1.61804^2 \)
- \( A(3) = 4 \leq 4.23612 = 1.61804^3 \)

**Induction step \((i \geq 4):**
- We have: \( A(j) \leq 1.61804^i \) for any \( j \leq i - 1 \).
- \( A(i) = A(i - 1) + A(i - 2) \leq 1.61804^{i-1} + 1.61804^{i-2} \leq 1.61804^i \)
- Note for this: \( 1.61804 + 1 \leq 1.61804^2 \).

Thus we have: \( n \leq |\text{Broad}_t^A(v_0)| = \sum_{i=0}^{t} |Rec_i^A(v_0)| \leq \sum_{i=0}^{t} A(i) \leq \sum_{i=0}^{t} 1.61804^i = \frac{1.61804^{t+1} - 1}{1.61804 - 1} \leq 3 \cdot 1.61804^t \)

\( t \geq 1.4404 \cdot \log_2 n - 3. \)

Proof of the second statement may be done in the same way.
Proof

- \( A(0) = 1 \leq 1 = 1.61804^0 \)
- \( A(1) = 1 \leq 1.61804 = 1.61804^1 \)
- \( A(2) = 2 \leq 2.61805 = 1.61804^2 \)
- \( A(3) = 4 \leq 4.23612 = 1.61804^3 \)
- Induction step \((i \geq 4)\):
  - We have: \( A(j) \leq 1.61804^i \) for any \( j \leq i - 1 \).
  - \( A(i) = A(i - 1) + A(i - 2) \leq 1.61804^{i-1} + 1.61804^{i-2} \leq 1.61804^i \)
  - Note for this: \( 1.61804 + 1 \leq 1.61804^2 \).

Thus we have:

\[
|\text{Broad}_t^A(v_0)| = \sum_{i=0}^t |\text{Rec}_i^A(v_0)| \leq \sum_{i=0}^t A(i) \leq \sum_{i=0}^t 1.61804^i = \frac{1.61804^{t+1} - 1}{1.61804 - 1} \leq 3 \cdot 1.61804^t
\]

\( t \geq 1.4404 \cdot \log_2 n - 3. \)

Proof of the second statement may be done in the same way.
Proof

- $A(0) = 1 \leq 1 = 1.61804^0$
- $A(1) = 1 \leq 1.61804 = 1.61804^1$
- $A(2) = 2 \leq 2.61805 = 1.61804^2$
- $A(3) = 4 \leq 4.23612 = 1.61804^3$

Induction step ($i \geq 4$):

- We have: $A(j) \leq 1.61804^j$ for any $j \leq i - 1$.
- $A(i) = A(i - 1) + A(i - 2) \leq 1.61804^{i-1} + 1.61804^{i-2} \leq 1.61804^i$
- Note for this: $1.61804 + 1 \leq 1.61804^2$.

Thus we have: $n \leq |\text{Broad}_t^A(v_0)| = \sum_{i=0}^{t} |\text{Rec}_i^A(v_0)| \leq \sum_{i=0}^{t} A(i) \leq \sum_{i=0}^{t} 1.61804^i = \frac{1.61804^{t+1} - 1}{1.61804 - 1} \leq 3 \cdot 1.61804^t$

- $t \geq 1.4404 \cdot \log_2 n - 3$.

Proof of the second statement may be done in the same way.
Proof

A(0) = 1 ≤ 1 = 1.61804^0

A(1) = 1 ≤ 1.61804 = 1.61804^1

A(2) = 2 ≤ 2.61805 = 1.61804^2

A(3) = 4 ≤ 4.23612 = 1.61804^3

Induction step (i ≥ 4):

- We have: A(j) ≤ 1.61804^j for any j ≤ i − 1.
- A(i) = A(i − 1) + A(i − 2) ≤ 1.61804^{i−1} + 1.61804^{i−2} ≤ 1.61804^i
- Note for this: 1.61804 + 1 ≤ 1.61804^2.

Thus we have: n ≤ |\textit{Broad}_t^A(v_0)| = \sum_{i=0}^{t} |\textit{Rec}_i^A(v_0)| ≤ \sum_{i=0}^{t} A(i) \leq \sum_{i=0}^{t} 1.61804^i = \frac{1.61804^{t+1}−1}{1.61804−1} \leq 3 \cdot 1.61804^t

- t ≥ 1.4404 \cdot \log_2 n − 3.

Proof of the second statement may be done in the same way.
Proof

- $A(0) = 1 \leq 1 = 1.61804^0$
- $A(1) = 1 \leq 1.61804 = 1.61804^1$
- $A(2) = 2 \leq 2.61805 = 1.61804^2$
- $A(3) = 4 \leq 4.23612 = 1.61804^3$

Induction step ($i \geq 4$):
- We have: $A(j) \leq 1.61804^j$ for any $j \leq i - 1$.
- $A(i) = A(i - 1) + A(i - 2) \leq 1.61804^{i-1} + 1.61804^{i-2} \leq 1.61804^i$
- Note for this: $1.61804 + 1 \leq 1.61804^2$.

Thus we have:

- $n \leq |\text{Broad}_t^A(v_0)| = \sum_{i=0}^{t} |\text{Rec}_i^A(v_0)| \leq \sum_{i=0}^{t} A(i) \leq \sum_{i=0}^{t} 1.61804^i = \frac{1.61804^{t+1} - 1}{1.61804 - 1} \leq 3 \cdot 1.61804^t$
- $t \geq 1.4404 \cdot \log_2 n - 3$

Proof of the second statement may be done in the same way.
Proof

- $A(0) = 1 \leq 1 = 1.61804^0$
- $A(1) = 1 \leq 1.61804 = 1.61804^1$
- $A(2) = 2 \leq 2.61805 = 1.61804^2$
- $A(3) = 4 \leq 4.23612 = 1.61804^3$
- Induction step ($i \geq 4$):
  - We have: $A(j) \leq 1.61804^j$ for any $j \leq i - 1$.
  - $A(i) = A(i - 1) + A(i - 2) \leq 1.61804^{i-1} + 1.61804^{i-2} \leq 1.61804^i$
  - Note for this: $1.61804 + 1 \leq 1.61804^2$.

Thus we have: $n \leq |Broad_t^A(v_0)| = \sum_{i=0}^{t} |Rec_i^A(v_0)| \leq \sum_{i=0}^{t} A(i) \leq \sum_{i=0}^{t} 1.61804^i = \frac{1.61804^{t+1} - 1}{1.61804 - 1} \leq 3 \cdot 1.61804^t$

- $t \geq 1.4404 \cdot \log_2 n - 3$.
- Proof of the second statement may be done in the same way.
Introduction

Broadcast Lower Bounds Simple Graphs Telephone-Mode Telegraph-Mode Sum.

Degree of the Nodes (3:26.8)

Proof

- $A(0) = 1 \leq 1 = 1.61804^0$
- $A(1) = 1 \leq 1.61804 = 1.61804^1$
- $A(2) = 2 \leq 2.61805 = 1.61804^2$
- $A(3) = 4 \leq 4.23612 = 1.61804^3$
- Induction step ($i \geq 4$):
  - We have: $A(j) \leq 1.61804^i$ for any $j \leq i - 1$.
  - $A(i) = A(i - 1) + A(i - 2) \leq 1.61804^{i-1} + 1.61804^{i-2} \leq 1.61804^i$
  - Note for this: $1.61804 + 1 \leq 1.61804^2$.
- Thus we have: $n \leq |\text{Broadcast}_t^A(v_0)| = \sum_{i=0}^{t} |\text{Rec}_i^A(v_0)| \leq \sum_{i=0}^{t} A(i) \leq \sum_{i=0}^{t} 1.61804^i = \frac{1.61804^{t+1}-1}{1.61804-1} \leq 3 \cdot 1.61804^t$
- $t \geq 1.4404 \cdot \log_2 n - 3$.
- Proof of the second statement may be done in the same way.
Proof

- $A(0) = 1 \leq 1.61804^0$
- $A(1) = 1 \leq 1.61804 = 1.61804^1$
- $A(2) = 2 \leq 2.61805 = 1.61804^2$
- $A(3) = 4 \leq 4.23612 = 1.61804^3$

Induction step ($i \geq 4$):
- We have: $A(j) \leq 1.61804^j$ for any $j \leq i - 1$.
- $A(i) = A(i - 1) + A(i - 2) \leq 1.61804^{i-1} + 1.61804^{i-2} \leq 1.61804^i$
- Note for this: $1.61804 + 1 \leq 1.61804^2$.

Thus we have:
- $n \leq |\text{Broadcast}^A(v_0)| = \sum_{i=0}^{t} |\text{Rec}_i^A(v_0)| \leq \sum_{i=0}^{t} A(i) \leq \sum_{i=0}^{t} 1.61804^i = \frac{1.61804^{t+1} - 1}{1.61804 - 1} \leq 3 \cdot 1.61804^t$
- $t \geq 1.4404 \cdot \log_2 n - 3$.

Proof of the second statement may be done in the same way.
Proof

- \(A(0) = 1 \leq 1 = 1.61804^0\)
- \(A(1) = 1 \leq 1.61804 = 1.61804^1\)
- \(A(2) = 2 \leq 2.61805 = 1.61804^2\)
- \(A(3) = 4 \leq 4.23612 = 1.61804^3\)

Induction step \((i \geq 4)\):

- We have: \(A(j) \leq 1.61804^j\) for any \(j \leq i - 1\).
- \(A(i) = A(i - 1) + A(i - 2) \leq 1.61804^{i-1} + 1.61804^{i-2} \leq 1.61804^i\)
- Note for this: \(1.61804 + 1 \leq 1.61804^2\).

Thus we have: \(n \leq |\text{Broadcast}^A_t(v_0)| = \sum_{i=0}^{t} |\text{Rec}^A_i(v_0)| \leq \sum_{i=0}^{t} A(i) \leq \sum_{i=0}^{t} 1.61804^i = \frac{1.61804^{t+1} - 1}{1.61804 - 1} \leq 3 \cdot 1.61804^t\)

- \(t \geq 1.4404 \cdot \log_2 n - 3\).

- Proof of the second statement may be done in the same way.
Proof

- $A(0) = 1 \leq 1 = 1.61804^0$
- $A(1) = 1 \leq 1.61804 = 1.61804^1$
- $A(2) = 2 \leq 2.61805 = 1.61804^2$
- $A(3) = 4 \leq 4.23612 = 1.61804^3$

Induction step ($i \geq 4$):

- We have: $A(j) \leq 1.61804^j$ for any $j \leq i - 1$.
- $A(i) = A(i - 1) + A(i - 2) \leq 1.61804^{i-1} + 1.61804^{i-2} \leq 1.61804^i$
- Note for this: $1.61804 + 1 \leq 1.61804^2$.

Thus we have: $n \leq |\text{Broadcast}^A_t(v_0)| = \sum_{i=0}^{t} |\text{Rec}^A_i(v_0)| \leq \sum_{i=0}^{t} A(i) \leq \sum_{i=0}^{t} 1.61804^i = \frac{1.61804^{t+1} - 1}{1.61804 - 1} \leq 3 \cdot 1.61804^t$

- $t \geq 1.4404 \cdot \log_2 n - 3$.

Proof of the second statement may be done in the same way.
Proof

- $A(0) = 1 \leq 1 = 1.61804^0$
- $A(1) = 1 \leq 1.61804 = 1.61804^1$
- $A(2) = 2 \leq 2.61805 = 1.61804^2$
- $A(3) = 4 \leq 4.23612 = 1.61804^3$

Induction step ($i \geq 4$):
- We have: $A(j) \leq 1.61804^j$ for any $j \leq i - 1$.
- $A(i) = A(i - 1) + A(i - 2) \leq 1.61804^{i-1} + 1.61804^{i-2} \leq 1.61804^i$
- Note for this: $1.61804 + 1 \leq 1.61804^2$.

Thus we have: $n \leq |\text{Broadcast}^A_t(v_0)| = \sum_{i=0}^{t} |\text{Rec}^A_i(v_0)| \leq \sum_{i=0}^{t} A(i) \leq \sum_{i=0}^{t} 1.61804^i = \frac{1.61804^{t+1} - 1}{1.61804 - 1} \leq 3 \cdot 1.61804^t$

- $t \geq 1.4404 \cdot \log_2 n - 3$.

Proof of the second statement may be done in the same way.
Recall

Definition (Gossip):
Given is $G = (V, E)$.
- Each node $w \in V$ has some information $I(w)$ and no node of $V \setminus \{w\}$ knows $I(w)$.
- Construct algorithm, where each node $v \in V$ collects information $\bigcup_{w \in V} I(w)$.

- By $\text{comm}(A)$ we denote the complexity (number of rounds) of a communication-algorithm.
- $r(G) = \min\{\text{comm}(A) \mid A \text{ is a one-way algorithm for the gossip-problem on } G\}$
- $r_2(G) = \min\{\text{comm}(A) \mid A \text{ is a two-way algorithm for the gossip-problem on } G\}$
Motivation

- Broadcast is a part of gossip.
- Many broadcasts have to “cooperate”. This makes the problem interesting.
- More important for algorithms on networks.
- Example: Distribute lower bounds for “Branch and Bound”.
- For gossip we get a difference between telegraph- and telephone-mode.
- We start with gossiping in the telephone-mode.
Motivation

- Broadcast is a part of gossip.
- Many broadcasts have to “cooperate”. This makes the problem interesting.
- More important for algorithms on networks.
- Example: Distribute lower bounds for “Branch and Bound”.
- For gossip we get a difference between telegraph- and telephone-mode.
- We start with gossiping in the telephone-mode.
Motivation

- Broadcast is a part of gossip.
- Many broadcasts have to “cooperate”. This makes the problem interesting.
- More important for algorithms on networks.
- Example: Distribute lower bounds for “Branch and Bound”.
- For gossip we get a difference between telegraph- and telephone-mode.
- We start with gossiping in the telephone-mode.
Motivation

- Broadcast is a part of gossip.
- Many broadcasts have to “cooperate”. This makes the problem interesting.
- More important for algorithms on networks.
- **Example:** Distribute lower bounds for “Branch and Bound”.
- For gossip we get a difference between telegraph- and telephone-mode.
- We start with gossiping in the telephone-mode.
Motivation

- Broadcast is a part of gossip.
- Many broadcasts have to “cooperate”. This makes the problem interesting.
- More important for algorithms on networks.
- Example: Distribute lower bounds for “Branch and Bound”.
- For gossip we get a difference between telegraph- and telephone-mode.
- We start with gossiping in the telephone-mode.
Motivation

- Broadcast is a part of gossip.
- Many broadcasts have to “cooperate”. This makes the problem interesting.
- More important for algorithms on networks.
- Example: Distribute lower bounds for “Branch and Bound”.
- For gossip we get a difference between telegraph- and telephone-mode.
- We start with gossiping in the telephone-mode.
Motivation

- Broadcast is a part of gossip.
- Many broadcasts have to “cooperate”. This makes the problem interesting.
- More important for algorithms on networks.
- Example: Distribute lower bounds for “Branch and Bound”.
- For gossip we get a difference between telegraph- and telephone-mode.
- We start with gossiping in the telephone-mode.
Lemma:

Let $G = (V, E)$ a graph with $n$ nodes. Then we have:

$$r(G) \geq r_2(G) \geq \begin{cases} \lceil \log_2 n \rceil & \text{n even}, \\ \lceil \log_2 n \rceil + 1 & \text{n odd}. \end{cases}$$

Proof: Only the case, where $n$ is odd, has to be proven.

- Show: $r_2(G) \geq \lceil \log_2 n \rceil + 1$.
- Let $A$ be a communication-algorithm for the gossip-problem. $A$ has communication rounds (matchings) $E_1, E_2, \cdots, E_k$.
- Show by induction: After $i$ rounds has each node at most $2^i$ pieces of information.
  - $i = 0$: Each node has $2^0 = 1$ pieces of information.
  - $i - 1 \rightarrow i$: at most $2^{i-1} + 2^{i-1} = 2^i$ pieces of information may be collected by any node.
- In round $k$ is at least one node $v$ inactive.
- $v$ has after $k$ rounds at most $2^{k-1}$ pieces of information.
Lemma:

Let $G = (V, E)$ a graph with $n$ nodes. Then we have:

$$r(G) \geq r_2(G) \geq \begin{cases} \lceil \log_2 n \rceil & n \text{ even}, \\ \lceil \log_2 n \rceil + 1 & n \text{ odd}. \end{cases}$$

Proof: Only the case, where $n$ is odd, has to be proven.

- **Show:** $r_2(G) \geq \lceil \log_2 n \rceil + 1$.

- Let $A$ be a communication-algorithm for the gossip-problem. $A$ has communication rounds (matchings) $E_1, E_2, \cdots, E_k$.

- Show by induction: After $i$ rounds has each node at most $2^i$ pieces of information.
  - $i = 0$: Each node has $2^0 = 1$ pieces of information.
  - $i - 1 \rightarrow i$: at most $2^{i-1} + 2^{i-1} = 2^i$ pieces of information may be collected by any node.

- In round $k$ is at least one node $v$ inactive.

- $v$ has after $k$ rounds at most $2^{k-1}$ pieces of information.
Lower Bound

Lemma:

Let $G = (V, E)$ a graph with $n$ nodes. Then we have:

$$r(G) \geq r_2(G) \geq \begin{cases} \lceil \log_2 n \rceil & n \text{ even}, \\ \lceil \log_2 n \rceil + 1 & n \text{ odd}. \end{cases}$$

Proof: Only the case, where $n$ is odd, has to be proven.

- Show: $r_2(G) \geq \lceil \log_2 n \rceil + 1$.

- Let $A$ be a communication-algorithm for the gossip-problem. $A$ has communication rounds (matchings) $E_1, E_2, \cdots, E_k$.

- Show by induction: After $i$ rounds has each node at most $2^i$ pieces of information.

  - $i = 0$: Each node has $2^0 = 1$ pieces of information.
  - $i - 1 \rightarrow i$: at most $2^{i-1} + 2^{i-1} = 2^i$ pieces of information may be collected by any node.

- In round $k$ is at least one node $v$ inactive.

- $v$ has after $k$ rounds at most $2^{k-1}$ pieces of information.
Lemma:

Let $G = (V, E)$ a graph with $n$ nodes. Then we have:

$$r(G) \geq r_2(G) \geq \begin{cases} \lceil \log_2 n \rceil & n \text{ even,} \\ \lceil \log_2 n \rceil + 1 & n \text{ odd.} \end{cases}$$

Proof: Only the case, where $n$ is odd, has to be proven.

- Show: $r_2(G) \geq \lceil \log_2 n \rceil + 1$.

- Let $A$ be a communication-algorithm for the gossip-problem. $A$ has communication rounds (matchings) $E_1, E_2, \cdots, E_k$.

- Show by induction: After $i$ rounds has each node at most $2^i$ pieces of information.
  - $i = 0$: Each node has $2^0 = 1$ pieces of information.
  - $i - 1 \rightarrow i$: at most $2^{i-1} + 2^{i-1} = 2^i$ pieces of information may be collected by any node.

- In round $k$ is at least one node $v$ inactive.

- $v$ has after $k$ rounds at most $2^{k-1}$ pieces of information.
Lemma:

Let $G = (V, E)$ a graph with $n$ nodes. Then we have:

$$r(G) \geq r_2(G) \geq \begin{cases} \left\lceil \log_2 n \right\rceil & n \text{ even,} \\ \left\lceil \log_2 n \right\rceil + 1 & n \text{ odd.} \end{cases}$$

Proof: Only the case, where $n$ is odd, has to be proven.

- **Show:** $r_2(G) \geq \left\lceil \log_2 n \right\rceil + 1$.

- Let $A$ be a communication-algorithm for the gossip-problem. $A$ has communication rounds (matchings) $E_1, E_2, \cdots, E_k$.

- **Show by induction:** After $i$ rounds has each node at most $2^i$ pieces of information.
  - **$i = 0$:** Each node has $2^0 = 1$ pieces of information.
  - **$i - 1 \rightarrow i$:** at most $2^{i-1} + 2^{i-1} = 2^i$ pieces of information may be collected by any node.

- In round $k$ is at least one node $v$ inactive.
- $v$ has after $k$ rounds at most $2^{k-1}$ pieces of information.
**Lemma:**

Let $G = (V, E)$ a graph with $n$ nodes. Then we have:

$$ r(G) \geq r_2(G) \geq \begin{cases} \lceil \log_2 n \rceil & n \text{ even}, \\ \lceil \log_2 n \rceil + 1 & n \text{ odd}. \end{cases} $$

**Proof:** Only the case, where $n$ is odd, has to be proven.

- **Show:** $r_2(G) \geq \lceil \log_2 n \rceil + 1$.
- **Let** $A$ be a communication-algorithm for the gossip-problem. $A$ has communication rounds (matchings) $E_1, E_2, \cdots, E_k$.
- **Show by induction:** After $i$ rounds has each node at most $2^i$ pieces of information.
  - $i = 0$: Each node has $2^0 = 1$ pieces of information.
  - $i - 1 \rightarrow i$: at most $2^{i-1} + 2^{i-1} = 2^i$ pieces of information may be collected by any node.
- **In round $k$** is at least one node $v$ inactive.
- **$v$ has after $k$ rounds** at most $2^{k-1}$ pieces of information.
Lemma:

Let $G = (V, E)$ a graph with $n$ nodes. Then we have:

$$r(G) \geq r_2(G) \geq \begin{cases} \lceil \log_2 n \rceil & n \text{ even}, \\ \lceil \log_2 n \rceil + 1 & n \text{ odd}. \end{cases}$$

Proof: Only the case, where $n$ is odd, has to be proven.

- Show: $r_2(G) \geq \lceil \log_2 n \rceil + 1$.
- Let $A$ be a communication-algorithm for the gossip-problem. $A$ has communication rounds (matchings) $E_1, E_2, \cdots, E_k$.
- Show by induction: After $i$ rounds has each node at most $2^i$ pieces of information.
  - $i = 0$: Each node has $2^0 = 1$ pieces of information.
  - $i - 1 \rightarrow i$: at most $2^{i-1} + 2^{i-1} = 2^i$ pieces of information may be collected by any node.
- In round $k$ is at least one node $v$ inactive.
- $v$ has after $k$ rounds at most $2^{k-1}$ pieces of information.
Lower Bound

Lemma:

Let \( G = (V, E) \) a graph with \( n \) nodes. Then we have:

\[
    r(G) \geq r_2(G) \geq \begin{cases} 
    \lceil \log_2 n \rceil & \text{n even}, \\
    \lceil \log_2 n \rceil + 1 & \text{n odd}.
\end{cases}
\]

Proof: Only the case, where \( n \) is odd, has to be proven.

- **Show:** \( r_2(G) \geq \lceil \log_2 n \rceil + 1 \).

- Let \( A \) be a communication-algorithm for the gossip-problem. \( A \) has communication rounds (matchings) \( E_1, E_2, \ldots, E_k \).

- **Show by induction:** After \( i \) rounds has each node at most \( 2^i \) pieces of information.
  
  - \( i = 0 \): Each node has \( 2^0 = 1 \) pieces of information.
  - \( i - 1 \to i \): at most \( 2^{i-1} + 2^{i-1} = 2^i \) pieces of information may be collected by any node.

- In round \( k \) is at least one node \( v \) inactive.

- \( v \) has after \( k \) rounds at most \( 2^{k-1} \) pieces of information.
Lemma:

Let $G = (V, E)$ a graph with $n$ nodes. Then we have:

$$r(G) \geq r_2(G) \geq \begin{cases} \lceil \log_2 n \rceil & \text{if } n \text{ even}, \\ \lceil \log_2 n \rceil + 1 & \text{if } n \text{ odd}. \end{cases}$$

Proof: Only the case, where $n$ is odd, has to be proven.

- Show: $r_2(G) \geq \lceil \log_2 n \rceil + 1$.

- Let $A$ be a communication-algorithm for the gossip-problem. 
  $A$ has communication rounds (matchings) $E_1, E_2, \cdots, E_k$.

- Show by induction: After $i$ rounds has each node at most $2^i$ pieces of information.
  - $i = 0$: Each node has $2^0 = 1$ pieces of information.
  - $i - 1 \to i$: at most $2^{i-1} + 2^{i-1} = 2^i$ pieces of information may be collected by any node.

- In round $k$ is at least one node $v$ inactive.

- $v$ has after $k$ rounds at most $2^{k-1}$ pieces of information.
Lemma:

For any graph $G = (V, E)$ with $|V| = n$ we have:

- $r(G) \leq 2n - 2$, and
- $r_2(G) \leq 2n - 3$.

Proof: Follows from the following known statements:

- $\minb(G) \leq n - 1$ for any graph $G = (V, E)$ with $|V| = n$.
- $r(G) \leq 2 \cdot \minb(G)$
- $r_2(G) \leq 2 \cdot \minb(G) - 1$
Lemma:

For any graph $G = (V, E)$ with $|V| = n$ we have:

- $r(G) \leq 2n - 2$, and
- $r_2(G) \leq 2n - 3$.

Proof: Follows from the following known statements:

- $\text{minb}(G) \leq n - 1$ for any graph $G = (V, E)$ with $|V| = n$.
- $r(G) \leq 2 \cdot \text{minb}(G)$
- $r_2(G) \leq 2 \cdot \text{minb}(G) - 1$
Lemma:

For any graph $G = (V, E)$ with $|V| = n$ we have:

- $r(G) \leq 2n - 2$, and
- $r_2(G) \leq 2n - 3$.

Proof: Follows from the following known statements:

- $\minb(G) \leq n - 1$ for any graph $G = (V, E)$ with $|V| = n$.
- $r(G) \leq 2 \cdot \minb(G)$
- $r_2(G) \leq 2 \cdot \minb(G) - 1$
Lemma:

For any graph $G = (V, E)$ with $|V| = n$ we have:

- $r(G) \leq 2n - 2$, and
- $r_2(G) \leq 2n - 3$.

Proof: Follows from the following known statements:

- $\min b(G) \leq n - 1$ for any graph $G = (V, E)$ with $|V| = n$.
- $r(G) \leq 2 \cdot \min b(G)$
- $r_2(G) \leq 2 \cdot \min b(G) - 1$
Simple Algorithm

Lemma:
For any graph $G = (V, E)$ with $|V| = n$ we have:
- $r(G) \leq 2n - 2$, and
- $r_2(G) \leq 2n - 3$.

Proof: Follows from the following known statements:
- $\min_{b}(G) \leq n - 1$ for any graph $G = (V, E)$ with $|V| = n$.
- $r(G) \leq 2 \cdot \min_{b}(G)$
- $r_2(G) \leq 2 \cdot \min_{b}(G) - 1$
Lemma:

For any graph $G = (V, E)$ with $|V| = n$ we have:

- $r(G) \leq 2n - 2$, and
- $r_2(G) \leq 2n - 3$.

Proof: Follows from the following known statements:

- $\text{minb}(G) \leq n - 1$ for any graph $G = (V, E)$ with $|V| = n$.
- $r(G) \leq 2 \cdot \text{minb}(G)$
- $r_2(G) \leq 2 \cdot \text{minb}(G) - 1$
Simple Algorithm

Lemma:

For any graph \( G = (V, E) \) with \( |V| = n \) we have:
- \( r(G) \leq 2n - 2 \), and
- \( r_2(G) \leq 2n - 3 \).

Proof: Follows from the following known statements:
- \( \minb(G) \leq n - 1 \) for any graph \( G = (V, E) \) with \( |V| = n \).
- \( r(G) \leq 2 \cdot \minb(G) \)
- \( r_2(G) \leq 2 \cdot \minb(G) - 1 \)
Simple Algorithm

Lemma:

For any graph $G = (V, E)$ with $|V| = n$ we have:

- $r(G) \leq 2n - 2$, and
- $r_2(G) \leq 2n - 3$.

Proof: Follows from the following known statements:

- $\minb(G) \leq n - 1$ for any graph $G = (V, E)$ with $|V| = n$.
- $r(G) \leq 2 \cdot \minb(G)$
- $r_2(G) \leq 2 \cdot \minb(G) - 1$
Simple Algorithm (Continuation)

Lemma:

We have:

- \( r(T_k(1)) = 2k \)
- \( r_2(T_k(1)) = 2k - 1 \)

Proof:

- Show: \( r(T_k(1)) \geq 2k \).
- \( r(T_k(1)) \) has one root and \( k \) leaves.
- The maximal matching is 1.
- In each round is only one leaf active.
- Each leaf has to send at least once.
- Each leaf has to receive at least once.
- Thus in total \( 2k \) rounds necessary.
- \( r_2(T_k(1)) \geq 2k - 1 \), is a simple exercise.
Simple Algorithm (Continuation)

Lemma:
We have:
- \( r(T_k(1)) = 2k \)
- \( r_2(T_k(1)) = 2k - 1 \)

Proof:
- Show: \( r(T_k(1)) \geq 2k \).
- \( r(T_k(1)) \) has one root and \( k \) leaves.
- The maximal matching is 1.
- In each round is only one leaf active.
- Each leaf has to send at least once.
- Each leaf has to receive at least once.
- Thus in total \( 2k \) rounds necessary.
- \( r_2(T_k(1)) \geq 2k - 1 \), is a simple exercise.
Simple Algorithm (Continuation)

Lemma:

We have:

- $r(T_k(1)) = 2k$
- $r_2(T_k(1)) = 2k - 1$

Proof:

- Show: $r(T_k(1)) \geq 2k$.
- $r(T_k(1))$ has one root and $k$ leaves.
- The maximal matching is 1.
- In each round is only one leaf active.
- Each leaf has to send at least once.
- Each leaf has to receive at least once.
- Thus in total $2k$ rounds necessary.
- $r_2(T_k(1)) \geq 2k - 1$, is a simple exercise.
**Simple Algorithm (Continuation)**

Lemma:

We have:
- \( r(T_k(1)) = 2k \)
- \( r_2(T_k(1)) = 2k - 1 \)

Proof:

- **Show**: \( r(T_k(1)) \geq 2k \).
- \( r(T_k(1)) \) has one root and \( k \) leaves.
- The maximal matching is 1.
- In each round is only one leaf active.
- Each leaf has to send at least once.
- Each leaf has to receive at least once.
- Thus in total \( 2k \) rounds necessary.
- \( r_2(T_k(1)) \geq 2k - 1 \), is a simple exercise.
Simple Algorithm (Continuation)

Lemma:
We have:
- \( r(T_k(1)) = 2k \)
- \( r_2(T_k(1)) = 2k - 1 \)

Proof:
- Show: \( r(T_k(1)) \geq 2k \).
- \( r(T_k(1)) \) has one root and \( k \) leaves.
- The maximal matching is 1.
- In each round is only one leaf active.
- Each leaf has to send at least once.
- Each leaf has to receive at least once.
- Thus in total \( 2k \) rounds necessary.
- \( r_2(T_k(1)) \geq 2k - 1 \), is a simple exercise.
Simple Algorithm (Continuation)

Lemma:
We have:
- \( r(T_k(1)) = 2k \)
- \( r_2(T_k(1)) = 2k - 1 \)

Proof:
- Show: \( r(T_k(1)) \geq 2k \).
- \( r(T_k(1)) \) has one root and \( k \) leaves.
- The maximal matching is 1.
- In each round is only one leaf active.
- Each leaf has to send at least once.
- Each leaf has to receive at least once.
- Thus in total \( 2k \) rounds necessary.
- \( r_2(T_k(1)) \geq 2k - 1 \), is a simple exercise.
Simple Algorithm (Continuation)

Lemma:

We have:

- $r(T_k(1)) = 2k$
- $r_2(T_k(1)) = 2k - 1$

Proof:

- Show: $r(T_k(1)) \geq 2k$.
- $r(T_k(1))$ has one root and $k$ leaves.
- The maximal matching is $1$.
- In each round is only one leaf active.
- Each leaf has to send at least once.
- Each leaf has to receive at least once.
- Thus in total $2k$ rounds necessary.
- $r_2(T_k(1)) \geq 2k - 1$, is a simple exercise.
Simple Algorithm (Continuation)

Lemma:

We have:

- $r(T_k(1)) = 2k$
- $r_2(T_k(1)) = 2k - 1$

Proof:

- **Show:** $r(T_k(1)) \geq 2k$.
- $r(T_k(1))$ has one root and $k$ leaves.
- The maximal matching is 1.
- In each round is only one leaf active.
- **Each leaf has to send at least once.**
- Each leaf has to receive at least once.
- Thus in total $2k$ rounds necessary.
- $r_2(T_k(1)) \geq 2k - 1$, is a simple exercise.
Simple Algorithm (Continuation)

Lemma:
We have:
- \( r(T_k(1)) = 2k \)
- \( r_2(T_k(1)) = 2k - 1 \)

Proof:
- Show: \( r(T_k(1)) \geq 2k \).
- \( r(T_k(1)) \) has one root and \( k \) leaves.
- The maximal matching is 1.
- In each round is only one leaf active.
- Each leaf has to send at least once.
- Each leaf has to receive at least once.
- Thus in total \( 2k \) rounds necessary.
- \( r_2(T_k(1)) \geq 2k - 1 \), is a simple exercise.
Simple Algorithm (Continuation)

Lemma:
We have:
- $r(T_k(1)) = 2k$
- $r_2(T_k(1)) = 2k - 1$

Proof:
- Show: $r(T_k(1)) \geq 2k$.
- $r(T_k(1))$ has one root and $k$ leaves.
- The maximal matching is 1.
- In each round is only one leaf active.
- Each leaf has to send at least once.
- Each leaf has to receive at least once.
- **Thus in total** $2k$ **rounds necessary.**
- $r_2(T_k(1)) \geq 2k - 1$, is a simple exercise.
Simple Algorithm (Continuation)

Lemma:

We have:
- \( r(T_k(1)) = 2k \)
- \( r_2(T_k(1)) = 2k - 1 \)

Proof:
- Show: \( r(T_k(1)) \geq 2k \).
- \( r(T_k(1)) \) has one root and \( k \) leaves.
- The maximal matching is 1.
- In each round is only one leaf active.
- Each leaf has to send at least once.
- Each leaf has to receive at least once.
- Thus in total \( 2k \) rounds necessary.
- \( r_2(T_k(1)) \geq 2k - 1 \), is a simple exercise.
Simple Algorithm (Continuation)

Lemma:

We have:

- \( r(T_k(1)) = 2k \)
- \( r_2(T_k(1)) = 2k - 1 \)

Proof:

- Show: \( r(T_k(1)) \geq 2k \).
- \( r(T_k(1)) \) has one root and \( k \) leaves.
- The maximal matching is 1.
- In each round is only one leaf active.
- Each leaf has to send at least once.
- Each leaf has to receive at least once.
- Thus in total \( 2k \) rounds necessary.
- \( r_2(T_k(1)) \geq 2k - 1 \), is a simple exercise.
Gossip on Lines

**Theorem:**

We have:

- \( r_2(L(n)) = n - 1 \) for any even number \( n \geq 2 \),
- \( r_2(L(n)) = n \) for any odd number \( n \geq 3 \),
- \( r(L(n)) = n \) for any even number \( n \geq 2 \) and
- \( r(L(n)) = n + 1 \) for any odd number \( n \geq 3 \).

**Proof:**

- Show: \( r_2(L(n)) \geq n - 1 \).
- Note: \( r_2(L(n)) \geq b(L(n)) \geq diam(L(n)) = n - 1 \)
Gossip on Lines

Theorem:

We have:

- \( r_2(L(n)) = n - 1 \) for any even number \( n \geq 2 \),
- \( r_2(L(n)) = n \) for any odd number \( n \geq 3 \),
- \( r(L(n)) = n \) for any even number \( n \geq 2 \) and
- \( r(L(n)) = n + 1 \) for any odd number \( n \geq 3 \).

Proof:

- Show: \( r_2(L(n)) \geq n - 1 \).
- Note: \( r_2(L(n)) \geq b(L(n)) \geq diam(L(n)) = n - 1 \).
Gossip on Lines

**Theorem:**

We have:

- \( r_2(L(n)) = n - 1 \) for any even number \( n \geq 2 \),
- \( r_2(L(n)) = n \) for any odd number \( n \geq 3 \),
- \( r(L(n)) = n \) for any even number \( n \geq 2 \) and
- \( r(L(n)) = n + 1 \) for any odd number \( n \geq 3 \).

**Proof:**

- Show: \( r_2(L(n)) \geq n - 1 \).
- Note: \( r_2(L(n)) \geq b(L(n)) \geq diam(L(n)) = n - 1 \)
Gossip on Lines

Theorem:
We have:
- $r_2(L(n)) = n - 1$ for any even number $n \geq 2$,
- $r_2(L(n)) = n$ for any odd number $n \geq 3$,
- $r(L(n)) = n$ for any even number $n \geq 2$ and
- $r(L(n)) = n + 1$ for any odd number $n \geq 3$.

Proof:
- Show: $r_2(L(n)) \geq n - 1$.
- Note: $r_2(L(n)) \geq b(L(n)) \geq diam(L(n)) = n - 1$.
Gossip on Lines

Theorem:

We have:

- $r_2(L(n)) = n - 1$ for any even number $n \geq 2$,
- $r_2(L(n)) = n$ for any odd number $n \geq 3$,
- $r(L(n)) = n$ for any even number $n \geq 2$ and
- $r(L(n)) = n + 1$ for any odd number $n \geq 3$.

Proof:

- Show: $r_2(L(n)) \geq n - 1$.
- Note: $r_2(L(n)) \geq b(L(n)) \geq diam(L(n)) = n - 1$
Gossip on Lines

Theorem:

We have:

- $r_2(L(n)) = n - 1$ for any even number $n \geq 2$,
- $r_2(L(n)) = n$ for any odd number $n \geq 3$,
- $r(L(n)) = n$ for any even number $n \geq 2$ and
- $r(L(n)) = n + 1$ for any odd number $n \geq 3$.

Proof:

- Show: $r_2(L(n)) \geq n - 1$.
- Note: $r_2(L(n)) \geq b(L(n)) \geq diam(L(n)) = n - 1$
Gossip on Lines

Theorem:

We have:

- \( r_2(L(n)) = n - 1 \) for any even number \( n \geq 2 \),
- \( r_2(L(n)) = n \) for any odd number \( n \geq 3 \),
- \( r(L(n)) = n \) for any even number \( n \geq 2 \) and
- \( r(L(n)) = n + 1 \) for any odd number \( n \geq 3 \).

Proof:

- Show: \( r_2(L(n)) \geq n - 1 \).
- Note: \( r_2(L(n)) \geq b(L(n)) \geq diam(L(n)) = n - 1 \)
Gossip on Lines

**Theorem:**

We have:

- $r_2(L(n)) = n - 1$ for any even number $n \geq 2$,
- $r_2(L(n)) = n$ for any odd number $n \geq 3$,
- $r(L(n)) = n$ for any even number $n \geq 2$ and
- $r(L(n)) = n + 1$ for any odd number $n \geq 3$.

**Proof:**

- **Show:** $r_2(L(n)) \geq n - 1$.
- **Note:** $r_2(L(n)) \geq b(L(n)) \geq diam(L(n)) = n - 1$
Gossip on Lines (Proof I)

- **Show:** \( r_2(L(n)) \leq n - 1 \) for \( n \) even.

- Consider algorithm \( A \), given by the following matchings:
  
  1. \( \{\{0, 1\}, \{n - 1, n - 2\}\} \),
  2. \( \{\{1, 2\}, \{n - 2, n - 3\}\} \),
  3. \( \{\{2, 3\}, \{n - 3, n - 4\}\} \),
  4. \( \ldots \)
  5. \( \{\{n/2 - 1, n/2\}\} \)
  6. \( \ldots \)
  7. \( \{\{2, 3\}, \{n - 3, n - 4\}\} \),
  8. \( \{\{1, 2\}, \{n - 2, n - 3\}\} \),
  9. \( \{\{0, 1\}, \{n - 1, n - 2\}\} \)
Gossip on Lines (Proof I)

- Show: $r_2(L(n)) \leq n - 1$ for $n$ even.

- Consider algorithm $A$, given by the following matchings:

  - $1$ \{\{0, 1\}, \{n - 1, n - 2\}\},
  - $2$ \{\{1, 2\}, \{n - 2, n - 3\}\},
  - $3$ \{\{2, 3\}, \{n - 3, n - 4\}\},
  - $4$ \ldots
  - $5$ \{\{n/2 - 1, n/2\}\}
  - $6$ \ldots
  - $7$ \{\{2, 3\}, \{n - 3, n - 4\}\},
  - $8$ \{\{1, 2\}, \{n - 2, n - 3\}\},
  - $9$ \{\{0, 1\}, \{n - 1, n - 2\}\}
Gossip on Lines (Proof I)

- Show: \( r_2(L(n)) \leq n - 1 \) for \( n \) even.

- Consider algorithm \( A \), given by the following matchings:

\[
\begin{align*}
1 & \quad \{\{0, 1\}, \{n - 1, n - 2\}\}, \\
2 & \quad \{\{1, 2\}, \{n - 2, n - 3\}\}, \\
3 & \quad \{\{2, 3\}, \{n - 3, n - 4\}\}, \\
4 & \quad \quad \quad \quad \quad \quad \ldots \\
5 & \quad \{\{n/2 - 1, n/2\}\} \\
6 & \quad \quad \quad \quad \quad \quad \ldots \\
7 & \quad \{\{2, 3\}, \{n - 3, n - 4\}\}, \\
8 & \quad \{\{1, 2\}, \{n - 2, n - 3\}\}, \\
9 & \quad \{\{0, 1\}, \{n - 1, n - 2\}\}
\end{align*}
\]

\[
\begin{align*}
\quad r_2(L(n)) &= n - 1 \quad (n \equiv 0 \pmod{2}) \\
\quad r_2(L(n)) &= n \quad (n \equiv 1 \pmod{2}) \\
\quad r(L(n)) &= n \quad (n \equiv 0 \pmod{2}) \\
\quad r(L(n)) &= n + 1 \quad (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof I)

- Show: \( r_2(L(n)) \leq n - 1 \) for \( n \) even.

- Consider algorithm \( A \), given by the following matchings:

\[
\begin{align*}
1 & \quad \{\{0, 1\}, \{n - 1, n - 2\}\}, \\
2 & \quad \{\{1, 2\}, \{n - 2, n - 3\}\}, \\
3 & \quad \{\{2, 3\}, \{n - 3, n - 4\}\}, \\
4 & \quad \ldots \\
5 & \quad \{\{n/2 - 1, n/2\}\} \\
6 & \quad \ldots \\
7 & \quad \{\{2, 3\}, \{n - 3, n - 4\}\}, \\
8 & \quad \{\{1, 2\}, \{n - 2, n - 3\}\}, \\
9 & \quad \{\{0, 1\}, \{n - 1, n - 2\}\}
\end{align*}
\]
Gossip on Lines (Proof I)

- Show: $r_2(L(n)) \leq n - 1$ for $n$ even.
- Consider algorithm $A$, given by the following matchings:

1. $\{\{0, 1\}, \{n - 1, n - 2\}\}$,
2. $\{\{1, 2\}, \{n - 2, n - 3\}\}$,
3. $\{\{2, 3\}, \{n - 3, n - 4\}\}$,
4.  
5. $\{\{n/2 - 1, n/2\}\}$
6.  
7. $\{\{2, 3\}, \{n - 3, n - 4\}\}$,
8. $\{\{1, 2\}, \{n - 2, n - 3\}\}$,
9. $\{\{0, 1\}, \{n - 1, n - 2\}\}$

\[
\begin{align*}
    r_2(L(n)) &= n - 1 \quad (n \equiv 0 \pmod{2}) \\
    r_2(L(n)) &= n \quad (n \equiv 1 \pmod{2}) \\
    r(L(n)) &= n \quad (n \equiv 0 \pmod{2}) \\
    r(L(n)) &= n + 1 \quad (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof I)

- **Show:** \( r_2(L(n)) \leq n - 1 \) for \( n \) even.

- **Consider algorithm \( A \), given by the following matchings:**

  1. \( \{0, 1\}, \{n - 1, n - 2\} \}
  2. \( \{1, 2\}, \{n - 2, n - 3\} \}
  3. \( \{2, 3\}, \{n - 3, n - 4\} \}
  4. \( \ldots \)
  5. \( \{n/2 - 1, n/2\} \)
  6. \( \ldots \)
  7. \( \{2, 3\}, \{n - 3, n - 4\} \}
  8. \( \{1, 2\}, \{n - 2, n - 3\} \}
  9. \( \{0, 1\}, \{n - 1, n - 2\} \)

\[
\begin{align*}
r_2(L(n)) &= n - 1 \quad (n \equiv 0 \pmod{2}) \\
r_2(L(n)) &= n \quad (n \equiv 1 \pmod{2}) \\
r(L(n)) &= n \quad (n \equiv 0 \pmod{2}) \\
r(L(n)) &= n + 1 \quad (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof I)

- Show: \( r_2(L(n)) \leq n - 1 \) for \( n \) even.

- Consider algorithm \( A \), given by the following matchings:

  1. \( \{\{0, 1\}, \{n - 1, n - 2\}\} \),
  2. \( \{\{1, 2\}, \{n - 2, n - 3\}\} \),
  3. \( \{\{2, 3\}, \{n - 3, n - 4\}\} \),
  4. \( \ldots \)
  5. \( \{\{n/2 - 1, n/2\}\} \)
  6. \( \ldots \)
  7. \( \{\{2, 3\}, \{n - 3, n - 4\}\} \),
  8. \( \{\{1, 2\}, \{n - 2, n - 3\}\} \),
  9. \( \{\{0, 1\}, \{n - 1, n - 2\}\} \)
Gossip on Lines (Proof I)

- Show: $r_2(L(n)) \leq n - 1$ for $n$ even.

- Consider algorithm $A$, given by the following matchings:

1. $\{\{0, 1\}, \{n - 1, n - 2\}\}$,
2. $\{\{1, 2\}, \{n - 2, n - 3\}\}$,
3. $\{\{2, 3\}, \{n - 3, n - 4\}\}$,
4. $\ldots$
5. $\{\{n/2 - 1, n/2\}\}$
6. $\ldots$
7. $\{\{2, 3\}, \{n - 3, n - 4\}\}$,
8. $\{\{1, 2\}, \{n - 2, n - 3\}\}$,
9. $\{\{0, 1\}, \{n - 1, n - 2\}\}$

\[
\begin{align*}
r_2(L(n)) &= n - 1 & (n \equiv 0 \pmod{2}) \\
r_2(L(n)) &= n & (n \equiv 1 \pmod{2}) \\
r(L(n)) &= n & (n \equiv 0 \pmod{2}) \\
r(L(n)) &= n + 1 & (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof I)

- **Show:** $r_2(L(n)) \leq n - 1$ for $n$ even.

- **Consider algorithm $A$, given by the following matchings:**

  1. $\{\{0, 1\}, \{n - 1, n - 2\}\}$,
  2. $\{\{1, 2\}, \{n - 2, n - 3\}\}$,
  3. $\{\{2, 3\}, \{n - 3, n - 4\}\}$,
  4. $\ldots$
  5. $\{\{n/2 - 1, n/2\}\}$
  6. $\ldots$
  7. $\{\{2, 3\}, \{n - 3, n - 4\}\}$,
  8. $\{\{1, 2\}, \{n - 2, n - 3\}\}$,
  9. $\{\{0, 1\}, \{n - 1, n - 2\}\}$
Gossip on Lines (Proof I)

- Show: \( r_2(L(n)) \leq n - 1 \) for \( n \) even.

- Consider algorithm \( A \), given by the following matchings:

1. \( \{\{0, 1\}, \{n - 1, n - 2\}\} \),
2. \( \{\{1, 2\}, \{n - 2, n - 3\}\} \),
3. \( \{\{2, 3\}, \{n - 3, n - 4\}\} \),
4. \( \ldots \)
5. \( \{\{n/2 - 1, n/2\}\} \)
6. \( \ldots \)
7. \( \{\{2, 3\}, \{n - 3, n - 4\}\} \),
8. \( \{\{1, 2\}, \{n - 2, n - 3\}\} \),
9. \( \{\{0, 1\}, \{n - 1, n - 2\}\} \)
Gossip on Lines (Proof I)

• Show: \( r_2(L(n)) \leq n - 1 \) for \( n \) even.

• Consider algorithm \( A \), given by the following matchings:

\[
\begin{align*}
&1. \{\{0, 1\}, \{n - 1, n - 2\}\}, \\
&2. \{\{1, 2\}, \{n - 2, n - 3\}\}, \\
&3. \{\{2, 3\}, \{n - 3, n - 4\}\}, \\
&4. \ldots \\
&5. \{\{n/2 - 1, n/2\}\} \\
&6. \ldots \\
&7. \{\{2, 3\}, \{n - 3, n - 4\}\}, \\
&8. \{\{1, 2\}, \{n - 2, n - 3\}\}, \\
&9. \{\{0, 1\}, \{n - 1, n - 2\}\}
\end{align*}
\]
Gossip on Lines (Proof I)

- Show: \( r_2(L(n)) \leq n - 1 \) for \( n \) even.

- Consider algorithm \( A \), given by the following matchings:

1. \{\{0, 1\}, \{n - 1, n - 2\}\},
2. \{\{1, 2\}, \{n - 2, n - 3\}\},
3. \{\{2, 3\}, \{n - 3, n - 4\}\},
4. \ldots
5. \{\{n/2 - 1, n/2\}\}
6. \ldots
7. \{\{2, 3\}, \{n - 3, n - 4\}\},
8. \{\{1, 2\}, \{n - 2, n - 3\}\},
9. \{\{0, 1\}, \{n - 1, n - 2\}\}

\[
\begin{align*}
r_2(L(n)) &= n - 1 \quad (n \equiv 0 \pmod{2}) \\
r_2(L(n)) &= n \quad (n \equiv 1 \pmod{2}) \\
r(L(n)) &= n \quad (n \equiv 0 \pmod{2}) \\
r(L(n)) &= n + 1 \quad (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof II)

- Show: \( r_2(L(n)) \leq n \) for \( n \) odd.

- Consider algorithm \( A \), given by the following matchings:

1. \( \{0, 1\} \),
2. \( \{1, 2\}, \{n - 1, n - 2\} \),
3. \( \{2, 3\}, \{n - 2, n - 3\} \),
4. \( \ldots \)
5. \( \{\lfloor n/2 \rfloor, \lceil n/2 \rceil \} \)
6. \( \ldots \)
7. \( \{2, 3\}, \{n - 2, n - 3\} \),
8. \( \{1, 2\}, \{n - 1, n - 2\} \),
9. \( \{0, 1\} \)
Gossip on Lines (Proof II)

- **Show:** $r_2(L(n)) \leq n$ for $n$ odd.

- **Consider algorithm $A$, given by the following matchings:**

  1. $\{0, 1\}$
  2. $\{1, 2\}, \{n - 1, n - 2\}$
  3. $\{2, 3\}, \{n - 2, n - 3\}$
  4. ...
  5. $\{\lfloor n/2 \rfloor, \lceil n/2 \rceil\}$
  6. ...
  7. $\{2, 3\}, \{n - 2, n - 3\}$
  8. $\{1, 2\}, \{n - 1, n - 2\}$
  9. $\{0, 1\}$

$$
\begin{align*}
r_2(L(n)) &= n - 1 \quad (n \equiv 0 \pmod{2}) \\
r_2(L(n)) &= n \quad (n \equiv 1 \pmod{2}) \\
r(L(n)) &= n \quad (n \equiv 0 \pmod{2}) \\
r(L(n)) &= n + 1 \quad (n \equiv 1 \pmod{2})
\end{align*}$$
Gossip on Lines (Proof II)

- Show: $r_2(L(n)) \leq n$ for $n$ odd.
- Consider algorithm $A$, given by the following matchings:

1. $\{0, 1\}$
2. $\{1, 2\}, \{n - 1, n - 2\}$
3. $\{2, 3\}, \{n - 2, n - 3\}$
4. ...
5. $\{\lfloor n/2 \rfloor, \lceil n/2 \rceil\}$
6. ...
7. $\{2, 3\}, \{n - 2, n - 3\}$
8. $\{1, 2\}, \{n - 1, n - 2\}$
9. $\{0, 1\}$
Gossip on Lines (Proof II)

- Show: $r_2(L(n)) \leq n$ for $n$ odd.

- Consider algorithm $A$, given by the following matchings:

1. $\{\{0, 1\}\}$
2. $\{\{1, 2\}, \{n-1, n-2\}\}$
3. $\{\{2, 3\}, \{n-2, n-3\}\}$
4. $\ldots$
5. $\{\{\lfloor n/2 \rfloor, \lceil n/2 \rceil\}\}$
6. $\ldots$
7. $\{\{2, 3\}, \{n-2, n-3\}\}$
8. $\{\{1, 2\}, \{n-1, n-2\}\}$
9. $\{\{0, 1\}\}$
Gossip on Lines (Proof II)

- Show: $r_2(L(n)) \leq n$ for $n$ odd.

- Consider algorithm $A$, given by the following matchings:

  1. $\{\{0, 1\}\}$
  2. $\{\{1, 2\}, \{n - 1, n - 2\}\}$
  3. $\{\{2, 3\}, \{n - 2, n - 3\}\}$
  4. ...
  5. $\{\{\lfloor n/2 \rfloor, \lceil n/2 \rceil\}\}$
  6. ...
  7. $\{\{2, 3\}, \{n - 2, n - 3\}\}$
  8. $\{\{1, 2\}, \{n - 1, n - 2\}\}$
  9. $\{\{0, 1\}\}$

\[
\begin{align*}
r_2(L(n)) &= n - 1 \quad (n \equiv 0 \pmod{2}) \\
r_2(L(n)) &= n \quad (n \equiv 1 \pmod{2}) \\
r(L(n)) &= n \quad (n \equiv 0 \pmod{2}) \\
r(L(n)) &= n + 1 \quad (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof II)

- Show: $r_2(L(n)) \leq n$ for $n$ odd.

- Consider algorithm $A$, given by the following matchings:

1. $\{0, 1\}$,
2. $\{1, 2\}, \{n - 1, n - 2\}$,
3. $\{2, 3\}, \{n - 2, n - 3\}$,
4. $\ldots$
5. $\{\lfloor n/2 \rfloor, \lceil n/2 \rceil\}$
6. $\ldots$
7. $\{2, 3\}, \{n - 2, n - 3\}$,
8. $\{1, 2\}, \{n - 1, n - 2\}$,
9. $\{0, 1\}$

\[
\begin{align*}
r_2(L(n)) &= n - 1 \quad (n \equiv 0 \pmod{2}) \\
r_2(L(n)) &= n \quad (n \equiv 1 \pmod{2}) \\
r(L(n)) &= n \quad (n \equiv 0 \pmod{2}) \\
r(L(n)) &= n + 1 \quad (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof II)

- Show: \( r_2(L(n)) \leq n \) for \( n \) odd.
- Consider algorithm \( A \), given by the following matchings:

1. \( \{0, 1\} \),
2. \( \{1, 2\}, \{n - 1, n - 2\} \),
3. \( \{2, 3\}, \{n - 2, n - 3\} \),
4. \( \ldots \)
5. \( \{\lfloor n/2 \rfloor, \lceil n/2 \rceil\} \)
6. \( \ldots \)
7. \( \{2, 3\}, \{n - 2, n - 3\} \),
8. \( \{1, 2\}, \{n - 1, n - 2\} \),
9. \( \{0, 1\} \)

\[
\begin{align*}
r_2(L(n)) &= n - 1 \quad (n \equiv 0 \pmod{2}) \\
r_2(L(n)) &= n \quad (n \equiv 1 \pmod{2}) \\
r(L(n)) &= n \quad (n \equiv 0 \pmod{2}) \\
r(L(n)) &= n + 1 \quad (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof II)

- Show: \( r_2(L(n)) \leq n \) for \( n \) odd.
- Consider algorithm \( A \), given by the following matchings:

  1. \( \{0, 1\} \),
  2. \( \{1, 2\}, \{n - 1, n - 2\} \),
  3. \( \{2, 3\}, \{n - 2, n - 3\} \),
  4. \[ \ldots \]
  5. \( \{\lfloor n/2 \rfloor, \lceil n/2 \rceil\} \)
  6. \[ \ldots \]
  7. \( \{2, 3\}, \{n - 2, n - 3\} \),
  8. \( \{1, 2\}, \{n - 1, n - 2\} \),
  9. \( \{0, 1\} \)

\[
\begin{align*}
r_2(L(n)) &= n - 1 & (n \equiv 0 \pmod 2) \\
r_2(L(n)) &= n & (n \equiv 1 \pmod 2) \\
r(L(n)) &= n & (n \equiv 0 \pmod 2) \\
r(L(n)) &= n + 1 & (n \equiv 1 \pmod 2)
\end{align*}
\]
Gossip on Lines (Proof II)

- Show: $r_2(L(n)) \leq n$ for $n$ odd.
- Consider algorithm $A$, given by the following matchings:

| 1 | $\{\{0, 1\}\}$ |
| 2 | $\{\{1, 2\}, \{n - 1, n - 2\}\}$ |
| 3 | $\{\{2, 3\}, \{n - 2, n - 3\}\}$ |
| 4 | $\ldots$ |
| 5 | $\{\{\lfloor n/2 \rfloor, \lceil n/2 \rceil\}\}$ |
| 6 | $\ldots$ |
| 7 | $\{\{2, 3\}, \{n - 2, n - 3\}\}$ |
| 8 | $\{\{1, 2\}, \{n - 1, n - 2\}\}$ |
| 9 | $\{\{0, 1\}\}$ |
Gossip on Lines (Proof II)

- Show: \( r_2(L(n)) \leq n \) for \( n \) odd.

- Consider algorithm \( A \), given by the following matchings:

\[
\begin{align*}
1 & \quad \{\{0, 1\}\}, \\
2 & \quad \{\{1, 2\}, \{n-1, n-2\}\}, \\
3 & \quad \{\{2, 3\}, \{n-2, n-3\}\}, \\
4 & \quad \ldots \\
5 & \quad \{\{\lfloor n/2 \rfloor, \lceil n/2 \rceil\}\} \\
6 & \quad \ldots \\
7 & \quad \{\{2, 3\}, \{n-2, n-3\}\}, \\
8 & \quad \{\{1, 2\}, \{n-1, n-2\}\}, \\
9 & \quad \{\{0, 1\}\}
\end{align*}
\]

\[
\begin{align*}
r_2(L(n)) &= n - 1 & (n \equiv 0 \pmod{2}) \\
r_2(L(n)) &= n & (n \equiv 1 \pmod{2}) \\
r(L(n)) &= n & (n \equiv 0 \pmod{2}) \\
r(L(n)) &= n + 1 & (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof II)

- Show: \( r_2(L(n)) \leq n \) for \( n \) odd.

- Consider algorithm \( A \), given by the following matchings:

  1. \( \{0, 1\} \)
  2. \( \{1, 2\}, \{n - 1, n - 2\} \)
  3. \( \{2, 3\}, \{n - 2, n - 3\} \)
  4. \( \ldots \)
  5. \( \{\lfloor n/2 \rfloor, \lceil n/2 \rceil\} \)
  6. \( \ldots \)
  7. \( \{2, 3\}, \{n - 2, n - 3\} \)
  8. \( \{1, 2\}, \{n - 1, n - 2\} \)
  9. \( \{0, 1\} \)
Gossip on Lines (Proof II)

- Show: \( r_2(L(n)) \leq n \) for \( n \) odd.

- Consider algorithm \( A \), given by the following matchings:

  1. \{0, 1\},
  2. \{1, 2\}, \{n - 1, n - 2\},
  3. \{2, 3\}, \{n - 2, n - 3\},
  4. \ldots
  5. \{[n/2], [n/2]\}
  6. \ldots
  7. \{2, 3\}, \{n - 2, n - 3\},
  8. \{1, 2\}, \{n - 1, n - 2\},
  9. \{0, 1\}

\[
\begin{align*}
r_2(L(n)) &= n - 1 & (n \equiv 0 \pmod{2}) \\
r_2(L(n)) &= n & (n \equiv 1 \pmod{2}) \\
r(L(n)) &= n & (n \equiv 0 \pmod{2}) \\
r(L(n)) &= n + 1 & (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof II)

Show: \( r_2(L(n)) \geq n \) for \( n \) odd.

Consider the flow of messages from the left to the right node. These could not be forwarded without delay. Because we would get a time-conflict in the center. Thus at least one messages has to be delayed. This provides the lower bound.

\[
\begin{align*}
r_2(L(n)) &= n - 1 \quad (n \equiv 0 \pmod{2}) \\
r_2(L(n)) &= n \quad (n \equiv 1 \pmod{2}) \\
r(L(n)) &= n \quad (n \equiv 0 \pmod{2}) \\
r(L(n)) &= n + 1 \quad (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof II)

- Show: $r_2(L(n)) \geq n$ for $n$ odd.
- Consider the flow of messages from the left to the right node.
  - These could not be forwarded without delay.
  - Because we would get a time-conflict in the center.
  - Thus at least one message has to be delayed.
  - This provides the lower bound.

\[ r_2(L(n)) = \begin{cases} n - 1 & (n \equiv 0 \pmod{2}) \\ n & (n \equiv 1 \pmod{2}) \end{cases} \]
\[ r(L(n)) = \begin{cases} n & (n \equiv 0 \pmod{2}) \\ n + 1 & (n \equiv 1 \pmod{2}) \end{cases} \]
Gossip on Lines (Proof II)

- Show: $r_2(L(n)) \geq n$ for $n$ odd.
- Consider the flow of messages from the left to the right node.
  - These could not be forwarded without delay.
- Because we would get a time-conflict in the center.
- Thus at least one messages has to be delayed.
- This provides the lower bound.

$$
\begin{align*}
  r_2(L(n)) &= n - 1 \quad (n \equiv 0 \pmod{2}) \\
  r_2(L(n)) &= n \quad (n \equiv 1 \pmod{2}) \\
  r(L(n)) &= n \quad (n \equiv 0 \pmod{2}) \\
  r(L(n)) &= n + 1 \quad (n \equiv 1 \pmod{2})
\end{align*}
$$
Gossip on Lines (Proof II)

- Show: $r_2(L(n)) \geq n$ for $n$ odd.
- Consider the flow of messages from the left to the right node.
- These could not be forwarded without delay.
- Because we would get a time-conflict in the center.
- Thus at least one messages has to be delayed.
- This provides the lower bound.

\[
egin{align*}
  r_2(L(n)) & = n - 1 & (n \equiv 0 \pmod{2}) \\
  r_2(L(n)) & = n & (n \equiv 1 \pmod{2}) \\
  r(L(n)) & = n & (n \equiv 0 \pmod{2}) \\
  r(L(n)) & = n + 1 & (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof II)

- Show: $r_2(L(n)) \geq n$ for $n$ odd.
- Consider the flow of messages from the left to the right node.
- These could not be forwarded without delay.
- Because we would get a time-conflict in the center.
- Thus at least one message has to be delayed.
- This provides the lower bound.

\[
\begin{align*}
r_2(L(n)) &= n - 1 \quad (n \equiv 0 \pmod{2}) \\
r_2(L(n)) &= n \quad (n \equiv 1 \pmod{2}) \\
r(L(n)) &= n \quad (n \equiv 0 \pmod{2}) \\
r(L(n)) &= n + 1 \quad (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof II)

- Show: $r_2(L(n)) \geq n$ for $n$ odd.
- Consider the flow of messages from the left to the right node.
- These could not be forwarded without delay.
- Because we would get a time-conflict in the center.
- Thus at least one message has to be delayed.
- This provides the lower bound.

$$
\begin{align*}
  r_2(L(n)) &= n - 1 & (n \equiv 0 \pmod{2}) \\
  r_2(L(n)) &= n & (n \equiv 1 \pmod{2}) \\
  r(L(n)) &= n & (n \equiv 0 \pmod{2}) \\
  r(L(n)) &= n + 1 & (n \equiv 1 \pmod{2})
\end{align*}
$$
Gossip on Lines (Proof II)

- Show: \( r_2(L(n)) \geq n \) for \( n \) odd.
- Consider the flow of messages from the left to the right node.
- These could not be forwarded without delay.
- Because we would get a time-conflict in the center.
- Thus at least one message has to be delayed.
- This provides the lower bound.

\[
\begin{align*}
r_2(L(n)) &= n - 1 \quad (n \equiv 0 \pmod{2}) \\
r_2(L(n)) &= n \quad (n \equiv 1 \pmod{2}) \\
r(L(n)) &= n \quad (n \equiv 0 \pmod{2}) \\
r(L(n)) &= n + 1 \quad (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof III)

Show: \( r(L(n)) \leq n \) for \( n \) even.

Consider algorithm \( A \), given by the following matchings:

1. \( \{(0, 1), (n - 1, n - 2)\} \),
2. \( \{(1, 2), (n - 2, n - 3)\} \),
3. \( \{(2, 3), (n - 3, n - 4)\} \),
4. \( \ldots \)
5. \( \{(n/2 - 1, n/2)\} \)
6. \( \{(n/2, n/2 - 1)\} \)
7. \( \ldots \)
8. \( \{(3, 2), (n - 4, n - 3)\} \),
9. \( \{(2, 1), (n - 3, n - 2)\} \),
10. \( \{(1, 0), (n - 2, n - 1)\} \)
Gossip on Lines (Proof III)

Show: $r(L(n)) \leq n$ for $n$ even.

Consider algorithm $A$, given by the following matchings:

1. $\{(0, 1), (n - 1, n - 2)\}$,
2. $\{(1, 2), (n - 2, n - 3)\}$,
3. $\{(2, 3), (n - 3, n - 4)\}$,
4. ...
5. $\{(n/2 - 1, n/2)\}$
6. $\{(n/2, n/2 - 1)\}$
7. ...
8. $\{(3, 2), (n - 4, n - 3)\}$,
9. $\{(2, 1), (n - 3, n - 2)\}$,
10. $\{(1, 0), (n - 2, n - 1)\}$
Gossip on Lines (Proof III)

- Show: \( r(L(n)) \leq n \) for \( n \) even.

- Consider algorithm \( A \), given by the following matchings:

1. \( \{(0, 1), (n - 1, n - 2)\} \),
2. \( \{(1, 2), (n - 2, n - 3)\} \),
3. \( \{(2, 3), (n - 3, n - 4)\} \),
4. \( \ldots \)
5. \( \{(n/2 - 1, n/2)\} \)
6. \( \{(n/2, n/2 - 1)\} \)
7. \( \ldots \)
8. \( \{(3, 2), (n - 4, n - 3)\} \),
9. \( \{(2, 1), (n - 3, n - 2)\} \),
10. \( \{(1, 0), (n - 2, n - 1)\} \)
**Gossip on Lines (Proof III)**

- Show: \( r(L(n)) \leq n \) for \( n \) even.

- Consider algorithm \( A \), given by the following matchings:

\[
\begin{align*}
1 & \{ (0,1), (n-1, n-2) \}, \\
2 & \{ (1,2), (n-2, n-3) \}, \\
3 & \{ (2,3), (n-3, n-4) \}, \\
4 & \ldots \\
5 & \{ (n/2 - 1, n/2) \} \\
6 & \{ (n/2, n/2 - 1) \} \\
7 & \ldots \\
8 & \{ (3,2), (n-4, n-3) \}, \\
9 & \{ (2,1), (n-3, n-2) \}, \\
10 & \{ (1,0), (n-2, n-1) \}
\end{align*}
\]
Gossip on Lines (Proof III)

- Show: $r(L(n)) \leq n$ for $n$ even.

- Consider algorithm $A$, given by the following matchings:

\[ \begin{align*}
1 \quad & \{(0, 1), (n - 1, n - 2)\}, \\
2 \quad & \{(1, 2), (n - 2, n - 3)\}, \\
3 \quad & \{(2, 3), (n - 3, n - 4)\}, \\
4 \quad & \{(n/2 - 1, n/2)\} \\
5 \quad & \{(n/2, n/2 - 1)\} \\
6 \quad & \{(n/2, n/2 - 1)\} \\
7 \quad & \{(n/2, n/2 - 1)\} \\
8 \quad & \{(3, 2), (n - 4, n - 3)\}, \\
9 \quad & \{(2, 1), (n - 3, n - 2)\}, \\
10 \quad & \{(1, 0), (n - 2, n - 1)\}
\end{align*} \]
Gossip on Lines (Proof III)

- Show: $r(L(n)) \leq n$ for $n$ even.
- Consider algorithm $A$, given by the following matchings:

1. $\{(0,1), (n-1, n-2)\}$,
2. $\{(1,2), (n-2, n-3)\}$,
3. $\{(2,3), (n-3, n-4)\}$,
4. $\ldots$,
5. $\{(n/2 - 1, n/2)\}$
6. $\{(n/2, n/2 - 1)\}$
7. $\ldots$,
8. $\{(3,2), (n-4, n-3)\}$,
9. $\{(2,1), (n-3, n-2)\}$,
10. $\{(1,0), (n-2, n-1)\}$

$r_2(L(n)) = n - 1$ (for $n \equiv 0 \pmod{2}$)
$r_2(L(n)) = n$ (for $n \equiv 1 \pmod{2}$)
$r(L(n)) = n$ (for $n \equiv 0 \pmod{2}$)
$r(L(n)) = n + 1$ (for $n \equiv 1 \pmod{2}$)
Gossip on Lines (Proof III)

- Show: $r(L(n)) \leq n$ for $n$ even.

- Consider algorithm $A$, given by the following matchings:

  1. $\{(0, 1), (n - 1, n - 2)\}$
  2. $\{(1, 2), (n - 2, n - 3)\}$
  3. $\{(2, 3), (n - 3, n - 4)\}$
  4. ...
  5. $\{(n/2 - 1, n/2)\}$
  6. $\{(n/2, n/2 - 1)\}$
  7. ...
  8. $\{(3, 2), (n - 4, n - 3)\}$
  9. $\{(2, 1), (n - 3, n - 2)\}$
  10. $\{(1, 0), (n - 2, n - 1)\}$
Gossip on Lines (Proof III)

- Show: \( r(L(n)) \leq n \) for \( n \) even.

- Consider algorithm \( A \), given by the following matchings:

  1. \( \{(0, 1), (n - 1, n - 2)\} \),
  2. \( \{(1, 2), (n - 2, n - 3)\} \),
  3. \( \{(2, 3), (n - 3, n - 4)\} \),
  4. \( \ldots \)
  5. \( \{(n/2 - 1, n/2)\} \)
  6. \( \{(n/2, n/2 - 1)\} \)
  7. \( \ldots \)
  8. \( \{(3, 2), (n - 4, n - 3)\} \),
  9. \( \{(2, 1), (n - 3, n - 2)\} \),
  10. \( \{(1, 0), (n - 2, n - 1)\} \)
Gossip on Lines (Proof III)

- Show: $r(L(n)) \leq n$ for $n$ even.

- Consider algorithm $A$, given by the following matchings:

1. $\{(0, 1), (n - 1, n - 2)\}$
2. $\{(1, 2), (n - 2, n - 3)\}$
3. $\{(2, 3), (n - 3, n - 4)\}$
4. $\ldots$
5. $\{(n/2 - 1, n/2)\}$
6. $\{(n/2, n/2 - 1)\}$
7. $\ldots$
8. $\{(3, 2), (n - 4, n - 3)\}$
9. $\{(2, 1), (n - 3, n - 2)\}$
10. $\{(1, 0), (n - 2, n - 1)\}$

$r_2(L(n)) = n - 1 \quad (n \equiv 0 \pmod{2})$
$r_2(L(n)) = n \quad (n \equiv 1 \pmod{2})$
$r(L(n)) = n \quad (n \equiv 0 \pmod{2})$
$r(L(n)) = n + 1 \quad (n \equiv 1 \pmod{2})$
**Gossip on Lines (Proof III)**

- **Show:** $r(L(n)) \leq n$ for $n$ even.

- **Consider algorithm $A$, given by the following matchings:**
  
  1. $\{(0, 1), (n - 1, n - 2)\}$,
  2. $\{(1, 2), (n - 2, n - 3)\}$,
  3. $\{(2, 3), (n - 3, n - 4)\}$,
  4. $\ldots$
  5. $\{(n/2 - 1, n/2)\}$
  6. $\{(n/2, n/2 - 1)\}$
  7. $\ldots$
  8. $\{(3, 2), (n - 4, n - 3)\}$,
  9. $\{(2, 1), (n - 3, n - 2)\}$,
  10. $\{(1, 0), (n - 2, n - 1)\}$

\[
\begin{align*}
r_2(L(n)) &= n - 1 \quad (n \equiv 0 \pmod{2}) \\
r_2(L(n)) &= n \quad (n \equiv 1 \pmod{2}) \\
r(L(n)) &= n \quad (n \equiv 0 \pmod{2}) \\
r(L(n)) &= n + 1 \quad (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof III)

- Show: \( r(L(n)) \leq n \) for \( n \) even.

- Consider algorithm \( A \), given by the following matchings:
  1. \( \{(0, 1), (n-1, n-2)\} \),
  2. \( \{(1, 2), (n-2, n-3)\} \),
  3. \( \{(2, 3), (n-3, n-4)\} \),
  4. \( \ldots \)
  5. \( \{(n/2 - 1, n/2)\} \)
  6. \( \{(n/2, n/2 - 1)\} \)
  7. \( \ldots \)
  8. \( \{(3, 2), (n-4, n-3)\} \),
  9. \( \{(2, 1), (n-3, n-2)\} \),
  10. \( \{(1, 0), (n-2, n-1)\} \)
Gossip on Lines (Proof III)

- Show: \( r(L(n)) \leq n \) for \( n \) even.

- Consider algorithm \( A \), given by the following matchings:

\[
\begin{align*}
1 & \quad \{(0, 1), (n - 1, n - 2)\}, \\
2 & \quad \{(1, 2), (n - 2, n - 3)\}, \\
3 & \quad \{(2, 3), (n - 3, n - 4)\}, \\
4 & \quad \ldots \\
5 & \quad \{(n/2 - 1, n/2)\} \\
6 & \quad \{(n/2, n/2 - 1)\} \\
7 & \quad \ldots \\
8 & \quad \{(3, 2), (n - 4, n - 3)\}, \\
9 & \quad \{(2, 1), (n - 3, n - 2)\}, \\
10 & \quad \{(1, 0), (n - 2, n - 1)\}
\end{align*}
\]
Gossip on Lines (Proof III)

- Show: \( r(L(n)) \leq n \) for \( n \) even.

- Consider algorithm \( A \), given by the following matchings:

1. \( \{(0, 1), (n - 1, n - 2)\} \)
2. \( \{(1, 2), (n - 2, n - 3)\} \)
3. \( \{(2, 3), (n - 3, n - 4)\} \)
4. \( \ldots \)
5. \( \{(n/2 - 1, n/2)\} \)
6. \( \{(n/2, n/2 - 1)\} \)
7. \( \ldots \)
8. \( \{(3, 2), (n - 4, n - 3)\} \)
9. \( \{(2, 1), (n - 3, n - 2)\} \)
10. \( \{(1, 0), (n - 2, n - 1)\} \)
Gossip on Lines (Proof IV)

\[ r_2(L(n)) = n - 1 \quad (n \equiv 0 \pmod{2}) \]
\[ r_2(L(n)) = n \quad (n \equiv 1 \pmod{2}) \]
\[ r(L(n)) = n \quad (n \equiv 0 \pmod{2}) \]
\[ r(L(n)) = n + 1 \quad (n \equiv 1 \pmod{2}) \]

- **Show:** \( r(L(n)) \geq n \) for \( n \) even.
- The proof is similar to the above one:
  - Consider the flow of messages from the left to the right node.
  - These could not be forwarded without delay.
  - Because we would get a time-conflict in the center.
  - Thus at least one message has to be delayed.
- This provides the lower bound.
Gossip on Lines (Proof IV)

- Show: \( r(L(n)) \geq n \) for \( n \) even.
- The proof is similar to the above one:
  - Consider the flow of messages from the left to the right node.
  - These could not be forwarded without delay.
  - Because we would get a time-conflict in the center.
  - Thus at least one message has to be delayed.
- This provides the lower bound.

\[
\begin{align*}
    r_2(L(n)) &= n - 1 & (n \equiv 0 \pmod{2}) \\
    r_2(L(n)) &= n & (n \equiv 1 \pmod{2}) \\
    r(L(n)) &= n & (n \equiv 0 \pmod{2}) \\
    r(L(n)) &= n + 1 & (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof IV)

- Show: $r(L(n)) \geq n$ for $n$ even.
- The proof is similar to the above one:
  - Consider the flow of messages from the left to the right node.
  - These could not be forwarded without delay.
  - Because we would get a time-conflict in the center.
  - Thus at least one message has to be delayed.
- This provides the lower bound.

\[
\begin{align*}
r_2(L(n)) &= n - 1 & (n \equiv 0 \pmod{2}) \\
r_2(L(n)) &= n & (n \equiv 1 \pmod{2}) \\
r(L(n)) &= n & (n \equiv 0 \pmod{2}) \\
r(L(n)) &= n + 1 & (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof IV)

- Show: \( r(L(n)) \geq n \) for \( n \) even.
- The proof is similar to the above one:
- Consider the flow of messages from the left to the right node.
- These could not be forwarded without delay.
- Because we would get a time-conflict in the center.
- Thus at least one messages has to be delayed.
- This provides the lower bound.

\[
\begin{align*}
  r_2(L(n)) &= n - 1 \quad (n \equiv 0 \pmod{2}) \\
  r_2(L(n)) &= n \quad (n \equiv 1 \pmod{2}) \\
  r(L(n)) &= n \quad (n \equiv 0 \pmod{2}) \\
  r(L(n)) &= n + 1 \quad (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof IV)

- Show: \( r(L(n)) \geq n \) for \( n \) even.
- The proof is similar to the above one:
- Consider the flow of messages from the left to the right node.
- These could not be forwarded without delay.
- **Because we would get a time-conflict in the center.**
- Thus at least one messages has to be delayed.
- This provides the lower bound.

\[
\begin{align*}
r_2(L(n)) &= n - 1 \quad (n \equiv 0 \pmod{2}) \\
r_2(L(n)) &= n \quad (n \equiv 1 \pmod{2}) \\
r(L(n)) &= n \quad (n \equiv 0 \pmod{2}) \\
r(L(n)) &= n + 1 \quad (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof IV)

Show: \( r(L(n)) \geq n \) for \( n \) even.

The proof is similar to the above one:

Consider the flow of messages from the left to the right node.

These could not be forwarded without delay.

Because we would get a time-conflict in the center.

Thus at least one messages has to be delayed.

This provides the lower bound.

\[
\begin{align*}
r_2(L(n)) &= n - 1 \quad (n \equiv 0 \, (\text{mod} \, 2)) \\
r_2(L(n)) &= n \quad (n \equiv 1 \, (\text{mod} \, 2)) \\
r(L(n)) &= n \quad (n \equiv 0 \, (\text{mod} \, 2)) \\
r(L(n)) &= n + 1 \quad (n \equiv 1 \, (\text{mod} \, 2))
\end{align*}
\]
Gossip on Lines (Proof IV)

- Show: \( r(L(n)) \geq n \) for \( n \) even.
- The proof is similar to the above one:
  - Consider the flow of messages from the left to the right node.
  - These could not be forwarded without delay.
  - Because we would get a time-conflict in the center.
  - Thus at least one messages has to be delayed.
- This provides the lower bound.

\[
\begin{align*}
r_2(L(n)) & = n - 1 \quad (n \equiv 0 \text{ (mod 2)}) \\
r_2(L(n)) & = n \quad (n \equiv 1 \text{ (mod 2)}) \\
r(L(n)) & = n \quad (n \equiv 0 \text{ (mod 2)}) \\
r(L(n)) & = n + 1 \quad (n \equiv 1 \text{ (mod 2)})
\end{align*}
\]
Gossip on Lines (Proof IV)

- Show: $r(L(n)) \geq n$ for $n$ even.

- The proof is similar to the above one:
  - Consider the flow of messages from the left to the right node.
  - These could not be forwarded without delay.
  - Because we would get a time-conflict in the center.
  - Thus at least one messages has to be delayed.

- This provides the lower bound.

\[
\begin{align*}
r_2(L(n)) &= n - 1 \quad (n \equiv 0 \pmod{2}) \\
r_2(L(n)) &= n \quad (n \equiv 1 \pmod{2}) \\
r(L(n)) &= n \quad (n \equiv 0 \pmod{2}) \\
r(L(n)) &= n + 1 \quad (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof V)

- Show: $r(L(n)) \leq n + 1$ for $n$ odd.

- Consider algorithm $A$, given by the following matchings:

  1. $(0, 1)$,
  2. $(1, 2), (n - 1, n - 2)$,
  3. $(2, 3), (n - 2, n - 3)$,
  4. 
  5. $\left\{ \left\lfloor \frac{n}{2} \right\rfloor, \left\lceil \frac{n}{2} \right\rceil \right\}$
  6. $\left\{ \left\lfloor \frac{n}{2} \right\rfloor, \left\lceil \frac{n}{2} \right\rceil \right\}$
  7. 
  8. $(3, 2), (n - 3, n - 2)$,
  9. $(2, 1), (n - 2, n - 1)$,
  10. $(1, 0)$

\[
\begin{align*}
r_2(L(n)) &= n - 1 \quad (n \equiv 0 \pmod{2}) \\
r_2(L(n)) &= n \quad (n \equiv 1 \pmod{2}) \\
r(L(n)) &= n \quad (n \equiv 0 \pmod{2}) \\
r(L(n)) &= n + 1 \quad (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof V)

Show: $r(L(n)) \leq n + 1$ for $n$ odd.

Consider algorithm $A$, given by the following matchings:

1. $\{(0, 1)\}$,
2. $\{(1, 2), (n - 1, n - 2)\}$,
3. $\{(2, 3), (n - 2, n - 3)\}$,
4. $\ldots$
5. $\{([n/2], [n/2])\}$
6. $\{([n/2], [n/2])\}$
7. $\ldots$
8. $\{(3, 2), (n - 3, n - 2)\}$,
9. $\{(2, 1), (n - 2, n - 1)\}$,
10. $\{(1, 0)\}$
Gossip on Lines (Proof V)

Show: \( r(L(n)) \leq n + 1 \) for \( n \) odd.

Consider algorithm \( A \), given by the following matchings:

1. \( \{(0, 1)\} \),
2. \( \{(1, 2), (n - 1, n - 2)\} \),
3. \( \{(2, 3), (n - 2, n - 3)\} \),
4. \( \ldots \)
5. \( \{([n/2], [n/2])\} \)
6. \( \{([n/2], [n/2])\} \)
7. \( \ldots \)
8. \( \{(3, 2), (n - 3, n - 2)\} \),
9. \( \{(2, 1), (n - 2, n - 1)\} \),
10. \( \{(1, 0)\} \)
Gossip on Lines (Proof V)

- Show: \( r(L(n)) \leq n + 1 \) for \( n \) odd.

- Consider algorithm \( A \), given by the following matchings:

1. \( \{(0, 1)\} \)
2. \( \{(1, 2), (n - 1, n - 2)\} \)
3. \( \{(2, 3), (n - 2, n - 3)\} \)
4. ...
5. \( \{[\lfloor n/2 \rfloor, \lceil n/2 \rceil]\} \)
6. \( \{[\lceil n/2 \rceil, \lfloor n/2 \rfloor]\} \)
7. ...
8. \( \{(3, 2), (n - 3, n - 2)\} \)
9. \( \{(2, 1), (n - 2, n - 1)\} \)
10. \( \{(1, 0)\} \)
Gossip on Lines (Proof V)

- Show: \( r(L(n)) \leq n + 1 \) for \( n \) odd.

- Consider algorithm \( A \), given by the following matchings:

1. \( \{(0, 1)\} \),
2. \( \{(1, 2), (n - 1, n - 2)\} \),
3. \( \{(2, 3), (n - 2, n - 3)\} \),
4. \( \ldots \)
5. \( \{([n/2], [n/2])\} \)
6. \( \{([n/2], [n/2])\} \)
7. \( \ldots \)
8. \( \{(3, 2), (n - 3, n - 2)\} \),
9. \( \{(2, 1), (n - 2, n - 1)\} \),
10. \( \{(1, 0)\} \)

\[
\begin{align*}
  r_2(L(n)) &= n - 1 & (n \equiv 0 \pmod{2}) \\
  r_2(L(n)) &= n & (n \equiv 1 \pmod{2}) \\
  r(L(n)) &= n & (n \equiv 0 \pmod{2}) \\
  r(L(n)) &= n + 1 & (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof V)

Show: $r(L(n)) \leq n + 1$ for $n$ odd.

Consider algorithm $A$, given by the following matchings:

1. \{$(0, 1)$\},
2. \{$(1, 2), (n - 1, n - 2)$\},
3. \{$(2, 3), (n - 2, n - 3)$\},
4. \ldots
5. \{$(\lfloor n/2 \rfloor, \lceil n/2 \rceil)$\}
6. \{$(\lceil n/2 \rceil, \lfloor n/2 \rfloor)$\}
7. \ldots
8. \{$(3, 2), (n - 3, n - 2)$\},
9. \{$(2, 1), (n - 2, n - 1)$\},
10. \{$(1, 0)$\}

\[
\begin{align*}
  r_2(L(n)) &= n - 1 \quad (n \equiv 0 \pmod{2}) \\
  r_2(L(n)) &= n \quad (n \equiv 1 \pmod{2}) \\
  r(L(n)) &= n \quad (n \equiv 0 \pmod{2}) \\
  r(L(n)) &= n + 1 \quad (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof V)

- Show: \( r(L(n)) \leq n + 1 \) for \( n \) odd.

- Consider algorithm \( A \), given by the following matchings:
  1. \( \{(0, 1)\} \),
  2. \( \{(1, 2), (n - 1, n - 2)\} \),
  3. \( \{(2, 3), (n - 2, n - 3)\} \),
  4. \( \ldots \)
  5. \( \{([n/2], [n/2])\} \)
  6. \( \{([n/2], [n/2])\} \)
  7. \( \ldots \)
  8. \( \{(3, 2), (n - 3, n - 2)\} \),
  9. \( \{(2, 1), (n - 2, n - 1)\} \),
  10. \( \{(1, 0)\} \)

\[
\begin{align*}
r_2(L(n)) &= n - 1 \quad (n \equiv 0 \pmod{2}) \\
r_2(L(n)) &= n \quad (n \equiv 1 \pmod{2}) \\
r(L(n)) &= n \quad (n \equiv 0 \pmod{2}) \\
r(L(n)) &= n + 1 \quad (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof V)

- Show: \( r(L(n)) \leq n + 1 \) for \( n \) odd.

- Consider algorithm \( A \), given by the following matchings:
  1. \( \{(0, 1)\} \)
  2. \( \{(1, 2), (n - 1, n - 2)\} \)
  3. \( \{(2, 3), (n - 2, n - 3)\} \)
  4. \( \ldots \)
  5. \( \{[\lfloor n/2 \rfloor, \lceil n/2 \rceil]\} \)
  6. \( \{\lfloor n/2 \rfloor, \lceil n/2 \rceil\} \)
  7. \( \ldots \)
  8. \( \{(3, 2), (n - 3, n - 2)\} \)
  9. \( \{(2, 1), (n - 2, n - 1)\} \)
  10. \( \{(1, 0)\} \)

\[
\begin{align*}
r_2(L(n)) &= n - 1 \quad (n \equiv 0 \pmod{2}) \\
r_2(L(n)) &= n \quad (n \equiv 1 \pmod{2}) \\
r(L(n)) &= n \quad (n \equiv 0 \pmod{2}) \\
r(L(n)) &= n + 1 \quad (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof V)

- Show: \( r(L(n)) \leq n + 1 \) for \( n \) odd.

- Consider algorithm \( A \), given by the following matchings:

1. \( \{(0, 1)\} \),
2. \( \{(1, 2), (n-1, n-2)\} \),
3. \( \{(2, 3), (n-2, n-3)\} \),
4. 
5. \( \{([n/2], [n/2])\} \)
6. \( \{([n/2], [n/2])\} \)
7. 
8. \( \{(3, 2), (n-3, n-2)\} \),
9. \( \{(2, 1), (n-2, n-1)\} \),
10. \( \{(1, 0)\} \)
Gossip on Lines (Proof V)

Show: \( r(L(n)) \leq n + 1 \) for \( n \) odd.

Consider algorithm \( A \), given by the following matchings:

1. \( \{(0,1)\} \),
2. \( \{(1,2), (n-1, n-2)\} \),
3. \( \{(2,3), (n-2, n-3)\} \),
4. \( \ldots \)
5. \( \{([n/2], \lceil n/2 \rceil)\} \)
6. \( \{([n/2], \lfloor n/2 \rfloor)\} \)
7. \( \ldots \)
8. \( \{(3,2), (n-3, n-2)\} \),
9. \( \{(2,1), (n-2, n-1)\} \),
10. \( \{(1,0)\} \)

\[
egin{align*}
r_2(L(n)) & = n - 1 \quad (n \equiv 0 \, (\text{mod } 2)) \\
r_2(L(n)) & = n \quad (n \equiv 1 \, (\text{mod } 2)) \\
r(L(n)) & = n \quad (n \equiv 0 \, (\text{mod } 2)) \\
r(L(n)) & = n + 1 \quad (n \equiv 1 \, (\text{mod } 2))
\end{align*}
\]
Gossip on Lines (Proof V)

- **Show:** $r(L(n)) \leq n + 1$ for $n$ odd.

- **Consider algorithm $A$, given by the following matchings:**

  1. $\{(0, 1)\}$,
  2. $\{(1, 2), (n - 1, n - 2)\}$,
  3. $\{(2, 3), (n - 2, n - 3)\}$,
  4. \[
  \{(\lfloor n/2 \rfloor, \lceil n/2 \rceil)\}
  \]
  5. $\{(\lfloor n/2 \rfloor, \lceil n/2 \rceil)\}$
  6. \[
  \{(\lfloor n/2 \rfloor, \lceil n/2 \rceil)\}
  \]
  7. \[
  \{(\lfloor n/2 \rfloor, \lceil n/2 \rceil)\}
  \]
  8. $\{(3, 2), (n - 3, n - 2)\}$,
  9. $\{(2, 1), (n - 2, n - 1)\}$,
  10. $\{(1, 0)\}$

\[
\begin{align*}
  r_2(L(n)) &= n - 1 & (n \equiv 0 \pmod{2}) \\
  r_2(L(n)) &= n & (n \equiv 1 \pmod{2}) \\
  r(L(n)) &= n & (n \equiv 0 \pmod{2}) \\
  r(L(n)) &= n + 1 & (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof V)

- Show: \( r(L(n)) \leq n + 1 \) for \( n \) odd.

- Consider algorithm \( A \), given by the following matchings:
  
  1. \( \{(0, 1)\} \),
  2. \( \{(1, 2), (n - 1, n - 2)\} \),
  3. \( \{(2, 3), (n - 2, n - 3)\} \),
  4. \( \ldots \)
  5. \( \{\lfloor n/2 \rfloor, \lceil n/2 \rceil\} \)
  6. \( \{\lceil n/2 \rceil, \lfloor n/2 \rfloor\} \)
  7. \( \ldots \)
  8. \( \{(3, 2), (n - 3, n - 2)\} \),
  9. \( \{(2, 1), (n - 2, n - 1)\} \),
  10. \( \{(1, 0)\} \)
Gossip on Lines (Proof V)

- Show: $r(L(n)) \leq n + 1$ for $n$ odd.

- Consider algorithm $A$, given by the following matchings:

  1. $\{(0,1)\}$,
  2. $\{(1,2), (n-1, n-2)\}$,
  3. $\{(2,3), (n-2, n-3)\}$,
  4. $\ldots$
  5. $\{([n/2], [n/2])\}$
  6. $\{([n/2], [n/2])\}$
  7. $\ldots$
  8. $\{(3,2), (n-3, n-2)\}$,
  9. $\{(2,1), (n-2, n-1)\}$,
  10. $\{(1,0)\}$
Gossip on Lines (Proof VI)

- Show: \( r(L(n)) \geq n + 1 \) for \( n \) odd.

  - The proof is similar to the above one:
  - Consider the flow of messages from the left to the right node.
  - These could not be forwarded without delay.
  - Because we would get a time-conflict in the center.
  - Thus at least one messages (w.l.o.g. the right) has to be delayed.
  - Now the right message has to move, because otherwise we would have already a delay of two.
  - But now we still do get a further delay.
  - Thus we have proven the lower bound.
Gossip on Lines (Proof VI)

- Show: \( r(L(n)) \geq n + 1 \) for \( n \) odd.

- The proof is similar to the above one:
  - Consider the flow of messages from the left to the right node.
  - These could not be forwarded without delay.
  - Because we would get a time-conflict in the center.
  - Thus at least one message (w.l.o.g. the right) has to be delayed.
  - Now the right message has to move, because otherwise we would have already a delay of two.
  - But now we still do get a further delay.
  - Thus we have proven the lower bound.
Gossip on Lines (Proof VI)

- Show: \( r(L(n)) \geq n + 1 \) for \( n \) odd.
- The proof is similar to the above one:
- Consider the flow of messages from the left to the right node.
- These could not be forwarded without delay.
- Because we would get a time-conflict in the center.
- Thus at least one message (w.l.o.g. the right) has to be delayed.
- Now the right message has to move, because otherwise we would have already a delay of two.
- But now we still do get a further delay.
- Thus we have proven the lower bound.

\[
\begin{align*}
r_2(L(n)) &= n - 1 \quad (n \equiv 0 \pmod{2}) \\
r_2(L(n)) &= n \quad (n \equiv 1 \pmod{2}) \\
r(L(n)) &= n \quad (n \equiv 0 \pmod{2}) \\
r(L(n)) &= n + 1 \quad (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof VI)

- Show: $r(L(n)) \geq n + 1$ for $n$ odd.
- The proof is similar to the above one:
- Consider the flow of messages from the left to the right node.
- These could not be forwarded without delay.
- Because we would get a time-conflict in the center.
- Thus at least one messages (w.l.o.g. the right) has to be delayed.
- Now the right message has to move, because otherwise we would have already a delay of two.
- But now we still do get a further delay.
- Thus we have proven the lower bound.

\[
\begin{align*}
  r_2(L(n)) &= n - 1 \quad (n \equiv 0 \pmod{2}) \\
  r_2(L(n)) &= n \quad (n \equiv 1 \pmod{2}) \\
  r(L(n)) &= n \quad (n \equiv 0 \pmod{2}) \\
  r(L(n)) &= n + 1 \quad (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof VI)

- Show: \( r(L(n)) \geq n + 1 \) for \( n \) odd.
- The proof is similar to the above one:
  - Consider the flow of messages from the left to the right node.
  - These could not be forwarded without delay.
  - Because we would get a time-conflict in the center.
- Thus at least one messages (w.l.o.g. the right) has to be delayed.
- Now the right message has to move, because otherwise we would have already a delay of two.
- But now we still do get a further delay.
- Thus we have proven the lower bound.
Gossip on Lines (Proof VI)

- **Show:** \( r(L(n)) \geq n + 1 \) for \( n \) odd.

- The proof is similar to the above one:
- Consider the flow of messages from the left to the right node.
- These could not be forwarded without delay.
- Because we would get a time-conflict in the center.
- **Thus at least one messages (w.l.o.g. the right) has to be delayed.**
- Now the right message has to move, because otherwise we would have already a delay of two.
- But now we still do get a further delay.
- Thus we have proven the lower bound.

\[
\begin{align*}
r_2(L(n)) &= n - 1 \quad (n \equiv 0 \pmod{2}) \\
r_2(L(n)) &= n \quad (n \equiv 1 \pmod{2}) \\
r(L(n)) &= n \quad (n \equiv 0 \pmod{2}) \\
r(L(n)) &= n + 1 \quad (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof VI)

- Show: \( r(L(n)) \geq n + 1 \) for \( n \) odd.
- The proof is similar to the above one:
  - Consider the flow of messages from the left to the right node.
  - These could not be forwarded without delay.
  - Because we would get a time-conflict in the center.
  - Thus at least one messages (w.l.o.g. the right) has to be delayed.
  - **Now the right message has to move, because otherwise we would have already a delay of two.**
  - But now we still do get a further delay.
  - Thus we have proven the lower bound.

\[
\begin{align*}
r_2(L(n)) &= n - 1 \quad (n \equiv 0 \pmod{2}) \\
r_2(L(n)) &= n \quad (n \equiv 1 \pmod{2}) \\
r(L(n)) &= n \quad (n \equiv 0 \pmod{2}) \\
r(L(n)) &= n + 1 \quad (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof VI)

- Show: \( r(L(n)) \geq n + 1 \) for \( n \) odd.

- The proof is similar to the above one:
- Consider the flow of messages from the left to the right node.
- These could not be forwarded without delay.
- Because we would get a time-conflict in the center.
- Thus at least one message (w.l.o.g. the right) has to be delayed.
- Now the right message has to move, because otherwise we would have already a delay of two.
- But now we still do get a further delay.
- Thus we have proven the lower bound.
Gossip on Lines (Proof VI)

- Show: \( r(L(n)) \geq n + 1 \) for \( n \) odd.
- The proof is similar to the above one:
- Consider the flow of messages from the left to the right node.
- These could not be forwarded without delay.
- Because we would get a time-conflict in the center.
- Thus at least one messages (w.l.o.g. the right) has to be delayed.
- Now the right message has to move, because otherwise we would have already a delay of two.
- But now we still do get a further delay.
- Thus we have proven the lower bound.

\[
\begin{align*}
r_2(L(n)) &= n - 1 \quad (n \equiv 0 \pmod{2}) \\
r_2(L(n)) &= n \quad (n \equiv 1 \pmod{2}) \\
r(L(n)) &= n \quad (n \equiv 0 \pmod{2}) \\
r(L(n)) &= n + 1 \quad (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on Lines (Proof VI)

- Show: \( r(L(n)) \geq n + 1 \) for \( n \) odd.
- The proof is similar to the above one:
  - Consider the flow of messages from the left to the right node.
  - These could not be forwarded without delay.
  - Because we would get a time-conflict in the center.
  - Thus at least one message (w.l.o.g. the right) has to be delayed.
  - Now the right message has to move, because otherwise we would have already a delay of two.
  - But now we still do get a further delay.
  - Thus we have proven the lower bound.

\[
\begin{align*}
  r_2(L(n)) &= n - 1 \quad (n \equiv 0 \pmod{2}) \\
  r_2(L(n)) &= n \quad (n \equiv 1 \pmod{2}) \\
  r(L(n)) &= n \quad (n \equiv 0 \pmod{2}) \\
  r(L(n)) &= n + 1 \quad (n \equiv 1 \pmod{2})
\end{align*}
\]
Gossip on arbitrary Trees

Lemma:
For any tree $T$ we have:
- $r(T) = 2 \cdot \min_b(T)$
- $r_2(T) = 2 \cdot \min_b(T) - 1$

Idea of the proof:
- We have already for any graph $G$: $r(G) \leq 2 \cdot \min_b(G)$.
- We have to show: $r(G) \geq 2 \cdot \min_b(G)$.
- Let $W = \bigcup_{w \in V} I(v)$ be the total information.
- Let $A$ be any communication algorithm on $T$.
- Let $t$ be the point in time, when some node knows $W$.
- Let $v$ one node, which after $t$ steps know $W$.
- Show: at time $t$ only node $v$ knows $W$. 
Gossip on arbitrary Trees

Lemma:

For any tree $T$ we have:

- $r(T) = 2 \cdot \minb(T)$
- $r_2(T) = 2 \cdot \minb(T) - 1$

Idea of the proof:

- We have already for any graph $G$: $r(G) \leq 2 \cdot \minb(G)$.
- We have to show: $r(G) \geq 2 \cdot \minb(G)$.
- Let $W = \bigcup_{w \in V} I(v)$ be the total information.
- Let $A$ be any communication algorithm on $T$.
- Let $t$ be the point in time, when some node knows $W$.
- Let $v$ one node, which after $t$ steps know $W$.
- Show: at time $t$ only node $v$ knows $W$. 
Gossip on arbitrary Trees

Lemma:
For any tree $T$ we have:
- $r(T) = 2 \cdot \min b(T)$
- $r_2(T) = 2 \cdot \min b(T) - 1$

Idea of the proof:
- We have already for any graph $G$: $r(G) \leq 2 \cdot \min b(G)$.
- We have to show: $r(G) \geq 2 \cdot \min b(G)$.
- Let $W = \bigcup_{w \in V} I(v)$ be the total information.
- Let $A$ be any communication algorithm on $T$.
- Let $t$ be the point in time, when some node knows $W$.
- Let $v$ one node, which after $t$ steps know $W$.
- Show: at time $t$ only node $v$ knows $W$. 
Gossip on arbitrary Trees

Lemma:
For any tree $T$ we have:
- $r(T) = 2 \cdot \min b(T)$
- $r_2(T) = 2 \cdot \min b(T) - 1$

Idea of the proof:
- We have already for any graph $G$: $r(G) \leq 2 \cdot \min b(G)$.
- We have to show: $r(G) \geq 2 \cdot \min b(G)$.
- Let $W = \bigcup_{v \in V} I(v)$ be the total information.
- Let $A$ be any communication algorithm on $T$.
- Let $t$ be the point in time, when some node knows $W$.
- Let $v$ one node, which after $t$ steps know $W$.
- Show: at time $t$ only node $v$ knows $W$. 
Gossip on arbitrary Trees

Lemma:
For any tree $T$ we have:
- $r(T) = 2 \cdot \min_b(T)$
- $r_2(T) = 2 \cdot \min_b(T) - 1$

Idea of the proof:
- We have already for any graph $G$: $r(G) \leq 2 \cdot \min_b(G)$.
- We have to show: $r(G) \geq 2 \cdot \min_b(G)$.
- Let $W = \bigcup_{w \in V} I(v)$ be the total information.
- Let $A$ be any communication algorithm on $T$.
- Let $t$ be the point in time, when some node knows $W$.
- Let $v$ one node, which after $t$ steps know $W$.
- Show: at time $t$ only node $v$ knows $W$. 
Gossip on arbitrary Trees

Lemma:

For any tree $T$ we have:

- $r(T) = 2 \cdot \min_b(T)$
- $r_2(T) = 2 \cdot \min_b(T) - 1$

Idea of the proof:

- We have already for any graph $G$: $r(G) \leq 2 \cdot \min_b(G)$.
- We have to show: $r(G) \geq 2 \cdot \min_b(G)$.
- Let $W = \bigcup_{w \in V} l(v)$ be the total information.
- Let $A$ be any communication algorithm on $T$.
- Let $t$ be the point in time, when some node knows $W$.
- Let $v$ one node, which after $t$ steps know $W$.
- Show: at time $t$ only node $v$ knows $W$. 
Gossip on arbitrary Trees

Lemma:
For any tree $T$ we have:
- $r(T) = 2 \cdot \text{minb}(T)$
- $r_2(T) = 2 \cdot \text{minb}(T) - 1$

Idea of the proof:
- We have already for any graph $G$: $r(G) \leq 2 \cdot \text{minb}(G)$.
- We have to show: $r(G) \geq 2 \cdot \text{minb}(G)$.
- Let $W = \bigcup_{w \in V} I(v)$ be the total information.
- Let $A$ be any communication algorithm on $T$.
- Let $t$ be the point in time, when some node knows $W$.
- Let $v$ one node, which after $t$ steps know $W$.
- Show: at time $t$ only node $v$ knows $W$. 
Gossip on arbitrary Trees

Lemma:
For any tree $T$ we have:
- $r(T) = 2 \cdot \min b(T)$
- $r_2(T) = 2 \cdot \min b(T) - 1$

Idea of the proof:
- We have already for any graph $G$: $r(G) \leq 2 \cdot \min b(G)$.
- We have to show: $r(G) \geq 2 \cdot \min b(G)$.
- Let $W = \bigcup_{w \in V} I(v)$ be the total information.
- Let $A$ be any communication algorithm on $T$.
- Let $t$ be the point in time, when some node knows $W$.
- Let $v$ one node, which after $t$ steps know $W$.
- Show: at time $t$ only node $v$ knows $W$. 
Gossip on arbitrary Trees

Lemma:
For any tree $T$ we have:
- $r(T) = 2 \cdot \minb(T)$
- $r_2(T) = 2 \cdot \minb(T) - 1$

Idea of the proof:
- We have already for any graph $G$: $r(G) \leq 2 \cdot \minb(G)$.
- We have to show: $r(G) \geq 2 \cdot \minb(G)$.
- Let $W = \bigcup_{w \in V} I(v)$ be the total information.
- Let $A$ be any communication algorithm on $T$.
- Let $t$ be the point in time, when some node knows $W$.
- Let $v$ one node, which after $t$ steps know $W$.
- Show: at time $t$ only node $v$ knows $W$. 
Gossip on arbitrary Trees

Lemma:

For any tree $T$ we have:

- $r(T) = 2 \cdot \min b(T)$
- $r_2(T) = 2 \cdot \min b(T) - 1$

Idea of the proof:

- We have already for any graph $G$: $r(G) \leq 2 \cdot \min b(G)$.
- We have to show: $r(G) \geq 2 \cdot \min b(G)$.
- Let $W = \bigcup_{w \in V} I(v)$ be the total information.
- Let $A$ be any communication algorithm on $T$.
- Let $t$ be the point in time, when some node knows $W$.
- Let $v$ one node, which after $t$ steps know $W$.
- Show: at time $t$ only node $v$ knows $W$. 
Gossip on arbitrary Trees

Lemma:
For any tree $T$ we have:
- $r(T) = 2 \cdot \minb(T)$
- $r_2(T) = 2 \cdot \minb(T) - 1$

Idea of the proof:
- We have already for any graph $G$: $r(G) \leq 2 \cdot \minb(G)$.
- We have to show: $r(G) \geq 2 \cdot \minb(G)$.
- Let $W = \bigcup_{w \in V} l(v)$ be the total information.
- Let $A$ be any communication algorithm on $T$.
- Let $t$ be the point in time, when some node knows $W$.
- Let $v$ one node, which after $t$ steps know $W$.
- Show: at time $t$ only node $v$ knows $W$. 
Gossip on arbitrary Trees (Proof I)

Let $u \neq v$ be an other node which knows $W$ after $t$ steps.

Let $(u, y_1, y_2, \ldots, y_k, v)$ be the unique path connecting $u$ and $v$.

If $v$ sends to $y_k$ at time $t$, then $v$ did know $W$ at time $t - 1$.

So we have to consider the case: $y_k$ sends to $v$ at time $t$:

- In this case $y_k$ sends $v$ some missing information.
- $y_k$ knows at time $t - 1$ the full information, which has to be send from $y_k$ to $v$.
- The information, which has to be send from $v$ to $y_k$, is already send.
- Then the node $y_k$ know $W$ at time $t - 1$.

Contradiction, the node $u$ does not exist.

Thus we have: $t \geq \min b(T) = b(v, T)$.
Gossip on arbitrary Trees (Proof I)

• Let $u \neq v$ be an other node which knows $W$ after $t$ steps.
• Let $(u, y_1, y_2, \ldots, y_k, v)$ be the unique path connecting $u$ and $v$.
• If $v$ sends to $y_k$ at time $t$, then $v$ did know $W$ at time $t - 1$.
• So we have to consider the case: $y_k$ sends to $v$ at time $t$:
  • In this case $y_k$ sends $v$ some missing information.
  • $y_k$ knows at time $t - 1$ the full information, which has to be send from $y_k$ to $v$.
  • The information, which has to be send from $v$ to $y_k$, is already send.
  • Then the node $y_k$ know $W$ at time $t - 1$.
• Contradiction, the node $u$ does not exist.
• Thus we have: $t \geq \min_b(T) = b(v, T)$. 
Gossip on arbitrary Trees (Proof I)

- Let \( u \neq v \) be an other node which knows \( W \) after \( t \) steps.
- Let \((u, y_1, y_2, \cdots, y_k, v)\) be the unique path connecting \( u \) and \( v \).
- If \( v \) sends to \( y_k \) at time \( t \), then \( v \) did know \( W \) at time \( t - 1 \).
- So we have to consider the case: \( y_k \) sends to \( v \) at time \( t \):
  - In this case \( y_k \) sends \( v \) some missing information.
  - \( y_k \) knows at time \( t - 1 \) the full information, which has to be send from \( y_k \) to \( v \).
  - The information, which has to be send from \( v \) to \( y_k \), is already send.
  - Then the node \( y_k \) know \( W \) at time \( t - 1 \).
- Contradiction, the node \( u \) does not exist.
- Thus we have: \( t \geq \min b(T) = b(v, T) \).
Gossip on arbitrary Trees (Proof I)

- Let $u \neq v$ be an other node which knows $W$ after $t$ steps.
- Let $(u, y_1, y_2, \ldots, y_k, v)$ be the unique path connecting $u$ and $v$.
- If $v$ sends to $y_k$ at time $t$, then $v$ did know $W$ at time $t - 1$.
- **So we have to consider the case: $y_k$ sends to $v$ at time $t$:**
  - In this case $y_k$ sends $v$ some missing information.
  - $y_k$ knows at time $t - 1$ the full information, which has to be send from $y_k$ to $v$.
  - The information, which has to be send from $v$ to $y_k$, is already send.
  - Then the node $y_k$ know $W$ at time $t - 1$.
- Contradiction, the node $u$ does not exist.
- Thus we have: $t \geq \min b(T) = b(v, T)$. 
Gossip on arbitrary Trees (Proof I)

- Let $u \neq v$ be an other node which knows $W$ after $t$ steps.
- Let $(u, y_1, y_2, \cdots, y_k, v)$ be the unique path connecting $u$ and $v$.
- If $v$ sends to $y_k$ at time $t$, then $v$ did know $W$ at time $t - 1$.
- So we have to consider the case: $y_k$ sends to $v$ at time $t$:
  - In this case $y_k$ sends $v$ some missing information.
  - $y_k$ knows at time $t - 1$ the full information, which has to be send from $y_k$ to $v$.
  - The information, which has to be send from $v$ to $y_k$, is already send.
  - Then the node $y_k$ know $W$ at time $t - 1$.
- Contradiction, the node $u$ does not exist.
- Thus we have: $t \geq \min_b(T) = b(v, T)$. 
Gossip on arbitrary Trees (Proof I)

- Let \( u \neq v \) be an other node which knows \( W \) after \( t \) steps.
- Let \((u, y_1, y_2, \ldots, y_k, v)\) be the unique path connecting \( u \) and \( v \).
- If \( v \) sends to \( y_k \) at time \( t \), then \( v \) did know \( W \) at time \( t - 1 \).
- So we have to consider the case: \( y_k \) sends to \( v \) at time \( t \):
  - In this case \( y_k \) sends \( v \) some missing information.
  - \( y_k \) knows at time \( t - 1 \) the full information, which has to be send from \( y_k \) to \( v \).
  - The information, which has to be send from \( v \) to \( y_k \), is already send.
  - Then the node \( y_k \) know \( W \) at time \( t - 1 \).

- Contradiction, the node \( u \) does not exist.
- Thus we have: \( t \geq \min b(T) = b(v, T) \).
Gossip on arbitrary Trees (Proof I)

- Let \( u \neq v \) be an other node which knows \( W \) after \( t \) steps.
- Let \((u, y_1, y_2, \cdots, y_k, v)\) be the unique path connecting \( u \) and \( v \).
- If \( v \) sends to \( y_k \) at time \( t \), then \( v \) did know \( W \) at time \( t - 1 \).
- So we have to consider the case: \( y_k \) sends to \( v \) at time \( t \):
  - In this case \( y_k \) sends \( v \) some missing information.
  - \( y_k \) knows at time \( t - 1 \) the full information, which has to be send from \( y_k \) to \( v \).
    - The information, which has to be send from \( v \) to \( y_k \), is already send.
    - Then the node \( y_k \) know \( W \) at time \( t - 1 \).
- Contradiction, the node \( u \) does not exist.
- Thus we have: \( t \geq \min b(T) = b(v, T) \).
Gossip on arbitrary Trees (Proof I)

- Let $u \neq v$ be an other node which knows $W$ after $t$ steps.
- Let $(u, y_1, y_2, \ldots, y_k, v)$ be the unique path connecting $u$ and $v$.
- If $v$ sends to $y_k$ at time $t$, then $v$ did know $W$ at time $t - 1$.
- So we have to consider the case: $y_k$ sends to $v$ at time $t$:
  - In this case $y_k$ sends $v$ some missing information.
  - $y_k$ knows at time $t - 1$ the full information, which has to be send from $y_k$ to $v$.
  - The information, which has to be send from $v$ to $y_k$, is already send.
  - Then the node $y_k$ know $W$ at time $t - 1$.

- Contradiction, the node $u$ does not exist.
- Thus we have: $t \geq \min b(T) = b(v, T)$.
Gossip on arbitrary Trees (Proof I)

- Let $u \neq v$ be an other node which knows $W$ after $t$ steps.
- Let $(u, y_1, y_2, \cdots, y_k, v)$ be the unique path connecting $u$ and $v$.
- If $v$ sends to $y_k$ at time $t$, then $v$ did know $W$ at time $t - 1$.
- So we have to consider the case: $y_k$ sends to $v$ at time $t$:
  - In this case $y_k$ sends $v$ some missing information.
  - $y_k$ knows at time $t - 1$ the full information, which has to be send from $y_k$ to $v$.
  - The information, which has to be send from $v$ to $y_k$, is already send.
  - Then the node $y_k$ know $W$ at time $t - 1$.

- Contradiction, the node $u$ does not exist.
- Thus we have: $t \geq \min b(T) = b(v, T)$. 

![Diagram of a tree with nodes $u$, $y_1$, $y_2$, $y_3$, $y_k$, and $v$.]
Gossip on arbitrary Trees (Proof I)

- Let \( u \neq v \) be an other node which knows \( W \) after \( t \) steps.
- Let \( (u, y_1, y_2, \cdots, y_k, v) \) be the unique path connecting \( u \) and \( v \).
- If \( v \) sends to \( y_k \) at time \( t \), then \( v \) did know \( W \) at time \( t - 1 \).
- So we have to consider the case: \( y_k \) sends to \( v \) at time \( t \):
  - In this case \( y_k \) sends \( v \) some missing information.
  - \( y_k \) knows at time \( t - 1 \) the full information, which has to be send from \( y_k \) to \( v \).
  - The information, which has to be send from \( v \) to \( y_k \), is already send.
  - Then the node \( y_k \) know \( W \) at time \( t - 1 \).
- Contradiction, the node \( u \) does not exist.
- Thus we have: \( t \geq \min b(T) = b(v, T) \).
Gossip on arbitrary Trees (Proof I)

- Let $u \neq v$ be an other node which knows $W$ after $t$ steps.
- Let $(u, y_1, y_2, \cdots, y_k, v)$ be the unique path connecting $u$ and $v$.
- If $v$ sends to $y_k$ at time $t$, then $v$ did know $W$ at time $t - 1$.
- So we have to consider the case: $y_k$ sends to $v$ at time $t$:
  - In this case $y_k$ sends $v$ some missing information.
  - $y_k$ knows at time $t - 1$ the full information, which has to be send from $y_k$ to $v$.
  - The information, which has to be send from $v$ to $y_k$, is already send.
  - Then the node $y_k$ know $W$ at time $t - 1$.
- Contradiction, the node $u$ does not exist.
- Thus we have: $t \geq \min_b(T) = b(v, T)$. 

\[ u \xrightarrow{} y_1 \xrightarrow{} y_2 \xrightarrow{} y_3 \xrightarrow{} y_k \xrightarrow{} v \]
Gossip on arbitrary Trees (Proof II)

- Consider the situation at node $v$ after round $t$.
- Let w.l.o.g. $v$ be the root of $T$.
- Let $v_1, v_2, \ldots, v_k$ be the successors of $v$.
- Let $T_1, T_2, \ldots, T_k$ be the subtrees with roots $v_1, v_2, \ldots, v_k$.
- In each subtree $T_i$ is some information $w_i$ missing.
- Only the node $v$ knows $\bigcup_{j=1}^{k} w_j$.
- Thus there are $b(v, T)$ steps to be done.
- We finally have $r(T) \geq \min b(T) + b(v, T) \geq 2 \cdot \min b(T)$
Gossip on arbitrary Trees (Proof II)

- Consider the situation at node \( v \) after round \( t \).
- Let w.l.o.g. \( v \) be the root of \( T \).
- Let \( v_1, v_2, \ldots, v_k \) be the successors of \( v \).
- Let \( T_1, T_2, \ldots, T_k \) be the subtrees with roots \( v_1, v_2, \ldots, v_k \).
- In each subtree \( T_i \) is some information \( w_i \) missing.
- Only the node \( v \) knows \( \bigcup_{j=1}^{k} w_j \).
- Thus there are \( b(v, T) \) steps to be done.
- We finally have \( r(T) \geq \min b(T) + b(v, T) \geq 2 \cdot \min b(T) \)
Consider the situation at node $v$ after round $t$.

Let w.l.o.g. $v$ be the root of $T$.

Let $v_1, v_2, \cdots, v_k$ be the successors of $v$.

Let $T_1, T_2, \cdots, T_k$ be the subtrees with roots $v_1, v_2, \cdots, v_k$.

In each subtree $T_i$ is some information $w_i$ missing.

Only the node $v$ knows $\bigcup_{j=1}^k w_j$.

Thus there are $b(v, T)$ steps to be done.

We finally have $r(T) \geq \min b(T) + b(v, T) \geq 2 \cdot \min b(T)$
Gossip on arbitrary Trees (Proof II)

- Consider the situation at node $v$ after round $t$.
- Let w.l.o.g. $v$ be the root of $T$.
- Let $v_1, v_2, \cdots, v_k$ be the successors of $v$.
- Let $T_1, T_2, \cdots, T_k$ be the subtrees with roots $v_1, v_2, \cdots, v_k$.
- In each subtree $T_i$ is some information $w_i$ missing.
- Only the node $v$ knows $\bigcup_{j=1}^{k} w_j$.
- Thus there are $b(v, T)$ steps to be done.
- We finally have $r(T) \geq \min b(T) + b(v, T) \geq 2 \cdot \min b(T)$
Consider the situation at node $v$ after round $t$.

Let w.l.o.g. $v$ be the root of $T$.

Let $v_1, v_2, \cdots, v_k$ be the successors of $v$.

Let $T_1, T_2, \cdots, T_k$ be the subtrees with roots $v_1, v_2, \cdots, v_k$.

In each subtree $T_i$ is some information $w_i$ missing.

Only the node $v$ knows $\bigcup_{j=1}^{k} w_j$.

Thus there are $b(v, T)$ steps to be done.

We finally have $r(T) \geq \min b(T) + b(v, T) \geq 2 \cdot \min b(T)$
Gossip on arbitrary Trees (Proof II)

- Consider the situation at node $v$ after round $t$.
- Let w.l.o.g. $v$ be the root of $T$.
- Let $v_1, v_2, \cdots, v_k$ be the successors of $v$.
- Let $T_1, T_2, \cdots, T_k$ be the subtrees with roots $v_1, v_2, \cdots, v_k$.
- In each subtree $T_i$ is some information $w_i$ missing.
- Only the node $v$ knows $\bigcup_{j=1}^{k} w_j$.
- Thus there are $b(v, T)$ steps to be done.
- We finally have $r(T) \geq \min b(T) + b(v, T) \geq 2 \cdot \min b(T)$
Gossip on arbitrary Trees (Proof II)

- Consider the situation at node $v$ after round $t$.
- Let w.l.o.g. $v$ be the root of $T$.
- Let $v_1, v_2, \ldots, v_k$ be the successors of $v$.
- Let $T_1, T_2, \ldots, T_k$ be the subtrees with roots $v_1, v_2, \ldots, v_k$.
- In each subtree $T_i$ is some information $w_i$ missing.
- Only the node $v$ knows $\bigcup_{j=1}^{k} w_j$.
- Thus there are $b(v, T)$ steps to be done.
- We finally have $r(T) \geq \min b(T) + b(v, T) \geq 2 \cdot \min b(T)$
Gossip on arbitrary Trees (Proof II)

- Consider the situation at node $v$ after round $t$.
- Let w.l.o.g. $v$ be the root of $T$.
- Let $v_1, v_2, \cdots, v_k$ be the successors of $v$.
- Let $T_1, T_2, \cdots, T_k$ be the subtrees with roots $v_1, v_2, \cdots, v_k$.
- In each subtree $T_i$ is some information $w_i$ missing.
- Only the node $v$ knows $\bigcup_{j=1}^{k} w_j$.
- Thus there are $b(v, T)$ steps to be done.
- We finally have $r(T) \geq \min b(T) + b(v, T) \geq 2 \cdot \min b(T)$
Gossip on arbitrary Trees (Proof II)

- Consider the situation at node \( v \) after round \( t \).
- Let w.l.o.g. \( v \) be the root of \( T \).
- Let \( v_1, v_2, \cdots, v_k \) be the successors of \( v \).
- Let \( T_1, T_2, \cdots, T_k \) be the subtrees with roots \( v_1, v_2, \cdots, v_k \).
- In each subtree \( T_i \) is some information \( w_i \) missing.
- Only the node \( v \) knows \( \bigcup_{j=1}^{k} w_j \).
- Thus there are \( b(v, T) \) steps to be done.
- We finally have \( r(T) \geq \min b(T) + b(v, T) \geq 2 \cdot \min b(T) \)
Consider the two-way mode: by a similar way we may prove:

- At time $t$ only two neighbours nodes $u$ and $v$ know the total information. We get in the similar way the second statement.
Consider the two-way mode: by a similar way we may prove:

- At time $t$ only two neighbours nodes $u$ and $v$ know the total information. We get in the similar way the second statement.
Gossip on arbitrary Trees (Proof III)

- Consider the two-way mode: by a similar way we may prove:
- At time $t$ only two neighbours nodes $u$ and $v$ know the total information. We get in the similar way the second statement.
Consider the two-way mode: by a similar way we may prove:

At time $t$ only two neighbours nodes $u$ and $v$ know the total information. We get in the similar way the second statement.
Lemma:

For all $m \geq 1$ and $k \geq 2$ we have:

- $r(T_k(m)) = 2 \min_b(T_k(m)) = 2 \cdot k \cdot m$.
- $r_2(T_k(m)) = 2 \min_b(T_k(m)) - 1 = 2 \cdot k \cdot m - 1$. 
1-Way Gossip on Cycles (Idea)

- Messages should traverse in both directions.
- Activate each $f(n)$-th node on the cycle.
- This will result in an additional $\Theta(f(n))$ steps.
- During the distribution we get $\Theta\left(\frac{n}{2\cdot f(n)}\right)$ delays.
- Thus we will choose $f(n) = \Theta(\sqrt{n})$.
- By this idea we may get a lower and upper bound.
1-Way Gossip on Cycles (Idea)

- Messages should traverse in both directions.
- **Activate each** $f(n)$-th node on the cycle.
- This will result in an additional $\Theta(f(n))$ steps.
- During the distribution we get $\Theta\left(\frac{n}{2 \cdot f(n)}\right)$ delays.
- Thus we will choose $f(n) = \Theta(\sqrt{n})$.
- By this idea we may get a lower and upper bound.
1-Way Gossip on Cycles (Idea)

- Messages should traverse in both directions.
- Activate each $f(n)$-th node on the cycle.
- This will result in an additional $\Theta(f(n))$ steps.
- During the distribution we get $\Theta(\frac{n}{2f(n)})$ delays.
- Thus we will choose $f(n) = \Theta(\sqrt{n})$.
- By this idea we may get a lower and upper bound.
1-Way Gossip on Cycles (Idea)

- Messages should traverse in both directions.
- Activate each $f(n)$-th node on the cycle.
- This will result in an additional $\Theta(f(n))$ steps.
- During the distribution we get $\Theta(\frac{n}{2f(n)})$ delays.
- Thus we will choose $f(n) = \Theta(\sqrt{n})$.
- By this idea we may get a lower and upper bound.
1-Way Gossip on Cycles (Idea)

- Messages should traverse in both directions.
- Activate each $f(n)$-th node on the cycle.
- This will result in an additional $\Theta(f(n))$ steps.
- During the distribution we get $\Theta\left(\frac{n}{2^{f(n)}}\right)$ delays.
- Thus we will choose $f(n) = \Theta(\sqrt{n})$.
- By this idea we may get a lower and upper bound.
1-Way Gossip on Cycles (Idea)

- Messages should traverse in both directions.
- Activate each $f(n)$-th node on the cycle.
- This will result in an additional $\Theta(f(n))$ steps.
- During the distribution we get $\Theta(\frac{n}{2f(n)})$ delays.
- Thus we will choose $f(n) = \Theta(\sqrt{n})$.
- By this idea we may get a lower and upper bound.
1-Way Gossip on Cycles (Idea)

- Messages should traverse in both directions.
- Activate each $f(n)$-th node on the cycle.
- This will result in an additional $\Theta(f(n))$ steps.
- During the distribution we get $\Theta\left(\frac{n}{2\cdot f(n)}\right)$ delays.
- Thus we will choose $f(n) = \Theta(\sqrt{n})$.
- By this idea we may get a lower and upper bound.
Gossip on Cycles (Idea)
Gossip on Cycles (Idea)
Gossip on Cycles (Idea)
Gossip on Cycles (Idea)
Gossip on Cycles (Idea)
Gossip on Cycles (Idea)
Gossip on Cycles (Idea)
Gossip on Cycles (Idea)
Gossip on Cycles (Idea of the algorithm)

- Split the cycle in $\Theta(\sqrt{n})$ blocks $B_i$.
- Within block $B_i$ ($i \in \{1, 2, 3, \cdots, k\}$ with $k \in \Theta(\sqrt{n})$) do the following:
  - Phase 1:
    - The nodes $v_i [u_i]$ start a “wave” to the left [right].
    - The messages of $v_i$ and $u_i$ are delayed $\Theta(\sqrt{n})$ times by the other messages.
    - After $n/2 + \Theta(\sqrt{n})$ round know nodes $z_i$ the total information.
  - Phase 2:
    - Each node $z_i$ distribute the total information to $\Theta(\sqrt{n})$ nodes.
- Note: If $n$ is even, we have always a delay of one and the synchronization is easy.
Gossip on Cycles (Idea of the algorithm)

- Split the cycle in $\Theta(\sqrt{n})$ blocks $B_i$.
- Within block $B_i$ ($i \in \{1, 2, 3, \cdots, k\}$ with $k \in \Theta(\sqrt{n})$) do the following:
  - Phase 1:
    - The nodes $v_i[ui]$ start a “wave” to the left [right].
    - The messages of $v_i$ and $u_i$ are delayed $\Theta(\sqrt{n})$ times by the other messages.
    - After $n/2 + \Theta(\sqrt{n})$ round know nodes $z_i$ the total information.
  - Phase 2:
    - Each node $z_i$ distribute the total information to $\Theta(\sqrt{n})$ nodes.
- Note: If $n$ is even, we have always a delay of one and the synchronization is easy.
Gossip on Cycles (Idea of the algorithm)

- Split the cycle in $\Theta(\sqrt{n})$ blocks $B_i$.
- Within block $B_i$ ($i \in \{1, 2, 3, \cdots, k\}$ with $k \in \Theta(\sqrt{n})$) do the following:
  - **Phase 1:**
    - The nodes $v_i[ u_i]$ start a “wave” to the left [right].
    - The messages of $v_i$ and $u_i$ are delayed $\Theta(\sqrt{n})$ times by the other messages.
    - After $n/2 + \Theta(\sqrt{n})$ round know nodes $z_i$ the total information.
  - **Phase 2:**
    - Each node $z_i$ distribute the total information to $\Theta(\sqrt{n})$ nodes.
- Note: If $n$ is even, we have always a delay of one and the synchronization is easy.
Gossip on Cycles (Idea of the algorithm)

- Split the cycle in $\Theta(\sqrt{n})$ blocks $B_i$.
- Within block $B_i$ ($i \in \{1, 2, 3, \ldots, k\}$ with $k \in \Theta(\sqrt{n})$) do the following:
  - **Phase 1:**
    - The nodes $v_i [u_i]$ start a “wave” to the left [right].
    - The messages of $v_i$ and $u_i$ are delayed $\Theta(\sqrt{n})$ times by the other messages.
    - After $n/2 + \Theta(\sqrt{n})$ round know nodes $z_i$ the total information.
  - **Phase 2:**
    - Each node $z_i$ distribute the total information to $\Theta(\sqrt{n})$ nodes.
- Note: If $n$ is even, we have always a delay of one and the synchronization is easy.
Gossip on Cycles (Idea of the algorithm)

- Split the cycle in $\Theta(\sqrt{n})$ blocks $B_i$.
- Within block $B_i$ ($i \in \{1, 2, 3, \ldots, k\}$ with $k \in \Theta(\sqrt{n})$) do the following:
  - **Phase 1:**
    - The nodes $v_i$ [$u_i$] start a “wave” to the left [right].
    - The messages of $v_i$ and $u_i$ are delayed $\Theta(\sqrt{n})$ times by the other messages.
    - After $n/2 + \Theta(\sqrt{n})$ round know nodes $z_i$ the total information.
  - **Phase 2:**
    - Each node $z_i$ distribute the total information to $\Theta(\sqrt{n})$ nodes.
- **Note:** If $n$ is even, we have always a delay of one and the synchronization is easy.
Gossip on Cycles (Idea of the algorithm)

- Split the cycle in $\Theta(\sqrt{n})$ blocks $B_i$.
- Within block $B_i$ ($i \in \{1, 2, 3, \cdots, k\}$ with $k \in \Theta(\sqrt{n})$) do the following:
  - Phase 1:
    - The nodes $v_i \ [u_i]$ start a “wave” to the left [right].
    - The messages of $v_i$ and $u_i$ are delayed $\Theta(\sqrt{n})$ times by the other messages.
    - After $n/2 + \Theta(\sqrt{n})$ round know nodes $z_i$ the total information.
  - Phase 2:
    - Each node $z_i$ distribute the total information to $\Theta(\sqrt{n})$ nodes.
- Note: If $n$ is even, we have always a delay of one and the synchronization is easy.
Gossip on Cycles (Idea of the algorithm)

- Split the cycle in $\Theta(\sqrt{n})$ blocks $B_i$.
- Within block $B_i$ ($i \in \{1, 2, 3, \ldots, k\}$ with $k \in \Theta(\sqrt{n})$) do the following:
  - Phase 1:
    - The nodes $v_i \ [u_i]$ start a “wave” to the left [right].
    - The messages of $v_i$ and $u_i$ are delayed $\Theta(\sqrt{n})$ times by the other messages.
    - After $n/2 + \Theta(\sqrt{n})$ round know nodes $z_i$ the total information.
  - Phase 2:
    - Each node $z_i$ distribute the total information to $\Theta(\sqrt{n})$ nodes.
- Note: If $n$ is even, we have always a delay of one and the synchronization is easy.
Gossip on Cycles (Idea of the algorithm)

- Split the cycle in $\Theta(\sqrt{n})$ blocks $B_i$.
- Within block $B_i$ ($i \in \{1, 2, 3, \ldots, k\}$ with $k \in \Theta(\sqrt{n})$) do the following:
  - Phase 1:
    - The nodes $v_i [u_i]$ start a “wave” to the left [right].
    - The messages of $v_i$ and $u_i$ are delayed $\Theta(\sqrt{n})$ times by the other messages.
    - After $n/2 + \Theta(\sqrt{n})$ round know nodes $z_i$ the total information.
  - Phase 2:
    - Each node $z_i$ distribute the total information to $\Theta(\sqrt{n})$ nodes.
- Note: If $n$ is even, we have always a delay of one and the synchronization is easy.
Gossip on Cycles (Idea of the algorithm)

- Split the cycle in $\Theta(\sqrt{n})$ blocks $B_i$.
- Within block $B_i$ ($i \in \{1, 2, 3, \ldots, k\}$ with $k \in \Theta(\sqrt{n})$) do the following:
  - **Phase 1:**
    - The nodes $v_i [u_i]$ start a “wave” to the left [right].
    - The messages of $v_i$ and $u_i$ are delayed $\Theta(\sqrt{n})$ times by the other messages.
    - After $n/2 + \Theta(\sqrt{n})$ round know nodes $z_i$ the total information.
  - **Phase 2:**
    - Each node $z_i$ distribute the total information to $\Theta(\sqrt{n})$ nodes.

**Note:** If $n$ is even, we have always a delay of one and the synchronization is easy.
Gossip on Cycles (Idea of the algorithm)

- Split the cycle in $\Theta(\sqrt{n})$ blocks $B_i$.
- Within block $B_i$ ($i \in \{1, 2, 3, \cdots, k\}$ with $k \in \Theta(\sqrt{n})$) do the following:
  - Phase 1:
    - The nodes $v_i [u_i]$ start a “wave” to the left [right].
    - The messages of $v_i$ and $u_i$ are delayed $\Theta(\sqrt{n})$ times by the other messages.
    - After $n/2 + \Theta(\sqrt{n})$ round know nodes $z_i$ the total information.
  - Phase 2:
    - Each node $z_i$ distribute the total information to $\Theta(\sqrt{n})$ nodes.
- Note: If $n$ is even, we have always a delay of one and the synchronization is easy.
Gossip on Cycles (Idea)

Theorem:

We have:

- \( r(C(n)) \leq \frac{n}{2} + \sqrt{2n} - 1 \) for even \( n \).
- \( r(C(n)) \leq \lceil \frac{n}{2} \rceil + \left\lceil 2 \cdot \sqrt{\lceil \frac{n}{2} \rceil} \right\rceil - 1 \) for odd \( n \).
- \( r(C(n)) \geq \frac{n}{2} + \sqrt{2n} - 1 \) for even \( n \).
- \( r(C(n)) \geq \lceil \frac{n}{2} \rceil + \left\lceil \sqrt{2n} - 1/2 \right\rceil - 1 \) for odd \( n \).

Proof: See literature.
Gossip on the Hypercube

Theorem:
For all $m \in \mathbb{N}$ we have: $r_2(HQ(m)) = m$

Proof:
- The lower bound is the diameter.
- Upper bound by the following algorithm:
  
  for $i = 1$ to $m$ do
  
  for all $a_1, a_2, \ldots, a_{m-1} \in \{0, 1\}$ do in parallel
  
  $a_1a_2\cdots a_{i-1}0a_ia_{i+1}\cdots a_{m-1}$ sends to
  $a_1a_2\cdots a_{i-1}1a_ia_{i+1}\cdots a_{m-1}$

Corollary:
For all $m \in \mathbb{N}$ we have: $r_2(K(2^m)) = m$
**Theorem:**

For all $m \in \mathbb{N}$ we have: $r_2(HQ(m)) = m$

**Proof:**

- The lower bound is the diameter.
- Upper bound by the following algorithm:

  ```
  for $i = 1$ to $m$ do
      for all $a_1, a_2, \ldots, a_{m-1} \in \{0, 1\}$ do in parallel
          $a_1a_2\cdots a_{i-1}0a_ia_{i+1}\cdots a_{m-1}$ sends to
          $a_1a_2\cdots a_{i-1}1a_ia_{i+1}\cdots a_{m-1}$
  ```

**Corollary:**

For all $m \in \mathbb{N}$ we have: $r_2(K(2^m)) = m$
Gossip on the Hypercube

**Theorem:**

For all $m \in \mathbb{N}$ we have: $r_2(HQ(m)) = m$

**Proof:**

- The lower bound is the diameter.

- Upper bound by the following algorithm:

  ```
  for $i = 1$ to $m$ do
    for all $a_1, a_2, \ldots, a_{m-1} \in \{0, 1\}$ do in parallel
      $a_1a_2\cdots a_{i-1}0a_ia_{i+1}\cdots a_{m-1}$ sends to
      $a_1a_2\cdots a_{i-1}1a_ia_{i+1}\cdots a_{m-1}$
  ```

**Corollary:**

For all $m \in \mathbb{N}$ we have: $r_2(K(2^m)) = m$
Gossip on the Hypercube

Theorem:
For all $m \in \mathbb{N}$ we have: $r_2(HQ(m)) = m$

Proof:
- The lower bound is the diameter.
- Upper bound by the following algorithm:
  
  ```
  for $i = 1$ to $m$ do
    for all $a_1, a_2, \cdots, a_{m-1} \in \{0, 1\}$ do in parallel
      $a_1 a_2 \cdots a_{i-1} 0a_i a_{i+1} \cdots a_{m-1}$ sends to
      $a_1 a_2 \cdots a_{i-1} 1a_i a_{i+1} \cdots a_{m-1}$
  ```

Corollary:
For all $m \in \mathbb{N}$ we have: $r_2(K(2^m)) = m$
Gossip on Graphs with $2 \cdot m$ Nodes (0. Idea)
Gossip on Graphs with $2 \cdot m$ Nodes (0. Idea)
Even Number of Nodes (3:50.3)

Gossip on Graphs with $2 \cdot m$ Nodes (0. Idea)
Gossip on Graphs with $2 \cdot m$ Nodes (0. Idea)
Gossip on Graphs with $2 \cdot m$ Nodes (0. Idea)
Gossip on Graphs with $2 \cdot m$ Nodes (0. Idea)
Gossip on Graphs with $2 \cdot m$ Nodes (0. Idea)
Gossip on Graphs with $2 \cdot m$ Nodes (0. Idea)
Gossip on Graphs with $2 \cdot m$ Nodes (1. Idea)

Implication:

For all $m \in \mathbb{N}$ we have:

$$r_2(K_{2^m}) = m$$

For all $m \in \mathbb{N}$ we have:

$$r_2(K_m) \leq \lceil \log m \rceil + 1$$
Gossip on Graphs with \(2 \cdot m\) Nodes (1. Idea)
Gossip on Graphs with $2 \cdot m$ Nodes (1. Idea)
Gossip on Graphs with $2 \cdot m$ Nodes (1. Idea)

Implication:
For all $m \in \mathbb{N}$ we have:
$$r_2(K_{2m}) = m$$
For all $m \in \mathbb{N}$ we have:
$$r_2(K_m) \leq \lceil \log m \rceil + 1$$
Gossip on Graphs with $2 \cdot m$ Nodes (1. Idea)

Implication:

For all $m \in \mathbb{N}$ we have:

$$r_2(K_{2 \cdot m}) = m$$

For all $m \in \mathbb{N}$ we have:

$$r_2(K_m) \leq \lceil \log m \rceil + 1$$
Gossip on Graphs with $2 \cdot m$ Nodes (1. Idea)

Implication: For all $m \in \mathbb{N}$ we have:

$$r_2(K(2^m)) = m$$

For all $m \in \mathbb{N}$ we have:

$$r_2(K(m)) \leq \lceil \log m \rceil + 1$$
Gossip on Graphs with $2 \cdot m$ Nodes (1. Idea)

Implication:
For all $m \in \mathbb{N}$ we have:
$$r_2(K_{2 \cdot m}) = m$$

For all $m \in \mathbb{N}$ we have:
$$r_2(K_m) \leq \lceil \log_2 m \rceil + 1$$
Gossip on Graphs with $2 \cdot m$ Nodes (1. Idea)
Gossip on Graphs with $2 \cdot m$ Nodes (1. Idea)

Implication:
For all $m \in \mathbb{N}$ we have:
$$r_2(K_{2m}) = m$$

For all $m \in \mathbb{N}$ we have:
$$r_2(K_m) \leq \lceil \log m \rceil + 1$$
Gossip on Graphs with $2 \cdot m$ Nodes (1. Idea)

Implication:

For all $m \in \mathbb{N}$ we have:

$$r_2(K(2^m)) = m$$

For all $m \in \mathbb{N}$ we have:

$$r_2(K(m)) \leq \lceil \log m \rceil + 1$$
Gossip on Graphs with $2 \cdot m$ Nodes (1. Idea)

Implication:

- For all $m \in \mathbb{N}$ we have: $r_2(K(2^m)) = m$
- For all $m \in \mathbb{N}$ we have: $r_2(K(m)) \leq \lceil \log m \rceil + 1$
Gossip on Graphs with $2 \cdot m$ Nodes (2. Idea)

- Too many nodes where inactive for too long time.
- These nodes could not double their information.
- Idea: Try to double the information of any node.
- Detailed idea: In each step each node has an “interval” of information.
- To make the doubling easy split the nodes into two groups.
- Both groups should be the same size.
- In the first step pairs of node from each group share their information.
Gossip on Graphs with $2 \cdot m$ Nodes (2. Idea)

- Too many nodes where inactive for too long time.
- These nodes could not double their information.
- Idea: Try to double the information of any node.
- Detailed idea: In each step each node has an “interval” of information.
- To make the doubling easy split the nodes into two groups.
- Both groups should be the same size.
- In the first step pairs of node from each group share their information.
Gossip on Graphs with $2 \cdot m$ Nodes (2. Idea)

- Too many nodes where inactive for too long time.
- These nodes could not double their information.
- **Idea**: Try to double the information of any node.
- Detailed idea: In each step each node has an “interval” of information.
- To make the doubling easy split the nodes into two groups.
- Both groups should be the same size.
- In the first step pairs of node from each group share their information.
Gossip on Graphs with $2 \cdot m$ Nodes (2. Idea)

- Too many nodes were inactive for too long time.
- These nodes could not double their information.
- Idea: Try to double the information of any node.
- Detailed idea: In each step each node has an “interval” of information.
- To make the doubling easy split the nodes into two groups.
- Both groups should be the same size.
- In the first step pairs of nodes from each group share their information.
Gossip on Graphs with $2 \cdot m$ Nodes (2. Idea)

- Too many nodes where inactive for too long time.
- These nodes could not double their information.
- Idea: Try to double the information of any node.
- Detailed idea: In each step each node has an “interval” of information.
- To make the doubling easy split the nodes into two groups.
- Both groups should be the same size.
- In the first step pairs of node from each group share their information.
Gossip on Graphs with $2 \cdot m$ Nodes (2. Idea)

- Too many nodes where inactive for too long time.
- These nodes could not double their information.
- Idea: Try to double the information of any node.
- Detailed idea: In each step each node has an “interval” of information.
- To make the doubling easy split the nodes into two groups.
- Both groups should be the same size.
- In the first step pairs of node from each group share their information.
Gossip on Graphs with $2 \cdot m$ Nodes (2. Idea)

- Too many nodes where inactive for too long time.
- These nodes could not double their information.
- Idea: Try to double the information of any node.
- Detailed idea: In each step each node has an "interval" of information.
- To make the doubling easy split the nodes into two groups.
- Both groups should be the same size.
- In the first step pairs of node from each group share their information.
Gossip on Graphs with \( 2 \cdot m \) Nodes (2. Idea)

- Too many nodes where inactive for too long time.
- These nodes could not double their information.
- Idea: Try to double the information of any node.
- Detailed idea: In each step each node has an “interval” of information.
- To make the doubling easy split the nodes into two groups.
- Both groups should be the same size.
- In the first step pairs of node from each group share their information.
Gossip on Graphs with $2 \cdot m$ Nodes (2. Idea)
Gossip on Graphs with $2 \cdot m$ Nodes (2. Idea)
Gossip on Graphs with $2 \cdot m$ Nodes (2. Idea)
Gossip on Graphs with $2 \cdot m$ Nodes (2. Idea)
Gossip on Graphs with $2 \cdot m$ Nodes (2. Idea)
Gossip on Graphs with $2 \cdot m$ Nodes (2. Idea)
Gossip on Graphs with $2 \cdot m$ Nodes (2. Idea)
Gossip on Graphs with $2 \cdot m$ Nodes (2. Idea)
Gossip on Graphs with $2 \cdot m$ Nodes (2. Idea)
Gossip on Graphs with $2 \cdot m$ Nodes (2. Idea)
Gossip on Graphs with $2 \cdot m$ Nodes

**Theorem:**

For all $m \in \mathbb{N}$ we have: $r_2(K(2m)) = \lceil \log 2m \rceil$

**Proof:** Split the nodes in groups $Q[i]$ and $R[i]$ ($0 \leq i \leq m - 1$).

- **Algorithm:**
  
  ```
  for all $i \in \{0, \cdots, m-1\}$ do in parallel
  Exchange the information between $Q[i]$ and $R[i]$
  
  for $t = 1$ to $\lceil \log_2 m \rceil$ do
    for all $i \in \{0, \ldots, m-1\}$ do in parallel
      Exchange the information between $Q[i]$ and $R[(i + 2^{t-1}) \mod m]$
  ```

- **Invariant:**
  
  - Let $\alpha[i]$ be the information of $Q[i]$ and $R[i]$ after their initial exchange.
  - After round $t$ know nodes $Q[i]$ and $R[(i + 2^{t-1}) \mod m]$:
    \[
    \bigcup_{0 \leq j \leq 2^t - 1} \alpha[(i + j) \mod m]
    \]

- The invariant is easy to be shown.
Gossip on Graphs with $2 \cdot m$ Nodes

Theorem:
For all $m \in \mathbb{N}$ we have: $r_2(K(2m)) = \lceil \log 2m \rceil$

Proof: Split the nodes in groups $Q[i]$ and $R[i]$ ($0 \leq i \leq m − 1$).

- **algorithm:**
  
  for all $i \in \{0, \ldots, m − 1\}$ do in parallel
  
  Exchange the information between $Q[i]$ and $R[i]$

  for $t = 1$ to $\lceil \log_2 m \rceil$ do
    
    for all $i \in \{0, \ldots, m − 1\}$ do in parallel
      
      Exchange the information between $Q[i]$ and $R[(i + 2^{t−1}) \mod m]$

- **Invariant:**
  
  Let $\alpha[i]$ be the information of $Q[i]$ and $R[i]$ after their initial exchange.

  After round $t$ know nodes $Q[i]$ and $R[(i + 2^{t−1}) \mod m]$:
  
  $\bigcup_{0 \leq j \leq 2^t−1} \alpha[(i + j) \mod m]$

  The invariant is easy to be shown.
Gossip on Graphs with $2 \cdot m$ Nodes

Theorem:
For all $m \in \mathbb{N}$ we have: $r_2(K(2m)) = \lceil \log 2m \rceil$

Proof: Split the nodes in groups $Q[i]$ and $R[i]$ ($0 \leq i \leq m - 1$).

- **algorithm:**
  - for all $i \in \{0, \ldots, m-1\}$ do in parallel
    - Exchange the information between $Q[i]$ and $R[i]$
  - for $t = 1$ to $\lceil \log_2 m \rceil$ do
    - for all $i \in \{0, \ldots, m-1\}$ do in parallel
      - Exchange the information between $Q[i]$ and $R[(i + 2^{t-1}) \mod m]$

- **Invariant:**
  - Let $\alpha[i]$ be the information of $Q[i]$ and $R[i]$ after their initial exchange.
  - After round $t$ know nodes $Q[i]$ and $R[(i + 2^{t-1}) \mod m]$: $\cup_{0 \leq j \leq 2^{t-1}} \alpha[(i + j) \mod m]$
  - The invariant is easy to be shown.
Gossip on Graphs with $2 \cdot m$ Nodes

Theorem:
For all $m \in \mathbb{N}$ we have: $r_2(K(2m)) = \lceil \log 2m \rceil$

Proof: Split the nodes in groups $Q[i]$ and $R[i]$ ($0 \leq i \leq m - 1$).

- algorithm:
  
  for all $i \in \{0, \ldots, m - 1\}$ do in parallel
  
  Exchange the information between $Q[i]$ and $R[i]$

  for $t = 1$ to $\lceil \log_2 m \rceil$ do
    
    for all $i \in \{0, \ldots, m - 1\}$ do in parallel
    
    Exchange the information between $Q[i]$ and $R[(i + 2^{t-1}) \mod m]$

- Invariant:
  
  Let $\alpha[i]$ be the information of $Q[i]$ and $R[i]$ after their initial exchange.

  After round $t$ know nodes $Q[i]$ and $R[(i + 2^{t-1}) \mod m]$: $\cup_{0 \leq j \leq 2^t-1} \alpha[(i + j) \mod m]$

- The invariant is easy to be shown.
Introduction

Broadcast

Lower Bounds

Simple Graphs

Telephone-Mode

Telegraph-Mode

Sum.

Even Number of Nodes  (3:54.5)

Gossip on Graphs with $2 \cdot m$ Nodes

Theorem:

For all $m \in \mathbb{N}$ we have: $r_2(K(2m)) = \lceil \log_2 m \rceil$

Proof: Split the nodes in groups $Q[i]$ and $R[i]$ ($0 \leq i \leq m - 1$).

- **algorithm:**
  - for all $i \in \{0, \cdots, m - 1\}$ do in parallel
    - Exchange the information between $Q[i]$ and $R[i]$
  - for $t = 1$ to $\lceil \log_2 m \rceil$ do
    - for all $i \in \{0, \ldots, m - 1\}$ do in parallel
      - Exchange the information between $Q[i]$ and $R[(i + 2^{t-1}) \mod m]$

- **Invariant:**
  - Let $\alpha[i]$ be the information of $Q[i]$ and $R[i]$ after their initial exchange.
  - After round $t$ know nodes $Q[i]$ and $R[(i + 2^{t-1}) \mod m]$:
    $\cup_{0 \leq j \leq 2^t - 1} \alpha[(i + j) \mod m]$

- The invariant is easy to be shown.
Gossip on Graphs with $2 \cdot m$ Nodes

**Theorem:**

For all $m \in \mathbb{N}$ we have: $r_2(K(2m)) = \lceil \log 2m \rceil$

**Proof:** Split the nodes in groups $Q[i]$ and $R[i]$ ($0 \leq i \leq m - 1$).

- **algorithm:**
  
  ```plaintext
  for all $i \in \{0, \cdots, m - 1\}$ do in parallel
  
  Exchange the information between $Q[i]$ and $R[i]
  
  for $t = 1$ to $\lceil \log_2 m \rceil$ do
  
  for all $i \in \{0, \ldots, m - 1\}$ do in parallel
  
  Exchange the information between $Q[i]$ and $R[(i + 2^{t-1}) \mod m]$
  ```

- **Invariant:**
  
  - Let $\alpha[i]$ be the information of $Q[i]$ and $R[i]$ after their initial exchange.
  - After round $t$ know nodes $Q[i]$ and $R[(i + 2^{t-1}) \mod m]$:
    $$\cup_{0 \leq j \leq 2^{t-1}} \alpha[(i + j) \mod m]$$

  - The invariant is easy to be shown.
Gossip on Graphs with $2 \cdot m$ Nodes

**Theorem:**
For all $m \in \mathbb{N}$ we have: $r_2(K(2m)) = \lceil \log 2m \rceil$

**Proof:** Split the nodes in groups $Q[i]$ and $R[i]$ ($0 \leq i \leq m - 1$).

- **algorithm:**
  
  ```
  for all $i \in \{0, \ldots, m - 1\}$ do in parallel
  Exchange the information between $Q[i]$ and $R[i]$
  for $t = 1$ to $\lceil \log_2 m \rceil$ do
    for all $i \in \{0, \ldots, m - 1\}$ do in parallel
      Exchange the information between $Q[i]$ and $R[(i + 2^{t-1}) \mod m]$
  ```

- **Invariant:**
  
  - Let $\alpha[i]$ be the information of $Q[i]$ and $R[i]$ after their initial exchange.
  
  - After round $t$ know nodes $Q[i]$ and $R[(i + 2^{t-1}) \mod m]$: $\bigcup_{0 \leq j \leq 2^{t-1}} \alpha[(i + j) \mod m]$

  The invariant is easy to be shown.
Gossip on Graphs with $2 \cdot m + 1$ Nodes (a try)

We need an extra round.

A nice proof with this idea will become complicated.

We will try to put some structure into the proof.
Gossip on Graphs with \( 2 \cdot m + 1 \) Nodes (a try)

- We need an extra round.
- A nice proof with this idea will become complicated.
- We will try to put some structure into the proof.
Gossip on Graphs with $2 \cdot m + 1$ Nodes (a try)

- We need an extra round.
- A nice proof with this idea will become complicated.
- We will try to put some structure into the proof.
Gossip on Graphs with $2 \cdot m + 1$ Nodes (a try)

- We need an extra round.
- A nice proof with this idea will become complicated.
- We will try to put some structure into the proof.
We need an extra round.
A nice proof with this idea will become complicated.
We will try to put some structure into the proof.
Gossip on Graphs with $2 \cdot m + 1$ Nodes (a try)

- We need an extra round.
- A nice proof with this idea will become complicated.
- We will try to put some structure into the proof.
We need an extra round.

A nice proof with this idea will become complicated.

We will try to put some structure into the proof.
We need an extra round.

A nice proof with this idea will become complicated.

We will try to put some structure into the proof.
Gossip on Graphs with $2 \cdot m + 1$ Nodes (a try)

- We need an extra round.
- A nice proof with this idea will become complicated.
- We will try to put some structure into the proof.
Gossip on Graphs with $2 \cdot m + 1$ Nodes (a try)

- We need an extra round.
- A nice proof with this idea will become complicated.
- We will try to put some structure into the proof.
Gossip on Graphs with $2 \cdot m + 1$ Nodes (a try)

- We need an extra round.
- A nice proof with this idea will become complicated.
- We will try to put some structure into the proof.
Gossip on Graphs with $2 \cdot m + 1$ Nodes (a try)

- We need an extra round.
- A nice proof with this idea will become complicated.
- We will try to put some structure into the proof.
We need an extra round.

A nice proof with this idea will become complicated.

We will try to put some structure into the proof.
Gossip on Graphs with $2 \cdot m + 1$ Nodes (Idea)

- **How could this be an idea?**
- We only have the edges of the first step.
- Idea: We could now choose a small even number of Nodes, which together have the total information.
- These nodes may perform the above gossip algorithm.
- In the last step we repeat the first round.
How could this be an idea?

- We only have the edges of the first step.
- Idea: We could now choose a small even number of Nodes, which together have the total information.
- These nodes may perform the above gossip algorithm.
- In the last step we repeat the first round.
How could this be an idea?

We only have the edges of the first step.

Idea: We could now choose a small even number of Nodes, which together have the total information.

These nodes may perform the above gossip algorithm.

In the last step we repeat the first round.
Gossip on Graphs with $2 \cdot m + 1$ Nodes (Idea)

- How could this be an idea?
- We only have the edges of the first step.
- Idea: We could now choose a small even number of Nodes, which together have the total information.
- These nodes may perform the above gossip algorithm.
- In the last step we repeat the first round.
How could this be an idea?

- We only have the edges of the first step.
- Idea: We could now choose a small even number of Nodes, which together have the total information.
- These nodes may perform the above gossip algorithm.
- In the last step we repeat the first round.
Gossip on Graphs with $2 \cdot m + 1$ Nodes (Idea)

- How could this be an idea?
- We only have the edges of the first step.
- Idea: We could now choose a small even number of nodes, which together have the total information.
- These nodes may perform the above gossip algorithm.
- In the last step we repeat the first round.
Gossip on Graphs with $2 \cdot m + 1$ Nodes

- Let $n = 2 \cdot m + 1$.
- Let $v_0, v_1, v_2, \cdots, v_{n-1}$ be all nodes.
- For all $i \in \{0, 1, \cdots, m - 1\}$ the node $v_{m+2+i}$ sends to $v_i$.
- The node $\{v_0, v_1, v_2, \cdots, v_m\}$ have now the total information.
- If $m + 1$ is even, perform a gossip on the nodes $\{v_0, v_1, v_2, \cdots, v_m\}$.
- If $m + 1$ is odd, perform a gossip on the nodes $\{v_0, v_1, v_2, \cdots, v_{m+1}\}$.
- For all $i \in \{0, 1, \cdots, m - 1\}$ the nodes $v_i$ send to $v_{m+2+i}$.
- Correctness follows direct by the construction.

Running time for $m + 1$ even:
\[
 r_2(K(m + 1)) + 2 = \lfloor \log_2(m + 1) \rfloor + 2 = \lfloor \log_2 \left( \frac{n+1}{2} \right) \rfloor + 2
\]
\[
 = \lfloor \log_2(n + 1) \rfloor + 1 = \lfloor \log_2 n \rfloor + 1
\]

Running time for $m + 1$ odd:
\[
 r_2(K(m + 2)) + 2 = \lfloor \log_2(m + 2) \rfloor + 2 = \lfloor \log_2 \left( \frac{n+3}{2} \right) \rfloor + 2
\]
\[
 = \lfloor \log_2(n + 3) \rfloor + 1 = \lfloor \log_2 n \rfloor + 1
\]
Gossip on Graphs with $2 \cdot m + 1$ Nodes

- Let $n = 2 \cdot m + 1$.
- Let $v_0, v_1, v_2, \cdots, v_{n-1}$ be all nodes.
- For all $i \in \{0, 1, \cdots, m - 1\}$ the node $v_{m+2+i}$ sends to $v_i$.
- The node $\{v_0, v_1, v_2, \cdots, v_m\}$ have now the total information.
- If $m + 1$ is even, perform a gossip on the nodes $\{v_0, v_1, v_2, \cdots, v_m\}$.
- If $m + 1$ is odd, perform a gossip on the nodes $\{v_0, v_1, v_2, \cdots, v_{m+1}\}$.
- For all $i \in \{0, 1, \cdots, m - 1\}$ the nodes $v_i$ send to $v_{m+2+i}$.
- Correctness follows direct by the construction.

Running time for $m + 1$ even:

$$r_2(K(m + 1)) + 2 = \lceil \log_2(m + 1) \rceil + 2 = \lceil \log_2 \left( \frac{n+1}{2} \right) \rceil + 2$$

$$= \lceil \log_2(n + 1) \rceil + 1 = \lceil \log_2 n \rceil + 1$$

Running time for $m + 1$ odd:

$$r_2(K(m + 2)) + 2 = \lceil \log_2(m + 2) \rceil + 2 = \lceil \log_2 \left( \frac{n+3}{2} \right) \rceil + 2$$

$$= \lceil \log_2(n + 3) \rceil + 1 = \lceil \log_2 n \rceil + 1$$
Gossip on Graphs with $2 \cdot m + 1$ Nodes

- Let $n = 2 \cdot m + 1$.
- Let $v_0, v_1, v_2, \ldots, v_{n-1}$ be all nodes.
- For all $i \in \{0, 1, \ldots, m - 1\}$ the node $v_{m+2+i}$ sends to $v_i$.
- The node $\{v_0, v_1, v_2, \ldots, v_m\}$ have now the total information.
- If $m + 1$ is even, perform a gossip on the nodes $\{v_0, v_1, v_2, \ldots, v_m\}$.
- If $m + 1$ is odd, perform a gossip on the nodes $\{v_0, v_1, v_2, \ldots, v_{m+1}\}$.
- For all $i \in \{0, 1, \ldots, m - 1\}$ the nodes $v_i$ send to $v_{m+2+i}$.
- Correctness follows direct by the construction.

- Running time for $m + 1$ even:
  \[ r_2(K(m + 1)) + 2 = \left\lfloor \log_2(m + 1) \right\rfloor + 2 = \left\lfloor \log_2 \left( \frac{n+1}{2} \right) \right\rfloor + 2 = \left\lfloor \log_2 n \right\rfloor + 1 \]

- Running time for $m + 1$ odd:
  \[ r_2(K(m + 2)) + 2 = \left\lfloor \log_2(m + 2) \right\rfloor + 2 = \left\lfloor \log_2 \left( \frac{n+3}{2} \right) \right\rfloor + 2 = \left\lfloor \log_2 n \right\rfloor + 1 \]
Gossip on Graphs with $2 \cdot m + 1$ Nodes

- Let $n = 2 \cdot m + 1$.
- Let $v_0, v_1, v_2, \ldots, v_{n-1}$ be all nodes.
- For all $i \in \{0, 1, \ldots, m-1\}$ the node $v_{m+2+i}$ sends to $v_i$.
- The node $\{v_0, v_1, v_2, \ldots, v_m\}$ have now the total information.
- If $m + 1$ is even, perform a gossip on the nodes $\{v_0, v_1, v_2, \ldots, v_m\}$.
- If $m + 1$ is odd, perform a gossip on the nodes $\{v_0, v_1, v_2, \ldots, v_{m+1}\}$.
- For all $i \in \{0, 1, \ldots, m-1\}$ the nodes $v_i$ send to $v_{m+2+i}$.
- Correctness follows direct by the construction.

Running time for $m + 1$ even:

$$r_2(K(m+1)) + 2 = \left\lfloor \log_2(m+1) \right\rfloor + 2 = \left\lfloor \log_2 \left( \frac{n+1}{2} \right) \right\rfloor + 2$$

$$= \left\lfloor \log_2(n+1) \right\rfloor + 1 = \left\lfloor \log_2 n \right\rfloor + 1$$

Running time for $m + 1$ odd:

$$r_2(K(m+2)) + 2 = \left\lfloor \log_2(m+2) \right\rfloor + 2 = \left\lfloor \log_2 \left( \frac{n+3}{2} \right) \right\rfloor + 2$$

$$= \left\lfloor \log_2(n+3) \right\rfloor + 1 = \left\lfloor \log_2 n \right\rfloor + 1$$
Gossip on Graphs with $2 \cdot m + 1$ Nodes

- Let $n = 2 \cdot m + 1$.
- Let $v_0, v_1, v_2, \ldots, v_{n-1}$ be all nodes.
- For all $i \in \{0, 1, \ldots, m - 1\}$ the node $v_{m+2+i}$ sends to $v_i$.
- The node $\{v_0, v_1, v_2, \ldots, v_m\}$ have now the total information.
- If $m + 1$ is even, perform a gossip on the nodes $\{v_0, v_1, v_2, \ldots, v_m\}$.
- If $m + 1$ is odd, perform a gossip on the nodes $\{v_0, v_1, v_2, \ldots, v_{m+1}\}$.
- For all $i \in \{0, 1, \ldots, m - 1\}$ the nodes $v_i$ send to $v_{m+2+i}$.
- Correctness follows direct by the construction.

Running time for $m + 1$ even:
\[
  r_2(K(m+1)) + 2 = \lceil \log_2(m+1) \rceil + 2 = \lceil \log_2 \left(\frac{n+1}{2}\right) \rceil + 2
\]
\[
  = \lceil \log_2(n+1) \rceil + 1 = \lceil \log_2 n \rceil + 1
\]

Running time for $m + 1$ odd:
\[
  r_2(K(m+2)) + 2 = \lceil \log_2(m+2) \rceil + 2 = \lceil \log_2 \left(\frac{n+3}{2}\right) \rceil + 2
\]
\[
  = \lceil \log_2(n+3) \rceil + 1 = \lceil \log_2 n \rceil + 1
\]
Gossip on Graphs with $2 \cdot m + 1$ Nodes

- Let $n = 2 \cdot m + 1$.
- Let $v_0, v_1, v_2, \cdots, v_{n-1}$ be all nodes.
- For all $i \in \{0, 1, \cdots, m-1\}$ the node $v_{m+2+i}$ sends to $v_i$.
- The node $\{v_0, v_1, v_2, \cdots, v_m\}$ have now the total information.
- If $m + 1$ is even, perform a gossip on the nodes $\{v_0, v_1, v_2, \cdots, v_m\}$.
- If $m + 1$ is odd, perform a gossip on the nodes $\{v_0, v_1, v_2, \cdots, v_{m+1}\}$.
- For all $i \in \{0, 1, \cdots, m-1\}$ the nodes $v_i$ send to $v_{m+2+i}$.
- Correctness follows direct by the construction.

Running time for $m + 1$ even:

$$r_2(K(m+1)) + 2 = \lceil \log_2(m+1) \rceil + 2 = \lceil \log_2 \left(\frac{n+1}{2}\right) \rceil + 2$$

$$= \lceil \log_2(n+1) \rceil + 1 = \lceil \log_2 n \rceil + 1$$

Running time for $m + 1$ odd:

$$r_2(K(m+2)) + 2 = \lceil \log_2(m+2) \rceil + 2 = \lceil \log_2 \left(\frac{n+3}{2}\right) \rceil + 2$$

$$= \lceil \log_2(n+3) \rceil + 1 = \lceil \log_2 n \rceil + 1$$
Gossip on Graphs with $2 \cdot m + 1$ Nodes

- Let $n = 2 \cdot m + 1$.
- Let $v_0, v_1, v_2, \ldots, v_{n-1}$ be all nodes.
- For all $i \in \{0, 1, \ldots, m-1\}$ the node $v_{m+2+i}$ sends to $v_i$.
- The node $\{v_0, v_1, v_2, \ldots, v_m\}$ have now the total information.
- If $m + 1$ is even, perform a gossip on the nodes $\{v_0, v_1, v_2, \ldots, v_m\}$.
- If $m + 1$ is odd, perform a gossip on the nodes $\{v_0, v_1, v_2, \ldots, v_{m+1}\}$.
- For all $i \in \{0, 1, \ldots, m-1\}$ the nodes $v_i$ send to $v_{m+2+i}$.
- Correctness follows direct by the construction.

Running time for $m + 1$ even:

$$r_2(K(m + 1)) + 2 = \lceil \log_2(m + 1) \rceil + 2 = \lceil \log_2 \left(\frac{n+1}{2}\right) \rceil + 2$$

$$= \lceil \log_2(n + 1) \rceil + 1 = \lceil \log_2 n \rceil + 1$$

Running time for $m + 1$ odd:

$$r_2(K(m + 2)) + 2 = \lceil \log_2(m + 2) \rceil + 2 = \lceil \log_2 \left(\frac{n+3}{2}\right) \rceil + 2$$

$$= \lceil \log_2(n + 3) \rceil + 1 = \lceil \log_2 n \rceil + 1$$
Let $n = 2 \cdot m + 1$.

Let $v_0, v_1, v_2, \ldots, v_{n-1}$ be all nodes.

For all $i \in \{0, 1, \ldots, m-1\}$ the node $v_{m+2+i}$ sends to $v_i$.

The node $\{v_0, v_1, v_2, \ldots, v_m\}$ have now the total information.

If $m+1$ is even, perform a gossip on the nodes $\{v_0, v_1, v_2, \ldots, v_m\}$.

If $m+1$ is odd, perform a gossip on the nodes $\{v_0, v_1, v_2, \ldots, v_{m+1}\}$.

For all $i \in \{0, 1, \ldots, m-1\}$ the nodes $v_i$ send to $v_{m+2+i}$.

Correctness follows direct by the construction.

Running time for $m + 1$ even:

$$r_2(K(m+1)) + 2 = \lceil \log_2(m+1) \rceil + 2 = \lceil \log_2 \left( \frac{n+1}{2} \right) \rceil + 2$$

Running time for $m + 1$ odd:

$$r_2(K(m+2)) + 2 = \lceil \log_2(m+2) \rceil + 2 = \lceil \log_2 \left( \frac{n+3}{2} \right) \rceil + 2 = \lceil \log_2 n \rceil + 1$$
Gossip on Graphs with $2 \cdot m + 1$ Nodes

- Let $n = 2 \cdot m + 1$.
- Let $v_0, v_1, v_2, \ldots, v_{n-1}$ be all nodes.
- For all $i \in \{0, 1, \ldots, m-1\}$ the node $v_{m+1+i}$ sends to $v_i$.
- The node $\{v_0, v_1, v_2, \ldots, v_m\}$ have now the total information.
- If $m + 1$ is even, perform a gossip on the nodes $\{v_0, v_1, v_2, \ldots, v_m\}$.
- If $m + 1$ is odd, perform a gossip on the nodes $\{v_0, v_1, v_2, \ldots, v_{m+1}\}$.
- For all $i \in \{0, 1, \ldots, m-1\}$ the nodes $v_i$ send to $v_{m+1+i}$.
- Correctness follows direct by the construction.

**Running time for $m + 1$ even:**
\[
\begin{align*}
\log_2(K(m+1)) + 2 &= \lceil \log_2(2 \cdot m + 1) \rceil + 2 \\
&= \lceil \log_2((n+1)/2) \rceil + 2 \\
&= \lceil \log_2(n+1) \rceil + 1 = \lceil \log_2 n \rceil + 1
\end{align*}
\]

**Running time for $m + 1$ odd:**
\[
\begin{align*}
\log_2(K(m+2)) + 2 &= \lceil \log_2(2 \cdot m + 2) \rceil + 2 \\
&= \lceil \log_2((n+3)/2) \rceil + 2 \\
&= \lceil \log_2(n+3) \rceil + 1 = \lceil \log_2 n \rceil + 1
\end{align*}
\]
Let \( n = 2 \cdot m + 1 \).

Let \( v_0, v_1, v_2, \cdots, v_{n-1} \) be all nodes.

For all \( i \in \{0, 1, \cdots, m - 1\} \) the node \( v_{m+2+i} \) sends to \( v_i \).

The node \( \{v_0, v_1, v_2, \cdots, v_m\} \) have now the total information.

If \( m + 1 \) is even, perform a gossip on the nodes \( \{v_0, v_1, v_2, \cdots, v_m\} \).

If \( m + 1 \) is odd, perform a gossip on the nodes \( \{v_0, v_1, v_2, \cdots, v_{m+1}\} \).

For all \( i \in \{0, 1, \cdots, m - 1\} \) the nodes \( v_i \) send to \( v_{m+2+i} \).

Correctness follows direct by the construction.

Running time for \( m + 1 \) even:
\[
\begin{align*}
r_2(K(m+1)) + 2 &= \lceil \log_2(m+1) \rceil + 2 = \lceil \log_2 \left( \frac{n+1}{2} \right) \rceil + 2 \\
&= \lceil \log_2(n+1) \rceil + 1 = \lceil \log_2 n \rceil + 1
\end{align*}
\]

Running time for \( m + 1 \) odd:
\[
\begin{align*}
r_2(K(m+2)) + 2 &= \lceil \log_2(m+2) \rceil + 2 = \lceil \log_2 \left( \frac{n+3}{2} \right) \rceil + 2 \\
&= \lceil \log_2(n+3) \rceil + 1 = \lceil \log_2 n \rceil + 1
\end{align*}
\]
Gossip on Graphs with $2 \cdot m + 1$ Nodes

- Let $n = 2 \cdot m + 1$.
- Let $v_0, v_1, v_2, \cdots, v_{n-1}$ be all nodes.
- For all $i \in \{0, 1, \cdots, m - 1\}$ the node $v_{m+2+i}$ sends to $v_i$.
- The node $\{v_0, v_1, v_2, \cdots, v_m\}$ have now the total information.
- If $m + 1$ is even, perform a gossip on the nodes $\{v_0, v_1, v_2, \cdots, v_m\}$.
- If $m + 1$ is odd, perform a gossip on the nodes $\{v_0, v_1, v_2, \cdots, v_{m+1}\}$.
- For all $i \in \{0, 1, \cdots, m - 1\}$ the nodes $v_i$ send to $v_{m+2+i}$.
- Correctness follows direct by the construction.

- **Running time for $m + 1$ even:**
  \[
  r_2(K(m + 1)) + 2 = \left\lceil \log_2(m + 1) \right\rceil + 2 = \left\lceil \log_2\left(\frac{n+1}{2}\right) \right\rceil + 2
  
  = \left\lceil \log_2(n + 1) \right\rceil + 1 = \left\lceil \log_2 n \right\rceil + 1
  \]

- **Running time for $m + 1$ odd:**
  \[
  r_2(K(m + 2)) + 2 = \left\lceil \log_2(m + 2) \right\rceil + 2 = \left\lceil \log_2\left(\frac{n+3}{2}\right) \right\rceil + 2
  
  = \left\lceil \log_2(n + 3) \right\rceil + 1 = \left\lceil \log_2 n \right\rceil + 1
  \]
1st Idea (Let the Knowledge grow)

- We need more rounds.
- A nice proof with this idea will become complicated.
- We will try to put some structure into the proof.
1st Idea (Let the Knowledge grow)

- We need more rounds.
- A nice proof with this idea will become complicated.
- We will try to put some structure into the proof.
**1st Idea (Let the Knowledge grow)**

- We need more rounds.
- A nice proof with this idea will become complicated.
- We will try to put some structure into the proof.
1st Idea (Let the Knowledge grow)

- We need more rounds.
- A nice proof with this idea will become complicated.
- We will try to put some structure into the proof.
$1^{st}$ Idea (Let the Knowledge grow)

- We need more rounds.
- A nice proof with this idea will become complicated.
- We will try to put some structure into the proof.
1st Idea (Let the Knowledge grow)

- We need more rounds.
- A nice proof with this idea will become complicated.
- We will try to put some structure into the proof.
1st Idea (Let the Knowledge grow)

- We need more rounds.
- A nice proof with this idea will become complicated.
- We will try to put some structure into the proof.
We need more rounds.

A nice proof with this idea will become complicated.

We will try to put some structure into the proof.
1st Idea (Let the Knowledge grow)

- We need more rounds.
- A nice proof with this idea will become complicated.
- We will try to put some structure into the proof.
1\textsuperscript{st} Idea (Let the Knowledge grow)

- We need more rounds.
- A nice proof with this idea will become complicated.
- We will try to put some structure into the proof.
1\textsuperscript{st} Idea (Let the Knowledge grow)

- We need more rounds.
- A nice proof with this idea will become complicated.
- We will try to put some structure into the proof.
We need more rounds.

A nice proof with this idea will become complicated.

We will try to put some structure into the proof.
1st Idea (Let the Knowledge grow)

- We need more rounds.
- A nice proof with this idea will become complicated.
- We will try to put some structure into the proof.
We need an additional two rounds.

\( v_x \) and \( w_y \) alternate as sender and receiver.

The information grows in blocks (intervals) in the nodes.

With this idea we may do the proof.

Only the first two rounds are special.
2\textsuperscript{nd} Idea (Let the Knowledge grow in a structured way)

- We need an additional two rounds.
- $v_x$ and $w_y$ alternate as sender and receiver.
- The information grows in blocks (intervals) in the nodes.
- With this idea we may do the proof.
- Only the first two rounds are special.
2\textsuperscript{nd} Idea (Let the Knowledge grow in a structured way)

- We need an additional two rounds.
- \(v_x\) and \(w_y\) alternate as sender and receiver.
- The information grows in blocks (intervals) in the nodes.
- With this idea we may do the proof.
- Only the first two rounds are special.
2\textsuperscript{nd} Idea (Let the Knowledge grow in a structured way)

- We need an additional two rounds.
- $v_x$ and $w_y$ alternate as sender and receiver.
- The information grows in blocks (intervals) in the nodes.
- With this idea we may do the proof.
- Only the first two rounds are special.
2\textsuperscript{nd} Idea (Let the Knowledge grow in a structured way)

- We need an additional two rounds.
- $v_x$ and $w_y$ alternate as sender and receiver.
- The information grows in blocks (intervals) in the nodes.
- With this idea we may do the proof.
- Only the first two rounds are special.
We need an additional two rounds.

$v_x$ and $w_y$ alternate as sender and receiver.

The information grows in blocks (intervals) in the nodes.

With this idea we may do the proof.

Only the first two rounds are special.
We need an additional two rounds.

$v_x$ and $w_y$ alternate as sender and receiver.

The information grows in blocks (intervals) in the nodes.

With this idea we may do the proof.

Only the first two rounds are special.
2\textsuperscript{nd} Idea (Let the Knowledge grow in a structured way)

- We need an additional two rounds.
- \(v_x\) and \(w_y\) alternate as sender and receiver.
- The information grows in blocks (intervals) in the nodes.
- With this idea we may do the proof.
- Only the first two rounds are special.
2\textsuperscript{nd} Idea (Let the Knowledge grow in a structured way)

- We need an additional two rounds.
- \(v_x\) and \(w_y\) alternate as sender and receiver.
- The information grows in blocks (intervals) in the nodes.
- With this idea we may do the proof.
- Only the first two rounds are special.
We need an additional two rounds.

\(v_x\) and \(w_y\) alternate as sender and receiver.

The information grows in blocks (intervals) in the nodes.

With this idea we may do the proof.

Only the first two rounds are special.
We need an additional two rounds.

\(v_x\) and \(w_y\) alternate as sender and receiver.

The information grows in blocks (intervals) in the nodes.

With this idea we may do the proof.

Only the first two rounds are special.
2nd Idea (Let the Knowledge grow in a structured way)

- We need an additional two rounds.
- \(v_x\) and \(w_y\) alternate as sender and receiver.
- The information grows in blocks (intervals) in the nodes.
- With this idea we may do the proof.
- Only the first two rounds are special.
2nd Idea (Let the Knowledge grow in a structured way)

- We need an additional two rounds.
- $v_x$ and $w_y$ alternate as sender and receiver.
- The information grows in blocks (intervals) in the nodes.
- With this idea we may do the proof.
- Only the first two rounds are special.
$2^{nd}$ Idea (Let the Knowledge grow in a structured way)

- We need an additional two rounds.
- $v_x$ and $w_y$ alternate as sender and receiver.
- The information grows in blocks (intervals) in the nodes.
- With this idea we may do the proof.
- Only the first two rounds are special.
We need an additional two rounds.

\( v_x \) and \( w_y \) alternate as sender and receiver.

The information grows in blocks (intervals) in the nodes.

With this idea we may do the proof.

Only the first two rounds are special.
2\textsuperscript{nd} Idea (Let the Knowledge grow in a structured way)

- We need an additional two rounds.
- \(v_x\) and \(w_y\) alternate as sender and receiver.
- The information grows in blocks (intervals) in the nodes.
- With this idea we may do the proof.
- Only the first two rounds are special.
2\textsuperscript{nd} Idea (Let the Knowledge grow in a structured way)

- After the first two rounds some node-pairs share their information.

- Consider this situation as the start:
  - All $v_x$ and $w_x$ have one information pair.
  - $v_i$ sends to $w_j$ and the $w_x$ have 2 information pairs.
  - $w_i$ sends to $v_j$ and the $v_x$ have 3 information pairs.
  - $v_i$ sends to $w_j$ and the $w_x$ have 5 information pairs.
  - $w_i$ sends to $v_j$ and the $v_x$ have 8 information pairs.
  - $v_i$ sends to $w_j$ and the $w_x$ have 13 information pairs.
  - $w_i$ sends to $v_j$ and the $v_x$ have 21 information pairs.
  - Thus the grow-rate and the algorithm is clearly visible.
2nd Idea (Let the Knowledge grow in a structured way)

- After the first two rounds some node-pairs share their information.
- Consider this situation as the start:
  - All \( v_x \) and \( w_x \) have one information pair.
  - \( v_i \) sends to \( w_j \) and the \( w_x \) have 2 information pairs.
  - \( w_i \) sends to \( v_j \) and the \( v_x \) have 3 information pairs.
  - \( v_i \) sends to \( w_j \) and the \( w_x \) have 5 information pairs.
  - \( w_i \) sends to \( v_j \) and the \( v_x \) have 8 information pairs.
  - \( v_i \) sends to \( w_j \) and the \( w_x \) have 13 information pairs.
  - \( w_i \) sends to \( v_j \) and the \( v_x \) have 21 information pairs.
  - Thus the grow-rate and the algorithm is clearly visible.
2nd Idea (Let the Knowledge grow in a structured way)

- After the first two rounds some node-pairs share their information.

- Consider this situation as the start:
  - All $v_x$ and $w_x$ have one information pair.
  - $v_i$ sends to $w_j$ and the $w_x$ have 2 information pairs.
  - $w_i$ sends to $v_j$ and the $v_x$ have 3 information pairs.
  - $v_i$ sends to $w_j$ and the $w_x$ have 5 information pairs.
  - $w_i$ sends to $v_j$ and the $v_x$ have 8 information pairs.
  - $v_i$ sends to $w_j$ and the $w_x$ have 13 information pairs.
  - $w_i$ sends to $v_j$ and the $v_x$ have 21 information pairs.
  - Thus the grow-rate and the algorithm is clearly visible.
After the first two rounds some node-pairs share their information.

Consider this situation as the start:

- All \( v_x \) and \( w_x \) have one information pair.
- \( v_i \) sends to \( w_j \) and the \( w_x \) have 2 information pairs.
- \( w_i \) sends to \( v_j \) and the \( v_x \) have 3 information pairs.
- \( v_i \) sends to \( w_j \) and the \( w_x \) have 5 information pairs.
- \( w_i \) sends to \( v_j \) and the \( v_x \) have 8 information pairs.
- \( v_i \) sends to \( w_j \) and the \( w_x \) have 13 information pairs.
- \( w_i \) sends to \( v_j \) and the \( v_x \) have 21 information pairs.

Thus the grow-rate and the algorithm is clearly visible.
2nd Idea (Let the Knowledge grow in a structured way)

- After the first two rounds some node-pairs share their information.
- Consider this situation as the start:
  - All $v_x$ and $w_x$ have one information pair.
  - $v_i$ sends to $w_j$ and the $w_x$ have 2 information pairs.
  - $w_i$ sends to $v_j$ and the $v_x$ have 3 information pairs.
  - $v_i$ sends to $w_j$ and the $w_x$ have 5 information pairs.
  - $w_i$ sends to $v_j$ and the $v_x$ have 8 information pairs.
  - $v_i$ sends to $w_j$ and the $w_x$ have 13 information pairs.
  - $w_i$ sends to $v_j$ and the $v_x$ have 21 information pairs.
- Thus the grow-rate and the algorithm is clearly visible.
After the first two rounds some node-pairs share their information.

Consider this situation as the start:

- All $v_x$ and $w_x$ have one information pair.
- $v_i$ sends to $w_j$ and the $w_x$ have 2 information pairs.
- $w_i$ sends to $v_j$ and the $v_x$ have 3 information pairs.
- $v_i$ sends to $w_j$ and the $w_x$ have 5 information pairs.
- $w_i$ sends to $v_j$ and the $v_x$ have 8 information pairs.
- $v_i$ sends to $w_j$ and the $w_x$ have 13 information pairs.
- $w_i$ sends to $v_j$ and the $v_x$ have 21 information pairs.
- Thus the grow-rate and the algorithm is clearly visible.
2\textsuperscript{nd} Idea (Let the Knowledge grow in a structured way)

- After the first two rounds some node-pairs share their information.
- Consider this situation as the start:
  - All $v_x$ and $w_x$ have one information pair.
  - $v_i$ sends to $w_j$ and the $w_x$ have 2 information pairs.
  - $w_i$ sends to $v_j$ and the $v_x$ have 3 information pairs.
  - $v_i$ sends to $w_j$ and the $w_x$ have 5 information pairs.
  - $w_i$ sends to $v_j$ and the $v_x$ have 8 information pairs.
  - $v_i$ sends to $w_j$ and the $w_x$ have 13 information pairs.
  - $w_i$ sends to $v_j$ and the $v_x$ have 21 information pairs.
- Thus the grow-rate and the algorithm is clearly visible.
2nd Idea (Let the Knowledge grow in a structured way)

- After the first two rounds some node-pairs share their information.
- Consider this situation as the start:
  - All $v_x$ and $w_x$ have one information pair.
  - $v_i$ sends to $w_j$ and the $w_x$ have 2 information pairs.
  - $w_i$ sends to $v_j$ and the $v_x$ have 3 information pairs.
  - $v_i$ sends to $w_j$ and the $w_x$ have 5 information pairs.
  - $w_i$ sends to $v_j$ and the $v_x$ have 8 information pairs.
  - $v_i$ sends to $w_j$ and the $w_x$ have 13 information pairs.
  - $w_i$ sends to $v_j$ and the $v_x$ have 21 information pairs.
  - Thus the grow-rate and the algorithm is clearly visible.
2nd Idea (Let the Knowledge grow in a structured way)

- After the first two rounds some node-pairs share their information.
- Consider this situation as the start:
  - All $v_x$ and $w_x$ have one information pair.
  - $v_i$ sends to $w_j$ and the $w_x$ have 2 information pairs.
  - $w_i$ sends to $v_j$ and the $v_x$ have 3 information pairs.
  - $v_i$ sends to $w_j$ and the $v_x$ have 5 information pairs.
  - $w_i$ sends to $v_j$ and the $v_x$ have 8 information pairs.
  - $v_i$ sends to $w_j$ and the $w_x$ have 13 information pairs.
  - $w_i$ sends to $v_j$ and the $v_x$ have 21 information pairs.
- Thus the grow-rate and the algorithm is clearly visible.
2nd Idea (Let the Knowledge grow in a structured way)

- After the first two rounds some node-pairs share their information.

- Consider this situation as the start:
  - All $v_x$ and $w_x$ have one information pair.
  - $v_i$ sends to $w_j$ and the $w_x$ have 2 information pairs.
  - $w_i$ sends to $v_j$ and the $v_x$ have 3 information pairs.
  - $v_i$ sends to $w_j$ and the $w_x$ have 5 information pairs.
  - $w_i$ sends to $v_j$ and the $v_x$ have 8 information pairs.
  - $v_i$ sends to $w_j$ and the $w_x$ have 13 information pairs.
  - $w_i$ sends to $v_j$ and the $v_x$ have 21 information pairs.
  - Thus the grow-rate and the algorithm is clearly visible.
2\textsuperscript{nd} Idea (Let the Knowledge grow in a structured way)

- After the first two rounds some node-pairs share their information.

- Consider this situation as the start:
  - All $v_x$ and $w_x$ have one information pair.
  - $v_i$ sends to $w_j$ and the $w_x$ have 2 information pairs.
  - $w_i$ sends to $v_j$ and the $v_x$ have 3 information pairs.
  - $v_i$ sends to $w_j$ and the $w_x$ have 5 information pairs.
  - $w_i$ sends to $v_j$ and the $v_x$ have 8 information pairs.
  - $v_i$ sends to $w_j$ and the $w_x$ have 13 information pairs.
  - $w_i$ sends to $v_j$ and the $v_x$ have 21 information pairs.
  - Thus the grow-rate and the algorithm is clearly visible.
Let $n = 2m$.

Gossip-Algorithm:

$t := 0$;

for all $i \in \{0, \ldots, m - 1\}$ do in parallel $R[i]$ sends to $Q[i]$;

for all $i \in \{0, \ldots, m - 1\}$ do in parallel $Q[i]$ sends to $R[i]$;

while $\text{fib}(2t + 1) < m$ do begin

$t := t + 1$;

for all $i \in \{0, \ldots, m - 1\}$ do in parallel

$R[(i + \text{fib}(2t - 1)) \mod m]$ sends to $Q[i]$;

if $\text{fib}(2t) < m$ then

for all $i \in \{0, \ldots, m - 1\}$ do in parallel

$Q[(i + \text{fib}(2t)) \mod m]$ sends to $R[i]$;

end;
Let $n = 2m$.

Gossip-Algorithm:

$t := 0;
\text{for all } i \in \{0, \ldots, m - 1\} \text{ do in parallel } R[i] \text{ sends to } Q[i];\n\text{for all } i \in \{0, \ldots, m - 1\} \text{ do in parallel } Q[i] \text{ sends to } R[i];\n\text{while } \text{fib}(2t + 1) < m \text{ do begin}\n\quad t := t + 1;\n\quad \text{for all } i \in \{0, \ldots, m - 1\} \text{ do in parallel}\n\quad \quad R[(i + \text{fib}(2t - 1)) \mod m] \text{ sends to } Q[i];\n\quad \text{if } \text{fib}(2t) < m \text{ then}\n\quad \quad \text{for all } i \in \{0, \ldots, m - 1\} \text{ do in parallel}\n\quad \quad \quad Q[(i + \text{fib}(2t)) \mod m] \text{ sends to } R[i]\n\text{end;}

$$
\begin{align*}
\text{fib}(0) &= \text{fib}(1) = 1 \\
\text{fib}(i) &= \text{fib}(i - 1) + \text{fib}(i - 2)
\end{align*}
$$
Let $n = 2m$.

Gossip-Algorithm:

$t := 0$;

for all $i \in \{0, \ldots, m - 1\}$ do in parallel $R[i]$ sends to $Q[i]$;

for all $i \in \{0, \ldots, m - 1\}$ do in parallel $Q[i]$ sends to $R[i]$;

while $\text{fib}(2t + 1) < m$ do begin

$t := t + 1$;

for all $i \in \{0, \ldots, m - 1\}$ do in parallel

$R[(i + \text{fib}(2t - 1)) \mod m]$ sends to $Q[i]$;

if $\text{fib}(2t) < m$ then

for all $i \in \{0, \ldots, m - 1\}$ do in parallel

$Q[(i + \text{fib}(2t)) \mod m]$ sends to $R[i]$

end;

\[\text{fib}(0) = \text{fib}(1) = 1\]
\[\text{fib}(i) = \text{fib}(i - 1) + \text{fib}(i - 2)\]
algorithm

- Let $n = 2m$.
- Gossip-Algorithm:
  
  ```
  t := 0;
  for all $i \in \{0, \ldots, m-1\}$ do in parallel $R[i]$ sends to $Q[i]$;
  for all $i \in \{0, \ldots, m-1\}$ do in parallel $Q[i]$ sends to $R[i]$;
  while $\text{fib}(2t+1) < m$ do begin
    $t := t + 1$;
    for all $i \in \{0, \ldots, m-1\}$ do in parallel
      $R[(i + \text{fib}(2t-1)) \mod m]$ sends to $Q[i]$;
    if $\text{fib}(2t) < m$ then
      for all $i \in \{0, \ldots, m-1\}$ do in parallel
        $Q[(i + \text{fib}(2t)) \mod m]$ sends to $R[i]$
  end;
  ```

```math
\text{fib}(0) = \text{fib}(1) = 1 \\
\text{fib}(i) = \text{fib}(i-1) + \text{fib}(i-2)
```
Let $n = 2m$.

Gossip-Algorithm:

$t := 0$;

for all $i \in \{0, \ldots, m - 1\}$ do in parallel $R[i]$ sends to $Q[i]$;

for all $i \in \{0, \ldots, m - 1\}$ do in parallel $Q[i]$ sends to $R[i]$;

while $\text{fib}(2t + 1) < m$ do begin

$t := t + 1$;

for all $i \in \{0, \ldots, m - 1\}$ do in parallel $R[(i + \text{fib}(2t - 1)) \mod m]$ sends to $Q[i]$;

if $\text{fib}(2t) < m$ then

for all $i \in \{0, \ldots, m - 1\}$ do in parallel $Q[(i + \text{fib}(2t)) \mod m]$ sends to $R[i]$

end;

\[
\begin{align*}
\text{fib}(0) &= \text{fib}(1) = 1 \\
\text{fib}(i) &= \text{fib}(i - 1) + \text{fib}(i - 2)
\end{align*}
\]
Theorem:

Let \( n = 2m \) and \( k = \min\{x \mid \text{fib}(x) \geq m\} \). Then we have \( r(K(n)) \leq k + 1 \).

Proof:

- The algorithm stops, if \( \text{fib}(2t + 1) \geq m \) or \( \text{fib}(2t) \geq m \) holds.
- The number of rounds within the loop is \( 2t \) or \( 2(t - 1) + 1 \).
- The total number of rounds is \( (k - 1) + 2 \).
- Correctness may be proven by the following invariant:
- Let \( a[i] \) be the information, which share \( R[i] \) and \( Q[i] \) after two rounds.
- After \( t \) loops we have:
  - \( Q[i] \) knows \( \bigcup_{0 \leq j \leq \text{fib}(2t+1)-1} \alpha[(i + j) \mod m] \)
  - \( R[i] \) knows \( \bigcup_{0 \leq j \leq \text{fib}(2t+2)-1} \alpha[(i + j) \mod m] \)
- The correctness is a direct result of this.
Theorem:

Let \( n = 2m \) and \( k = \min\{x \mid \text{fib}(x) \geq m\} \). Then we have \( r(K(n)) \leq k + 1 \).

Proof:

- The algorithm stops, if \( \text{fib}(2t + 1) \geq m \) or \( \text{fib}(2t) \geq m \) holds.
- The number of rounds within the loop is \( 2t \) or \( 2(t - 1) + 1 \).
- The total number of rounds is \( k - 1 + 2 \).
- Correctness may be proven by the following invariant:
- Let \( a[i] \) be the information, which share \( R[i] \) and \( Q[i] \) after two rounds.
- After \( t \) loops we have:
  - \( Q[i] \) knows \( \cup_{0 \leq j \leq \text{fib}(2t+1)-1} \alpha[(i + j) \mod m] \)
  - \( R[i] \) knows \( \cup_{0 \leq j \leq \text{fib}(2t+2)-1} \alpha[(i + j) \mod m] \)
- The correctness is a direct result of this.
One-Way-Gossip

Theorem:
Let $n = 2m$ and $k = \min\{x \mid \text{fib}(x) \geq m\}$. Then we have $r(K(n)) \leq k + 1$.

Proof:

- The algorithm stops, if $\text{fib}(2t + 1) \geq m$ or $\text{fib}(2t) \geq m$ holds.
- The number of rounds within the loop is $2t$ or $2(t - 1) + 1$.
- The total number of rounds is $(k - 1) + 2$.
- Correctness may be proven by the following invariant:
- Let $a[i]$ be the information, which share $R[i]$ and $Q[i]$ after two rounds.
- After $t$ loops we have:
  - $Q[i]$ knows $\cup_{0 \leq j \leq \text{fib}(2t+1)-1} \alpha[(i + j) \mod m]$
  - $R[i]$ knows $\cup_{0 \leq j \leq \text{fib}(2t+2)-1} \alpha[(i + j) \mod m]$
- The correctness is a direct result of this.
One-Way-Gossip

Theorem:
Let $n = 2m$ and $k = \min\{x \mid \text{fib}(x) \geq m\}$. Then we have $r(K(n)) \leq k + 1$.

Proof:

- The algorithm stops, if $\text{fib}(2t + 1) \geq m$ or $\text{fib}(2t) \geq m$ holds.
- The number of rounds within the loop is $2t$ or $2(t - 1) + 1$.
- The total number of rounds is $(k - 1) + 2$.
- Correctness may be proven by the following invariant:
- Let $a[i]$ be the information, which share $R[i]$ and $Q[i]$ after two rounds.
- After $t$ loops we have:
  - $Q[i]$ knows $\cup_{0 \leq j \leq \text{fib}(2t+1)-1} \alpha[(i + j) \mod m]$
  - $R[i]$ knows $\cup_{0 \leq j \leq \text{fib}(2t+2)-1} \alpha[(i + j) \mod m]$
- The correctness is a direct result of this.
One-Way-Gossip

Theorem:
Let $n = 2m$ and $k = \min\{x \mid \text{fib}(x) \geq m\}$. Then we have $r(K(n)) \leq k + 1$.

Proof:
- The algorithm stops, if $\text{fib}(2t + 1) \geq m$ or $\text{fib}(2t) \geq m$ holds.
- The number of rounds within the loop is $2t$ or $2(t - 1) + 1$.
- The total number of rounds is $(k - 1) + 2$.
- Correctness may be proven by the following invariant:
  - Let $a[i]$ be the information, which share $R[i]$ and $Q[i]$ after two rounds.
  - After $t$ loops we have:
    - $Q[i]$ knows $\bigcup_{0 \leq j \leq \text{fib}(2t+1)-1} a[(i + j) \mod m]$
    - $R[i]$ knows $\bigcup_{0 \leq j \leq \text{fib}(2t+2)-1} a[(i + j) \mod m]$
  - The correctness is a direct result of this.
Theorem:
Let $n = 2m$ and $k = \min\{x \mid \text{fib}(x) \geq m\}$. Then we have $r(K(n)) \leq k + 1$.

Proof:
- The algorithm stops, if $\text{fib}(2t + 1) \geq m$ or $\text{fib}(2t) \geq m$ holds.
- The number of rounds within the loop is $2t$ or $2(t - 1) + 1$.
- The total number of rounds is $(k - 1) + 2$.
- Correctness may be proven by the following invariant:
  - Let $a[i]$ be the information, which share $R[i]$ and $Q[i]$ after two rounds.
  - After $t$ loops we have:
    - $Q[i]$ knows $\cup_{0 \leq j \leq \text{fib}(2t+1)-1} \alpha[(i + j) \mod m]$
    - $R[i]$ knows $\cup_{0 \leq j \leq \text{fib}(2t+2)-1} \alpha[(i + j) \mod m]$
  - The correctness is a direct result of this.
One-Way-Gossip

Theorem:
Let $n = 2m$ and $k = \min\{x \mid \text{fib}(x) \geq m\}$. Then we have $r(K(n)) \leq k + 1$.

Proof:
- The algorithm stops, if $\text{fib}(2t + 1) \geq m$ or $\text{fib}(2t) \geq m$ holds.
- The number of rounds within the loop is $2t$ or $2(t - 1) + 1$.
- The total number of rounds is $(k - 1) + 2$.
- Correctness may be proven by the following invariant:
- Let $a[i]$ be the information, which share $R[i]$ and $Q[i]$ after two rounds.
  - After $t$ loops we have:
    - $Q[i]$ knows $\bigcup_{0 \leq j \leq \text{fib}(2t+1)-1} \alpha[(i + j) \mod m]$
    - $R[i]$ knows $\bigcup_{0 \leq j \leq \text{fib}(2t+2)-1} \alpha[(i + j) \mod m]$
- The correctness is a direct result of this.
One-Way-Gossip

**Theorem:**
Let \( n = 2m \) and \( k = \min\{x \mid \text{fib}(x) \geq m\} \). Then we have \( r(K(n)) \leq k + 1 \).

**Proof:**
- The algorithm stops, if \( \text{fib}(2t + 1) \geq m \) or \( \text{fib}(2t) \geq m \) holds.
- The number of rounds within the loop is \( 2t \) or \( 2(t - 1) + 1 \).
- The total number of rounds is \( (k - 1) + 2 \).
- Correctness may be proven by the following invariant:
- Let \( a[i] \) be the information, which share \( R[i] \) and \( Q[i] \) after two rounds.
- After \( t \) loops we have:
  - \( Q[i] \) knows \( \bigcup_{0 \leq j \leq \text{fib}(2t + 1) - 1} a[(i + j) \mod m] \)
  - \( R[i] \) knows \( \bigcup_{0 \leq j \leq \text{fib}(2t + 2) - 1} a[(i + j) \mod m] \)
- The correctness is a direct result of this.
One-Way-Gossip

Theorem:
Let $n = 2m$ and $k = \min\{x \mid \text{fib}(x) \geq m\}$. Then we have $r(K(n)) \leq k + 1$.

Proof:

- The algorithm stops, if $\text{fib}(2t + 1) \geq m$ or $\text{fib}(2t) \geq m$ holds.
- The number of rounds within the loop is $2t$ or $2(t - 1) + 1$.
- The total number of rounds is $(k - 1) + 2$.
- Correctness may be proven by the following invariant:
  - Let $a[i]$ be the information, which share $R[i]$ and $Q[i]$ after two rounds.
  - After $t$ loops we have:
    - $Q[i]$ knows $\bigcup_{0 \leq j \leq \text{fib}(2t+1)-1} \alpha[(i + j) \mod m]$
    - $R[i]$ knows $\bigcup_{0 \leq j \leq \text{fib}(2t+2)-1} \alpha[(i + j) \mod m]$
- The correctness is a direct result of this.
Theorem:
Let $n = 2m$ and $k = \min\{x \mid \text{fib}(x) \geq m\}$. Then we have $r(K(n)) \leq k + 1$.

Proof:
- The algorithm stops, if $\text{fib}(2t + 1) \geq m$ or $\text{fib}(2t) \geq m$ holds.
- The number of rounds within the loop is $2t$ or $2(t - 1) + 1$.
- The total number of rounds is $(k - 1) + 2$.
- Correctness may be proven by the following invariant:
- Let $a[i]$ be the information, which share $R[i]$ and $Q[i]$ after two rounds.
- After $t$ loops we have:
  - $Q[i]$ knows $\cup_{0 \leq j \leq \text{fib}(2t+1)-1} \alpha[(i + j) \mod m]$.
  - $R[i]$ knows $\cup_{0 \leq j \leq \text{fib}(2t+2)-1} \alpha[(i + j) \mod m]$.
- The correctness is a direct result of this.
Theorem:

Let \( n = 2m \) and \( k = \min\{x \mid \text{fib}(x) \geq m\} \). Then we have \( r(K(n)) \leq k + 1 \).

Proof:

- The algorithm stops, if \( \text{fib}(2t + 1) \geq m \) or \( \text{fib}(2t) \geq m \) holds.
- The number of rounds within the loop is \( 2t \) or \( 2(t - 1) + 1 \).
- The total number of rounds is \( (k - 1) + 2 \).
- Correctness may be proven by the following invariant:

Let \( a[i] \) be the information, which share \( R[i] \) and \( Q[i] \) after two rounds.

After \( t \) loops we have:

- \( Q[i] \) knows \( \bigcup_{0 \leq j \leq \text{fib}(2t + 1) - 1} \alpha[(i + j) \mod m] \)
- \( R[i] \) knows \( \bigcup_{0 \leq j \leq \text{fib}(2t + 2) - 1} \alpha[(i + j) \mod m] \)

The correctness is a direct result of this.
**Theorem:**

Let $n = 2m - 1$ and $k = \min\{x \mid \text{fib}(x) \geq m\}$. Then we have $r(K(n)) \leq k + 2$.

Proof: Using the same idea as for the two-way mode.

**Theorem:**

Let $n$ even. Then we have: $r(K(n)) \geq 2 + \lceil \log_{\frac{1}{2}}(1+\sqrt{5}) \frac{n}{2} \rceil$.

Proof: See literature (Idea is given the following).

<table>
<thead>
<tr>
<th>$n$</th>
<th>2</th>
<th>4</th>
<th>6</th>
<th>8</th>
<th>10</th>
<th>12</th>
<th>14</th>
<th>16</th>
<th>18</th>
<th>20</th>
<th>22</th>
</tr>
</thead>
<tbody>
<tr>
<td>Upper Bound</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>6</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>8</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>Lower Bound</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>6</td>
<td>6</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
</tr>
</tbody>
</table>
One-Way-Gossip

Theorem:
Let \( n = 2m - 1 \) and \( k = \min\{x \mid \text{fib}(x) \geq m\} \). Then we have \( r(K(n)) \leq k + 2 \).

Proof: Using the same idea as for the two-way mode.

Theorem:
Let \( n \) even. Then we have: \( r(K(n)) \geq 2 + \lceil \log_{\frac{1}{2}(1+\sqrt{5})} \frac{n}{2} \rceil \).

Proof: See literature (Idea is given the following).

<table>
<thead>
<tr>
<th>( n )</th>
<th>2</th>
<th>4</th>
<th>6</th>
<th>8</th>
<th>10</th>
<th>12</th>
<th>14</th>
<th>16</th>
<th>18</th>
<th>20</th>
<th>22</th>
</tr>
</thead>
<tbody>
<tr>
<td>Upper Bound</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>6</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>8</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>Lower Bound</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>6</td>
<td>6</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
</tr>
</tbody>
</table>
One-Way-Gossip

**Theorem:**

Let $n = 2m - 1$ and $k = \min \{x \mid \text{fib}(x) \geq m\}$. Then we have $r(K(n)) \leq k + 2$.

Proof: Using the same idea as for the two-way mode.

**Theorem:**

Let $n$ even. Then we have: $r(K(n)) \geq 2 + \lceil \log_{\frac{1}{2}(1+\sqrt{5})} \frac{n}{2} \rceil$.

Proof: See literature (Idea is given the following).

<table>
<thead>
<tr>
<th>$n$</th>
<th>2</th>
<th>4</th>
<th>6</th>
<th>8</th>
<th>10</th>
<th>12</th>
<th>14</th>
<th>16</th>
<th>18</th>
<th>20</th>
<th>22</th>
</tr>
</thead>
<tbody>
<tr>
<td>Upper Bound</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>6</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>8</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>Lower Bound</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>6</td>
<td>6</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
</tr>
</tbody>
</table>
Idea for the lower Bound

**Situation:**
- Algorithm with “fibonacci growth”.
- No idea to enlarge this growth.

**Construction of a lower bound:**
- Start with an arbitrary algorithm.
- Use only the restriction of the algorithm.
- Abstract.

We will now try to do the abstraction.

Try the get the core-problem.

The core-problem ist:
- “Fibonacci growth” could not be improved.
Idea for the lower Bound

**Situation:**
- Algorithm with “fibonacci growth”.
- No idea to enlarge this growth.

**Construction of a lower bound:**
- Start with an arbitrary algorithm.
- Use only the restriction of the algorithm.
- Abstract.

**We will now try to do the abstraction.**

**Try the get the core-problem.**

**The core-problem ist:**
- “Fibonacci growth” could not be improved.
Idea for the lower Bound

- **Situation:**
  - Algorithm with “fibonacci growth”.
  - No idea to enlarge this growth.

- **Construction of a lower bound:**
  - Start with an arbitrary algorithm.
  - Use only the restriction of the algorithm.
  - Abstract.

- We will now try to do the abstraction.

- Try the get the core-problem.

- The core-problem ist:
  - “Fibonacci growth” could not be improved.
Idea for the lower Bound

- **Situation:**
  - Algorithm with “fibonacci growth”.
  - No idea to enlarge this growth.

- **Construction of a lower bound:**
  - Start with an arbitrary algorithm.
  - Use only the restriction of the algorithm.
  - Abstract.

- We will now try to do the abstraction.
- Try the get the core-problem.
- The core-problem ist:
  - “Fibonacci growth” could not be improved.
Idea for the lower Bound

- **Situation:**
  - Algorithm with “fibonacci growth”.
  - No idea to enlarge this growth.

- **Construction of a lower bound:**
  - **Start with an arbitrary algorithm.**
  - Use only the restriction of the algorithm.
  - Abstract.

- We will now try to do the abstraction.
- Try the get the core-problem.
- The core-problem ist:
  - “Fibonacci growth” could not be improved.
Idea for the lower Bound

- **Situation:**
  - Algorithm with “fibonacci growth”.
  - No idea to enlarge this growth.

- **Construction of a lower bound:**
  - Start with an arbitrary algorithm.
  - Use only the restriction of the algorithm.
  - Abstract.

- We will now try to do the abstraction.
- Try the get the core-problem.
- The core-problem ist:
  - “Fibonacci growth” could not be improved.
Idea for the lower Bound

○ Situation:
  ○ Algorithm with “fibonacci growth”.
  ○ No idea to enlarge this growth.

○ Construction of a lower bound:
  ○ Start with an arbitrary algorithm.
  ○ Use only the restriction of the algorithm.
  ○ Abstract.

○ We will now try to do the abstraction.

○ Try the get the core-problem.

○ The core-problem ist:
  ○ “Fibonacci growth” could not be improved.
Idea for the lower Bound

- **Situation:**
  - Algorithm with “fibonacci growth”.
  - No idea to enlarge this growth.

- **Construction of a lower bound:**
  - Start with an arbitrary algorithm.
  - Use only the restriction of the algorithm.
  - Abstract.

- **We will now try to do the abstraction.**

- Try the get the core-problem.

- The core-problem ist:
  - “Fibonacci growth” could not be improved.
Idea for the lower Bound

Situation:
- Algorithm with “fibonacci growth”.
- No idea to enlarge this growth.

Construction of a lower bound:
- Start with an arbitrary algorithm.
- Use only the restriction of the algorithm.
- Abstract.

We will now try to do the abstraction.

Try the get the core-problem.

The core-problem ist:
- “Fibonacci growth” could not be improved.
Idea for the lower Bound

- **Situation:**
  - Algorithm with “fibonacci growth”.
  - No idea to enlarge this growth.

- **Construction of a lower bound:**
  - Start with an arbitrary algorithm.
  - Use only the restriction of the algorithm.
  - Abstract.

- We will now try to do the abstraction.

- Try the get the core-problem.

- **The core-problem ist:**
  - “Fibonacci growth” could not be improved.
Idea for the lower Bound

- **Situation:**
  - Algorithm with “fibonacci growth”.
  - No idea to enlarge this growth.

- **Construction of a lower bound:**
  - Start with an arbitrary algorithm.
  - Use only the restriction of the algorithm.
  - Abstract.

- We will now try to do the abstraction.
- Try the get the core-problem.
- The core-problem ist:
  - “Fibonacci growth” could not be improved.
Idea for the lower Bound

- Situation:
  - Algorithm with “fibonacci growth”.
  - No idea to enlarge this growth.

- Construction of a lower bound:
  - Start with an arbitrary algorithm.
  - Use only the restriction of the algorithm.
  - Abstract.

- We will now try to do the abstraction.

- Try the get the core-problem.

- The core-problem ist:
  - “Fibonacci growth” could not be improved.
1. Abstraction

**Definition:**

The **Network Counting Problem**:

- **Given a directed graph** \( G = (V, E) \).
- Each node stores a number.
- Initial just the number 1 is stored.
- The receiver add the number from the sender to his number after one communication.
- The objective is: all nodes should store a number larger then \( |V| \).
- With \( nc(G) \) we denote the minimal rounds to achieve this objective.

**Lemma:**

For any graph \( G \) we have: \( r(G) \geq nc(G) \).
1. Abstraction

**Definition:**

The **Network Counting Problem**:
- Given a directed graph $G = (V, E)$.
- Each node stores a number.
- Initial just the number 1 is stored.
- The receiver add the number from the sender to his number after one communication.
- The objective is: all nodes should store a number larger then $|V|$.
- With $nc(G)$ we denote the minimal rounds to achieve this objective.

**Lemma:**

For any graph $G$ we have: $r(G) \geq nc(G)$.
1. Abstraction

**Definition:**

The **Network Counting Problem**:
- Given a directed graph $G = (V, E)$.
- Each node stores a number.
- **Initial just the number 1 is stored.**
- The receiver add the number from the sender to his number after one communication.
- The objective is: all nodes should store a number larger then $|V|$.
- With $nc(G)$ we denote the minimal rounds to achieve this objective.

**Lemma:**

For any graph $G$ we have: $r(G) \geq nc(G)$. 
1. Abstraction

**Definition:**

The **Network Counting Problem**:

- Given a directed graph $G = (V, E)$.
- Each node stores a number.
- Initial just the number 1 is stored.
- The receiver adds the number from the sender to his number after one communication.
- The objective is: all nodes should store a number larger than $|V|$.
- With $nc(G)$ we denote the minimal rounds to achieve this objective.

**Lemma:**

For any graph $G$ we have: $r(G) \geq nc(G)$.
1. Abstraction

**Definition:**

The **Network Counting Problem:**

- Given a directed graph $G = (V, E)$.
- Each node stores a number.
- Initial just the number 1 is stored.
- The receiver add the number from the sender to his number after one communication.
- **The objective is:** all nodes should store a number larger than $|V|$.
- With $nc(G)$ we denote the minimal rounds to achieve this objective.

**Lemma:**

For any graph $G$ we have: $r(G) \geq nc(G)$. 
1. Abstraction

**Definition:**

The **Network Counting Problem**:

- Given a directed graph $G = (V, E)$.
- Each node stores a number.
- Initial just the number 1 is stored.
- The receiver add the number from the sender to his number after one communication.
- The objective is: all nodes should store a number larger then $|V|$.

- **With** $nc(G)$ **we denote the minimal rounds to achieve this objective.**

**Lemma:**

For any graph $G$ we have: $r(G) \geq nc(G)$. 
1. Abstraction

Definition:
The Network Counting Problem:
- Given a directed graph $G = (V, E)$.
- Each node stores a number.
- Initial just the number 1 is stored.
- The receiver add the number from the sender to his number after one communication.
- The objective is: all nodes should store a number larger than $|V|$.
- With $nc(G)$ we denote the minimal rounds to achieve this objective.

Lemma:
For any graph $G$ we have: $r(G) \geq nc(G)$. 
1. Abstraction

**Definition:**

The **Network Counting Problem**:

- Given a directed graph $G = (V, E)$.
- Each node stores a number.
- Initial just the number 1 is stored.
- The receiver add the number from the sender to his number after one communication.
- The objective is: all nodes should store a number larger than $|V|$.
- With $nc(G)$ we denote the minimal rounds to achieve this objective.

**Lemma:**

For any graph $G$ we have: $r(G) \geq nc(G)$.
2. Abstraction

- Let $G = (\{v_1, v_2, v_3, \cdots, v_n\}, E)$ be a directed Graph.
- Each node $v_i$ stores after $t$ rounds the number $z_i^t$.
- One situation of the network counting problem could be described by a vector:
  - Initial: $(1, 1, 1, \cdots, 1)^T$.
  - After $t$ rounds: $(z_1^t, z_2^t, z_3^t, z_n^t)^T$.
- One round of an algorithm for the network counting problem is given by a matrix $B$:
  - $A$ is a $n \times n$ matrix.
  - $a_{ij} = 1$ node $j$ sends to node $i$.
  - $A$ contains on the diagonal only ones.
  - $A$ has in each row at most two ones.
  - $A$ has in each column at most two ones.
  - If $a_{ij} = a_{kl} = 1$ ($i \neq j \neq k \neq l$), then we have $l \neq i \neq k$ and $l \neq j \neq k$.
  - Thus we get: $A \cdot (z_1^t, z_2^t, z_3^t, z_n^t)^T = (z_1^{t+1}, z_2^{t+1}, z_3^{t+1}, z_n^{t+1})^T$.
2. Abstraction

- Let $G = (\{v_1, v_2, v_3, \ldots, v_n\}, E)$ be a directed Graph.
- Each node $v_i$ stores after $t$ rounds the number $z_i^t$.
- One situation of the network counting problem could be described by a vector:
  - Initial: $(1, 1, 1, \ldots, 1)^T$.
  - After $t$ rounds: $(z_1^t, z_2^t, z_3^t, z_n^t)^T$.
- One round of an algorithm for the network counting problem is given by a matrix $B$:
  - $A$ is a $n \times n$ matrix.
  - $a_{ij} = 1$ node $j$ sends to node $i$.
  - $A$ contains on the diagonal only ones.
  - $A$ has in each row at most two ones.
  - $A$ has in each column at most two ones.
  - If $a_{ij} = a_{kl} = 1$ ($i \neq j \neq k \neq l$), then we have $l \neq i \neq k$ and $l \neq j \neq k$.
  - Thus we get: $A \cdot (z_1^t, z_2^t, z_3^t, z_n^t)^T = (z_1^{t+1}, z_2^{t+1}, z_3^{t+1}, z_n^{t+1})^T$
2. Abstraction

Let $G = (\{v_1, v_2, v_3, \cdots, v_n\}, E)$ be a directed Graph.

Each node $v_i$ stores after $t$ rounds the number $z_i^t$.

One situation of the network counting problem could be described by a vector:

- Initial: $(1, 1, 1, \cdots, 1)^T$.
- After $t$ rounds: $(z_1^t, z_2^t, z_3^t, \cdots, z_n^t)^T$.

One round of an algorithm for the network counting problem is given by a matrix $B$:

- $A$ is a $n \times n$ matrix.
- $a_{ij} = 1$ node $j$ sends to node $i$.
- $A$ contains on the diagonal only ones.
- $A$ has in each row at most two ones.
- $A$ has in each column at most two ones.
- If $a_{ij} = a_{kl} = 1$ ($i \neq j \neq k \neq l$), then we have $l \neq i \neq k$ and $l \neq j \neq k$.
- Thus we get: $A \cdot (z_1^t, z_2^t, z_3^t, \cdots, z_n^t)^T = (z_1^{t+1}, z_2^{t+1}, z_3^{t+1}, \cdots, z_n^{t+1})^T$.
2. Abstraction

- Let $G = (\{v_1, v_2, v_3, \ldots, v_n\}, E)$ be a directed Graph.
- Each node $v_i$ stores after $t$ rounds the number $z_i^t$.
- One situation of the network counting problem could be described by a vector:
  - Initial: $(1, 1, 1, \ldots, 1)^T$.
  - After $t$ rounds: $(z_1^t, z_2^t, z_3^t, z_n^t)^T$.
- One round of an algorithm for the network counting problem is given by a matrix $B$:
  - $A$ is a $n \times n$ matrix.
  - $a_{ij} = 1$ node $j$ sends to node $i$.
  - $A$ contains on the diagonal only ones.
  - $A$ has in each row at most two ones.
  - $A$ has in each column at most two ones.
  - If $a_{ij} = a_{kl} = 1$ ($i \neq j \neq k \neq l$), then we have $l \neq i \neq k$ and $l \neq j \neq k$.
  - Thus we get: $A \cdot (z_1^t, z_2^t, z_3^t, z_n^t)^T = (z_1^{t+1}, z_2^{t+1}, z_3^{t+1}, z_n^{t+1})^T$.
2. Abstraction

- Let $G = (\{v_1, v_2, v_3, \cdots, v_n\}, E)$ be a directed Graph.
- Each node $v_i$ stores after $t$ rounds the number $z_i^t$.
- One situation of the network counting problem could be described by a vector:
  - Initial: $(1, 1, 1, \cdots, 1)^T$.
  - After $t$ rounds: $(z_1^t, z_2^t, z_3^t, z_n^t)^T$.
- One round of an algorithm for the network counting problem is given by a matrix $B$:
  - $A$ is a $n \times n$ matrix.
  - $a_{ij} = 1$ node $j$ sends to node $i$.
  - $A$ contains on the diagonal only ones.
  - $A$ has in each row at most two ones.
  - $A$ has in each column at most two ones.
  - If $a_{ij} = a_{kl} = 1$ ($i \neq j \neq k \neq l$), then we have $l \neq i \neq k$ and $l \neq j \neq k$.
  - Thus we get: $A \cdot (z_1^t, z_2^t, z_3^t, z_n^t)^T = (z_1^{t+1}, z_2^{t+1}, z_3^{t+1}, z_n^{t+1})^T$.
2. Abstraction

- Let $G = (\{v_1, v_2, v_3, \cdots, v_n\}, E)$ be a directed Graph.
- Each node $v_i$ stores after $t$ rounds the number $z_i^t$.
- One situation of the network counting problem could be described by a vector:
  - Initial: $(1, 1, 1, \cdots, 1)^T$.
  - After $t$ rounds: $(z_1^t, z_2^t, z_3^t, z_n^t)^T$.
- One round of an algorithm for the network counting problem is given by a matrix $B$:
  - $A$ is a $n \times n$ matrix.
  - $a_{ij} = 1$ node $j$ sends to node $i$.
  - $A$ contains on the diagonal only ones.
  - $A$ has in each row at most two ones.
  - $A$ has in each column at most two ones.
  - If $a_{ij} = a_{kl} = 1$ ($i \neq j \neq k \neq l$), then we have $l \neq i \neq k$ and $l \neq j \neq k$.
- Thus we get: $A \cdot (z_1^t, z_2^t, z_3^t, z_n^t)^T = (z_1^{t+1}, z_2^{t+1}, z_3^{t+1}, z_n^{t+1})^T$.
2. Abstraction

- Let $G = (\{v_1, v_2, v_3, \cdots, v_n\}, E)$ be a directed Graph.
- Each node $v_i$ stores after $t$ rounds the number $z_i^t$.
- One situation of the network counting problem could be described by a vector:
  - Initial: $(1, 1, 1, \cdots, 1)^T$.
  - After $t$ rounds: $(z_1^t, z_2^t, z_3^t, z_n^t)^T$.
- One round of an algorithm for the network counting problem is given by a matrix $B$:
  - $A$ is a $n \times n$ matrix.
  - $a_{ij} = 1$ node $j$ sends to node $i$.
  - $A$ contains on the diagonal only ones.
  - $A$ has in each row at most two ones.
  - $A$ has in each column at most two ones.
  - If $a_{ij} = a_{kl} = 1$ ($i \neq j \neq k \neq l$), then we have $l \neq i \neq k$ and $l \neq j \neq k$.
  - Thus we get: $A \cdot (z_1^t, z_2^t, z_3^t, z_n^t)^T = (z_1^{t+1}, z_2^{t+1}, z_3^{t+1}, z_n^{t+1})^T$
2. Abstraction

- Let $G = (\{v_1, v_2, v_3, \ldots, v_n\}, E)$ be a directed Graph.
- Each node $v_i$ stores after $t$ rounds the number $z_i^t$.
- One situation of the network counting problem could be described by a vector:
  - Initial: $(1, 1, 1, \ldots, 1)^T$.
  - After $t$ rounds: $(z_1^t, z_2^t, z_3^t, z_n^t)^T$.
- One round of an algorithm for the network counting problem is given by a matrix $B$:
  - $A$ is a $n \times n$ matrix.
  - $a_{ij} = 1$ node $j$ sends to node $i$.
  - $A$ contains on the diagonal only ones.
  - $A$ has in each row at most two ones.
  - $A$ has in each column at most two ones.
  - If $a_{ij} = a_{kl} = 1$ ($i \neq j \neq k \neq l$), then we have $l \neq i \neq k$ and $l \neq j \neq k$.
  - Thus we get: $A \cdot (z_1^t, z_2^t, z_3^t, z_n^t)^T = (z_1^{t+1}, z_2^{t+1}, z_3^{t+1}, z_n^{t+1})^T$.
2. Abstraction

- Let $G = (\{v_1, v_2, v_3, \cdots, v_n\}, E)$ be a directed Graph.
- Each node $v_i$ stores after $t$ rounds the number $z_i^t$.
- One situation of the network counting problem could be described by a vector:
  - Initial: $(1, 1, 1, \cdots, 1)^T$.
  - After $t$ rounds: $(z_1^t, z_2^t, z_3^t, z_n^t)^T$.
- One round of an algorithm for the network counting problem is given by a matrix $B$:
  - $A$ is a $n \times n$ matrix.
  - $a_{ij} = 1$ node $j$ sends to node $i$.
  - $A$ contains on the diagonal only ones.
  - $A$ has in each row at most two ones.
  - $A$ has in each column at most two ones.
  - If $a_{ij} = a_{kl} = 1$ ($i \neq j \neq k \neq l$), then we have $l \neq i \neq k$ and $l \neq j \neq k$.
  - Thus we get: $A \cdot (z_1^t, z_2^t, z_3^t, z_n^t)^T = (z_1^{t+1}, z_2^{t+1}, z_3^{t+1}, z_n^{t+1})^T$
2. Abstraction

- Let $G = (\{v_1, v_2, v_3, \cdots, v_n\}, E)$ be a directed Graph.
- Each node $v_i$ stores after $t$ rounds the number $z_i^t$.
- One situation of the network counting problem could be described by a vector:
  - Initial: $(1, 1, 1, \cdots, 1)^T$.
  - After $t$ rounds: $(z_1^t, z_2^t, z_3^t, z_n^t)^T$.
- One round of an algorithm for the network counting problem is given by a matrix $B$:
  - $A$ is a $n \times n$ matrix.
  - $a_{ij} = 1$ node $j$ sends to node $i$.
  - $A$ contains on the diagonal only ones.
  - $A$ has in each row at most two ones.
  - $A$ has in each column at most two ones.
  - If $a_{ij} = a_{kl} = 1$ ($i \neq j \neq k \neq l$), then we have $l \neq i \neq k$ and $l \neq j \neq k$.
  - Thus we get: $A \cdot (z_1^t, z_2^t, z_3^t, z_n^t)^T = (z_1^{t+1}, z_2^{t+1}, z_3^{t+1}, z_n^{t+1})^T$.
2. Abstraction

Let $G = (\{v_1, v_2, v_3, \cdots, v_n\}, E)$ be a directed Graph.

Each node $v_i$ stores after $t$ rounds the number $z_i^t$.

One situation of the network counting problem could be described by a vector:

- Initial: $(1, 1, 1, \cdots, 1)^T$.
- After $t$ rounds: $(z_1^t, z_2^t, z_3^t, \cdots, z_n^t)^T$.

One round of an algorithm for the network counting problem is given by a matrix $B$:

- $A$ is a $n \times n$ matrix.
- $a_{ij} = 1$ node $j$ sends to node $i$.
- $A$ contains on the diagonal only ones.
- $A$ has in each row at most two ones.
- $A$ has in each column at most two ones.
- If $a_{ij} = a_{kl} = 1$ ($i \neq j \neq k \neq l$), then we have $l \neq i \neq k$ and $l \neq j \neq k$.

Thus we get: $A \cdot (z_1^t, z_2^t, z_3^t, \cdots, z_n^t)^T = (z_1^{t+1}, z_2^{t+1}, z_3^{t+1}, \cdots, z_n^{t+1})^T$
2. Abstraction

- Let $G = (\{v_1, v_2, v_3, \cdots, v_n\}, E)$ be a directed Graph.
- Each node $v_i$ stores after $t$ rounds the number $z_i^t$.
- One situation of the network counting problem could be described by a vector:
  - Initial: $(1, 1, 1, \cdots, 1)^T$.
  - After $t$ rounds: $(z_1^t, z_2^t, z_3^t, z_n^t)^T$.
- One round of an algorithm for the network counting problem is given by a matrix $B$:
  - $A$ is a $n \times n$ matrix.
  - $a_{ij} = 1$ node $j$ sends to node $i$.
  - $A$ contains on the diagonal only ones.
  - $A$ has in each row at most two ones.
  - $A$ has in each column at most two ones.
  - If $a_{ij} = a_{kl} = 1$ ($i \neq j \neq k \neq l$), then we have $l \neq i \neq k$ and $l \neq j \neq k$.
  - Thus we get: $A \cdot (z_1^t, z_2^t, z_3^t, z_n^t)^T = (z_1^{t+1}, z_2^{t+1}, z_3^{t+1}, z_n^{t+1})^T$
2. Abstraction

- Let $G = (\{v_1, v_2, v_3, \ldots, v_n\}, E)$ be a directed Graph.
- Each node $v_i$ stores after $t$ rounds the number $z^t_i$.
- One situation of the network counting problem could be described by a vector:
  - Initial: $(1, 1, 1, \ldots, 1)^T$.
  - After $t$ rounds: $(z^t_1, z^t_2, z^t_3, \ldots, z^t_n)^T$.
- One round of an algorithm for the network counting problem is given by a matrix $B$:
  - $A$ is a $n \times n$ matrix.
  - $a_{ij} = 1$ node $j$ sends to node $i$.
  - $A$ contains on the diagonal only ones.
  - $A$ has in each row at most two ones.
  - $A$ has in each column at most two ones.
  - If $a_{ij} = a_{kl} = 1$ ($i \neq j \neq k \neq l$), then we have $l \neq i \neq k$ and $l \neq j \neq k$.
  - Thus we get: $A \cdot (z^t_1, z^t_2, z^t_3, \ldots, z^t_n)^T = (z^{t+1}_1, z^{t+1}_2, z^{t+1}_3, \ldots, z^{t+1}_n)^T$. 
2. Abstraction

Let $G = (\{v_1, v_2, v_3, \cdots, v_n\}, E)$ be a directed Graph.

Each node $v_i$ stores after $t$ rounds the number $z_i^t$.

One situation of the network counting problem could be described by a vector:
- Initial: $(1, 1, 1, \cdots, 1)^T$.
- After $t$ rounds: $(z_1^t, z_2^t, z_3^t, z_n^t)^T$.

One round of an algorithm for the network counting problem is given by a matrix $B$:
- $A$ is a $n \times n$ matrix.
- $a_{ij} = 1$ node $j$ sends to node $i$.
- $A$ contains on the diagonal only ones.
- $A$ has in each row at most two ones.
- $A$ has in each column at most two ones.
- If $a_{ij} = a_{kl} = 1$ ($i \neq j \neq k \neq l$), then we have $l \neq i \neq k$ and $l \neq j \neq k$.
- Thus we get: $A \cdot (z_1^t, z_2^t, z_3^t, z_n^t)^T = (z_1^{t+1}, z_2^{t+1}, z_3^{t+1}, z_n^{t+1})^T$. 

2. Abstraction (Continuation)

- We consider now matrices of the above form.

- These are matrices $A$, for which there is a transformation $T$ with:

  $$TAT^{-1} = \begin{pmatrix} B & B & 0 \\ & & \\ 0 & & 1 \\ \end{pmatrix}.$$

  and $B = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$.

- We will estimate the growth, which these matrices provide for the network counting problem.
2. Abstraction (Continuation)

- We consider now matrices of the above form.

- These are matrices $A$, for which there is a transformation $T$ with:

\[
TAT^{-1} = \begin{pmatrix}
    B & B & 0 \\
    B & \ddots & \ddots \\
    0 & \ddots & B & 1 \\
    \vdots & \ddots & \ddots & 1
\end{pmatrix}.
\]

and $B = \begin{pmatrix} 11 \\ 01 \end{pmatrix}$.

- We will estimate the growth, which these matrices provide for the network counting problem.
We consider now matrices of the above form.

These are matrices $A$, for which there is a transformation $T$ with:

$$TAT^{-1} = \begin{pmatrix} B & 0 \\ \cdot & B \\ \cdot & \cdot & 1 \\ 0 & \cdot & \cdot & 1 \end{pmatrix}.$$

and $B = \begin{pmatrix} 11 \\ 01 \end{pmatrix}$.

We will estimate the growth, which these matrices provide for the network counting problem.
2. Abstraction (Continuation)

- We consider now matrices of the above form.
- These are matrices $A$, for which there is a transformation $T$ with:

$$TAT^{-1} = \begin{pmatrix} B & 0 \\ B & \ddots \\ \vdots & \ddots & B \\ 0 & \ddots & \ddots & 1 \\ & \ddots & \ddots & \ddots \\ & & 0 & \ddots & 1 \end{pmatrix}.$$ 

and $B = \begin{pmatrix} 11 \\ 01 \end{pmatrix}$.

- We will estimate the growth, which these matrices provide for the network counting problem.
Recollection (Norm, 3. Abstraction)

- Let $||.||$ be the vector norm over $\mathbb{R}^n$. Then we have:
  - $||x|| = 0 \iff x = 0^n$,
  - $||\alpha \cdot x|| = |\alpha| \cdot ||x||$,
  - $||x + y|| \leq ||x|| + ||y||$
  - this holds for all $\alpha \in \mathbb{R}, x, y \in \mathbb{R}^n$

- The matrix norm for a vector norm $||.||$ is defined by $||A|| = sup_{x \neq 0} \frac{||Ax||}{||x||}$. Then we have:
  - $||A|| = 0 \iff A = 0$
  - $||A + B|| \leq ||A|| + ||B||$
  - $||\alpha A|| = |\alpha| \cdot ||A||$
  - $||A \cdot B|| \leq ||A|| \cdot ||B||$
  - $||A \cdot x|| \leq ||A|| \cdot ||x||$
  - this holds for all $A, B \in \mathbb{R}^{n^2}, x \in \mathbb{R}^n, \alpha \in \mathbb{R}, \alpha \geq 0$.

- Here we use: $||x|| = \sqrt{\sum_{i=1}^{n} |x_i|^2}$ for ein $x = (x_1, .., x_n)$,

- Known: $||A|| = \text{Spectral Norm}(A) = \sqrt{|\lambda_{max}(A^T \cdot A)|}$ with: $\lambda_{max}$ is the largest Eigenvalue.
Recollection (Norm, 3. Abstraction)

- Let $||\cdot||$ be the vector norm over $\mathbb{R}^n$. Then we have:
  - $||x|| = 0 \iff x = 0^n$,
  - $||\alpha \cdot x|| = |\alpha| \cdot ||x||$,
  - $||x + y|| \leq ||x|| + ||y||$
  - this holds for all $\alpha \in \mathbb{R}, x, y \in \mathbb{R}^n$

- The matrix norm for a vector norm $||\cdot||$ is defined by $||A|| = \sup_{x \neq 0} \frac{||Ax||}{||x||}$.
  Then we have:
  - $||A|| = 0 \iff A = 0$
  - $||A + B|| \leq ||A|| + ||B||$
  - $||\alpha A|| = |\alpha| \cdot ||A||$
  - $||A \cdot B|| \leq ||A|| \cdot ||B||$
  - $||A \cdot x|| \leq ||A|| \cdot ||x||$
  - this holds for all $A, B \in \mathbb{R}^{n^2}, x \in \mathbb{R}^n, \alpha \in \mathbb{R}, \alpha \geq 0$.

- Here we use: $||x|| = \sqrt{\sum_{i=1}^{n} |x_i|^2}$ for ein $x = (x_1, \ldots, x_n)$,

- Known: $||A|| = \text{Spectral Norm}(A) = \sqrt{|\lambda_{\text{max}}(A^T \cdot A)|}$ with: $\lambda_{\text{max}}$ is the largest Eigenvalue.
Recollection (Norm, 3. Abstraction)

Let $||.||$ be the vector norm over $\mathbb{R}^n$. Then we have:

- $||x|| = 0 \iff x = 0^n$,
- $||\alpha \cdot x|| = |\alpha| \cdot ||x||$,
- $||x + y|| \leq ||x|| + ||y||$
- this holds for all $\alpha \in \mathbb{R}, x, y \in \mathbb{R}^n$

The matrix norm for a vector norm $||.||$ is defined by $||A|| = \sup_{x \neq 0} \frac{||Ax||}{||x||}$. Then we have:

- $||A|| = 0 \iff A = 0$
- $||A + B|| \leq ||A|| + ||B||$
- $||\alpha A|| = \alpha \cdot ||A||$
- $||A \cdot B|| \leq ||A|| \cdot ||B||$
- $||A \cdot x|| \leq ||A|| \cdot ||x||$
- this holds for all $A, B \in \mathbb{R}^{n^2}, x \in \mathbb{R}^n, \alpha \in \mathbb{R}, \alpha \geq 0$.

Here we use: $||x|| = \sqrt{\sum_{i=1}^{n} |x_i|^2}$ for ein $x = (x_1, .., x_n)$,

Known: $||A|| = \text{Spectral Norm}(A) = \sqrt{\lambda_{\text{max}}(A^T \cdot A)}$ with: $\lambda_{\text{max}}$ is the largest Eigenvalue.
Recollection (Norm, 3. Abstraction)

- Let $||..||$ be the vector norm over $\mathbb{R}^n$. Then we have:
  - $||x|| = 0 \iff x = 0^n$,
  - $||\alpha \cdot x|| = |\alpha| \cdot ||x||$,
  - $||x + y|| \leq ||x|| + ||y||$
  - this holds for all $\alpha \in \mathbb{R}, x, y \in \mathbb{R}^n$

- The matrix norm for a vector norm $||..||$ is defined by $||A|| = \sup_{x \neq 0} \frac{||Ax||}{||x||}$.
  Then we have:
  - $||A|| = 0 \iff A = 0$
  - $||A + B|| \leq ||A|| + ||B||$
  - $||\alpha A|| = \alpha \cdot ||A||$
  - $||A \cdot B|| \leq ||A|| \cdot ||B||$
  - $||A \cdot x|| \leq ||A|| \cdot ||x||$
  - this holds for all $A, B \in \mathbb{R}^{n^2}, x \in \mathbb{R}^n, \alpha \in \mathbb{R}, \alpha \geq 0$.

- Here we use: $||x|| = \sqrt{\sum_{i=1}^{n} |x_i|^2}$ for ein $x = (x_1, .., x_n)$,

- Known: $||A|| = \text{Spectral Norm}(A) = \sqrt{\lambda_{\text{max}}(A^T \cdot A)}$ with: $\lambda_{\text{max}}$ is the largest Eigenvalue.
Recollection (Norm, 3. Abstraction)

- Let \(\|\cdot\|\) be the vector norm over \(\mathbb{R}^n\). Then we have:
  - \(\|x\| = 0 \iff x = 0^n\),
  - \(\|\alpha \cdot x\| = |\alpha| \cdot \|x\|\),
  - \(\|x + y\| \leq \|x\| + \|y\|\)
  - **this holds for all** \(\alpha \in \mathbb{R}, x, y \in \mathbb{R}^n\)

- The matrix norm for a vector norm \(\|\cdot\|\) is defined by \(\|A\| = \sup_{x \neq 0} \frac{\|Ax\|}{\|x\|}\).

  Then we have:
  - \(\|A\| = 0 \iff A = 0\)
  - \(\|A + B\| \leq \|A\| + \|B\|\)
  - \(\|\alpha A\| = \alpha \cdot \|A\|\)
  - \(\|A \cdot B\| \leq \|A\| \cdot \|B\|\)
  - \(\|A \cdot x\| \leq \|A\| \cdot \|x\|\)
  - **this holds for all** \(A, B \in \mathbb{R}^{n^2}, x \in \mathbb{R}^n, \alpha \in \mathbb{R}, \alpha \geq 0\).

- Here we use: \(\|x\| = \sqrt{\sum_{i=1}^{n} |x_i|^2}\) for ein \(x = (x_1, \ldots, x_n)\),

- Known: \(\|A\| = \text{Spectral Norm}(A) = \sqrt{\lambda_{\text{max}}(A^T \cdot A)}\) with: \(\lambda_{\text{max}}\) is the largest Eigenvalue.
Recollection (Norm, 3. Abstraction)

- Let \(|\ldots|\) be the vector norm over \(\mathbb{R}^n\). Then we have:
  - \(|x| = 0 \iff x = 0^n\),
  - \(|\alpha \cdot x| = |\alpha| \cdot |x|\),
  - \(|x + y| \leq |x| + |y|\)
  - this holds for all \(\alpha \in \mathbb{R}, x, y \in \mathbb{R}^n\)

- The matrix norm for a vector norm \(|\ldots|\) is defined by \(|A| = \sup_{x \neq 0} \frac{|Ax|}{|x|}\).
Then we have:
  - \(|A| = 0 \iff A = 0\)
  - \(|A + B| \leq |A| + |B|\)
  - \(|\alpha A| = \alpha \cdot |A|\)
  - \(|A \cdot B| \leq |A| \cdot |B|\)
  - \(|A \cdot x| \leq |A| \cdot |x|\)
  - this holds for all \(A, B \in \mathbb{R}^{n^2}, x \in \mathbb{R}^n, \alpha \in \mathbb{R}, \alpha \geq 0\).

Here we use: \(|x| = \sqrt{\sum_{i=1}^n |x_i|^2}\) for \(x = (x_1, \ldots, x_n)\),

- Known: \(|A| = \text{Spectral Norm}(A) = \sqrt{|\lambda_{\max}(A^T \cdot A)|}\) with: \(\lambda_{\max}\) is the largest Eigenvalue.
Recollection (Norm, 3. Abstraction)

- Let $||.||$ be the vector norm over $\mathbb{R}^n$. Then we have:
  - $||x|| = 0 \iff x = 0^n$,
  - $||\alpha \cdot x|| = |\alpha| \cdot ||x||$,
  - $||x + y|| \leq ||x|| + ||y||$
  - this holds for all $\alpha \in \mathbb{R}, x, y \in \mathbb{R}^n$

- The matrix norm for a vector norm $||.||$ is defined by $||A|| = sup_{x \neq 0} \frac{||Ax||}{||x||}$. Then we have:
  - $||A|| = 0 \iff A = 0$
  - $||A + B|| \leq ||A|| + ||B||$
  - $||\alpha A|| = |\alpha| \cdot ||A||$
  - $||A \cdot B|| \leq ||A|| \cdot ||B||$
  - $||A \cdot x|| \leq ||A|| \cdot ||x||$
  - this holds for all $A, B \in \mathbb{R}^{n^2}, x \in \mathbb{R}^n, \alpha \in \mathbb{R}, \alpha \geq 0$.

- Here we use: $||x|| = \sqrt{\sum_{i=1}^{n} |x_i|^2}$ for $x = (x_1, .., x_n)$,

- Known: $||A|| = \text{Spectral Norm}(A) = \sqrt{|\lambda_{max}(A^T \cdot A)|}$ with: $\lambda_{max}$ is the largest Eigenvalue.
Recollection (Norm, 3. Abstraction)

- Let \( ||.|| \) be the vector norm over \( \mathbb{R}^n \). Then we have:
  - \( ||x|| = 0 \iff x = 0^n \),
  - \( ||\alpha \cdot x|| = |\alpha| \cdot ||x|| \),
  - \( ||x + y|| \leq ||x|| + ||y|| \)
  - this holds for all \( \alpha \in \mathbb{R}, x, y \in \mathbb{R}^n \)

- The matrix norm for a vector norm \( ||.|| \) is defined by \( ||A|| = \sup_{x \neq 0} \frac{||Ax||}{||x||} \).
  Then we have:
  - \( ||A|| = 0 \iff A = 0 \)
  - \( ||A + B|| \leq ||A|| + ||B|| \)
  - \( ||\alpha A|| = |\alpha| \cdot ||A|| \)
  - \( ||A \cdot B|| \leq ||A|| \cdot ||B|| \)
  - \( ||A \cdot x|| \leq ||A|| \cdot ||x|| \)
  - this holds for all \( A, B \in \mathbb{R}^{n^2}, x \in \mathbb{R}^n, \alpha \in \mathbb{R}, \alpha \geq 0 \).

- Here we use: \( ||x|| = \sqrt{\sum_{i=1}^{n} |x_i|^2} \) for \( x = (x_1, \ldots, x_n) \),

- Known: \( ||A|| = \text{Spectral Norm}(A) = \sqrt{|\lambda_{max}(A^T \cdot A)|} \) with: \( \lambda_{max} \) is the largest Eigenvalue.
Recollection (Norm, 3. Abstraction)

- Let $||\cdot||$ be the vector norm over $\mathbb{R}^n$. Then we have:
  - $||x|| = 0 \iff x = 0^n$,
  - $||\alpha \cdot x|| = |\alpha| \cdot ||x||$,
  - $||x + y|| \leq ||x|| + ||y||$
  - this holds for all $\alpha \in \mathbb{R}, x, y \in \mathbb{R}^n$

- The matrix norm for a vector norm $||\cdot||$ is defined by $||A|| = \sup_{x \neq 0} \frac{||Ax||}{||x||}$. Then we have:
  - $||A|| = 0 \iff A = 0$
  - $||A + B|| \leq ||A|| + ||B||$
  - $||\alpha A|| = \alpha \cdot ||A||$
  - $||A \cdot B|| \leq ||A|| \cdot ||B||$
  - $||A \cdot x|| \leq ||A|| \cdot ||x||$
  - this holds for all $A, B \in \mathbb{R}^{n^2}, x \in \mathbb{R}^n, \alpha \in \mathbb{R}, \alpha \geq 0$.

- Here we use: $||x|| = \sqrt{\sum_{i=1}^{n} |x_i|^2}$ for ein $x = (x_1, \ldots, x_n)$,

- Known: $||A|| = \text{Spectral Norm}(A) = \sqrt{|\lambda_{max}(A^T \cdot A)|}$ with: $\lambda_{max}$ is the largest Eigenvalue.
Recollection (Norm, 3. Abstraction)

- Let $||\cdot||$ be the vector norm over $\mathbb{R}^n$. Then we have:
  - $||x|| = 0 \iff x = 0^n$,
  - $||\alpha \cdot x|| = |\alpha| \cdot ||x||$,
  - $||x + y|| \leq ||x|| + ||y||$
  - this holds for all $\alpha \in \mathbb{R}, x, y \in \mathbb{R}^n$

- The matrix norm for a vector norm $||\cdot||$ is defined by $||A|| = \sup_{x \neq 0} \frac{||Ax||}{||x||}$. Then we have:
  - $||A|| = 0 \iff A = 0$
  - $||A + B|| \leq ||A|| + ||B||$
  - $||\alpha A|| = |\alpha| \cdot ||A||$
  - $||A \cdot B|| \leq ||A|| \cdot ||B||$
  - $||A \cdot x|| \leq ||A|| \cdot ||x||$
  - this holds for all $A, B \in \mathbb{R}^{n^2}, x \in \mathbb{R}^n, \alpha \in \mathbb{R}, \alpha \geq 0$.

- Here we use: $||x|| = \sqrt{\sum_{i=1}^{n} |x_i|^2}$ for $\text{ein } x = (x_1, \ldots, x_n)$,

- Known: $||A|| = \text{Spectral Norm}(A) = \sqrt{|\lambda_{\max}(A^T \cdot A)|}$ with: $\lambda_{\max}$ is the largest Eigenvalue.
Recollection (Norm, 3. Abstraction)

- Let $||..||$ be the vector norm over $\mathbb{R}^n$. Then we have:
  - $||x|| = 0 \Leftrightarrow x = 0^n$
  - $||\alpha \cdot x|| = |\alpha| \cdot ||x||$
  - $||x + y|| \leq ||x|| + ||y||$
  - this holds for all $\alpha \in \mathbb{R}, x, y \in \mathbb{R}^n$

- The matrix norm for a vector norm $||..||$ is defined by $||A|| = \sup_{x \neq 0} \frac{||Ax||}{||x||}$.
Then we have:
  - $||A|| = 0 \Leftrightarrow A = 0$
  - $||A + B|| \leq ||A|| + ||B||$
  - $||\alpha A|| = \alpha \cdot ||A||$
  - $||A \cdot B|| \leq ||A|| \cdot ||B||$
  - $||A \cdot x|| \leq ||A|| \cdot ||x||$
  - this holds for all $A, B \in \mathbb{R}^{n^2}, x \in \mathbb{R}^n, \alpha \in \mathbb{R}, \alpha \geq 0$.

- Here we use: $||x|| = \sqrt{\sum_{i=1}^{n} |x_i|^2}$ for ein $x = (x_1, .., x_n)$,

- Known: $||A|| = \text{Spectral Norm}(A) = \sqrt{|\lambda_{\text{max}}(A^T \cdot A)|}$ with: $\lambda_{\text{max}}$ is the largest Eigenvalue.
Recollection (Norm, 3. Abstraction)

- Let \( ||..|| \) be the vector norm over \( \mathbb{R}^n \). Then we have:
  - \( ||x|| = 0 \iff x = 0^n \),
  - \( ||\alpha \cdot x|| = |\alpha| \cdot ||x|| \),
  - \( ||x + y|| \leq ||x|| + ||y|| \)
  - this holds for all \( \alpha \in \mathbb{R}, x, y \in \mathbb{R}^n \)

- The matrix norm for a vector norm \( ||..|| \) is defined by \( ||A|| = \sup_{x \neq 0} \frac{||Ax||}{||x||} \).
  Then we have:
  - \( ||A|| = 0 \iff A = 0 \)
  - \( ||A + B|| \leq ||A|| + ||B|| \)
  - \( ||\alpha A|| = \alpha \cdot ||A|| \)
  - \( ||A \cdot B|| \leq ||A|| \cdot ||B|| \)
  - \( ||A \cdot x|| \leq ||A|| \cdot ||x|| \)
  - this holds for all \( A, B \in \mathbb{R}^{n^2}, x \in \mathbb{R}^n, \alpha \in \mathbb{R}, \alpha \geq 0 \).

- Here we use: \( ||x|| = \sqrt{\sum_{i=1}^{n} |x_i|^2} \) for \( x = (x_1, ..., x_n) \),

- Known: \( ||A|| = \text{Spectral Norm}(A) = \sqrt{|\lambda_{max}(A^T \cdot A)|} \) with: \( \lambda_{max} \) is the largest Eigenvalue.
Recollection (Norm, 3. Abstraction)

- Let \( ||..|| \) be the vector norm over \( \mathbb{R}^n \). Then we have:
  - \( ||x|| = 0 \iff x = 0^n \),
  - \( ||\alpha \cdot x|| = |\alpha| \cdot ||x|| \),
  - \( ||x + y|| \leq ||x|| + ||y|| \)
  - this holds for all \( \alpha \in \mathbb{R}, x, y \in \mathbb{R}^n \)

- The matrix norm for a vector norm \( ||..|| \) is defined by \( ||A|| = \sup_{x \neq 0} \frac{||Ax||}{||x||} \).
  Then we have:
  - \( ||A|| = 0 \iff A = 0 \)
  - \( ||A + B|| \leq ||A|| + ||B|| \)
  - \( ||\alpha A|| = |\alpha| \cdot ||A|| \)
  - \( ||A \cdot B|| \leq ||A|| \cdot ||B|| \)
  - \( ||A \cdot x|| \leq ||A|| \cdot ||x|| \)
  - this holds for all \( A, B \in \mathbb{R}^{n^2}, x \in \mathbb{R}^n, \alpha \in \mathbb{R}, \alpha \geq 0 \).

- Here we use: \( ||x|| = \sqrt{\sum_{i=1}^{n} |x_i|^2} \) for ein \( x = (x_1, .., x_n) \),

- Known: \( ||A|| = \text{Spectral Norm}(A) = \sqrt{|\lambda_{\text{max}}(A^T \cdot A)|} \) with: \( \lambda_{\text{max}} \) is the largest Eigenvalue.
Recollection (Norm, 3. Abstraction)

- Let $||.||$ be the vector norm over $\mathbb{R}^n$. Then we have:
  - $||x|| = 0 \Leftrightarrow x = 0^n$,
  - $||\alpha \cdot x|| = |\alpha| \cdot ||x||$,
  - $||x + y|| \leq ||x|| + ||y||$
  - this holds for all $\alpha \in \mathbb{R}, x, y \in \mathbb{R}^n$

- The matrix norm for a vector norm $||.||$ is defined by $||A|| = \sup_{x \neq 0} \frac{||Ax||}{||x||}$. Then we have:
  - $||A|| = 0 \Leftrightarrow A = 0$
  - $||A + B|| \leq ||A|| + ||B||$
  - $||\alpha A|| = |\alpha| \cdot ||A||$
  - $||A \cdot B|| \leq ||A|| \cdot ||B||$
  - $||A \cdot x|| \leq ||A|| \cdot ||x||$
  - this holds for all $A, B \in \mathbb{R}^{n \times n}, x \in \mathbb{R}^n, \alpha \in \mathbb{R}, \alpha \geq 0$.

- Here we use: $||x|| = \sqrt{\sum_{i=1}^{n} |x_i|^2}$ for every $x = (x_1, .., x_n)$,

- Known: $||A|| = \text{Spectral Norm}(A) = \sqrt{|\lambda_{\text{max}}(A^T \cdot A)|}$ with: $\lambda_{\text{max}}$ is the largest Eigenvalue.
Recollection (Norm, 3. Abstraction)

Let $||.||$ be the vector norm over $\mathbb{R}^n$. Then we have:

- $||x|| = 0 \iff x = 0^n$,
- $||\alpha \cdot x|| = |\alpha| \cdot ||x||$,
- $||x + y|| \leq ||x|| + ||y||$

this holds for all $\alpha \in \mathbb{R}, x, y \in \mathbb{R}^n$

The matrix norm for a vector norm $||.||$ is defined by $||A|| = \sup_{x \neq 0} \frac{||Ax||}{||x||}$.

Then we have:

- $||A|| = 0 \iff A = 0$
- $||A + B|| \leq ||A|| + ||B||$
- $||\alpha A|| = \alpha \cdot ||A||$
- $||A \cdot B|| \leq ||A|| \cdot ||B||$
- $||A \cdot x|| \leq ||A|| \cdot ||x||$

this holds for all $A, B \in \mathbb{R}^{n \times n}, x \in \mathbb{R}^n, \alpha \in \mathbb{R}, \alpha \geq 0$.

Here we use: $||x|| = \sqrt{\sum_{i=1}^{n} |x_i|^2}$ for $x = (x_1, .., x_n)$,

Known: $||A|| = \text{Spectral Norm}(A) = \sqrt{\lambda_{max}(A^T \cdot A)}$ with: $\lambda_{max}$ is the largest Eigenvalue.
2. Abstraction (Continuation)

- We compute the spectral norm:
  - \( \|A\| = \|TAT^{-1}\| = \|B\| \).
  - \( B^T \cdot B = \begin{pmatrix} 10 \\ 11 \end{pmatrix} \begin{pmatrix} 11 \\ 01 \end{pmatrix} = \begin{pmatrix} 11 \\ 12 \end{pmatrix} \).
  - \( (2 - \lambda)(1 - \lambda) - 1 = 0 \)
  - \( \lambda^2 - 3\lambda + 1 = 0 \)
  - \( \lambda_{max}(B^T B) = \frac{3}{2} + \sqrt{\frac{5}{4}} \)
  - \( \|A\| = \sqrt{\lambda_{max}(A^T A)} = \frac{1}{2}(1 + \sqrt{5}) \)
2. Abstraction (Continuation)

- We compute the spectral norm:
  \[ \|A\| = \|TAT^{-1}\| = \|B\|. \]

- \[ B^T \cdot B = \begin{pmatrix} 10 \\ 11 \end{pmatrix} \begin{pmatrix} 11 \\ 01 \end{pmatrix} = \begin{pmatrix} 11 \\ 12 \end{pmatrix}. \]

- \[ \Rightarrow (2 - \lambda)(1 - \lambda) - 1 = 0 \]
- \[ \Rightarrow \lambda^2 - 3\lambda + 1 = 0 \]
- \[ \Rightarrow \lambda_{\text{max}}(B^T B) = \frac{3}{2} + \sqrt{\frac{5}{4}} \]
- \[ \Rightarrow \|A\| = \sqrt{\lambda_{\text{max}}(A^T A)} = \frac{1}{2}(1 + \sqrt{5}) \]
2. Abstraction (Continuation)

- We compute the spectral norm:
  - $||A|| = ||TAT^{-1}|| = ||B||$.
  - $B^T \cdot B = \begin{pmatrix} 10 \\ 11 \end{pmatrix} \begin{pmatrix} 11 \\ 01 \end{pmatrix} = \begin{pmatrix} 11 \\ 12 \end{pmatrix}$.
  - $\Rightarrow (2 - \lambda)(1 - \lambda) - 1 = 0$
  - $\Rightarrow \lambda^2 - 3\lambda + 1 = 0$
  - $\Rightarrow \lambda_{\text{max}}(B^T B) = \frac{3}{2} + \sqrt{\frac{5}{4}}$
  - $||A|| = \sqrt{\lambda_{\text{max}}(A^T A)} = \frac{1}{2}(1 + \sqrt{5})$
2. Abstraction (Continuation)

- We compute the spectral norm:

\[ ||A|| = ||TAT^{-1}|| = ||B||. \]

\[ B^T \cdot B = \begin{pmatrix} 10 \\ 11 \end{pmatrix} \begin{pmatrix} 11 \\ 01 \end{pmatrix} = \begin{pmatrix} 11 \\ 12 \end{pmatrix}. \]

\[ \Rightarrow (2 - \lambda)(1 - \lambda) - 1 = 0 \]

\[ \Rightarrow \lambda^2 - 3\lambda + 1 = 0 \]

\[ \Rightarrow \lambda_{\text{max}}(B^T B) = \frac{3}{2} + \sqrt{\frac{5}{4}} \]

\[ ||A|| = \sqrt{\lambda_{\text{max}}(A^T A)} = \frac{1}{2}(1 + \sqrt{5}) \]
2. Abstraction (Continuation)

- We compute the spectral norm:
  \[ \|A\| = \|TA\|^{-1} = \|B\|. \]

- \[ B^T \cdot B = \begin{pmatrix} 10 \\ 11 \end{pmatrix} \begin{pmatrix} 11 \\ 01 \end{pmatrix} = \begin{pmatrix} 11 \\ 12 \end{pmatrix}. \]

- \[ \Rightarrow (2 - \lambda)(1 - \lambda) - 1 = 0 \]

- \[ \Rightarrow \lambda^2 - 3\lambda + 1 = 0 \]

- \[ \Rightarrow \lambda_{\text{max}}(B^T B) = \frac{3}{2} + \sqrt{\frac{5}{4}} \]

- \[ \|A\| = \sqrt{\lambda_{\text{max}}(A^T A)} = \frac{1}{2}(1 + \sqrt{5}) \]
2. Abstraction (Continuation)

- We compute the spectral norm:
  
  \[ \|A\| = \|TAT^{-1}\| = \|B\|. \]

  \[ B^T \cdot B = \begin{pmatrix} 10 & 11 \\ 11 & 01 \end{pmatrix} = \begin{pmatrix} 11 \\ 12 \end{pmatrix}. \]

- \( (2 - \lambda)(1 - \lambda) - 1 = 0 \)
- \( \lambda^2 - 3\lambda + 1 = 0 \)
- \( \Rightarrow \lambda_{\text{max}}(B^T B) = \frac{3}{2} + \sqrt{\frac{5}{4}} \)
- \( \|A\| = \sqrt{\lambda_{\text{max}}(A^T A)} = \frac{1}{2}(1 + \sqrt{5}) \)
2. Abstraction (Continuation)

- We compute the spectral norm:
  
  \[ ||A|| = ||TAT^{-1}|| = ||B||.\]
  
  \[ B^T \cdot B = \begin{pmatrix} 10 \\ 11 \end{pmatrix} \begin{pmatrix} 11 \\ 01 \end{pmatrix} = \begin{pmatrix} 11 \\ 12 \end{pmatrix}.\]

- \[ (2 - \lambda)(1 - \lambda) - 1 = 0 \]
- \[ \lambda^2 - 3\lambda + 1 = 0 \]
- \[ \lambda_{\text{max}}(B^T B) = \frac{3}{2} + \sqrt{\frac{5}{4}} \]

- \[ ||A|| = \sqrt{\lambda_{\text{max}}(A^T A)} = \frac{1}{2} (1 + \sqrt{5}) \]
2. Abstraction (Continuation)

- We compute the spectral norm:

\[ ||A|| = ||TAT^{-1}|| = ||B||. \]

\[ \begin{bmatrix} 10 \\ 11 \end{bmatrix} \begin{bmatrix} 11 \\ 01 \end{bmatrix} = \begin{bmatrix} 11 \\ 12 \end{bmatrix}. \]

\[ \Rightarrow (2 - \lambda)(1 - \lambda) - 1 = 0 \]

\[ \Rightarrow \lambda^2 - 3\lambda + 1 = 0 \]

\[ \Rightarrow \lambda_{\max}(B^T B) = \frac{3}{2} + \sqrt{\frac{5}{4}} \]

\[ ||A|| = \sqrt{\lambda_{\max}(A^T A)} = \frac{1}{2}(1 + \sqrt{5}) \]
Theorem:

A algorithm, solving the network counting problem needs \(2 + \lceil \log_{\frac{1}{2}}(1+\sqrt{5}) \frac{n}{2} \rceil\) rounds.

Proof:

- Let \(A_j, 1 \leq j \leq r\) be matrices, which solve the problem in \(r\) rounds.
- \(\alpha := (\alpha_1, \alpha_2, \cdots, \alpha_n)^T = A_{r-2} \cdot \cdots \cdot A_2 \cdot A_1 \cdot (1,1,\cdots,1)\).
- \(\|\alpha\| \leq (\prod_{i=1}^{r-2} ||A_i||) \cdot ||(1,\ldots,1)|| \leq (\frac{1}{2}(1 + \sqrt{5}))^{r-2} \cdot \sqrt{n}\)
- Let \(\text{inf}(i, t)\) be the number, which have the nodes \(v_i\) after \(t\) rounds.
- After round \(t\) we have: \(\text{inf}(i, t) \geq n\) for all \(i \in \{1, 2, \cdots, n\}\).
- After round \(t - 1\) we have: \(\text{inf}(i, t - 1) \geq n\) for at least \(n/2\) nodes.
- There could be some \(i\) with: \(\text{inf}(i, t - 2) \geq n\).
- But if \(\alpha_i < n\) and \(\text{inf}(i, t - 1) \geq n\), then there exists \(j\) with: \(\alpha_i + \alpha_j \geq n\).
Theorem:

A algorithm, solving the network counting problem needs \( 2 + \lceil \log_2 \left( \frac{1}{2} (1 + \sqrt{5}) \right) \frac{n}{2} \rceil \) rounds.

Proof:

- Let \( A_j \), \( 1 \leq j \leq r \) be matrices, which solve the problem in \( r \) rounds.
- \( \alpha := (\alpha_1, \alpha_2, \cdots, \alpha_n)^T = A_{r-2} \cdots A_2 \cdot A_1 \cdot (1, 1, \cdots, 1) \).
- \( \|\alpha\| \leq \left( \prod_{i=1}^{r-2} \|A_i\| \right) \cdot \|(1, \ldots, 1)\| \leq \left( \frac{1}{2} (1 + \sqrt{5}) \right)^{r-2} \cdot \sqrt{n} \)
- Let \( \inf(i, t) \) be the number, which have the nodes \( v_i \) after \( t \) rounds.
- After round \( t \) we have: \( \inf(i, t) \geq n \) for all \( i \in \{1, 2, \cdots, n\} \).
- After round \( t - 1 \) we have: \( \inf(i, t - 1) \geq n \) for at least \( n/2 \) nodes.
- There could be some \( i \) with: \( \inf(i, t - 2) \geq n \).
- But if \( \alpha_i < n \) and \( \inf(i, t - 1) \geq n \), then there exists \( j \) with: \( \alpha_i + \alpha_j \geq n \).
Theorem:

A algorithm, solving the network counting problem needs $2 + \left\lceil \log_2 \left( \frac{1 + \sqrt{5}}{2} n \right) \right\rceil$ rounds.

Proof:

- Let $A_j, 1 \leq j \leq r$ be matrices, which solve the problem in $r$ rounds.
- $\alpha := (\alpha_1, \alpha_2, \ldots, \alpha_n)^T = A_{r-2} \cdot \ldots \cdot A_2 \cdot A_1 \cdot (1, 1, \ldots, 1)$.
- $||\alpha|| \leq \left( \prod_{i=1}^{r-2} ||A_i|| \right) \cdot ||(1, \ldots, 1)|| \leq \left( \frac{1}{2} (1 + \sqrt{5}) \right)^{r-2} \cdot \sqrt{n}$

- Let $inf(i, t)$ be the number, which have the nodes $v_i$ after $t$ rounds.
- After round $t$ we have: $inf(i, t) \geq n$ for all $i \in \{1, 2, \ldots, n\}$.
- After round $t - 1$ we have: $inf(i, t - 1) \geq n$ for at least $n/2$ nodes.
- There could be some $i$ with: $inf(i, t - 2) \geq n$.
- But if $\alpha_i < n$ and $inf(i, t - 1) \geq n$, then there exists $j$ with: $\alpha_i + \alpha_j \geq n$. 
Theorem:

A algorithm, solving the network counting problem needs \( 2 + \lceil \log_{\frac{1}{2}}(1 + \sqrt{5}) \frac{n}{2} \rceil \) rounds.

Proof:

- Let \( A_j, 1 \leq j \leq r \) be matrices, which solve the problem in \( r \) rounds.
- \( \alpha := (\alpha_1, \alpha_2, \cdots, \alpha_n)^T = A_{r-2} \cdots A_2 \cdot A_1 \cdot (1, 1, \cdots, 1) \).
- \( \|\alpha\| \leq (\prod_{i=1}^{r-2} \|A_i\|) \cdot \|(1, \ldots, 1)\| \leq (\frac{1}{2}(1 + \sqrt{5}))^{r-2} \cdot \sqrt{n} \)
- Let \( \text{inf}(i, t) \) be the number, which have the nodes \( v_i \) after \( t \) rounds.
- After round \( t \) we have: \( \text{inf}(i, t) \geq n \) for all \( i \in \{1, 2, \cdots, n\} \).
- After round \( t - 1 \) we have: \( \text{inf}(i, t - 1) \geq n \) for at least \( n/2 \) nodes.
- There could be some \( i \) with: \( \text{inf}(i, t - 2) \geq n \).
- But if \( \alpha_i < n \) and \( \text{inf}(i, t - 1) \geq n \), then there exists \( j \) with: \( \alpha_i + \alpha_j \geq n \).
Theorem:

A algorithm, solving the network counting problem needs $2 + \left\lceil \log_2 \left( \frac{1 + \sqrt{5}}{2} \right) n \right\rceil$ rounds.

Proof:

- Let $A_j, 1 \leq j \leq r$ be matrices, which solve the problem in $r$ rounds.
- $\alpha := (\alpha_1, \alpha_2, \cdots, \alpha_n)^T = A_{r-2} \cdots A_2 \cdot A_1 \cdot (1, 1, \cdots, 1)$.
- $||\alpha|| \leq \left( \prod_{i=1}^{r-2} ||A_i|| \right) \cdot ||(1, \ldots, 1)|| \leq \left( \frac{1}{2} (1 + \sqrt{5}) \right)^{r-2} \cdot \sqrt{n}$

- Let $inf(i, t)$ be the number, which have the nodes $v_i$ after $t$ rounds.
- After round $t$ we have: $inf(i, t) \geq n$ for all $i \in \{1, 2, \cdots, n\}$.
- After round $t - 1$ we have: $inf(i, t - 1) \geq n$ for at least $n/2$ nodes.
- There could be some $i$ with: $inf(i, t - 2) \geq n$.
- But if $\alpha_i < n$ and $inf(i, t - 1) \geq n$, then there exists $j$ with: $\alpha_i + \alpha_j \geq n$.
Theorem:

A algorithm, solving the network counting problem needs \(2 + \lceil \log_{\frac{1}{2}}(1+\sqrt{5}) \cdot \frac{n}{2} \rceil\) rounds.

Proof:

1. Let \(A_j, 1 \leq j \leq r\) be matrices, which solve the problem in \(r\) rounds.
2. \(\alpha := (\alpha_1, \alpha_2, \cdots, \alpha_n)^T = A_{r-2} \cdots A_2 \cdot A_1 \cdot (1, 1, \cdots, 1)\).
3. \(\|\alpha\| \leq \left(\prod_{i=1}^{r-2} \|A_i\|\right) \cdot \|(1, \ldots, 1)\| \leq \left(\frac{1}{2}(1 + \sqrt{5})\right)^{r-2} \cdot \sqrt{n}\)
4. Let \(\text{inf}(i, t)\) be the number, which have the nodes \(v_i\) after \(t\) rounds.
5. After round \(t\) we have: \(\text{inf}(i, t) \geq n\) for all \(i \in \{1, 2, \cdots, n\}\).
6. After round \(t-1\) we have: \(\text{inf}(i, t-1) \geq n\) for at least \(n/2\) nodes.
7. There could be some \(i\) with: \(\text{inf}(i, t-2) \geq n\).
8. But if \(\alpha_i < n\) and \(\text{inf}(i, t-1) \geq n\), then there exists \(j\) with: \(\alpha_i + \alpha_j \geq n\).
Theorem:

A algorithm, solving the network counting problem needs \(2 + \lceil \log_{\frac{1}{2}}(1+\sqrt{5}) \frac{n}{2} \rceil\) rounds.

Proof:

- Let \(A_j, 1 \leq j \leq r\) be matrices, which solve the problem in \(r\) rounds.
- \(\alpha := (\alpha_1, \alpha_2, \cdots, \alpha_n)^T = A_{r-2} \cdots A_2 A_1 (1, 1, \cdots, 1)\).
- \(||\alpha|| \leq (\prod_{i=1}^{r-2} ||A_i||) \cdot ||(1, \ldots, 1)|| \leq (\frac{1}{2}(1 + \sqrt{5}))^{r-2} \cdot \sqrt{n}\)
- Let \(\inf(i, t)\) be the number, which have the nodes \(v_i\) after \(t\) rounds.
- After round \(t\) we have: \(\inf(i, t) \geq n\) for all \(i \in \{1, 2, \cdots, n\}\).
- After round \(t - 1\) we have: \(\inf(i, t - 1) \geq n\) for at least \(n/2\) nodes.
- There could be some \(i\) with: \(\inf(i, t - 2) \geq n\).
- But if \(\alpha_i < n\) and \(\inf(i, t - 1) \geq n\), then there exists \(j\) with: \(\alpha_i + \alpha_j \geq n\).
Theorem:

A algorithm, solving the network counting problem needs $2 + \lceil \log_{\frac{1}{2}} \left( 1 + \sqrt{5} \right) \frac{n}{2} \rceil$ rounds.

Proof:

- Let $A_j, 1 \leq j \leq r$ be matrices, which solve the problem in $r$ rounds.
- $\alpha := (\alpha_1, \alpha_2, \ldots, \alpha_n)^T = A_{r-2} \cdot \ldots \cdot A_2 \cdot A_1 \cdot (1, 1, \ldots, 1)$.
- $\|\alpha\| \leq \left( \prod_{i=1}^{r-2} \|A_i\| \right) \cdot \|(1, \ldots, 1)\| \leq \left( \frac{1}{2} (1 + \sqrt{5}) \right)^{r-2} \cdot \sqrt{n}$
- Let $inf(i, t)$ be the number, which have the nodes $v_i$ after $t$ rounds.
- After round $t$ we have: $inf(i, t) \geq n$ for all $i \in \{1, 2, \ldots, n\}$.
- After round $t-1$ we have: $inf(i, t-1) \geq n$ for at least $n/2$ nodes.
- There could be some $i$ with: $inf(i, t-2) \geq n$.
- But if $\alpha_i < n$ and $inf(i, t-1) \geq n$, then there exists $j$ with: $\alpha_i + \alpha_j \geq n$. 
Theorem:

A algorithm, solving the network counting problem needs $2 + \lceil \log_{\frac{1}{2}}(1+\sqrt{5}) \cdot \frac{n}{2} \rceil$ rounds.

Proof:

- Let $A_j, 1 \leq j \leq r$ be matrices, which solve the problem in $r$ rounds.
- $\alpha := (\alpha_1, \alpha_2, \cdots, \alpha_n)^T = A_{r-2} \cdot \cdots \cdot A_2 \cdot A_1 \cdot (1, 1, \cdots, 1)$.
- $||\alpha|| \leq (\prod_{i=1}^{r-2} ||A_i||) \cdot ||(1,\ldots,1)|| \leq (\frac{1}{2}(1+\sqrt{5}))^{r-2} \cdot \sqrt{n}$
- Let $inf(i,t)$ be the number, which have the nodes $v_i$ after $t$ rounds.
- After round $t$ we have: $inf(i,t) \geq n$ for all $i \in \{1,2,\cdots,n\}$.
- After round $t-1$ we have: $inf(i,t-1) \geq n$ for at least $n/2$ nodes.
- There could be some $i$ with: $inf(i,t-2) \geq n$.
- But if $\alpha_i < n$ and $inf(i,t-1) \geq n$, then there exists $j$ with: $\alpha_i + \alpha_j \geq n$. 
Theorem:

A algorithm, solving the network counting problem needs $2 + \lceil \log_{\frac{1}{2}}(1 + \sqrt{5}) \frac{n}{2} \rceil$ rounds.

Proof:

- Let $A_j, 1 \leq j \leq r$ be matrices, which solve the problem in $r$ rounds.
- $\alpha := (\alpha_1, \alpha_2, \cdots, \alpha_n)^T = A_{r-2} \cdot \cdots \cdot A_2 \cdot A_1 \cdot (1, 1, \cdots, 1)$.
- $||\alpha|| \leq (\prod_{i=1}^{r-2} ||A_i||) \cdot ||(1, \ldots, 1)|| \leq (\frac{1}{2}(1 + \sqrt{5}))^{r-2} \cdot \sqrt{n}$
- Let $\inf(i, t)$ be the number, which have the nodes $v_i$ after $t$ rounds.
- After round $t$ we have: $\inf(i, t) \geq n$ for all $i \in \{1, 2, \cdots, n\}$.
- After round $t-1$ we have: $\inf(i, t-1) \geq n$ for at least $n/2$ nodes.
- There could be some $i$ with: $\inf(i, t-2) \geq n$.
- But if $\alpha_i < n$ and $\inf(i, t-1) \geq n$, then there exists $j$ with: $\alpha_i + \alpha_j \geq n$. 
Continuation

\[ \alpha := (\alpha_1, \alpha_2, \ldots, \alpha_n)^T = A_{r-2} \cdots A_2 A_1 \cdot (1, 1, \ldots, 1) \]

- Let
  - \( c_1 \) be the number of cases with: \( \alpha_i \geq n \),
  - \( c_2 \) be the number of cases with: \( \alpha_i < n \) and \( \alpha_j \geq n \),
  - \( c_3 \) be the number of cases with: \( \alpha_i < n, \alpha_j < n \) and \( \alpha_i + \alpha_j \geq n \).

- Then we have: \( c_1 \geq c_2 \) and \( c_1 + c_2 + c_3 \geq n/2 \).

- Thus we also get: \( 2c_1 + c_3 \geq \frac{n}{2} \)

- \[ ||\alpha|| = \sqrt{\sum_{i=1}^{n} \alpha_i^2} \geq \sqrt{c_1 n^2 + c_3 \cdot 2 \cdot \frac{n^2}{4}} \geq n \cdot \sqrt{\frac{1}{2} (2c_1 + c_3)} \geq \frac{n}{2} \sqrt{n}. \]

- We already have:
  \[ ||\alpha|| \leq \left( \prod_{i=1}^{r-2} ||A_i|| \right) \cdot ||(1, \ldots, 1)|| \leq (\frac{1}{2} (1 + \sqrt{5}))^{r-2} \cdot \sqrt{n}. \]

- And we get:
  \[ \frac{n}{2} \cdot \sqrt{n} \leq ||\alpha|| \leq \Phi^{r-2} \cdot \sqrt{n}, \]

- From which we conclude:
  \[ r \geq 2 + \left[ \log_{\frac{1}{2} (1 + \sqrt{5})} \frac{n}{2} \right] \]
Continuation

\[ \alpha := (\alpha_1, \alpha_2, \ldots, \alpha_n)^T = A_{r-2} \cdots A_2 \cdot A_1 \cdot (1, 1, \ldots, 1) \]

- Let
  - \( c_1 \) be the number of cases with: \( \alpha_i \geq n \),
  - \( c_2 \) be the number of cases with: \( \alpha_i < n \) and \( \alpha_j \geq n \),
  - \( c_3 \) be the number of cases with: \( \alpha_i < n, \alpha_j < n \) and \( \alpha_i + \alpha_j \geq n \).

- Then we have: \( c_1 \geq c_2 \) and \( c_1 + c_2 + c_3 \geq n/2 \).

- Thus we also get: \( 2c_1 + c_3 \geq \frac{n}{2} \).

- \( ||\alpha|| = \sqrt{\sum_{i=1}^{n} \alpha_i^2} \geq \sqrt{c_1 n^2 + c_3 \cdot 2 \cdot \frac{n^2}{4}} \geq n \cdot \sqrt{\frac{1}{2}(2c_1 + c_3)} \geq \frac{n}{2} \sqrt{n} \).

- We already have:
  \[ ||\alpha|| \leq (\prod_{i=1}^{r-2} ||A_i||) \cdot ||(1, \ldots, 1)|| \leq \left( \frac{1}{2}(1 + \sqrt{5}) \right)^{r-2} \cdot \sqrt{n} \).

- And we get:
  \[ \frac{n}{2} \cdot \sqrt{n} \leq ||\alpha|| \leq \Phi^{r-2} \cdot \sqrt{n}, \]

- From which we conclude:
  \[ r \geq 2 + \left[ \log_{\frac{1}{2}(1+\sqrt{5})} \frac{n}{2} \right] \]
Continuation

\[ \alpha := (\alpha_1, \alpha_2, \cdots, \alpha_n)^T = A_{r-2} \cdots A_2 A_1 (1, 1, \cdots, 1) \]

- Let
  - \( c_1 \) be the number of cases with: \( \alpha_i \geq n \),
  - \( c_2 \) be the number of cases with: \( \alpha_i < n \) and \( \alpha_j \geq n \),
  - \( c_3 \) be the number of cases with: \( \alpha_i < n \), \( \alpha_j < n \) and \( \alpha_i + \alpha_j \geq n \).

- Then we have: \( c_1 \geq c_2 \) and \( c_1 + c_2 + c_3 \geq n/2 \).
- Thus we also get: \( 2c_1 + c_3 \geq \frac{n}{2} \)

\[ \|\alpha\| = \sqrt{\sum_{i=1}^{n} \alpha_i^2} \geq \sqrt{c_1 n^2 + c_3 \cdot 2 \cdot \frac{n^2}{4}} \geq n \cdot \sqrt{\frac{1}{2} (2c_1 + c_3)} \geq \frac{n}{2} \sqrt{n}. \]

- We already have:
  \[ \|\alpha\| \leq \prod_{i=1}^{r-2} \|A_i\| \cdot \|(1, \ldots, 1)\| \leq (\frac{1}{2} (1 + \sqrt{5}))^{r-2} \cdot \sqrt{n}. \]

- And we get:
  \[ \frac{n}{2} \cdot \sqrt{n} \leq \|\alpha\| \leq \Phi^{r-2} \cdot \sqrt{n}, \]

- From which we conclude:
  \[ r \geq 2 + \left[ \log_{\frac{1}{2} (1 + \sqrt{5})} \frac{n}{2} \right] \]
Let

- $c_1$ be the number of cases with: $\alpha_i \geq n$,
- $c_2$ be the number of cases with: $\alpha_i < n$ and $\alpha_j \geq n$,
- $c_3$ be the number of cases with: $\alpha_i < n$, $\alpha_j < n$ and $\alpha_i + \alpha_j \geq n$.

Then we have: $c_1 \geq c_2$ and $c_1 + c_2 + c_3 \geq n/2$.

Thus we also get: $2c_1 + c_3 \geq \frac{n}{2}$

\[ \|\alpha\| = \sqrt{\sum_{i=1}^{n} \alpha_i^2} \geq \sqrt{c_1 n^2 + c_3 \cdot 2 \cdot \frac{n^2}{4}} \geq n \cdot \sqrt{\frac{1}{2} (2c_1 + c_3)} \geq \frac{n}{2} \sqrt{n}. \]

We already have:

\[ \|\alpha\| \leq \prod_{i=1}^{r-2} \|A_i\| \cdot \|(1, ..., 1)\| \leq \left(\frac{1}{2} (1 + \sqrt{5})\right)^{r-2} \cdot \sqrt{n}. \]

And we get:

\[ \frac{n}{2} \cdot \sqrt{n} \leq \|\alpha\| \leq \Phi^{r-2} \cdot \sqrt{n}, \]

From which we conclude:

\[ r \geq 2 + \left\lceil \log_{\frac{1}{2} (1 + \sqrt{5})} \frac{n}{2} \right\rceil \]
Continuation

\[ \alpha := (\alpha_1, \alpha_2, \cdots, \alpha_n)^T = A_{r-2} \cdots A_2 A_1 \cdot (1, 1, \cdots, 1) \]

- Let
  - \( c_1 \) be the number of cases with: \( \alpha_i \geq n \),
  - \( c_2 \) be the number of cases with: \( \alpha_i < n \) and \( \alpha_j \geq n \),
  - \( c_3 \) be the number of cases with: \( \alpha_i < n \), \( \alpha_j < n \) and \( \alpha_i + \alpha_j \geq n \).

- Then we have: \( c_1 \geq c_2 \) and \( c_1 + c_2 + c_3 \geq n/2 \).
- Thus we also get: \( 2c_1 + c_3 \geq \frac{n}{2} \)
- \( ||\alpha|| = \sqrt{\sum_{i=1}^{n} \alpha_i^2} \geq \sqrt{c_1 n^2 + c_3 \cdot 2 \cdot \frac{n^2}{4}} \geq n \cdot \sqrt{\frac{1}{2} (2c_1 + c_3)} \geq \frac{n}{2} \sqrt{n} \).
- We already have:
  \[ ||\alpha|| \leq (\prod_{i=1}^{r-2} ||A_i||) \cdot ||(1, \ldots, 1)|| \leq (\frac{1}{2} (1 + \sqrt{5}))^{r-2} \cdot \sqrt{n}. \]
- And we get:
  \[ \frac{n}{2} \cdot \sqrt{n} \leq ||\alpha|| \leq \Phi^{r-2} \cdot \sqrt{n}, \]
- From which we conclude:
  \[ r \geq 2 + \left\lceil \log_\frac{1}{2} (1+\sqrt{5}) \frac{n}{2} \right\rceil \]
Continuation

\[ \alpha := (\alpha_1, \alpha_2, \cdots, \alpha_n)^T = A_{r-2} \cdots A_2 \cdot A_1 \cdot (1, 1, \cdots, 1) \]

- Let
  - \( c_1 \) be the number of cases with: \( \alpha_i \geq n \),
  - \( c_2 \) be the number of cases with: \( \alpha_i < n \) and \( \alpha_j \geq n \),
  - \( c_3 \) be the number of cases with: \( \alpha_i < n, \alpha_j < n \) and \( \alpha_i + \alpha_j \geq n \).

- Then we have: \( c_1 \geq c_2 \) and \( c_1 + c_2 + c_3 \geq n/2 \).

- Thus we also get: \( 2c_1 + c_3 \geq \frac{n}{2} \)

\[ ||\alpha|| = \sqrt{\sum_{i=1}^{n} \alpha_i^2} \geq \sqrt{c_1 n^2 + c_3 \cdot 2 \cdot \frac{n^2}{4}} \geq n \cdot \sqrt{\frac{1}{2} (2c_1 + c_3)} \geq \frac{n}{2} \sqrt{n}. \]

- We already have:

\[ ||\alpha|| \leq (\prod_{i=1}^{r-2} ||A_i||) \cdot ||(1, \ldots, 1)|| \leq \left( \frac{1}{2} (1 + \sqrt{5}) \right)^{r-2} \cdot \sqrt{n}. \]

- And we get:

\[ \frac{n}{2} \cdot \sqrt{n} \leq ||\alpha|| \leq \Phi^{r-2} \cdot \sqrt{n}, \]

- From which we conclude:

\[ r \geq 2 + \left[ \log_\frac{1}{2} (1+\sqrt{5}) \frac{n}{2} \right] \]
Continuation

\[ \alpha := (\alpha_1, \alpha_2, \ldots, \alpha_n)^T = A_{r-2} \cdots A_2 A_1 \cdot (1, 1, \ldots, 1) \]

- Let
  - \( c_1 \) be the number of cases with: \( \alpha_i \geq n \),
  - \( c_2 \) be the number of cases with: \( \alpha_i < n \) and \( \alpha_j \geq n \),
  - \( c_3 \) be the number of cases with: \( \alpha_i < n, \alpha_j < n \) and \( \alpha_i + \alpha_j \geq n \).

- Then we have: \( c_1 \geq c_2 \) and \( c_1 + c_2 + c_3 \geq n/2 \).
- Thus we also get: \( 2c_1 + c_3 \geq \frac{n}{2} \).

- \( \|\alpha\| = \sqrt{\sum_{i=1}^{n} \alpha_i^2} \geq \sqrt{c_1 n^2 + c_3 \cdot 2 \cdot \frac{n^2}{4}} \geq \sqrt{n} \cdot \sqrt{\frac{1}{2} (2c_1 + c_3)} \geq \frac{n}{2} \sqrt{n} \).

- We already have:
  \[ \|\alpha\| \leq (\prod_{i=1}^{r-2} ||A_i||) \cdot ||(1, \ldots, 1)|| \leq (\frac{1}{2} (1 + \sqrt{5}))^{r-2} \cdot \sqrt{n} \].

- And we get:
  \[ \frac{n}{2} \cdot \sqrt{n} \leq \|\alpha\| \leq \Phi^{r-2} \cdot \sqrt{n} \],

- From which we conclude:
  \[ r \geq 2 + \left\lceil \log_{\frac{1}{2} \left( 1 + \sqrt{5} \right)} \frac{n}{2} \right\rceil \]
Continuation

\[ \alpha := (\alpha_1, \alpha_2, \ldots, \alpha_n)^T = A_{r-2} \cdots A_2 \cdot A_1 \cdot (1, 1, \ldots, 1) \]

- Let
  - \( c_1 \) be the number of cases with: \( \alpha_i \geq n \),
  - \( c_2 \) be the number of cases with: \( \alpha_i < n \) and \( \alpha_j \geq n \),
  - \( c_3 \) be the number of cases with: \( \alpha_i < n \), \( \alpha_j < n \) and \( \alpha_i + \alpha_j \geq n \).

- Then we have: \( c_1 \geq c_2 \) and \( c_1 + c_2 + c_3 \geq n/2 \).

- Thus we also get: \( 2c_1 + c_3 \geq \frac{n}{2} \)

- \( \|\alpha\| = \sqrt{\sum_{i=1}^{n} \alpha_i^2} \geq \sqrt{c_1n^2 + c_3 \cdot 2 \cdot \frac{n^2}{4}} \geq n \cdot \sqrt{\frac{1}{2}(2c_1 + c_3)} \geq \frac{n}{2} \sqrt{n} \).

- We already have:
  \[ \|\alpha\| \leq \left( \prod_{i=1}^{r-2} \|A_i\| \right) \cdot \|(1, \ldots, 1)\| \leq (\frac{1}{2}(1 + \sqrt{5}))^{r-2} \cdot \sqrt{n}. \]

- And we get:
  \[ \frac{n}{2} \cdot \sqrt{n} \leq \|\alpha\| \leq \Phi^{r-2} \cdot \sqrt{n}, \]

- From which we conclude:
  \[ r \geq 2 + \left[ \log_{\frac{1}{2}(1+\sqrt{5})} \frac{n}{2} \right] \]
Continuation

\[ \alpha := (\alpha_1, \alpha_2, \cdots, \alpha_n)^T = A_{r-2} \cdots A_2 A_1 \cdot (1, 1, \cdots, 1) \]

- Let
  - \( c_1 \) be the number of cases with: \( \alpha_i \geq n \),
  - \( c_2 \) be the number of cases with: \( \alpha_i < n \) and \( \alpha_j \geq n \),
  - \( c_3 \) be the number of cases with: \( \alpha_i < n \), \( \alpha_j < n \) and \( \alpha_i + \alpha_j \geq n \).

- Then we have: \( c_1 \geq c_2 \) and \( c_1 + c_2 + c_3 \geq n/2 \).

- Thus we also get: \( 2c_1 + c_3 \geq \frac{n}{2} \)

- \[ ||\alpha|| = \sqrt{\sum_{i=1}^{n} \alpha_i^2} \geq \sqrt{c_1 n^2 + c_3 \cdot 2 \cdot \frac{n^2}{4}} \geq n \cdot \sqrt{\frac{1}{2}(2c_1 + c_3)} \geq \frac{n}{2} \sqrt{n}. \]

- We already have:
  \[ ||\alpha|| \leq (\prod_{i=1}^{r-2} ||A_i||) \cdot ||(1, \ldots, 1)|| \leq (\frac{1}{2}(1 + \sqrt{5}))^{r-2} \cdot \sqrt{n}. \]

- And we get:
  \[ \frac{n}{2} \cdot \sqrt{n} \leq ||\alpha|| \leq \Phi^{r-2} \cdot \sqrt{n}, \]

- From which we conclude:
  \[ r \geq 2 + \left[ \log_{\frac{1}{2}(1+\sqrt{5})} \frac{n}{2} \right] \]
Continuation

\[ \alpha := (\alpha_1, \alpha_2, \ldots, \alpha_n)^T = A_{r-2} \cdot \ldots \cdot A_2 \cdot A_1 \cdot (1, 1, \ldots, 1) \]

- Let
  - \( c_1 \) be the number of cases with: \( \alpha_i \geq n \),
  - \( c_2 \) be the number of cases with: \( \alpha_i < n \) and \( \alpha_j \geq n \),
  - \( c_3 \) be the number of cases with: \( \alpha_i < n, \alpha_j < n \) and \( \alpha_i + \alpha_j \geq n \).

Then we have: \( c_1 \geq c_2 \) and \( c_1 + c_2 + c_3 \geq n/2 \).

Thus we also get: \( 2c_1 + c_3 \geq \frac{n}{2} \)

\[ \|\alpha\| = \sqrt{\sum_{i=1}^{n} \alpha_i^2} \geq \sqrt{c_1 n^2 + c_3 \cdot 2 \cdot \frac{n^2}{4}} \geq n \cdot \sqrt{\frac{1}{2} (2c_1 + c_3)} \geq \frac{n}{2} \sqrt{n}. \]

We already have:

\[ \|\alpha\| \leq \left( \prod_{i=1}^{r-2} \|A_i\| \right) \cdot \|(1, \ldots, 1)\| \leq \left( \frac{1}{2} (1 + \sqrt{5}) \right)^{r-2} \cdot \sqrt{n}. \]

And we get:

\[ \frac{n}{2} \cdot \sqrt{n} \leq \|\alpha\| \leq \Phi^{r-2} \cdot \sqrt{n}, \]

From which we conclude:

\[ r \geq 2 + \left\lceil \log_{\frac{1}{2}} \left( 1 + \sqrt{5} \right) \frac{n}{2} \right\rceil \]
Let

- $c_1$ be the number of cases with: $\alpha_i \geq n$,
- $c_2$ be the number of cases with: $\alpha_i < n$ and $\alpha_j \geq n$,
- $c_3$ be the number of cases with: $\alpha_i < n$, $\alpha_j < n$ and $\alpha_i + \alpha_j \geq n$.

Then we have: $c_1 \geq c_2$ and $c_1 + c_2 + c_3 \geq n/2$.

Thus we also get: $2c_1 + c_3 \geq \frac{n}{2}$

$||\alpha|| = \sqrt{\sum_{i=1}^{n} \alpha_i^2} \geq \sqrt{c_1 n^2 + c_3 \cdot 2 \cdot \frac{n^2}{4}} \geq n \cdot \sqrt{\frac{1}{2} (2c_1 + c_3)} \geq \frac{n}{2} \sqrt{n}$.

We already have:

$||\alpha|| \leq (\prod_{i=1}^{r-2} ||A_i||) \cdot ||(1, \ldots, 1)|| \leq (\frac{1}{2} (1 + \sqrt{5}))^{r-2} \cdot \sqrt{n}$.

And we get:

$\frac{n}{2} \cdot \sqrt{n} \leq ||\alpha|| \leq \Phi^{r-2} \cdot \sqrt{n}$.

From which we conclude:

$r \geq 2 + \left\lceil \log_{\frac{1}{2} (1 + \sqrt{5})} \frac{n}{2} \right\rceil$.
Quality of these Bounds

Lemma:

Let \( n = 2m \) and let:

- \( t_1 := 1 + k \), with \( k \) is the smallest number with \( m \leq F(k) \) and
- \( t_2 := 2 + \lceil \log_2 \frac{1}{2}(1+\sqrt{5}) \cdot m \rceil. \)

Then we have \( t_1 = t_2 \) for infinite many \( m \) and \( t_1 \leq t_2 + 1 \) for all \( m \).

Proof:

- Let \( \Phi = \frac{1}{2}(1 + \sqrt{5}) \).
- Then we have: \( \Phi^2 = \Phi + 1 \).
- Furthermore we have \( \Phi^{i-2} \leq F(i) \leq \Phi^{i-1} \) for all \( i \geq 2 \).
- Consider \( n \in \mathbb{N} \) with: \( n = 2 \cdot F(k) \) for some \( k \).
  - Then we have: \( t_1 = k + 1 \) and \( t_2 = 2 + \lceil \log_\Phi F(k) \rceil = 2 + k - 1 = k + 1 \).
  - From which we get: \( t_1 = t_2 \) for these \( n \).
Quality of these Bounds

Lemma:

Let $n = 2m$ and let:

- $t_1 := 1 + k$, with $k$ is the smallest number with $m \leq F(k)$ and
- $t_2 := 2 + \lceil \log_{\Phi} \left( \frac{1}{2}(1 + \sqrt{5}) \right) m \rceil$.

Then we have $t_1 = t_2$ for infinite many $m$ and $t_1 \leq t_2 + 1$ for all $m$.

Proof:

- Let $\Phi = \frac{1}{2}(1 + \sqrt{5})$.
- Then we have: $\Phi^2 = \Phi + 1$.
- Furthermore we have $\Phi^{i-2} \leq F(i) \leq \Phi^{i-1}$ for all $i \geq 2$.
- Consider $n \in \mathbb{N}$ with: $n = 2 \cdot F(k)$ for some $k$.
  - Then we have: $t_1 = k + 1$ and $t_2 = 2 + \lceil \log_{\Phi} F(k) \rceil = 2 + k - 1 = k + 1$.
  - From which we get: $t_1 = t_2$ for these $n$. 
Quality of these Bounds

Lemma:

Let \( n = 2m \) and let:

- \( t_1 := 1 + k \), with \( k \) is the smallest number with \( m \leq F(k) \) and
- \( t_2 := 2 + \lceil \log_\frac{1}{2} (1 + \sqrt{5}) \cdot m \rceil \).

Then we have \( t_1 = t_2 \) for infinite many \( m \) and \( t_1 \leq t_2 + 1 \) for all \( m \).

Proof:

- Let \( \Phi = \frac{1}{2} (1 + \sqrt{5}) \).
- Then we have: \( \Phi^2 = \Phi + 1 \).
- Furthermore we have \( \Phi^{i-2} \leq F(i) \leq \Phi^{i-1} \) for all \( i \geq 2 \).
- Consider \( n \in \mathbb{N} \) with: \( n = 2 \cdot F(k) \) for some \( k \).
  - Then we have: \( t_1 = k + 1 \) and
    \[
    t_2 = 2 + \lceil \log_\Phi F(k) \rceil = 2 + k - 1 = k + 1.
    \]
  - From which we get: \( t_1 = t_2 \) for these \( n \).
Quality of these Bounds

Lemma:

Let $n = 2m$ and let:

- $t_1 := 1 + k$, with $k$ is the smallest number with $m \leq F(k)$ and
- $t_2 := 2 + \lceil \log_{\frac{1}{2}}(1 + \sqrt{5}) m \rceil$.

Then we have $t_1 = t_2$ for infinite many $m$ and $t_1 \leq t_2 + 1$ for all $m$.

Proof:

- Let $\Phi = \frac{1}{2} (1 + \sqrt{5})$.
- Then we have: $\Phi^2 = \Phi + 1$.
- Furthermore we have $\Phi^{i-2} \leq F(i) \leq \Phi^{i-1}$ for all $i \geq 2$.
- Consider $n \in \mathbb{N}$ with: $n = 2 \cdot F(k)$ for some $k$.
  - Then we have: $t_1 = k + 1$ and $t_2 = 2 + \lceil \log_{\Phi} F(k) \rceil = 2 + k - 1 = k + 1$.
  - From which we get: $t_1 = t_2$ for these $n$. 
Quality of these Bounds

Lemma:

Let \( n = 2m \) and let:

- \( t_1 := 1 + k \), with \( k \) is the smallest number with \( m \leq F(k) \) and
- \( t_2 := 2 + \lceil \log_\frac{1}{2} (1 + \sqrt{5}) m \rceil \).

Then we have \( t_1 = t_2 \) for infinite many \( m \) and \( t_1 \leq t_2 + 1 \) for all \( m \).

Proof:

- Let \( \Phi = \frac{1}{2} (1 + \sqrt{5}) \).
- Then we have: \( \Phi^2 = \Phi + 1 \).
- Furthermore we have \( \Phi^{i-2} \leq F(i) \leq \Phi^{i-1} \) for all \( i \geq 2 \).
- Consider \( n \in \mathbb{N} \) with: \( n = 2 \cdot F(k) \) for some \( k \).
  - Then we have: \( t_1 = k + 1 \) and
    \[ t_2 = 2 + \lceil \log_\Phi F(k) \rceil = 2 + k - 1 = k + 1. \]
  - From which we get: \( t_1 = t_2 \) for these \( n \).
Quality of these Bounds

Lemma:

Let $n = 2m$ and let:

- $t_1 := 1 + k$, with $k$ is the smallest number with $m \leq F(k)$ and
- $t_2 := 2 + \lceil \log_{\frac{1}{2}(1+\sqrt{5})} m \rceil$.

Then we have $t_1 = t_2$ for infinite many $m$ and $t_1 \leq t_2 + 1$ for all $m$.

Proof:

- Let $\Phi = \frac{1}{2}(1 + \sqrt{5})$.
- Then we have: $\Phi^2 = \Phi + 1$.
- Furthermore we have $\Phi^{i-2} \leq F(i) \leq \Phi^{i-1}$ for all $i \geq 2$.
- Consider $n \in \mathbb{N}$ with: $n = 2 \cdot F(k)$ for some $k$.
  - Then we have: $t_1 = k + 1$ and $t_2 = 2 + \lceil \log_{\Phi} F(k) \rceil = 2 + k - 1 = k + 1$.
  - From which we get: $t_1 = t_2$ for these $n$. 
Quality of these Bounds

Lemma:
Let \( n = 2m \) and let:
- \( t_1 := 1 + k \), with \( k \) is the smallest number with \( m \leq F(k) \) and
- \( t_2 := 2 + \lceil \log_{\frac{1}{2}}(1+\sqrt{5}) m \rceil \).

Then we have \( t_1 = t_2 \) for infinite many \( m \) and \( t_1 \leq t_2 + 1 \) for all \( m \).

Proof:
- Let \( \Phi = \frac{1}{2} (1 + \sqrt{5}) \).
- Then we have: \( \Phi^2 = \Phi + 1 \).
- Furthermore we have \( \Phi^{i-2} \leq F(i) \leq \Phi^{i-1} \) for all \( i \geq 2 \).
- Consider \( n \in \mathbb{N} \) with: \( n = 2 \cdot F(k) \) for some \( k \).
  - Then we have: \( t_1 = k + 1 \) and
    \[ t_2 = 2 + \lceil \log_{\Phi} F(k) \rceil = 2 + k - 1 = k + 1. \]
  - From which we get: \( t_1 = t_2 \) for these \( n \).
Quality of these Bounds

Lemma:

Let $n = 2m$ and let:

- $t_1 := 1 + k$, with $k$ is the smallest number with $m \leq F(k)$ and
- $t_2 := 2 + \left\lceil \log_{\frac{1}{2}(1+\sqrt{5})} m \right\rceil$.

Then we have $t_1 = t_2$ for infinite many $m$ and $t_1 \leq t_2 + 1$ for all $m$.

Proof:

- Let $\Phi = \frac{1}{2}(1 + \sqrt{5})$.
- Then we have: $\Phi^2 = \Phi + 1$.
- Furthermore we have $\Phi^{i-2} \leq F(i) \leq \Phi^{i-1}$ for all $i \geq 2$.
- Consider $n \in \mathbb{N}$ with: $n = 2 \cdot F(k)$ for some $k$.
  - Then we have: $t_1 = k + 1$ and $t_2 = 2 + \left\lceil \log_{\Phi} F(k) \right\rceil = 2 + k - 1 = k + 1$.
  - From which we get: $t_1 = t_2$ for these $n$. 
Quality of these Bounds

**Lemma:**

Let \( n = 2m \) and let:

- \( t_1 := 1 + k \), with \( k \) is the smallest number with \( m \leq F(k) \) and
- \( t_2 := 2 + \left\lceil \log_{\frac{1}{2}} (1 + \sqrt{5}) \right\rceil m \).

Then we have \( t_1 = t_2 \) for infinite many \( m \) and \( t_1 \leq t_2 + 1 \) for all \( m \).

**Proof:**

- Let \( \Phi = \frac{1}{2} (1 + \sqrt{5}) \).
- Then we have: \( \Phi^2 = \Phi + 1 \).
- Furthermore we have \( \Phi^{i-2} \leq F(i) \leq \Phi^{i-1} \) for all \( i \geq 2 \).
- Consider \( n \in \mathbb{N} \) with: \( n = 2 \cdot F(k) \) for some \( k \).
  - Then we have: \( t_1 = k + 1 \) and \( t_2 = 2 + \left\lceil \log_{\Phi} F(k) \right\rceil = 2 + k - 1 = k + 1 \).
  - From which we get: \( t_1 = t_2 \) for these \( n \).
Quality of these Bounds

Lemma:

Let \( n = 2m \) and let:

- \( t_1 := 1 + k \), with \( k \) is the smallest number with \( m \leq F(k) \) and
- \( t_2 := 2 + \lceil \log_{\frac{1}{2}}(1 + \sqrt{5}) \ m \rceil \).

Then we have \( t_1 = t_2 \) for infinite many \( m \) and \( t_1 \leq t_2 + 1 \) for all \( m \).

Proof:

- Let \( \Phi = \frac{1}{2} (1 + \sqrt{5}) \).
- Then we have: \( \Phi^2 = \Phi + 1 \).
- Furthermore we have \( \Phi^{i-2} \leq F(i) \leq \Phi^{i-1} \) for all \( i \geq 2 \).
- Consider \( n \in \mathbb{N} \) with: \( n = 2 \cdot F(k) \) for some \( k \).
  - Then we have: \( t_1 = k + 1 \) and
    \( t_2 = 2 + \lceil \log_{\Phi} F(k) \rceil = 2 + k - 1 = k + 1 \).
  - From which we get: \( t_1 = t_2 \) for these \( n \).
Lemma:

Let \( n = 2m \) and let:

- \( t_1 := 1 + k \), with \( k \) is the smallest number with \( m \leq F(k) \) and
- \( t_2 := 2 + \lceil \log_\frac{1}{2}(1+\sqrt{5}) m \rceil \).

Then we have \( t_1 = t_2 \) for infinite many \( m \) and \( t_1 \leq t_2 + 1 \) for all \( m \).

Proof:

- Let \( \Phi = \frac{1}{2}(1 + \sqrt{5}) \).
- Then we have: \( \Phi^2 = \Phi + 1 \).
- Furthermore we have \( \Phi^{i-2} \leq F(i) \leq \Phi^{i-1} \) for all \( i \geq 2 \).
- Consider \( n \in \mathbb{N} \) with: \( n = 2 \cdot F(k) \) for some \( k \).
  - Then we have: \( t_1 = k + 1 \) and
    \[
    t_2 = 2 + \lceil \log_\Phi F(k) \rceil = 2 + k - 1 = k + 1.
    \]
  - From which we get: \( t_1 = t_2 \) for these \( n \).
Lemma:

Let \( n = 2m \) and let:

- \( t_1 := 1 + k \), with \( k \) is the smallest number with \( m \leq F(k) \) and
- \( t_2 := 2 + \lceil \log_{\frac{1}{2}}(1 + \sqrt{5}) m \rceil \).

Then we have \( t_1 = t_2 \) for infinite many \( m \) and \( t_1 \leq t_2 + 1 \) for all \( m \).

Proof:

- Setze \( \Phi = \frac{1}{2}(1 + \sqrt{5}) \).
- Then we have \( \Phi^{i-2} \leq F(i) \leq \Phi^{i-1} \) for all \( i \geq 2 \).
- Let \( n = 2 \cdot m \) arbitrary.
  - Let \( i \) be defined by: \( \Phi^{i-1} < m \leq \Phi^i \), then we have: \( t_2 = 2 + i \).
  - Let \( k \) be the smallest number with \( F(k) \geq m \).
  - Note: \( \Phi^{k-2} \leq F(k) \leq \Phi^{k-1} \).
  - Then we have: \( i = k - 1 \) oder \( i = k - 2 \).
  - From which we conclude: \( t_1 = k + 1 \leq i + 3 \).
Quality of these Bounds (Part 2)

Lemma:

Let \( n = 2m \) and let:

- \( t_1 := 1 + k \), with \( k \) is the smallest number with \( m \leq F(k) \) and
- \( t_2 := 2 + \lceil \log_{\frac{1}{2}}(1+\sqrt{5}) m \rceil \).

Then we have \( t_1 = t_2 \) for infinite many \( m \) and \( t_1 \leq t_2 + 1 \) for all \( m \).

Proof:

- Setze \( \Phi = \frac{1}{2}(1 + \sqrt{5}) \).
- Then we have \( \Phi^{i-2} \leq F(i) \leq \Phi^{i-1} \) for all \( i \geq 2 \).
- Let \( n = 2 \cdot m \) arbitrary.
  - Let \( i \) be defined by: \( \Phi^{i-1} < m \leq \Phi^i \), then we have: \( t_2 = 2 + i \).
  - Let \( k \) be the smallest number with \( F(k) \geq m \).
  - Note: \( \Phi^{k-2} \leq F(k) \leq \Phi^{k-1} \).
  - Then we have: \( i = k - 1 \) oder \( i = k - 2 \).
  - From which we conclude: \( t_1 = k + 1 \leq i + 3 \).
Quality of these Bounds (Part 2)

Lemma:

Let \( n = 2m \) and let:

- \( t_1 := 1 + k \), with \( k \) is the smallest number with \( m \leq F(k) \) and
- \( t_2 := 2 + \lceil \log_{\frac{1}{2}}(1+\sqrt{5}) \, m \rceil \).

Then we have \( t_1 = t_2 \) for infinite many \( m \) and \( t_1 \leq t_2 + 1 \) for all \( m \).

Proof:

- Setze \( \Phi = \frac{1}{2}(1 + \sqrt{5}) \).
- Then we have \( \Phi^{i-2} \leq F(i) \leq \Phi^{i-1} \) for all \( i \geq 2 \).
- Let \( n = 2 \cdot m \) arbitrary.
  - Let \( i \) be defined by: \( \Phi^{i-1} < m \leq \Phi^i \), then we have: \( t_2 = 2 + i \).
  - Let \( k \) be the smallest number with \( F(k) \geq m \).
  - Note: \( \Phi^{k-2} \leq F(k) \leq \Phi^{k-1} \).
  - Then we have: \( i = k - 1 \) oder \( i = k - 2 \).
  - From which we conclude: \( t_1 = k + 1 \leq i + 3 \).
Quality of these Bounds (Part 2)

Lemma:

Let $n = 2m$ and let:

- $t_1 := 1 + k$, with $k$ is the smallest number with $m \leq F(k)$ and
- $t_2 := 2 + \lceil \log_\frac{1}{2} (1+\sqrt{5}) \cdot m \rceil$.

Then we have $t_1 = t_2$ for infinite many $m$ and $t_1 \leq t_2 + 1$ for all $m$.

Proof:

- Setze $\Phi = \frac{1}{2} (1 + \sqrt{5})$.
- Then we have $\Phi^{i-2} \leq F(i) \leq \Phi^{i-1}$ for all $i \geq 2$.
- Let $n = 2 \cdot m$ arbitrary.
  - Let $i$ be defined by: $\Phi^{i-1} < m \leq \Phi^i$, then we have: $t_2 = 2 + i$.
  - Let $k$ be the smallest number with $F(k) \geq m$.
  - Note: $\Phi^{k-2} \leq F(k) \leq \Phi^{k-1}$.
  - Then we have: $i = k - 1$ oder $i = k - 2$.
  - From which we conclude: $t_1 = k + 1 \leq i + 3$. 
Quality of these Bounds (Part 2)

**Lemma:**

Let $n = 2m$ and let:
- $t_1 := 1 + k$, with $k$ is the smallest number with $m \leq F(k)$ and
- $t_2 := 2 + \lceil \log_{\frac{1}{2}}(1+\sqrt{5})m \rceil$.

Then we have $t_1 = t_2$ for infinite many $m$ and $t_1 \leq t_2 + 1$ for all $m$.

**Proof:**

- Setze $\Phi = \frac{1}{2}(1 + \sqrt{5})$.
- Then we have $\Phi^{i-2} \leq F(i) \leq \Phi^{i-1}$ for all $i \geq 2$.
- Let $n = 2 \cdot m$ arbitrary.
  - Let $i$ be defined by: $\Phi^{i-1} < m \leq \Phi^i$, then we have: $t_2 = 2 + i$.
  - Let $k$ be the smallest number with $F(k) \geq m$.
  - Note: $\Phi^{k-2} \leq F(k) \leq \Phi^{k-1}$.
  - Then we have: $i = k - 1$ oder $i = k - 2$.
  - From which we conclude: $t_1 = k + 1 \leq i + 3$. 


Quality of these Bounds (Part 2)

Lemma:

Let \( n = 2m \) and let:

- \( t_1 := 1 + k \), with \( k \) is the smallest number with \( m \leq F(k) \) and
- \( t_2 := 2 + \lceil \log_{\frac{1}{2}}(1+\sqrt{5}) \cdot m \rceil \).

Then we have \( t_1 = t_2 \) for infinite many \( m \) and \( t_1 \leq t_2 + 1 \) for all \( m \).

Proof:

- Setze \( \Phi = \frac{1}{2}(1 + \sqrt{5}) \).
- Then we have \( \Phi^{i-2} \leq F(i) \leq \Phi^{i-1} \) for all \( i \geq 2 \).
- Let \( n = 2 \cdot m \) arbitrary.
  - Let \( i \) be defined by: \( \Phi^{i-1} < m \leq \Phi^i \), then we have: \( t_2 = 2 + i \).
  - Let \( k \) be the smallest number with \( F(k) \geq m \).
  - Note: \( \Phi^{k-2} \leq F(k) \leq \Phi^{k-1} \).
  - Then we have: \( i = k - 1 \) oder \( i = k - 2 \).
  - From which we conclude: \( t_1 = k + 1 \leq i + 3 \).
## Summary (Telefon-Mode)

| Graph | $|V|$ | diam | Lower Bound | Upper Bound |
|-------|------|------|-------------|-------------|
| $K_n$ | $n$ | 1 | $\lceil \log_2 n \rceil + \text{odd}(n)$ | $\lceil \log_2 n \rceil + \text{odd}(n)$ |
| $H_k$ | $2^k$ | $k$ | $n - \text{even}(n)$ | $n - \text{even}(n)$ |
| $P_n$ | $n$ | $n - 1$ | $\lceil \frac{n}{2} \rceil + \text{odd}(n)$ | $\lceil \frac{n}{2} \rceil + \text{odd}(n)$ |
| $C_n$ | $n$ | $\lfloor \frac{n}{2} \rfloor - 2$ | $\lfloor \frac{5k}{2} \rfloor - 2$ | $\lfloor \frac{5k}{2} \rfloor - 2$, $k$ even |
| $\text{CCC}_k$ | $k \cdot 2^k$ | $\lfloor \frac{5k}{2} \rfloor - 2$ | $\lfloor \frac{5k}{2} \rfloor - 1$, $k$ odd | $2k - 1$ |
| $SE_k$ | $2^k$ | $2k - 1$ | $2k - 1$ | $2k + 5$ |
| $BF_k$ | $k \cdot 2^k$ | $\lfloor \frac{3k}{2} \rfloor$ | $1.9770k$ | $2k + 5$ |
| $DB_k$ | $2^k$ | $k$ | $1.5965k$ | $2k + 5$ |
# Summary (Telegraph-Mode)

| Graph | | \(|V|\) | diam | Lower Bound | Upper Bound |
|-------|----------|--------|-----------|-------------|-------------|
| \(K_n\) | | \(n\) | 1 | \(1.44 \log_2 n\) | 1.44log\(_2\)n |
| \(H_k\) | | \(2^k\) | \(k\) | 1.44\(k\) | 1.88\(k\) |
| \(P_n\) | | \(n\) | \(n - 1\) | \(n + \text{odd}(n)\) | \(n + \text{odd}(n)\) |
| \(C_n\) | \(n\) even | \(\lfloor \frac{n}{2} \rfloor\) | \(\frac{n}{2} + \lceil \sqrt{2n} \rceil - 1\) | \(\frac{n}{2} + \lceil \sqrt{2n} \rceil - 1\) |
| | \(n\) odd | \(\lceil \frac{n}{2} \rceil\) | \(\lceil \frac{n}{2} \rceil + \lceil \sqrt{2n} - \frac{1}{2} \rceil - 1\) | \(\lceil \frac{n}{2} \rceil + \lceil 2\sqrt{\lceil \frac{n}{2} \rceil} \rceil - 1\) |
| \(CCC_k\) | \(k \cdot 2^k\) | \(\lfloor \frac{5k}{2} \rfloor - 2\) | \(\lfloor \frac{5k}{2} \rfloor - 2\) | \(\lceil \frac{7k}{2} \rceil + \lceil 2\sqrt{\lceil \frac{k}{2} \rceil} \rceil - 2\) |
| \(SE_k\) | | \(2^k\) | \(2k - 1\) | \(2k - 1\) | \(3k + 3\) |
| \(BF_k\) | | \(k \cdot 2^k\) | \(\lfloor \frac{3k}{2} \rfloor\) | 1.9770\(k\) | \(\lceil \frac{5k}{2} \rceil + \lceil 2\sqrt{\lceil \frac{k}{2} \rceil} \rceil - 1\) |
| \(DB_k\) | | \(2^k\) | \(k\) | 1.5965\(k\) | 3\(k + 3\) |
**Literature**

Legend

- Not of relevance
- Implicitly used basics
- Idea of proof or algorithm
- Structure of proof or algorithm
- Full knowledge