Algorithmic Graph Theory (SS2016)
Chapter 8
Gossiping

Walter Unger

Lehrstuhl für Informatik 1

14:00, December 21, 2018
<table>
<thead>
<tr>
<th>Contents</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>1</td>
</tr>
<tr>
<td>- Recall and Motivation</td>
<td></td>
</tr>
<tr>
<td>- First Results</td>
<td></td>
</tr>
<tr>
<td>Simple Graphs</td>
<td>2</td>
</tr>
<tr>
<td>- Lines</td>
<td></td>
</tr>
<tr>
<td>- Trees</td>
<td></td>
</tr>
<tr>
<td>- Graphs with Bridges</td>
<td></td>
</tr>
<tr>
<td>Networks</td>
<td>3</td>
</tr>
<tr>
<td>- Cycles</td>
<td></td>
</tr>
<tr>
<td>- HQ</td>
<td></td>
</tr>
<tr>
<td>- Hypercube</td>
<td></td>
</tr>
<tr>
<td>- CCC and BF</td>
<td></td>
</tr>
<tr>
<td>Complexity</td>
<td>4</td>
</tr>
<tr>
<td>- Results</td>
<td></td>
</tr>
<tr>
<td>Telephone-Mode</td>
<td>5</td>
</tr>
<tr>
<td>- Even Number of Nodes</td>
<td></td>
</tr>
<tr>
<td>- Odd Number of Nodes</td>
<td></td>
</tr>
<tr>
<td>Telegraph-Mode</td>
<td>6</td>
</tr>
<tr>
<td>- Upper Bound</td>
<td></td>
</tr>
<tr>
<td>- Lower Bound</td>
<td></td>
</tr>
<tr>
<td>Summary</td>
<td>7</td>
</tr>
<tr>
<td>- Telefon-Mode</td>
<td></td>
</tr>
<tr>
<td>- Telegraph-Mode</td>
<td></td>
</tr>
</tbody>
</table>
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.
- $r(G) = \min\{\text{comm}(A) \mid A \text{ is a one-way algorithm for the gossip-problem on } G\}$
- $r_2(G) = \min\{\text{comm}(A) \mid A \text{ is a two-way algorithm for the gossip-problem on } G\}$
**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.
- \( r(G) = \min\{\text{comm}(A) \mid A \text{ is a one-way algorithm for the gossip-problem on } G\} \)
- \( r_2(G) = \min\{\text{comm}(A) \mid A \text{ is a two-way algorithm for the gossip-problem on } G\} \)
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 $comm(A)$ we denote the complexity (number of rounds) of a communication-algorithm.

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

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

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

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:

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

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

- Show: $r_2(G) \geq \left\lfloor \log_2 n \right\rfloor + 1$.
- 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 \( G = (V, E) \) be 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) \( E_1, E_2, \ldots, E_k \).
  - 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.
    - \( i - 1 \rightarrow i \): 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.
**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:

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

**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:

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

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.
  - $i - 1 \rightarrow i$: 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:

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:

- $\min b(G) \leq n - 1$ 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:

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

Proof: Follows from the following known statements:

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

- $\minb(G) \leq n - 1$ 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:

- $\minb(G) \leq n - 1$ 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 $|V| = n$.
- $r(G) \leq 2 \cdot \minb(G)$
- $r_2(G) \leq 2 \cdot \minb(G) - 1$
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:

- $\minb(G) \leq n - 1$ 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.
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:
1. \( r(T_k(1)) = 2k \)
2. \( r_2(T_k(1)) = 2k - 1 \)

Proof:

1. Show: \( r(T_k(1)) \geq 2k \).
2. \( r(T_k(1)) \) has one root and \( k \) leaves.
3. The maximal matching is 1.
4. In each round is only one leaf active.
5. Each leaf has to send at least once.
6. Each leaf has to receive at least once.
7. Thus in total \( 2k \) rounds necessary.
8. \( r_2(T_k(1)) \geq 2k - 1 \), is a simple exercise.
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.
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: \( r_2(L(n)) \geq b(L(n)) \geq diam(L(n)) = n - 1 \)
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: \( r_2(L(n)) \geq b(L(n)) \geq diam(L(n)) = n - 1 \)
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: $r_2(L(n)) \geq b(L(n)) \geq diam(L(n)) = n - 1$
**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: $r_2(L(n)) \geq b(L(n)) \geq \text{diam}(L(n)) = n - 1$
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: \( r_2(L(n)) \geq b(L(n)) \geq diam(L(n)) = n - 1 \)
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: \( r_2(L(n)) \geq b(L(n)) \geq diam(L(n)) = n - 1 \)
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:

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

| $r_2(L(n))$ | $n - 1$ | $(n \equiv 0 \pmod{2})$
|-------------|---------|-----------------
| $r_2(L(n))$ | $n$     | $(n \equiv 1 \pmod{2})$
| $r(L(n))$   | $n$     | $(n \equiv 0 \pmod{2})$
| $r(L(n))$   | $n + 1$ | $(n \equiv 1 \pmod{2})$
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. $\{0, 1\}, \{n - 1, n - 2\}$

\[
r_2(L(n)) = n - 1 \quad (n \equiv 0 \pmod{2})
\]
\[
r_2(L(n)) = n \quad (n \equiv 1 \pmod{2})
\]
\[
r(L(n)) = n \quad (n \equiv 0 \pmod{2})
\]
\[
r(L(n)) = n + 1 \quad (n \equiv 1 \pmod{2})
\]

\[
G_0 \quad G_1 \quad G_2 \quad G_3 \quad G_4 \quad G_5 \quad G_6 \quad G_7 \quad G_8 \quad G_9
\]
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:

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:
  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:
  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. $\{0, 1\}, \{n - 1, n - 2\}$
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:

\[
\begin{align*}
\text{(1)} & \quad \{0, 1\}, \\
\text{(2)} & \quad \{1, 2\}, \{n - 1, n - 2\}, \\
\text{(3)} & \quad \{2, 3\}, \{n - 2, n - 3\}, \\
\text{(4)} & \quad \ldots \\
\text{(5)} & \quad \{\lceil n/2 \rceil, \lfloor n/2 \rfloor\} \\
\text{(6)} & \quad \ldots \\
\text{(7)} & \quad \{2, 3\}, \{n - 2, n - 3\}, \\
\text{(8)} & \quad \{1, 2\}, \{n - 1, n - 2\}, \\
\text{(9)} & \quad \{0, 1\}
\end{align*}
\]
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. $\{0, 1\}$

$$r_2(L(n)) = n - 1 \quad (n \equiv 0 \pmod{2})$$
$$r_2(L(n)) = n \quad (n \equiv 1 \pmod{2})$$
$$r(L(n)) = n \quad (n \equiv 0 \pmod{2})$$
$$r(L(n)) = n + 1 \quad (n \equiv 1 \pmod{2})$$
Gossip on Lines (Proof 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. $\{[n/2], [n/2]\}$
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. $\{0, 1\}$

\[
\begin{align*}
r_2(L(n)) & = n - 1 \quad (n \equiv 0 \pmod{2}) \\
r_2(L(n)) & = n \quad (n \equiv 1 \pmod{2}) \\
r(L(n)) & = n \quad (n \equiv 0 \pmod{2}) \\
r(L(n)) & = n + 1 \quad (n \equiv 1 \pmod{2})
\end{align*}
\]
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. $\{0, 1\}$

\[ r_2(L(n)) = \begin{cases} n - 1 & (n \equiv 0 \pmod{2}) \\ n & (n \equiv 1 \pmod{2}) \end{cases} \]
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. $\{\lfloor n/2 \rfloor, \lceil n/2 \rceil\}$
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. $\{\lfloor n/2 \rfloor, \lceil n/2 \rceil\}$
  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. $\{0, 1\}$
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: $r_2(L(n)) \geq n$ 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.

$\begin{align*}
r_2(L(n)) &= n - 1 & (n \equiv 0 \mod 2) \\
r_2(L(n)) &= n & (n \equiv 1 \mod 2) \\
r(L(n)) &= n & (n \equiv 0 \mod 2) \\
r(L(n)) &= n + 1 & (n \equiv 1 \mod 2)
\end{align*}$
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.
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 messages has to be delayed.
- This provides the lower bound.
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. $\{(2, 3), (n - 3, n - 4)\}$,
...,
5. $\{(n/2 - 1, n/2)\}$,
6. $\{(n/2, n/2 - 1)\}$,
...,
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. ...
  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. ... 
  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. \( \{(1, 0), (n - 2, n - 1)\} \)
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. $\cdots$
5. $\{(n/2 - 1, n/2)\}$
6. $\{(n/2, n/2 - 1)\}$
7. $\cdots$
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. ...
  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. \( \{(1, 0), (n - 2, n - 1)\} \)

\[
\begin{align*}
r_2(L(n)) &= n - 1 & (n \equiv 0 \text{ (mod 2)}) \\
r_2(L(n)) &= n & (n \equiv 1 \text{ (mod 2)}) \\
r(L(n)) &= n & (n \equiv 0 \text{ (mod 2)}) \\
r(L(n)) &= n + 1 & (n \equiv 1 \text{ (mod 2)})
\end{align*}
\]
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 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.

$$r_2(L(n)) = n - 1 \quad (n \equiv 0 \pmod{2})$$
$$r_2(L(n)) = n \quad (n \equiv 1 \pmod{2})$$
$$r(L(n)) = n \quad (n \equiv 0 \pmod{2})$$
$$r(L(n)) = n + 1 \quad (n \equiv 1 \pmod{2})$$
Gossip on Lines (Proof 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*}
\forall v_0 & \quad v_1 \quad v_2 \quad v_3 \quad v_4 \quad v_5 \quad v_6 \quad v_7 \quad v_8 \quad v_9 \\

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 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.

\[
\begin{align*}
\text{Show: } r(L(n)) &\geq n \text{ for } n \text{ even.} \\
\text{The proof is similar to the above one:} \\
\text{Consider the flow of messages from the left to the right node.} \\
\text{These could not be forwarded without delay.} \\
\text{Because we would get a time-conflict in the center.} \\
\text{Thus at least one message has to be delayed.} \\
\text{This provides the lower bound.}
\end{align*}
\]
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.
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.
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. ...
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. $\{(\lfloor n/2 \rfloor, \lceil n/2 \rceil)\}$
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. $\{(1, 0)\}$

Diagram:

```
\begin{tikzpicture}
    \node (v0) at (0,0) {$v_0$};
    \node (v1) at (1,0) {$v_1$};
    \node (v2) at (2,0) {$v_2$};
    \node (v3) at (3,0) {$v_3$};
    \node (v4) at (4,0) {$v_4$};
    \node (v5) at (5,0) {$v_5$};
    \node (v6) at (6,0) {$v_6$};
    \node (v7) at (7,0) {$v_7$};
    \node (v8) at (8,0) {$v_8$};
    \draw (v0) -- (v1);
    \draw (v1) -- (v2);
    \draw (v2) -- (v3);
    \draw (v3) -- (v4);
    \draw (v4) -- (v5);
    \draw (v5) -- (v6);
    \draw (v6) -- (v7);
    \draw (v7) -- (v8);
\end{tikzpicture}
```
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. $\{ (1, 0) \}$

Diagram:

```
[Diagram of vertices labeled 0 to 8 with matchings indicated.]
```
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. $\{(1, 0)\}$

\[ r_2(L(n)) = n - 1 \quad (n \equiv 0 \mod 2) \]
\[ r_2(L(n)) = n \quad (n \equiv 1 \mod 2) \]
\[ r(L(n)) = n \quad (n \equiv 0 \mod 2) \]
\[ r(L(n)) = n + 1 \quad (n \equiv 1 \mod 2) \]
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. \( \ldots \)
  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 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. $\{(1, 0)\}$
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. \( \{\lfloor n/2 \rfloor, \lceil n/2 \rceil\}\)
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 \( n \text{ odd} \).
- 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 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.

\[
\begin{align*}
\quantity{r_2(L(n))} &= \quantity{n - 1} \quad (n \equiv 0 \pmod{2}) \\
\quantity{r_2(L(n))} &= \quantity{n} \quad (n \equiv 1 \pmod{2}) \\
\quantity{r(L(n))} &= \quantity{n} \quad (n \equiv 0 \pmod{2}) \\
\quantity{r(L(n))} &= \quantity{n + 1} \quad (n \equiv 1 \pmod{2})
\end{align*}
\]
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.

\[
\begin{align*}
 r_2(L(n)) &= n \quad (n \equiv 0 \pmod{2}) \\
 r(L(n)) &= n \quad (n \equiv 1 \pmod{2}) \\
 r(L(n)) &= n + 1 \quad (n \equiv 0 \pmod{2}) \\
 r(L(n)) &= n + 1 \quad (n \equiv 1 \pmod{2})
\end{align*}
\]
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 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.
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.

![Graph diagram]

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

- $r(T) = 2 \cdot \minb(T)$
- $r_2(T) = 2 \cdot \minb(T) - 1$

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

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:
- $r(T) = 2 \cdot \min b(T)$
- $r_2(T) = 2 \cdot \min b(T) - 1$

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:

1. $r(T) = 2 \cdot \min b(T)$
2. $r_2(T) = 2 \cdot \min b(T) - 1$

Idea of the proof:

1. We have already for any graph $G$: $r(G) \leq 2 \cdot \min b(G)$.
2. We have to show: $r(G) \geq 2 \cdot \min b(G)$.
3. Let $W = \bigcup_{w \in V} I(v)$ be the total information.
4. Let $A$ be any communication algorithm on $T$.
5. Let $t$ be the point in time, when some node knows $W$.
6. Let $v$ one node, which after $t$ steps know $W$.
7. 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 $W$. 
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 $W$. 
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 $W = \bigcup_{w \in V} I(v)$ 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 $W$. 
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: \( t \geq \min_b(T) = b(v, T) \).
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: \( t \geq \min_b(T) = b(v, T) \).
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: $t \geq \min b(T) = b(v, T)$. 

![](image.png)
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: \( t \geq \min b(T) = b(v, T) \).
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: \( t \geq \min b(T) = b(v, T) \).
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: $t \geq \min_b(T) = b(v, T)$.
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 $(u, y_1, y_2, \ldots, y_k, v)$ 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: $t \geq \min b(T) = b(v, T)$. 

```
      u   y1  y2  y3  yk   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 $r(T) \geq \minb(T) + b(v, T) \geq 2 \cdot \minb(T)$
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 $r(T) \geq \min b(T) + b(v, T) \geq 2 \cdot \min b(T)$
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 $r(T) \geq \min b(T) + b(v, T) \geq 2 \cdot \min b(T)$
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 $v_1, v_2, \ldots, v_k$ 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 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.
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.
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.
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.
Lemma:

For all $m \geq 1$ and $k \geq 2$ we have:

- $r(T_k(m)) = 2 \min b(T_k(m)) = 2 \cdot k \cdot m$.
- $r_2(T_k(m)) = 2 \min b(T_k(m)) - 1 = 2 \cdot k \cdot m - 1$. 
Lemma:

For all $m \geq 1$ and $k \geq 2$ we have:

- $r(T_k(m)) = 2 \min_b(T_k(m)) = 2 \cdot k \cdot m$.
- $r_2(T_k(m)) = 2 \min_b(T_k(m)) - 1 = 2 \cdot k \cdot m - 1$. 
Lemma:

For all \( m \geq 1 \) and \( k \geq 2 \) we have:

- \( r(T_k(m)) = 2 \min_b(T_k(m)) = 2 \cdot k \cdot m \).
- \( r_2(T_k(m)) = 2 \min_b(T_k(m)) - 1 = 2 \cdot k \cdot m - 1 \).
Lemma:

Let $G = (V, E)$ be a graph with bridge $e \in E$, which is separated by $e$ in components $G_1$ and $G_2$, then we have

$$r(G) \geq \text{minb}(G) + 1 + \min\{\text{minb}(G_1), \text{minb}(G_2)\}$$

Proof: Let $W = \bigcup_{v \in V} I(v)$ be the total information.

Let $t \geq \text{minb}(G)$ the time, when a node $w$ knows $W$.

- If $w \in G_1$ hold, then do no node from $G_2$ know $W$.
- Then there are still $1 + \text{minb}(G_2)$ steps to do.
- If $w \in G_2$ hold, then do no node from $G_1$ know $W$.
- Then there are still $1 + \text{minb}(G_1)$ steps to do.
- Thus we have: $r(G) \geq \text{minb}(G) + 1 + \min\{\text{minb}(G_1), \text{minb}(G_2)\}$. 
Lemma:

Let $G = (V, E)$ be a graph with bridge $e \in E$, which is separated by $e$ in components $G_1$ and $G_2$, then we have

$$r(G) \geq \min b(G) + 1 + \min \{\min b(G_1), \min b(G_2)\}$$

Proof: Let $W = \bigcup_{v \in V} I(v)$ be the total information.

Let $t \geq \min b(G)$ the time, when a node $w$ knows $W$.

- If $w \in G_1$ hold, then do no node from $G_2$ know $W$.
- Then there are still $1 + \min b(G_2)$ steps to do.

- If $w \in G_2$ hold, then do no node from $G_1$ know $W$.
- Then there are still $1 + \min b(G_1)$ steps to do.

Thus we have: $r(G) \geq \min b(G) + 1 + \min \{\min b(G_1), \min b(G_2)\}$.
Lemma:

Let $G = (V, E)$ be a graph with bridge $e \in E$, which is separated by $e$ in components $G_1$ and $G_2$, then we have
\[ r(G) \geq \min b(G) + 1 + \min \{ \min b(G_1), \min b(G_2) \} \]

Proof: Let $W = \bigcup_{v \in V} I(v)$ be the total information. Let $t \geq \min b(G)$ the time, when a node $w$ knows $W$.

- If $w \in G_1$ hold, then do no node from $G_2$ know $W$.
- Then there are still $1 + \min b(G_2)$ steps to do.
- If $w \in G_2$ hold, then do no node from $G_1$ know $W$.
- Then there are still $1 + \min b(G_1)$ steps to do.
- Thus we have: $r(G) \geq \min b(G) + 1 + \min \{ \min b(G_1), \min b(G_2) \}$. 

![Graph with Bridge](image.png)
Lemma:

Let $G = (V, E)$ be a graph with bridge $e \in E$, which is separated by $e$ in components $G_1$ and $G_2$, then we have

$$r(G) \geq \min b(G) + 1 + \min\{\min b(G_1), \min b(G_2)\}$$

Proof: Let $W = \bigcup_{v \in V} I(v)$ be the total information.

Let $t \geq \min b(G)$ the time, when a node $w$ knows $W$.

- If $w \in G_1$ hold, then do no node from $G_2$ know $W$.
- Then there are still $1 + \min b(G_2)$ steps to do.
- If $w \in G_2$ hold, then do no node from $G_1$ know $W$.
- Then there are still $1 + \min b(G_1)$ steps to do.
- Thus we have: $r(G) \geq \min b(G) + 1 + \min\{\min b(G_1), \min b(G_2)\}$. 

Lemma:

Let $G = (V, E)$ be a graph with bridge $e \in E$, which is separated by $e$ in components $G_1$ and $G_2$, then we have

$r(G) \geq \minb(G) + 1 + \min\{\minb(G_1), \minb(G_2)\}$

Proof: Let $W = \bigcup_{v \in V} I(v)$ be the total information.

Let $t \geq \minb(G)$ the time, when a node $w$ knows $W$.

- If $w \in G_1$ hold, then do no node from $G_2$ know $W$.
- Then there are still $1 + \minb(G_2)$ steps to do.
- If $w \in G_2$ hold, then do no node from $G_1$ know $W$.
- Then there are still $1 + \minb(G_1)$ steps to do.
- Thus we have: $r(G) \geq \minb(G) + 1 + \min\{\minb(G_1), \minb(G_2)\}$. 
Graphs with Bridges

Lemma:

Let $G = (V, E)$ be a graph with bridge $e \in E$, which is separated by $e$ in components $G_1$ and $G_2$, then we have

$$r(G) \geq \min b(G) + 1 + \min \{\min b(G_1), \min b(G_2)\}$$

Proof: Let $W = \bigcup_{v \in V} I(v)$ be the total information.

Let $t \geq \min b(G)$ the time, when a node $w$ knows $W$.

- If $w \in G_1$ hold, then do no node from $G_2$ know $W$.
- Then there are still $1 + \min b(G_2)$ steps to do.
- If $w \in G_2$ hold, then do no node from $G_1$ know $W$.
- Then there are still $1 + \min b(G_1)$ steps to do.
- Thus we have: $r(G) \geq \min b(G) + 1 + \min \{\min b(G_1), \min b(G_2)\}$.  

---

![Diagram of a graph with two components $G_1$ and $G_2$ connected by a bridge $e$]
Lemma:

Let $G = (V, E)$ be a graph with bridge $e \in E$, which is separated by $e$ in components $G_1$ and $G_2$, then we have
$$r(G) \geq \min b(G) + 1 + \min \{\min b(G_1), \min b(G_2)\}$$

Proof: Let $W = \bigcup_{v \in V} I(v)$ be the total information.
Let $t \geq \min b(G)$ the time, when a node $w$ knows $W$.

- If $w \in G_1$ hold, then do no node from $G_2$ know $W$.
- Then there are still $1 + \min b(G_2)$ steps to do.
- If $w \in G_2$ hold, then do no node from $G_1$ know $W$.
- Then there are still $1 + \min b(G_1)$ steps to do.
- Thus we have: $r(G) \geq \min b(G) + 1 + \min \{\min b(G_1), \min b(G_2)\}$.
Lemma:

Let $G = (V, E)$ be a graph with bridge $e \in E$, which is separated by $e$ in components $G_1$ and $G_2$, then we have:

$$r_2(G) \geq \min b(G) + \min \{\min b(G_1), \min b(G_2)\}$$

Proof: Let $t \geq \min b(G)$ be the time, when node $w$ knows $W$ the first time. As before we may prove:

- Let $i \in \{1, 2\}$. If $w \in G_i$ and $v_{3-i}$ does not know $W$, then no node from $G_{3-i}$ knows $W$. There are still $1 + \min b(G_{3-i})$ steps to do.
- If $v_1$ and $v_2$ know $W$ at time $t$, then no other node knows $W$. There are still $\min \{\min b(G_1), \min b(G_2)\}$ Steps to do.
- Thus we have: $r_2(G) \geq \min b(G) + \min \{\min b(G_1), \min b(G_2)\}$. 
Lemma:

Let $G = (V, E)$ be a graph with bridge $e \in E$, which is separated by $e$ in components $G_1$ and $G_2$, then we have:

$$r_2(G) \geq \min b(G) + \min \{ \min b(G_1), \min b(G_2) \}$$

Proof: Let $t \geq \min b(G)$ be the time, when node $w$ knows $W$ the first time. As before we may prove:

- Let $i \in \{1, 2\}$. If $w \in G_i$ and $v_{3-i}$ does not know $W$, then no node from $G_{3-i}$ knows $W$. There are still $1 + \min b(G_{3-i})$ steps to do.
- If $v_1$ and $v_2$ know $W$ at time $t$, then no other node knows $W$. There are still $\min \{ \min b(G_1), \min b(G_2) \}$ Steps to do.
- Thus we have: $r_2(G) \geq \min b(G) + \min \{ \min b(G_1), \min b(G_2) \}$. 

- \begin{align*}
  G_1 & \quad v_1 \quad \text{v}_2 \quad G_2
\end{align*}
Lemma:

Let $G = (V, E)$ be a graph with bridge $e \in E$, which is separated by $e$ in components $G_1$ and $G_2$, then we have:

$r_2(G) \geq \min b(G) + \min \{\min b(G_1), \min b(G_2)\}$

Proof: Let $t \geq \min b(G)$ be the time, when node $w$ knows $W$ the first time. As before we may prove:

- Let $i \in \{1, 2\}$. If $w \in G_i$ and $v_{3-i}$ does not know $W$, then no node from $G_{3-i}$ knows $W$. There are still $1 + \min b(G_{3-i})$ steps to do.

- If $v_1$ and $v_2$ know $W$ at time $t$, then no other node knows $W$. There are still $\min \{\min b(G_1), \min b(G_2)\}$ Steps to do.

Thus we have: $r_2(G) \geq \min b(G) + \min \{\min b(G_1), \min b(G_2)\}$. 
Lemma:

Let $G = (V, E)$ be a graph with bridge $e \in E$, which is separated by $e$ in components $G_1$ and $G_2$, then we have:

$$r_2(G) \geq \min b(G) + \min \{ \min b(G_1), \min b(G_2) \}$$

Proof: Let $t \geq \min b(G)$ be the time, when node $w$ knows $W$ the first time. As before we may prove:

- Let $i \in \{1, 2\}$. If $w \in G_i$ and $v_{3-i}$ does not know $W$, then no node from $G_{3-i}$ knows $W$. There are still $1 + \min b(G_{3-i})$ steps to do.

- If $v_1$ and $v_2$ know $W$ at time $t$, then no other node knows $W$. There are still $\min \{ \min b(G_1), \min b(G_2) \}$ Steps to do.

- Thus we have: $r_2(G) \geq \min b(G) + \min \{ \min b(G_1), \min b(G_2) \}$. 
Lemma:

Let $G = (V, E)$ be a graph with bridge $e \in E$, which is separated by $e$ in components $G_1$ and $G_2$, then we have:

$$r_2(G) \geq \min b(G) + \min \{ \min b(G_1), \min b(G_2) \}$$

Proof: Let $t \geq \min b(G)$ be the time, when node $w$ knows $W$ the first time. As before we may prove:

- Let $i \in \{1, 2\}$. If $w \in G_i$ and $v_{3-i}$ does not know $W$, then no node from $G_{3-i}$ knows $W$. There are still $1 + \min b(G_{3-i})$ steps to do.
- If $v_1$ and $v_2$ know $W$ at time $t$, then no other node knows $W$. There are still $\min \{ \min b(G_1), \min b(G_2) \}$ Steps to do.
- Thus we have: $r_2(G) \geq \min b(G) + \min \{ \min b(G_1), \min b(G_2) \}$. 

![Graph with Bridges](image)
Theorem:

We have:
- \( r_2(C(k)) = k/2 \) for even \( k \).
- \( r_2(C(k)) = \lceil k/2 \rceil + 1 \) for odd \( k \).

Idea of the proof (\( k \) even): [\( k \) odd: an easy exercise]

- Let \( k \) be even.
- \( r_2(C(k)) \geq k/2 \) results by the diameter.
- \( r_2(C(k)) \leq k/2 \) is true by the following algorithm:
  - \{0, 1\}, \{2, 3\}, \{4, 5\}, \ldots, \{2i, 2i + 1\}, \ldots, \{n − 2, n − 1\}
  - \{1, 2\}, \{3, 4\}, \{5, 6\}, \ldots, \{2i − 1, 2i\}, \ldots, \{n − 1, 0\}
  - \{0, 1\}, \{2, 3\}, \{4, 5\}, \ldots, \{2i, 2i + 1\}, \ldots, \{n − 2, n − 1\}
  - \{1, 2\}, \{3, 4\}, \{5, 6\}, \ldots, \{2i − 1, 2i\}, \ldots, \{n − 1, 0\}
- \ldots

Note: After \( i \) rounds knows each node \( 2 \cdot i \) Informationen.
Gossip on Cycles

Theorem:

We have:

- \( r_2(C(k)) = k/2 \) for even \( k \).
- \( r_2(C(k)) = \lceil k/2 \rceil + 1 \) for odd \( k \).

Idea of the proof (\( k \) even): [\( k \) odd: an easy exercise]

- Let \( k \) be even.
- \( r_2(C(k)) \geq k/2 \) results by the diameter.
- \( r_2(C(k)) \leq k/2 \) is true by the following algorithm:
  - \( \{0, 1\}, \{2, 3\}, \{4, 5\}, \ldots, \{2i, 2i+1\}, \ldots, \{n-2, n-1\} \)
  - \( \{1, 2\}, \{3, 4\}, \{5, 6\}, \ldots, \{2i-1, 2i\}, \ldots, \{n-1, 0\} \)
  - \( \{0, 1\}, \{2, 3\}, \{4, 5\}, \ldots, \{2i, 2i+1\}, \ldots, \{n-2, n-1\} \)
  - \( \{1, 2\}, \{3, 4\}, \{5, 6\}, \ldots, \{2i-1, 2i\}, \ldots, \{n-1, 0\} \)
  - \( \ldots \)

- Note: After \( i \) rounds knows each node \( 2 \cdot i \) Informationen.
Gossip on Cycles

Theorem:

We have:
- \( r_2(C(k)) = \frac{k}{2} \) for even \( k \).
- \( r_2(C(k)) = \lceil \frac{k}{2} \rceil + 1 \) for odd \( k \).

Idea of the proof (\( k \) even): [\( k \) odd: an easy exercise]
- Let \( k \) be even.
- \( r_2(C(k)) \geq \frac{k}{2} \) results by the diameter.
- \( r_2(C(k)) \leq \frac{k}{2} \) is true by the following algorithm:
  - \( \{0, 1\}, \{2, 3\}, \{4, 5\}, \ldots, \{2i, 2i+1\}, \ldots, \{n-2, n-1\} \)
  - \( \{1, 2\}, \{3, 4\}, \{5, 6\}, \ldots, \{2i-1, 2i\}, \ldots, \{n-1, 0\} \)
  - \( \{0, 1\}, \{2, 3\}, \{4, 5\}, \ldots, \{2i, 2i+1\}, \ldots, \{n-2, n-1\} \)
  - \( \{1, 2\}, \{3, 4\}, \{5, 6\}, \ldots, \{2i-1, 2i\}, \ldots, \{n-1, 0\} \)
  - \( \ldots \)

- Note: After \( i \) rounds knows each node \( 2 \cdot i \) Informationen.
Gossip on Cycles

Theorem:

We have:
- \( r_2(C(k)) = k/2 \) for even \( k \).
- \( r_2(C(k)) = \lceil k/2 \rceil + 1 \) for odd \( k \).

Idea of the proof (\( k \) even): \( [k \text{ odd: an easy exercise}]

- Let \( k \) be even.
- \( r_2(C(k)) \geq k/2 \) results by the diameter.
- \( r_2(C(k)) \leq k/2 \) is true by the following algorithm:
  - \( \{0, 1\}, \{2, 3\}, \{4, 5\}, \ldots, \{2i, 2i+1\}, \ldots, \{n-2, n-1\} \)
  - \( \{1, 2\}, \{3, 4\}, \{5, 6\}, \ldots, \{2i-1, 2i\}, \ldots, \{n-1, 0\} \)
  - \( \{0, 1\}, \{2, 3\}, \{4, 5\}, \ldots, \{2i, 2i+1\}, \ldots, \{n-2, n-1\} \)
  - \( \{1, 2\}, \{3, 4\}, \{5, 6\}, \ldots, \{2i-1, 2i\}, \ldots, \{n-1, 0\} \)
  - \( \ldots \)

- Note: After \( i \) rounds knows each node \( 2 \cdot i \) Informationen.
**Gossip on Cycles**

**Theorem:**

We have:

- $r_2(C(k)) = k/2$ for even $k$.
- $r_2(C(k)) = \lceil k/2 \rceil + 1$ for odd $k$.

**Idea of the proof (k even): [k odd: an easy exercise]**

- Let $k$ be even.
- $r_2(C(k)) \geq k/2$ results by the diameter.
- $r_2(C(k)) \leq k/2$ is true by the following algorithm:
  1. $\{0, 1\}, \{2, 3\}, \{4, 5\}, \cdots, \{2i, 2i + 1\}, \cdots, \{n-2, n-1\}$
  2. $\{1, 2\}, \{3, 4\}, \{5, 6\}, \cdots, \{2i-1, 2i\}, \cdots, \{n-1, 0\}$
  3. $\{0, 1\}, \{2, 3\}, \{4, 5\}, \cdots, \{2i, 2i + 1\}, \cdots, \{n-2, n-1\}$
  4. $\{1, 2\}, \{3, 4\}, \{5, 6\}, \cdots, \{2i-1, 2i\}, \cdots, \{n-1, 0\}$
  5. $\cdots$

- Note: After $i$ rounds knows each node $2 \cdot i$ Informationen.
Gossip on Cycles

Theorem:

We have:
- \( r_2(C(k)) = \frac{k}{2} \) for even \( k \).
- \( r_2(C(k)) = \lceil \frac{k}{2} \rceil + 1 \) for odd \( k \).

Idea of the proof (\( k \) even): [\( k \) odd: an easy exercise]

- Let \( k \) be even.
- \( r_2(C(k)) \geq \frac{k}{2} \) results by the diameter.
- \( r_2(C(k)) \leq \frac{k}{2} \) is true by the following algorithm:
  1. \( \{0, 1\}, \{2, 3\}, \{4, 5\}, \ldots, \{2i, 2i + 1\}, \ldots, \{n - 2, n - 1\} \)
  2. \( \{1, 2\}, \{3, 4\}, \{5, 6\}, \ldots, \{2i - 1, 2i\}, \ldots, \{n - 1, 0\} \)
  3. \( \{0, 1\}, \{2, 3\}, \{4, 5\}, \ldots, \{2i, 2i + 1\}, \ldots, \{n - 2, n - 1\} \)
  4. \( \{1, 2\}, \{3, 4\}, \{5, 6\}, \ldots, \{2i - 1, 2i\}, \ldots, \{n - 1, 0\} \)
  5. \( \ldots \)

- Note: After \( i \) rounds knows each node \( 2 \cdot i \) Informationen.
Theorem:

We have:

- \( r_2(C(k)) = \frac{k}{2} \) for even \( k \).
- \( r_2(C(k)) = \lceil \frac{k}{2} \rceil + 1 \) for odd \( k \).

Idea of the proof (\( k \) even): [\( k \) odd: an easy exercise]

- Let \( k \) be even.
- \( r_2(C(k)) \geq \frac{k}{2} \) results by the diameter.
- \( r_2(C(k)) \leq \frac{k}{2} \) is true by the following algorithm:
  
  1. \{\{0, 1\}, \{2, 3\}, \{4, 5\}, \ldots, \{2i, 2i + 1\}, \ldots, \{n - 2, n - 1\}\}
  2. \{\{1, 2\}, \{3, 4\}, \{5, 6\}, \ldots, \{2i - 1, 2i\}, \ldots, \{n - 1, 0\}\}
  3. \{\{0, 1\}, \{2, 3\}, \{4, 5\}, \ldots, \{2i, 2i + 1\}, \ldots, \{n - 2, n - 1\}\}
  4. \{\{1, 2\}, \{3, 4\}, \{5, 6\}, \ldots, \{2i - 1, 2i\}, \ldots, \{n - 1, 0\}\}
  5. \ldots

- Note: After \( i \) rounds knows each node \( 2 \cdot i \) Informationen.
Gossip on Cycles

Theorem:

We have:
- \( r_2(C(k)) = \frac{k}{2} \) for even \( k \).
- \( r_2(C(k)) = \lceil \frac{k}{2} \rceil + 1 \) for odd \( k \).

Idea of the proof (\( k \) even): [\( k \) odd: an easy exercise]
- Let \( k \) be even.
- \( r_2(C(k)) \geq \frac{k}{2} \) results by the diameter.
- \( r_2(C(k)) \leq \frac{k}{2} \) is true by the following algorithm:
  1. \{0, 1\}, \{2, 3\}, \{4, 5\}, \ldots, \{2i, 2i + 1\}, \ldots, \{n - 2, n - 1\}
  2. \{1, 2\}, \{3, 4\}, \{5, 6\}, \ldots, \{2i - 1, 2i\}, \ldots, \{n - 1, 0\}
  3. \{0, 1\}, \{2, 3\}, \{4, 5\}, \ldots, \{2i, 2i + 1\}, \ldots, \{n - 2, n - 1\}
  4. \{1, 2\}, \{3, 4\}, \{5, 6\}, \ldots, \{2i - 1, 2i\}, \ldots, \{n - 1, 0\}
  5. \ldots

- Note: After \( i \) rounds knows each node \( 2 \cdot i \) Informationen.
Theorem:

We have:
- \( r_2(C(k)) = k/2 \) for even \( k \).
- \( r_2(C(k)) = \lceil k/2 \rceil + 1 \) for odd \( k \).

Idea of the proof (\( k \) even): [\( k \) odd: an easy exercise]
- Let \( k \) be even.
- \( r_2(C(k)) \geq k/2 \) results by the diameter.
- \( r_2(C(k)) \leq k/2 \) is true by the following algorithm:
  1. \( \{0, 1\}, \{2, 3\}, \{4, 5\}, \ldots, \{2i, 2i + 1\}, \ldots, \{n - 2, n - 1\} \)
  2. \( \{1, 2\}, \{3, 4\}, \{5, 6\}, \ldots, \{2i - 1, 2i\}, \ldots, \{n - 1, 0\} \)
  3. \( \{0, 1\}, \{2, 3\}, \{4, 5\}, \ldots, \{2i, 2i + 1\}, \ldots, \{n - 2, n - 1\} \)
  4. \( \{1, 2\}, \{3, 4\}, \{5, 6\}, \ldots, \{2i - 1, 2i\}, \ldots, \{n - 1, 0\} \)
  5. \ldots

Note: After \( i \) rounds knows each node \( 2 \cdot i \) Informationen.
Gossip on Cycles

Theorem:

We have:

- \( r_2(C(k)) = k/2 \) for even \( k \).
- \( r_2(C(k)) = \lceil k/2 \rceil + 1 \) for odd \( k \).

Idea of the proof (\( k \) even): [\( k \) odd: an easy exercise]

- Let \( k \) be even.
- \( r_2(C(k)) \geq k/2 \) results by the diameter.
- \( r_2(C(k)) \leq k/2 \) is true by the following algorithm:

\[
\begin{align*}
1 & \{0, 1, 2, 3, 4, 5, \ldots, 2i, 2i+1, \ldots, n-2, n-1\} \\
2 & \{1, 2, 3, 4, 5, 6, \ldots, 2i-1, 2i, \ldots, n-1, 0\} \\
3 & \{0, 1, 2, 3, 4, 5, \ldots, 2i, 2i+1, \ldots, n-2, n-1\} \\
4 & \{1, 2, 3, 4, 5, 6, \ldots, 2i-1, 2i, \ldots, n-1, 0\} \\
5 & \ldots
\end{align*}
\]

- Note: After \( i \) rounds knows each node \( 2 \cdot i \) Informationen.
Theorem:

We have:

- \( r_2(C(k)) = \frac{k}{2} \) for even \( k \).
- \( r_2(C(k)) = \lceil \frac{k}{2} \rceil + 1 \) for odd \( k \).

Idea of the proof (\( k \) even): [\( k \) odd: an easy exercise]

- Let \( k \) be even.
- \( r_2(C(k)) \geq \frac{k}{2} \) results by the diameter.
- \( r_2(C(k)) \leq \frac{k}{2} \) is true by the following algorithm:

  1. \( \{0, 1\}, \{2, 3\}, \{4, 5\}, \ldots, \{2i, 2i + 1\}, \ldots, \{n - 2, n - 1\} \)
  2. \( \{1, 2\}, \{3, 4\}, \{5, 6\}, \ldots, \{2i - 1, 2i\}, \ldots, \{n - 1, 0\} \)
  3. \( \{0, 1\}, \{2, 3\}, \{4, 5\}, \ldots, \{2i, 2i + 1\}, \ldots, \{n - 2, n - 1\} \)
  4. \( \{1, 2\}, \{3, 4\}, \{5, 6\}, \ldots, \{2i - 1, 2i\}, \ldots, \{n - 1, 0\} \)
  5. \ldots

- Note: After \( i \) rounds knows each node \( 2 \cdot i \) Informationen.
Theorem:

We have:
- \( r_2(C(k)) = \frac{k}{2} \) for even \( k \).
- \( r_2(C(k)) = \lceil \frac{k}{2} \rceil + 1 \) for odd \( k \).

Idea of the proof (\( k \) even): [\( k \) odd: an easy exercise]

- Let \( k \) be even.
- \( r_2(C(k)) \geq \frac{k}{2} \) results by the diameter.
- \( r_2(C(k)) \leq \frac{k}{2} \) is true by the following algorithm:
  1. \{\{0, 1\}, \{2, 3\}, \{4, 5\}, \ldots, \{2i, 2i + 1\}, \ldots, \{n - 2, n - 1\}\}
  2. \{\{1, 2\}, \{3, 4\}, \{5, 6\}, \ldots, \{2i - 1, 2i\}, \ldots, \{n - 1, 0\}\}
  3. \{\{0, 1\}, \{2, 3\}, \{4, 5\}, \ldots, \{2i, 2i + 1\}, \ldots, \{n - 2, n - 1\}\}
  4. \{\{1, 2\}, \{3, 4\}, \{5, 6\}, \ldots, \{2i - 1, 2i\}, \ldots, \{n - 1, 0\}\}
  5. \ldots

- Note: After \( i \) rounds knows each node \( 2 \cdot i \) Informationen.
Gossip on Cycles

Theorem:

We have:
- $r_2(C(k)) = k/2$ for even $k$.
- $r_2(C(k)) = \lceil k/2 \rceil + 1$ for odd $k$.

Idea of the proof ($k$ even): [$k$ odd: an easy exercise]
- Let $k$ be even.
- $r_2(C(k)) \geq k/2$ results by the diameter.
- $r_2(C(k)) \leq k/2$ is true by the following algorithm:
  1. $\{\{0, 1\}, \{2, 3\}, \{4, 5\}, \ldots, \{2i, 2i+1\}, \ldots, \{n-2, n-1\}\$
  2. $\{\{1, 2\}, \{3, 4\}, \{5, 6\}, \ldots, \{2i-1, 2i\}, \ldots, \{n-1, 0\}\$
  3. $\{\{0, 1\}, \{2, 3\}, \{4, 5\}, \ldots, \{2i, 2i+1\}, \ldots, \{n-2, n-1\}\$
  4. $\{\{1, 2\}, \{3, 4\}, \{5, 6\}, \ldots, \{2i-1, 2i\}, \ldots, \{n-1, 0\}\$
  5. $\ldots$

- Note: After $i$ rounds knows each node $2 \cdot i$ Informationen.
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 $\Theta\left(\frac{n}{2f(n)}\right)$ 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 $\Theta(\frac{n}{2\cdot f(n)})$ 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 \( \Theta(\frac{n}{2^f(n)}) \) 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.
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** $B_i$ ($i \in \{1, 2, 3, \ldots, k\}$ 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 $v_i \ [u_i]$ 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 $n/2 + \Theta(\sqrt{n})$ rounds 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 $z_i$ distributes 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 $B_i$ ($i \in \{1, 2, 3, \ldots, k\}$ 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 $n/2 + \Theta(\sqrt{n})$ rounds 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 \( n \).
- \( r(C(n)) \leq \left\lceil \frac{n}{2} \right\rceil + \left\lceil 2 \cdot \sqrt{\left\lceil \frac{n}{2} \right\rceil} \right\rceil - 1 \) for odd \( n \).
- \( r(C(n)) \geq \frac{n}{2} + \sqrt{2n} - 1 \) for even \( n \).
- \( r(C(n)) \geq \left\lceil \frac{n}{2} \right\rceil + \left\lceil \sqrt{2n} - \frac{1}{2} \right\rceil - 1 \) for odd \( n \).

Proof: See literature.
Gossip on Cycles (Idea)

**Theorem:**

We have:

- $r(C(n)) \leq \frac{n}{2} + \sqrt{2n} - 1$ for even $n$.
- $r(C(n)) \leq \lceil \frac{n}{2} \rceil + \lceil 2 \cdot \sqrt{\lceil \frac{n}{2} \rceil} \rceil - 1$ for odd $n$.
- $r(C(n)) \geq \frac{n}{2} + \sqrt{2n} - 1$ for even $n$.
- $r(C(n)) \geq \lceil \frac{n}{2} \rceil + \lceil \sqrt{2n} - \frac{1}{2} \rceil - 1$ for odd $n$.

**Proof:** See literature.
Gossip on Cycles (Idea)

**Theorem:**

We have:

- \( r(C(n)) \leq \frac{n}{2} + \sqrt{2n} - 1 \) for even \( n \).
- \( r(C(n)) \leq \lceil \frac{n}{2} \rceil + \lceil 2 \cdot \sqrt{\lceil \frac{n}{2} \rceil} \rceil - 1 \) for odd \( n \).
- \( r(C(n)) \geq \frac{n}{2} + \sqrt{2n} - 1 \) for even \( n \).
- \( r(C(n)) \geq \lceil \frac{n}{2} \rceil + \lceil \sqrt{2n} - 1/2 \rceil - 1 \) for odd \( n \).

**Proof:** See literature.
Theorem:

We have:
- $r(C(n)) \leq n/2 + \sqrt{2n} - 1$ for even $n$.
- $r(C(n)) \leq \lceil n/2 \rceil + \lceil 2 \cdot \sqrt{n/2} \rceil - 1$ for odd $n$.
- $r(C(n)) \geq n/2 + \sqrt{2n} - 1$ for even $n$.
- $r(C(n)) \geq \lceil n/2 \rceil + \lceil \sqrt{2n} - 1/2 \rceil - 1$ for odd $n$.

Proof: See literature.
Theorem:

We have:

- \( r(C(n)) \leq \frac{n}{2} + \sqrt{2n} - 1 \) for even \( n \).
- \( r(C(n)) \leq \lceil \frac{n}{2} \rceil + \lceil 2 \cdot \sqrt{\lceil \frac{n}{2} \rceil} \rceil - 1 \) for odd \( n \).
- \( r(C(n)) \geq \frac{n}{2} + \sqrt{2n} - 1 \) for even \( n \).
- \( r(C(n)) \geq \lceil \frac{n}{2} \rceil + \lceil \sqrt{2n - 1/2} \rceil - 1 \) 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:
  \[
  \text{for } i = 1 \text{ to } m \text{ do } \\
  \quad \text{for all } a_1, a_2, \ldots, a_{m-1} \in \{0, 1\} \text{ do in parallel } \\
  \quad \quad a_1 a_2 \cdots a_{i-1}0a_i a_{i+1} \cdots a_{m-1} \text{ sends to } \\
  \quad \quad a_1 a_2 \cdots a_{i-1}1a_i a_{i+1} \cdots a_{m-1}
  \]

Corollary:
For all \( m \in \mathbb{N} \) we have: \( r_2(K(2^m)) = m \)
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
      $a_1 a_2 \cdots a_{i-1} 0 a_i a_{i+1} \cdots a_{m-1}$ sends to
      $a_1 a_2 \cdots a_{i-1} 1 a_i a_{i+1} \cdots a_{m-1}$

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
  
  $a_1a_2\cdots a_{i-1}0a_i a_{i+1}\cdots a_{m-1}$ sends to
  
  $a_1a_2\cdots a_{i-1}1a_i a_{i+1}\cdots a_{m-1}$

**Corollary:**

For all $m \in \mathbb{N}$ we have: $r_2(K(2^m)) = m$
Gossip on 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
    $a_1 a_2 \cdots a_{i-1} 0 a_i a_{i+1} \cdots a_{m-1}$ sends to
    $a_1 a_2 \cdots a_{i-1} 1 a_i a_{i+1} \cdots a_{m-1}$

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: $r_2(K(2^m)) = m$
Consider one-way mode:

- Start with the first phase of the gossip-algorithm for cycles on all cycles.
- Then each $\Theta(\sqrt{n})$-th node on each cycle knows the total information of its cycles.
- In $\Theta(\sqrt{n})$ waves distribute this information down and between the cycles.
- After $\Theta(n)$ steps knows each $\Theta(\sqrt{n})$-th node of each cycle the total information.
- The final part is the second phase of the gossip-algorithm of cycles on all cycles.
- All nodes know now the total information.
Consider one-way mode:

- **Start with the first phase of the gossip-algorithm for cycles on all cycles.**
- Then each $\Theta(\sqrt{n})$-th node on each cycle knows the total information of its cycles.
- In $\Theta(\sqrt{n})$ waves distribute this information down and between the cycles.
- After $\Theta(n)$ steps knows each $\Theta(\sqrt{n})$-th node of each cycle the total information.
- The final part is the second phase of the gossip-algorithm of cycles on all cycles.
- All nodes know now the total information.
CCC and BF (Idea)

- Consider one-way mode:
  - Start with the first phase of the gossip-algorithm for cycles on all cycles.
  - Then each $\Theta(\sqrt{n})$-th node on each cycle knows the total information of its cycles.
  - In $\Theta(\sqrt{n})$ waves distribute this information down and between the cycles.
  - After $\Theta(n)$ steps knows each $\Theta(\sqrt{n})$-th node of each cycle the total information.
  - The final part is the second phase of the gossip-algorithm of cycles on all cycles.
  - All nodes know now the total information.
Consider one-way mode:

- Start with the first phase of the gossip-algorithm for cycles on all cycles.
- Then each $\Theta(\sqrt{n})$-th node on each cycle knows the total information of its cycles.
- In $\Theta(\sqrt{n})$ waves distribute this information down and between the cycles.
- After $\Theta(n)$ steps knows each $\Theta(\sqrt{n})$-th node of each cycle the total information.
- The final part is the second phase of the gossip-algorithm of cycles on all cycles.
- All nodes know now the total information.
Consider one-way mode:
- Start with the first phase of the gossip-algorithm for cycles on all cycles.
- Then each $\Theta(\sqrt{n})$-th node on each cycle knows the total information of its cycles.
- In $\Theta(\sqrt{n})$ waves distribute this information down and between the cycles.
- After $\Theta(n)$ steps knows each $\Theta(\sqrt{n})$-th node of each cycle the total information.
- The final part is the second phase of the gossip-algorithm of cycles on all cycles.
- All nodes know now the total information.
Consider one-way mode:

- Start with the first phase of the gossip-algorithm for cycles on all cycles.
- Then each $\Theta(\sqrt{n})$-th node on each cycle knows the total information of its cycles.
- In $\Theta(\sqrt{n})$ waves distribute this information down and between the cycles.
- After $\Theta(n)$ steps knows each $\Theta(\sqrt{n})$-th node of each cycle the total information.
- The final part is the second phase of the gossip-algorithm of cycles on all cycles.
- All nodes know now the total information.
Consider one-way mode:

- Start with the first phase of the gossip-algorithm for cycles on all cycles.
- Then each $\Theta(\sqrt{n})$-th node on each cycle knows the total information of its cycles.
- In $\Theta(\sqrt{n})$ waves distribute this information down and between the cycles.
- After $\Theta(n)$ steps knows each $\Theta(\sqrt{n})$-th node of each cycle the total information.
- The final part is the second phase of the gossip-algorithm of cycles on all cycles.
- All nodes know now the total information.
Consider two-way mode:
- Start with the gossip algorithm for cycles on all cycles.
- Each node of the cycle knows now the total information of its cycle.
- In $\Theta(n/2)$ waves distribute this information down and between the cycles.
- After $\Theta(n)$ steps knows each node the total information.
Consider two-way mode:

- Start with the gossip algorithm for cycles on all cycles.
- Each node of the cycle knows now the total information of its cycle.
- In $\Theta(n/2)$ waves distribute this information down and between the cycles.
- After $\Theta(n)$ steps knows each node the total information.
CCC and BF (Idea)

- Consider two-way mode:
  - Start with the gossip algorithm for cycles on all cycles.
  - Each node of the cycle knows now the total information of its cycle.
  - In $\Theta(n/2)$ waves distribute this information down and between the cycles.
  - After $\Theta(n)$ steps knows each node the total information.
 CCC and BF (Idea)

- Consider two-way mode:
  - Start with the gossip algorithm for cycles on all cycles.
  - Each node of the cycle knows now the total information of its cycle.
  - In $\Theta(n/2)$ waves distribute this information down and between the cycles.
  - After $\Theta(n)$ steps knows each node the total information.
Consider two-way mode:

- Start with the gossip algorithm for cycles on all cycles.
- Each node of the cycle knows now the total information of its cycle.
- In $\Theta(n/2)$ waves distribute this information down and between the cycles.
- After $\Theta(n)$ steps knows each node the total information.
Theorem:

Let \( k \geq 3 \), then we have:

- \( r(\text{CCC}(k)) \leq r(\text{C}(k)) + 3k - 1 \leq \lceil \frac{7k}{2} \rceil + \left\lfloor 2\sqrt{\left\lceil \frac{k}{2} \right\rceil} \right\rfloor - 2. \)
- \( r(\text{BF}(k)) \leq r(\text{C}(k)) + 2k \leq \lceil \frac{5k}{2} \rceil + \left\lfloor 2\sqrt{\left\lceil \frac{k}{2} \right\rceil} \right\rfloor - 1. \)
- \( r_2(\text{CCC}(k)) \leq \frac{k}{2} + 2k = 5 \cdot \lceil \frac{k}{2} \rceil \) for even \( k \).
- \( r_2(\text{CCC}(k)) \leq \left\lceil \frac{k}{2} \right\rceil + 2k + 2 = 5 \cdot \lceil \frac{k}{2} \rceil \) for odd \( k \).
- \( r_2(\text{BF}(k)) \leq \frac{k}{2} + 2k = 5 \cdot \lceil \frac{k}{2} \rceil \) for even \( k \).
- \( r_2(\text{BF}(k)) \leq \left\lceil \frac{k}{2} \right\rceil + 2k + 2 = 5 \cdot \lceil \frac{k}{2} \rceil \) for odd \( k \).
Theorem:

Let $k \geq 3$, then we have:

- $r(CCC(k)) \leq r(C(k)) + 3k - 1 \leq \left\lfloor \frac{7k}{2} \right\rfloor + \left\lceil 2\sqrt{\left\lfloor \frac{k}{2} \right\rfloor} \right\rceil - 2$.
- $r(BF(k)) \leq r(C(k)) + 2k \leq \left\lfloor \frac{5k}{2} \right\rfloor + \left\lceil 2\sqrt{\left\lfloor \frac{k}{2} \right\rfloor} \right\rceil - 1$.
- $r_2(CCC(k)) \leq \frac{k}{2} + 2k = 5 \cdot \left\lfloor \frac{k}{2} \right\rfloor$ for even $k$.
- $r_2(CCC(k)) \leq \left\lfloor \frac{k}{2} \right\rfloor + 2k + 2 = 5 \cdot \left\lfloor \frac{k}{2} \right\rfloor$ for odd $k$.
- $r_2(BF(k)) \leq \frac{k}{2} + 2k = 5 \cdot \left\lfloor \frac{k}{2} \right\rfloor$ for even $k$.
- $r_2(BF(k)) \leq \left\lfloor \frac{k}{2} \right\rfloor + 2k + 2 = 5 \cdot \left\lfloor \frac{k}{2} \right\rfloor$ for odd $k$. 
Theorem:

Let \( k \geq 3 \), then we have:

- \( r(\text{CCC}(k)) \leq r(C(k)) + 3k - 1 \leq \left\lceil \frac{7k}{2} \right\rceil + \left\lceil 2\sqrt{\left\lceil \frac{k}{2} \right\rceil} \right\rceil - 2. \)

- \( r(\text{BF}(k)) \leq r(C(k)) + 2k \leq \left\lceil \frac{5k}{2} \right\rceil + \left\lceil 2\sqrt{\left\lceil \frac{k}{2} \right\rceil} \right\rceil - 1. \)

- \( r_2(\text{CCC}(k)) \leq \frac{k}{2} + 2k = 5 \cdot \left\lceil \frac{k}{2} \right\rceil \) for even \( k \).

- \( r_2(\text{CCC}(k)) \leq \left\lceil \frac{k}{2} \right\rceil + 2k + 2 = 5 \cdot \left\lceil \frac{k}{2} \right\rceil \) for odd \( k \).

- \( r_2(\text{BF}(k)) \leq \frac{k}{2} + 2k = 5 \cdot \left\lceil \frac{k}{2} \right\rceil \) for even \( k \).

- \( r_2(\text{BF}(k)) \leq \left\lceil \frac{k}{2} \right\rceil + 2k + 2 = 5 \cdot \left\lceil \frac{k}{2} \right\rceil \) for odd \( k \).
Theorem:

Let \( k \geq 3 \), then we have:

- \( r(\text{CCC}(k)) \leq r(\text{C}(k)) + 3k - 1 \leq \left\lceil \frac{7k}{2} \right\rceil + \left\lceil 2\sqrt{\left\lceil \frac{k}{2} \right\rceil} \right\rceil - 2. \)

- \( r(\text{BF}(k)) \leq r(\text{C}(k)) + 2k \leq \left\lceil \frac{5k}{2} \right\rceil + \left\lceil 2\sqrt{\left\lceil \frac{k}{2} \right\rceil} \right\rceil - 1. \)

- \( r_2(\text{CCC}(k)) \leq \frac{k}{2} + 2k = 5 \cdot \left\lceil \frac{k}{2} \right\rceil \) for even \( k \).
- \( r_2(\text{CCC}(k)) \leq \left\lceil \frac{k}{2} \right\rceil + 2k + 2 = 5 \cdot \left\lceil \frac{k}{2} \right\rceil \) for odd \( k \).
- \( r_2(\text{BF}(k)) \leq \frac{k}{2} + 2k = 5 \cdot \left\lceil \frac{k}{2} \right\rceil \) for even \( k \).
- \( r_2(\text{BF}(k)) \leq \left\lceil \frac{k}{2} \right\rceil + 2k + 2 = 5 \cdot \left\lceil \frac{k}{2} \right\rceil \) for odd \( k \).
Theorem:

Let $k \geq 3$, then we have:

- $r(\text{CCC}(k)) \leq r(\text{C}(k)) + 3k - 1 \leq \lceil \frac{7k}{2} \rceil + \left\lfloor 2\sqrt{\frac{k}{2}} \right\rfloor - 2$.

- $r(\text{BF}(k)) \leq r(\text{C}(k)) + 2k \leq \lceil \frac{5k}{2} \rceil + \left\lfloor 2\sqrt{\frac{k}{2}} \right\rfloor - 1$.

- $r_2(\text{CCC}(k)) \leq \frac{k}{2} + 2k = 5 \cdot \lceil \frac{k}{2} \rceil$ for even $k$.

- $r_2(\text{CCC}(k)) \leq \lceil \frac{k}{2} \rceil + 2k + 2 = 5 \cdot \lceil \frac{k}{2} \rceil$ for odd $k$.

- $r_2(\text{BF}(k)) \leq \frac{k}{2} + 2k = 5 \cdot \lceil \frac{k}{2} \rceil$ for even $k$.

- $r_2(\text{BF}(k)) \leq \lceil \frac{k}{2} \rceil + 2k + 2 = 5 \cdot \lceil \frac{k}{2} \rceil$ for odd $k$. 
Theorem:

Let $k \geq 3$, then we have:

- $r(\text{CCC}(k)) \leq r(C(k)) + 3k - 1 \leq \left\lceil \frac{7k}{2} \right\rceil + \left\lceil 2\sqrt{\frac{k}{2}} \right\rceil - 2$.
- $r(\text{BF}(k)) \leq r(C(k)) + 2k \leq \left\lceil \frac{5k}{2} \right\rceil + \left\lceil 2\sqrt{\frac{k}{2}} \right\rceil - 1$.
- $r_2(\text{CCC}(k)) \leq \frac{k}{2} + 2k = 5 \cdot \left\lceil \frac{k}{2} \right\rceil$ for even $k$.
- $r_2(\text{CCC}(k)) \leq \left\lceil \frac{k}{2} \right\rceil + 2k + 2 = 5 \cdot \left\lceil \frac{k}{2} \right\rceil$ for odd $k$.
- $r_2(\text{BF}(k)) \leq \frac{k}{2} + 2k = 5 \cdot \left\lceil \frac{k}{2} \right\rceil$ for even $k$.
- $r_2(\text{BF}(k)) \leq \left\lceil \frac{k}{2} \right\rceil + 2k + 2 = 5 \cdot \left\lceil \frac{k}{2} \right\rceil$ for odd $k$. 
Theorem:

Let \( k \geq 3 \), then we have:

\[ r(\text{CCC}(k)) \leq r(C(k)) + 3k - 1 \leq \left\lceil \frac{7k}{2} \right\rceil + \left\lfloor 2\sqrt{\left\lfloor \frac{k}{2} \right\rfloor} \right\rfloor - 2. \]

\[ r(\text{BF}(k)) \leq r(C(k)) + 2k \leq \left\lceil \frac{5k}{2} \right\rceil + \left\lfloor 2\sqrt{\left\lfloor \frac{k}{2} \right\rfloor} \right\rfloor - 1. \]

\[ r_2(\text{CCC}(k)) \leq \frac{k}{2} + 2k = 5 \cdot \left\lfloor \frac{k}{2} \right\rfloor \text{ for even } k. \]

\[ r_2(\text{CCC}(k)) \leq \left\lceil \frac{k}{2} \right\rceil + 2k + 2 = 5 \cdot \left\lceil \frac{k}{2} \right\rceil \text{ for odd } k. \]

\[ r_2(\text{BF}(k)) \leq \frac{k}{2} + 2k = 5 \cdot \left\lceil \frac{k}{2} \right\rceil \text{ for even } k. \]

\[ r_2(\text{BF}(k)) \leq \left\lceil \frac{k}{2} \right\rceil + 2k + 2 = 5 \cdot \left\lceil \frac{k}{2} \right\rceil \text{ for odd } k. \]
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.
Introduction

Simple Graphs

Networks

Complexity

Telephone-Mode

Telegraph-Mode

Sum.

Walter Unger 21.12.2018 14:00 SS2016

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.
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.
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**

**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.
Theorem:
The two-way and one-way gossip-problem on trees is in \( P \)

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.
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 \( P \)

Proof: simple exercise.

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

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 $\mathcal{NPC}$

Proof: Same way as the for the broadcast-problem.
Theorem:
The two-way and one-way gossip-problem on trees is in \( P \)

Proof: simple exercise.

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

Proof: Same way as for the broadcast-problem.
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 (0. Idea)
Gossip on Graphs with $2 \cdot m$ Nodes (0. Idea)
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)
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)

Implication:

For all $m \in \mathbb{N}$ we have:

$$r_{2}(K(2^{m})) = m$$

For all $m \in \mathbb{N}$ we have:

$$r_{2}(K(m)) \leq \lceil \log_{2} m \rceil + 1$$
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:
$$r_2(K(2^m)) = m$$

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:

$r_2(K(2^m)) = m$

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.
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 \(i \in \{0, \ldots, m - 1\}\) 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 $R[(i + 2^{t-1}) \mod m]$:  
  $\bigcup_{0 \leq j \leq 2^t - 1} \alpha[(i + j) \mod m]$

- 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:**
  
  ```text
  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 $R[(i + 2^{t-1}) \mod m]$:
    \[ \bigcup_{0 \leq j \leq 2^t - 1} \alpha[(i + j) \mod m] \]

- The invariant is easy to be shown.
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 $2 \cdot m + 1$ 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.
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.
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 (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 $v_0, v_1, v_2, \ldots, v_{n-1}$ 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:

$$r_2(K(m+1)) + 2 = \lceil \log_2(n+1) \rceil + 2 = \lceil \log_2 \left( \frac{n+1}{2} \right) \rceil + 2$$

Running time for $m + 1$ odd:

$$r_2(K(m+2)) + 2 = \lceil \log_2(n+3) \rceil + 1 = \lceil \log_2 n \rceil + 1$$
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:
\[
\begin{align*}
  r_2(K(m+1)) + 2 & = \left\lceil \log_2(m+1) \right\rceil + 2 \quad = \left\lceil \log_2 \left( \frac{n+1}{2} \right) \right\rceil + 2 \\
                    & = \left\lceil \log_2(n+1) \right\rceil + 1 \quad = \left\lceil \log_2 n \right\rceil + 1
\end{align*}
\]

Running time for $m + 1$ odd:
\[
\begin{align*}
  r_2(K(m+2)) + 2 & = \left\lceil \log_2(m+2) \right\rceil + 2 \quad = \left\lceil \log_2 \left( \frac{n+3}{2} \right) \right\rceil + 2 \\
                    & = \left\lceil \log_2(n+3) \right\rceil + 1 \quad = \left\lceil \log_2 n \right\rceil + 1
\end{align*}
\]
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:
\[
\log_2 (m + 1) + 2 = \left\lceil \log_2 (n + 1) \right\rceil + 1 = \left\lceil \log_2 n \right\rceil + 1
\]

Running time for $m + 1$ odd:
\[
\log_2 (m + 2) + 2 = \left\lceil \log_2 (n + 3) \right\rceil + 1 = \left\lceil \log_2 n \right\rceil + 1
\]
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:
\[
q_2(K(m+1)) + 2 \quad = \quad \lceil \log_2(m+1) \rceil + 2 \quad = \quad \lceil \log_2 \left( \frac{n+1}{2} \right) \rceil + 2 \\
\quad = \quad \lceil \log_2(n+1) \rceil + 1 \quad = \quad \lceil \log_2 n \rceil + 1
\]

Running time for $m + 1$ odd:
\[
q_2(K(m+2)) + 2 \quad = \quad \lceil \log_2(m+2) \rceil + 2 \quad = \quad \lceil \log_2 \left( \frac{n+3}{2} \right) \rceil + 2 \\
\quad = \quad \lceil \log_2(n+3) \rceil + 1 \quad = \quad \lceil \log_2 n \rceil + 1
\]
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:

$$\begin{align*}
    r_2(K(m+1)) + 2 &= \lceil \log_2(m+1) \rceil + 2 \\
    &= \lceil \log_2(n+1) \rceil + 1 \\
    &= \lceil \log_2 \left( \frac{n+1}{2} \right) \rceil + 1
\end{align*}$$

Running time for $m + 1$ odd:

$$\begin{align*}
    r_2(K(m+2)) + 2 &= \lceil \log_2(m+2) \rceil + 2 \\
    &= \lceil \log_2(n+3) \rceil + 1 \\
    &= \lceil \log_2 \left( \frac{n+3}{2} \right) \rceil + 1
\end{align*}$$
Gossip on Graphs with $2 \cdot m + 1$ Nodes

- Let $n = 2 \cdot m + 1$.
- Let $v_0, v_1, v_2, \ldots, v_{n-1}$ 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 $v_0, v_1, v_2, \ldots, v_{n-1}$ 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:**
\[
\begin{align*}
r_2(K(m + 1)) + 2 & = \lfloor \log_2(m + 1) \rfloor + 2 \\
& = \lfloor \log_2(n) \rfloor + 1
\end{align*}
\]

**Running time for $m + 1$ odd:**
\[
\begin{align*}
r_2(K(m + 2)) + 2 & = \lfloor \log_2(m + 2) \rfloor + 2 \\
& = \lfloor \log_2(n + 3) \rfloor + 1
\end{align*}
\]
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:

$$r_2(K(m+1)) + 2 = \lceil \log_2(m + 1) \rceil + 2 = \lceil \log_2\left(\frac{n+1}{2}\right) \rceil + 2 = \lceil \log_2(n) \rceil + 1$$

Running time for $m + 1$ odd:

$$r_2(K(m+2)) + 2 = \lceil \log_2(m + 2) \rceil + 2 = \lceil \log_2\left(\frac{n+3}{2}\right) \rceil + 2 = \lceil \log_2(n) \rceil + 1$$
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.
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.
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.
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.
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.
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\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.
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.
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\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.
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.
We need an additional two rounds.

$\nu_x$ and $\nu_y$ 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.
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)

- 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.
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.
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.
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.
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.
algorithm

- Let \( n = 2m \).

- Gossip-Algorithm:
  
  \[ \begin{align*}
  t &:= 0; \\
  \text{for all } i \in \{0, \ldots, m-1\} & \text{ do in parallel } R[i] \text{ sends to } Q[i]; \\
  \text{for all } i \in \{0, \ldots, m-1\} & \text{ do in parallel } Q[i] \text{ sends to } R[i]; \\
  \text{while } \text{fib}(2t+1) < m \text{ do begin} \\
  & t := t + 1; \\
  & \text{for all } i \in \{0, \ldots, m-1\} \text{ do in parallel} \\
  & \quad R[(i + \text{fib}(2t-1)) \mod m] \text{ sends to } Q[i]; \\
  & \quad \text{if } \text{fib}(2t) < m \text{ then} \\
  & \quad \quad \text{for all } i \in \{0, \ldots, m-1\} \text{ do in parallel} \\
  & \quad \quad \quad Q[(i + \text{fib}(2t)) \mod m] \text{ sends to } R[i] \\
  \text{end;}
\]
Let \( n = 2m \).

Gossip-Algorithm:
\[
\begin{align*}
t &:= 0; \\
\text{for all } i &\in \{0, \ldots, m-1\} \text{ do in parallel } R[i] \text{ sends to } Q[i]; \\
\text{for all } i &\in \{0, \ldots, m-1\} \text{ do in parallel } Q[i] \text{ sends to } R[i]; \\
\text{while } \text{fib}(2t + 1) < m \text{ do begin} \\
\quad t &:= t + 1; \\
\quad \text{for all } i &\in \{0, \ldots, m-1\} \text{ do in parallel} \\
\quad &\quad R[(i + \text{fib}(2t - 1)) \mod m] \text{ sends to } Q[i]; \\
\quad \text{if } \text{fib}(2t) < m \text{ then} \\
\quad &\quad \text{for all } i &\in \{0, \ldots, m-1\} \text{ do in parallel} \\
\quad &\quad &\quad Q[(i + \text{fib}(2t)) \mod m] \text{ sends to } R[i] \\
\text{end};
\end{align*}
\]
Let $n = 2m$.

Gossip-Algorithm:

$\begin{align*}
\text{Let } n &= 2m. \\
\text{Gossip-Algorithm:} & \quad t := 0; \\
& \quad \text{for all } i \in \{0, \ldots, m - 1\} \text{ do in parallel } R[i] \text{ sends to } Q[i]; \\
& \quad \text{for all } i \in \{0, \ldots, m - 1\} \text{ do in parallel } Q[i] \text{ sends to } R[i]; \\
& \quad \text{while } \text{fib}(2t + 1) < m \text{ do begin} \\
& \quad \quad t := t + 1; \\
& \quad \quad \text{for all } i \in \{0, \ldots, m - 1\} \text{ do in parallel} \\
& \quad \quad \quad R[(i + \text{fib}(2t - 1)) \mod m] \text{ sends to } Q[i]; \\
& \quad \quad \text{if } \text{fib}(2t) < m \text{ then} \\
& \quad \quad \quad \text{for all } i \in \{0, \ldots, m - 1\} \text{ do in parallel} \\
& \quad \quad \quad \quad Q[(i + \text{fib}(2t)) \mod m] \text{ sends to } R[i] \\
& \quad \text{end;}
\end{align*}$
Let \( n = 2m \).

Gossip-Algorithm:
\[
\begin{align*}
    t &:= 0; \\
    &\text{for all } i \in \{0, \ldots, m-1\} \text{ do in parallel } R[i] \text{ sends to } Q[i]; \\
    &\text{for all } i \in \{0, \ldots, m-1\} \text{ do in parallel } Q[i] \text{ sends to } R[i]; \\
    &\text{while } \text{fib}(2t + 1) < m \text{ do begin} \\
    &\quad t := t + 1; \\
    &\quad \text{for all } i \in \{0, \ldots, m-1\} \text{ do in parallel} \\
    &\quad \quad R[(i + \text{fib}(2t - 1)) \mod m] \text{ sends to } Q[i]; \\
    &\quad \text{if } \text{fib}(2t) < m \text{ then} \\
    &\quad \quad \text{for all } i \in \{0, \ldots, m-1\} \text{ do in parallel} \\
    &\quad \quad \quad Q[(i + \text{fib}(2t)) \mod m] \text{ sends to } R[i] \\
    &\text{end;}
\end{align*}
\]
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;
Theorem:

Let $n = 2^m$ 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.
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 $k = \min\{x \mid fib(x) \geq m\}$. Then we have $r(K(n)) \leq k + 1$.

Proof:
- The algorithm stops, if $fib(2t + 1) \geq m$ or $fib(2t) \geq m$ 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 $\cup_{0 \leq j \leq fib(2t+1)-1} \alpha[(i + j) \mod m]$
  - $R[i]$ knows $\cup_{0 \leq j \leq fib(2t+2)-1} \alpha[(i + j) \mod m]$
- The correctness is a direct result of this.

$\begin{align*}
fib(0) &= fib(1) = 1 \\
fib(i) &= fib(i - 1) + fib(i - 2)
\end{align*}$
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 \( U_{0 \leq j \leq \text{fib}(2t+1)-1} \alpha[(i + j) \mod m] \)
  - \( R[i] \) knows \( U_{0 \leq j \leq \text{fib}(2t+2)-1} \alpha[(i + j) \mod m] \)
- 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 $\cup_{0 \leq j \leq \text{fib}(2t+1)-1} \alpha[(i + j) \mod m]$
  - $R[i]$ 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.
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 \( n = 2m - 1 \) 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: \( r(K(n)) \geq 2 + \left\lceil \log_{\frac{1}{2}(1+\sqrt{5})} \frac{n}{2} \right\rceil \).

**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>
**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>8</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.
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: $r(G) \geq nc(G)$. 

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: $r(G) \geq nc(G)$. 
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 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: $r(G) \geq nc(G)$. 
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 an $n \times n$ 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 $z^t_i$.
- One situation of the network counting problem could be described by a vector:
  - Initial: $(1, 1, 1, \cdots, 1)^T$.
  - After $t$ rounds: $(z^t_1, z^t_2, z^t_3, z^t_n)^T$.

- 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: $A \cdot (z^t_1, z^t_2, z^t_3, z^t_n)^T = (z^{t+1}_1, z^{t+1}_2, z^{t+1}_3, z^{t+1}_n)^T$
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

- 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 $z^t_i$.
- One situation of the network counting problem could be described by a vector:
  - Initial: $(1, 1, 1, \ldots, 1)^T$.
  - After $t$ rounds: $(z^t_1, z^t_2, z^t_3, z^t_n)^T$.

- 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: $A \cdot (z^t_1, z^t_2, z^t_3, z^t_n)^T = (z^{t+1}_1, z^{t+1}_2, z^{t+1}_3, z^{t+1}_n)^T$
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 $z^t_i$.
- One situation of the network counting problem could be described by a vector:
  - Initial: $(1, 1, 1, \cdots, 1)^T$.
  - After $t$ rounds: $(z^t_1, z^t_2, z^t_3, z^t_n)^T$.
- 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: $A \cdot (z^t_1, z^t_2, z^t_3, z^t_n)^T = (z^{t+1}_1, z^{t+1}_2, z^{t+1}_3, z^{t+1}_n)^T$
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: \((z^t_1, z^t_2, z^t_3, z^t_n)^T\).

- 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: \( A \cdot (z^t_1, z^t_2, z^t_3, z^t_n)^T = (z^{t+1}_1, z^{t+1}_2, z^{t+1}_3, z^{t+1}_n)^T \)
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 (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:

$$TAT^{-1} = \begin{pmatrix} B & 0 \\ B & 1 \\ 0 & 1 \end{pmatrix}.$$  

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:

$$
\begin{pmatrix}
B & 0 \\
B & B \\
0 & 1 \\
\end{pmatrix}
$$

$$
T A T^{-1} =
\begin{pmatrix}
B & 0 \\
B & B \\
0 & 1 \\
\end{pmatrix}
$$

and $B = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$.

- We will estimate the growth, which these matrices provide for the network counting problem.
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 \( A, B \in \mathbb{R}^{n \times n}, x \in \mathbb{R}^n, \alpha \in \mathbb{R}, \alpha \geq 0 \).

Here we use: \( \| x \| = \sqrt{\sum_{i=1}^{n} |x_i|^2} \) for 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 \( \| \cdot \| \) 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 \( x = (x_1, \ldots, x_n) \).

- 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)

1. Let $||\cdot||$ 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 $\alpha \in \mathbb{R}, x, y \in \mathbb{R}^n$

2. The matrix norm for a vector norm $||\cdot||$ 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 $A, B \in \mathbb{R}^{n^2}, x \in \mathbb{R}^n, \alpha \in \mathbb{R}, \alpha \geq 0$.

3. Here we use: $||x|| = \sqrt{\sum_{i=1}^{n} |x_i|^2}$ for $x = (x_1, \ldots, x_n)$.

4. Known: $||A|| = $ 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 $\text{e}i n$ $x = (x_1, ..., x_n)$.

- 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:
  - $\|x\| = 0 \Leftrightarrow x = 0^n$,
  - $\|\alpha \cdot x\| = |\alpha| \cdot \|x\|$,
  - $\|x + y\| \leq \|x\| + \|y\|$
  - this holds for all $\alpha \in \mathbb{R}, x, y \in \mathbb{R}^n$

- The matrix norm for a vector norm $\|\cdot\|$ 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 $x = (x_1, \ldots, x_n)$.

- 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 $A, B \in \mathbb{R}^{n \times n}, x \in \mathbb{R}^n, \alpha \in \mathbb{R}, \alpha \geq 0$.

- Here we use: $||x|| = \sqrt{\sum_{i=1}^{n} |x_i|^2}$ for every $x = (x_1, \ldots, x_n)$.

- 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 $x = (x_1, ..., x_n)$,

- Known: $||A|| = \text{Spectral Norm}(A) = \sqrt{\lambda_{\text{max}}(A^T \cdot A)}$ with: $\lambda_{\text{max}}$ is the largest Eigenvalue.
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.
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 each $x = (x_1, \ldots, x_n)$.

- 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 $||x||$ 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 $e \in x = (x_1, ..., x_n)$.

- 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 $A, B \in \mathbb{R}^{n^2}, x \in \mathbb{R}^n, \alpha \in \mathbb{R}, \alpha \geq 0$.

- Here we use: $||x|| = \sqrt{\sum_{i=1}^{n} |x_i|^2}$ for $\text{e}in \ x = (x_1, .., x_n)$.

- 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 $x = (x_1, \ldots, x_n)$.

- 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 $e_i$ in $x = (x_1, \ldots, x_n)$.

- 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 $||x||$ 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 $\text{e}in \ x = (x_1, \ldots, x_n)$.

- Known: $||A|| = \text{Spectral Norm}(A) = \sqrt{\lambda_{max}(A^T \cdot A)}$ with: $\lambda_{max}$ is the largest Eigenvalue.
2. Abstraction (Continuation)

- We compute the spectral norm:
  \[ \|A\| = \|TA(T^{-1})\| = \|B\|. \]
  \[ B^T \cdot B = \begin{pmatrix} 10 & 11 \\ 11 & 01 \end{pmatrix} = \begin{pmatrix} 11 \\ 12 \end{pmatrix}. \]
  \[ \Rightarrow (2 - \lambda)(1 - \lambda) - 1 = 0 \]
  \[ \Rightarrow \lambda^2 - 3\lambda + 1 = 0 \]
  \[ \Rightarrow \lambda_{\text{max}}(B^T B) = \frac{3}{2} + \sqrt{\frac{5}{4}} \]
  \[ \|A\| = \sqrt{\lambda_{\text{max}}(AT A)} = \frac{1}{2}(1 + \sqrt{5}) \]
2. Abstraction (Continuation)

- We compute the spectral norm:
  
  \[ ||A|| = ||TAT^{-1}|| = ||B||.\]

\[ B^T \cdot B = \begin{pmatrix} 10 & 11 \\ 11 & 01 \end{pmatrix} = \begin{pmatrix} 11 \\ 12 \end{pmatrix}.\]

\[ \Rightarrow (2 - \lambda)(1 - \lambda) - 1 = 0 \]
\[ \Rightarrow \lambda^2 - 3\lambda + 1 = 0 \]
\[ \Rightarrow \lambda_{\text{max}}(B^T B) = \frac{3}{2} + \sqrt{\frac{5}{4}} \]

\[ ||A|| = \sqrt{\lambda_{\text{max}}(A^T A)} = \frac{1}{2}(1 + \sqrt{5}) \]
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:
  - $||A|| = ||TAT^{-1}|| = ||B||$.
  - $B^T \cdot B = \begin{pmatrix} 10 & 11 \\ 11 & 01 \end{pmatrix} = \begin{pmatrix} 11 & 12 \end{pmatrix}$.
  - $\Rightarrow (2 - \lambda)(1 - \lambda) - 1 = 0$
  - $\Rightarrow \lambda^2 - 3\lambda + 1 = 0$
  - $\Rightarrow \lambda_{\max}(B^T B) = \frac{3}{2} + \sqrt{\frac{5}{4}}$
  - $||A|| = \sqrt{\lambda_{\max}(A^T A)} = \frac{1}{2}(1 + \sqrt{5})$
2. Abstraction (Continuation)

- We compute the spectral norm:
  - $||A|| = ||TAT^{-1}|| = ||B||$.
  - $B^T \cdot B = \begin{pmatrix} 10 \\ 11 \end{pmatrix} \begin{pmatrix} 11 \\ 01 \end{pmatrix} = \begin{pmatrix} 11 \\ 12 \end{pmatrix}$.
  - $\Rightarrow (2 - \lambda)(1 - \lambda) - 1 = 0$
  - $\Rightarrow \lambda^2 - 3\lambda + 1 = 0$
  - $\Rightarrow \lambda_{\text{max}}(B^T B) = \frac{3}{2} + \sqrt{\frac{5}{4}}$
  - $||A|| = \sqrt{\lambda_{\text{max}}(A^T A)} = \frac{1}{2}(1 + \sqrt{5})$
2. Abstraction (Continuation)

- We compute the spectral norm:
- \( ||A|| = ||TAT^{-1}|| = ||B||. \)
- \( B^T \cdot B = \begin{pmatrix} 10 \\ 11 \end{pmatrix} \begin{pmatrix} 11 \\ 01 \end{pmatrix} = \begin{pmatrix} 11 \\ 12 \end{pmatrix}. \)
- \( \Rightarrow (2 - \lambda)(1 - \lambda) - 1 = 0 \)
- \( \Rightarrow \lambda^2 - 3\lambda + 1 = 0 \)
- \( \Rightarrow \lambda_{\text{max}}(B^T B) = \frac{3}{2} + \sqrt{\frac{5}{4}} \)
- \( ||A|| = \sqrt{\lambda_{\text{max}}(A^T A)} = \frac{1}{2}(1 + \sqrt{5}) \)
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.
- $\alpha := (\alpha_1, \alpha_2, \ldots, \alpha_n)^T = A_r \cdot \ldots \cdot A_2 \cdot A_1 \cdot (1, 1, \ldots, 1)$.
- $||\alpha|| \leq (\prod_{i=1}^{r-2} ||A_i||) \cdot ||(1, \ldots, 1)|| \leq (\frac{1}{2} (1 + \sqrt{5}))^{r-2} \cdot \sqrt{n}$
- 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 $i \in \{1, 2, \ldots, n\}$.
- 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 $2 + \left\lceil \log_{\frac{1}{2}}(1 + \sqrt{5}) \cdot \frac{n}{2} \right\rceil$ rounds.

**Proof:**

- Let $A_j, 1 \leq j \leq r$ be matrices, which solve the problem in $r$ rounds.
- $\alpha := (\alpha_1, \alpha_2, \cdots, \alpha_n)^T = A_{r-2} \cdot \cdots \cdot A_2 \cdot A_1 \cdot (1, 1, \cdots, 1)$.
- $||\alpha|| \leq (\prod_{i=1}^{r-2} ||A_i||) \cdot ||(1, \ldots, 1)|| \leq (\frac{1}{2}(1 + \sqrt{5}))^{r-2} \cdot \sqrt{n}$
- 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 $2 + \lceil \log_{\frac{1}{2}} \left( 1 + \sqrt{5} \right)^{\frac{n}{2}} \rceil$ 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 $2 + \lceil \log_{\frac{1}{2}}(1 + \sqrt{5}) \frac{n}{2} \rceil$ 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 $\inf(i, t)$ 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: $\alpha_i + \alpha_j \geq n$. 
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.
- \(\alpha := (\alpha_1, \alpha_2, \ldots, \alpha_n)^T = A_{r-2} \cdots A_2 \cdot A_1 \cdot (1, 1, \cdots, 1)\).
- \(||\alpha|| \leq (\prod_{i=1}^{r-2} ||A_i||) \cdot ||(1, ..., 1)|| \leq (\frac{1}{2} (1 + \sqrt{5}))^{r-2} \cdot \sqrt{n}\)
- 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.
- $\alpha := (\alpha_1, \alpha_2, \ldots, \alpha_n)^T = A_{r-2} \cdot \ldots \cdot A_2 \cdot A_1 \cdot (1, 1, \ldots, 1)$.
- $||\alpha|| \leq (\prod_{i=1}^{r-2} ||A_i||) \cdot ||(1, \ldots, 1)|| \leq (\frac{1}{2} (1 + \sqrt{5}))^{r-2} \cdot \sqrt{n}$
- 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 $i \in \{1, 2, \ldots, n\}$.
- 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.
- \(\alpha := (\alpha_1, \alpha_2, \cdots, \alpha_n)^T = A_{r-2} \cdot \cdots \cdot A_2 \cdot A_1 \cdot (1, 1, \cdots, 1)\).
- \(\|\alpha\| \leq (\prod_{i=1}^{r-2} ||A_i||) \cdot ||(1, ..., 1)|| \leq (\frac{1}{2} (1 + \sqrt{5}))^{r-2} \cdot \sqrt{n}\)
- 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 $2 + \lceil \log_{\frac{1}{2}}(1 + \sqrt{5}) \frac{n}{2} \rceil$ 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 $\inf(i, t)$ 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: \(\alpha_i + \alpha_j \geq n\).
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: \( 2c_1 + c_3 \geq \frac{n}{2} \).

\[ ||\alpha|| = \sqrt{\sum_{i=1}^{n} \alpha_i^2} \geq \sqrt{c_1 n^2 + c_3 \cdot 2 \cdot \frac{n^2}{4}} \geq n \cdot \sqrt{\frac{1}{2}(2c_1 + c_3)} \geq \frac{n}{2} \cdot \sqrt{n}. \]

We already have:

\[ ||\alpha|| \leq \left( \prod_{i=1}^{r-2} ||A_i|| \right) \cdot ||(1, ..., 1)|| \leq (\frac{1}{2}(1 + \sqrt{5}))^{r-2} \cdot \sqrt{n}. \]

And we get:

\[ \frac{n}{2} \cdot \sqrt{n} \leq ||\alpha|| \leq \Phi^{r-2} \cdot \sqrt{n}, \]

From which we conclude:

\[ r \geq 2 + \left\lceil \log_{\frac{1}{2}(1+\sqrt{5})} \frac{n}{2} \right\rceil \]
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:

$$r \geq 2 + \left[\log_{\frac{1}{2}(1+\sqrt{5})} \frac{n}{2}\right]$$
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: $2c_1 + c_3 \geq \frac{n}{2}$

$$||\alpha|| = \sqrt{\sum_{i=1}^{n} \alpha_i^2} \geq \sqrt{c_1 n^2 + c_3 \cdot 2 \cdot \frac{n^2}{4}} \geq n \cdot \sqrt{\frac{1}{2} (2c_1 + c_3)} \geq \frac{n}{2} \cdot \sqrt{n}.$$  

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:

$$r \geq 2 + \left\lceil \log_{\frac{\Phi}{2}} \left( \frac{n}{2} \right) \right\rceil$$
Continuation

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:

\[r \geq 2 + \lceil \log_{\frac{1}{2}(1+\sqrt{5})} \frac{n}{2} \rceil
\]
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:

$r \geq 2 + \lceil \log_{\frac{1}{2}(1+\sqrt{5})} \frac{n}{2} \rceil$
\begin{itemize}
  \item Let
    \begin{itemize}
      \item $c_1$ be the number of cases with: $\alpha_i \geq n$,
      \item $c_2$ be the number of cases with: $\alpha_i < n$ and $\alpha_j \geq n$,
      \item $c_3$ be the number of cases with: $\alpha_i < n$, $\alpha_j < n$ and $\alpha_i + \alpha_j \geq n$.
    \end{itemize}
  \item Then we have: $c_1 \geq c_2$ and $c_1 + c_2 + c_3 \geq n/2$.
  \item Thus we also get: $2c_1 + c_3 \geq \frac{n}{2}$
  \item $\|\alpha\| = \sqrt{\sum_{i=1}^{n} \alpha_i^2} \geq \sqrt{c_1 n^2 + c_3 \cdot 2 \cdot \frac{n^2}{4}} \geq n \cdot \sqrt{\frac{1}{2} (2c_1 + c_3)} \geq \frac{n}{2} \sqrt{n}$.
  \item We already have:
    \[ \|\alpha\| \leq \left( \prod_{i=1}^{r-2} \|A_i\| \right) \cdot \|(1, \ldots, 1)\| \leq \left( \frac{1}{2} (1 + \sqrt{5}) \right)^{r-2} \cdot \sqrt{n}. \]
  \item And we get:
    \[ \frac{n}{2} \cdot \sqrt{n} \leq \|\alpha\| \leq \Phi^{r-2} \cdot \sqrt{n}, \]
  \item From which we conclude:
    \[ r \geq 2 + \left\lceil \log_{\frac{1}{2} (1+\sqrt{5})} \frac{n}{2} \right\rceil \]
\end{itemize}
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:

$$r \geq 2 + \left\lceil \log \frac{1}{2} \left(1+\sqrt{5}\right) \frac{n}{2} \right\rceil.$$
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:

\[||\alpha|| \leq \left( \prod_{i=1}^{r-2} ||A_i|| \right) \cdot ||(1, ..., 1)|| \leq \left( \frac{1}{2}(1 + \sqrt{5}) \right)^{r-2} \cdot \sqrt{n}.\]

And we get:

\[\frac{n}{2} \cdot \sqrt{n} \leq ||\alpha|| \leq \Phi^{r-2} \cdot \sqrt{n},\]

From which we conclude:

\[r \geq 2 + \lceil \log_{\frac{1}{2}(1+\sqrt{5})} \frac{n}{2} \rceil\]
Continuation

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:
$$r \geq 2 + \left\lceil \log_{\frac{1}{2} (1+\sqrt{5})} \frac{n}{2} \right\rceil$$
Continuation

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:

\[
r \geq 2 + \left[ \log_{\frac{1}{2} (1+\sqrt{5})} \frac{n}{2} \right].
\]
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 $n$. 

8:54 Lower Bound

1/11
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
- $t_2 := 2 + \lceil \log_{\frac{1}{2}} (1 + \sqrt{5}) \rceil m$.

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 $t_2 = 2 + \lceil \log_\Phi F(k) \rceil = 2 + k - 1 = k + 1$.
  - 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 \( n \).
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
- \( t_2 := 2 + \lceil \log_2 \left( 1 + \sqrt{5} \right) m \rceil \).

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
- $t_2 := 2 + \lceil \log_\frac{1}{2} (1 + \sqrt{5}) m \rceil$.

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 $n$. 

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 (Part 2)

**Lemma:**

Let \( n = 2m \) and let:

\[ t_1 := 1 + k, \text{ with } k \text{ is the smallest number with } m \leq F(k) \text{ and } \]
\[ t_2 := 2 + \lceil \log_{\frac{1}{2}(1+\sqrt{5})} m \rceil. \]

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
- $t_2 := 2 + \lceil \log_{\frac{1}{2}(1+\sqrt{5})} m \rceil$.

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
- $t_2 := 2 + \lceil \log_{\frac{1}{2}}(1+\sqrt{5}) \cdot m \rceil$.

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
- $t_2 := 2 + \left\lceil \log_{\frac{1}{2}(1+\sqrt{5})} m \right\rceil$.

Then we have $t_1 = t_2$ for infinite many $m$ and $t_1 \leq t_2 + 1$ for all $m$.

**Proof:**

- 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: $t_1 = k + 1 \leq i + 3$. 
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: \( t_1 = k + 1 \leq i + 3 \).
## Summary (Telefon-Mode)

| Graph  | $|V|$ | $diam$ | Lower Bound | Upper Bound |
|--------|------|--------|-------------|-------------|
| $K_n$  | $n$  | 1      | $\lceil \log_2 n \rceil + odd(n)$ | $\lceil \log_2 n \rceil + odd(n)$ |
| $H_k$  | $2^k$ | $k$    | $k$         | $k$         |
| $P_n$  | $n$  | $n - 1$ | $n - even(n)$ | $n - even(n)$ |
| $C_n$  | $n$  | $\lceil \frac{n}{2} \rceil$ | $\lceil \frac{n}{2} \rceil + odd(n)$ | $\lceil \frac{n}{2} \rceil + odd(n)$ |
| $CCC_k$ | $k \cdot 2^k$ | $\left\lfloor \frac{5k}{2} \right\rfloor - 2$ | $\left\lfloor \frac{5k}{2} \right\rfloor - 2$ | $\left\lfloor \frac{5k}{2} \right\rfloor - 2, k$ even |
| $SE_k$ | $2^k$ | $2k - 1$ | $2k - 1$ | $2k + 5$ |
| $BF_k$ | $k \cdot 2^k$ | $\left\lfloor \frac{3k}{2} \right\rfloor$ | $1.9770k$ | $2.25 \cdot k + o(k)$ |
| $DB_k$ | $2^k$ | $k$ | $1.5965k$ | $2k + 5$ |
Summary (Telegraph-Mode)

| Graph | $|V|$ | diam | Lower Bound | Upper Bound |
|-------|------|------|-------------|-------------|
| $K_n$ | $n$  | 1    | $1.44 \log_2 n$ | $1.44 \log_2 n$ |
| $H_k$ | $2^k$ | $k$  | $1.44k$ | $1.88k$ |
| $P_n$ | $n$  | $n-1$ | $n + \text{odd}(n)$ | $n + \text{odd}(n)$ |
| $C_n$ | $n$ even | $\lceil \frac{n}{2} \rceil$ | $\frac{n}{2} + \lceil \sqrt{2n} \rceil - 1$ | $\frac{n}{2} + \lceil \sqrt{2n} \rceil - 1$ |
|       | $n$ odd | $\lceil \frac{n}{2} \rceil$ | $\frac{n}{2} + \lceil \sqrt{2n} - \frac{1}{2} \rceil - 1$ | $\frac{n}{2} + \lceil 2 \sqrt{\frac{n}{2}} \rceil - 1$ |
| $CCC_k$ | $k \cdot 2^k$ | $\lceil \frac{5k}{2} \rceil - 2$ | $\lceil \frac{5k}{2} \rceil - 2$ | $\lceil \frac{7k}{2} \rceil + \lceil 2 \sqrt{\frac{k}{2}} \rceil - 2$ |
| $SE_k$ | $2^k$ | $2k - 1$ | $2k - 1$ | $3k + 3$ |
| $BF_k$ | $k \cdot 2^k$ | $\lceil \frac{3k}{2} \rceil$ | $1.9770k$ | $\lceil \frac{5k}{2} \rceil + \lceil 2 \sqrt{\frac{k}{2}} \rceil - 1$ |
| $DB_k$ | $2^k$ | $k$ | $1.5965k$ | $3k + 3$ |
Literatur

J. Hromkovič, et al.:
Dissemination of Information in Communication Networks:
Broadcasting, Gossiping, Leader Election, and Fault-Tolerance.
Legend

- : Not of relevance
- : implicitly used basics
- : idea of proof or algorithm
- : structure of proof or algorithm
- : Full knowledge