Distributed Computing over Communication Networks:

Locality Lower Bounds
Vertex Coloring: Results so far?

E.g., on trees in $\log^*(n)$ time, down to 6 colors...

... and then shift-down: down to 3 colors (same complexity).
From trees to rings...

How to color a ring?
Ring Coloring

Possible output:

See exercise session:
Algo for trees can be adapted!
So $\log^*(n)$ time...!
Assume unique node IDs:

Lower bound for # colors without communication?

\( n \)

Lower bound for # colors with one communication round?

\( \log n \)

Lower bound for # colors with two communication rounds?

\( \log \log n \)

Lower bound for # colors with \( \log^* n \) communication rounds?

\( O(1) \)
3-color a ring: \( \log^*(n) \) time is optimal!

Class of algos?
Need assumptions!
1. synchronous, **directed** ring (communication in both directions and nodes can differentiate between clockwise and counter-clockwise)
2. IDs from 1...n (not in order, otherwise trivial!)
3. unbounded message size

The stronger assumptions for which the lower bound is still high the better for us!

Remember „local algorithm“ is symmetric: each node executes the same code! We will see: dan differentiate only in terms of neighborhoods...
Canonical Form of Distributed Algorithm?

What can a distributed algorithm do or learn in $r$ rounds?

1. Initially, all nodes only know their own ID
2. As information needs at least $r$ rounds to travel $r$ hops, a node can only learn about $r$-hop neighborhood!

Note that any local $r$-round algorithm can be brought into canonical form!

**Canonical Form**

1. First, in $r$ rounds: send initial state to nodes at distance $r$
2. Then: compute output based on complete information about $r$-hop neighborhood

In other words: we can emulate any local algorithm by making all communication first and then do all local computations! Why?

Example „leader election“:

Whether nodes only forward highest ID so far or whether all information is collected first and later selected does not make a difference!
No Deterministic Local Algorithms Can Do More...

We can do all communication first and then do all local computations!

How to prove this?

Let $A$ be any $r$-round algorithm. We can show that the canonical form algorithm $C$ can compute all possible messages that $A$ may send as well.

By induction over distance of nodes...:
if we can compute messages of first $i$ rounds in $(r-i+1)$-neighborhood, we have all information to compute first $(i+1)$ round messages in $(r-i)$-neighborhood. See „Skript“. 😊
Implication?

**Takeaway**

A local coloring algorithm can be seen as a function which takes neighborhoods and outputs colors.
Local Views

This motivates the following definition:

**r-Hop View**

We call the collection of the initial states of all nodes in the r-neighborhood of a node v the „r-hop view of v“.

Due to our canonical form lemma, this means that:

**Deterministic r-Round Algo**

A deterministic r-round algorithm A is a function that maps every possible r-hop view to the set of possible outputs.

Implication for nodes with same view? Must produce *same output*, in any algorithm!
So if *any local algorithm* can be *emulated* by a canonic algorithm, the question remains:

**How good can a canonic algorithm maximally be?**
Rings

How do r-hop views of our rings look like?
E.g., 1-hop view of 4?
Rings

How do r-hop views of our rings look like? E.g., 1-hop view of 4?
Rings

How do r-hop views of our rings look like?
E.g., 2-hop view of 4?
Rings

How do r-hop views of our rings look like? E.g., 1-hop view of 4?
How do r-hop views of our rings look like?

Generally:

The r-hop view of a ring is a \((2r+1)\) tuple:

\[(l_{-r}, l_{-r+1}, ..., l_0, ..., l_r)\]

where \(l_0\) is ID/label of considered node \(v\).

A deterministic coloring algorithm maps these tuples to colors!

Question: why tuple and not set? Sense of orientation! 😊
**Ring Colorings**

When is a coloring valid?

Consider two r-hop views:

\[(l_{-r}, l_{-r+1}, \ldots, l_0, \ldots, l_r)\]

and

\[(l'_{-r}, l'_{-r+1}, \ldots, l'_0, \ldots, l'_r)\]

where \(l'_i = l_{i+1}\) for \(-r \leq i \leq r-1\) and \(l'_r \neq l_i\) for \(-r \leq i \leq r\), so what?

Then the two views can originate from adjacent nodes in the ring! So?

**So every algorithm needs to assign different colors to the two views!**

1-hop view of 2:

1-hop view of 1:

(1,2,3) and (4,1,2) must give different colors (for 1 and 2, respectively!)
Neighborhood Graphs?

What if we define a neighborhood graph: neighborhoods are nodes, and connected if they are conflicting (i.e., views may originate from two adjacent nodes)?

Assume we color the neighborhood graph as follows: „view node“ has color of the node the neighborhood is computed from by deterministic local r-round algo.

How does the coloring of the neighborhood graph look like then?

Same neighborhood = same color, and?
Neighborhood Graphs?

Given collected neighborhoods, canonic coloring ALG colors adjacent nodes differently:

So corresponding views/nodes in neighborhood graph must have different colors too, so valid coloring for neighborhood graph:
Neighborhood Graph

„Formal“ definition:

**Neighborhood Graph**

The r-neighborhood graph $N_r(G)$ consists of all r-hop views of G (for all nodes) which are connected iff they could originate from two adjacent nodes.

This lemma motivates the concept:

**Lemma**

There is an r-round algorithm that colors graphs G with c colors iff the chromatic number of the neighborhood graph is $\chi(N_r(G)) \leq c$.

Proof?
Lemma

There is an **r-round algorithm** that colors graphs G with c colors iff the chromatic number of the neighborhood graph is \( \chi(N_r(G)) \leq c \).

Proof:

Because r-round algorithm defines legal coloring on neighborhood graph! (Everything else could yield conflict: neighborhood graph contains all possible conflicts.)

We know: local coloring algo is a function that maps r-hop view to color, so to every node of \( N_r(G) \).

This coloring is legal: by the definition of r-hop neighborhood graphs, adjacent nodes of \( N_r(G) \) must have different colors, since the corresponding nodes in the underlying graph are also adjacent. (But maybe slightly more than c colors are needed, so “\( \leq \)”...)

QED

So how do neighborhood graphs of rings look like? How to color them? And how to exploit the lemma to get a lower bound?
How to find a good lower bound with this lemma?

We have to show that $\chi(N_r(G))$ is small only for a large $r$...

So how does $N_r(G)$ of a ring look like?

For example of our initial ring graph?
Nr(Given Ring)?

0-hop neighborhood graph?

\[ \chi(G) = 2 \text{ or } 3 \]

1-hop neighborhood graph?

\[ \chi(G) = 2 \text{ or } 3 \]

2-hop neighborhood graph?

\[ \chi(G) = 2 \text{ or } 3 \]

So 0 or 1 round to 3-color?!?

Attention: We are interested in neighborhood graphs of families of graphs / rings!
A given graph is easy (neighborhoods trivial)! ☺
Nr(Ring)?

r-hop neighborhood graph for ring family (n=6 known)?

$N_0 = ?$

Complete graph: every node could be neighbor of every other node

$\chi(N_0) = ?$

= $n$

$N_1 = ?$

Any 0-local algorithm can only choose its ID as a color…: $n$ colors

$\chi(N_1) = ???$

Not easy although quite regular…
What happens for larger neighborhoods?

Intuitively, the larger the considered neighborhood, the less conflicts are possible! Chromatic number declines for larger $r$... (We will see: in logarithmically „per hop“!) At some point, the graph family member is clear!
Main question now: What is $\chi(N_r(\text{Ring}))$??

Difficult... So let’s focus on a graph which is similar, but has less conflicts and hence its chromatic number can be used instead for the lower bound!

What graphs are good then?

E.g., subgraphs...: less conflicts, so weaker lower bound when applying our lemma!
Instead of defining neighborhood graphs for rings:

**B_{k,n} Graph**

Assume two integers \( k, n \) where \( n \geq k \). The \( B_{k,n} \) graph consists of the nodes of \( k \)-tuples of increasing node labels (from \( \{1,...,n\} \)). There is a directed edge from node \( \alpha \) to node \( \beta \) iff \( \forall i \in \{1,...,k-1\}: \beta_i = \alpha_{i+1} \).

**Example: k=2, n=4**

\[
V(B_{k,n}) = \{(1,2),(1,3),(1,4),(2,3),(2,4),(3,4)\}
\]

\[
E(B_{k,n}) = \{((1,2),(2,3)), ((2,3),(3,4)), ((1,2),(2,4)) \}
\]
Neighborhood Graph of Ring

What does this have to do with rings?!

**Lemma**

Viewed as an undirected graph, $B_{2r+1,n}$ is a subgraph of the $r$-neighborhood graph of $n$-node rings with node labels from $\{1,\ldots,n\}$.

Example: Neighborhood $r=1$ (so $k=3$), $n=4$

\[
\begin{align*}
V(B_{k,n}) &= ? \\
&= \{(1,2,3),(1,2,4), (1,3,4),(2,3,4)\}
\end{align*}
\]

\[
\begin{align*}
E(B_{k,n}) &= ? \\
&= \{\{(1,2,3),(2,3,4)\}\}
\end{align*}
\]

Indeed! Neighborhood of 2 and 3! But only a subgraph! (Why?)
Lemma

Viewed as an undirected graph, $B_{2r+1,n}$ is a subgraph of the $r$-neighborhood graph of $n$-node rings with node labels from $\{1,\ldots,n\}$.

Proof?

The set of $k$-tuples of increasing labels is a subset of all the $k$-tuples / nodes (in our example, views of node 1 and 4 are missing).

Two nodes are only connected in $B_{2r+1,n}$ if there is also an edge in the neighborhood graph (because labels are ordered, the views must come from adjacent nodes): not more edges/conflicts.

What does it mean?!

Chromatic number of $B_{2r+1,n}$ good for lower bound of our problem!
- We have to compute lower bound for $\chi(B_{2r+1,n})$!
- How? With another helper graph... 😊
Diline Graph

The directed line graph (diline graph) $DL(G)$ of a directed graph $G=(V,E)$ is defined as follows: $V(DL(G))=E$, and there is a directed edge $((w,x),(y,z))$ iff $x=y$.

In other words: $DL(G)$ consists of the node representing the edges of $G$, and two nodes are connected if the corresponding edges „follow“ after each other.

Example:

What is the relation to $B_{k,n}$?!
Recursive Construction

$B_{k,n}$ can be recursively defined by directed line graphs!

**Lemma**

$B_{k+1,n} = DL(B_{k,n})$

Really? Example: $k=2$, $n=4$?

- $V(B_{k,n}) = \{(1,2),(1,3),(1,4),(2,3),(2,4),(3,4)\}$
- $E(B_{k,n}) = \{((1,2),(2,3)), ((2,3),(3,4)), ((1,2),(2,4)), ((1,3),(3,4))\}$

Example: $k=3$, $n=4$?

- $V(B_{k,n}) = \{(1,2,3),(1,2,4),(1,3,4),(2,3,4)\}$
- $E(B_{k,n}) = \{((1,2,3),(2,3,4))\}$
Recursive Construction

**Lemma**

\[ B_{k+1,n} = DL(B_{k,n}) \]

**Proof?**

By the definition of \( B_{k,n} \), two nodes \( \alpha, \beta \) are connected if the first \( k-1 \) labels in \( \beta \) are the same as the last \( k-1 \) labels of \( \alpha \).

Therefore, the pair \((\alpha, \beta)\) can be represented by a \((k+1)\) tuple \( \gamma=(\gamma_1, \ldots, \gamma_{k+1}) \) with \( \gamma_1=\alpha_1 \), \( \gamma_i=\beta_{i-1}=\alpha_i \) for \( 2 \leq i \leq k \), and \( \gamma_{k+1}=\beta_k \). The labels of \( \gamma \) are increasing too! So \( B_{k+1,n} \) has the same node set as \( DL(B_{k,n}) \).

What about the edges?
Recursive Construction

**Lemma**

\[ B_{k+1,n} = DL(B_{k,n}) \]

Proof (continued for edges...)

There is an edge between two nodes \((\alpha, \beta)\) and \((\alpha', \beta')\) of \(DL(B_{k,n})\) if \(\beta = \alpha'\).

This is equivalent to that the two corresponding \((k+1)\)-tuples \(\gamma\) and \(\gamma'\) are neighbors in \(B_{k+1,n}\):

- the last \(k\) labels of \(\gamma\) are equivalent to the first \(k\) labels of \(\gamma'\).

QED

So, \(B_{k,n}\) graphs are simply „iterated line graphs“!
Chromatic Numbers

Implication for colorings, coloring $G$ vs $DL(G)$?

Proof idea?

Given a $c$-coloring of $DL(G)$ we construct a $2^c$ coloring of $G$ (so minimal coloring of $G$ can only be smaller).

How does coloring of $G$ and $DL(G)$ relate?

Note: A $c$-coloring of the diline graph $DL(G)$ can be seen as a coloring of the edges of $G$ such that no two adjacent edges have the same color (definition of $DL(G)$).

$\chi(DL(G)) \geq \log_2(\chi(G))$

Attention: these graphs are generally not rings of course! 😊
Chromatic Numbers

Implication for colorings, coloring $G$ vs $DL(G)$?

Proof idea (continued...)

For a node $v \in G$, let $S_v$ denote the set of colors of its outgoing edges in the graph. Let $(u,v)$ be a directed edge in $G$ and let $x$ be the color of $(u,v)$. Thus: $x \in S_u$.

No edge $(v,w)$ can have color $x$, so $x \notin S_v$, so $S_u \neq S_v$: neighboring nodes in $G$ must have different „out-edge-color-sets“!

We can use these color sets $S$ to obtain a vertex coloring of $G$: the color of a node $u$ is $S_u$. This coloring must be legal!

As we can have at most $2^c$ subsets (of $c$ vertex colors of $DL(G)$ and hence edge colors of $G$), the coloring has at most $2^c$ colors.

QED
Chromatic Numbers

Chromatic number of $B_{k,n}$?

Recall: Gives lower bound for r-hop coloring algo!
Intuitively: Each time the local view is increased, the chromatic number goes down at most by $\log$!

Lemma

$\chi(B_{1,n}) = n$ and $\chi(B_{k,n}) \geq \log^{(k-1)} n$

Proof idea?

$B_{1,n}$ is the complete graph.
For larger $k$, it holds by induction due to our lemmas!

QED
Finally: Lower Bound

Combining everything gives our lower bound! 🤪

**Lower Bound**

Any deterministic distributed algorithm to color a ring with 3 or less colors needs at least \((\log^* n)/2-1\) rounds.

Proof idea?

We need to show that \(\chi(B_{2r+1,n}) > 3\) for all \(r < (\log^* n)/2-1\).
We know that \(\chi(B_{2r+1,n}) \geq \log^{(2r)} n\).
And \(B_{2r+1,n}\) is subgraph of neighborhood graph we actually want!
The rest is simple maths...

QED
Literature for further reading:

- Peleg’s book (as always 😊 )

End of lecture