Material covered:

Two-party communication lower bound of the equality function:
Follows section 11.3 in “Roger Wattenhofer, notes for Lecture 11 “Hard Problems” (co-authored with Stephan Holzer) of the lecture Principles of Distributed Computing taught at ETH Zurich.

Lower bound for Spanning Tree verification and Approximation of MST that matches the runtime of the exact MST-algorithm from lecture 5:
LAST LECTURE:
- \( \text{MST} \in \tilde{O}(\frac{\text{poly}}{\text{log}^2 n}) \) via \((\text{poly}, \text{poly} \log n)\) fragments

TODAY:
- \( \tilde{\Omega}(\frac{\text{poly}}{\text{log}^2 n}) \) for two-party communication complexity of equality verification
- \( \tilde{\Omega}(\frac{\text{poly}}{\text{log}^2 n}) \) for spanning tree verification
- \( \tilde{\Omega}(\frac{\text{poly}}{\text{log}^2 n}) \) for \( \text{poly}(n) \)-approximation of \( \text{MST} \)
- \( \tilde{\Omega}(\frac{\text{poly}}{\text{log}^2 n}) \) for \( \text{poly}(n) \)-approximation of (weighted) single source shortest paths \( \text{SSSP} \)

RESTATE: CONGEST MODEL / MST

DEF: TWO-PARTY COMMUNICATION COMPLEXITY

\[
\begin{align*}
A & \xleftarrow{1\text{ bit/round}} B \\
\text{INPUT: } X & \in [0,1)^n \\
& = (x_1, \ldots, x_n) \\
\text{OUTPUT: } Y & \in [0,1)^n \\
& = (y_1, \ldots, y_n) \\
S(X,Y) & \in [0,1)^n \\
S : & \rightarrow [0,1)^n \\
CC(S) & = \# \text{ bits exchanged by the best protocol in the worst case,}
\end{align*}
\]

IDEA: PROVE A TWO-PARTY LOWER BOUND
- TRANSFER IT INTO A MORE DISTRIBUTED SETTING.

NOTE: \( \tilde{\Omega}(\cdot) \) AND \( \tilde{O}(\cdot) \) IGNORE \( \text{polylog}(n) \)-FACTORS
**GENERAL FRAMEWORK:**

\[ S(x, y)' \]

\[ C(x) = \text{known} \]

\[ G_x \]

\[ G_y \]

\[ A \]

\[ B \]

- **TAKES \( \Omega (g(2)) \) ROUNDS TO EXCHANGE \( C(S) \) BIT.**
- **GRAPH HAS CERTAIN PROPERTY \( P \) IF \( S(x, y) = 1 \).**

\[ \Rightarrow \] VERIFYING THE PROPERTY TAKES \( \Omega (g(2)) \) ROUNDS.

**EXAMPLE:**

- \( P = "\text{SUBGRAPH } H_{xy} \text{ INDUCED BY MARKED EDGES (RED)} \text{ IS SPANNING TREE OF } G_{xy}" \)
  - \( k = \sqrt{n} \), \( g(\sqrt{n}) = \tilde{\Omega}(\sqrt{n}) \), \( D = O(\log n) \)
  - \( S = EQ \), \( C(S) = \Omega (k) \)

\[ \Rightarrow \] VERIFYING IF \( H_{xy} \) IS ST TAKES \( \tilde{\Omega}(\sqrt{n}) \) (+D)
DEF: EQUALITY VERIFICATION

\[ \text{EQ}(x, y) := \begin{cases} 1 : x = y \\ 0 : x \neq y \end{cases} \]

DEF: MATRIX REPRESENTING \( S \)

\[ M^S_{x,y} := S(x, y) \]

EXAMPLE: FOR EQ IN CASE \( k = 3 \)

\[
\begin{bmatrix}
\text{EQ} & 000 & 001 & 010 & 011 & 100 & 101 & 110 & 111 \\
000 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
001 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
010 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
011 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
100 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
101 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
110 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
111 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{bmatrix}
\]

DEF: \( R \subseteq \mathbb{E}^2 \times \mathbb{E} \) IS (COMBINATORIAL) MONOCHROMATIC RECTANGLE IF

- \((x_1, y_1) \in R\) AND \((x_2, y_2) \in R\) IMPLIES \((x_1, y_2) \in R\)
- THERE IS A FIXED \( z \) S.T. \( S(x, y) = z \) FOR ALL \((x, y) \in R\)

EXAMPLE: \( R_1 := \mathbb{E}^{0.11} \times \mathbb{E}^{0.11} \)
- \( R_2 := \mathbb{E}^{0.11, 100, 101, 110} \times \mathbb{E}^{0.000, 001} \)
- \( R_3 := \mathbb{E}^{0.000, 001, 101} \times \mathbb{E}^{0.11, 100, 110} \)
- \( R_4 := \mathbb{E}^{0.000, 001} \times \mathbb{E}^{0.000, 001} \)

\( R_1, R_2, R_3 \) ARE MONOCHROMATIC, \( R_4 \) IS NOT.
NOTE: • WHEN EXCHANGING A BIT A AND B CAN ELIMINATE THE CORRESPONDING ROWS.
• CAN STOP WHEN A MONOCHROMATIC RECTANGLE IS LEFT
• MAYBE A AND B CAN DO BETTER THAN SENDING THE GIVEN BITS?

DEF: FOOLING SET
$S \subseteq \{0,1\}^d \times \{0,1\}^d$ FOOLS $S$ IF THERE IS A FIXED $\epsilon > 0$ SUCH THAT:
• $S(x,y) = \epsilon$ FOR EACH $(x,y) \in S$
• FOR $(x_1,y_1) \neq (x_2,y_2)$ RECTANGLE $\exists x_1,x_2 \in \{x_1,y_1,y_2\}^d$ IS NOT MONOCHROMATIC
  • $S(x_1,y_2) \neq \epsilon$ (OR AND)
  • $S(x_2,y_1) \neq \epsilon$

EXAMPLE: $S = \{000,000\}, \{001,001\}$ FOOLS EQ.

LEMMA: $S$ FOOLS $S$, THEN $CC(S) = \Omega(\log|S|)$

PROOF: ASSUME ALG ALWAYS NEEDS $\leq \log|S|$ ROUNDS TO COMPUTE $S$

$\Rightarrow |\{0,1\}^d|^t = 2^t \leq |S|$ ACTION PATTERNS (BIT SEQUENCES SENT)

$\Rightarrow \exists (x_1,y_1),(x_2,y_2) \in S$ s.t. ALG PRODUCES SAME PATTERN $P$

$\Rightarrow$ A PRODUCES $P$ ON $(x_1,y_2)$ AND $(x_2,y_1)$ AS WELL

$\Rightarrow$ AFTER $t$ ROUNDS A DOES NOT KNOW IF B'S INPUT WAS $y_1$ OR $y_2$ AND B

WAS IT A'S INPUT $x_1$ OR $x_2$?

BY DEF. OF FOOLING SET:
• $S(x_1,y_2) \neq S(x_1,y_2) \Rightarrow A$ DOES NOT KNOW THE SOLUTION
• $S(x_2,y_1) \neq S(x_1,y_1) \Rightarrow B$ DOES NOT KNOW THE SOLUTION

$\Box$ 4 ASSUMPTION
**Theorem:** \( CC(EQ) = \Omega (2^k) \)

**Proof:**
\[ S = \{ (x, x) \mid x \in \mathbb{E}^k \} \]
fools EQ and has size \( 2^k \)

Apply Lemma I.

---

**Lemma II.** Assume \( x_i + \overline{t_i} \neq 0 \) for all \( 1 \leq i \leq k \). Deciding if \( H_{xy} \) is a spanning tree of \( G_{xy} \) takes \( \tilde{O}(\sqrt{n}) \) rounds in the following construction.

Set \( k' := \sqrt{m} \log n - 1 \)

\( H_{xy} \) consists of all \( k' + 1 \) paths and edges \( e_i \) s.t. \( x_i = 1 \) as well as edges \( e'_i \) if \( \overline{t_i} = 1 \).

**Example for** \( H_{xy} \): 
\( x = 1001, y = 1000 \) (\( k = 4 \))

\[ H_{xy} \subseteq G_{xy} \]

Note that \( H_{xy} \) is not a ST of \( G_{xy} \).
PROOF OF LEMMA: 1) \( H \text{ ALWAYS SPANS } G: \)
\[ x_i + \overline{f_i} \neq 0 \implies \text{THERE IS ALWAYS AN EDGE (ASSUMPTION) } e_i \text{ OR } e_i' \text{ ON AT LEAST ONE SIDE IN } H_{xy}. \]

2) \( H \) IS A TREE IF \( x = y \)
\[ x_i + \overline{f_i} = 1 \text{ AS } x = y \implies \text{ALWAYS EXACTLY ONE EDGE } e_i \text{ OR } e_i' \text{ IN } H_{xy}. \]

3) \( G_x \text{ NEEDS } \Omega(\sqrt{m}) \text{ ROUNDS TO EXCHANGE JUST ONE BIT WITH } G_y. \)

AS WE CAN DECIDE IF \( x = y \) WHEN WE CAN DECIDE IF \( H_{xy} \) IS STEF OF \( G_{xy}. \) (FOLLOWS FROM 1) AND 2)), ONE NEEDS TO EXCHANGE
\[ C(EQ) = \Omega(n) = \Omega(\sqrt{m}) \text{ BIT. } \]

\( \square \)

QUESTION: WHY IS THIS CHEATING? 
- BECAUSE \( D = \sqrt{n}/\log n \)
  - DON'T NEED EQ-LOWER BOUND

- NO USE OF "LIMITED BANDWIDTH" IN THE ARGUMENT, WORKS IN THE LOCAL MODEL AS WELL

SOLUTION 2:
- REDUCE DIAMETER.
**DEF:** An \( i \)-**shortcut** between nodes \( u, v \) in level \( d \) is a \( u,v \)-path of length \( 2(d-i)+1 \). E.g.

**Claim:** An \( i \)-SC covers \( \sqrt{m}/2^{i+1} \) hops at level \( d \).

- All \( i \)-SC cover \( \sqrt{m}/2 \) hops at level \( d \).

**Proof:**

\[
\sum_{i=0}^{d} 2^i \cdot \frac{\sqrt{m}}{2^{i+1}} = \frac{d \cdot \sqrt{m}}{2} \approx \frac{\sqrt{m}}{2} \qquad \Box
\]

**LEM:** Assume nodes can only forward received messages. \( G_x \) and \( G_y \) cannot exchange \( \sqrt{m} \log n \) bit in time \( \leq \sqrt{m}/3 \log n \).
Proof: 

# Paths of length \( \left( \frac{\sqrt{n}}{3 \lg n} \right) - 1 \) that the shortcuts can contribute:

\[
\sum_{L \geq \frac{\sqrt{n}}{2}} \left( \frac{\sqrt{n}}{\lg n} - \frac{\sqrt{n}}{3 \lg n+1} \right)
\]

Normal path length \((\frac{n}{4+1})\) Desired path length

# of edges in normal paths that can be covered.

\[\leq 3 \lg n.\]

Example:

\[\sum \text{Cases like 3+4 cannot happen too often at the same time!}\]

\[\Rightarrow \text{Within time } \frac{\sqrt{n}}{3 \lg n} - 1 \text{ at most}\]

\[\left( \frac{\sqrt{n}}{3 \lg n} - 1 \right) \cdot 3 \lg n \leq \sqrt{n} \]

Messages can be exchanged.

Each message contains \( \lg n \) bit.

(IF WE ASSUME \( \Theta(\lg n) \) BIT WE NEED TO CHOOSE CONSTANTS DIFFERENTLY).
**Note:** In the above argument we even use several tree-edges several times in the same timeslot.

**Question:** How do we remove the assumption $x_i + \overline{y}_i \neq 0$?

**Def:** $x' := x_0 \overline{x} = (x_1, \ldots, x_k, \overline{x}_1, \ldots, \overline{x}_k) \in \{0,1\}^{2k}$ for given $x$

$y' := \overline{y}_0 y = (\overline{y}_1, \ldots, \overline{y}_k, y_1, \ldots, y_k) \in \{0,1\}^{2k}$

**Lem:** Iff $x \equiv y$ there is an $z \in \{0,1\}^k$ s.t. $x'_z = y'_z = 0$

**Proof:** case $x \equiv y$ $\Rightarrow$ \exists $i \in \{1, \ldots, k\}$ s.t. $x_i \equiv y_i$

$\Rightarrow$ Either $x_i = \overline{y}_i = 0$ or $\overline{x}_i = y_i = 0$

$$x'_z = \overline{y}_i, \quad x_i = y_i, \quad \overline{x}_i + \overline{y}_i$$

Case $x \equiv y$ $\Rightarrow$ Always $x_i + \overline{y}_i = 1 = x_i + y_i$ \[\square\]

**Final Construction:**

Extend $H_{xy}$ s.t. a spanning tree of the tree is included and the blocks 182 are connected by one edge.
CLAIM: \( H_{xy} \) IS DISCONNECTED IFF \( \exists i \) s.t. \( x_i = y_i = 0 \)
is tree otherwise.

THIS COMPETES THE PROOF OF LEM.I.

DEF: SPANNING TREE T \( \lambda(n) \)-APPROXIMATES MST T_{MST} IF
\[
\omega(T_{MST}) \leq \omega(T) \leq \lambda(n) \cdot \omega(T_{MST})
\]

THM: \( \lambda(n) \)-APPROXIMATING AN MST TAKES \( \Omega(n + D) \) ROUNDS
E.g. poly(n)

PROOF: ASSIGN WEIGHT 1 (OR 0) TO EDGES IN H.
\[
\omega(H) = \begin{cases} \lambda(n) \cdot n & \text{if } u \text{ NOT in } H, \\ \lambda(n) \cdot n + n - 2 & \text{if } u \text{ is tree, then } H \text{ is MST} \\ \lambda(n) \cdot n + n - 2 & \text{if } u \text{ is not a tree, then } H \text{ is disconnected (by construction of } H) \\ \end{cases}
\]

\( \Rightarrow \) MST HAS WEIGHT \( \lambda(n) \cdot n + n - 2 \)

\( \Rightarrow \) IF AN ALGORITHM COULD \( \lambda(n) \)-APPROXIMATE AN MST IT COULD DECIDE CORRECTLY IF \( H_{xy} \) IS A SPANNING TREE

THM: \( \lambda(n) \)-APPROXIMATING SSSP TAKES TIME \( \Omega(\sqrt{n}) \).

PROOF: USE THE ABOVE WEIGHTS.

\( \Rightarrow \) IF H IS A TREE, THEN ALL SHORTEST PATHS HAVE LENGTH \( \leq n-1 \).

\( \Rightarrow \) IF H IS NOT A TREE, THERE IS AT LEAST ONE SHORTEST PATH OF LENGTH \( \lambda(n) \cdot n \). THIS CAN BE DETECTED IN TIME \( O(1) \).
NOTE: IN THE PROOF WE ASSUMED THAT NODES CAN ONLY FORWARD MESSAGES. USING SIMULATION ARGUMENTS THIS ASSUMPTION CAN BE REMOVED IN OUR CONSTRUCTION.

NOTE: USING THE SET-DISJOINTNESS FUNCTION INSTEAD OF EQ, ONE CAN EXTEND THESE DETERMINISTIC LOWER BOUNDS TO RANDOMIZED LOWER BOUNDS.

NOTE: IT TURNED OUT THAT IN THE CONGEST MODEL VERIFYING CAN BE HARDER THAN COMPUTING!

E.G. SPANNING TREE

* VERIFYING THAT A PROPOSED SOLUTION IS INDEED A SPANNING TREE TAKES $\tilde{O}(\sqrt{m} + d)$

* COMPUTING A SOLUTION (BFS-TREE) TAKES $O(d)$.

COMPARE THIS TO CLASSIC COMPUTING THEORY; E.G. SAT (SATISFIABILITY)

* VERIFYING AN ASSIGNMENT FOR SAT IS IN P

* COMPUTING A SOLUTION FOR SAT IS $\text{NP}$

* COMPUTING MIGHT BE HARDER THAN VERIFYING!