236610
Distributed Graph Algorithms

Spring 2018
Keren Censor-Hillel
MST in CONGEST

- **BFS-based** algorithm in $O(n)$ rounds
- **GHS algorithm** in $O(n\log n)$ rounds
- **GKP algorithm** in $O(\sqrt{n}\log^*n + D)$ rounds

**Question:** $D$ is necessary. What about $\sqrt{n}$?
- We will see that $\sqrt{n}$ is also a lower bound
- But first, a simpler lower bound proof
Computing the Diameter

• A BFS tree gives a $2$-approximation to the diameter
  
  $D = \max\{d(u,v) \mid u,v \text{ in } V\}$

• Can be computed in $O(D)$ rounds in CONGEST

• What about exact diameter?
• Better approximation factors?
Diameter in CONGEST

• Exact diameter can be computed in $O(n)$ rounds
  – Even APSP (all-pairs-shortest-paths)

• Any algorithm for computing the exact diameter requires $\Omega(n/\log n)$ rounds
  – Even when $D$ is small
Approximating the Diameter in CONGEST

• A $(3/2-\varepsilon)$-approximation of $D$ requires $\Omega(n/\log^3 n)$ rounds

• A $3/2$-approximation of $D$ can be computed in $O((n/\log n)^{1/2}+D)$ rounds

• A threshold at $3/2$
Diameter in CONGEST

• **Theorem**: Any algorithm for computing the exact diameter requires $\Omega(n/\log n)$ rounds

• **Proof**: Reduction from 2-party Set Disjointness.

• **Reminder**: If we reduce solving A to solving B, and we show that solving A is hard, then solving B is hard.
2-party communication

• Two players, Alice and Bob

• Inputs: $x^A = (x^A_1, \ldots, x^A_k)$, $x^B = (x^B_1, \ldots, x^B_k)$ in $\{0,1\}^k$

• Players exchange bits according to a protocol $\pi$

• Outputs: $y^A$, $y^B$ in $\{0,1\}$
2-party communication

• At the beginning and after each bit that is sent in the protocol $\pi$:
  – Both players know the bit
  – Both players know whether the protocol is finished (and their output) or who sends the next bit

• The sequence of sent bits in $\pi$ is the transcript
2-party communication

• The \textit{communication complexity} of a protocol $CC(\pi)$: Total number of bits sent in the transcript

• The \textit{communication complexity} of a problem: The minimal complexity over all protocols that solve the problem
2-party communication

• **Set Disjointness:**

• **Inputs:**

\[ x^A = (x^A_1, ..., x^A_k), \quad x^B = (x^B_1, ..., x^B_k) \text{ in } \{0,1\}^k \]

• **Outputs:**

\[ y^A = y^B = 1 \text{ if and only if } \]

there is an \( i \in \{1, ..., k\} \) such that \( x^A_i = x^B_i = 1 \)
Lower Bound for Set Disjointness

• **Theorem**: The communication complexity of set disjointness is $k+1$

• **Proof**:
  • There is a protocol that solves set disjointness with $\text{CC}(\pi) = k+1$
    – Alice sends all of her input bits to Bob ($k$ bits)
    – Bob sends Alice the result ($1$ bit)
Lower Bound for Set Disjointness

- **Theorem**: The communication complexity of **set disjointness** is $k+1$

- **Proof**:
  - Any protocol $\pi$ that solves set disjointness has $\text{CC}(\pi)=k+1$
    - Later if we have time
Computing D – Base Graph $G_{\text{base}}$
Computing D

Clique A\(^1\)  Clique B\(^1\)

Clique A\(^2\)  Clique B\(^2\)

a  b
Computing D

Clique $A^1$  Clique $B^1$

Clique $A^2$  Clique $B^2$

a  b
Computing $D$

Clique $A^1$  

Clique $B^1$

Clique $A^2$  

Clique $B^2$
Input-Based Graph

- Given an input \( x^A = (x^A_1, \ldots, x^A_k) \), \( x^B = (x^B_1, \ldots, x^B_k) \) to Set Disjointness, define \( G \) as follows:
  - \( G \) contains all of \( G_{base} \)
  - For \( (i,j)=1,\ldots,k \), the spike from \( A^1_i \) to \( A^2_j \) is in \( G \) if and only if \( x^{A}_{i,j}=0 \)
    - \( (i,j)=(i-1)\sqrt{k}+j \)
  - For \( (i,j)=1,\ldots,k \), the spike from \( B^1_i \) to \( B^2_j \) is in \( G \) if and only if \( x^{B}_{i,j}=0 \)
Computing D – Input-Based Graph

Clique A¹     Clique B¹

Clique A²     Clique B²

a       b
Computing D – Input-Based Graph

Clique A

Clique B

Clique A'

Clique B'
Computing D – Input-Based Graph

Clique A¹

Clique B¹

Clique A²

Clique B²

a

b
Computing $D$

• **Claim 1**: If the inputs are disjoint then $D(G)=2$. Otherwise, $D(G)=3$.

• **Proof**: By case analysis.
Diameter in CONGEST

• **Theorem**: Any algorithm for computing the exact diameter requires $\Omega(n/\log n)$ rounds


• **Illustration**
Computing D

Clique $A^1$  Clique $B^1$

Clique $A^2$  Clique $B^2$

a  b
Diameter in CONGEST

• **Theorem**: Any algorithm for computing the exact diameter requires $\Omega(n / \log n)$ rounds.

• **Proof** (cont.):

At most $O(n \log n)$ bits can be sent over the cut in a round. But $k = \Theta(n^2)$, so the number of rounds is $\Omega(n^2 / n \log n) = \Omega(n / \log n)$. 

Keren Censor-Hillel, Spring 2018
(3/2-\(\varepsilon\))-Approximation of D

- **Theorem**: Any algorithm for computing a (3/2-\(\varepsilon\))-approximation of the diameter requires \(\Omega(n^{1/2}/\log n)\) rounds

- **Illustration**
(3/2-\(\varepsilon\))-Approximation of D
(3/2-\(\epsilon\))-Approximation of D

- **Theorem**: Any algorithm for computing a (3/2-\(\epsilon\))-approximation of the diameter requires \(\Omega(n^{1/2}/\log n)\) rounds.

- **Proof**: Reduction from Set Disjointness. Alice simulates \(a\) and nodes in \(A^1, A^2\) and \(A^3\), Bob simulates \(b\) and nodes in \(B^1, B^2\) and \(B^3\).
(3/2-\(\varepsilon\))-Approximation of D

- **Theorem**: Any algorithm for computing a (3/2-\(\varepsilon\))-approximation of the diameter requires \(\Omega(n^{1/2}/\log n)\) rounds.

- **Proof**: At most \(O(n^{1/2}\log n)\) bits can be sent over the cut in a round. But now \(k=\Theta(n)\), so the number of rounds is \(\Omega(n/n^{1/2}\log n)=\Omega(n^{1/2}/\log n)\).

Bounds by [Frischknecht, Holzer, Wattenhofer]
MST in CONGEST

• **BFS-based** algorithm in $O(n)$ rounds

• **GHS algorithm** in $O(n \log n)$ rounds

• **GKP algorithm** in $O(\sqrt{n \log^* n} + D)$ rounds

• **Question:** $D$ is necessary. What about $\sqrt{n}$?
Lower Bound for MST in CONGEST

- **Theorem**: Any MST algorithm in CONGEST needs $\Omega(\sqrt{n/\log n})$ rounds.

- **Main tool**: A reduction from the problem of 2-party set-disjointness

Lower bound by Das-Sarma et al.
Lower Bound for MST in CONGEST

- **First step**: A reduction from the problem of verification of a connected spanning subgraph to MST

- **Input** for node \( v \): for every \( \{u,v\} \in E \), is \( \{u,v\} \in H? \)

- **Output** for \( v \): is \( H \) a connected spanning graph of \( G \)
  - all-accept-all-reject
  - all-accept-one-reject (\( D \) rounds give all-reject)
Reduction from the verification of a connected spanning subgraph to MST

• Lemma (reduction):

If $A$ is an algorithm for $\text{MST}$ in CONGEST which completes in $O(T)$ rounds, then

there is an algorithm $A'$ for the verification of a connected spanning subgraph in CONGEST which completes in $O(T+D)$ rounds.
Reduction from the verification of a connected spanning subgraph to MST

• **Proof:**

• The nodes assign the following weights to each edge $e = \{u, v\}$ in $E$:
  
  – If $e \in H$, assign $w(e) \leftarrow 0$
  – Otherwise, assign $w(e) \leftarrow 1$
Reduction from the verification of a connected spanning subgraph to MST

- **Proof:**
  - If \( e \in H \), assign \( w(e) \leftarrow 0 \)
  - Otherwise, assign \( w(e) \leftarrow 1 \)

- \( w(MST) = 0 \) if and only if \( H \) is a connected spanning subgraph

- Also, each node can locally check whether the MST contains one of its edges that is not in \( H \).
Lower bound for the verification of a connected spanning subgraph

• **Theorem**: Any algorithm for the verification of a connected spanning subgraph in CONGEST needs $\Omega(\sqrt{n/\log n})$ rounds.

• This implies the same lower bound for the construction of an MST, by the Reduction Lemma.

• **Proof**: By a reduction from the problem of 2-party set-disjointness
Lower bound for the verification of a connected spanning subgraph

• **Theorem:** Any algorithm for the verification of a connected spanning subgraph in CONGEST needs $\Omega(vn/\log n)$ rounds.

• This implies the same lower bound for the construction of an MST, by the Reduction Lemma.
Base Graph

• Illustration

• The number of nodes is $n = \Theta(kd^q)$
• The diameter is $D = 2q + 2$
Base Graph

A = \( u^q_0 \) to \( v^1_0 \) to \( \ldots \) to \( v^{k-1}_0 \)

B = \( u^q_{dq-1} \) to \( v^0_{dq-1} \) to \( v^1_{dq-1} \) to \( \ldots \) to \( v^{k-1}_{dq-1} \)

spikes from \( u^q_i \) to \( v^j_i \)

complete \( d \)-ary tree of depth \( q \)

\( k \) paths of length \( d^q \)
Input-Based Graph

- Given an input $x^A = (x^A_1, \ldots, x^A_k), x^B = (x^B_1, \ldots, x^B_k)$ to Set Disjointness, define $H$ in $G$ as follows:
  - All paths are in $H$
  - All tree edges are in $H$
  - All spikes from $u^{q_i}$ for $i=1, \ldots, d^q-2$ are not in $H$
  - For $j=1, \ldots, k$, the spike from $u^{q_0}$ to $v^j_0$ is in $H$ if and only if $x^A_j = 0$
  - For $j=1, \ldots, k$, the spike from $u^{q_{d^q-1}}$ to $v^j_{d^q-1}$ is in $H$ if and only if $x^B_j = 0$
Input-Based Graph

A complete $d$-ary tree of depth $q$

spikes from $u^q_i$ to $v^j_i$

$k$ paths of length $d^q$
Is H Connected?

• **Claim 1:** $H$ is not connected if and only if there is a $j=1,...,k$ such that $x^A_j = x^B_j = 1$

• **Proof:** If $x^A_j = x^B_j = 1$ then both edges $(u_q^0, v^j_0)$ and $(u_{dq-1}^q, v^j_{dq-1})$ are not in $H$, and hence the $j$-th path is a connected component of $H$. Otherwise, $(u_q^0, v^j_0)$ or $(u_{dq-1}^q, v^j_{dq-1})$ is in $H$ for all $j$, and all paths are connected to the tree.
Claim 2: If $\text{ALG}$ is a distributed algorithm for verifying a connected spanning subgraph in $R$ rounds with $R \leq (d^q - 1)/2$ then Alice and Bob can simulate $\text{ALG}$ by exchanging $O(d^q R \log n)$ bits.
Lower bound for the verification of a connected spanning subgraph

• **Theorem:** Any algorithm for *the verification of a connected spanning subgraph* in CONGEST needs $\Omega(\sqrt{n/\log n})$ rounds.

• **Proof:** By **Claim 1**, if A and B simulate **ALG** then they solve Set Disjointness. Since the CC of Set Disjointness is $\Omega(k)$, by **Claim 2**, $R=\Omega(\min(d^q,k/dq\log n))$
Lower bound for the verification of a connected spanning subgraph

- $R = \Omega(\min(d^q, k/dq\log n))$
- Now choose $k = d^{q+1}q\log n$
- Then $R = \Omega(d^q)$

- Recall $n = \Theta(kd^q)$, then $n = \Theta(d^{2q+1}q\log n)$
- Then $R = \Omega(d^q) = \Omega((n/q\log n)^{q/(2q+1)})$
  $= \Omega((n/q\log n)^{1/2-1/(2(2q+1))})$

- Now choose $q = \log n$, then $R = \Omega(\sqrt{n}/\log n)$
Simulation

• **Claim 2**: If ALG is a distributed algorithm for verifying a connected spanning subgraph in $R$ rounds with $R \leq (d^q - 1)/2$ then Alice and Bob can **simulate ALG** by exchanging $O(dqR \log n)$ bits

• **Proof**: We show that after exchanging $O(dqR \log n)$ bits, Alice knows the local state of $A = u^q_{q_0}$ after $R$ rounds of ALG, and Bob knows the local state of $B = u^q_{dq-1}$ after $R$ rounds of ALG.
Simulation

- \( T(S) = \) tree nodes with descendants in \( S \)

- \( L_0 = V \setminus \{B\} \)
- \( L'_i = \{u^q_j \mid j \leq d^q-1-i\} \)
- \( L_i = \bigcup_{1 \leq t \leq k} \{v^t_j \mid j \leq d^q-1-i\} \cup L'_i \cup T(L'_i) \)

- \( R_0 = V \setminus \{A\} \)
- \( R'_i = \{u^q_j \mid j \geq i\} \)
- \( R_i = \bigcup_{1 \leq t \leq k} \{v^t_j \mid j \geq i\} \cup R'_i \cup T(R'_i) \)
Input-Based Graph

A = $u^q_0$

$A = \{v^0_0, v^1_0, \ldots, v^{k-1}_0\}$

$B = u^q_{dq-1}$

$B = \{v^0_{dq-1}, v^1_{dq-1}, \ldots, v^{k-1}_{dq-1}\}$

spikes from $u^q_i$ to $v^j_i$

$k$ paths of length $d^q$

complete $d$-ary tree of depth $q$
There are $k$ paths of length $d^q$ from $u_{q_0}^0$ to $v_{j_i}^{i_0}$.

A complete $d$-ary tree of depth $q$ is shown with $L_0$.
spikes from $u^q_i$ to $v^j_i$

$k$ paths of length $d^q$

complete $d$-ary tree of depth $q$

$A = u^q_0$

$B = u^q_{dq-1}$

$R_0$
k paths of length $d^q$

spikes from $u^q_i$ to $v^j_i$

complete $d$-ary tree of depth $q$
The diagram illustrates a complete $d$-ary tree of depth $q$. The tree consists of $k$ paths of length $d^q$. Spikes from $u_{q_i}^i$ to $v_{j_i}^i$ are shown.
spikes from $u^q_i$ to $v^j_i$

$k$ paths of length $d^q$

complete $d$-ary tree of depth $q$

$A = u^q_0$

$B = u^q_{dq-1}$

$R_3$
The state of B after t rounds

• To know the state of nodes in $R_t$ after $t$ rounds, it is enough to know the state of nodes in $N(R_t)$ after $t-1$ rounds.

• To know the state of nodes in $R_t$ after $t$ rounds, it is enough to know the state of nodes in $R_{t-1}$ after $t-1$ rounds and the messages sent from $V \setminus R_{t-1}$ to $R_t$ in round $t$. 

Keren Censor-Hillel, Spring 2018
The state of $B$ after $t$ rounds

• By induction on $t$, there are at most $dq$ messages from $V\backslash R_{t-1}$ to $R_t$ in round $t$, and at most $dq$ messages from $V\backslash L_{t-1}$ to $L_t$ in round $t$

1. All are on into the leftmost/rightmost tree nodes in $R_t/L_t$ at every level

2. $V\backslash R_{t-1}$ is a subset of $L_{t-1}$

3. $V\backslash L_{t-1}$ is a subset of $R_{t-1}$

• In total, $O(dqR\log n)$ bits

• Items 2 and 3 hold only for $R \leq (d^a-1)/2$
Lower Bound for MST in CONGEST

- **Theorem**: Any MST algorithm in CONGEST needs $\Omega(\sqrt{n}/\log n)$ rounds.
Local/Global Problems

• We need $\Omega(D)$ rounds for BFS, MST

• Do we need $\Omega(D)$ rounds for all interesting problems?

• No:
  – BFS, MST are global problems
  – some problems are local problems