The NOF Communication Complexity of Multi-Party Pointer Jumping

Joshua Brody

Dartmouth College
Hanover, NH USA

IAS Computer Science/Discrete Math Seminar
December 7, 2009
Talk Outline

- Multi-Party Communication Games
- The Multi-Party Pointer Jumping Problem
- Upper Bounds
- Restricted Protocols
- Conclusions
Multi-party Communication Games
Multi-Party Communication Games

Input $x = (x_1, \ldots, x_k)$ is split between $k$ players.

Goal: minimize communication needed to compute $f(x)$.

Our model of communication:

- Player $i$ sees every input except $x_i$ (NOF model).
- One-way communication: each player speaks once and in order.
- Blackboard communication: all players see every message sent.
Pointer Jumping

Vertices:
- \( k - 1 \) layers, plus start vertex
- layers have \( n \) vertices

Compute \( mpj_k \), the bit reached by following pointers from the start vertex.
Pointer Jumping

Vertices:
- \( k - 1 \) layers, plus start vertex
- layers have \( n \) vertices

Input:
- \( k - 1 \) layers of pointers
- \( n \) bit string

Compute \( \text{mpj}^k \) = bit reached by following pointers from start vertex.
Pointer Jumping

Vertices:
- \( k - 1 \) layers, plus start vertex
- layers have \( n \) vertices

Input:
- \( k - 1 \) layers of pointers
- \( n \) bit string

Compute \( MPJ_k = \) bit reached by following pointers from start vertex.
Pointer Jumping: non-Boolean version

Vertices:
- $k$ layers, plus start vertex
- layers have $n$ vertices

Input:
- $k$ layers of pointers

Compute $\widehat{\text{MPJ}}_k = \text{vertex}$ reached by following pointers from start vertex.
Layers of Edges are Functions

Formal Definition:

- Inputs:
  - \( i \in \{0, 1\}^n \)
  - \( f_2, \ldots, f_{k-1} : \{0, 1\}^n \rightarrow \{0, 1\}^n \)
- Output:
  - \( mpj_k : x \leftarrow f_{k-1} \circ \cdots \circ f_2(i) \)
Layers of Edges are Functions

Formal Definition:

Inputs:
- \(i \in [n]\)
- \(f_2, \ldots, f_{k-1} : [n] \rightarrow [n]\)
- \(x \in \{0, 1\}^n\)

Output:
- \(\text{MPJ}_k := x[f_{k-1} \circ \cdots \circ f_2(i)]\)
Pointer Jumping: Trivial Bounds

• One-way: any order except $P_1, P_2, \ldots, P_k$: $O(\log n)$

• One way: in the order $P_1, P_2, \ldots, P_k$: $O(n)$
Pointer Jumping: Trivial Bounds

- One-way: any order except $P_1, P_2, \ldots, P_k$: $O(\log n)$
- One way: in the order $P_1, P_2, \ldots, P_k$: $O(n)$
  - Problem seems hard. Maybe $n^{\Omega(1)}$ lower bound?
Motivation

\(\text{ACC}^0\) complexity class: \(\text{AC}^0\) plus \(\text{MOD}_m\) gates.

- No function \(f \not\in \text{ACC}^0\) is known.
- If \(f : \{0, 1\}^n \rightarrow \{0, 1\}\) and \(f \in \text{ACC}^0\), then \(f\) has deterministic NOF protocol with \(\text{poly}(\log n)\) communication, for \(k = \text{poly}(\log n)\) players.

\[\text{[Yao'90], [Håstad-Goldmann'91], [Beigel-Tarui'94]}\]
ACC⁰ complexity class: AC⁰ plus MODₘ gates.

- No function \( f \not\in \text{ACC}^0 \) is known.
- If \( f : \{0,1\}^n \rightarrow \{0,1\} \) and \( f \in \text{ACC}^0 \), then \( f \) has deterministic NOF protocol with \( \text{poly}(\log n) \) communication, for \( k = \text{poly}(\log n) \) players.

[Yao'90], [Håstad-Goldmann'91], [Beigel-Tarui'94]

Recently pointer jumping has been used to prove lower bounds in:

- threshold circuits [Razborov-Wigderson'93]
- proof size [Beame-Pitassi-Segerlind'05]
- matroid intersection queries [Harvey'08]
- randomly-ordered data streams [Chakrabarti-Cormode-McGregor'08]
Previous Result Highlights

Far from proving $\text{MPJ}_{\text{poly}(\log n)} \not\in \text{ACC}^0$

- $\Omega(\sqrt{n})$ for $\text{MPJ}_3$ [Wigderson’97]
- $\Omega(n^{1/(k-1)} / k^k)$ for $\text{MPJ}_k$ [Viola-Wigderson’07]
- lower bounds for restricted protocols
Far from proving $\text{MPJ}_{\text{poly}(\log n)} \notin \text{ACC}^0$

- $\Omega(\sqrt{n})$ for $\text{MPJ}_3$ [Wigderson’97]
- $\Omega\left(n^{1/(k-1)/k^k}\right)$ for $\text{MPJ}_k$ [Viola-Wigderson’07]
- lower bounds for restricted protocols
- $O\left(n \log^{(k-1)} n\right)$ for $\text{MPJ}_k$ [Damm-Jukna-Sgall’96]
- $O\left(n \frac{\log \log n}{\log n}\right)$ for $\text{MPJ}_3$ when middle layer is a permutation. [Pudlák-Rödl-Sgall ’97]
Previous Result Highlights

Far from proving $\text{MPJ}_{\text{poly}(\log n)} \not\in \text{ACC}^0$

- $\Omega(\sqrt{n})$ for $\text{MPJ}_3$  
  [Wigderson’97]
- $\Omega(n^{1/(k-1)/k^k})$ for $\text{MPJ}_k$  
  [Viola-Wigderson’07]
- lower bounds for restricted protocols
- $O\left(n \log^{(k-1)} n\right)$ for $\text{MPJ}_k$  
  [Damm-Jukna-Sgall’96]
- $O\left(n \frac{\log \log n}{\log n}\right)$ for $\text{MPJ}_3$ when middle layer is a permutation.  
  [Pudlák-Rödl-Sgall ’97]

Our Results

- $O\left(n \sqrt{\frac{\log \log n}{\log n}}\right)$ for $\text{MPJ}_3$  
  [B.-Chakrabarti’08]
- bounds for restricted protocols  
  [B.’09]
Previous Result Highlights

Far from proving $\text{MPJ}_{\text{poly}(\log n)} \not\in \text{ACC}^0$

- $\Omega(\sqrt{n})$ for $\text{MPJ}_3$  [Wigderson’97]
- $\Omega(n^{1/(k-1)/k^k})$ for $\text{MPJ}_k$  [Viola-Wigderson’07]
- lower bounds for restricted protocols (2nd half of talk)
- $O\left(n \log^{(k-1)} n\right)$ for $\overline{\text{MPJ}}_k$  [Damm-Jukna-Sgall’96]
- $O\left(n \frac{\log \log n}{\log n}\right)$ for $\text{MPJ}_3$ when middle layer is a permutation.  [Pudlák-Rödl-Sgall ’97]

Our Results

- $O\left(n \sqrt{\frac{\log \log n}{\log n}}\right)$ for $\text{MPJ}_3$  [B.-Chakrabarti’08]
- bounds for restricted protocols (2nd half of talk)  [B.’09]
Talk Outline

- Multi-Party Communication Games
- The Multi-Party Pointer Jumping Problem
- Upper Bounds
- Restricted Protocols
- Conclusions
The Damm-Jukna-Sgall Protocol

3 players:

$P_1$ sends $\log \log n$ bits of $f_2(i)$ for each $i \Rightarrow n \log \log n$ bits.

$P_2$ sends $f_3(j)$ for each possible $j \Rightarrow n^{2 \log \log n} = n$ bits.

$P_3$ outputs $f_3(f_2(i))$.

$P_1 P_3 P_2$

$k$ players:

$P_1$ sends $\log (k - 1)n$ bits for each pointer.

$P_2$ sends $\log (k - 2)n$ bits for each of $n/\log (k - 2)n$ possible pointers.

...$

Total communication: $O(n \log (k - 1)n)$ bits.
The Damm-Jukna-Sgall Protocol

3 players:

- \( P_1 \) sends \( \log \log n \) bits of \( f_2(i) \) for each \( i \)

\[ P_1 \rightarrow P_3 \]

\[ P_2 \]

\[ 1 \rightarrow 2 \rightarrow 3 \rightarrow n \]
The Damm-Jukna-Sgall Protocol

3 players:

- \( P_1 \) sends \( \log \log n \) bits of \( f_2(i) \) for each \( i \)
- \( P_2 \) sends \( f_3(j) \) for each possible \( j \)

Total communication:

\[ O(n \log(n)) \] bits.
The Damm-Jukna-Sgall Protocol

3 players:

- $P_1$ sends $\log \log n$ bits of $f_2(i)$ for each $i$
- $P_2$ sends $f_3(j)$ for each possible $j$
- $P_3$ outputs $f_3(f_2(i))$.
3 players:

- $P_1$ sends $\log \log n$ bits of $f_2(i)$ for each $i$ \Rightarrow $n \log \log n$ bits.
- $P_2$ sends $f_3(j)$ for each possible $j$ \Rightarrow $\frac{n}{2 \log \log n} \log n = n$ bits.
- $P_3$ outputs $f_3(f_2(i))$. 

The Damm-Jukna-Sgall Protocol

Total communication: $O(n \log \log (k - 1))$ bits.
The Damm-Jukna-Sgall Protocol

3 players:
- $P_1$ sends $\log \log n$ bits of $f_2(i)$ for each $i$ $\Rightarrow n \log \log n$ bits.
- $P_2$ sends $f_3(j)$ for each possible $j$ $\Rightarrow \frac{n}{2^{\log \log n}} \log n = n$ bits.
- $P_3$ outputs $f_3(f_2(i))$.

$k$ players:
- $P_1$ sends $\log^{(k-1)} n$ bits for each pointer.
- $P_2$ sends $\log^{(k-2)} n$ bits for each of $n / \log^{(k-2)} n$ possible pointers.

...
The Damm-Jukna-Sgall Protocol

3 players:
- $P_1$ sends $\log \log n$ bits of $f_2(i)$ for each $i$ ⇒ $n \log \log n$ bits.
- $P_2$ sends $f_3(j)$ for each possible $j$ ⇒ $\frac{n}{2\log \log n} \log n = n$ bits.
- $P_3$ outputs $f_3(f_2(i))$.

$k$ players:
- $P_1$ sends $\log^{(k-1)} n$ bits for each pointer.
- $P_2$ sends $\log^{(k-2)} n$ bits for each of $n/\log^{(k-2)} n$ possible pointers.

... Total communication: $O(n \log^{(k-1)} n)$ bits.
The Pudlák-Rödl-Sgall Protocol

**Step 0:** Generate random bipartite graph $H$
The Pudlák-Rödl-Sgall Protocol

**Step 0:** Generate random bipartite graph $H$
The Pudlák-Rödl-Sgall Protocol

Step 0: Generate random bipartite graph $H$

Step 1: $P1$ sees $\pi$, knows $H$

- creates graph $G_{\pi}$ on vertices in second layer
- $(a, b) \in E$ iff $(y_{\pi^{-1}(a)}, x_b)$ and $(y_{\pi^{-1}(b)}, x_a)$ in $H$
The Pudlák-Rödl-Sgall Protocol

**Step 0:** Generate random bipartite graph $H$

**Step 1:** $P_1$ sees $\pi$, knows $H$

- creates graph $G_\pi$ on vertices in second layer
- $(a, b) \in E$ iff $(y_{\pi^{-1}(a)}, x_b)$ and $(y_{\pi^{-1}(b)}, x_a)$ in $H$
**The Pudlák-Rödl-Sgall Protocol**

**Step 0:** Generate random bipartite graph $H$

**Step 1:** $P_1$ sees $\pi$, knows $H$

- creates graph $G_\pi$ on vertices in second layer
- $(a, b) \in E$ iff $(y_{\pi^{-1}(a)}, x_b)$ and $(y_{\pi^{-1}(b)}, x_a)$ in $H$
- Let $C_1, \ldots, C_r$ be a clique cover of $G_\pi$
- For each $1 \leq i \leq r$, $P_1$ sends parity of bits in $C_i$
The Pudlák-Rödl-Sgall Protocol

Step 2: $P_2$ sees $i$, knows $H$

- sends $x_j$ for each $(y_i, x_j) \in H$
The Pudlák-Rödl-Sgall Protocol

Step 2: $P_2$ sees $i$, knows $H$
- sends $x_j$ for each $(y_i, x_j) \in H$
The Pudlák-Rödl-Sgall Protocol

Step 2: $P_2$ sees $i$, knows $H$
- sends $x_j$ for each $(y_i, x_j) \in H$

Step 3: $P_3$ sees $i, \pi$, knows $H$
- $C :=$ clique containing $\pi(i)$
The Pudlák-Rödl-Sgall Protocol

Step 2: P2 sees $i$, knows $H$
- sends $x_j$ for each $(y_i, x_j) \in H$

Step 3: P3 sees $i, \pi$, knows $H$
- $C :=$ clique containing $\pi(i)$
- Note: $j \neq \pi(i) \in C \Rightarrow (j, \pi(i)) \in G_{\pi}$
The Pudlák-Rödl-Sgall Protocol

Step 2: $P_2$ sees $i$, knows $H$
- sends $x_j$ for each $(y_i, x_j) \in H$

Step 3: $P_3$ sees $i, \pi$, knows $H$
- $C :=$ clique containing $\pi(i)$
- Note: $j \neq \pi(i) \in C \Rightarrow (j, \pi(i)) \in G_\pi$
- $\therefore (y_i, x_j) \in H \Rightarrow P_2$ sent $x_j$. 
The Pudlák-Rödl-Sgall Protocol

**Step 2:** $P_2$ sees $i$, knows $H$
- sends $x_j$ for each $(y_i, x_j) \in H$

**Step 3:** $P_3$ sees $i, \pi$, knows $H$
- $C :=$ clique containing $\pi(i)$
- Note: $j \neq \pi(i) \in C \Rightarrow (j, \pi(i)) \in G_\pi$
- $\therefore (y_i, x_j) \in H \Rightarrow P_2$ sent $x_j$.
- $P_3$ takes clique bit, XORs out all $x_j \neq x_{\pi(i)}$, recovers $x_{\pi(i)}$. 
Lemma: [PRS'96], [Bollobás'88]

There exists a bipartite graph $H$ such that for all $i, \pi$

1. $G_\pi$ has $O\left(n \frac{\log \log n}{\log n}\right)$ cliques

2. $y_i$ has outdegree $O\left(n \frac{\log \log n}{\log n}\right)$
A General Protocol: 3 players

Idea: - Run PRS several times in parallel.
A General Protocol: 3 players

Idea:

- Run \( \text{PRS} \) several times in parallel.

- Pick permutations \( \pi_1, \pi_2, \ldots, \pi_d \) such that \( f(i) = \pi_j(i) \) for some permutation.
A General Protocol: 3 players

Idea:  
- Run PRS several times in parallel.
- Pick permutations \( \pi_1, \pi_2, \ldots, \pi_d \) such that \( f(i) = \pi_j(i) \) for some permutation.
A General Protocol: 3 players

Idea: - Run PRS several times in parallel.
- Pick permutations $\pi_1, \pi_2, \ldots, \pi_d$ such that $f(i) = \pi_j(i)$ for some permutation.
A General Protocol: 3 players

<table>
<thead>
<tr>
<th>P1</th>
<th>P2</th>
<th>P3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Idea:  
- Run PRS several times in parallel.
- Pick permutations $\pi_1, \pi_2, \ldots, \pi_d$ such that $f(i) = \pi_j(i)$ for some permutation.
A General Protocol: 3 players

Idea:  
- Run PRS several times in parallel.
- Pick permutations $\pi_1, \pi_2, \ldots, \pi_d$ such that $f(i) = \pi_j(i)$ for some permutation.
- $P3$ determines which permutation matches $f(i)$.
A General Protocol: 3 players

Idea:  - Run PRS several times in parallel.
      - Pick permutations $\pi_1, \pi_2, \ldots, \pi_d$ such that $f(i) = \pi_j(i)$ for some permutation.
      - $P3$ determines which permutation matches $f(i)$.

It turns out we can’t do this efficiently, but we can get close enough.
Technical Details

Definition: A set of permutations $A \subseteq S_n$ $d$-covers $f$ if for all $i \in [n]$, one of the following conditions holds:

- There exists $\pi \in A$ such that $\pi(i) = f(i)$.
- $f(i)$ has a large preimage: $|f^{-1}(f(i))| > d$. 
Technical Details

Definition: A set of permutations \( A \subseteq S_n \) \( d \)-covers \( f \) if for all \( i \in [n] \), one of the following conditions holds:

- There exists \( \pi \in A \) such that \( \pi(i) = f(i) \).
- \( f(i) \) has a large preimage: \( |f^{-1}(f(i))| > d \).

Lemma: We can always find a set of \( d \) permutations that \( d \)-covers \( f \).
Technical Details

**Definition:** A set of permutations $A \subseteq S_n$ $d$-covers $f$ if for all $i \in [n]$, one of the following conditions holds:

- There exists $\pi \in A$ such that $\pi(i) = f(i)$.
- $f(i)$ has a large preimage: $|f^{-1}(f(i))| > d$.

**Lemma:** We can always find a set of $d$ permutations that $d$-covers $f$.

**Note:** There can be at most $n/d$ points with large preimages.
A General Protocol: 3 players

Players agree on $d$ and a $d$-covering set $A_d(f)$ for each $f$. 
A General Protocol: 3 players

Players agree on $d$ and a $d$-covering set $A_d(f)$ for each $f$.

- $P1$ sends $\{\alpha(\pi, x)\}_{\pi \in A_d(f)}$.
- $P1$ also sends $x[j]$ for any $j$ with a large preimage.
A General Protocol: 3 players

Players agree on $d$ and a $d$-covering set $A_d(f)$ for each $f$.

- $P1$ sends $\{\alpha(\pi, x)\}_{\pi \in A_d(f)}$.
- $P1$ also sends $x[j]$ for any $j$ with a large preimage.
- $P2$ sends $\{\beta(i, x, \alpha)\}_\alpha$. 

With $d = \sqrt{\log n \log \log n}$, the protocol costs $O(n \sqrt{\log \log n \log n})$. 

Joshua Brody
A General Protocol: 3 players

Players agree on $d$ and a $d$-covering set $A_d(f)$ for each $f$.

- $P_1$ sends $\{\alpha(\pi, x)\}_{\pi \in A_d(f)}$.
- $P_1$ also sends $x[j]$ for any $j$ with a large preimage.
- $P_2$ sends $\{\beta(i, x, \alpha)\}_\alpha$.
- $P_3$ recovers $x[f(i)]$ from PRS or from $P_1$’s extra bits.
A General Protocol: 3 players

Players agree on $d$ and a $d$-covering set $A_d(f)$ for each $f$.

- $P1$ sends $\{\alpha(\pi, x)\}_{\pi \in A_d(f)}$.
- $P1$ also sends $x[j]$ for any $j$ with a large preimage.
- $P2$ sends $\{\beta(i, x, \alpha)\}_\alpha$.
- $P3$ recovers $x[f(i)]$ from PRS or from $P1$’s extra bits.

With $d = \sqrt{\frac{\log n}{\log \log n}}$, the protocol costs $O \left(n \sqrt{\frac{\log \log n}{\log n}} \right)$. 
Talk Outline

• Multi-Party Communication Games

• The Multi-Party Pointer Jumping Problem

• Upper Bounds

• Restricted Protocols

• Conclusions
Restricted Protocols

Partial progress: protocols with more restricted forms of information sharing

- **Myopic protocols**: $P_j$ only sees layers $1, \ldots, (j - 1)$ as well as layer $(j + 1)$ of graph. (i.e., limited visibility of layers ahead)

  [Gronemeier’06]

- **Conservative protocols**: $P_j$ sees layers $(j + 1), \ldots, k$ of graph, plus composition of layers $1, \ldots, (j - 1)$. Doesn’t see individual layers $1, \ldots, (j - 1)$ themselves. (i.e., limited visibility of layers behind)

  [Damm-Jukna-Sgall’96]
Restricted Protocols

Partial progress: protocols with more restricted forms of information sharing

- **Myopic protocols:** $P_j$ only sees layers $1, \ldots, (j - 1)$ as well as layer $(j + 1)$ of graph. (i.e., limited visibility of layers ahead)

  [Gronemeier’06]

- **Conservative protocols:** $P_j$ sees layers $(j + 1), \ldots, k$ of graph, plus composition of layers $1, \ldots, (j - 1)$. Doesn’t see individual layers $1, \ldots, (j - 1)$ themselves. (i.e., limited visibility of layers behind)

  [Damm-Jukna-Sgall’96]

Note: The DJS protocol for $\widehat{\text{MPJ}}_k$ is both myopic and conservative!
Restricted Protocols

Partial progress: protocols with more restricted forms of information sharing

- **Myopic protocols**: $P_j$ only sees layers $1, \ldots, (j - 1)$ as well as layer $(j + 1)$ of graph. (i.e., limited visibility of layers ahead)
  
  \[ [\text{Gronemeier'06}] \]

- **Conservative protocols**: $P_j$ sees layers $(j + 1), \ldots, k$ of graph, plus composition of layers $1, \ldots, (j - 1)$. Doesn’t see individual layers $1, \ldots, (j - 1)$ themselves. (i.e., limited visibility of layers behind)

  \[ [\text{Damm-Jukna-Sgall'96}] \]

Note: The DJS protocol for $\widehat{\text{MPJ}}_k$ is both myopic and conservative!

\[ [\text{Chakrabarti'07}] \] gave randomized lower bounds for restricted protocols:

- **myopic**: $\Omega(n/k)$ bits.
- **conservative**: $\Omega(n/k^2)$ bits.
Restricted Protocols

Partial progress: protocols with more restricted forms of information sharing

- **Myopic protocols**: $P_j$ only sees layers $1, \ldots, (j - 1)$ as well as layer $(j + 1)$ of graph. (i.e., limited visibility of layers ahead)
  
  [Gronemeier’06]

- **Conservative protocols**: $P_j$ sees layers $(j + 1), \ldots, k$ of graph, plus composition of layers $1, \ldots, (j - 1)$. Doesn’t see individual layers $1, \ldots, (j - 1)$ themselves. (i.e., limited visibility of layers behind)
  
  [Damm-Jukna-Sgall’96]

Note: The DJS protocol for $\widehat{MPJ}_k$ is both myopic and conservative!

[Chakrabarti’07] gave randomized lower bounds for restricted protocols:

- **myopic**: $\Omega(n/k)$ bits.

- **conservative**: $\Omega(n/k^2)$ bits.

For the rest of this talk: all protocols are myopic.
Our Results

Question: Can there be any nontrivial myopic protocol for $MPJ_k$?
Our Results

Question: Can there be any nontrivial myopic protocol for $\text{MPJ}_k$?

No, but in an interesting way...
Our Results

Question: Can there be any nontrivial myopic protocol for MPJ$_k$?

No, but in an interesting way...

Theorem: In any myopic protocol for MPJ$_k$, some player must send at least $n/2$ bits.
Our Results

Question: Can there be any nontrivial myopic protocol for $\text{MPJ}_k$?

No, but in an interesting way...

**Theorem:** In any myopic protocol for $\text{MPJ}_k$, some player must send at least $n/2$ bits.

Definitions:

- $\text{cost}(\mathcal{P}) := \text{cost of largest message of } \mathcal{P}$.
- $\text{tcost}(\mathcal{P}) := \text{total cost of } \mathcal{P}$.
- $\delta n$-bit protocol: $\text{cost}(\mathcal{P}) = \delta n$. 
Main Theorem: There exists a decreasing function $\phi : \mathbb{N} \rightarrow \mathbb{R}$ with $\lim_{k \rightarrow \infty} \phi(k) = \frac{1}{2}$ such that

1. Any deterministic protocol for $\text{MPJ}_k$ costs at least $\phi(k)n$ bits.
Main Theorem: There exists a decreasing function $\phi : \mathbb{N} \rightarrow \mathbb{R}$ with $\lim_{k \to \infty} \phi(k) = \frac{1}{2}$ such that

1. Any deterministic protocol for $\text{MPJ}_k$ costs at least $\phi(k)n$ bits.

2. There exists a protocol $\mathcal{P}$ for $\text{MPJ}_k$ with $\text{cost}(\mathcal{P}) = \phi(k)n + o(n)$. 
Detailed Results

Main Theorem: There exists a decreasing function $\phi : \mathbb{N} \to \mathbb{R}$ with $\lim_{k \to \infty} \phi(k) = \frac{1}{2}$ such that

1. Any deterministic protocol for $\text{MPJ}_k$ costs at least $\phi(k)n$ bits.
2. There exists a protocol $\mathcal{P}$ for $\text{MPJ}_k$ with $\text{cost}(\mathcal{P}) = \phi(k)n + o(n)$.

Theorem: Any deterministic protocol for $\text{MPJ}_k$ has total cost at least $n$.

Theorem: If $\mathcal{P}$ is a deterministic protocol for $\text{MPJ}_k$, then

$$\text{cost}(\mathcal{P}) \geq n \left( \log^{(k-1)} n \right) \left( 1 - o(1) \right).$$

Theorem: Any randomized protocol for $\text{MPJ}_k$ has

$$\text{cost}(\mathcal{P}) = \Omega \left( \frac{n}{k \log n} \right).$$
Generalized Pointer Jumping

$\text{MPJ}_{m,k}$: just like $\text{MPJ}_k$, except $m \leq n$ vertices in first layer.
**Generalized Pointer Jumping**

\[ \text{MPJ}_{m, k} : \text{just like MPJ}_k, \text{ except } m \leq n \text{ vertices in first layer.} \]
**Round Elimination Lemma**

**Base Case Lemma:** Any protocol $\mathcal{P}$ for $\text{MPJ}_{m,2}$ has $\text{cost}(\mathcal{P}) \geq m$ (INDEX)

**Round Elimination Lemma:** Let $k \geq 3$. If there is a $\delta n$-bit protocol $\mathcal{P}$ for $\text{MPJ}_{m,k}$, then there is a $\delta n$-bit protocol $\mathcal{Q}$ for $\text{MPJ}_{m',k-1}$ with $m' = n \cdot 2^{-\delta n/m}$. 
Round Elimination Lemma

Base Case Lemma: Any protocol $\mathcal{P}$ for $\text{MPJ}_{m,2}$ has $\text{cost}(\mathcal{P}) \geq m$ (INDEX)

Round Elimination Lemma: Let $k \geq 3$. If there is a $\delta n$-bit protocol $\mathcal{P}$ for $\text{MPJ}_{m,k}$, then there is a $\delta n$-bit protocol $\mathcal{Q}$ for $\text{MPJ}_{m',k-1}$ with $m' = n \cdot 2^{-\delta n/m}$.

Message Sets:

- P1's input: $f_2 \in [n]^m$
- $M := M_m = \{f_2 : \text{P1 sends } m \text{ on input } f_2\}$.
- Fix $m$ to maximize $|M|$; then $|M| \geq \frac{n^m}{2^{\delta n}}$. 
Round Elimination Lemma

Base Case Lemma: Any protocol $\mathcal{P}$ for $\text{MPJ}_{m,2}$ has $\text{cost}(\mathcal{P}) \geq m$ (INDEX).

Round Elimination Lemma: Let $k \geq 3$. If there is a $\delta n$-bit protocol $\mathcal{P}$ for $\text{MPJ}_{m,k}$, then there is a $\delta n$-bit protocol $\mathcal{Q}$ for $\text{MPJ}_{m',k-1}$ with $m' = n \cdot 2^{-\delta n/m}$.

Message Sets:

- P1’s input: $f_2 \in [n]^m$
- $M := M_m = \{f_2 : \text{P1 sends } m \text{ on input } f_2\}$.
- Fix $m$ to maximize $|M|$; then $|M| \geq \frac{n^m}{2^{\delta n}}$.

Definition: For $\mathcal{F} \subseteq [n]^m$, $\text{Range}(i, \mathcal{F}) := \{f_2(i) : f_2 \in \mathcal{F}\}$.
**Round Elimination Lemma**

**Base Case Lemma:** Any protocol \( \mathcal{P} \) for \( \text{MPJ}_{m,2} \) has \( \text{cost}(\mathcal{P}) \geq m \) (INDEX)

**Round Elimination Lemma:** Let \( k \geq 3 \). If there is a \( \delta n \)-bit protocol \( \mathcal{P} \) for \( \text{MPJ}_{m,k} \), then there is a \( \delta n \)-bit protocol \( \mathcal{Q} \) for \( \text{MPJ}_{m',k-1} \) with \( m' = n \cdot 2^{-\delta n/m} \).

**Message Sets:**

- P1’s input: \( f_2 \in [n][m] \)
- \( M := M_m = \{ f_2 : \text{P1 sends } m \text{ on input } f_2 \} \).
- Fix \( m \) to maximize \( |M| \); then \( |M| \geq \frac{n^m}{2^\delta n} \).

**Definition:** For \( \mathcal{F} \subseteq [n][m] \), \( \text{Range}(i, \mathcal{F}) := \{ f_2(i) : f_2 \in \mathcal{F} \} \)

**Range Lemma:** If \( |\mathcal{F}| \geq (m')^m \), then \( \exists i \) with \( |\text{Range}(i, \mathcal{F})| \geq m' \)
Proof of Round Elimination Lemma

Base Case Lemma: Any protocol $\mathcal{P}$ for $\text{MPJ}_{m,2}$ has $\text{cost}(\mathcal{P}) \geq m$ (INDEX)

Round Elimination Lemma: Let $k \geq 3$. If there is a $\delta n$-bit protocol $\mathcal{P}$ for $\text{MPJ}_{m,k}$, then there is a $\delta n$-bit protocol $\mathcal{Q}$ for $\text{MPJ}_{m',k-1}$ with $m' = n \cdot 2^{-\delta n/m}$.

Proof:
Proof of Round Elimination Lemma

Base Case Lemma: Any protocol $\mathcal{P}$ for $\text{MPJ}_{m,2}$ has $\text{cost}(\mathcal{P}) \geq m$ (INDEX).

Round Elimination Lemma: Let $k \geq 3$. If there is a $\delta n$-bit protocol $\mathcal{P}$ for $\text{MPJ}_{m,k}$, then there is a $\delta n$-bit protocol $\mathcal{Q}$ for $\text{MPJ}_{m',k-1}$ with $m' = n \cdot 2^{-\delta n/m}$.

Proof:

- Fix $M$. Note: $|M| \geq \frac{n^m}{2^{\delta n}} = 2^{m \log n - \delta n} = (m')^m$.

- By Range Lemma, $\exists \ i \in [m]$ s.t. $|\text{Range}(i, M)| \geq m'$. Fix $i$.

- For each $j \in [m']$, fix $g_j \in M$ s.t. $g_j(i) = j$.

- Protocol $\mathcal{Q}$: on input $(j, f_3, \ldots, f_{k-1}, x)$, players simulate $\mathcal{P}$ on input $(i, g_j, f_3, \ldots, f_{k-1}, x)$.
Analysis

Define

- \( a_0 := 0, a_\ell := \delta 2^{a_\ell - 1} \)
- \( m_\ell := n 2^{-a_\ell} \)

Definition: Let \( \phi(k) := \) least \( \delta \) such that \( a_{k-1} \geq 1 \)
Analysis

Define

- \( a_0 := 0, a_\ell := \delta 2^{a_{\ell-1}} \)
- \( m_\ell := n 2^{-a_\ell} \)

Definition: Let \( \phi(k) := \) least \( \delta \) such that \( a_{k-1} \geq 1 \)
Analysis

Define

- \( a_0 := 0 \), \( a_\ell := \delta 2^{a_{\ell - 1}} \)
- \( m_\ell := n 2^{-a_\ell} \)

Definition: Let \( \phi(k) := \text{least } \delta \text{ such that } a_{k-1} \geq 1 \)
Define

- $a_0 := 0$, $a_\ell := \delta 2^{a_\ell - 1}$
- $m_\ell := n2^{-a_\ell}$

**Definition:** Let $\phi(k) := \text{least } \delta \text{ such that } a_{k-1} \geq 1$
Define

- \( a_0 := 0, a_\ell := \delta 2^{a_{\ell-1}} \)
- \( m_\ell := n^{2^{-a_\ell}} \)

**Definition:** Let \( \phi(k) := \text{least } \delta \text{ such that } a_{k-1} \geq 1 \)
Define

- \( a_0 := 0, a_\ell := \delta 2^{a_{\ell-1}} \)
- \( m_\ell := n2^{-a_\ell} \)

**Definition:** Let \( \phi(k) := \) least \( \delta \) such that \( a_{k-1} \geq 1 \)
Analysis

Define

- $a_0 := 0, a_\ell := \delta 2^{a_{\ell-1}}$
- $m_\ell := n 2^{-a_\ell}$

Definition: Let $\phi(k) := \text{least } \delta \text{ such that } a_{k-1} \geq 1$

Claim: $\lim_{k \to \infty} \phi(k) = 1/2$ (Induction)

Round elimination $(m = m_\ell)$:

$$m' = n 2^{-\frac{\delta n}{m_\ell}} = n 2^{-\delta n/n 2^{-a_\ell}} = n 2^{-\delta 2^{a_\ell}} = n 2^{-a_{\ell+1}} = m_{\ell+1}$$
Proof of Main Theorem

**Theorem:** Any myopic protocol $\mathcal{P}$ for $\text{MPJ}_k = \text{MPJ}_{n,k}$ has

$$\text{cost}(\mathcal{P}) \geq n\phi(k).$$

**Proof:**

Joshua Brody
Proof of Main Theorem

**Theorem:** Any myopic protocol \( \mathcal{P} \) for \( \text{MPJ}_k = \text{MPJ}_{n,k} \) has

\[
\text{cost}(\mathcal{P}) \geq n\phi(k).
\]

**Proof:**

\( \delta n \)-bit protocol for \( \text{MPJ}_{m_0,k} \) \( \Rightarrow \)

\[ \ldots k - 2 \text{ round eliminations} \ldots \Rightarrow \]

\( \delta n \)-bit protocol for \( \text{MPJ}_{m_{k-2},2} \) \( \Rightarrow \)

\[ \delta n \geq n2^{-a_{k-2}} = m_{k-2} \quad \text{(Base Case Lemma)} \Rightarrow \]

\[ a_{k-1} = \delta 2^{a_{k-2}} \geq 1 \Rightarrow \]

\[ \delta \geq \phi(k) \quad \text{(by def. of } \phi(k) \text{)} \]
A Sketch of Matching Upper Bound

Idea: Cover \([n]^m\) with sets \(S_1, \ldots, S_t \subseteq [n]^m\) s.t.

\[|\text{Range}(i, S)| = m' \text{ for all } i, S.\]

Packing lower bound: \(t \geq 2^{\delta n}\).

Claim: \(t \leq 2^{\delta n + o(n)}\). (Prob. Method)

Protocol:

- P1 sends \(S \ni f_2\). \((\text{cost } = \delta n + o(n))\)
- Players 2, \ldots, k see \(i\), set \([m'] := \text{Range}(i, S)\).
- Players 2, \ldots, k run \(\text{MPJ}_{m',k-1}\) protocol on \((f_2(i), f_3, \ldots, x)\).
Randomizing the Lower Bound

Round Elimination Lemma: Let $k \geq 3$. If there is a $\delta n$-bit, $\varepsilon$-error distributional protocol $P$ for $\text{MPJ}_{m,k}$, then there is a $\delta n$-bit, $\varepsilon'$-error protocol $Q$ for $\text{MPJ}_{m',k-1}$ with $m' = n \cdot 2^{-2\delta n/m}$ and $\varepsilon' = 2n\varepsilon$.

Proof:
Randomizing the Lower Bound

Round Elimination Lemma: Let $k \geq 3$. If there is a $\delta n$-bit, $\varepsilon$-error distributional protocol $P$ for $\text{MPJ}_{m,k}$, then there is a $\delta n$-bit, $\varepsilon'$-error protocol $Q$ for $\text{MPJ}_{m',k-1}$ with $m' = n \cdot 2^{-2\delta n/m}$ and $\varepsilon' = 2n\varepsilon$.

Proof:

- $z := (f_3, \ldots, f_{k-1}, x)$
- Call $(i, f_2)$ bad if $\Pr_z[\text{error } | (i, f_2)] > 2n\varepsilon$

- Call $f_2$ bad if $\Pr_i[(i, f_2) \text{ bad } | f_2] \geq 1/n$
Randomizing the Lower Bound

Round Elimination Lemma: Let $k \geq 3$. If there is a $\delta n$-bit, $\epsilon$-error distributional protocol $P$ for $\text{MPJ}_{m,k}$, then there is a $\delta n$-bit, $\epsilon'$-error protocol $Q$ for $\text{MPJ}_{m',k-1}$ with $m' = n \cdot 2^{-2\delta n/m}$ and $\epsilon' = 2n\epsilon$.

Proof:

- $z := (f_3, \ldots, f_{k-1}, x)$
- Call $(i, f_2)$ bad if $\Pr_z[\text{error} | (i, f_2)] > 2n\epsilon$  
  $\Rightarrow \Pr[(i, f_2) \text{ bad}] < 1/2n$  \hspace{1cm} \text{(Markov)}

- Call $f_2$ bad if $\Pr_i[(i, f_2) \text{ bad} | f_2] \geq 1/n$  
  $\Rightarrow \Pr[f_2 \text{ bad}] < 1/2$  \hspace{1cm} \text{(Markov)}
Randomizing the Lower Bound

Round Elimination Lemma: Let $k \geq 3$. If there is a $\delta n$-bit, $\varepsilon$-error distributional protocol $\mathcal{P}$ for $\text{MPJ}_{m,k}$, then there is a $\delta n$-bit, $\varepsilon'$-error protocol $\mathcal{Q}$ for $\text{MPJ}_{m',k-1}$ with $m' = n \cdot 2^{-2\delta n/m}$ and $\varepsilon' = 2n\varepsilon$.

Proof:

- $z := (f_3, \ldots, f_{k-1}, x)$
- Call $(i, f_2)$ bad if $\Pr_z[\text{error }|(i, f_2)] > 2n\varepsilon$
  $\Rightarrow \Pr[(i, f_2) \text{ bad}] < 1/2n$ (Markov)
- Call $f_2$ bad if $\Pr_i[(i, f_2) \text{ bad } | f_2] \geq 1/n$
  $\Rightarrow \Pr[f_2 \text{ bad}] < 1/2$ (Markov)

Note: $f_2$ good $\Rightarrow (i, f_2)$ good for all $i$. 
Randomizing the Lower Bound

Round Elimination Lemma: Let $k \geq 3$. If there is a $\delta n$-bit, $\varepsilon$-error distributional protocol $\mathcal{P}$ for $\text{MPJ}_{m,k}$, then there is a $\delta n$-bit, $\varepsilon'$-error protocol $Q$ for $\text{MPJ}_{m',k-1}$ with $m' = n \cdot 2^{-2\delta n/m}$ and $\varepsilon' = 2n\varepsilon$.

Proof:

- $z := (f_3, \ldots, f_{k-1}, x)$
- Call $(i, f_2)$ bad if $\Pr_z[\text{error } | (i, f_2)] > 2n\varepsilon$
  $\Rightarrow \Pr[(i, f_2) \text{ bad}] < 1/2n$ \hspace{1cm} (Markov)
- Call $f_2$ bad if $\Pr_i[(i, f_2) \text{ bad } | f_2] \geq 1/n$
  $\Rightarrow \Pr[f_2 \text{ bad}] < 1/2$ \hspace{1cm} (Markov)
  Note: $f_2$ good $\Rightarrow (i, f_2)$ good for all $i$.
- Follow deterministic proof
  $M := M_m = \{\text{good } f_2 : \text{P1 sends } m \text{ on input } f_2\}$ \ldots
Conclusions/Open Problems

Conclusions

• Still far from proving $\text{MPJ}_k \not\in \text{ACC}^0$

• Provided the first $o(n)$ protocol for $\text{MPJ}_k$

• Characterized maximum communication complexity of myopic protocols up to $1 + o(1)$ factors.

• Lower bound technique applies to $\text{MPJ}_k$ and $\widehat{\text{MPJ}}_k$ and does randomize; seems promising for other problems.

Open Problems

1. Settle $D(\text{MPJ}_k)$

2. Possible first step: improve bound on $\text{MPJ}_3$

3. Relax protocol restrictions: 2-myopic, ...
Thank you!

Questions?
Contact jbrody@cs.dartmouth.edu