# “Robust” Lower Bounds

*for Communication and Stream Computation*



**Amit Chakrabarti** Dartmouth College
**Graham Cormode** AT&T Labs
**Andrew McGregor** UC San Diego

# Communication Complexity

# Communication Complexity

- *Goal:* Evaluate $f(x_1, \ldots, x_n)$ when input is split among p players:

$x_1 \ldots x_{10}$

$x_{11} \ldots x_{20}$

$x_{21} \ldots x_{30}$

How much communication is required to evaluate f?
Consider randomized, blackboard, one-way, multi-round, ...

# Communication Complexity

- *Goal:* Evaluate $f(x_1, \ldots, x_n)$ when input is split among p players:

$x_1 \ldots x_{10}$ $x_{11} \ldots x_{20}$ $x_{21} \ldots x_{30}$

How much communication is required to evaluate f?
Consider randomized, blackboard, one-way, multi-round, ...

# Communication Complexity

- *Goal:* Evaluate $f(x_1, \ldots, x_n)$ when input is split among p players:


  How much communication is required to evaluate f?
  Consider randomized, blackboard, one-way, multi-round, ...

- *How important is the split?*

  Is f hard for many splits or only hard for a few bad splits?

  Previous work on worst and best partitions.

  [Aho, Ullman, Yannakakis '83] [Papadimitriou, Sipser '84]

# Communication Complexity

- *Goal:* Evaluate $f(x_1, \ldots, x_n)$ when input is split among p players:


  How much communication is required to evaluate f?
  Consider randomized, blackboard, one-way, multi-round, ...

- *How important is the split?*

  Is f hard for many splits or only hard for a few bad splits?

  Previous work on worst and best partitions.

  [Aho, Ullman, Yannakakis '83] [Papadimitriou, Sipser '84]

- *Consider random partitions:*

  Define error probability over coin flips and random split.

# Stream Computation

[Morris '78] [Munro, Paterson '78] [Flajolet, Martin '85]
[Alon, Matias, Szegedy '96] [Henzinger, Raghavan, Rajagopalan '98]

- *Goal:* Evaluate $f(x_1, \ldots, x_n)$ given sequential access:

↓

$x_1\ x_2\ x_3\ x_4\ x_5\ \ldots \quad \ldots\ x_n$

# Stream Computation

[Morris '78] [Munro, Paterson '78] [Flajolet, Martin '85]
[Alon, Matias, Szegedy '96] [Henzinger, Raghavan, Rajagopalan '98]

- *Goal:* Evaluate $f(x_1, \ldots, x_n)$ given sequential access:

$$\downarrow$$

$$x_1\ x_2\ x_3\ x_4\ x_5\ \ldots \qquad \ldots\ x_n$$

# Stream Computation

[Morris '78] [Munro, Paterson '78] [Flajolet, Martin '85]
[Alon, Matias, Szegedy '96] [Henzinger, Raghavan, Rajagopalan '98]

- *Goal:* Evaluate $f(x_1, \ldots, x_n)$ given sequential access:

↓

$x_1\ x_2\ x_3\ x_4\ x_5\ \ldots \qquad \ldots\ x_n$

How much working memory is required to evaluate f?
Consider randomized, approximate, multi-pass, etc.

# Stream Computation

[Morris '78] [Munro, Paterson '78] [Flajolet, Martin '85]
[Alon, Matias, Szegedy '96] [Henzinger, Raghavan, Rajagopalan '98]

- *Goal:* Evaluate $f(x_1, \ldots, x_n)$ given sequential access:

  ↓

  $x_1\ x_2\ x_3\ x_4\ x_5\ \ldots\quad \ldots\ x_n$

  How much working memory is required to evaluate f?
  Consider randomized, approximate, multi-pass, etc.

- *Random-order streams:* Assume f is order-invariant:

  Upper Bounds: e.g., stream of i.i.d. samples.

  Lower Bounds: is a "hard" problem hard in practice?

[Munro, Paterson '78] [Demaine, López-Ortiz, Munro '02]
[Guha, McGregor '06, '07a, '07b] [Chakrabarti, Jayram, Patrascu '08]

# Stream Computation

[Morris '78] [Munro, Paterson '78] [Flajolet, Martin '85]
[Alon, Matias, Szegedy '96] [Henzinger, Raghavan, Rajagopalan '98]

- *Goal:* Evaluate $f(x_1, \ldots, x_n)$ given sequential access:

  $$x_1\ x_2\ x_3\ x_4\ x_5\ \ldots\ \ \ \ldots\ x_n$$

  How much working memory is required to evaluate f?
  Consider randomized, approximate, multi-pass, etc.

- *Random-order streams:* Assume f is order-invariant:

  Upper Bounds: e.g., stream of i.i.d. samples.

  Lower Bounds: is a "hard" problem hard in practice?

  [Munro, Paterson '78] [Demaine, López-Ortiz, Munro '02]
  [Guha, McGregor '06, '07a, '07b] [Chakrabarti, Jayram, Patrascu '08]

- Random-partition-CC bounds give random-order bounds

# Results

- *t-party Set-Disjointess:* Any protocol for $\Omega(t^2)$-player random-partition requires $\Omega(n/t)$ bits communication.

  ∴ 2-approx. for $k^{th}$ freq. moments requires $\Omega(n^{1-3/k})$ space.

- *Median:* Any $p$-round protocol for $p$-player random-partition requires $\Omega(m^{f(p)})$ where $f(p)=1/3^p$

  ∴ Polylog(m)-space algorithm requires $\Omega(\log\log m)$ passes.

- *Gap-Hamming:* Any one-way protocol for 2-player random-partition requires $\Omega(n)$ bits communicated.

  ∴ $(1+\varepsilon)$-approx. for $F_0$ or entropy requires $\Omega(\varepsilon^{-2})$ space.

- *Index:* Any one-way protocol for 2-player random-partition (with duplicates) requires $\Omega(n)$ bits communicated.

  ∴ Connectivity of a graph $G=(V, E)$ requires $\Omega(|V|)$ space.

# The Challenge...

2 5 6 ... 21 23

1 8 8 ... 24 24

0 0 0 ... 25 25

# The Challenge...

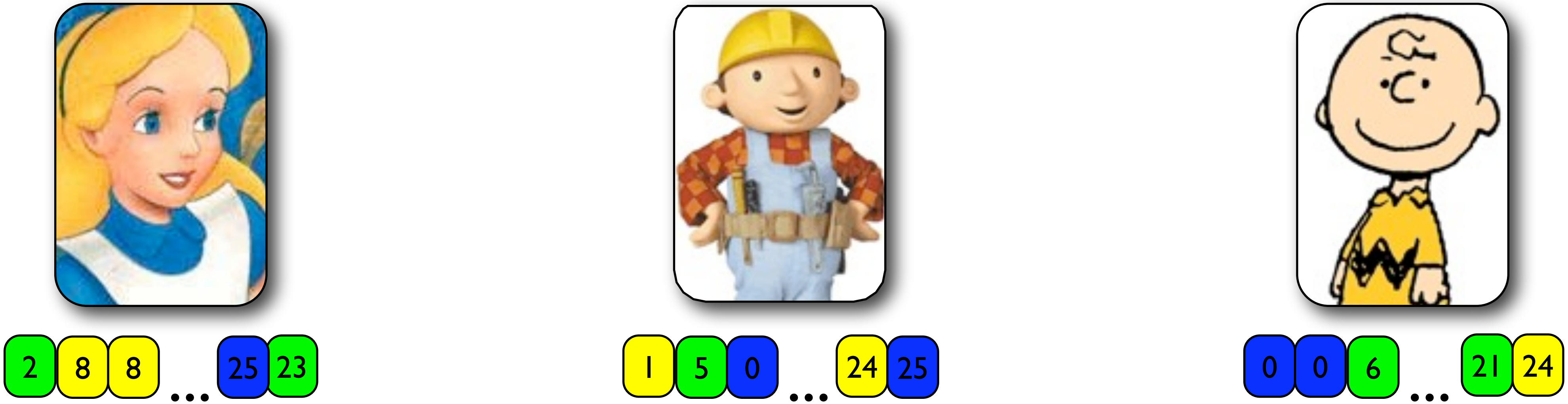
2 5 6 ... 21 23

1 8 8 ... 24 24

0 0 0 ... 25 25

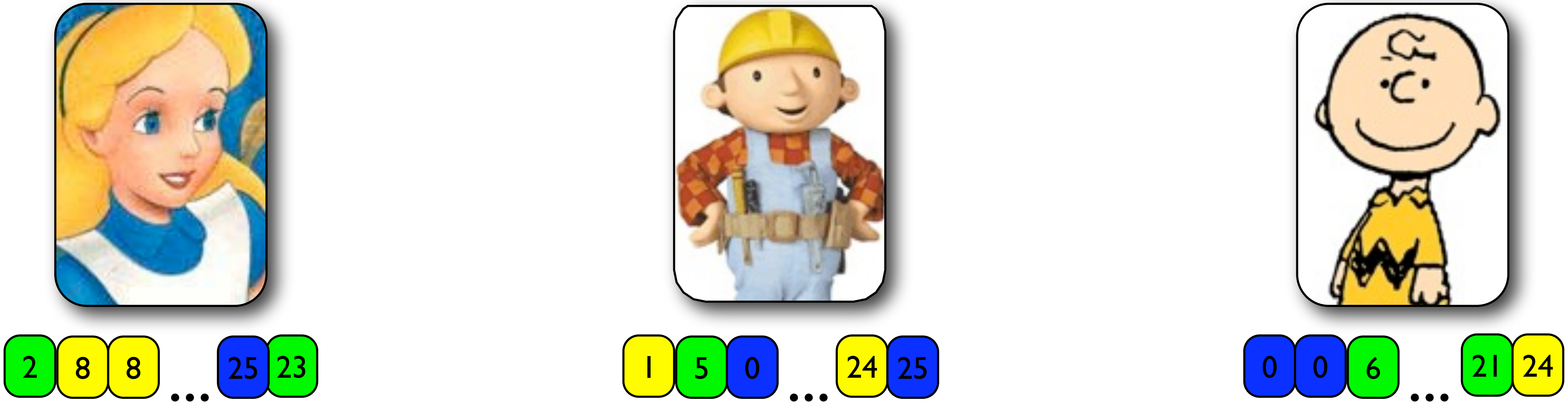
- *Naive reduction from fixed-partition-CC:*
  1. Players determine random partition, send necessary data.
  2. Simulate protocol on random partition.

# The Challenge...




- *Naive reduction from fixed-partition-CC:*
  1. Players determine random partition, send necessary data.
  2. Simulate protocol on random partition.

# The Challenge...




- *Naive reduction from fixed-partition-CC:*
  1. Players determine random partition, send necessary data.
  2. Simulate protocol on random partition.
- *Problem:* Seems to require too much communication.

# The Challenge...

2 8 8 ... 25 23

1 5 0 ... 24 25

0 0 6 ... 21 24

- *Naive reduction from fixed-partition-CC:*
  1. Players determine random partition, send necessary data.
  2. Simulate protocol on random partition.
- *Problem:* Seems to require too much communication.
- *Consider random input and public coins:*

  Issue #1: Need independence of input and partition.

  Issue #2: Generalize information statistics techniques.

*a)* Disjointness

*b)* Selection

*a)* Disjointness

*b)* Selection

# Multi-Party Set-Disjointness

- *Instance:* *t* x *n* matrix,

$$X = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \end{pmatrix}$$

and define, $\mathsf{DISJ}_{n,t} = \bigvee_i \mathsf{AND}_t(x_{1,i}, \ldots, x_{t,i})$

# Multi-Party Set-Disjointness

- *Instance:* $t$ x $n$ matrix,

$$X = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \end{pmatrix}$$

  and define, $\mathrm{DISJ}_{n,t} = \bigvee_i \mathrm{AND}_t(x_{1,i}, \ldots, x_{t,i})$

- *Unique intersection:* Each column has weight 0, 1, or $t$ and at most one column has weight $t$.

# Multi-Party Set-Disjointness

- *Instance:* *t* x *n* matrix,

$$X = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \end{pmatrix}$$

  and define, $\mathrm{DISJ}_{n,t} = \bigvee_i \mathrm{AND}_t(x_{1,i}, \ldots, x_{t,i})$

- *Unique intersection:* Each column has weight 0, 1, or *t* and at most one column has weight *t*.

- *Thm:* $\Omega(n/t)$ bound if *t*-players each get a row.

[Kalyanasundaram, Schnitger '92] [Razborov '92]
[Chakrabarti, Khot, Sun '03] [Bar-Yossef, Jayram, Kumar, Sivakumar '04]

# Multi-Party Set-Disjointness

- *Instance:* *t* x *n* matrix,

$$X = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \end{pmatrix}$$

  and define, $\mathrm{DISJ}_{n,t} = \bigvee_i \mathrm{AND}_t(x_{1,i}, \ldots, x_{t,i})$

- *Unique intersection:* Each column has weight 0, 1, or *t* and at most one column has weight *t*.

- *Thm:* $\Omega(n/t)$ bound if *t*-players each get a row.

  [Kalyanasundaram, Schnitger '92] [Razborov '92]
  [Chakrabarti, Khot, Sun '03] [Bar-Yossef, Jayram, Kumar, Sivakumar '04]

- *Thm:* $\Omega(n/t)$ bound for random partition for $\Omega(t^2)$ players.

# *Generalize Information Statistics Approach...*

[Chakrabarti, Shi, Wirth, Yao '01] [Chakrabarti, Khot, Sun '03] [Bar-Yossef, Jayram, Kumar, Sivakumar '04]

# *Generalize Information Statistics Approach...*

[Chakrabarti, Shi, Wirth, Yao '01] [Chakrabarti, Khot, Sun '03] [Bar-Yossef, Jayram, Kumar, Sivakumar '04]

- $\Pi(X)$ is transcript of $\delta$-error protocol $\Pi$ on *random input* $X \sim \mu$.

# ***Generalize Information Statistics Approach…***

[Chakrabarti, Shi, Wirth, Yao ’01] [Chakrabarti, Khot, Sun ’03] [Bar-Yossef, Jayram, Kumar, Sivakumar ’04]

- $\Pi(X)$ is transcript of $\delta$-error protocol $\Pi$ on *random input* $X$~$\mu$.
- *Information Cost:* $\text{icost}(\Pi) = I(X : \Pi(X))$

  Lower bound on the length of the protocol

  Amenable to direct-sum results…

# *Generalize Information Statistics Approach...*

[Chakrabarti, Shi, Wirth, Yao '01] [Chakrabarti, Khot, Sun '03] [Bar-Yossef, Jayram, Kumar, Sivakumar '04]

- $\Pi(X)$ is transcript of $\delta$-error protocol $\Pi$ on *random input* $X \sim \mu$.
- *Information Cost:* $\text{icost}(\Pi) = I(X : \Pi(X))$

  Lower bound on the length of the protocol

  Amenable to direct-sum results...

$\text{icost}(\Pi) \geq \sum_j I(X^j : \Pi(X))$

where $X^j$ is $j^{\text{th}}$ column of matrix $X$

*Step 1:*

# Generalize Information Statistics Approach...

[Chakrabarti, Shi, Wirth, Yao '01] [Chakrabarti, Khot, Sun '03] [Bar-Yossef, Jayram, Kumar, Sivakumar '04]

- $\Pi(X)$ is transcript of δ-error protocol Π on *random input* $X \sim \mu$.
- *Information Cost:* $\text{icost}(\Pi) = I(X : \Pi(X))$

  Lower bound on the length of the protocol

  Amenable to direct-sum results...

Step 1:

$$\text{icost}(\Pi) \geq \sum_j I(X^j : \Pi(X))$$

where $X^j$ is $j$ th column of matrix $X$

Step 2:

$$I(X^j : \Pi(X)) \geq \text{icost}(\Pi')$$

where Π' is "best" δ-error protocol for $\text{AND}_t$

# *Generalize Information Statistics Approach...*

[Chakrabarti, Shi, Wirth, Yao '01] [Chakrabarti, Khot, Sun '03] [Bar-Yossef, Jayram, Kumar, Sivakumar '04]

- $\Pi(X)$ is transcript of δ-error protocol Π on *random input* $X \sim \mu$.
- *Information Cost:* $\mathrm{icost}(\Pi) = I(X : \Pi(X))$

  Lower bound on the length of the protocol

  Amenable to direct-sum results...

**Step 1:** $\mathrm{icost}(\Pi) \geq \sum_j I(X^j : \Pi(X))$

where $X^j$ is $j$ th column of matrix $X$

**Step 2:** $I(X^j : \Pi(X)) \geq \mathrm{icost}(\Pi')$

where Π' is "best" δ-error protocol for $\mathrm{AND}_t$

**Step 3:** $\mathrm{icost}(\Pi') \geq \Omega(1/t)$

assuming Π' is private-coin, one-way protocol

# *Generalize Information Statistics Approach...*

[Chakrabarti, Shi, Wirth, Yao '01] [Chakrabarti, Khot, Sun '03] [Bar-Yossef, Jayram, Kumar, Sivakumar '04]

- $\Pi(X)$ is transcript of $\delta$-error protocol $\Pi$ on *random input* $X \sim \mu$.
- *Information Cost:* $\text{icost}(\Pi) = I(X : \Pi(X))$
  Lower bound on the length of the protocol
  Amenable to direct-sum results...

*Step 1:* $\text{icost}(\Pi) \geq \sum_j I(X^j : \Pi(X))$

where $X^j$ is $j^{\text{th}}$ column of matrix $X$

*Step 2:* $I(X^j : \Pi(X)) \geq \text{icost}(\Pi')$

where Π' is "best" δ-error protocol for $\text{AND}_t$

*Step 3:* $\text{icost}(\Pi') \geq \Omega(1/t)$

assuming Π' is private-coin, one-way protocol

- *Necessary Generalization:*
  Step 1: Condition "icost" on public coins.
  Step 2: Error of Π' is best $\delta$+Birthday$(t,p)$ error protocol.
  Step 3: Generalize result for public-coin protocols.

# Frequency Moments

# Frequency Moments

- ***Define:*** $F_k(S) = \sum_i (\text{freq. of i})^k$

# Frequency Moments

- ***Define:*** $F_k(S) = \sum_i (\text{freq. of i})^k$
- ***Reduction from set-disjointness:*** [Alon, Matias, Szegedy '99]

$$S = \{i : x_{ij} = 1\}$$

$$\begin{aligned} F_k(S) &\geq t^k \text{ if } \mathrm{DISJ}_{n,t}(X) = 1 \\ F_k(S) &\leq n \text{ if } \mathrm{DISJ}_{n,t}(X) = 0 \end{aligned}$$

# Frequency Moments

- *Define:* $F_k(S) = \sum_i (\text{freq. of i})^k$
- *Reduction from set-disjointness:* [Alon, Matias, Szegedy '99]

$$S = \{i : x_{ij} = 1\}$$

$$\begin{aligned} F_k(S) &\geq t^k \text{ if } \mathrm{DISJ}_{n,t}(X) = 1 \\ F_k(S) &\leq n \text{ if } \mathrm{DISJ}_{n,t}(X) = 0 \end{aligned}$$

- *Thm:* $\Omega(n^{1-3/k})$ space bound for random order streams.
- *Proof:* Set $t^k=2n$ to prove $\Omega(n^{1-1/k})$ total communication

  Per-message communication is $\Omega(n^{1-1/k}/p)= \Omega(n^{1-3/k})$

# Frequency Moments

- *Define:* $F_k(S) = \sum_i (\text{freq. of i})^k$
- *Reduction from set-disjointness:* [Alon, Matias, Szegedy '99]

$$S = \{i : x_{ij} = 1\}$$

$$F_k(S) \geq t^k \text{ if } \mathrm{DISJ}_{n,t}(X) = 1$$

$$F_k(S) \leq n \text{ if } \mathrm{DISJ}_{n,t}(X) = 0$$

- *Thm:* $\Omega(n^{1-3/k})$ space bound for random order streams.
- *Proof:* Set $t^k$=2$n$ to prove $\Omega(n^{1-1/k})$ total communication

  Per-message communication is $\Omega(n^{1-1/k}/p) = \Omega(n^{1-3/k})$
- *Open Problem:* $\Omega(n^{1-2/k})$ bound for random order?

*a)* Disjointness

*b)* Selection

# Selection in Streams

# Selection in Streams

- Find median of stream of m values in polylog($m$) space.

# Selection in Streams

- Find median of stream of m values in polylog($m$) space.
- *Thm:* For adversarial-order stream, $\Theta(\lg m / \lg \lg m)$ pass

  [Munro, Paterson '78] [Guha, McGregor '07a]

# Selection in Streams

- Find median of stream of m values in polylog($m$) space.
- *Thm:* For adversarial-order stream, $\Theta(\lg m / \lg\lg m)$ pass

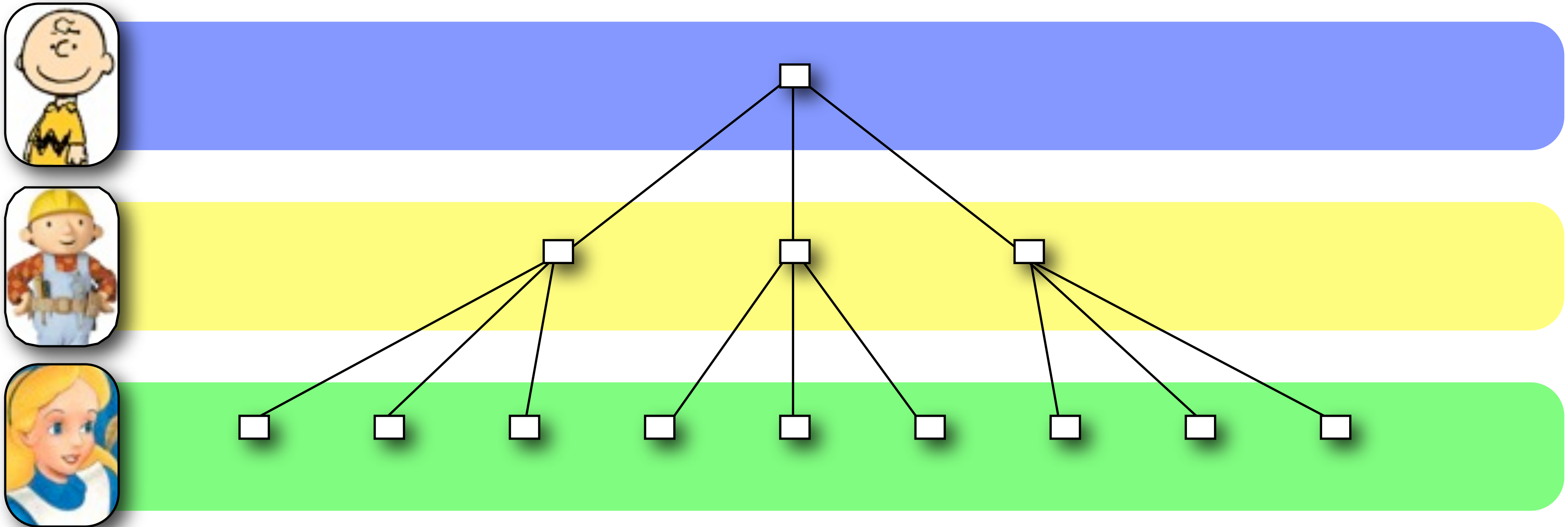
  [Munro, Paterson '78] [Guha, McGregor '07a]
- *Thm:* For random-order stream, $\Theta(\lg\lg m)$ pass

  [Guha, McGregor '06] [Chakrabarti, Jayram, Patrascu '08]

# Selection in Streams

- Find median of stream of m values in polylog($m$) space.
- *Thm:* For adversarial-order stream, $\Theta(\lg m / \lg \lg m)$ pass
  [Munro, Paterson '78] [Guha, McGregor '07a]
- *Thm:* For random-order stream, $\Theta(\lg \lg m)$ pass
  [Guha, McGregor '06] [Chakrabarti, Jayram, Patrascu '08]
- *Our result:* Using random-partition-CC techniques we get simpler and tighter pass/space trade-offs...

# Tree Pointer Jumping (TPJ)...



- *Instance:* Function on nodes of $(p+1)$-level, $t$-ary tree,

  if $v$ is an internal node: $f$ maps $v$ to a child of $v$

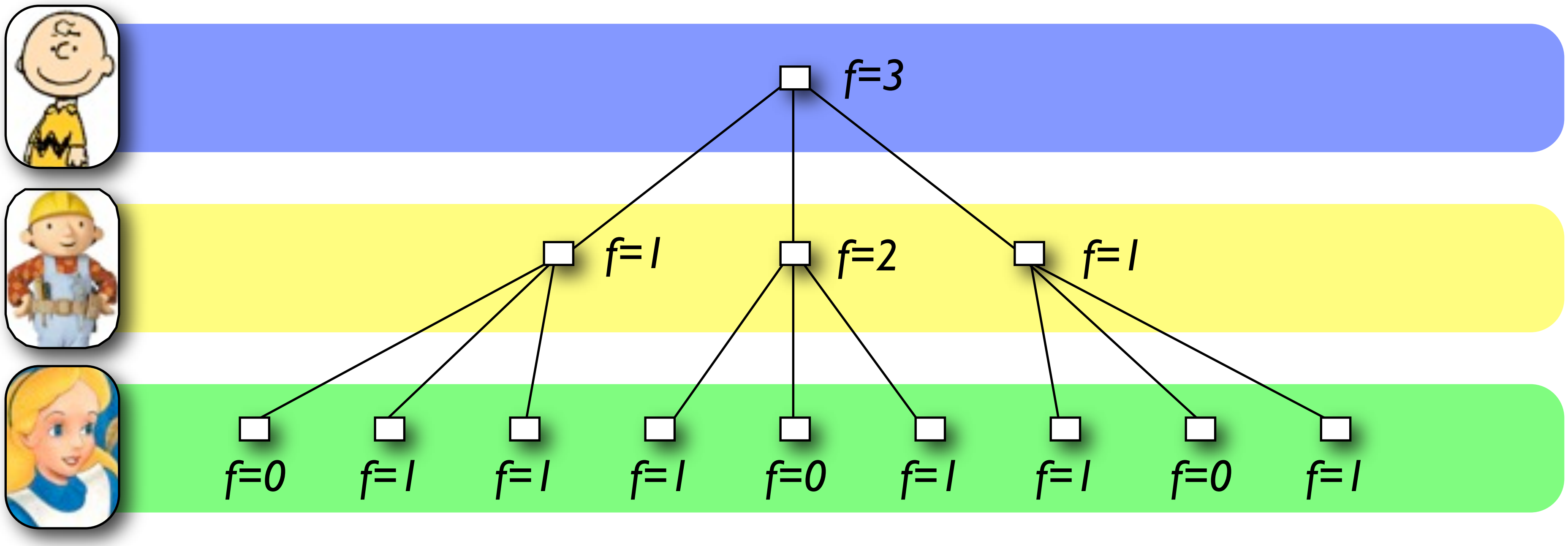
  if $v$ is a leaf: $f$ maps $v$ to $\{0,1\}$

- *Goal:* Compute $f(f(\ldots f(v_{root})\ldots))$.

- *Thm:* With $p$-players, if $i^{th}$ player knows $f(v)$ when level$(v)=i$:

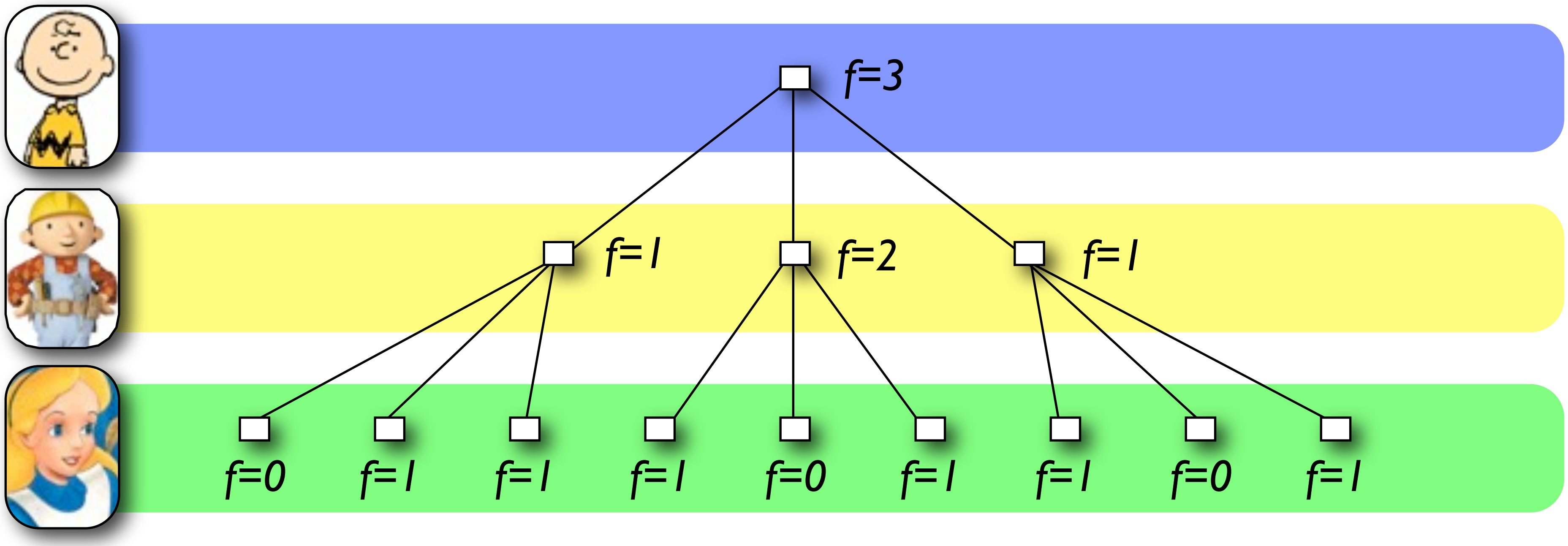
  Any $p$-round protocol requires $\Omega(t)$ communication.

# *Tree Pointer Jumping (TPJ)...*




- *Instance:* Function on nodes of ($p$+1)-level, $t$-ary tree,
  - if $v$ is an internal node: $f$ maps $v$ to a child of $v$
  - if $v$ is a leaf: $f$ maps $v$ to {0,1}
- *Goal:* Compute $f(f(\ldots f(v_{root})\ldots))$.
- *Thm:* With $p$-players, if $i^{th}$ player knows $f(v)$ when level($v$)=$i$:
  Any $p$-round protocol requires $\Omega(t)$ communication.

# Tree Pointer Jumping (TPJ)...



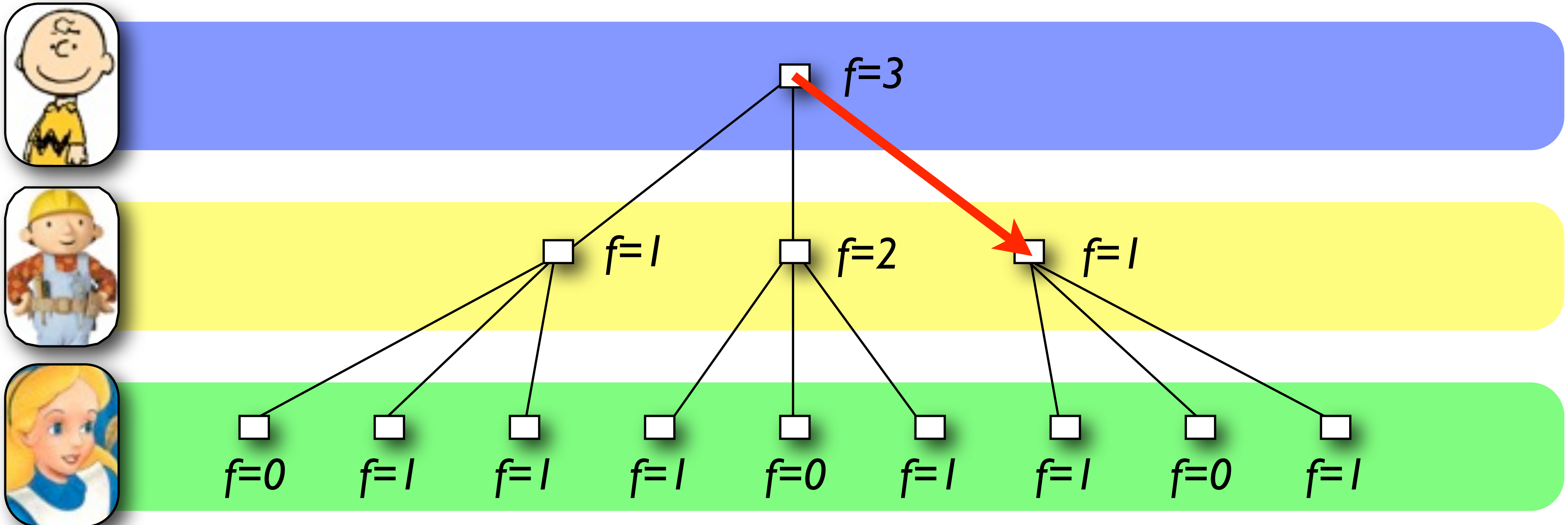

- *Instance:* Function on nodes of ($p$+1)-level, $t$-ary tree,

  if $v$ is an internal node: $f$ maps $v$ to a child of $v$

  if $v$ is a leaf: $f$ maps $v$ to {0,1}

- *Goal:* Compute $f(f(\ldots f(v_{root})\ldots))$.

# *Tree Pointer Jumping (TPJ)...*



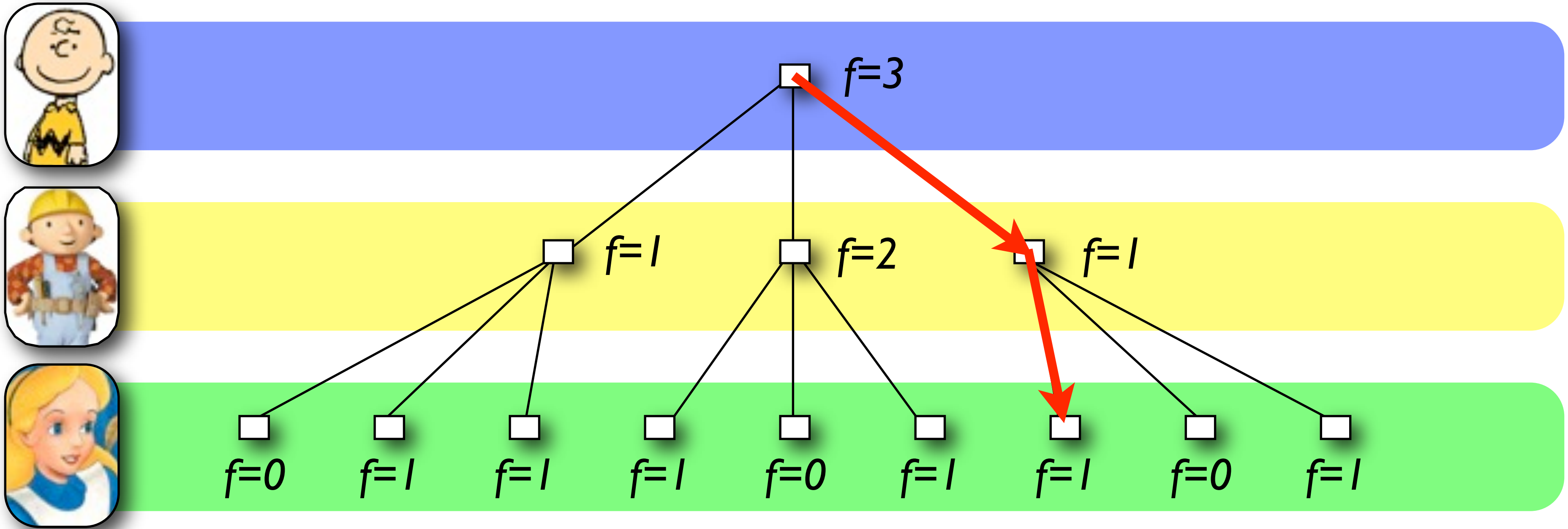

- *Instance:* Function on nodes of ($p$+1)-level, $t$-ary tree,

  if $v$ is an internal node: $f$ maps $v$ to a child of $v$

  if $v$ is a leaf: $f$ maps $v$ to {0,1}

- *Goal:* Compute $f(f(\ldots f(v_{root})\ldots))$.

# Tree Pointer Jumping (TPJ)...



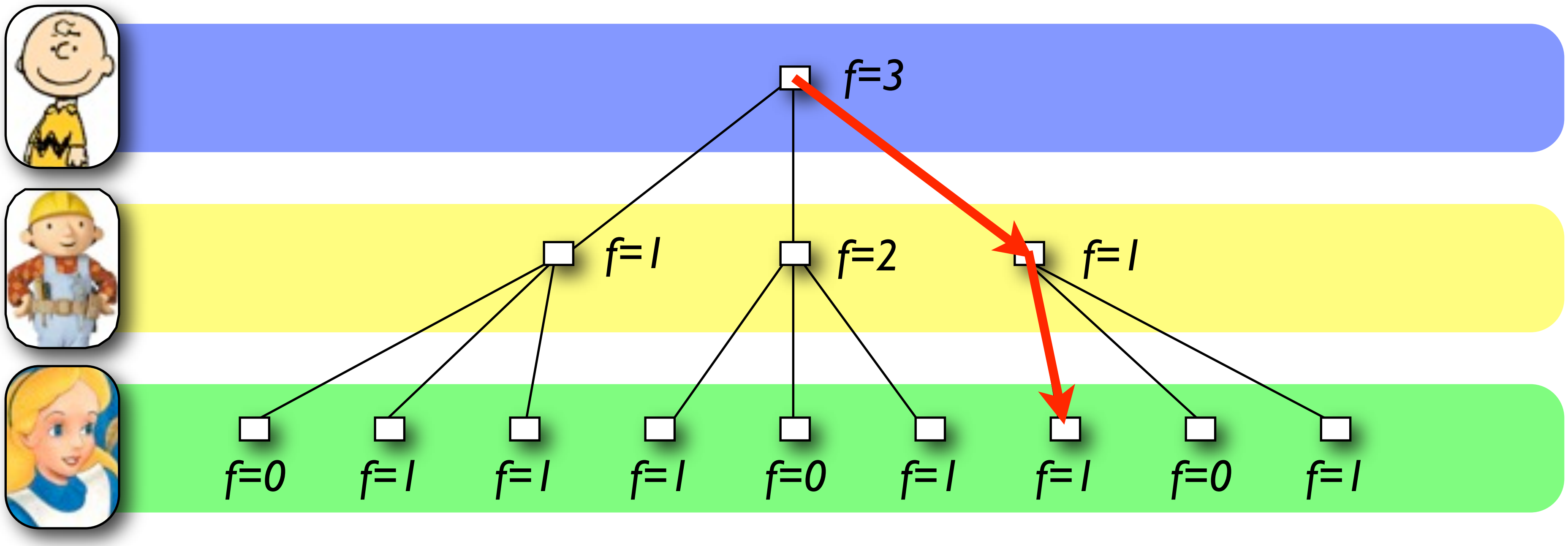

- *Instance:* Function on nodes of $(p+1)$-level, $t$-ary tree,

  if $v$ is an internal node: $f$ maps $v$ to a child of $v$

  if $v$ is a leaf: $f$ maps $v$ to $\{0,1\}$

- *Goal:* Compute $f(f(\ldots f(v_{root})\ldots))$.

# Tree Pointer Jumping (TPJ)...




- *Instance:* Function on nodes of $(p+1)$-level, $t$-ary tree,

  if $v$ is an internal node: $f$ maps $v$ to a child of $v$

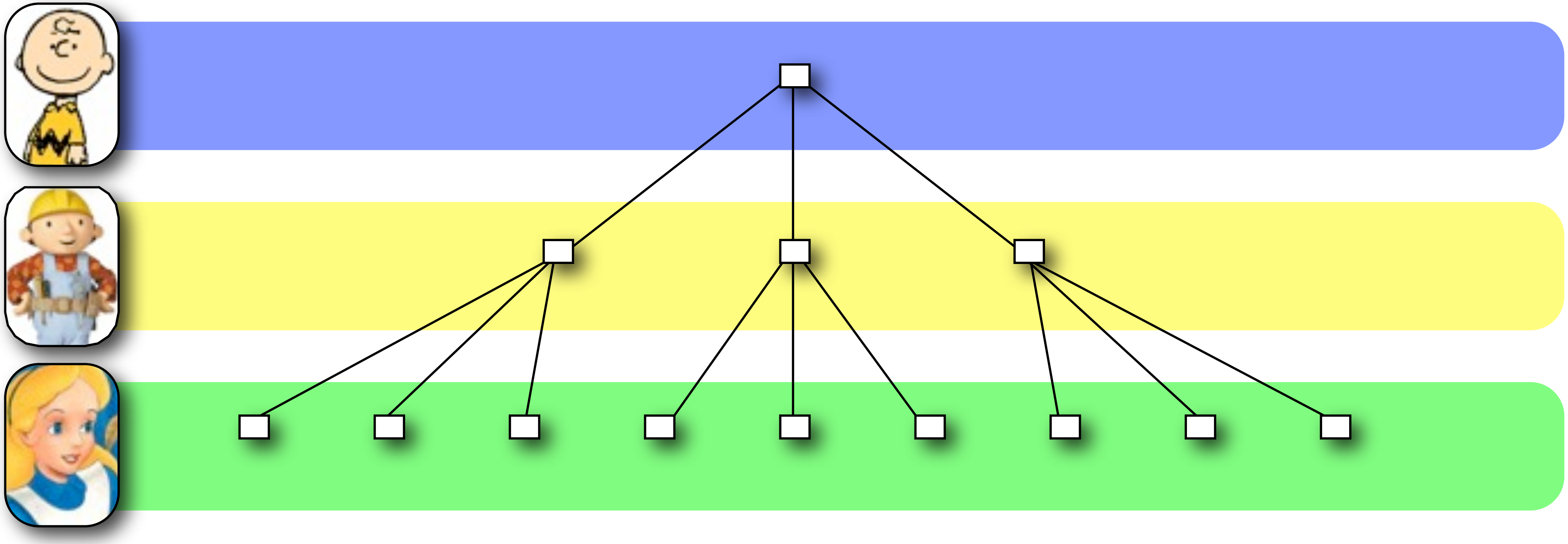
  if $v$ is a leaf: $f$ maps $v$ to $\{0,1\}$

- *Goal:* Compute $f(f(\ldots f(v_{root})\ldots))$.

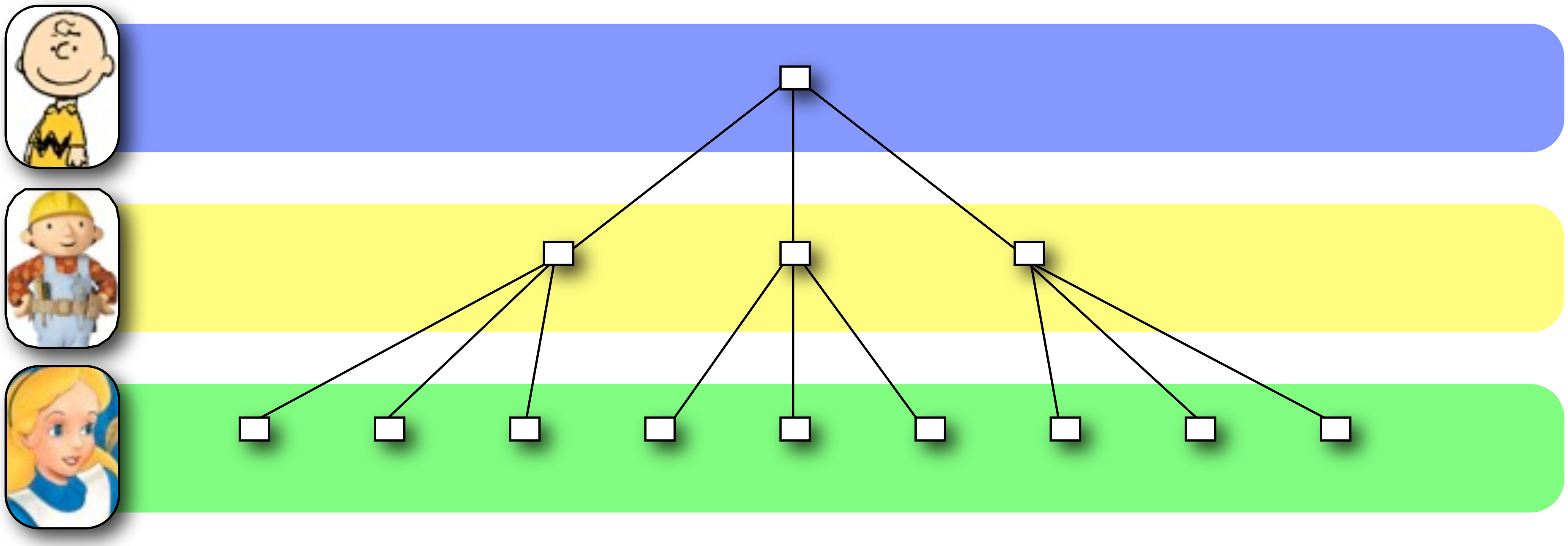
- *Thm:* With $p$-players, if $i^{th}$ player knows $f(v)$ when level$(v)=i$:

  Any $p$-round protocol requires $\Omega(t)$ communication.

# *Reduction from TPJ to Median...*

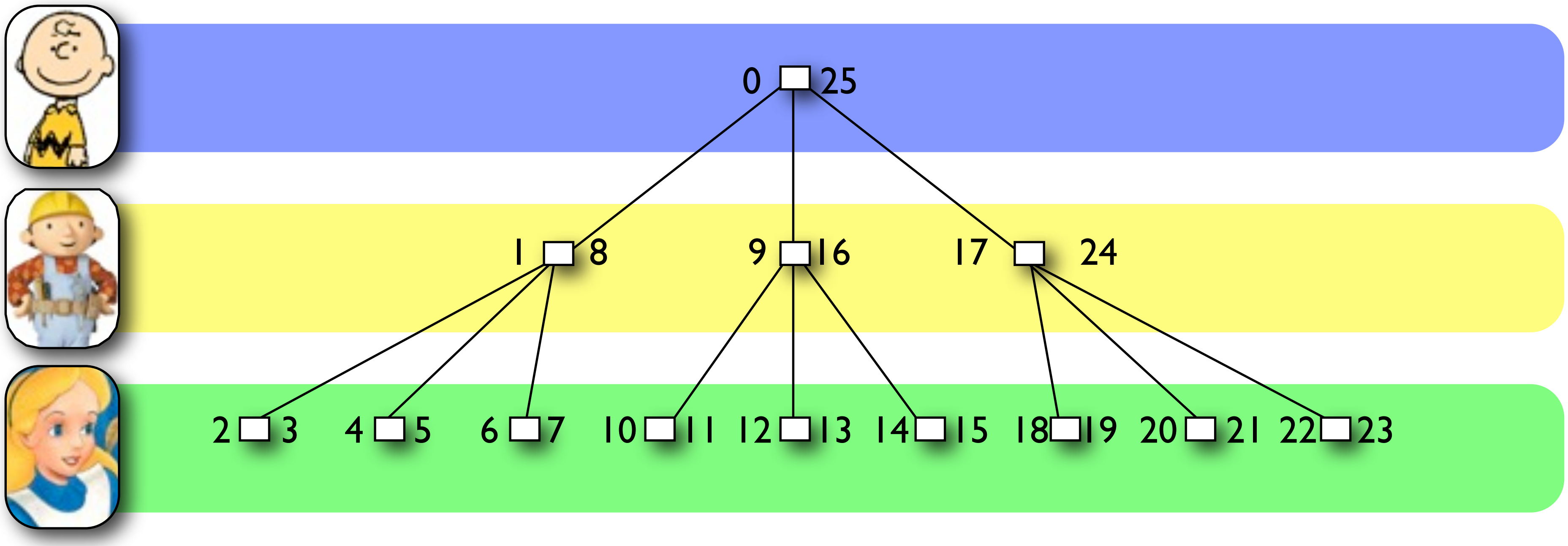


# ***Reduction from TPJ to Median...***



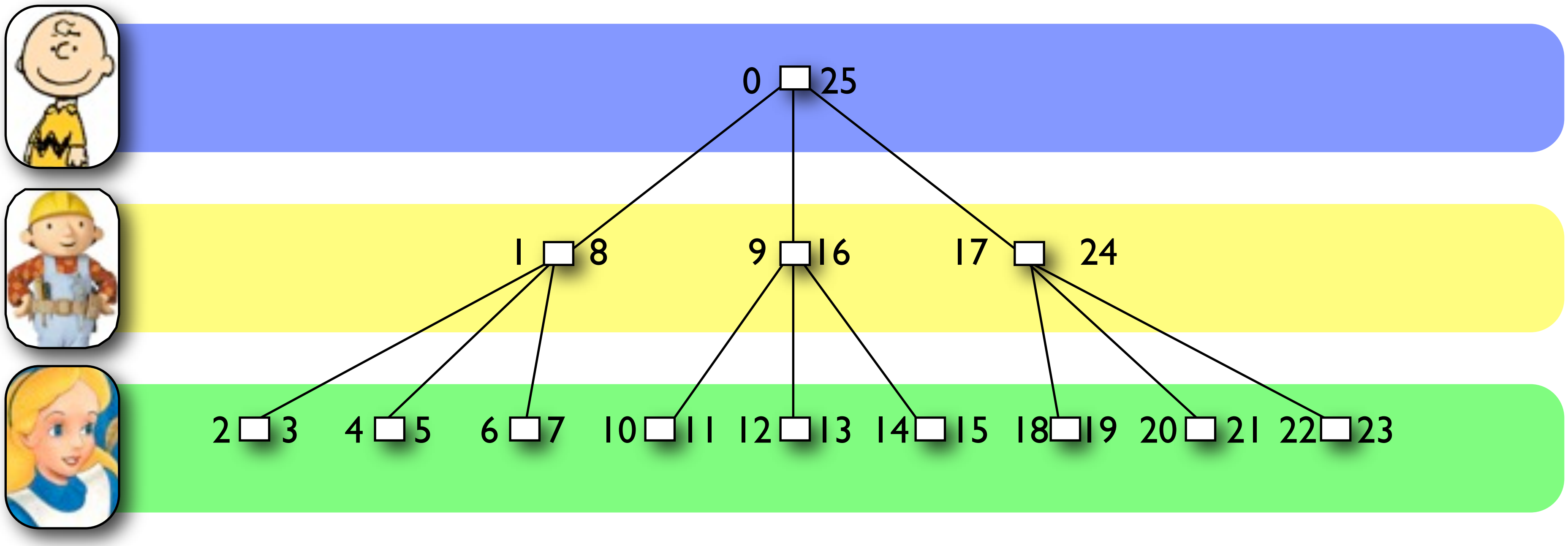
- With each node v associate two values $\alpha(v) < \beta(v)$ such that $\alpha(v) < \alpha(u) < \beta(u) < \beta(v)$ for any descendent $u$ of $v$.

# *Reduction from TPJ to Median...*



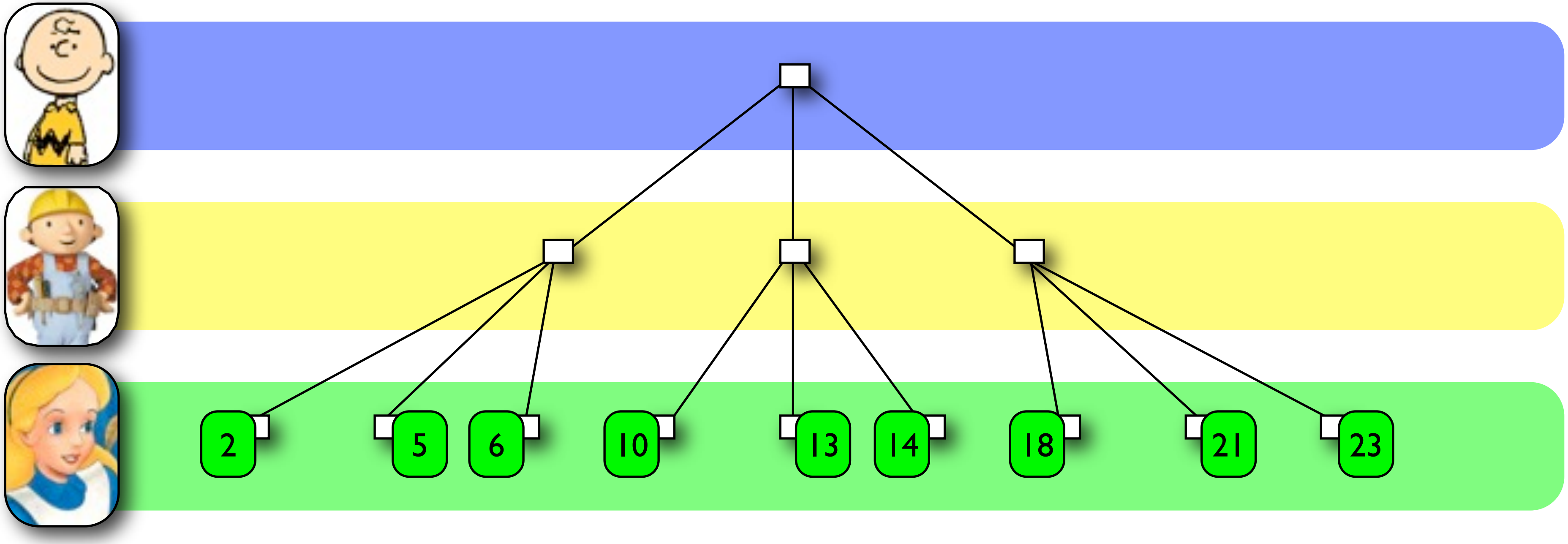

- With each node v associate two values $\alpha(v) < \beta(v)$ such that $\alpha(v) < \alpha(u) < \beta(u) < \beta(v)$ for any descendent *u* of *v*.

# Reduction from TPJ to Median...



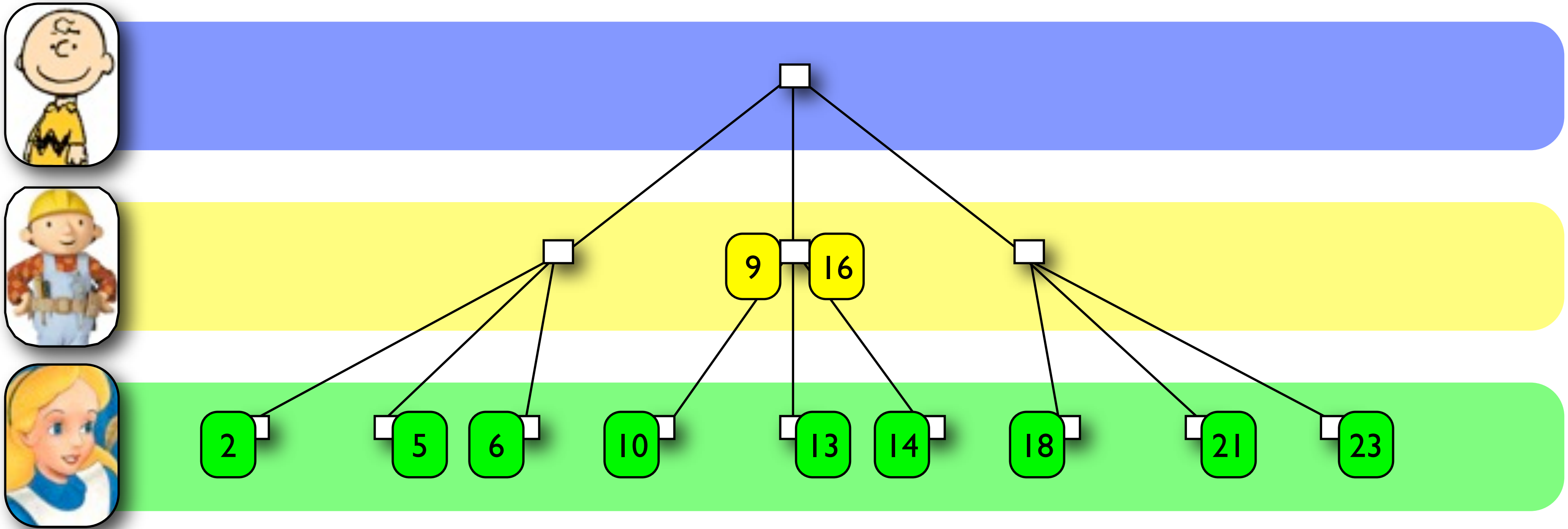

- With each node v associate two values $\alpha(v) < \beta(v)$ such that $\alpha(v) < \alpha(u) < \beta(u) < \beta(v)$ for any descendent $u$ of $v$.
- *For each node:* Generate multiple copies of $\alpha(v)$ and $\beta(v)$ such that median of values corresponds to TPJ solution.

# *Reduction from TPJ to Median...*



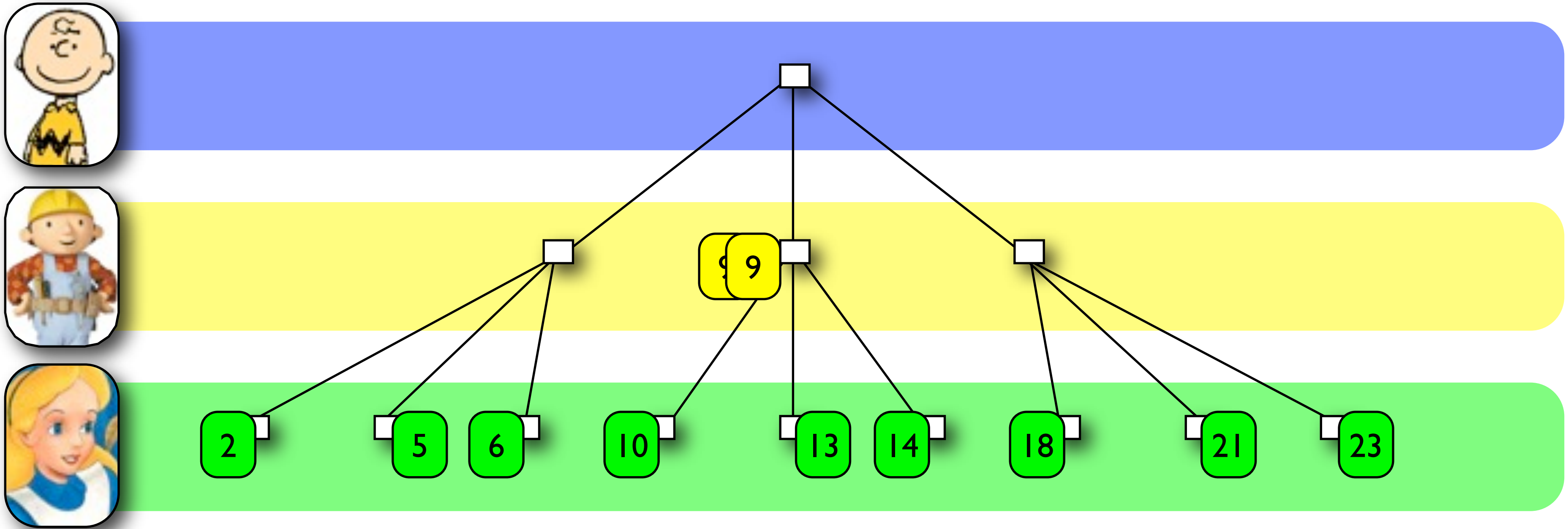

- With each node v associate two values α(*v*) < β(*v*) such that α(*v*) < α(*u*) < β(*u*) < β(*v*) for any descendent *u* of *v*.
- *For each node:* Generate multiple copies of α(*v*) and β(*v*) such that median of values corresponds to TPJ solution.

# *Reduction from TPJ to Median...*




- With each node v associate two values α(*v*) < β(*v*) such that α(*v*) < α(*u*) < β(*u*) < β(*v*) for any descendent *u* of *v*.
- *For each node:* Generate multiple copies of α(*v*) and β(*v*) such that median of values corresponds to TPJ solution.

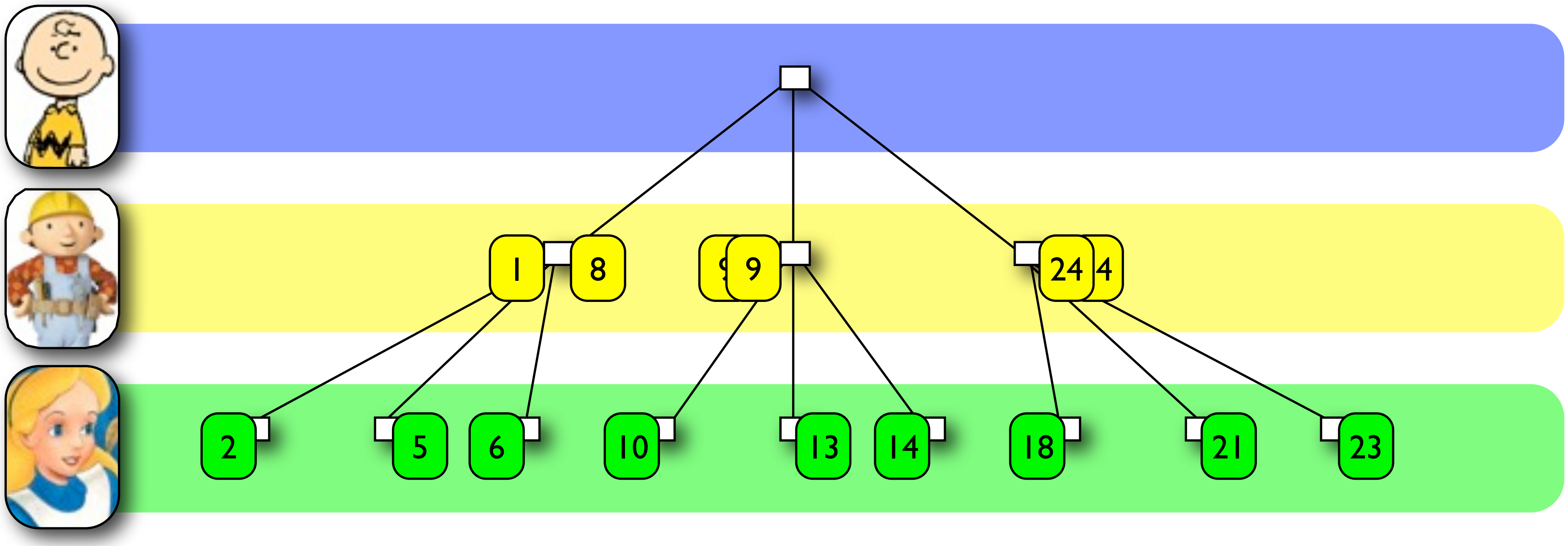
# *Reduction from TPJ to Median...*




- With each node v associate two values $\alpha(v) < \beta(v)$ such that $\alpha(v) < \alpha(u) < \beta(u) < \beta(v)$ for any descendent *u* of *v*.

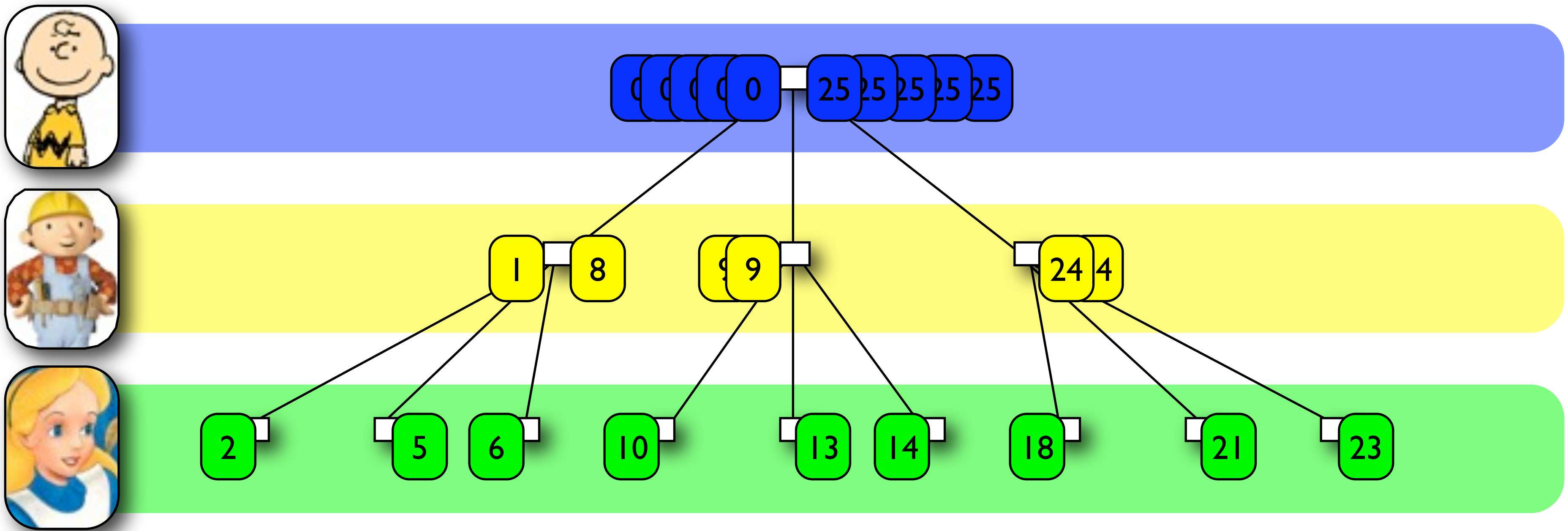
- *For each node:* Generate multiple copies of $\alpha(v)$ and $\beta(v)$ such that median of values corresponds to TPJ solution.

# Reduction from TPJ to Median…



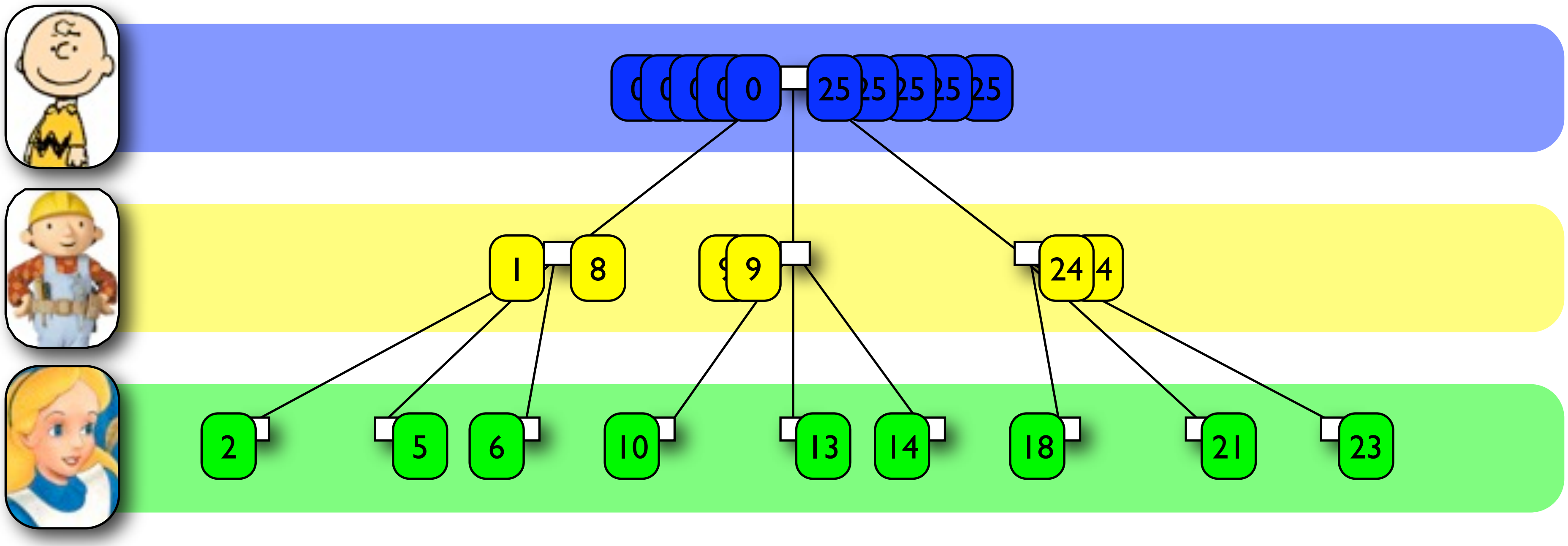

- With each node v associate two values $\alpha(v) < \beta(v)$ such that $\alpha(v) < \alpha(u) < \beta(u) < \beta(v)$ for any descendent *u* of *v*.
- *For each node:* Generate multiple copies of $\alpha(v)$ and $\beta(v)$ such that median of values corresponds to TPJ solution.

# *Reduction from TPJ to Median...*




- With each node v associate two values $\alpha(v) < \beta(v)$ such that $\alpha(v) < \alpha(u) < \beta(u) < \beta(v)$ for any descendent *u* of *v*.
- *For each node:* Generate multiple copies of $\alpha(v)$ and $\beta(v)$ such that median of values corresponds to TPJ solution.

# *Reduction from TPJ to Median...*




- With each node v associate two values $\alpha(v) < \beta(v)$ such that $\alpha(v) < \alpha(u) < \beta(u) < \beta(v)$ for any descendent *u* of *v*.
- *For each node:* Generate multiple copies of $\alpha(v)$ and $\beta(v)$ such that median of values corresponds to TPJ solution.
- Relationship between *t* and # copies determines bound.

# *Simulating Random-Partition Protocol...*

# *Simulating Random-Partition Protocol...*

- Consider node *v* where *f*(*v*) is known to Bob.

# *Simulating Random-Partition Protocol...*

α α α β β β β β

- Consider node *v* where *f*(*v*) is known to Bob.

# *Simulating Random-Partition Protocol...*



α α α β β β β β

- Consider node *v* where *f*(*v*) is known to Bob.
- *Creating Instance of Random-Partition Median Finding:*
  1) Using public coin, players determine partition of tokens and set half to α and half to β.

# *Simulating Random-Partition Protocol...*

- Consider node *v* where *f*(*v*) is known to Bob.
- *Creating Instance of Random-Partition Median Finding:*
  1) Using public coin, players determine partition of tokens and set half to α and half to β.

# *Simulating Random-Partition Protocol...*

β α β

α β α

α β

- Consider node *v* where *f*(*v*) is known to Bob.
- *Creating Instance of Random-Partition Median Finding:*
  1) Using public coin, players determine partition of tokens and set half to α and half to β.

# *Simulating Random-Partition Protocol...*

β α β

α β α

α β

- Consider node *v* where *f*(*v*) is known to Bob.
- *Creating Instance of Random-Partition Median Finding:*
  1) Using public coin, players determine partition of tokens and set half to α and half to β.

  2) Bob "fixes" balance of tokens under his control.

# *Simulating Random-Partition Protocol...*

β α β

α β β

α β

- Consider node *v* where *f*(*v*) is known to Bob.
- <u>*Creating Instance of Random-Partition Median Finding:*</u>
  1) Using public coin, players determine partition of tokens and set half to α and half to β.

  2) Bob "fixes" balance of tokens under his control.

# *Simulating Random-Partition Protocol...*

β α β    α β β    α β

- Consider node *v* where *f*(*v*) is known to Bob.
- *Creating Instance of Random-Partition Median Finding:*
  1) Using public coin, players determine partition of tokens and set half to α and half to β.

  2) Bob "fixes" balance of tokens under his control.
- *Thm:* Partition looks random if total number of tokens is greater than (max bias)$^2$. Hence, $m = \exp(2^p \lg t)$.

# *Summary*

Introduced notion of Robust Lower Bounds

Tight communication bounds for *disjointness*, *indexing*, *gap-hamming*, and improved *selection* bound.

Data streams bounds including *frequency moments*, *connectivity*, *entropy*, $F_0$, *quantile estimation*, ...

Many open problems... *Thanks!*