Kolmogorov complexity version of Slepian-Wolf coding

Marius Zimand

Towson University

STOC 2017, Montreal
When we compress correlated pieces of data,

**Distributed** Compression = **Centralized** Compression

and this is true even for a very general definition of correlation based on Kolmogorov complexity.
Distributed compression: a simple example

- Alice knows a line $\ell$; Bob knows a point $P \in \ell$; They want to send $\ell$ and $P$ to Zack.
- $\ell: 2n$ bits of information (intercept, slope in GF$[2^n]$).
- $P: 2n$ bits of information (the 2 coord. in GF$[2^n]$).
- Total information in $(\ell, P) = 3n$ bits; mutual information of $\ell$ and $P = n$ bits.
- If Alice and Bob get together, they need to send $3n$ bits. What if they compress separately?
Distributed compression: a simple example

- Alice knows a line $\ell$; Bob knows a point $P \in \ell$; They want to send $\ell$ and $P$ to Zack.
- $\ell : 2n$ bits of information (intercept, slope in GF$[2^n]$).
- $P : 2n$ bits of information (the 2 coord. in GF$[2^n]$).
- Total information in $(\ell, P) = 3n$ bits; mutual information of $\ell$ and $P = n$ bits.
- If Alice and Bob get together, they need to send $3n$ bits.

What if they compress separately?

**QUESTION 1:**
Alice can send $2n$ bits, and Bob $n$ bits. Is the geometric correlation between $\ell$ and $P$ crucial for these compression lengths?

**Ans:** No. Same is true (modulo a polylog($n$) overhead.) if Alice and Bob each have $2n$ bits of information, with mutual information $n$, in the sense of Kolmogorov complexity.
Distributed compression: a simple example

- Alice knows a line \( \ell \); Bob knows a point \( P \in \ell \); They want to send \( \ell \) and \( P \) to Zack.
- \( \ell \): 2\( n \) bits of information (intercept, slope in GF\([2^n]\)).
- \( P \): 2\( n \) bits of information (the 2 coord. in GF\([2^n]\)).
- Total information in \((\ell, P) = 3n\) bits; mutual information of \( \ell \) and \( P = n \) bits.
- If Alice and Bob get together, they need to send 3\( n \) bits.
  What if they compress separately?

**QUESTION 2:**

Can Alice send 1.5\( n \) bits, and Bob 1.5\( n \) bits? Can Alice send 1.74\( n \) bits, and Bob 1.26\( n \) bits?

**Ans:** Yes and Yes (modulo a polylog\((n)\) overhead.)
IT vs. AIT

IT (à la Shannon)
- Data is the realization of a random variable $X$.
- The model: a stochastic process generates the data.
- Amount of information in the data: $H(X)$ (Shannon entropy).

AIT (Kolmogorov complexity)
- Data is just an individual string $x$.
- There is no generative model.
- Amount of information in the data: $C(x) =$ minimum description length.
**IT vs. AIT**

### IT (à la Shannon)
- Data is the realization of a random variable $X$.
- The model: a stochastic process generates the data.
- Amount of information in the data: $H(X)$ (Shannon entropy).

### AIT (Kolmogorov complexity)
- Data is just an individual string $x$.
- There is no generative model.
- Amount of information in the data: $C(x) = \text{minimum description length}$. 

```plaintext
0000000000000000
```

Kolmogorov Slepian-Wolf

2017 4 / 17
IT vs. AIT

IT (à la Shannon)
- Data is the realization of a random variable $X$.
- The model: a stochastic process generates the data.
- Amount of information in the data: $H(X)$ (Shannon entropy).

AIT (Kolmogorov complexity)
- Data is just an individual string $x$.
- There is no generative model.
- Amount of information in the data: $C(x) = \text{minimum description length}$.
**IT vs. AIT**

**IT (à la Shannon)**
- Data is the realization of a random variable $X$.
- The model: a stochastic process generates the data.
- Amount of information in the data: $H(X)$ (Shannon entropy).

**AIT (Kolmogorov complexity)**
- Data is just an individual string $x$.
- There is no generative model.
- Amount of information in the data: $C(x)$ = minimum description length.

**Kolmogorov complexity**

Fix $U$ a universal Turing machine.

- $p$ is a description of $x$ if $U(p) = x$.
- $p$ is a description of $x$ given $y$ if $U(p, y) = x$.

$C(x) = \min \{ |p| \mid p$ is a description of $x \}.$

$C(x \mid y) = \min \{ |p| \mid p$ is a description of $x$ given $y \}.$
Distributed compression (IT view): Slepian-Wolf Theorem

- The classic Slepian-Wolf Th. is the analog of Shannon Source Coding Th. for the distributed compression of memoryless sources.
- Memoryless source: \((X_1, X_2)\) consists of \(n\) independent draws from a joint distribution \(p(b_1, b_2)\) on pair of bits.
- Encoding: \(E_1: \{0, 1\}^n \to \{0, 1\}^{n_1}, E_2: \{0, 1\}^n \to \{0, 1\}^{n_2}\).
- Decoding: \(D: \{0, 1\}^{n_1} \times \{0, 1\}^{n_2} \to \{0, 1\}^n \times \{0, 1\}^n\).
- Goal: \(D(E_1(X_1), E_2(X_2)) = (X_1, X_2)\) with probability \(1 - \epsilon\).
- It is necessary that \(n_1 + n_2 \geq H(X_1, X_2) - \epsilon n\),
  \(n_1 \geq H(X_1 | X_2) - \epsilon n\), \(n_2 \geq H(X_2 | X_1) - \epsilon n\).

Theorem (Slepian, Wolf, 1973)

*There exist encoding/decoding functions \(E_1, E_2\) and \(D\) satisfying the goal such that*

\[n_1 + n_2 \geq H(X_1, X_2) + \epsilon n, \quad n_1 \geq H(X_1 | X_2) + \epsilon n, \quad n_2 \geq H(X_2 | X_1) + \epsilon n.\]

It holds for any constant number of sources.
Slepian-Wolf Th.: Some comments

Theorem (Slepian, Wolf, 1973)

There exist encoding/decoding functions $E_1$, $E_2$ and $D$ such that $n_1 + n_2 \geq H(X_1, X_2) + \epsilon n$, $n_1 \geq H(X_1 \mid X_2) + \epsilon n$, $n_2 \geq H(X_2 \mid X_1) + \epsilon n$.

- Even if $(X_1, X_2)$ are compressed together, the sender still needs to send $\approx H(X_1, X_2)$ many bits.
- **Strength of S.-W. Th.**: distributed compression = centralized compression, for memoryless sources.
- **Shortcoming of S.-W. Th.**: Memoryless sources are very simple. The theorem has been extended to stationary and ergodic sources (Cover, 1975), which are still pretty lame.
Recall: Alice knows a line $\ell$; Bob knows a point $P \in \ell$; They want to send $\ell$ and $P$ to Zack.

There is no generative model.

Correlation can be described with the complexity profile: $C(\ell) = 2n$, $C(P) = 2n$, $C(\ell, P) = 3n$.

Is it possible to have distributed compression based only on the complexity profile?

If yes, what are the possible compression lengths?
Recall: Alice knows a line $\ell$; Bob knows a point $P \in \ell$; They want to send $\ell$ and $P$ to Zack.

There is no generative model.

Correlation can be described with the complexity profile: $C(\ell) = 2n$, $C(P) = 2n$, $C(\ell, P) = 3n$.

Is it possible to have distributed compression based only on the complexity profile?

If yes, what are the possible compression lengths?

**Necessary conditions:** Suppose we want encoding/decoding procedures so that $D(E_1(x_1), E_2(x_2)) = (x_1, x_2)$ with probability $1 - \epsilon$, for all strings $x_1, x_2$.

Then, for infinitely many $x_1, x_2$,

\[
|E_1(x_1)| + |E_2(x_2)| \geq C(x_1, x_2) + \log(1 - \epsilon) - O(1) \\
|E_1(x_1)| \geq C(x_1 \mid x_2) + \log(1 - \epsilon) - O(1) \\
|E_2(x_2)| \geq C(x_2 \mid x_1) + \log(1 - \epsilon) - O(1)
\]
MAIN RESULT: Kolmogorov complexity version of the Slepian-Wolf Theorem

There exist probabilistic poly.-time algorithms $E_1$, $E_2$ and algorithm $D$ such that for all integers $n_1, n_2$ and $n$-bit strings $x_1, x_2$, if $n_1 + n_2 \geq C(x_1, x_2)$, $n_1 \geq C(x_1 \mid x_2)$, $n_2 \geq C(x_2 \mid x_1)$, then

- $E_i$ on input $(x_i, n_i)$ outputs a string $p_i$ of length $n_i + O(\log^3 n)$, for $i = 1, 2$,
- $D$ on input $(p_1, p_2)$ outputs $(x_1, x_2)$ with probability $1 - 1/n$.

There is an analogous version for any constant number of sources.
Some comments

- Compression takes polynomial time. Decompression is slower than any computable function. This is unavoidable at this level of optimality (compression at close to minimum description length).
Some comments

- Compression takes polynomial time. Decompression is slower than any computable function. This is unavoidable at this level of optimality (compression at close to minimum description length).

- If we use time/space-bounded Kolmogorov complexity, decompression is somewhat better. For the line/point example, decompression is in linear space.
Some comments

- Compression takes polynomial time. Decompression is slower than any computable function. This is unavoidable at this level of optimality (compression at close to minimum description length).
- If we use time/space-bounded Kolmogorov complexity, decompression is somewhat better. For the line/point example, decompression is in linear space.
- Compression for individual strings is also done by Lempel-Ziv algorithms. They compress optimally for finite-state procedures. We compress at close to minimum description length.

At the high level, the proof follows the approach from a paper of Andrei Romashchenko (2005). Technical machinery is different.

The classical S.-W. Th. can be obtained from the Kolmogorov complexity version (because if $X$ is memoryless, $H(X) - c \epsilon \sqrt{n} \leq C(X) \leq H(X) + c \epsilon \sqrt{n}$ with prob. $1 - \epsilon$).

The $O(\log \frac{3}{n})$ overhead can be reduced to $O(\log n)$, but compression is no longer in polynomial time.
Some comments

- Compression takes polynomial time. Decompression is slower than any computable function. This is unavoidable at this level of optimality (compression at close to minimum description length).
- If we use time/space-bounded Kolmogorov complexity, decompression is somewhat better. For the line/point example, decompression is in linear space.
- Compression for individual strings is also done by Lempel-Ziv algorithms. They compress optimally for finite-state procedures. We compress at close to minimum description length.
- At the high level, the proof follows the approach from a paper of Andrei Romashchenko (2005). Technical machinery is different.
Some comments

- Compression takes polynomial time. Decompression is slower than any computable function. This is unavoidable at this level of optimality (compression at close to minimum description length).
- If we use time/space-bounded Kolmogorov complexity, decompression is somewhat better. For the line/point example, decompression is in linear space.
- Compression for individual strings is also done by Lempel-Ziv algorithms. They compress optimally for finite-state procedures. We compress at close to minimum description length.
- At the high level, the proof follows the approach from a paper of Andrei Romashchenko (2005). Technical machinery is different.
- The classical S.-W. Th. can be obtained from the Kolmogorov complexity version (because if $X$ is memoryless, $H(X) - c\epsilon \sqrt{n} \leq C(X) \leq H(X) + c\epsilon \sqrt{n}$ with prob. $1 - \epsilon$).
Some comments

- Compression takes polynomial time. Decompression is slower than any computable function. This is unavoidable at this level of optimality (compression at close to minimum description length).

- If we use time/space-bounded Kolmogorov complexity, decompression is somewhat better. For the line/point example, decompression is in linear space.

- Compression for individual strings is also done by Lempel-Ziv algorithms. They compress optimally for finite-state procedures. We compress at close to minimum description length.

- At the high level, the proof follows the approach from a paper of Andrei Romashchenko (2005). Technical machinery is different.

- The classical S.-W. Th. can be obtained from the Kolmogorov complexity version (because if $X$ is memoryless, $H(X) - c_\epsilon \sqrt{n} \leq C(X) \leq H(X) + c_\epsilon \sqrt{n}$ with prob. $1 - \epsilon$).

- The $O(\log^3 n)$ overhead can be reduced to $O(\log n)$, but compression is no longer in polynomial time.
Proof sketch
Graphs with the rich owner property

Bipartite graph $G$, with left degree $D$; parameters $k, \delta$;

$x$’s neighbors

$\forall B \subseteq L$, all nodes in $B$ except at most $\delta \cdot |B|$ are rich owners w.r.t. $B$
Graphs with the rich owner property

Bipartite graph $G$, with left degree $D$; parameters $k, \delta$;

$x$ is a rich owner w.r.t $B$ if

**small regime case:** $|B| \leq 2^k$

$x$ owns $(1 - \delta)$ of $N(x)$

**large regime case:** $|B| > 2^k$

at least fraction $(1 - \delta)$ of $y \in N(x)$ have

$\deg_B(y) \leq (2/\delta^2)|B|D/2^k$

("close" to avg. right degree if $|R| \approx 2^k")
Graphs with the rich owner property

Bipartite graph $G$, with left degree $D$; parameters $k, \delta$;

$x$ is a rich owner w.r.t $B$ if

**small regime case:** $|B| \leq 2^k$

$x$ owns $(1 - \delta)$ of $N(x)$

**large regime case:** $|B| > 2^k$

at least fraction $(1 - \delta)$ of $y \in N(x)$ have

$\deg_B(y) \leq (2/\delta^2)|B|D/2^k$

(“close” to avg. right degree if $|R| \approx 2^k$)

$G$ has the $(k, \delta)$ rich owner property:

$\forall B \subseteq L$,

all nodes in $B$ except at most $\delta \cdot |B|$ are rich owners w.r.t. $B$. 
Graphs with the rich owner property

Bipartite graph $G$, with left degree $D$; parameters $k, \delta$;

$x$ is a rich owner w.r.t $B$ if

**small regime case:** $|B| \leq 2^k$

$x$ owns $(1 - \delta)$ of $N(x)$

**large regime case:** $|B| > 2^k$

at least fraction $(1 - \delta)$ of $y \in N(x)$ have

\[
\deg_B(y) \leq (2/\delta^2)|B|D/2^k
\]

(“close” to avg. right degree if $|R| \approx 2^k$)

$G$ has the $(k, \delta)$ rich owner property:

$\forall B \subseteq L$, all nodes in $B$ except at most $\delta \cdot |B|$ are rich owners w.r.t. $B$
Graphs with the rich owner property

Bipartite graph $G$, with left degree $D$; parameters $k, \delta$;

$x$ is a rich owner w.r.t $B$ if

**small regime case:** $|B| \leq 2^k$
$x$ owns $(1 - \delta)$ of $N(x)$

**large regime case:** $|B| > 2^k$
at least fraction $(1 - \delta)$ of $y \in N(x)$ have
$\deg_B(y) \leq (2/\delta^2)|B|D/2^k$ 
("close" to avg. right degree if $|R| \approx 2^k$)

$G$ has the $(k, \delta)$ rich owner property:
$\forall B \subseteq L$,
all nodes in $B$ except at most $\delta \cdot |B|$ are rich owners w.r.t. $B$
Theorem (based on the (Raz-Reingold-Vadhan 2002) extractor)

There exists a poly.-time computable (uniformly in $n$, $k$ and $1/\delta$) graph with the rich owner property for parameters $(k, \delta)$ with:

- $L = \{0, 1\}^n$
- $R = \{0, 1\}^{k+O(\log^3(n/\delta))}$
- $D(\text{left degree}) = 2^{O(\log^3(n/\delta))}$
Proof sketch (cont. 1)

Suppose that compression lengths satisfy
$n_1 \geq C(x_1 \mid x_2), \ n_2 \geq C(x_2 \mid x_1),\n n_1 + n_2 \geq C(x_1, x_2).$
Proof sketch (cont. 1)

- Suppose that compression lengths satisfy
  \[ n_1 \geq C(x_1 \mid x_2), \quad n_2 \geq C(x_2 \mid x_1), \]
  \[ n_1 + n_2 \geq C(x_1, x_2). \]

- Alice uses graph \( G_1 \) with
  \( (n_1 + 1, \delta = 1/n^2) \) rich owner property,
  Bob uses graph \( G_2 \) with \( (n_2 + 1, \delta = 1/n^2) \)
  rich owner property.
Proof sketch (cont. 1)

- Suppose that compression lengths satisfy
  \[ n_1 \geq C(x_1 | x_2), \quad n_2 \geq C(x_2 | x_1), \]
  \[ n_1 + n_2 \geq C(x_1, x_2). \]

- Alice uses graph \( G_1 \) with \((n_1 + 1, \delta = 1/n^2)\) rich owner property,
  Bob uses graph \( G_2 \) with \((n_2 + 1, \delta = 1/n^2)\) rich owner property.

- **Compression:** Alice chooses \( p_1 \) a random neighbor of \( x_1 \), Bob chooses \( p_2 \) a random neighbor of \( x_2 \).
Proof sketch (cont. 1)

- Suppose that compression lengths satisfy
  \( n_1 \geq C(x_1 \mid x_2), \ n_2 \geq C(x_2 \mid x_1), \)
  \( n_1 + n_2 \geq C(x_1, x_2). \)

- Alice uses graph \( G_1 \) with
  \( (n_1 + 1, \delta = 1/n^2) \) rich owner property,
  Bob uses graph \( G_2 \) with \( (n_2 + 1, \delta = 1/n^2) \)
  rich owner property.

- **Compression:** Alice chooses \( p_1 \) a random
  neighbor of \( x_1 \), Bob chooses \( p_2 \) a random
  neighbor of \( x_2 \).

- **Decompression:** Zack needs to
  reconstruct \( x_1, x_2 \) from \( p_1, p_2 \).
Suppose that compression lengths satisfy
\[ n_1 \geq C(x_1 \mid x_2), \quad n_2 \geq C(x_2 \mid x_1), \quad n_1 + n_2 \geq C(x_1, x_2). \]

Alice uses graph \( G_1 \) with \((n_1 + 1, \delta = 1/n^2)\) rich owner property,
Bob uses graph \( G_2 \) with \((n_2 + 1, \delta = 1/n^2)\) rich owner property.

**Compression:** Alice chooses \( p_1 \) a random neighbor of \( x_1 \), Bob chooses \( p_2 \) a random neighbor of \( x_2 \).

**Decompression:** Zack needs to reconstruct \( x_1, x_2 \) from \( p_1, p_2 \).

**Idea:** For \( i = 1, 2 \), find \( B_i \) in the “small regime”, containing \( x_i \) as a rich owner.
Then with prob \( 1 - \delta \), \( x_i \) owns \( p_i \), so from \( p_i \) we can reconstruct \( x_i \).
Assume first that the decompressor knows the complexity profile $C(x_1), C(x_2), C(x_1, x_2)$. 

Case 1 (easy case): $C(x_2) \leq n$. 

Take $B_2 = \{ x | C(x) \leq C(x_2) \}$. $B_2$ is in the "small regime," $x_2$ is rich owner. So, with prob $1 - \delta$, $x_2$ owns $p_2$, so it can be reconstructed from $p_2$. 

Take $B_1 = \{ x | C(x | x_2) \leq C(x_1 | x_2) \}$. $B_1$ is in the "small regime," $x_1$ is a rich owner. So, with prob $1 - \delta$, $x_1$ owns $p_1$, so it can be reconstructed from $p_1$. 

G_1 x_1 p_1 B_1 G_2 x_2 p_2 B_2
Decompression - 1

- Assume first that the decompressor knows the complexity profile $C(x_1), C(x_2), C(x_1, x_2)$.
- **Case 1 (easy case):** $C(x_2) \leq n_2$. 
Assume first that the decompressor knows the complexity profile $C(x_1), C(x_2), C(x_1, x_2)$.

**Case 1 (easy case):** $C(x_2) \leq n_2$.

Take $B_2 = \{x \mid C(x) \leq C(x_2)\}$. $B_2$ is in the “small regime,” $x_2$ is rich owner.
Assume first that the decompressor knows the complexity profile $C(x_1), C(x_2), C(x_1, x_2)$.

**Case 1 (easy case):** $C(x_2) \leq n_2$.

Take $B_2 = \{x \mid C(x) \leq C(x_2)\}$. $B_2$ is in the “small regime,” $x_2$ is rich owner.

So, with prob $1 - \delta$, $x_2$ owns $p_2$, so it can be reconstructed from $p_2$. 
Assume first that the decompressor knows the complexity profile \( C(x_1), C(x_2), C(x_1, x_2). \)

**Case 1 (easy case):** \( C(x_2) \leq n_2. \)

Take \( B_2 = \{ x \mid C(x) \leq C(x_2) \}. \) \( B_2 \) is in the “small regime,” \( x_2 \) is rich owner.

So, with prob \( 1 - \delta \), \( x_2 \) owns \( p_2 \), so it can be reconstructed from \( p_2. \)

Take \( B_1 = \{ x \mid C(x \mid x_2) \leq C(x_1 \mid x_2) \}. \) \( B_1 \) is in the “small regime,” \( x_1 \) is a rich owner.
Assume first that the decompressor knows the complexity profile $C(x_1), C(x_2), C(x_1, x_2)$.

**Case 1 (easy case):** $C(x_2) \leq n_2$.

Take $B_2 = \{x \mid C(x) \leq C(x_2)\}$. $B_2$ is in the “small regime,” $x_2$ is rich owner.

So, with prob $1 - \delta$, $x_2$ owns $p_2$, so it can be reconstructed from $p_2$.

Take $B_1 = \{x \mid C(x \mid x_2) \leq C(x_1 \mid x_2)\}$. $B_1$ is in the “small regime,” $x_1$ is a rich owner.

So, with prob $1 - \delta$, $x_1$ owns $p_1$, so it can be reconstructed from $p_1$. 
Case 2 (hard case): $C(x_2) > n_2$. 
Case 2 (hard case): $C(x_2) > n_2$.

With some work, it can be shown that

$C(p_2) \approx n_2$ and $C(x_1 \mid p_2) \approx

C(x_1, x_2) - n_2 < (n_1 + n_2) - n_2 = n_1$. 

$G_1x_1p_1B_1G_2x_2p_2B_2Marius Zimand (Towson University) Kolmogorov Slepian-Wolf 2017 15 / 17$
Case 2 (hard case): $C(x_2) > n_2$.

With some work, it can be shown that $C(p_2) \approx n_2$ and $C(x_1 \mid p_2) \approx C(x_1, x_2) - n_2 < (n_1 + n_2) - n_2 = n_1$.

Take $B_1 = \{ x \mid C(x \mid p_2) \leq C(x_1, x_2) - n_2 \}$. $B_1$ is in the “small regime,” $x_1$ is rich owner.

Take $B_2 = \{ x \mid C(x \mid x_2) \leq C(x_1 \mid x_2) - n_2 \}$. $B_2$ is in the “small regime,” $x_2$ is a rich owner.
Case 2 (hard case): $C(x_2) > n_2$.

With some work, it can be shown that $C(p_2) \approx n_2$ and $C(x_1 | p_2) \approx C(x_1, x_2) - n_2 < (n_1 + n_2) - n_2 = n_1$.

Take

$$B_1 = \{x \mid C(x | p_2) \leq C(x_1, x_2) - n_2\}.$$  

$B_1$ is in the “small regime,” $x_1$ is rich owner.

So, with prob $1 - \delta$, $x_1$ owns $p_1$, so it can be reconstructed from $p_1$. 

\[ \]
Case 2 (hard case): $C(x_2) > n_2$.

With some work, it can be shown that $C(p_2) \approx n_2$ and $C(x_1 \mid p_2) \approx C(x_1, x_2) - n_2 < (n_1 + n_2) - n_2 = n_1$.

Take $B_1 = \{x \mid C(x \mid p_2) \leq C(x_1, x_2) - n_2\}$. $B_1$ is in the “small regime,” $x_1$ is rich owner.

So, with prob $1 - \delta$, $x_1$ owns $p_1$, so it can be reconstructed from $p_1$.

Take $B_2 = \{x \mid C(x \mid x_1) \leq C(x_1 \mid x_2)\}$. $B_2$ is in the “small regime,” $x_2$ is a rich owner.
Case 2 (hard case): $C(x_2) > n_2$.

With some work, it can be shown that $C(p_2) \approx n_2$ and $C(x_1 | p_2) \approx C(x_1, x_2) - n_2 < (n_1 + n_2) - n_2 = n_1$.

Take $B_1 = \{x | C(x | p_2) \leq C(x_1, x_2) - n_2\}$. $B_1$ is in the “small regime,” $x_1$ is rich owner.

So, with prob $1 - \delta$, $x_1$ owns $p_1$, so it can be reconstructed from $p_1$.

Take $B_2 = \{x | C(x | x_1) \leq C(x_1 | x_2)\}$. $B_2$ is in the “small regime,” $x_2$ is a rich owner.

So, with prob $1 - \delta$, $x_2$ owns $p_2$, so it can be reconstructed from $p_2$. 

\[ G_1 \] 
\[ G_2 \]
How to lift the assumption that the decompressor knows the complexity profile $C(x_1), C(x_2), C(x_1, x_2)$.
Decompression - 3

- How to lift the assumption that the decompressor knows the complexity profile $C(x_1), C(x_2), C(x_1, x_2)$.
- Try in parallel all possibilities for $C(x_1), C(x_2), C(x_1, x_2)$. We run the decompressor for each one till it finds the first neighbors of $p_1$ and $p_2$ in the corresponding $B_i$-sets (Note: some may never find any neighbors).
How to lift the assumption that the decompressor knows the complexity profile $C(x_1), C(x_2), C(x_1, x_2)$.

Try in parallel all possibilities for $C(x_1), C(x_2), C(x_1, x_2)$. We run the decompressor for each one till it finds the first neighbors of $p_1$ and $p_2$ in the corresponding $B_i$-sets (Note: some may never find any neighbors).

For the right guess of the profile, $p_1$ and $p_2$ have unique neighbors in the $B_i$-sets, and they are $x_1$ and $x_2$. Using extra hashing, we can isolate $x_1$ and $x_2$ from the strings produced by the parallel procedures with incorrect guesses. Cost of hashing: $O(\log n)$ bits, because there are $O(n^3)$ parallel procedures.
How to lift the assumption that the decompressor knows the complexity profile $C(x_1), C(x_2), C(x_1, x_2)$.

Try in parallel all possibilities for $C(x_1), C(x_2), C(x_1, x_2)$. We run the decompressor for each one till it finds the first neighbors of $p_1$ and $p_2$ in the corresponding $B_i$-sets (Note: some may never find any neighbors).

For the right guess of the profile, $p_1$ and $p_2$ have unique neighbors in the $B_i$-sets, and they are $x_1$ and $x_2$.

Using extra hashing, we can isolate $x_1$ and $x_2$ from the strings produced by the parallel procedures with incorrect guesses. Cost of hashing: $O(\log n)$ bits, because there are $O(n^3)$ parallel procedures.
Merci beaucoup.