On Row-by-Row Coding for 2-D Constraints

Ido Tal    Tuvi Etzion    Ron M. Roth

Computer Science Department
Technion, Haifa 32000, Israel
Graph Representable Constraint

1-D Constraints

- Let $G(V, E, L)$ be an edge labeled graph, $L : E \rightarrow \Sigma$.
- Example:

```
0 0 0 0 0
1 1 1 1
0 0 0 0 0
1 1 1 1
```

- $S = S(G)$ is the set of all words that are generated by paths in $G$.
- The capacity of $S$ is given by

$$\text{cap}(S) = \lim_{\ell \to \infty} \frac{1}{\ell} \cdot \log_2 |S \cap \Sigma^\ell|.$$
Parallel Encoding

An $M$-track, rate $R$, parallel encoder for a constraint $S \subseteq \Sigma^*$

- We write to $M$ tracks (columns).
An $M$-track, rate $R$, parallel encoder for a constraint $S \subseteq \Sigma^*$

- We write to $M$ tracks (columns).
An $M$-track, rate $R$, parallel encoder for a constraint $S \subseteq \Sigma^*$

- We write to $M$ tracks (columns).
- At each time slot, a symbol is written to each track (we produce a row).
- The row written is a function of the state of the encoder and of the current $M \cdot R$ information bits.
Parallel Encoding

An $M$-track, rate $R$, parallel encoder for a constraint $S \subseteq \Sigma^*$

- We write to $M$ tracks (columns).
- At each time slot, a symbol is written to each track (we produce a row).
- The row written is a function of the state of the encoder and of the current $M \cdot R$ information bits.

\[
\begin{array}{cccccc}
g_1^{(0)} & g_2^{(0)} & \cdots & g_k^{(0)} & \cdots & g_M^{(0)} \\
\end{array}
\]
Parallel Encoding

An $M$-track, rate $R$, parallel encoder for a constraint $S \subseteq \Sigma^*$

- We write to $M$ tracks (columns).
- At each time slot, a symbol is written to each track (we produce a row).
- The row written is a function of the state of the encoder and of the current $M \cdot R$ information bits.

| $g_1^{(0)}$ | $g_1^{(1)}$ | $g_2^{(0)}$ | $g_2^{(1)}$ | $g_k^{(0)}$ | $g_k^{(1)}$ | $\cdots$ | $\cdots$ | $g_M^{(0)}$ | $g_M^{(1)}$ |
Parallel Encoding

An $M$-track, rate $R$, parallel encoder for a constraint $S \subseteq \Sigma^*$

- We write to $M$ tracks (columns).
- At each time slot, a symbol is written to each track (we produce a row).
- The row written is a function of the state of the encoder and of the current $M \cdot R$ information bits.
Parallel Encoding

An $M$-track, rate $R$, parallel encoder for a constraint $S \subseteq \Sigma^*$

- We write to $M$ tracks (columns).
- At each time slot, a symbol is written to each track (we produce a row).
- The row written is a function of the state of the encoder and of the current $M \cdot R$ information bits.
- Each track must contain an element of $S$. 

| $g_1^{(0)}$ | $g_2^{(0)}$ | $\ldots$ | $g_k^{(0)}$ | $\ldots$ | $g_M^{(0)}$ |
| $g_1^{(1)}$ | $g_2^{(1)}$ | $\ldots$ | $g_k^{(1)}$ | $\ldots$ | $g_M^{(1)}$ |
| $\vdots$ | $\vdots$ | $\ddots$ | $\vdots$ | $\ddots$ | $\vdots$ |
| $g_1^{(t)}$ | $g_2^{(t)}$ | $\ldots$ | $g_k^{(t)}$ | $\ldots$ | $g_M^{(t)}$ |
Parallel Encoding

An $M$-track, rate $R$, parallel encoder for a constraint $S \subseteq \Sigma^*$

- We write to $M$ tracks (columns).
- At each time slot, a symbol is written to each track (we produce a row).
- The row written is a function of the state of the encoder and of the current $M \cdot R$ information bits.
- Each track must contain an element of $S$. 

\[
\begin{array}{ccccccc}
& g_1^{(0)} & g_2^{(0)} & \cdots & g_k^{(0)} & \cdots & g_M^{(0)} \\
& (1) & (1) & & (1) & & (1) \\
g_1 & g_2 & \cdots & g_k & \cdots & g_M \\
\vdots & \vdots & & \vdots & & \vdots \\
g_1^{(t)} & g_2^{(t)} & \cdots & g_k^{(t)} & \cdots & g_M^{(t)} \\
\in S
\end{array}
\]
### Parallel Decoding

**An $M$-track $(m, a)$-SBD decoder**

- At time slot $t$, the respective input bits are recovered from rows $t - m, t - m + 1, \ldots, t + a$

\[
\begin{array}{cccccc}
\vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\
\begin{array}{cccc}
(\text{t} - m)^{(t-m)} & (\text{t} - m)^{(t-m)} & \cdots & (\text{t} - m)^{(t-m)} & \cdots & (\text{t} - m)^{(t-m)} \\
g_1 & g_2 & \cdots & g_k & \cdots & g_M \\
\vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\
(\text{t})^{(t)} & (\text{t})^{(t)} & \cdots & (\text{t})^{(t)} & \cdots & (\text{t})^{(t)} \\
g_1 & g_2 & \cdots & g_k & \cdots & g_M \\
\vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\
(\text{t} + a)^{(t+a)} & (\text{t} + a)^{(t+a)} & \cdots & (\text{t} + a)^{(t+a)} & \cdots & (\text{t} + a)^{(t+a)} \\
g_1 & g_2 & \cdots & g_k & \cdots & g_M \\
\vdots & \vdots & \cdots & \vdots & \cdots & \vdots
\end{array}
\end{array}
\]
Main Results

Main results of our parallel encoding/decoding scheme

- We approach $\text{cap}(S(G))$ as the number of tracks, $M$, grows.
- The vertical size of the decoding window is constant in $M$.
- For a constant graph size, encoding and decoding time is $O(M \log^2 M \log \log M)$.
Consider as an example the square constraint [WeeksBlahut98]:

The elements are all the binary arrays in which an entry may equal ‘1’ only if all its eight neighbors are ‘0’.

\[
\begin{array}{cccc}
1 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 \\
\end{array}
\]

A graph which produces all \( \ell \times 4 \) arrays that satisfy this constraint:

Thus, if the number of columns is reasonably small, we can reduce our 2-D constraint to a 1-D constraint.
2-D Constraints (Cont.)

- We use this as follows:
- Partition the 2-D array into two alternating type strips:

```
0 0 1 0 | 0 1 0 1 0 0 0 0 1
1 0 0 0 | 0 0 0 0 0 0 1 0 0
0 0 0 1 | 0 0 1 0 0 0 0 0 0
1 0 0 0 | 0 0 0 0 1 0 0 0 0
```
2-D Constraints (Cont.)

- We use this as follows:
- Partition the 2-D array into two alternating type strips:
  - $M$ data strips of width 4.

\[
\begin{array}{cccccccccccccccc}
0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\]
We use this as follows:

Partition the 2-D array into two alternating type strips:
- \(M\) data strips of width 4.
- \(M - 1\) merging strips of width 1.

\[
\begin{array}{cccccccccccc}
0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\
\end{array}
\]
2-D Constraints (Cont.)

- We use this as follows:
- Partition the 2-D array into two alternating type strips:
  - \( M \) data strips of width 4.
  - \( M - 1 \) merging strips of width 1.

\[
\begin{array}{cccc|cccc|cccc}
0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\]

- Think of each of the data strips as a track.
We use this as follows:

Partition the 2-D array into two alternating type strips:

- $M$ data strips of width 4.
- $M - 1$ merging strips of width 1.

Think of each of the data strips as a track.
Fill all the merging strips with ‘0’ bits.
2-D Constraints (Cont.)

- We use this as follows:
- Partition the 2-D array into two alternating type strips:
  - $M$ data strips of width 4.
  - $M - 1$ merging strips of width 1.

- Think of each of the data strips as a track.
- Fill all the merging strips with ‘0’ bits.
- We may now use an $M$-track parallel encoder in order to encode information to the array in a row-by-row manner.
2-D Constraints (Cont.)

- We use this as follows:
- Partition the 2-D array into two alternating type strips:
  - $M$ data strips of width 4.
  - $M - 1$ merging strips of width 1.

```
0 0 1 0 0 1 0 1 0 0 0 0 0 1
```

- Think of each of the data strips as a track.
- Fill all the merging strips with ‘0’ bits.
- We may now use an $M$-track parallel encoder in order to encode information to the array in a row-by-row manner.
2-D Constraints (Cont.)

- We use this as follows:
- Partition the 2-D array into two alternating type strips:
  - \( M \) data strips of width 4.
  - \( M - 1 \) merging strips of width 1.

\[
\begin{array}{ccccccc}
0 & 0 & 1 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 \\
\end{array}
\]

- Think of each of the data strips as a track.
- Fill all the merging strips with ‘0’ bits.
- We may now use an \textcolor{red}{M-track parallel encoder} in order to encode information to the array in a row-by-row manner.
2-D Constraints (Cont.)

- We use this as follows:
- Partition the 2-D array into two alternating type strips:
  - $M$ data strips of width 4.
  - $M - 1$ merging strips of width 1.

```
0 0 1 0 | 0 1 0 1 0 | 0 0 0 0 1
1 0 0 0 | 0 0 0 0 0 | 0 1 0 0 0
0 0 0 1 | 0 0 1 0 0 | 0 0 0 0 0
```

- Think of each of the data strips as a track.
- Fill all the merging strips with ‘0’ bits.
- We may now use an $M$-track parallel encoder in order to encode information to the array in a row-by-row manner.
2-D Constraints (Cont.)

- We use this as follows:
- Partition the 2-D array into two alternating type strips:
  - $M$ data strips of width 4.
  - $M - 1$ merging strips of width 1.

<table>
<thead>
<tr>
<th>0</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

- Think of each of the data strips as a track.
- Fill all the merging strips with ‘0’ bits.
- We may now use an $M$-track parallel encoder in order to encode information to the array in a row-by-row manner.
We use this as follows:

- Partition the 2-D array into two alternating type strips:
  - $M$ data strips of width 4.
  - $M - 1$ merging strips of width 1.

```
0 0 1 0 | 0 1 0 1 0 0 0 0 1
1 0 0 0 | 0 0 0 0 0 1 0 0 0
0 0 0 1 | 0 0 1 0 0 0 0 0 0
1 0 0 0 | 0 0 0 0 1 0 0 0 0
```

- Think of each of the data strips as a track.
- Fill all the merging strips with ‘0’ bits.
- We may now use an $M$-track parallel encoder in order to encode information to the array in a row-by-row manner.
- Enlarging the width of the data strips gives a better encoding rate, at the expense of the encoder’s complexity.
The description of our $M$-track parallel encoder for $S = S(G)$ is defined by its respective multiplicity matrix $D$:

1. Let $A_G = (a_{i,j})$ be the adjacency matrix of $G$.
2. A nonnegative integer matrix $D = (d_{i,j})_{i,j \in V}$ is a valid multiplicity matrix with respect to $G$ and $M$ if

\[
1 \cdot D \cdot 1^T \leq M, \quad (1)
\]
\[
1 \cdot D = 1 \cdot D^T, \quad \text{and} \quad (2)
\]
\[
d_{i,j} > 0 \text{ only if } a_{i,j} > 0. \quad (3)
\]

Our aim is to find a multiplicity matrix such that the respective encoder has rate close to $\text{cap}(S)$. 
For the sake of exposition, assume that $G$ does not contain parallel edges.

Let $\mathcal{P}_D : E \to [0, 1]$ be the Markov chain on $G$ defined as follows:

$$\mathcal{P}_D(i \to j) = d_{i,j} / (1 \cdot D \cdot 1^T).$$

Since we required that $1 \cdot D = 1 \cdot D^T$, we have that $\mathcal{P}_D$ is stationary.

Essentially, the encoder “mimics” $\mathcal{P}_D$.

The rate of the encoder approaches $\text{cap}(S)$ when $1 \cdot D \cdot 1^T$ approaches $M$ and $\mathcal{P}_D$ is close to the maxentropic Markov chain on $G$. 
Encoder Example

\[
\begin{align*}
A_G &= \begin{pmatrix}
1 & 1 & 0 \\
1 & 0 & 1 \\
1 & 0 & 0 \\
\end{pmatrix} \\
D &= \begin{pmatrix}
4 & 3 & 0 \\
2 & 0 & 1 \\
1 & 0 & 0 \\
\end{pmatrix}
\end{align*}
\]

\[M = 12\]
Encoder Example

\[ A_G = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \quad D = \begin{pmatrix} 4 & 3 & 0 \\ 2 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \]

\[ 1 \cdot D \cdot 1^T = 11 \]

\[ M = 12 \]
Encoder Example

\[
A_G = \begin{pmatrix}
1 & 1 & 0 \\
1 & 0 & 1 \\
1 & 0 & 0
\end{pmatrix}
\quad
D = \begin{pmatrix}
4 & 3 & 0 \\
2 & 0 & 1 \\
1 & 0 & 0
\end{pmatrix}
\]

\[1 \cdot D \cdot 1^T = 11\]

\[M = 12\]
Encoder Example

$A_G = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$  
$D = \begin{pmatrix} 4 & 3 & 0 \\ 2 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$

- $1 \cdot D \cdot 1^T = 11$
- $1 \cdot D^T = (7, 3, 1)$

$M = 12$
Encoder Example

\[ A_G = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \quad D = \begin{pmatrix} 4 & 3 & 0 \\ 2 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \]

- \[ 1 \cdot D \cdot 1^T = 11 \]
- \[ 1 \cdot D^T = (7, 3, 1) \]

\[ M = 12 \]
Encoder Example

\[ A_G = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \quad D = \begin{pmatrix} 4 & 3 & 0 \\ 2 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \]

- \(1 \cdot D \cdot 1^T = 11\)
- \(1 \cdot D^T = (7, 3, 1)\)

\[ M = 12 \]
Encoder Example

$$\begin{align*}
A_G &= \begin{pmatrix}
1 & 1 & 0 \\
1 & 0 & 1 \\
1 & 0 & 0
\end{pmatrix} \\
D &= \begin{pmatrix}
4 & 3 & 0 \\
2 & 0 & 1 \\
1 & 0 & 0
\end{pmatrix}
\end{align*}$$

- $1 \cdot D \cdot 1^T = 11$
- $1 \cdot D^T = (7, 3, 1)$

$M = 12$
Encoder Example

\[ A_G = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \quad D = \begin{pmatrix} 4 & 3 & 0 \\ 2 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \]

- \( 1 \cdot D \cdot 1^T = 11 \)
- \( 1 \cdot D^T = (7, 3, 1) \)

\[ M = 12 \]
Encoder Example

$A_G = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$ \quad $D = \begin{pmatrix} 4 & 3 & 0 \\ 2 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$

- $1 \cdot D \cdot 1^T = 11$
- $1 \cdot D^T = (7, 3, 1)$

$M = 12$
Encoder Example

\[ A_G = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \quad D = \begin{pmatrix} 4 & 3 & 0 \\ 2 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \]

- \( 1 \cdot D \cdot 1^T = 11 \)
- \( 1 \cdot D^T = (7, 3, 1) \)

\[
\begin{array}{cccccccccccc}
\alpha & \alpha & \alpha & \alpha & \alpha & \alpha & \alpha & \beta & \beta & \beta & \gamma & \alpha \\
\downarrow a & \downarrow a & \downarrow a & \downarrow a & \downarrow b & \downarrow b & \downarrow b & \downarrow c & \downarrow c & \downarrow d & \downarrow e & \alpha \\
\alpha & \alpha & \alpha & \alpha & \beta & \beta & \beta & \alpha & \alpha & \gamma & \alpha \\
\end{array}
\]

\( M = 12 \)
Encoder Example

\[\begin{align*}
A_G &= \begin{pmatrix}
1 & 1 & 0 \\
1 & 0 & 1 \\
1 & 0 & 0
\end{pmatrix} \\
D &= \begin{pmatrix}
4 & 3 & 0 \\
2 & 0 & 1 \\
1 & 0 & 0
\end{pmatrix}
\end{align*}\]

- \(1 \cdot D \cdot 1^T = 11\)
- \(1 \cdot D^T = (7, 3, 1) = 1 \cdot D\)

\[\begin{array}{cccccccccccc}
\alpha & \alpha & \alpha & \alpha & \alpha & \alpha & \alpha & \beta & \beta & \beta & \beta & \gamma \\
\downarrow & \downarrow & \downarrow & \downarrow & b & b & b & c & c & d & e & \alpha
\end{array}\]

\(M = 12\)
Encoder Example

\[ AG = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \quad D = \begin{pmatrix} 4 & 3 & 0 \\ 2 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \]

- \( 1 \cdot D \cdot 1^T = 11 \)
- \( 1 \cdot D^T = (7, 3, 1) = 1 \cdot D \)

\[ M = 12 \]
### Encoder Example

![Graph Diagram]

\[ A_G = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \quad D = \begin{pmatrix} 4 & 3 & 0 \\ 2 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \]

- \( 1 \cdot D \cdot 1^T = 11 \)
- \( 1 \cdot D^T = (7, 3, 1) = 1 \cdot D \)

\[ M = 12 \]
Encoder Example

\[ A_G = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \quad D = \begin{pmatrix} 4 & 3 & 0 \\ 2 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \]

- \( 1 \cdot D \cdot 1^T = 11 \)
- \( 1 \cdot D^T = (7, 3, 1) = 1 \cdot D \)

\( M = 12 \)
Encoder Example

\[
\begin{align*}
A_G &= \begin{pmatrix}
1 & 1 & 0 \\
1 & 0 & 1 \\
1 & 0 & 0
\end{pmatrix} &
D &= \begin{pmatrix}
4 & 3 & 0 \\
2 & 0 & 1 \\
1 & 0 & 0
\end{pmatrix}
\end{align*}
\]

- \(1 \cdot D \cdot 1^T = 11\)
- \(1 \cdot D^T = (7, 3, 1) = 1 \cdot D\)

\[
M = 12
\]
Encoder Example

\[ A_G = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \quad D = \begin{pmatrix} 4 & 3 & 0 \\ 2 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \]

- \( 1 \cdot D \cdot 1^T = 11 \)
- \( 1 \cdot D^T = (7, 3, 1) = 1 \cdot D \)

\[ M = 12 \]
**Encoder Example**

\[ AG = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \quad D = \begin{pmatrix} 4 & 3 & 0 \\ 2 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \]

- \( 1 \cdot D \cdot 1^T = 11 \)
- \( 1 \cdot D^T = (7, 3, 1) = 1 \cdot D \)
- \( \Delta = (\prod_{i \in V} r_i!) / (\prod_{i,j \in V} d_{i,j}! \cdot a_{i,j}^{-d_{i,j}}) \)

\[ M = 12 \]
Maxentropic Distribution

- Let $\mathcal{P}^*: E \to [0, 1]$ be the maxentropic stationary Markov chain on $G$.
- For an as yet unspecified $M'$, define:

$$P = (p_{i,j}), \quad p_{i,j} = M'\mathcal{P}^*(i \to j).$$
Halevy and Roth’s Solution

- **If**, when taking $M' = M$, all the entries of $P$ were integers, then we could take $D = P$.

- We would have $R(D) = \frac{\log_2 \Delta}{M} \xrightarrow{M \to \infty} \text{cap}(S(G))$.

- Solution [HalevyRoth]: Perturb a related matrix such that its entries are rational, and take $M = M'$ large enough.

- Problem: $M$ unrealistically large.
Take $M' = M - \lfloor |V| \text{diam}(G)/2 \rfloor$.

We say that an integer matrix $\tilde{P} = (\tilde{p}_{i,j})$ is a good quantization of $P = (p_{i,j})$ if

\begin{align*}
M' &= \sum_{i,j \in V} p_{i,j} = \sum_{i,j \in V} \tilde{p}_{i,j}, \\
\lfloor \sum_{j \in V} p_{i,j} \rfloor &\leq \sum_{j \in V} \tilde{p}_{i,j} \leq \lceil \sum_{j \in V} p_{i,j} \rceil, \\
|p_{i,j}| &\leq \tilde{p}_{i,j} \leq \lceil p_{i,j} \rceil, \quad \text{and—} \\
\lfloor \sum_{i \in V} p_{i,j} \rfloor &\leq \sum_{i \in V} \tilde{p}_{i,j} \leq \lceil \sum_{i \in V} p_{i,j} \rceil.
\end{align*}
Lemma

There exists a matrix $\tilde{P}$ which is a good quantization of $P$. Furthermore, such a matrix can be found by an efficient algorithm.

Partial Proof.

- Formulate the above as an integer flow problem.
- A fractional solution exists.
- Thus, an integer solution exists.
Example

\[ M = 12 \]

\[ A_G = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \]
Example

\[ M = 12 \quad M' = 9 \]

\[ A_G = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \]
Example

\[ M = 12 \quad M' = 9 \]

\[
A_G = \begin{pmatrix}
1 & 1 & 0 \\
1 & 0 & 1 \\
1 & 0 & 0
\end{pmatrix}
\]

\[
P = \begin{pmatrix}
3.05 & 2.53 & 0 \\
1.64 & 0 & 0.89 \\
0.89 & 0 & 0
\end{pmatrix}
\]

\[
\tilde{P} = \begin{pmatrix}
4 & 2 & 0 \\
2 & 0 & 1 \\
0 & 0 & 0
\end{pmatrix}
\]
Example

\[ M = 12 \quad M' = 9 \]

\[ A_G = \begin{pmatrix}
1 & 1 & 0 \\
1 & 0 & 1 \\
1 & 0 & 0
\end{pmatrix} \]

\[ P = \begin{pmatrix}
3.05 & 2.53 & 0 \\
1.64 & 0 & 0.89 \\
0.89 & 0 & 0
\end{pmatrix} \]

\[ \tilde{P} = \begin{pmatrix}
4 & 2 & 0 \\
2 & 0 & 1 \\
0 & 0 & 0
\end{pmatrix} \]
Example

\[ M = 12 \quad M' = 9 \]

\[ A_G = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \]

\[ P = \begin{pmatrix} 3.05 & 2.53 & 0 \\ 1.64 & 0 & 0.89 \\ 0.89 & 0 & 0 \end{pmatrix} \]

\[ \tilde{P} = \begin{pmatrix} 4 & 2 & 0 \\ 2 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \]
Example

\[ M = 12 \quad M' = 9 \]

\[ A_G = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \]

\[ P = \begin{pmatrix} 3.05 & 2.53 & 0 \\ 1.64 & 0 & 0.89 \\ 0.89 & 0 & 0 \end{pmatrix} \]

\[ \tilde{P} = \begin{pmatrix} 4 & 2 & 0 \\ 2 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \]
Example

\[ M = 12 \quad M' = 9 \]

\[ A_G = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \]

\[ P = \begin{pmatrix} 3.05 & 2.53 & 0 \\ 1.64 & 0 & 0.89 \\ 0.89 & 0 & 0 \end{pmatrix} \]

\[ \tilde{P} = \begin{pmatrix} 4 & 2 & 0 \\ 2 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \]
\( \tilde{P} \) is an integer matrix (a good quantization of \( P \)).

However, \( \tilde{P} \) is generally not a valid multiplicity matrix:

- We might have that \( 1 \cdot (\tilde{P})^T \neq 1 \cdot \tilde{P} \) (the respective Markov chain is not stationary).
Theorem

Let \( \tilde{P} = (\tilde{p}_{i,j}) \) be a good quantization of \( P \). There exists a multiplicity matrix \( D = (d_{i,j}) \) with respect to \( G \) and \( M \), such that

1. \( d_{i,j} \geq \tilde{p}_{i,j} \) for all \( i, j \in V \), and—
2. \( M' - \lfloor |V| \text{diam}(G)/2 \rfloor \leq 1 \cdot D \cdot 1^T \leq M \)

(\( M' = M - \lfloor |V| \text{diam}(G)/2 \rfloor \)). Moreover, the matrix \( D \) can be found by an efficient algorithm.

Proof makes use of network flow as well.
Main Theorem

Theorem

Let $G$ be a deterministic graph with memory $m$. For $M$ sufficiently large, one can efficiently construct an $M$-track $(m, 0)$-SBD parallel encoder for $S = S(G)$ at a rate $R$ such that

$$R \geq \text{cap}(S(G)) \left(1 - \frac{|V| \text{diam}(G)}{2M}\right)$$

$$- O \left(\frac{|V|^2 \log (M \cdot a_{\text{max}}/a_{\text{min}})}{M - |V| \text{diam}(G)/2}\right),$$

where $a_{\text{min}}$ (respectively, $a_{\text{max}}$) is the smallest (respectively, largest) nonzero entry in $A_G$.

Proof makes use of the multiplicity matrix guaranteed by previous theorem.