Graph and String Parameters: Connections Between Pathwidth, Cutwidth and the Locality Number

Katrin Casel$^1$, Joel D. Day$^2$, Pamela Fleischmann$^3$, Tomasz Kociumaka$^4$, Florin Manea$^3$, Markus L. Schmid$^5$

$^1$ HPI, University of Potsdam, Germany
$^2$ Loughborough University, UK
$^3$ Kiel University, Germany
$^4$ University of Warsaw, Poland, and Bar-Ilan University, Israel
$^5$ Trier University, Germany

Theorietag 2019 – Marburg
A Solitaire Game on Strings

The Game

Given: String $\alpha$ over (finite) alphabet $\Sigma = \{a_1, a_2, \ldots, a_n\}$.

Objective: Mark all symbols $a_1, a_2, \ldots, a_n$ in some order (all occ. of the same symbol in parallel), such that there are only few contiguous blocks of marked symbols in the word.
A Solitaire Game on Strings

The Game

Given: String $\alpha$ over (finite) alphabet $\Sigma = \{a_1, a_2, \ldots, a_n\}$.

Objective: Mark all symbols $a_1, a_2, \ldots, a_n$ in some order (all occ. of the same symbol in parallel), such that there are only few contiguous blocks of marked symbols in the word.

Example

a d a b a d b d a e c b c b

marking sequence:

marked blocks:

maximum number of marked blocks:
A Solitaire Game on Strings

The Game

Given: String $\alpha$ over (finite) alphabet $\Sigma = \{a_1, a_2, \ldots, a_n\}$.

Objective: Mark all symbols $a_1, a_2, \ldots, a_n$ in some order (all occ. of the same symbol in parallel), such that there are only few contiguous blocks of marked symbols in the word.

Example

$$a \ d \ a \ b \ a \ d \ b \ d \ a \ e \ c \ b \ c \ b$$

marking sequence: $b$

marked blocks: 4
maximum number of marked blocks: 4
A Solitaire Game on Strings

The Game

Given: String \( \alpha \) over (finite) alphabet \( \Sigma = \{a_1, a_2, \ldots, a_n\} \).

Objective: Mark all symbols \( a_1, a_2, \ldots, a_n \) in some order (all occ. of the same symbol in parallel), such that there are only few contiguous blocks of marked symbols in the word.

Example

a d a b a d b d a e c b c b

marking sequence: b, c

marked blocks: 3
maximum number of marked blocks: 4
The Game

Given: String $\alpha$ over (finite) alphabet $\Sigma = \{a_1, a_2, \ldots, a_n\}$.

Objective: Mark all symbols $a_1, a_2, \ldots, a_n$ in some order (all occ. of the same symbol in parallel), such that there are only few contiguous blocks of marked symbols in the word.

Example

$\text{a d a b a d b d a e c b c b}$

marking sequence: $b, c, e$

marked blocks: 3
maximum number of marked blocks: 4
A Solitaire Game on Strings

The Game

Given: String $\alpha$ over (finite) alphabet $\Sigma = \{a_1, a_2, \ldots, a_n\}$.

Objective: Mark all symbols $a_1, a_2, \ldots, a_n$ in some order (all occ. of the same symbol in parallel), such that there are only few contiguous blocks of marked symbols in the word.

Example

```
 a d a b a d b d a e c b c b
```

marking sequence: b, c, e, d

marked blocks: 4
maximum number of marked blocks: 4
A Solitaire Game on Strings

The Game

Given: String $\alpha$ over (finite) alphabet $\Sigma = \{a_1, a_2, \ldots, a_n\}$.

Objective: Mark all symbols $a_1, a_2, \ldots, a_n$ in some order (all occ. of the same symbol in parallel), such that there are only few contiguous blocks of marked symbols in the word.

Example

```
adabaadbdaecbcb
```

marking sequence: b, c, e, d, a

marked blocks: 1
maximum number of marked blocks: 4
A Solitaire Game on Strings

The Game

Given: String $\alpha$ over (finite) alphabet $\Sigma = \{a_1, a_2, \ldots, a_n\}$.

Objective: Mark all symbols $a_1, a_2, \ldots, a_n$ in some order (all occ. of the same symbol in parallel), such that there are only few contiguous blocks of marked symbols in the word.

Example

a d a b a d b d a e c b c b

marking sequence:

marked blocks:

maximum number of marked blocks:
A Solitaire Game on Strings

The Game

Given: String $\alpha$ over (finite) alphabet $\Sigma = \{a_1, a_2, \ldots, a_n\}$.

Objective: Mark all symbols $a_1, a_2, \ldots, a_n$ in some order (all occ. of the same symbol in parallel), such that there are only few contiguous blocks of marked symbols in the word.

Example

| a | d | a | b | a | d | b | d | a | e | c | b | c | b |

marking sequence: d

marked blocks: 3

maximum number of marked blocks: 3
The Game

Given: String $\alpha$ over (finite) alphabet $\Sigma = \{a_1, a_2, \ldots, a_n\}$.

Objective: Mark all symbols $a_1, a_2, \ldots, a_n$ in some order (all occ. of the same symbol in parallel), such that there are only few contiguous blocks of marked symbols in the word.

Example

$a \ d \ a \ b \ a \ d \ b \ d \ a \ e \ c \ b \ c \ b$

marking sequence: $d$, $a$

marked blocks: $3$

maximum number of marked blocks: $3$
A Solitaire Game on Strings

The Game

Given: String $\alpha$ over (finite) alphabet $\sum = \{a_1, a_2, \ldots, a_n\}$.

Objective: Mark all symbols $a_1, a_2, \ldots, a_n$ in some order (all occ. of the same symbol in parallel), such that there are only few contiguous blocks of marked symbols in the word.

Example

\[ a \ d \ a \ b \ a \ d \ b \ d \ a \ e \ c \ b \ c \ b \]

marking sequence: \(d, a, b\)

marked blocks: 3

maximum number of marked blocks: 3
A Solitaire Game on Strings

The Game

Given: String $\alpha$ over (finite) alphabet $\Sigma = \{a_1, a_2, \ldots, a_n\}$.

Objective: Mark all symbols $a_1, a_2, \ldots, a_n$ in some order (all occ. of the same symbol in parallel), such that there are only few contiguous blocks of marked symbols in the word.

Example

```
adabaadbdaeccbcb
```
marking sequence: d, a, b, c

marked blocks: 2
maximum number of marked blocks: 3
A Solitaire Game on Strings

The Game

Given: String $\alpha$ over (finite) alphabet $\Sigma = \{a_1, a_2, \ldots, a_n\}$.

Objective: Mark all symbols $a_1, a_2, \ldots, a_n$ in some order (all occ. of the same symbol in parallel), such that there are only few contiguous blocks of marked symbols in the word.

Example

$\text{a d a b a d b d a e c b c b}$

marking sequence: d, a, b, c, e

marked blocks: 1
maximum number of marked blocks: 3
The Locality Number

Let $X = \{x_1, x_2, \ldots, x_n\}$ be a finite alphabet.

**Marking Sequence**

Any ordered list $\sigma$ of the symbols from $X$ (or, equivalently, a bijection $\sigma : \{1, 2, \ldots, |X|\} \rightarrow X$) is a **marking sequence**.
### The Locality Number

Let $X = \{x_1, x_2, \ldots, x_n\}$ be a finite alphabet.

#### Marking Sequence

Any ordered list $\sigma$ of the symbols from $X$ (or, equivalently, a bijection $\sigma : \{1, 2, \ldots, |X|\} \rightarrow X$) is a **marking sequence**.

#### Marking Number

The **marking number** $\pi_\sigma(\alpha)$ (of $\sigma$ with respect to $\alpha$) is the maximum number of marked blocks obtained while marking $\alpha$ according to $\sigma$. 
The Locality Number

Let $X = \{x_1, x_2, \ldots, x_n\}$ be a finite alphabet.

### Marking Sequence

Any ordered list $\sigma$ of the symbols from $X$ (or, equivalently, a bijection $\sigma : \{1, 2, \ldots, |X|\} \rightarrow X$) is a **marking sequence**.

### Marking Number

The **marking number** $\pi_\sigma(\alpha)$ (of $\sigma$ with respect to $\alpha$) is the maximum number of marked blocks obtained while marking $\alpha$ according to $\sigma$.

### Locality Number

A string $\alpha$ over $X$ is **$k$-local** $\iff \pi_\sigma(\alpha) \leq k$, for some marking sequence $\sigma$.

The **locality number** of $\alpha$ is $\text{loc}(\alpha) = \min\{k \mid \alpha \text{ is } k\text{-local}\}$.
The Locality Number

Example

Let \( \alpha = \text{adabadbdaecbcb} \), \( \sigma_1 = (b, c, e, d, a) \), \( \sigma_2 = (d, a, b, c, e) \).
The Locality Number

Example

Let $\alpha = \text{adabadbdaecbcb}$, $\sigma_1 = (b, c, e, d, a)$, $\sigma_2 = (d, a, b, c, e)$

$\pi_{\sigma_1}(\alpha) = 4 \implies \text{loc}(\alpha) \leq 4$
The Locality Number

Example

Let $\alpha = \text{adabaddaecedcbb}$, $\sigma_1 = (b, c, e, d, a)$, $\sigma_2 = (d, a, b, c, e)$

$\pi_{\sigma_1}(\alpha) = 4 \ (\Rightarrow \ \text{loc}(\alpha) \leq 4)$

$\pi_{\sigma_2}(\alpha) = 3 \ (\Rightarrow \ \text{loc}(\alpha) \leq 3)$
The Locality Number

Example

Let \( \alpha = \text{adabadbdaecbcb} \), \( \sigma_1 = (b, c, e, d, a) \), \( \sigma_2 = (d, a, b, c, e) \)

\[ \pi_{\sigma_1}(\alpha) = 4 \implies \text{loc}(\alpha) \leq 4 \]

\[ \pi_{\sigma_2}(\alpha) = 3 \implies \text{loc}(\alpha) \leq 3 \]

\( \text{loc}(\alpha) = 3 \)
## The Locality Number

### Example

Let \( \alpha = \text{adabadbdaecbcb} \), \( \sigma_1 = (b, c, e, d, a) \), \( \sigma_2 = (d, a, b, c, e) \)

\[
\pi_{\sigma_1}(\alpha) = 4 \implies \text{loc}(\alpha) \leq 4
\]

\[
\pi_{\sigma_2}(\alpha) = 3 \implies \text{loc}(\alpha) \leq 3
\]

\[
\text{loc}(\alpha) = 3
\]

### Motivation

Pattern matching with variables.

Marking sequence = dynamic programming algorithm

\( \sim \) XP-algorithms w.r.t. parameter \( \text{loc}(\alpha) \).
## Known Results and Open Problems

### Computing the locality number

**Loc**

<table>
<thead>
<tr>
<th>Input:</th>
<th>String ( \alpha \in \Sigma^* ), ( k \in \mathbb{N} ).</th>
</tr>
</thead>
<tbody>
<tr>
<td>Question:</td>
<td>( \text{loc}(\alpha) \leq k )?</td>
</tr>
</tbody>
</table>

**MinLoc** denotes the corresponding minimisation problem.

**Known Results**

- \( \text{Loc} \in \text{XP} \) w.r.t. parameter \( k \) (i.e., in \( \text{P} \) for fixed \( k \)).

**Open Problems**

- Is \( \text{Loc} \) NP-complete?
- Is \( \text{Loc} \in \text{FPT} \) (w.r.t. \( k \) or \( |\Sigma| \))?
- Are there good approximation algorithms for MinLoc?
**Known Results and Open Problems**

### Computing the locality number

<table>
<thead>
<tr>
<th>Loc</th>
<th>Input: String $\alpha \in \Sigma^*$, $k \in \mathbb{N}$.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Question: $\text{loc}(\alpha) \leq k$?</td>
</tr>
</tbody>
</table>

MinLoc denotes the corresponding minimisation problem.
Known Results and Open Problems

Computing the locality number

Loc
Input: String $\alpha \in \Sigma^*$, $k \in \mathbb{N}$.
Question: $\text{loc}(\alpha) \leq k$?

MinLoc denotes the corresponding minimisation problem.

Known Results

Loc $\in \text{XP}$ w.r.t. parameter $k$ (i.e., in P for fixed $k$).
## Known Results and Open Problems

### Computing the locality number

**Loc**

**Input:** String $\alpha \in \Sigma^*$, $k \in \mathbb{N}$.

**Question:** $\text{loc}(\alpha) \leq k$?

MinLoc denotes the corresponding minimisation problem.

### Known Results

Loc $\in \text{XP}$ w.r.t. parameter $k$ (i.e., in P for fixed $k$).

### Open Problems

- Is Loc NP-complete?
- Is Loc $\in \text{FPT}$ (w.r.t. $k$ or $|\Sigma|$)?
- Are there good approximation algorithms for MinLoc?
Let $G = (V, E)$ be a (multi)graph with $V = \{v_1, \ldots, v_n\}$.

**Cuts**

**Cut**: partition $(V_1, V_2)$ of $V$. 
Let $G = (V, E)$ be a (multi)graph with $V = \{v_1, \ldots, v_n\}$.

### Cuts

**Cut**: partition $(V_1, V_2)$ of $V$.

**Cut set**: $C(V_1, V_2) = \{\{x, y\} \in E \mid x \in V_1, y \in V_2\}$. 

**Cutwidth** 

**Cutwidth of $L$**: $cw(L) = \max \{|C(V_j^1, V_j^2)| \mid 0 \leq j \leq n\}$

**Cutwidth of $G$**: $cw(G) = \min \{cw(L) \mid L \text{ is linear arrangement for } G\}$. 

**Linear Arrangements and Cut width**

**Linear arrangement of $G$**: sequence $L = (v_{j_1}^1, v_{j_2}^2, \ldots, v_{j_n}^n)$, where $(j_1, j_2, \ldots, j_n)$ is a permutation of $(1, 2, \ldots, n)$.
Cutwidth

Let $G = (V, E)$ be a (multi)graph with $V = \{v_1, \ldots, v_n\}$.

**Cuts**

**Cut**: partition $(V_1, V_2)$ of $V$.

**Cut set**: $C(V_1, V_2) = \{\{x, y\} \in E \mid x \in V_1, y \in V_2\}$.

**Size of a cut**: $|C(V_1, V_2)|$. 
Let $G = (V, E)$ be a (multi)graph with $V = \{v_1, \ldots, v_n\}$.

**Cuts**

**Cut**: partition $(V_1, V_2)$ of $V$.

**Cut set**: $C(V_1, V_2) = \{\{x, y\} \in E \mid x \in V_1, y \in V_2\}$.

**Size of a cut**: $|C(V_1, V_2)|$.

**Linear Arrangements and Cutwidth**

**Linear arrangement of $G$**: sequence $L = (v_{j_1}, v_{j_2}, \ldots, v_{j_n})$, where $(j_1, j_2, \ldots, j_n)$ is a permutation of $(1, 2, \ldots, n)$. 
Cutwidth

Let $G = (V, E)$ be a (multi)graph with $V = \{v_1, \ldots, v_n\}$.

Cuts

**Cut**: partition $(V_1, V_2)$ of $V$.

**Cut set**: $\mathcal{C}(V_1, V_2) = \{\{x, y\} \in E \mid x \in V_1, y \in V_2\}$.

**Size of a cut**: $|\mathcal{C}(V_1, V_2)|$.

Linear Arrangements and Cutwidth

**Linear arrangement of $G$**: sequence $L = (v_{j_1}, v_{j_2}, \ldots, v_{j_n})$, where $(j_1, j_2, \ldots, j_n)$ is a permutation of $(1, 2, \ldots, n)$.

**Cutwidth of $L$**: $\text{cw}(L) = \max\{|\mathcal{C}(\{v_{j_1}, v_{j_2}, \ldots, v_{j_i}\}, \{v_{j_{i+1}}, \ldots, v_{j_n}\})| \mid 0 \leq i \leq n\}$.
**Cutwidth**

Let $G = (V, E)$ be a (multi)graph with $V = \{v_1, \ldots, v_n\}$.

### Cuts

- **Cut**: partition $(V_1, V_2)$ of $V$.
- **Cut set**: $C(V_1, V_2) = \{\{x, y\} \in E \mid x \in V_1, y \in V_2\}$.
- **Size of a cut**: $|C(V_1, V_2)|$.

### Linear Arrangements and Cutwidth

- **Linear arrangement of $G$**: sequence $L = (v_{j_1}, v_{j_2}, \ldots, v_{j_n})$, where $(j_1, j_2, \ldots, j_n)$ is a permutation of $(1, 2, \ldots, n)$.
- **Cutwidth of $L$**: $\text{cw}(L) = \max\{|C(\{v_{j_1}, v_{j_2}, \ldots, v_{j_i}\}, \{v_{j_{i+1}}, \ldots, v_{j_n}\})| \mid 0 \leq i \leq n\}$
- **Cutwidth of $G$**: $\text{cw}(G) = \min\{\text{cw}(L) \mid L \text{ is lin. arr. for } G\}.$
Graph $G$:

![Graph](image)

Linear arrangement with cut width 5:

$u \quad v \quad w \quad x \quad y \quad z$

Linear arrangement with cut width 3:

$w \quad u \quad x \quad v \quad y \quad z$

$\text{cw}(G) = 3$
Cutwidth – Example

Graph $G$:

Linear arrangement with cutwidth 5:
Cutwidth – Example

Graph $G$: 

Linear arrangement with cutwidth 5:
Cutwidth – Example

Graph $G$:

Linear arrangement with cutwidth 5:

Linear arrangement with cutwidth 3:

$$cw(G) = 3$$
Cutwidth – Example

Graph $G$:

Linear arrangement with cutwidth 5:

Linear arrangement with cutwidth 3:
**Cutwidth – Example**

Graph $G$:

```
  y
 /|
/  |
u---v---z
  |   |
  w---x
```

Linear arrangement with cutwidth 5:

```
u--v--w--x--y--z
```

Linear arrangement with cutwidth 3:

```
w--u--x--v--y--z
```

$\text{cw}(G) = 3$
Computing the Cutwidth

Cutwidth problem

Cutwidth

Input: (Multi)graph $G$, $k \in \mathbb{N}$.
Question: $\text{cw}(\alpha) \leq k$?
Computing the Cutwidth

<table>
<thead>
<tr>
<th>Cutwidth problem</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cutwidth</td>
</tr>
<tr>
<td>Input: (Multi)graph $G$, $k \in \mathbb{N}$.</td>
</tr>
<tr>
<td>Question: $cw(\alpha) \leq k$?</td>
</tr>
</tbody>
</table>

MinCutwidth denotes the corresponding minimisation problem.
Computing the Cutwidth

Cutwidth problem

Cutwidth

- **Input:** (Multi)graph $G$, $k \in \mathbb{N}$.
- **Question:** $\text{cw}(\alpha) \leq k$?

MinCutwidth denotes the corresponding minimisation problem.

**Known Results**

- Cutwidth is NP-complete.
- Cutwidth $\in$ FPT (w.r.t. $k$).
- Exact exponential algorithms, linear fpt-algorithms, approximation algorithms...
Loc \leq \text{Cutwidth}

\Sigma = \{a, b, c, d\}
\alpha = abcabcdbada
k = 2.
Loc $\leq$ Cutwidth

$\Sigma = \{a, b, c, d\}$
$\alpha = abcpcbada$
$k = 2.$

Construct multigraph $H_{\alpha,k} = (V, E)$:

- nodes $a, b, c, d$
- edges between $a, b, c, d$ based on $\alpha$
Loc $\leq$ Cutwidth

$\Sigma = \{a, b, c, d\}$
$\alpha = \text{abcbcdbada}$
$k = 2.$

Construct multigraph $H_{\alpha,k} = (V, E)$:

```
\begin{tikzpicture}
  \node [shape=circle,draw=black] (A) at (0,0) {a};
  \node [shape=circle,draw=black] (B) at (1,0) {b};
  \node [shape=circle,draw=black] (C) at (0,-1) {c};
  \node [shape=circle,draw=black] (D) at (1,-1) {d};

  \path (A) edge (B);

\end{tikzpicture}
```
Loc $\leq$ Cutwidth

$\Sigma = \{a, b, c, d\}$
$\alpha = abcbcdbada$
$k = 2.$

Construct multigraph $H_{\alpha,k} = (V, E)$:

\begin{center}
\begin{tikzpicture}
  \node (a) at (0,0) {a};
  \node (b) at (1,0) {b};
  \node (c) at (0,-1) {c};
  \node (d) at (1,-1) {d};
  \draw (a) -- (b);
\end{tikzpicture}
\end{center}

Lemma $cw(H_{\alpha,k}) = 2k$ if and only if $lo c(\alpha) \leq k$. 
Loc $\leq$ Cutwidth

$\Sigma = \{a, b, c, d\}$
$\alpha = abc bcd bda$
$k = 2.$

Construct multigraph $H_{\alpha, k} = (V, E)$:

Diagram of the multigraph $H_{\alpha, k}$ with vertices $a$, $b$, $c$, $d$ and edges $ab$, $bc$, $cd$, $db$. The marking sequence is $(c, b, d, a)$. 

Lemma $\text{cw}(H_{\alpha, k}) = 2k$ if and only if $\text{loc}(\alpha) \leq k$. 

Loc \leq \text{Cutwidth}

\[ \Sigma = \{a, b, c, d\} \]
\[ \alpha = abc\text{bcdbada} \]
\[ k = 2. \]

Construct multigraph
\[ H_{\alpha,k} = (V, E): \]

\[ a \rightarrow b \]
\[ c \rightarrow d \]
\[ \]
Construct multigraph $H_{\alpha,k} = (V, E)$:

$$\Sigma = \{a, b, c, d\}$$

$$\alpha = abc\text{bcd}bada$$

$$k = 2.$$
Loc \leq \text{Cutwidth}

\[ \Sigma = \{a, b, c, d\} \]
\[ \alpha = abc\text{bcd}b\text{ada} \]
\[ k = 2. \]

Construct multigraph
\[ H_{\alpha,k} = (V, E): \]

\[ \alpha = \text{abc\text{bcd}b\text{ada}} \]
\[ k = 2. \]
Loc \leq \text{Cutwidth}

\[ \Sigma = \{a, b, c, d\} \]
\[ \alpha = abc\text{bcdabada} \]
\[ k = 2. \]

Construct multigraph \( H_{\alpha,k} = (V, E) \):

![Graph Diagram]
Loc ≤ Cutwidth

\[ \Sigma = \{a, b, c, d\} \]
\[ \alpha = \text{abcycdbada} \]
\[ k = 2. \]

Construct multigraph

\[ H_{\alpha, k} = (V, E) : \]

\[ \text{Marking sequence: } (c, b, d, a) \]
Loc \leq \text{Cutwidth}

\Sigma = \{a, b, c, d\}
\alpha = abcbcdbada
k = 2.

Construct multigraph $H_{\alpha,k} = (V, E)$:
Loc ≤ Cutwidth

$$\Sigma = \{a, b, c, d\}$$
$$\alpha = abcdbada$$
$$k = 2.$$
Loc $\leq$ Cutwidth

$\Sigma = \{a, b, c, d\}$
$\alpha = abcbcdbada$
$k = 2.$

Construct multigraph $H_{\alpha, k} = (V, E)$:

$\Sigma = \{a, b, c, d\}$
$\alpha = abcbcdbada$
$k = 2.$
$\Sigma = \{a, b, c, d\}$

$\alpha = abc\text{bcdbada}$

$k = 2$.

Construct multigraph $H_{\alpha, k} = (V, E)$:
Construct multigraph $H_{\alpha,k} = (V, E)$:

Marking sequence: $(c, b, d, a)$
Loc $\leq$ Cutwidth

$\Sigma = \{a, b, c, d\}$
$\alpha = abc\text{bcdbada}$
$k = 2$.

Construct multigraph $H_{\alpha,k} = (V, E)$:

Marking sequence: $(c, b, d, a)$
Loc $\leq$ Cutwidth

$\Sigma = \{a, b, c, d\}$
$\alpha = abcdbcdbada$
$k = 2.$

Construct multigraph $H_{\alpha,k} = (V, E)$:

$\alpha = a \ b \ c \ b \ c \ d \ b \ a \ d \ a$

Marking sequence: $(c, b, d, a)$
\[ \Sigma = \{a, b, c, d\} \]
\[ \alpha = abcdbcdbada \]
\[ k = 2. \]

Construct multigraph \( H_{\alpha, k} = (V, E) \):

Marking sequence: \((c, b, d, a)\)
Loc $\leq$ Cutwidth

$\Sigma = \{a, b, c, d\}$
$\alpha = abcdbadbada$
$k = 2.$

Construct multigraph $H_{\alpha, k} = (V, E)$:

Marking sequence: $(c, b, d, a)$
Loc $\leq$ Cutwidth

$\Sigma = \{a, b, c, d\}$
$\alpha = abccbdaba$
$k = 2.$

Construct multigraph $H_{\alpha,k} = (V, E)$:

Marking sequence: $(c, b, d, a)$
Loc $\leq$ Cutwidth

$\Sigma = \{a, b, c, d\}$
$\alpha = abcbcdabada$
$k = 2.$

Construct multigraph $H_{\alpha,k} = (V, E)$:

Marking sequence: $(c, b, d, a)$
Loc $\leq$ Cutwidth

$\Sigma = \{a, b, c, d\}$
$\alpha = abc\text{bcdbada}$
$k = 2.$

Construct multigraph $H_{\alpha,k} = (V, E)$:

Marking sequence: (c, b, d, a)
Loc \leq \text{Cutwidth}

\[ \Sigma = \{a, b, c, d\} \]
\[ \alpha = \text{abcdbada} \]
\[ k = 2. \]

Construct multigraph \( H_{\alpha,k} = (V, E) \):

Marking sequence: \((c, b, d, a)\)

Lemma

\( \text{cw}(H_{\alpha,k}) = 2k \) if and only if \( \text{loc}(\alpha) \leq k \).
Cutwidth ≤ Loc

$G = (V, E)$:

![Graph Diagram]

$\alpha = \{x, v\}$

Lemma

$\forall e \in E: cw(G) \leq loc(\alpha e) \leq cw(G) + 1$

$\exists e \in E: loc(\alpha e) = cw(G)$. 


Cutwidth $\leq$ Loc

$G = (V, E)$:

$u \quad v \quad z$

$w \quad x$
Cutwidth $\leq \text{Loc}$

$G = (V, E)$:
Cutwidth $\leq$ Loc

$G = (V, E)$:

$\alpha \{x, v\} = x$

Lemma

$\forall e \in E$: $cw(G) \leq lo c(\alpha e) \leq cw(G) + 1$

$\exists e \in E$: $lo c(\alpha e) = cw(G)$. 
Cutwidth $\leq$ Loc

$G = (V, E)$:

$\alpha = \{x, v\}$

Lemma

$\forall e \in E: cw(G) \leq lo(c(\alpha e)) \leq cw(G) + 1$

$\exists e \in E: lo(c(\alpha e)) = cw(G)$.
Cutwidth $\leq$ Loc

$G = (V, E)$:

$x \ w \ u$
Cutwidth $\leq$ Loc

$G = (V, E)$:

\[ \{x, v\} = x \quad w \quad u \quad x \]
Cutwidth $\leq$ Loc

$G = (V, E)$:

$\{x, v\} = x w u x w$

$\exists e \in E: \text{loc}(\alpha e) = \text{cw}(G)$.
Cutwidth $\leq$ Loc

$G = (V, E)$:

$\alpha \{ x, v \} = \alpha (G)$

Lemma: $\forall e \in E$, $cw(G) \leq loc(\alpha e) \leq cw(G) + 1$.

$\exists e \in E$, $loc(\alpha e) = cw(G)$. 

$w \ x \ u \ x \ w \ u$
Cutwidth $\leq$ Loc

$G = (V, E)$:

\[
x w u x w u x
\]
Cutwidth $\leq$ Loc

$G = (V, E)$:

$\alpha\{x, v\} = x \ x \ u \ x \ w \ u \ x \ v$

Lemma

$\forall e \in E: cw(G) \leq loc(\alpha e) \leq cw(G) + 1$

$\exists e \in E: loc(\alpha e) = cw(G)$. 
**Cutwidth \( \leq \) Loc**

\[ G = (V, E): \]

\begin{align*}
&x \ x \ w \ u \ x \ w \ u \ x \ v \ u
\end{align*}
Cutwidth $\leq$ Loc

$G = (V, E)$:

$x\;w\;u\;x\;w\;u\;x\;v\;u\;v\;y\;z\;v\;y\;z\;v$
Cutwidth $\leq$ Loc

$G = (V, E)$:

$$\alpha\{x, v\} = x \overset{w}{\rightarrow} u \overset{x}{\rightarrow} w \overset{u \times v}{\rightarrow} u \overset{v}{\rightarrow} v \overset{y, z, v}{\rightarrow} y \overset{z, v}{\rightarrow} z \overset{v}{\rightarrow} v$$
Cutwidth $\leq$ Loc

$G = (V, E)$:

$\alpha\{x, v\} = x w u x w u x v u v y z v y z v$

Lemma

$\forall e \in E : cw(G) \leq loc(\alpha_e) \leq cw(G) + 1$

$\exists e \in E : loc(\alpha_e) = cw(G)$. 
Consequences
## Consequences

<table>
<thead>
<tr>
<th>Approximation Meta-Theorem</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th></th>
<th>MinCutwidth</th>
<th>MinLoc</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run time:</td>
<td>$O(f(</td>
<td>E</td>
</tr>
<tr>
<td>Appr. ratio:</td>
<td>$r(\text{opt},</td>
<td>E</td>
</tr>
</tbody>
</table>
### Approximation Meta-Theorem

<table>
<thead>
<tr>
<th></th>
<th>MinCutwidth</th>
<th>MinLoc</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Run time:</strong></td>
<td>$O(f(</td>
<td>E</td>
</tr>
<tr>
<td><strong>Appr. ratio:</strong></td>
<td>$r(\text{opt},</td>
<td>E</td>
</tr>
<tr>
<td><strong>Run time:</strong></td>
<td>$O(n(f(</td>
<td>E</td>
</tr>
<tr>
<td><strong>Appr. ratio:</strong></td>
<td>$r(\text{opt},</td>
<td>E</td>
</tr>
</tbody>
</table>
Consequences

Approximation Meta-Theorem

<table>
<thead>
<tr>
<th></th>
<th>MinCutwidth</th>
<th>MinLoc</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run time:</td>
<td>$O(f(</td>
<td>E</td>
</tr>
<tr>
<td>Appr. ratio:</td>
<td>$r(\text{opt},</td>
<td>E</td>
</tr>
</tbody>
</table>

Theorem

The problem Loc

- is NP-complete,
  (even if every symbol has at most 3 occurrences)
## Consequences

### Approximation Meta-Theorem

<table>
<thead>
<tr>
<th></th>
<th>MinCutwidth</th>
<th>MinLoc</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Run time:</strong></td>
<td>$O(f(</td>
<td>E</td>
</tr>
<tr>
<td><strong>Appr. ratio:</strong></td>
<td>$r(\text{opt},</td>
<td>E</td>
</tr>
<tr>
<td><strong>Run time:</strong></td>
<td>$O(n(f(</td>
<td>E</td>
</tr>
<tr>
<td><strong>Appr. ratio:</strong></td>
<td>$r(\text{opt},</td>
<td>E</td>
</tr>
</tbody>
</table>

### Theorem

The problem Loc

- is NP-complete,
  - (even if every symbol has at most 3 occurrences)
- can be solved in $O^*(2|\Sigma|)$,
## Consequences

### Approximation Meta-Theorem

<table>
<thead>
<tr>
<th></th>
<th>MinCutwidth</th>
<th>MinLoc</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run time:</td>
<td>$O(f(</td>
<td>E</td>
</tr>
<tr>
<td>Appr. ratio:</td>
<td>$r(\text{opt},</td>
<td>E</td>
</tr>
</tbody>
</table>

### Theorem

The problem Loc

- is NP-complete,
  (even if every symbol has at most 3 occurrences)
- can be solved in $O^*(2^{2\Sigma})$,
- in FPT (w.r.t. parameter $k$), with linear fpt-algorithm.
Path-Decompositions and Pathwidth

Path-decompositions as tree-decomposition

A path-decomposition is a tree-decomposition the underlying tree-structure of which is a path.
Path-Decompositions and Pathwidth

<table>
<thead>
<tr>
<th>Path-decompositions as tree-decomposition</th>
</tr>
</thead>
<tbody>
<tr>
<td>A path-decomposition is a tree-decomposition the underlying tree-structure of which is a path.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Path-decompositions as marking procedures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Let $G = (V, E)$ be a graph.</td>
</tr>
<tr>
<td><strong>Path-decomposition</strong> $Q$ of $G$:</td>
</tr>
</tbody>
</table>
Path-Decompositions and Pathwidth

Path-decompositions as tree-decomposition

A path-decomposition is a tree-decomposition the underlying tree-structure of which is a path.

Path-decompositions as marking procedures

Let $G = (V, E)$ be a graph.

Path-decomposition $Q$ of $G$:

- Initially all vertices are blue.
Path-Decompositions and Pathwidth

Path-decompositions as tree-decomposition

A path-decomposition is a tree-decomposition the underlying tree-structure of which is a path.

Path-decompositions as marking procedures

Let $G = (V, E)$ be a graph.

**Path-decomposition** $Q$ of $G$:

- Initially all vertices are blue.
- Until all vertices are blue again,
  - color a vertex red that has never been red before, or
  - color a red vertex blue again,
### Path-Decompositions and Pathwidth

<table>
<thead>
<tr>
<th>Path-decompositions as tree-decomposition</th>
</tr>
</thead>
<tbody>
<tr>
<td>A path-decomposition is a tree-decomposition the underlying tree-structure of which is a path.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Path-decompositions as marking procedures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Let $G = (V, E)$ be a graph.</td>
</tr>
<tr>
<td><strong>Path-decomposition</strong> $Q$ of $G$:</td>
</tr>
<tr>
<td>▶ Initially all vertices are blue.</td>
</tr>
<tr>
<td>▶ Until all vertices are blue again,</td>
</tr>
<tr>
<td>▶ color a vertex red that has never been red before, or</td>
</tr>
<tr>
<td>▶ color a red vertex blue again,</td>
</tr>
<tr>
<td>▶ such that each two adjacent vertices are red at the same time.</td>
</tr>
</tbody>
</table>
Path-Decompositions and Pathwidth

Path-decompositions as tree-decomposition

A path-decomposition is a tree-decomposition the underlying tree-structure of which is a path.

Path-decompositions as marking procedures

Let $G = (V, E)$ be a graph.

Path-decomposition $Q$ of $G$:

- Initially all vertices are blue.
- Until all vertices are blue again,
  - color a vertex red that has never been red before, or
  - color a red vertex blue again,
- such that each two adjacent vertices are red at the same time.

$pw(Q)$: Max. number of marked vertices.
Path-Decompositions and Pathwidth

Path-decompositions as tree-decomposition
A path-decomposition is a tree-decomposition the underlying
tree-structure of which is a path.

Path-decompositions as marking procedures
Let $G = (V, E)$ be a graph.
Path-decomposition $Q$ of $G$:
- Initially all vertices are blue.
- Until all vertices are blue again,
  - color a vertex red that has never been red before, or
  - color a red vertex blue again,
- such that each two adjacent vertices are red at the same time.

$pw(Q)$: Max. number of marked vertices.
$pw(G)$: Min. $pw(Q)$ over all path-decompositions.
Computing the Pathwidth

<table>
<thead>
<tr>
<th>Pathwidth problem</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pathwidth</td>
</tr>
<tr>
<td>Input: Graph $G$, $k \in \mathbb{N}$.</td>
</tr>
<tr>
<td>Question: $pw(\alpha) \leq k$?</td>
</tr>
</tbody>
</table>
## Computing the Pathwidth

### Pathwidth problem

**Input:** Graph $G$, $k \in \mathbb{N}$.

**Question:** $\text{pw}(\alpha) \leq k$?

MinPathwidth denotes the corresponding minimisation problem.
Computing the Pathwidth

Pathwidth problem

Pathwidth
Input: Graph $G$, $k \in \mathbb{N}$.
Question: $pw(\alpha) \leq k$?

MinPathwidth denotes the corresponding minimisation problem.

Known Results

- Pathwidth is NP-complete.
- Pathwidth $\in$ FPT (w.r.t. $k$).
- Exact exponential algorithms, linear fpt-algorithms, approximation algorithms...
Loc $\leq$ Pathwidth

$\alpha = c a b a c a b a c$
$\alpha = c a b a c a b a c$

$G_\alpha$: 

![Graph diagram]
$\alpha = \text{c a b a c a b a c}$

$G_\alpha$: 

[Graph diagram]
$\alpha = c a b a c a b a c$

$\mathcal{G}_\alpha$: 

![Graph with nodes and edges representing $\mathcal{G}_\alpha$.]
Loc $\leq$ Pathwidth

$\alpha = c\, a\, b\, a\, c\, a\, b\, a\, c$

$G_\alpha$: 
Results

Lemma

\[ \text{loc}(\alpha) \leq \text{pw}(G_{\alpha}) \leq 2 \text{loc}(\alpha). \]
### Lemma

$\text{loc}(\alpha) \leq \text{pw}(G_\alpha) \leq 2\text{loc}(\alpha)$.

### Lemma

$\exists \alpha : \text{pw}(G_\alpha) = 2\text{loc}(\alpha)$,

$\exists \beta : \text{loc}(\beta) = \text{pw}(G_\beta)$. 
Results

Lemma
\[ \text{loc}(\alpha) \leq \text{pw}(G_\alpha) \leq 2 \text{loc}(\alpha). \]

Lemma
\[ \exists \alpha : \text{pw}(G_\alpha) = 2 \text{loc}(\alpha), \]
\[ \exists \beta : \text{loc}(\beta) = \text{pw}(G_\beta). \]

Theorem
There is an \( O(\sqrt{\log(\text{opt}) \log(n)}) \)-approx. algo. for MinLoc.
## Consequences for Cutwidth

<table>
<thead>
<tr>
<th>MinCutwidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Leighton and Rao, JACM 1999: $O(\log(n) \log(n))$-approximation.</td>
</tr>
<tr>
<td>(Based on more general approximation techniques for edge-seperators)</td>
</tr>
</tbody>
</table>
## Consequences for Cutwidth

<table>
<thead>
<tr>
<th>MinCutwidth</th>
<th>Leighton and Rao, JACM 1999: (O(\log(n) \log(n)))-approximation.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(Based on more general approximation techniques for edge-seperators)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MinPathwidth</th>
<th>Feige et al., SICOMP 2008: (O(\sqrt{\log(\text{opt}) \log(n)}))-approximation.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(Based on more general approximation techniques for vertex-seperators)</td>
</tr>
</tbody>
</table>
**Consequences for Cutwidth**

<table>
<thead>
<tr>
<th>MinCutwidth</th>
<th>Leighton and Rao, JACM 1999: $O(\log(n) \log(n))$-approximation.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(Based on more general approximation techniques for edge-seperators)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MinPathwidth</th>
<th>Feige et al., SICOMP 2008: $O(\sqrt{\log(\text{opt}) \log(n)})$-approximation.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(Based on more general approximation techniques for vertex-seperators)</td>
</tr>
</tbody>
</table>

| MinCutwidth $\leq$ MinLoc $\leq$ MinPathwidth | There is an $O(\sqrt{\log(\text{opt}) \log(h)})$-approximation algorithm for MinCutwidth on multigraphs with $h$ edges. |
Direct Reduction: MinCutwidth $\leq$ MinPathwidth