Generalizations to and from Chordal Graphs

Frank Kammer and Torsten Tholey

University of Augsburg, Germany

August 2010
Chordal Graphs

intersection graphs of subtrees in a tree

Frank Kammer
University of Augsburg
interval graphs $\subset$ chordal graphs
A clique tree of a chordal graph $G = (V, E)$ is ...

A tree $T = (W, F)$ with a mapping $B$ for each node $w \in W$ to a so-called \textit{bag} $V' \subseteq V$ such that

- $G[B(w)]$ is a maximal clique

$G = \bigcup_{w \in W} G[B(w)]$
A clique tree of a chordal graph $G = (V, E)$ is ...

A tree $T = (W, F)$ with a mapping $B$ for each node $w \in W$ to a so-called bag $V' \subseteq V$ such that

- $G[B(w)]$ is a maximal clique
- $G = \bigcup_{w \in W} G[B(w)]$
A clique tree of a chordal graph $G = (V, E)$ is ...

A tree $T = (W, F)$ with a mapping $B$ for each node $w \in W$ to a so-called bag $V' \subseteq V$ such that:

- $G[B(w)]$ is a maximal clique
- $G = \bigcup_{w \in W} G[B(w)]$
- each $v \in V$ occurs exactly in the bags of a subtree of $T$. 
A weak clique tree of a chordal graph $G = (V, E)$ is ...

A tree $T = (W, F)$ with a mapping $B$ for each node $w \in W$ to a so-called bag $V' \subseteq V$ such that:

- $G[B(w)]$ is a maximal clique
- $G = \bigcup_{w \in W} G[B(w)]$
- each $v \in V$ occurs exactly in the bags of a subtree of $T$. 

Frank Kammer

University of Augsburg
Given: A graph $G$ and $k$ vertex pairs $(s_1, t_1), \ldots, (s_k, t_k)$. 

$k$-Disjoint Paths Problem (k-DPP)
Given: A graph $G$ and $k$ vertex pairs $(s_1, t_1), \ldots, (s_k, t_k)$. 
Given: A graph $G$ and $k$ vertex pairs $(s_1, t_1), \ldots, (s_k, t_k)$.

Goal: Find $k$ pairwise vertex disjoint paths $P_1, \ldots, P_k$ such that $s_i$ and $t_i$ are the endpoints of $P_i$ ($1 \leq i \leq k$).
Given: A graph $G$ and $k$ vertex pairs $(s_1, t_1), \ldots, (s_k, t_k)$.

Goal: Find $k$ pairwise vertex disjoint paths $P_1, \ldots, P_k$ such that $s_i$ and $t_i$ are the endpoints of $P_i$ ($1 \leq i \leq k$).


**k-Disjoint Paths Problem (k-DPP)**

- **Given:** A graph $G$ and $k$ vertex pairs $(s_1, t_1), \ldots, (s_k, t_k)$.
- **Goal:** Find an extension of the initial coloring such that vertices of each color induce a connected component.
Results

ℓ-DPP on general undirected graphs:
- Robertson and Seymour [JCTB’95]: $O(n^3)$ time
- Perković and Reed [IJFCS’00]: $O(n^2)$ time

DPP on general undirected graphs:
- Lynch [SIGDA Newsletter’75]: NP-hard

Motivation

- VLSI design
- routing problems
- minor search
- ...
Results

\(\ell\)-DPP on general undirected graphs:
- Robertson and Seymour [JCTB’95]: \(O(n^3)\) time
- Perković and Reed [IJFCS’00]: \(O(n^2)\) time

DPP on general undirected graphs:
- Lynch [SIGDA Newsletter’75]: NP-hard

\(\ell\)-DPP on undirected chordal graphs:
- Kammer und Tholey [WG’09]: FPT \(O(f(\ell)n)\)

DPP on undirected chordal graphs:
- Kammer und Tholey [WG’09]: NP-hard
Chordal Graphs, i.e., Graphs with a Clique Tree

Observation if $\forall w' \neq w'' : B(w') \not\subseteq B(w'')$

Each bag of a node $w$ is a separator in $G$, unless $\text{deg}(w) = 1$. 
Observation if $\forall w' \neq w'' : B(w') \not\subseteq B(w'')$

Each bag of a node $w$ is a separator in $G$, unless $\deg(w) = 1$. 
Observation if $\forall w' \neq w'' : B(w') \not\subseteq B(w'')$

Each bag of a node $w$ is a separator in $G$, unless $\text{deg}(w) = 1$. 

Frank Kammer
University of Augsburg
Observation if $\forall w' \neq w'' : B(w') \not\subseteq B(w'')$

Each bag of a node $w$ is a separator in $G$, unless $\deg(w) = 1$.

For a color $c$, take $w_1$ and $w_2$ such that their bags contain $c$. Connecting $c \Rightarrow$ the nodes on the $w_1$-$w_2$-path have a bag

- with $\geq 1$ $c$-colored vertex
Chordal Graphs, i.e., Graphs with a Clique Tree

Observation if \( \forall w' \neq w'' : B(w') \not\subset B(w'') \):

Each bag of a node \( w \) is a separator in \( G \), unless \( \text{deg}(w) = 1 \).

For a color \( c \), take \( w_1 \) and \( w_2 \) such that their bags contain \( c \).

Connecting \( c \Rightarrow \) the nodes on the \( w_1-w_2 \)-path have a bag

- with \( \geq 1 \) \( c \)-colored vertex
Chordal Graphs, i.e., Graphs with a Clique Tree

Observation if $\forall w' \neq w'' : B(w') \not\subseteq B(w'')$

Each bag of a node $w$ is a separator in $G$, unless $\deg(w) = 1$.

For a color $c$, take $w_1$ and $w_2$ such that their bags contain $c$. connecting $c \Rightarrow$ the nodes on the $w_1$-$w_2$-path have a bag
- with $\geq 1$ $c$-colored vertex
Observation if $\forall w' \neq w'' : B(w') \not\subseteq B(w'')$

Each bag of a node $w$ is a separator in $G$, unless $\text{deg}(w) = 1$.

For a color $c$, take $w_1$ and $w_2$ such that their bags contain $c$. Connecting $c \Rightarrow$ the nodes on the $w_1$-$w_2$-path have a bag
- with $\geq 1$ $c$-colored vertex and
- w.l.o.g. with $\leq 2$ $c$-colored vertices.

Frank Kammer
University of Augsburg
Chordal Graphs, i.e., Graphs with a Clique Tree

Observation if $\forall w' \neq w'': B(w') \not\subseteq B(w'')$

Each bag of a node $w$ is a separator in $G$, unless $\text{deg}(w) = 1$.

For a color $c$, take $w_1$ and $w_2$ such that their bags contain $c$. Connecting $c \Rightarrow$ the nodes on the $w_1$-$w_2$-path have a bag

- with $\geq 1$ $c$-colored vertex and
- w.l.o.g. with $\leq 2$ $c$-colored vertices.

Ignore remaining nodes.
A clique tree for an $n$-vertex graph $G$ can be found in linear time.

Some facts

- A clique tree for an $n$-vertex graph $G$ has $\leq n$ nodes
Chordal Graphs, i.e., Graphs with a Clique Tree

Some facts

A clique tree for an $n$-vertex graph $G$
- has $\leq n$ nodes and
- can be found in linear time.
Some facts

A clique tree for an $n$-vertex graph $G$
- has $\leq n$ nodes and
- can be found in linear time.

Width of a clique tree

Maximum size of a bag of the clique tree minus 1.
Sketch of an algorithm

 Traverse weak clique tree bottom-up.
Sketch of an algorithm

- Traverse weak clique tree bottom-up.
Sketch of an algorithm

- Traverse weak clique tree bottom-up.
Algorithm

Sketch of an algorithm

- Traverse weak clique tree bottom-up.
- At each node \( w \) of the clique tree: Compute all valid colorings for \( B(w) \).
Algorithm

Sketch of an algorithm

- Traverse weak clique tree bottom-up.
- At each node \( w \) of the clique tree:
  Compute all valid colorings for \( B(w) \).

Properties of valid colorings for the bag \( B(w) \) of a node \( w \)

- No initially colored vertex in \( B(w) \) is recolored.
Algorithm

Sketch of an algorithm

- Traverse weak clique tree bottom-up.
- At each node \( w \) of the clique tree:
  Compute all valid colorings for \( B(w) \).

Properties of valid colorings for the bag \( B(w) \) of a node \( w \)

- No initially colored vertex in \( B(w) \) is recolored.
- Each color in \( B(w) \) is used \( \leq 2 \) times.
Algorithm

Sketch of an algorithm

- Traverse weak clique tree bottom-up.
- At each node \( w \) of the clique tree:
  Compute all valid colorings for \( B(w) \).

Properties of valid colorings for the bag \( B(w) \) of a node \( w \)

- No initially colored vertex in \( B(w) \) is recolored.
- Each color in \( B(w) \) is used \( \leq 2 \) times.
- For each child \( w' \) of \( w \):
  Extension of a valid coloring of \( B(w) \cup B(w') \).
Sketch of an algorithm

- Traverse weak clique tree bottom-up.
- At each node $w$ of the clique tree: Compute all valid colorings for $B(w)$.

Properties of valid colorings for the bag $B(w)$ of a node $w$

- No initially colored vertex in $B(w)$ is recolored.
- Each color in $B(w)$ is used $\leq 2$ times.
- For each child $w'$ of $w$:
  - Extension of a valid coloring of $B(w) \cup B(w')$.
- For each $w' \in N(w)$ with $B(w) \cap B(w')$ separating color $c$:
  - One vertex of $B(w) \cap B(w')$ is colored with $c$. 
Algorithm

Sketch of an algorithm

- Traverse weak clique tree bottom-up.
- At each node \( w \) of the clique tree: Compute all valid colorings for \( B(w) \).

Valid coloring \( B(w) \) corresponds to an extendable coloring for tree below \( w \).

Properties of valid colorings for the bag \( B(w) \) of a node \( w \)

- No initially colored vertex in \( B(w) \) is recolored.
- Each color in \( B(w) \) is used \( \leq 2 \) times.
- For each child \( w' \) of \( w \):
  - Extension of a valid coloring of \( B(w) \cup B(w') \).
- For each \( w' \in N(w) \) with \( B(w) \cap B(w') \) separating color \( c \): One vertex of \( B(w) \cap B(w') \) is colored with \( c \).
Algorithm

Sketch of an algorithm

- Traverse weak clique tree bottom-up.
- At each node $w$ of the clique tree: Compute all valid colorings for $B(w)$.

$\Rightarrow$ At the root we obtain a solution for the $k$-DPP.

Properties of valid colorings for the bag $B(w)$ of a node $w$

- No initially colored vertex in $B(w)$ is recolored.
- Each color in $B(w)$ is used $\leq 2$ times.
- For each child $w'$ of $w$:
  - Extension of a valid coloring of $B(w) \cup B(w')$.
- For each $w' \in N(w)$ with $B(w) \cap B(w')$ separating color $c$:
  - One vertex of $B(w) \cap B(w')$ is colored with $c$. 

Frank Kammer
University of Augsburg
**Algorithm**

**Sketch of an algorithm**

- Traverse weak clique tree bottom-up.
- At each node $w$ of the clique tree:
  - Compute all valid colorings for $B(w)$.

If clique tree has width $r$, $\leq (r + 2)^{2^k}$ colorings are analyzed for each bag.

**Properties of valid colorings for the bag $B(w)$ of a node $w$**

- No initially colored vertex in $B(w)$ is recolored.
- Each color in $B(w)$ is used $\leq 2$ times.
- For each child $w'$ of $w$:
  - Extension of a valid coloring of $B(w) \cup B(w')$.
- For each $w' \in N(w)$ with $B(w) \cap B(w')$ separating color $c$:
  - One vertex of $B(w) \cap B(w')$ is colored with $c$. 

Frank Kammer  
University of Augsburg
Sketch of an algorithm

- Traverse weak clique tree bottom-up.
- At each node $w$ of the clique tree:
  Compute all valid colorings for $B(w)$.

Since an $n$-vertex chordal graph $G$ has clique tree width $r \leq n$, the $k$-DPP is solvable on $G$ in polynomial time.

Properties of valid colorings for the bag $B(w)$ of a node $w$

- No initially colored vertex in $B(w)$ is recolored.
- Each color in $B(w)$ is used $\leq 2$ times.
- For each child $w'$ of $w$:
  Extension of a valid coloring of $B(w) \cup B(w')$.
- For each $w' \in N(w)$ with $B(w) \cap B(w')$ separating color $c$:
  One vertex of $B(w) \cap B(w')$ is colored with $c$. 
Example
Valid colorings for $B_{11}$

$B_{11} \cap B_6 = \{x\}$ is a separator for green $\Rightarrow x$ must be green.
Valid colorings for $B_{12}$

Two valid colorings: One colors $x$ and the other $d$ in blue.
Valid colorings for $B_{12}$

Two valid colorings: One colors $x$ and the other $d$ in blue.
Valid colorings for $B_6$

Simultaneous extend valid colorings of $B_{11}$ and $B_{12}$. 

Frank Kammer
University of Augsburg
Example

Valid colorings for $B_6$

For connecting green, additionally $k$ must be green.
Valid colorings for $B_7$

Extend valid colorings $\Rightarrow$ $k$ must be green & $d$ must be blue.
Valid colorings for $B_7$

Moreover, exactly one of $a$, $f$ or $q$ must be green.
Valid colorings for $B_7$

Moreover, exactly one of $a$, $f$ or $q$ must be green.
Valid colorings for $B_7$

Moreover, exactly one of $a$, $f$ or $q$ must be green.
Valid colorings for $B_1$

$j$ must be green since $B_1 \cap B_2 = \{d, j\}$ is a separator for green.

Frank Kammer
University of Augsburg
Valid colorings for $B_1$

Again, exactly one of $a$, $f$ or $q$ must be green.
Valid colorings for $B_3$

*Extend valid colorings $\Rightarrow j$ must be green & $d$ must be blue.*
Valid colorings for $B_3$

Similar argumentation: Either s green & y blue or vice versa.
Valid colorings for $B_3$

*Similar argumentation: Either $s$ green & $y$ blue or vice versa.*
Valid colorings for $B_3$

$$(B_8 \cap B_3) \setminus \{d, j, s, y\} = \{h\} \Rightarrow h \text{ must be red.}$$
Valid colorings for $B_3$

Exactly one of $e$, $n$ or $t$ must be red. In sum, 6 valid colorings.
Valid colorings for $B_4$

Each valid coloring of $B_3$ can be extended. $s$ remains colored.
Valid colorings for $B_4$

We can extend only valid colorings of $B_5$ with $s$ uncolored, $v$ red.
Recall: For each node $x$ of $T$, its bag $B(x)$ has only $\leq 2$ vertices for each color $c$. 
In our example: For each edge \( \{x, y\} \) of \( T \) used by a color \( c \), its edge bag \( B(\{x, y\}) = B(x) \cap B(y) \) has one \( c \)-colored vertex.
For each edge \( \{x, y\} \) of \( T \),
\[
|B(\{x, y\})| \geq \#(\text{colors using } \{x, y\}); \text{ remove rest.}
\]
For each edge \( \{x, y\} \) of \( T \),
guarantee \( |B(\{x, y\})| \geq \#(\text{colors using}\{x, y\}) \); remove rest.
Goal: Remove occurrences of a vertex v such that the nodes whose bags contain v induce a subtree.
Iterate over $c$: For each $B_i$, if a $c$-colored vertex $v$ is below $B_i$, take up to $\ell \in \mathbb{N}$ vertices of $B_i$ closest to $v$ for $B_i$. 

Frank Kammer
University of Augsburg
Example for $\ell = 1$

Iterate over $c$: For each $B_i$, if a $c$-colored vertex $v$ is below $B_i$, take up to $\ell \in \mathbb{IN}$ vertices of $B_i$ closest to $v$ for $B_i$. 

Frank Kammer
University of Augsburg
Example for $\ell = 1$

Iterate over $c$: For each $B_i$, if a $c$-colored vertex $v$ is below $B_i$, take up to $\ell \in \mathbb{IN}$ vertices of $B_i$ closest to $v$ for $B_i$. 
Example for $\ell = 1$

Iterate over $c$: For each $B_i$, if a $c$-colored vertex $v$ is below $B_i$, take up to $\ell \in \mathbb{N}$ vertices of $B_i$ closest to $v$ for $B_i$. 
Example for $\ell = 1$

Iterate over $c$: For each $B_i$, if a $c$-colored vertex $v$ is below $B_i$, take up to $\ell \in \mathbb{N}$ vertices of $B_i$ closest to $v$ for $B_i$ & child bags.
**Example for $\ell = 1$**

**Lemma:** If a vertex is chosen at a bag $B_i$, then it is chosen at all ancestors of $B_i$.

$\Rightarrow$ Each vertex is part of bags of 1 subtree.
Lemma: If a vertex is chosen at a bag $B_i$, then it is chosen at all ancestors of $B_i$. 
$\Rightarrow$ Each vertex is part of bags of 1 subtree.
Example for $\ell = 1$

Lemma: If a vertex is chosen at a bag $B_i$, then it is chosen at all ancestors of $B_i$. $\implies$ Each vertex is part of bags of 1 subtree.

Frank Kammer
University of Augsburg
Lemma: If a vertex is chosen at a bag $B_i$, then it is chosen at all ancestors of $B_i$. ⇒ Each vertex is part of bags of 1 subtree.
Lemma: If a vertex is chosen at a bag $B_i$, then it is chosen at all ancestors of $B_i$. $\Rightarrow$ Each vertex is part of bags of 1 subtree.
Lemma: For each color, we can independently choose vertices as described.
Lemma: For each color, we can independently choose vertices as described. We get a weak clique tree \((T', B')\) for some \(G'\).
Example for $\ell = 1$

Theorem: For connecting $k$ colors, $\ell = 2k$ is enough.
Example for $\ell = 1$

Theorem: For connecting $k$ colors, $\ell = 2k$ is enough.

$\text{solution in } (T, B) \implies \text{solution in weak clique tree } (T', B')$
solution in \((T, B)\) \(\Rightarrow\) solution in \((T', B')\)

A path of \(T\) whose endpoints contain two red terminals.

Frank Kammer
University of Augsburg
solution in \((T, B) \Rightarrow \text{solution in } (T', B')\)

Frank Kammer
University of Augsburg
A solution for red in \((T, B)\) such that each color occurs at most twice in a bag of \(B\).
solution in \((T, B) \Rightarrow \text{solution in } (T', B')\)

small circles \(\Leftrightarrow\) bags of \((T', B')\)
solution in \((T, B) \Rightarrow \) solution in \((T', B')\)

\[
B(\{x, y\}) = B(x) \cap B(y) \quad B'(\{x, y\}) = B'(x) \cap B'(y)
\]
solution in \((T, B) \Rightarrow \) solution in \((T', B')\)

**Invariant:** For each edge \(\{x, y\}\) of \(T'\) and each color \(c\),

\[
\text{there are} \leq 2 \text{ } c\text{-colored vertices in } B'(\{x, y\}).
\]
solution in \((T, B) \Rightarrow\) solution in \((T', B')\)

**Invariant:** For each edge \(\{x, y\}\) of \(T'\) and each color \(c\),

\[
\text{there are } \leq 2 \text{ } c\text{-colored vertices in } B'(\{x, y\}).
\]

an edge bag of \((T, B)\) is small \(\Leftrightarrow\) less than \(\ell\) vertices
solution in \((T, B) \Rightarrow solution in \((T', B')\)

**Invariant:** For each edge \(\{x, y\}\) of \(T'\) and each color \(c\),

there are \(\leq 2\) \(c\)-colored vertices in \(B'(\{x, y\})\).

**Lemma:**

\[ B(\{x, y\}) \text{ is small } \Rightarrow B'(\{x, y\}) = B(\{x, y\}) \]
solution in \((T, B) \Rightarrow \) solution in \((T', B')\)

**Invariant:** For each edge \(\{x, y\}\) of \(T'\) and each color \(c\),

\[
\text{there are } \leq 2 \text{-} c\text{-colored vertices in } B'(\{x, y\}).
\]

Color break for an edge \(\{x, y\}\): No red vertex is in \(B'(\{x, y\})\).
Invariant: For each edge \( \{x, y\} \) of \( T' \) and each color \( c \), there are \( \leq 2 \) \( c \)-colored vertices in \( B'(\{x, y\}) \).

Color break for an edge \( \{x, y\} \): No red vertex is in \( B'(\{x, y\}) \).
Invariant: For each edge \( \{x, y\} \) of \( T' \) and each color \( c \), there are \( \leq 2 \) \( c \)-colored vertices in \( B'(\{x, y\}) \).

Finally, if \( B'(x) \) of a node \( x \) has \( \geq 3 \) \( c \)-colored vertices, we can uncolor one vertex in \( B'(x) \).
**Invariant:** For each edge \( \{x, y\} \) of \( T' \) and each color \( c \), there are \( \leq 2 \) \( c \)-colored vertices in \( B'(\{x, y\}) \).

Finally, if \( B'(x) \) of a node \( x \) has \( \geq 3 \) \( c \)-colored vertices, we can uncolor one vertex in \( B'(x) \).
Invariant: For each edge \( \{x, y\} \) of \( T' \) and each color \( c \), there are \( \leq 2 \) \( c \)-colored vertices in \( B'(\{x, y\}) \).

Corollary (for fixed \( k \))

A bag of \( (T', B') \) contains \( \leq 2k\ell = 4k^2 = O(1) \) vertices.
Invariant: For each edge \( \{x, y\} \) of \( T' \) and each color \( c \), there are \( \leq 2 \) \( c \)-colored vertices in \( B'(\{x, y\}) \).

Corollary (for fixed \( k \))

The \( k \)-DPP is solvable in linear time on chordal graphs.
1-in-3 SAT

**Given:** a formula in 3-CNF

**Goal:** find an assignment s.t. 1 literal is true in every clause
1-in-3 SAT

**Given:** a formula in 3-CNF

**Goal:** find an assignment s.t. 1 literal is true in every clause

**monotone formula**

every literal is positive
1-in-3 SAT

**Given:** a formula in 3-CNF
**Goal:** find an assignment s.t. 1 literal is true in every clause

**monotone formula**
every literal is positive

**cubic**
every variable occurs exactly three times
1-in-3 SAT

**Given:** a formula in 3-CNF

**Goal:** find an assignment s.t. 1 literal is true in every clause

---

**monotone formula**

every literal is positive

---

**cubic**

every variable occurs exactly three times

---

Moore and Robson [Discrete and Comput. Geom. '01]

1-in-3 SAT restricted to monotone cubic formulas is NP-hard
Some notation

- the graph $G$ is defined by its clique tree
Some notation

- The graph $G$ is defined by its clique tree
- Black lines identify the same vertex

Diagram:

```
\begin{align*}
& a_1 \quad a_2 \\
& b_1 \quad b_2 \\
& y_1 \quad y_2 \\
& z_1 \quad z_2
\end{align*}
```
Some notation

- The graph $G$ is defined by its clique tree.
- Black lines identify the same vertex.
- The vertices of each bag induce a clique in $G$.
Some notation

- The graph $G$ is defined by its clique tree.
- Black lines identify the same vertex.
- The vertices of each bag induce a clique in $G$.
- Only some edges are shown (in gray).

Frank Kammer
University of Augsburg
Observation

\( a_1 \) and \( a_2 \) can be connected to
Observation

\(a_1\) and \(a_2\) can be connected to only \(y_1\) and \(z_1\)

if \(b_1\) and \(b_2\) also want to be connected.
Observation

$a_1$ and $a_2$ can be connected to only $y_1$ and $z_1$
if $b_1$ and $b_2$ also want to be connected.

all vertices are used by the paths!
NP-hardness of the DPP on Chordal Graphs

Reduction from 1-in-3 SAT restricted to monotone cubic formulas
Reduction from 1-in-3 SAT restricted to monotone cubic formulas
Create a blue gadget for each variable.
Reduction from 1-in-3 SAT restricted to monotone cubic formulas

Create a yellow gadget for each clause.
NP-hardness of the DPP on Chordal Graphs

Reduction from 1-in-3 SAT restricted to monotone cubic formulas

If a variable $x_1$ occurs in a clause $C_1$, identify a square vertex in the gadgets for $x_1$ and $C_1$. 
Reduction from 1-in-3 SAT restricted to monotone cubic formulas

If a variable $x_1$ occurs in a clause $C_1$, identify a square vertex as well as a triangle vertex in the gadgets for $x_1$ and $C_1$. 

Frank Kammer
Reduction from 1-in-3 SAT restricted to monotone cubic formulas

Moreover, for some $i, j \in \{1, 2, 3\}$ (e.g., $i = 2$ and $j = 1$) the instance has a terminal-pair $(a_i, y_j)$ and $(b_i, z_j)$. 
Reduction from 1-in-3 SAT restricted to monotone cubic formulas

The vertex identification and the terminal pairs \((a_i, y_j), (b_i, z_j)\) is applied to each occurrence of a variable in a clause.
Reduction from 1-in-3 SAT restricted to monotone cubic formulas

The vertex identification and the terminal pairs \((a_i, y_j), (b_i, z_j)\) is applied to each occurrence of a variable in a clause.
To show: 1-in-3 truth assignment $\Rightarrow$ solution for the DPP

- variable is true $\Rightarrow$
  - a-terminal $\sim\triangledown\sim$ y-terminal

- variable is false $\Rightarrow$
  - a-terminal $\sim\blacksquare\sim$ y-terminal
To show: 1-in-3 truth assignment $\iff$ solution for the DPP

**Lemma:** All paths from the A-terminals leave a variable gadget by the $\triangle$ or $\blacksquare$-vertices.
To show: 1-in-3 truth assignment $\iff$ solution for the DPP

**Lemma:** All paths from the A-terminals leave a variable gadget by the $\triangle$ or $\blacksquare$-vertices. **Def:** A-Paths use $\triangle \iff$ variable is true
NP-hardness of the DPP on Chordal Graphs

1-in-3 truth assignment $\iff$ solution for the DPP
1-in-3 truth assignment ⇔ solution for the DPP

DPP is NP-hard on chordal graphs.
Generalizations

New results by generalizing algorithms on trees

On undirected chordal n-vertex graphs:

- Linear time for the k-DPP  
  (i.e., DPP is FPT on chordal graphs)
- DPP is NP-hard

Next, some ideas to generalize algorithms on chordal graphs by introducing 3 complexity parameters.
Definition

The neighbors of each vertex can be divided into \( k \) cliques.

\( k \)-approximation for

- Minimum Dominating Set
- Minimum Independent Dominating Set
- Maximum Weighted Independent Set
- ...
Definition

The neighbors of each vertex can be divided into $k$ cliques.

$k$-approximation for

- Minimum Dominating Set
- Minimum Independent Dominating Set
- Maximum Weighted Independent Set
- ...
$k$-groupable graphs: unit-disk graphs

Frank Kammer
unit disk graphs are 7-groupable

\[ R = (2 \cos(30) - 1)r \]
unit disk graphs are 7-groupable

\[ R = (2 \cos(30) - 1)r \]
unit disk graphs are 7-groupable

\[ R = (2 \cos(30) - 1)r \]
$k$-groupable graphs: unit-square graphs
unit square graphs are 10-groupable
**Definition**

There is a numbering of the vertices such that, for each vertex, its incident vertices of larger number can be divided into $k$ cliques.

*called successor cliques*

**$k$-approximation for**

- Maximum Weighted Independent Set
- Maximum Weighted Clique
- Minimum Clique Partition
- ...

---

**k-eliminable graphs**

Frank Kammer
University of Augsburg
**Definition**

There is a numbering of the vertices such that, for each vertex, its incident vertices of larger number can be divided into $k$ cliques.

-called successor cliques-

**$k$-approximation for**

- Maximum Weighted Independent Set
- Maximum Weighted Clique
- Minimum Clique Partition
- ...
Lemma

Disk graphs are 7-eliminable.
For each object, ratio of the radius of the outball and of the inball is bounded.
**Definition**

There is an orientation of the edges such that the endpoints of the outgoing edges of each vertex can be partitioned into \( k \) cliques.

---

**2k-approximation for**

- Maximum Weighted Independent Set
- Maximum Weighted Clique
- Minimum Vertex Coloring
- ...
Definition

There is an orientation of the edges such that the endpoints of the outgoing edges of each vertex can be partitioned into $k$ cliques.

2$k$-approximation for

- Maximum Weighted Independent Set
- Maximum Weighted Clique
- Minimum Vertex Coloring
- ...
For each object, ratio of the radius of the outball and of the inball is bounded.
Max Independent Set on $k$-eliminable graphs

An algorithm with a $k$-approximation:

- Set $S = \emptyset$.
- Process vertices in order of the numbering:
  - For $v$, test if not deleted (i.e., $N(v) \cap S = \emptyset$).
  - If so, add $v$ into $S$ and delete $N[v]$.
- Return $S$. 
An algorithm with a $2k$-approximation:

- Find order $v_1, \ldots, v_n$ with $v_i$ in $G[v_i, \ldots, v_n]$ having not less outgoing than incoming edges.
An algorithm with a $2k$-approximation:

- Find order $v_1, \ldots, v_n$ with $v_i$ in $G[v_i, \ldots, v_n]$ having not less outgoing than incoming edges.
- Color $v_n, v_{n-1}, \ldots, v_1$ by the lowest available number in $\mathbb{N}$.

$G[v_i, \ldots, v_n]$:

- Largest successor clique: size $x$
- $|\text{incoming edges}| \leq |\text{outgoing edges}|$
Min Vertex Coloring on $k$-orientable graphs

An algorithm with a $2k$-approximation:

- Find order $v_1, \ldots, v_n$ with $v_i$ in $G[v_i, \ldots, v_n]$ having not less outgoing than incoming edges.
- Color $v_n, v_{n-1}, \ldots, v_1$ by the lowest available number in $\mathbb{N}$.
- In every step, the assigned number is less $2k$ times OPT.

$$|\text{incoming edges}| \leq |\text{outgoing edges}|$$
The 3 Parameters

The interaction of the graph classes

- $k$-groupable
- $k$-eliminable
- $k$-orientable
- Chordal
- Treewidth $k$

Lemma: All 3 parameters are NP-hard.

Proof: Reduction from minimum clique partition problem.