Constraint Satisfaction and Fair Multi-Objective Optimization Problems: Foundations, Complexity, and Islands of Tractability

Gianluigi Greco and Francesco Scarcello
Constraint Satisfaction Problems

- Distribute the goods/presents to the kids
- Goods are indivisible
Constraint Satisfaction Problems

- Distribute the goods/presents to the kids
- Goods are indivisible

Variables
- \( K_1, K_2, \text{ and } K_3 \)

Domain
- \( g_1, g_2, ..., g_7 \)

Constraints
- \( K_1 \in \{g_1, g_2, g_3\} \)
- \( K_2 \in \{g_2, g_3, g_4, g_5\} \)
- \( K_3 \in \{g_4, g_5, g_6, g_7\} \)
- \( K_1 \neq K_2, K_2 \neq K_3 \)
• Distribute the goods/presents to the kids
• Goods are indivisible

**Solution**
- $K_1 \rightarrow g_2$
- $K_2 \rightarrow g_4$
- $K_3 \rightarrow g_6$

**Variables**
- $K_1, K_2, K_3$

**Domain**
- $g_1, g_2, \ldots, g_7$

**Constraints**
- $K_1 \in \{g_1, g_2, g_3\}$
- $K_2 \in \{g_2, g_3, g_4, g_5\}$
- $K_3 \in \{g_4, g_5, g_6, g_7\}$
- $K_1 \neq K_2, K_2 \neq K_3$
Optimization Functions in CSPs

- Distribute the goods/presents to the kids
- Goods are indivisible

- Kids have preferences over the presents

- Valuation Function: $\mathcal{F} = \langle w, + \rangle$
  - $w(K_1/g_1) = 3$
  - $w(K_1/g_2) = 10$
  - $\ldots$

- Value of the Solution
  - $w(K_1/g_1) + w(K_2/g_4) + w(K_3/g_6) = 21$

- Optimal (MAX) Solution
  - Maximizes the social welfare
Multi-Objective Optimization

- Different Valuations: \( \{F_1, F_2, F_3\} \)
  - \( F_h \) is defined on \( K_h \)
- Combination Strategies:
Multi-Objective Optimization

- **Different Valuations:** $\{\mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3\}$
  - $\mathcal{F}_h$ is defined on $K_h$

- **Combination Strategies:**
  - MAX-SUM *(social welfare)*: 10 2 9
Multi-Objective Optimization

- Different Valuations: \( \{F_1, F_2, F_3\} \)
  - \( F_h \) is defined on \( K_h \)

- Combination Strategies:
  - MAX-SUM (social welfare) \( \begin{array}{c} 10 \ 2 \ 9 \end{array} \)
  - LEX \( \begin{array}{c} 10 \ 3 \ 2 \end{array} \)
Multi-Objective Optimization

- **Different Valuations:** \( \mathcal{F}_h \) is defined on \( K_h \)
- **Combination Strategies:**
  - MAX-SUM (social welfare) \( \begin{array}{c} \text{10} \\ \text{2} \\ \text{9} \end{array} \)
  - LEX \( \begin{array}{c} \text{10} \\ \text{3} \\ \text{2} \end{array} \)
  - PARETO \( \begin{array}{c} \text{3} \\ \text{6} \\ \text{9} \end{array} \)

\( g_1 \) \( g_2 \) \( g_3 \) \( g_4 \) \( g_5 \) \( g_6 \) \( g_7 \)
Related Literature

- **Different Valuations:**
  - $F_h$ is defined on $K_h$

- **Combination Strategies:**
  - MAX-SUM (social welfare)  
    - 10 2 9  
    - [Bistarelli et al., Rossi et al.]
  - LEX  
    - 10 3 2  
    - [Freuder et al.]
  - PARETO  
    - 3 6 9  
    - [Torrens and Faltings]
Santa’s goal is to distribute presents in a way that the least lucky kid is as happy as possible.
Related Literature

The Santa Claus Problem:

Santa’s goal is to distribute presents in a way that the least lucky kid is as happy as possible.

Social welfare = 19 (max 21)
The Santa Claus Problem:

Santa’s goal is to distribute presents in a way that the least lucky kid is as happy as possible.

Social welfare = 19 (max 21)

Related Literature

FAIR OPTIMIZATION

- MAX-MIN

[Snow and Freuder, Dubois and Fortemps, Bouveret and Lematre]
Santa’s goal is to distribute presents in a way that the least lucky kid is as happy as possible.

Social welfare = 19 (max 21)

Related Literature

The Santa Claus Problem:

FAIR OPTIMIZATION

- MAX-MIN

Limited expressiveness
- functions on one variable/constraint

No complexity analysis

[Snow and Freuder, Dubois and Fortemps, Bouveret and Lematre]
Overview

Complexity

Model

Decomposition Methods
The Model

- $L = \{ \mathcal{F}_1, \ldots, \mathcal{F}_n \}$ is a set of valuation functions

- $\mathcal{F}_i = \langle w_i, \oplus_i \rangle$ is such that
  - $w_i : \bar{X}_i \times \mathcal{U} \mapsto \mathbb{R}$, with $\bar{X}_i \subseteq \text{Var}$
  - $\oplus_i$ is a commutative, associative, and closed binary operator

- $\mathcal{F}_i(\theta) = \bigoplus \{X/u \in \theta | X \in \bar{X}_i\} w_i(X, u)$

\[\max_\theta \ \min_{\mathcal{F} \in L} \mathcal{F}(\theta)\]
The Model

- $L = \{ F_1, \ldots, F_n \}$ is a set of valuation functions
- $F_i = \langle w_i, \oplus_i \rangle$ is such that
  - $w_i : \overline{X}_i \times \mathcal{U} \mapsto \mathbb{R}$, with $\overline{X}_i \subseteq \text{Var}$
  - $\oplus_i$ is a commutative, associative, and closed binary operator
- $F_i(\theta) = \bigoplus \{ X/u \in \theta | X \in \overline{X}_i \} w_i(X, u)$

The Santa Claus Problem:

(possible solutions)

\[
\begin{array}{ccc}
F_1 & F_2 & F_3 \\
10 & 2 & 9 \\
10 & 3 & 2 \\
3 & 6 & 9 \\
4 & 6 & 6 \\
4 & 6 & 9 \\
\vdots & \vdots & \vdots \\
\end{array}
\]

$\max_\theta \ \min_{F \in L} F(\theta)$
The Model

- \( L = \{ \mathcal{F}_1, \ldots, \mathcal{F}_n \} \) is a set of valuation functions
- \( \mathcal{F}_i = \langle w_i, \oplus_i \rangle \) is such that
  - \( w_i : \bar{X}_i \times \mathcal{U} \mapsto \mathbb{R} \), with \( \bar{X}_i \subseteq \text{Var} \)
  - \( \oplus_i \) is a commutative, associative, and closed binary operator
- \( \mathcal{F}_i(\theta) = \bigoplus \{ X/\forall u \in \theta | X \in \bar{X}_i \} \ w_i(X, u) \)

The Santa Claus Problem:
(possible solutions)

\[
\begin{array}{ccc}
{\mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3} \\
10 & 2 & 9 \\
10 & 3 & 2 \\
3 & 6 & 9 \\
4 & 6 & 6 \\
4 & 6 & 9 \\
\vdots & \vdots & \vdots \\
\end{array}
\]

\[
\max_{\theta} \ \min_{\mathcal{F} \in L} \mathcal{F}(\theta)
\]
The Model

- \( L = \{ F_1, \ldots, F_n \} \) is a set of valuation functions
- \( F_i = \langle w_i, \oplus_i \rangle \) is such that
  - \( w_i : \bar{X}_i \times U \rightarrow \mathbb{R}, \) with \( \bar{X}_i \subseteq Var \)
  - \( \oplus_i \) is a commutative, associative, and closed binary operator
- \( F_i(\theta) = \bigoplus \{ X/u \in \theta|X \in \bar{X}_i \} w_i(X, u) \)

The Santa Claus Problem:

\( \{ F_1, F_2, F_3 \} \)

\[
\begin{array}{ccc}
10 & 2 & 9 \\
10 & 3 & 2 \\
3 & 6 & 9 \\
4 & 6 & 6 \\
4 & 6 & 9 \\
\vdots & \vdots & \vdots \\
\end{array}
\]

\( \max_\theta \min_{F \in L} F(\theta) \)

\( \text{lex} \max_\theta \min_{F \in L} F(\theta) \)
The Model

- \( L = \{ \mathcal{F}_1, \ldots, \mathcal{F}_n \} \) is a set of valuation functions
- \( \mathcal{F}_i = \langle w_i, \oplus_i \rangle \) is such that
  - \( w_i : \tilde{X}_i \times \mathcal{U} \rightarrow \mathbb{R} \), with \( \tilde{X}_i \subseteq \text{Var} \)
  - \( \oplus_i \) is a commutative, associative, and closed binary operator
- \( \mathcal{F}_i(\theta) = \bigoplus \{ X/u \in \theta \mid X \in \tilde{X}_i \} \ w_i(X, u) \)

**The Santa Claus Problem:**

| \{\mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3\} |
|---|---|---|
| 10 2 9 |
| 10 3 2 |
| 3 6 9 |
| 4 6 6 |
| 4 6 9 |

(possible solutions)

\[ \max_{\theta} \min_{\mathcal{F} \in L} \mathcal{F}(\theta) \]

\[ \text{lex} \max_{\theta} \min_{\mathcal{F} \in L} \mathcal{F}(\theta) \]
Overview

Complexity

Decomposition Methods

Model
Complexity of (LEX)Max-Min Solutions

- Constraint satisfaction is NP-hard
  - Even without optimization functions…

- Tractable classes of CSPs
  - Based on the values in the constraint relations
  - Based on the structure of the constraint scopes
Complexity of (LEX)Max-Min Solutions

- **Constraint satisfaction is NP-hard**
  - Even without optimization functions…

- **Tractable classes of CSPs**
  - Based on the values in the constraint relations
  - Based on the structure of the constraint scopes
    - Treewidth [Dechter & Pearl]
Complexity of (LEX)Max-Min Solutions

- Constraint satisfaction is NP-hard
  - Even without optimization functions…

- Tractable classes of CSPs
  - Based on the values in the constraint relations
  - Based on the structure of the constraint scopes

 CONSTRAINT HYPERGRAPH

\[ K_1 \neq K_2 \]

\[ K_1 \cap K_3 \]

\[ K_2 \neq K_3 \]
Complexity of (LEX)Max-Min Solutions

- **Constraint satisfaction is NP-hard**
  - Even without optimization functions…

- **Tractable classes of CSPs**
  - Based on the values in the constraint relations
  - Based on the structure of the constraint scopes

**CONSTRAINT HYPERGRAPH**

- $K_1 \neq K_2$
- $K_2 \neq K_3$
- $K_1$ and $K_4$ are further variable/constraints
Complexity of (LEX)Max-Min Solutions

- JOIN TREE
  - Vertices correspond to the hyperedges
  - Each variable induces a connected subtree

- CONSTRAINT HYPERGRAPH

![Diagram with hyperedges and variable/constraint relationships]
Complexity of (LEX)Max-Min Solutions

- **JOIN TREE**
  - Vertices correspond to the hyperedges
  - Each variable induces a connected subtree

- **CONSTRAINT HYPERGRAPH**

  Further variable/constraints

\[ K_1 \neq K_2 \]
\[ K_2 \neq K_3 \]
Complexity of (LEX)Max-Min Solutions

- **JOIN TREE**
  - Vertices correspond to the hyperedges
  - Each variable induces a connected subtree

- **ACYCLIC CSP**

- **CONSTRAINT HYPERGRAPH**
  - Further variable/constraints
  - $K_1 \neq K_2$
  - $K_2 \neq K_3$
  - $K_1 \neq K_2$
  - $K_2 \neq K_3$
### Complexity of Acyclic Instances

#### Restrictions

- $\text{max}_{\mathcal{F} \in L} |\text{dom}(\mathcal{F})| \leq D$
- $|L| \leq F$

<table>
<thead>
<tr>
<th>$[D, F]$</th>
<th>1</th>
<th>$h$</th>
<th>$\infty$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$k$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\infty$</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Complexity of Acyclic Instances

<table>
<thead>
<tr>
<th>([D, F])</th>
<th>1</th>
<th>(h)</th>
<th>(\infty)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(k)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(\infty)</td>
<td></td>
<td>(\text{in P})</td>
<td></td>
</tr>
</tbody>
</table>

[Gottlob et al.]

- **Restrictions on** \(L = \{\mathcal{F}_1, \ldots, \mathcal{F}_n\}\)
  - \(\max_{\mathcal{F} \in L} |\text{dom}(\mathcal{F})| \leq D\)
  - \(|L| \leq F\)
Complexity of Acyclic Instances

<table>
<thead>
<tr>
<th>[D, F]</th>
<th>1</th>
<th>h</th>
<th>∞</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>in P</td>
<td></td>
<td></td>
</tr>
<tr>
<td>k</td>
<td>in P</td>
<td></td>
<td></td>
</tr>
<tr>
<td>∞</td>
<td>in P</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

[Gottlob et al.]

- **Restrictions on** $L = \{F_1, \ldots, F_n\}$
  - $\max_{F \in L} |\text{dom}(F)| \leq D$
  - $|L| \leq F$
### Complexity of Acyclic Instances

#### Restrictions

- **[Gottlob et al.]**

<table>
<thead>
<tr>
<th>$[D, F]$</th>
<th>1</th>
<th>$h$</th>
<th>$\infty$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><strong>in P</strong></td>
<td></td>
<td><strong>in P</strong></td>
</tr>
<tr>
<td>$k$</td>
<td><strong>in P</strong></td>
<td></td>
<td><strong>in P</strong></td>
</tr>
<tr>
<td>$\infty$</td>
<td><strong>in P</strong></td>
<td></td>
<td><strong>in P</strong></td>
</tr>
</tbody>
</table>

*Dynamic programming*

---

**Restrictions on** $L = \{\mathcal{F}_1, \ldots, \mathcal{F}_n\}$

- $\max_{\mathcal{F} \in L} |\text{dom}(\mathcal{F})| \leq D$
- $|L| \leq F$
### Complexity of Acyclic Instances

#### Restrictions

- **Restrictions on** $L = \{ \mathcal{F}_1, \ldots, \mathcal{F}_n \}$
  - $\max_{\mathcal{F} \in L} |\text{dom}(\mathcal{F})| \leq D$
  - $|L| \leq F$

#### Dynamic Programming

<table>
<thead>
<tr>
<th>$[D, F]$</th>
<th>1</th>
<th>$h$</th>
<th>$\infty$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>in P</td>
<td>in P</td>
<td>in P</td>
</tr>
<tr>
<td>$k$</td>
<td>in P</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\infty$</td>
<td>in P</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Complexity of Acyclic Instances

<table>
<thead>
<tr>
<th>[D, F]</th>
<th>1</th>
<th>h</th>
<th>(\infty)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(\in P)</td>
<td>(\in P)</td>
<td>(\in P)</td>
</tr>
<tr>
<td>(k)</td>
<td>(\in P)</td>
<td>(\in P)</td>
<td>(\in P)</td>
</tr>
<tr>
<td>(\infty)</td>
<td>(\in P)</td>
<td>weakly (\text{NP-hard})</td>
<td>(\in P)</td>
</tr>
</tbody>
</table>

- Reduction from «Partition»
- Pseudo-polynomial

Restrictions on \(L = \{\mathcal{F}_1, \ldots, \mathcal{F}_n\}\)
- \(\max_{\mathcal{F} \in L} |\text{dom}(\mathcal{F})| \leq D\)
- \(|L| \leq F\)
## Complexity of Acyclic Instances

### Restrictions

- **Restrictions on** $L = \{ F_1, \ldots, F_n \}$
  - $\max_{F \in L} |\text{dom}(F)| \leq D$
  - $|L| \leq F$

### Table

<table>
<thead>
<tr>
<th>$[D, F]$</th>
<th>$1$</th>
<th>$h$</th>
<th>$\infty$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1$</td>
<td>in P</td>
<td>in P</td>
<td>in P</td>
</tr>
<tr>
<td>$k$</td>
<td>in P</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\infty$</td>
<td>in P</td>
<td>weakly NP-hard</td>
<td>NP-hard</td>
</tr>
</tbody>
</table>

### Reductions
- Reduction from «Partition»
- Reduction from «Set Packing»

### Dynamic Programming
- Pseudo-polynomial

---

[Gottlob et al.]
### Complexity of Acyclic Instances

<table>
<thead>
<tr>
<th>([D, F])</th>
<th>1</th>
<th>(h)</th>
<th>(\infty)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(\text{in } P)</td>
<td>(\text{in } P)</td>
<td>(\text{in } P)</td>
</tr>
<tr>
<td>(k)</td>
<td>(\text{in } P)</td>
<td>(\text{weakly NP-hard})</td>
<td>(\text{NP-hard})</td>
</tr>
<tr>
<td>(\infty)</td>
<td>(\text{in } P)</td>
<td></td>
<td>(\text{NP-hard})</td>
</tr>
</tbody>
</table>

- **Restrictions on** \(L = \{F_1, \ldots, F_n\}\)
  - \(\max_{F \in L} |\text{dom}(F)| \leq D\)
  - \(|L| \leq F\)

- Reduction from «Set Packing»
- Pseudo-polynomial
- Dynamic programming

---

[Gottlob et al.]

---
### Complexity of Acyclic Instances

<table>
<thead>
<tr>
<th>([D, F])</th>
<th>1</th>
<th>(h)</th>
<th>(\infty)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1)</td>
<td>in (P)</td>
<td>in (P)</td>
<td>in (P)</td>
</tr>
<tr>
<td>(k)</td>
<td>in (P)</td>
<td>(\text{in } P)</td>
<td>NP-hard</td>
</tr>
<tr>
<td>(\infty)</td>
<td>in (P)</td>
<td>weakly NP-hard</td>
<td>NP-hard</td>
</tr>
</tbody>
</table>

- **Restrictions on** \(L = \{F_1, \ldots, F_n\}\)
  - \(\max_{F \in L} \vert \text{dom}(F) \vert \leq D\)
  - \(\vert L \vert \leq F\)

- Reduction from «Set Packing»
- Reduction from «Partition»
- Dynamic programming

[Gottlob et al.]

---

[D, F]: \([D, F]\) represents the domain and feature attributes.

\(h\): A parameter indicating a specific condition.

\(\infty\): Indicates an upper bound.

**in \(P\)**: Indicates the problem is solvable in polynomial time.

**NP-hard**: Indicates the problem is NP-hard.

**weakly NP-hard**: Indicates the problem is weakly NP-hard.

\(\text{a novel machinery is needed}\)**: Indicates the need for a new approach.
Guards for Valuation Functions

Decomposition Methods
Guards for Valuation Functions

- A set of variables $W$ is a *guard* for a valuation function if
  - separates the hypergraph in components where its domain variables do not occur together with any variable occurring in other valuation functions
A set of variables $W$ is a guard for a valuation function if
- separates the hypergraph in components where its domain variables do not occur together with any variable occurring in other valuation functions.
Guards for Valuation Functions

- A set of variables $W$ is a **guard** for a valuation function if
  - separates the hypergraph in components where its domain variables do not occur together with any variable occurring in other valuation functions.
Guards for Valuation Functions

A set of variables $W$ is a guard for a valuation function if

- separates the hypergraph in components where its domain variables do not occur together with any variable occurring in other valuation functions.
Guards for Valuation Functions

- A set of variables $W$ is a guard for a valuation function if
  - separates the hypergraph in components where its domain variables do not occur together with any variable occurring in other valuation functions

$W$ defines 3 components

- covers $X_1$, which is in the domain of $F_1$
- $C_2$ (and $C_4$) does not contain variables in the domain of $F_2$ and $F_3$
Guards for Valuation Functions

defines 3 components

covers \( X_1 \), which is in the domain of \( \mathcal{F}_1 \)

\( \mathcal{C}_2 \) (and \( \mathcal{C}_1 \)) does not contain variables in the domain of \( \mathcal{F}_2 \) and \( \mathcal{F}_3 \)

is a guard for \( \mathcal{F}_1 \); in fact, it is also a guard for the other functions
Key Ideas

Guards for Valuation Functions

Decomposition Methods
Decomposition Methods

- **Common Ideas**
  - Generalize the notion of graph or hypergraph acyclicity
  - Associate a width to each instance, expressing its degree of cyclicity
  - Polynomial time algorithms for bounded-width CSP instances, running in $O(n^{w+1} \cdot \log n)$
  - Bounded-width CSP instances can be recognized in polynomial time
  - Bounded-width decompositions can be computed in polynomial time

- **Noticeable Examples**
  - Tree decompositions
  - (Generalized) Hypertree decompositions
Generalized Hypertree Decompositions

\[ a(S, X, X', C, F) \quad b(S, Y, Y', C', F') \quad c(C, C', Z) \quad d(X, Z) \]
\[ e(Y, Z) \quad f(F, F', Z') \quad g(X', Z') \quad h(Y', Z') \]
\[ j(J, X, Y, X', Y') \quad p(B, X', F) \quad q(B', X', F) \]

\[ j(J, X, Y, X', Y') \]

\[ a(S, X, X', C, F), \ b(S, Y, Y', C', F') \]

\[ j(_, X, Y, _, _), \ c(C, C', Z) \]
\[ d(X, Z) \quad e(Y, Z) \]

\[ j(_, _, X', Y'), \ f(F, F', Z') \]
\[ g(X', Z'), \ f(F, _, Z') \]
\[ h(Y', Z') \]

\[ p(B, X', F) \quad q(B', X', F) \]
Basic Conditions

- We group edges

\[ j(J,X,Y,X',Y') \]

\[ a(S,X,X',C,F), \ b(S,Y,Y',C',F') \]

\[ j(_,X,Y,_,_), \ c(C,C',Z) \]

\[ j(_,_,X',Y'), \ f(F,F',Z') \]

\[ d(X,Z) \]

\[ e(Y,Z) \]

\[ g(X',Z'), \ f(F,_,Z') \]

\[ h(Y',Z') \]

\[ p(B,X',F) \]

\[ q(B',X',F) \]
Basic Conditions (2/2)

Edges can partially be used
Connectdness Condition

\[ j(J,X,Y,X',Y') \]

\[ a(S,X,X',C,F), b(S,Y,Y',C',F') \]

\[ j(_,X,Y,_,_), c(C,C',Z) \]

\[ d(X,Z), e(Y,Z) \]

\[ j(_,_,_,X',Y'), f(F,F',Z') \]

\[ g(X',Z'), f(F,_,Z') \]

\[ h(Y',Z') \]

\[ p(B,X',F), q(B',X',F) \]
Hypertree Decompositions (HTD)

HTD = Generalized HTD + Special Condition

Each variable not used at some vertex $v$

Does not appear in the subtrees rooted at $v$
Guards for Valuation Functions

Decomposition Methods
Decomposition Methods and Guards

Input CSP $\Psi_{hw}$ \rightarrow \{A_1 \rightarrow \cdots \rightarrow A_n\} set of equivalent acyclic instances
Decomposition Methods and Guards

An instance is guarded via the given method if there is an output $A_i$ such that each valuation function is guarded by some hyperedge.
An instance is guarded via the given method if there is an output $A_i$ such that each valuation function is guarded by some hyperedge.

**Decomposition Methods and Guards**

Input CSP $\Psi_{hw} \rightarrow$ set of equivalent acyclic instances $\{A_1, \ldots, A_n\}$

**Is guarded via hypertree decomposition (width k=3)**

- $F_1$
- $F_2$
- $F_3$
## Main Results

<table>
<thead>
<tr>
<th>$[D, F]$</th>
<th>1</th>
<th>$h$</th>
<th>$\infty$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>in $P$</td>
<td>in $P$</td>
<td>in $P$</td>
</tr>
<tr>
<td>$k$</td>
<td>in $P$</td>
<td></td>
<td>NP-hard</td>
</tr>
<tr>
<td>$\infty$</td>
<td>in $P$</td>
<td>weakly NP-hard</td>
<td>NP-hard</td>
</tr>
</tbody>
</table>
### Main Results

<table>
<thead>
<tr>
<th>[D, F]</th>
<th>1</th>
<th>h</th>
<th>(\infty)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 (k)</td>
<td>(\in P)</td>
<td>(\in P)</td>
<td>(\in P)</td>
</tr>
<tr>
<td>(\infty)</td>
<td>(\in P)</td>
<td>weakly NP-hard</td>
<td>NP-hard</td>
</tr>
</tbody>
</table>

in \(P\), if guarded via \(\Psi_{hw}\)
Proof Idea

Guard for $\mathcal{F}_1$

in $P$, if guarded via $\Psi_{hw}$

variables in the domain of $\mathcal{F}_1$. 
Proof Idea

in $\mathcal{P}$, if guarded via $\Psi_{hw}$

guard for $\mathcal{F}_1[X_1]$

variables in the domain of $\mathcal{F}_1$. 
Proof Idea

In $P$, if guarded via $\Psi_{hw}$, solutions with optimal values, computed via dynamic programming, in $P$, if guarded via $\Psi_{hw}$

variables in the domain of $F_1$.
Proof Idea

variables in the domain of $F_1$.

solutions with optimal values, computed via dynamic programming

acyclic instance, with 1 function over $n$ variables

in $P$, if guarded via $\Psi_{hw}$
**Main Results**

<table>
<thead>
<tr>
<th>[D, F]</th>
<th>1</th>
<th>h</th>
<th>$\infty$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1$</td>
<td>in $P$</td>
<td>in $P$</td>
<td>in $P$</td>
</tr>
<tr>
<td>$k$</td>
<td>in $P$</td>
<td></td>
<td>NP-hard</td>
</tr>
<tr>
<td>$\infty$</td>
<td>in $P$</td>
<td>weakly NP-hard</td>
<td>NP-hard</td>
</tr>
</tbody>
</table>

in $P$, if guarded via $\Psi_{hw}$
### Main Results

<table>
<thead>
<tr>
<th>[D, F]</th>
<th>1</th>
<th>h</th>
<th>∞</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/k</td>
<td>(\text{in } P)</td>
<td>(\text{in } P)</td>
<td>(\text{in } P)</td>
</tr>
<tr>
<td>(\infty)</td>
<td>(\text{in } P)</td>
<td>\text{weakly NP-hard}</td>
<td>\text{NP-hard}</td>
</tr>
</tbody>
</table>

(acyclic) instances of this kind are always guarded via \(\Psi_{hw}\) width: \(h \times k + 1\)
# Main Results

<table>
<thead>
<tr>
<th></th>
<th>(D, F)</th>
<th>1</th>
<th>(h)</th>
<th>(\infty)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\frac{1}{k})</td>
<td>in (P)</td>
<td>in (P)</td>
<td>in (P)</td>
<td>in (P)</td>
</tr>
<tr>
<td>(\infty)</td>
<td>in (P)</td>
<td>weakly (NP)-hard</td>
<td>NP-hard</td>
<td>NP-hard</td>
</tr>
</tbody>
</table>

(acyclic) instances of this kind are always guarded via \(\Psi_{hw}\) width: \(h \times k+1\)
Overview

Complexity

Model

Decomposition Methods

Thank you!