# Aggregating Conditionally Lexicographic Preferences Using Answer Set Programming Solvers

**Xudong Liu** and Miroslaw Truszczynski

Department of Computer Science
University of Kentucky
Lexington, KY, USA

# Overview

# Multi-Agent Decision Making

A group of students at UK are deciding a joint vacation in 2014 and they may consider the following issues:

- Time: spring, summer, fall, winter, etc
- Transportation: drive, fly, etc
- Destination: Chicago, Beijing, Miami, Brussels, etc
- Duration: three days, one week, ten days, etc

⋮ ⋮

Combinations of values from the domains of issues form vacation options (or alternatives); in such cases, alternatives come from combinatorial domains. Each student has her own preferences (linear ordering) over the alternatives. A solution is a vacation option considering students' preferences.

# Multi-Agent Decision Making

Aggregating agents' individual preferences over alternatives to achieve collaborative decisions.

1. Alternatives: in *combinatorial* domains
2. Individual preferences: *conditionally lexicographic preferences (LP trees)*
3. Aggregation: based on *positional scoring voting rules*
4. Collaborative decisions: alternatives with the highest score

# Contributions

1. New computational complexity results for the *winner* and the *evaluation* problems based on a special class of positional scoring rules when votes are LP trees
2. Logic programs modeling the aggregation problems
3. Insight into effectiveness of *Answer-Set Programming (ASP)* in modeling and solving these problems
   - Two ASP dialects compared

# Voting in Combinatorial Domains

1. Issues: $\mathcal{I} = \{X_1, X_2, \ldots, X_p\}$, with $D(X_i) = \{0_i, 1_i\}$
2. Alternatives: $\mathcal{X} = D(X_1) \times D(X_2) \times \ldots \times D(X_p)$ , $|\mathcal{X}| = 2^p$ (denoted by $m$)
3. Vote: a strict total order on $\mathcal{X}$
4. Profile: a finite set of votes collected from $n$ voters

# Positional Scoring Rules

1. Scoring vector: $w = (w_0, \ldots, w_{m-1})$ of natural numbers , where $w_0 \geq w_1 \geq \ldots \geq w_{m-1}$ and $w_0 > w_{m-1}$.
2. Let $v = o_0 > o_1 > \ldots > o_{m-1}$ be a vote over $\mathcal{X}$, the score of alternative $o_i$ in vote $v$ is $w_i$, denoted by $s_w(v, o_i)$
3. Let $\mathcal{V}$ be a profile of votes $v$, the score of alternative $o_i$ in profile $\mathcal{V}$:

$$s_w(\mathcal{V}, o_i) = \sum_{v \in \mathcal{V}} s_w(v, o_i)$$

# Positional Scoring Rules

- Borda: $(m-1, m-2, \ldots, 0)$
- $k$-approval: $(1, \ldots, 1, 0, \ldots, 0)$ with $k$ being the number of 1's
- 2-valued $(k, l)$-approval: $(a, \ldots, a, b, \ldots, b, 0 \ldots, 0)$, where $a$ and $b$ are constants $(a > b)$ and the numbers of $a$'s and $b$'s equal to $k$ and $l$, respectively.

# LP Trees

1. An *LP tree* [1] $\mathcal{L}$ over $\mathcal{I} = \{X_1, \ldots, X_p\}$ is a *binary tree*
2. Each node $t$ in $\mathcal{L}$ is labeled by an issue from $\mathcal{I}$, denoted by $lss(t)$, and with *preference information* ($0 > 1$ or $1 > 0$)
3. Each issue appears **exactly once** on each path from the root to a leaf

Intuitively, the issue labeling the root of an LP tree is of highest importance. Alternatives with the preferred value of that issue (upper half) are preferred over alternatives with the non-preferred one (lower half).

[1] Richard Booth, Yann Chevaleyre, Jérôme Lang, Jérôme Mengin, and Chattrakul Sombattheera. Learning conditionally lexicographic preference relations, 2010.

# LP Trees: Example

$D(Time) = \{summer(s), winter(w)\}$, $D(Dest) = \{Chicago(c), Miami(m)\}$,
$D(Tran) = \{drive(d), fly(f)\}$

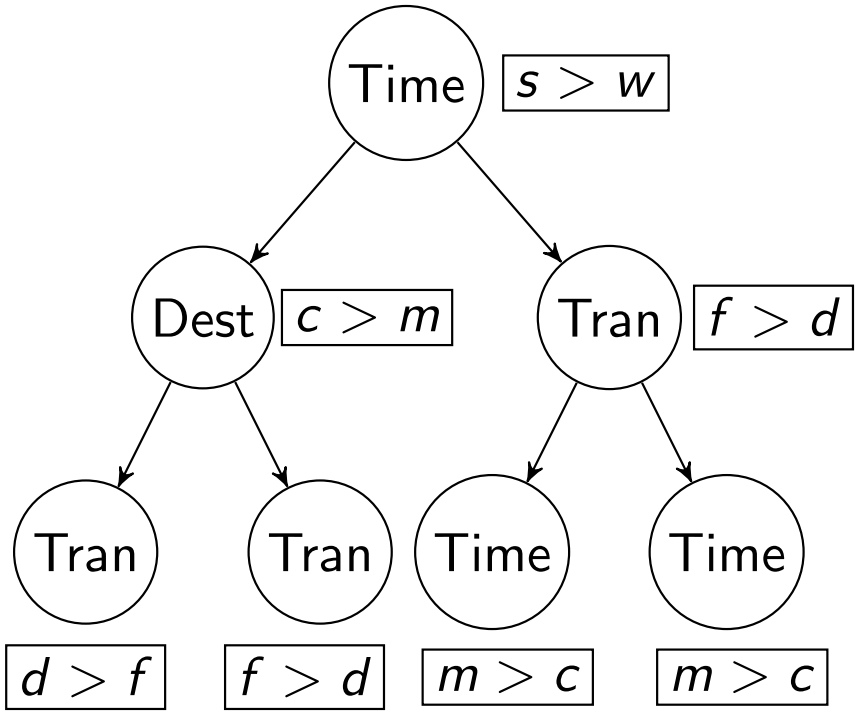
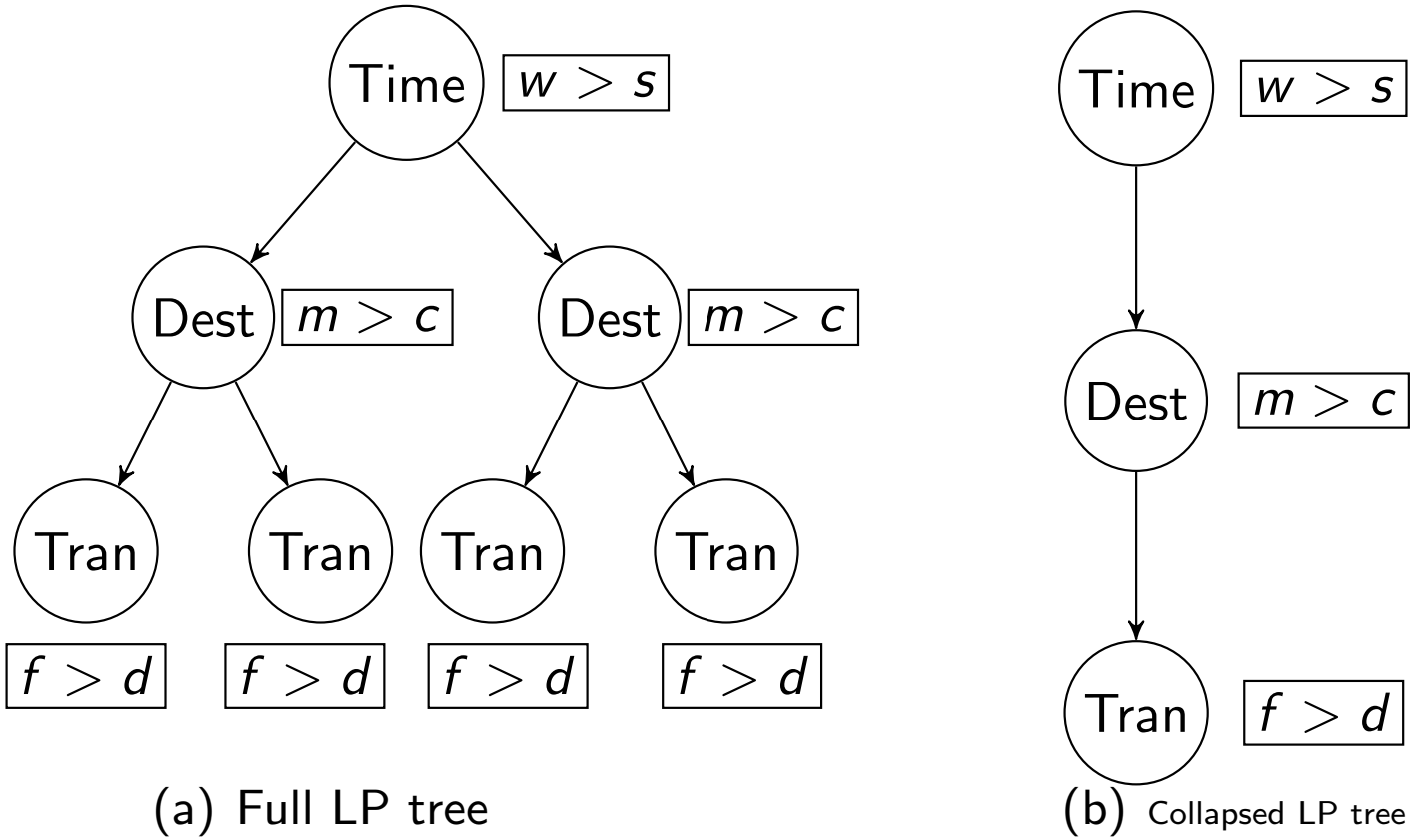



Figure: LP tree

$scd \succ scf \succ smf \succ smd \succ wfm \succ wfc \succ wdm \succ wdc$

# Collapsed LP Trees

$D(Time) = \{summer(s), winter(w)\}$, $D(Dest) = \{Chicago(c), Miami(m)\}$,
$D(Tran) = \{drive(d), fly(f)\}$



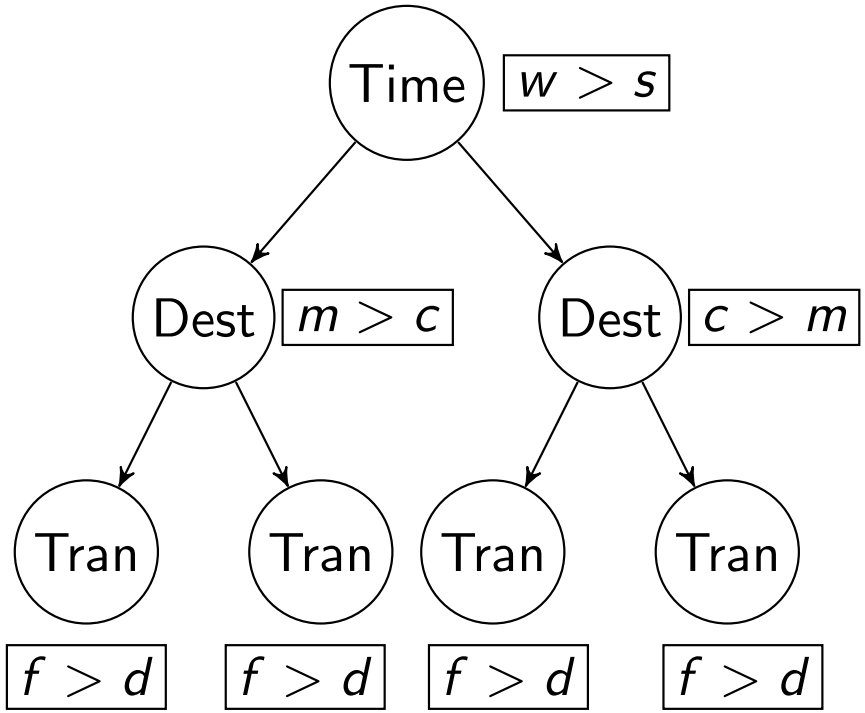

(a) Full LP tree

(b) Collapsed LP tree

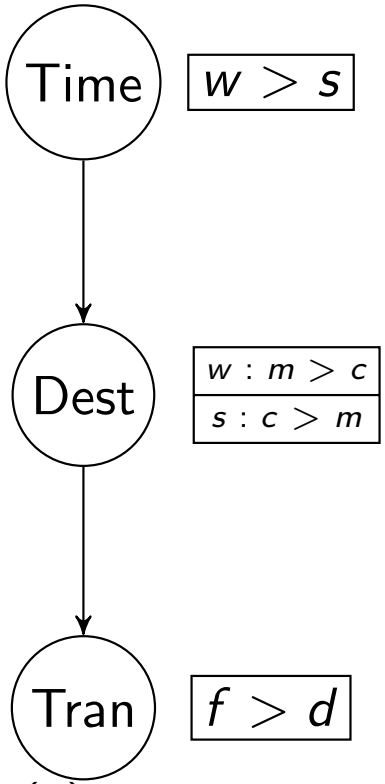
Figure: Collapse to UI-UP

# Collapsed LP Trees

$D(Time) = \{summer(s), winter(w)\}$, $D(Dest) = \{Chicago(c), Miami(m)\}$,
$D(Tran) = \{drive(d), fly(f)\}$




(a) Full LP tree

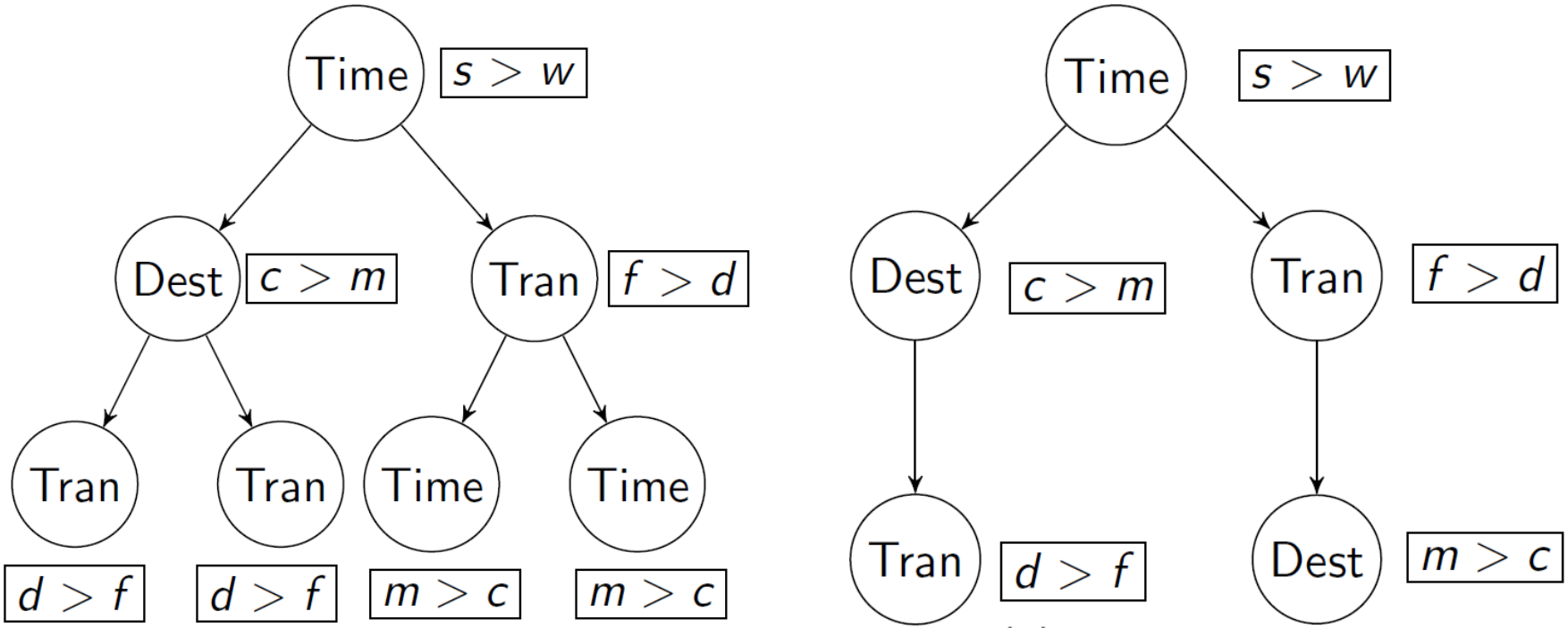



(b) Collapsed LP tree

Figure: Collapse to UI-CP

# Collapsed LP Trees

$D(Time) = \{summer(s), winter(w)\}$, $D(Dest) = \{Chicago(c), Miami(m)\}$,
$D(Tran) = \{drive(d), fly(f)\}$




(a) Full LP tree (b) Collapsed LP tree

Figure: Collapse to CI-UP

# Collapsed LP Trees

$D(Time) = \{summer(s), winter(w)\}$, $D(Dest) = \{Chicago(c), Miami(m)\}$,
$D(Tran) = \{drive(d), fly(f)\}$

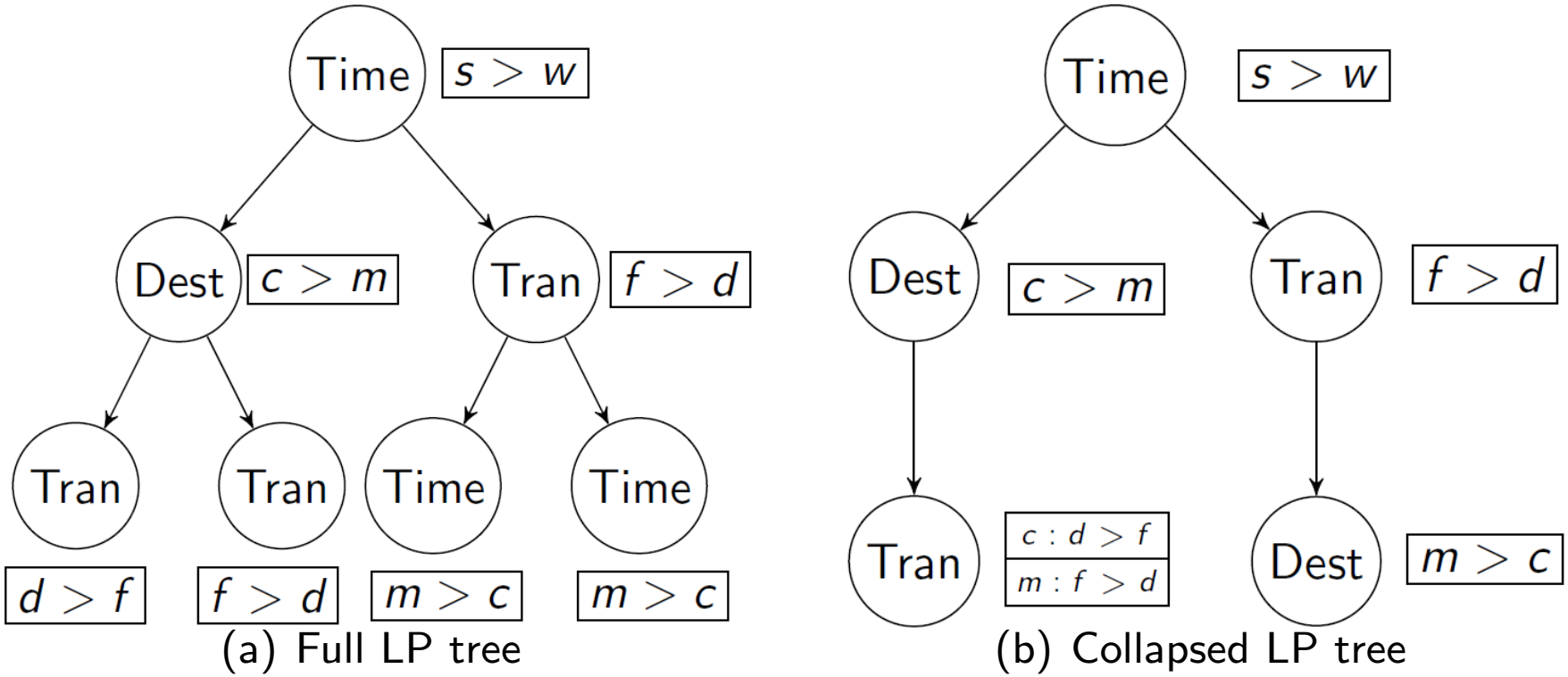



Figure: Collapse to CI-CP

# Voting Rules Studied

## Effective Implicit Positional Scoring Rules

Let $r$ be a positional scoring rule, and $w$ its underlying scoring vector. Rule $r$ is *effective implicit* if, given $m$ and $i$ $(0 \leq i \leq m-1)$, there is an algorithm that computes the value $w_i$ in time polynomial in the sizes of $m$.

- Borda: $w_i = m - i - 1$
- $k$-approval: if $i \leq k-1$, $w_i = 1$; otherwise, $w_i = 0$
- $(k, l)$-approval: if $i \leq k-1$, $w_i = a$; if $k \leq i \leq l-1$, $w_i = b$; otherwise, $w_i = 0$

# Problems

## The Winner Problem

Let $r$ be an effective implicit positional scoring rule with a scoring vector $w$. Given a profile $\mathcal{V}$ of $n$ LP trees over $p$ issues, the *winner* problem is to compute an alternative $o \in \mathcal{X}$ with the maximum score $s_w(\mathcal{V}, o)$.

## The Evaluation Problem

Let $r$ be an effective implicit positional scoring rule with a scoring vector $w$. Given a profile $\mathcal{V}$ of $n$ LP trees over $p$ issues and a positive integer $R$, the *evaluation* problem is to decide whether there exists an alternative $o \in \mathcal{X}$ such that $s_w(\mathcal{V}, o) \geq R$.

# Computational Complexity

- The computational complexity of the winner and the evaluation problems for positional scroing rules has not been fully understood
- Some computational complexity results are known for some special cases, we aim at expanding the space of known results for more general positional scoring rules

# Computational Complexity: Borda

| | CP | UP |
|---|---|---|
| CI | NPC | NPC |
| UI | NPC | P |

(a) Evaluation

| | CP | UP |
|---|---|---|
| CI | NP-Hard | NP-Hard |
| UI | NP-Hard | P |

(b) Winner

Figure: Borda [2]

[2] Jérôme Lang, Jérôme Mengin, and Lirong Xia. Aggregating conditionally lexicographic preferences on multi-issue domains, 2012.

# Computational Complexity: $k$-Approval

1. if $k = 2^{p-1}$, the winner and the evaluation problems are in P.
2. if $k = c \cdot 2^{p-M}$ ($c$ and $M$ are constants) and $k \neq 2^{p-1}$, we have

| | CP | UP |
|---|---|---|
| CI | NPC | NPC |
| UI | NPC | NPC |

(a) Evaluation

| | CP | UP |
|---|---|---|
| CI | NP-Hard | NP-Hard |
| UI | NP-Hard | NP-Hard |

(b) Winner

Figure: $k$-approval [3]

[3] Jérôme Lang, Jérôme Mengin, and Lirong Xia. Aggregating conditionally lexicographic preferences on multi-issue domains, 2012.

# Computational Complexity: $(k, l)$-Approval

Our research considers yet another class of positional scoring rules: $(k, l)$-approval

## $(2^{p-2}, 2^{p-2})$-Approval Evaluation Problem

Let $w$ be the scoring vector for $(2^{p-2}, 2^{p-2})$-approval. The problem to decide for a given $\{UI, CI\} \times \{UP, CP\}$ profile $\mathcal{V}$ and an integer $R$ whether there is an alternative $o$ such that $s_w(\mathcal{V}, o) \geq R$ is NP-complete.

## Proof.

Hardness follows from a polynomial reduction from the NP-complete problem *2-MINSAT*.

# Computational Complexity: $(k, l)$-Approval

| | CP | UP |
|---|---|---|
| CI | NPC | NPC |
| UI | NPC | NPC |

(a) Evaluation

| | CP | UP |
|---|---|---|
| CI | NP-Hard | NP-Hard |
| UI | NP-Hard | NP-Hard |

(b) Winner

Figure: $(2^{p-2}, 2^{p-2})$-approval [4]

[4] The same results obtained for $(2^{p-3}, 2^{p-3})$-approval

# Answer-Set Programming (ASP)

1. Answer-Set Programming [5] [6] is a language for modeling search and optimization problems defined by constraints
2. Rich syntax
   - Constraints: $:\text{-}\ not\ a, b.$
   - Aggregates: $1\{b, c, d\}2.$
   - Definitions: $goal :\text{-}\ a.$ $\quad goal :\text{-}\ b, c.$
3. Formal semantics is given by *answer sets* [7]
4. Fast solvers are available
   - DLV, Cmodels, *clingo*

[5] V.W. Marek and M. Truszczynski. Stable models and an alternative logic programming paradigm, 1999.

[6] Niemelä. Logic Programs with Stable Model Semantics as a Constraint Programming Paradigm, 1999.

[7] Michael Gelfond and Vladimir Lifschitz. The Stable Model Semantics For Logic Programming, 1988

# ASP Dialects

1. Some ASP extensions provide syntax to model *numeric constraints* directly
2. There are solvers that collect numeric constraints and use specialized propagation techniques to solve them
   - *clingcon*: combines, based on Satisfiability Modulo Theories, ASP with non-Boolean Constraint Programming

# Encodings in ASP

1. Encoded LP trees as logic programs
2. Encoded the winner and the evaluation problems in two ASP dialects
   - $\{$*winner*,*evaluation*$\} \times \{$*Borda*, *k-approval*, *(k,l)-approval*$\} \times \{$*clingo*, *clingcon*$\}$
3. The answer sets computed correspond to the solutions to the winner and the evaluation problems

# Random LP Profiles

- To experiment with LP profiles, we developed methods to randomly generate *encodings* of a special type of CI-CP LP tree of size linear in the number of issues

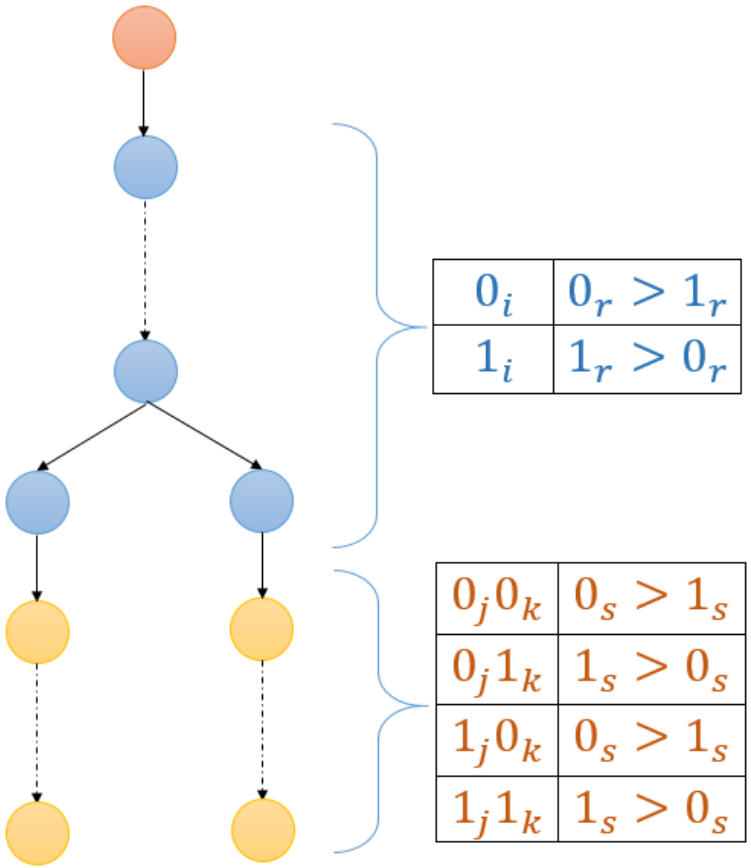


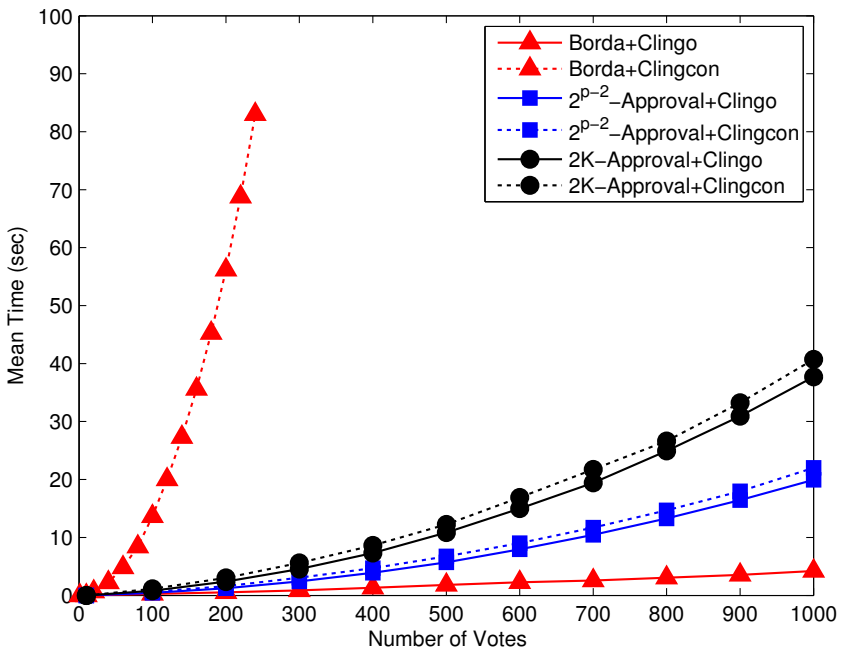

Figure: Random LP tree

# Experimentation

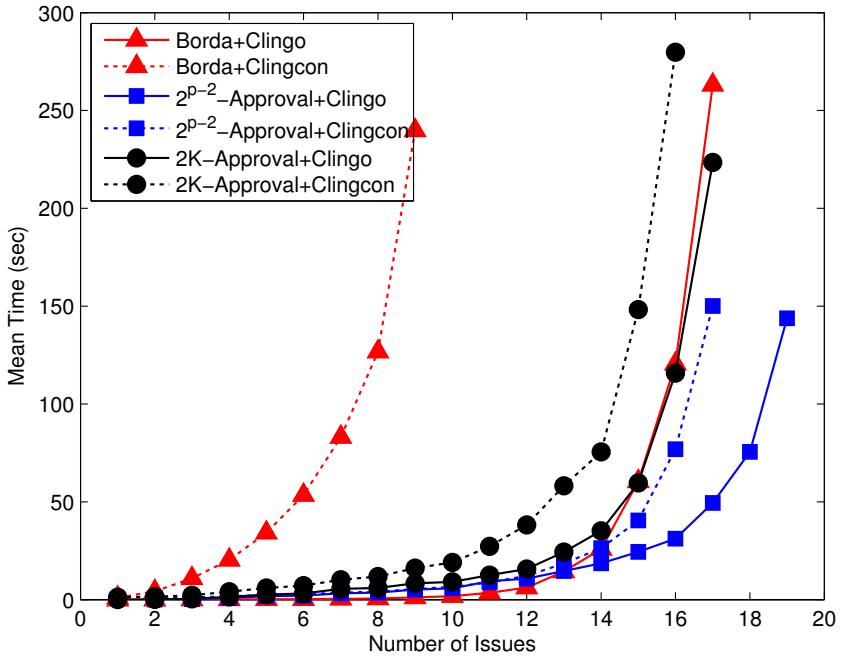
- Focused on NP-hard cases
- The winner problem: $\{$*FI(10)*, *FV(500)*$\}$ $\times$ $\{$*Borda*, $2^{p-2}$-*approval*, *2K-approval* [8] $\}$ $\times$ $\{$*clingo*, *clingcon*$\}$
- The evaluation problem: for a randomly generated profile, computed winning score *WS* and solved the evaluation problem with various *thresholds* (percentages of *WS*): $\{$*5%*$\cdot$*WS*, *10%*$\cdot$*WS*, $\ldots$, *100%*$\cdot$*WS*, $WS+1\}$ $\times$ $\{$*Borda*, $2^{p-2}$-*approval*, *2K-approval*$\}$ $\times$ $\{$*clingo*, *clingcon*$\}$

[8] 2K-approval: $(2,\ldots,2,1,\ldots,1,0,\ldots,0)$ with the numbers of 2's and 1's equal to $2^{p-2}$

# Varying $p$ and $n$



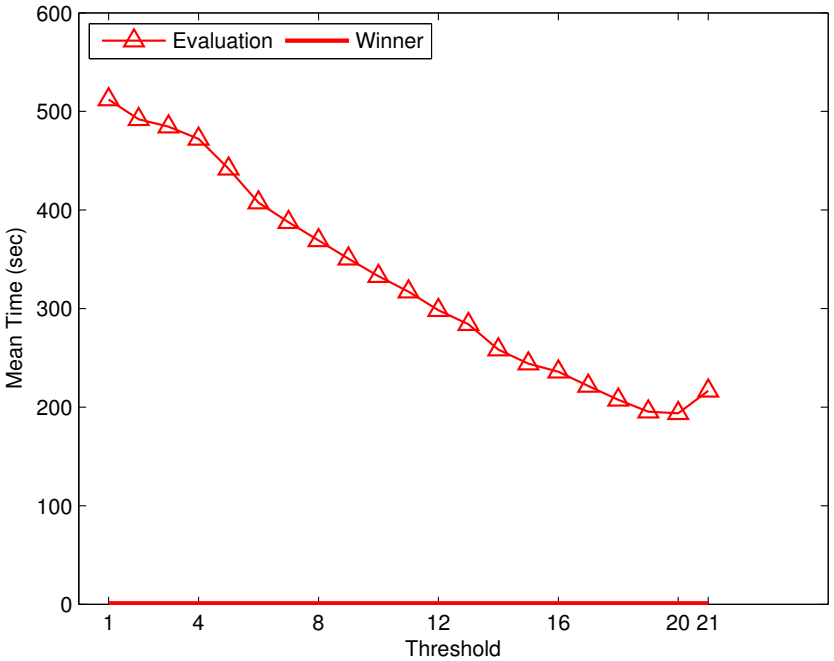

(a) Fixed #issues (10)



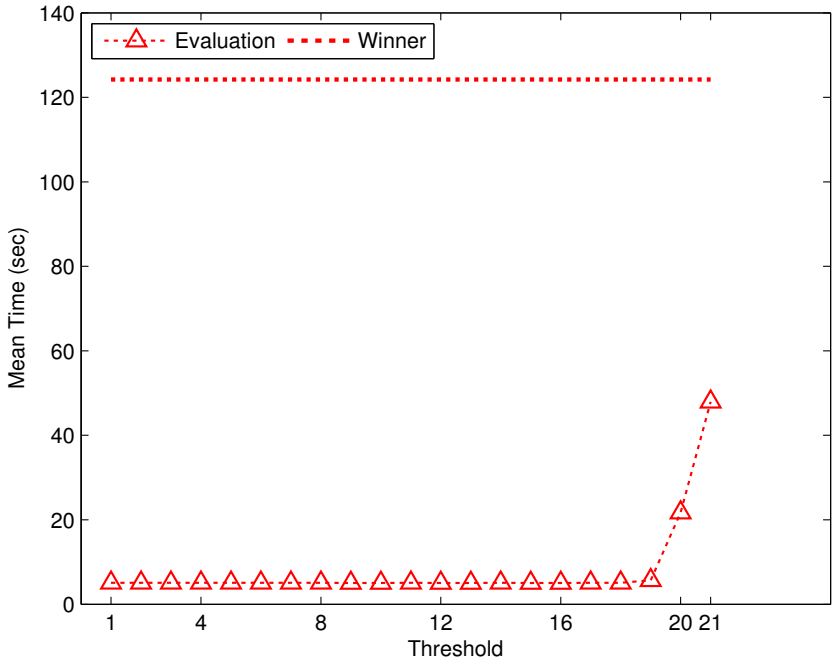

(b) Fixed #votes (500)

Figure: The winner problem

# Evaluation: Borda




(a) *clingo* (500 votes/3 issues)




(b) *clingcon* (500 votes/8 issues)

Figure: Borda

# Evaluation: $2^{p-2}$-Approval

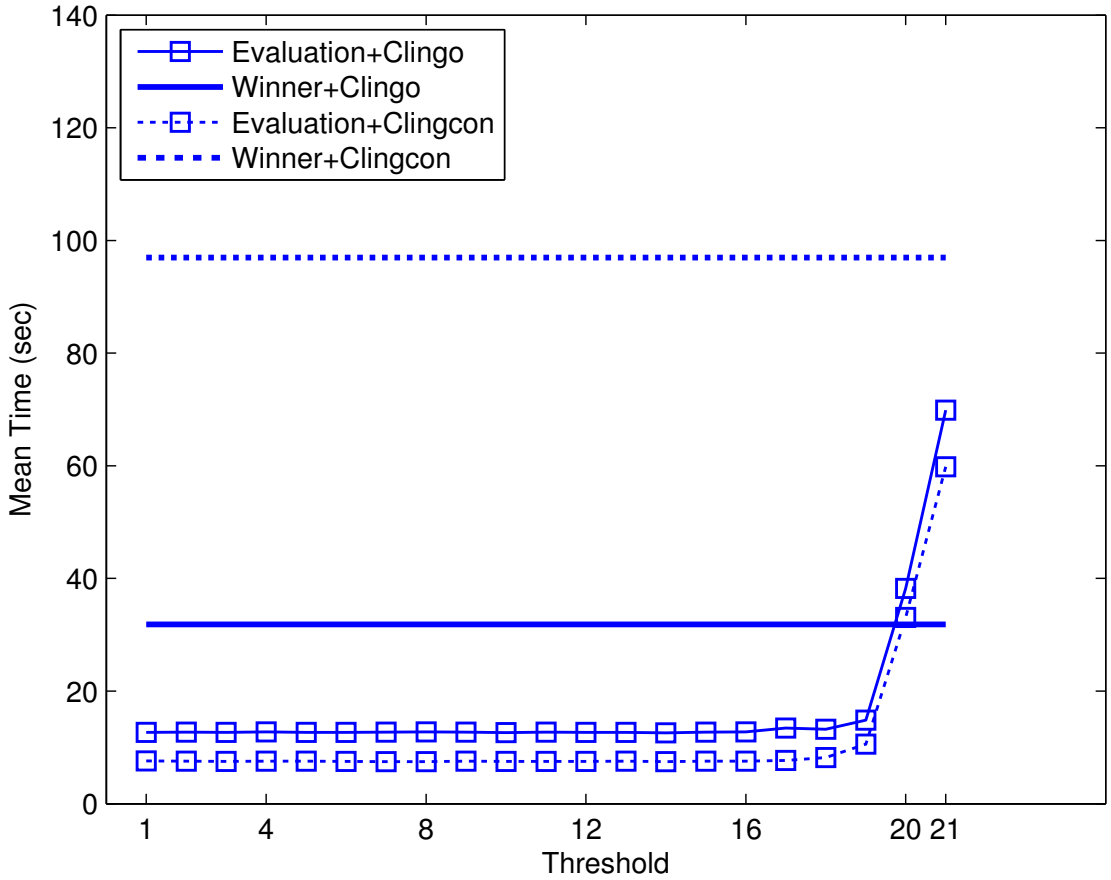



Figure: 500 votes/16 issues

# Evaluation: $2K$-Approval

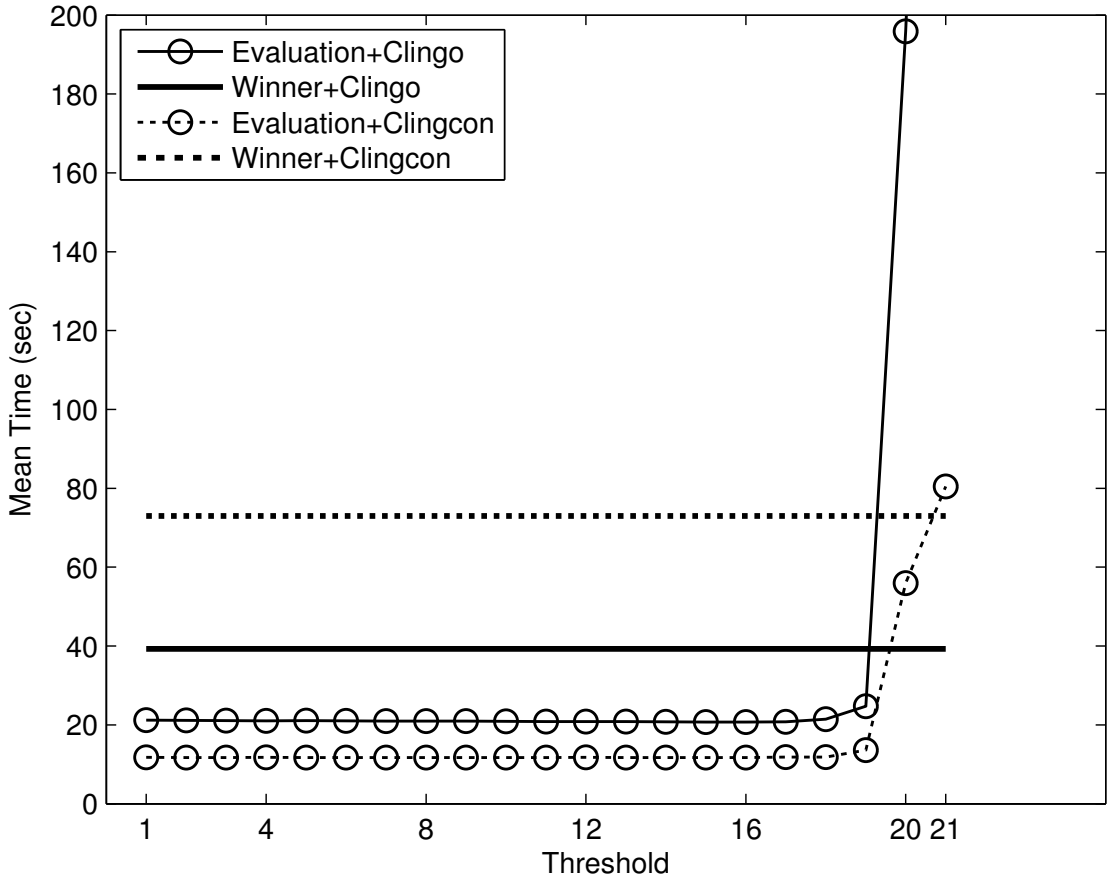



Figure: 500 votes/14 issues

# Future Work

1. Generate richer classes of random LP trees
2. Aggregate preferences in other formalisms, such as conditional preference networks (CP-nets) [9] and answer set optimization preferences [10]

This work was supported by the NSF grant IIS-0913459.

[9] C. Boutilier, R. Brafman, C. Domshlak, H. Hoos, and D. Poole. CP-nets: A tool for representing and reasoning with conditional ceteris paribus preference statements, 2004.

[10] Gerhard Brewka, Ilkka Niemela, and Miroslaw Truszczynski. Answer set optimization, 2003.