Preferences, Planning, and Control

Ronen I. Brafman
Department of Computer Science
Ben-Gurion University

September 18, 2008

Outline

Collaborators

- **Carmel Domshlak**
- Craig Boutilier
- Yuri Chernyavsky
- Holger Hoos
- David Poole
- Yannis Dimopolous

Why Preferences?

Goal concept

- Central to classical planning
- Rigid: all or nothing

Goal concept

- Central to classical planning
- Rigid: all or nothing

Goal concept inadequate:

- It is difficult to formulate a goal when you're not familiar with a domain
  - Planning a vacation in a place you don't know well
  - Information retrieval
- Autonomous systems in uncertain environments can't ask user for revised goals

The Alternative

Preference relations

- Convey more complete information about user objectives
- Can be (repeatedly) consulted when primary goal unachievable

The Alternative

Preference relations

- Convey more complete information about user objectives
- Can be (repeatedly) consulted when primary goal unachievable

A *preference* relation over a set Ω is a transitive binary relation $\succeq$ over Ω. If for every $o, o' \in \Omega$ either $o \succeq o'$ or $o' \succeq o$ then $\succeq$ is a *total* order. Otherwise, it is a *partial* order.

Preferences are simple to specify if:

Single objective with natural order

- Minimize cost
- Maximize quality

Very small set of simple alternatives

- Marriott $\succ$ Best-Western $\succ$ Student Housing $\succ$ A bench across the opera house

Easy to Understand; Hard to Get.

Preference Specification is Difficult!

Preferences are simple to specify if:

Single objective with natural order

- Minimize cost
- Maximize quality

Very small set of simple alternatives

- Marriott $\succ$ Best-Western $\succ$ Student Housing $\succ$ A bench across the opera house

Preferences are difficult to specify if:

Multiple objectives

- Minimize cost *and* maximize quality $\Rightarrow$ Complicated tradeoffs

Large set of alternatives

- Hundreds of MP3 players

Preference Languages

Basic assumption

Outcomes/alternatives are structured – have attributes

Allow users to describe preference order *implicitly*

- Users provide *preference statements*
- Languages that mimic natural language utterances, make it easier to receive information from users
- Statements interpreted as partial order over set of alternatives

Preference Elicitation

Elicitation Techniques

- Limit amount of explicit information provided by user
- Reduce user's cognitive burden (fewer, simpler questions)
- Domain knowledge + previous input $\Rightarrow$ focused questions

Why Preferences?

Summary

- Many applications call for replacing goals with preferences
- Explicit preference relationships are hard to construct and obtain
- Preference languages help users implicitly express a preference order using natural statements
- Preference elicitation technique focus user's effort on most relevant preference information

CP-nets: A Graphical Preference Model

CP-nets
[Boutilier,B,Domshlak, Hoos & Poole 2004]

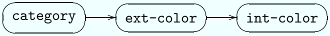
Preference expression

s_1	I prefer minivans to SUVs
s_2	In minivans, I prefer red exterior to white
s_3	In SUVs, I prefer white exterior to red
s_4	In white cars, I prefer dark interior to bright
s_5	In red cars, I prefer bright interior to dark

Outcome space

	category	ext-color	int-color
t_1	minivan	red	bright
t_2	minivan	red	dark
t_3	minivan	white	bright
t_4	minivan	white	dark
t_5	SUV	red	bright
t_6	SUV	red	dark
t_7	SUV	white	bright
t_8	SUV	white	dark

CP-net

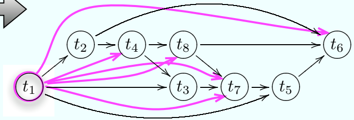


$C_{mv} \succ C_{suv}$

C_{mv}	$E_r \succ E_w$
C_{suv}	$E_w \succ E_r$

E_r	$I_b \succ I_d$
E_w	$I_d \succ I_b$

Preference order



What is the Graphical Representation Good For?
CP-nets

1. Convenient(?) input/elicitation tool
2. Convenient “map of independence”
3. Graph structure related to query processing complexity
4. Some algorithms utilize topological ordering over CP-net

Various queries given a set of preference statements S

- **Verification** Does S convey an ordering?
- **Optimization** Find $o \in \Omega$, such that $\forall o' \in \Omega : o' \not\succ o$.
- **Comparison** Given $o, o' \in \Omega$, does $S \models o \succ o'$?
- **Sorting** Given $\Omega' \subseteq \Omega$, order Ω' consistently with S.

Queries, Complexity, and Graphical Structure

Various queries given a set of preference statements S

Verification Does S convey an ordering?

- "YES" for acyclic CP-nets! [Boutilier *et. al.*2004]
- Tractable for *certain* classes of cyclic CP-nets [B&Domshlak2002]
- PSPACE-hard in general [Goldsmith *et. al.* 2005]

Optimization Find $o \in \Omega$, such that $\forall o' \in \Omega : o' \not\succ o$.

Comparison Given $o, o' \in \Omega$, does $S \models o \succ o'$?

Sorting Given $\Omega' \subseteq \Omega$, order Ω' consistently with S.

Queries, Complexity, and Graphical Structure

Various queries given a set of preference statements S

Verification Does S convey an ordering?

Optimization Find $o \in \Omega$, such that $\forall o' \in \Omega : o' \not\succ o$.

- Linear time for acyclic CP-nets.
- Tractable for *certain* classes of cyclic CP-nets

Comparison Given $o, o' \in \Omega$, does $S \models o \succ o'$?

Sorting Given $\Omega' \subseteq \Omega$, order Ω' consistently with S.

Pairwise Comparison: Given $o, o' \in \Omega$, does $S \models o \succ o'$?

[Boutilier,B,Domshlak,Hoos& Poole 2004][Goldsmith,Lang,Truszczyński&Wilson2005]

Boolean variables

Graph topology	Comparison
Directed Tree	$O(n^2)$
Polytree (indegree $\leq k$)	$O(2^{2k} n^{2k+3})$
Polytree	NP-complete
δ-Connected	NP-complete
DAG	PSPACE-hard
General case	PSPACE-complete

Multi-valued variables

NP-hard...

Queries, Complexity, and Graphical Structure

Various queries given a set of preference statements S

- **Verification** Does S convey an ordering?
- **Optimization** Find $o \in \Omega$, such that $\forall o' \in \Omega : o' \not\succ o$.
- **Comparison** Given $o, o' \in \Omega$, does $S \models o \succ o'$?
- **Sorting** Given $\Omega' \subseteq \Omega$, order Ω' consistently with S.

Some Good News

Sorting is easy!

For *acyclic* CP-nets, sorting is doable in $O(n \log n)$ time

CP-Nets

Summary

- **Language**: Conditional preferences over single attributes
  Summer $\wedge$ Family: Eilat $\succ$ Jerusalem
- **Interpretation**:
  - Statements interpreted using *ceteris paribus* semantics
  - Statements combined via union and transitive closure
- **Representation**: Annotated, directed, graph
  - **Nodes**: Attributes
  - **Edges**: Direct dependency (condition $\rightarrow$ conditioned)
  - **Annotations**: Conditional preference tables (CPTs)
- **Model**: Partial order
- **Complexity**: Related to graph properties

Preferences & Planning: Planning with Goal Preferences

Goal oriented planning $\Rightarrow$ preference-based planning

- Replace goal in domain description with a CP-net
- Find a plan for best feasible goal

Goal oriented planning $\Rightarrow$ preference-based planning

- Replace goal in domain description with a CP-net
- Find a plan for best feasible goal

Can we solve it effectively in practice?

Goal oriented planning $\Rightarrow$ preference-based planning

- Replace goal in domain description with a CP-net
- Find a plan for best feasible goal

Can we solve it effectively in practice?

For an acyclic CP-net — a qualified *yes*

Solving Planning with Preference Problems
[B&Chernyavski2005]

- Planning $\Rightarrow$ CSP [Do&Kambhampati2001]
- Planning with CP-nets $\Rightarrow$ CSP + CP-nets
- CSP + CP-net = (Discrete, qualitative) Constrained optimization
- Constrained optimization = Find best (according to CP-net) feasible (according to CSP) solution
- Constraint solver $\Rightarrow$ Constraint optimizer

Constraint Solver $\longrightarrow$ Optimizer

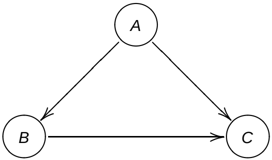
Preferences as constraints on solver [Boutilier,B,Domshlak,Hoos&Poole2004]

Conceptually simple algorithm:

- Use your favorite DPLL/Tree-search-based CSP solver
- CP-net constrains variable/value orderings
- Parents must be assigned before children
- Preferred values must be assigned first
- First solution is optimal!

Solving Constrained Optimization

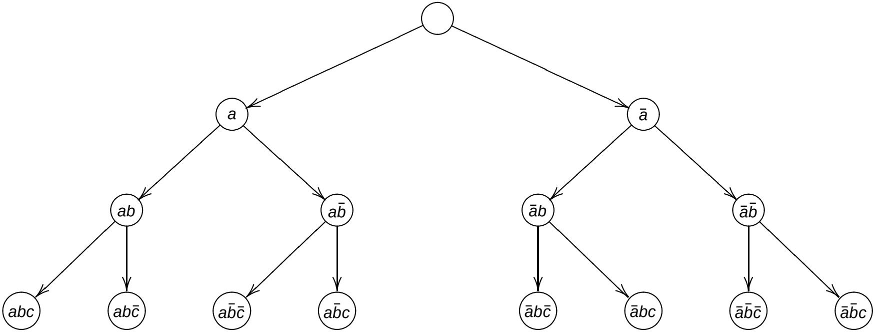
$a \succ \bar{a}$



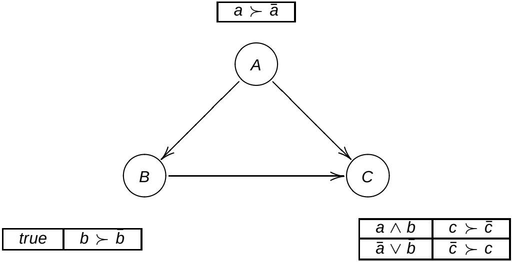
true	$b \succ \bar{b}$

$a \wedge b$	$c \succ \bar{c}$
$\bar{a} \vee \bar{b}$	$\bar{c} \succ c$

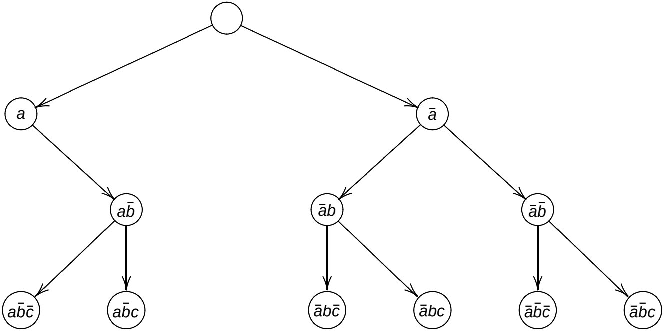
Constraints: $a \longrightarrow \neg b$



Solving Constrained Optimization: Pruning



Constraints: $a \longrightarrow \neg b$

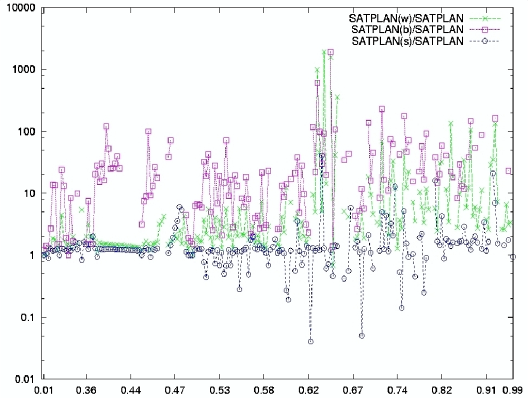


Some Results
[B&Chernyavski05]

# scenario	time[sec.] (avg.)		
	BGP-CSP (time1)	Oracle (time2)	OG&T (time3)
1	0.42	0.01	1.21
2	2.60	0.10	7.65
3	10.37	0.32	215.61
4	2.83	0.20	12.71
5	34.88	0.20	133.65
6	0.13	0.03	1.02
7	0.49	0.08	2.26
8	3.44	0.08	69.61
9	2.92	0.19	3.88
10	2.11	0.26	3.78
11	19.31	0.43	46.56
12	36.85	0.42	72.01
13	15.37	0.72	53.61
14	25.82	1.58	151.51
15	11.37	0.91	59.43

Table: Average plan search times

Preference with Planning as Satisfiability
[Giunchiglia&Maratea06]



Many others

- Oversubscription Planning [Smith2004,van den Briel,Sanchez Nigenda,Do&Kambhampati2004,Workshop,Competition,...]
- Qualitative preferences [Bienvenu,Fritz,&McIlraith2006,...]

Planning with Goal Preferences

Summary

- Planning with preferences can be solved as a constrained optimization problem
- Given a CP-net, we can do constrained optimization using a CSP solver + ordering meta-constraints
- Efficiency depends on # of variables involved
- With few goal variables, method is quite efficient

Preferences & Planning: Graphical Models

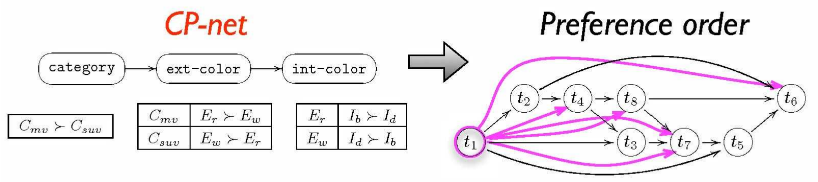
CP-nets query complexity

Complexity is related to simple properties of an intuitive graph

Planning complexity

Can we related planning complexity to simple properties of an intuitive graph?

Planning for Dominance



- Variables = Attributes
- States = Outcomes
- Actions = Rows in CPTs
  - Action transform an outcome to a less preferred outcome
- *Cond*: $X = x_i \succ X = x_j \;\Rightarrow$
  Precondition: *Cond* $\wedge X = x_i$; **Effect**: $X = x_j$
  - a_1: **Pre**: *category = minivan* **Eff**: *category = SUV*
  - a_2: **Pre**: *category = minivan* $\wedge$ *ext = red* **Eff**: *ext = white*
- $o_1 \succ o_2$? $\Rightarrow$ Is there a plan from o_1 to o_2?

Planning for Dominance

- CP-net “actions” have unary effects.
  - Planning with unary effects is as hard as regular planning
- CP-net “actions” have special properties
  - Actions are never reversible

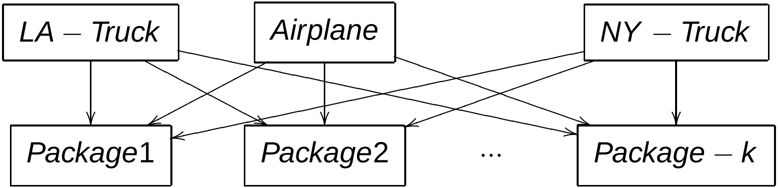
Can we relate properties of a similar graphical structure for planning to plan generation complexity?

- CPT $\Rightarrow$ action
- CP-net $\Rightarrow$?

CP-nets $\Rightarrow$ *Causal Graph*

- Nodes: state variables
- Edges: an edge from x to x' if x is a precondition of an action that affects x'

Causal Graph for Logistics Domain



Boolean variables [B&Domshlak2003]

Graph topology	**Plan Generation**
Directed Tree	$O(n^2)$
Polytree (indegree $\leq k$)	$O(2^{2k} n^{2k+3})$
Polytree	NP-complete
δ-Connected (indegree $\leq k$)	NP-complete
DAG	PSPACE-hard
General case	PSPACE-complete

Multi-valued variables

NP-hard...[Domshlak&Dinitz2001][Chen&Gimenez2008]

What about Standard Planning?

- Causal graph based analysis works!
  - Same definition for general operators
  - Complexity related to graph structure through tree-width

What about Standard Planning?

- Causal graph based analysis works!
  - Same definition for general operators
  - Complexity related to graph structure through tree-width

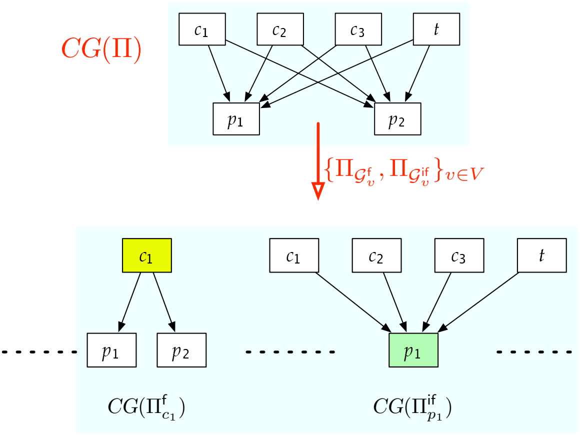
Theorem [B & Domshlak, 2007]
Given bounded-tree width, when short, “balanced” plans exist, they can be computed in polynomial time.

Causal Graph and Complexity

What else can we do with the causal graph?

Tractable Substructures [Katz&Domshlak 2008]

Use tractable substructures as abstractions



Tractable Substructures [Katz&Domshlak 2008]

Use tractable substructures as abstractions

Heuristics and Causal Graphs [Helmert2004,Geffner& Helmert2008]

We can compute heuristic values by propagating information along the causal graph

Causal Graph and Complexity

What else can we do with the causal graph?

Tractable Substructures [Katz&Domshlak 2008]

Use tractable substructures as abstractions

Heuristics and Causal Graphs [Helmert2004,Geffner& Helmert2008]

We can compute heuristic values by propagating information along the causal graph

Distributed Systems [B&Domshlak 2008]

Agent-interaction graph generalizes causal graph in distributed systems

Back to Preferences

Non-Unary Operators

Non-unary operators in planning

$$\Downarrow$$

Preferences over multiple attributes

Example

In a England, I prefer Fish, Chips, and Beer to Veal, Potatoes, and Wine

Main Obstacle: Length

- Dominance queries are like plan existence queries
- In planning, we can focus on short plans
- In preference handling, we can't

Research Challenge

From Planning to Complex Preferences

- Are there tractable classes of planning problems that induce tractable reasoning about complex preferences
  - This is not only about CP-nets!
- Can we reason efficiently with preferences over more than one attribute

Summary

- Causal Graph—

Summary

- Causal Graph—
  - Planning complexity
  - Heuristics
  - CP-nets motivated its initial development

Planning for/with Elicitation

Elicitation as planning under uncertainty [Boutilier2002]

- State: user's true preferences
  - We are uncertain about its value
- Actions: queries to user + "stop" (time to decide)
- Cost & Reward:
  - Query cost depends on cognitive difficulty
  - "Stop": *true* value of *perceived* best outcome
- Planning problem: maximize expected reward
- Challenge: trade of cost of queries with value of better knowledge of true preference model
- Can be formulated precisely as a POMDP

Elicitation as planning under uncertainty [Boutilier2002]

- State: user's true preferences + environment state
  - We are uncertain about user's preference
  - We may be uncertain about true world state
- Actions: queries to user + regular actions
- Cost & Reward:
  - Query cost depends on cognitive difficulty
  - Regular action's value depend on user's preference
- Planning problem: maximize expected reward
- Challenge: trade of cost of queries with value of better knowledge of true preference model
- Can be formulated precisely as a POMDP

Research Challenge

Efficient Planning for Elicitation

POMDPs for elicitation have special structure

- Special state space
- Special actions
- Fixed state in the first case

Can we provide planning algorithms that reason effectively in this domain?

Planning for Elicitation

Summary

- Preference elicitation can be formulated as, and integrated elegantly into, decision theoretic planning models

Preferences & Control:
Relational Preference Models

Relational Preference and Control

Motivation: Command & Control GUI

Imagine controling emergency forces in NYC

- Dynamic situation
  - New events (fire, injured, etc.)
  - Personal/equipment change (forces added/removed)
- Much information to monitor
  - Sensors on personal, equipment, buildings
- Much relevant information to access
  - Maps, building specs, simulations,...

Our Goal:

- Proactively manage decision maker's display
- Same system can work when personal and equipment change
- Same system can be used in New York and Sydney

How do We Solve This Problem?

1. Model this is as a decision-theoretic planning problem
   - Not there yet!
   - Dynamic universe, relational model, huge state space, huge action space, probabilities hard to assess
2. Let user provide an explicit policy
   - Users wouldn't be able to handle this
   - Dynamic universe, relational model, huge state space, huge action space
3. Something in between: preferences over choices+optimization

A Simple Relational Preference Language

The language

1. A set of rules
2. Class definitions (possibly implicit)

rule-body $\rightarrow$ rule-head : $\langle(v_1, w_1), \ldots, (v_k, w_k)\rangle$

- *rule-body:* $\text{class}_1(x_1) \wedge \ldots \wedge \text{class}_k(x_k) \wedge \alpha_1 \wedge \ldots \wedge \alpha_m$
  - α_i: *x_i.path REL value or x_i.path$_i$ REL x_j.path$_j$*.
- *rule-head: x_j.path*
  - *x_j.path* denotes a controllable attribute.

A Simple Relational Preference Language

Examples

$$(1)\ \text{fireman}(x) \wedge \text{fire}(y) \wedge x.\text{location} = y.\text{location} \rightarrow x.\text{camera.display} \ : \ \langle(\text{“on”},4),(\text{“off”},0)\rangle.$$

$$(2)\ \text{fireman}(x) \wedge \text{fire}(y) \wedge x.\text{location} = y.\text{location} \rightarrow x.\text{camera.display} \ : \ \langle(\text{“on”},4)\rangle.$$

$$(3)\ \text{fireman}(x) \wedge x.\text{camera.display} = \textit{“on”} \rightarrow x.\text{rank} \ : \langle(\text{“high”},4)\rangle$$

Preference Rules + DB of objects $\Rightarrow$ value function over the possible assignments to their controlable attributes.

A relational extension of GAI value functions.

$$v_{\mathcal{R},\mathcal{O}}(\bar{a}) = \sum_{\text{ground instances } r' \text{ of } r \in \mathcal{R} \text{ satisfied by } \bar{a}} w(r, v(\bar{a}, r))$$

- $\mathcal{R}$: set of rules
- $\mathcal{O}$: set of objects
- $\bar{a}$: an assignment to their controlable attributes
- $w(r, v(\bar{a}, r))$: weight assigned in r to its head given $\bar{a}$.

Example

$$(1)\ \text{fireman}(x) \wedge \text{fire}(y) \wedge x.\text{location} = y.\text{location}$$
$$\rightarrow x.\text{camera.display} \ :\ \langle(\text{“on”},4),(\text{“off”},0)\rangle.$$

Rule (1) + fireman(Alice), fireman(Bob), fire(Fire1)

- First instantiation: $(1a)$ Alice.location $=$ Fire1.location $\rightarrow$ Alice.camera.display $:$ $\langle$(“on”,4),(“off”,0)$\rangle$
- Second instantiation: $(1b)$ Bob.location $=$ Fire1.location $\rightarrow$ Bob.camera.display $:$ $\langle$(“on”,4),(“off”,0)$\rangle$

Optimization

Methods

- Local search
- Branch & Bound
- Transform into probabilistic relational model:
  - Input (RPR): $b \rightarrow h\langle(v_1, w_1), \ldots, (v_k, w_k)\rangle$.
  - Output (ML): $\{b \wedge h = v_i : w_i\}$.

Preference Rules Vs. Rules

Rules are rigid

- Context sensitivity must be built in explicitly
- Can be inconsistent

Preference rules are flexible

- Context sensitivity built in
- Inference replaced by optimization
- Easier to define approximations

Demo

Conclusions

- To support complex applications, personalization, and autonomy, we need preference handling techniques
- Preference handling and planning have many synergies:
  - Planning with preferences can be done using simple techniques that transform a solver into an optimizer
  - Strong relationship between CP-nets and causal graphs and complexity questions in preference and planning
  - Causal graphs have emerged as an interesting and useful representation of a planning domain
  - Preference elicitation can be formulated as a decision-theoretic planning problem