Eliciting, Modeling, and Reasoning about Preferences using CP-nets

Ronen I. Brafman
Ben-Gurion University
Preferences Are Important…

• Important on their own:
  – Needed even when no uncertainty present
• Hard to get:
  – Preferences at least as hard to elicit as likelihood
• Major bottleneck:
  – Major obstacle to the deployment of decision-support and decision automation software
  – Much harder to learn (arguably, impossible)
Preferences Neglected

• In the past 20 years, research focus was on likelihood elicitation
  – Bayesian nets the primary impetus
  – Much work on other formalisms
  – Important applications – primarily diagnosis

• Until recently, little progress in preference representation and management
Why Were BNs Successful?

- Structure
- Independence
CP-Nets

An attempt to import the essential ideas behind Bayes Nets into preference modeling

• Structure:
  – DiGraph with state variables as nodes
  – Edges denote direct influences

• Independence
  – Preferential independence is used to reduce required information
Preferential Independence

If my preferences over the values of a variable $v$ do not depend on the values of some other variables, then $v$ is preferentially independent of all other variables.

For processor speed, I prefer 1000 MHz to 800 MHz (all else being equal)

A subset of variables $X$ is preferentially independent of its complement $Y = V - X$ if and only if, for all assignments $\bar{x}_1, \bar{x}_2, \bar{y}_1, \bar{y}_2$ holds

$$\bar{x}_1 \bar{y}_1 \geq \bar{x}_2 \bar{y}_1 \text{ iff } \bar{x}_1 \bar{y}_2 \geq \bar{x}_2 \bar{y}_2$$
Conditional Preferential Independence

If my preferences over values of $v$ depend on, and only on, the values of $v_1, \ldots, v_k$, then $v$ is conditionally preferentially independent of $V-\{v_1, \ldots, v_k\}$, given an assignment to $v_1, \ldots, v_k$.

I prefer a 19” screen to a 17” screen if video card is Sony’s

Let $X, Y, Z$ be a partition of $V$ into three disjoint non-empty sets. $X$ is conditionally preferentially independent of $Y$ given $\overline{Z}$ if and only if, for all $\overline{x_1}, \overline{x_2}, \overline{y_1}, \overline{y_2}$ holds

$$\overline{x_1}\overline{y_1}\overline{z} \geq \overline{x_2}\overline{y_1}\overline{z} \quad \text{iff} \quad \overline{x_1}\overline{y_2}\overline{z} \geq \overline{x_2}\overline{y_2}\overline{z}$$
CP-Nets

• As in BNs, we have
  – Qualitative graphical structure
  – Quantification of the relation between parents and child
    • Quantitative quantification $\Rightarrow$ utility functions
    • Qualitative quantification $\Rightarrow$ preference relations

• Most work to-date concentrates on the qualitative version and assumes no uncertainty

• Important potential applications:
  – Configuration problems
  – Database search
Historical Notes

• Large body of work on preference elicitation in past 5 years
  – Graphical representations [Bacchus & Grove], EU-nets [La Murra & Shoham], Value of information [Chajewska et al.], POMDP [Boutilier], Minmax Regret based elicitation [Boutilier et al.], KBANN [Haddawy et al.], LPs [Doyle and McGeachie, Blyth], and more

• Many formalism (soft constraints, preference logics,…)

• CP-nets timeline
  – Boutilier, Brafman, Geib & Poole, AAAI Spring Sym’97
  – Boutilier, Brafman, Hoos & Poole, UAI’99
  – UCP-nets: Boutilier, Bacchus & Brafman, 2001
  – TCP-nets: Brafman & Domshlak, UAI’02

• Other work: Dimopolous; Rossi; Veneble; Walsh; Wilson,…
Outline

• Introduction
• Specifying Preference Relations with CP-nets+
  – Motivation and desiderata
  – Language and semantics
  – Consistency
  – Uniqueness
  – Queries
  – Constrained Optimization
• Specifying Utility Functions with UCP-nets
• Cycles
Huge assortment of models

Customizable accessories

Customizable mechanics
On a red sport car I prefer a sunroof ...
Applications

• **Product Configuration**
  – Find an optimal feasible configuration

• **Searching large databases on the web**
  – Find best available flight

• **Personalization**
  – Display content most appropriate for user
  – Adopt presentation to user device, preferences
Common Properties

• Uncertainty not a serious issue
  ➔ Utility functions are not needed
• Lay users
  ➔ No/little training required
  ➔ As effort-less as possible
• On-line/consumer application
  ➔ Expert decision analyst not available
  ➔ Fast response time desirable
What We Want from a Preference Model

- Supports simple elicitation process based on **intuitive** and **natural** statements about preferences

On a red sports car I prefer a sunroof ...
Some Natural Preference Statements

• I prefer 1000 MHz processor to 800 MHz processor

• I prefer 19in screen to 17in screen if the video card is Sony’s

• CPU speed is more important than CPU manufacturer
The Language

• Value preferences
  – Absolute: I prefer $v_1$ to $v_2$ for variables $X$.
  – Conditional: I prefer $v_1$ to $v_2$ for variables $X$ if $Y=y$ and $Z=z$.

• Relative importance
  – Absolute: $X$ is more important than $Y$
  – Conditional: $X$ is more important than $Y$ if $Z=z$
Interpretation:

**Ceteris Paribus (CP) Semantics**

- *Ceteris Paribus* (Lat.) – All else being equal.
  - The preference holds only when comparing two outcomes that differ in the variables mentioned.
- **Example:** “I prefer wine to beer with my meal”
- Interpretation: Given two *identical* meals, one with wine and one with beer, I prefer the former.
“ I prefer red wine to white wine with my meal, ceteris paribus, given that meat is served”

That is: given two identical meals in which meat is served, I prefer red wine to white wine.

Tells us nothing about two identical meals in which meat is NOT served.
CP Statements and Independence

• *Ceteris Paribus* preference statements induce independence relations:
  – If my preference for wine depends on (and only on) the main course, then wine choice is conditionally preferentially independent of all other variables given the main course value


**CP-nets** *(Boutilier, Brafman, Hoos, Poole, UAI ’99)*

A qualitative, graphical model of preferences, that captures and organizes statements of conditional preferential independence.

- Each node represents a domain variable.
- The immediate parents $Parents(v)$ of a variable $v$ in the network are those variables that affect user’s preference over the values of $v$.

  - $Parents(screen\ size) = \{\text{video card manuf.}\}$
  - $Parents(operating\ system) = \{\text{processor speed, screen size}\}$

  Formally, a child is conditionally preferentially independent of all nodes given its parents’ values.

  Provides an ordering over the values of the node for every possible parent context.
Example of a CP-net

(a \land b) \lor (\overline{a} \land \overline{b}) : c \succ \overline{c}

(a \land \overline{b}) \lor (\overline{a} \land b) : \overline{c} \succ c
CP-nets

• Can be used as a device for helping users describe and structure their preferences
• Can be used as a representation tool for natural language statements
Semantics and Consistency

Any **acyclic** CP-net defines a (consistent) partial order over the outcome space.

\[(a \land b) \lor (\overline{a} \land \overline{b}) : c \succ \overline{c}\]

\[(a \land \overline{b}) \lor (\overline{a} \land b) : \overline{c} \succ c\]
Uniqueness

Two fully specified CP-nets are different IFF they induce different partial orders
Cyclic CP-Nets

• A theory of cyclic CP-nets is emerging
• Computational problems typically harder
• We’ll concentrate on acyclic networks.
• Time permitting, we’ll discuss cycles, too
Example

Dinner Configuration
Suppose that dinner consist of a main course, a soup, and a wine.

Preferences:

• I strictly prefer a steak to a fish fillet.

• I prefer to open with a fish soup if the main course is a steak, and with a vegetable soup if the main course is a fish fillet.

• I prefer a red wine with a vegetable soup, and a white wine with a fish soup.
Main Course

Soup

Wine

$s \succ ff$

$s : fs \succ vs$

$ff : vs \succ fs$

$fs : w \succ r$

$vs : r \succ w$

$ff \land fs \land r$

$ff \land vs \land w$

$s \land vs \land w$

$s \land fs \land r$

$s \land vs \land r$

$s \land vs \land w$
Relative Importance Relations

• Relative importance statements are very natural
• They express the fact that one variable’s value is more important than another’s
• CP-nets induce *implicit* importance relations between nodes and their descendents
Induced Importance Relations in CP-nets

- \( fish \succ vegetable \)
- \( fish: white \succ red \)
- \( vegetable: red \succ white \)
- \( vegetable \land white \)
- parent preferences violated
- \( vegetable \land red \)
- \( fish \land red \)
- \( fish \land white \)
- child preferences violated

July, 7, 2004
UAI’04 Tutorial
Ronen Brafman
Relative Importance

Processor type is more important to me than operating system (all else being equal).

If it is more important to me that the value of $X$ be high than the value of $Y$ be high, then $X$ is more important than $Y$.

$$X > Y$$

Operating system is more important than processor type (all else being equal), if the PC is used primarily for graphical applications.

If, given $z \in \text{Dom}(Z)$, it is more important to me that the value of $X$ be high than the value of $Y$ be high, then $X$ is conditionally more important than $Y$.

$$X >_{z} Y$$
\[ a \succ \overline{a} \]

\[ a : b \succ \overline{b} \quad \overline{a} : \overline{b} \succ b \]

\[ b : c \succ \overline{c} \quad \overline{b} : \overline{c} \succ c \]

\[ S(C, D) = \{B, E\} \]

\[ be : C \succ D \quad \overline{be} : D \succ C \quad b\overline{e} : D \succ C \]

- **nodes \equiv variables**
- **cp-arcs** (directed)
- **i-arcs** (directed)
- **ci-arcs** (undirected)
- **cp-tables**
- **ci-tables**
Example

Choosing a Flight to a Conference in USA
Parameters & Values

- **Day of the flight**
  - *One* or *Two days* before the conference.

- **Airline**
  - *British Airlines* or *KLM*.

- **Departure time**
  - *Morning* or *night*.

- **Stop-overs**
  - *Direct* flight, or a flight with a *stop-over* in Europe.

- **Class**
  - *Economy* or *business*. 
My Preferences
Flight Day - $D$

I have a family and much work, so I prefer to leave a day before the conference.

$1d \succ 2d$
Airline - A

I prefer British Airways to KLM because they have a better frequent-flyer program

$$ba \succ klm$$
Among the flights leaving two days before the conference I prefer to take an evening/night flight, because it will allow me to work longer at the day of the flight.

However, among the flights leaving one day before the conference I prefer to take a morning/noon flight, because I hate to arrive at the last moment.

\[1d : m \succ n\]

\[2d : n \succ m\]
I am a smoker, and I find long non-smoking day flights difficult to cope with. Thus, I prefer a stop-over in Europe.

However, on night flights I usually sleep (and don’t smoke), thus I prefer direct flights which are shorter.

\[ m : 1l \succ 0l \]
\[ n : 0l \succ 1l \]
Class - C

I sleep well in night flights, regardless of the class, and so at night, I prefer economy which is much cheaper.

During the day I prefer to pay for a seat in business class so that I can enjoy the food, wine, and comfortable seats.

\[ m : b \succ e \]

\[ n : e \succ b \]
Day of the flight

Departure Time

Stop-overs

Class

Airline
Day of the flight

Departure Time

Stop-overs

Class

Airline

1d \succ 2d

1d : m \succ n
2d : n \succ m

ba \succ klm

m : 1l \succ 0l
n : 0l \succ 1l

m : b \succ e
n : e \succ b
Relative Importance

Getting a more preferred flying time is more important to me than getting the preferred airline.

\[ T > A \]
1d \succ 2d

1d : m \succ n
2d : n \succ m

m : 1l \succ 0l
n : 0l \succ 1l

ba \succ klm

m : b \succ e
n : e \succ b
Conditional Relative Importance

1. On a *KLM, day flight*, an intermediate stop in Amsterdam is more important to me than sitting in business class.

2. Given a *British Airways, night flight*, having a direct flight is more important to me than getting a cheaper economy sit.

3. On a *British Airways, day flight*, sitting in business class is more important to me than having a stop-over.

\[ S \triangleright_{(m \land klm)} C \quad S \triangleright_{(n \land ba)} C \quad C \triangleright_{(m \land ba)} S \]
Day of the flight

Departure Time

Airline

Stop-overs

Class

1d ≻ 2d

1d : m ≻ n
2d : n ≻ m

ba ≻ klm

m : 1l ≻ 0l
n : 0l ≻ 1l

m∧klm : L ▷ S
n∧ba : L ▷ S
m∧ba : S ▷ L

m : b ≻ e
n : e ≻ b
Importance: Alternative Specification

“$A$ is more important than $B$”

“I prefer better values for $A$ regardless of $B$”.

More generally [Wilson04]:

If $Cond$ then $a_1 > a_2$ regardless of $B_1, \ldots, B_k$
Queries on CP-Nets
Queries

• Comparison of two outcomes \( o, o' \) given a CP-net \( N \):
  – Dominance: does \( o > o' \) hold according to \( N \)?
  – Weak Dominance/order: does \( o \triangleright o' \) hold according to \( N \)?

• Set Ordering:
  – ORD: Given a CP-net \( N \), order a set of outcomes \( O \) consistently with \( N \).

• Optimization:
  – Unconstrained: find the optimal outcome
  – Constrained (BEST): given an (explicit/implicit) set of outcomes \( O \), find one/some/all elements in \( O \) that are not dominated by any other outcome in \( O \).
Dominance queries

Given a CP-net $N$ and a pair of assignments $\alpha$ and $\beta$, determine whether

$$N \models \alpha > \beta$$

If this relation holds, $\alpha$ is preferred to $\beta$, and we say that $\alpha$ dominates $\beta$ with respect to $N$.

A sequence of improving flips from one assignment to another (flipping sequence) is a proof that the latter assignment is preferred to the former.
Dominance testing for CP-nets with binary variables.

<table>
<thead>
<tr>
<th>CP-net graph</th>
<th>Complexity</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tree</td>
<td>Linear?</td>
<td>Backtrack free algorithm in BBHP ’99</td>
</tr>
</tbody>
</table>
Dominance testing for CP-nets with binary variables.

<table>
<thead>
<tr>
<th>CP-net graph</th>
<th>Complexity</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tree</td>
<td>$O(n^2)$</td>
<td>Lower bound</td>
</tr>
</tbody>
</table>
Dominance testing for CP-nets with binary variables.

<table>
<thead>
<tr>
<th>CP-net graph</th>
<th>Complexity</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tree</td>
<td>$O(n^2)$</td>
<td>Lower bound</td>
</tr>
<tr>
<td>Polytree</td>
<td>$O(2^{2k}n^{2k+3})$</td>
<td>$k$ – maximal indegree</td>
</tr>
</tbody>
</table>

The table above summarizes the complexity and remarks for different CP-net graphs. The complexity for a tree structure is $O(n^2)$, and for a polytree, it is $O(2^{2k}n^{2k+3})$. The remarks include the condition that $k$ is the maximal indegree.
Dominance testing for CP-nets with binary variables.

<table>
<thead>
<tr>
<th>CP-net graph</th>
<th>Complexity</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tree</td>
<td>$O(n^2)$</td>
<td>Lower bound</td>
</tr>
<tr>
<td>Polytree</td>
<td>$O(2^{2k} n^{2k+3})$</td>
<td>$k$ - maximal indegree</td>
</tr>
<tr>
<td>Singly-connected DAG</td>
<td>NP-complete</td>
<td>Reduction from 3SAT</td>
</tr>
</tbody>
</table>
Dominance testing for CP-nets with binary variables.

<table>
<thead>
<tr>
<th>CP-net graph</th>
<th>Complexity</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tree</td>
<td>$O(n^2)$</td>
<td>Lower bound</td>
</tr>
<tr>
<td>Polytree</td>
<td>$O(2^{2k} n^{2k+3})$</td>
<td>$k$ - maximal indegree</td>
</tr>
<tr>
<td>Singly-connected DAG</td>
<td>NP-complete</td>
<td>Reduction from 3SAT</td>
</tr>
<tr>
<td>$\delta$-connected DAG</td>
<td>NP-complete</td>
<td>Minimal flipping sequences are polynomially bounded</td>
</tr>
</tbody>
</table>
Dominance testing for CP-nets with binary variables.

<table>
<thead>
<tr>
<th>CP-net graph</th>
<th>Complexity</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tree</td>
<td>$O(n^2)$</td>
<td>Lower bound</td>
</tr>
<tr>
<td>Polytree</td>
<td>$O(2^{2k} n^{2k+3})$</td>
<td>$k$ - maximal indegree</td>
</tr>
<tr>
<td>Singly-connected DAG</td>
<td>NP-complete</td>
<td>Reduction from 3SAT</td>
</tr>
<tr>
<td>$\delta$-connected DAG</td>
<td>NP-complete</td>
<td>Minimal flipping sequences are polynomially bounded</td>
</tr>
<tr>
<td>DAG</td>
<td>?</td>
<td>NP or harder?</td>
</tr>
</tbody>
</table>
Pair Ordering – A Cheaper Alternative

Dominance query:

Given a CP-net \( N \) and a pair of assignments \( \alpha \) and \( \beta \), ask whether \( N \models \alpha > \beta \).

Ordering query:

Given a CP-net \( N \) and a pair of assignments \( \alpha \) and \( \beta \), ask whether \( N \not\models \beta > \alpha \).

If \( N \models \beta > \alpha \), there exists a complete (total) preference ordering consistent with \( N \) in which \( \alpha > \beta \).

In such a case we say that \( \alpha \) is \textit{consistently orderable over} \( \beta \) with respect to \( N \).
Claim 1:

Let $N$ be a CP-net, and $\alpha$, $\beta$ be a pair of complete assignments. If there exist a variable $X$ in $N$, such that:

1. $\alpha$ and $\beta$ assign the same values to all ancestors of $X$ in $N$, and

2. given the assignment provided by $\alpha$ (and $\beta$) to $\text{Parents}(X)$, $\alpha$ assigns a more preferred value to $X$ than that assigned by $\beta$

then $N \not\models \beta > \alpha$. 
Ordering Queries -- II

Condition provided by Claim 1 is:

- Testable in time linear in the number of variables,
- Sufficient BUT not necessary for $N \not\models \beta > \alpha$.

“Partial necessity” – either $N \not\models \beta > \alpha$ or $N \models \alpha > \beta$ (or both) will be determined by the procedure used to answer the ordering queries.
Claim 2:

Given a CP-net $\mathcal{N}$ over $n$ variables and a set of complete assignments $o_1, \ldots, o_m$, ordering these assignments consistently with $\mathcal{N}$ can be done using ordering queries only, in time $O(nm^2)$. 
Example: ESPN Promo

• Presentation with five elements
  – Video featuring upcoming broadcast
  – 2 image ads
  – Video ad
  – Running text with scores/news

• Each media element is a variable

• Additional variables denote user properties, device properties, bandwidth

• Variable values = different content and quality options
CP-Net for ESPN Promo

Gender

Nationality

Sports Video → Ad1 → Ad2 ← Text

Text
Constraints

• Unplayed transmitted material ≤ buffer size
• Transmission rate ≤ bandwidth
• Two ads for same company not allowed
• Ads for alcoholic beverages to underage users not allowed
• Bounds on ratio between height and width
• …
Optimization
Finding the preferentially optimal outcome for an acyclic network is straightforward!

\[ a \succ \bar{a} \]
\[ b \succ \bar{b} \]
\[ (a \land b) \lor (\bar{a} \land \bar{b}) : c \succ \bar{c} \]
\[ (a \land \bar{b}) \lor (\bar{a} \land b) : \bar{c} \succ c \]
\[ c : e \succ \bar{e} \]
\[ \bar{c} : \bar{e} \succ e \]
\[ d : f \succ \bar{f} \]
\[ \bar{d} : \bar{f} \succ f \]
Constrained Optimization

Input:
• Constraints (defining what’s feasible)
• Preferences (defining what’s desirable)

Output:
• One undominated, feasible solution or
• A set of undominated, feasible solutions or
• All undominated, feasible solutions
Solving Constrained Optimization Problems

• Basic idea: Generate & Test
  – Generate outcomes
  – Test for feasibility
  – Test for optimality

• Looks bad – testing for optimality is difficult
Ordered Generate & Test

- Generate outcomes in non-increasing order
- Test for feasibility
- Check for optimality:
  - First feasible outcome is optimal!
  - If more than one is needed:
    - Maintain set of optimal solutions
    - Compare new feasible solutions against current optimal set using dominance testing
Compare

If dominated ... 

Otherwise ...
Generating a Non-increasing Sequence of Outcomes

- Select a topological ordering over variables (consistent with the CP-net structure)
- Build an assignment tree (search tree) by instantiating variables in this order
  - Variable values are ordered based on the CPT
- Leaf nodes, ordered left to right provide a non-increasing sequence of outcomes
\[(a \land b) \lor (\overline{a} \land \overline{b}): c \succ \overline{c}\]

\[(a \land \overline{b}) \lor (\overline{a} \land b): \overline{c} \succ c\]
Improvements

• We are generating a search tree, so we can do all the standard pruning and CSP techniques
  – In fact, we can view the CP-net as imposing a constraint on the variable orderings for the underlying constraint solver

• We can also prune a branch by using bounds (branch and bound) when:
  – We assign a variable to a less preferred value
  – Current set of constraints as strong as for some previous value of this variable

• We can decompose problem is CP-net+Constraints graph contains several connected components
Constraints: $a \Rightarrow b$

Diagram:

- $a$
- $\bar{a}$
- $ab$
- $a \Rightarrow b$
- $\bar{a}b$
- Dominated
- $abc$
- $ab\bar{c}$
- $\bar{a}bc$
- $\bar{abc}$
Anytime Behavior

• *First* feasible solution is optimal!
  – No overhead beyond standard CSP solution
    • Variable ordering is restricted
    • Practical effect has not been investigated yet

• No item withdrawn from set of current solutions
  – Anytime property: set always expanded

• For more than one solution dominance testing needed
  – Can lead to considerable computational overhead
Research Issues

- Language: can we enhance the language while maintaining its intuitiveness?
- Consistency and Complexity for TCP-nets
- Consistency and Constrained Optimization for cyclic nets -- more later.
- Utility functions
- Applications
Quantitative Quantification
Generalized Additive Independence

• $X_1, ..., X_k$ (possibly overlapping) variable sets s.t. $V = \cup X_i$

• $X_1, ..., X_k$ are \textit{generalized additive independent (GAI)} if:
  - For any two distributions $P_1, P_2$, with identical marginals over the $X_i$, expected utility w.r.t. $U$ is the same for $P_1$ and $P_2$

• [Fishburn,BG95]: $X_1, ..., X_k$ are GAI iff $U$ can be written as $U(v) = \sum_i f_i(x_i)$ for suitable utility factors $f_i$
  - Dominance queries easy
  - Optimization queries harder (requires dynamic programming)
UCP-Networks

• Basic structure:
  Graphical structure of a CP-net, but we quantify with conditional utilities

• First-cut semantics: $U(x)$ is sum of utility factors; e.g.,

$$U(abcd) = f_A(a) + f_B(b) + f_C(abc) + f_D(cd) = 5 + 4 + .2 + .9 = 10.1$$

• Thus, utility computation is linear
Elicitation Process for UCP-Nets

1. Identify variables of interest
2. Order variables in (approximate) decreasing order of importance
3. Identify immediate influences on value preference for each variable (= Parents)
4. Quantify influence for a variable as follows:
   • For each assignment to parents and arbitrary fixed assignment to all other variables, assign 0 to the worst element and appropriate values to the other elements
The CPI-Restriction

• Directionality incorporates the stronger *Ceteris Paribus* semantics:

  Preference order over a UCP-net must be consistent with the CP-net it induces

• We refer to this as the *CPI Restriction*

• Can we test it effectively?
Testing CPI Condition

• Not all quantifications satisfy CPI conditions
• A local test exists to verify CPI conditions

- X **dominates its children** if for all $x_1, x_2$ s.t. $f_X(x_1, u) \geq f_X(x_2, u), u, z, y$:

$$f_X(x_1, u) - f_X(x_2, u) \geq \sum_i f_Y(y_i|x_1uiz_i) - f_Y(y_i|x_2uiz_i)$$

- $G$ is a UCP-net **iff** each variable dominates its children
The CPI Restriction

• Directional model w/o CPI still useful for structuring the elicitation process
• UCP-nets satisfying CPI provide a specific form of GAI decomposition
• Optimization-related properties of CP-nets now apply
• Given a GAI-decomposition, the structure of an equivalent UCP+CPI may be different
Preference Compilation

- Similar conditions exist for making a UCP-net consistent with a TCP-net.
- Structure is more complex, though.
- Significance: we can use value/utility functions as a rich semantic model for qualitative statements.
- Qualitative statements = (linear) constraints on the value function parameters
- Structure of value function (i.e., parameters) captures additional independence assumptions.
- In UAI’04, we show what UCP-net structure is sufficient to compile a given TCP-net.
- Suggests following elicitation methodology: start with qualitative statements; compile; refine.
Cyclic Networks
Cyclic Networks

What happens when we have cyclic preferences?

Example:

Consider the following cyclic CP-network (where the assignments do not form a cycle)
Cyclic Networks

Example:
By changing the CPTs, we get a cycle

\[
\begin{align*}
  &b: a > \bar{a} \\
  &\bar{b}: \bar{a} > a \\
  &a: b > \bar{b} \\
  &\bar{a}: \bar{b} > b
\end{align*}
\]

\[
\begin{array}{c}
\text{A} \\
\text{B}
\end{array}
\]

\[
\begin{array}{c}
\text{ab} \\
\bar{a} \\
\bar{b} \\
\text{ab}
\end{array}
\]

\[
\begin{array}{c}
\text{ab} \\
\text{ab} \\
\text{ab} \\
\text{ab}
\end{array}
\]

By changing the CPTs, we get a cycle.
Cyclic Networks

Example:

Consider the following CP-net:

- **A**
  - d: a > a
  - d: a > a

- **B**
  - a: b > b
  - a: b > b

- **C**
  - b: c > c
  - b: c > c

- **D**
  - c: d > d
  - c: d > d

Dotted lines indicate the direction of the influence.
Cyclic Networks

Example:
The derived relation
And its cycle
Cyclic Networks

- Cycle of preferences = inconsistency?
- Should we give up on a specification if it induces a cyclic relation between outcomes?
- Or, can we do something useful with such a specification?
Cyclic Networks

Possible solution: interpret preferences as weak (i.e., $\geq$ and not as $>$). Thus, members of a cycle are equally preferred.

Reminder: $o > o' \iff o \geq o' \& o' \nless o$
Cyclic Networks

Clear candidates for pareto-optimal assignments.

abcd  abcd

All other

abcd  abcd
Cyclic Networks

• All CP-nets are consistent now.
• Some are simply not informative, though.
• Yet, any clear preference we had before, remains valid under this, seemingly weaker, semantics.
Cyclic Networks

Definitions:

1. \( o \) is *strongly optimal* if \( \exists \ o' \ s.t. \ o' \geq o \)

2. \( o \) is *weakly optimal* if \( \exists \ o' \ s.t. \ o' > o \)
Cyclic Networks

Consider the following CP-net:

\[
\begin{align*}
& c: \bar{a} > a \\
& \bar{c}: a > \bar{a} \\
& d > \bar{d}
\end{align*}
\]

\[
\begin{align*}
& a: \bar{b} > b \\
& \bar{a}: b > \bar{b} \\
& b: \bar{c} > c \\
& \bar{b}: c > \bar{c}
\end{align*}
\]

From this network we derive two classes of assignments \( d^* \) (\( d^* \)'s \( D \) value is \( d \)) and \( \bar{d}^* \) (\( \bar{d}^* \)'s \( D \) value is \( \bar{d} \)).

Weakly optimal

\[\begin{align*}
& \rightarrow d^* \\
& \downarrow \\
& \bar{d}^*
\end{align*}\]
Computing Strongly Optimal Assignments

We can compute strongly optimal assignment (when they exist) as follows:

1. Select a cutset $C$
2. For every assignment $A$ to the cutset:
   1. Reduce the graph by removing variables in $C$
   2. Compute $O' =$ optimal assignment to reduced graph
   3. Compute $A' =$ best assignment to variables in original graph given that variables not in $C$ are assigned $O'$
   4. If $A' = A$ then $A'O'$ is a strongly optimal assignment
Summary

- Preferences are important!
- We need them to support user decisions
- We need to develop tools for eliciting, representing, and reasoning with them
- CP-nets attempt to utilize ideas that have been successful in probabilistic reasoning -- structure and independence -- to provide such capabilities
- Clearly, there’s much room for development in this area.