Preferences, Queries, Logics

Jan Chomicki
University at Buffalo

DBRank, August 29, 2011
Plan of the talk

1. Preference relations
2. Preference queries
3. Advanced topics
Part I

Preference relations
“And what is your preference in wine—single or double figures?”
**Preference relations**

<table>
<thead>
<tr>
<th>Universe of objects</th>
</tr>
</thead>
<tbody>
<tr>
<td>- constants: uninterpreted, numbers,...</td>
</tr>
<tr>
<td>- individuals (entities)</td>
</tr>
<tr>
<td>- tuples</td>
</tr>
<tr>
<td>- sets</td>
</tr>
</tbody>
</table>

A preference relation is a binary relation between objects, where one object is better than another. A preference relation is an abstract, uniform way of talking about (relative) desirability, worth, cost, timeliness, and their combinations.
Preference relations

Universe of objects
- constants: uninterpreted, numbers,...
- individuals (entities)
- tuples
- sets

Preference relation $>$
- binary relation between objects
- $x > y \equiv x \text{ is\_better\_than } y \equiv x \text{ dominates } y$
- an abstract, uniform way of talking about (relative) desirability, worth, cost, timeliness,..., and their combinations
- preference relations used in preference queries
Buying a car

Salesman: What kind of car do you prefer?
Customer: The newer the better, if it is the same make. And cheap, too.
Salesman: Which is more important for you: the age or the price?
Customer: The age, definitely.
Salesman: Those are the best cars, according to your preferences, that we have in stock.
Customer: Wait...it better be a BMW.
Salesman: What kind of car do you prefer?
Buying a car

Salesman: What kind of car do you prefer?
Customer: The newer the better, if it is the same make. And cheap, too.
Salesman: What kind of car do you prefer?
Customer: The newer the better, if it is the same make. And cheap, too.
Salesman: Which is more important for you: the age or the price?
Buying a car

Salesman: What kind of car do you prefer?
Customer: The newer the better, if it is the same make. And cheap, too.
Salesman: Which is more important for you: the age or the price?
Customer: The age, definitely.
Buying a car

Salesman: What kind of car do you prefer?
Customer: The newer the better, if it is the same make. And cheap, too.
Salesman: Which is more important for you: the age or the price?
Customer: The age, definitely.
Salesman: Those are the best cars, according to your preferences, that we have in stock.
Buying a car

Salesman: What kind of car do you prefer?
Customer: The newer the better, if it is the same make. And cheap, too.
Salesman: Which is more important for you: the age or the price?
Customer: The age, definitely.
Salesman: Those are the best cars, according to your preferences, that we have in stock.
Customer: Wait...it better be a BMW.
Preferences in perspective

Applications of preferences and preference queries
Preferences in perspective

Applications of preferences and preference queries

1. decision making
2. e-commerce
3. digital libraries
4. personalization
Preferences in perspective

Applications of preferences and preference queries

1 decision making
2 e-commerce
3 digital libraries
4 personalization

Preferences are multi-disciplinary

- economic theory: von Neumann, Arrow, Sen
- philosophy: Aristotle, von Wright
- psychology: Slovic
- artificial intelligence: Boutilier, Brafman
- databases: Kießling, Kossmann
Properties of preference relations

irreflexivity:
\( \forall x \in X, x \not\leq x \)

asymmetry:
\( \forall x, y \in X, x \not\leq y \lor y \not\leq x \)

transitivity:
\( \forall x, y, z \in X, x \not\leq y \land y \not\leq z \Rightarrow x \not\leq z \)

negative transitivity:
\( \forall x, y, z \in X, x \leq y \land y \leq z \Rightarrow x \leq z \)

connectivity:
\( \forall x, y \in X, x \not\leq y \lor y \not\leq x \lor x = y \)

Orders

strict partial order (SPO): irreflexive and transitive

weak order (WO): negatively transitive SPO

total order: connected SPO
### Properties of preference relations

<table>
<thead>
<tr>
<th>Property</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>irreflexivity</strong></td>
<td>$\forall x. x \not\succ x$</td>
</tr>
<tr>
<td><strong>asymmetry</strong></td>
<td>$\forall x, y. x \succ y \Rightarrow y \not\succ x$</td>
</tr>
<tr>
<td><strong>transitivity</strong></td>
<td>$\forall x, y, z. (x \succ y \land y \succ z) \Rightarrow x \succ z$</td>
</tr>
<tr>
<td><strong>negative transitivity</strong></td>
<td>$\forall x, y, z. (x \not\succ y \land y \not\succ z) \Rightarrow x \not\succ z$</td>
</tr>
<tr>
<td><strong>connectivity</strong></td>
<td>$\forall x, y. x \succ y \lor y \succ z \lor z \succ x \lor x = y$</td>
</tr>
</tbody>
</table>
Properties of preference relations

<table>
<thead>
<tr>
<th>Properties of &gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>irreflexivity: ( \forall x. x \not\succ x )</td>
</tr>
<tr>
<td>asymmetry: ( \forall x, y. x &gt; y \Rightarrow y \not\succ x )</td>
</tr>
<tr>
<td>transitivity: ( \forall x, y, z. (x &gt; y \land y &gt; z) \Rightarrow x &gt; z )</td>
</tr>
<tr>
<td>negative transitivity: ( \forall x, y, z. (x \not\succ y \land y \not\succ z) \Rightarrow x \not\succ z )</td>
</tr>
<tr>
<td>connectivity: ( \forall x, y. x &gt; y \lor y &gt; x \lor x = y )</td>
</tr>
</tbody>
</table>

Orders

- strict partial order (SPO): irreflexive and transitive
- weak order (WO): negatively transitive SPO
- total order: connected SPO
Weak and total orders

**Weak order**

```
 a
|   |
 c--d--e
|     |
 f    b
```

**Total order**

```
 a
|   |
 d  f
```
Order properties of preference relations

Irreflexivity, asymmetry: uncontroversial.

Transitivity: captures rationality of preference not always guaranteed: voting paradoxes helps with preference querying.

Negative transitivity: scoring functions represent weak orders.

We assume that preference relations are SPOs.
Order properties of preference relations

Irreflexivity, asymmetry: uncontroversial.
Order properties of preference relations

Irreflexivity, asymmetry: uncontroversial.

Transitivity:
- captures rationality of preference
- not always guaranteed: voting paradoxes
- helps with preference querying
Order properties of preference relations

Irreflexivity, asymmetry: uncontroversial.

Transitivity:
- captures *rationality* of preference
- not always guaranteed: voting paradoxes
- helps with *preference querying*

Negative transitivity:
- *scoring functions* represent weak orders
Order properties of preference relations

Irreflexivity, asymmetry: uncontroversial.

Transitivity:
- captures \textit{rationality} of preference
- not always guaranteed: voting paradoxes
- helps with \textit{preference querying}

Negative transitivity:
- \textit{scoring functions} represent weak orders

We assume that preference relations are SPOs.
Not every SPO is a WO
Not every SPO is a WO

**Indifference**

\[ x \sim y \equiv x \not\succ y \land y \not\succ x. \]
Not every SPO is a WO

**Indifference ~**

\[ x \sim y \equiv x \not\succ y \land y \not\succ x. \]

**Canonical example**

\[ mazda \succ kia, mazda \sim^i vw, kia \sim^i vw \]
Not every SPO is a WO

Indifference $\sim$

$$x \sim y \equiv x \succ y \land y \succ x.$$  

Canonical example

$mazda \succ kia$, $mazda \sim^i vw$, $kia \sim^i vw$

Violation of negative transitivity

$mazda \succ vw$, $vw \succ kia$, $mazda \succ kia$
Preference specification

Explicit preference relations
Finite sets of pairs: bmw, mazda, kia, ...

Implicit preference relations
Can be infinite but finitely representable
Defined using logic formulas in some constraint theory:

\[ p_1, y_1, p_1 \preceq q_1 \preceq p_2, y_2, p_2 \]

for relation Car, Make, Year, Price.

Defined using preference constructors (Preference SQL)
Defined using real-valued scoring functions:

\[ F(p_{m1}, y_1, p_1 \preceq q_1 \preceq p_{m2}, y_2, p_2) \]
Preference specification

Explicit preference relations

Finite sets of pairs: bmw > mazda, mazda > kia,...
Preference specification

Explicit preference relations

Finite sets of pairs: bmw > mazda, mazda > kia,...

Implicit preference relations

- can be infinite but finitely representable
- defined using logic formulas in some constraint theory:

\[
p_{m_1, y_1, p_1 q} \succeq p_{m_2, y_2, p_2 q} \quad \text{for relation Car, Make, Year, Price}
\]
Preference specification

Explicit preference relations

Finite sets of pairs: bmw > mazda, mazda > kia,...

Implicit preference relations

- can be infinite but finitely representable
- defined using logic formulas in some constraint theory:

\[(m_1, y_1, p_1) >_1 (m_2, y_2, p_2) \equiv y_1 > y_2 \lor (y_1 = y_2 \land p_1 < p_2)\]

for relation Car(Make, Year, Price).
Preference specification

Explicit preference relations

Finite sets of pairs: bmw > mazda, mazda > kia,...

Implicit preference relations

- can be infinite but finitely representable
- defined using logic formulas in some constraint theory:

\[ (m_1, y_1, p_1) >_1 (m_2, y_2, p_2) \equiv y_1 > y_2 \lor (y_1 = y_2 \land p_1 < p_2) \]

for relation Car(Make, Year, Price).
- defined using preference constructors (Preference SQL)
Preference specification

Explicit preference relations

**Finite** sets of pairs: bmw > mazda, mazda > kia,…

Implicit preference relations

- can be **infinite** but **finitely representable**
- defined using **logic formulas** in some constraint theory:

  $$(m_1, y_1, p_1) >_1 (m_2, y_2, p_2) \equiv y_1 > y_2 \lor (y_1 = y_2 \land p_1 < p_2)$$

  for relation $\text{Car}(\text{Make}, \text{Year}, \text{Price})$.
- defined using **preference constructors** (Preference SQL)
- defined using real-valued **scoring functions**:
### Preference specification

#### Explicit preference relations

**Finite** sets of pairs: `bmw > mazda, mazda > kia,...`

#### Implicit preference relations

- can be **infinite** but **finitely representable**
- defined using **logic formulas** in some constraint theory:

  \[(m_1, y_1, p_1) >_1 (m_2, y_2, p_2) \equiv y_1 > y_2 \lor (y_1 = y_2 \land p_1 < p_2)\]

  for relation `Car(Make, Year, Price)`.

- defined using **preference constructors** (Preference SQL)
- defined using real-valued **scoring functions**: \[F(m, y, p) = \alpha \cdot y + \beta \cdot p\]
Preference specification

Explicit preference relations

Finite sets of pairs: bmw > mazda, mazda > kia,...

Implicit preference relations

- can be infinite but finitely representable
- defined using logic formulas in some constraint theory:
  \[(m_1, y_1, p_1) >_1 (m_2, y_2, p_2) \equiv y_1 > y_2 \lor (y_1 = y_2 \land p_1 < p_2)\]
  for relation Car(Make, Year, Price).
- defined using preference constructors (Preference SQL)
- defined using real-valued scoring functions: \[F(m, y, p) = \alpha \cdot y + \beta \cdot p\]
  \[(m_1, y_1, p_1) >_2 (m_2, y_2, p_2) \equiv F(m_1, y_1, p_1) > F(m_2, y_2, p_2)\]
Logic formulas

Constants

Object (tuple) attributes

Comparison operators: =, <, >, ≤, ≥...

Arithmetic operators: +, −, ×...

Boolean connectives: ∧, ∨, ¬...

Quantifiers: ∀, ∃

Quantifiers usually can be eliminated (quantifier elimination)

No database relations
The language of logic formulas

- constants
- object (tuple) attributes
- comparison operators: $=, 
eq, <, >, \ldots$
- arithmetic operators: $+, \cdot, \ldots$
- Boolean connectives: $\lnot$, $\land$, $\lor$
- quantifiers:
  - $\forall, \exists$
  - usually can be eliminated (quantifier elimination)
- no database relations
Representability

**Definition**
A scoring function \( f \) represents a preference relation \( \succeq \) if for all \( x, y \),
\[ f(p_x q) \succeq f(p_y q) \]

**Necessary condition for representability**
The preference relation \( \succeq \) is a weak order.

**Sufficient condition for representability**
\( \succeq \) is a weak order the domain is countable or some continuity conditions are satisfied (studied in decision theory).
Representability

Definition

A scoring function $f$ represents a preference relation $\succ$ if for all $x, y$

$$x \succ y \equiv f(x) > f(y).$$
### Defineability

A scoring function $f$ represents a preference relation $>$ if for all $x, y$

$$x > y \equiv f(x) > f(y).$$

### Necessary condition for representability

The preference relation $>$ is a weak order.
Representability

Definition
A scoring function $f$ represents a preference relation $\succ$ if for all $x, y$

\[ x \succ y \iff f(x) > f(y). \]

Necessary condition for representability
The preference relation $\succ$ is a weak order.

Sufficient condition for representability
- $\succ$ is a weak order
- the domain is countable or some continuity conditions are satisfied (studied in decision theory)
Preference constructors [Kießling, 2002]

Good values
Prefer $v_P$ over $v_R$.

POS(Make, t mazda,vw u)

Bad values
Prefer $v_R$ over $v_P$.

NEG(Make, t yugo u)

Explicit preference
Preference encoded by a finite directed graph.

EXP(Make, t (bmw,ford),...,(mazda,kia) u)

Value comparison
Prefer larger/smaller values.

HIGHEST(Year)
LOWEST(Price)

Distance
Prefer values closer to $v_0$.

AROUND(Price,12K)
Good values

Prefer $v \in S_1$ over $v \notin S_1$. 
Good values

Prefer $v \in S_1$ over $v \notin S_1$.

POS(Make, \{mazda, vw\})
Preference constructors [Kießling, 2002]

**Good values**

Prefer \( v \in S_1 \) over \( v \notin S_1 \).

**Bad values**

Prefer \( v \notin S_1 \) over \( v \in S_1 \).

\[
\text{POS} (\text{Make}, \{\text{mazda}, \text{vw}\})
\]
## Preference constructors [Kießling, 2002]

<table>
<thead>
<tr>
<th>Good values</th>
<th>Bad values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prefer $v \in S_1$ over $v \notin S_1$.</td>
<td>Prefer $v \notin S_1$ over $v \in S_1$.</td>
</tr>
</tbody>
</table>

**POS(Make, {mazda, vw})**

**NEG(Make, {yugo})**

- **Explicit preference**
  - Preference encoded by a finite directed graph.
  - \( \text{EXP(Make, (bmw, ford), ...,(mazda, kia))} \)

- **Value comparison**
  - Prefer larger/smaller values.
  - \( \text{HIGHEST(Year)} \)
  - \( \text{LOWEST(Price)} \)

- **Distance**
  - Prefer values closer to $v_0$.
  - \( \text{AROUND(Price, 12K)} \)
Preference constructors [Kießling, 2002]

Good values
Preference $v \in S_1$ over $v \notin S_1$.

Bad values
Preference $v \notin S_1$ over $v \in S_1$.

Explicit preference
Preference encoded by a finite directed graph.

POS(Make,{mazda,vw})

NEG(Make,{yugo})
### Preference constructors [Kießling, 2002]

<table>
<thead>
<tr>
<th>Good values</th>
<th>POS(Make, {mazda, vw})</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prefer $v \in S_1$ over $v \notin S_1$.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bad values</th>
<th>NEG(Make, {yugo})</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prefer $v \notin S_1$ over $v \in S_1$.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Explicit preference</th>
<th>EXP(Make, {(bmw, ford), ..., (mazda, kia)})</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preference encoded by a finite directed graph.</td>
<td></td>
</tr>
</tbody>
</table>
Preference constructors [Kießling, 2002]

<table>
<thead>
<tr>
<th>Good values</th>
<th>Prefer $v \in S_1$ over $v \notin S_1$.</th>
</tr>
</thead>
<tbody>
<tr>
<td>POS(Make, {mazda, vw})</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bad values</th>
<th>Prefer $v \notin S_1$ over $v \in S_1$.</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEG(Make, {yugo})</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Explicit preference</th>
<th>Preference encoded by a finite directed graph.</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXP(Make, {(bmw, ford), ..., (mazda, kia)})</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value comparison</th>
<th>Prefer larger/smaller values.</th>
</tr>
</thead>
</table>

Preference constructors [Kießling, 2002]

**Good values**
Prefer $v \in S_1$ over $v \notin S_1$.

**Bad values**
Prefer $v \notin S_1$ over $v \in S_1$.

**Explicit preference**
Preference encoded by a finite directed graph.

**Value comparison**
Prefer larger/smaller values.

- $\text{POS}(\text{Make},\{\text{mazda},\text{vw}\})$
- $\text{NEG}(\text{Make},\{\text{yugo}\})$
- $\text{EXP}(\text{Make},\{(\text{bmw},\text{ford}),\ldots, (\text{mazda},\text{kia})\})$
- $\text{HIGHEST}(\text{Year})$
- $\text{LOWEST}(\text{Price})$
Preference constructors [Kießling, 2002]

- **Good values**
  Prefer $v \in S_1$ over $v \notin S_1$.

- **Bad values**
  Prefer $v \notin S_1$ over $v \in S_1$.

- **Explicit preference**
  Preference encoded by a finite directed graph.
  
  $$\text{EXP}(\text{Make}, \{(\text{bmw}, \text{ford}), ..., (\text{mazda}, \text{kia})\})$$

- **Value comparison**
  Prefer larger/smaller values.
  
  $$\text{HIGHEST}(\text{Year})$$
  $$\text{LOWEST}(\text{Price})$$

- **Distance**
  Prefer values closer to $v_0$.
  
  $$\text{AROUND}(\text{Price}, 12K)$$
Preference constructors [Kießling, 2002]

**Good values**

Prefer \( v \in S_1 \) over \( v \notin S_1 \).

**POS(Make, \{mazda, vw\})**

**Bad values**

Prefer \( v \notin S_1 \) over \( v \in S_1 \).

**NEG(Make, \{yugo\})**

**Explicit preference**

Preference encoded by a finite directed graph.

**EXP(Make, \{(bmw, ford), \ldots, (mazda, kia)\})**

**Value comparison**

Prefer larger/smaller values.

**HIGHEST(Year)**

**LOWEST(Price)**

**Distance**

Prefer values closer to \( v_0 \).

**AROUND(Price, 12K)**
Combining preferences

Preference composition

Combining preferences about objects of the same kind does not increase dimensionality.

Preference accumulation

Defining preferences over objects in terms of preferences over simpler objects increases dimensionality (preferences over Cartesian product).
Combining preferences

Preference composition

- combining preferences about objects of the same kind
- dimensionality is not increased
- representing preference aggregation, revision, ...
Combining preferences

Preference composition

- combining preferences about objects of the same kind
- dimensionality is not increased
- representing preference aggregation, revision, ...

Preference accumulation

- defining preferences over objects in terms of preferences over simpler objects
- dimensionality is increased (preferences over Cartesian product).
Combining preferences: composition
Combining preferences: composition

Boolean composition

\[ x \succ^\cup y \equiv x \succ^1 y \lor x \succ^2 y \]

and similarly for \( \cap \).
Combining preferences: composition

Boolean composition

\[ x >^\cup y \equiv x >_1 y \lor x >_2 y \]

and similarly for \( \cap \).

Prioritized composition

\[ x >^{lex} y \equiv x >_1 y \lor (y >^1 x \land x >_2 y). \]
Combining preferences: composition

**Boolean composition**

\[ x >^\lor y \equiv x >_1 y \lor x >_2 y \]

and similarly for \( \land \).

**Prioritized composition**

\[ x >^{\text{lex}} y \equiv x >_1 y \lor (y \nmid_1 x \land x >_2 y). \]

**Pareto composition**

\[ x >^{\text{Par}} y \equiv (x >_1 y \land y \nmid_2 x) \lor (x >_2 y \land y \nmid_1 x). \]
Preference composition
Preference composition

Preference relation $\succ_1$

- bmw
- ford
- mazda
- kia
Preference composition

Preference relation $>_1$

```
  bmw
 /|
/  |
ford---mazda
 |
 v
 kia
```

Preference relation $>_2$

```
  ford
   |
   v
   kia
    |
     v
    mazda
     |
      v
      bmw
```
Preference composition

Preference relation $>_1$

- bmw
- ford
- mazda
- kia

Preference relation $>_2$

- ford
- mazda
- kia
- bmw

Prioritized composition

- bmw
  - ford
    - mazda
    - kia
Preference composition

Preference relation $\succ_1$

bmw

ford mazda

kia

Preference relation $\succ_2$

ford kia

mazda

bmw

Prioritized composition

bmw

ford

mazda

kia

Pareto composition

ford bmw

kia mazda
Combining preferences: accumulation [Kiessling, 2002]
Combining preferences: accumulation [Kiessling, 2002]

Prioritized accumulation: \( >^{pr} = (>_1 \& >_2) \)

\[(x_1, x_2) >^{pr} (y_1, y_2) \equiv x_1 >_1 y_1 \lor (x_1 = y_1 \land x_2 >_2 y_2).\]
Combining preferences: accumulation [Kiessling, 2002]

Prioritized accumulation: $>^{pr} = (>_1 \& >_2)$

$$(x_1, x_2) >^{pr} (y_1, y_2) \equiv x_1 >_1 y_1 \lor (x_1 = y_1 \land x_2 >_2 y_2).$$

Pareto accumulation: $>^{pa} = (>_1 \otimes >_2)$

$$(x_1, x_2) >^{pa} (y_1, y_2) \equiv (x_1 >_1 y_1 \land x_2 \geq_2 y_2) \lor (x_1 \geq_1 y_1 \land x_2 >_2 y_2).$$
Combining preferences: accumulation [Kiessling, 2002]

Prioritized accumulation: \( >^{pr} = (>_1 \& >_2) \)

\[
(x_1, x_2) >^{pr} (y_1, y_2) \equiv x_1 >_1 y_1 \lor (x_1 = y_1 \land x_2 >_2 y_2).
\]

Pareto accumulation: \( >^{pa} = (>_1 \otimes >_2) \)

\[
(x_1, x_2) >^{pa} (y_1, y_2) \equiv (x_1 >_1 y_1 \land x_2 \geq_2 y_2) \lor (x_1 \geq_1 y_1 \land x_2 >_2 y_2).
\]

Properties

- closure
- associativity
- commutativity of Pareto accumulation
Skylines

Given single-attribute total preference relations $\preceq_A^1, \ldots, \preceq_A^n$ for a relational schema $R_{\mathit{A^1, \ldots, A^n}}$, the skyline preference relation $\preceq_{\mathit{sky}}$ is defined as $\preceq_{\mathit{sky}} \preceq_{\mathit{A}}$. Unfolding the definition $p_x^1, \ldots, x_n^q \preceq_{\mathit{sky}} p_y^1, \ldots, y_n^q \preceq_{\mathit{i}} x_i^\preceq_{\mathit{A}} y_i$.  

Jan Chomicki
Preferences, Queries, Logics
DBRank, August 29, 2011 21 / 46
Given single-attribute total preference relations $>_{A_1}, \ldots, >_{A_n}$ for a relational schema $R(A_1, \ldots, A_n)$, the skyline preference relation $>^{sky}$ is defined as

$$>^{sky} = >_{A_1} \otimes >_{A_2} \otimes \cdots \otimes >_{A_n}.$$ 

Unfolding the definition

$$(x_1, \ldots, x_n) >^{sky} (y_1, \ldots, y_n) \equiv \bigwedge_i x_i \geq_{A_i} y_i \land \bigvee_i x_i >_{A_i} y_i.$$
Skyline in Euclidean space
Skyline in Euclidean space

Two-dimensional Euclidean space

\[(x_1, x_2) >_{\text{sky}} (y_1, y_2) \equiv x_1 \geq y_1 \land x_2 > y_2 \lor x_1 > y_1 \land x_2 \geq y_2\]

Skyline consists of \(>_{\text{sky}}\)-maximal vectors
Skyline in Euclidean space

Two-dimensional Euclidean space

\[(x_1, x_2) \succ_{sky} (y_1, y_2) \equiv x_1 \geq y_1 \land x_2 > y_2 \lor x_1 > y_1 \land x_2 \geq y_2\]

Skyline consists of \(\succ_{sky}\)-maximal vectors
Skyline properties

Invariance

A skyline preference relation is unaffected by scaling or shifting in any dimension.

Maxima

A skyline consists of the maxima of monotonic scoring functions.

Skyline is not a weak order.

\[
p \preceq q \iff p_i \leq q_i \text{ for all } i
\]
Skyline properties

Invariance

A skyline preference relation is unaffected by scaling or shifting in any dimension.
## Skyline properties

### Invariance
A skyline preference relation is unaffected by **scaling** or **shifting** in any dimension.

### Maxima
A skyline consists of the maxima of **monotonic** scoring functions.
Skyline properties

Invariance
A skyline preference relation is unaffected by scaling or shifting in any dimension.

Maxima
A skyline consists of the maxima of monotonic scoring functions.

Skyline is not a weak order

\[(2, 0) \succ_{\text{sky}} (0, 2), (0, 2) \succ_{\text{sky}} (1, 0), (2, 0) \succ_{\text{sky}} (1, 0)\]
Skyline in SQL

Grouping
Designating attributes not used in comparisons (DIFF).

Example
```
SELECT * FROM Car
SKYLINE Price MIN,
Year MAX,
Make DIFF
```
Grouping

Designating attributes not used in comparisons (DIFF).

Example

```sql
SELECT * FROM Car
SKYLINE Price MIN,
    Year MAX,
    Make DIFF
```
Part II

Preference queries
Winnow [Ch., 2002]

new relational algebra operator
ω (other names: Best, BMO)
retrieves the non-dominated (best) elements in a database relation
can be expressed in terms of other operators

Definition

Given a preference relation $\preceq$ and a database relation $r$:

$$\omega_{\preceq} \text{max}_r.$$  

Notation: If a preference relation $\preceq$ is defined using a formula $C$, then we write $\omega_{C} \text{max}_r$, instead of $\omega_{\preceq} \text{max}_r$.

Skyline query $\omega_{\preceq} \text{sky}_r$ computes the set of maximal vectors in $r$ (the skyline set).

Jan Chomicki
Preferences, Queries, Logics
DBRank, August 29, 2011 27 / 46
Winnow [Ch., 2002]

Winnow
- new relational algebra operator $\omega$ (other names: Best, BMO)
- retrieves the non-dominated (best) elements in a database relation
- can be expressed in terms of other operators
Winnow [Ch., 2002]

Winnow

- new relational algebra operator $\omega$ (other names: Best, BMO)
- retrieves the non-dominated (best) elements in a database relation
- can be expressed in terms of other operators

Definition

Given a preference relation $\succ$ and a database relation $r$:

$$\omega_\succ(r) = \{ t \in r \mid \neg \exists t' \in r. t' \succ t \}.$$
Winnow [Ch., 2002]

**Winnow**
- new relational algebra operator $\omega$ (other names: Best, BMO)
- retrieves the non-dominated (best) elements in a database relation
- can be expressed in terms of other operators

**Definition**

Given a preference relation $\succ$ and a database relation $r$:

$$\omega_\succ(r) = \{ t \in r \mid \neg \exists t' \in r. \; t' \succ t \}.$$  

Notation: If a preference relation $\succ_C$ is defined using a formula $C$, then we write $\omega_C(r)$, instead of $\omega_{\succ_C}(r)$. 
Winnow [Ch., 2002]

Winnow

- new relational algebra operator $\omega$ (other names: Best, BMO)
- retrieves the non-dominated (best) elements in a database relation
- can be expressed in terms of other operators

Definition

Given a preference relation $>_{\mathcal{P}}$ and a database relation $r$:

$$\omega_{>_{\mathcal{P}}}(r) = \{t \in r \mid \neg \exists t' \in r. t' >_{\mathcal{P}} t\}.$$ 

Notation: If a preference relation $>_{\mathcal{C}}$ is defined using a formula $\mathcal{C}$, then we write $\omega_{\mathcal{C}}(r)$, instead of $\omega_{>_{\mathcal{C}}}(r)$.

Skyline query

$\omega_{>_{\text{sky}}}(r)$ computes the set of maximal vectors in $r$ (the skyline set).
Example of winnow
Example of winnow

Relation $Car(\text{Make}, \text{Year}, \text{Price})$

Preference relation:

$$(m, y, p) >_1 (m', y', p') \equiv y > y' \lor (y = y' \land p < p').$$
Example of winnow

Relation \textit{Car}(\textit{Make}, \textit{Year}, \textit{Price})

Preference relation:

$$(m, y, p) >_1 (m', y', p') \equiv y > y' \lor (y = y' \land p < p').$$

<table>
<thead>
<tr>
<th>Make</th>
<th>Year</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>mazda</td>
<td>2009</td>
<td>20K</td>
</tr>
<tr>
<td>ford</td>
<td>2009</td>
<td>15K</td>
</tr>
<tr>
<td>ford</td>
<td>2007</td>
<td>12K</td>
</tr>
</tbody>
</table>
Example of winnow

Relation $\text{Car}(\text{Make}, \text{Year}, \text{Price})$

Preference relation:

$$(m, y, p) >_1 (m', y', p') \equiv y > y' \lor (y = y' \land p < p').$$

<table>
<thead>
<tr>
<th>Make</th>
<th>Year</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>mazda</td>
<td>2009</td>
<td>20K</td>
</tr>
<tr>
<td>ford</td>
<td>2009</td>
<td>15K</td>
</tr>
<tr>
<td>ford</td>
<td>2007</td>
<td>12K</td>
</tr>
</tbody>
</table>
Generalizations of winnow

Iterating winnow

\[ \omega \]

\[ r \]

\[ q \]

\[ H \]

\[ \omega \]

\[ 1 \]

\[ p \]

\[ r \]

\[ q \]

\[ \omega \]

\[ p \]

\[ r \]

\[ q \]

\[ 0 \]

\[ \delta \]

\[ i \]

\[ n \]

\[ \omega \]

\[ i \]

\[ p \]

\[ r \]

\[ q \]

\[ t \]

\[ t \]

\[ P \]

\[ r \]

\[ q \]

\[ t \]

\[ P \]

\[ r \]

\[ q \]

\[ t \]

\[ 1 \]

\[ 1 \]

\[ p \]

\[ r \]

\[ q \]

\[ t \]

\[ P \]

\[ r \]

\[ q \]

\[ t \]

\[ 1 \]

\[ t \]

\[ \pm \]

\[ k \]

\[ u \]

\[ \delta \]

\[ k \]

\[ u \]
Generalizations of winnow

Iterating winnow

\[ \omega_0^0(r) = \emptyset \]

\[ \omega_{n+1}(r) = \omega(r - \bigcup_{0 \leq i \leq n} \omega_i(r)) \]
Generalizations of winnow

Iterating winnow

\[ \omega^0_\geq(r) = \emptyset \]
\[ \omega^{n+1}_\geq(r) = \omega_\geq(r - \bigcup_{0 \leq i \leq n} \omega^i_\geq(r)) \]

Ranking

Rank tuples by their minimum distance from a winnow tuple:

\[ \eta_\geq(r) = \{(t, i) \mid t \in \omega^i_C(r)\} \].
Generalizations of winnow

Iterating winnow

\[
\omega^0_>(r) = \emptyset \\
\omega^{n+1}_>(r) = \omega_>(r - \bigcup_{0 \leq i \leq n} \omega^i_>(r))
\]

Ranking

Rank tuples by their minimum distance from a winnow tuple:

\[
\eta_>(r) = \{(t, i) \mid t \in \omega^i_C(r)\}.
\]

k-band

Return the tuples dominated by at most \(k\) tuples:

\[
\omega_>(r) = \{t \in r \mid \#\{t' \in r \mid t' > t\} \leq k\}.
\]
Preference SQL

The language

basic preference constructors

Pareto/prioritized accumulation

new SQL clause

PREFERRING
groupwise preferences

native implementation

Winnow in Preference SQL

```
SELECT * FROM Car
PREFERRING HIGHEST(Year)
CASCADE LOWEST(Price)
```
Preference SQL

The language

- basic preference constructors
- Pareto/prioritized accumulation
- new SQL clause `PREFERRING`
- groupwise preferences
- native implementation

Winnow in Preference SQL

```
SELECT * FROM Car
PREFERRING HIGHEST(Year)
CASCADE LOWEST(Price)
```
## The language

- basic preference constructors
- Pareto/prioritized accumulation
- new SQL clause PREFERENCES
- groupwise preferences
- native implementation

## Winnow in Preference SQL

```sql
SELECT * FROM Car
PREFERENCES HIGHEST(Year)
    CASCADE LOWEST(Price)
```
Algebraic laws [Ch., 2002; Kießling, 2002]
Commutativity of winnow with selection

If the formula

$$\forall t_1, t_2. [\alpha(t_2) \land \gamma(t_1, t_2)] \Rightarrow \alpha(t_1)$$

is valid, then for every $r$

$$\sigma_\alpha(\omega_\gamma(r)) = \omega_\gamma(\sigma_\alpha(r)).$$
Commutativity of winnow with selection

If the formula

$$\forall t_1, t_2. [\alpha(t_2) \land \gamma(t_1, t_2)] \Rightarrow \alpha(t_1)$$

is valid, then for every $r$

$$\sigma_{\alpha}(\omega_{\gamma}(r)) = \omega_{\gamma}(\sigma_{\alpha}(r)).$$

Under the preference relation

$$(m, y, p) >_{C_1} (m', y', p') \iff y > y' \land p \leq p' \lor y \geq y' \land p < p'$$

the selection $\sigma_{Price<20K}$ commutes with $\omega_{C_1}$ but $\sigma_{Price>20K}$ does not.
Using information about integrity constraints to:

- eliminate redundant occurrences of winnow.
- make more efficient computation of winnow possible.

Eliminating redundancy

Given a set of integrity constraints $F$, $\omega_C$ is redundant w.r.t. $F$ iff $F$ implies the formula

$$R_{t_1} \land t_2 \land t_2 \land \neg t_1^C.$$

Jan Chomicki
Semantic query optimization [Ch., 2004]

Using information about integrity constraints to:
- eliminate redundant occurrences of winnow.
- make more efficient computation of winnow possible.

**Eliminating redundancy**

Given a set of integrity constraints $F$, $\omega_C$ is redundant w.r.t. $F$ iff $F$ implies the formula

\[ \forall t_1, t_2. \ R(t_1) \land R(t_2) \Rightarrow t_1 \sim_C t_2. \]
Integrity constraints

Constraint-generating dependencies (CGD) [Baudinet et al., 1995]

CGD entailment

Decidable by reduction to the validity of \( \exists \)-formulas in the constraint theory (assuming the theory is decidable).
Constraint-generating dependencies (CGD) [Baudinet et al., 1995]

\[ \forall t_1, \ldots, t_n. [R(t_1) \land \cdots \land R(t_n) \land \gamma(t_1, \ldots, t_n)] \Rightarrow \gamma'(t_1, \ldots, t_n). \]
Integrity constraints

Constraint-generating dependencies (CGD) [Baudinet et al., 1995]

\[ \forall t_1 \ldots \forall t_n. [R(t_1) \land \cdots \land R(t_n) \land \gamma(t_1, \ldots t_n)] \Rightarrow \gamma'(t_1, \ldots t_n). \]

CGD entailment

\textbf{Decidable} by reduction to the validity of } \forall \text{-formulas in the constraint theory (assuming the theory is decidable).}
Part III

Advanced topics
Preference modification

Given a preference relation $\succ$ and additional preference or indifference information $I$, construct a new preference relation $\succ_1$ whose contents depend on $\succ$ and $I$.

General postulates fulfillment: the new information $I$ should be completely incorporated into $\succ_1$ with minimal change: $\succ_1$ should be as close to $\succ$ as possible. Closure: $\succ_1$ should have the same order-theoretic (SPO, WO) properties as $\succ$. Finiteness or finite representability of $\succ$ should also be preserved in $\succ_1$. 
Preference modification

Goal

Given a preference relation $>$ and additional preference or indifference information $\mathcal{I}$, construct a new preference relation $>^\prime$ whose contents depend on $>$ and $\mathcal{I}$. 
Preference modification

Goal

Given a preference relation $>$ and additional preference or indifference information $\mathcal{I}$, construct a new preference relation $>$' whose contents depend on $>$ and $\mathcal{I}$.

General postulates

- **fulfillment**: the new information $\mathcal{I}$ should be completely incorporated into $>$'
- **minimal change**: $>$' should be as close to $>$ as possible
- **closure**:
  - order-theoretic (SPO, WO) properties of $>$ should be preserved in $>$'
  - finiteness or finite representability of $>$ should also be preserved in $>$'
Preference revision [Ch., 2007]

Setting new information: revising preference relation

Composition operator $\theta$: union, prioritized or Pareto composition

Composition eliminates (some) preference conflicts

Additional assumptions: interval orders

$$TC_p \theta q$$

to guarantee SPO

VW, 2009

VW, 2008

VW, 2007

Kia, 2009

Kia, 2008

Kia, 2007
Preference revision [Ch., 2007]

**Setting**

- new information: *revising* preference relation $>_{0}$
- composition operator $\theta$: union, prioritized or Pareto composition
- composition eliminates (some) preference conflicts
- additional assumptions: interval orders
- $>^{\prime} = TC(>_{0} \theta >)$ to guarantee SPO
Preference revision [Ch., 2007]

Setting

- new information: revising preference relation $\succ_0$
- composition operator $\theta$: union, prioritized or Pareto composition
- composition eliminates (some) preference conflicts
- additional assumptions: interval orders
- $\succ' = TC(\succ_0 \theta \succ)$ to guarantee SPO
Setting

- new information: revising preference relation $\succ_0$
- composition operator $\theta$: union, prioritized or Pareto composition
- composition eliminates (some) preference conflicts
- additional assumptions: interval orders
- $\succ' = TC(\succ_0 \theta \succ)$ to guarantee SPO
Preference revision [Ch., 2007]

Setting

- new information: revising preference relation $\succ_0$
- composition operator $\theta$: union, prioritized or Pareto composition
- composition eliminates (some) preference conflicts
- additional assumptions: interval orders
- $\succ' = TC(\succ_0 \theta \succ)$ to guarantee SPO

Diagram:

- VW, 2009
- VW, 2008
- VW, 2007
- Kia, 2009
- Kia, 2008
- Kia, 2007

Arrows indicate preference relations.
Preference contraction [Mindolin, Ch., 2011]
Preference contraction [Mindolin, Ch., 2011]

Setting

- new information: contractor relation \( CON \)
- \( >' \): maximal subset of \( > \) disjoint with \( CON \)
Preference contraction [Mindolin, Ch., 2011]

Setting

- new information: contractor relation $CON$
- $\triangleright'$: maximal subset of $>$ disjoint with $CON$

Graph:

```
VW, 2009
   / \        /
VW, 2008  VW, 2007
      / \      / \
VW, 2006  VW, 2005
```
Setting

- new information: contractor relation $CON$
- $>$': maximal subset of $>$ disjoint with $CON$
Setting

- new information: contractor relation \( CON \)
- \( >' \): maximal subset of \( > \) disjoint with \( CON \)
Substitutability [Balke et al., 2006]
Substitutability [Balke et al., 2006]

Setting

- new information: set of indifference pairs
- additional preferences are added to achieve object substitutability
Substitutability [Balke et al., 2006]

**Setting**
- new information: set of *indifference* pairs
- additional preferences are added to achieve *object substitutability*

![Diagram]

- VW, 2009
  - VW, 2008
    - VW, 2007
  - Kia, 2008
  - Kia, 2007
- Kia, 2009
  - Kia, 2008
  - Kia, 2007
Substitutability [Balke et al., 2006]

Setting

- new information: set of indifference pairs
- additional preferences are added to achieve object substitutability

Substitutability [Balke et al., 2006]

Setting

- new information: set of indifference pairs
- additional preferences are added to achieve object substitutability
Preferences over finite sets

Set preferences

Induced:

\[ \forall x \in P_X, y \in P_Y. x \preceq y \]

Aggregate:

\[ \sum A_p X q \preceq \sum A_p Y q \]

Set preference queries

find the best subsets of a given set

restrictions on cardinality
Preferences over finite sets

Set preferences

Induced:

\[ X \succ Y \equiv \forall x \in X. \exists y \in Y. x > y \]

Aggregate:

\[ X \succ Y \equiv sum_A(X) > sum_A(Y) \]
Preferences over finite sets

Set preferences

\[ X \gg Y \equiv \forall x \in X. \exists y \in Y. x > y \]

Aggregate:

\[ X \gg Y \equiv \text{sum}_A(X) > \text{sum}_A(Y) \]

Set preference queries

- find the best subsets of a given set
- restrictions on cardinality
Preferences over set profiles [Zhang, Ch., 2011]
### Preferences over set profiles [Zhang, Ch., 2011]

<table>
<thead>
<tr>
<th>Name</th>
<th>Area</th>
<th>Rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>Newton</td>
<td>Physics</td>
<td>9</td>
</tr>
<tr>
<td>Einstein</td>
<td>Physics</td>
<td>10</td>
</tr>
<tr>
<td>Gargamel</td>
<td>Alchemy</td>
<td>0</td>
</tr>
</tbody>
</table>

**Preferences**

- **P₁**: at most one physicist
- **P₂**: higher total rating

**Jan Chomicki**

Preferences, Queries, Logics

DBRank, August 29, 2011
Preferences over set profiles [Zhang, Ch., 2011]

<table>
<thead>
<tr>
<th>Name</th>
<th>Area</th>
<th>Rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>Newton</td>
<td>Physics</td>
<td>9</td>
</tr>
<tr>
<td>Einstein</td>
<td>Physics</td>
<td>10</td>
</tr>
<tr>
<td>Gargamel</td>
<td>Alchemy</td>
<td>0</td>
</tr>
</tbody>
</table>

**Preferences**

- 2-element subsets
- $P_1$: *at most one* physicist
- $P_2$: higher total *rating*
- $P_1$ more important than $P_2$
Set profile \((F_1, F_2)\)

\[
F_1(S) \equiv \text{SELECT COUNT(Name) FROM S WHERE Area='Physics'}
\]

\[
F_2(S) \equiv \text{SELECT SUM(Rating) FROM S}
\]
Set profile \((F_1, F_2)\)

\[
F_1(S) \equiv \text{SELECT COUNT(Name) FROM S WHERE Area='Physics'}
\]

\[
F_2(S) \equiv \text{SELECT SUM(Rating) FROM S}
\]

Set preference relations

\[
 X \gg_1 Y \equiv F_1(X) \leq 1 \land F_1(Y) > 1
\]

\[
 X \gg_2 Y \equiv F_2(X) > F_2(Y)
\]
Set profile \((F_1, F_2)\)

\[ F_1(S) \equiv \text{SELECT COUNT(Name) FROM S WHERE Area='Physics'} \]
\[ F_2(S) \equiv \text{SELECT SUM(Rating) FROM S} \]

Set preference relations

\[ X \gg_1 Y \equiv F_1(X) \leq 1 \land F_1(Y) > 1 \]
\[ X \gg_2 Y \equiv F_2(X) > F_2(Y) \]

Prioritized composition

\[ X \gg Y \equiv X \gg_1 Y \lor (Y \gg_1 X \land X \gg_2 Y) \]
Prospective research topics

Definability of scoring functions representing preference relations of CP-nets and other graphical models of preferences

Extrinsic preference relations
Preference relations that are not fully defined by tuple contents:

\[ x \not\sim y \]

\[ D_{n_1, n_2} \]

Dissatisfied

\[ p_x, n_1 \land \neg p_y, n_2 \land n_1 \land n_2 \]

Incomplete preferences

tuple scores and probabilities [Soliman et al., 2007]

uncertain tuple scores

disjunctive preferences:

\[ a \not\sim b \lor a \not\sim c \]

Jan Chomicki
Preferences, Queries, Logics
DBRank, August 29, 2011
Prospective research topics

Definability

- of scoring functions representing preference relations
- of CP-nets and other graphical models of preferences
Prospective research topics

Definability
- of scoring functions representing preference relations
- of CP-nets and other graphical models of preferences

Extrinsic preference relations
Preference relations that are not fully defined by tuple contents:

\[ x > y \equiv \exists n_1, n_2. \text{Dissatisfied}(x, n_1) \land \text{Dissatisfied}(y, n_2) \land n_1 < n_2. \]
Prospective research topics

**Definability**
- of scoring functions representing preference relations
- of CP-nets and other graphical models of preferences

**Extrinsic preference relations**
Preference relations that are not fully defined by tuple contents:

\[ x > y \equiv \exists n_1, n_2. Dissatisfied(x, n_1) \land Dissatisfied(y, n_2) \land n_1 < n_2. \]

**Incomplete preferences**
- tuple scores and **probabilities** [Soliman et al., 2007]
- **uncertain** tuple scores
- **disjunctive** preferences: \( a > b \lor a > c \)
Prospective applications

- Databases: preference queries as decision components, workflows, event systems, personalization of query results
- Multi-agent systems: conflict resolution, negotiating joint preferences and decisions
- Social media: preference similarity and stability, preference aggregation
Prospective applications

Databases

- preference queries as decision components: workflows, event systems
- personalization of query results
Prospective applications

Databases
- preference queries as decision components: workflows, event systems
- personalization of query results

Multi-agent systems
- conflict resolution
- negotiating joint preferences and decisions
- negotiation preferences
Prospective applications

Databases
- preference queries as decision components: workflows, event systems
- personalization of query results

Multi-agent systems
- conflict resolution
- negotiating joint preferences and decisions
- negotiation preferences

Social media
- preference similarity and stability
- preference aggregation
Preference queries vs. Top-K queries
### Preference queries vs. Top-K queries

<table>
<thead>
<tr>
<th>Preference queries</th>
<th>Top-K queries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary preference relations</td>
<td>Scoring functions</td>
</tr>
<tr>
<td>Clear declarative reading</td>
<td>“Mysterious” formulation</td>
</tr>
<tr>
<td></td>
<td>Nondeterminism</td>
</tr>
<tr>
<td>No relational data model extension</td>
<td>Rank-relations [Li et al., 2005]</td>
</tr>
<tr>
<td>Structured data</td>
<td>Structured and unstructured data</td>
</tr>
<tr>
<td>Manual construction</td>
<td>Automatic construction</td>
</tr>
<tr>
<td>Preference SQL</td>
<td>Mainstream DBMS</td>
</tr>
</tbody>
</table>
Acknowledgments

- Paolo Ciaccia
- Parke Godfrey
- Jarek Gryz
- Werner Kießling
- Denis Mindolin
- Sławek Staworko
- Xi Zhang

Companion paper
