Towards a Decision Query Language

Jan Chomicki
University at Buffalo
http://www.cse.buffalo.edu/~chomicki
Decisions, decisions,...

• decision scope: Possible vacation destinations? For how long? For how much?

• desirability: I prefer a beach to a large city.

• uncertainty: Enough parking space? Too crowded?

• resources: Where to book?

Some requirements are hard, others are soft.
What is required

Languages in which possible choices and decision criteria of agents can be formulated.

Essential features:

- data and queries
- constraints
- preferences
- *uncertainty, risk,*...
Preferences

Ordering the choices in terms of:

- desirability, coolness, ...
- reliability
- cost, convenience
- timeliness...

Two options:

- binary preference relations: what’s better
- numeric utility functions: scores.
Many different preference relations

Between two hawks, which flies the higher pitch;
Between two dogs, which hath the deeper mouth;
Between two blades, which bears the better temper;
Between two horses, which doth bear him best;
Between two girls, which hath the merriest eye.

W. Shakespeare, King Henry VI.
Decision querying

Find the best answers to a query, instead of all the answers.

“Find the lowest price for this book on the Web...

... but also keep in mind my preference for amazon.com.”

What to do with the obtained information is not addressed:

“We report, you decide.”
Preferences as first-order formulas

[Chomicki, EDBT’02].

Relation \( Book(Title, Vendor, Price) \).

Preference:

\[(i, v, p) \succ_{C_1} (i', v', p') \equiv i = i' \land p < p'.\]

Indifference:

\[(i, v, p) \sim_{C_1} (i', v', p') \equiv i \neq i' \lor p = p'.\]

Utility functions?
Relational algebra embedding

[Chomicki, EDBT’02; Kiessling, VLDB’02]:

New \textit{winnow} operator returning the tuples in the given instance that are \textbf{not dominated} by any other tuple in the instance.

<table>
<thead>
<tr>
<th>Book</th>
<th>Title</th>
<th>Vendor</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_1$</td>
<td>The Flanders Panel</td>
<td>amazon.com</td>
<td>$14.75</td>
</tr>
<tr>
<td>$t_2$</td>
<td>The Flanders Panel</td>
<td>fatbrain.com</td>
<td>$13.50</td>
</tr>
<tr>
<td>$t_3$</td>
<td>The Flanders Panel</td>
<td>bn.com</td>
<td>$18.80</td>
</tr>
<tr>
<td>$t_4$</td>
<td>Green Guide: Greece</td>
<td>bn.com</td>
<td>$17.30</td>
</tr>
</tbody>
</table>
Application scenarios

E-commerce:

- B2C: comparison shopping
- B2B: e-procurement (Cosima [Kiessling, CEC’04])
- E-services

Personalization:

- personalized query results [Koutrika et al. ICDE’04]
- personalized interaction

Configuration:

- “soft” constraints
Plan of the talk

1. Preference relations vs. utility functions.
2. Query languages.
3. Applications: skylines, linear optimization.
5. Preference query optimization.
6. Extensions.
7. Related work.
8. Future work.
Definitions

Preference relation: a binary relation \( \succ \) between the tuples of a given relation.

Preference formula: a first-order formula defining a preference relation.

Intrinsic preference formula: the definition uses only built-in predicates.

Typical properties of preference relations: irreflexivity, and transitivity (\( \Rightarrow \) strict partial orders), can be effectively checked for intrinsic preference formulas with \( =, \neq, <, >, \leq, \geq \).
Weak orders

Weak order: a strict partial order with transitive indifference.
Preference constructors [Kiessling, VLDB’02]

Atomic:

- LOWEST, HIGHEST
- POS, NEG, and combinations
- AROUND, BETWEEN, SCORE

Composite:

- unidimensional: intersection, disjoint union
- multidimensional: Pareto and lexicographic composition

Strict partial orders, definable using first-order formulas.
Utility (scoring) functions

An approach grounded in utility theory:

1. construct a real-valued function $u$ such that:

   \[ t_1 \succ t_2 \equiv u(t_1) > u(t_2) \]

2. return the answers that maximize $u$ in the given instance.

Typically, top $K$ answers are requested.
Properties of scoring functions

+ can be implemented using SQL3 user-defined functions
  [Agrawal et al, SIGMOD’00] [Hristidis et al., SIGMOD’01]

+ provide an ordering of all the answers

+ capture preference intensity

+ can be numerically aggregated
  – need to be hand-crafted for every input
  – hard to logically aggregate
  – not expressive enough: only weak order pref. relations.
Non-existence of utility functions

<table>
<thead>
<tr>
<th>t_1</th>
<th>The Flanders Panel</th>
<th>amazon.com</th>
<th>$14.75</th>
</tr>
</thead>
<tbody>
<tr>
<td>t_2</td>
<td>The Flanders Panel</td>
<td>fatbrain.com</td>
<td>$13.50</td>
</tr>
<tr>
<td>t_3</td>
<td>The Flanders Panel</td>
<td>bn.com</td>
<td>$18.80</td>
</tr>
<tr>
<td>t_4</td>
<td>Green Guide: Greece</td>
<td>bn.com</td>
<td>$17.30</td>
</tr>
</tbody>
</table>

The set of constraints

\[ \{ u(t_2) > u(t_1) > u(t_3), u(t_4) = u(t_1), u(t_4) = u(t_2) \} \]

is unsatisfiable.
Winnow

Given a preference relation $\succ$ defined using a preference formula $C$:

$$\omega_C(r) = \{t \in r | \neg \exists t' \in r. t' \succ t\}.$$ 

Example ("preference for amazon.com"):

$$(i, v, p) \succ_2 (i', v', p') \equiv i = i'$$ 

$$\land v = 'amazon.com' \land v' \neq 'amazon.com'$$
<table>
<thead>
<tr>
<th>$t_1$</th>
<th>The Flanders Panel</th>
<th>amazon.com</th>
<th>$14.75</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_2$</td>
<td>The Flanders Panel</td>
<td>fatbrain.com</td>
<td>$13.50</td>
</tr>
<tr>
<td>$t_3$</td>
<td>The Flanders Panel</td>
<td>bn.com</td>
<td>$18.80</td>
</tr>
<tr>
<td>$t_4$</td>
<td>Green Guide: Greece</td>
<td>bn.com</td>
<td>$17.30</td>
</tr>
</tbody>
</table>
Skyline queries

Find all the tuples that are not dominated by any other tuple in every dimension [Börzsönyi et al, ICDE’01] (Pareto set).

Skylines contain maxima of monotone scoring functions.
Skyline in SQL

\[
\text{SELECT ... FROM ... WHERE ...}
\]
\[
\text{GROUP BY ... HAVING ...}
\]
\[
\text{SKYLINE OF A1[MIN|MAX|DIFF],..., An[MIN|MAX|DIFF]}
\]

Skyline:

\[
\text{SKYLINE OF A DIFF, B MAX, C MIN}
\]

maps to the preference formula:

\[
(x, y, z) \succ (x', y', z') \equiv x = x' \land y \geq y' \land z \leq z' \land (y > y' \lor z < z').
\]
Linear optimization queries

Query formulation:

Find the input tuples that maximize $\sum_{i=1}^{n} a_i x_i$.

The preference relation:

$\bar{x} \succ \bar{y} \equiv \sum_{i=1}^{n} a_i x_i > \sum_{i=1}^{n} a_i y_i$.

Convex hulls contain maxima of positive linear scoring functions.
Winnow evaluation

General methods:

- translation to relational algebra/SQL (Preference SQL [Kiessling et al, VLDB’02])
- BNL: Block-Nested-Loops [Börzsönyi et al, ICDE’01]
- $\beta$-tree [Torlone, Ciaccia, SEBD’03]
Special methods:

- **skyline queries:**
  - **SFS:** Sort-Filter-Skyline [Chomicki et al, ICDE’03]
  - nearest-neighbor search [Kossmann et al., VLDB’02], [Papadias et al, SIGMOD’03].

- **linear optimization queries (top K answers):**
  - convex hull [Chang et al., SIGMOD’00]
  - ranked views [Hristidis et al., SIGMOD’01]
  - ...

1. initialize the window \( W \) and the temporary file \( F \) to empty;

2. repeat the following until the input is empty:

3. for every tuple \( t \) in the input:
   - \( t \) is dominated by a tuple in \( W \) ⇒ ignore \( t \),
   - \( t \) dominates some tuples in \( W \) ⇒ eliminate them and insert \( t \) into \( W \),
   - \( t \) is incomparable with all tuples in \( W \) ⇒ insert \( t \) into \( W \) (if there is room), otherwise add \( t \) to \( F \);

4. output the tuples from \( W \) that were added there when \( F \) was empty,

5. make \( F \) the input, clear F.
1. sort the input w.r.t. any monotone scoring function;
2. initialize the window $W$ and the temporary file $F$ to empty;
3. repeat the following until the input is empty:
   4. for every tuple $t$ in the input:
      - $t$ is dominated by a tuple in $W \Rightarrow$ ignore $t$,
      - $t$ is incomparable with all tuples in $W \Rightarrow$ insert $t$ into $W$ (if there is room), otherwise add $t$ to $F$;
4. output the tuples from $W$.
5. make $F$ the input, clear $F$. 
Optimization of preference queries

Algebraic query optimization.

Semantic query optimization.

Cost-based query optimization.
Algebraic laws [Chomicki, TODS’03]

Commutativity with selection:

If the formula

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

is valid, then for every \(r\)

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

Under the preference relation

\[(i, v, p) \succ C_1 (i', v', p') \equiv i = i' \land p < p'\]

the selection \(\sigma_{Price<20}\) commutes with \(\omega_{C_1}\) but \(\sigma_{Price>20}\) does not.
Distributivity over Cartesian product: For every $r_1$ and $r_2$

$$\omega_C(r_1 \times r_2) = \omega_C(r_1) \times r_2.$$ 

Commutativity of winnow: If $C_1(t_1, t_2) \Rightarrow C_2(t_1, t_2)$ and $\succ C_1$ and $\succ C_2$ are strict partial orders, then for all finite instances $r$:

$$\omega_{C_1}(\omega_{C_2}(r)) = \omega_{C_2}(\omega_{C_1}(r)) = \omega_{C_2}(r).$$ 

Also commutativity with projection.
Semantic query optimization

[Chomicki, CDB’04].

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 $F$ entails 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 (CGDs) [Baudinet et al, ICDT’95]:

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

Entailment is decidable for CGDs by reduction to the validity of $\forall$-formulas in the constraint theory.
Cost-based optimization

For skylines [Buchta, 1989; Godfrey, FOIKS’04]:

The expected cardinality of a $d$-dimensional skyline of $n$ tuples is equal to $H_{d-1,n}$, the $d-1$-order harmonic of $n$ (under attribute independence).

Asymptotically: $H_{d,n} \in \Theta((\ln n)^d/d!)$. 

Some values:

$$H_{2,10^6} = 104$$
$$H_{6,10^6} = 14,087$$
Extension: extrinsic preference

Extrinsic preference relation: depends not only on the components of the tuples being compared but also on other factors:

- the presence or absence of other tuples in the database
- computed or aggregate values.

Solution: winnow + SQL.
Preference for a lower total cost of a book (including shipping and handling).

<table>
<thead>
<tr>
<th>Vendor</th>
<th>SH</th>
</tr>
</thead>
<tbody>
<tr>
<td>amazon.com</td>
<td>$6.99</td>
</tr>
<tr>
<td>fatbrain.com</td>
<td>$3.99</td>
</tr>
<tr>
<td>bn.com</td>
<td>$5.99</td>
</tr>
</tbody>
</table>

Apply winnow to the following view:

```
CREATE VIEW TotalCost(Title, Vendor, Cost) AS
SELECT Book.Title, Book.Vendor, Book.Price + SHCosts.SH
FROM Book, SHCosts WHERE Book.Vendor = SHCosts.Vendor
```

Problem: computing Cartesian products.
Extension: preferences between sets

A best set does not necessarily consist of the best individuals:

- bundling [Chang et al, EC’03]
- complementarity
- diversity $\Rightarrow$ College Admissions Problem

Design query language extensions in which:

- sets are first-class citizens: powerset? nondeterminism?
- solutions can be constrained
- set winnow is available.
Other related work

Preference queries [Lacroix, Lavency, VLDB’87]:

Pick the tuples of $R$ satisfying $Q \land P_1 \land P_2$; if none, pick the tuples satisfying $Q \land P_1 \land \neg P_2$; if none, pick the tuples satisfying $Q \land \neg P_1 \land P_2$.

This can be expressed as

$$\omega_{C_2}(\omega_{C_1}(\sigma_Q(R)))$$

where $C_1(t_1, t_2) \equiv P_1(t_1) \land \neg P_1(t_2)$ and $C_2(t_1, t_2) \equiv P_2(t_1) \land \neg P_2(t_2)$. 
Datalog with preferences [Kiessling et al, 1994], [Govindarajan et al, 2000]:

- clausally-defined preference relations
- extension of Datalog, requires a special evaluation method.

Other areas:

- AI: inference of propositional preferences, “soft” constraints
- philosophy: axiomatizations of preference
- economics: modelling economic behavior.
Future work

Preference modelling and management:

- elicitation: how to construct preference formulas?
- aggregation
- modelling risk and uncertainty

Decision components:

- preferences between actions and plans: workflows, ECA systems
- preferences between E-services

Preferences for XML?