

Semantic Optimization of Preference Queries

Jan Chomicki

University at Buffalo

<http://www.cse.buffalo.edu/~chomicki>

Querying with Preferences

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

Why:

- **too many** answers
- only the best answers **will do**.

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

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

Preferences as constraint formulas

[Chomicki, EDBT'02].

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

Preference:

$$(i, v, p) \succ_{C_1} (i', v', p') \equiv i = i' \wedge p < p'.$$

Indifference:

$$(i, v, p) \sim_{C_1} (i', v', p') \equiv i \neq i' \vee p = p'.$$

Relational algebra embedding

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

New **winnow** operator returning the tuples in the given instance that are **not dominated** by any other tuple in the instance.

Book	Title	Vendor	Price
t_1	The Flanders Panel	amazon.com	\$14.75
t_2	The Flanders Panel	fatbrain.com	\$13.50
t_3	The Flanders Panel	bn.com	\$18.80
t_4	Green Guide: Greece	bn.com	\$17.30

Semantic query optimization

Integrity constraints are **logical formulas**:

$$\forall x, y, y'. P(x, y) \wedge P(x, y') \Rightarrow y = y'.$$

Traditional SQO techniques: Predicate Elimination, Join Introduction/Elimination, ...

Here: **new SQO techniques** for optimizing window.

Plan of the talk

1. Preference relations and winnow.
2. Simplifying the evaluation of preference queries.
3. Dependency entailment.
4. Applications:
 - Preference SQL.
 - Skyline queries.
5. 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 $=, \neq, <, >, \leq, \geq$.

Winnow

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

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

Evaluation:

- special-purpose algorithms like BNL [Börzsönyi et al, ICDE'01], or
- relational algebra:

$$Book(I, V, P) - \pi_{I, V, P}(Book(I, V, P) \bowtie_{I=I', P > P'} Book(I', V', P')).$$

Algebraic properties of winnow

[Chomicki, TODS'03].

Commutativity with selection:

If the formula

$$\forall t_1, t_2. (\alpha(t_2) \wedge \gamma(t_1, t_2)) \Rightarrow \alpha(t_1)$$

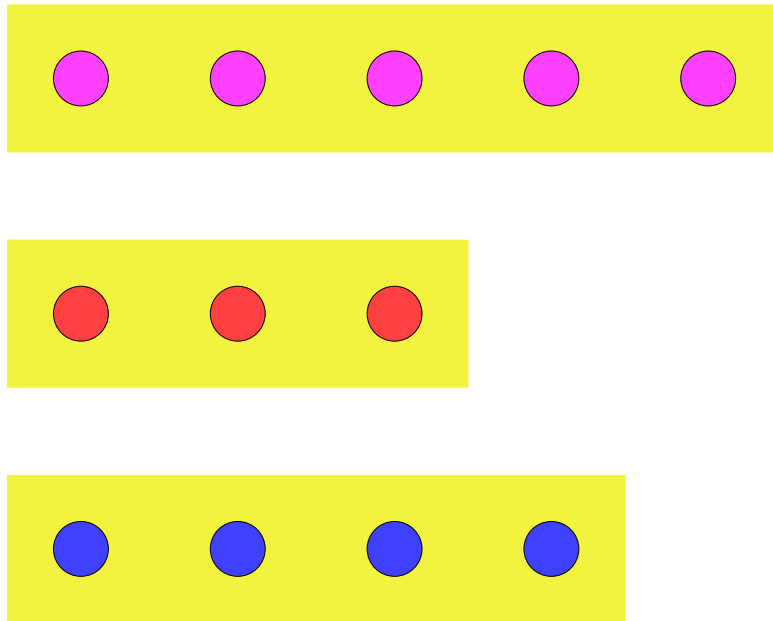
is valid, then for every r

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

Similar properties for other relational algebra operators.

Weak orders

Weak order: a strict partial order with transitive indifference.



“Hard” vs. “easy” orders

Weak orders are **easy**:

- winnow can be computed **in a single pass**.

Empty orders are **easier**:

- winnow can be **eliminated**.

How easy orders arise

Integrity constraints:

- present in the database schema
- preserved by RA operators
- generated by RA operators.

[Simmen et al., SIGMOD'96]:

The operator $\sigma_{Title="ABC"}(Book)$ generates the FD $f_1: \emptyset \rightarrow Title$.

Specific results

Set of integrity constraints F .

1. ω_C is **redundant w.r.t.** F iff F entails the formula

$$\forall t_1, t_2. R(t_1) \wedge R(t_2) \Rightarrow t_1 \sim_C t_2.$$

2. \succ_C is a **weak order relative to** F iff F entails the formula

$$\forall t_1, t_2, t_3. R(t_1) \wedge R(t_2) \wedge R(t_3) \Rightarrow \neg(t_1 \succ_C t_2 \wedge t_1 \sim_C t_3 \wedge t_2 \sim_C t_3).$$

3. $\omega_C(r)$ **satisfies a dependency** f for every instance r iff f is entailed by the formula $\forall t_1, t_2. R(t_1) \wedge R(t_2) \Rightarrow t_1 \sim_C t_2$.

The right class of integrity constraints?

Constraint-generating dependencies (CGDs) [Baudinet, Chomicki, Wolper, ICDT'95, JCSS'99]:

$$\forall t_1 \dots \forall t_n. [R(t_1) \wedge \dots \wedge R(t_n) \wedge \gamma(t_1, \dots, t_n)] \Rightarrow \gamma'(t_1, \dots, t_n).$$

Entailment is **decidable** for CGDs by reduction to the validity of \forall -formulas in the constraint theory.

Example

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

FD $f_1: \emptyset \rightarrow Title$.

Preference relation:

$$(i, v, p) \succ_{C_1} (i', v', p') \equiv i = i' \wedge p < p'.$$

is a **weak order relative to** f_1 because the formula

$$i_1 = i_2 = i_3 \wedge p_1 < p_2 \wedge (i_1 \neq i_3 \vee p_1 = p_3) \wedge (i_2 \neq i_3 \vee p_2 = p_3)$$

is **unsatisfiable**.

Preference SQL

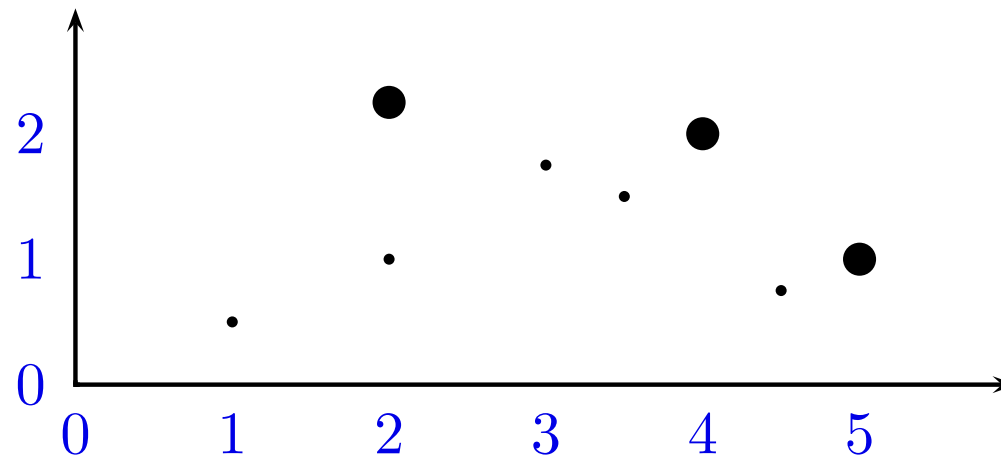
[Kiessling et al., VLDB 2002]:

- atomic and composite preference specifications
- winnow but no logical framework
- implementation: Preference SQL compiled to SQL
- deployed applications: personalized search engines and shopping agents

```
SELECT * FROM Book
PREFERRING MIN(Price)
GROUPING Title
```

Skyline queries

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



Skyline in SQL

```
SELECT ... FROM ... WHERE ...  
  GROUP BY ... HAVING ...  
  SKYLINE OF A1[MIN|MAX|DIFF], ..., An[MIN|MAX|DIFF]
```

Skyline:

```
SKYLINE OF A DIFF, B MAX, C MIN
```

maps to the preference formula:

$$(x, y, z) \succ (x', y', z') \equiv x = x' \wedge y \geq y' \wedge z \leq z' \wedge (y > y' \vee z < z').$$

Linear optimization queries

Query formulation:

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

The preference relation:

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

But only **weak orders** representable using **scoring functions**.

Future work

Semantic query optimization:

- integration with traditional techniques
- richer integrity constraints: decidability of finite entailment essential
- richer queries: ranking.

Preference queries:

- preference elicitation and aggregation
- complex preferences
- XML?