Preference Queries over Sets

Xi Zhang   Jan Chomicki

SUNY at Buffalo

April 15, 2011

Tuple Preferences vs Set Preferences

Tuple Preferences

Well known preferences: top-k, skyline etc.

Set Preferences

Preferences between sets of tuples.

Motivating Example

Alice is buying 3 books as gifts.

Title	Genre	Rating	Price	Vendor
a_1	sci-fi	5.0	$15.00	Amazon
a_2	biography	4.8	$20.00	B&N
a_3	sci-fi	4.5	$25.00	Amazon
a_4	romance	4.4	$10.00	B&N
a_5	sci-fi	4.3	$15.00	Amazon
a_6	romance	4.2	$12.00	B&N
a_7	biography	4.0	$18.00	Amazon
a_8	sci-fi	3.5	$18.00	Amazon
...	...	...	...	...

Motivating Example

Alice is buying 3 books as gifts.

Title	Genre	Rating	Price	Vendor
a_1	sci-fi	5.0	$15.00	Amazon
a_2	biography	4.8	$20.00	B&N
a_3	sci-fi	4.5	$25.00	Amazon
a_4	romance	4.4	$10.00	B&N
a_5	sci-fi	4.3	$15.00	Amazon
a_6	romance	4.2	$12.00	B&N
a_7	biography	4.0	$18.00	Amazon
a_8	sci-fi	3.5	$18.00	Amazon
...	...	...	...	...

She has the following wishes...

- (C1) Spend as little money as possible.
- (C2) Get one sci-fi book.
- (C3) Prioritize (C2) over (C1)

Motivating Example

Alice is buying 3 books as gifts.

Title	Genre	Rating	Price	Vendor
a_1	sci-fi	5.0	$15.00	Amazon
a_2	biography	4.8	$20.00	B&N
a_3	sci-fi	4.5	$25.00	Amazon
a_4	romance	4.4	$10.00	B&N
a_5	sci-fi	4.3	$15.00	Amazon
a_6	romance	4.2	$12.00	B&N
a_7	biography	4.0	$18.00	Amazon
a_8	sci-fi	3.5	$18.00	Amazon
...	...	...	...	...

She has the following wishes...

- (C1) Spend as little money as possible. the cheapest 3 books
- (C2) Get one sci-fi book.
- (C3) Prioritize (C2) over (C1)

Tuple Preference - Winnow (Chomicki [Cho03])

- Tuple Preference: t_1 is *preferred to* t_2

$$t_1 \succ_C t_2 \quad \Leftrightarrow \quad t_1.rating = \text{'sci-fi'} \wedge t_1.price < t_2.price$$

Tuple Preference - Winnow (Chomicki [Cho03])

- Tuple Preference: t_1 is *preferred to* t_2

$$t_1 \succ_C t_2 \quad \Leftrightarrow \quad t_1.rating = \text{'sci-fi'} \wedge t_1.price < t_2.price$$

- Winnow Operator ω_C: return all tuples in a database relation r for which there is no *preferred* tuple in r

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

Tuple Preference - Winnow (Chomicki [Cho03])

- Tuple Preference: t_1 is *preferred to* t_2

$$t_1 \succ_C t_2 \quad \Leftrightarrow \quad t_1.rating = \text{'sci-fi'} \wedge t_1.price < t_2.price$$

- Winnow Operator ω_C: return all tuples in a database relation r for which there is no *preferred* tuple in r

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

- Set Preference - tuple set $\{t_1, t_2, \ldots, t_k\}$ is *preferred to* tuple set $\{t'_1, t'_2, \ldots, t'_k\}$
- Fixed cardinality (k) assumption

Profile-based Set Preference

k-**subsets**: subsets of relation r, with *fixed* cardinality k

Set Pref.	Quantities of Interest	Desired Value or Order
(C1)	total cost	$<$
(C2)	# of sci-fi books	1
(C3)	total cost, # of sci-fi books	(C2)$\rhd$(C1)

Profile-based Set Preference

k-**subsets**: subsets of relation r, with *fixed* cardinality k

Set Pref.	Quantities of Interest	Desired Value or Order
(C1)	total cost	$<$
(C2)	# of sci-fi books	1
(C3)	total cost, # of sci-fi books	(C2)$\rhd$(C1)

$\downarrow$

features $\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_m$

Profile-based Set Preference

k-**subsets**: subsets of relation r, with *fixed* cardinality k

Set Pref.	Quantities of Interest	Desired Value or Order
(C1)	total cost	$<$
(C2)	# of sci-fi books	1
(C3)	total cost, # of sci-fi books	(C2)$\rhd$(C1)

↓ ↓

features $\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_m$

preferences over profiles

profile $= \langle \mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_m \rangle$

Features and Set Preferences

Features: mini-SQL Queries

$\mathcal{F}_1 \equiv$ `SELECT sum(price) FROM $S`

$\mathcal{F}_2 \equiv$ `SELECT count(title) FROM $S WHERE genre='sci-fi'`

Features and Set Preferences

Features: mini-SQL Queries

$\mathcal{F}_1 \equiv$ `SELECT sum(price) FROM $S`

$\mathcal{F}_2 \equiv$ `SELECT count(title) FROM $S WHERE genre='sci-fi'`

Set Preferences

$$
\begin{aligned}
s_1 \succ_{C1} s_2 \Leftrightarrow\ & \mathcal{F}_1(s_1) < \mathcal{F}_1(s_2). \\
s_1 \succ_{C2} s_2 \Leftrightarrow\ & \mathcal{F}_2(s_1) = 1 \wedge \mathcal{F}_2(s_2) \neq 1. \\
s_1 \succ_{C3} s_2 \Leftrightarrow\ & (\mathcal{F}_2(s_1) = 1 \wedge \mathcal{F}_2(s_2) \neq 1) \\
& \vee (\mathcal{F}_2(s_1) = 1 \wedge \mathcal{F}_2(s_2) = 1 \wedge \mathcal{F}_1(s_1) < \mathcal{F}_1(s_2)) \\
& \vee (\mathcal{F}_2(s_1) \neq 1 \wedge \mathcal{F}_2(s_2) \neq 1 \wedge \mathcal{F}_1(s_1) < \mathcal{F}_1(s_2)).
\end{aligned}
$$

Additive Features

- Assume additive features in this talk.
- Efficient optimizations exist for additive features.

Additive Features

- Assume additive features in this talk.
- Efficient optimizations exist for additive features.

Additive Feature

A feature $\mathcal{F}$ is additive iff for every subset s of relation r, and every $t \in r - s$

$$\mathcal{F}(s \cup \{t\}) = \mathcal{F}(s) + f(t)$$

$$\mathcal{F}(\{t\}) = f(t)$$

Computing the "Best" Sets

Naive Algorithm

- Generate all k-subsets of relation r and compute their profiles.
- Run the winnow operator over all the profiles and get the "best" profiles.

Computing the "Best" Sets

Naive Algorithm

- Generate all k-subsets of relation r and compute their profiles.
- Run the winnow operator over all the profiles and get the "best" profiles.

Too many candidate k-subsets!

Computing the "Best" Sets

Naive Algorithm

- Generate all k-subsets of relation r and compute their profiles.
- Run the winnow operator over all the profiles and get the "best" profiles.

Too many candidate k-subsets!

Example

$k = 3, |r| = 1000 \quad \Rightarrow \quad \binom{1000}{3} = 166167000$ candidate subsets

Superpreference Optimization

Goal: Generate as few candidate k-subsets as possible

Superpreference Optimization

Goal: Generate as few candidate k-subsets as possible

"Superpreference"

Find a "superpreference" ($\succ^+$) over the relation r, such that

$$t_1 \succ^+ t_2 \Leftrightarrow \quad s' \cup \{t_1\} \ggcurly_C s' \cup \{t_2\}.$$

for every (k-1)-subset s' of r containing neither t_1 nor t_2.

Superpreference Optimization

Goal: Generate as few candidate k-subsets as possible

"Superpreference"

Find a "superpreference" $(\succ^{+})$ over the relation r, such that

$$t_1 \succ^{+} t_2 \Leftrightarrow \quad s' \cup \{t_1\} \ggcurly_C s' \cup \{t_2\}.$$

for every (k-1)-subset s' of r containing neither t_1 nor t_2.

Pruning Condition

Let $cover(t) = \{t' \in r | t' \succ^{+} t\}$, i.e. all tuples preferred to t.

$$\exists t \in s, cover(t) \nsubseteq s \Rightarrow s \text{ is not a best subset.}$$

Superpreference Optimization

Goal: Generate as few candidate k-subsets as possible

"Superpreference"

Find a "superpreference" ($\succ^+$) over the relation r, such that

$$t_1 \succ^+ t_2 \Leftrightarrow \quad s' \cup \{t_1\} \ggcurly_C s' \cup \{t_2\}.$$

for every (k-1)-subset s' of r containing neither t_1 nor t_2.

Pruning Condition

Let $cover(t) = \{t' \in r | t' \succ^+ t\}$, i.e. all tuples preferred to t.

$$\exists t \in s, cover(t) \nsubseteq s \Rightarrow s \text{ is not a best subset.}$$

Systematic Construction

Possible if all the features are additive.

Example - “Superpreference”

Set preference: (C5)$\cap$(C6)

(C5) Alice wants to spend as little money as possible on sci-fi books.

(C6) Alice wants the average rating of books to be as high as possible.

Example - "Superpreference"

Set preference: (C5)$\cap$(C6)

(C5) Alice wants to spend as little money as possible on sci-fi books.

(C6) Alice wants the average rating of books to be as high as possible.

Features:

$\mathcal{F}_5 \equiv$ `SELECT sum(price) FROM $S WHERE genre='sci-fi'`

$\mathcal{F}_6 \equiv$ `SELECT avg(rating) FROM $S`

Example - "Superpreference"

Set preference: (C5)∩(C6)
(C5) Alice wants to spend as little money as possible on sci-fi books.
(C6) Alice wants the average rating of books to be as high as possible.

Features:
$\mathcal{F}_5 \equiv$ `SELECT sum(price) FROM $S WHERE genre='sci-fi'`
$\mathcal{F}_6 \equiv$ `SELECT avg(rating) FROM $S`

Profile preference:

$$s_1 \ggg_C s_2 \equiv \mathcal{F}_5(s_1) < \mathcal{F}_5(s_2) \wedge \mathcal{F}_6(s_1) > \mathcal{F}_6(s_2)$$

Example - "Superpreference"

Set preference: (C5)∩(C6)
(C5) Alice wants to spend as little money as possible on sci-fi books.
(C6) Alice wants the average rating of books to be as high as possible.

Features:
$\mathcal{F}_5 \equiv$ `SELECT sum(price) FROM $S WHERE genre='sci-fi'`
$\mathcal{F}_6 \equiv$ `SELECT avg(rating) FROM $S`

Profile preference:

$$s_1 \ggcurly_C s_2 \equiv \mathcal{F}_5(s_1) < \mathcal{F}_5(s_2) \land \mathcal{F}_6(s_1) > \mathcal{F}_6(s_2)$$

"Superpreference" formula (assuming $price > 0$)

$$t_1 \succ^+ t_2 \equiv t_1.rating > t_2.rating \land t_2.genre = \text{'sci-fi'} \land (t_1.price < t_2.price \lor t_1.genre \neq \text{'sci-fi'}).$$

M-relation

Goal

Avoid redundancy in generating profiles

Book:

Title	Genre	Rating	Price	Vendor
a_1	sci-fi	5.0	$15.00	Amazon
a_2	biography	4.8	$20.00	B&N
a_3	sci-fi	4.5	$25.00	Amazon
a_4	romance	4.4	$10.00	B&N
a_5	sci-fi	4.3	$15.00	Amazon
a_6	romance	4.2	$12.00	B&N
a_7	biography	4.0	$18.00	Amazon
a_8	sci-fi	3.5	$18.00	Amazon
a_9	romance	4.0	$20.00	Amazon
a_{10}	history	4.0	$19.00	Amazon

Redundancy Example

$$\begin{aligned} & profile_\Gamma(\{a_1, a_2, a_7\}) \\ = \; & profile_\Gamma(\{a_1, a_2, a_9\}) \\ = \; & \langle 15.00, 13.8 \rangle \end{aligned}$$

Exchangeable Tuples a_7, a_9

For any 2-subset s of $Book \backslash \{a_7, a_9\}$

$$profile_\Gamma(s \cup \{a_7\}) = profile_\Gamma(s \cup \{a_9\})$$

Profile $\Gamma = \{\mathcal{F}_5, \mathcal{F}_6\}$

$\mathcal{F}_5 \equiv$ `SELECT sum(price) FROM $S WHERE genre='sci-fi'`

$\mathcal{F}_6 \equiv$ `SELECT sum(rating) FROM $S`

M-relation Generation

Book:

Title	Genre	Rating	Price	Vendor
...	...	...	...	...
a_7	biography	4.0	$18.00	Amazon
a_8	sci-fi	3.5	$18.00	Amazon
a_9	romance	4.0	$20.00	Amazon
a_{10}	history	4.0	$19.00	Amazon

M-relation:

	A_5	A_6	A_{cnt}
...	...	...	...
$m_{7,9,10}$	$0.00	4.0	3
m_8	$18.00	3.5	1

Profile $\Gamma = \{\mathcal{F}_5, \mathcal{F}_6\}$

$\mathcal{F}_5 \equiv$ `SELECT sum(price) FROM $S WHERE genre='sci-fi'`

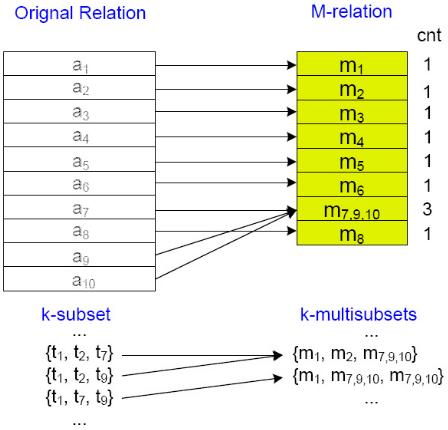
$\mathcal{F}_6 \equiv$ `SELECT sum(rating) FROM $S`

M-relation Generation SQL

```
SELECT CASE WHEN r.genre='sci-fi' THEN r.price ELSE 0 END AS A5,
       r.rating AS A6, count(*) AS Acnt FROM r
GROUP BY A5, A6
```

Set Preference via M-relation

- Set preference over the original relations ⇒ set preference over its M-relation



Conclusions and Future Work

Conclusions

- A formal "logic + SQL" framework for specifying restricted set preferences and implementing set preference queries.
- Effective optimizations yielding improvements of several orders of magnitude.

Conclusions and Future Work

Conclusions

- A formal "logic + SQL" framework for specifying restricted set preferences and implementing set preference queries.
- Effective optimizations yielding improvements of several orders of magnitude.

Future Work

- Query optimization for non-additive features.
- Set preference elicitation.
- Embedding "best-subset" generation in relational query languages.
- Additional set ranking or browsing techniques for result navigation.
- Relaxing the fixed cardinality assumption: Superpreference depends on it assumption while M-relation does not.

Thank you!

Dataset and Set Preferences

- **Dataset**
  - 8000 book quotes from Amazon
  - Schema: $\langle title, genre, rating, price, vendor\rangle$
- **Features**

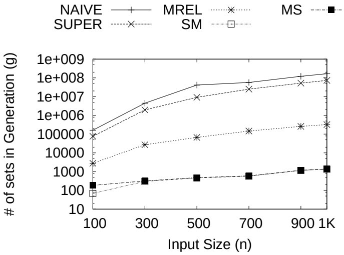
$\mathcal{F}_5 \equiv$ `SELECT sum(price) FROM $S WHERE genre='sci-fi'`
$\mathcal{F}_6 \equiv$ `SELECT sum(rating) FROM $S`
$\mathcal{F}_7 \equiv$ `SELECT sum(rating) FROM $S WHERE genre='sci-fi'`
$\mathcal{F}_8 \equiv$ `SELECT sum(price) FROM $S`
$\mathcal{F}_9 \equiv$ `SELECT count(title) FROM $S WHERE genre='sci-fi' and price < 20.00`
$\mathcal{F}_{10} \equiv$ `SELECT sum(rating) FROM $S WHERE rating >= 4.0`

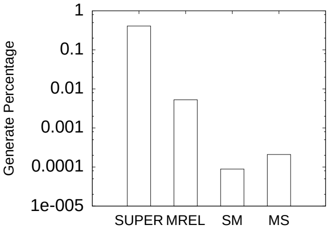
- **Set Preferences**

Set Pref. Name	Profile Schema Γ	Profile Pref. Formula C
SP1	$\langle \mathcal{F}_5, \mathcal{F}_6\rangle$	$\mathcal{F}_5(s_1) < \mathcal{F}_5(s_2) \wedge \mathcal{F}_6(s_1) > \mathcal{F}_6(s_2)$
SP2	$\langle \mathcal{F}_9, \mathcal{F}_{10}\rangle$	$\mathcal{F}_9(s_1) > \mathcal{F}_9(s_2) \wedge \mathcal{F}_{10}(s_1) < \mathcal{F}_{10}(s_2)$
SP3	$\langle \mathcal{F}_{11}, \mathcal{F}_{12}\rangle$	$\mathcal{F}_{11}(s_1) > \mathcal{F}_{11}(s_2) \wedge \mathcal{F}_{12}(s_1) > \mathcal{F}_{12}(s_2)$

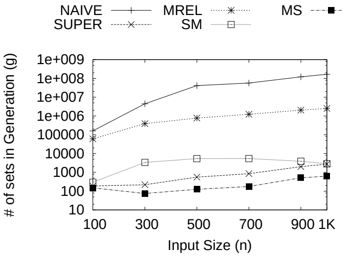
Performance of Different Algorithms

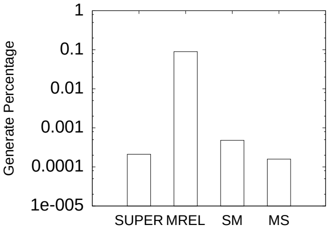
Set Pref 1





Set Pref 2

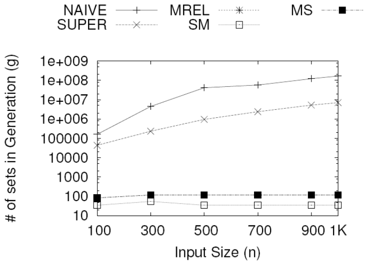


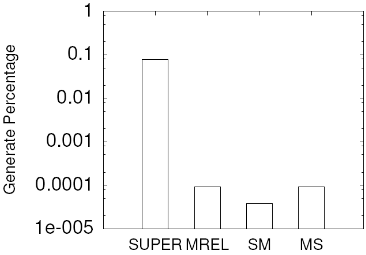


Performance of Different Algorithms

Set

Pref 3





Related Work

- Guha et al. [GGK+03]
  - Problem: find an optimal subset of tuples
  - Set property: $\mathit{aggr}(A) < \mathit{parameter}$
  - Set preference: objective function $\min / \max$
- Binshtok et al. [BBS+07]
  - Problem: find a optimal subset of items
  - Set property: predicate
  - Set preference: TCP-net or scoring function
  - Consider subsets of any cardinality, in the fixed-cardinality case, it is subsumed by our framework
- desJardins and Wagstaff [dW05]
  - Consider fixed-cardinality set preference
  - Consider two set features: *diversity* and *depth*

References I

Maxim Binshtok, Ronen I. Brafman, Solomon Eyal Shimony, Ajay Mani, and Craig Boutilier.
Computing optimal subsets.
In *AAAI*, pages 1231–1236, 2007.

Jan Chomicki.
Preference formulas in relational queries.
ACM Trans. Database Syst., 28(4):427–466, 2003.

Marie desJardins and Kiri Wagstaff.
DD-pref: A language for expressing preferences over sets.
In *AAAI*, pages 620–626, 2005.

References II

Sudipto Guha, Dimitrios Gunopulos, Nick Koudas, Divesh Srivastava, and Michail Vlachos.
Efficient approximation of optimization queries under parametric aggregation constraints.
In *VLDB*, pages 778–789, 2003.