Preference Elicitation with Subjective Features

Craig Boutilier

Department of Computer Science

University of Toronto

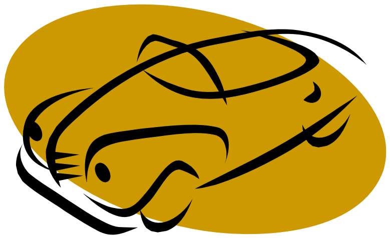
joint work with Kevin Regan and Paolo Viappiani;

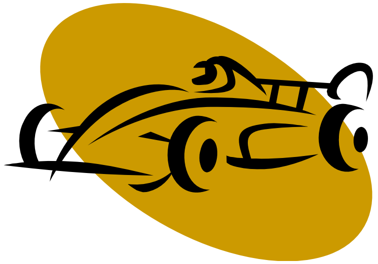
(and Laurent Charlin and Rich Zemel)

The Preference Bottleneck in AI*

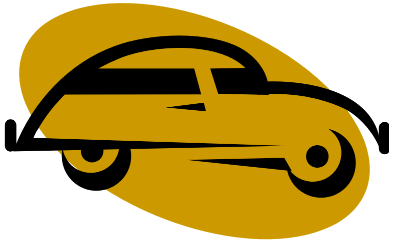
- Decisions on behalf of individuals (organizations)
  - match individuals to desired products, services, information, people, behaviors, courses of action
- Decision theory provides foundations for automated decision support systems
  - actions, outcomes, dynamics, utilities: MEU
- *But what is the objective function?*
  - user *preferences (or utilities)* are often unknown
  - vary much more widely than dynamics

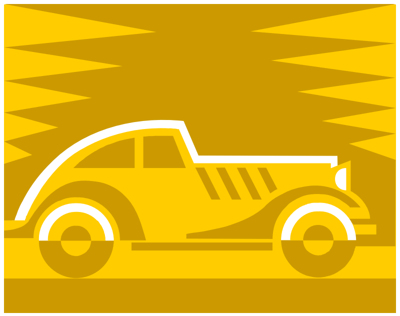
Product Configuration





Luggage Capacity?
Two Door? Cost?
Engine Size?
Color? Options?





The *Preference Bottleneck*

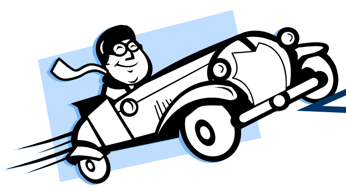
- The usual difficult questions:
  - decomposition of preferences
  - difficulty assessing precise tradeoffs, …
- Other difficult questions:
  - what are sources of preference information?
  - what preference info is *relevant* to the task at hand?
  - when is the elicitation effort *worth the improvement* it offers in terms of decision quality?
  - what *decision criterion* to use given partial utility info?

Beyond Stylized Queries

- Good progress using standard query-response models:
  - comparison queries, standard gambles, stated-choice methods,…
  - easy to formulate, formalize, analyze
- Drawbacks:
  - no exploration or construction of preferences
  - data-intensive at level of individual users
  - impact of cognitive biases (framing, anchoring, endowment,…)
  - *fixed vocabulary*
- Addressed in:
  - conversational recommendation systems
  - collaborative filtering, conjoint analysis
  - behavioral DT/economics, …
  - *but decision-theoretic (or social choice) foundations are weak*

Constructed (Subjective) Features

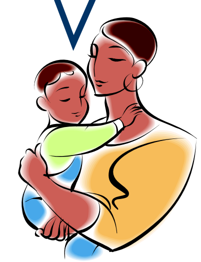
- "Catalog" attributes usually fix universe of discourse
  - e.g., luggage cap, engine size, city l /100km, crash test ratings, …
- Users may care about combinations of such attributes
  - *car safety*: function of size, airbag config, crash test ratings, …
  - but different users have different definitions



SAFETY MATTERS TO ME:
Brakes: brembo V hawk
Tires: high performance
FuelControl: Autoshutoff
etc.

SAFETY MATTERS TO ME:
AirBags: front & side
RearPassCrashRating: 4* V 5*
ChildRestr: TypeB V TypeC

SAFETY MATTERS TO ME:
AirBags: front & side
StabilityControl: Yes
Tires: Cat3 V Cat4 V Cat5
etc.



Subjective Features

- Goal: *personalized, constructed features* in the dialog
  - move beyond fixed vocabulary of catalog attributes
  - support more natural interactions, using features in which user naturally conceives of her preferences
  - key point: personalized features admit objective definitions

- Genuinely “subjective” features
  - judgments that defy definition in terms of objective features
    - e.g., want a car that is “sporty looking” or “cute”

Overview

- Minimax Regret Models of Preference Elicitation
  - robust optimization under utility uncertainty: minimax regret
  - refining utility uncertainty: regret-based query strategies
- User-defined (constructed) features
  - pure concept elicitation: assume known utility function
  - simultaneous concept and utility elicitation
- Subjective features in collaborative filtering

One-shot Decision Problem

- Finite set of *decisions* ***X***
  - decision (configuration) variables $X = \{X_1 \ldots X_n\}$
  - feasible set ***X*** defined by constraints, product DB, etc.
- *Utility function u:* ***X*** $\rightarrow$ *[0,1]*
  - simplified model: equates decisions with outcomes
- Optimal decision ***x**** maximizes utility
- Utility representation critical to assessment
  - some structural form usually assumed
  - so *u* parameterized compactly (weight vector *w*)
    - *e.g., linear/additive, generalized additive models*

Additive Utility Models

- *Additive models* commonly used in practice
  - *local value functions* v_i plus *scaling factors* λ_I

$$u(\mathbf{x}) = \sum_{i=1}^{n} u_i(x_i) = \sum_{i=1}^{n} \lambda_i v_i(x_i).$$

  - e.g., *u(Car) = 0.3 v_1(Color) + 0.2 v_2(Doors) + 0.5 v_3(Power)* and *v_1(Color) : cherryred:1.0, metallicblue:0.7, …, grey:0.0*
  - assess local VFs with *local SG queries*

$$x_i \sim \langle p, x_i^{\top}; 1-p, x_i^{\perp} \rangle \Longleftrightarrow v_i(x_i) = p$$

  - assess scaling factors with 2n "global" queries

$$\lambda_j = u(\mathbf{x}^{\top j}) - u(\mathbf{x}^{\perp j})$$

Elicitation Tradeoffs

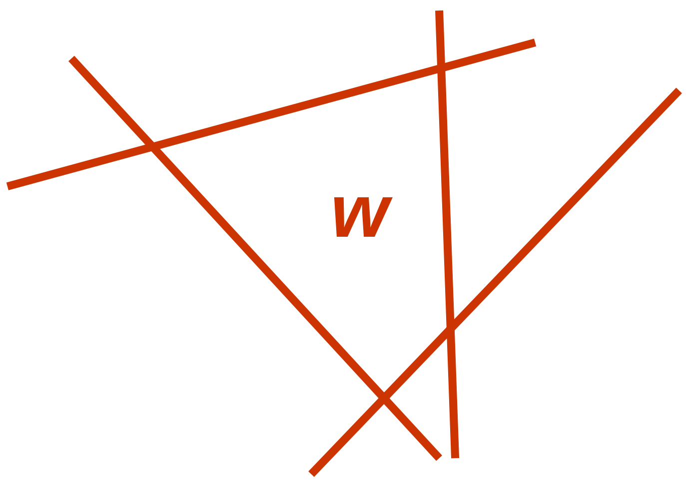
- Burden of complete utility information too much to bear
  - large number of parameters to assess
  - unreasonable precision required
  - cost (cognitive, communication, computational, revelation) may outweigh benefit
    - can often make *optimal* decisions without full utility information (exploiting feasibility constraints)
- General approach: incremental elicitation until a decision can be made that is “good enough”
  - simple queries: *constrain* parameters, don’t *identify* parameters

Elicitation in Additive Models: Comparisons

- *Somewhat simplified view (ignoring calibration across features)*

- *Comparison queries* (is ***x*** preferred to ***x'*** ?)
  - impose linear constraints on parameters
    - $\Sigma_k\ u_k(x_k) > \Sigma_k\ u_k(x_k)$
  - local variants possible (on single attributes)

$u_c(\text{red})+u_d(\text{2door})+u_e(\text{280hp}) > u_c(\text{blue})+u_d(\text{2door})+u_e(\text{280hp})$



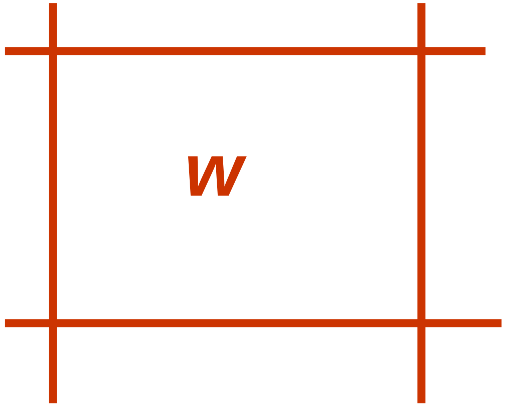
Elicitation in Additive Models: Bound Queries*

- *Somewhat simplified view (ignoring calibration across features)*

- *Bound queries* (is $u_k(x_k) > v$?)
  - response tightens bound on specific utility parameter
  - a boolean version a (global/local) standard gamble query
    - "Do you prefer x_k to $(x^T_k, p;\ x^{\perp}_k, 1\text{-}p)$?"

$$u_{enginesize}(280hp) > 0.4$$

$$u_{color}(red) < 0.7$$



Other Modes of Interaction*

- Choose from set of alternatives
- Ranking set of alternatives
- Graphical manipulation of parameters
  - bound queries: allow tightening of bound (user controlled)
  - approximate valuations: user-controlled degree of precision (useful for quasi-linear environments)

A General Framework for Elicitation and Interactive Decision Making

- *B: beliefs about user's utility function u*
- *Opt(B): "optimal" decision given incomplete, noisy, and/or imprecise beliefs about u*

- Repeat until *B* meets some termination condition
  - ask user some query (propose some interaction) *q*
  - observe user response *r*
  - update *B* given *r*
- Return/recommend *Opt(B)*

- Our queries leave us with *strict* utility uncertainty
  - need some form of robust optimization
  - use *minimax regret* to make decisions and suggest queries

Minimax Regret

- Utility uncertainty given by *feasible set W*
  - e.g., *W* defined by linear constraints on *w*

$u_e(280hp) > 0.4$

$u_c(red)+u_d(2door)+u_e(280hp) > u_c(blue)+u_d(2door)+u_e(280hp)$

  - *Regret of x under w:* $R(x, \mathbf{w}) = \max_{x' \in X} u(x'; \mathbf{w}) - u(x; \mathbf{w})$
  - *Max regret of x under W:* $MR(x, W) = \max_{\mathbf{w} \in W} R(x, \mathbf{w})$
  - *Minimax regret and optimal allocation:*

$$x_W^* = \arg\min_{x \in X} MR(x, W)$$

Minimax Regret: An Example

- Simple example to contrast maxmin, MMR

	U1	U2	U3	Min	MR
D1	8	2	1	1	5
D2	7	7	1	1	1
D3	2	2	2	2	6

- Maxmin: recommends D3 (too cautious?)
- MMR recommends D2
  - might be worse than D3, but never by more than a little

Why Minimax Regret?*

- Minimizes regret in presence of adversary
  - provides bound worst-case loss (cf. maximin)
  - *robustness* in the face of utility function uncertainty
- In contrast to Bayesian methods:
  - useful when priors not readily available
  - can be more tractable; see **[CKP00/02, Bou02]**
  - user unwilling to “leave money on the table” **[BSS04]**
  - preference aggregation settings **[BSS04]**
  - effective elicitation even *if* priors available **[WB03]**

Computing Minimax Regret

- Difficulties computing minimax regret:
  - underlying optimization generally an IP
  - minimax (integer) program (not straight min or max)
  - generally quadratic objective

$$MMR(W) = \min_{x \in X} \max_{w \in W} \max_{x' \in X} w \cdot x' - w \cdot x$$

- General Approach:
  - Bender’s decomposition and constraint generation to break minimax program
  - Various encoding tricks to linearize quadratic terms
    - details and formulation depend on domain

Minimax Regret: Bender's Reformulation*

- With unknown utility parameters in *W*

$$MMR(W) = \min_{x \in X} \max_{w \in W} \max_{x' \in X} u(x';w) - u(x;w)$$

- Linear IP formulation (infinitely many constraints)

$$\begin{aligned} &\min_{x \in X} \ \delta \\ &s.t. \quad \delta \geq u(x';w) - u(x;w) \quad \forall x' \in X, \forall w \in W \end{aligned}$$

- Linear IP formulation (exponentially many constraints)

$$\begin{aligned} &\min_{x \in X} \ \delta \\ &s.t. \quad \delta \geq u(x_w^*;w) - u(x;w) \quad \forall w \in V(W) \end{aligned}$$

Constraint Generation*

- Avoid W-vertex enumeration: *constraint generation*
- Let $Gen = \{(x',w)\}$ for some feasible x', $w \in W$
  - solve $$\min_{x \in X} \delta$$ $$s.t. \quad \delta \geq u(x';w) - u(x;w) \quad \forall (x',w) \in Gen$$
    - let solution be x^* with objective value δ^*
  - compute max regret $MR(x^*,W)$ of solution x^*
    - solution has max regret r, witness (x'', w'')
  - if $r > \delta^*$, add (x'', w'') to *Gen*, repeat; else terminate
    - note: (x'', w'') is *maximally* violated constraint

Computing Max Regret**

- Objective is naturally quadratic

$$MR(x,W) = \max_{w \in W} \max_{x' \in X} u(x';w) - u(x;w)$$

- However, feature instantiations are discrete
  - quadratic terms: products of integer, continuous vars
  - easily linearized by introduction of auxiliary variables

$$MR(\mathbf{x},\mathbf{U}) = \max_{\{I_{\mathbf{x}'[k]}, X'_i, U_{\mathbf{x}[k]}, Y_{\mathbf{x}'[k]}\}} \sum_k \left(\sum_{\mathbf{x}'[k]} Y_{\mathbf{x}'[k]} \right) - U_{\mathbf{x}[k]}$$

Details of Linearization***

- Replace quadratic term $U_{x'[k]}\, I_{x'[k]}$ by new variable $Y_{x'[k]}$
  - assume (loose) upper bound on each utility parameter $u_{x'[k]}$
  - constraints ensure $Y_{x'[k]} = 0$ if $I_{x'[k]} = 0$; and $Y_{x'[k]} = U_{x'[k]}$ otherwise

$$MR(\mathbf{x}, \mathbf{U}) = \max_{\{I_{\mathbf{x}'[k]}, X_i', U_{\mathbf{x}[k]}, Y_{\mathbf{x}'[k]}\}} \sum_k \left(\sum_{\mathbf{x}'[k]} Y_{\mathbf{x}'[k]} \right) - U_{\mathbf{x}[k]}$$

$$\text{subject to} \begin{cases} Y_{\mathbf{x}'[k]} \le I_{\mathbf{x}'[k]} u_{\mathbf{x}'[k]}^{\uparrow} & \forall k, \mathbf{x}'[k] \\ Y_{\mathbf{x}'[k]} \le U_{\mathbf{x}'[k]} & \forall k, \mathbf{x}'[k] \\ \mathcal{A}, \mathcal{C} \text{ and } \mathcal{U} \end{cases}$$

Max Regret for Parameter Bounds***

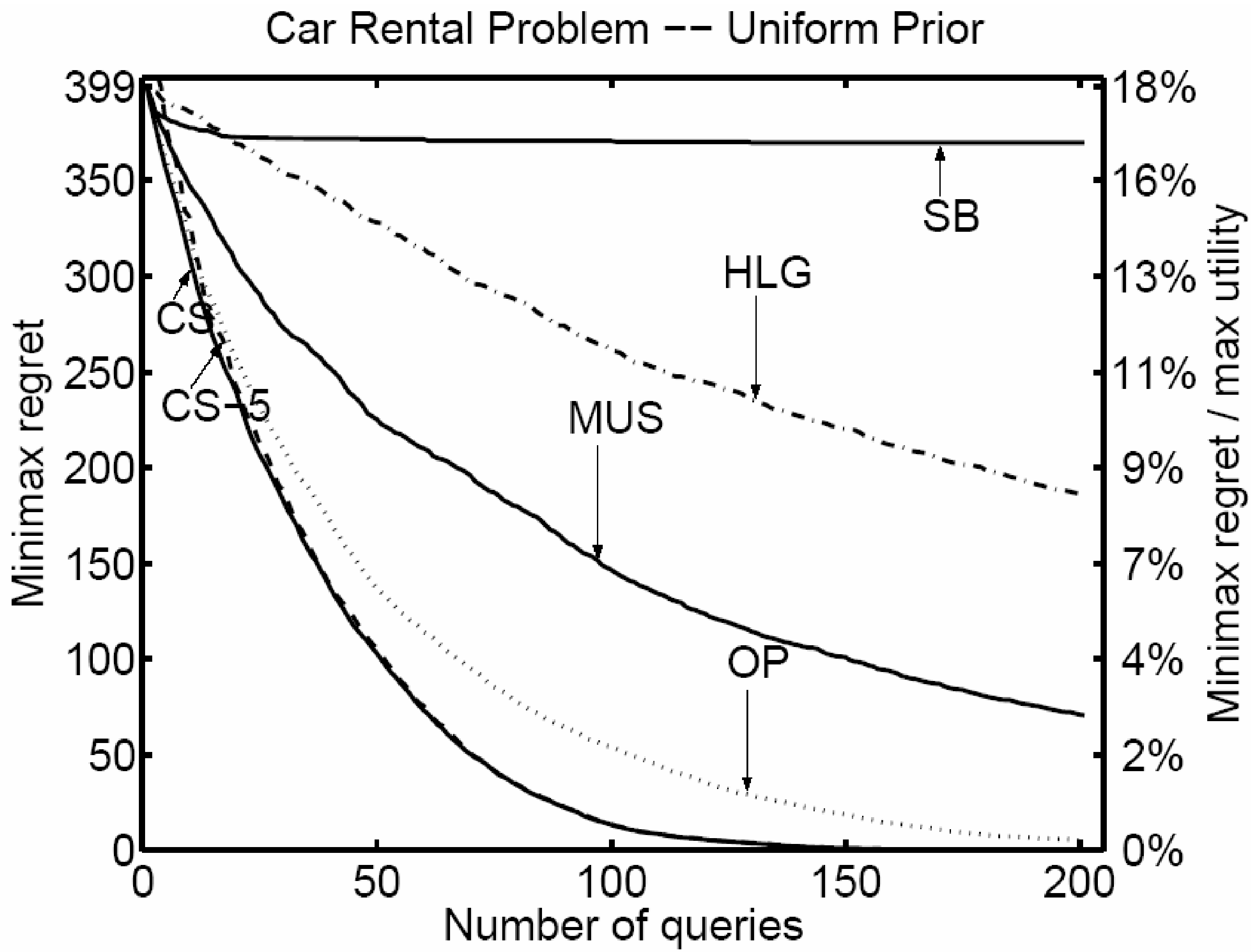
- Max regret computation is even simpler in the case of simple upper and lower bounds on utility parameters
  - hyperrectangular polytope (see bound queries)
  - requires integer variables only (selection of outcome)
  - pairwise regret is constant for each local configuration

Regret-based Query Strategies

- Have explored a variety of query *strategies*
  - what query should I ask user to reduce minimax regret “quickly”?
- *Current solution strategy (CSS)* works well in practice
  - ask queries that impact utility parameters of current minimax optimal solution or current adversarial witness
  - realization depends on precise form of query

Is x^*: <red,2door,280hp>
Preferred to x^w: <blue,4door,195hp> ?

Results (Car Rental, Unif)



26 vars; 61 billion configs

36 factors, at most 5 vars; 150 parameters

Users drawn using uniform prior over parameters (45 trials)

Gaussian priors similar

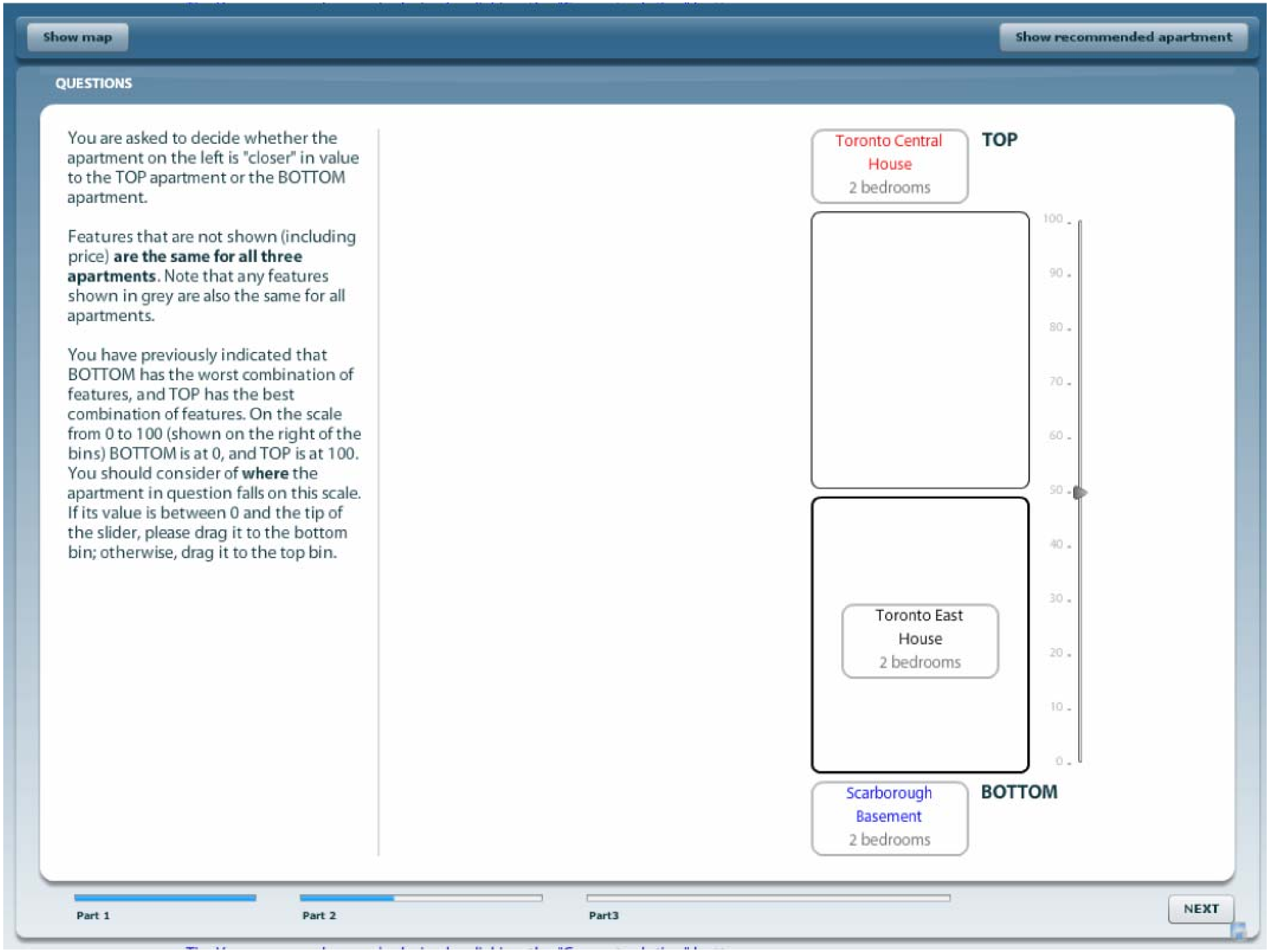
Apartment Search (DB= 100, 9 attributes, 7 factors)

Main | Solution | Database

DATABASE

ID	PRICE	Area	Building type	No. of bedrooms	Furniture	Laundry	Parking	Smoking restrictions
25	850	East Toronto	House	2 bedrooms	Unfurnished	Laundry available	Parking not available	Smoking allowed
26	1200	West Toronto	House	3 bedrooms	Unfurnished	Laundry not available	Parking not available	Smoking not allowed
27	1000	Scarborough	Basement	2 bedrooms	Furnished	Laundry available	Parking not available	Smoking not allowed
28	1400	Downtown	High-rise	1 bedroom	Unfurnished	Laundry available	Parking available	Smoking allowed
29	750	West Toronto	House	1 bedroom	Unfurnished	Laundry available	Parking not available	Smoking not allowed
30	650	East Toronto	Basement	1 bedroom	Furnished	Laundry available	Parking available	Smoking allowed
31	1200	Downtown	High-rise	1 bedroom	Furnished	Laundry available	Parking available	Smoking allowed
32	650	West Toronto	Basement	1 bedroom	Furnished	Laundry available	Parking not available	Smoking not allowed
33	1100	Downtown	Basement	2 bedrooms	Unfurnished	Laundry available	Parking available	Smoking allowed
34	600	Scarborough	Basement	1 bedroom	Unfurnished	Laundry available	Parking not available	Smoking allowed
35	1200	West Toronto	Basement	2 bedrooms	Furnished	Laundry not available	Parking not available	Smoking allowed
36	700	West Toronto	Basement	1 bedroom	Unfurnished	Laundry available	Parking not available	Smoking allowed
37	745	Downtown	High-rise	1 bedroom	Furnished	Laundry not available	Parking available	Smoking not allowed
38	775	Downtown	High-rise	1 bedroom	Unfurnished	Laundry available	Parking not available	Smoking not allowed
39	650	Scarborough	Basement	1 bedroom	Unfurnished	Laundry available	Parking available	Smoking allowed
40	900	East Toronto	High-rise	1 bedroom	Unfurnished	Laundry available	Parking available	Smoking not allowed
41	900	Scarborough	Basement	2 bedrooms	Furnished	Laundry available	Parking available	Smoking allowed
42	750	Scarborough	Basement	2 bedrooms	Unfurnished	Laundry not available	Parking not available	Smoking allowed
43	995	Downtown	High-rise	1 bedroom	Unfurnished	Laundry available	Parking available	Smoking not allowed
44	1360	Downtown	High-rise	2 bedrooms	Unfurnished	Laundry available	Parking available	Smoking not allowed
45	650	Scarborough	Basement	1 bedroom	Furnished	Laundry available	Parking not available	Smoking allowed
46	1100	West Toronto	House	1 bedroom	Furnished	Laundry available	Parking not available	Smoking not allowed

Apartment Search (DB= 100, 9 attributes, 7 factors)



Apartment Search (DB= 100, 9 attributes, 7 factors)

Apartment Search (DB= 100, 9 attributes, 7 factors)

$1150?

Toronto Central
House
2 bedrooms
Unfurnished
Laundry available
Parking available
No dishwasher
Storage room
Air-conditioned

Would you be willing to pay $1150 or more for this apartment?

Yes   No

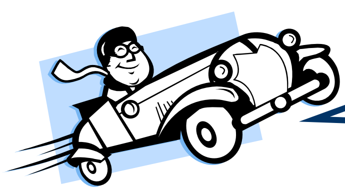
Effectiveness of Regret-based Elicitation

- Recent user study **[Braziunas, B; under review]** suggests:
  - minimax regret is comprehensible, reasonably intuitive
    - some query types more acceptable than others
  - converges on near-optimal decisions in multiattribute databases
  - converges much more quickly (time, “effort”) than search through a small database
  - user satisfaction with engagement is very high

- Other lessons
  - additive models not realistic, but…
  - machinery needed to elicit GAI models “soundly” is overkill

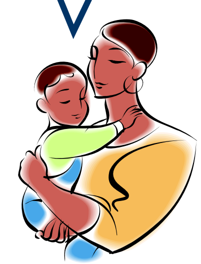
Non-catalog, Constructed Features

- Call configuration vars $X = \{X_1 \ldots X_n\}$ *catalog attributes*
  - catalog spec; those objective features that define the item
- Users may care about combinations of such attributes
  - *car safety*: function of size, airbag config, crash test ratings, …



Brakes: brembo v hawk
Tires: high performance
FuelControl: Autoshutoff
etc.

AirBags: front&side
RearPassCrashRating: 4* v 5*
ChildRestr: TypeB v TypeC



AirBags: front&side
StabilityControl: Yes
Tires: Cat3 v Cat4 v Cat5
etc.

Fundamental Objectives

- Keeny's VFT: *fundamental* vs. *means* objectives
  - **Carenini** made these distinctions for recommenders
  - **FindMe** static compound critiques ("more sporty") or Stolze user types ("family snaps" vs. "professional")
  - **BBGP97**: *configurable* vs. *functional* variables in pref. elicitation
    - X configurable; Y functional; mapping $f: X \rightarrow Y$
    - constraints over X, elicitation over Y

Tradeoffs in Eliciting User Features

- Change to preference model not *necessarily* needed
  - but fundamentally changes nature of interaction
  - *need to elicit open-ended, user-initiated features*
    - possibly defaults with tweakable definitions
- Goal: elicit as little as necessary to make a good decision
  - utility model, feasibility constraints mean (often) near-optimal decisions possible with weak knowledge of definitions
  - want to trade off elicitation effort with decision quality

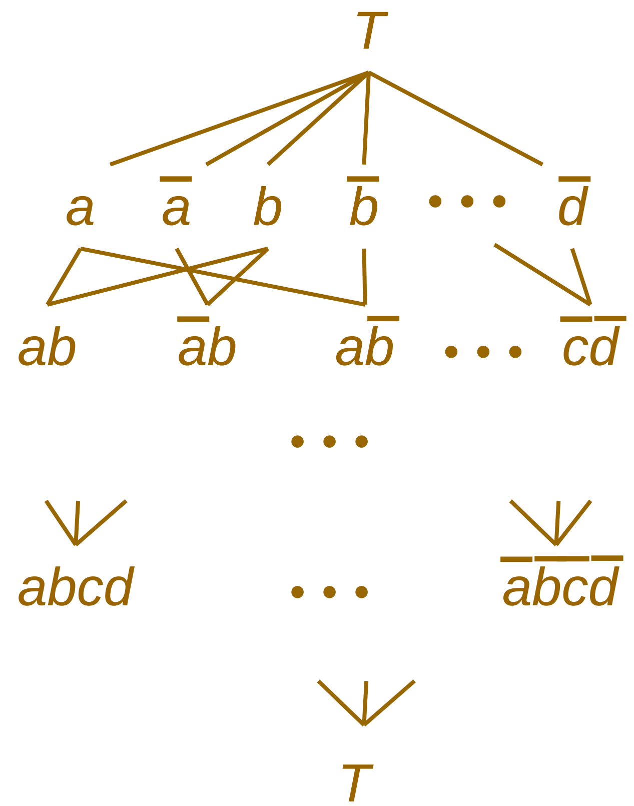
> - "Safe car" requires feature X=x1 plus other unspecified stuff;
> - X=x1 implies Y /= y2;
> - Y=y2 more important that "safety";
> - no value in further elicitation of defn: no safe car is optimal

Eliciting Constructed Features: Model

- *Initial focus: known utility function, uncertain definition*
- Product space $\boldsymbol{X} \subseteq Dom\{X_1 \ldots X_n\}$
  - reward $r(\boldsymbol{X})$ reflects utility for catalog features
  - concept $c(\boldsymbol{X})$ drawn from some hypothesis space H
  - bonus p: additional utility for an $\boldsymbol{x}$ satisfying $c(\boldsymbol{x})$
  - *utility* $u(\boldsymbol{x}) = r(\boldsymbol{x}) + p\, c(\boldsymbol{x})$
- How do we elicit *c*?
  - concept learning: "accurate" identification of *c*
    - e.g., MB model, PAC model, query model
  - our goal: learn *just enough* about *c* to identify a single good/optimal instance; and minimize user queries
    - e.g., compare **[BJSZ04]** (eliciting value functions)

Concept Learning: Key Aspects

- Hypothesis space H
  - e.g., nonmonotone conjunctions
  - e.g., *a, $\bar{a}b$, $b\bar{c}d$, $ab\bar{c}d$*
- Queries used to help identify concept *C*
  - e.g., membership queries:
    - *is abcd in C?*
- Version space $V \subseteq H$
  - $c \in V$ iff c respects prior knowledge, responses, etc.
- Strategies, performance metrics
- Simple algorithm for nonmonotone conjunctions
  - ask random membership queries until positive instance *p* found
  - negate literals in p one at a time and ask query of that instance
  - exponential ; linear once *p* found



Version Spaces and MMR

- Let $V \subseteq H$ be current version space
  - $c \in V$ iff c respects prior knowledge, responses, etc.
- If choice ***x*** must be made, use minimax regret

$$MR(\mathbf{x}; V) = \max_{c \in V} \max_{\mathbf{x}' \in \mathbf{X}} u(\mathbf{x}'; c) - u(\mathbf{x}; c)$$

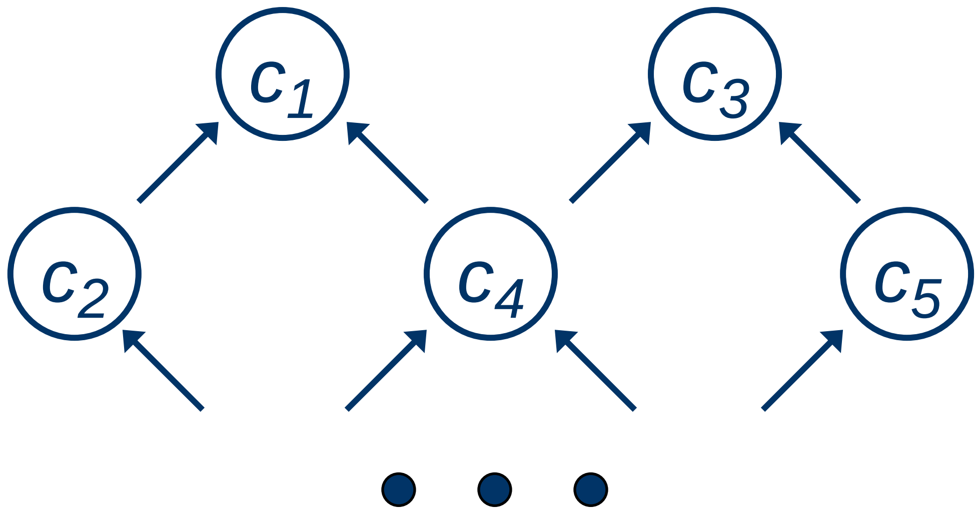
$$MMR(V) = \min_{\mathbf{x} \in \mathbf{X}} MR(\mathbf{x}, V)$$

$$\mathbf{x}^*_V = \arg\min_{\mathbf{x} \in \mathbf{X}} MR(\mathbf{x}, V)$$

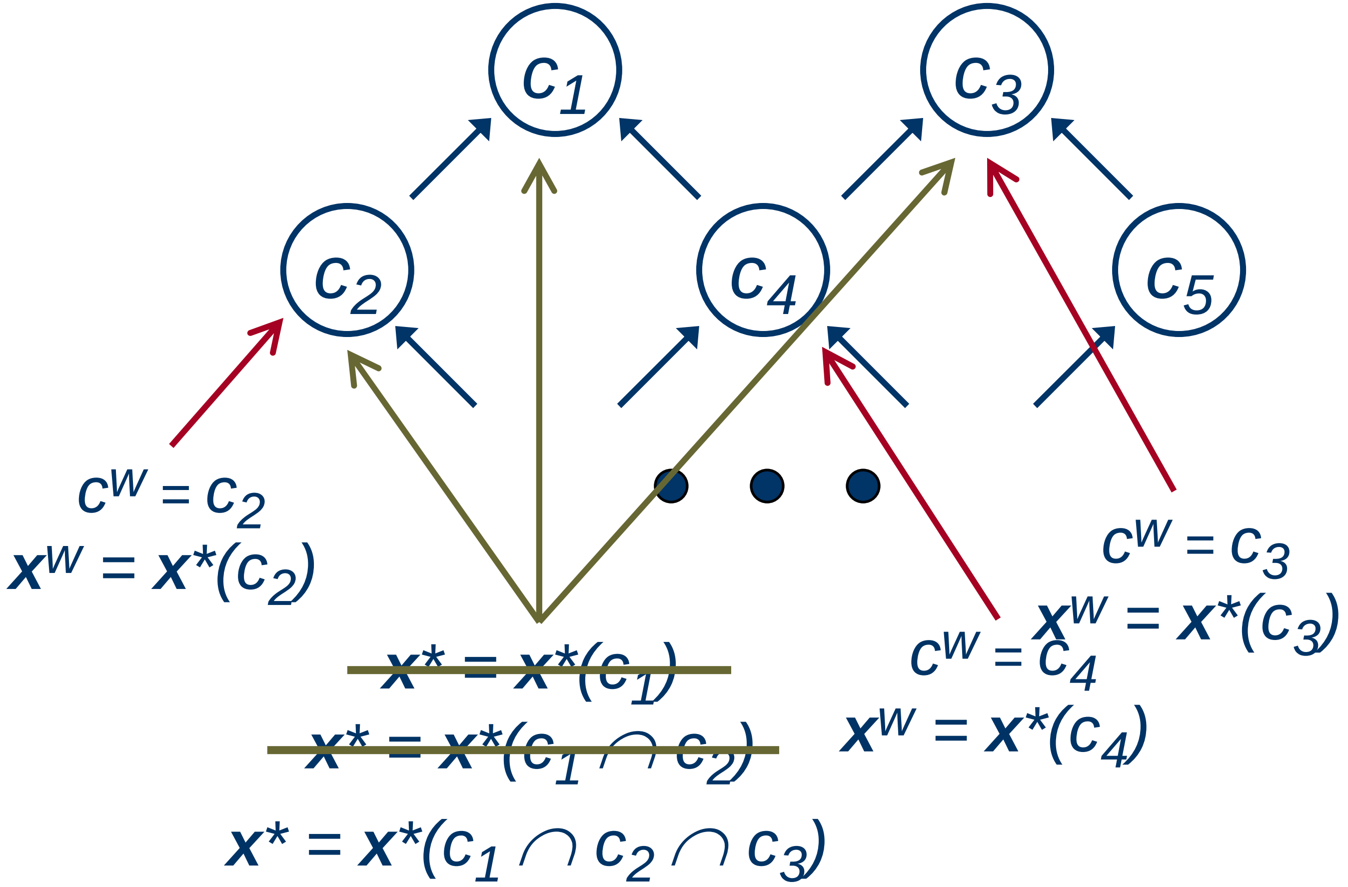
  - If *MMR(V)* = ε, ***x**** is ε–optimal
- Can determine optimal ***x*** with little info about *c*
  - e.g., if r is constant; if Δ(r) > p; etc.

Characterizing MMR-Optimal Soln

- MMR-optimal soln $\mathbf{x}^*$, $\mathbf{x}^w$, c^w: interesting structure
  - $\mathbf{x}+$, $r^+ = r(\mathbf{x}^+) = max\ \{ r(\mathbf{x}) : \mathbf{x} \in \mathbf{X}\}$
  - general-specific lattice $\geq$ over V: $c \geq c'$ iff $c' \subseteq c$
  - $\mathbf{x}^*(c)$: best satisfying c; $r^*(c) = max\ \{ r(\mathbf{x}) : \mathbf{x} \in c,\ \mathbf{x} \in \mathbf{X}\}$
    - induces reward-ordering over V: $r^*(c_1) \geq r^*(c_2) \geq \ldots$
    - reward ordering respects GS ordering



Characterizing MMR-Optimal Soln



Characterizing MMR-Optimal Soln

- Order $r_1 > r_2 \ldots > r_m$ elements of $\{r_c^* : c \in V\}$
- Let $C_i = \{c \in V : r_c^* = r_i\}$ and $S_i = \cap C_i$

Proposition 1 *If* $\mathbf{x}_V^*$ *is not consistent with* V*, then* $\mathbf{x}_V^* \in X^+$ *(and all elements of* X^+ *have identical max regret).*

Proposition 2 *If* $\mathbf{x}_V^* \notin X^+$*, then: (a)* $\mathbf{x}_V^*$ *is consistent with* V*; (b)* $\mathbf{x}_V^* \in \arg\max\{r(\mathbf{x}) : \mathbf{x} \in S_1 \cap \ldots \cap S_i\}$ *for some* $i \geq 1$*; and (c) either* $c^w \in C_1$*, or* $c^w \in C_{i+1}$*.*

Observation 2 $\mathbf{x}_V^* \in c^w$ *only if* $\mathbf{x}^w \in c^w$*.*

Computing MMR: Conjunctions

- Often *max { r(**x**) : **x** ∈ **X**}* defined by a MIP
  - want to compute MMR by encoding within MIP
  - special case: conjunctions, memberships queries
    - *e.g., "Do you consider this to be a safe car?"*
- Let $\mathbf{x}_c = \arg\max_{\mathbf{x} \in \mathbf{X}} u(\mathbf{x}; c)$, then *MMR(V)* is:

$$
\begin{aligned}
\min \quad & \delta \\
\text{s.t.} \quad & \delta \geq r(\mathbf{x}_c) - r(X_1, \cdots X_n) + p(\mathbf{x}_c, c) - pI^c \quad \forall c \in V \\
& I^c \leq X_j \quad \forall c \in V, \forall x_j \in c \\
& I^c \leq 1 - X_j \quad \forall c \in V, \forall \overline{x}_j \in c
\end{aligned}
$$

Computing MMR: Conjunctions*

- Constraint generation: avoid enumeration of *V*
  - solve w/ subset of *V*, find max violated constraint, add if nonzero
  - max violated constraint: concept that maximizes regret $MR(\mathbf{x}^*, V)$
  - Let E^+, E^- be positive, negative instances

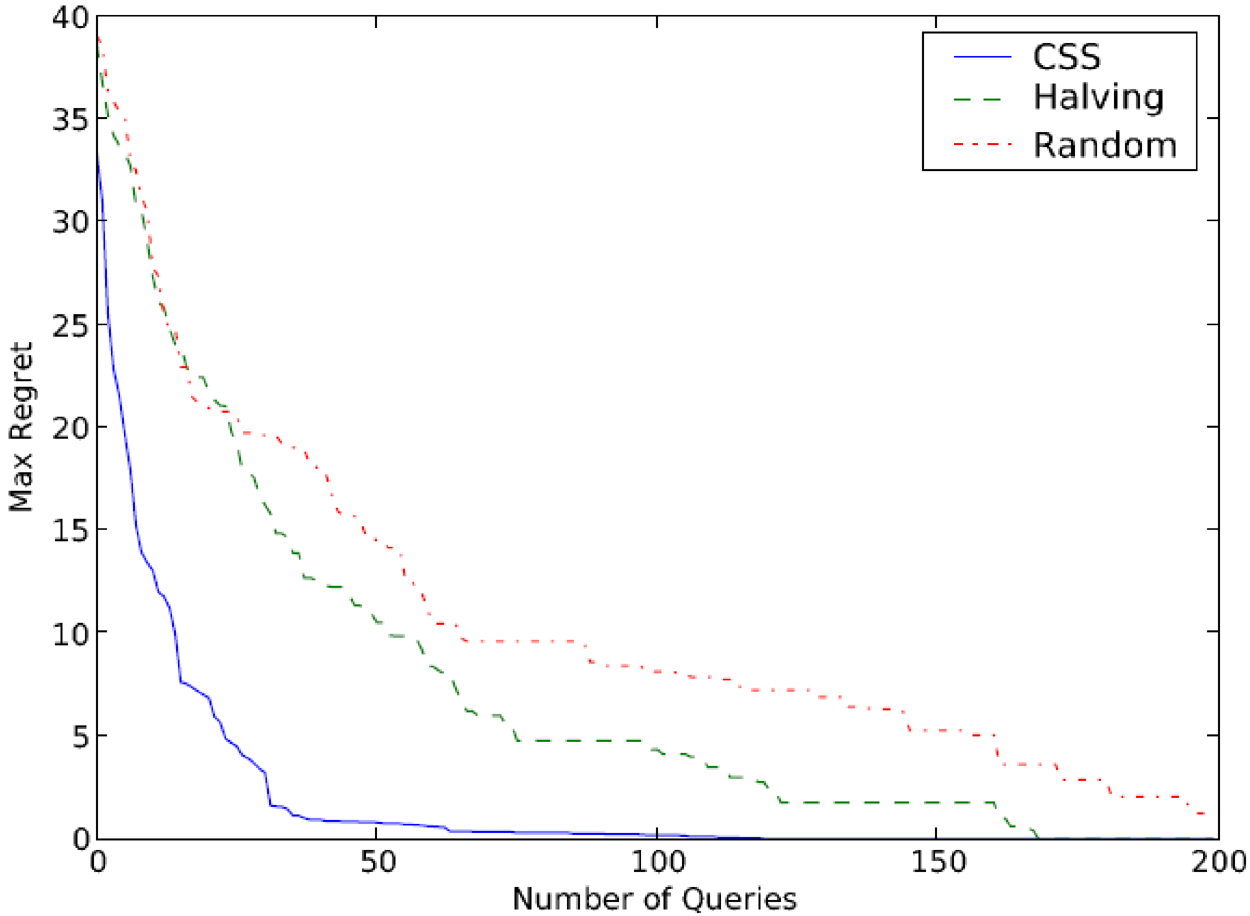
$$
\begin{aligned}
\max \quad & r(X_1, \cdots, X_n) - r(\mathbf{x}) + pB^w - pB^x \\
\text{s.t.} \quad & B^w + I(x_j) \leq X_j + 1.5 \quad \forall j \leq n \\
& B^w + I(\overline{x}_j) \leq (1 - X_j) + 1.5 \quad \forall j \leq n \\
& B^x \geq 1 - \sum_{j:\mathbf{x}[j] \text{ positive}} I(\overline{x}_j) - \sum_{j:\mathbf{x}[j] \text{ negative}} I(x_j) \\
& \sum_j I(\neg \mathbf{y}[j]) = 0 \quad \forall \mathbf{y} \in E^+ \\
& \sum_j I(\neg \mathbf{y}[j]) \geq 1 \quad \forall \mathbf{y} \in E^- \\
& (X_1, \cdots, X_n) \in \mathbf{X}
\end{aligned}
$$

Query Strategies

- Aim: reduce regret quickly
- Several strategies using membership queries:
  - *Halving*: aims to learn concept directly
    - "random" query $\boldsymbol{x}$ until positive response; then refine (unique) most specific concept in V (negate one literal at a time)
  - *Current Soln (CS)*: tackle regret directly
    - If $\boldsymbol{x}^*$, $\boldsymbol{x}^w$ both in c^w : query $\boldsymbol{x}^w$ (unless certain)
    - If $\boldsymbol{x}^w$ in c^w but not $\boldsymbol{x}^*$: query $\boldsymbol{x}^*$ (unless certain)
    - If $\boldsymbol{x}^*$, $\boldsymbol{x}^w$ both not in c^w : query $\boldsymbol{x}^w$ if $\boldsymbol{x}^w$ is V-consistent, o.w. $\boldsymbol{x}^*$
  - Several variants show modest improvements

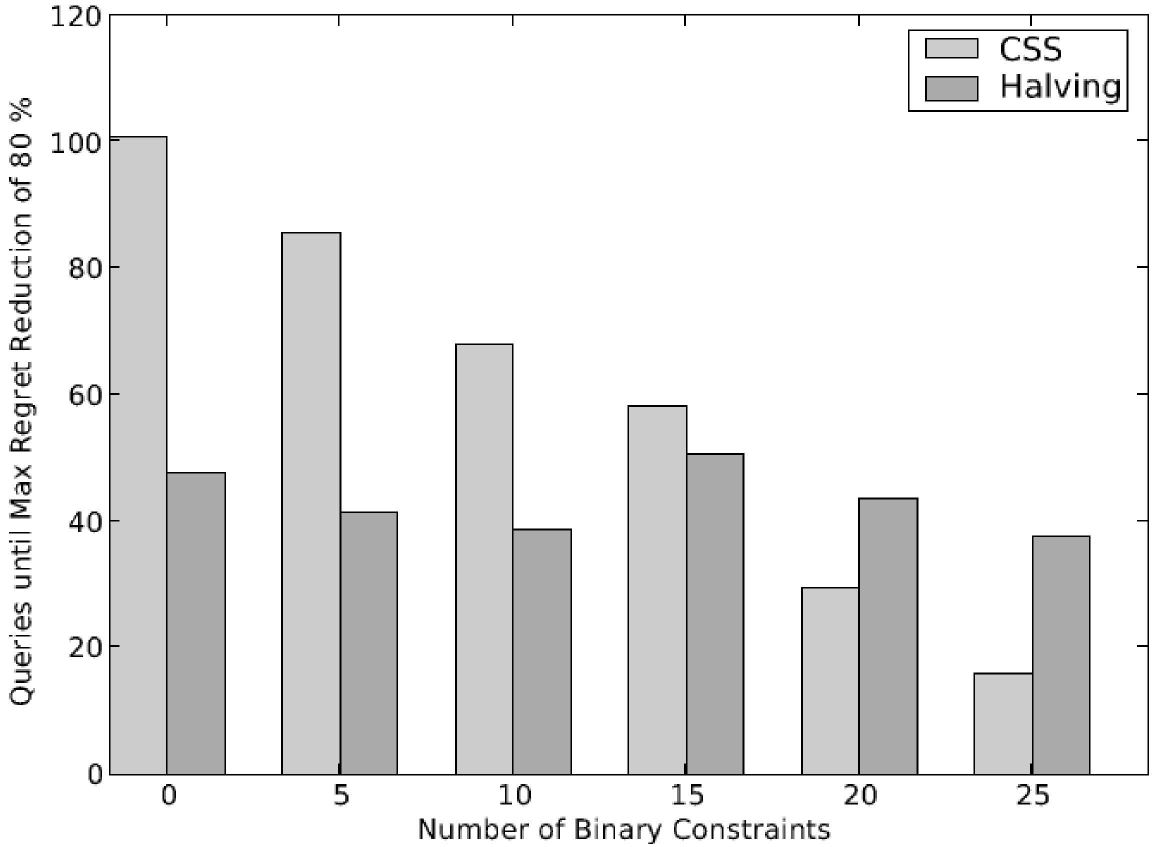
“Typical” Results

- 30 variables, 20 random binary constraints, concepts have size 10, random reward/bonus, bonus = 25% of max reward



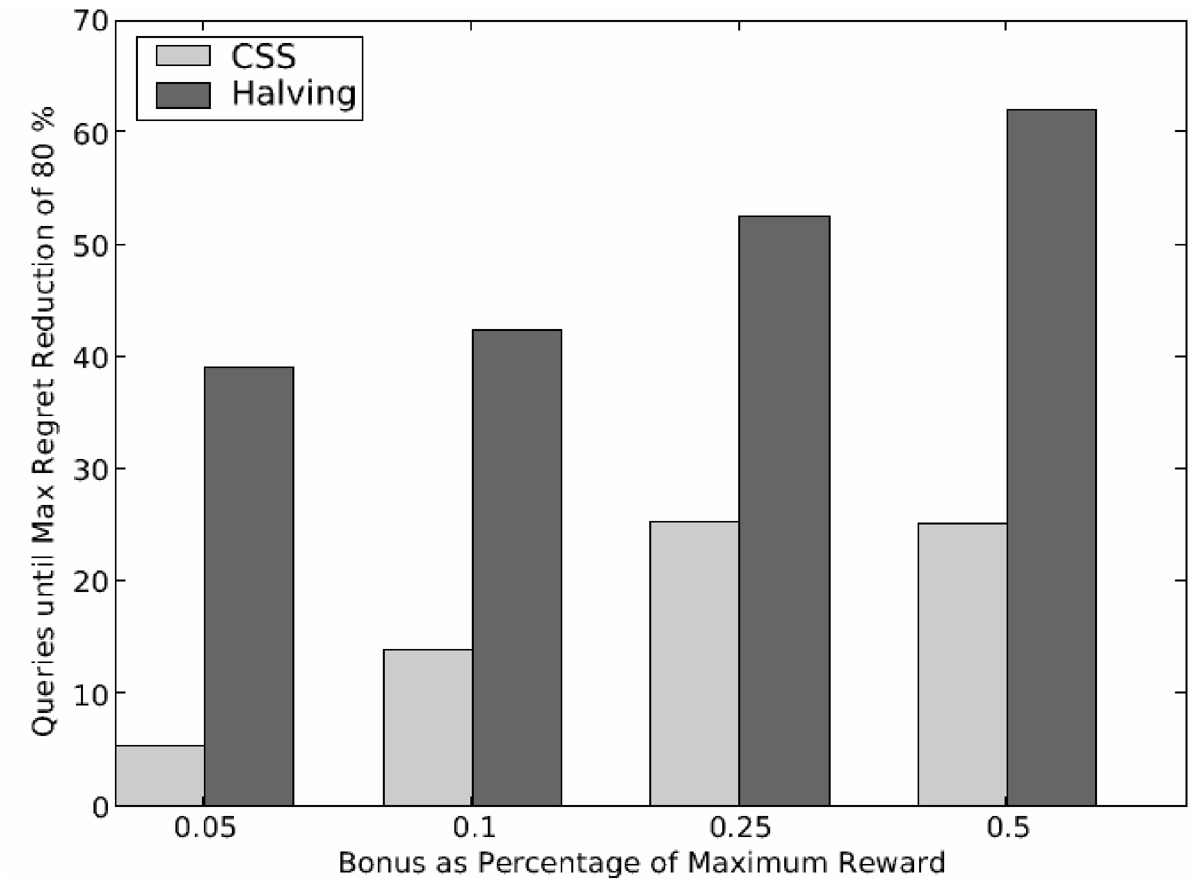
Varying Constraint Tightness

- Tighter constraints: sparser solution sets, more variability in *r** values, more concepts in *V* without positive instances in ***X***
  - shown: number of queries to reach regret reduction of 80%



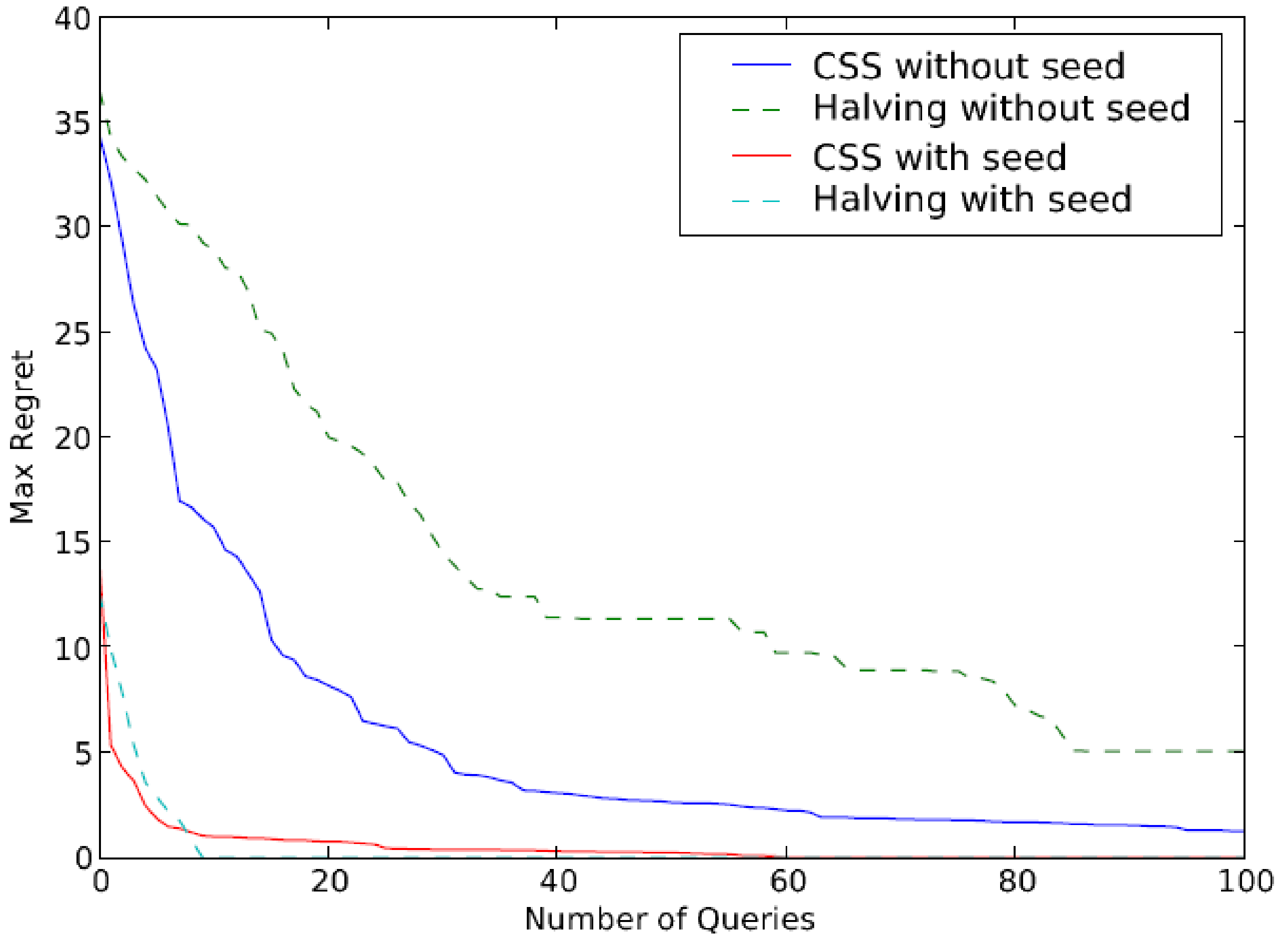
Varying Relative Bonus

- Greater bonus value: refining the concept becomes more critical
  - shown: queries to reach regret reduction of 80% (20 constraints)



Positive Instance as Seed

- Once positive instance found, true "halving" kicks in
  - assume user identifies a positive example immediately



Incorporating Utility Uncertainty

- Utility (reward, bonus) not really known
  - require *simultaneous utility and feature elicitation*
  - doing one "completely" followed by other is wasteful
- Challenges
  - what are appropriate query strategies (tradeoffs)
  - semantics of elicitation queries more complicated

Utility and Concept Uncertainty: Model

- As before: product space $\boldsymbol{X} \subseteq Dom\{X_1 \ldots X_n\}$
  - reward $r(\boldsymbol{X})$ reflects utility for catalog features
  - concept $c(\boldsymbol{X})$ drawn from some hypothesis space H
  - bonus b: additional utility for an $\boldsymbol{x}$ satisfying $c(\boldsymbol{x})$
  - *utility* $u(\boldsymbol{x}) = r(\boldsymbol{x}) + b\ c(\boldsymbol{x})$
- As before, concept $c(\boldsymbol{X})$ unknown
- In addition, utility parameters w (including b) unknown
  - assume additive utility model: $r(\boldsymbol{x},w) = \sum_k u_k(x_k)$
- Minimax regret: over utility *and* concept uncertainty

MMR with Utility, Concept Uncertainty

- $V \subseteq H$: current version space; *W*: current utility polytope
- If choice ***x*** must be made, use minimax regret

$$MR(\mathbf{x}; W, V) = \max_{w \in W} \max_{c \in V} \max_{\mathbf{x}' \in \mathbf{X}} u(\mathbf{x}'; w, c) - u(\mathbf{x}; w, c)$$

$$MMR(W, V) = \min_{\mathbf{x} \in \mathbf{X}} MR(\mathbf{x}; W, V)$$

$$\mathbf{x}^*_{W,V} = \arg\min_{\mathbf{x} \in \mathbf{X}} MR(\mathbf{x}; W, V)$$

  - If $MMR(W,V) = \varepsilon$, ***x**** is ε–optimal
  - Note: definition can be generalized if *W, V* coupled
- Can determine optimal ***x*** with little info about *w, c*

Computing MMR: Conjunctions

- Compute MMR by encoding within MIP
  - special case: conjunctions, memberships queries
- Let $\mathbf{x}^*_{w,c} = \arg\max u(\mathbf{x}; w, c)$
  - $b(\mathbf{x}_{w,c}, c)$ constant: w_b if $c(\mathbf{x})$; 0 otherwise.
- Then *MMR(V)* is:

$$
\begin{aligned}
\min \quad & \delta \\
\text{s.t. } & \delta \geq r(\mathbf{x}^*_{w,c}) - r(X_1, \cdots X_n) \\
& \qquad + b(\mathbf{x}^*_{w,c}, c) - w_b I^c \quad \forall c \in V, \forall w \in W \\
& I^c \leq X_j \qquad \forall c \in V, \forall x_j \in c \\
& I^c \leq 1 - X_j \qquad \forall c \in V, \forall \overline{x}_j \in c
\end{aligned}
$$

Computing MMR: Conjunctions*

- Constraint generation: avoid enumeration of *W*,*V*
  - max violated constraint: concept that maximizes regret *MR(**x***,W,V)*
  - Let E^+, E^- be positive, negative instances

$$
\begin{aligned}
\max \quad & \sum_{j \le n} Y_j + Z^a - \sum_{j \le n} w_j \mathbf{x}[j] - Z^x \\
\text{s.t.} \quad & B^a + I(x_j) \le X_j + 1.5 \quad \forall j \le n \\
& B^a + I(\overline{x}_j) \le (1 - X_j) + 1.5 \quad \forall j \le n \\
& B^x \ge 1 - \sum_{j:\mathbf{x}[j]\text{ positive}} I(\overline{x}_j) - \sum_{j:\mathbf{x}[j]\text{ negative}} I(x_j) \\
& \sum_j I(\neg \mathbf{y}[j]) = 0 \quad \forall \mathbf{y} \in E^+ \\
& \sum_j I(\neg \mathbf{y}[j]) \ge 1 \quad \forall \mathbf{y} \in E^- \\
& Y_j \le X_j w_j\!\uparrow; \quad Y_j \le w_j \quad \forall j \le n \\
& Z^a \le B^a w_b\!\uparrow; \quad Z^a \le w_b \\
& B^x w_b\!\downarrow \le Z^x; \quad B^x w_b\!\uparrow \le Z^x + w_b\!\uparrow - w_b \\
& (w_1, \cdots, w_n, w_b) \in W; \quad (X_1, \cdots, X_n) \in \mathbf{X}
\end{aligned}
$$

Comparison Queries in Joint Model

- User prefers ***x*** to ***y***
  - with no feature uncertainty: linear constraint ***wx*** > ***wy***
  - with feature uncertainty, more complicated
    - ask membership queries: linear; e.g., ***wx*** + *p* > ***wy***
    - *unknown* membership: conditional constraints

$$w\mathbf{x} - w\mathbf{y} > 0 \text{ if } c(\mathbf{x}), c(\mathbf{y})$$

$$w\mathbf{x} + p - w\mathbf{y} > 0 \text{ if } c(\mathbf{x}), \neg c(\mathbf{y}) \;*$$

$$w\mathbf{x} - w\mathbf{y} - p > 0 \text{ if } \neg c(\mathbf{x}), c(\mathbf{y})$$

$$w\mathbf{x} - w\mathbf{y} > 0 \text{ if } \neg c(\mathbf{x}), \neg c(\mathbf{y})$$

- linearized in MIP in a straightforward fashion

$$* \quad w\mathbf{x} + b - w\mathbf{y} > [\sum_{j \le n} I(\neg \mathbf{x}[j]) + (1 - I(\neg \mathbf{y}[k]))]\,\Delta_{\downarrow} \quad \forall k \le n$$

Query Strategies

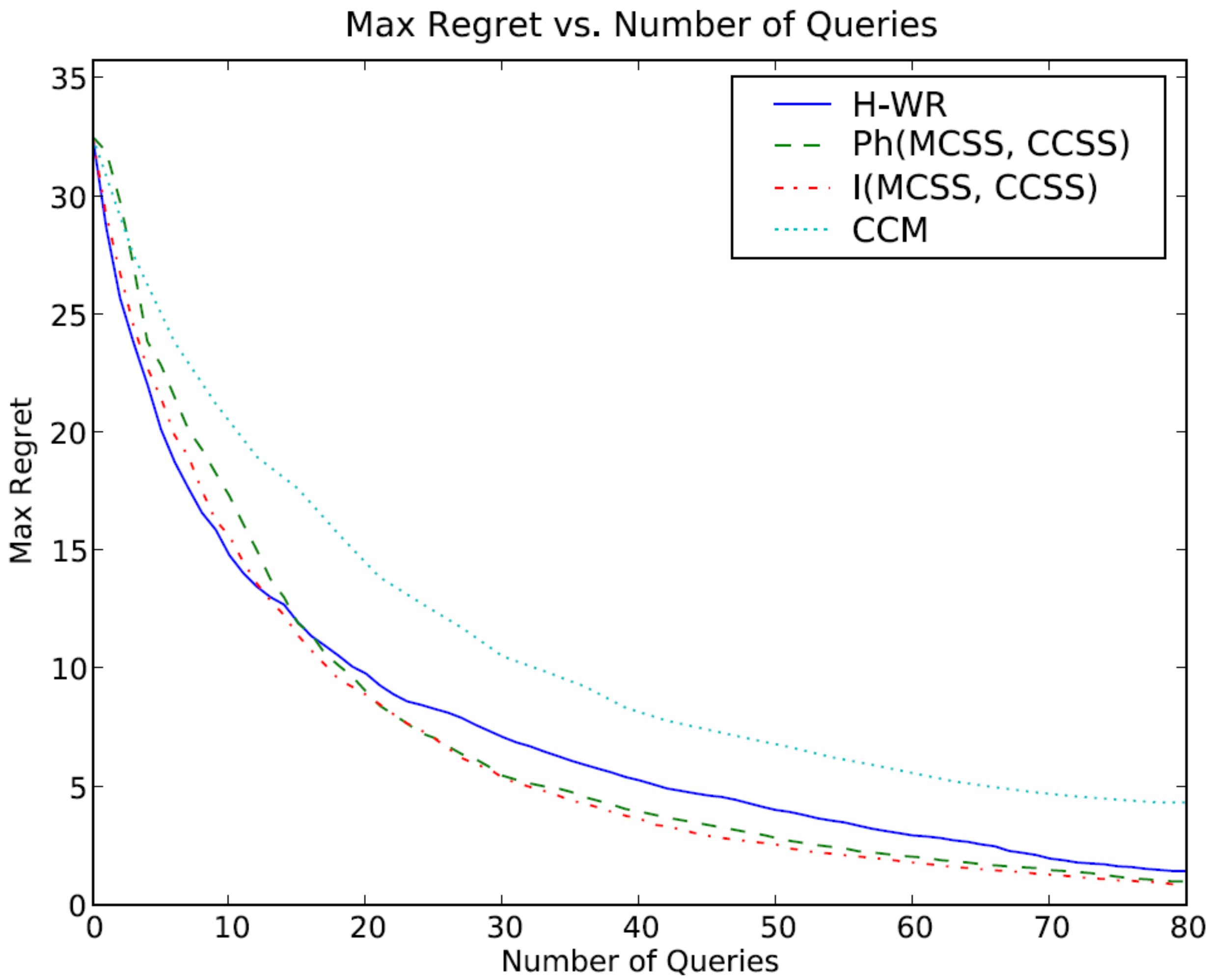
- Focus on comparison queries, membership queries
  - Membership: *halving*, *current soln (MCSS)*, defined as before
  - Comparison queries: use only current solution (CCSS)
- Key question: when to ask membership vs. comparison
  - which is more valuable: refining concept, refining utility polytope
  - decompose MR of current soln into *concept regret, utility regret*

$$rr = r(\mathbf{x}^a; w) - r(\mathbf{x}^*; w); \quad cr = w_b(c(\mathbf{x}^a) - c(\mathbf{x}^*))$$

- Five strategies explored:
  - *Phased*: Ph(Halving, CCSS) and Ph(MCSS, CCSS)
    - stalling: ask a comparison query
  - *Interleaved*: I(Halving, CCSS) and I(MCSS, CCSS)
    - query choice: whichever of concept or utility regret is greater
  - *Combined* comparison-membership queries: CCM
    - uses CCSS to generate comparison

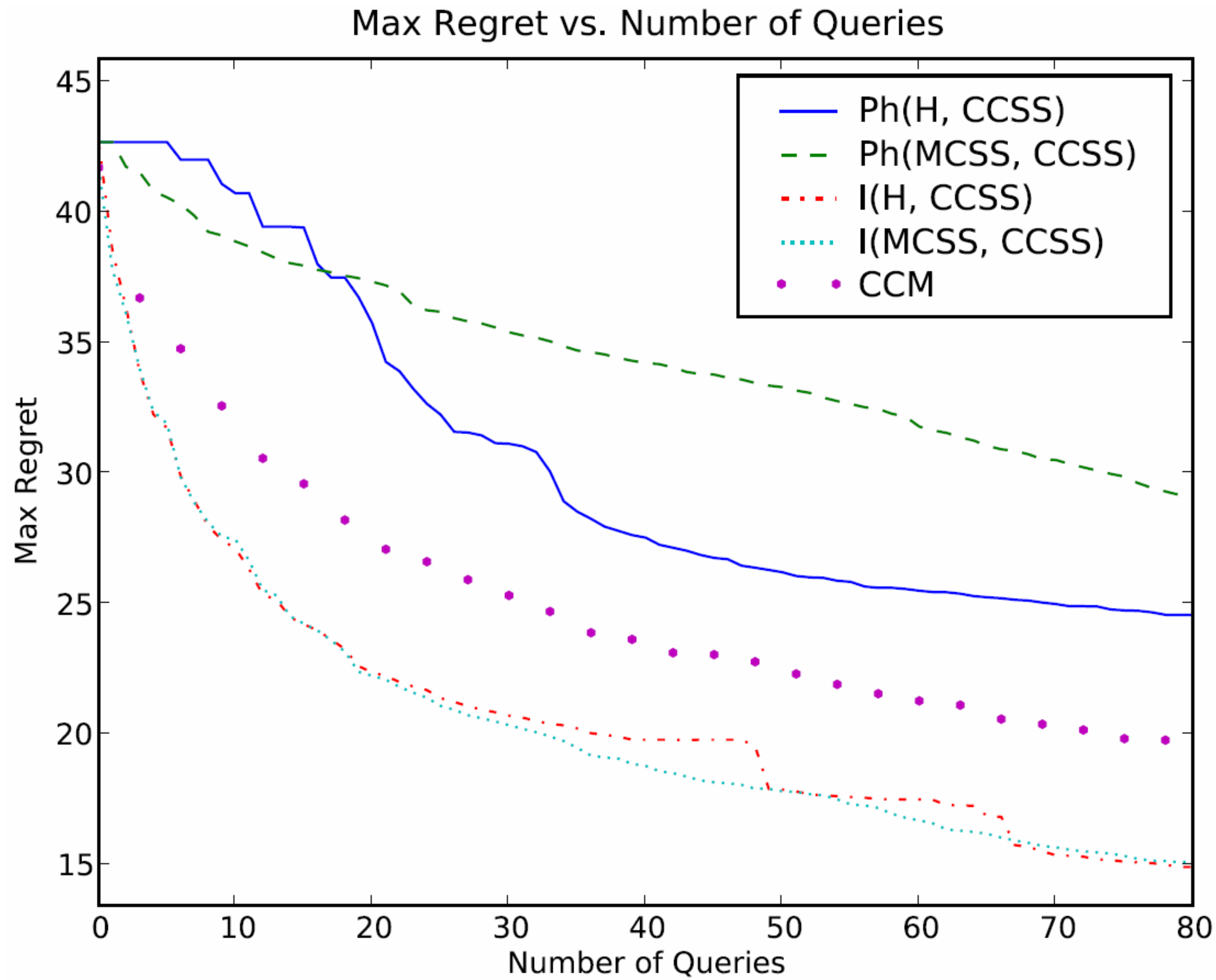
Empirical Results

- 20 variables, 60 random binary constraints, random concepts (5 vars max, 3.33 on avg), random reward/bonus and initial uncertain bounds (30runs)



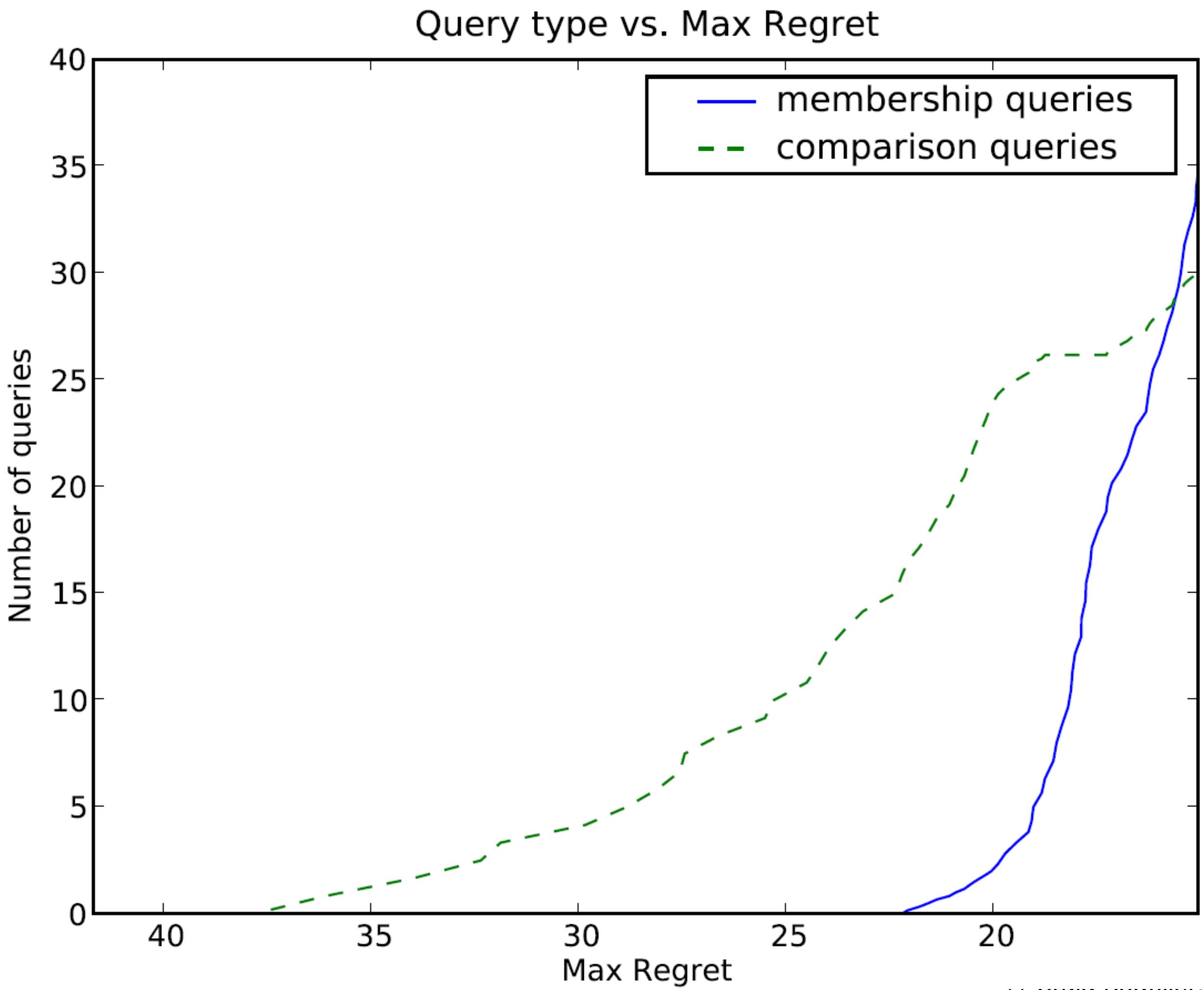
Empirical Results

- 30 variables, 90 random binary constraints, random concepts (10 vars max, 6.67 on avg), random reward/bonus and initial uncertain bounds (20runs)



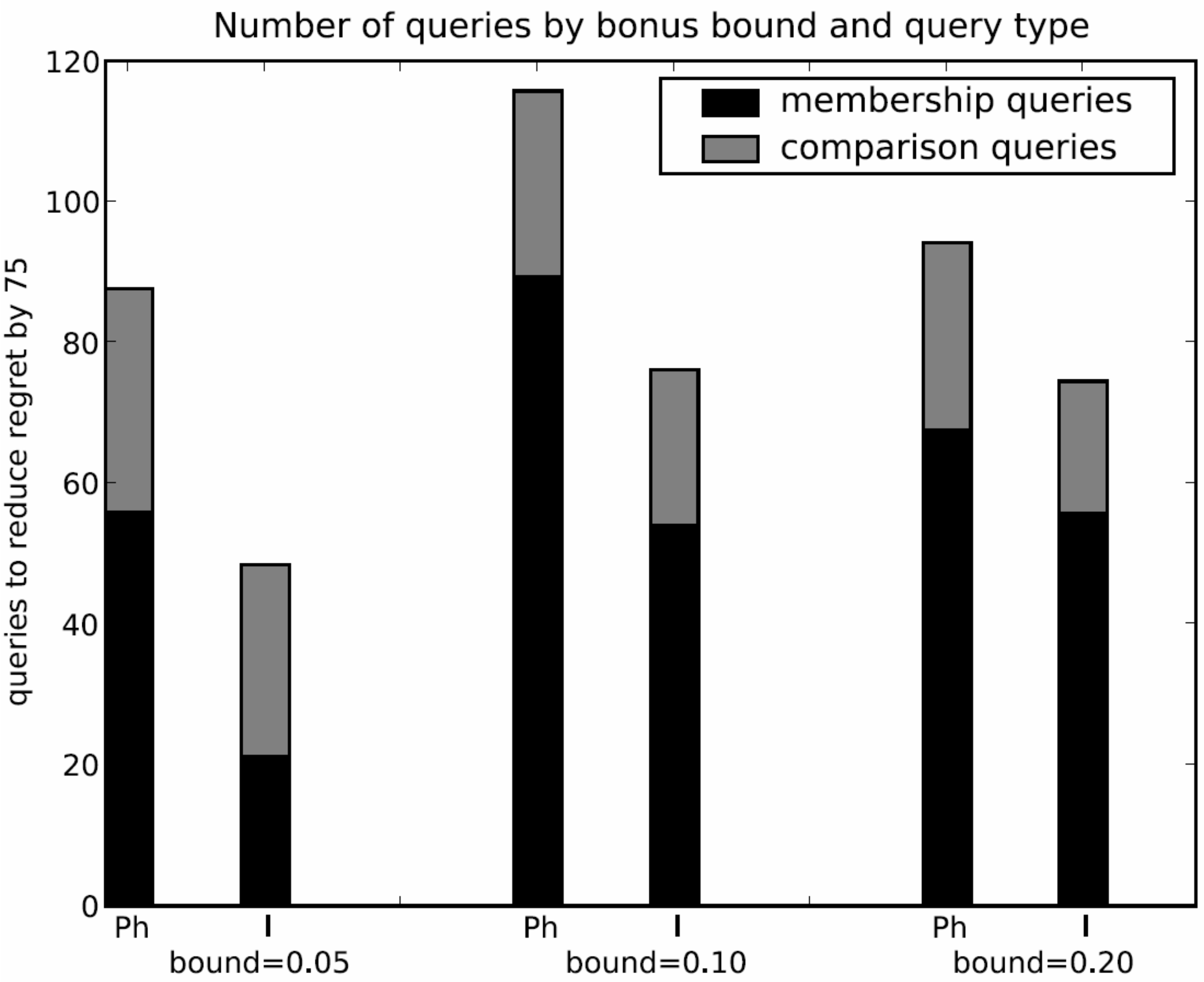
Query mix for interleaved I(MCSS, CCSS)*

- 30 variables, 90 random binary constraints, random concepts (10 vars max, 6.67 on avg), random reward/bonus and initial uncertain bounds (20runs)



Varying Relative Bonus*

- Greater bonus value: refining concept becomes more critical
  - shown: queries to reach regret reduction of 75% (20 variables, 60 constraints)



Constructed Features: Summary

- Formal view of constructed features
  - allows “on the fly” elicitation of “fundamental” objectives
- Elicitation of feature definitions
  - attention focused on relevant constraints on user definition
  - some first steps toward integrated feature and utility elicitation
- First steps only
  - more general hypothesis classes (including fuzzier concepts)
  - richer concept query classes (some more natural?)
  - better strategies for integration

Subjective Features

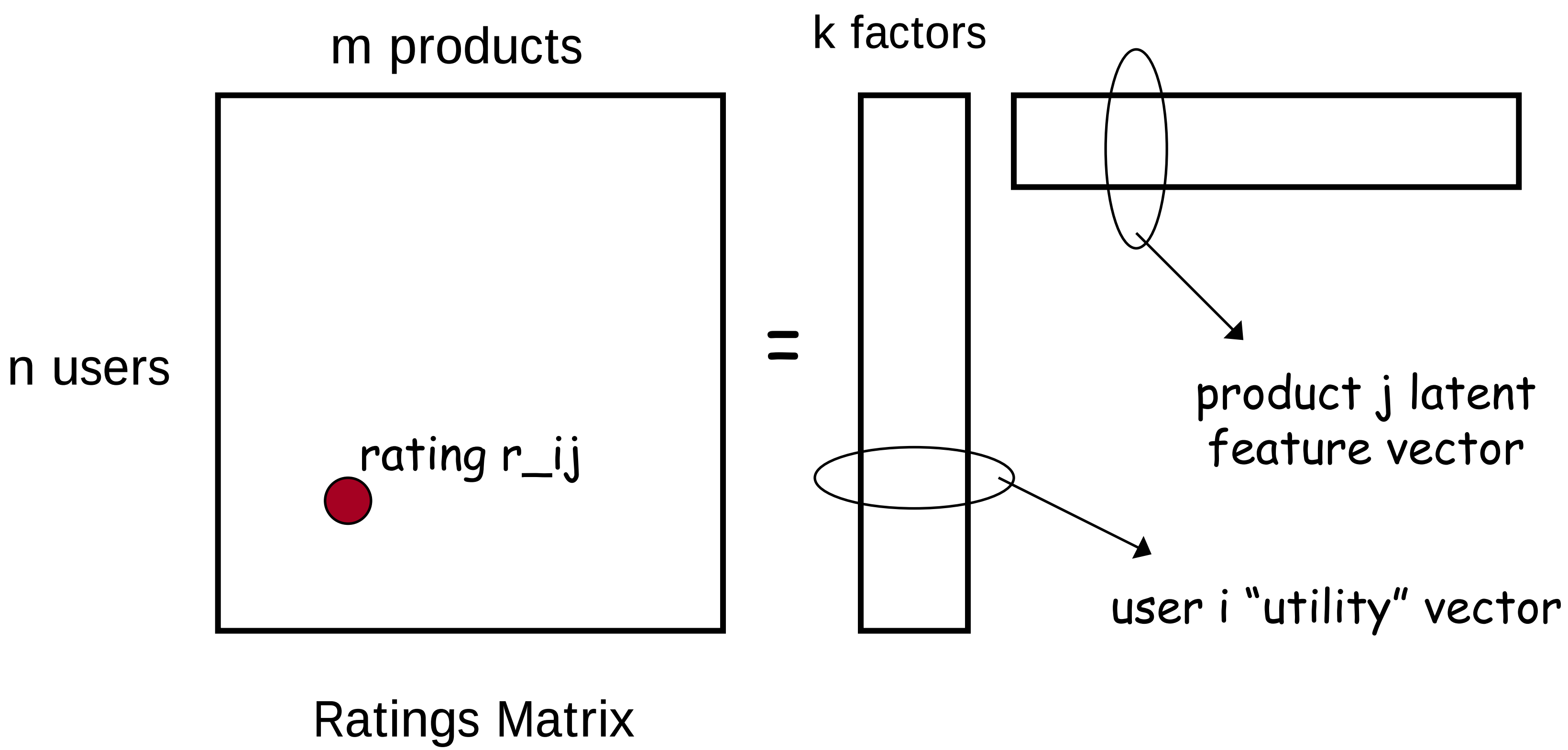
- Consider other subjective features (not constructed):
  - your assessment of "cute" car differs from my wife's
  - no "functional definition" from catalog features





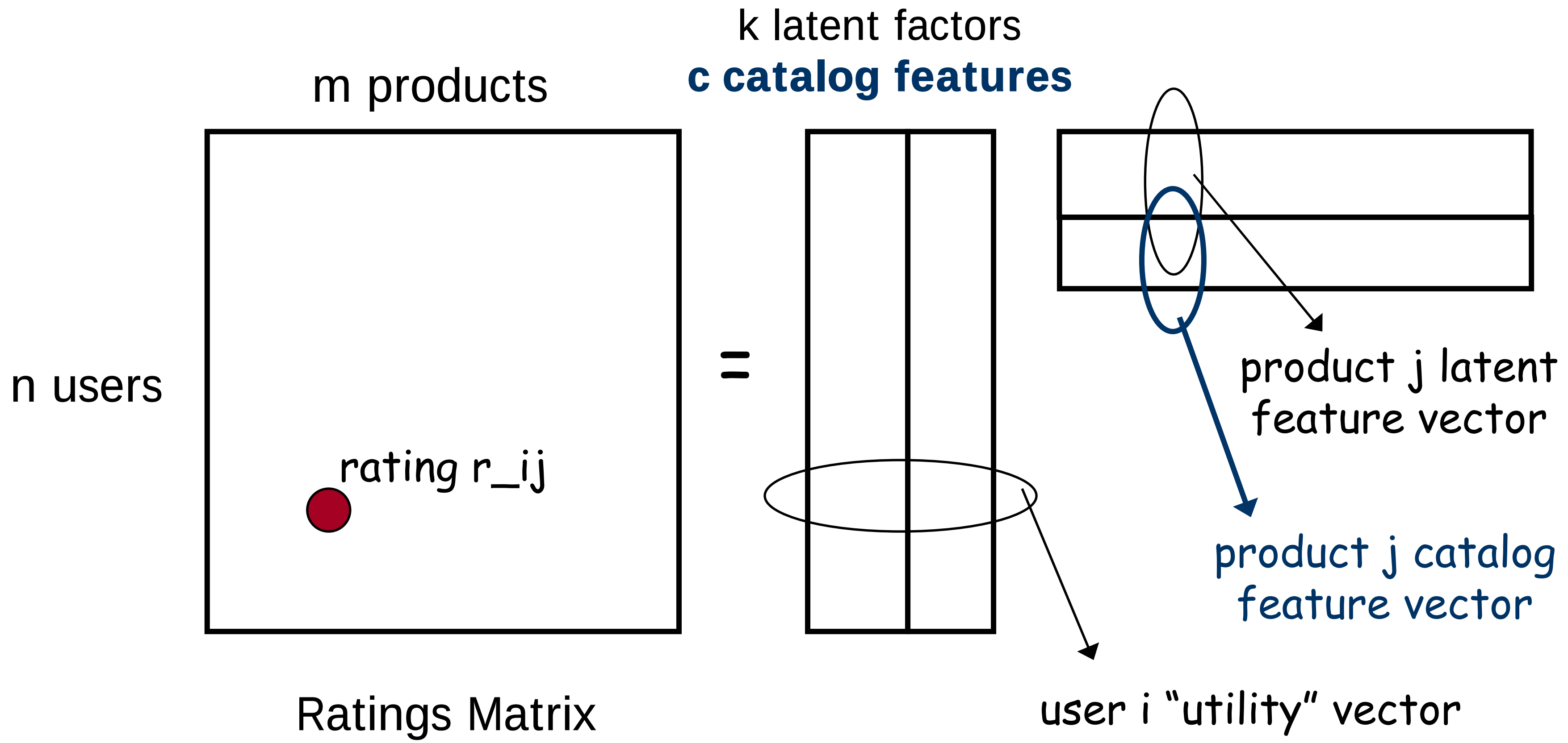
Collaborative Filtering (stylized)

- *Matrix factorization perspective*



- *Can add active component to improve ratings (BZM)*

CF: Catalog Features

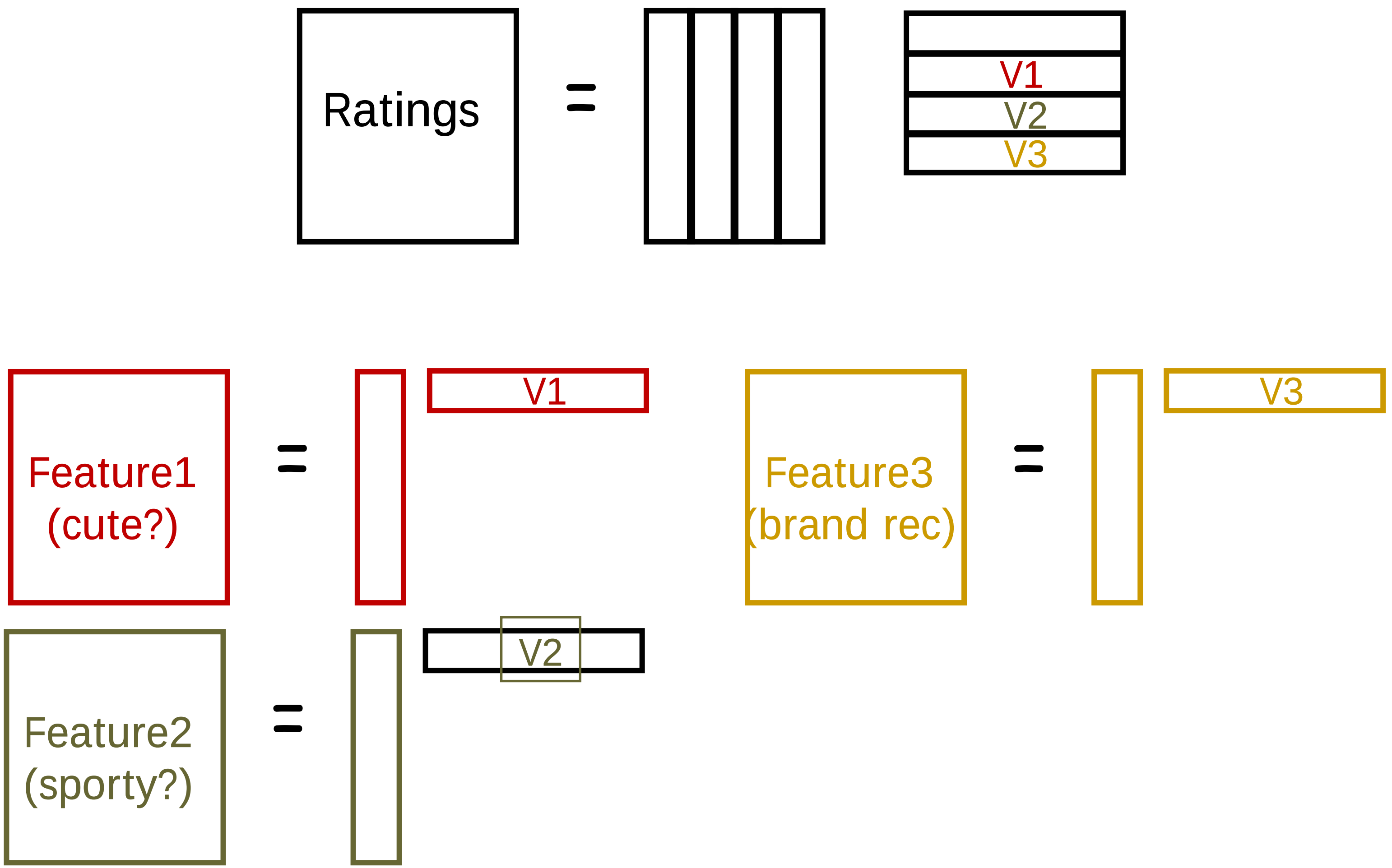


- Combining active CF, elicitation an interesting problem for such content-collaborative recommendation systems
  - must apply PMF with constraints on user utility vectors, full information on catalog product features

Subjective Features in CF

- Suppose subjective keywords for some items
  - e.g., various people label cars “cute”, “sporty”, …
    - see Dudek for movies with semantic labels
- Goal:
  - use subjective feature assessment to predict utility
  - problem: unobservable for novel items
  - solution: simultaneously predict your assessment for novel items as well
  - assume (incomplete) set of subjective assessments over the user-item population

Subjective Features [see also Singh&Gordon 08]



Subjective Features [w Charlin, Zemel]

- Leverage collaborative aspects to assess SFs
  - solved through iterative (componentwise optimization)
  - prelim results (synthetic data) encouraging
- Key questions:
  - learning, optimzt'n of with catalog features/constraints
  - active elicitation of ratings? subjective features?
    - *e.g., show picture of car: "You think this one's cute?"*
  - learning *visual features* (subjective/objective)
  - *sentiment analysis*: treat as objective features

Toward Conversational Recommenders

- Overall goal: make decision support/recommender systems more “accessible” to naïve users
- Several preliminary steps
  - constructed features (“on the fly” fundamental objectives)
  - collaborative models for subjective feature assessment
  - conversational/critiquing models using MMR
    - semantics of critiques, set recommendations/queries
- Other important directions
  - biases (framing, anchoring, thresholds, hyperbolic discounts, …)
    - overcoming, or quantifying/accounting for them
  - linguistic cues to strength of preference
  - query/interaction costs
  - mechanism design and social choice