Preference elicitation for interactive learning of Optimization Modulo Theory problems

Andrea Passerini

Dipartimento di Ingegneria e Scienza dell'Informazione
Università degli Studi di Trento, Italy

joint work with Paolo Campigotto and Roberto Battiti

A housing problem

I would like to build a house in a safe area, close to my parents and to the kindergarten, with a garden if there are no parks nearby. My maximum budget is 300,000 euro.

MAX-SMT formulation

Variable	Description
x_1	has garden
x_2	has park nearby
x_3	crime rate
x_4	distance from parents
x_5	distance from kindergarten

$\max_{\mathbf{x}}$ $\boxed{w_1 p_1 + w_2 p_2 + w_3 (p_3 \wedge p_4)}$ client utility

subj to:

client const

$$p_1 = (\neg x_2 \Rightarrow x_1)$$
$$p_2 = (x_3 \leq \theta_1)$$
$$p_3 = (x_4 \leq \theta_2)$$
$$p_4 = (x_5 \leq \theta_3)$$

$$price(\mathbf{x}) \leq 300000$$

$$x_4 \geq \theta_4$$
$$x_5 \geq \theta_5$$

company const

soft constraints

hard constraints

Satisfiability modulo theory

- Satisfiability problem over *predicates* rather than *Boolean variables*
- Predicates defined over *theories* of interest (e.g. FOL, linear arithmetic)
- Solution requires truth assignment of predicates + consistent value of predicate variables (e.g. integer or real variables..)
- MAX-SMT extends SMT as MAX-SAT extends SAT (very recent research trend)

Fine we are done! not quite..

I would like to build a house in a safe area, close to my parents and to the kindergarten, with a garden if there are no parks nearby. My maximum budget is 300,000 euro.

Who is capable of such a precise and exhaustive explanation?

Solving an unknown MAX-SMT problem

- exact problem formulation unknown (human DM)
- set of candidate *catalogue* features is available (the **variables**)
- set of candidate constraints over the features (combinations of **predicates**)
- True (unknown) utility is weighted sum of *few* constraints over *few* variables
- DM feedback as pairwise preferences btw solutions

Solving an unknown MAX-SMT problem

- exact problem formulation unknown (human DM)
- set of candidate *catalogue* features is available (the **variables**)
- set of candidate constraints over the features (combinations of **predicates**)
- True (unknown) utility is weighted sum of *few* constraints over *few* variables
- DM feedback as pairwise preferences btw solutions

Solving an unknown MAX-SMT problem

- exact problem formulation unknown (human DM)
- set of candidate *catalogue* features is available (the **variables**)
- set of candidate constraints over the features (combinations of **predicates**)
- True (unknown) utility is weighted sum of *few* constraints over *few* variables
- DM feedback as pairwise preferences btw solutions

Solving an unknown MAX-SMT problem

- exact problem formulation unknown (human DM)
- set of candidate *catalogue* features is available (the **variables**)
- set of candidate constraints over the features (combinations of **predicates**)
- True (unknown) utility is weighted sum of *few* constraints over *few* variables
- DM feedback as pairwise preferences btw solutions

Solving an unknown MAX-SMT problem

- exact problem formulation unknown (human DM)
- set of candidate *catalogue* features is available (the **variables**)
- set of candidate constraints over the features (combinations of **predicates**)
- True (unknown) utility is weighted sum of *few* constraints over *few* variables
- DM feedback as pairwise preferences btw solutions

Problem formulation

Catalogue features

- Set of features characterizing candidate solutions
- E.g. for housing problem

feature	Description	type
x_1	house type	ord
x_2	garden	Bool
x_3	garage	Bool
x_4	commercial facilities nearby	Bool
x_5	public green areas nearby	Bool
x_6	distance from downtown	num
x_7	crime rate	num
x_9	public transit service quality index	num
x_{10}	distance from parents house	num
....	...	...

Problem formulation

Possible predicates

- All predicates constructible from catalogue features
- E.g. for housing problem

predicate	Description	formula
p_1	has garden	x_2
p_2	has garage	x_3
p_3	has park nearby	x_5
p_4	close to downtown	$x_6 < \theta_1$
p_5	low crime rate area	$x_7 < \theta_2$
p_6	high quality transit service	$x_8 > \theta_3$
....	...	...

Problem formulation

Possible constraints

- All constraints constructible from predicates
- E.g. for housing problem

predicate	Description	formula
c_1	has garden	p_1
c_2	garden if no park nearby	$\neg p_3 \rightarrow p_1$
c_3	good transportation if far from downtown	$\neg p_4 \rightarrow p_6$
c_4	garage if high crime rate	$\neg p_5 \rightarrow p_2$
....	...	...

Problem formulation

Feasible set

- The set of feasible solutions is defined by known hard constraints
- E.g. for housing problem

hard constraint
price $\leq \theta_1$
location-based taxes and fees $\leq \theta_2 =>$
not public green areas nearby
crime rate $\leq \theta_4 =>$downtown distance $\geq \theta_5$
working place distance + parents house distance $\geq \theta_6$
working place distance + high schools distance $\geq \theta_7$
parents house distance + high schools distance $\geq \theta_8$
garden $=>$ house type $\geq \theta_{13}$

Problem formulation

User utility

- User utility is linear combination of (some) constraints
- We use polynomial mapping $\phi(\mathbf{p})$ from predicates to all products (i.e. conjunctions) of up to d predicates
- User utility formalized as:

$$f(\mathbf{p}) = \mathbf{w}^T \phi(\mathbf{p})$$

Note

- True (unknown) f extremely **sparse** (most weights should be zero)

Learning utility

Ranking task

- User feedback comes as ranking of candidate solutions
- Need feature sparsity: *1-norm regularization*
- Dataset $\mathcal{D}$ of pairwise comparisons $(\mathbf{p}_i, \mathbf{p}_j)$ ($\mathbf{p}_i$ better than $\mathbf{p}_j$)
- Support vector ranking with 1-norm regularization:

$$\min_{\mathbf{w}} \sum_{(\mathbf{p}_i,\mathbf{p}_j):\mathbf{p}_i \prec \mathbf{p}_j} \left[1 - \mathbf{w}^T \left(\Phi(\mathbf{p}_i) - \Phi(\mathbf{p}_j)\right)\right]_+ + \lambda||\mathbf{w}||_1$$

Optimizing utility

MAX-SMT problem

- A given utility f (+ hard constraints) defines a MAX-SMT problem
- Optimizing utility consists of solving the MAX-SMT problem
- Diversity should be injected to avoid premature convergence (utility is learned..)
  - E.g. re-optimize adding diversification constraints

Preference elicitation algorithm

Algorithm

- Select s random configurations
- Ask the DM for their ranking and initialize $\mathcal{D}$
- **while** (termination_criterion)
  - Learn ranking function as:

$$\min_{\mathbf{w}} \sum_{(\mathbf{p}_i,\mathbf{p}_j):\mathbf{p}_i \prec \mathbf{p}_j} \left[1 - \mathbf{w}^T\left(\Phi(\mathbf{p}_i) - \Phi(\mathbf{p}_j)\right)\right]_+ + \lambda||\mathbf{w}||_1$$

  - Optimize learned MAX-SMT and get $s/2$ new configs
  - Ask the DM for their ranking and update $\mathcal{D}$
- **return** configuration optimizing the final MAX-SMT

Preference elicitation algorithm

Algorithm

- Select s random configurations
- Ask the DM for their ranking and initialize $\mathcal{D}$
- **while** (termination_criterion)
  - Learn ranking function as:

$$\min_{\mathbf{w}} \sum_{(\mathbf{p}_i,\mathbf{p}_j):\mathbf{p}_i \prec \mathbf{p}_j} \left[1 - \mathbf{w}^T\left(\Phi(\mathbf{p}_i) - \Phi(\mathbf{p}_j)\right)\right]_+ + \lambda ||\mathbf{w}||_1$$

  - Optimize learned MAX-SMT and get $s/2$ new configs
  - Ask the DM for their ranking and update $\mathcal{D}$
- **return** configuration optimizing the final MAX-SMT

Preference elicitation algorithm

Algorithm

- Select s random configurations
- Ask the DM for their ranking and initialize $\mathcal{D}$
- **while** (termination_criterion)
  - Learn ranking function as:

$$\min_{\mathbf{w}} \sum_{(\mathbf{p}_i,\mathbf{p}_j):\mathbf{p}_i \prec \mathbf{p}_j} \left[1 - \mathbf{w}^T \left(\Phi(\mathbf{p}_i) - \Phi(\mathbf{p}_j)\right)\right]_+ + \lambda ||\mathbf{w}||_1$$

  - Optimize learned MAX-SMT and get $s/2$ new configs
  - Ask the DM for their ranking and update $\mathcal{D}$
- **return** configuration optimizing the final MAX-SMT

Preference elicitation algorithm

Algorithm

- Select s random configurations
- Ask the DM for their ranking and initialize $\mathcal{D}$
- **while** (termination_criterion)
  - Learn ranking function as:

$$\min_{\mathbf{w}} \sum_{(\mathbf{p}_i,\mathbf{p}_j):\mathbf{p}_i \prec \mathbf{p}_j} \left[1 - \mathbf{w}^T \left(\Phi(\mathbf{p}_i) - \Phi(\mathbf{p}_j)\right)\right]_+ + \lambda ||\mathbf{w}||_1$$

  - Optimize learned MAX-SMT and get $s/2$ new configs
  - Ask the DM for their ranking and update $\mathcal{D}$
- **return** configuration optimizing the final MAX-SMT

Preference elicitation algorithm

Algorithm

- Select s random configurations
- Ask the DM for their ranking and initialize $\mathcal{D}$
- **while** (termination_criterion)
  - Learn ranking function as:

$$\min_{\mathbf{w}} \sum_{(\mathbf{p}_i,\mathbf{p}_j):\mathbf{p}_i \prec \mathbf{p}_j} \left[1 - \mathbf{w}^T \left(\Phi(\mathbf{p}_i) - \Phi(\mathbf{p}_j)\right)\right]_+ + \lambda ||\mathbf{w}||_1$$

  - Optimize learned MAX-SMT and get $s/2$ new configs
  - Ask the DM for their ranking and update $\mathcal{D}$
- **return** configuration optimizing the final MAX-SMT

Preference elicitation algorithm

Algorithm

- Select s random configurations
- Ask the DM for their ranking and initialize $\mathcal{D}$
- **while** (termination_criterion)
  - Learn ranking function as:

$$\min_{\mathbf{w}} \sum_{(\mathbf{p}_i, \mathbf{p}_j): \mathbf{p}_i \prec \mathbf{p}_j} \left[1 - \mathbf{w}^T \left(\Phi(\mathbf{p}_i) - \Phi(\mathbf{p}_j)\right)\right]_+ + \lambda ||\mathbf{w}||_1$$

  - Optimize learned MAX-SMT and get $s/2$ new configs
  - Ask the DM for their ranking and update $\mathcal{D}$
- **return** configuration optimizing the final MAX-SMT

Preference elicitation algorithm

Algorithm

- Select s random configurations
- Ask the DM for their ranking and initialize $\mathcal{D}$
- **while** (termination_criterion)
  - Learn ranking function as:

$$\min_{\mathbf{w}} \sum_{(\mathbf{p}_i,\mathbf{p}_j):\mathbf{p}_i \prec \mathbf{p}_j} \left[1 - \mathbf{w}^T\left(\Phi(\mathbf{p}_i) - \Phi(\mathbf{p}_j)\right)\right]_+ + \lambda||\mathbf{w}||_1$$

  - Optimize learned MAX-SMT and get $s/2$ new configs
  - Ask the DM for their ranking and update $\mathcal{D}$
- **return** configuration optimizing the final MAX-SMT

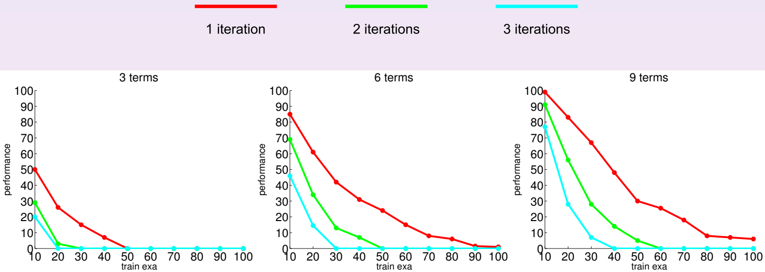
Experiments

Housing problem

- 15 features, 40 predicates, max term size 3, term weights in [1, 100].
- inaccurate feedback: ($p = 0.1$ of incorrect ranking)
- Evaluate by approximation error w.r.t. gold solution (found optimizing true utility)
- Experiments for growing number of terms (medians over 400 runs)

Experiments

Results



Pros

- Fewest knowledge assumed on utility
- Simple feedback required to DM
- Handling of inaccurate feedback
- Large class of problems can be modelled

Cons

- lack of optimality guarantees
- small-scale problems (explicit mapping):
  - OK for human DM
  - can use implicit mapping (kernels) but :
    - lower results (no feature sparsity)
    - problems with tight SAT - Theory solvers integration
- simple preference elicitation strategy
  - can try Bayesian approaches but need to retain feature sparsity