Brain-machine optimization: learning objective functions by interacting with the final user

Roberto Battiti and Paolo Campigotto

DISI – Dipartimento di Ingegneria e Scienza dell'Informazione
University of Trento (Italy)

ECCO XXII, May 17-19, 2009 Jerusalem, Israel
Topics: (machine) learning and optimization

Reactive Search Optimization
“learning on the job”
– avoiding user intervention
– internal learning loop

… suitable if single clear $f(x)$ to optimize
but in many real-world cases… no $f(x)$

Multiobjective Optimization
interactive: “user in the learning/optimizing loop”
– couple internal and external learning through systematic machine learning techniques
Reactive search optimization

= 

ON-LINE MACHINE LEARNING FOR OPTIMIZATION 

= 

on-line dynamic and adaptive search 

= 

on-line reinforcement learning for optimization

Reactive Search: Learning on the Job
Operations research (optimization)

Computer Science

Machine learning and neural nets

RSO

(c) R. Battiti

Reactive Search
The role of the user
The role of the user

- choices and free parameters
- the user as a crucial learning component ("trial and error")

Parameter tuning is a typical “learning” process where experiments are designed, with the support of statistical estimation (parameter identification) tools.
Automated tuning through machine learning

- **Automation.** The time-consuming tuning phase is now substituted by an automated process.

- **Complete and unambiguous documentation.** The algorithm becomes self-contained: its quality can be judged independently from the designer.

Complexity is shifted
Final user → algorithm developer
On-line tuning

➢ Take into account:
  - Problem-dependent
  - Task-dependent
  - Local properties in configuration space (see local search), parameters are dynamically tuned based on optimization state and previous history
integration of sub-symbolic machine learning techniques into search heuristics. The word *reactive* hints at a ready response to events *during* the search through an internal online feedback loop for the *self-tuning* of critical parameters.

Methodologies of interest for Reactive Search include machine learning and statistics, in particular reinforcement learning, active or query learning, transfer learning, neural networks.

[www.reactive-search.org](http://www.reactive-search.org)
RSO context

$f(x)$ is given (either analytically or as a black box)

the emphasis is on learning local models and using them while optimizing (*no additional knowledge from DM required*)

... but in some cases $f(x)$ to optimize is not given, modeling user preferences is a crucial issue

Try asking a decision maker: “give me the $f(x)$ that you are optimizing”
Moltiobjective optimization

intermediate (classical) case:
some criteria are given \( f_1(x) f_2(x) \ldots f_k(x) \)
but are not easily combined into a single \( f(x) \)

…provide efficient vector solutions \( (f_1,\ldots,f_k) \)
leave to the user the possibility to decide
(and to learn about possibilities and “real” objectives, even if not formalized)
Efficient frontier: an example

Pareto Front

no other feasible solution is strictly better in one objective and at least as good for the other ones

Objective space

Image of Feasible Region in the Objective Space
Preference information

- Critical task: identify the best solution for the DM from the efficient frontier

- Based on the DM preference information usage:
  - A priori MOO methods
  - A posteriori MOO methods
  - Interactive MOO methods (IM)
A priori methods

- Assumptions about preference information before optimization process
- DM specifies preference on the objectives a priori
- Drawbacks:
  - Very difficult task for DM
  - DM often does not know before how realistic his expectations are (no learning possibilities)
**A posteriori Methods**

- The Pareto optimal set (or part of it) is generated and presented to the DM who selects the most preferred among the alternatives.

- **Drawbacks:**
  - Generation is computationally expensive: find all the non dominated solutions!
  - Hard for the DM the selection among a large set of alternatives
  - Presenting / displaying the alternatives to the DM
**Interactive methods**

- Solutions generation phases alternated to solution evaluation phases requiring user interaction

- Effective approach
  - Only a subset of the Pareto optimal set has to be generated and evaluated by the DM
  - The DM drives the search process
  - The DM gets to know the problem better (*learning on the job*)
Interactive methods

- How information is provided to DM
- How preference information is obtained from DM
- How the search process is updated based on the preference information
- How the original MOO problem is transformed into a single-objective optimization problem (scalarization process)

e.g. optimize: $\sum w_i f_i(x)$
Interactive methods

- Assumptions:
  - DM states the decision variables, the constraints, the model, the objectives but he cannot prioritize the importance of the objectives
  - Model parameter vector (MPV): parameter vector in the MOO method capturing the preference information (e.g. weights $w_i$)
    - MPV defined => final solution determined
  - Value function: DM makes decisions based on an underlying (unknown) function representing the preference of the DM among the criterion vectors
Notation

- Input MOO problem
  \[
  \text{minimize} \quad \{f_1(x), f_2(x), \ldots, f_k(x)\}
  \]
  \[
  \text{subject to} \quad x \in S,
  \]

- K objective functions:
  \[
  f_i : \mathbb{R}^n \rightarrow \mathbb{R}
  \]

- Decision variables vectors:
  \[(x_1, x_2, \ldots, x_n)\]

- Feasible region S

- Criterion values:
  \[z_i = f_i(x) \text{ for all } i = 1, \ldots, k\]

- Image of the feasible region:
  \[Z (= f(S))\]
Notation

- Lower bound: ideal criterion vector $z^* \in \mathbb{R}^k$
  - Obtained by minimizing each objective individually
- Upper bound: Nadir criterion vector $z^{nad}$
  - (estimated) worst objective values of the Pareto Optimal solutions
Interactive methods

- Possible classification:
  - Trade-off based methods
  - Reference point methods
  - Classification based methods

- Neural network based approaches
Neural Network based approaches

- Key issue: modelling the preference information of the DM by using a Neural Network (NN)
- NN based model used to predict how much the DM appreciate a given (Pareto optimal) solution
- Two methods in the literature:
  - FFANN – Feed Forward Artificial Neural Networks procedure (Sun et al. 1996, 2000)
  - IMOM - Intelligent Interactive Multiobjective Optimization method (Huang et al. 2005)
FFANN procedure

- Iterative procedure
- NN model of the preference information structure
  - Input: criterion vector
  - Output: preference value

1. Use trained NN to determine best NN input
   \[ \max \text{NN}(f(x)) \] [not nec. Pareto-optimal]

2. Use trained NN to filter Pareto-optimal points to present to DM for evaluation
FFANN-2 procedure  (Sun et al. 2000)

- At each iteration:
  - Step 1: randomly select $2P$ values in the current model parameter vector space generate $2P$ non-dominated criterion vectors
  - Step 2: compute the preference values for the $2P$ criterion vectors using the trained NN and select the $P$ vectors with highest output values
  - Step 3: present the selected $P$ vectors to DM for evaluation. The criterion vector $z_b$ with highest value is the best solution at current iteration. If FU is satisfied, stop
FFANN-2 procedure  (Sun et al. 2000)

- Step 4: train the NN with the P criterion vectors as input and the DM evaluation as the desired output
- Step 5: based on $z_b$ update the model parameter vector space

- NN model reduces the burden on the DM of evaluating the generated solutions
- Drawbacks: NN model cannot help in searching improved solutions
Intelligent Interactive MO Method
IMOM (Huang et al. 2005)

- Iterative procedure
- NN model of the preference information structure
  - Input: model parameter vector
  - Output: preference value
- At each iteration reduce the model parameter vector space based on DM preference information
IMOM (Huang et al. 2005)

- At each iteration:
  - Step 1: pick the best value for the model parameter vector \( mpv \) obtained at previous iteration, randomly select \( P-1 \) values in the current model parameter vector space \( \Omega \) and generate \( P \) pareto optimal solutions
  - Step 2: If the designer is satisfied with one on the \( P \) solutions, stop. Otherwise, DM evaluates the \( P \) criterion vectors
IMOM (Huang et al. 2005)

- Step 3: **train the NN model** with the preference information of the DM obtained in the last several iterations

- Step 4: Obtain the best $mpv$ w.r.t. the highest preference value by solving
  \[
  \max_{mpv} \text{NN}(mpv) \\
  \text{s.t. } mpv \in \Omega
  \]

- Step 5: modify and reduce the model parameter vector space based on the best $mpv$ obtained at step 4
IMOM (Huang et al. 2005)

Initialization

- Modify and reduce the model parameter vector space
- Solve an optimization problem and obtain the weight vector corresponding to the highest preference value
- Build the ANN model of the designer’s preference structure
- Determine the preference value of each generated Pareto solution

Generate randomly $K$ weight vectors in the model parameter vector space

Calculate the Pareto solution corresponding to each of the $K$ weight vectors

The designer is satisfied with one of the generated Pareto solutions?

- Output the Pareto solution the designer is satisfied with
- End
IMOM (Huang et al. 2005)
Current work: learning for multiobjective optimization

Context is the same: learning user preferences

1. Train a predictor able to reproduce user preferences
2. Use the learned predictor to guide the search in place of the user

Emphasis is different:

DM time is a scarce and costly resource. It is crucial to minimize the number of queries made to the user and their complexity.
Current work: learning for multiobjective optimization

Design:

- Question structure (comparison, qualitative evaluation,...)
- Way in which the predictor is updated based on the preference information obtained in an online manner
- Detect the most informative questions (active learning)
support vector ranking

Algorithm

- Given:
  - A set of candidate solutions \( \{f(x_1), \ldots, f(x_N)\} \)
  - A set of user preferences as \( f(x_i) > f(x_j) \)
- Learn a large margin ranking function \( w^T f(x) \):
  \[
  \min_w \|w\|^2
  \]
- Satisfying constraints:
  \[
  w^T f(x_i) > w^T f(x_j)
  \]

Extensions

- Soft constraints can be employed adding penalties for violated ones
- A complex non-linear ranker can be learned by the kernel trick
Online support vector ranking

<table>
<thead>
<tr>
<th>Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The ranker is trained with an initial set of user preferences</td>
</tr>
<tr>
<td>2. Ranker guided search and ranker refinement steps are alternated:</td>
</tr>
<tr>
<td>1. the ranker guides the search emulating user preferences</td>
</tr>
<tr>
<td>2. produced solutions are ranked by the user, and the ranker is refined adding the new constraints</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Transfer learning</th>
</tr>
</thead>
<tbody>
<tr>
<td>The learned ranker could be used for related multiobjective optimization tasks, with same objectives but different feasible sets</td>
</tr>
</tbody>
</table>
PRELIMINARY CALL

Learning and Intelligent OptimizationN
LION 4, Jan 18-22, 2010, Venice, Italy

Technical Co-Sponsorship (pending): IEEE Computational Intelligence Society, Microsoft Research, Associazione Italiana per l'Intelligenza Artificiale

Microsoft Research

Industrial sponsorship (pending)

Proceedings published by:

THE END