Multiobjectove Grammatically-based Genetic Programming

Author: Ing. Jiří Petrlík
TID 2011

Contents

Multiobjective optimization problem

- To choose a suitable solution we should consider more than one objective.
- In the case of digital circuits design we should consider delay, prize, transistor count, etc. Some of them are opposite.
- Usually there is no the best solution. We must choose compromise.

Multiobjective optimization problem

- Formally multiobjective problem is a vector function f which maps a vector of m parameters to a vector of n objectives.

$$\begin{aligned}
&\text{min/max} \quad y = f(x) = (f_1(x), f_2(x), ..., f_n(x)) \\
&\text{subject to } x = (x_1, x_2, ..., x_m) \in X \\
&\qquad\qquad\quad\ \ y = (y_1, y_2, ..., y_n) \in Y
\end{aligned}$$

- X is the parameter space
- Y is the objective space [2]

Fitness assignment strategies

1. Aggregation methods
    - Transform results of multiple objective functions into one scalar function.
    - For example weighted sum approach.
    - To use these methods we need domain knowledge.
    - They don't provide family of solutions.
2. Pareto-based fitness assignment
    - uses Pareto dominance

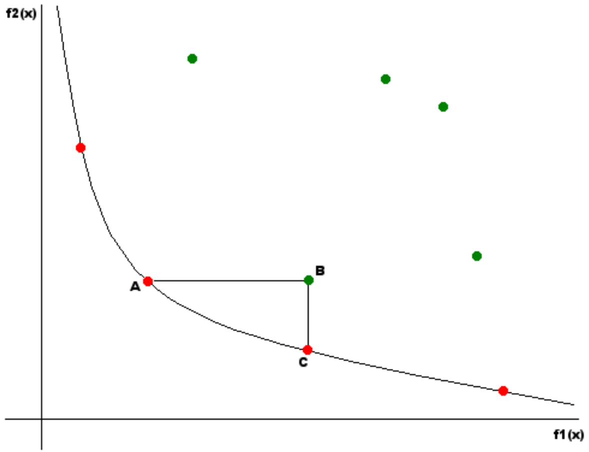
Pareto dominance

- Intuition: Solution a is better than solution b if and only if a is better or the same quality in all objectives and better in at least one objective.
- For minimization problem a dominate b ($a \succ b$), if and only if
  - $\forall i \in \{1, 2, ..., n\} : f_i(a) \leq f_i(b) \wedge$
  - $\exists j \in \{1, 2, ..., n\} : f_j(a) < f_j(b)$ [2]
- Solution a covers solution b ($a \succeq b$), if and only if
  - $a \succ b \vee$
  - $f(a) = f(b)$ [2]

Pareto-optimal front

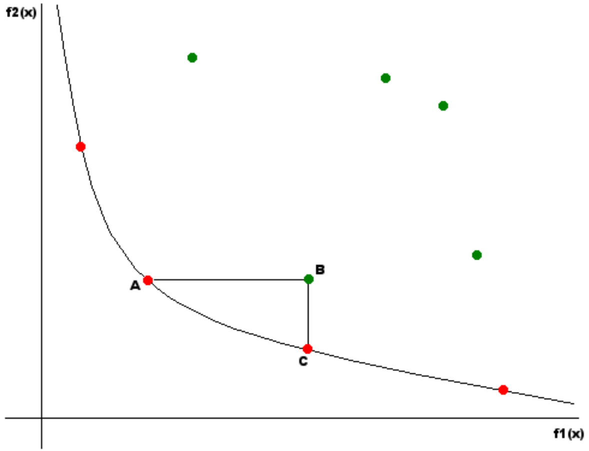
- Solution $a \in X$ is Pareto optimal if and only if there is no other solution $b \in X$ which dominates a in the search space. [2]
- Pareto-optimal front is a set of Pareto optimal solutions. [2]
- We usually want to find solutions on the Pareto front, or near the Pareto front.

Pareto-optimal front



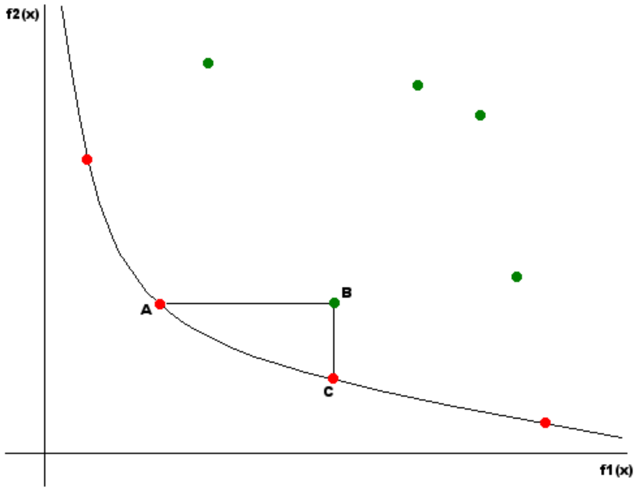
- $A \succ C$? No because $f_2(A) > f_2(C)$ breaks the first condition.
- $C \succ A$? No because $f_1(C) > f_1(A)$ breaks the first condition.

Pareto-optimal front



- $A \succ B$? Yes, because:
  - $f_1(A) \leq f_1(B)$ and $f_2(A) \leq f_2(B)$ (condition 1. is passed)
  - $f_1(A) < f_1(B)$ (condition 2. is passed)

Pareto-optimal front



- $A \succ A$? No because $f_1(A) \nless f_1(a)$ and $f_2(A) \nless f_2(A)$ breaks the second condition.

Genetic algorithm

1. Randomly generate set of solutions (first population P_0).
2. Evaluate quality of candidate solutions.
3. If termination condition was passed, then finish. (max. iteration count, sufficient solution found)
4. Reproduction phase (operators crossover and mutation).
5. Choose solutions into the new population (P_{t+1}).

Genetic algorithm - three main difficulties

1. How to represent solutions.
    - Binari vector, vector of integers, etc.
    - Directed graph, **tree**.
2. How to evaluate quality of solution.
    - single objective problems $\times$ multiobjective problems
3. How to implement operators crosover and mutation.

How to represent solutions

1. Production rule sequence encoding
   - Linear genome - typically vector of integers, or binary string.
   - It's necessary to use mapping before evaluating quality of solution.
2. Solution encoding individual
   - Tree representation. It will be shown in next slides.

Solution-encoding individual

- We must define context free grammar (N,T,P,S).
- Solutions are represented as trees in which non-leaf vertices are nonterminal symbols and leaf vertices are terminal symbols.

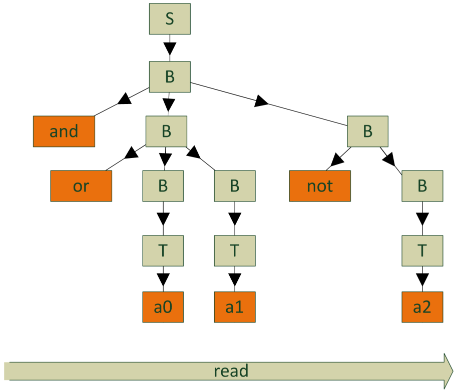
Example of simple program [1]

a_0	a_1	a_2	out
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

- Context free grammar:
- $N = \{S, B, T\}$
- $T = \{and, or, not, a_0, a_1, a_2\}$

$$\begin{aligned} S &\rightarrow B \\ B &\rightarrow and\ B\ B \mid or\ B\ B \mid not\ B \mid T \\ T &\rightarrow a_0 \mid a_1 \mid a_2 \end{aligned}$$

Example of simple program [1]



$$and(or(a_0, a_1), not(a_2))$$

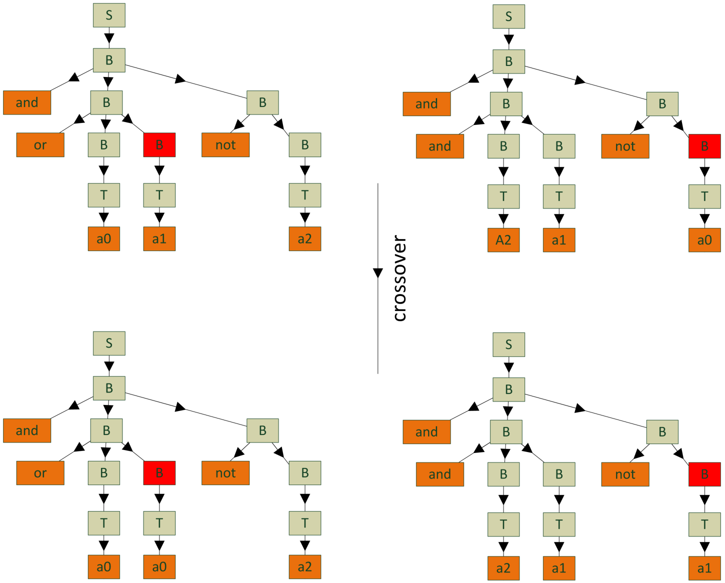
Creating the initial population

- Let p: $A \rightarrow \alpha$ is a production, l_p is a number of minimum derivation steps to create string of terminals ($A \Rightarrow \alpha \Rightarrow^{l_p} \beta$ where $\beta \in T^*$).
- To generete P_0 call procedure Generate(S,D).
- D is the maximal depth of tree.
- Generate(A,D)
    1. Randomly select production $p : A \rightarrow \alpha$ with $l_p < D$.
    2. Connect each symbols from α to A
    3. For each nonterminal $B_j \in \alpha$ call Generate(B_j,$D - 1$).

Crossover

1. Select two programs ρ_1 and ρ_2.
2. Randomly select one non-terminal $A \in \rho_1$.
3. If $A \notin \rho_2$ go back to step 2.
4. Randomly select $A \in \rho_2$.
5. Swap subtree under non-terminal A from ρ_1 with subtree under non-terminal A from ρ_2. [1]

Crossover



S
B
and
B
B
or
B
B
not
B
T
T
T
a0
a1
a2
and
and
B
B
not
B
A2
a1
a0
crossover
a0
a0
a2
a2
a1
a1

Mutation

- Randomly select one progam ρ.
- Randomly select one non-leaf vertex $A \in \rho$.
- Delete subtree under the chosen vertex.
- For vertex call procedure Generate(A,D) and connect new tree under the chosen vertex.

Advantages of grammar based genetic programming

- Can incorporate more knowledge about the problem into algorithm.
- Enables to use data types.
- Has wide range of applications (symbolic function regression, clustering, search for topology of neural network, data mining, evolving rule sets etc.)

Why multiobjective?

Why use multiobjective genetic algorithms in grammar based genetic programming:

- To avoid "bloat" - situation when depth of trees rise and quality of programms is constant.
- When we need to optimize program on more than one objective.

NSGAII

- Authors: Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal
- Uses nondominating sorting approach.
- Uses density estimation approach.
- Doesn't need any additional parameters like sharing parameter. [3]

Literature

[1] Whigham P. A. Grammaticaly-based genetic programming, proceedings of the workshop on GP: from theory to real-worl applications, Tahoe City, 1995, pp. 33-41
[2] Zitzler E., Deb K. Thiele L. Comparision of multiobjective evolutionary algorithms: Empirical results (revised edition), Technical Report 70, Computer Engeneering and Networks Laboratory (TIK), Swiss Federal Institute of Technology Zurich, Switzerland
[3] Deb K., Agrawal S., Pratap A., Meyarivan T. A fast and elitist nondominated sorting genetic algorithm for multiobjective optimization: NSGA-II, in Parallel Problem Solving from Nature - PPSN VI, Berlin, pp. 849-858. Springer
[4] Pappa G. L., Freitas A. A. Evolving rule induction algorithms with multi-objective grammar-based genetic programming, Knowledge and Information Systems, Vol 19., Num. 3, 2007, pp. 283-309

End

Thank you for your attention.

?