MOEA/D

MOEA/D as Cellular GA

Yuri Lavinas
Master Student - University of Tsukuba

System and Information Engineering

05/09/2018

What is MOP?

Multiobjective Optimization Problem have m multiple objective functions that must be optimized simultaneously.
Minimize[1] $F(x) = (f_1(x), f_2(x), ..., f_m(x))$,
subject to x in Ω.

- $F(x)$ objective functions;
- f_i is the i-th objective to be minimized;
- x is the decision vector;
- Ω is the decision space.

[1] All definitions are for minimization. Following inequalities should be reversed if the goal is to maximize.

Why is MOP interesting?

1. Many real-world scientific and engineering are MOP.
    - Water quality control, Groundwater pollution remediation, Design of marine vehicles, … Coello et al. [2007].
    - Petrol extraction.
2. Hard problems: to balance the interests of the multi-objective as a whole is hard.

What is Pareto Set?

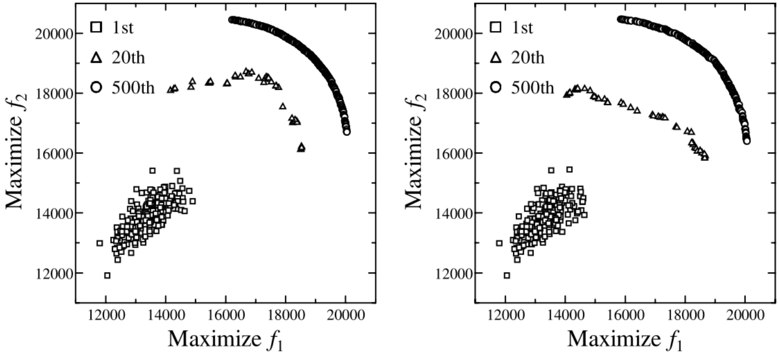
Objectives may be conflicting - The goal is to find good trade-offs.

- Set of solutions.

Set of *optimum solutions* - Pareto set.

- Non-dominated solutions: no single solution provides a better trade-off in all objectives.

What is Pareto Set?



(a) The weighted sum. (b) The weighted Tchebycheff.

From Ishibuchi et al. [2009].

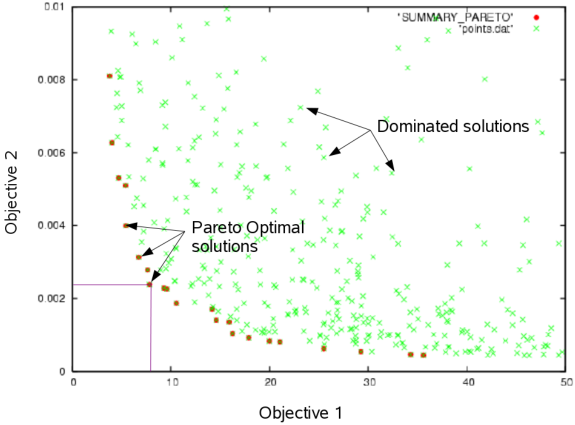
Non-dominated solutions

1. Let $u = (u_1, ..., u_m)$ and $v = (v_1, ..., v_m)$ vectors in Ω (the decision space).
   - $\forall i : u$ dominates v if $f_i(u) \leq f_i(v)$ and $\exists j : f_j(u) < f_j(v)$.
   - u dominates v, v is dominated by u, u is better that v.
2. A point x^* in Ω is called *Pareto Optimal* if no other point dominates x^*.

Pareto Front

1. The set of all Pareto Optimal is called the *Pareto Set*.
   - $P^* = \{x \in \Omega : \nexists y \in \Omega$ and $F(y) \leq F(x)\}$
2. **Pareto Front** is the image of the Pareto Set in the objective space.
   - PF $= \{F(x) = (f_i(x), ..., f_m(x)) : x \in P^*\}$

Pareto Front



From: http://www.cenaero.be/Page.asp?docid=27103&langue=EN

Decompostion

MOEA/D represents a class of population-based meta-heuristics for solving MultiObjective Problems (MOPs).

- It is based on decomposition - one kind of scalarizing function
- One multi-objective problem becomes various single-objective sub-problems.
- All sub-problems are solved in parallel.
- A decomposition strategy generates weight vectors that defines the sub-problems.

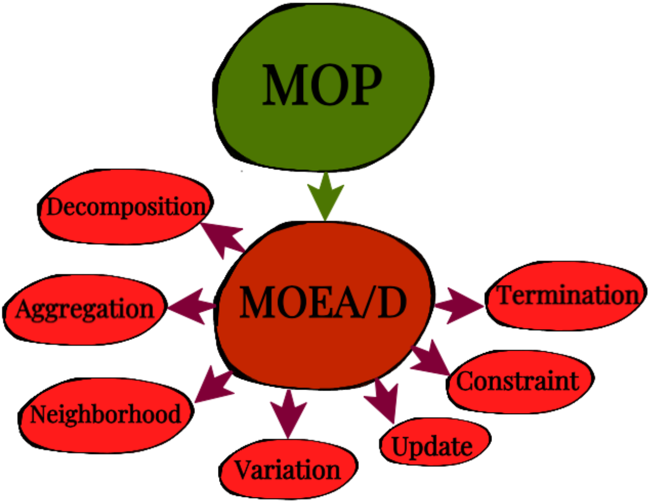
Why use decomposition?

- It may be good at generating an even distribution of solutions in MOPs;
- It reduces the computation complexity when compared to other algorithms (NSGA-II) (at each generation), Zhang et al. [2009];
- Fitness assignment and diversity maintenance become easier to handle.

Components of the MOEA/D

- Decomposition strategy: decomposes w/ weight vectors;
- Aggregation function: weight vector => single-objective sub-problems;
- **Neighborhood assignment strategy**: Relationship between sub-problems;
- Variation Stack: New candidates solutions;
- Update Strategy: Maintain/discard candidate solutions;
- Constraint handling method: Constraint violation;
- Termination Criteria: when to stop the search.

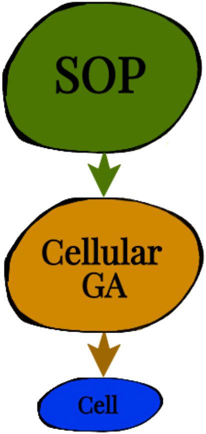
Simple Visualization

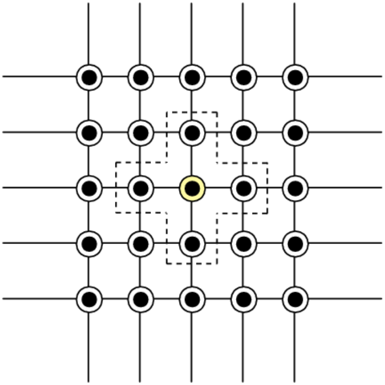


What is Cellular GA?

- Cellular GA (cGA) is a distributed model of the Genetic Algorithm (GA).
  - From Gong et al. [2015]: A cGA has one population but arranges the individuals on the grid, called **cell**.
- Cellular GA is well explored in the context of single-objective problems (SOP).
  - Each individual is only compared with its neighborhood.
  - Each individual uses only its neighborhood for updates.
  - It is highly parallelized: Synchronous x Asynchronous.

Cell and neighbors



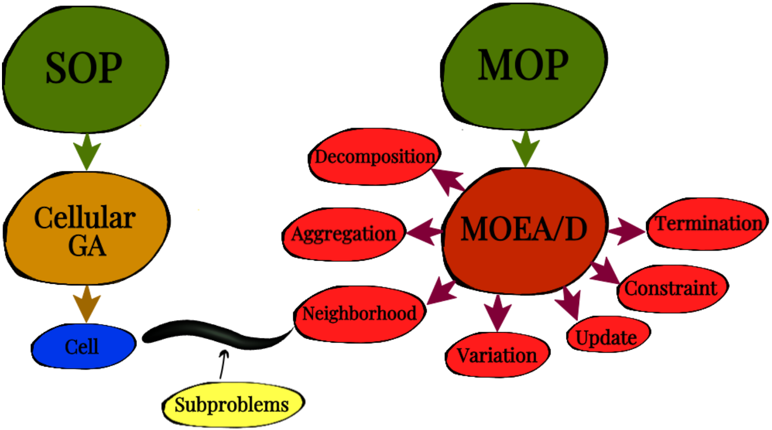


From: Robles et al. [2009]

Research focus

- Goal: To propose improvements to the MOEA/D by incorporating (well-)established techniques from the Cellular GA.
- Why? MOEA/D can be viewed as a cGA.
  - A cell can be seen as a specific "Neighborhood assignment strategy", where each cell has its own weight vector.

Research focus



MOEA/D as Cellular GA

1. Fitness of a solution is calculated at each cell using a scalarizing fitness function with the weight vector associated with that cell.
    - The cells are located in the (m-1)-dimensional subspace -> satisfying the normalization condition:
        - $w_1 + ... + w_m = 1$ and $w_i \geq 0$ for i = 1, 2 ,. . . ,m.
    - Two neighborhood structures:
        - Local selection.
        - Local replacement, both defined using the Euclidean distance between cells.

MOEA/D as Cellular GA

2. Parent solutions are randomly selected from the **selection neighbors**.
   2.2 Offspring is generated from the parents by crossover, mutation and/or repair.
3. Generated offspring is compared with the **replacement neighbors**, considering the weight vector.

Obs: The local selection and local replacement are performed for the next cell after they are completed for the current cell.

Discussion

- The main characteristic feature of MOEA/D as a cGA is the use of **local replacement** in addition to local selection.
  - Generated offspring for a cell is compared with not only the current solution of the cell but also its neighbors for possible replacement.
- Local replacement neighborhood has greater effect on the performance than local selection neighborhood.
  - Increasing its size tends to be better.

Discussion

- Good results with very large populations.
- Multiple neighbors can be replaced by one good offspring - less diversity.
  - Decreasing the size of competition neighborhood increases the non-dominated solutions, but degrades the search ability of the MOEA/D.

MOEA/D

1. Dual populations (external population).
   - Store the visited non-dominated solutions with a weight vector, and then add/remove sub-problems given crowded/sparse regions by adjusting the weight vector, Qi et al. [2014].
   - Stored the non-dominated solutions such that the Hyper Volume is maximized, Jiang et al. [2016].
   - External population is updated using non-dominates sorting and crowding distance <- NSGA-II. Also it is used to guide the allocation of computational resources to a sub-problem given its contribution, Cai et al. [2015].
2. Multigrid scalarizing scheme or a single grid with different scalarizing functions are alternately assigned for each weight vector Ishibuchi et al. [2010].

MOEA/D as Cellular GA

1. Asynchronous - could degenerate the algorithm.
   - Dynamic computation resource allocation to different sub-problems, Zhang et al. [2009] and, Zhou and Zhang [2016].
2. Island model with different parameter values and/or reproduction operators and/or scalarizing scheme - aiming to balance convergence and diversity, and to explore the objective space better.
   - Neighborhood relationship should be defined in the objective space and should be adaptive, Giagkiozis et al. [2014].
   - (self-)Adaptive replacement strategies, Wang et al. [2016] and Zhang et al. [2016].

Basic References

Most of this presentation was based on the works of Trivedi et al. [2017] and Zhang and Li [2007].

Xinye Cai, Yexing Li, Zhun Fan, and Qingfu Zhang. An external archive guided multiobjective evolutionary algorithm based on decomposition for combinatorial optimization. *IEEE Transactions on Evolutionary Computation*, 19(4):508–523, 2015.

Carlos A Coello Coello, Gary B Lamont, David A Van Veldhuizen, et al. *Evolutionary algorithms for solving multi-objective problems*, volume 5. Springer, 2007.

Ioannis Giagkiozis, Robin C Purshouse, and Peter J Fleming. Generalized decomposition and cross entropy methods for many-objective optimization. *Information Sciences*, 282:363–387, 2014.

Yue-Jiao Gong, Wei-Neng Chen, Zhi-Hui Zhan, Jun Zhang, Yun Li, Qingfu Zhang, and Jing-Jing Li. Distributed evolutionary algorithms and their models: A survey of the state-of-the-art. *Applied Soft Computing*, 34:286–300, 2015.

Hisao Ishibuchi, Yuji Sakane, Noritaka Tsukamoto, and Yusuke Nojima. Adaptation of scalarizing functions in moea/d: An adaptive scalarizing function-based multiobjective evolutionary algorithm. In *International Conference on Evolutionary Multi-Criterion Optimization*, pages 438–452. Springer, 2009.

Hisao Ishibuchi, Yuji Sakane, Noritaka Tsukamoto, and Yusuke Nojima. Simultaneous use of different scalarizing functions in moea/d. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, pages 519–526. ACM, 2010.

Siwei Jiang, Liang Feng, Dazhi Yang, Chen Kim Heng, Yew-Soon Ong, Allan NengSheng Zhang, Puay Siew Tan, and Zhihua Cai. Towards adaptive weight vectors for multiobjective evolutionary algorithm based on decomposition. In *Evolutionary Computation (CEC), 2016 IEEE Congress on*, pages 500–507. IEEE, 2016.

Yutao Qi, Xiaoliang Ma, Fang Liu, Licheng Jiao, Jianyong Sun, and Jianshe Wu. Moea/d with adaptive weight adjustment. *Evolutionary computation*, 22(2):231–264, 2014.

Ignacio Robles, Rafael Alcalá, José Manuel Benítez, and Francisco Herrera. Evolutionary parallel and gradually distributed lateral tuning of fuzzy rule-based systems. *Evolutionary Intelligence*, 2 (1-2):5, 2009.

Anupam Trivedi, Dipti Srinivasan, Krishnendu Sanyal, and Abhiroop Ghosh. A survey of multiobjective evolutionary algorithms based on decomposition. *IEEE Transactions on Evolutionary Computation*, 21(3):440–462, 2017.

Zhenkun Wang, Qingfu Zhang, Aimin Zhou, Maoguo Gong, and Licheng Jiao. Adaptive replacement strategies for moea/d. *IEEE transactions on cybernetics*, 46(2):474–486, 2016.

Hu Zhang, Xiujie Zhang, Xiao-Zhi Gao, and Shenmin Song. Self-organizing multiobjective optimization based on decomposition with neighborhood ensemble. *Neurocomputing*, 173:1868–1884, 2016.

Qingfu Zhang and Hui Li. Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on evolutionary computation*, 11(6):712–731, 2007.

Qingfu Zhang, Wudong Liu, and Hui Li. The performance of a new version of moea/d on cec09 unconstrained mop test instances. In *Evolutionary Computation, 2009. CEC'09. IEEE Congress on*, pages 203–208. IEEE, 2009.

Aimin Zhou and Qingfu Zhang. Are all the subproblems equally important? resource allocation in decomposition-based multiobjective evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 20(1):52–64, 2016.

Table of Contents