# MULTIOBJECTIVE OPTIMIZATION VIA CO-EVOLUTION FOR THE JOB SHOP SCHEDULING PROBLEM

ESQUIVEL S. C., FERRERO S.W., GALLARD R.H.

Proyecto UNSL-338403[1]
Departamento de Informática
**Universidad Nacional de San Luis (UNSL)**
**Ejército de los Andes 950 - Local 106**
5700 - San Luis, Argentina.
E-mail: {esquivel,swf,rgallard}@unsl.edu.ar
Phone: + 54 652 20823
Fax : +54 652 30224

## Abstract

A job shop is a facility that produces goods according to specified production plans under several domain-dependent constraints. Job Shop Scheduling (JSS) attempts to provide optimal schedules. Common variables to optimize are total completion time (makespan), machine idleness, lateness and total weighted completion time. According to this variables different objectives can be devised.
Multiobjective optimization, also known as vector-valued criteria or multicriteria optimization, have long been used in many application areas where a problem involve multiple objectives, often conflicting, to be met or optimized.
Co-evolution, as an extended evolutive model, can be applied to solve multicriteria optimization for the JSS problem using a plain aggregative approach.
This presentation will show the design, implementations and results of a co-evolutive approach solving a multiobjective optimization problem involving the makespan, machine idleness and total weighted completion time as criteria to be optimized.

[1] The Research Group is supported by the Universidad Nacional de San Luis and the ANPCYT (National Agency to Promote Science and Technology).

## Introduction

In multiobjective optimization Fonseca and Flemming [2] classified as plain aggregative approaches those methods where a single objective function resulting as a numerical combination of objectives values is to be optimized. Here decisions on multicriteria are made before searching. The unique objective function obtained by aggregation of multiple objectives is used to establish a total order in the solution's space. This measure provides then a basis for selection of individuals.
In this way, a genetic algorithm performs as usual finding the fittest individuals for that single aggregated function. Bhanu and Lee [1] and Vemuri and Cedeño [5], worked on this linear combination approach.

In nature, individuals do not evolve in isolation. Instead, there is a co-evolution [4] that involves interactions between individuals of diverse evolving populations (species).
Co-evolution can be applied to solve multicriteria optimization for the JSS problem using a plain aggregative approach. The idea is to create one sub-population for each criterion and evolve them until convergence. At this point another evolutionary process begin actuating on the whole population whose objective is an aggregation of the partial objectives. This evolution step continues until reaching convergence. After that the whole population is subdivided into sub-populations and the original process is started again. A final stop criterion is defined to terminate the entire process.

As a result, it is expected that at the end of the co-evolutive process, we obtain a set of good performers which optimizing the aggregated objective are also the best on their corresponding partial objective.

## 2. The Job Shop Scheduling Problem (JSSP)

This scheduling is related to the allocation of limited resources (machines) to jobs over time. This is a decision making process that has as a goal the optimization of one or more objectives.
The model considered here assumes that the system consists of a number of different machines and only one job may execute on a machine at a time. All schedules and jobs are non-preemptive. Jobs can have distinct priorities and all of them are available at production initiating time. There are no restriction on due dates for the jobs. Each job visits all machines, only once, following a predetermined sequence of machines, called a *route*. Consequently a job can be seen as composed by various steps, called *operations*.
So, a job can be represented by a vector where its components are the successive operations to be performed. These components are 2-tuples of the form (machine, duration), specifying the machine where the job must be allocated and the time spent in that machine.
An instance of the JSSP is matrix where the rows specifies the jobs as above described. This matrix is called the *instance matrix* for the specific JSSP.
Associated with the instance matrix is a *priority list*, which is used at the building stage of a schedule to solve conflicts between jobs requiring resources.
Let us consider the following example:

Given a JSSP, with three jo[illegible] machines, the instance matrix $I$ and prio[illegible]

Instance matrix $I$

| | $O_1$ | $O_2$ |
|---|---|---|
| $J_1$ | (2,4) | (1,2) |
| $J_2$ | (1,3) | (2,8) |
| $J_3$ | (2,7) | (1,4) |

Priority list $L$

| | | |
|---|---|---|
| [illegible] | [illegible] | [illegible] |

A schedule specified by the following Gantt chart can be built,

| | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $M_2$ | $J_1$ | | | | $J_2$ | | | | | | | | $J_3$ | | | | | | | | | |
| $M_1$ | $J_2$ | | | | $J_1$ | | | | | | | | | | | | | | | $J_3$ | | |
| Time | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 |

As we can see, given an instance of the JSSP, different schedules can be built when we use different priority lists.

### 3. The co-evolutionary approach for multiciteria optimization

Given an instance of he JSSP, of $n$ jobs and $m$ machines we considered three performance parameters to minimize:

**MS** = max $\{C_1,\ldots,C_n\}$, where $C_i$ is the completion time of Job $i$. It is the *makespan* an it is equivalent to the completion time of the last job to leave the system.

**TIT**= $\sum_{j=1}^{m} IT_j$ where $IT_j$ is the idle time of machine $j$. It is the *total idle time.*

**WCT**= $\sum_{i=1}^{n} w_i C_i$, where $w_i$ is the weight associated to the completion time of Job $i$. It is the weighted completion time.

Our co-evolutionary approach considered the 3-criteria problem optimizing three objective functions, $f_1$, $f_2$ and $f_3$, corresponding to *MS*, *TIT* and *WCT*, respectively and the aggregation $f = \alpha f_1 + \beta f_2 + \gamma f_3$.

An individual in any population is an integer vector representing a priority list. Consequently a population evolves by creating new individuals (priority lists) optimizing some of the above mentioned objective functions (criteria).

There exist two different evolutionary processes:

1) *Independent evolution.* Here three populations evolve independently, each one optimizing one of the above mentioned criteria. This process is performed until each population reach convergence.

2. *Unif*[illegible] convergence of independent popu[illegible] are merged into a single population and the evolutionary process optimize now the aggregation *f*. That means that the independent evolved individuals are now submitted to a new environment. This stage involves interactions between individuals of diverse evolving populations under new environmental conditions.

Both processes are repeated until the termination criterion for the unified population holds.

**The whole co-evolutive process can be delineated as follows:**

Begin *Co-evolutive process*

1. Initialize 3 distinct populations of size *s* (one for each objective)
2. Evolve independently each population until the termination criterion $\theta_1$ holds.
3. Merge the independently evolved population into a single unified population of size 3*s*.
4. Evolve the unified population until the termination criterion $\theta_1$ holds.
5. If the termination criterion $\theta_2$ holds then stop

    **else rank individuals according to:**

    *f1* and create a new population with the best one third of the unified population.

    *f2* and create a new population with the best one third of the unified population.

    *f3* and create a new population with the best one third of the unified population.

    go to step 2.

End of *Co-evolutive process*.

Termination criterion $\theta_1$ stops the evolutionary process when after 10 consecutive generations the difference between mean population fitness values remains less than 1%.
Termination criterion $\theta_2$ stops the co-evolutionary process when after 5 consecutive generations of the unified population the difference between mean population fitness values remains less than 5%.

## 4. Experiments and results

The co-evolutionary approach was contrasted against an evolutionary approach. In both cases simple but non canonical genetic algorithms were used.
For our experiments, randomised initial populations of size fixed to 30 individuals were used to optimize each criterion, using integer representation, elitism, one point crossover and big creep mutation. The number of generations was bounded by the corresponding termination criterion and probabilities for crossover and mutation were fixed to 0.65 and 0.03 respectively. Ten instances of two types, small and big, with known optimal makespan were used. small instances were of 10 jobs and 5 machines while big instances were of 20 jobs and 10 machines.

...worthwhile to remark that when looking at the makespan, *MS*, a reduction of the mean error was observed when the co-evolutive approach was contrasted against the evolutive approach.
On the small instances the error of 2.482 % went down to 2.398% and on the big instances the value of 9.142% was reduced to 8.822%.
Regarding to the other performance variables, *TIT* and *WCT*, mean error values could not be established because optimum values are unknown. Nevertheless the values found by the co-evolutionary approach are in general slightly better than those obtained with the evolutionary approach.
Referring to the minimum value of the aggregation $f$ after co-evolution it worth saying that it is quite near of the value obtained by computing the linear combination of the best independent evolved values.

## 5. Conclusions

This presentation shows an application of co-evolution to face an aggregative multicriteria optimization approach for the JSSP.
Independent and unified evolution were introduced to obtain partial optimization of three distinct criteria, $f_1$, $f_2$ and $f_3$, and the aggregation criterion $f$ at the same time.
Results of these preliminary experiments with co-evolution show some enhancements when compared with the conventional evolutionary approach.
Future work include diverse parameter settings, recombination and hybridization expecting better performance for the method proposed here.

## 6. Bibliography


[1] Bhanu B, Lee S., *Genetic learning for adaptive image segmentation*, (Boston MA: Kluwer). 1994.

[2] Fonseca C. M., Fleming P. J.- *Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization* - Proc. of the 5th In. Conf. on Genetic Algorithms, pp 416-423, Urbana- Champaign, IL, Morgan Kaufmann, 1993

[3] Leitmann G., Marzollo A., *Multicriteria Decision Making* – CISM No 211, Springer Verlag, Wien , NY, 1975.

[4] Paredis J., Coevolutionary constraint satisfaction. Proc. 3rd Ann. Conference on Parallel Problem Solving from Nature (New York: Springer) pp 46-55, 1994.

[5] Vemuri R., Cedeño W., *A new genetic algorithm for multiobjective optimization in water resource management*, Proc. of the 1st IEEE International Conf. on Evolutionary Computation (ICEC'94), pp 495-500, Orlando, USA, 1994.