IEEE CEC 2018

A Decomposition-based Local Search Algorithm for Multi-objective Sequence Dependent Setup Times Permutation Flowshop Scheduling

Murilo Zangari, Ademir Constantino, and Josu Ceberio

State University of Maringa, Brazil
University of the Basque Country, Spain

July 11, 2018

Schedule

Flowshop Scheduling Problem (FSP)

- FSP serves as a model for several real-world problems from manufacturing, engineering, and other fields of application
- Different objectives have been considered as optimization goals, e.g., i) *Makespan, ii) Total Flowtime, and iii) Total Tardiness*

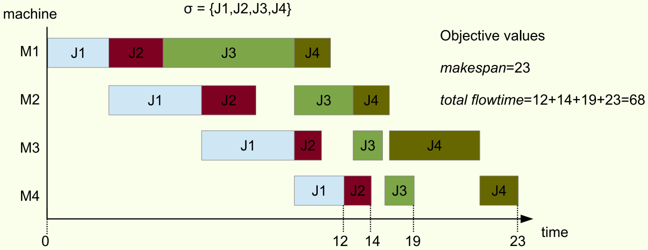


Figure: An example of flow-shop scheduling problem with 4 jobs and 4 machines

- Problem restriction: When the sequence of jobs is the same for all machines, the problem is denoted Permutation Flowshop Scheduling (PFSP)
- Goal: To find a permutation of jobs such that a criteria is minimized

Flowshop Scheduling Problem (FSP)

- **Setup Times**: In FSP, setups can reflect some non-productive operations that have to be processed on machines that are not part of the processing times of the jobs
- **Sequence-dependent:** When the setups dependents on the job being processed and on the next job in the sequence
- **Multi-objective optimization:** Multiple PFSP objectives to be optimized simultaneously.

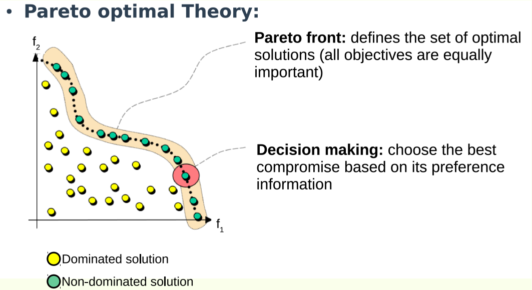


Figure: An example of maximization of two objectives

PFSP solvers

- Heuristics Algorithms
- Meta-heuristics Algorithms
  - Evolutionary Strategies (e.g., genetic search)
  - Simulated Annealing
  - Iterated Local Search (ILS)
- Hybrid Approaches
  - Example: Heuristic + Evolutionary Strategies + local search
  - Goal: An effective balance between *Convergence* $\times$ *Diversity*

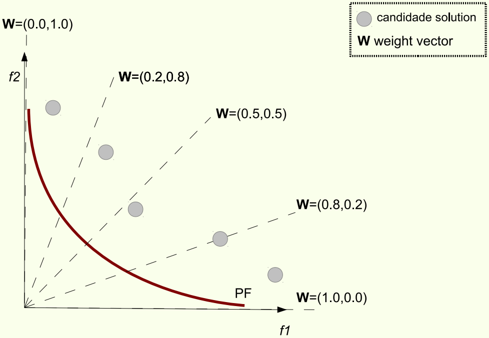
Concept Definition: Iterated Local Search

- A solution $\mathbf{x}'$ is a neighbor solution of $\mathbf{x}$ if $\mathbf{x}'$ can be achieved by a single move from $\mathbf{x}$, and its depends on a basic underlying *operator* and a given *distance* between any two solutions

- **Iterated Local Search (ILS):** consists of repeatedly applying local search procedures. When the search is trapped in a local optimal solution, ILS can perturb the solution to allow the search to escape from the trap without losing many of the good properties of the current solution

Concept Definition: Multi-objective EA based on Decomposition

- **MOEA/D framework**: The idea is to decompose a multi-objective problem into a number of scalar single-objective subproblems (by using a set of weight vectors and an aggregation function). Each subproblem is optimized using an EA and the information mainly from its several neighboring subproblems
- Well effective for solving combinatorial optimization problems with 2 and 3 objectives.



Multi-objective ILS based on Decomposition

- Ingredients:
  - Decomposition strategy: *Weighted Sum Approach*
  - Heuristic initialization of the population using variants of the *NEH heuristic*
  - Local search operators in a space of permutations: *1-insert* and *1-interchange* (*exploitation*)
  - Shaking procedure: to move the current solution of a subproblem to another region of the search (*exploration*)

Multi-objective ILS based on Decomposition

- **Input:**
  - N: population size
  - $\boldsymbol{W}^1, ..., \boldsymbol{W}^N$: distributed weight vectors
  - T: neighborhood size for update
  - n_r: maximum replacements allowed by a new solution
  - n_{sh}: number of random *insert moves*
- **Output:**
  - *Pop*: The final set of N solutions
  - *External Pareto* (all non-dominated solutions found)
- **Initialization**: $Pop := \{\sigma^1, ..., \sigma^N\}$
- *While a stopping criterion is not met:*
  - **Search Process:** For each $k \in 1, ..., N$:
    - **Generate** σ' using a LS move on σ^k and compute $F(\sigma')$
    - **Update** *Pop* with σ' according to the scalar aggregation function and the T closest neighbor subproblems
    - **Check** if the subproblem has not been improved after n generations
    - **Update** *External Pareto*

Experimental setup

- **Benchmark:** 220 instances extended from the Taillard benchmark that vary according to the number of jobs $n = \{20, 50, 100\}$, number of machines $m = \{10, 20\}$, and setup times.
- **Bi-objective case:** *makespan* and *total weighted tardiness*
- **Comparison:**
  - The best-known results from the literature (in the form of approximated *Pareto fronts*) obtained by the best performer algorithms (RIPG and MOSA_VM)
  - MOEA/D (which employs genetic operators, specially toileted for permutation problems)
- **Performance assessment:** 1) Hypervolume, 2) Coverage, and 3) *Empirical Attained function* (EAF)

Experimental setup

- Parameters settings: $N = 100$, $T = 20$, $n_r = 2$, $n_{sh} = 14$ random *insert moves*
- Maximum Generations: $1000n$
- 20 independent runs for each algorithm and problem instance
- Statistical tests: *Friedman's* ranked based at 95% of confidence level and *post-hoc* Nemenyi

Results: MOLS/D components

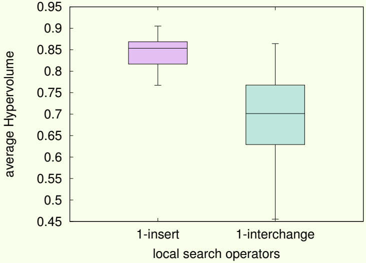


Figure: Boxplot of the average *HV* values obtained by MOLS/D using the *1-insert move* and the *1-interchange move*.

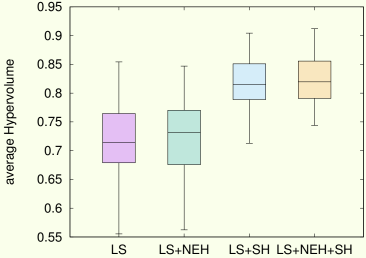
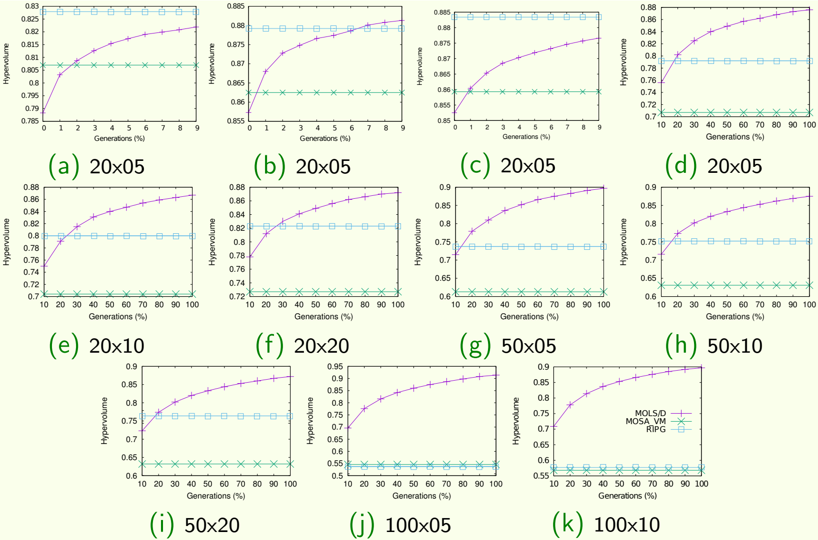


Figure: Boxplot of the average *HV* values obtained by four algorithm configurations: (i) without both heuristic initialization and shaking procedure (LS), (ii) only with the heuristic initialization (LS+NEH), (iii) only with the shaking procedure (LS+SH), and (iv) with all the components together (LS+NEH+SH).

Results: Comparison to the best-known results from literature



(a) 20x05 (b) 20x05 (c) 20x05 (d) 20x05

(e) 20x10 (f) 20x20 (g) 50x05 (h) 50x10

(i) 50x20 (j) 100x05 (k) 100x10

Figure: Average *HV* values obtained by MOLS/D throughout 10 different stages of the search compared to the *HV* obtained by the reference sets of MOSA and RIPG (constant lines) for the different problem scales (11 in total)

Results: Comparison results

Table: Average *HV* values obtained by MOLS/D, MOEA/D, MOSA_VM, and RIPG

problem $n \times m$	SSD50				SSD125			
	MOLS/D	MOEA/D	MOSA_VM	RIPG	MOLS/D	MOEA/D	MOSA_VM	RIPG
20×05	**0.847**	0.649	0.832	**0.852**	**0.822**	0.542	0.799	**0.835**
20×10	**0.893**	0.774	0.874	**0.890**	**0.867**	0.684	0.852	**0.871**
20×20	**0.874**	0.712	0.856	**0.881**	**0.863**	0.668	0.834	**0.870**
50×05	**0.877**	0.413	0.704	0.788	**0.869**	0.357	0.656	0.812
50×10	**0.869**	0.440	0.706	0.801	**0.856**	0.412	0.659	0.816
50×20	**0.867**	0.487	0.717	0.817	**0.872**	0.445	0.678	0.831
100×05	**0.897**	0.348	0.612	0.735	**0.874**	0.251	0.509	0.764
100×10	**0.875**	0.382	0.632	0.752	**0.862**	0.309	0.517	0.767
100×20	**0.873**	0.377	0.633	0.763	**0.885**	0.347	0.554	0.787
200×10	**0.914**	0.320	0.549	0.538	**0.906**	0.285	0.420	0.549
200×20	**0.896**	0.375	0.565	0.574	**0.906**	0.369	0.479	0.627

Results: Diff-EAF tool

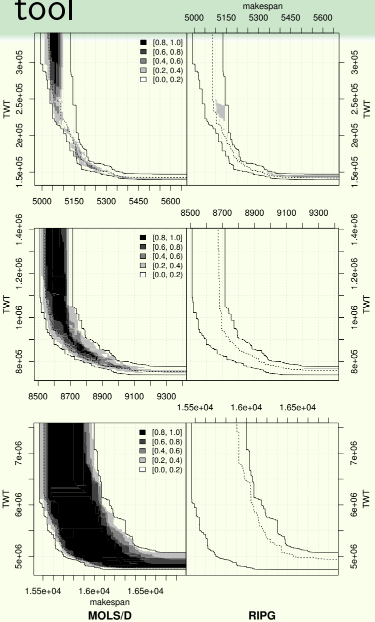


Figure: Diff-EAF between MOLS/D (left) and RIPG (right) for SSD50_051 (50 $\times$ 20) (top), SSD50_071 (100 $\times$ 20) (middle), and SSD50_101 (200 $\times$ 20) (bottom).

Final Considerations

- A simple yet efficient population-based algorithm: Multi-objective Iterated Local Search Algorithm based on Decomposition
- Hybrid approach: heuristic + ILS strategy + diversity mechanism
- Experimental study using 220 benchmark instances with different problem scales
- Contributions:
  - MOLS/D outperforms a tailored MOEA/D
  - MOLS/D is able to achieve better results than the state-of-the-art approaches for the benchmark considered
  - We made our results (in the form of approximated *Pareto fronts*) available for further investigations by other researches. Available at https://github.com/murilozangari/sdst results.
- Future work:
  - The application of MOLS/D to solve i) SDST flowshop with three objectives, ii) flexible job-shop scheduling, and iii) other kind of permutation problems.

Acknowledgment

This work has been supported by:

- PNPD/CAPES (Brazilian Program of Post-Doctoral)
- CNPq (Productivity Grant Nos. 306754/2015-0)
- Research Groups 2013-2018 (IT-609-13) programs (Basque Government)
- TIN2016-78365-R (Spanish Ministry of Economy, Industry, and Competitiveness)

Thank you!

UEM Universidade Estadual de Maringá

GPEA Grupo de Pesquisa em Engenharia de Algoritmo

eman ta zabal zazu
Universidad del País Vasco — Euskal Herriko Unibertsitatea