Syntheses

March 2019

Synthesis 18/03

- **Multi-objective Knapsack problem (Maximization)** :
  n : number of items.
  p : number of criteria.
  $u_{i,j}$: utility of item i for the j-th criteria.
  w_i : weight of item i.
  W : capacity of the backpack.
  x_i : decision variable (1 if we add the item i, 0 otherwise).

$$
\begin{aligned}
& & z_1 &= \sum_{i=1}^{n} u_{i,1}\ x_i \\
& Maximize & \ldots & \\
& & z_p &= \sum_{i=1}^{n} u_{i,p}\ x_i \\
& s.t & \sum_{i=1}^{n} w_i x_i &\leq W \\
& & x_i &\in \{0,1\}, \quad i = 1, ..., n
\end{aligned}
$$

  Our experiments are based on a bi-objective knapsack problem but is adapted for more than 2 objective.

- **The Decision Maker preferences** will be simulated by a hidden WS-function f_λ :

$$
f_\lambda(x) = \sum_{j=1}^{p} \lambda_j\ z_j
$$
$$
\sum_{j=1}^{p} \lambda_j = 1
$$

- **Multi-Objective Local Search algorithms (MOLS)** :
  is a generalization of a simple Local Search by considering only one criteria. It follows a common structure, we define below for our problem :

  1. Initial Population : What is the size of the population and how are they generated ? one alternative (for simplification) generated randomly.
  2. Next exploration : Which alternative is explored next (from the archive) ? FIFO.
  3. Neighborhood exploration : How do we generate new solutions ?
     For each item in the current backpack :
     —— Temporarily remove it from the Backpack.
     —— Generate a random set of weights $\mathbf{c}_i$, $i = 1, ..., p$ of a WS function and compute $r_{(.)} = \frac{f_{\mathbf{c}}(.)}{w_.}$
     —— Fill the backpack with the remaining items by adding them in the decreasing order of $r_{(.)}$.

     → NEED TO DIVERSIFY THE NEIGHBORHOOD.

  4. Neighborhood selection part : we focus on two approaches

     – **Pareto Local Search** :
       Select only neighbors that are non-dominated using a Pareto dominance relation.
       → ++Good quality of efficient front, ++Diversity, - -time consuming.

– **Aggregate function (Weighted Sum $f_\omega(.)$)** :
Select the neighbor with the highest WS-function value.
→ ++Fast computation, **- -**Diversity, **- -**Quality of solution.

- **Experimental results** :
How the quality of the solution evolve when at first no information about the DMs preferences is given, then starting to get more information to finally knowing the exact set of weights (WS) of the DM.

The sample size of the experiment was about **30** tests and fixed a timeout was required for large instances (**3 minutes**) .

– Different experiments shown in figure **1a**, **2a** and **3a**. It presents the evolution of the average gap considering the information rate (0% to 100%).

– The average execution time (second) was plotted respectively in figure **1b**, **2b** and **3c**.

– It is also interesting to measure the quality of the front regarding the information rate. We used two types of indicator :

– Average minimal distance (figures **1c**, **2c** and **3c**) : compute an euclidean distance between the founded solutions χ of the MOLS algorithm and the optimal front $\mathcal{OPT}$:

$$D(\chi, \mathcal{OPT}) = \frac{\sum_{r \in \mathcal{OPT}} min_{x \in \chi} \ dist(r,x)}{|\mathcal{OPT}|}$$

– Proportion Reference (figures **1d**, **2d** and **3d**) : measure the proportion of Pareto optimal front in χ.

- **Combining two type of selection Pareto and WS** :
Starts with a WS selective approach (exploit) for a fixed number of exploration then switch into a Pareto dominance (explore).
The aim behind this : usually taking time in the beginning of a local search method is time consuming and less effective.

– Results are presented in last figures of PDF file.
It shows the search space evolution using Pareto, WS and the combination of both (WS-PLS). We also plot the optima front (OPT).

- **Note** :
Due to a less divers neighborhood, the approaches tends to stuck very quickly (few iterations) into a local optima no matter how large the problem size is.