A genetic algorithm for learning the parameters of an SRMP preference model

Bastien Pasdeloup[1] and **Arwa Khannoussi** [2] and **Alexandru-Liviu Olteanu** [3] and **Patrick Meyer** [4]

Abstract. In the domain of Multiple Criteria Decision Aiding, decision makers are faced with problems involving multiple conflicting criteria. Preference models are used to reach a decision in such situations. To tune the parameters of those models, preference elicitation algorithms are used, generally using so-called holistic judgments as inputs. In this work, we focus on a specific preference model called ranking based on multiple reference profiles. In the literature, mixed-integer linear programming and constraint programming techniques have already been proposed to tune the model parameters. However these approaches have difficulties to handle realistic large scale problems. We propose here an evolutionary metaheuristic in order to address this issue, which we test using extensive numerical experiments in order to highlight its performance and limits. We show that the proposed metaheuristic has the capacity to reproduce learning inputs very well, while having an important generalization power.

1 Introduction and related work

Difficult decisions usually involve multiple, often conflicting, perspectives over a set of alternatives. In this paper, we consider the ranking problem in Multiple Criteria Decision Aiding (MCDA), and, more specifically, the *Ranking based on Multiple reference Profiles* (RMP) [8, 2] preference model. We focus on a specific case of the RMP model in which the importance of criteria is represented by additive weights. RMP is based on pairwise comparisons, but instead of directly comparing alternatives, it compares each alternative to a set of predefined reference profiles. The objective is to construct a preference relation on the set of alternatives based on the way each alternative compares with the specified reference profiles.

The RMP method belong to the class of MCDA methods that are based on outranking relations [9]. An outranking relation, denoted as $\succsim$, is a pairwise relation between two alternatives a and b, indicating whether a is "*at least as good as*" b. Generally, this involves determining if there are sufficient arguments to state that a is at least as good as b, and if there is no major reason to refute the statement. The use of reference profiles is rather wide spread among MCDA methods, and is based on psychological evidence [10] which suggests that decision makers (DMs) often base their decisions on so-called references which correspond to their current expectations on the decision problem. A few examples can be found in [3, 1, 9, 5].

In order to be used in practice, the parameters of the RMP ranking model (criteria weights, performances of the reference profiles, a lexicographic order of the reference profiles) need to be tuned so that the model accurately reflects the perspective of the DM. The interaction between an analyst and a DM in order to set the values of a preference model is called preference elicitation. The direct approach requires the DM give numerical values for the model parameters, while the indirect approach uses holistic information given by the DM in order to infer the model parameters [4]. We focus here on the indirect approach where the information provided by the DM consists in pairwise comparisons over a few alternatives only.

Previous works on inferring the parameters of the RMP model [7] involve exact resolution approaches that require significant computational resources and time. A few approaches involving metaheuristics may also be found in [11, 6]. We extend these works by proposing an evolutionary metaheuristic in order to make the inference process more tractable and allow for its application in real-life situations. We also extend these studies by considering larger problem instances.

2 A genetic algorithm to tune the SRMP parameters

We do not present the Simplified RMP (SRMP) model here. The interested reader can refer to [7]. The proposed inference algorithm follows the classical elements of a genetic algorithm:

- A *solution* consists of the performances of the reference profiles, criteria weights and a lexicographic order of the reference profiles.
- The *fitness function*, which allows to evaluate the quality of a solution during the search, calculates the percentage of input statements correctly modeled by the current solution (train accuracy).
- The genetic algorithm *initializes* a population of solutions and then improves it through repetitive applications of *mutation*, *crossover* and *selection* operators. The *initial population* is generated randomly, allowing the entire range of possible solutions.
- We then initialize the next population with some of the current best solutions and some new random solutions in order to maintain diversity. The rest of the population is obtained through a reproduction method, where some *parent solutions* are *selected* to create *children solutions*. Individual solutions are selected based on the value of their fitness using a *roulette wheel* procedure. To generate the next population of solutions from those selected, crossover and mutation operators are used. A *crossover* operator produces a child solution from two parent solutions, by keeping some characteristics of the parents. The child is then transformed through *mutation* operators to diversify the population. It is finally inserted in the population if it improves the fitness of at least one of its parents.
- The process ends when the population does not improve anymore during a certain number of iterations.

In this work, multiple crossover and mutation operators are introduced. For the crossovers we consider: *exchange of weights between*

[1] IMT Atlantique, Lab-STICC, UMR CNRS 6285, F-29238 Brest, France
[2] IMT Atlantique, LS2N, F-44300 Nantes, France
[3] Lab-STICC, UMR 6285, CNRS, Université Bretagne Sud, France
[4] IMT Atlantique, Lab-STICC, UMR CNRS 6285, F-29238 Brest, France

two solutions, *exchange of the lexicographical order between two solutions*, *exchange of profiles between two solutions*, *mixing of the profiles between two solutions*, and *mixing profiles and associated weights between solutions*. For the mutations, we consider: *random perturbation of the weights*, *random perturbation of the profiles*, *contraction of the profiles* (i.e. moving them closer one to another), and *dilatation of the profiles* (i.e. increasing the distance between them).

The hyperparameters of the genetic algorithm are: *number of input pairwise comparisons*, *choice of selection strategy*, *number of parents to be selected with selection strategy*, *population size*, *parameters of the mutation operators*, *stopping criterion*, *number of profiles in the solutions*, *survival probability* (i.e. probability for a child to be kept in the new population even if its fitness is lower than those of the parents), *elitism ratio* (i.e. number of best solutions that are systematically kept in the next population), and *random ratio* (i.e. number of new random solutions to inject in the next population).

3 Setting the hyperparameters of the algorithm

To set the hyperparameters of the genetic algorithm we consider a problem with 11 criteria, 3 profiles, 50 alternatives in the learning set, and 100 pairwise comparisons provided to the algorithm to learn the RMP parameters. The size of the problem makes it difficult (even impossible) to solve using existing approaches.

To evaluate the quality of a specific configuration of the genetic algorithm, we calculate different accuracies. The *train accuracy* measures the ability of the algorithm to reproduce the decisions of the ground truth DM on comparisons already made (i.e., those used to learn the RMP model). The *test accuracy 1* measures the ability of the algorithm to generalize to unseen comparisons of alternatives from the same dataset as the one used to train the model. And finally the *test accuracy 2* measures the ability of the algorithm to generalize to unseen comparisons of alternatives from a different dataset.

We have performed a grid search for each pair of operators (crossover, mutation) in isolation of others, and have measured the average accuracy over 10 problem instances for each configuration of the parameters. The following hyperparameters have been fixed as follows to make the grid search tractable:

- Parents selection strategy: roulette;
- Number of parents: 2;
- Stopping criterion: 50 iterations with no change among the $\lceil \frac{e}{p} \rceil$ best solutions in the population, where e is the elitism ratio and p the population size;
- Number of profiles in estimated solution: 3. Note that this hyperparameter simplifies the problem, as it is fixed knowing the ground truth has 3 profiles. In practice, one should iterate over this parameter to find the best setting.

To determine useful pairs of operators, we also realized numerous simulations on the same problem instances, where all operators have a chance of being used during the process. This gives a baseline of average performance for pairs of operators used all across the process. Among the proposed operators, the crossovers standing out are: *mixing of the profiles between two solutions*, and *mixing profiles and associated weights between solutions*. Regarding mutations, those that outperform others are: *random perturbation of the weights* and *contraction of the profiles*. We have thus restricted the list of possible operators to those listed above, and have allowed random selection of the corresponding pairs to produce children.

4 Performance of the genetic algorithm

With hyperparameters of the genetic algorithm now set, we have studied the performance of the algorithm on artificial problems.

In Figure 1 one can observe the train accuracy in red. It decreases slowly when the number of input pairwise comparisons increases. This is normal, as it becomes harder to reproduce the input pairs when their number increases. The accuracies on the two test sets increase however with the number of input pairwise comparisons. This is to be expected, since more preferential information in the input allows to learn the preferential parameters in a more faithful way.

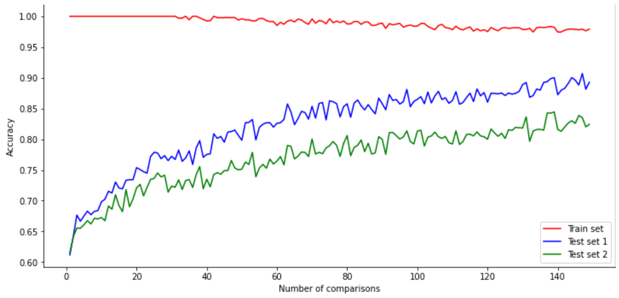

Figure 1. Average accuracies of the inferred RMP models as a function of the number of comparisons used to learn the models.

Regarding computation times, obviously the training time increases with the number of comparisons. Our measurements show that, with 140 pairwise comparisons, computation time lies around 1 minute.

REFERENCES

[1] C.A. Bana e Costa and J-C. Vansnick, 'MACBETH—An interactive path towards the construction of cardinal value functions', *International transactions in operational Research*, **1**(4), 489–500, (1994).

[2] D. Bouyssou and T. Marchant, 'Multiattribute preference models with reference points', *European Journal of Operational Research*, **229**(2), 470–481, (2013).

[3] C-L. Hwang, Y-J. Lai, and T-Y. Liu, 'A new approach for multiple objective decision making', *Computers & Operations Research*, **20**(8), 889–899, (1993).

[4] E. Jacquet-Lagrèze and Y. Siskos, 'Preference disaggregation: 20 years of mcda experience', *European Journal of Operational Research*, **130**(2), 233–245, (2001).

[5] A. Leroy, V. Mousseau, and M. Pirlot, 'Learning the parameters of a multiple criteria sorting method', in *Algorithmic Decision Theory*, eds., R. Brafman, F. Roberts, and A. Tsoukiàs, volume 6992, pp. 219–233. Springer, (2011).

[6] J. Liu, W. Ouerdane, and V. Mousseau, 'A metaheuristic approach for preference learning in multicriteria ranking based on reference points', in *Proceedings of the 2nd workshop "From multiple criteria Decision Aid to Preference Learning" (DA2PL)*, pp. 76–86, (2014).

[7] A-L. Olteanu, K. Belahcène, V. Mousseau, W. Ouerdane, A. Rolland, and J. Zheng, 'Preference elicitation for a ranking method based on multiple reference profiles', *4OR: A Quarterly Journal of Operations Research*, (2021).

[8] A. Rolland, 'Reference-based preferences aggregation procedures in multi-criteria decision making', *European Journal of Operational Research*, **225**(3), 479–486, (2013).

[9] B. Roy, 'The outranking approach and the foundations of ELECTRE methods', *Theory and Decision*, **31**, 49–73, (1991).

[10] A. Tversky and D. Kahneman, 'Loss aversion in riskless choice: A reference-dependent model', *The Quarterly Journal of Economics*, **106**(4), 1039–1061, (11 1991).

[11] L. Zheng, A. Rolland, and V. Mousseau, 'Preference elicitation for a ranking method based on multiple reference profiles', Technical report, Laboratoire Génie Industriel, Ecole Centrale Paris, Paris, (August 2012).