ACCOMPANYING LETTER

Dear RCRA Special Issue Editors,

in the following you will find the answers to the critcisms contained in the RCRA reviews.

Note also that the paper, with respect to the one presented at RCRA workshop, has been extended by analyzing the proposed algorithm also using an additional objective criterion.

Best Regards,
Valentino Santucci
Marco Baioletti
Alfredo Milani

Regarding review 1, the main criticisms and their respective answers are:

1. **Criticism:** *It would be interesting, if possible, to give a bit more details and intuitions about the reasons why differential mutation works on continuous problems (the so-called contour matching property) and why/if the proposed adaptation to a discrete permutation problem is expected to exploit the same ideas.*

   Answer: The “contour matching property” is vaguely defined also in the work where it is introduced. Actually, it claims that, thanks to the automatically adapted distribution of individuals' differences, DE population is prone to: (1) focus on newly discovered areas of the problem landscape that deserve to be “investigated”, (2) move computational resources (individuals) in the most promising areas. This is not easily formalizable and, to the best of our knowledge, there is no such formalization in literature. Anyway, our combinatorial definition of the differential mutation (DM) exactly mimics its continuous counterpart. Indeed, both continuous and permutation-based DMs (1) work in a metric space (the metric distances are, respectively, Euclidean and Kendal-Tau distance) and (2) perform the same geometric operations but by exploiting different algebraic concepts in order to implement them.

2. **Criticism**: *Surprisingly, the choice of the generator set is not motivated by the reasons why differential mutation is expected to work, but by consideration on the smoothness of the objective function landscape. Did you investigate the others generator sets T and I in some experiments?*

   Answer: The motivation bringing to the choice of ST as generating set have been reformulated in Section 2. Furthermore, it worths to note that, in order to experimentally investigate T and I, randomized “sorting” algorithms for these generators have to be implemented. Selection and insertion sort are not exactly what is needed. Anyway, we are currently working to obtain these algorithms.

3. **Criticism**: *In fact, my main comment and question about the paper is that, although differential mutation seems to be an interesting and original tool in this type of problem, the experiments do not really provide an evidence for that. The experiments seem to clearly indicate that the overall approach performs very well on the problem, but the differential evolution is only one of the ingredients of the approach beside the crossover, the way selection is performed, the use of a constructive heuristic in the initial population, the local search, the restart, etc. It would for instance be very informative to see how the approach behaves when differential mutation (in line 4 of algorithm 2) is replaced by, let's say, the random selection of an individual in the population or a purely random individual, or the result of a path relinking between two randomly selected individuals, ... Otherwise, in the end, even if the idea of differential mutation looks attractive, there is no real proof that it is the main ingredient to the success of the approach.*

   Answer: It worths to note that: (1) AGA (see paper references) employs our same crossover and their performances seems to be inferior, (2) population restart should be regarded as a sort of endemic component of differential evolution (DE); indeed, DE population is not able to evolve when all the individuals are the same, also in the original and continuous DE, (3) hybridization of population-based algorithms with local searches is a widely exploited technique in the field (see for example AGA or HGM-EDA), (4) the same is true for the construction of initial solutions by means of heuristic methods (see for example AGA or IG). Summarizing, local searches, restarts, and guided initialization are widely adopted in any other state-of-the-art algorithm. Anyway, we are working on an experiment that allows to objectively made more clear the contribution of our differential mutation.

4. **Criticism**: *As far as I understand it, the result of differential mutation for an individual xi, results in a totally different individual that is built from randomly selected individuals x_r0, x_r1, x_r2 (not including*

xi). This is quite different from the usual definition of mutation in classical evolutionary algorithm where a mutation is usually a rather small random change for a given individual so that the mutant keeps many characteristics of the original individual. I'm wondering if this the usual definition in Differential evolution algorithms but it makes the understanding of algorithm 2 a bit difficult as in fact, if I understand well, in line 4, DifferentialMutation(i) does not really depend on i.

Answer: Classical differential evolution explicitly denies to allow x_r0, x_r1 or x_r2 to be equal to x_i. It can be read on the original DE paper or every other fair DE description in literature. This rule is also quite reasonable since the generated mutant is the solution to be recombined with x_i during the crossover. Finally, DifferentialMutation(i) procedure has to know i to avoid to use it.

Regarding review 2, the main criticisms and their respective answers are:

1. **Criticism**: *The paper is clear, although the presentation could be improved by e.g. providing more details about the genetic algorithm approach and terminology (they are ok for an expert in the field, but may result obscure to non experts).*

   Answer: Our approach is based on Differential Evolution, not on genetic algorithms (although both are evolutionary algorithms). Anyway, to address this issue we need to have details of the points looking obscure to non experts.

2. **Criticism**: *The paper seems also correct. However, the statement of the paper, i.e. finding the best permutation to optimize the TFT, is partly solved. Indeed, over 79/120 considered benchmarks, i.e. 65% the proposed approach finds the best solution. In the other cases as far as the reader can understand, the results is worse than the best.*

   Answer: Also in other combinatorial problems it is quite unusual to find algorithms that performs the best on every possible instance of a large benchmark suite. On TFT, DEP is the best algorithm on more than half of the suite. Moreover, note that DEP is a meta-heuristic and not an exact algorithm (PFSP is NP hard). Anyway, maybe we do not understood the question.

3. **Criticism**: *The plots in Figure 2 are not properly explained. What are they about? What's on the x and y axes? How one could read it?*

   Answer: The plots have been removed and replaced by a textual argumentation on the choice of ST (see also answer to review 1, criticism 2).

4. **Criticism**: *The results in table 1 are summarized in section 4. However, there is no analysis of the reason certain results/behaviors have been obtained. For instance, do you have an explanation on the reason why for the 500jobs DEP does not show the lowest Friedman's average rank? Why for the 20 jobs all the algorithms have the same behavior? Why for the 500 jobs, DEP is outperformed by AGA and VNS_4?*

   Answer: A plausible reason of why DEP does not perform well on 500 jobs problems with respect to the competitors is explained in section 4.1 (“[...] The only weakness for DEP is found in problems with 500 jobs, [...]”). A possible reason of why for the 20 jobs all the algorithms have the same behavior is also explained in section 4.1 (“[...] This may suggest that the best values obtained are probably the optimal values for these instances. [...]”).

Regarding review 3, the main criticisms and their respective answers are:

1. **Criticism**: *Not being at all familiar with this area of research, it is puzzling to see the use of an explicitly-called 'bubble sort' algorithm. I assume there is a reason for this choice, but it is not clarified in the paper.*

   Answer: The name “bubble sort” is motivated by the fact that algorithm 1 is actually a randomized version of bubble sort. The motivations of why it is introduced are explained just after the algorithm in Section 2. Indeed, RandBS sorts the permutation in input (but it can sort every array of comparable elements) and, as a second result, it returns the sequence of adjacent swaps made to sort the input permutation.

Regarding review 4, the main criticisms and their respective answers are:

1. **Criticism**: *The figures are incorrectly numbered, with the text referring to Figures 1a and 1b while the actual caption showing Fig. 2. It is also not entirely clear what they are presenting.*

   Answer: The figures have been removed and replaced by a textual argumentation (see also the answer to review 1, criticism 2).

2. **Criticism**: *Section 3 also merits expanding all the respective sections to explain the rationale and the general idea rather than just referring to other sources, because this makes the paper unreadable if the reader is not already familiar with all the mentioned techniques and algorithms.*

   Answer: jDE has been briefly described. Crossover, selection and local search looks to be already described. Do you need also a description of the (well known) constructive heuristics used in the initialization?