A Comparison of Generating Sets for the Algebraic Differential Evolution

M. Baioletti [1] V. Santucci [2] A. Milani [1] U. Bartoccini [2]

[1]Department of Mathematics and Computer Science
University of Perugia
Perugia, Italy

[2]University for Foreigners of Perugia
Perugia, Italy

July, 13 2019
ECPERM workshop @ GECCO 2019

Outline

Introduction

- The aim of this paper is twofold
- We have extended ADEP (Algebraic Differential Evolution for Permutation) by adding a new generating set based on the reversal operation
- We have made a comparative empirical analysis on generating sets for ADEP applied to TSP

Algebraic Framework

- Algebraic Differential Evolution for Permutation (ADEP) is a discrete DE based on the Algebraic Framework (AF) introduced and studied in [BSM14,BSM16]
- The mutation operator of ADEP

  $$v \leftarrow x_{r_0} \oplus F \odot (x_{r_1} \ominus x_{r_2})$$

  works directly on permutations
- The AF was successfully applied to famous permutation based problems, like PFSP, LOP, and LOPCC, and to other form of problems, like Bayesian Network Structural Learning and Multidimensional Two Way Number Partitioning Problem

Finitely generated groups

- The key idea of AF is that in many problems, the search space X is a Finitely Generated Group with respect to a solution composition operator $\circ$ and a generating set G
- Hence
  - $\circ$ is a binary operation on X satisfying the group properties, i.e., closure, associativity, existence of a neutral element (or identity e), and invertibility (x^{-1}); and
  - $G \subseteq X$ is a finite generating set of the group, i.e., any $x \in X$ has a (not necessarily unique) minimal-length decomposition $\langle g_1, \ldots, g_l \rangle$, with $g_1, \ldots, g_l \in G$, i.e.,

$$x = g_1 \circ \cdots \circ g_l$$

Operators

- Using $(X, \circ, G)$ we can provide the formal definitions of the operators $\oplus, \ominus, \odot$ according to the principles of the Algebraic Framework
- Let $x, y \in X$ and $\langle g_1, \ldots, g_k, \ldots, g_{|x|} \rangle$ be a minimal decomposition of x, then
    - $x \oplus y := x \circ y$
    - $x \ominus y := y^{-1} \circ x$
    - $F \odot x := g_1 \circ \cdots \circ g_k$, with $k = \lceil F \cdot |x| \rceil$ and $F \in [0, 1]$

Permutations

- The set of all permutations of $[n] = \{1, \ldots, n\}$, together with the permutation composition $\circ : [n] \times [n] \to [n]$, form the Symmetric group $\mathcal{S}(n)$
- $\mathcal{S}(n)$ has many generating sets
- The most famous are ASW (based on *adjacent swap moves*), EXC (based on *exchange moves*), and INS (based on *insertion moves*)

Generating sets

- $ASW = \{\sigma_i : 1 \leq i < n\}$
  $\pi \circ \sigma_i$ swaps the i-th and $(i+1)$-th items of π. For instance, let $\pi = \langle 3, 5, 2, 4, 1 \rangle$, thus
  $\pi \circ \sigma_3 = \langle 3, 5, \underline{2, 4}, 1 \rangle \circ \langle 1, 2, 4, 3, 5 \rangle = \langle 3, 5, \underline{4, 2}, 1 \rangle$.
- $EXC = \{\epsilon_{ij} : 1 \leq i < j \leq n\}$
  $\pi \circ \epsilon_{ij}$ swaps the i-th and j-th items of π. For instance,
  $\pi \circ \epsilon_{2,5} = \langle 3, \underline{5}, 2, 4, \underline{1} \rangle \circ \langle 1, 5, 3, 4, 2 \rangle = \langle 3, \underline{1}, 2, 4, \underline{5} \rangle$.
- $INS = \{\iota_{ij} : 1 \leq i, j \leq n\}$
  $\pi \circ \iota_{ij}$ shifts the i-th item in π to position j. For instance,
  $\pi \circ \iota_{3,5} = \langle 3, 5, \underline{2}, 4, \underline{1} \rangle \circ \langle 1, 2, 4, 5, 3 \rangle = \langle 3, 5, 4, 1, \underline{2} \rangle$.

Randomized decomposers

- A decomposition of a permutation $\pi \in \mathcal{S}_n$ can be obtained by a sorting algorithm that only performs moves from the chosen generating set *ASW*, *EXC*, or *INS*
- Optimal randomized decomposers for *ASW*, *EXC*, and *INS* have been implemented by means of generalized and randomized variants of, respectively, the bubble-sort, selection-sort, and insertion-sort algorithms
- They have been called *RandBS*, *RandSS*, *RandIS* and each one exploits a different algebraic property of permutations

Reversal

- A reversal $R(i, j)$, with $1 \leq i < j \leq n$, is an operation which, given a permutation $\pi \in \mathcal{S}(n)$, reverses the items in the interval $[i, j]$ of the ordering represented by π.
- For instance, the reversal $R(4, 7)$ applied to $\pi = \langle 4, 9, 5, \underline{3, 8, 2, 1}, 7, 6 \rangle$ produces the permutation $\langle 4, 9, 5, \underline{1, 2, 8, 3}, 7, 6 \rangle$.
- Reversals are widely studied in computational biology
- The reversals form the generating set $REV = \{\rho_{ij} : 1 \leq i < j \leq n\}$
- There is a bijective correspondance between reversals and 2-OPT moves

Sorting by reversal

- Any permutation can be decomposed as a composition of at most $n - 1$ reversals
- Finding the minimal decomposition is a NP-hard problem
- There exists fast approximated algorithm which produce a decomposition whose length is at most a small multiple of the minimal decomposition length
- Among them, one of the most suitable to be randomized is the greedy algorithm *KS* proposed by Kecegioglu and Sankoff
- Its approximation factor guaranteed is 2, though, the decomposition is often very close to the optimal length

Randomized decomposer for reversal

- *KS* uses the concepts of *breakpoint* and *decreasing strip*.
- A breakpoint of a permutation π is a position i such that $|\pi(i) - \pi(i+1)| > 1$
- $\pi(i), \pi(i+1), \ldots, \pi(j-1), \pi(j)$ is a decreasing strip if its items are in decreasing order

KS algorithm

```
function $\mathrm{KS}(\pi \in \mathcal{S}(n))$
    $l := 0$
    while $\pi$ contains at least a breakpoint do
        $l := l + 1$
        Let $\rho_{i_l,j_l}$ a reversal that removes the most breakpoints of $\pi$, resolving ties among those that remove one breakpoint in favour of reversals that leave a decreasing strip
        $\pi := \pi \circ \rho_{i_l,j_l}$
    end while
    return $\langle \rho_{i_1,j_1}, \ldots, \rho_{i_l,j_l} \rangle$
end function
```

RandRS

- We have randomized *KS* in order to use it in DEP
- *RandRS* works by iteratively choosing (and applying to π) anyone of the reversals $\rho \in REV$ satisfying one of the following properties in priority order:

  (P1) $nb(\pi \circ \rho) = nb(\pi) - 2$

  (P2) $nb(\pi \circ \rho) = nb(\pi) - 1$ and $\pi \circ \rho$ has at least one decreasing strip

  (P3) $nb(\pi \circ \rho) = nb(\pi) - 1$

  (P4) $nb(\pi \circ \rho) = nb(\pi)$

  where $nb(\pi)$ denotes the number of breakpoints in the permutation π
- The reversal ρ is randomly selected by means of 4 *reservoir sampling* variables

RandRS2

- Although randomized, *RandRS* is anyway a greedy algorithm that, at each iteration, selects a reversal according to the priorities among the rules (P1–P4).
- It may happen that sometimes only one reversal can be chosen and this aspect may lead to a diversity lost problem when *RandRS* is embedded into an evolutionary algorithm
- For this reason, we have also devised another simpler randomized decomposer, called *RandRS2*, which iteratively chooses a random reversal satisfying anyone of the properties (P1–P4) without any priority among them

Properties of Generating sets

Generating Set	Cardinality	Diameter	Time Complexity
ASW	$n-1$	$\binom{n}{2}$	$\Theta(n^2)$
EXC	$\binom{n}{2}$	$n-1$	$\Theta(n)$
INS	$(n-1)^2$	$n-1$	$\Theta(n^2)$
REV	$\binom{n}{2}$	$\leq n-1$	$\Theta(n^2)$

Feature of ADEP

- ADEP directly evolves a population $\{\pi_1, \ldots, \pi_N\}$ of N permutations.
- The last city in the TSP tour is fixed and we use a permutation of the remaining $n-1$ cities in order to encode a TSP solution
- jDE is used in order to automatically adapt the value of the parameter $F \in (0, 1]$.
- After some preliminary experiments, three crossover operators have been selected: *MPX* (Maximal Preservative CRX), *ER* (Edge Recombination) and *OX1* (Order Crossover).
- The crowding selection scheme is adopted in order to slow down population convergence
- A restart mechanism is used when all the population individuals converge to the same solution.
- At the end of the evolution, a local search operator, based on the 2-OPT neighborhood, is applied to the best solution

ADEP

```
Randomly initialize the population $\{\pi_1, \ldots, \pi_N\}$
**for** $gen \leftarrow 1$ **to** $MaxGen$ **do**
    **for** $i \leftarrow 1$ **to** $N$ **do**
        $\nu_i \leftarrow \text{DifferentialMutation}(i, \{\pi_1, \ldots, \pi_N\}, F)$
        $\upsilon_i \leftarrow \text{Crossover}(\pi_i, \nu_i)$
        Evaluate $f(\upsilon_i)$
    **end for**
    Select the population for the next iteration
    **if** restart criterion is satisfied **then**
        Reinitialize the population
    **end if**
**end for**
Apply local search to the best individual $\pi_{best}$
```

Experimental settings

- We have conducted an experimental analysis on 25 TSP instances selected from TSPLIB, whose size varies in the range $n \in [14, 100]$.
- The main goal of this analysis is to investigate the performances of the different generating sets when ADEP is applied to the TSP
- Moreover, we want to compare RandRS with RandRS2 (for sake of simplicity, we consider it as an additional generating set *REV2*)
- By combining the 5 generating sets and the 3 crossovers, 15 ADEP configurations have been considered
- All the configurations use a population size of 100 individuals and have been executed 20 times per instance, with a 100 000 generations as termination criterion

Comparing the generating sets

	ASW	EXC	INS	REV	REV2
Avg ARPD	7.35	3.97	2.89	2.72	2.80
Avg Rank	11.26	7.65	7.34	6.92	6.83

- In the table we have aggregated the average ARPD and the average ranks for each generating set.
- *REV* and *REV2* are the best best generating sets, while *INS* is very close to them and *EXC* and *ASW* are clearly worse, either considering the average ARPD or the average rank
- The comparison between *REV* and *REV2* does not produce a clear winner, indeed *REV* has the best ARPD value, while *REV2* reaches the smallest average rank

Conclusions

- In this work we have studied the performances of ADEP, equipped with four different generating sets, on the TSP
- In particular, we have analyzed the three already proposed generating sets *ASW*, *EXC* and *INS* and the newly introduced generating set *REV*, based on the reversal moves
- Two implementations of the *REV* randomized decomposer have been considered
- The experimental analysis shows that the reversal moves lead to better results, whichever crossover operator is used

Future works

- As a future work, we intend to study the impact of the generating sets in other permutation-based optimization problems
- Furthermore, we are also interested in making a similar analysis with other discrete algorithm based on the algebraic framework like, for instance, the algebraic PSO