Exploiting the algebraic properties of the permutation space in Evolutionary Computation

Valentino Santucci

University for Foreigners of Perugia, Italy

Keynote Talk @ ECPERM Workshop - GECCO 2020 - 8th July 2020

Outline

- Permutations in EC
- Permutations in Group Theory
- Algebraic properties of combinatorial spaces
- An algebraic framework for EC
- Algebraic EAs and operators
- Practical applications
- Conclusions and open questions

Why permutations in EC?

- **Represent solutions** of important COPs

  Ex: $\pi = \left\langle \begin{array}{c} 12345678 \\ 47231586 \end{array} \right\rangle$

Why permutations in EC?

- **Represent solutions** of important COPs
  Ex: $\pi = \left\langle \begin{array}{c} 12345678 \\ 47231586 \end{array} \right\rangle$
- **Orderings of items**: $\pi(i)$ is the item in position i
  Ex: item 4 is in position 1, item 7 is in position 2, ...
  Useful in the *Permutation Flowshop Scheduling Problem* (PFSP)

Why permutations in EC?

- **Represent solutions** of important COPs
  Ex: $\pi = \left\langle \begin{array}{c} 12345678 \\ 47231586 \end{array} \right\rangle$
- **Orderings of items**: $\pi(i)$ is the item in position i
  Ex: item 4 is in position 1, item 7 is in position 2, ...
  Useful in the *Permutation Flowshop Scheduling Problem* (PFSP)
- **Rankings on items**: $\pi(i)$ is the position of the item i
  Ex: item 1 is in position 4, item 2 is in position 7, ...
  Useful in the *Linear Ordering Problem* (LOP)

Why permutations in EC?

- **Represent solutions** of important COPs
  Ex: $\pi = \left\langle \begin{array}{c} 12345678 \\ 47231586 \end{array} \right\rangle$
- **Orderings of items**: $\pi(i)$ is the item in position i
  Ex: item 4 is in position 1, item 7 is in position 2, ...
  Useful in the *Permutation Flowshop Scheduling Problem* (PFSP)
- **Rankings on items**: $\pi(i)$ is the position of the item i
  Ex: item 1 is in position 4, item 2 is in position 7, ...
  Useful in the *Linear Ordering Problem* (LOP)
- **Bijective functions**: the item $\pi(i)$ is assigned to the item i
  Ex: item 4 is assigned to item 1, item 7 is assigned to item 2, ...
  Useful in the *Quadratic Assignment Problem* (QAP)

Why permutations in EC?

- **Represent solutions** of important COPs
  Ex: $\pi = \left\langle \begin{array}{c} 12345678 \\ 47231586 \end{array} \right\rangle$
- **Orderings of items**: $\pi(i)$ is the item in position i
  Ex: item 4 is in position 1, item 7 is in position 2, ...
  Useful in the *Permutation Flowshop Scheduling Problem* (PFSP)
- **Rankings on items**: $\pi(i)$ is the position of the item i
  Ex: item 1 is in position 4, item 2 is in position 7, ...
  Useful in the *Linear Ordering Problem* (LOP)
- **Bijective functions**: the item $\pi(i)$ is assigned to the item i
  Ex: item 4 is assigned to item 1, item 7 is assigned to item 2, ...
  Useful in the *Quadratic Assignment Problem* (QAP)
- **Routing among items**: item $\pi(i)$ is connected to item $\pi(i+1)$
  Ex: item 4 is connected to item 7, item 7 is connected to item 2, ...
  Useful in the *Traveling Salesman Problem* (TSP)

Neighborhoods in the permutation space

- They allow to build-up *local search* based meta-heuristics

Neighborhoods in the permutation space

- They allow to build-up *local search* based meta-heuristics
- **ASW**: swaps between adjacent items
  Ex: $\langle 53241 \rangle$ and $\langle 53421 \rangle$ are neighbors
  It modifies the relative order of only two items

Neighborhoods in the permutation space

- They allow to build-up *local search* based meta-heuristics
- **ASW**: swaps between adjacent items
  Ex: $\langle 53241 \rangle$ and $\langle 53421 \rangle$ are neighbors
  It modifies the relative order of only two items
- **EXC**: swaps between generic items
  Ex: $\langle 53241 \rangle$ and $\langle 13245 \rangle$ are neighbors
  It modifies the absolute positions of only two items

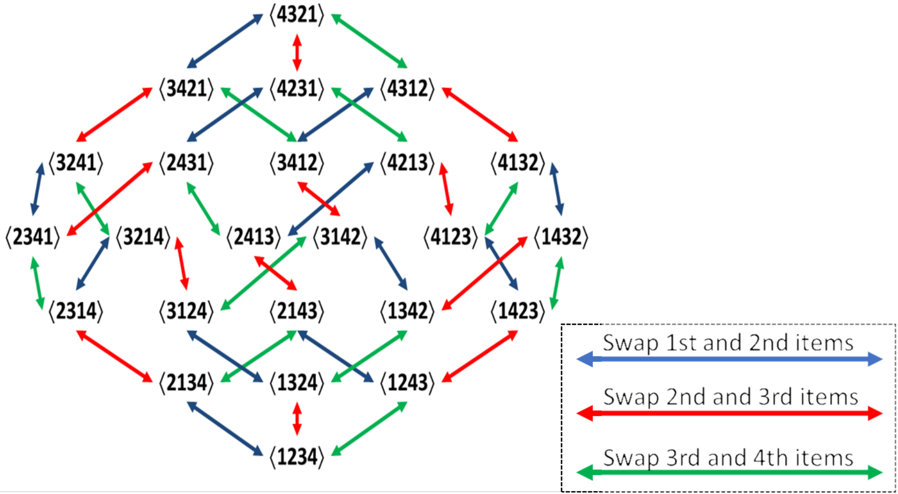
Neighborhoods in the permutation space

- They allow to build-up *local search* based meta-heuristics
- **ASW**: swaps between adjacent items
  Ex: $\langle 53241 \rangle$ and $\langle 53421 \rangle$ are neighbors
  It modifies the relative order of only two items
- **EXC**: swaps between generic items
  Ex: $\langle 53241 \rangle$ and $\langle 13245 \rangle$ are neighbors
  It modifies the absolute positions of only two items
- **INS**: insertions or shifts of an item
  Ex: $\langle 53241 \rangle$ and $\langle 52431 \rangle$ are neighbors
  It modifies the relative order of a bunch of items but with respect to only one item

Neighborhoods in the permutation space

- They allow to build-up *local search* based meta-heuristics
- **ASW**: swaps between adjacent items
  Ex: $\langle 53241 \rangle$ and $\langle 53421 \rangle$ are neighbors
  It modifies the relative order of only two items
- **EXC**: swaps between generic items
  Ex: $\langle 53241 \rangle$ and $\langle 13245 \rangle$ are neighbors
  It modifies the absolute positions of only two items
- **INS**: insertions or shifts of an item
  Ex: $\langle 53241 \rangle$ and $\langle 52431 \rangle$ are neighbors
  It modifies the relative order of a bunch of items but with respect to only one item
- **REV**: reversals of a chunk of items
  Ex: $\langle 53241 \rangle$ and $\langle 54231 \rangle$ are neighbors
  It modifies only two connections

Permutation space under the ASW neighborhood



Group Theory: permutations form a group

- **Composition** of permutations
  $\tau = \pi \circ \rho$ is defined as $\tau(i) = \pi(\rho(i))$ for $i = 1, \ldots, n$

  Ex: $\left\langle \begin{matrix} 1234 \\ 4213 \end{matrix} \right\rangle \circ \left\langle \begin{matrix} 1234 \\ 2143 \end{matrix} \right\rangle = \left\langle \begin{matrix} 1234 \\ 2431 \end{matrix} \right\rangle$

Group Theory: permutations form a group

- **Composition** of permutations
  $\tau = \pi \circ \rho$ is defined as $\tau(i) = \pi(\rho(i))$ for $i = 1, \dots, n$

  Ex: $\left\langle \begin{matrix} 1234 \\ 4213 \end{matrix} \right\rangle \circ \left\langle \begin{matrix} 1234 \\ 2143 \end{matrix} \right\rangle = \left\langle \begin{matrix} 1234 \\ 2431 \end{matrix} \right\rangle$

- **Identity** permutation $e = \langle 12 \dots n \rangle$ is the neutral element

Group Theory: permutations form a group

- **Composition** of permutations
  $\tau = \pi \circ \rho$ is defined as $\tau(i) = \pi(\rho(i))$ for $i = 1, \ldots, n$
  Ex: $\left\langle \begin{matrix} 1234 \\ 4213 \end{matrix} \right\rangle \circ \left\langle \begin{matrix} 1234 \\ 2143 \end{matrix} \right\rangle = \left\langle \begin{matrix} 1234 \\ 2431 \end{matrix} \right\rangle$

- **Identity** permutation $e = \langle 12 \ldots n \rangle$ is the neutral element

- **Inverse** permutation: there exists a unique π^{-1} s.t. $\pi \circ \pi^{-1} = e$
  Ex: $\pi = \left\langle \begin{matrix} 1234 \\ 4213 \end{matrix} \right\rangle \to \left\langle \begin{matrix} 4213 \\ 1234 \end{matrix} \right\rangle \to \left\langle \begin{matrix} 1234 \\ 3241 \end{matrix} \right\rangle \to \left\langle \begin{matrix} 1234 \\ 3241 \end{matrix} \right\rangle = \pi^{-1}$

Group Theory: permutations form a group

- **Composition** of permutations
  $\tau = \pi \circ \rho$ is defined as $\tau(i) = \pi(\rho(i))$ for $i = 1, \ldots, n$

  Ex: $\left\langle \begin{matrix} 1234 \\ 4213 \end{matrix} \right\rangle \circ \left\langle \begin{matrix} 1234 \\ 2143 \end{matrix} \right\rangle = \left\langle \begin{matrix} 1234 \\ 2431 \end{matrix} \right\rangle$

- **Identity** permutation $e = \langle 12 \ldots n \rangle$ is the neutral element

- **Inverse** permutation: there exists a unique π^{-1} s.t. $\pi \circ \pi^{-1} = e$

  Ex: $\pi = \left\langle \begin{matrix} 1234 \\ 4213 \end{matrix} \right\rangle \to \left\langle \begin{matrix} 4213 \\ 1234 \end{matrix} \right\rangle \to \left\langle \begin{matrix} 1234 \\ 3241 \end{matrix} \right\rangle \to \left\langle \begin{matrix} 1234 \\ 3241 \end{matrix} \right\rangle = \pi^{-1}$

- The **Symmetric group** of n items is denoted by $\mathcal{S}_n$

How neighborhoods and algebra relate to each other?

- Under **ASW**, all the neighbors of $\langle 2413 \rangle$ are:
  - $\langle 4213 \rangle$
  - $\langle 2143 \rangle$
  - $\langle 2431 \rangle$

How neighborhoods and algebra relate to each other?

- Under **ASW**, all the neighbors of $\langle 2413 \rangle$ are:
  - $\langle 4213 \rangle$
  - $\langle 2143 \rangle$
  - $\langle 2431 \rangle$
- They can be algebraically obtained by:
  - $\langle 2413 \rangle \circ \langle 2134 \rangle = \langle 4213 \rangle$
  - $\langle 2413 \rangle \circ \langle 1324 \rangle = \langle 2143 \rangle$
  - $\langle 2413 \rangle \circ \langle 1243 \rangle = \langle 2431 \rangle$

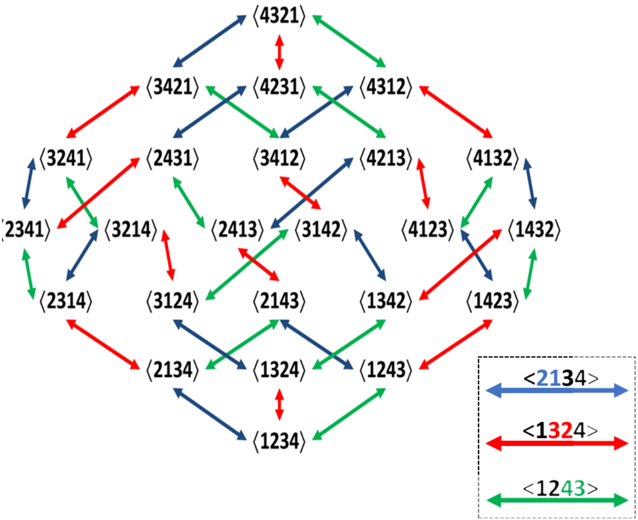
How neighborhoods and algebra relate to each other?

- Under **ASW**, all the neighbors of $\langle 2413 \rangle$ are:
  - $\langle 4213 \rangle$
  - $\langle 2143 \rangle$
  - $\langle 2431 \rangle$

- They can be algebraically obtained by:
  - $\langle 2413 \rangle \circ \langle 2134 \rangle = \langle 4213 \rangle$
  - $\langle 2413 \rangle \circ \langle 1324 \rangle = \langle 2143 \rangle$
  - $\langle 2413 \rangle \circ \langle 1243 \rangle = \langle 2431 \rangle$

- Let $\sigma_1 = \langle 2134 \rangle$, $\sigma_2 = \langle 1324 \rangle$, $\sigma_3 = \langle 1243 \rangle$ and $\mathtt{ASW} = \{\sigma_1, \sigma_2, \sigma_3\}$ then:
  - $\mathtt{ASW} \subset \mathcal{S}_n$
  - $\mathtt{ASW}$ generates all the permutations in $\mathcal{S}_n$
  - any permutation can be factorized as a product of generators in ASW
  - $\mathcal{S}_n$ is a **finitely generated group**

Cayley graph for the ASW generating set



All the generating sets

- $\texttt{ASW} = \{\sigma_i : 1 \leq i < n\}$, where σ_i is the identity permutation with the items i and $i+1$ exchanged.
  Ex: $\langle 13245 \rangle$ swaps 2nd and 3rd items
  Distance induced: **Kendall's-τ distance**

All the generating sets

- $\texttt{ASW} = \{\sigma_i : 1 \leq i < n\}$, where σ_i is the identity permutation with the items i and $i+1$ exchanged.
  Ex: $\langle 13245 \rangle$ swaps 2nd and 3rd items
  Distance induced: **Kendall's-τ distance**
- $\texttt{EXC} = \{\epsilon_{ij} : 1 \leq i < j \leq n\}$, where ϵ_{ij} is the identity permutation with the items i and j exchanged.
  Ex: $\langle 14325 \rangle$ swaps 2nd and 4th items
  Distance induced: **Cayley distance**

All the generating sets

- $\texttt{ASW} = \{\sigma_i : 1 \leq i < n\}$, where σ_i is the identity permutation with the items i and $i+1$ exchanged.
  Ex: $\langle 13245 \rangle$ swaps 2nd and 3rd items
  Distance induced: **Kendall's-τ distance**
- $\texttt{EXC} = \{\epsilon_{ij} : 1 \leq i < j \leq n\}$, where ϵ_{ij} is the identity permutation with the items i and j exchanged.
  Ex: $\langle 14325 \rangle$ swaps 2nd and 4th items
  Distance induced: **Cayley distance**
- $\texttt{INS} = \{\iota_{ij} : 1 \leq i, j \leq n\}$, where ι_{ij} is the identity permutation where the item i is shifted to position j.
  Ex: $\langle 14235 \rangle$ shifts 4th item to 2nd position
  Distance induced: **Ulam distance**

All the generating sets

- $\texttt{ASW} = \{\sigma_i : 1 \leq i < n\}$, where σ_i is the identity permutation with the items i and $i+1$ exchanged.
  Ex: $\langle 13245 \rangle$ swaps 2nd and 3rd items
  Distance induced: **Kendall's-τ distance**
- $\texttt{EXC} = \{\epsilon_{ij} : 1 \leq i < j \leq n\}$, where ϵ_{ij} is the identity permutation with the items i and j exchanged.
  Ex: $\langle 14325 \rangle$ swaps 2nd and 4th items
  Distance induced: **Cayley distance**
- $\texttt{INS} = \{\iota_{ij} : 1 \leq i, j \leq n\}$, where ι_{ij} is the identity permutation where the item i is shifted to position j.
  Ex: $\langle 14235 \rangle$ shifts 4th item to 2nd position
  Distance induced: **Ulam distance**
- $\texttt{REV} = \{\rho_{ij} : 1 \leq i < j \leq n\}$, where ρ_{ij} is the identity permutation where the chunk between positions i and j is reversed.
  Ex: $\langle 14325 \rangle$ reverses the chunk from 2nd to 4th positions
  Distance induced: **reversals' distance**

Relations among the generating sets

- $\mathtt{ASW}$ is a proper subset of $\mathtt{EXC}, \mathtt{INS}, \mathtt{REV}$
- $\mathtt{INS} \cap \mathtt{EXC} = \mathtt{ASW}$
- $\mathtt{INS} \cap \mathtt{REV} = \mathtt{ASW}$
- $\mathtt{EXC} \cap \mathtt{REV} = \mathtt{ASW} \cup \{\epsilon_{ij} \in \mathtt{EXC} : |i - j| = 2\}$
- $\mathtt{INS} = \mathtt{INS}_{\mathtt{bw}} \cup \mathtt{INS}_{\mathtt{fw}}$
- $\mathtt{INS}_{\mathtt{bw}} \cap \mathtt{INS}_{\mathtt{fw}} = \mathtt{ASW}$

Relations among the generating sets

- $\mathtt{ASW}$ is a proper subset of $\mathtt{EXC}, \mathtt{INS}, \mathtt{REV}$
- $\mathtt{INS} \cap \mathtt{EXC} = \mathtt{ASW}$
- $\mathtt{INS} \cap \mathtt{REV} = \mathtt{ASW}$
- $\mathtt{EXC} \cap \mathtt{REV} = \mathtt{ASW} \cup \{\epsilon_{ij} \in \mathtt{EXC} : |i - j| = 2\}$
- $\mathtt{INS} = \mathtt{INS}_{\mathtt{bw}} \cup \mathtt{INS}_{\mathtt{fw}}$
- $\mathtt{INS}_{\mathtt{bw}} \cap \mathtt{INS}_{\mathtt{fw}} = \mathtt{ASW}$

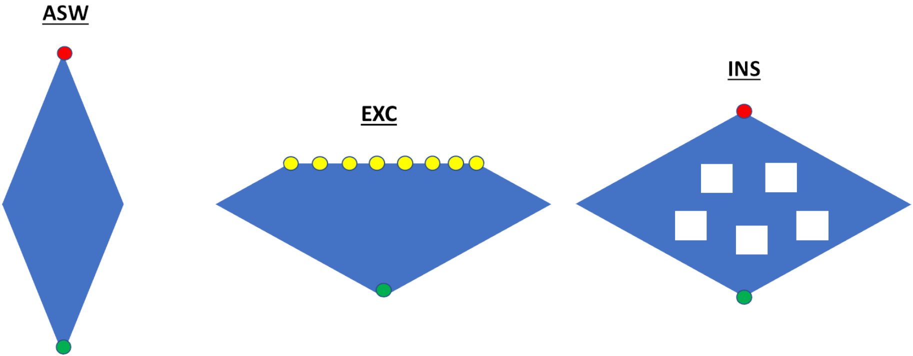
- Useful for:
  - Designing Variable Neighborhood Search algorithms
  - Designing perturbation step in Iterative Local Search algorithms
  - A priori smoothness estimation in Fitness Landscape Analysis

Properties of the generating sets

- **Cardinality** = number of neighbors
- **Diameter** of the Cayley graph (search space)
- **Number of longest permutations** (whose shortest-path distance from the identity equals the diameter)
- **Abstract convexity**: any permutation resides in a shortest path between the identity and a longest permutation?
- **Lattice** structure: *meet* and *join* are well defined?

GSet	Card.	Diameter	#Longest Perm.	Abst.Convex	Lattice
ASW	$n-1$	$\binom{n}{2}$	1	Yes	Yes
EXC	$\binom{n}{2}$	$n-1$	$(n-1)!$	Yes	No
INS	$(n-1)^2$	$n-1$	1	No	No
REV	$\binom{n}{2}$	$\leq n-1$	?	?	?

Visual intuitions about space structures



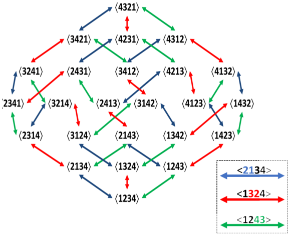
is the identity permutation

is the reversed identity

is a cyclic permutation

is a «hole» (due to «non-convexity»)

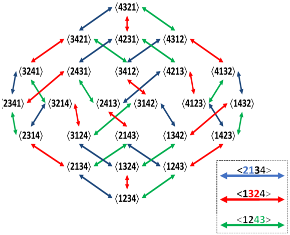
From simple search moves to composite search moves



- **Simple search move** = a single generator =
  = a *special* permutation
- **Composite search move** = a sequence of *simple search moves* = a sequence of generators =
  = **a generic permutation**
  (since a generating set generates the group)

- **Difference permutation** = composition of the **generators in a shortest path**

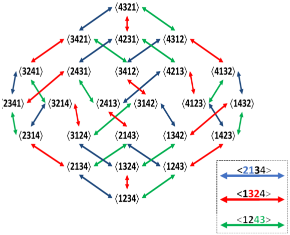
From simple search moves to composite search moves



- **Simple search move** = a single generator =
  = a *special* permutation
- **Composite search move** = a sequence of *simple search moves* = a sequence of generators =
  = **a generic permutation**
  (since a generating set generates the group)

- **Difference permutation** = composition of the **generators in a shortest path**
- **Minimal factorization** of the difference permutation contains generators in a shortest path

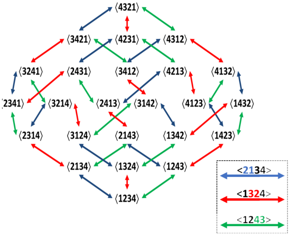
From simple search moves to composite search moves



- **Simple search move** = a single generator = = a *special* permutation
- **Composite search move** = a sequence of *simple search moves* = a sequence of generators = = **a generic permutation** (since a generating set generates the group)

- **Difference permutation** = composition of the **generators in a shortest path**
- **Minimal factorization** of the difference permutation contains generators in a shortest path
- **Distance** = length of a shortest path = length of a minimal factorization of the difference permutation

From simple search moves to composite search moves



- **Simple search move** = a single generator = = a *special* permutation
- **Composite search move** = a sequence of *simple search moves* = a sequence of generators = = **a generic permutation** (since a generating set generates the group)

- **Difference permutation** = composition of the **generators in a shortest path**
- **Minimal factorization** of the difference permutation contains generators in a shortest path
- **Distance** = length of a shortest path = length of a minimal factorization of the difference permutation
- **Single representation for solutions and differences**
- **Strong analogy with points and vectors in the Euclidean space**

Equations of Motion in Continuous EAs

Can be consistently redefined for discrete spaces?

Differential Evolution (DE)

$$\boldsymbol{u} \leftarrow \boldsymbol{x_0} + F \cdot (\boldsymbol{x_1} - \boldsymbol{x_2})$$

where: $\boldsymbol{u}, \boldsymbol{x_0}, \boldsymbol{x_1}, \boldsymbol{x_2} \in \mathbb{R}^n$, while $F \in \mathbb{R}^+$

Particle Swarm Optimization (PSO)

$$v \leftarrow w \cdot \boldsymbol{v} + c_1 r_1 \cdot (\boldsymbol{p} - \boldsymbol{x}) + c_2 r_2 \cdot (\boldsymbol{g} - \boldsymbol{x})$$

$$\boldsymbol{x} \leftarrow \boldsymbol{x} + \boldsymbol{v}$$

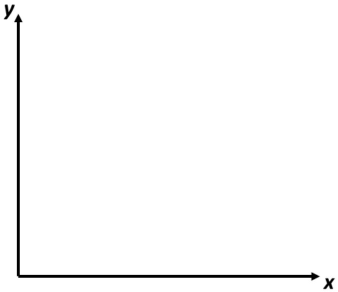
where: $\boldsymbol{x}, \boldsymbol{v}, \boldsymbol{p}, \boldsymbol{g} \in \mathbb{R}^n$, while $w, c_1, c_2, r_1, r_2 \in \mathbb{R}^+$

Continuous DE vs Algebraic DE

Differential mutation in action

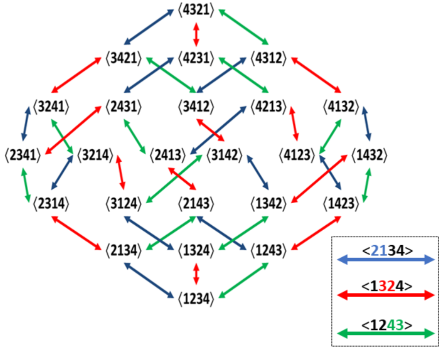
Continuous DE

$$u \leftarrow x_0 + F \cdot (x_1 - x_2)$$



Algebraic DE

$$\upsilon \leftarrow \pi_0 \oplus F \odot (\pi_1 \ominus \pi_2)$$

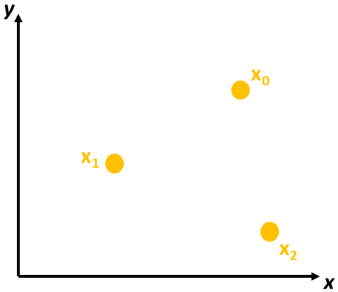


Continuous DE vs Algebraic DE

Differential mutation in action

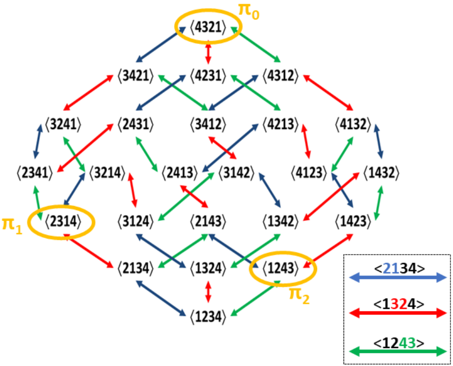
Continuous DE

$$u \leftarrow x_0 + F \cdot (x_1 - x_2)$$



Algebraic DE

$$\upsilon \leftarrow \pi_0 \oplus F \odot (\pi_1 \ominus \pi_2)$$

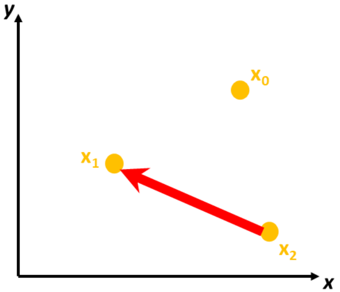


Continuous DE vs Algebraic DE

Differential mutation in action

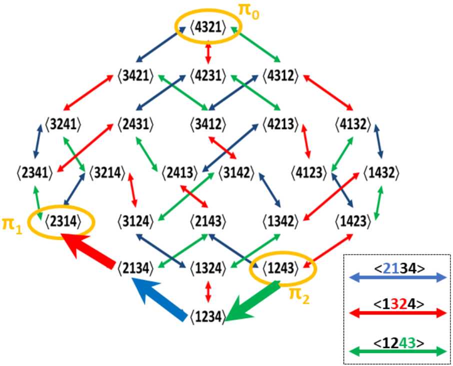
Continuous DE

$$u \leftarrow x_0 + F \cdot (x_1 - x_2)$$



Algebraic DE

$$\upsilon \leftarrow \pi_0 \oplus F \odot (\pi_1 \ominus \pi_2)$$

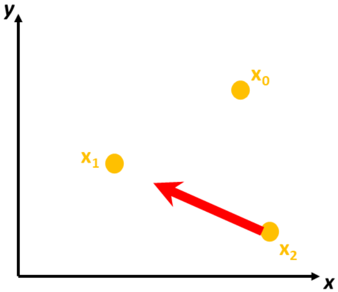


Continuous DE vs Algebraic DE

Differential mutation in action

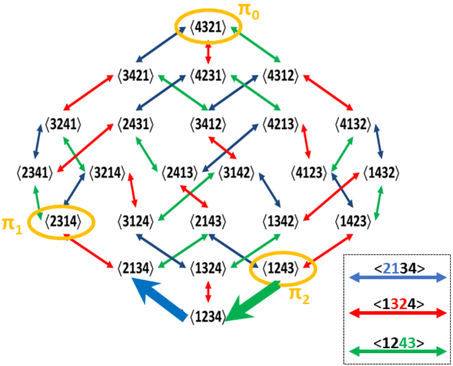
Continuous DE

$$\boldsymbol{u} \leftarrow \boldsymbol{x}_0 + \boldsymbol{F} \cdot (\boldsymbol{x}_1 - \boldsymbol{x}_2) \qquad F = 2/3$$



Algebraic DE

$$\upsilon \leftarrow \pi_0 \oplus \boldsymbol{F} \odot (\pi_1 \ominus \pi_2) \qquad F = 2/3$$

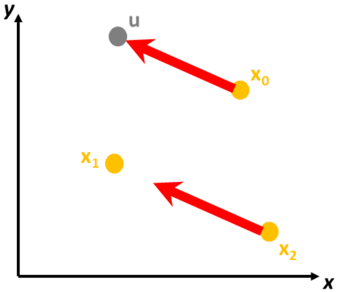


Continuous DE vs Algebraic DE

Differential mutation in action

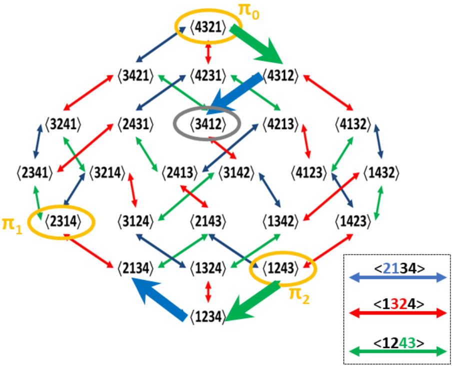
Continuous DE

$$\boldsymbol{u} \leftarrow \boldsymbol{x_0} + \boldsymbol{F} \cdot (\boldsymbol{x_1} - \boldsymbol{x_2}) \qquad F = 2/3$$



Algebraic DE

$$\upsilon \leftarrow \pi_0 \oplus \boldsymbol{F} \odot (\pi_1 \ominus \pi_2) \qquad F = 2/3$$



Algebraic Differential Evolution for Permutations

Classical continuous DE

The key operation of DE is the differential mutation which generates a mutant $u \in \mathbb{R}^n$ according to

$$u \leftarrow x_0 + F \cdot (x_1 - x_2)$$

where $x_0, x_1, x_2 \in \mathbb{R}^n$ are three distinct population individuals and $F > 0$ is the *scale factor parameter* of DE.

Algebraic DE for Permutations (ADEP)

The key operation of ADEP is the differential mutation which generates a mutant $\upsilon \in \mathcal{S}_n$ according to

$$\upsilon \leftarrow \pi_0 \oplus F \odot (\pi_1 \ominus \pi_2)$$

where $\pi_0, \pi_1, \pi_2 \in \mathcal{S}_n$ are three distinct population individuals and $F > 0$ is the *scale factor paramter* of ADEP.

Algebraic Framework for Permutations

Discrete operators for equations of motion in EAs

Let:

- $\mathtt{ASW}_n = \{\sigma_1, \ldots, \sigma_{n-1}\}$ be the "adjacent swap" generators of $\mathcal{S}_n$
- $\pi, \rho \in \mathcal{S}_n$
- $\langle \sigma_{i_1}, \ldots, \sigma_{i_k}, \ldots, \sigma_{i_l} \rangle$ be a minimal factorization of π whose length is l
- $a \in [0, 1]$

Discrete operators are defined as follows:

- $\pi \oplus \rho := \pi \circ \rho$
- $\pi \ominus \rho := \rho^{-1} \circ \pi$
- $a \odot \pi := \sigma_{i_1} \circ \cdots \circ \sigma_{i_k}$, with $k = \lceil a \cdot l \rceil$

Discrete sum and difference

- Algebraically, they **do not rely on the chosen generating set**
- They are **deterministic**
- They are **consistent to each other**:

$$\pi = \rho \oplus (\pi \ominus \rho) = \rho \circ (\rho^{-1} \circ \pi) = \pi$$

Discrete scalar multiplication

- Minimal factorization is **not unique**

Discrete scalar multiplication

- Minimal factorization is **not unique**
- Choose one at random $\implies$ **stochastic operator**

Discrete scalar multiplication

- Minimal factorization is **not unique**
- Choose one at random $\implies$ **stochastic operator**
- A **stochastic factorization algorithm** is required for every generating set

Discrete scalar multiplication

- Minimal factorization is **not unique**
- Choose one at random $\implies$ **stochastic operator**
- A **stochastic factorization algorithm** is required for every generating set
- **COMMON FACTORIZATION SCHEME**
  Identify a measurable property of a permutation such that:

  Then:

Discrete scalar multiplication

- Minimal factorization is **not unique**
- Choose one at random $\Longrightarrow$ **stochastic operator**
- A **stochastic factorization algorithm** is required for every generating set
- **COMMON FACTORIZATION SCHEME**
  Identify a measurable property of a permutation such that:
  - the identity permutation has maximum/minimum value;
  - the value of such property can be increased/decreased by only using simple moves corresponding to the chosen generating set.

  Then:

Discrete scalar multiplication

- Minimal factorization is **not unique**
- Choose one at random $\implies$ **stochastic operator**
- A **stochastic factorization algorithm** is required for every generating set
- **COMMON FACTORIZATION SCHEME**
  Identify a measurable property of a permutation such that:
  - the identity permutation has maximum/minimum value;
  - the value of such property can be increased/decreased by only using simple moves corresponding to the chosen generating set.

  Then:
  - sort the input permutation by only using the chosen moves/generators;
  - keep track of the sequence of selected generators;
  - reverse such sequence and invert every generator;

Stochastic factorization algorithms

- **(ASW)** `RandBS`: randomized variant of bubble-sort
  Monotonically reduces the **inversion count** of a permutation by only applying adjacent swaps.
  Complexity: $\Theta(n^2)$, Optimal: yes

Stochastic factorization algorithms

- **(ASW)** `RandBS`: randomized variant of bubble-sort
  Monotonically reduces the **inversion count** of a permutation by only applying adjacent swaps.
  Complexity: $\Theta(n^2)$, Optimal: yes
- **(EXC)** `RandSS`: randomized variant of selection-sort
  Monotonically increases the **number of cycles** of a permutation by only applying exchanges.
  Complexity: $\Theta(n)$, Optimal: yes

Stochastic factorization algorithms

- **(ASW)** `RandBS`: randomized variant of bubble-sort
  Monotonically reduces the **inversion count** of a permutation by only applying adjacent swaps.
  Complexity: $\Theta(n^2)$, Optimal: yes
- **(EXC)** `RandSS`: randomized variant of selection-sort
  Monotonically increases the **number of cycles** of a permutation by only applying exchanges.
  Complexity: $\Theta(n)$, Optimal: yes
- **(INS)** `RandIS`: randomized variant of insertion-sort
  Monotonically increases the **length of a longest increasing subsequence** by only applying insertions.
  Complexity: $\Theta(n^2)$, Optimal: yes

Stochastic factorization algorithms

- **(ASW)** `RandBS`: randomized variant of bubble-sort
  Monotonically reduces the **inversion count** of a permutation by only applying adjacent swaps.
  Complexity: $\Theta(n^2)$, Optimal: yes
- **(EXC)** `RandSS`: randomized variant of selection-sort
  Monotonically increases the **number of cycles** of a permutation by only applying exchanges.
  Complexity: $\Theta(n)$, Optimal: yes
- **(INS)** `RandIS`: randomized variant of insertion-sort
  Monotonically increases the **length of a longest increasing subsequence** by only applying insertions.
  Complexity: $\Theta(n^2)$, Optimal: yes
- **(REV)** `RandRS`: randomized variant of the Kececioglu-Sankoff algorithm
  Reduces the **number of breakpoints** by only applying reversals.
  Complexity: $\Theta(n^2)$, Optimal: no

How much random we are?

- All the stochastic factorization algorithms perform a **random walk in the sub-graph** formed by the union of all the shortest paths from the identity to the permutation to factorize.
- **Cannot increase entropy** (over the set of minimal factorizations) without increasing computational complexity.

Multiplication by any positive scalar?

- Let $\pi, \rho \in \mathcal{S}_n$ and $a \geq 0$
- Let $|\pi|$ be the length of a minimal factorization of π
- Let $\rho \sqsubseteq \pi$ iff a minimal factorization of ρ is a prefix of a minimal factorization of π

Multiplication by any positive scalar?

- Let $\pi, \rho \in \mathcal{S}_n$ and $a \geq 0$
- Let $|\pi|$ be the length of a minimal factorization of π
- Let $\rho \sqsubseteq \pi$ iff a minimal factorization of ρ is a prefix of a minimal factorization of π
- $\rho = a \odot \pi$ has to satisfy:

  (C1) $|\rho| = \lceil a \cdot |\pi| \rceil$
  (C2) if $a \in [0, 1]$ then $\rho \sqsubseteq \pi$
  (C3) if $a \geq 1$ then $\pi \sqsubseteq \rho$

Multiplication by any positive scalar?

- Let $\pi, \rho \in \mathcal{S}_n$ and $a \geq 0$
- Let $|\pi|$ be the length of a minimal factorization of π
- Let $\rho \sqsubseteq \pi$ iff a minimal factorization of ρ is a prefix of a minimal factorization of π
- $\rho = a \odot \pi$ has to satisfy:

  (C1) $|\rho| = \lceil a \cdot |\pi| \rceil$
  (C2) if $a \in [0, 1]$ then $\rho \sqsubseteq \pi$
  (C3) if $a \geq 1$ then $\pi \sqsubseteq \rho$

- Previous definition ($a \in [0, 1]$) satisfies (C1) and (C2)

Multiplication by any positive scalar?

- Let $\pi, \rho \in \mathcal{S}_n$ and $a \geq 0$
- Let $|\pi|$ be the length of a minimal factorization of π
- Let $\rho \sqsubseteq \pi$ iff a minimal factorization of ρ is a prefix of a minimal factorization of π
- $\rho = a \odot \pi$ has to satisfy:

  (C1) $|\rho| = \lceil a \cdot |\pi| \rceil$
  (C2) if $a \in [0, 1]$ then $\rho \sqsubseteq \pi$
  (C3) if $a \geq 1$ then $\pi \sqsubseteq \rho$
- Previous definition ($a \in [0, 1]$) satisfies (C1) and (C2)
- **In line with the geometric interpretation of the Euclidean space** (use L2 norm and linear dependency among vectors)

Multiplication by any positive scalar?

- Let $\pi, \rho \in \mathcal{S}_n$ and $a \geq 0$
- Let $|\pi|$ be the length of a minimal factorization of π
- Let $\rho \sqsubseteq \pi$ iff a minimal factorization of ρ is a prefix of a minimal factorization of π
- $\rho = a \odot \pi$ has to satisfy:
  - (C1) $|\rho| = \lceil a \cdot |\pi| \rceil$
  - (C2) if $a \in [0, 1]$ then $\rho \sqsubseteq \pi$
  - (C3) if $a \geq 1$ then $\pi \sqsubseteq \rho$
- Previous definition ($a \in [0, 1]$) satisfies (C1) and (C2)
- **In line with the geometric interpretation of the Euclidean space** (use L2 norm and linear dependency among vectors)
- Computation when $a \geq 1$: take a shortest path of the inputted permutation and extend it in such way that the extended path is a shortest path in its own, thus the endpoint permutation is the result.

Issues when $a > 1$

- **Issue1**: $|a \odot \pi|$ may be larger than the diameter
- **Motivation1**: Search space is finite
- **Solution1**: **Truncate if diameter is exceeded**

Issues when $a > 1$

- **Issue1**: $|a \odot \pi|$ may be larger than the diameter
- **Motivation1**: Search space is finite
- **Solution1**: **Truncate if diameter is exceeded**

- **Issue2**: $a \odot \pi$ may not exists for any $\pi \in \mathcal{S}_n$ when `INS` is used (because of the non-convexity)
- **Motivation2**: *longest increasing subsequence* cannot always be reduced by *shifting away* an item
- **Solution2**: **reverse, consider the longest decreasing subsequence, reverse again** $\implies$ it is like using a surrogate weight which is in (non-strict) monotonic relation with the `INS` weight

Is it only intuition?

- Lot of **continuous EAs behaviors can be replicated** in permutation space
  Is this justified only by intuition?

Is it only intuition?

- Lot of **continuous EAs behaviors can be replicated** in permutation space
  Is this justified only by intuition?
- The algebraic framework works **with any finitely generated group**:

Is it only intuition?

- Lot of **continuous EAs behaviors can be replicated** in permutation space
  Is this justified only by intuition?
- The algebraic framework works **with any finitely generated group**:
  - **Bit-strings**
    Operator: `xor`
    Generators: bit-strings with only one 1-bit

Is it only intuition?

- Lot of **continuous EAs behaviors can be replicated** in permutation space
  Is this justified only by intuition?
- The algebraic framework works **with any finitely generated group**:
  - **Bit-strings**
    Operator: `xor`
    Generators: bit-strings with only one 1-bit
  - **Integer vectors**
    Operator: $+$
    Generators: vectors with all 0s except one ± 1 entry

Is it only intuition?

- Lot of **continuous EAs behaviors can be replicated** in permutation space
  Is this justified only by intuition?
- The algebraic framework works **with any finitely generated group**:
  - **Bit-strings**
    Operator: `xor`
    Generators: bit-strings with only one 1-bit
  - **Integer vectors**
    Operator: $+$
    Generators: vectors with all 0s except one ± 1 entry
  - **Enumerable subset of real vectors**
    whose entries are "multiples" of a chosen constant $\hbar > 0$
    Operator: $+$
    Generators: vectors with all 0s except one $\pm\hbar$ entry
    Pushing $\hbar \to 0$ tends to replicate the continuous behaviour in discretized equations of motion

Properties of $\oplus, \ominus, \odot$

- Properties which are satisfied:
  (X is the discrete set of solutions, e.g. permutations)

  (i) $\oplus$ is associative;
  (ii) $\oplus$ is commutative iff $\circ$ is commutative;
  (iii) e is the neutral element for $\oplus$;
  (iv) $x \oplus x^{-1} = x^{-1} \oplus x = e$ for each $x \in X$;
  (v) $1 \odot x = x$ for each $x \in X$;
  (vi) $a \odot (b \odot x) = (ab) \odot x$ for each $x \in X$ and $a, b \geq 0$;
  (vii) $0 \odot x = e$ for each $x \in X$;
  (viii) $x \oplus (y \ominus x) = y$ for each $x, y \in X$.

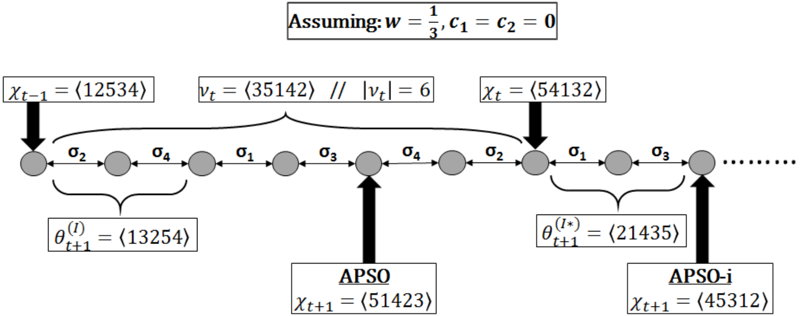
- **Distributive properties are missing**

$$w \odot \nu \neq [(1 + w) \odot \nu] \ominus \nu$$

  where $w \geq 0$ and $\nu \in \mathcal{S}_n$.
  Caution on PSO inertial term

Inertia-preserving Algebraic PSO

- Replace the inertial term $\theta^{(I)} = w \odot \nu$ with $\theta^{(I*)} = [(1+w) \odot \nu] \ominus \nu$



Algebraic Crossover

- **Continuous arithmetic crossover** are convex combinations:
  - $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ are the **parents**
  - $a, b \geq 0$ s.t. $a + b = 1$ are crossover **parameters**
  - $\boldsymbol{z} = a\boldsymbol{x} + b\boldsymbol{y}$ is the **offspring**

Algebraic Crossover

- **Continuous arithmetic crossover** are convex combinations:
  - $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ are the **parents**
  - $a, b \geq 0$ s.t. $a + b = 1$ are crossover **parameters**
  - $\boldsymbol{z} = a\boldsymbol{x} + b\boldsymbol{y}$ is the **offspring**
- Trick: $\boldsymbol{z} = a\boldsymbol{x} + b\boldsymbol{y} = \boldsymbol{y} + a(\boldsymbol{x} - \boldsymbol{y}) = \boldsymbol{x} + b(\boldsymbol{y} - \boldsymbol{x})$

Algebraic Crossover

- **Continuous arithmetic crossover** are convex combinations:
  - $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ are the **parents**
  - $a, b \geq 0$ s.t. $a + b = 1$ are crossover **parameters**
  - $\boldsymbol{z} = a\boldsymbol{x} + b\boldsymbol{y}$ is the **offspring**
- Trick: $\boldsymbol{z} = a\boldsymbol{x} + b\boldsymbol{y} = \boldsymbol{y} + a(\boldsymbol{x} - \boldsymbol{y}) = \boldsymbol{x} + b(\boldsymbol{y} - \boldsymbol{x})$
- **Discretization** $(x, y, z \in \mathcal{S}_n)$:

$$z = \begin{cases} y \oplus a \odot (x \ominus y) & \text{with prob. } 0.5 \\ x \oplus b \odot (y \ominus x) & \text{with prob. } 0.5 \end{cases}$$

Algebraic Crossover

- **Continuous arithmetic crossover** are convex combinations:
  - $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ are the **parents**
  - $a, b \geq 0$ s.t. $a + b = 1$ are crossover **parameters**
  - $\boldsymbol{z} = a\boldsymbol{x} + b\boldsymbol{y}$ is the **offspring**
- Trick: $\boldsymbol{z} = a\boldsymbol{x} + b\boldsymbol{y} = \boldsymbol{y} + a(\boldsymbol{x} - \boldsymbol{y}) = \boldsymbol{x} + b(\boldsymbol{y} - \boldsymbol{x})$
- **Discretization** $(x, y, z \in \mathcal{S}_n)$:

$$z = \begin{cases} y \oplus a \odot (x \ominus y) & \text{with prob. } 0.5 \\ x \oplus b \odot (y \ominus x) & \text{with prob. } 0.5 \end{cases}$$

- Both expressions have the **same support**

Algebraic Crossover

- **Continuous arithmetic crossover** are convex combinations:
  - $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ are the **parents**
  - $a, b \geq 0$ s.t. $a + b = 1$ are crossover **parameters**
  - $\boldsymbol{z} = a\boldsymbol{x} + b\boldsymbol{y}$ is the **offspring**
- Trick: $\boldsymbol{z} = a\boldsymbol{x} + b\boldsymbol{y} = \boldsymbol{y} + a(\boldsymbol{x} - \boldsymbol{y}) = \boldsymbol{x} + b(\boldsymbol{y} - \boldsymbol{x})$
- **Discretization** $(x, y, z \in \mathcal{S}_n)$:

$$z = \begin{cases} y \oplus a \odot (x \ominus y) & \text{with prob. } 0.5 \\ x \oplus b \odot (y \ominus x) & \text{with prob. } 0.5 \end{cases}$$

- Both expressions have the **same support**
- **The offspring lies in a shortest path between the parents** (it is a *geometric crossover* according to "Moraglio et al.")

Algebraic Crossover

- **Continuous arithmetic crossover** are convex combinations:
  - $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ are the **parents**
  - $a, b \geq 0$ s.t. $a + b = 1$ are crossover **parameters**
  - $\boldsymbol{z} = a\boldsymbol{x} + b\boldsymbol{y}$ is the **offspring**
- Trick: $\boldsymbol{z} = a\boldsymbol{x} + b\boldsymbol{y} = \boldsymbol{y} + a(\boldsymbol{x} - \boldsymbol{y}) = \boldsymbol{x} + b(\boldsymbol{y} - \boldsymbol{x})$
- **Discretization** $(x, y, z \in \mathcal{S}_n)$:

$$z = \begin{cases} y \oplus a \odot (x \ominus y) & \text{with prob. } 0.5 \\ x \oplus b \odot (y \ominus x) & \text{with prob. } 0.5 \end{cases}$$

- Both expressions have the **same support**
- **The offspring lies in a shortest path between the parents** (it is a *geometric crossover* according to "Moraglio et al.")
- If greedy strategies are employed $\Longrightarrow$ **Path Relinking**

AX for ASW

Radcliffe properties for precedences

- $[x, y]$ contains the permutations in all the shortest paths between $x, y \in \mathcal{S}_n$
- $z \in [x, y]$ is the offspring of the AX

AX for ASW

Radcliffe properties for precedences

- $[x, y]$ contains the permutations in all the shortest paths between $x, y \in \mathcal{S}_n$
- $z \in [x, y]$ is the offspring of the AX
- **Any permutation is a consistent set of $\binom{n}{2}$ pairwise precedences of items**

- $[x, y]$ contains the permutations in all the shortest paths between $x, y \in \mathcal{S}_n$
- $z \in [x, y]$ is the offspring of the AX
- **Any permutation is a consistent set of $\binom{n}{2}$ pairwise precedences of items**

- AX has the **Radcliffe properties**:
  - a precedence in z is a precedence in x or y (**AX transmits precedences**)
  - common precedences of x and y are precedences in z (**AX is respectful**)
  - $[x, y]$ contains permutations formed by all the consistent combinations of the precedences in x and y (**AX is assorting** if the scalar parameter is chosen randomly)

Further possibilities

- **Lattice structure** of `ASW`: extend AX codomain (Baioletti et al., 2018a)

Further possibilities

- **Lattice structure** of `ASW`: extend AX codomain (Baioletti et al., 2018a)
- **Variable Neighborhoods**: use different neighborhoods for the different individuals in the population (Baioletti et al., 2020b)

Further possibilities

- **Lattice structure** of `ASW`: extend AX codomain (Baioletti et al., 2018a)
- **Variable Neighborhoods**: use different neighborhoods for the different individuals in the population (Baioletti et al., 2020b)
- **Product groups**: model structured representations like e.g. DAGs (Baioletti et al., 2018b)

Further possibilities

- **Lattice structure** of `ASW`: extend AX codomain (Baioletti et al., 2018a)
- **Variable Neighborhoods**: use different neighborhoods for the different individuals in the population (Baioletti et al., 2020b)
- **Product groups**: model structured representations like e.g. DAGs (Baioletti et al., 2018b)
- **Group actions**: extend the algebraic framework to non-group spaces like e.g. permutations with repetition (Baioletti et al., 2020a)

Further possibilities

- **Lattice structure** of `ASW`: extend AX codomain (Baioletti et al., 2018a)
- **Variable Neighborhoods**: use different neighborhoods for the different individuals in the population (Baioletti et al., 2020b)
- **Product groups**: model structured representations like e.g. DAGs (Baioletti et al., 2018b)
- **Group actions**: extend the algebraic framework to non-group spaces like e.g. permutations with repetition (Baioletti et al., 2020a)
- **Permutation as a set of precedences**: iteratively construct a permutation precedence by precedence (Baioletti et al., 2017; Santucci & Ceberio, 2020)

Further possibilities

- **Lattice structure** of `ASW`: extend AX codomain (Baioletti et al., 2018a)
- **Variable Neighborhoods**: use different neighborhoods for the different individuals in the population (Baioletti et al., 2020b)
- **Product groups**: model structured representations like e.g. DAGs (Baioletti et al., 2018b)
- **Group actions**: extend the algebraic framework to non-group spaces like e.g. permutations with repetition (Baioletti et al., 2020a)
- **Permutation as a set of precedences**: iteratively construct a permutation precedence by precedence (Baioletti et al., 2017; Santucci & Ceberio, 2020)
- **Cycle structure**: interesting for TSP and VRP (future work)

... but is it working?

- ADE and APSO significantly outperform the naive DE and PSO equipped with the popular random-key decoder on a variety of benchmarks (Santucci, Baioletti, & Milani, 2019; Santucci et al., 2020)
- ADE obtained:
  - state-of-the-art results on the PFSP, LOPCC and MDTWNPP (Baioletti et al., 2020b; Santucci et al., 2016; Santucci, Baioletti, Di Bari, et al., 2019)
  - competitive results on the LOP, TSP and SRLFP (Baioletti et al., 2015; Baioletti, Milani, Santucci, & Bartoccini, 2019; Di Bari et al., 2020)
  - peak results among EAs for Bayesian networks learning (Baioletti et al., 2018b)
- Algebraic crossovers are competitive with classical permutation crossover operators (Baioletti et al., 2018a)
- Representation as set of precedences allowed to obtain state-of-the-art results on the LOP (Santucci & Ceberio, 2020)
- Identify preferred pairs of items to exchange in order to exit basins of attraction of QAP instances (Baioletti, Milani, Santucci, & Tomassini, 2019)

Conclusions and Open Questions

- The rich algebraic structure of permutations can be fruitfully exploited for:
  - Study structure and properties of the search space
  - Designing algorithms and operators in EC
  - Study the search behaviour of an algorithm

Conclusions and Open Questions

- The rich algebraic structure of permutations can be fruitfully exploited for:
  - Study structure and properties of the search space
  - Designing algorithms and operators in EC
  - Study the search behaviour of an algorithm

- **Open questions**:
  - Can algebraic properties be used to derive expected runtime analyses?
  - Can algebraic properties be used to build a tunable instance generator for the permutation space?
  - Is a review article about EC for permutation problems useful to the community? :-)

Bibliography I

Baioletti, M., Milani, A., & Santucci, V. (2018a). Algebraic crossover operators for permutations. In *Proc. of IEEE CEC 2018*.

Baioletti, M., Milani, A., & Santucci, V. (2015). Linear ordering optimization with a combinatorial differential evolution. In *Proc. of 2015 IEEE SMC 2015*.

Baioletti, M., Milani, A., & Santucci, V. (2017). A new precedence-based ant colony optimization for permutation problems. In *Proc. of SEAL 2017*.

Baioletti, M., Milani, A., & Santucci, V. (2018b). Learning bayesian networks with algebraic differential evolution. In *Proc. of PPSN 2018*.

Baioletti, M., Milani, A., & Santucci, V. (2020a). An algebraic approach for the search space of permutations with repetition. In *Proc. of EVOCOP 2020*.

Baioletti, M., Milani, A., & Santucci, V. (2020b). Variable neighborhood algebraic differential evolution: An application to the linear ordering problem with cumulative costs. *Information Sciences*, *507*, 37–52.

Baioletti, M., Milani, A., Santucci, V., & Bartoccini, U. (2019). An experimental comparison of algebraic differential evolution using different generating sets. In *Proc. of the GECCO 2019*.

Baioletti, M., Milani, A., Santucci, V., & Tomassini, M. (2019). Search moves in the local optima networks of permutation spaces: The QAP case. In *Proc. of the GECCO 2019*.

Bibliography II

Di Bari, G., Baioletti, M., & Santucci, V. (2020). An experimental evaluation of the algebraic differential evolution algorithm on the single row facility layout problem. In *Proc. of the GECCO 2020.*

Santucci, V., Baioletti, M., & Milani, A. (2016). Algebraic differential evolution algorithm for the permutation flowshop scheduling problem with total flowtime criterion. *IEEE Transactions on Evolutionary Computation*, *20*(5), 682–694.

Santucci, V., Baioletti, M., & Milani, A. (2019). Tackling permutation-based optimization problems with an algebraic particle swarm optimization algorithm. *Fundamenta Informaticae*, *167*(1-2), 133–158.

Santucci, V., Baioletti, M., Di Bari, G., & Milani, A. (2019). A binary algebraic differential evolution for the multidimensional two-way number partitioning problem. In *Proc. of EvoCOP 2019.*

Santucci, V., Baioletti, M., & Milani, A. (2020). An algebraic framework for swarm and evolutionary algorithms in combinatorial optimization. *Swarm and Evolutionary Computation*, *55*.

Santucci, V., & Ceberio, J. (2020). Using pairwise precedences for solving the linear ordering problem. *Applied Soft Computing*, *87*.

Thanks for the attention!!!

valentino.santucci@unistrapg.it