Geometry of Evolutionary Algorithms

Alberto Moraglio
University of Birmingham, UK
albmor@gmail.com
http://www.albertomoraglio.com

GECCO'12 Companion, July 7–11, 2012, Philadelphia, PA, USA.
ACM 978-1-4503-1178-6/12/07.

Instructor Biography

- Education:
  - Master in Computer Engineering from Polytechnic University of Turin, Italy
  - PhD in Computer Science from Essex University, UK (2007)
- Positions:
  - Researcher, HP Laboratories, UK
  - Assistant Professor, University of Coimbra, Portugal
  - Research Fellow, University of Kent, UK
  - Research Fellow, University of Birmingham, UK
- Research Interests:
  - Foundational Principles of Evolutionary Computation
  - Bridging Theory and Practice in Evolutionary Computation
- Main contributions to the field:
  - Geometric View of Evolutionary Algorithms (>50 publications)

Agenda

- **Motivations, Research Questions and Methodology**
- **Geometric Interpretation of Search Operators**
- **Fitness Landscape of Geometric Operators**
- **Unification of Evolutionary Algorithms**
- **Principled Design of Crossover Operators**
- **Principled Generalization of Search Algorithms**
- **Unified Theory of Evolutionary Algorithms**
- **A Vision of the Future**
- **Conclusions**

Objectives of the Tutorial

- **Introduce the Geometric View of Evolutionary Algorithms**
- **Provide a unifying framework to think intuitively, formally and generally about Evolutionary Algorithms across Representations**
- **Give a comprehensive overview of the benefits of the Geometric View**
- **Illustrate a way to bridge Theory and Practice**
- **Give evidence of general principles behind Evolutionary Search**
- **Think about a desirable future scenario**
- **Gather ideas, suggestions and criticisms from the participants!**

Motivations, Research Questions and Methodology

Fragmentation

- Different flavors of (traditional) Evolutionary Algorithms:
  - Very many variations on each flavors
  - It is desirable to have a coherent picture (De Jong)
- Evolutionary Algorithms are very similar:
  - Algorithmically irrelevant differences (e.g., application domain and phenotype interpretation)
  - Algorithmic elements that can be freely exchanged (e.g., selection scheme)
- Real difference:
  - Solution representation (e.g., binary strings, real vectors)
  - Search operators (i.e., mutation and crossover)
- **Is there a deeper unity encompassing all Evolutionary Algorithms beyond the specific representation?**

Practice: ad-hoc operators design

- For every new problem and new solution representation search operators are designed ad-hoc
- No systematic way of designing new search operators
  - No guidelines or only informal rule-of-thumbs (heuristic)
  - Not applicable to all representations/problems (limited scope)
  - Mostly for mutation and less for crossover (simple operators)
  - Application of guidelines to specific representation is a black art (vague)

Practice: ad-hoc operators design

- No formal thinking about search operator design
  - **Can we formally define mutation and crossover in general for any representation?**
  - **Can we formally derive representation-specific operators for any target representation?**
  - **Can we automatically construct representation-specific operators from their general definitions?**

Practice: vague meta-heuristics

- Meta-heuristic: a space/problem-independent algorithmic template of a search algorithm that can be specified to new spaces/problems
  - Neighbourhood-based (e.g., local search) vs. Representation-based (e.g., evolutionary algorithms)
- Meta-heuristics have vague non-formal definitions
  - **Can we formally define a meta-heuristic in a space/problem independent way?**
  - **Can we formally specify it to any target space without ad-hoc adaptations?**
  - **Can we prove general search properties of a meta-heuristic?**

Practice: vague meta-heuristics

- New meta-heuristics can be obtained by generalizing search algorithms defined on specific representations
  - E.g., Particle Swarm Optimization can be generalized from continuous to combinatorial spaces
  - **Is there a formal/systematic way of generalizing search algorithms for specific search spaces to (formal) meta-heuristics?**

Theory: rigorous XOR general

- No general theory (general principles)
  - General "theories" are not rigorous (e.g., landscape analysis (Merz))
  - General theories are not about performance (e.g., modern schema theories (Poli), dynamical systems (Stephens))
  - Rigorous theory about performance are very problem specific (e.g., run-time analysis (Wegner))
  - **Are there truly general principles common to all evolutionary algorithms across representations?**
  - **Is a general rigorous theory of performance possible?**

Theory: relevant to practice XOR rigorous

- No practically useful theory (analysis vs. design)
  - Most theories are about algorithm analysis (descriptive)
  - Theories of algorithm/operators design (prescriptive):
    - Heuristic/not formal (e.g., Building-Blocks (Goldberg), Locality (Rothlauf))
    - Formal but without performance guarantee (e.g., Forma Analysis (Radcliffe))
  - **Is a general formal theory of algorithm design that guarantees some form of performance possible?**

Geometric Framework

- Recombination and mutation across representations admit surprisingly simple geometric characterizations relating parents and offspring (geometric operators).
- Formalizes and simplifies the relationship between representations, search operators, distance of the search space/neighbourhood structure, and fitness landscape.
- Allows us to extend the geometric intuition and reasoning valid on continuous spaces to combinatorial spaces.
- The geometric team:
  - My PhD work + 50 publications with many co-authors
  - Other people working on it by their own initiative ☺

Other Formal Unifying Frameworks

- Radcliffe: formal theory of representations based on equivalence classes
- Poli: unification of schema theorem for genetic algorithms and genetic programming
- Stephens: EAs unification using dynamical systems and coarse graining
- Rowe: theory of representations based on group theory
- Stadler: theory of landscapes which links representations and search operators based on algebraic combinatorics

Geometric Interpretation of Search Operators

Metric Space

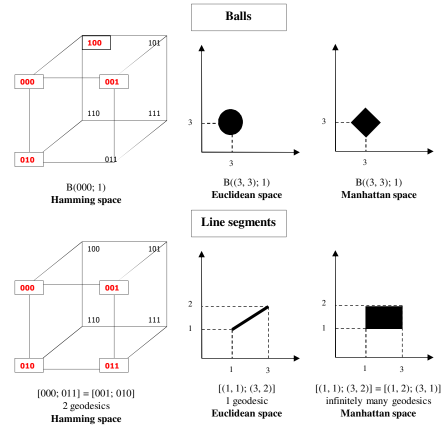
$$d(x,y) \geq 0$$
$$d(x,y) = 0 \Leftrightarrow x = y$$
$$d(x,y) = d(y,x)$$
$$d(x,z) + d(z,y) \geq d(x,y)$$

Balls & Segments

$$B(x; r) = \{ y \in S \mid d(x, y) \le r \}$$

$$[x; y] = \{ z \in S \mid d(x, z) + d(z, y) = d(x, y) \}$$

Squared Balls & Chunky Segments

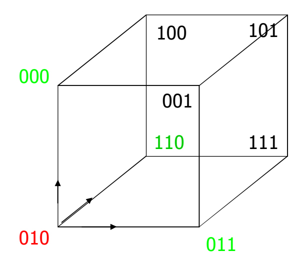


Geometric Crossover & Mutation

- Geometric operators are defined on the structure of the search space by means of simple geometric shapes, like balls and segments. These shapes are used to delimit the region of space that includes all possible offspring with respect to the location of their parents.
- **Geometric crossover**: a recombination operator is a geometric crossover under the metric *d* if all its offspring are in the *d*-metric segment between its parents.
- **Geometric mutation**: a mutation operator is a r-geometric mutation under the metric d if all its offspring are in the d-ball of radius r centred in the parent.

Example of Geometric Mutation

Traditional one-point mutation is 1-geometric under Hamming distance.

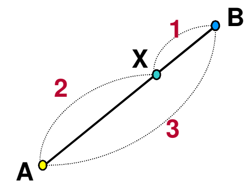


Neighbourhood structure naturally associated with the shortest path distance.

Example of Geometric Crossover

The traditional crossover is geometric under the Hamming distance.

A	0	1	1	0	1
B	1	1	0	1	1
X	0	1	0	1	1



H(A,X) + H(X,B) = H(A,B)

Accessibility & Probability

- Geometric operators are defined in terms of **accessibility**: where to find offspring relative to parents positions.
- More fine-grained classes of geometric operators which include the **probability** of generating offspring are possible.
- For example, in **uniform geometric crossover** under *d* offspring are uniformly distributed on the *d*-metric segment.
- Traditional uniform crossover for binary strings is uniform geometric crossover under Hamming distance.

Uniform Crossover & Uniform Mutation

Uniform geometric crossover:

$$f_{UX}(z \mid x, y) = \Pr\{UX = z \mid P1 = x, P2 = y\} = \frac{\delta(z \in [x, y])}{|[x, y]|}$$

$$\mathrm{Im}[UX(x, y)] = \{z \in S \mid f_{UX}(z \mid x, y) > 0\} = [x, y]$$

Uniform geometric ε-mutation:

$$f_{UM\varepsilon}(z \mid x) = \Pr\{UM = z \mid P = x\} = \frac{\delta(z \in B(x, \varepsilon))}{|B(x, \varepsilon)|}$$

$$\mathrm{Im}[UM_{\varepsilon}(x)] = \{z \in S \mid f_{M\varepsilon}(z \mid x) > 0\} = B(x, \varepsilon)$$

Representation-Search Space Duality

- Cartesian duality: via equating points in the plane and their coordinate geometric object (e.g., a line) have algebraic dual (e.g., a corresponding linear equation in the coordinates of the points on the line).
- An analogous duality applies to geometric operators:
  - coordinates that represent a point in a plane = representation (e.g., binary string) that represent a point in the search space (e.g., hamming space)
  - the same geometric operator can be defined in geometric terms in terms of spatial relations and, at the same time, in can be defined in algebraic terms in terms of manipulation of the underlying representation.

Representation-Search Space Duality

- Example: traditional uniform crossover can be defined:
  (i) geometrically as uniform geometric crossover on the Hamming space
  (ii) algebraically by how the binary strings representing the parents are probabilistically recombined to obtain binary strings representing their offspring
- Algebraic vs. Geometric:
  - Operational (implementation) vs. Declarative (specification)
  - Representation-specific (no distance) vs. Representation-independent (no representation)

Fitness Landscape of Geometric Operators

Fitness landscapes & search operators

- Visual **metaphor** to understand search behaviour
- Used in problem hardness studies
- A fitness landscape is a triple:
  - Fitness function f
  - Solution set S
  - Structure on the search space (e.g., d/Nhd)
- Fitness landscapes are **induced** by search operators:
  - In a search algorithm one can find f and S but not d or Nhd
  - So fitness landscapes do not exist!
  - What is the fitness landscape seen by a search algorithm then?
  - The structure of the search space hence the fitness landscape is "induced" by the search operators.
  - What this actually means is not clear!

Traditional View

- One operator, one landscape (Terry Jones).
- The structure of the space induced by a mutation operator is a graph with nodes representing candidate solutions and weighted edges indicating the probability of producing a certain offspring given a certain parent.
- Different mutation operators induce different structures, hence different landscapes.
- Problem 1: when a search algorithm has two operators (e.g., mutation and crossover) each of them see a different fitness landscape. What is the fitness landscape seen by the search algorithm?

Crossover Landscape

- What is the structure induced by crossover?
- As crossover has two parents edges, each **pair of nodes** are linked by edges to nodes representing possible offspring.
- This structure is not a graph, it is an hyper-graph.
- Problem 2: the natural spatial interpretation of graph is lost, these fitness landscapes have difficult interpretation.
- There are other approaches to induce structure of the search space from recombination operators by theoreticians (e.g., Stadler) or practitioners (e.g., Vanneschi)

Geometric Landscape

- The structure of the landscape is given by the distance associated with the geometric operators.
- As mutation and crossover operator can be defined using the same distance **they see the same fitness landscape**, which is also the landscape seen by the search algorithm.
- Mutation and crossover **navigate the same fitness landscape in different ways**, as mutation produces offspring (i.e., accesses) a ball around the parent, and crossover accesses the segment between the parents.
- Probabilities of accessing offspring are **spatial distributions** (weights on nodes) on balls and segments.

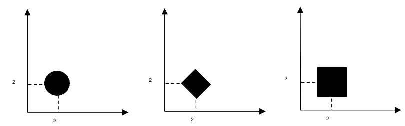
Benefits

- Same fitness landscape for mutation, crossover and search algorithm. This allows to understand how they interact.
- Simple fitness landscape for crossover and more complex search operators.
- Intuitive interpretation of search dynamics in the search space and how it relates with the topography of the fitness landscape.
- Rigorous and complete description of the search. It can be used to **prove** performance of search algorithms on fitness landscapes.
- Unifies neighbourhood search view and representation-based search view, that are now seen as dual and equivalent.

Geometric Unification of Evolutionary Algorithms

Minkowski spaces – real vectors

Balls

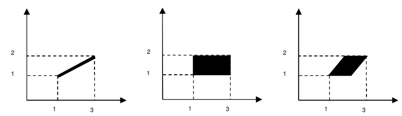


B((2, 2); 1)
Euclidean space

B((2, 2); 1)
Manhattan space

B((2, 2); 1)
Chessboard space

Segments



[(1, 1); (3, 2)]
1 geodesic
Euclidean space

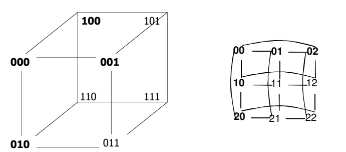
[(1, 1); (3, 2)] = [(1, 2); (3, 1)]
infinitely many geodesics
Manhattan space

[(1, 1); (3, 2)]
infinitely many geodesics
Chessboard space

Pre-existing operators – real vectors

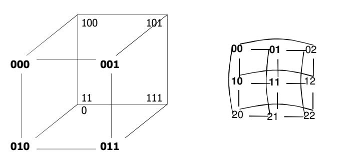
- **Mutations:**
  - bounded spherical mutation: geometric under Euclidean distance
  - creep mutation: geometric under Chessboard distance
- **Recombinations:**
  - blend crossover: geometric under Euclidean distance
  - box crossover: geometric under Manhattan distance
  - discrete crossover: geometric under Manhattan distance
  - extended-line & extended-box crossovers: **non-geometric**

Hamming spaces – n-ary strings



B(000; 1)
Hamming space H(3,2)

B(00;1)
Hamming space H(2,3)



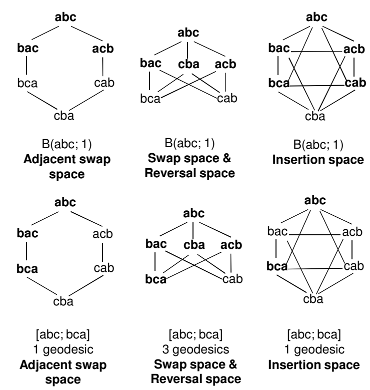
[000; 011] = [001; 010]
2 geodesics
Hamming space H(3,2)

[00;11]=[01;10]
2 geodesics
Hamming space H(2,3)

Pre-existing operators – n-ary strings

- **Mutations**:
  - point-mutations for binary and n-ary strings:
    1-geometric mutation under Hamming distance
  - position-wise mutations:
    n-geometric mutation under Hamming distance
    (with probability distribution only function of the distance)
- **Recombinations**:
  - all mask-based crossovers (including 1-point, 2-point, uniform) for binary and n-ary strings:
    geometric crossover under Hamming distance
  - intermediate recombination for **integer vectors**:
    geometric crossover under Manhattan distance on integer vectors

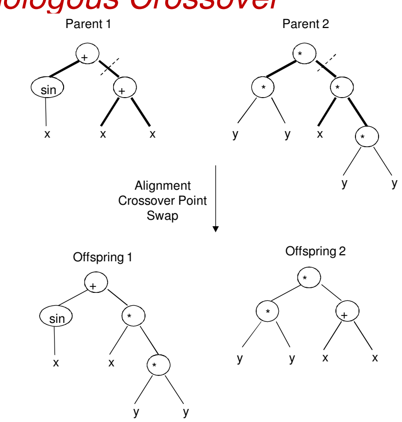
Cayley spaces - permutations



Pre-existing operators – permutations

- **Mutations:**
  - single edit-move mutations: 1-geometric mutation under corresponding edit distance
- **Recombinations**:
  - PMX: geometric crossover under swap distance
  - Cycle crossover: geometric crossover under swap distance & Hamming distance (restricted to permutations)
  - Cut-and-fill crossovers (adaptations of 1-point crossover): geometric crossovers under swap and adjacent swap distances
  - Merge crossover: geometric crossover under insertion distance
  - Davis's order crossover: non-geometric crossover
  - Most recombinations for permutations are geometric crossovers

Syntactic tree spaces – Homologous Crossover



Pre-existing operators – syntactic trees

- **Mutations:**
  - point and sub-tree mutations: geometric mutation under structural Hamming distance on trees (mutations towards the root have larger radius)
- **Recombinations**:
  - Koza's sub-tree swap crossover: non-geometric
  - Homologous crossover: geometric under structural Hamming distance

Variable length sequence spaces – Homologous Recombination

```
Parent1=AGCACACA
Parent2=ACACACTA
```

best inexact alignment (with gaps):

```
AGCA|CAC-A → Child1=AGCACACTA
A-CA|CACTA → Child2=ACACACA
```

Pre-existing operators – sequences: Biological Recombination

- **Mutation:**
  - insertion, deletion or substitution of a single amino acid: 1-geometric mutation under Levenshtein distance
- **Recombination:**
  - Homologous recombination for variable length sequences (1-point, 2-points, n-points, uniform): geometric crossover under Levenshtein distance
  - More realistic models of homologous biological recombination with respects to gap size and base-pairs matching preference: geometric crossovers under weighted and block-based Levenshtein distance

Significance of Unification

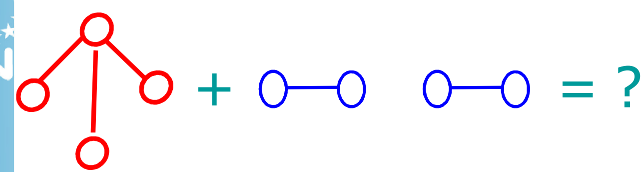
- Most of the pre-existing crossover operators for major representations fit the geometric definition
- Established pre-existing operators have emerged from experimental work done by generations of practitioners over decades
- **Geometric crossover compresses in a simple formula an empirical phenomenon**

Principled Design of Crossover Operators

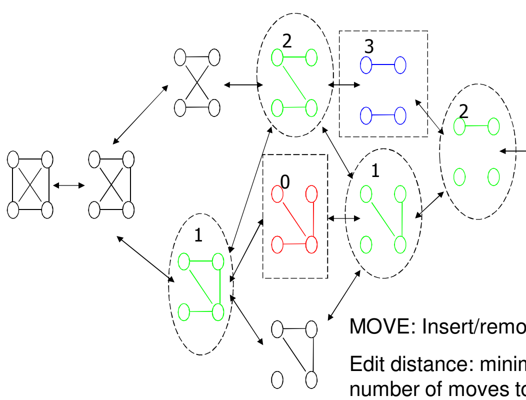
Crossover Principled Design

- Domain specific solution representation is effective
- **Problem**: for non-standard representations it is not clear how crossover should look like
- **But**: given a problem you may know already a **good neighbourhood structure/distance/mutation**
- Geometric Interpretation of Crossover:
  - **your representation and space structure =>**
  - **specific geometric crossover by plugging the space structure in the definition =>**
  - **operational definition of crossover manipulating the underlying representation**

Crossover Design: Graph Example



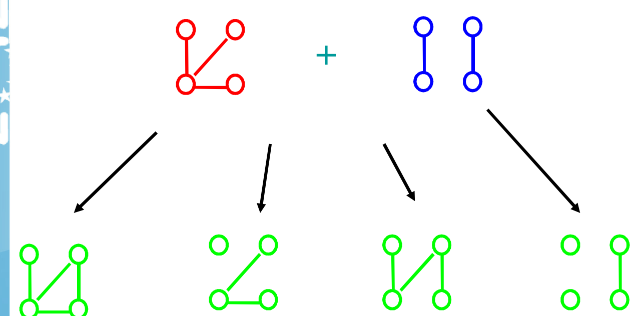
Non-labelled graph neighbourhood



MOVE: Insert/remove an edge

Edit distance: minimum number of moves to transform a graph to the other

Offspring



Operational Geometric Crossover

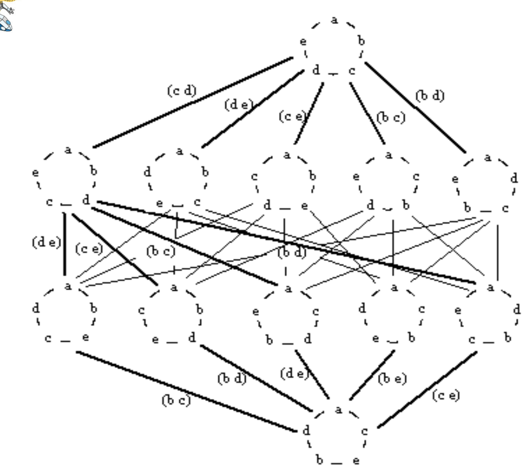
- Edit distance has a natural dual interpretation:
  - measure of distance in the search space
  - measure of similarity on the underlying representation
  - this can be used to help identifying an operational definition of crossover representation (implementation) which corresponds its geometric definition in terms of distance (specification)
- For graphs under ins/del edge edit distance the operational crossover is as follows:
  - Pair up the nodes of the parent graphs such that there are the minimum number of edges mismatches
  - Recombine the aligned parent graphs using a recombination mask on the edges
  - This recombination **implements exactly** the geometric crossover

Crossover Design: TSP Example

- Edit distance duality for permutations:
  - producing offspring in the segment between parents on a space generated by moves of type x (e.g., swaps) ⇔
  - producing offspring permutations on **minimal sorting trajectories** to sort a parent permutation into the other using move of type x
- Sorting Crossovers:
  - Geometric crossover for permutations can be implemented using traditional sorting algorithms and returning as offspring a partially sorted permutation
  - Adj. Swap -> bubble sort
  - Swap -> selection sort,
  - Insertion ->insertion sort
- Pre-existing geometric crossovers for permutations are sorting crossovers in disguise

Crossover Design: TSP example

- A known **good** neighbourhood structure for TSP is 2opt structure = space of circular permutations endowed with reversal edit distance
- Geometric crossover for TSP =
  picking offspring on the minimal sorting trajectories by sorting one parent circular permutation toward the other parent by reversals (sorting circular permutations by reversals)



Operational Geometric Crossover for TSP

- BAD NEWS: sorting circular permutations by reversals is NP-Hard!
- GOOD NEWS: there are approximation algorithms that sort within a bounded error to optimality
- A 2-approximation algorithm sorts by reversals using sorting trajectories that are at most twice the length of the minimal sorting trajectories
- Approximation algorithms can be used to build approximated geometric crossovers for TSP
- In experiments, this crossover beats Edge Recombination which is the best for TSP

Product Geometric Crossover

- It is a simple and general method to build more complex geometric crossovers from simple geometric crossovers
- GX1:AxA→A geometric under d1
- GX2:BxB→ B geometric under d2
- A product crossover of GX1 and GX2 is an operator defined on the cartesian product of their domains PGX:(A,B)x(A,B)→(A,B) that applies GX1on the first projection and GX2 on the second projection. **GX1 and GX2 do not need to be independent and can be based on different representations.**
- Theorem: PGX is a geometric crossover under the distance d = d1+d2

Crossover Design: Sudoku Example

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

Fill in the grid so that every row, every column, and every 3x3 box contains the digits 1 through 9

4 types of constraints:
1) Fixed Elements
2) Rows are permutations
3) Columns are permutations
4) Boxes are permutations

Crossover Design: Sudoku Example

- We start from an initial population of solutions (filled grids) correct with respect to constraints 1) and 2)
- We want a geometric crossover defined on the entire Sudoku grid that preserves constraints 1) and 2) so that we search a smaller search space
- Constraints 3) and 4) are treated as soft constrains and the fitness of a solution is the number of unsatisfied constraints (to minimze)
- The Hamming distance between grids gives rise to a smooth landscape because close grids have similar fitness

Crossover Design: Sudoku Example

- The cycle crossover on a row preserves constraints 1) and 2) and it is geometric under Hamming distance
- For the product geometric crossover theorem, the row-wise cycle crossover is geometric under Hamming distance on the entire grid
- The fitness landscape seen by this crossover is smooth
- This crossover performed very well in experiments compared with other recombinations

Path-relinking = Crossover

- The meta-heuristic path-relinking (Glover) searches on a path between solutions in the neighbourhood structure (not necessarily on a shortest path/segment). It has been successfully applied to many combinatorial problems.
- From a design viewpoint, geometric crossover can be understood as a **formalized generalization** (to metric spaces) of path-relinking that gives a formal recipe to design new crossover operators rather than suggesting heuristically how to search the neighbourhood structure.
- Geometric crossover **unifies** the notions of recombination (i) as manipulation of the parental representation and (ii) as neighbourhood search between parental location. It shows that the dichotomy neighbourhood search vs. representation-based search is only illusory and that essentially **path-relinking is dual and equivalent to crossover.**

Principled Generalization of Search Algorithms

Motivations

- Problem: ad hoc extensions of continuous search algorithms to combinatorial spaces.
  Is there a **systematic** way?
- Solution: Principled generalization: formal generalization of continuous search algorithm via geometric interpretation of operators
- Applied to
  - Particle Swarm, Differential Evolution, Nelder&Mead
  - Binary strings, Permutations, GP trees

Generalization Methodology

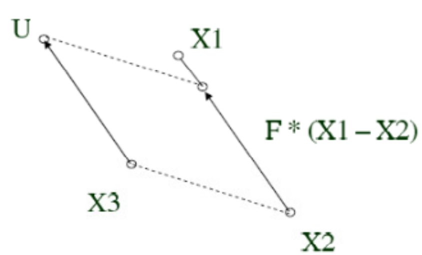
1. Take a continuous optimization algorithm
2. Rewrite search operators using geometric objects as functions of only the Euclidean distance
3. Substitute Euclidean distance with a generic metric → formal geometric algorithm
4. Plug a new distance in the formal algorithm → instance of the algorithm for a new space
5. Rewrite the search operators getting rid of the distance and using the associated representation

Differential Evolution Example

initialize population of N_p real vectors at random
while stop criterion not met **do**
  for all vector $X(i)$ in the population **do**
    pick at random 3 distinct vectors from the current population $X1, X2, X3$
    $\boxed{\text{create mutant vector } U = X3 + F \cdot (X1 - X2) \text{ where } F \text{ is the scale factor parameter}}$ DM
    $\boxed{\text{set } V \text{ as the result of the discrete recombination of } U \text{ and } X(i) \text{ with probability } Cr}$ DX
    if $f(V) \geq f(X(i))$ **then**
      set the i^{th} vector in the next population $Y(i) = V$
    else
      set $Y(i) = X(i)$
    end if
  end for
  for all vector $X(i)$ in the population **do**
    set $X(i) = Y(i)$
  end for
end while

Differential Mutation

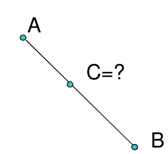
Construction of U using vectors



$U = X3 + F * (X1 - X2)$

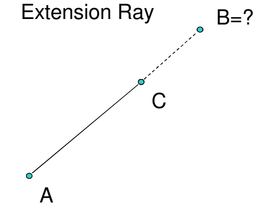
Convex Combination & Extension Ray

Convex Combination



C=CX(A,B)

Extension Ray

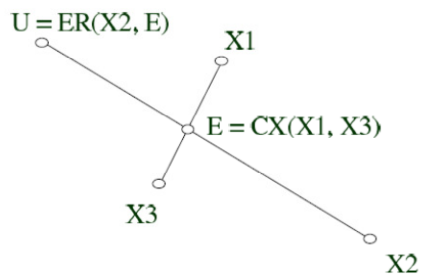


B=ER(A,C)

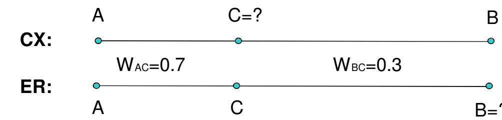
- Extension ray is the inverse operation of convex combination
- **They are well-defined in any metric space**

Differential Mutation

Construction of U using convex combination and extension ray



Weighted CX and ER



- C = CX(A,B) with weights WAC and WBC
- Weights are attraction coefficients
- Distances inversely proportional to weights
- B = ER(A,C) with weights WAC and WBC
- Weights have the same meaning in CX & ER
- But different givens and unknowns

Formal Geometric Differential Evolution

initialize population of N_p configurations at random
while stop criterion not met **do**
  for all configuration $X(i)$ in the population **do**
    pick at random 3 distinct configurations from the current population $X1, X2, X3$
    set $W = \frac{1}{1+F}$ where F is the scale factor parameter
    create intermediate configuration E as the convex combination $CX(X1, X3)$ with weights $(1-W, W)$
    create mutant configuration U as the extension ray $ER(X2, E)$ with weights $(W, 1-W)$
    create candidate configuration V as the convex combination $CX(U, X(i))$ with weights $(Cr, 1-Cr)$ where Cr is the recombination parameter
    if $f(V) \geq f(X(i))$ **then**
      set the i^{th} configuration in the next population $Y(i) = V$
    else
      set $Y(i) = X(i)$
    end if
  end for
  for all configuration $X(i)$ in the population **do**
    set $X(i) = Y(i)$
  end for
end while

DM

DX

Specialization (Hamming space)

- The GDE is a formal algorithm that is specialized to the Hamming space once all its operators (DM and DX) are specialized to the Hamming space
- DM and DX can be rewritten solely in terms of convex combination and extension ray combination
- So, to obtain the specialization of the GDE to the Hamming space, we only need the specializations of convex combination and extension ray

Convex Combination & Extension Ray (Hamming space)

- **Convex combination**: it is a form of biased uniform crossover which prefers bits form one or the other parents according to their weights
- **Extension ray recombination**: the offspring C of binary extension ray originating in parent A and passing through parent B can be obtained by starting from B and with a suitable probability flipping those bits that**, at the same time**, increase the Hamming distance form B and from A
- These operators are provably conforming to the geometric formal definitions of convex combination and extension ray under Hamming distance

Results

- When ported from continuous to Hamming space all the algorithms (DE, PSO, NM) worked very well **out-of-the-box**. This shows that continuous algorithm can be ported using this methodology to discrete spaces.
- When specified to permutations and GP trees spaces a number of surprising behaviours appeared.
- As we applied the very same algorithms to different spaces, the cause of their specific behaviours are specific geometric properties of the underlying search space they are applied to. This allows us in principle to create a taxonomy of search spaces according to their corresponding effects on search behaviour.
- Relevant properties: symmetry, curvature, deformation.

Unified Theory of Evolutionary Algorithms

Formal Evolutionary Algorithm

- Geometric Crossover can be understood as a functional form taking the distance *d* as argument.
- An evolutionary algorithm with geometric crossover can be understood as a function of the metric *d* (*d* is a parameter as e.g., the mutation rate).
- From an abstract point of view, an evolutionary algorithm with geometric crossover with any metric is a well-defined representation-independent formal specification of a search algorithms whose properties derive form the metric axioms (formal evolutionary algorithm (see also Radcliffe)).

Abstract Evolutionary Search

- What happens if we “run” a formal evolutionary algorithm?
- A formal model of a search algorithm can be used to infer (some properties of) the behaviour of a partially-specified algorithm, where the metric parameter is left unspecified.
- Abstract evolutionary search: the behaviour obtained by “running” a formal evolutionary algorithm. This can be described axiomatically (from the metric axioms).
- The abstract evolutionary search process is the behaviour of the formal evolutionary algorithm on ALL possible search (metric) spaces and associated representations.

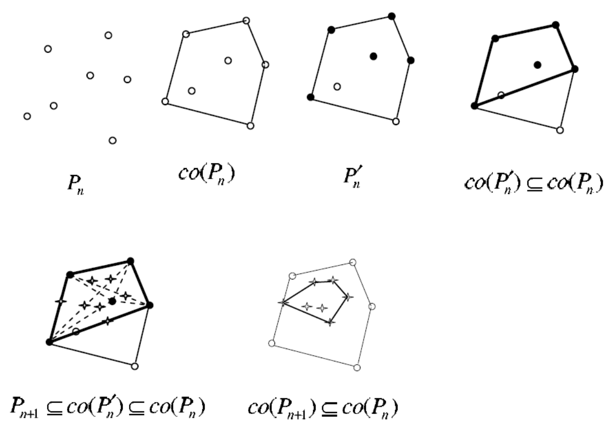
Abstract Convex Evolutionary Search

- Theorem: For any evolutionary algorithm repeating the cycle selection, crossover, replacement we have that the convex hulls of the populations form a nested chain:

$$co(pop_{n+1}) \subseteq co(pop_n) \subseteq \ldots \subseteq co(pop_1) \subseteq co(pop_0)$$

- **This is very general**: it holds for any representation, any distance, any problem (landscape), any offspring distribution of geometric crossover, any selection and replacement. It even applies to varying population sizes.

Abstract Convex Evolutionary Search

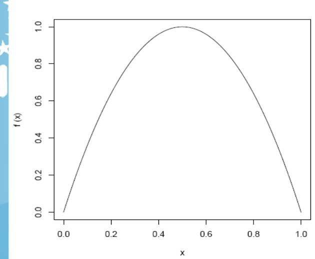


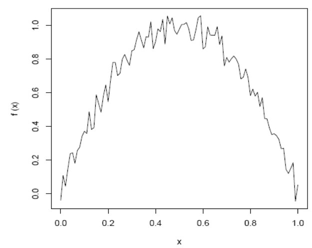
Matching Abstract Search & Abstract Landscape

- NFLT: any non-futile theory which aims at proving performance better than random search of a class of search algorithms must indicate w.r.t. which class of fitness landscapes.
- Are there general conditions on the fitness landscape that guarantee good performance of the convex search for any space/representation?
- At an abstract level, all evolutionary algorithms (with geometric crossover) present a unitary behaviour.
  Is there a class of fitness landscapes well-defined at an abstract level that leads to good performance independently from the specific *d*?

Concave Fitness Landscapes

Convex search works well on (approximately) concave landscapes





This generalises to general metric spaces with ANY representation

Steady-Improvement Theorem

- *On a concave fitness landscape, by applying geometric crossover to parents sampled uniformly at random from ANY population of parents, the expected average fitness of the offspring population is not less than the average fitness of the parent population.*
- As (non-adversary) selection cannot get the fitness of the offspring worse, **this is a statement about the one-step performance of the formal evolutionary algorithm on an abstract fitness landscape.**
- Performance degrades nicely as landscapes become less concave.

Two Remarks

- 1) Good News: this result shows that concave landscapes in this sense are extremely “crossover-friendly” as normally to achieve avg. fitness of the offspring not worse than the avg. fitness of the parent one does require selection!
- 2) Bad News: this result cannot be reiterated to obtain not trivial lower-bound after n-steps.

Work in Progress

- Looking at fitness landscapes arising from combinatorial problems (big valley HP)
- N-step performance (curvature of the concave landscape)
- How can mutation be naturally included in this framework? (from accessibility to probability)
- How far can a theory be pushed forward at this level of abstraction? Only time will tell…

A Vision of the Future: Automatic Evolutionary Problem Solving

A Future Scenario

- Goal: automated design of efficient EAs for any problem
- Time line:
  - PAST: original GA: we thought we had a magic solver → NFL said no
  - PRESENT: black art: how to tailor EA to the problem at hand?
  - FUTURE (theory): formal general **theory of design of provably efficient** EA
  - FUTURE (practice): **automated design**, **automated implementation**, theory-led parameter settings

"Magic" Evolutionary Meta Solver

- INPUT: Problem Description
  -> Magic Evolutionary Meta Solver ->
  OUTPUT: Solution with Guaranteed Approximation
- NFL does not apply because the Meta Solver uses **full knowledge of the problem** to derive a problem-tailored evolutionary algorithm which is provably efficient by the theory
- At this point the human designer would be made redundant, people would not even know or care what is inside the magic box, they will just use it!
- This is a desirable **remote future** scenario, is it in principle at all possible? Is it pure science fiction?

From Problem to Solution

- INPUT: problem description
- 1) **Formulation**: choice of solution representation and space structure (e.g., distance, neighbourhood structure) such that the problem is turned into a EA easy class (e.g., "smooth" landscape)
- 2) **Adaptation**: the EA scheme is applied to the chosen representation and space structure
- 3) **Implementation**: the specific EA for the problem at hand with a given representation and structure is derived
- 4) **Tuning**: parameter values are chosen
- 5) **Execution**: the problem specific algorithm is executed and the best solution obtained
- OUTPUT: solution

Automatic Formulation

- A theory should be **abstract** and accept as input parameters: landscapes based on different representations and neighbourhood structures
- A theory should relate performance guarantee of the EA on the landscape as a function of its **degree** of smoothness
- From the algebraic description of the problem, the system should be able to **infer** the degree of smoothness (e.g., Lipchitz continuity) without experiments for any choice of representation and neighbourhood structure
- The choice of representation and neighbourhood structure available have to be restricted to those that admit an **efficient implementation** of search operators

Automatic Formulation

- Each combination of representation and neighbourhood structure gives rise to a certain degree of smoothness of the landscape for the problem at hand
- Choose the combination of representation and neighbourhood structure such that the theory predicts the best performance **guarantee**
- As the theory is sound, the solution obtained by the problem-specific EA that will be constructed will meet this **guarantee**

Automatic Adaptation

- Automatic Adaptation: the formal specification of the problem specific EA can be obtained unambiguously by instantiation of the formal EA on the specific fitness landscape (solution representation, neighbourhood structure and fitness function)

Automatic Implementation

- Automatic Implementation: the implementation of the specification of the problem-specific EA can be obtained by deriving operational descriptions in terms of representation manipulation of the geometric operators for the specific representation and space structure. This can be done using a library of pre-implemented operators meeting the specifications, by operators compositions or by operator synthesis.
- Differently from pre-existing software-suite that allow the user to build custom EA by combining components, the specific EA obtained as above has a **formal semantic** dictated by the theory which certifies that the solution found by the specific EA will conform to the **theoretical performance guarantees**

Automatic Parameter Setting

- The performance guarantee produced by the theory is expressed as a functions of the parameters of the EA (e.g., crossover rate, selection intensity). The optimal values of the parameters can be obtained analytically by choosing those values that give the best guarantee.

How far are we?

- The geometric framework makes in principle possible the outlined scenario as it covers all the necessary conceptual steps
- The necessary theory for the performance guarantee can be hard to obtain, but it may be possible (e.g., the general results on convex evolutionary search is promising)
- The formal synthesis machinery to pass from specifications to implementation may be within reach (see also Fonseca). If we replace the missing formal theory by a "heuristic theory" or by experiments, we could be able to implement a prototype of the system.

Conclusions and Future Work

Summary

- **Motivations:**
  - fragmentation, no formal design, no general theory.
- **Geometric Operators:**
  - Search operators can be defined formally across representations using simple geometric shapes.
- **Fitness Landscape:**
  - Geometric mutation and crossover see the same simple fitness landscape but they navigate it in different ways.
- **Unification:**
  - Many crossover operators across representations are geometric crossover under some distance.

Summary

- **Crossover Principled Design:**
  - Plugging in an edit distance associated with a new representation it is possible to derive operational definition of the specific geometric crossover.
- **Principled Generalization of Search Algorithms:**
  - Generalization by replacing the Euclidean distance in continuous search algorithm with a generic metric.
- **Unified Theory of Evolutionary Algorithms:**
  - All evolutionary algorithms with geometric crossover do a form of convex search. They work well on concave-like landscapes.
- **A Vision of the Future:**
  - Automated design of efficient EAs for any problem.

Current/Future Work

- Generalizing:
  - Established Algorithms, e.g., Estimation of Distribution Algorithms
  - Established Concepts, e.g., Schema
  - Older and newer theories, e.g., Schema Theorem, Run-Time Analysis
- Reformulating non-geometric theories in geometric terms:
  - Elementary Landscapes (Stadler)
  - Forma Analysis (Radcliffe)
- Formalizing and making rigorous practical theories geometric in flavour:
  - Landscape Analysis, e.g., Global Convexity (Boese)
  - Locality and Redundancy of Genotype-Phenotype map (Rothlauf)
- Applying the framework to specific domain & problems:
  - Semantic Crossover for Genetic Programming (Krawiek)

Take home message

- There are fundamental geometric principles lurking behind the scene of all evolutionary algorithms, which are made explicit by the geometric view.
- The geometric view is also a unifying way of thinking about evolutionary algorithms which is general, rigorous and intuitive at the same time, with interesting consequences for (bridging) theory and practice.
- I hope from now on you will **think geometrically** about whatever aspect of evolution interests you! ☺
- Collaborations are most welcome!

Thank you!

References

- K. D. Boese, A. B. Kahng, and S. Muddu. "A new adaptive multi-start technique for combinatorial global optimizations", *Operations Research Letters*, 15:101–113, 1994
- K. DeJong. "*Evolutionary computation: a unified approach" ,The MIT Press, 2006*
- *S. Droste, T. Jansen, and I. Wegener. "On the analysis of the (1+1) Evolutionary Algorithm", Theoretical Computer Science (2002), 276, 51-81*
- *C. Fonseca. "Using geometric crossover as a formal specification for Python programs", personal communication, 2009*
- *F. Glover, M. Laguna, and R. Marti, "Fundamentals of scatter search and path relinking", Control and Cybernetics 29 (2000), no. 3, 653–684*
- *D. E. Goldberg, "Genetic algorithms in search, optimization and machine learning", Addison-Wesley, 1989*

References

- *S. Gustafson and L. Vanneschi, "Operator-based tree distance" in genetic programming, IEEE Transactions on Evolutionary Computation 12(4) (2008), 506–524*
- *T. Jones, "Evolutionary algorithms, fitness landscapes and search", Ph.D. thesis, University of New Mexico, 1995*
- *K. Krawiec and P. Lichocki. "Approximating geometric crossover in semantic space", Genetic and Evolutionary Computation Conference, pages 987–994, 2009*
- W. Langdon and R. Poli. "*Foundations of Genetic Programming", Springer-Verlag, 2002*
- P. Merz and B. Freisleben. "Fitness landscapes and memetic algorithms", In D. Corne, M. Dorigo, and F. Glover, editors, *New ideas in optimization.* McGraw-Hill, 1999
- N. Radcliffe. "Equivalence class analysis of genetic algorithms", *Complex Systems, 5:183–205, 1991*

References

- C. M. Reidys and P. F. Stadler. "Combinatorial landscapes", *SIAM Review, 44:3–54, 2002*
- F. Rothlauf. "*Representations for Genetic and Evolutionary Algorithms", Springer, 2002*
- J. E. Rowe, M. D. Vose, and A. H. Wright. "Group properties of crossover and mutation", *Evolutionary Computation Journal, 10(2):151–184, 2002*
- C. R. Stephens and A. Zamora. "EC theory: A unified viewpoint", In *Proceedings of the Genetic and Evolutionary Computation Conference, pages* 1394–1405, 2003
- P. D. Surry and N. J. Radcliffe. "Formal algorithms + formal representations = search strategies", In *Proceedings of Parallel Problem Solving from Nature Conference, pages 366–375, 1996*
- D. H. Wolpert and W. G. Macready. "No free lunch theorems for optimization", *IEEE Transaction on Evolutionary Computation, 1(1):67–82, 1996*

References

- A. Moraglio, "Geometric Theory of Representations for Evolutionary Algorithms", Springer (forthcoming).
- Y. Borenstein, A. Moraglio (Editors), "Theory and Principled Methods for the Design of Metaheuristics", 2011, Springer.
- A. Moraglio, J. Togelius, S. Silva "Geometric Differential Evolution for Combinatorial and Programs Spaces", Evolutionary Computation Journal, 2011
- H-Y. Kim, Y. Yoon, A. Moraglio, B-R. Moon "Geometric Crossover for Real-coded Genetic Algorithms", Information Sciences Journal, 2010
- Y. Yoon, Y.-H. Kim, A. Moraglio, B.-R. Moon, "Geometric Interpretation of Genotype-Phenotype Mapping and Induced Crossovers", Theoretical Computer Science Journal, 2010
- A. Moraglio, R. Poli "Topological Crossover for the Permutation Representation", Italian Journal Intelligenza Artificiale, 2010

References

- A. Moraglio, C. Di Chio, J. Togelius, R. Poli "Geometric particle swarm optimisation", Journal of Artificial Evolution and Applications, online article ID 143624, 14 pages, Volume 2008, 2008
- A. Moraglio, H-Y. Kim, Y. Yoon, B-R. Moon "Geometric Crossovers for Multiway Graph Partitioning", Evolutionary Computation Journal, volume 15, issue 4, pages 445-474, 2007
- A. Moraglio, Y. Borenstein "A Gaussian Random Field Model of Smooth Fitness Landscapes" in "Theory and Principled Methods for the Design of Metaheuristics", Y. Borenstein, A. Moraglio (editors), Springer, 2011
- A. Moraglio, H-Y. Kim, Y. Yoon, B-R. Moon "Geometric Crossovers for Multiway Graph Partitioning", in "Theory and Principled Methods for the Design of Metaheuristics", Y. Borenstein, A. Moraglio (editors), Springer, 2011

References

- A. Moraglio, H-Y. Kim, Y. Yoon “Geometric Surrogate-Based Optimisation for Permutation-Based Problems”, GECCO 2011
- A. Moraglio, S. Silva “Geometric Nelder-Mead Algorithm on the Space of Genetic Programs”, GECCO 2011
- A. Moraglio, A. Kattan “Geometric Generalisation of Surrogate Model Based Optimisation to Combinatorial Spaces”, European Conference on Combinatorial Optimisation, 2011
- A. Moraglio "Abstact Evolutionary Convex Search", Workshop on the Foundations of Genetic Algorithms, 2011
- A. Alentorn, A. Moraglio, C. G. Johnson “Binary Nelder-Mead Algorithm for Market Neutral Portfolio Optimization”, IEEE UK Conference on Computational Intelligence, 2010
- A. Moraglio "One-Point Geometric Crossover", Proceedings of Parallel Problem Solving from Nature 2010

References

- A. Moraglio, J. Togelius "Geometric Nelder-Mead Algorithm for the Permutation Representation", Proceedings of IEEE World Conference on Computational Intelligence 2010
- A. Moraglio, S. Silva "Geometric Differential Evolution on the Space of Genetic Programs", Proceedings of European Conference on Genetic Programming, pages 171-183, 2010
- A. Moraglio, C. Johnson "Geometric Generalization of Nelder-Mead Algorithm", Proceedings of European Conference on Evolutionary Computation in Combinatorial Optimisation, pages 190-201, 2010
- A. Moraglio, J. Togelius “Geometric Differential Evolution”, Genetic and Evolutionary Computation Conference, pages 1705-1712, 2009
- A. Moraglio, J. Togelius “Inertial Geometric Particle Swarm Optimization”, IEEE Congress on Evolutionary Computation, pages 1973-1980, 2009

References

- A. Moraglio, Y. Borenstein “A Gaussian Random Field Model of Smooth Fitness Landscapes”, Workshop on Foundations of Genetic Algorithms, pages 171-182, 2009
- J. Togelius, R. De Nardi, A. Moraglio “Geometric PSO + GP = Particle Swarm Programming”, IEEE Congress on Evolutionary Computation, pages 3594-3600, 2008
- C. Di Chio, A. Moraglio, R. Poli “Geometric Particle Swarm Optimization on Binary and Real Spaces: from Theory to Practice”, Particle Swarms: the Second Decade – Genetic and Evolutionary Computation Conference workshop, 2007
- A. Moraglio, J. Togelius “Geometric PSO for the Sudoku Puzzle”, Genetic and Evolutionary Computation Conference, pages 118 - 125, 2007
- Y. Yoon, H-Y. Kim, A. Moraglio, B-R. Moon “Geometric Crossover for Real-Vector Representation”, Genetic and Evolutionary Computation Conference, page 1539, 2007

References

- A. Moraglio, C. Di Chio, R. Poli “Geometric particle swarm optimisation”, European Conference on Genetic Programming, pages 125-136, 2007
- A. Moraglio, R. Poli “Inbreeding Properties of Geometric Crossover and Non-geometric Recombinations”, Foundations of Genetic Algorithms, pages 1-14, 2007
- A. Moraglio, H-Y. Kim, Y. Yoon, B-R. Moon, R. Poli “Cycle Crossover for Permutations with Repetitions: Application to Graph Partitioning”, Evolutionary Algorithms: Bridging Theory and Practice - *Parallel Problem Solving from Nature* workshop, 2006
- A. Moraglio, R. Poli “Geometric Crossover for Sets, Multisets and Partitions”, *Parallel Problem Solving from Nature*, pages 1038-1047, 2006
- A. Moraglio, R. Poli “Product Geometric Crossover”, *Parallel Problem Solving from Nature,* pages 1018-1027, 2006

References

- A. Moraglio, R. Poli "Inbreeding Properties of Geometric Crossover and Non-geometric Recombinations", Evolutionary Computation Workshop - European Conference on Artificial Intelligence, 2006
- R. Seehuus, A. Moraglio "Geometric Crossover for Protein Motif Discovery", Workshop on Adaptive Representations - Genetic and Evolutionary Computation Conference, 2006
- A. Moraglio, R. Poli, R. Seehuus "Geometric Crossover for Biological Sequences", Workshop on Adaptive Representations - Genetic and Evolutionary Computation Conference, 2006
- A. Moraglio, J. Togelius, S. Lucas "Product Geometric Crossover for the Sudoku Puzzle", IEEE Congress on Evolutionary Computation, pages 470-476, 2006
- A. Moraglio, H-Y. Kim, Y. Yoon, B-R. Moon, R. Poli "Generalized Cycle Crossover for Graph Partitioning", Genetic and Evolutionary Computation Conference, pages 1421-1422, 2006

References

- H-Y. Kim, Y. Yoon, A. Moraglio, B-R. Moon "Geometric Crossover for Multiway Graph Partitioning", Genetic and Evolutionary Computation Conference, pages 1217-1224, 2006
- A. Moraglio "Geometric Unification of Evolutionary Algorithms", European Graduate Student Workshop on Evolutionary Computation – European Conference on Genetic Programming, 2006
- A. Moraglio, R. Poli, R. Seehuus "Geometric Crossover for Biological Sequences", European Conference on Genetic Programming, pages 121-132, 2006
- A. Moraglio, R. Poli "Topological Crossover for the Permutation Representation", Italian Workshop on Evolutionary Computation - Italian Association of Artificial Intelligence Conference, 2005
- A. Moraglio, R. Poli "Geometric Landscape of Homologous Crossover for Syntactic Trees", IEEE Congress on Evolutionary Computation, pages 427- 434, 2005

References

- A. Moraglio, R. Poli "Topological Crossover for the Permutation Representation", Workshop on Theory of Representations - Genetic and Evolutionary Computation Conference, 2005
- A. Moraglio "Geometric Unification of Evolutionary Algorithms", British Colloquium for Theoretical Computer Science, page 251, 2005
- A. Moraglio, R. Poli "Topological Interpretation of Crossover", Genetic and Evolutionary Computation Conference, pages 1377-1388, 2004
- A. Moraglio "*Towards a Geometric Unification of Evolutionary Algorithms*", PhD thesis, University of Essex, 2007