A Gestalt Genetic Algorithm: Less details for better search

Christophe Philemotte
cphilemo@iridia.ulb.ac.be

Hugues Bersini
bersini@ulb.ac.be

IRIDIA - CoDE
Université Libre de Bruxelles CP 194/6
50 Avenue F.D. Roosevelt
B-1050 Brussels
Belgium

ABSTRACT

The basic idea to defend in this paper is that an adequate perception of the search space, sacrificing most of the precision, can paradoxically accelerate the discovery of the most promising solution zones. While any search space can be observed at any scale according to the level of details, there is nothing inherent to the classical metaheuristics to naturally account for this multi-scaling. Nevertheless, the wider the search space the longer the time needed by any metaheuristic to discover and exploit the "promising" zones. Any possibility to compress this time is welcome. Abstracting the search space during the search is one such possibility. For instance, a common Ordering Genetic Algorithm (o-GA) is not well suited to efficiently resolve very large instances of the Traveling Salesman Problem (TSP). The mechanism presented here (reminiscent of *Gestalt* psychology) aims at abstracting the search space by substituting the variables of the problems with macro-versions of them. This substitution allows any given metaheuristic to tackle the problem at various scales or through different multi-resolution lenses. In the TSP problem to be treated here, the towns will simply be aggregated into regions and the metaheuristics will apply on this new one-level-up search space. The whole problem becomes now how to discover the most appropriate regions and to merge this discovery with the running of the o-GA at the new level.

Categories and Subject Descriptors

I.2.8 [**Problem Solving, Control Methods, and Search**]: Heuristic methods

General Terms

Algorithms, Design, Experimentation

Keywords

Genetic Algorithm, Gestalt, Traveling Salesman Problem

1. INTRODUCTION

From macrocosm to microcosm, all biological systems are hierarchically structured in levels. They are often described as being more than the sum of their parts. We discussed in previous papers this key question often neglected: "for who or for which other observing system" [2, 18]. This is also the fundaments of *Gestalt* theory, as established by Max Wertheimer, Wolfgang Kölher and Kurt Koffka [1]. In *Gestalt* thinking, the perception is a combination of individual sensations, in which the environment is perceived in terms of its structure, through related macroscopic properties.

According to the Intrinsic Emergence concept [23, 3, 3], a complex system is mechanically observed by a functional device which is intrinsically defined by the system itself. Through this mechanical device, macroscopic and functional information are reintroduced in the process of the system. As in *Gestalt* theory, the system presents some form of internal feedback of the whole i.e "a whole perceived by something else but able to act on itself". Intrinsic Emergence and *Gestalt* psychology are very related concepts. Some authors [2, 18, 19, 20] have proposed a concrete interpretation of this concept in the EC framework: alterating of the perception of the search space in a useful way. It is done by considering only some of the details of the search space and by creating indivisible macroscopic genes, a "*Gestalt*".

In biology, cluster of genes and the processes that transmit them through inheritance have been noticed for many years. In 1959, Demerec and Hartman wrote: "regardless of how the gene clusters originated, natural selection must act to prevent their separation" and "the mere existence of such arrangements shows that they must be beneficial, conferring an evolutionary advantage on individuals and populations which exhibit them" [8]. In the last decades, with the boom of bioinformatics, the biologists have got the means to focus on and to more deeply investigate the importance of gene clustering in natural evolution. This mechanism could be interesting in EC framework just as the Darwin's evolution model was the starting point of EC.

In EC, several works have be done in this conceptual vein [2, 18, 19, 20, 7, 6, 26]. In this paper, we aim at developing a new version of the Observer Algorithm ([18, 19, 20]) but still persisting with this same conceptual framework. In particular, we will apply it to the TSP which is indeed a very well suited application to illustrate the whole idea. We warn the reader that the goal is to better understand this kind of mechanism and not to propose another best known algorithm to tackle the TSP. We still are at the beginning of our investigation of this idea.

From a cognitive perspective, a human perfecly follows this *Gestalt* approach when engaged in some form of TSP. For instance, Krolak has noticed that human builds some cluster of towns to resolve the problem [15] and does not take into consideration all details when finding the shortest path for a car trip. He first considers some regions as a whole, and performs the task at this level of abstraction. Afterwards, he focuses on the details and eventually refines the path. The Observer Algorithm (OA) to be presented and discussed in this paper behaves in a very similar way, improving an o-GA through the use of high-level perception of the search space: a *Gestalt* GA.

As in the Intrinsic Emergence concept, the observer is defined as a process that first builds, then evaluates and finally improves some lenses. The lenses alter the way the o-GA is observing the search space while exploring and exploiting it. The quality of these lenses is computed from the performance of the o-GA optimization. As a result of this evaluation, new lenses are proposed and supplied to the next run of the o-GA. OA and o-GA act simultaneously and in a very cooperative way.

In Section 2 and 3, we introduce both the TSP and o-GA. Section 4 describes the generic algorithm idea adopting the *Gestalt* framework. Section 5 explains the particular encoding that is required by this algorithm and its effect on the TSP. In Section 6, the intertwining between the o-GA and the OA is detailed. The obtained results on two particular instances of TSP (`dsj1000` and `p654`) taken from the TSPLIB are shown and discussed in Section 7.

2. TSP

In combinatorial optimization, the TSP is certainly one of the most studied, popular and representative problem. It is easy to state, to comprehend and typical of many real-world applications: finding the shortest tour trough a set of towns which have to be visited exactly once. Since it is a NP-hard problem [12], most of the efforts have been focused on the design of algorithms that can find satisfactory solutions in a decent time. Roughly, two families of algorithms are typically applied: local search and global search algorithms. Local search methods such as or-Opt or Lin-Kernighan (LK) rely on problem-specific knowledge [16, 17, 21]. It is no more the case for many global search methods, like EC, Ant Colony Optimization, Neural Networks, or Simulated Annealing, which want to remain as blind as possible to preserve their universal character. Whenever they are confronted by a very wide search space, complete blindness is definitely something to recover from. As a matter of fact, the current state-of-the-art methods to efficiently deal with TSP testify for this weakness and most of the global search methods are outperformed by their local search counterparts. The current best performing algorithms somewhat hybridize global and local search methods [24, 25].

3. O-GA

One of the most popular global search method applied to TSP is GA. It is well known that the performance of GA strongly depends on the way the problem is encoded in the individuals to be evolved [22]. While in GA the binary string representation is commonly used, it is not well suited to the TSP and generally not to ordering problems. This representation implies some kind of repairing processes which somewhat goes against the basic principles of GA. In the case of the TSP, the path representation [13] is the most simple and natural encoding driving to satisfactory performances. More complex representations like graphic image [14] allow for a considerable improvement in the results. Yet these complex representations are less flexible than the path representation.

Many crossover operators have been designed for solving the kind of combinatorial problems for which solutions are just permutations. They range from general operators such as the Partially Match Crossover (PMX) [13] to more problem-specific operators such as the Edge Recombination Crossover (ERX) [5]. In general, the most successful operators are combined further with local search methods. For instance, the Distance-Preserving Crossover (DPX) coupled to the LK method is one of the state-of-the-art crossover successfully applied to the TSP [11]. Inversion or mutation operators can as well be improved by the integration of some problem-specific methods or remain very unspecific.

4. A GESTALT TYPE OF ENHANCEMENT

In [19, 20], the authors have suggested an algorithm to improve the exploration process of a Simple GA (SGA). This algorithm allows the presence of masks to hide many details of the search space so as to accelerate the discovery of promising solution zones. A classical GA was applied on this new, abstracted and reduced search space, and indeed allowed the expected acceleration during the discovery process. The new algorithm they proposed was applied to the Hierarchical IF-and-only-iF problem (HIFF) and experiments demonstrated a considerable improvement with respect to the basic Simple GA. The whole exercise can simply be stated as trying to find for each problem the right observer. By skipping many details and drastically reducing the search space, an observer can allow a most rapid and efficient traveling though the original search space. If this is the general idea and remains a widely applicable one, any problem may still have its own unique and ideal way to define and optimize the observer. In our original experiments like in the experiments to follow, a double GA is used: one in the original search space and one in the space of the observers that observe this search space.

We propose here an extended and more solid form of the algorithm presented in [19] and applied on the HIFF problem. For sake of clarity, we will expose this new extended version on the TSP problem. Now despite this precise choice of this very familiar problem, we hope the readers will accept that the basic idea of this space-abstracting observer is a very general one that can easily be transposed to any metaheuristic and for any optimization problem (see Figure 1). Here, this notion of observer is considered in a more precise way. The notion rather consists in a process which both observes and then evaluates how well a given metaheuristic performs in the search space. Let's suppose that a given step of the algorithm, some lenses exist that allow to observe the search space with much less precision. The classical optimization goes on but over the search space perceived through these lenses. Afterwards, the quality of these lenses is assessed on the basis of the performance of this optimization. Based upon these evaluations, a new generation of lenses is proposed and the complete sequence is repeated

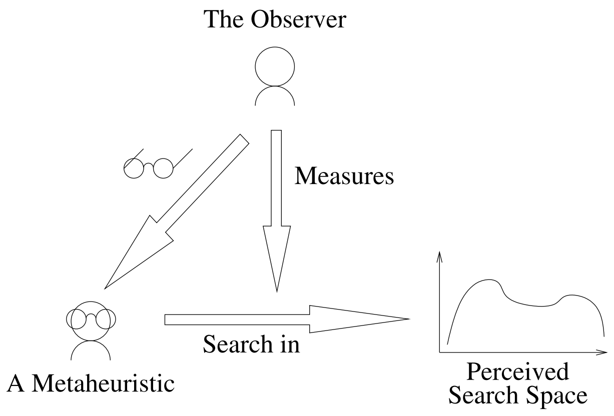

Figure 1: Let's suppose that a given step of the algorithm, some lenses exist that allow to observe the search space with much less precision. The classical optimization goes on but over the search space perceived through these lenses. Afterwards, the quality of these lenses is assessed on the basis of the performance of this optimization. Based upon these evaluations, a new generation of lenses is proposed and the complete sequence is repeated again up to find the most adequate lenses.

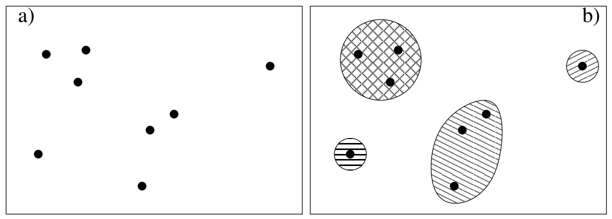

Figure 2: The cities (a) are grouped into regions (b). The search is then operated at this new level.

again up to find the most adequate lenses. In general, these lenses can be viewed as Grouping Templates (GT) which are applied on the encoding of the problem at hand. In the specific case of the TSP, the cities are regrouped into regions, and the restriction of the search space is explained by a search over the possible permutations of the regions instead of the cities (see Figure 2). The complete algorithm merges two different searches: one over the lenses spaces and one over the original search space. The search over the definition of the lenses is guided by evaluations provided by the search executed using these lenses over the original search space. In [19, 20], the Observer mainly supervises the given metaheuristic and always restarts the experiments from the same observation. In this paper, the Observer mechanism acts all along the given metaheuristic searches for good candidate solutions and at fixed time intervals.

The first step evaluates the quality of the lenses (i.e. a GT) according to the candidate solutions found after some generations of the metaheuristic. The lens is rewarded on the basis of its assistance to the task of the metaheuristic. The way this evaluation is computed is discussed in the following. The second step builds new lenses on the basis of the previous ones and their respective quality. The third step closes the loop by applying the new lenses to the candidate solutions populations and the complete sequence is repeated again. Through each lens, the metaheuristic perceives the search space in a complete new way. Some parts will not be viewed as a juxtaposition of candidate solutions. They are viewed as a whole, i.e. a macroscopic candidate solution. The effect of the lenses is thus to make the metaheuristic operating in another manner. The metaheuristic operates while adopting a multi-scale perception of the search space. It is very reminiscent of *Gestalt* psychology.

Algorithm 1 The new Observer Algorithm

```
initialize lenses population
initialize solutions population
while not Stop-Condition do
  if some lenses are obsoletes then
    evaluate the lenses efficiency
    build new lenses w.r.t. their efficiency
    alter the solutions population
  end if
  operate the given metaheuristic on the solutions population
end while
```

5. GROUPING TEMPLATE

A second optimization process is performed in the space of possible GTs. In previous works, the authors proposed to generate the GTs randomly [2], or evolutionarily [18, 19, 20]. The GT were encoded as a mask for computing the assignment of each gene to each group [18, 19, 20]. This strategy had an important drawback: not all possible grouping schemes could be represented. Here, we propose to consider the GTs search as a grouping problem and therefore to apply a Grouping Genetic Algorithm (GGA).

The GGA was originally proposed by Falkenauer [9]. By an appropriate encoding and genetic operators, it is designed to extend the range of GA applicability to grouping applications. In the rest of the chapter, we will show and explain how the repartition of the cities in the possible regions is indeed an instance of a grouping problem. The encoding consists of two parts: the cities part and the regions part. The cities part is composed of the n cities of the TSP (more generally the n objects to be optimally grouped). The regions part is a permutation of the k region labels which are assigned to each city belonging to this region (see Figure 3). For instance, if our problem consists in 10 cities labelled 0, 1, 2, 3, 4, 5, 6, 7, 8 and 9, a grouping template could be defined (in the case of five regions labelled a, b, c, d and e) as such *abbceddeac* : *abedc* i.e. $a = \{0, 8\}$, $b = \{1, 2\}$, $c = \{3, 9\}$, $d = \{5, 6\}$ and $e = \{4, 7\}$.

The GGA operators act over the regions part of the chromosomes and correspondingly modify the cities part. The same Grouping operators as the one originally described by Falkenauer in [9] will be used with small changes. We refer to [9] for details about the original operators. The inversion operator is not modified. The small changes mainly apply to the heuristic of reassignment of the cities by indeed taking profit of the spatial specificities of the problem. This heuristic is used both by the mutation and the crossover operators. Here, an unassigned city x is preferentially grouped with one

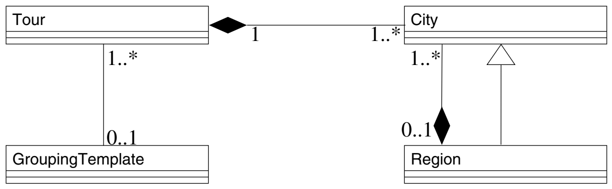

Figure 3: An object view of the adapted encoding of TSP solutions. A solution is associated to a GT. The GT defines how the cities are grouped in the solution. But it does not define the order into the region or the order of the region. A city could be considered as a simple city or a region.

of its spatial neighbors y (see Figure 4.c). The neighborhood size is a parameter S_n. In the case of the mutation operator, another change needs to be explained. If a mutation occurs, the selected city could be assigned either to an exiting region or to a new region. The probability of the occurrence is $P_{occ} = N_R/N_C$ (N_R is the number of regions, N_C is the number of cities) so that this probability increases with the number of regions. The crossover and inversion operators are always applied. The mutation is applied with a given probability P_m.

The initialization of the population is realized in a uniform random way. The selection of parents is a classic tournament of size 2. The parents are recombined in order to produce two new GTs. This low variability between two successive generations aims at maintaining a sufficient diversity in the GT population.

To be selected, the GTs are evaluated on the basis of the performance of the associated solution population on which they apply. Each GT acts on a restricted part of the whole solution population. The size of the partition is fixed and each GT applies on a subpopulation composed of a constant number of individuals. The evaluation of one GT acting over its own subpopulation is expressed as:

$$f_{GT} = \sqrt{\overline{f_{sol}} \times \min(f_{sol})}\ , \qquad (1)$$

where $\overline{f_{sol}}$ is the average of the individual fitness over the solution population and $\min(f_{sol})$ is the minimum of the individual fitness values. A GT should be as good as it produces a population of average good quality with some elites. The efficiency of a GT is indeed only dependent on the quality of its associated subpopulation of solutions. The TSP is a minimization problem and obviously the discovery of the best GT is also one.

6. O-GA ENHANCED BY OA

For the experiments to be presented in this paper, this whole Observation and *Gestalt* mechanisms have been grafted to an o-GA. Right now, we are not at all running after the best TSP algorithm but simply showing how this addition facilitates the exploration of wide landscapes whatever basic operators are engaged in this exploration. The Observer addition boils down to macroscopically sample the search space in order to more rapidly discover promising zones. Due to the addition of the GTs, o-GA does not operate directly at the city level but rather at the region level (See Figure 4.a and 4.b). The o-Ga can not look inside a region and so it can

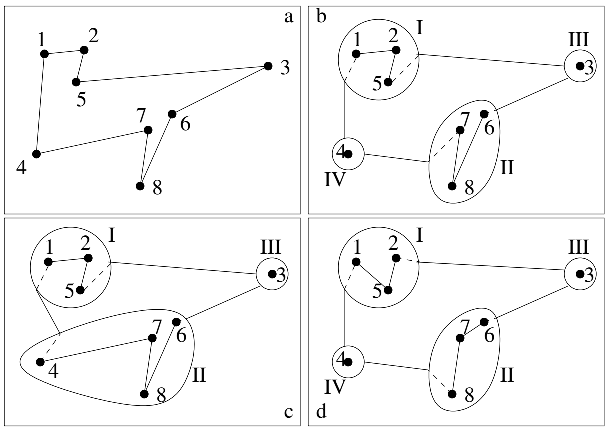

Figure 4: a) A possible tour T through eight cities. b) A GT x defines four regions: $I = \{1, 2, 5\}$, $II = \{6, 7, 8\}$, $III = \{3\}$ and $IV = \{4\}$. This GT is applied on the given plain tour T. The order in and of each region is then defined. From this point, the intra-order is fixed and can only be modified by new GT. c) From the GT x and another one, we obtain a new GT y. Our tour T is associated to this GT y. The region IV is deleted. The city 4 is moved to the region II. Because of its previous relative order from the case b), the city 4 becomes a border city of the region II. d) For another possible tour S and the GT x, we can obtain other intra-order.

Algorithm 2 GGA Implementation of the Observer Algorithm

```
initialize GT population
for all GT do
  initialize its solutions subpopulation
end for
while not Stop-Condition do
  if some GT are obsoletes then
    evaluate each GT
    select two GT parents with a tournament
    apply crossover to these parents
    replace the worst GT with the two generated children
    mutate the two children with some probability
    apply inversion to these children
    mix the solutions pop of the selected GT parents
    create two sol pop with and associate them to children
  end if
  operate a tep of the given metaheuristic on the solutions population
end while
```

not modify the ordered sequence of cities constituting this region. The GT creates a kind of barrier in the representation: a *Gestalt* encoding of the problem. This barrier offers two ways of evaluating the quality of a given GT. Inside a region, different order are possible (see Figure 4.b and 4.c). The more orders we have, the more precise the sampling of the quality of the corresponding region is. If independently of such sequence, this region remains a good one, it is indeed an adequate macrovariable to build a new search on. If the partition is good and all regions constitute an adequate division of the search space, the genetic operators will better operate because they won't need to take all details into account.

The o-GA uses overlapping populations, tournament selection of size 2, PMX or ERX and swap adjacent loci mutation. The population is uniformly initialized and associated to a population of different GTs, each GT dedicated to one sub-part of the complete population. When following a crossover operation, two GTs produce two new GTs, their corresponding solutions populations are mixed in order to generate two new solutions populations for the offspring. This is akin to a migration of some candidate solutions to the new sub-populations.

7. RESULTS

We have designed two different experiments. The first experiment compares the deviation from the optimum of the best found solution fitness $\delta_f = (f_{sol} - f_{opt})/f_{opt}$ in function of each algorithm, i.e. o-GA with PMX, o-GA with ERX, and their versions coupled with the OA (see Figure 5 and 6). The computation time[1] has been chosen as computation effort measure because of the possible load that OA implementation introduces. The number of trials is 50. Only 600 seconds are given to algorithms to find the best possible solution. The total size of the candidate solutions population is 1000. The mutation rate for the o-GA and OA is 0.2. The considered neighbor size S_n for the OA is 10. The worsts GTs become obsolete after 100 steps of o-GA and are replaced by offspring. The initial GTs define only two regions. The results are shown with boxplot defined by non parametric statistics: minimum, first quartile, median, third quartile and maximum. The tackled instances of TSP are `dsj1000` and `p654` from the TSPLIB[2]. They have been chosen for their cluster structure. The second experiment gives us an perspective of one run of OA added to o-GA with PMX for `dsj1000` during 6000s. The evolution of δ_f is plotted as a function of the computational time (see Figure 7).

The o-GA with PMX gives the worst results for both TSP instances (see Figure 5 and 6). It quickly gets stuck in local optima and it does not succeed in finding a better solution. PMX is a generic crossover. Indeed, it is designed to be a trade-off between the transmission of relative order, absolute order and adjancy relations present in the parent permutations to offspring permutations. PMX is more flexible but performs worse than crossover suited to a specific problem. The obtained results are hence not surprising.

The o-GA with ERX gives the best central tendency for both instances (see Figure 5 and 6). It performs well on TSP,

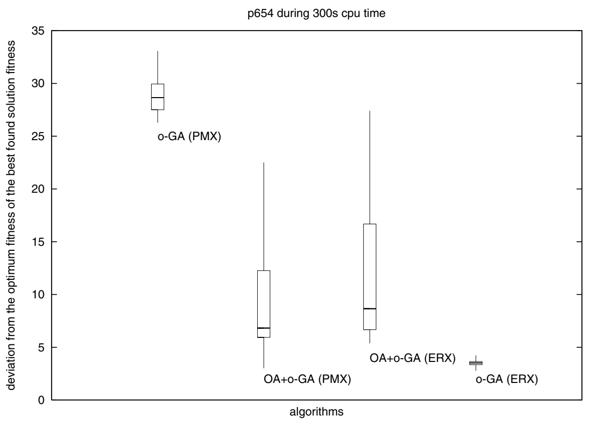

Figure 5: Obtained results for the TSP instance `p654` with the four algorithms, i.e. o-GA with PMX, o-GA with ERX, and their versions coupled with the OA. The x axis represents the deviation of the best found solution fitness from the optimum fitness after 600 seconds of computation time. The y axis indexes the four different algorithms. For each algorithm, a boxplot is shown and represents two statistical measurements: one of central tendency and second of statistical dispersion.

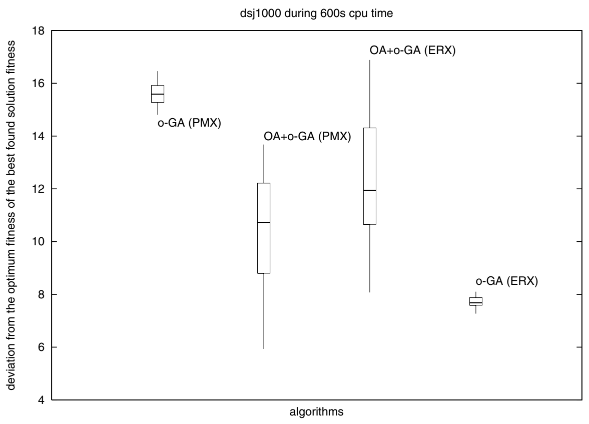

Figure 6: Obtained results for the TSP instance `dsj1000` with the four algorithms, i.e. o-GA with PMX, o-GA with ERX, and their versions coupled with the OA. The x axis represents the deviation of the best found solution fitness from the optimum fitness after 600 seconds of computation time. The y axis indexes the four different algorithms. For each algorithm, a boxplot is shown and represents two statistical measurements: one of central tendency and second of statistical dispersion.

[1]The used machine is an iBook G4 rev.2005 1,40Ghz with 1GB of RAM. The operating system is a Gentoo Linux.

[2]http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/

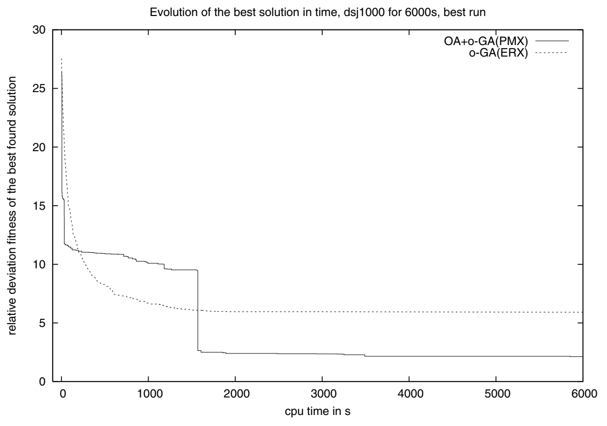

Figure 7: A representative run of the o-GA with PMX coupled with OA and the o-GA with ERX on the `dsj1000` TSP instance. The deviation from the optimum fitness of the best found solution fitness is plotted as a function of the computation time in seconds. The maximum computation time is 6000s. o-GA(PMX)+OA: Four phases can be noticed. First, a quick and drastic improvement in term of fitness. Second, a slowly decreasing plateau during which the OA is looking for good definition of the regions. Third, a strong rupture. Fourth, a nearly plateau very slowly decreasing to the optimum. o-GA(ERX): A smooth exponential decrease.

and is one of the best known pure genetic crossover for tackling TSP. ERX considers a solution as a tour of cities and is defined over circular permutations. It aims at transmitting adjancy relations from parents to offspring. The dispersion is very small, and the algorithm behavior is then very reproductible.

The o-GA with ERX coupled with OA provides the second worse central tendency and a large dispersion for both instances (see Figures 5 and 6). In this case, OA clearly degrades the power of ERX. The adjancency relation is adequate to the city level but it is not suited to the region level. Its interpretation of the permutation of region becomes unrelevant at this level.

The o-GA with PMX coupled with OA provides the second best central tendency, a wide dispersion and the best minimum for both instances (5 and 6). This time, OA provides good results with the PMX. The interpretation of the PMX is still valid at the region level. Absolute order, relative order and adjancy relations are relevant properties w.r.t. the regions definition (i.e. the GTs). The wide dispersion of results makes our approach not so much reproductible as ERX is. This is a clue for improving the cooperation between OA and the helped algorithm. It is clear that the chosen implementation uses many different evolutionary techniques. We have to better understand the role of the symbiose between the macroscopic and microscopic level. The only way for designing a better implementation of OA.

For the second experiment, we have chosen a representative run of the o-GA with PMX coupled with OA and of the o-GA with ERX on the `dsj1000` TSP instance. The obtained results with this run are close to the central tendency. With the corresponding seed, we have run it again but for a longer time, i.e. 6000 seconds. We can notice four phases (see Figure 7). At first, the used regions are big. o-GA with PMX can quickly converge to a first promising area in the search space. This drastically improves the candidate solutions. In a second phase, OA begins to look after good regions definition. This takes about 1500s. Afterwards, a big rupture occurs. This means that the OA finds a better way to help the o-GA in its processing and provides to the o-GA a good perception of the search space. In a last phase, the couple of algorithms perform with more difficulties. The minimization becomes slower. This behavior tell us that the refining process of OA is not good. The assignment of a city to a new city occurs with the probability P_{occ}. By analizing the obtained regions, we have noticed that regions are only maded of two cities on average. This is linked to the previously given probability. It is a clue to improve the refining process of our OA implementation. For the o-GA with ERX run, the candidate solutions are improved following a smooth exponential decrease. The final plateau is slightly worse than the results obtained during the run of the o-GA with PMX coupled with OA. These final results make us confident into the proposed OA. Because of the duration for finding a good region, we have to run longer our experiments. The comparison will be more precise with other algorithms.

8. CONCLUSION

This work is motivated by linked concepts such as *Gestalt* theory, Intrinsic Emergence, and facts of biological genes cluster. In this framework, we suggest a new design of the OA which is described in [19]. We add this version to o-GA to tackle TSP. TSP is chosen for well illustrating this *Gestalt* GA. With OA, a city could be considered as a city or a region. o-GA then perfoms at a different level of abstraction. The new design includes different techniques: naive cooperative coevolution between GT and solution tour, a new encoding of GT and a GGA to optimize the region definition (i.e. GT).

The obtained results are good and show an improvement when a generic crossover like PMX is used. From these results, we have exposed some if the weakness of our design such as the refining process. These experiments encourage us in this conceptual approach.

But behind this small experimental and descriptive investigation of this *Gestalt* algorithm, we have underlined the role that leveled structure could play in nature inspired metaheuristic such as EC. The Intrinsic Emergence concept modelizes the information flow through the barrier between the parts and the whole. And the obtained results provide good confidence in its realization by OA.

9. REFERENCES

[1] M. Ash. *Gestalt Psychology In German Culture 1890 - 1967*. Cambridge University Press, 1995.

[2] H. Bersini. Whatever emerges should be intrinsically useful. In *Artificial life 9*, pages 226–231. The MIT Press, 2004.

[3] J. Crutchfield. Is anything ever new? considering emergence. In D. P. G. Cowan and D. Melzner, editors, *Integrative Themes*, volume XIX of *Santa Fe Institute Studies in the Sciences of Complexity.*

Addison-Wesley Publishing Company, Reading, Massachusetts, 1994.

[4] J. Crutchfield and M. Mitchell. The evolution of emergent computation. In *Proceedings of the National Academy of Science*, volume 23, page 103, 1995.

[5] D. F. D. Whitley, T. Starkweather. Scheduling problems and traveling salesman: The genetic edge recombination operator. In *Proceedings of the 3rd International Conference on Genetic Algorithms (ICGA)*, pages 133–140, 1989.

[6] A. Defaweux, T. Lenaerts, and J. I. van Hemert. Evolutionary transitions as a metaphor for evolutionary optimisation. In *Advances in Artificial Life, 8th European Conference, ECAL 2005, Proceedings*, volume 3630 of *Lecture Notes in Computer Science*, pages 342–352. Springer, 2005.

[7] A. Defaweux, T. Lenaerts, J. I. van Hemert, and J. Parent. Transition models as an incremental approach for problem solving in evolutionary algorithms. In *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*, pages 599–606, New York, NY, USA, 2005. ACM Press.

[8] M. Demerec and P. Hartman. Complex loci in microorganims. *Annual Review of Microbiology*, 13:377–406, 1959.

[9] E. Falkenauer. A new representation and operators for genetic algorithms applied to grouping problems. *Evolutionary Computation*, 2(2):123–144, 1994.

[10] D. Fogel. Applying evolutionary programming to selected travelling salesman problems. *Cybernetics and Systems : an International Journal*, 24:27–36, 1993.

[11] B. Freisleben and P. Merz. A genetic local search algorithm for solving symmetric and asymmetric traveling salesman problems. In *Proceedings of IEEE International Conference on Evolutionary Computation 1996*, pages 616–621, 1996.

[12] M. R. Garey, R. L. Graham, and D. S. Johnson. Some np-complete geometric problems. In *STOC '76: Proceedings of the eighth annual ACM symposium on Theory of computing*, pages 10–22, New York, NY, USA, 1976. ACM Press.

[13] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Professional, 1989.

[14] S. Jung and B. R. Moon. Toward minimal restriction of genetic encoding and crossovers for the two-dimensional euclidean tsp. *IEEE Trans. Evolutionary Computation*, 6(6):557–565, 2002.

[15] P. Krolak, W. Felts, and G. Marble. A man-machine approach toward solving the traveling salesman problem. *Commununications of the ACM*, 14(5):327–334, 1971.

[16] S. Lin and B. Kernighan. An effective heuristic algorithm for traveling salesman problem. *Operations Research*, 220(4598):498–516, 1973.

[17] I. Or. *Traveling salesman-type problems and their relation to the logistics of regional blood banking*. PhD thesis, Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL, 1976.

[18] C. Philemotte and H. Bersini. Coevolution of effective observers and observed multi-agents system. In *Advances in Artificial Life, 8th European Conference, ECAL 2005, Proceedings*, volume 3630 of *Lecture Notes in Computer Science*, pages 785–794. Springer, 2005.

[19] C. Philemotte and H. Bersini. How an optimal observer can collapse the search space. In *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 1273–1280, New York, NY, USA, 2006. ACM Press.

[20] C. Philemotte and H. Bersini. How an optimal observer can smooth a landscape. In *IEEE Congress on Evolutionary Computation 2006*, pages 2293–2300. IEEE, 2006.

[21] G. Reinelt. *The traveling Salesman: Computational Solutions for TSP Applications*. Springer-Verlag, 1994.

[22] F. Rothlauf and D. E. Goldberg. *Representations for Genetic and Evolutionary Algorithms*. Physica-Verlag, 2002.

[23] L. Steels. Towards a theory of emergent functionality. In *Proceedings of the first international conference on simulation of adaptive behavior on From animals to animats*, pages 451–461, Cambridge, MA, USA, 1990. MIT Press.

[24] T. Stützle. *Local Search Algorithms for Combinatorial Problems - Analysis, Algorithms, and New Applications*. PhD thesis, Department of Computer Science, Darmstadt University of Technology, 1998.

[25] T. Walters. Repair and brood selection in the traveling salesman problem. In A. Eiben, T. Back, M. Schoenauer, and H.-P. Schwefel, editors, *Proceedings of the Fifth International Conference on Parallel Problem Solving from Nature, Amsterdam, The Netherlands*, volume 1498 of *Lecture Notes in Computer Science*, pages 813–822. Springer-Verlag, 1998.

[26] R. A. Watson. *Compositional evolution: interdisciplinary investigations in evolvability, modularity, and symbiosis*. PhD thesis, Brandeis University, 2002. Adviser-Jordan B. Pollack.