Comparisons of meta-heuristics for the layered digragh drawing problem

BRUNO PINAUD PASCALE KUNTZ REMI LEHN
Ecole Polytechnique de l'Université de Nantes
Université de Nantes
La Chantrerie BP50609 44306 Nantes
FRANCE

Abstract: In this paper we present a Genetic Algorithm (GA) with problem-specific operators for the layered digraph drawing problem and compare it with two very different meta-heuristics: Tabu search and Multi-start descents. Tabu search has previously proved to be better than the classical deterministic local heuristic often employed in graph drawing. Here, we show on a set of 282 medium-sized graphs that GAs lead to similar results to Tabu for graphs of relatively high density and better results for graphs with medium density. Comparisons with multi-start descents tend to confirm the good quality of the solutions found by using GAs.

Key-Words: meta-heuristics, combinatorial optimization, graph drawing, genetic algorithm, tabu search, multi-start descents

1 Introduction

Today graph drawing problems know a renewed interest in particular in the field of "information visualization" (Web browsing, cartography, ...) [5]. Graphs can be used at the same time as theoretical models and as efficient visualization supports which often allow access to complex structures without getting bogged down in mathematical detail [1].

The problem of drawing a graph G is generally set as a combinatorial optimization problem [2]: producing a layout of G on a given support (plane, ...) according to a drawing convention that optimizes some measurable aesthetics of which one of the most important for readability is arc crossing minimization [13]. In this paper, we focus on an approach often used for hierarchical relationship representations: the *layered drawings of directed graph*s. In this case, vertices of G are arranged on vertical layers and the arcs linking vertices are represented by oriented line segments which flow in the same direction. Minimizing arc crossing for this layout problem could seem intuitively easier than the general problem of minimizing arc crossing on a plane since the choice of geometric coordinates is here replaced by a choice of vertex ordering on each layer. Unfortunately, it remains *NP*-complete [4].

The importance of the problem in numerous applications has stimulated the development of various heuristics from local transformations to meta-heuristics. The easiest deterministic local transformations are based on simple permutations on each layer [3, 6]. The most popular ones are the greedy-switching and the so-called "averaging heuristics". The greedy-switching permutation iteratively switches consecutive pairs of vertices if it decreases the crossing number. The averaging heuristic which includes the barycenter heuristic [14] and the median heuristic [4] is based on a shrewd remark: arc-crossings tend to be minimized when connected vertices are placed facing with each other. Roughly speaking, these algorithms compute the average position -e.g. the barycenter or median of their neighbors- for each vertex and sort them on each layer according to the obtained values.

To overcome the difficulties inherent to these local approaches (see [2] for details), different meta-heuristics have been proposed in the past years: Tabu search [10], GRASP [11], Genetic algorithms and Evolutionary algorithms [15]. In particular, GAs have been shown to be particularly promising for a generalization of the problem presented here which takes dynamic updates of the graph into account [9]. However, as far

as we know, there have been few comparisons between GAs and other meta-heuristics.

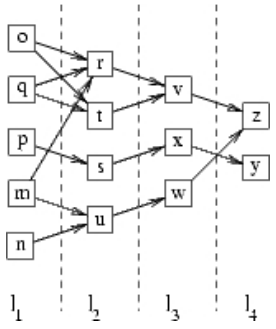
In order to better understand the GA behavior for the layered drawing problem, we here propose to compare it with two approaches based on very different principles: the Multi-start descents and the Tabu search which allows an exploration of the search space from a restricted set of potential solutions which does not necesarily improve the objective function.

The first section presents a Genetic Algorithm with problem-specific mutation and crossover operators. In particular, this version introduces a hybridization with an adaptation of the averaging heuristics presented above in the mutation phase. In the second section, GAs are compared with Tabu search and Multi-start descents on a large set of random layered medium-sized digraphs.

2 Problem formulation

Let $G = (V, A)$ be an acyclic digraph with a set V of n vertices and a set A of m arcs, and $L = \{l_1, l_2, ..., l_K\}$ be a set of K layers. The distribution of the vertices on each layer is here considered to be given: V_i, $i = 1, ..., K$, is the subset of vertices which must be placed on L_i, $i = 1, ..., K$. In a layered drawing (see figure below), every arc (u, v) flows in the same direction: if $u \in V_i$ then $v \in V_j$ where $i < j$. Moreover, as usual for this problem, we suppose that the graph is proper i.e. each arc $(u, v) \in A$ is connected to vertices on consecutive layers: if $u \in V_i$ then $v \in V_{i+1}$. In practice, it is easy to come down to this hypothesis when replacing an arc whose length λ is greater than one by a path of $\lambda - 1$ dummy vertices on consecutive layers.

The layered drawing problem consists in defining a vertex ordering on each layer so that the associated drawing minimizes the arc crossing number. In the following, the vertex ordering on $l_k \in L$ is denoted by σ_k: $\sigma_k(u) = p$ means that the vertex $u \in V_k$ is on the p^{th} position on l_k. And, the arc crossing number for a drawing D_i is denoted by $f(D_i)$.



The above figure is an example of a layered digraph with 5 arc crossings ($f(D) = 5$). In this example $V_1 = \{o, q, p, m, n\}$, $\sigma_1(o) = 1$, $\sigma_1(q) = 2$, etc. and $V_2 = \{r, t, s, u\}$, $\sigma_2(r) = 1$, $\sigma_2(s) = 3$, etc.

3 A genetic based approach

It is well-known that in the basic framework, GAs work with a population of potential solutions coded by genotypes which stochastically evolves by means of three basic operators: selection, recombination and mutation.

As for us, a genotype codes the vertex ranks in the successive orderings $\sigma_1, \sigma_2, ..., \sigma_k$ associated with each layer. The selection is determined by a classical roulette wheel based on the objective function f. Differences with the basic scheme occur at mutation and recombination stages.

3.1 Mutation operators and hybridization

Besides the classical bitflip operator, which here exchanges two vertices in a layer randomly chosen, we apply three problem-specific operators close to the deterministic heuristics (switching, median and barycenter heuristics) presented in section 1. Variants of these operators have already been successfully tested in the other meta-heuristics previously quoted. Note that in this case a local improvement is introduced in the "mutation" phase -in the sense of a hybridization strategy with a local search. Each local operator is here defined for any layer l_k of a layout D_i.

Switching (S). A drawing D_j is deduced from a drawing D_i by swapping two adjacent vertices in the same layer l_k if $f(D_j) < f(D_i)$.

Barycenter (B). A vertex v on l_k is repositioned at the average position $avg(v)$ of the connected vertices in both l_{k-1} and l_{k+1}:

$$avg(v) = \frac{\dfrac{\sum_{u \in N_{k-1}(v)} \sigma_{k-1}(u)}{|V_{k-1}|} + \dfrac{\sum_{u \in N_{k+1}(v)} \sigma_{k+1}(u)}{|V_{k+1}|}}{|N_{k-1}(v)| + |N_{k+1}(v)|}$$

where $N_{k-1}(v)$ (resp. $N_{k+1}(v)$) is the set of the neighbors of v on l_{k-1} (resp. l_{k+1}). Ratios $1/|V_{k-1}|$ and $1/|V_{k+1}|$ are introduced in the sum to normalize the vertex position on each layer according to their cardinality. The average position is computed for each ver-

tex of l_k and we retain the new vertex ordering σ'_k obtained after sorting the average values: $\sigma'_k(u) > \sigma'_k(v)$ if $avg(u) > avg(v)$.

Median (**M**). This operator is very close to the previous one. Let us recall that the median of a sorted set of numbers is the number med so that half of the numbers are smaller than med and the other half are greater than med. If $N_{k-1}(v) = \{u_1, u_2, ..., u_p\}$ and $N_{k+1}(v) = \{w_1, w_2, ..., w_q\}$ then the median position $med(v)$ of the vertex v is the median of the following sorted set $\{\frac{\sigma_{k-1}(u_1)}{|V_{k-1}|}, ..., \frac{\sigma_{k-1}(u_p)}{|V_{k-1}|}, \frac{\sigma_{k+1}(w_1)}{|V_{k+1}|}, ..., \frac{\sigma_{k+1(w_q)}}{|V_{k+1}|}\}$. Like for the barycenter, we retain the new vertex ordering σ''_k obtained after sorting the median values: $\sigma''_k(u) > \sigma''_k(v)$ if $med(u) > med(v)$.

Each operator is iteratively applied on each layer with a given probability.

3.2 Crossover operators

Contrary to GA applications for graph drawing on a plane where only real coordinates are taken into account (e.g. [12], [8]), the main difficulty for ordinal coding is to define a crossover which guarantees a feasible solution. In a previous paper [9], we compared two specialized crossover operators which had been previously applied to problems encoded as permutation such as the Traveling Salesman Problem: Order Crossover 1 and Partially Mapped Crossover (see [7] for details). We have showed that the second one was significatively too time consuming and that the results where not any better for our problem. Hence, we have here used a variant of Order Crossover 1 for an "intra-layer" operator (C_{intra}) and a usual one point crossover for an "inter-layer" operator (C_{inter}).

Let g_1 and g_2 be two genotypes called parents. The "inter-layer" operator works as follows: a crossover point cp is randomly chosen between two layers to create two new genotypes g_3 and g_4. The genotype g_3 is composed of the g_1 orderings in the layers preceding cp and the g_2 orderings in the layers after cp. Similarly, g_4 is defined by the orderings in the remaining layers: the first layers of g_2 and the last layers of g_1. For the "intra-layer" operator a crossover point is randomly chosen inside a layer, and then, the layers of the same index in the parents are mixed together to make up two new valid genotypes. The first crossover operator acts like a "vertical cut" in the drawing; It allows to interlace layers without changing the vertex ordering. And, the second operator acts as an "horizontal cut" in the drawing which can be used to construct new orderings different from those of the two parents.

4 Experimental comparisons

We compare GAs with multi-start descents and the Tabu algorithm developed in [10] which has been experimentally proved to be more efficient than the classical local deterministic heuristics.

To make experimental computations easier we have developed a layered acyclic digraph generator. Comparisons have been made on a set of 282 connected graphs with $4 \leq K \leq 40$ and $3 \leq |V_k| \leq 15$.

4.1 Results for Multi-start descents and Tabu search

Multi-start descents (MsD). A set of 1000 layouts of a graph G is randomly generated, and each layout is improved by an iterative application on each layer of the three "mutation" operators **S**, **M** and **B**. The best layout is retained at each step.

Tabu search (TS). We have chosen the TABU2 version presented in [10] as it gives better results than most basic implementations. Let us very briefly define the main steps. The search starts with an "intensification phase": a single layer is selected and the algorithm tries to find an optimal or near-optimal ordering of this layer, considering that the orderings of adjacent layers are fixed. This layer is then considered tabu as long as its adjacent layers are not modified. When all layers are tabu, a "diversification phase" is applied on randomly chosen vertices by the application of a switching procedure.

For GAs we have chosen the following probabilities for the different operators: $Pr(S) = 0.05$, $Pr(M) = Pr(B) = 0.2$, $Pr(C_{inter}) = Pr(C_{intra}) = 0.2$. We have tested other distributions but the results are very close for small variations (± 0.1 around these values). The initial population contains 30 drawings of a same graph.

In order to better understand the behavior differences between the different approaches we use two criteria: the relative height to the best found solution and the graph density. The relative height $h_H^{H'}(D_i)$ of a drawing D_i produced by a meta-heuristic H to the best solution produced by a meta-heuristic H' is defined by $h_H^{H'}(D_i) = 1 - \frac{f(D_i) - \widehat{f}_{H'}}{f_{init} - f_{H'}}$ where $\widehat{f}_{H'}$ is the best value found by H' and f_{init} the arc crossing number of the drawing given by the generator. If $h_H^{H'}(D_i)$ is close to 1, then the quality of D_i is similar to the quality of the

Best Methods	% of graphs	Avg Density *(std dev)*	h_{GA} *(std dev)*	h_{TS} *(std dev)*
MsD=TS=GA	30.5	0.8 *(0.19)*		
MsD	15.25	0.5 *(0.18)*	$h_{GA}^{MsD} = 0.96$ *(0.08)*	$h_{TS}^{MsD} = 0.9$ *(0.04)*
MsD=GA	54.25	0.61 *(0.22)*		$h_{TS}^{MsD} = h_{TS}^{GA} = 0.85$ *(0.12)*

Table 1: Numerical comparisons of Genetic Algorithm (GA), Multi-Starts descents (MsD) and Tabu Search (TS). (MsD, TS and GA give equivalent results for 30.5% of graphs,...)

best solution found by H'. The graph density $d(G)$ of a graph G is often introduced for graph class discrimination. The usual definition is the ratio of the arc number m on the arc number of a complete graph with a maximal arc number. In a layered digraph, the maximal graph has $m_{max} = \sum_{k=2}^{K} |V_{k-1}| \times |V_k|$ arcs. Hence, the density is here defined by $d(G) = m/m_{max}$.

Comparative results are given in table 1. In all cases, TS has never produced better drawings than GAs or MsDs. The cases of congruence between the three approaches seem to correspond to graphs with a relatively high density. Vertices of these graphs have numerous neighbors and consequently there are less efficient local permutations than for graphs with a medium density. For 15.25% of cases -which corresponds to graphs with a median density-, MsDs lead to better results than TS and GAs. However, the results of GAs are closer to the best obtained drawings ($h_{GA}^{MsD} = 0.96$) than those obtained by TS. Note that MsDs have been here introduced only to compare the quality of the results. In practice, it is obviously unthinkable to generate a set of 1000 drawings and to apply a descent on each of them. Yet, it is interesting to observe that GAs, which are distinctly quicker, lead to equivalent results than MsDs for 54.25% of the graphs.

4.2 Convergence

Figure 1 describes the average convergence of GAs. The number of generations necessary to reach a good quality solution is relatively small compared to basic GAs for other combinatorial optimization problems. Two factors contribute to explain this behavior. In our implementation, the initial population of drawings produced by the generator has been slightly improved. The vertex ordering of the first genotypes fits different depth first searchs of the graph: a directed one starting with vertices having a null inferior half-degree, a backward one starting with vertices with a null superior half-degree, etc.. This tends to initialize the exploration process with potentially suitable solutions. Moreover, the introduction of hybridization in mutation operators significantly improves the drawing quality at each generation.

Note that the optimization of the computation time was not our priority in this paper. Consequently, for this first stage, the three meta-heuristics have been developed in Perl which allows to notably reduce the development time by using complex data structures. This is the reason why computation times are high here: around 15 minutes for Tabu, 35 seconds for GAs and 70 minutes for MsDs on an SGI Origin 2000. In C++ language, Laguna et al. [10] have measured a mean computation time of around one minute. Moreover, the Tabu version chosen here for the quality of its results was known to converge more slowly than other versions. Besides, the first experiments with a new version of GAs coded in C show a convergence within a very few seconds.

5 Conclusion

In this paper we have presented a GA with problem-specific operators for the layered digraph drawing problem and compared it with two "opposite" meta-heuristics: Tabu search and Multi-start descents. Tabu search has been previously proved to be better than the classical deterministic local heuristic often employed in graph drawing. Here, we show on a set of 282 medium sized graphs that GAs lead to similar results to Tabu for graphs of relatively high density and better results for graphs with medium density. Comparisons with 1000 descents tend to confirm the good quality of the solutions found by GAs.

At present, our research is going on in three directions. We extend our comparisons to a class of larger graphs in order to better understand the influence of the density criterium on the different meta-heuristic behaviors. Moreover, GRASP has been recently adapted for

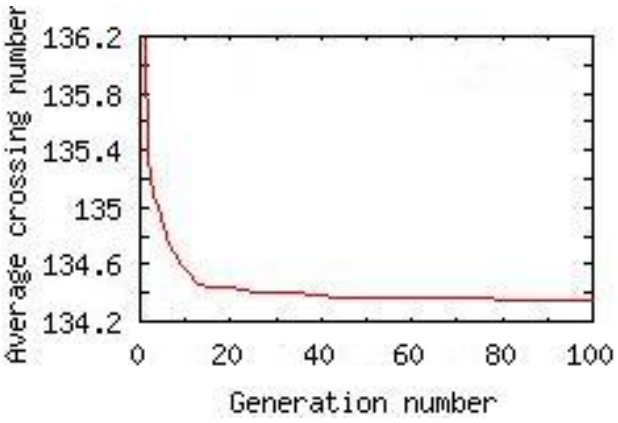


Fig. 1: Average convergence of GAs

the same drawing problem [11] and from the first published results, this approach seems more efficient than Tabu search; therefore, it will be interesting to compare it with our GA implementation. Lastly, as noticed in the previous section, we now focus on the improvement of the computation time.

References

[1] W. Buntine. Graphical models for discovering knowledge. In *Advances in Knowledge Discovery and Data Mining*, pages 59–82. 1996.

[2] G. Di-Battista, P. Eades, R. Tamassia, and I.-G. Tollis. *Graph drawing – Algorithms for the visualization of graphs*. Prentice-Hall, 1999.

[3] P. Eades and D. Kelly. Heuristics for reducing crossings in 2-layered networks. *Ars Combinatorics*, 21:89–98, 1986.

[4] P. Eades and N. Wormald. Edge crossings in drawings of bipartite graphs. *Algorithmica*, 11:379–403, 1994.

[5] I. Herman, G. Melançon, and M.S. Marshall. Graph visualization and navigation in information visualization: a survey. *IEEE Trans. on Visualization and Computer Graphics*, 6(11):24–43, 2000.

[6] M. Jünger and P. Mutzel. 2-layer straightline crossing minimization: performance of exact and heuristic algorithms. *J. of Graph Algorithms and Applications*, 1(1):1–25, 1997.

[7] H. Kargupta, K. Deb, and D. Goldberg. Ordering genetic algorithms and deception. In *Proc. Parallel Problem Solving from Nature*, volume 2, pages 47–56. Elsevier Sc., 1992.

[8] C. Kosak, J. Marks, and S. Shieber. A parallel genetic algorithm for network-diagram layout. In *Proc. of the 4^{th} Int. Conf. on Generic Algorithms*, pages 458–465. Morgan-Kaufmann, 1991.

[9] P. Kuntz, R. Lehn, and H. Briand. Dynamic rule graph drawing by genetic search. In *Proc. of the IEEE Int. Conf. on System Man and Cybernetics*, 2000.

[10] M. Laguna, R. Marti, and V. Valls. Arc crossing minimization in hierarchical design with tabu search. *Computers and Operations Res.*, 24(12):1175–1186, 1997.

[11] R. Marti. Arc crossing minimization in graphs with grasp. *IIE Trans.*, 33(10):913–919, 2001.

[12] A. Ochoa-Rodrìguez and A. Rosete-Suàrez. Automatic graph drawing by genetic search. In *Proc. of the 11^{th} Int. Conf. on CAD, CAM, Robotics and Manufactories of the Future*, pages 982–987, 1995.

[13] H. Purchase. Effective information visualisation: a study of graph drawing aesthetics and algorithms. *Interacting with computers*, 13(2):477–506, 2000.

[14] K. Sugiyama, S. Tagawa, and M. Toda. Methods for visual understanding of hierarchical systems. *IEEE Trans. Sys. Man and Cybernetics*, 11:109–125, 1981.

[15] J. Utech, J. Branke, H. Schmeck, and P. Eades. An evolutionary algorithm for drawing directed graphs. In *Proc. of the Int. Conf. on Imaging Science, Systems and Technology*, pages 154–160. CSREA Press, 1998.