

# Dynamic rule graph drawing by genetic search

Pascale Kuntz, Rémi Lehn, Henri Briand

**Abstract**— The recent importance given to the integration of the user in a KDD process, which gives him the opportunity to direct his mining towards his specific own needs, requires the development of new highly interactive visualization tools. For graph based representation of discovered knowledge this means that layout algorithms must dynamically take modifications into account. In this paper, we present a genetic approach to draw a series of layered directed graphs which model relationships between association rules. We develop new problem-specific genetic operators and show that genetic algorithms are well-adapted to solve a multiobjective problem : meeting static aesthetic requirements such as minimizing arc crosses and preserving the “user’s mental map” when a transformation is interactively performed on the graph. Experimental results are presented on several randomly generated series of graphs.

**Keywords**— Computational Cybernetics, Data Mining and Knowledge Discovery

## I. INTRODUCTION

TOGETHER with chart graphics and maps, network diagrams or graphs are one of the most popular graphical forms used for information presentation. They can be used at the same time as theoretical models and as visualization supports. They often allow access to complex abstract structures without getting bogged down in the mathematical detail [1]. For such reasons, graphs are well adapted to visually represent relationships between rules extracted from large datasets.

Since open problems on visualization set in the last decade have been identified [2], great efforts have been made in graphical database query interfaces and representations of multidimensional data, and today several tools integrate graph models (e.g. [3]; [4]). However, with the development of more and more efficient knowledge extraction algorithms and of human-centered approaches [5] new needs are coming to light in the mining and post-treatment phases of a KDD process [6].

We here restrict ourselves to non-supervised approaches of rule extraction for which different graph models have been proposed in the literature. Let  $O$  be the set of objects of a database described by a finite set  $I$  of items. For purely logical rules, i.e. without any counter-examples, Galois lattices [7] have known a renewed interest in combinatorial data analysis for the search of implications in binary data [8]. In this case, each node is a particular pair composed of a subset of  $O$  and a subset of  $I$  and the set of pairs is partially ordered by the standard set inclusion relation applied to  $O$  and  $I$ . It can be represented by a Hasse diagram where arcs represent the inclusion relation. Unfortunately, determining a Galois lattice along with its Hasse diagram is a computationally difficult problem, and the visual representation become inextricable for numerous items. An ex-

tension of this model to the search of “quasi-implications” where the strict inclusion is replaced by a statistical measure has been recently proposed [9] but experiments only deal with small size data sets at the moment. In other common rule representation models, the graph’s nodes are just itemsets which correspond to premises and conclusions of the rules, and an arc directly represents an implication relation whose validity is assessed by a quality measure [10]. More sophisticated approaches are inspired by hierarchical classification algorithms and represent implication relationships by hierarchical trees [11].

Whatever the selected model is, in the vast majority of the applications, a layout of the rule graph is a kind of final synthesis obtained after a complex data processing line. It is considered as an appropriate visualisation support to give an insight into results that would be more difficult to get from looking at long lists of alpha-numerical characters. From the algorithmical point of view for the layout problem, the vertex and arc sets are given as input in accordance with the model, and a drawing satisfying some intelligibility criteria is furnished as output. This is referred to as a static drawing problem. However, the recent importance given to the integration of the user in a KDD process, which gives him the opportunity to direct his mining towards his own needs, requires the development of new highly interactive visualization tools. For graphs, this means that layout algorithms must take into account modifications (e.g. insertions or deletions of nodes and arcs) dynamically at different time scales.

Generally speaking, numerous efficient graph drawing algorithms have been proposed for many criteria [12] but, few of them take the interactivity with the user into account [13]. If a modification is performed, the algorithm runs again and produces a new drawing which may be thoroughly different from the previous one. As noticed by Papakostas et al [14] “this is a waste of human resources to continually re-analyse the entire drawing and also of computational resources to re-computed the entire layout after each modification”. And, in the context of data mining, these limitations may become a real obstacle: important changes between two consecutive layouts in a dynamical rule extraction process, where nodes and arcs can be added depending on user requests, highly disturb the interpretation task. This is the reason why such a process must be associated with a drawing method which preserves the “user’s mental map” as much as possible. More formally, this problem can be set as a multi-objective problem: produce at each step  $t$  a layout  $L(G_t)$  of the graph  $G_t$  that satisfies common readability requirements, as for instance arc crossings minimization, and so that  $L(G_t)$  remains “similar” to the layout  $L(G_{t-1})$  of the graph  $G_{t-1}$  proposed at the previous step.

IRIN, Université de Nantes, Rue Christian Pauc, 44300 Nantes, France, Remi.Lehn@irin.univ-nantes.fr

In this paper, we develop a genetic algorithm (GA) based approach for dynamically drawing a directed graph model of association rules, simpler than a Galois lattice. GAs are stochastic global search methods that have proven to be efficient for many combinatorial optimization problems. In particular, several authors have reported promising applications of GAs or Evolutionary Algorithms for different static drawing problems of directed graphs ([15], [16], [17], [18], [19]). Here, we develop new problem-specific genetic operators for the layered drawing of our rule graph model and show that GAs are particularly well-adapted for the dynamic layout problem. Due to their intrinsic parallelism they compute a set of potentially good solutions and indirectly allow to solve our multi-objective problem: the set of potential solutions is generated with GA operators according to fitness function which measure aesthetics, and a layout is chosen among them according to its resemblance to a previous layout.

## II. MODELING OF THE PROBLEM

Each subset  $X_i$  -often called itemset- of  $I$  is associated with a unique subset  $\xi_i$  of  $O$ :  $\xi_i$  is the set of objects for which every item of  $X_i$  is present.

For instance,  $X_i = \{i_1, i_4, i_5\}$  associated with the subset  $\xi_i = \{o_{12}, o_{42}, o_{791}, o_{3244}\}$  means that  $o_{12}$ ,  $o_{42}$ ,  $o_{791}$  and  $o_{3244}$  are the only objects of  $O$  for which  $i_1$ ,  $i_4$  and  $i_5$  are all present. Although our model may be generalized to more complex data, we here restrict ourselves to binary items.

### A. The rule graph model

Here, a rule graph is an acyclic directional graph  $G = (V, A)$  where each vertex of  $V$  is an potentially interesting itemset, and each arc of  $A$  represents a significant implication between two itemsets. Formally, a vertex is a subset of  $I$  and there exists an arc between two vertices  $X$  and  $X'$  if the rule  $X \rightarrow X' \setminus X$  is valid according to statistical measures. For instance, if  $X = \{i_4, i_7\}$  and  $X' = \{i_4, i_7, i_9\}$ , an arc between  $X$  and  $X'$  corresponds to the association rule  $[i_4 \wedge i_7] \rightarrow i_9$ .

We have developed a user-driven algorithm for the association rule extraction where a series of graph  $(G_t)_{t=0,T}$  is generated dynamically by the requests of the user. Roughly speaking, at each step  $t$ , the user selects a vertex associated with an itemset he is focusing on, and the interesting associated rules are automatically generated by a modified local version of the well-known A Priori algorithm [20] (the precise description of the process and its cognitive foundations are far beyond the scope of this paper and we refer to [21] for details). Then, a new graph  $G_{t+1}$  is drawn to update the knowledge. The process starts with a discrete graph  $G_0$  which only contains vertices representing itemsets of cardinality one often present in the database -in other words, items common to numerous objects-, and stops at a step  $T$  fixed by the user with a graph  $G_T$  which contains all the rules discovered during the process.

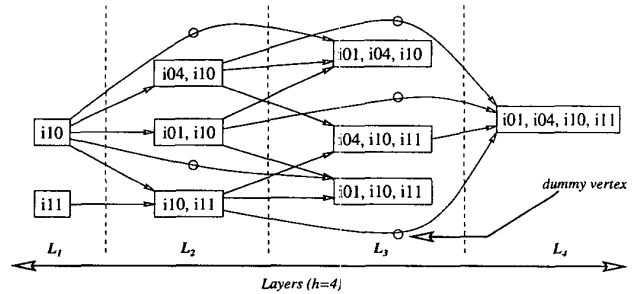


Fig. 1. Layered drawing of the rule graph

### B. Intelligibility requirements for the graph layout

A property always required for drawing graphs which model abstract concepts is intelligibility. This covers physical constraints of the drawing support and aesthetic criteria. Aesthetics are closely linked to the application; they specify graphic properties that should help the reader to understand and remember the information represented in the graph.

In our case, polyline drawings with vertices arranged in vertical layers are well adapted. The vertex set  $V$  is partitioned into  $h$  subsets associated with  $h$  layers  $L_1, \dots, L_h$  so that for any arc  $(X, X') \in A$  where  $X$  is placed on  $L_i$  and  $X'$  on  $L_j$  then  $i > j$ . In the model, each layer is associated with a degree of precision in the knowledge state: layers on the left correspond to general descriptions made by small itemsets, whereas layers on the right correspond to more specific descriptions (Fig. 1). Thus, vertices are here supposed to be pre-assigned to the layers, and graphs are supposed to be acyclic.

In addition to this drawing convention, readability criteria, often defined by combinatorial optimization goals, must be specified. One of the most sensitive criterion for the interpretation is the number of arc crossings [22]. Note that the general problem of minimizing arc crossings is NP-complete and remains so even for the layered digraph drawing where the crossing number depends on the ordering of the vertices within each layer only [23].

In order to avoid long lines which can create confusion we also consider the minimization of the sum of the arc lengths. These two constraints are here treated separately: a GA generates a vertex ordering on each layer which tends to minimize arc crossings and a classical hill-climbing locally adjusts the vertex coordinates to minimize arc lengths.

### C. Dynamic integration

One major difficulty is to propose a pragmatic definition of the intuitive concept of “mental map’s stability” when interpreting a layout. In the general case, let us denote by  $P$  the set of points of a geometric space  $S$  which represents the vertices of a graph  $G$  on a layout  $L(G)$ , and let  $T$  be a function which transforms  $P$  into another representation  $P'$  of the same graph. Eades et al., [24] propose a definition of the stability which depends on the choice of an equivalence relation  $\equiv$  on the set of the finite  $S$  sub-

sets: the transformation  $T$  preserves the representation  $P$  if  $P \equiv P' = T(P)$ . They give different pragmatic definitions of  $\equiv$  one of which is based on order equivalences:  $P$  and  $P'$  have the same “orthogonal ordering” if the relative positions of each pair of points are the same in given directions (e.g. up, down, left, right).

In this paper we adapt this notion to the comparison of the layouts of different graphs  $G_t$  and  $G_{t+1}$  obtained at consecutive steps. We only take into account the number of vertex permutations between two layouts as this criterion is the most sensitive for the interpretation. Let  $\bar{c}_k$  (resp.  $c_k$ ) be the number of vertex couples on layer  $k$  whose order has changed (resp. not changed) between  $L(G_t)$  and  $L(G_{t+1})$ . The two layouts are compared via a dissimilarity coefficient  $\Delta$ :  $\Delta(L(G_t), L(G_{t+1})) = 1 - \frac{1}{h} \sum_{k=1, h} \frac{\bar{c}_k}{c_k}$ . And,  $P_t \equiv P_{t+1}$  if  $\Delta(L(G_t), L(G_{t+1})) = 0$ .

### III. A GENETIC BASED APPROACH

GAs work with a population of potential solutions which stochastically evolves by means of three basic operators: selection, recombination and mutation. And, when designing a GA, the proper choice of the solution coding and the genetic operators is of the utmost importance as it can have a significant influence on the optimization process. In the following, we therefore discuss coding and associated operators in more details.

#### A. Solution coding and selection

To simplify the drawing problem, a classical representation is previously made so that the graph is “proper” i.e. each arc is incident to vertices placed on two adjacent layers only. This is easily achieved by inserting so-called dummy vertices: an arc incident to vertices  $X$  and  $X'$ , respectively placed on layers  $k$  and  $k'$  so that  $|k - k'| > 1$  is transformed into a path of  $|k - k'|$  arcs using  $|k - k'| - 1$  dummy vertices (circles on figure 1). Moreover, a set of possible positions numbered downwards is associated with each layer. Hence, the problem consists in defining a permutation  $\sigma_k$  on each  $L_k$ , where  $\sigma_k(X)$  is the rank of  $X$  on  $L_k$ , so that the total arc crossing number is minimized.

Although for most graph drawing applications, the potential solutions are coded by a string of the real coordinates of the vertex positions, an ordinal representation is more appropriate here. A genotype codes the vertex ranks for each layer one after the other: the ranks of the  $n_1$  vertices of  $L_1$  are placed on the left in the genotype, then the ranks of the  $n_2$  vertices of  $L_2$  are added to it and so on.

In the classical GA scheme (e.g. [25]), the initial population is randomly chosen, and every subsequent generation is built from the current one with the genetic operators. An alternative consists in initializing the process by a small population which increases by insertion of new solutions created by a reproduction mechanism and whose evolution is controlled by an “aging” threshold: only solutions recent enough can be selected for reproduction. Experimental comparisons have led us to prefer this alternative and to select an initial population with genotypes whose vertex ordering fits different depth first searches 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, ...). This tends to initialize the exploration with potentially suitable solutions.

The selection is determined by a classical roulette wheel based on the fitness function. In order to compare the crossing numbers of different layouts without knowledge of a maximal bound, the fitness is here defined by  $2^{-f(L(G))}$  where  $f(L(G))$  is the number of arc crossings in the layout  $L(G)$ .

#### B. Mutation

Four mutation operators are applied according to a probability, and to ensure that the genotype codes a valid solution, each mutation operator is local and can modify genes of a unique layer -randomly chosen- at each time. Two of them are common local search operators in permutation problems: the random local permutation and the 2-opt. The two other are problem-specific and are inspired by drawing heuristics [26].

**Permutation based on vertex degrees.** As a permutation with few neighbors disturbs the drawing usually less than a permutation with highly connected vertices, local permutations for vertices with small degrees are favored. The choice of two vertices to be exchanged depends on a probability inversely proportional to their degrees.

**Permutation based on median.** This permutation stemmed from one of the most common heuristic applied to the 2-layer drawing problem which aims at placing a vertex on one layer  $L_{i,i=1,2}$  in front of the “middle” (which can be a median<sup>1</sup> or a barycenter) of its neighbors on  $L_{j,j \neq i}$ . This attempts to minimize arc length locally and contributes to a better drawing. The transposition of this idea to the permutation scheme consists in selecting a vertex  $X$  randomly on a layer  $L_k$  and exchanging its rank with the rank of the vertex of  $L_k$  which is the closest to the middle of the median of the successors of  $X$  on  $L_{k+1}$  and of the median of its predecessors on  $L_{k-1}$ . More formally, the classical definition of the median must be adapted here. The median of a vertex  $X$  on  $L_k$  is the vertex of  $L_k$  whose rank in the order defined by  $\sigma_k$  is the closest to

$$\frac{1}{2} \left( + \frac{\frac{n_k}{n_{k-1}} \sum_{Y \text{ predecessor of } X} \sigma_{k-1}(Y)}{\deg^-(X)} \right) \quad (1)$$

where  $\deg^-(X)$  (resp.  $\deg^+(X)$ ) is the number of  $X$  predecessors (resp. successors) and  $n_k$  is the vertex number on  $L_k$ . For instance let us consider a vertex  $X$  at rank 1 on  $L_k$  with 5 predecessors on  $L_{k-1}$  at respective ranks 1, 3, 5, 6, 8 and 3 successors on  $L_{k+1}$  at respective ranks 3, 5 and 6 (Fig. 2). If  $n_{k-1} = 8$ ,  $n_k = 4$  and  $n_{k+1} = 6$  then formula (1) equals 2.7 and consequently the vertex  $X$  is swapped with the vertex at rank 3 on  $L_k$ .

<sup>1</sup>For real coordinates on each vertical layer, the median of the predecessors of a vertex is the  $y$ -coordinate s.t. half of the  $y$ -coordinates of the predecessor are greater and the other half are smaller.

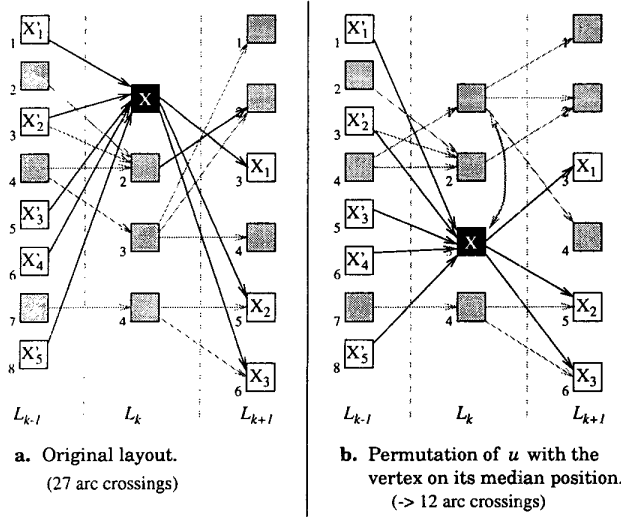


Fig. 2. Permutation based on median

Generally speaking, the impact of each of these mutation operators depends on the graph properties. In case of few dummy vertices, the local permutation combined with 2-opt gives good results whereas in the opposite case -i.e. when arcs are greater and vertex degrees are heterogeneous- more specific operators noticeably improve the results. As for dynamic drawing these properties are not known *a priori* and may evolve during the mining process, we here keep all of the operators with a fixed activating probability at each step (0.02 for local permutation, 0.01 for 2-opt, 0.04 for permutation based on vertex degrees and 0.08 for permutation based on median).

### C. Crossover

It is well-known that the main difficulty for ordinal codings is to define a crossover which guarantees a feasible solution. Here, two genotypes are selected and the crossover is applied on one layer  $L_k$  randomly chosen. We have compared two specialized crossover operators which have been previously applied to problems encoded as permutation -referred to as "ordering GA" - (we refer to [27] for a description): Order Crossover 1 (OC1) and Partially Mapped Crossover (PMX). And to avoid side effects as for instance the recombination of two identical genotypes which leads to a new different one, we have included the variants proposed in [28]. Figure 3 shows the evolution of the fitness with each operator for the same computational time. The time required by the PMX compared to OC1 acts as a brake upon the search space exploration here. The number of generations obtained with OC1 is about three time superior to PMX and profits of PMX in the optimization process are quickly overtaken. Let us remark that PMX has been developed for Traveling Salesman Problems of large sizes in particular whereas, in our case, the complexity lies in the relationships between layers, and each layer is limited to about 70 vertices for readability requirements.

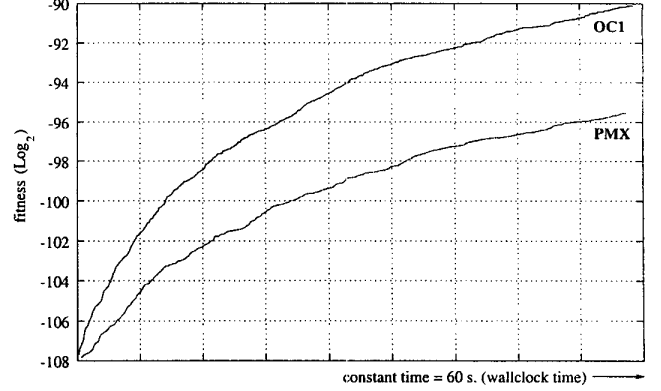


Fig. 3. Comparison with OC1 and PMX crossover operators (evolution of average fitness for 100 drawings of a random 50 edges/40 vertices/10 layers digraph).

Due to these results we here retain the following definition of OC1. An interval  $I$  of random size is selected on the layer  $L_k$  of the first genotype and  $I$  is duplicated in the offspring's genotype at the same place. This one is completed by genes of the second genotype which are not in  $I$  by starting at the end of  $I$  and by following the second genotype's order. For instance, let us suppose that the first (resp. the second) genotype contains the sequence  $i_4 - i_1 - i_2 - i_5 - i_3 - i_6$  (resp.  $i_1 - i_2 - i_4 - i_6 - i_5 - i_3$ ) for the layer  $L_k$ . If  $I = i_2 - i_5 - i_3$  then the layer  $L_k$  of the offspring contains  $i_4 - i_6 - i_2 - i_5 - i_3 - i_1$ . The other layers are either the same as the first genotype or the same as the second genotype according to a random choice.

### D. Dynamic constraint

The stability constraint between two consecutive layouts is not introduced right from the start but from a certain size  $n_s$  of the graph. Up to  $n_s$  (fixed here at 20 vertices) only fitness  $f$  is taken into account in the optimization process. Indeed, when graphs are small, the user can record the whole of the information and cope with changes easily. Moreover, at the beginning of the mining process, there are few extracted rules and the "shapes" of the associated graphs are not well defined.

Let  $t$  be the step of the drawing  $L(G_t)$  of the graph with  $n_s$  vertices. At  $t + 1$  new vertices are added. The GA described above runs during a fixed number of generations in order to maximize  $f$  and we denote by  $P_{t+1}^*$  the last population obtained. The dissimilarity  $\Delta$  is computed between each layout of  $P_{t+1}^*$  and  $L(G_t)$ , and among the  $k$  (fixed here at 20) closest layouts, the one with the largest fitness is selected.

In the coding, when a vertex is added, a new place is allocated on each genotype and a rank is added for the new vertex. For arc addition the only change is concerned with the fitness whose value is calculated again.

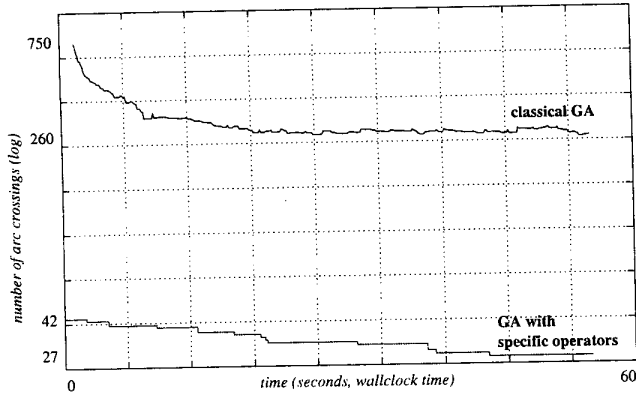


Fig. 4. Comparison with a graph drawing algorithm based on classical GA operators (average values of 20 drawings of a random 45 edges/31 vertices/6 layers digraph).

#### IV. EXPERIMENTAL RESULTS

In order to prove the interest of our new problem-specific GA operators, we first compare results obtained on “static” graphs to those obtained with basic GA operators applied to a real coding of vertex positions on each layer [29]. Figure 4 shows the fitness values obtained for the same computation time (60 seconds wallclock time) on 20 drawings of a random graph.

Figure 6 illustrates the integration of the dynamic constraint. Two consecutive layouts  $L(G_{t-1})$  and  $L(G_t)$  are presented: one vertex and six arcs are added on  $G_t$ . And figure 7 describes a typical situation of the multi-objective optimization. At each step  $t$ , vertex and arcs are randomly added to the previous graph  $G_{t-1}$ . The initial graph  $G_0$  is reduced to a single random arc. At each  $t$ , the GA tends to maintain a layout  $L(G_t)$  very close to the previous one  $L(G_{t-1})$  as long as possible. But, at a certain step  $t^*$  too many changes in the graph have been performed and the layouts close to the previous one  $L(G_{t-1})$  become unacceptable according to the fitness; they disappear and leave their places with other potential solutions. The evolution of the dissimilarities  $\Delta(L(G_{t-1}), L(G_t))$  between selected layouts highlights “peaks” and “holes” which are explained by this double optimization process. Let us notice that this phenomenon is closely linked to the parameter values which govern the evolution; in particular, it softens when the size of the population decreases.

When we compare the evolutions of the fitness of the selected layout  $L^*(G_t)$  (i.e. the layout which minimizes  $\Delta(L^*(G_{t-1}), L^*(G_t))$ ) and of the best layout  $L_f(G_t)$  according to the fitness (i.e. the layout which maximizes  $f$ ), we see that both curves follow a similar evolution with the  $L_f(G_t)$  fitness obviously better than -or equal to- the  $L^*(G_t)$  one.

#### V. CONCLUSION

In this paper we presented a graph model well-adapted to interactive association rule visualization and we developed a genetic algorithm for the dynamical drawing of

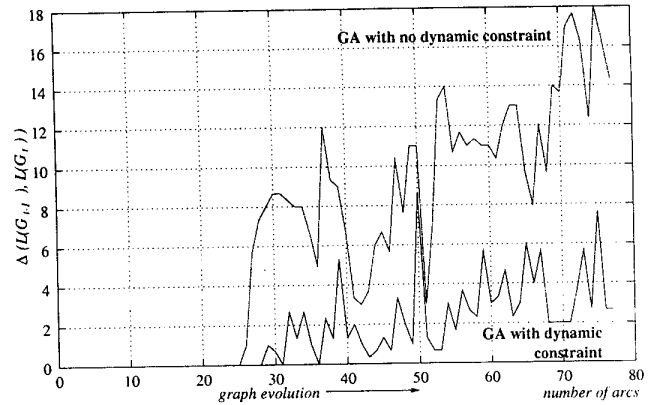


Fig. 5. Comparison of the fitness values at each step for the selected layouts and the best ones according to the fitness (average value on 10 runs on the same serie of graphs).

layered directed graphs. We showed that ordering coding and problem-specific operators inspired by classical drawing heuristics and permutation problems significantly improve previous results obtained with a classical scheme. Numerical tests allow us to think that GAs are very strong candidate to solve this class of drawing problems. Our algorithm is now included in a rule mining software [30] and first experiments on real databases from marketing and human resources highlight the information gain obtained in the extraction process with such a visualization support.

Future work includes further evaluations that need to determine which dissimilarities are most appropriate for modeling the user mental map’s stability. Additional experiments are also necessary to better understand the impact of constraint’s stability on the search space. In particular, we are applying an approach coming for multidimensional scaling to discover resemblance relationships between layouts within a same population.

#### REFERENCES

- [1] W. Buntine, “Graphical models for discovering knowledge,” In Fayyad et al. [31], pp. 59–82.
- [2] L. Treinish, D.M. Butler D.M., H. Senay, G.G. Grinstein, and S.T. Bryson, “Grand challenge in visualization software,” in *Proc. Visualization*. 1992, pp. 366–371, Boston, Mass.
- [3] M.P. Consens and A.O. Mendelzon, “Hy+: a hypergraph-based query and visualization system,” in *Proc. ACM SIGMOD Int. Conf. on Management of Data*, 1993, pp. 511–516.
- [4] R.A. Becker, S.G. Eick, and G.J. Wills, “Visualizing network data,” *IEEE Trans. on Visualizations and Graphics*, vol. 1, no. 1, pp. 16–28, 1995.
- [5] J.R. Brachman and T. Anand, “The process of knowledge discovery in databases: a human-centered approach,” In Fayyad et al. [31], pp. 37–58.
- [6] W.J. Frawley, G. Piatetsky-Shapiro, and C.J. Matheus, “Knowledge discovery in databases: an overview,” in *Knowledge Discovery in Databases*, pp. 1–27. AAAI Press, 1991.
- [7] G. Birkoff, *Lattice Theory*, A.M.S., 1967.
- [8] V. Duquenne, “Latticial structures in data analysis,” *Theoretical Computer Science*, vol. 217, no. 2, pp. 407–436, 1999.
- [9] J. M. Bernard and S. Poitrenaud, “Multivariate Bayesian implicative analysis for a binary questionnaire : quasi-implications and simplified Galois lattice,” 1999.
- [10] M. Klemettinen, H. Mannila, H. Toivonen P. Ronkainen, and I. Verkamo, “Finding interesting rules from large sets of discovered association rules,” in *Proc. of the 3<sup>rd</sup> Int. Conf. on*

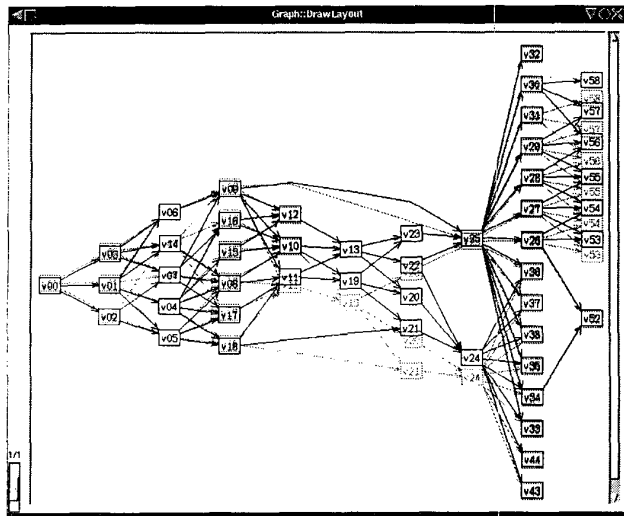


Fig. 6. Graph updating with dynamic constraint – on this figure, 1 vertex and 6 arcs have been added (vertex  $v_{13}$  and its incident arcs);  $L(G_t)$  is darker and  $L(G_{t-1})$  is lighter.  $\Delta(L(G_{t-1}), L(G_t)) = 2$

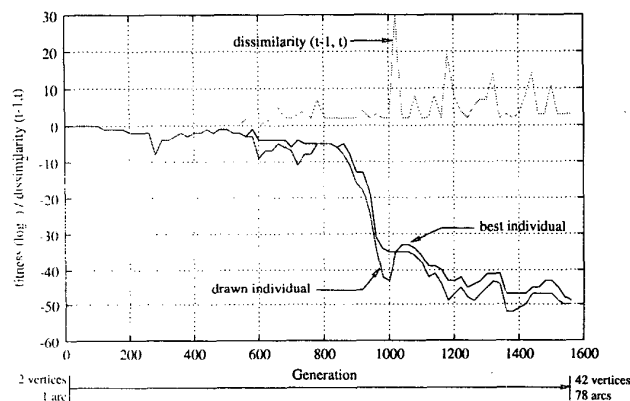


Fig. 7. Typical situation of the multi-objective optimization for dynamical layouts

- Information and Knowledge Management. 1994, pp. 401–407, ACM Press.
- [11] R. Gras, H. Briand, and P. Peter, "Structuration sets with implication intensity," in *Proc. of the Int. Conf. on Ordinal and Symbolic Data Analysis*, 1996, pp. 147–156.
  - [12] G. Di Battista, P. Eades, R. Tamassia, and I.-G. Tollis, *Graph drawing - Algorithms for the visualization of graphs*, Prentice-Hall, 1999.
  - [13] R.-F. Cohen, G. Di Battista, R. Tamassia, I.-G. Tollis, and P. Bertolazzi, "A framework for dynamic graph drawing," in *Proc. of ACM Symposium on Computational Geometry*. 1992, pp. 261–270, ACM Press.
  - [14] A. Papakostas, J.-M. Six, and I.-G. Tollis, "Experimental and theoretical results in interactive orthogonal graph drawing," in *Proc. Graph Drawing'96*. 1997, vol. 1190 of *Lect. Notes in Comp. Sc.*, pp. 371–386, Springer Verlag.
  - [15] L.J. Groves, Z. Michalewicz, P.V. Elia, and C.Z. Janikow, "Genetic algorithms for drawing directed graphs," in *Proc. of the 5th Int. Symp. on Methodologies for Intelligent Systems*. 1990, pp. 268–276, Elsevier.
  - [16] T. Masui, "Graphic object layout with interactive genetic algorithms," in *Proc. of the 1992 IEEE Work. on Visual Languages*. 1992, pp. 74–80, IEEE Computer Soc. Press.
  - [17] E. Makinen and M. Sieranta, "Genetic algorithms for drawing bipartite graphs," *Int. J. of Comput. Math.*, vol. 53, no. 3-4, pp. 157–166, 1994.
  - [18] A. Ochoa and A. Rosete, "Automatic graph drawing by genetic search," in *Proc. of the 11th Int. Conf. on CAD, CAM, Robotics and Manufactories of the Future*, 1995, pp. 982–987.
  - [19] 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*. 1998, pp. 154–160, CSREA Press.
  - [20] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A.-I. Verkamo, "Fast discovery of association rules," in *Fayyad et al.* [31], pp. 307–328.
  - [21] P. Kuntz, F. Guillet, R. Lehn, and H. Briand, "A user-driven process for mining association rules," in *to appear in Proc. of Principles and Practice of Knowledge Discovery in Databases*. 2000, Springer-Verlag.
  - [22] H. Purchase, "Which aesthetic has the greatest effect on human understanding ?," in *Proc. Graph Drawing'97 - Lect. Notes in Comp. Sc.* 1997, vol. 1353, pp. 248–261, Springer Verlag.
  - [23] P. Eades and N. Wormald, "Edge crossings in drawings of bipartite graphs," *Algorithmica*, vol. 11, pp. 379–403, 1994.
  - [24] P. Eades, W. Lai, K. Misue, and K. Sugiyama, "Preserving the mental map of a diagram," in *Proc. of Compugraphics*, 1991, pp. 24–33.
  - [25] D.-E. Goldberg, *Genetic Algorithms in Search Optimization and Machine Learning*, Addison-Wesley, 1989.
  - [26] E.R. Gansner, E. Koutsofios, S.C. North, and K.P. Vo, "A technique for drawing directed graphs," *IEEE Trans. Soft. Eng.*, vol. 19, pp. 214–230, 1993.
  - [27] H. Kargupta, K. Deb, and D. Goldberg, "Ordering genetic algorithms and deception," in *Proc. Parallel Problem Solving from Nature*. 1992, vol. 2, pp. 47–56, Elsevier Sc.
  - [28] D. Whitley and N. Yoo, "Modeling simple genetic algorithms for permutation problems," in *Foundations of Genetic Algorithms III*. Morgan Kaufmann, 1995.
  - [29] F. Guillet, P. Kuntz, and R. Lehn, "A genetic algorithm for visualizing networks of association rules," in *Proc. of the 12th Int. Conf. on Industrial and Engineering App. AI and Expert Sys.* 1999, pp. 145–154, Lect. Notes in Comp. Sc., Springer-Verlag.
  - [30] R. Lehn, F. Guillet, and P. Kuntz, "Felix: an interactive rule mining interface in a kdd process," in *Proc. of the 10th Mini Euro Conference: Human Centered Processes*, ENST Bretagne, France, 1999, pp. 169–174.
  - [31] U.M. Fayyad, G. Piatetsky-Sapiro, and P. Smyth, Eds., *Advances in Knowledge Discovery and Data Mining*, AAAI Press, 1996.