View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by El Servicio de Difusión de la Creación Intelectual

An automatic graph layout procedure to visualize correlated data

Mario Inostroza-Ponta, Regina Berretta,
Alexandre Mendes, and Pablo Moscato

Newcastle Bioinformatics Initiative
School of Electrical Engineering and Computer Science
Faculty of Engineering and Built Environment
The University of Newcastle, Callaghan, NSW, 2308, Australia

and
ARC Centre in Bioinformatics
Contact email:Pablo.Moscato@newcastle.edu.au

Abstract. This paper introduces an automatic procedure to assist on the interpretation of a large dataset when a similarity metric is available. We propose a visualization approach based on a graph layout methodology that uses a Quadratic Assignment Problem (QAP) formulation. The methodology is presented using as testbed a time series dataset of the Standard & Poor's 100, one the leading stock market indicators in the United States. A weighted graph is created with the stocks represented by the nodes and the edges' weights are related to the correlation between the stocks' time series. A heuristic for clustering is then proposed; it is based on the graph partition into disconnected subgraphs allowing the identification of clusters of highly-correlated stocks. The final layout corresponds well with the perceived market notion of the different *industrial sectors*. We compare the output of this procedure with a traditional dendogram approach of hierarchical clustering.

1 Introduction

The Standard & Poor's 100 index is one the leading stock market indicators in the United States. It measures the performance of the 100 largest U.S. companies, corresponding to over US$ 6 trillion in terms of market capitalization[1] and it is composed of stocks from different sectors.In the stock market, the changes of the value of a given company are highly correlated with the time series of its stock price. Two contributions to the study of market dynamics ([1];[2]) reported on the application of *Self-Organizing Maps* and *Chaotic Map Synchronization* to two different datasets composed of the price variation of the stocks in the Dow Jones index. A graph-based approach using 6,546 *financial instruments* (stocks, indexes, etc.) traded in the US markets has also been recently introduced [3].

[1] http://www2.standardandpoors.com/spf/pdf/index/factsheet_sp100.pdf

In this paper we propose a new graph layout visualization method and use it to uncover interesting relationships between the stocks of the S&P100 index used as a case study. We will consider that each stock corresponds to a node of a graph; the edges' weights will be related to the correlation between stocks. The method recursively divides the graph in disconnected subgraphs. Once the subgraphs (clusters) are defined, we solve a sequence of Quadratic Assignment Problems (QAP) using a memetic algorithm (MA) which will determine their relative position in the layout. Finally, another instance of the QAP is solved to find how the clusters are distributed, now considering each cluster as a single element.

The Quadratic Assignment Problem (QAP) belongs to the *NP-hard* [4] class and is a well-studied combinatorial optimization problem [5, 6, 7]. Informally, we are given a set of n facilities and m locations ($m \geq n$), and the task is to assign each facility to a location taking into account the flow between each facility and the distance between the locations. The objective is to minimize the overall transportation cost between all the facilities. For our case study, we will use the correlation between stocks to determine the flow between facilities. The locations will be points in a grid, with the distances between them given by the Euclidean metric. We have as input a flow matrix between the stocks, and from this matrix we create a weighted graph. We can understand this graph as a proximity graph; its edges will also have a strong influence in the layout process as will be described later. The result is a graph layout where clusters of stocks with similar dynamical behavior are promptly identified, and no user-intervention is required during the process.

The use of MAs to address the QAP can be dated back to Carrizo et al.(1992) [8] and Merz and Freisleben (1999) [9]. In this paper, we employ a similar MA to those successfully used before for other combinatorial optimization problems, including Number Partitioning [10] and the Asymmetric Travelling Salesman [11] problems among others. Two local search methods are used; one of them has an embedded Tabu Search [12].

This paper is organized as follows. In Sec. 2 we describe the graph layout procedure. Section 3 describes the memetic algorithm for the QAP. The result of applying this method on the S&P100 dataset is presented in Sec. 4, followed by the conclusions in Sec. 5.

2 Graph Layout Procedure

The graph layout procedure proposed in this paper is composed of 3 steps: *creation of a distance matrix*, *proximity graph clustering algorithm* and *creation of QAP instances that will be solved using the MA*. We explain each step using the S&P100 dataset.

2.1 Distance Matrix

To create the distance matrix D of the S&P100 dataset, we took the second derivative of the weekly closing price variation of the stocks that compose the index, between the years 1999 and 2004. The work of Ausloos and Ivanova (2002) [13] advocates the use of the second derivative (which represents the acceleration of the stock price), arguing in their studies of "*pressure, acceleration and force indicators*" that it contains more information than the first derivative. The expression of the three-point rule for the second derivative of the stock price at time t is given by

$$y_i(t) = \frac{P_i(t-h) - 2.P_i(t) + P_i(t+h)}{h^2}, \tag{1}$$

where $P_i(t)$ represents the closing price of the stock i in the week t and h represents the interval used to calculate the derivative; in this case, $h = 1$ week. At the end, we normalize the result by dividing it by $P_i(t-h)$, so to eliminate any bias introduced by the actual price of the stock. The distance matrix $D = \{d_{ij}\}$ is defined as $d_{ij} = 1 - \rho_{ij}$, where ρ_{ij} is the Pearson correlation between stocks i and j using the values calculated with function 1. The two most correlated stocks ($\rho = 0.802$) are Schlumberger Ltd. and Baker Hughes Inc., while the two most anti-correlated ($\rho = -0.38$) are Alcoa Inc. and Anheuser-Busch Co. There are only 459 pairs of stocks with $\rho < 0$.

2.2 Proximity graph clustering algorithm

We use the matrix D to build our *ad-hoc* proximity graph using the *minimum spanning tree* and the *k-nearest neighbors* graphs, which we will refer to as G_{MST} and G_{kNN} respectively, as follows: Initially, we create a complete undirected weighted graph $G(V, E, w)$, using the matrix D, where the weight $w_{ij} = d_{ij}$. The minimum spanning tree $G_{MST}(V, E_{MST})$ is defined as a connected, acyclic subgraph containing all the nodes of G and whose edges sum has minimum total weight. The graph G_{kNN} is represented by $G_{kNN}(V, E_{kNN})$, where $e_{ij} \in E_{kNN}$ iff j is one of the k nearest neighbors of i. Our proximity graph, namely $G_{cluster}(V, E_{cluster})$, is constructed such that $E_{cluster} = E_{MST} \cap E_{kNN}$. This type of proximity graphs was used also in González-Barrios and Quiroz (2003) [14]. In this work we decided to set k as the minimal value such that G_{kNN} is still connected while in Ref. [14] they have a different approach.

2.3 Creating and solving QAP instances

We consider the QAP with n elements and $m > n$ positions. The QAP has as input a matrix $F = \{f_{ij}\}$ of flows between the n elements and a matrix $L = \{l_{ij}\}$ of distances between m grid locations. The objective is to assign the n elements to the m locations such that the function $Cost(S) = \sum_{i=1}^{n} \sum_{j=1}^{n} f_{ij} l_{S(i)S(j)}$ is

minimized, where the notation $S(i)$ represents the assigned location of element i in solution S. The flow matrix F is created using distance matrix D according to:

$$f_{ij} = \begin{cases} \frac{1000}{d_{ij}} & \text{if } e_{ij} \in E_{cluster}; \\ \frac{1}{d_{ij}} & \text{otherwise.} \end{cases} \quad (2)$$

Clearly, higher (respectively lower) flows will correspond to elements that are similar (respectively dissimilar). A good solution for the QAP will thus put the elements with a high flow closer in the layout, which is exactly our goal. Additionally, two elements with an edge in $G_{cluster}$ have their flow multiplied by a factor of 1,000, thus enforcing their proximity in the final layout. The matrix L is generated from the distances of points in a square grid of $m = g^2$ positions, with $m \gg n$. In this work, we set $\lceil g = 2\sqrt{n} \rceil$ and l_{qp} is the Euclidean distance between each locations p and q for all $1 \leq q, p \leq m$).

Assume that the graph $G_{cluster}$ contains c disconnected subgraphs ($G^1_{cluster}$, $G^2_{cluster}, ..., G^c_{cluster}$). Then each subgraph $G^i_{cluster}$ becomes a QAP instance and is solved separately. Finally, we solve one last QAP, where each element is a subgraph $G^i_{cluster}$. The instance for this problem is created by building a fully connected graph $G_C(V_C, E_C, w_C)$ where $|V_C| = c$ and the weight $w_{C_{ij}}$ corresponds to the flow between subgraphs $G^i_{cluster}$ and $G^j_{cluster}$, calculated as:

$$w_{C_{ij}} = \frac{\sum_{p \in G_i} \sum_{q \in G_j} f_{pq}}{|V^i_{cluster}| * |V^j_{cluster}|}. \quad (3)$$

In the next section, we will describe the main characteristics of the memetic algorithm used to tackle the QAP problem.

3 Memetic Algorithm

Memetic Algorithms (MAs) is a name that designates a class of powerful population-based metaheuristics with many successful practical applications ([15, 16, 17]). In our MA implementation (see the pseudo-code in Figure 1), we have a population of *agents* composed of two solutions (namely *pocket* and *current*). The idea behind this is that while the *current* solutions are constantly being modified by recombination and mutation, the *pockets* maintain a memory of the best solutions found. The population is organized with a hierarchical ternary tree structure, divided in four overlapped subpopulations of four agents each (one *leader* and three *supporters*). The *supporters* of the first subpopulation are the *leaders* of the others. This population structure has been used before [10] and in Ref. [11] this was the best structure in a comprehensive test of alternative topologies. The method **updatePop**() is responsible for making the best solutions climb the tree towards the upper agents. The method initially verifies the *pocket* solution of each agent, checking whether it is worse than the *current* one. Whenever that happens, the *pocket* is replaced by the

```
memeticAlgorithm()
    pop = initializePop(); updatePop(pop)
    repeat
        for i=0 to 12
            offspring =recombination(selectSol(parentA,parentB))
            localSearchTS(offspring)
        updatePop(pop); 8-neighborLS(agent^0_pocket)
    until max_number_of_generations
```

Fig. 1. Pseudo-code of the memetic algorithm implemented for the QAP.

current. Then, for each subpopulation, the method replaces the leader's *pocket* solution with the best supporter's *pocket* whenever the latter has better cost. A solution is represented as an integer array S of size n, where $S(i) = k$ means that the element i is assigned to location k. The agents are initialized with random solutions, where the elements are spread uniformly at random across all the available locations. Also during the initialization step, we optimize the *pocket* solutions by applying a local search that incorporate a Tabu Search (see Section 3.2).

3.1 Recombination

Concerning the selection of the parent solutions, the method **selectSol**() uses two strategies, depending on whether the population has lost diversity or not. We consider that a population is diverse if its *pocket* solutions differ at least in one value from a set of 20% of randomly chosen positions. If *diversity has not been lost*, one of the parents is the *pocket* solution of a leader agent. The second parent is the *pocket* solution from a supporter agent *within the same subpopulation*. The new solution created replaces the *current* solution of the supporter agent selected. On the contrary, if *diversity has been lost*, both parents are *pocket* solutions from supporter agents. However, in this case *the agents belong to different subpopulations*. The offspring replaces the *current* solution in one of the supporter agent. Once the parents were selected, a recombination algorithm is used to create a new solution. Our memetic algorithm uses a similar recombination to that introduced by Merz [9] and it is explained with the help of a step-by-step example described in Figure 2. Initially, all the elements assigned to the same location in both parents are copied to the offspring (elements A and E). Afterwards, we select at random an unassigned element from the offspring, say D, and look at its location in one of the parents, say parent 2. Thus, the method assigns location #3 to element D. Next, we look at the location of D in parent 1 (i.e. location #1) and check which element is in location #1 in parent 2 (i.e. element G), assigning its location to the offspring (i.e. element G goes to location #1). The process is repeated, now checking the location of element G in parent 1 (location #4). However, as location #4 is not present in parent 2, the process stops. We repeat the process starting with element H in parent

1. After processing all the elements in the offspring, element B still does not have a location because both locations #3 and #12 have already been taken. This does not happen when $n = m$. In Ref. [9] the authors do not envision this possibility because they considered only the case $n = m$. In this case, we consider a straight path between those locations and choose a random location over it, in this case location #6. If all the locations along the line have already been taken, a random one from any of the parents is chosen. Complementary to

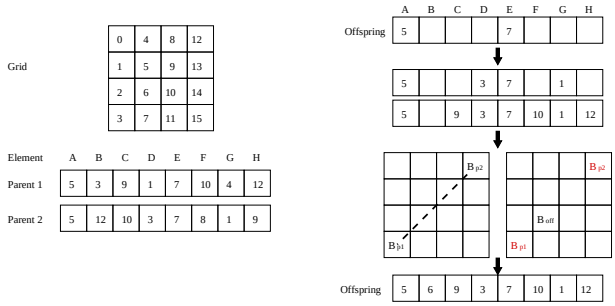

Fig. 2. A step-by-step description of the crossover procedure for the QAP problem.

the recombination operator, the mutation swaps the locations of three randomly selected elements in the solution. We use a 3-element swap scheme because in the **localSearchTS**() method (explained next in Section 3.2) all the 2-element swap movements are already considered. Mutation is always applied over the offspring after recombination.

3.2 Local Search algorithms

We implemented two local search methods (see Figure 1). (**localSearchTS**()) includes a Tabu Search implementation [12] and it is described next. The neighborhood of a solution S is defined by the swap of all pairs of elements of S. The algorithm chooses the swap that causes the best improvement in the QAP objective function. If such swap does not exist, we perform the swap that least worsens the solution. After a swap is done, any swap that brings the elements back to their previous positions become *tabu* for a number of iterations. However, a *tabu* swap shall be accepted if the objective function value of the new solution is better than the incumbent – i.e. an *aspiration criterion*. This local search is applied on each *pocket* solution of the population at the beginning of the MA and on each *current* solution after the recombination phase.

The second local search method (**8-neighborLS**()) iteratively selects an element at random and tries to move it to the eight surrounding locations in the grid, using a best-improvement strategy. Every time an element is moved to a new position, we test again its eight adjacent locations, until no improvement is possible anymore. The process iterates for all elements of the solution. As

this algorithm performs just a fine-tuning of the solution, it is applied only to the *pocket* solution of the leader agent of the population.

4 Computational Results

The memetic algorithm was coded using Java SDK 1.5.1 and generated the graph layout for the S&P100 dataset in less than 20 seconds of CPU time in a 3.0 GHz Pentium IV machine with 512Mb of RAM. The resulting graph contains 10 clusters and is shown in Figure 3. In this instance all the elements are labelled according to the industrial sector that they belong to; this allows us to better analyze the quality of the layout. Initially, this analysis takes into consideration each cluster defined by the proximity graph, as we expect those clusters to reflect the classification by industrial sector. Then, within each cluster, we will analyze any relevant structure uncovered by the QAP. Because of the space restrictions, we only give the analysis of one cluster.

Cluster #8 could be easily classified as a *services* cluster because 10 of its 17 elements belong to that sector. However, a better classification of the elements in this cluster could be obtained using the information from the layout produced by our method. In the left side of the layout there are four companies related with the packaging industry (**Alcoa**, **Du Pont**, **Allegheny Technologies** and **3M**) and two related with paper products (**OfficeMax** and **Weyerhaeuser**). These companies have been joined together with **International Paper**, which has a participation in both industries. Next to them, we can find the two railroad companies, **Norfolk Southern** and **Burlington Northern Santa Fe**. Finally there is a group of seven companies (**Black & Decker**, **Limited Brands**, **May Department Stores**, **Wal-Mart**, **Radioshack**, **Home Depot** and **Sears**) mainly related with the stores industry. The last company of this cluster is **Rockwell Automation**. It has no clear relation with the other companies, but as a *conglomerate* we cannot consider it an outlier. To compare our layout we use the classical dendogram (Figure 4) obtained with a hierarchical clustering method provided by the European Bioinformatics Institute (EBI)[2], using average linkage (UPGMA) clustering based on "correlation measure based distance" (uncentered). Even though the clustering methods developed at EBI are aimed to analyze biological datasets, their hierarchical clustering is a general approach which can be used in datasets from any source. The input is also the second derivatives of the weekly stock prices. While some technological sectors seems present, the dendogram analysis has its problems. Clusters are only defined when we "cut" the tree. Our methodology managed to automatically separate most of the sectors into distinct natural clusters uncovering similarities in the dynamics of groups of stocks. In addition, for the clusters without a sound sector majority, the QAP created a layout where the elements from different sectors were organized into smaller groups (e.g. clusters #7 and #8). The

[2] http://ep.ebi.ac.uk/EP/EPCLUST/

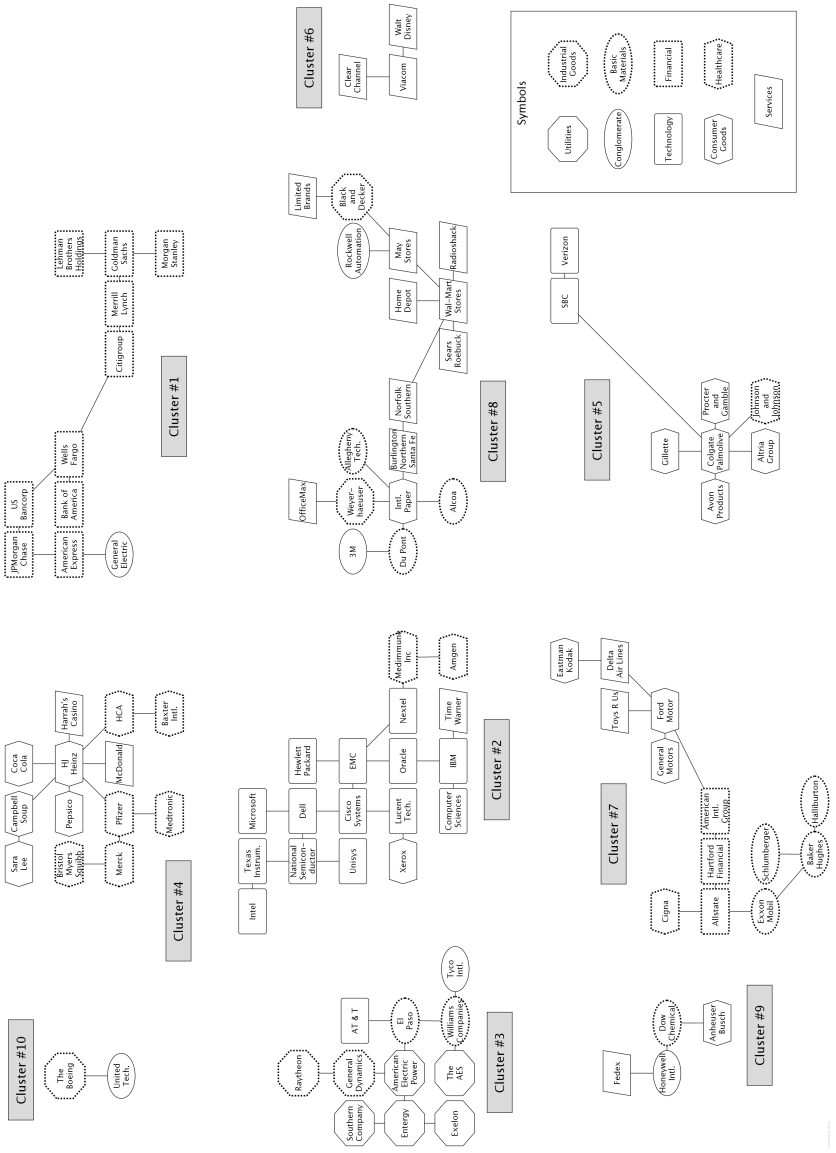

Fig. 3. Graph layout for the S&P100 dataset. The memetic algorithm solved one QAP for each cluster and an extra QAP considering each cluster as a single element, obtaining the final layout. Each shape indicates a different industrial sector represented in the dataset.

quality of the results for the S&P100 dataset supports the use of this method as a new clustering/visualization tool for other time-series data analysis problems. Our visualization methodology is not restricted to the clustering method used

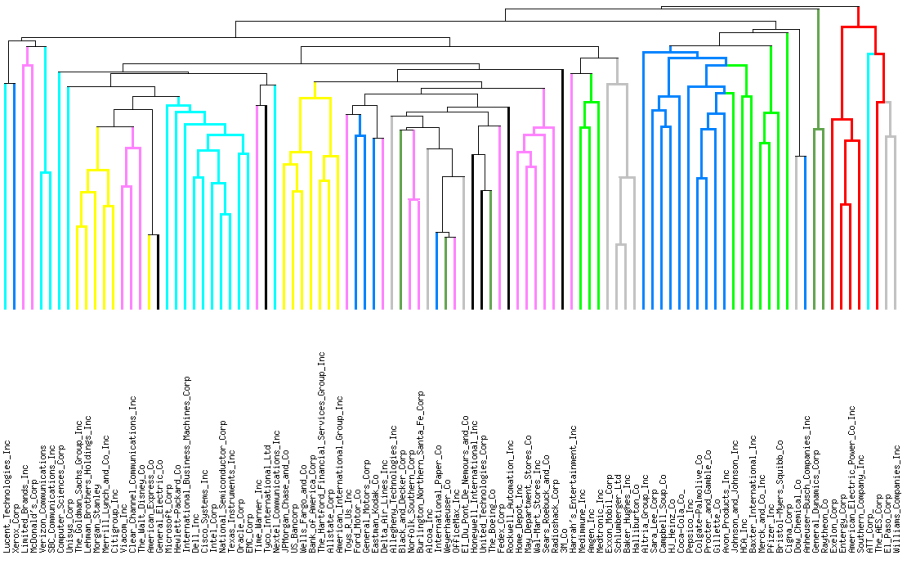

Fig. 4. Dendogram representing the average linkage hierarchical clustering from EBI.

here and the use of the QAP model and a memetic algorithm can be a good alternative for another approach to the automatic layout of weighted graphs.

5 Conclusions

A methodology for correlated data visualization based on the Quadratic Assignment Problem (QAP) was introduced in this paper. It uses a recursive partition approach that divides the dataset into small clusters. A memetic algorithm solves a separate QAP for each subgraph. The final solution obtained by our methodology correctly identifies the majority of the sectors present in a stock market dataset. In some cases, elements from distinct sectors but similar dynamical behaviors are located within the same component of the proximity graph. The success in obtaining a high-quality layout for a financial market instance – which is a challenge due to their inherent near-chaotic behaviour – makes us believe that this technique can be very useful for other problem domains.

References

1. V.V. Gafiychuk, B. Yo. Datsko, and J. Izmaylova. Analysis of data clusters obtained by self-organizing methods. *Physica A: Statistical Mechanics and its Applications*, 341:547–555, 2004.

2. N. Basalto, R. Bellotti, F. De Carlo, P. Facchi, and S. Pascazio. Clustering stock market companies via chaotic map synchronization. *Physica A: Statistical Mechanics and its Applications*, 345(1-2):196–206, 2005.
3. V. Boginski, S. Butenko, and P.M. Pardalos. On structural properties of the market graph. In A. Nagurney, editor, *Innovations in Financial and Economic Networks*, pages 28–45. Edward Elgar Publishing Inc, 2003.
4. S. Sahni and T. González. P-complete approximation problems. *Journal of the Association for Computing Machinery*, 23(3):555–565, 1976.
5. R. Burkard, E. Çela, P. Pardalos, and L. Pitsoulis. The quadratic assignment problem. In P. Pardalos and D. Du, editors, *Handbook of Combinatorial Optimization*, pages 241–338. Kluwer Academic Publishers, 1998.
6. E. Taillard. Robust taboo search for the quadratic assignment problem. *Parallel Computing*, 17(4-5):443–455, 1991.
7. C.A. Oliveira, P.M. Pardalos, and M.G.C. Resende. Grasp with path-relinking for the quadratic assignment problem. In C.C. Ribeiro and S.L. Martins, editors, *Lecture Notes in Computer Science*, volume 3059, pages 356–368. Springer-Verlag, 2004.
8. J. Carrizo, F.G. Tinetti, and P. Moscato. A computational ecology for the quadratic assignment problem. In *Proceedings of the 21st Meeting on Informatics and Operations Research*, Buenos Aires, Argentina, August, 1992.
9. P. Merz and B. Freisleben. A comparison of memetic algorithms, tabu search and ant colonies for the quadratic assigment problem. In *Proceedings of the 1999 International Congress of Evolutionary Computation (CEC'99)*, Washington DC, USA, 6-9 July, 1999.
10. R. Berretta and P. Moscato. The number partitioning problem: An open challenge for evolutionary computation? In D. Corne and M. Dorigo, editors, *New Ideas in Optimization*, pages 261–278. McGraw-Hill, 1999.
11. L.S. Buriol, P.M. Franca, and P. Moscato. A new memetic algorithm for the asymmetric traveling salesman problem. *Journal of Heuristics*, 10(3):483–506, 2004.
12. F. Glover and M. Laguna. *Tabu Search*. Kluwer Academic Publishers, Norwell, Massachusetts, 1997.
13. M. Ausloos and K. Ivanova. Mechanistic approach to generalized technical analysis of share prices and stock market indices. *The European Physical Journal B*, 27:177–187, 2002.
14. J.M. González-Barrios and A.J. Quiroz. A clustering procedure based on the comparison between the k nearest neighbors graph and the minimal spanning tree. *Statistics & Probability Letters*, 62(1):23–34, 2003.
15. P. Moscato, C. Cotta, and A. Mendes. Memetic algorithms. In G. Onwubolu and B. Babu, editors, *New Optimization Techniques in Engineering*, pages 53–86. Springer-Verlag, 2004.
16. P. Moscato. Memetic algorithms. In P. Pardalos and M. Resende, editors, *Handbook of Applied Optimization*. Oxford University Press, New York, NY, USA, 2002.
17. P. Moscato and C. Cotta. Memetic algorithms. In T.F. Gonzalez, editor, *Handbook of Approximation Algorithms and Metaheuristics*. Chapman & Hall/CRC, 2006. to appear.