The effect of the asymmetry of road transportation networks on the traveling salesman problem

Alejandro Rodríguez,*, Rubén Ruiz

* Grupo de Sistemas de Optimización Aplicada, Instituto Tecnológico de Informática, Universitat Politècnica de València, Pza. Ferrándiz Carbonell 2, 03801 Alcoy, Spain

A R T I C L E   I N F O

Available online 16 September 2011

Keywords:
Asymmetry
Asymmetric traveling salesman problem
Algorithms
Geographic information systems
Road transportation networks

A B S T R A C T

The routing of vehicles on road transportation networks is an area of great importance to transportation planners within scientific literature. This field includes well known and studied problems like traveling salesman problems or TSP or the more realistic asymmetric variant or ATSP, whose applications extend to other areas of transport and operations research. This work studies the effect that the asymmetry of road transportation networks, geographical location and territory have over TSP and ATSP methods. We conduct comprehensive experiments in order to assess the effects that these factors have on some of the best known algorithms for the TSP/ATSP. We demonstrate that all these factors have a significant influence in solution time and quality. Furthermore, we show that the solutions obtained with Euclidean matrices and those obtained with real distance matrices differ significantly.

1. Introduction

The traveling salesman problem, or TSP for short, is one of the most well known and thoroughly studied combinatorial optimization problems [1]. The objective is to find the minimum cost (usually minimum distance) route visiting a set of locations, where each location is visited exactly once. The tour must start and finish at the same location. A solution to the TSP problem is represented by a permutation of the locations. The TSP is a well known NP-Hard problem.

In routing problems, and more precisely, in the TSP (symmetric and asymmetric), there is a distance or cost matrix. Each element in the matrix contains the travel distance, time, or any other cost function between any two locations o, d, where o ≠ d. Usually, travel time, speeds and costs are a function of the distances between locations or nodes.

A wide range of the research work on the TSP can be applied to other discrete optimization problems, and to applications in several fields such as genome reconstruction, scheduling operations, machine movements for hole drilling in circuit boards or other objects, etc. and of course, its application in the routing problems and transport like the routing of aircrafts, ships, school buses, etc. [2]. As is well known, the Asymmetric Traveling Salesman Problem (ATSP) where the distance matrix is not necessarily symmetric, is a more general case than the Symmetric Traveling Salesman Problem (STSP) where the cost or distance matrix is always assumed to be symmetric. In related scientific literature these two versions of the TSP are often investigated independently, with a strong bias towards the second. Apart from the list of n locations, the input data for the TSP is just the distance matrix. As a result, carefully estimating distances between nodes are extremely important. The need for real matrices and distances has been highlighted several times in the TSP literature [3], and also for Vehicle Routing Problems (CVRP) [4], or for other variants as well [5]. Although in some forms of transportation by air, sea and train the Euclidean distances (symmetric TSP) can be a reasonable approximation; in other cases, the Euclidean distances may constitute a gross underestimation of reality, especially in urban transportation, where distances on road networks can be highly asymmetric.

In this work we deal with the issue of asymmetry in the distance matrix from road transportation networks. The main objective of this research is to measure the effect that the asymmetry of the road network has in solving the TSP, both the symmetric and the asymmetric variant. As we will demonstrate and measure, symmetric solutions—those obtained with symmetric and Euclidean distance matrices—have little in common with regard to sequence and total distance with real solutions (those obtained with asymmetric and real distances). There is usually a very significant difference between the solution of an asymmetric instance and the solution of a symmetric one based on an approximation of the real distances. Not only is the total distance significantly different, but so is the sequence of nodes in...
the solution. Furthermore, different state-of-the-art methods for the STSP and ATSP are shown to differ in effectiveness and in efficacy when tested against asymmetric real distances, compared against original performance in Euclidean settings. Some methods even no longer work when faced with asymmetric matrices. However, it is not the intention of this paper to carry out a comparison about state-of-the-art methods. Some other interesting factors that also affect the level of asymmetry and the performance of STSP and ATSP methods, like territory, geographical location and problem size are also studied. More precisely, this paper addresses the following research questions: What is the effect of the asymmetry over the effectiveness and efficiency of the main TSP/ATSP heuristics? Is it feasible to reduce the Asymmetric Traveling Salesman Problem to a symmetric one? How do all the factors behave for different problem sizes? What is the most adequate heuristic in each case?

The remainder of this paper is organized as follows: Section 2 further substantiates the importance of considering asymmetry in routing problems. Section 3 elaborates on the research questions and hypotheses, together with the studied factors and variables, experimental design and computational tests. Section 4 presents a thorough analysis of the different results from many perspectives, like CPU times, quality of the solutions and quantitative and qualitative comparisons. Finally, the conclusions of this work are presented in Section 5.

2. The real world is asymmetric

Given any TSP instance with \( n \) locations or nodes, the distance matrix between any possible pair of nodes \( o,d \), where \( o,d \in \{0,\ldots,n\} \), is denoted by \( C_{[n\times n]} \) and is a square matrix where the diagonal is usually disregarded. This matrix has \( n \times (n-1) \) elements with all the distances. In the vast majority of the routing literature, the locations or nodes are determined by their coordinates in a 2D plane and the distances between each pair of nodes are calculated by the simple Euclidean distance, given by the Pythagorean formula. In this case, it is straightforward to see that the distance between the nodes \( o \) and \( d \) is the same as the distance between \( o \) and \( d \), i.e., \( C_{od} = C_{do} \), for all \( o,d \in \{0,\ldots,n\} \). In this case, the matrix \( C \) can be summarized by an upper or lower triangular matrix with \( n \times (n-1)/2 \) elements. A slightly more elaborated approach for obtaining the matrix \( C \) is to calculate the orthodromic distance between the geolocations of two nodes. Basically, the orthodromic distance is the shortest distance between any two points on the surface of a sphere, measured along a straight path on the surface of the sphere itself. This is often referred to as the great-circle distance. Orthodromic distances are also symmetric in nature. Note that orthodromic distances are much more accurate than Euclidean distances when measuring long distances in Earth as Euclidean distances would traverse the Earth nucleus, not considering the Earth’s curvature.

It has been known for many decades [6] that Euclidean or orthodromic distances have little resemblance to real distances between nodes or locations that are linked through transportation networks or roads. As a matter of fact, the Euclidean or orthodromic distance is a very loose and weak lower bound of the shortest path that communicates any two nodes in a transportation network. Furthermore, when one considers the nature of traffic, one-way streets and the intricate layout of most roads, it is straightforward to see that, to some degree or another, real distance matrices are not symmetric. This degree of asymmetry cannot be easily estimated as it varies widely according to different factors. Long distances are likely to be more symmetric due to two-way highways. However, connecting locations in the historical centers of some big cities is likely to return asymmetric distances.

The usage of Euclidean or orthodromic distances is simply motivated by the large cost and difficulty in obtaining the real distances matrix \( C \). Even nowadays, one needs to calculate \( n \times (n-1) \) shortest paths, each one constituting an enormous effort as real transportation networks, for example inside a country, typically contain billions of nodes and arcs. Geographic Information Systems (GIS) and geo-spatial databases, along with their Advanced Programming Interfaces (API) facilitate, to some extent, this herculean task. In any case, this possibility is relatively recent as rich GIS systems capable of doing such calculations have only existed in the mainstream market since the mid 1990s. Before this date, and since the early 1970s, researchers have tried to calculate the real distance matrix indirectly from the Euclidean or orthodromic one. For example, some researchers tried to estimate real distances after multiplying the orthodromic matrix by a given factor [7]. Other works developed some functions to estimate real distances [8]. This idea was further exploited by other authors that developed distance estimation functions depending not only on the zone where nodes are located, but also on total traveled distance [6]. Many problems arise when using these functions. The proposed functions have to be adjusted mathematically and empirically (which more or less implies some validation, that in turn needs some real distance matrices). This adjustment process is an objective function dependent and also depends on the precision desired. Other authors demonstrated that this adjustment process is also dependent on the territory and other characteristics like geometry of the zone, type of transportation network, orographic accidents, natural obstacles and the like [9,10]. Therefore, distance estimation cannot be carried out over the basis of a single function or without a deep and careful study, including parameter adjustment. While we do not advocate that such functions are not useful in any environment (some strategic decisions with aggregated information might benefit from such functions, where some degree of approximation is accepted), we support the idea given in [9] that such functions are not acceptable in real operational settings.

The fact is that the existing literature has usually been concerned more with the symmetric TSP than with its asymmetric counterpart. It is clear that the first is a simpler, and more basic, problem. The best known polynomial-time heuristic with a known quality guarantee for the Euclidean TSP, due to Arora [11], is able to guarantee a \( 1+\varepsilon \) approximation factor with respect to the optimal solution, for any fixed error factor \( \varepsilon > 0 \). In contrast, the best known polynomial-time heuristic with a known quality guarantee for the ATSP (given by Asadpour et al. [12]) can only guarantee a \( \log n \) approximation factor, where \( n \) is the number of locations.

It is evident that the ATSP is a much more difficult problem, and as commented, a significant portion of the TSP literature considers Euclidean distances without even raising the issue of real distances. There is also a rich literature on the ATSP generalization and formulations, as for example, the papers of Gouveia and Pires [13] and Fischetti et al. [14], among many others like [11,15,16]. However, authors do not actually study, to the best of our knowledge, in its full complexity, the effect that different degrees of asymmetry and factors affecting asymmetry have over solution methodologies. In order to cope with all these complexities, modern GIS systems must be employed [17], together with a deep understanding of the effect of the asymmetry and other interesting factors over the calculation of real distance matrices and TSP resolution.

3. Studying the effect of asymmetry

As previously stated, we are interested in either confirming or refuting the following hypotheses: (1) Asymmetry strongly effects
the effectiveness and efficiency of the main TSP and ATSP heuristics. (2) The location of the nodes in the real world generates different levels of asymmetry and therefore also conditions TSP methods. (3) It is not always feasible to reduce the ATSP to the TSP [18] for solving real ATSP problems with TSP heuristics. (4) The size of the problem interacts with asymmetry and also affects TSP algorithms. In order to assess these hypotheses we carry out a complete comparative study of the different solutions provided by TSP methods, with real characteristics and dimensions as commented in Fischetti et al. [14]. A large set of TSP instances is generated to this end.

A full factorial experimental design is employed [19], where each generated problem instance is defined by a series of factors that are further described in the following.

3.1. Factors and instances generated

**Territory:** It is the geographical region where the instance is located. This region is bounded by a quadrant defined by two pairs of opposed geographical coordinates (latitude and longitude). This is a qualitative ordinal factor that has been tested at three variants, of increasing size, related with the Iberian peninsula (our area of interest), as shown in the leftmost picture at Fig. 1. The three regions are referred to as short, medium and large distance, respectively.

In the short distance, locations are placed in the geographical area of influence of a big city. As a result, the minimum distances between pairs of nodes are conditioned by urban transportation networks (one-way streets, traffic circles, city center, etc.). Medium distance includes short distance plus larger distances entailing regional transportation through paths, regional roads, city communication rings, etc. Lastly, large distance territories are further conditioned by large distance roads, highways and intercity communications.

**Location:** It is the placement of the nodes inside the territory. This can be random or might follow a given pattern. Three variants are defined for this nominal qualitative factor: random, grid and radial. Fig. 2 shows some examples over a given territory.

In the grid location distribution, the territory is divided into square zones. The node is placed at the center of each zone, albeit slightly displaced by a random vector. Radial distribution has a central location that services the remaining \( n - 1 \) nodes, which are radially distributed at an angular equidistance equal to \( \alpha = 2\pi/(n-1) \). Fig. 1(right) shows a map with 500 radially distributed locations in a large territory.

**Number of nodes:** This number \( n \) determines the size \( n \times (n-1)/2 \) of the matrix or, \( 2n \times (n-1) \) if it is transformed (the transformation process is detailed next). It is a quantitative factor with 10 levels: \( n = \{ 50, 100, 150, \ldots, 500 \} \).

**Symmetry:** For each generated instance, the distance matrix \( C \) is calculated in different ways. This qualitative nominal factor considers strictly symmetric or asymmetric matrices with the following studied variants:

- Orthodromic: It is the symmetric matrix with great-circle distances.
- Asymmetric: Asymmetric matrix where the distances have been calculated with the aid of a GIS, i.e., distances are actually shortest distances between locations as per the real network of roads and streets.
- Minimum arc from each pair: It is a symmetric matrix where distances have been extracted from the asymmetric matrix in a special way. Given any two distinct nodes \( o \) and \( d \), the distance for the matrix is the minimum of the two ways, i.e., the distance satisfies \( \min(c_{od}, c_{do}) \). This results in a symmetric matrix.
- Maximum arc from each pair: Similar to the previous one but taking the maximum of the two ways: \( \max(c_{od}, c_{do}) \).
- Transformed: A symmetric matrix is constructed from the asymmetric one using a well known mathematical transformation due to Jonker and Volgenant [18]. This transformation is correlated with the number of nodes in the instance as the transformation multiplies the size of the distance matrix by a factor of four. Each location or node is split into two nodes, one real, and a second virtual node. The distance between a real node and its corresponding virtual sibling is set to a very small

---

**Fig. 1.** Different territories in the Iberian peninsula (left). Example of an instance with locations following a radial distribution in a large distance territory (right).

**Fig. 2.** Examples of locations in random (left), grid (middle) and radial (right) distributions.
favorable cost (usually $-\infty$). This results in real and virtual
nodes to be consecutively placed in the final TSP tour. The
original asymmetric “from”–“to” ways are assigned to dis-
tances between real nodes in the transformed matrix whereas
original asymmetric “to”–“from” distances (i.e., the way back
distances) are assigned to the virtual nodes. All other possible
distances are assigned a very unfavorable value ($+\infty$). A
simple $3 \times 3$ symmetric matrix and its corresponding trans-
formation are given in expression (1) below:

$$
\begin{pmatrix}
0 & c_{12} & c_{13} \\
c_{21} & 0 & c_{23} \\
c_{31} & c_{32} & 0
\end{pmatrix}
\quad \Rightarrow
\begin{pmatrix}
0 & \infty & \infty & -\infty & c_{21} & c_{31} \\
\infty & 0 & \infty & -\infty & c_{32} \\
-\infty & \infty & 0 & \infty & c_{23} \\
-\infty & c_{12} & c_{13} & 0 & \infty \\
c_{21} & -\infty & c_{23} & 0 & \infty \\
c_{31} & c_{32} & -\infty & \infty & \infty & 0
\end{pmatrix}
$$

(1)

All four factors, together with their corresponding levels or
variants are gathered in Table 1.

As we can see, the last row of Table 1 contains the total
number of levels or variants for each factor. Since we employ a
full factorial experimental design, we have $3 \times 3 \times 10 \times 5 = 450$
treatments after combining all levels or variants. For each treat-
ment, five different instances are generated, for a grand total of
2250 TSP/ATSP instances. All these instances are publicly available
at http://soa.iti.es/problem-instances.

3.2. Response variables

A we will detail later, several state-of-the-art TSP methods are
used for solving the proposed instances. Using Design of Ex-
periments allows to study the effect that each considered factor
(including the different algorithms) have over one or more response
variables.

Solutions obtained after solving each instance are analyzed
mainly at two levels: quantitative (mainly tour length) and
qualitative (sequence of nodes or locations in the tour). As regards
the last qualitative assessment, the literature is marred with
papers that propose indicators for measuring the differences
between solution objects, as for example [20]. In our case,
measuring the differences between two TSP tours is commonly
covered out by counting the number of $k$–opt movements that
are needed to transform one tour $s$ into another $s'$. This needs a
non-polynomial CPU time as a function of $n$. Therefore, we employ
simpler measures of a distance $d$ between two tours or $d(s,s')$:

Relative percentage deviation from the best solution found $\Delta s^*$:

It is the relative deviation (in percentage) of the tour length
obtained after solving a given TSP instance $i$ with algorithm $A$
($S_i$) from the lowest known tour length for that instance ($S^*$).

It is calculated as follows:

$$
\Delta S^* = \frac{S_i - S^*}{S^*} \cdot 100
$$

Table 1
Factors for the instances along with their levels and variants.

<table>
<thead>
<tr>
<th>Territory (T)</th>
<th>Location (L)</th>
<th>Number of nodes (n)</th>
<th>Symmetry (M)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Short distance</td>
<td>Random</td>
<td>50</td>
<td>Orthodromic (O)</td>
</tr>
<tr>
<td>Medium distance</td>
<td>Grid</td>
<td>100</td>
<td>Asymmetric (A)</td>
</tr>
<tr>
<td>Large distance</td>
<td>Radial</td>
<td>150</td>
<td>Minimum arc (P)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Maxmum arc (G)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>500</td>
<td>Transformed (T)</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>10</td>
<td>5</td>
</tr>
</tbody>
</table>

Hamming distance $d_h$: It is a well known indicator that
measures the differences between vectors, proposed by Hamming
[21]. Basically, it takes two tours $s$ and $s'$ and adds 1 to the
indicator counter each time a position in the tour is occupied by
different nodes at both tours. For example, given $s=[2, 5, 3, 1, 4, 6]$ and
$s'=[1, 2, 3, 4, 5, 6]$, the Hamming distance is 4. There is a
problem as regards the TSP since the relative order of nodes in the
tour is as important as their absolute positions. Take a second
example $s=[6, 1, 2, 3, 4, 5]$ and $s'=[1, 2, 3, 4, 5, 6]$. In this case, the
Hamming distance is 6, even though the route is almost the same
(the only difference being the starting/ending node. However, this
indicator is simple to calculate (it just requires $O(n)$ steps) it is
easy to understand and to interpret.

Adjacency distance $d_A$: Together with the Hamming distance,
it makes sense to measure also the number of equal adjacent nodes
between two routes $s$ and $s'$, where the nodes need not be located
in the same absolute positions at the two tours. More specifically,
this is achieved by checking if the arc between nodes $e$ and $e+1$
at solution $s=(e,e+1)$—exists in any place of sequence $s'$. As
this result, the adjacency distance counts the number of distinct arcs
between two tours, with the maximum possible distance being
$n-1$. For example, given $s=[1, 2, 3, 4, 5, 6]$ and $s'=\{1, 2, 3, 4, 5\}$,
$d_A=3$, since arcs (1,6) (6,2) (5,1) of $s'$ are not present at $s$.

Using efficient data structures, $d_A$ can be calculated in $O(n)$
steps.

Note that in our measurements, both sequences are shifted so
that vertex 1 is the first vertex in the sequence, in order to have a
more precise measure of the distances (Hamming and Adjacency).

CPU time: It is the real elapsed CPU time that was needed when
solving a given instance with an algorithm. This excludes input/
output operations as well as all other system overheads, as detailed
in Alba [22].

Asymmetry in distance matrices: We are particularly interested
in measuring the asymmetry degree of matrices. Strictly speaking,
a matrix is asymmetric if it exists at least one pair $a$, $d$ such that
$c_{ad} \neq c_{da}$, where $d \in n \neq d$. Furthermore, this is even true if
$c_{ad} = c_{da} + c$, for any arbitrarily low value of $c$. Obviously, this
binary asymmetry indicator is not very informative and more
precise indicators are needed. We employ the following alter-
natives: $Alf(a)$: It indicates the asymmetry degree by counting
the number of asymmetric pairs of distances (pairs $a$, $d$ that satisfy
$c_{ad} \neq c_{da}$) over the total number of pairs $n$, using the $x_o(a,d)$
definition and expression (3) below, where $a$ is a pair of arcs $(od,do)$.
$\alpha$ takes values in the $[0\%, 100\%]$ interval. Delta
$\delta$: It measures the asymmetry degree in more detail by actually
looking at how different are asymmetric pairs (in distance). It is
interpreted with expression (4) below. Weight: It just sums all
the distances of the $C$ matrix, i.e., $\sum_{a=1}^{n} \sum_{d=1}^{n} c_{ad}$. Average
weight

$$
\| x_o(a,d) = \left\{ \begin{array}{ll}
0 & \text{if } c_{ad} = c_{da} \\
1 & \text{if } c_{ad} \neq c_{da}
\end{array} \right.
$$

$$
\alpha = \frac{2 \sum_{a=1}^{n} \sum_{d=1}^{n} x_o(a,d)}{n^2 - n} \cdot 100
$$

$$
\delta_a = \frac{|c_{ad} - c_{da}|}{\min(c_{ad}, c_{da})} \cdot 100 \quad \forall a,d \in n, \quad a \neq d
$$

3.3. Solution process

For solving all instances, we employ a high performance
computing cluster with 30 blades, each one containing 16 GB of
RAM memory and two Intel XEON E5420 processors running at
2.5 GHz. Note that each processor has four physical computing
cores (eight per blade). At this stage, it is worth mentioning the
sheer computation effort needed for calculating real distance
matrices (all instances where $M=A$ as per Table 1), especially when compared against orthodromic matrices. 450 instances in the set of 2250 contain real distances. These have been calculated by doing a humongous number of shortest route requests between pairs of nodes to Google Maps. This took 196.5 single blade equivalent CPU days. This is in stark contrast with the 21 s needed for calculating the same matrices but with orthodromic distances.

A direct outcome of this computational effort is a large set of 450 ATSP instances where distances are actually real, corresponding to current transportation networks in Spain, following all previous factors already mentioned in earlier sections. This set is complementary to the well-known TSPLIB95 dataset where only 19 synthetic ATSP instances can be found. These ATSP instances have random integer distances at each arc with $n$ sizes between 17 and 443. As indicated, these instances are publicly available. Each instance is solved with a wide range of TSP heuristics:

- Nearest neighbor algorithm ($A=\text{NN}$) as described in Flood [3]. A simple heuristic, yet with reasonable performance.
- Concorde TSP solver\(^1\) ($A=\text{CO}$). A very powerful state-of-the-art exact branch-and-cut algorithm for the TSP. It is described in Applegate et al. [24]. Parameters: default options.
- Lin–Kernighan heuristic of Lin and Kernighan [25] ($A=\text{LK}$). One of the most well known powerful and well-known heuristics.
- Improved Lin–Kernighan of Helsgaun [26] ($A=\text{HE}$). This is currently considered as one of the state-of-the-art methods for solving the TSP. Parameters: author’s recommendations.
- Memetic algorithm of Nagata and Kobayashi [27] and Nagata [28] ($A=\text{NA}$). Also one of the most important and adaptive heuristics. Parameters: 10 trials, population size = 100, 30 children, 2 parents.
- Branch-and-cut method of Fischetti et al. [14] ($A=\text{FI}$). In our experiments, only for problem size $n=50$. Parameters: optimized compilation, internal limit of 150,000 branching nodes.
- Improved GKS/TB COP heuristics of [29] ($A=\text{GO}$), based on the Helsgaun code. Parameters: author’s recommendations.

As we can see, the selection of TSP heuristics is motivated either by simplicity, asymmetry adaptation or by current state-of-the-art performance. Note that not all studied heuristics are capable of working over asymmetric matrices. For example, the LK and CO methods are specifically designed for the TSP and not for the ATSP [30]. In these cases, the transformed matrix ($M=T$) is employed instead of the real asymmetric one. This results in 2250 instances $\times$ 8 algorithms $\times$ 450 asymmetric matrices $\times$ 2 non-ATSP heuristics (LK and CO) $\times$ 450 ($M=T$ unsupported on CO) $\times$ 2250 (instances $n>50$ size on FI) = 14,625 computational experiments. All these experiments needed 830 single blade equivalent CPU hours. No CPU time limit was imposed to any algorithm. The algorithms were implemented and ran following the instructions of their respective authors.

4. Analysis of results

All results are supported by statistical analyses. We mainly use the multifactor Analysis of Variance (ANOVA) technique where we control all studied factors. Three different groups of response variables are considered: CPU times needed by the algorithms, quantitative and qualitative comparison of symmetric (TSP) and asymmetric (ATSP) tours. All results are detailed in the following sections. Since the ANOVA is a parametric technique, one needs to check the three main hypotheses which are normality, homoscedasticity and independence of the residuals. The residuals resulting from the experiment were analyzed and no serious deviations were observed.

4.1. CPU times

Some of the most interesting results are observed when analyzing the CPU times needed by the algorithms capable of solving ATSP problems. The resulting ANOVA table is given in Table 2.

At a 95% confidence level ($\alpha=0.05$), all single factors and six double factor interactions are statistically significant. Among significant factors, importance is observed by the magnitude of the $F$-Ratio. For example, the $F$-Ratio of the factor “Algorithm” is not less than 3145.2. This means that the differences among the different algorithms generate 3145.2 more variance than the variance obtained within each algorithm. Therefore, the type of algorithm has a very strong and statistically significant influence over the CPU time.

The ANOVA technique mainly points out statistical significance. For a further understanding of the behavior of any studied factor, we need descriptive plots. We have included plots with points and smoothed lines for comparing the CPU time as a function of the size of the matrices for all combinations of symmetry (types of matrices) and algorithms factors. All these plots are shown in Fig. 3 where the X-axis gives the size of the matrix ($n$) and the Y-axis the CPU time in seconds, with a logarithmic scale. Each row in the plot corresponds to a type of matrix and each column to an algorithm. Note that there are no plots for algorithms LK and CO for asymmetric matrices, and GO for transformed matrices ($M=T$).

We confirm that the type of algorithm has a strong influence over CPU time. As expected, the improved Lin–Kernighan methods of Helsgaun (HE) and Concorde (CO) are the most computationally demanding algorithms. The size of the matrix also affects CPU time directly and exponentially and this is the case for all types of matrices, symmetric and asymmetric and for all algorithms. Also expected is the matrix transformation process of

<table>
<thead>
<tr>
<th>Source</th>
<th>Sum of squares</th>
<th>Degrees of freedom</th>
<th>Mean square</th>
<th>$F$-Ratio</th>
<th>$p$-value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Main effects</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A:Territory</td>
<td>2900.6</td>
<td>2</td>
<td>1450.3</td>
<td>17.7</td>
<td>0.0000</td>
</tr>
<tr>
<td>B:Location</td>
<td>1028.8</td>
<td>2</td>
<td>514.4</td>
<td>6.3</td>
<td>0.0019</td>
</tr>
<tr>
<td>C:Symmetry</td>
<td>2502.2</td>
<td>3</td>
<td>834.1</td>
<td>10.2</td>
<td>0.0000</td>
</tr>
<tr>
<td>D:FI</td>
<td>753 030</td>
<td>8</td>
<td>94 128.8</td>
<td>1148.7</td>
<td>0.0000</td>
</tr>
<tr>
<td>E:Algorithm</td>
<td>1 544 66</td>
<td>6</td>
<td>257 723</td>
<td>3145.2</td>
<td>0.0000</td>
</tr>
<tr>
<td><strong>Interactions</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AB</td>
<td>351.6</td>
<td>4</td>
<td>87.9</td>
<td>1.1</td>
<td>0.3681</td>
</tr>
<tr>
<td>AC</td>
<td>1221.9</td>
<td>6</td>
<td>203.7</td>
<td>2.5</td>
<td>0.0210</td>
</tr>
<tr>
<td>AD</td>
<td>2664.9</td>
<td>16</td>
<td>166.6</td>
<td>2.1</td>
<td>0.0086</td>
</tr>
<tr>
<td>AE</td>
<td>11786.4</td>
<td>12</td>
<td>982.2</td>
<td>12.0</td>
<td>0.0000</td>
</tr>
<tr>
<td>BC</td>
<td>236.5</td>
<td>6</td>
<td>39.4</td>
<td>0.5</td>
<td>0.8229</td>
</tr>
<tr>
<td>BD</td>
<td>1788.7</td>
<td>16</td>
<td>111.8</td>
<td>1.4</td>
<td>0.1491</td>
</tr>
<tr>
<td>BE</td>
<td>4835</td>
<td>12</td>
<td>402.9</td>
<td>4.9</td>
<td>0.0000</td>
</tr>
<tr>
<td>CD</td>
<td>3348.8</td>
<td>24</td>
<td>139.5</td>
<td>1.7</td>
<td>0.0174</td>
</tr>
<tr>
<td>DE</td>
<td>1 035 66</td>
<td>48</td>
<td>21 638.7</td>
<td>264.1</td>
<td>0.0000</td>
</tr>
<tr>
<td>Residual</td>
<td>849 253</td>
<td>10 364</td>
<td>81.9</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\(^1\) http://www.tsp.gatech.edu/concorde.html
matrices ($M = T$), which results in enormous CPU time increases. This is a logical result that validates the whole experiment, as the size of the original asymmetric matrix is multiplied by two in the transformation process. However, and as shown in Fig. 3 with symmetric matrices, HE is actually slower than CO (about three times slower). This is an unexpected result as CO is an exact procedure and HE, albeit extremely effective, cannot guarantee optimality. The matrix transformation ($M = T$) affects much more CO than HE as the CPU time increases approximately by a factor of 7. The problem is that CO only works with symmetric matrices and the transformation is the only possible way of dealing with asymmetric problems. The corollary is that CO is far more sensible to the size of the TSP to solve.

As regards the other studied factors, the location affects the CPU time for all methods. The means plot of Fig. 4 contains the interaction between algorithm and location factors for asymmetric matrices only. The means are plotted in the middle of Tukey’s Honest Significant Difference (HSD) 95% confidence intervals. Overlapping intervals denote that the means contained within them are not statistically different. Grid locations result in slightly higher CPU times for all methods compared to the Random and Radial locations; except for the FI algorithm as it consumes more CPU time for Random location ($n = 50$). In the case of symmetric matrices (not shown in the figure), CPU times increase sharply (20%) for the CO method for Grid locations and this difference is statistically significant. To the best of our knowledge, there are no reported studies that analyze how the distribution of the nodes or clients and road transportation networks affect the CPU times of state-of-the-art methods.

Other interesting findings affect the territory factor. Fig. 5 shows the interaction between algorithms and the territory factor for asymmetric matrices only. In general, and especially for FI, there is a preference of algorithms for short distances territories. A possible explanation is that in short distances there is more variability in the distances between nodes and possibly this helps in the process of finding a solution. In case of symmetric matrices (not shown in the figure), CPU time increases up to 37% for CO in cases of long and medium distances compared with the short distances territories. A contribution of this work is to demonstrate that the different degrees of asymmetry (and differences between distances) of the road transportation network affect studied methods accordingly.

Finally, CPU time is affected, on average, by the symmetry factor (type of distance matrix). If we remove the transformed matrix ($M = T$) which we have already seen increase in CPU times by orders of magnitude, the result obtained is shown in Fig. 6. We see that there are no statistically significant differences between the $M = P$ and $M = G$ matrices. Recall that these represent symmetric matrices where the distances are the minimum and maximum distances, respectively, between the from–to and to–from asymmetric distances in the matrix. This means that the

![Fig. 3. CPU time behavior for algorithms (CO, LK, HE, NA, FI, GO), types and sizes of matrices.](image)

![Fig. 4. Means plot with Tukey’s Honest Significant Difference (HSD) 95% confidence intervals for the interaction between location and algorithm factors, where the response variable is CPU time ($M = A$).](image)
4.2. Quality of solutions

It has to be reminded that the objective at this step is not to measure which algorithm, among the tested ones, is the best. The focus is rather on studying how the considered factors affect the measure which algorithm, among the tested ones, is the best. The literature, publications and new results are often disputed with Fig. 5. Means plot with Tukey's Honest Significant Difference (HSD) 95% confidence intervals for the symmetry factor (type of matrix), where the response variable is CPU time (M=A).

Fig. 6. Means plot with Tukey's Honest Significant Difference (HSD) 95% confidence intervals for the interaction between territory and algorithm factors, where the response variable is CPU time (M=T).

differences in CPU time cannot be attributed to the magnitude of the distances, but rather to the differences in the distances themselves. We also observe how asymmetric matrices (M=A) need significantly more CPU time than regular orthodromic matrices (M=O).

4.2. Quality of solutions

It has to be reminded that the objective at this step is not to measure which algorithm, among the tested ones, is the best. The focus is rather on studying how the considered factors affect the quality of the solutions provided by the algorithms. Table 3 provides the number of times that each algorithm provides the best solution (N, S*), and the corresponding rate (% S*) under three different settings. The second and third columns indicate matrices M=(O, P, G) (1350 experiments per algorithm). The fourth, fifth, sixth and seventh columns indicate asymmetric (M=T) cases, respectively, with 450 experiments per algorithm. Note that in the case of A=GO there is no data for M=T, and for A=FI there is only data for n=50 problem size.

As expected, CO always produces the optimum solution for the 1350 symmetric instances. Expectedly, HE’s success rate is high at almost a 69% and much better than LK’s at 46%. The simple heuristics NN and 2O are rarely successful. For asymmetric matrices (M=A), FI is clearly dominant on small sized problems (n=50). For other asymmetric cases, HE is clearly dominant and no further results can be drawn from our comparison since LK and CO do not accept asymmetric matrices. Most surprisingly, CO does not obtain the optimum solutions in all cases for M=T. An obvious explanation is that in the transformation process, some values in the matrix are −∞ or +∞ and this creates numerical instability problems inside CO that result in small deviations from the true optimum solution. These ∞ values were also the reason for not experimenting with M=T for the GO method. With these results, we can now conclude that not only CO needs an exponentially greater CPU time for transformed matrices, but also that the results cannot be trusted. Naturally, with modifications inside the CO code, there is the possibility that transformed matrices could be considered without glitches. Another interesting outcome is that for transformed matrices, NA and LK outperform HE. However, the transformation process is actually not needed for HE or NA and we cannot conclude that LK or NA are preferred over HE for transformed matrices.

Numerous statistical analyses were performed in order to check the influence of the studied factors over the quality of the solutions. Multiple ANOVA experiments were performed, which are not fully detailed due to space restrictions. It has to be noted that since no maximum CPU time was given to all tested algorithms, the effect of the different studied factors over solution quality is about 1% or less (contrary to the previous observed effects on CPU time). While this might be seen as a marginal effect, it has to be reminded that in the TSP state-of-the-art literature, publications and new results are often disputed with improvements of less than 2% in solution quality [26]. However, almost all factors resulted statistically significant in all tests carried out. Table 4 shows average ΔS values, defined by expression (2) in previous sections, for the different tested algorithms as a function of the type of matrix (symmetry factor). Again, we see the large deterioration in FI, HE, NA and CO with transformed matrices (M=T). For large asymmetric matrices, HE, NA and GO are good choices. Whereas for all other matrices, either CO or HE clearly dominate. Once again we see that for transformed matrices, it is even better to use LK than HE.

After studying the different matrices’ asymmetry degree and all other studied factors, we found that the asymmetry degree of a
matrix \((\delta)\) and the territory factor are strongly related. In our experiments we have observed and demonstrated that the asymmetry of the arcs on short distances territories is much higher than medium or large distances territories, which is relevant and interesting for the case of city logistics problems. It is logical to think that these differences in the degree of asymmetry affect the behavior of algorithms, as shown at Table 5.

It is relevant and interesting to observe how depending on the type of algorithm, there are preferences for more symmetrical or asymmetrical environments, or what is equivalent: long distances (CO, HE, NA) and medium distances (LK) versus short distances (FI, GO, NN, 2O). Furthermore, although not shown, this effect is observed for all matrix sizes and specially for transformed matrices. We now analyze the behavior of the different algorithms against the location factor in Table 6. As shown, no overly strong effects are observed (albeit all differences are statistically significant for location factor). It is interesting to note that although the road network in the Iberian Peninsula is characterized by a radial structure centered in the capital Madrid, the degree of asymmetry increases slightly in such locations. Other road networks in other countries could be an important relationship between location and asymmetry of the arcs that conditions more the behavior of algorithms.

Lastly, it is worth mentioning that matrix size has a very small impact on \(\Delta S_i^r\) values. The effect is less than 0.06% in the worst case. Fig. 7 shows the averages of the symmetry and algorithms factors (excluding M=T). The horizontal axis shows the size of the matrix \(n\) and the vertical axis (different scale for each symmetry factor) shows the percentage deviation \(\Delta S_i^r\) over the best solution. Note the vertical scale and values for M=T matrices, where the effect of this type of matrix and size \(n\) on \(\Delta S_i^r\) is higher.

### 4.3. Quantitative and qualitative assessment

After studying the different matrices’ asymmetry degree and all other studied factors, we found that the asymmetry degree of a matrix \((\delta)\) and the average weight \((\text{Weight})\) of the different distances in the matrix are strongly related. If there is a relation between symmetric and asymmetric matrices in the form of an increased average weight, it is logical to think that the symmetric solution of the TSP could be “augmented” in order to carefully estimate the real ATSP solution (as regards the total tour length).

---

**Table 5**

Average \(\Delta S_i^r\) values according to algorithm and territory factors.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Short</th>
<th>Medium</th>
<th>Large</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>NN</td>
<td>24.28</td>
<td>21.88</td>
<td>21.95</td>
<td>22.70</td>
</tr>
<tr>
<td>2O</td>
<td>119.02</td>
<td>136.98</td>
<td>145.51</td>
<td>133.84</td>
</tr>
<tr>
<td>LK</td>
<td>0.42</td>
<td>0.41</td>
<td>0.44</td>
<td>0.42</td>
</tr>
<tr>
<td>CO</td>
<td>0.41</td>
<td>0.30</td>
<td>0.18</td>
<td>0.30</td>
</tr>
<tr>
<td>HE</td>
<td>3.40</td>
<td>1.09</td>
<td>1.00</td>
<td>1.83</td>
</tr>
<tr>
<td>NA</td>
<td>2.92</td>
<td>0.76</td>
<td>0.65</td>
<td>1.45</td>
</tr>
<tr>
<td>FI</td>
<td>4.79</td>
<td>5.50</td>
<td>5.39</td>
<td>5.23</td>
</tr>
<tr>
<td>GO</td>
<td>0.31</td>
<td>0.38</td>
<td>0.40</td>
<td>0.36</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>19.44</strong></td>
<td><strong>20.91</strong></td>
<td><strong>21.93</strong></td>
<td></td>
</tr>
</tbody>
</table>

**Table 6**

Average \(\Delta S_i^r\) values according to algorithm and location factors.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Random</th>
<th>Grid</th>
<th>Radial</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>NN</td>
<td>23.49</td>
<td>22.52</td>
<td>22.10</td>
<td>22.70</td>
</tr>
<tr>
<td>2O</td>
<td>150.47</td>
<td>124.90</td>
<td>126.14</td>
<td>133.84</td>
</tr>
<tr>
<td>LK</td>
<td>0.45</td>
<td>0.31</td>
<td>0.50</td>
<td>0.42</td>
</tr>
<tr>
<td>CO</td>
<td>0.25</td>
<td>0.53</td>
<td>0.12</td>
<td>0.30</td>
</tr>
<tr>
<td>HE</td>
<td>1.98</td>
<td>1.52</td>
<td>1.99</td>
<td>1.83</td>
</tr>
<tr>
<td>NA</td>
<td>1.54</td>
<td>1.06</td>
<td>1.73</td>
<td>1.45</td>
</tr>
<tr>
<td>FI</td>
<td>5.56</td>
<td>6.73</td>
<td>3.38</td>
<td>5.23</td>
</tr>
<tr>
<td>GO</td>
<td>0.44</td>
<td>0.27</td>
<td>0.39</td>
<td>0.36</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>23.02</strong></td>
<td><strong>19.73</strong></td>
<td><strong>19.54</strong></td>
<td></td>
</tr>
</tbody>
</table>

---

**Fig. 7.** Average \(\Delta S_i^r\) values according to the size of the matrix \(n\), algorithms (CO, LK, HE, NA, FI, GO) and symmetry factors.
This is needed since, as we have already stated, the TSP tour length is a loose lower bound of the real ATSP tour length. Similarly, it is important to check the tour length of the TSP solution, when calculated with the ATSP matrix and vice versa. In order to check all these questions we use the following indicators, which are strongly based on the previously defined response variables.

\( \Delta ATSP: \) It is the percentage increase of the tour length of the ATSP solution as regards the TSP solution:

\[
\Delta ATSP = \frac{ATSP - TSP}{TSP} \cdot 100
\]  

(5)

TSP is the tour length of the symmetric problem, calculated with symmetric orthodromic matrices. ATSP is the asymmetric problem tour length, calculated with real distance matrices (\( M = A \)). Note that algorithms LK and CO cannot solve the ATSP. In these cases the transformed matrix (\( M = T \)) is used instead.

\( TSPA: \) The TSP solution is calculated with asymmetric matrices; i.e., we take the solution of a TSP problem and recalculate it with the real distances. Obviously, the tour length will increase (\( TSPA \geq TSP \)).

\( ATSP_A: \) It is the opposite case as \( TSPA \). The ATSP solution is calculated with the symmetric matrix.

\( \Delta ATSP_A: \) It is the percentage increase of \( TSPA \) against ATSP. It could be positive or negative.

\( \Delta ATSP_0: \) It is the percentage increase or decrease of \( ATSP_0 \) against TSP.

\( \Delta dH: \) It is the percentage of differences in the TSP solution against ATSP. Values close to 100% qualitatively indicate that the TSP solution is very different from the ATSP. It is based on the previously defined Hamming distance \( dH: \)

\[
\Delta dH = \frac{dH}{n} \cdot 100
\]  

(6)

\( \Delta da: \) It is the percentage of different arcs, over the total arcs that the TSP sequence has over the ATSP solution. It is based on the previous adjacency distance \( da: \)

\[
\Delta da = \frac{dA}{n+1} \cdot 100
\]  

(7)

![Fig. 8. Histograms with the \( \Delta ATSP \) (left) and \( ATSP_0 \) (right) frequency distributions.](image)

We calculate all previous indicators for all experiments, namely 3 territories \( \times 3 \) locations \( \times 10 \) different matrix sizes \( \times 8 \) different algorithms \( \times 5 \) replicates which results in 3195 data. The results are good. The average \( \Delta ATSP \) indicator reaches a value of 80.1%. This indicates a huge difference between the ATSP and TSP solutions. Note that the minimum observed value for this indicator is an already large 32.9% (the maximum being an impressive 196.9%). The frequency distributions of the \( \Delta ATSP \) values are given as an histogram in Fig. 8 (left). It is observed that in a large percentage of the cases, the increase is between 50% and 100%. Fig. 8 (right) shows a second histogram, this time for \( ATSP_A \). The distribution is clearly skewed towards positive values, with an average of 13.6%. Exactly, 39.8% of the cases show differences equal or larger than 10%.

A very strong result, especially with the second histogram, is that there is a large difference between solving symmetric and asymmetric problems. The idea that orthodromic or Euclidean distances for solving the TSP are valid (seen in some commercial routing software) and that after all, one can later calculate real distances with the solution is utterly wrong. As shown, on average, there are differences that amount to a 13.6%. Note that these differences are extremely large for TSP problems where algorithms fight in the state-of-the-art segment for less than 2% improvements. As regards \( ATSP_0 \), the distribution (not shown) is again positive, with an average of 14.3%. In a 50.02% of the cases, the differences are equal or greater than 10%. Once again, it is shown that it is not the same solving an asymmetric ATSP and measuring the tour length of the sequence with symmetric distances.

In our opinion, this is the main contribution of this research work. From our perspective, the commonly accepted assumption of considering Euclidean distances does not hold when solutions are calculated with real asymmetric distances. The TSP solutions deteriorate enormously when calculated with real matrices. Furthermore, there is little guarantee that good algorithms for the TSP will work equally good for the ATSP. Often, the degradation in performance will be significantly greater than the observed differences between competing methods. Another significant result is to qualitatively observe the big differences between the sequences obtained with symmetric TSP and asymmetric ATSP problems. The average \( \Delta dA \), with a value of 71.0% indicates that the symmetric sequences are almost entirely different from the asymmetric ones. Values are even greater if one uses the Hamming distance \( \Delta dH \). The minimum value for the \( \Delta dA \) value is as high as 6%. This means that, in the best case, a full 6% of the sequence is different. Fig. 9 (left) shows the frequency
distribution of $\Delta dH$. Note how it is heavily skewed towards values very close to 100%. Fig. 9 (right) shows the corresponding histogram for $\Delta dA$.

### 4.4. Some examples

In this section we depict two examples. The purpose is to graphically show the large differences between symmetric and asymmetric solutions. Only two examples have been randomly selected due to obvious space constraints.

**CC-0503HE**: The first example corresponds to instance CC-0503. A total of 50 nodes are grid located in a short distance territory. We compare the best solution obtained with algorithm HE. There are huge differences between the symmetric optimal tour length of 244.2 km and the 423.3 km of the different optimal tour length calculated with the asymmetric distances. Therefore, $\Delta ATSP = 73.32\%$. Additionally, we have that $\Delta dA = 90.20\%$. Fig. 10 (left) has two optimum solutions superimposed. The symmetric in blue and the real one in red.

**MR-0504CO**: In this second example we show the results of the instance MR-0504 solved with CO algorithm. We have again 50 nodes radially distributed in a medium distance territory. The total tour length in the symmetric tour is 1369.7 km versus 2097.9 km for the asymmetric tour length. This results in a $\Delta ATSP$ of 53.17%, with a $\Delta dA$ of 98.04%. Fig. 10 (right) shows the details.

### 5. Conclusions and further work

In this work we have studied the effect that the asymmetry has over the solution process of the TSP/ATSP. We have shown that solving the TSP differs much from solving the ATSP at so many levels: tour length, adjacency of nodes, hamming distance and CPU time. Furthermore, all these differences are strongly affected by other and new studied factors: degree of symmetry, territory and location of the nodes in the road transportation network. During the research, thousands of instances have been solved with some of the best well known (including some state-of-the-art) algorithms for the TSP and ATSP.

We have been able to confirm, as expected, that the algorithm used strongly affects the CPU time. HE and CO are the most computationally demanding methods, much more than the other simplistic heuristics or modern meta-heuristics. We have also shown that the territory, location and size of the problem factors all affect the different methods in a sound and statistically significant way. As is known, the problem size increases the CPU time exponentially. An interesting and novel contribution of this paper is the study of the effect on CPU time and quality of solutions due to the factors of territory, location and their relationship with the asymmetry degree of the road transportation network. Particularly for symmetric matrices, locations in grid have a significant effect on CPU times, resulting in harder to solve problems. This effect is amplified with transformed
matrices. It has been demonstrated that the territory has an impact on CPU time, especially for short distance territories. For symmetric transformed matrices, the territory factor is relevant as regards the quality of the solution. This effect is observed for all matrix sizes. Another conclusion of this study is that the transformation process has a profound effect over CPU times. Furthermore, this transformation process has shown not to be entirely feasible for algorithms HE and, especially, for CO. If one closely compares HE and CO for symmetric matrices, we observe that HE is about three times slower than CO. This is unexpected, since CO is an exact branch-and-cut method and HE a (powerful) heuristic. However, the transformation process increases the computation time more than sevenfold, and affects CO much more. The result is that asymmetry has a deep impact over algorithms, either from the quality of the solutions standpoint, or from the CPU time whenever matrix transformation is needed.

Comprehensive statistical experiments further demonstrate that there is an inverse relation between the territory factor and the average $\Delta$ATSP indicator value. This confirms that there are quantitative differences between the TSP and ATSP solutions. The differences between the two solutions are smaller for large distances than for short distances. Furthermore, we found relations between the location factor and the $\Delta$ATSP indicator. In this case, the existing differences are greater as the size of the problems grows and they are greater for radially and randomly placed locations than for grid ones. The size of the matrix also conditions the differences between the TSP and ATSP solutions. In any case, solving a TSP and later calculating the real tour length with real distances is not a viable solution process.

The asymmetry is not just a binary condition of the problem. There are different degrees of asymmetry as the territory and the location of the nodes in the road transportation network. This results in different degrees of complexity, and the effect on CPU time and solution quality. Some algorithms (HE, FI, NA, GO) are better equipped to solve problems with a high degree of asymmetry or short distances, and in certain location patterns. Others (LK, CO, HE, NA) provide better performance in symmetrical environments or long distances. This paper hopes to inspire future research on the development and testing of new and improved algorithms, not only taking into account the asymmetry condition, but also new factors studied here. New instances have been made available to the scientific community. Further work stems from the possibility of providing effective methods for calculating real asymmetric travel matrices, as this imposes today a clear entry barrier for those researchers not willing to use the typical Euclidean matrices. Extending this study to more complex problems, like for the Capacitated Vehicle Routing Problem CVRP or for the Heterogeneous Fleet variant (HFCVRP) is of interest to see if more realistic routing problems are equally affected by asymmetry.

Acknowledgments

The authors would like to warmly thank Prof. Keld Helsgaun, Prof. Yuichi Nagata, Profs. Boris Goldengorin and Gerold Jäger, and especially Prof. Matteo Fischetti, for facilitating the code of their great algorithms. This work is partially funded by the Spanish Ministry of Science and Innovation, under the project “SMPA – Advanced Parallel Multiobjective Sequencing: Practical and Theoretical Advances” with reference DPI2008-03511/DPI. The authors should also thank the IMPIVA—Institute for the Small and Medium Valencian Enterprise, for the project OSC with references IMIDIC/2008/137, IMIDIC/2009/198 and IMIDIC/2010/175 and the Polytechnic University of Valencia, for the project PPAR with reference 3147.

References