A Parallel Approach for Solving a Large-Scale Traveling Salesman Problem

Roman Bazylevych¹, Rémy Dupas², *Bhanu Prasad³, Bohdan Kuz⁴, Roman Kutelmakh⁵, Lubov Bazylevych⁶

¹University of Information Technology and Management in Rzeszow, Poland
²Université de Bordeaux 1 - IMS UMR 5218 CNRS-LAPS, France
³Department of Computer and Information Sciences, Florida Agricultural and Mechanical University, Tallahassee, Florida 32307, USA
⁴-⁵Lviv Polytechnic National University, Lviv, Ukraine
⁶Institute of Applied Problems of Mechanics and Mathematics, NASU, Lviv, Ukraine

E-mails: ¹rbazylevych@wsiz.rzeszow.pl, ²remy.dupas@gmail.com, ³prasad@cis.famu.edu,
⁴bohdankuz@rambler.ru, ⁵rkutelmakh@ua.fm, ⁶lbaz@iapmm.lviv.ua

*Contact author

Abstract. A parallel approach for solving a large-scale Euclidean Traveling Salesman Problem (TSP) is presented. The approach consists of the following stages: partitioning the input data set into clusters, solving the TSP for each cluster to get partial solutions, merging the partial solutions to form a complete solution, and optimizing the complete solution. Lin-Kernighan-Helsgaun (LKH) algorithm is used to solve TSP in each cluster. Rings are used to form a complete solution. Several optimization strategies are presented to improve the quality of the complete solution. The approach minimizes the run time of the overall solution. The quality loss of the solution obtained with this approach is negligible when compared to the best known solutions.

Keywords: parallel approach, combinatorial optimization, decomposition, clustering, TSP.

1. Introduction

The Traveling Salesman Problem (TSP) is NP-hard and, for a large-scale problem, it is not possible to find the optimal solution in a reasonable amount of time [1, 2, 3, 4, 5, 6]. Many applications such as integrated circuit design, scheduling, analysis and synthesis of chemical structures, and continuous line drawings can be formally described as TSPs [1, 2]. A lot of research has been done on finding approximate solutions that are close to the optimal solutions [3, 4, 5].

Existing heuristic methods for solving TSP have time complexity of $O(N^2)$ or above, where $N$ is the number of points [1, 3, 7, 16, 17, 18]. Hence, these methods are not very effective to solve large-scale TSP problems. Applegate et al. [6] have developed the optimal solution for a TSP problem that has 85900-points and it is the
largest problem that is solved optimally until now. This solution required nearly 136 years of CPU time.

Some algorithms for solving TSP have used “clusters” and they can execute in parallel (i.e., execute using multiple processors at the same time) and provide high quality solutions [7, 8, 9]. The most recent work by Bazylevych et al. [10] has provided a “ring” algorithm that can be parallelized. The goal of this research is to develop a parallel approach for solving a large-scale Euclidean TSP problem and evaluate the performance of that approach.

The rest of the paper is organized as follows. Sections 2 and 3 focused on the problem formulation and the important stages of the developed approach. Section 4 provides the details of clusters’ formation. Section 5 provides the details of partial solutions and merging (i.e., joining) them to get a complete solution. Section 6 provides some approaches for optimizing the complete solution. Section 7 provides some experimental results of the developed approach. Finally, Section 8 concludes the research.

2. Problem Formulation

Given a set \( P = \{p_1, p_2, \ldots, p_N\} \) of \( N \) points that are described by their coordinates \( p_i = (x_i, y_i) \), the distance function \( \text{dist}: P \times P \rightarrow R \), where \( R \) is the set of real numbers, is defined by the Euclidean distance or by any other metric as follows:

\[
\text{dist}(p_i, p_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}
\]

\( \forall i, j \in \{1, 2, \ldots, N\} \) and \( r_{ij} \in R \).

The problem is to minimize the length of the closed route that visits each point of \( P \) only once and returns back to the starting point. That is, the problem is to find

\[
M = <m_1, m_2, \ldots, m_N, m_1> \in P \ \forall \ k \in \{1, 2, \ldots, N\}:
\]

\[
\text{length of} \ M = \sum_{i=1}^{N-1} \text{dist}(m_i, m_{i+1}) + \text{dist}(m_N, m_1) \rightarrow \text{minimum}.
\]

The problem is symmetric if \( \text{dist}(p_i, p_j) = \text{dist}(p_j, p_i) \ \forall \ i, j \in \{1, 2, \ldots, N\} \) and it is asymmetric otherwise. The problem considered in this research is of symmetric type.

3. Main Stages of the Developed Approach

The developed parallel approach for solving a large-scale TSP consists of the following stages:

1. Decomposition of the set \( P \) of given points into clusters.
2. Finding TSP solutions for each cluster to get partial solutions.
3. Merging the partial solutions to get a complete solution for \( P \).
4. Optimizing the complete solution.
P is divided into a limited number of clusters and the size of each cluster (i.e., the number of points in each cluster) is very small when compared to that of P. Lin-Kernighan-Helsgaun (LKH) algorithm [3, 5, 14] is used to find TSP solutions in each cluster. These solutions are called partial solutions because these solutions are not yet merged together to form a complete solution.

The run time T of the complete solution (i.e., the run time of the TSP solution for the entire P) of the developed approach is given by

\[ T = t_c + t_{ps} + t_{js} \]

where \( t_c \) is the time to divide \( P \) into clusters, \( t_{ps} \) is the time required to find the partial solutions and \( t_{js} \) is the time required to merge the partial solutions. The maximum degree of parallelism (i.e., the maximum number of processors that can run in parallel) during partial solutions’ finding stage is equal to the total number of clusters formed in that stage. The duration for finding the partial solutions in parallel is:

\[ t_{ps} = t_0 \]

where \( t_0 \) is the time to solve the TSP in a single cluster that has \( n_0 \) points. The partial solutions are merged to form a complete solution. The merging stage uses adjacent clusters. All the clusters together form a planar graph, if the border between any two adjacent clusters is considered as an edge and the ends of those edges are considered as vertices of the planar graph. Three or more different borders meet at every vertex. The chromatic number of that planar graph is 4, according to the four color theorem [11]. This number determines the number of steps needed to find, in parallel, the complete solution at Stage 3 of the developed approach. More details about these steps are provided in Section 5.

The number of vertices’ groups of the planar graph, each group painted in a different color, shows the maximum degree of parallelism achieved while merging the partial solutions. That is, \( t_{js} = 4 \cdot t_0 \). If the number of points in a ring is same as that of the cluster then the time for finding a solution in one ring is approximately equal to the time for finding a partial solution. More precisely, the time for finding the solution in one ring is little bit more because there are additional operations such as finding the temporary route segments (see Step 3 of Section 5 for more information). The time required to perform these additional operations is not much when compared to the time for finding the TSP solution within the ring.

In general, \( t_{js} > 4 \cdot t_0 \). This is because a pair of rings may overlap even if they are not adjacent (see Section 5 for more discussion) and hence such rings must be considered sequentially but not in parallel to find their TSP solutions. In that case, the number of steps required to join the clusters (i.e., Stage 3 of the developed approach) will be more that 4, where 4 is the chromatic number of the planar graph.

Based on the experimental results, \( t_{js} << t_0 \). For example, for Mona Lisa test case [12] with 100,000 points, \( t_i = 1 \) second and \( t_0 = 956 \) seconds. The run time for the complete solution is

\[ T = t_c + t_0 + 4 \cdot t_0 = 4 \cdot t_0 \]

More precisely,

\[ T > 5 \cdot t_0 \]

The maximum speedup for solving the entire TSP is achieved when the total number of available processors is equal to the total number of clusters formed. Optimization of the complete solution (i.e., Stage 4 of the developed approach) can also be performed in parallel. The main restriction for the optimization is that the local optimization areas (Section 6 contains more details about local optimization areas), where parallel optimization is performed, should not overlap with each other.

The speedup (\( g \)) of the developed approach is:
\[ g = \frac{T(N)}{T} \approx \frac{T(N)}{5 \cdot t_0} = \frac{T(N)}{5 \cdot T(n_0)} = \frac{T(N)}{5 \cdot T(N / k)}, \]

where \( T(N) \) is the run time to solve the entire TSP without clustering, \( k \) is the total number of clusters formed, \( N \) is the total number of points in \( P \) and \( t_0 = T(n_0) \) is the run time to solve TSP in a cluster with number of points \( n_0 = N/k \).

Consider a TSP algorithm that has no clusters and has time complexity \( O(n^m) \) for a data set with \( n \) points and for a nonnegative number \( m \). The speedup \( g \) that can be achieved by the developed clustering approach, when compared with the considered TSP algorithm is:

\[ g \approx \frac{N^m}{5 \left( \frac{N}{k} \right)^m} = \frac{k^m}{5}. \]

More precisely, \( g < \frac{k^m}{5} \).

The minimum number of processors required to achieve the maximum performance is determined by the number of clusters \( k \) that are formed. Because \( n_0 \ll N \) and \( k = N/n_0 \), a significant run time gain is achieved while obtaining the complete solution in parallel.

4. Decomposition of Points into Clusters

During the first stage of TSP solution process, the input set \( P \) of points is divided into clusters. Delaunay triangulation [13] is performed on \( P \) and the result is a set of triangular faces. By having Delaunay triangulation, no point in \( P \) will be inside the circumcircle of any triangle. The total length of the Delaunay triangulation edges is minimal. The computational complexity of Delaunay triangulation is \( O(N \log N) \) [15].

The clustering process consists of the following steps:

1. Form a set of Delaunay triangles by performing Delaunay triangulation of \( P \). Select any Delaunay triangle from the middle of \( P \). Call that triangle a “fragment” of the cluster to be formed.
2. (2a). Select all the Delaunay triangles that are adjacent to the “fragment” (i.e., “fragment” of the cluster to be formed) and are not already selected in the entire clustering process and then merge them with the fragment. (2b). If the number of points in the fragment is within 100 range of a specified number (for example 800) then call the fragment a cluster and go to Step 3 to create another possible cluster. Else go to Step 2a.
3. Select any Delaunay triangle that is not already selected in the entire clustering process and is adjacent to a cluster and call it a “fragment” (i.e., “fragment” of the cluster to be formed) and go to Step 2. Stop the clustering process if no such triangle is available.

Note: Step 2a constitutes one step of “wave propagation” while creating a cluster and this concept is used in Section 5 also.
Figure 1. a is Mona Lisa picture and b is the clustering of a.

The border of each cluster consists of Delaunay triangulation edges. In addition, the optimal solution for the TSP contains Delaunay triangulation edges. The main limitation parameter for clustering is the maximum number of points \((n_0)\) in each cluster. Increasing the value of this parameter will increase the run time of the whole process but decreasing the value of this parameter will reduce the quality of the solution. In other words, if the total number of clusters is small then the quality of the overall solution will improve but it requires more run time. Some tradeoff must be chosen to achieve a reasonable run time and solution quality.

Figure 2. Partial solutions for a fragment of clusters of Figure 1b. These solutions are not merged together.

At the end of the clusters’ formation stage, there may be some clusters with a relatively small number of points. Any cluster with a small number of points will be merged with an adjacent one with which it has more common edges. An additional parameter is introduced to set the lowest number of points in a cluster. For example, the clustering example based on Delaunay triangulation of Mona Lisa test [12] is shown in Figure 1. This example is characterized as follows: the size of \(P\) is 100,000 points, the number of clusters that are created is 122, approximate number of points in each cluster is 800, and the lowest number of points in a cluster is 600.
5. Finding and Merging the Partial Solutions

The second and third stages of the developed approach are explained here. In the second stage, TSP solutions are computed, by using the LKH algorithm [3, 5, 14], for each cluster that was formed in the first stage. These solutions are called partial solutions and they can be computed in parallel. Figure 2 contains the partial solutions for a fragment of clusters shown in Figure 1b.

The process of merging the partial solutions is performed in the third stage. Ring method [10] is used to merge the partial solutions. Basically, a ring represents a merging zone between a cluster and its neighboring clusters. The merging process consists of the following steps:

1. Building an initial "ring": Select any cluster at random. The border of the ring is the border points (i.e., vertices) that belong to the Delaunay triangular edges of the selected cluster. Each border edge of the ring has two incident border triangles, one is inside the cluster and another one is outside. All such incident border triangles will be considered for initial wave propagation (wave propagation is explained in Section 4).

2. Building the final ring: The final ring consists of all the points (indicated by \(d\) in Figure 3) that are obtained by performing wave propagation by starting from the border triangles mentioned in Step 1 and continue this wave propagation up to a given number of triangles, both inside and outside zones of the cluster. If the number of steps in wave propagation is large enough, the final ring covers the entire cluster in which the initial ring was started. The number of steps is determined by the number of points (for example, 800) that must belong to the “ring”. Note that the number of points is a parameter of the algorithm.

3. Temporarily replacing the route segments: If there is a route segment, whose first and last points (i.e., start and end vertices) only are on or in between the ring borders (i.e., no other points of that route segment are on or in between the ring borders) (\(b\) and \(c\) are such route segments in Figure 3) then temporarily replace that route segment with a single edge that connects the first and last points of that route segment. Let us call this edge “temporary single edge” and call that route segment “temporary route segment”.

4. Solving the TSP for the ring: Apply LKH on all the points that are on or in between the ring borders, in such a way that the LKH solution should include all temporary single edges, if any, that were formed in Step 3.

5. Replacing the temporary single edges: replace each temporary single edge by the temporary route segment that was replaced, in Step 3, by this temporary single edge.

6. Repeating the process to form another ring: Repeat all the previous steps by randomly selecting another cluster that is not already considered in the process. Stop the process if no more clusters are available for selection.

A cluster consists of multiple Delaunay triangles. Steps 1-5 are intended to merge a cluster (i.e., the partial solution of that cluster) with the adjacent clusters (i.e., the partial solutions of the adjacent clusters), by using the rings, to form a complete solution that passes through all points of \(P\).
Before start merging the partial solutions, the clusters will be divided into 4 different groups according to the four color theorem [11]. All the clusters belong to a single group are nonadjacent, that is, adjacent clusters belong to different groups. The parallel merging process, using rings, can be started by considering nonadjacent clusters only. This is because if adjacent clusters are considered for parallel merging then the rings that form around these adjacent clusters will overlap with each other and create problem as soon as they are ready to expand. Also, once the parallel merging process is started by considering nonadjacent clusters, each ring considers its adjacent clusters only, for merging. Since the adjacent clusters are divided into 4 different groups, at least 4 steps are needed to merge the clusters in parallel. This is because, at any given instance of time, a given cluster can be used by a single ring to merge that cluster with another cluster.

While merging the partial solutions in parallel, if the width of a ring is too large then two or more rings may overlap, even though the rings are formed by considering nonadjacent clusters. If the overlap occurs, one of the overlapping rings will be dropped from consideration. In that situation, the number of steps involved in the parallel merging process will be more than 4 because the ring that was dropped will be considered as part of an additional step.

![Figure 3. A sample ring.](image)

Figure 3. A sample ring. $a$ (i.e., the dashed lines) represents the ring borders, $b$ represents the internal route segments, $c$ represents the external route segments, $d$ represents the points, $e$ represents the border of the cluster that is considered, and $f$ represents the border between two adjacent clusters.

6. Optimization

The complete solution $M_0$ requires further optimization. It can be achieved by reducing the length of the route in Local Optimization Areas (LOAs). A LOA consists of a small number of points located in a close proximity, where a better TSP solution (than the one that was obtained by considering a large number of points) can
be obtained in a short amount of time.

**Figure 4.** Scanning around the cluster’s border. The gray areas, $S_i$ and $S_{i+1}$, are LOAs. The shaded portion, $F$, is the overlapping area between $S_i$ and $S_{i+1}$. The arrow represents the direction in which LOAs are selected for scanning.

The parameters are analyzed in more detail. Based on the experimental results obtained, it is better to choose $n_{sa}$ between 800-1200 points to obtain high quality solutions with a reasonable run time. In addition, experimental results show that if $n_{sa}$ is within 30-50% range of $n_{sa}$ then the developed approach provides good quality results.

**Figure 5.** Scanning along the route. The gray areas, $S_i$ and $S_{i+1}$, are LOAs. The shaded portion, $F$, is the overlapping area between $S_i$ and $S_{i+1}$.

To improve $M_{in}$, three different optimization strategies were developed as part of this research. In all these strategies, entire $P$ (i.e., the entire surface) will be covered by LOAs and the neighboring LOA locations must overlap with each other. The quality of the final solution depends on the following parameters: size (i.e., number of points) of LOA and the size of the overlapping area between adjacent LOAs. Increasing these sizes will improve the solution’s quality but it also increases the run time. Some tradeoff must be chosen between these parameters. The optimization strategies work as follows:

1. By scanning (i.e., thoroughly searching) the border area of each cluster for LOAs, as shown in Figure 4. To do this, the centers of the LOAs will be at the cluster’s border points. Once the center of a LOA is determined, other points in that LOA are determined by wave propagation around the
center (i.e., by performing the similar kind of wave propagation explained in Section 4 but by considering the “center of LOA” in place of “fragment” of Section 4).

2. By scanning along $M_0$ for LOAs, as shown in Figure 5. The centers of these LOAs will be on the points of $M_0$. The points in a LOA are determined by wave propagation around its center, as mentioned earlier.

3. By selecting LOAs in the form of geometrical areas such rectangles, as shown in Figure 6. The next LOA needs to be determined, once the scanning within the current LOA is over. In this figure, the size of each LOA is $a \times b$ square units. The second LOA is obtained by shifting $a/2$ units horizontally from the first one. By shifting $b/2$ units vertically from the second LOA, the third one is obtained, and so on. This kind of shifting gives a 50% overlap between successive LOAs.

The parameters and other details of these optimization strategies are:

1. The size of a LOA, indicated by $n_{oa}$.
2. The size of the overlapping area (i.e., number of common points or percentage of common points) between adjacent LOAs selected for scanning the surface. This size is indicated by $n_{oa}$.
3. The basic TSP algorithm, such as LKH [3, 5], that is used to find a solution within a LOA. Note that the “temporary single edge” and “temporary route segment” arrangements of Section 5 are used here but by considering the LOA in place of “ring” of Section 5.

![Figure 6. Scanning the surface. $S_{11}$ represents the initial LOA. $S_{12}$ represents the LOA obtained by a horizontal shift of $a/2$ units from $S_{11}$. $S_{21}$ represents the LOA obtained by a vertical shift of $b/2$ units from $S_{12}$. $S_{22}$ represents the LOA obtained by a horizontal shift of $a/2$ units from $S_{21}$. The area of each LOA is $a \times b$ square units.](image)

Different non-adjacent LOAs will be optimized in parallel in a single optimization strategy. The LOAs can be in any geometrical shape such as square, rectangle, and circle. Different LOAs in a given optimization strategy can be of different shapes but in this research, the shapes of all the LOAs used in an optimization strategy are the same. After applying a LOA (i.e., determining the LOA and applying TSP within that
LOA), or after applying more than one LOA in parallel, if the total length of the entire solution is less than that of the existing solution then the new solution found in the LOA is accepted for that area. The number of vertices’ groups of a planar graph that can be considered independently (i.e., to optimize in parallel, by using LOAs) is determined by the chromatic number of that planar graph and this chromatic number is 4. There are \(\frac{w}{4}\) optimization areas in each such group, where \(w\) is the number of LOAs that cover the entire surface \(P\). For all the clusters in a single group, the TSP can be performed simultaneously, by using LOAs, to optimize the solution.

The experimental results show that the quality of the optimization results will improve when the values of \(n_{sa}\) and \(n_{oa}\) are increased but this will increase the run time. An optimal trade-off is obtained by repeating the scanning, in a given optimization strategy, by changing the values of the parameters and the scanning order. For example, by scanning in counter-clockwise direction rather than clockwise direction in Figure 4 or a sequence of vertical shifts followed by a sequence of horizontal shifts rather than a horizontal shift followed by a vertical shift in Figure 6 and provide the resultant solution to the next optimization strategy to operate on. Experimental results also show that a considerable improvement in the quality is achieved by no more than 2-3 full cycles of optimization by using the same scanning strategy.

![Figure 7. Optimization areas (in gray) for specific critical surface zones.](image)

Apart from the previous three optimization strategies, a fourth one is also developed in this research. In this strategy, the LOAs do not cover the entire search space and they are formed at each specific critical zone, where three or more clusters are adjacent to each other, as shown in Figure 7. To optimize the solution in such critical zones, LOAs with a relatively more number of points will be considered. Typically, the number of points in each such LOA is in the range of 1200-1500.

Four different optimization strategies were specified. No two or more strategies can be applied in parallel because: (i) each strategy needs to be applied in parallel, at as many simultaneous locations as possible, on the search surface and (ii) an optimization strategy, except the first one, will operate on the TSP solution obtained from the previous strategy (the first optimization strategy operates on \(M_0\)).

However, a single optimization strategy can be applied in parallel at more than one location if these locations are not overlapping.
7. Experimental Results

The parallelization approach was investigated in terms of the quality of the solution and run time. The TSP test-cases were taken from the Concorde website [12]. Experiments were conducted on a PC with Athlon II X2 240 processor, 2.8 GHz CPU, and 2 GB RAM.

Table 1. Experimental results showing the tour quality of the developed approach.

<table>
<thead>
<tr>
<th>Test</th>
<th>Problem size (size of P)</th>
<th>Length of the complete solution</th>
<th>Length of the optimized complete solution</th>
<th>Length of the best known solution [12]</th>
<th>Tour quality %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mona Lisa 100K</td>
<td>100000</td>
<td>5 758 988</td>
<td>5 757 516</td>
<td>5 757 191</td>
<td>0.00565</td>
</tr>
<tr>
<td>Van gogh 120K</td>
<td>120000</td>
<td>6 545 620</td>
<td>6 544 127</td>
<td>6 543 622</td>
<td>0.00772</td>
</tr>
<tr>
<td>Venus 140K</td>
<td>140000</td>
<td>6 812 666</td>
<td>6 811 271</td>
<td>6 810 696</td>
<td>0.00844</td>
</tr>
<tr>
<td>Pareja 160K</td>
<td>160000</td>
<td>7 622 498</td>
<td>7 620 636</td>
<td>7 619 976</td>
<td>0.00866</td>
</tr>
<tr>
<td>Courbet 180K</td>
<td>180000</td>
<td>7 891 519</td>
<td>7 889 462</td>
<td>7 888 759</td>
<td>0.00891</td>
</tr>
<tr>
<td>Earring 200K</td>
<td>200000</td>
<td>8 174 726</td>
<td>8 174 507</td>
<td>8 171 712</td>
<td>0.03420</td>
</tr>
</tbody>
</table>

Table 2. Experimental results showing the required time for the developed approach vs. LKH.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Time (minutes) for the developed approach</th>
<th>Time (minutes) for LKH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mona Lisa 100K</td>
<td>121</td>
<td>446</td>
</tr>
<tr>
<td>Van gogh 120K</td>
<td>178</td>
<td>605</td>
</tr>
<tr>
<td>Venus 140K</td>
<td>213</td>
<td>980</td>
</tr>
<tr>
<td>Pareja 160K</td>
<td>229</td>
<td>1148</td>
</tr>
<tr>
<td>Courbet 180K</td>
<td>280</td>
<td>1435</td>
</tr>
<tr>
<td>Earring 200K</td>
<td>295</td>
<td>1722</td>
</tr>
</tbody>
</table>

The following parameters were used in the experiment:

- The number of points in a cluster is 800-900.
- Internal depth of a “ring” is 10 triangles (i.e., the ring covered 10 Delaunay triangles while propagating the wave inside the given cluster).
- External depth of a “ring” is 15 triangles (i.e., the ring covered 15 Delaunay triangles while propagating the wave outside the given cluster).
- $n_{sa} = 800$.
- $n_{oa} = 400$.

Note that the number of points in each cluster and $n_{sa}$ need not be the same, although they were the same in this experiment.

The experimental results are provided in Tables 1 and 2. Note that all four optimization strategies of Section 6 were used sequentially, i.e., one strategy after the other while deriving the results of these two tables.
In Table 1, the “Tour quality %” column shows the comparison, in terms of the solution quality, between the developed approach and the best known solutions from the Concorde website [12]. A significant feature of the developed approach is its computational complexity, which is almost linear, as shown in Figure 8. The computational complexity includes Delaunay triangulation also. These results demonstrate that the developed approach can be used for large-scale problems to get high quality solutions in a reasonable amount of time. For example, for a 200K points TSP, the solution quality of the developed approach is only 0.0342% less than that of the best known solution of Concorde website [12].

![Figure 8](image.png)

**Figure 8.** Time vs. size of the developed approach.

In Table 2, the run time of the developed approach is compared with that of LKH. “Time (minutes) for the developed approach” column shows the time required to find the optimized complete solution sequentially on our computer. “Time (minutes) for LKH column” shows the time required to perform LKH on our computer, without any decomposition.

One of the parameters of LKH is \( n/2 \), where \( n \) is the number of points in the TSP that we are solving using LKH. While deriving the values of Tables 1 and 2, the values of \( n \) were set as follows: \( n_0 \) while applying LKH for finding the partial solutions, \( n_{sa} \) while applying LKH on a LOA, and \( N \) while applying LKH to obtain “Time (minutes) for LKH” column of Table 2.

8. Conclusion

An important feature of the developed approach is the fact that the optimization algorithms can be executed in parallel. The approach can take advantage of parallel resources available in the current and future platforms. The well-known heuristic method LKH was used as the basic algorithm for solving TSP in each cluster, in each ring, and in each LOA. The developed approach needs to be evaluated on a multiprocessor system. If the number of available processors is limited then the number of clusters that were initially formed needs to be modified accordingly, so
that both these numbers match. The relation between the number of clusters and the quality of the solution needs to be investigated. Finally, the relation between the number of clusters and the required time (see Table 2) also needs to be investigated.

Acknowledgement

The authors are thankful to Dr. Keld Helsgaun of the Department of Computer Science, Roskilde University in Denmark, for the LKH source code [14].

References