An Almost Time Optimal Route Planning Method for Complex Manufacturing Topologies

Matthias Jörgl(✉), Hubert Gattringer, and Andreas Müller

Institute of Robotics, Johannes Kepler University Linz,
Altenbergerstr. 69, 4040 Linz, Austria
{matthias.joergl,hubert.gattringer,a.mueller}@jku.at
http://www.robotik.jku.at

Abstract. This paper focuses on time and distance optimal route planning for complex manufacturing topologies. The main idea of the proposed algorithm is to transform the manufacturing process into an undirected weighted graph, which can be treated as the well-known Traveling Salesman Problem (TSP). The optimal solution for the TSP is found with the help of Linear Integer Programming, which yields to Nondeterministic Polynomial-time (NP-hard). In order to overcome unpractical computation times for more complex tasks the optimal solution is approximated with the help of the Minimum Spanning Tree (MST) algorithm and alternatively with the Christofides algorithm. Simulation results and a comparison of the denoted methods are shown.

Keywords: Route planning · Integer linear programming · Traveling salesman problem · Christofides heuristic · Minimum spanning tree heuristic

1 Introduction

A main trend in today's automation industry is to reduce process time in order to increase production rate. In many applications like welding, punching and cutting, the execution sequence of the welds, holes etc. is crucial for the process time. Finding the optimal topology can be modeled as an undirected weighted graph and thereby treated in the same way as the well-known Traveling Salesman Problem (TSP). A main focus of this work is the choice of the weights of the graph, which have a strong influence on the quality of the solution.

The optimal solution of the TSP can be found with the help of Integer Linear Programming. While finding the optimal solution for the TSP is possible for small scaled problems, for more complex tasks the solution can only be approached. In this contribution the solution is approximated with the help of the Minimum Spanning Tree (MST) algorithm and with the help of the Christofides algorithm, which is an expansion of the MST. Both methods yield an approximation of the optimal solution in polynomial-time.

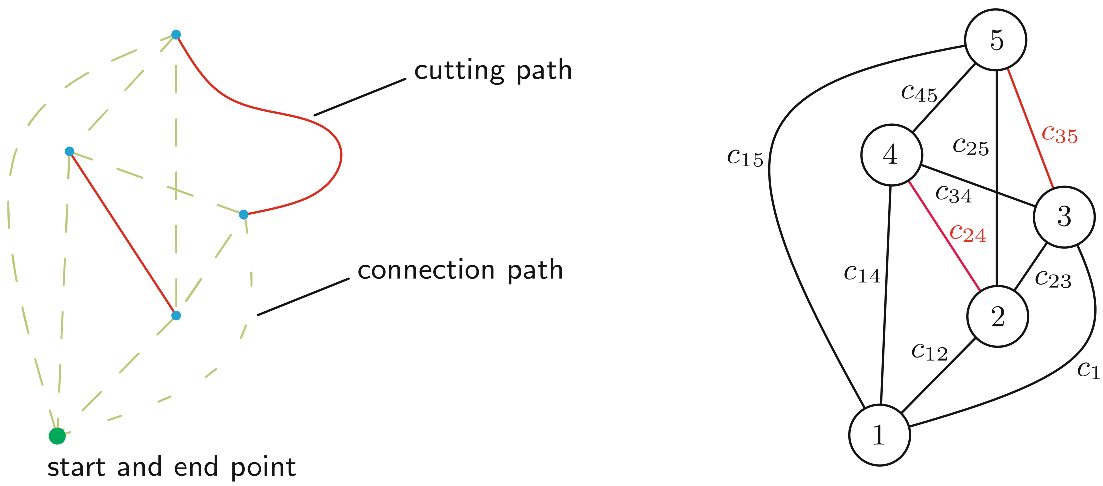
The considered paper is organized as follows: The problem formulation of a typically laser cutting process is dealt with in Sect. 2. Finding the optimal

R. Moreno-Díaz et al. (Eds.): EUROCAST 2015, LNCS 9520, pp. 673–680, 2015.
DOI: 10.1007/978-3-319-27340-2_83

topology with the help of Integer Linear Programming is shown in Sect. 3. In Sect. 4 the proposed heuristic methods will be discussed in more detail. Simulation results and a comparison to demonstrate the efficiency of all methods are shown in Sect. 5 and concluding remarks are given in Sect. 6.

2 Problem Formulation

The first part of finding an execution sequence is to transform the manufacturing process into an undirected weighted graph $G = (V, E)$, comprising a set V of vertices and a set E of edges with nonnegative edge costs $c\colon E \rightarrow \mathbb{R}^{+}$. Alternatively, a directed graph could also be used to manage that problem, but will lead to unpractical computation times. Now, finding an optimal route can be treated in the same way as the TSP. Figure 1 shows a typical layout for a laser cutting process (LCP) with the corresponding graph on the right side.



(a) Layout of a cutting process (b) Undirected weighted graph

Fig. 1. Layout of a cutting process with all possible connections with corresponding weighted graph

The trajectories of individual manufacturing processes are defined as basis splines and are connected via straight lines. A trajectory using basis splines is generally computed to

$$\mathbf{p}(u) = \sum_{j=0}^{n} \mathbf{d}_j \, N_j^d(u), \tag{1}$$

with number of control points n, control points $\mathbf{d}$, basis spline degree d and basis functions $N_j^d(u)$, see [2] for further details.

Due to the large number of edges, the weighting must be kept as simple as possible. Assuming that the execution direction of a contour has no influence on the process time, each start and each end point is assigned to a node of the graph. Edges, which connect contours, are weighted in relation to the process time of the machine using the weighting function

$$c_{ij} = \frac{|x_i - x_j|}{V_{x,max}} + \frac{|y_i - y_j|}{V_{y,max}}, \tag{2}$$

with the maximum axis velocities $V_{x,max}$ and $V_{y,max}$ of a two axis linear robot (e.g. a laser cutting machine). The parameters x_i and y_i contain the position of each node of the graph. Contrary, the whole path length of the process can also be reduced with

$$c_{ij} = \sqrt{|x_i - x_j|^2 + |y_i - y_j|^2}, \tag{3}$$

which increases the life time of the machine by using only distances for weighting.

3 Integer Linear Program Formulation

To achieve an optimal route, the TSP can be formulated into an Integer Linear Program

$$\begin{aligned}
&\min_{\mathbf{x}} \sum_{i\in V} \sum_{j\in V\setminus\{i\}} c_{ij}\, x_{ij} \\
\text{s.t.} \quad & \sum_{j\in V\setminus\{i\}} x_{ij} = 1 \\
& \sum_{i\in V\setminus\{j\}} x_{ji} = 1 \\
& u_i \in \mathbf{Z} \\
& u_i - u_j + n\, x_{ij} \le n-1,\ 1 \le i \ne j \le n \\
& 0 \le x_{ij} \le 1,\ x_{ij}\, \text{integer},
\end{aligned}$$

with the integer variables

$$x_{ij} = \begin{cases} 1 & \text{if}\, arc(i,i)\, \text{is in the tour} \\ 0 & \text{otherwise.} \end{cases}$$

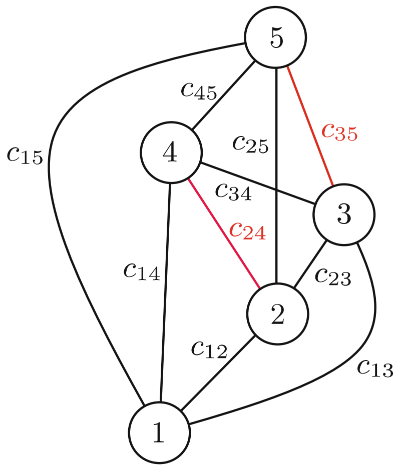
The first set of equalities guarantees that each node of the graph can be arrived from exactly one other node, and the second set of equalities guarantees that from each node only one departure to exactly one other node is possible. The last constraints enforce that there is only a single tour covering all nodes, see e.g. [1]. To ensure that every contour will be executed by the machine, the edge weights between start and end point of each contour have to be treated separately. A functional approach is to weight them negative in contrast to the contour connections. To obtain a solution, the Integer Linear Program was solved with a standard solver.

4 Heuristic Methods

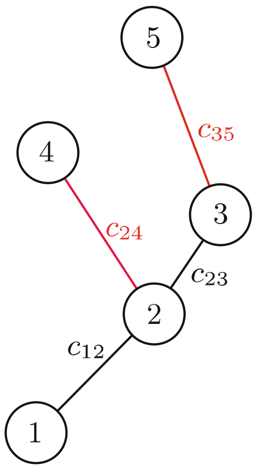
There is a high computational burden when using Linear Programming for solving the TSP for complex tasks. Hence, in this section, two heuristic methods of finding an approximation of the optimal tour are shown. Beside the great number of heuristic methods, the MST - heuristic and the Christofides heuristic are outlined and compared in this paper.

4.1 MST - Heuristic

The first step of the MST - heuristic is to transform the manufacturing process into an undirected weighted graph introduced in Sect. 2 in order to get a graph $G = (V, E)$. The next step is to construct a minimum spanning tree T. A minimum spanning tree is a subgraph T of G, that contains all the vertices and is a tree with weight less than on equal to the weight of every other tree. In Fig. 2 the corresponding minimum spanning tree of the undirected weighted graph is pictured.



(a) Undirected weighted graph



(b) Minimum Spanning tree

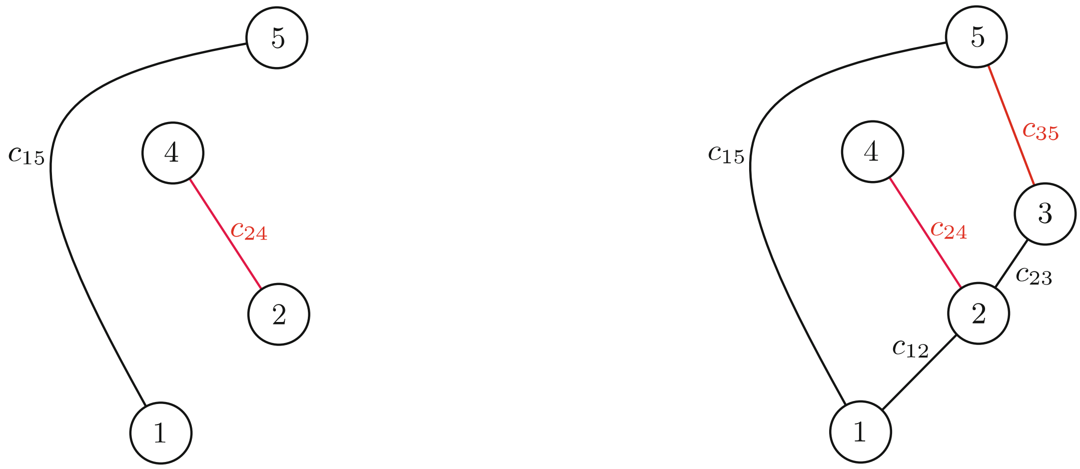
Fig. 2. Minimum spanning tree of the undirected weighted graph

To achieve the minimum spanning tree of the graph G, the Prims's algorithm is used. The Prims's algorithm initialize the tree T with a single vertex, chosen arbitrarily from the graph G. After initialization, the algorithm grows the tree by one edge, which is not yet in the tree and which has minimum weight. After transforming it to the tree T, the algorithm repeats this step until all vertices are in the tree. For detailed information according this algorithm we refer to [5]. Finding a minimum spanning tree with the help of Prim's method yields to a time complexity of $\mathcal{O}(n^2 log_2(n))$, which is the most consuming one. To ensure that the tree T includes all cutting paths of the manufacturing process, the costs of these paths must be smaller than the smallest contour connection. Finally, our goal is to find an Eulerian tour in T, which proceeds to a Hamiltonian tour. A tour is said to be Eulerian in a graph G, if every edge of E is traversed exactly once. Whereupon a Hamiltonian tour is a tour containing each vertex exactly once.

In summary, the output of the whole algorithm provides an approximation of the optimal route for a complex distribution of contours (e.g. a cutting process), where each contour is processed exactly once.

4.2 Christofides Heuristic

The MST - heuristic can be extended with the help of perfect minimum matching, which yields to the Christofides algorithm. A perfect minimum matching M in G is a set of pairwise edges with odd degree, which matches all vertices of the graph with minimal total cost, see [6]. After calculating the perfect minimal matching M, the algorithm unite the minimal spanning tree T with the set M, see Fig. 3. Hence, impasses can be identificated and are connected together in such a manner that the sum of these paths has minimal total process time (path length). This part is the most time consuming one, having a time complexity of $\mathcal{O}(n^3)$.



(a) Perfect minimum matching M of edges with odd degree

(b) $T \cup M$

Fig. 3. The perfect minimal matching M united with the minimal spanning tree T

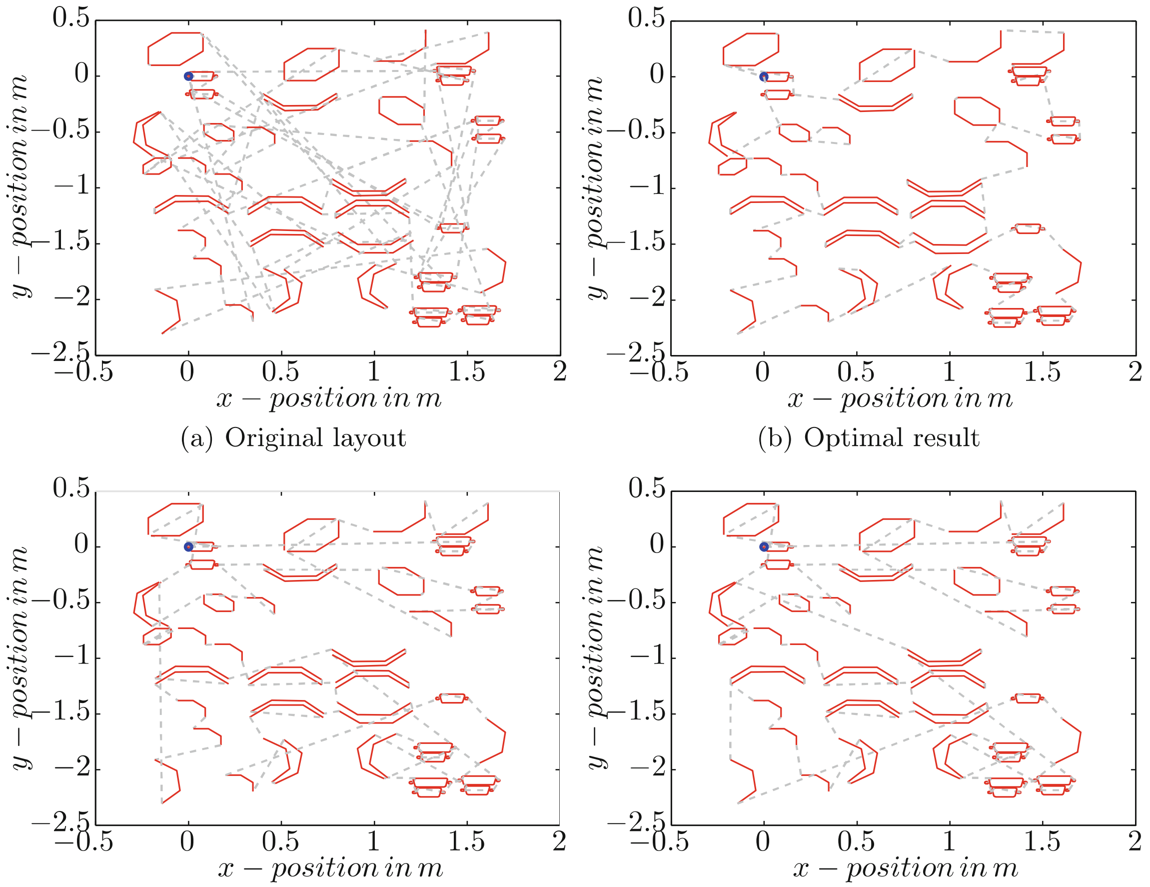
Due to the fact, that the costs of the cutting paths are smaller than the smallest contour connection, the triangle inequality

$$c(u, v) \leq c(u, w) + c(w, v),\ \forall u,\ v,\ w\ \in V, \tag{5}$$

of the graph G is not fulfilled. Thereby, in this case the benefit of the classical Christofides algorithm does not apply the estimation of the upper limit of the heuristic solution, which is at worst 50 % higher than the optimal ones (see [3,4] for more details). The last step of the Christofides algorithm is also to find a Hamiltonian tour as described in Sect. 4.1.

5 Results

In order to show the efficiency of the proposed algorithms, simulation results are given. Figure 4 pictures a comparison between the unsorted and sorted execution sequences of a typical laser cutting process, where the costs are in relation to the path lengths. Note, that the presented manufacturing process consists of 80 cutting paths. The solid profiles in Fig. 4 shows the corresponding control polygons of the cutting paths and the dashed lines represent the contour connections.



(a) Original layout (b) Optimal result

(c) MST - Heuristic (d) Christofides Heuristic

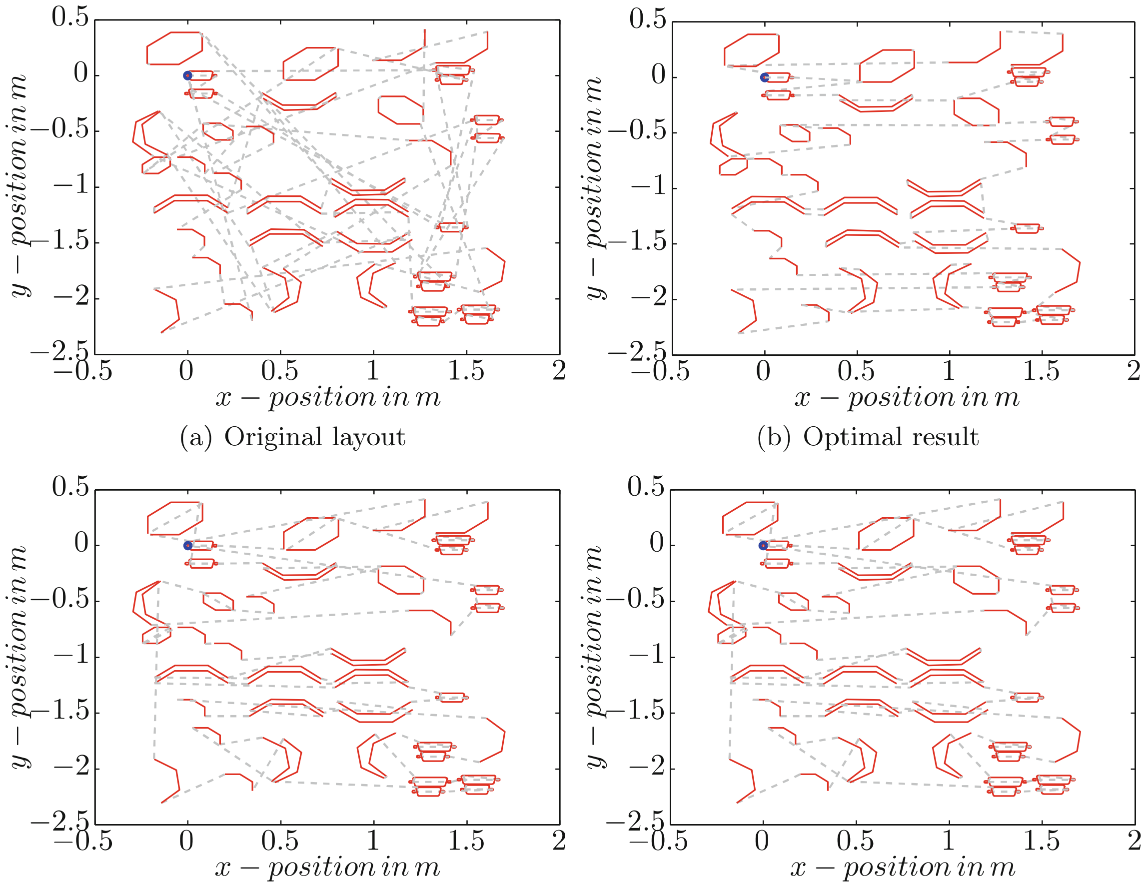
Fig. 4. Results of optimization (minimal distance)

Obviously, the whole path length of the process has been drastically reduced with the help of all algorithms. It turns out, that both heuristic methods produce similar results and reduce the whole path length by about 35 %, with a maximum computation time of 1.58 s on a standard PC. Whereat the optimal solution already leads to an unpractical computation time of 50.1 s. In this case, the Christofides algorithm has no advantages over the MST - heuristic. Instead the computation time is about 8 % higher. A summary of the results are presented in Table 1.

Contrary to the previous plots, Fig. 5 presents the results of the time optimal approach, where all edges are weighted in relation to the process time of the

Table 1. Results - distance optimization

Algorithm	Computation time	Process time	Total time	Process distance
Linear programming	50.10 s	72.66 s	122.76 s	34.03 m
MST	1.45 s	84.80 s	86.25 s	44.86 m
Christofides	1.58 s	83.69 s	85.27 s	44.06 m
Random layout	–	132.75 s	132.75 s	74.26 m



(a) Original layout (b) Optimal result

(c) MST - Heuristic (d) Christofides Heuristic

Fig. 5. Results of optimization (minimal time)

machine. In order to evaluate the effect of different axis velocities, $V_{x,max}$ is five times higher than $V_{y,max}$. Therefore, it is clearly evident that the main movement of all methods results along the x -axis. The results of the simulation are summarized in the Table 2.

An interesting observation is that both heuristic methods lead to the same approximation of the optimal route. This means, that the perfect minimum matching of the Christofides heuristic has no influence to the solution. Here, the heuristic methods reduce the total process time by about 59 %, whereat the optimal solution also leads to unpractical computation time.

Table 2. Results - time optimization

Algorithm	Computation time	Process time	Total time	Process distance
Linear programming	59.30 s	68.84 s	128.14 s	41.30 m
MST	1.47 s	79.07 s	80.54 s	49.05 m
Christofides	1.61 s	79.07 s	80.68 s	49.05 m
Random layout	–	132.75 s	132.75 s	74.26 m

6 Conclusion

In this paper, two heuristic methods of finding an almost time or distance optimal route for complex manufacturing topologies have been developed. Comparisons to the optimal solutions of a random layout are drawn. There, one can see obviously, that the presented heuristic methods lead to practical computation time on the one hand as well as they reduce the whole path length by about 35 % and the total process time by about 59 %.

Acknowledgments. This work has been supported by the Austrian COMET-K2 program of the Linz Center of Mechatronics (LCM), and was funded by the Austrian federal government and the federal state of Upper Austria.

References

1. Dantzig, G.B., Fulkerson, D.R., Johnson, S.M.: Solution of a large-scale traveling-salesman problem. Oper. Res. **2**, 393–410 (1954)
2. de Boor, C.: A Practical Guide to Splines. Mathematics of Computation. Springer, New York (1978)
3. Christofides, N.: Worst-case analysis of a new heuristic for the travelling salesman problem. Research Report, Carnegie-Mellon University, Pittsburgh (1976)
4. Christofides, N., Mingozzi, A., Toth, P.: Exact algorithms for the vehicle routing problem, based on spanning tree and shortest path relaxations. Math. Program. **20**(1), 255–282 (1981)
5. Prim, R.C.: Shortest connecting networks and some generalizations. Bell Syst. Tech. J. **36**, 1389–1401 (1957)
6. Gabow, H.N.: A scaling algorithm for weighted matching on general graphs. Ph.D. thesis, Stanford University (1974)