**Applied Algorithmics Group**

aa-01-2005 August 31, 2005

# Hybrid Evolutionary Approaches to CNC Drill Route Optimization

**Submission to International Conference on Computational Intelligence for Modelling Control and Automation – CIMCA'2005 November 28–30, 2005, Vienna, Austria**

Simon Sigl
ssigl@cosy.sbg.ac.at

Helmut A. Mayer
helmut@cosy.sbg.ac.at

Correspondence to:

Helmut Mayer
Universität Salzburg
Fachbereich Scientific Computing
Jakob–Haringer–Straße 2
A–5020 Salzburg
AUSTRIA

Telephone: +43-662-8044-6315
FAX: +43-662-8044-611

# Hybrid Evolutionary Approaches to CNC Drill Route Optimization

**Abstract**

*A problem in computer numeric control (CNC) manufacturing is the minimization of the distance a drilling tool has to move in order to drill holes at given locations. After introducing the problem being an instance of the traveling salesman problem (TSP) we describe the Route Optimizer RO3 based on an evolutionary algorithm (EA). In experiments with real–world problem instances we were able to improve the results of RO3 by hybridization with the 2–opt heuristic to route lengths being 6% above the optimum. In its hybrid version RO3 achieves machine time savings of about 10% compared to visual optimization by a human expert.*

## 1 Introduction

When constructing playground equipment, a certain manufacturing step requires the drilling of holes into wooden and plastic parts. The order in which the holes are to be drilled influences the time this process needs on a CNC machine[1], and should therefore be optimized. As the drilling device has to be steered to the location of each hole exactly once, the problem of drill route optimization is an instance of the (symmetric, single–objective, euclidean) Traveling Salesman Problem (TSP).

Solving the TSP means finding the route with minimal total cost, which is known to be an NP–hard problem. The starting point for the presented industrial application was a heuristic that reasonably reduces the cost, i.e., the tool path length. Since Evolutionary Algorithms (EAs) [1] have proven to be a useful approach to find (sub)optimal tours for the TSP, we devised an EA that further decreases tool path lengths, and therefore, increases productivity.

We were able to increase the effectiveness of our EA by applying the local search heuristic 2–opt to tours generated by evolution. Since the *Concorde TSP Solver*[2] is able to generate optimal solutions for our problem instances, we can objectively assess the quality of the solutions found by our approach. Note that Concorde may be used freely only for academic purposes.

This work is structured as follows: in the next section we introduce the real–world problem at hand, and discuss its constraints in detail. The problem description is followed by the intrinsics of our approach to solve the problem with an EA. Subsequently, we assess the quality of solutions using real-world data from the problem domain. Finally, we draw conclusions and identify possible improvements to be investigated in the future.

[1]This project is done in cooperation with *Ludwig Schröckeneder Spielplatzeinrichtungen*, Siggerwiesen 39, 5101 Bergheim, Austria

[2]http://www.tsp.gatech.edu/concorde.html

## 2 Drill Route Optimization

The manufacturing company *Ludwig Schröckeneder Spielplatzeinrichtungen* uses a multi–functional CNC device capable of processing wooden and plastic material to produce various parts of equipment for children's playgrounds. Processing these parts includes the drilling of holes located on a plane. The total distance a drilling tool has to be moved in order to drill holes at specific locations is directly proportional to the time consumed by this task. As a consequence, we want to optimize the route taken by the drilling device in order to speed up the production process.

### 2.1 Problem Description

The CNC programs for the presented application can typically be divided into two parts: drilling and shaping. In the first phase several holes (typically 100–150) are drilled in order to be able to fix the raw material to a base board, so that the parts cannot move during shaping in phase two. Here, we focus on optimization of the drill task.

The CNC programs for drilling are developed on a workstation using a special 2D–CAD design tool that generates CNC sources. These programs are then transmitted to a *Siemens Sinumerik 810M* control system that performs the controlling of machine actions according to the programmed commands. Initially, the drilling sequence specified in a program is pre–optimized by the CNC programmer by visual inspection of the problem instance. Since this optimization step, which generates routes considerably superior to a random sequence, is only based on human experience, there is room for improvements.

For a seamless integration into the production cycle a route optimizer running on the workstation parses the CNC program, and, after optimizing the drilling sequence, rewrites the program that then contains the altered sequence. This process is illustrated in Figure 1.

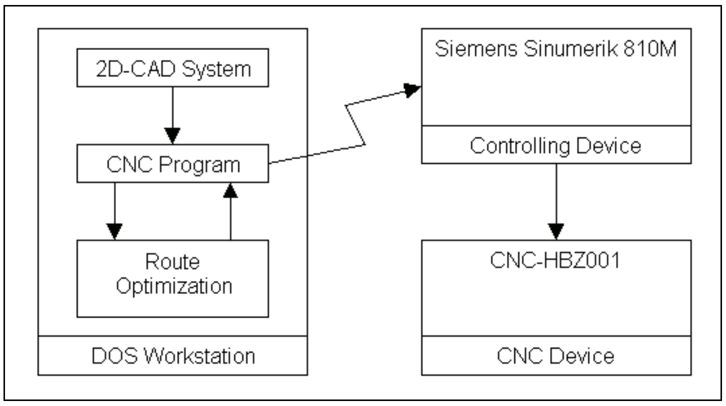

Figure 1: Environment of the route optimizer.

At the beginning of the drilling process a tool change has to be made in order to provide the machine with the appropriate drill tool. Such tool changes are performed at fixed locations. During drilling the CNC program (Figure 2) may command a tool-change in order to drill holes with different diameters. This divides the holes to be drilled into subsets with common diameters (from today's view, one or two sets are the rule for most programs), requiring tool changes for each subset. We call these subsets *drilling sections*, which introduce a special constraint: while optimizing the drill sequence, we are not allowed to move a hole from one drilling section to another, i.e., a sequence may not traverse the borders imposed by tool changes. This constraint reduces the complexity of the problem, as it narrows the search space by dividing the problem into smaller parts. At the end of the drill process another tool change takes place to equip the machine with a milling head in order to shape the part.

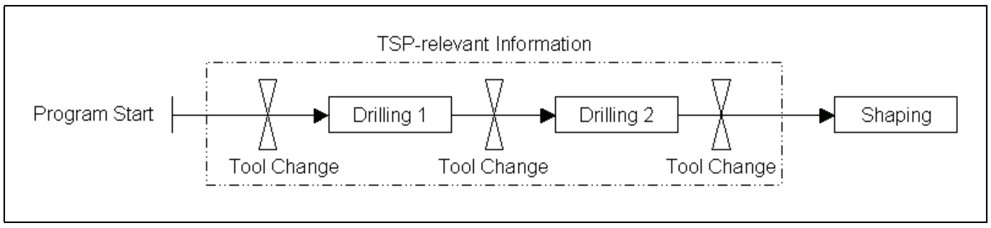

Figure 2: Typical structure of the CNC program.

## 2.2 Previous Approaches

Preceding this project, two attempts to solve the drill route optimization problem have been made. The first program RO1 (Route Optimizer Version 1) used a heuristic, which proved to be of of little benefit. Its successor RO2 is an implementation of a basic EA that is more effective than RO1. Its results represent a base line for performance evaluation of the system described in this work.

RO2 is basically an implementation of a (1,1)-Evolution Strategy, meaning that there is only one individual subjected to evolutionary processes. A mutation is accepted if it improves the quality of the solution, otherwise it is discarded. This approach has some known potential drawbacks such as narrow sampling of the search space (due to the population size of 1), the absence of a recombination operator, and the inherent tendency to converge to a local maximum (even more so, when starting with a good heuristic solution (Section 2.1)).

Nevertheless, RO2 was able to perform route length reductions up to 15% compared to the initial route generated by the CNC programmer. In order to overcome the shortcomings mentioned above, and to further improve the quality of optimization we designed RO3 presented in the following section.

## 3 The Route Optimizer RO3

RO3 is, like its predecessor RO2, an implementation of an Evolutionary Algorithm. When approaching a TSP with EAs, first of all, we have to define genotype encoding and fitness function. Also, we have to select (possibly problem–specific) genetic operators, and set a variety of parameters, e.g., operator probabilities, population size, initial population, and selection method.

### 3.1 Route Encoding and Fitness

A TSP round–trip (or tour) can be nicely represented by a permutation of the locations to be visited. A parsimonious encoding of such a permutation is an array of references to the locations. The fitness of the solution (phenotype represented by the permutation genotype) is simply the length of the corresponding tour (in millimeters), which we intend to minimize.

### 3.2 Genetic Operators

The obvious problem with naively altering a route is that it may be transformed to an invalid one. For example, the route $1 \rightarrow 2 \rightarrow 3$ randomly changed at the second locus may result in $1 \rightarrow 3 \rightarrow 3$ violating the problem constraints. Thus, we would like to employ a mutation operator always generating valid routes, i.e., permutations.

A well–known method to implement such a mutation operator are transpositions. Given the tour $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$, we may swap locations 3 and 5 and get the valid tour $1 \rightarrow 2 \rightarrow 5 \rightarrow 4 \rightarrow 3$. Provably [2], all possible permutations can be constructed by performing a finite number of transpositions, therefore it is theoretically possible to sample the whole search space solely by applying this mutation operator.

The concept of crossover is promising in the context of the TSP, as recombining good subroutes may effectively lead to better solutions. Two popular crossover operators for the TSP are *Partially Mapped Crossover* (PMX) [3] and *Ordered Crossover* (OX) [4].

Basically, PMX and OX implement 2–point crossover exchanging genetic material located inbetween the two random crossover locations (crossover section), which corresponds to direct inheritance of subroutes. As this operation may produce invalid routes, i.e., eliminate or duplicate cities (points), these inconsistencies have to be repaired. With OX the relative order of genes outside the crossover section is reconstructed, while PMX swaps a minimal number of genes outside the crossover section with genes inside.

We compared these operators using a representative problem instance with 123 holes and 2 drilling sections. OX, PMX, and a hybrid operator (application of OX or PMX with a probability of 0.5) were used in 50 evolutionary runs with 8,000 generations each. As can be seen in Table 1, the most promising results have been achieved applying OX only (hence, used for RO3), however, the differences are small.

| Crossover Operator | Average Fitness |
|---|---|
| PMX | 29,611 |
| OX | 29,405 |
| Hybrid | 29,597 |

Table 1: Comparison of crossover operators (averaged on 50 runs).

In order to determine "good" mutation and crossover rates, we ran some preliminary experiments with different rates, and averaged the measured results on 25 evolution runs for each setting. At a value of 0.5 for both rates, the best results have been recorded, al-though again, the differences were small. The averaged development of fitness is displayed in Figure 3 along with the baseline heuristic.

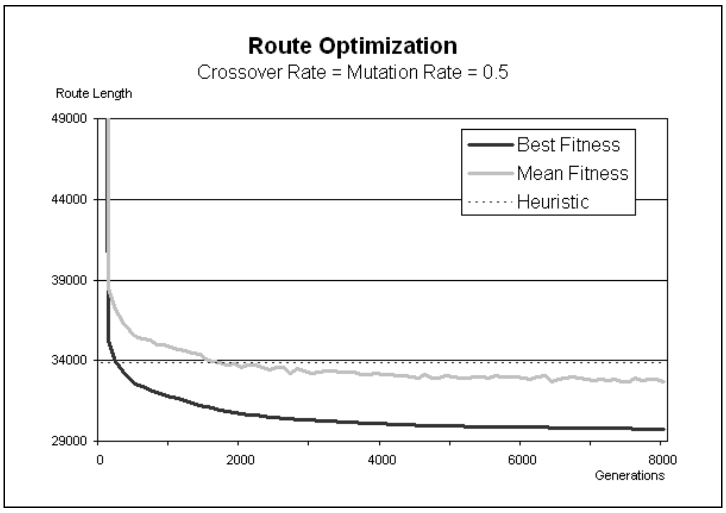

Figure 3: RO3 fitness statistics (averaged on 25 runs).

In order to promote the potential to escape from local optima, we introduced *Catastrophes* in RO3. If, within 1,000 generations, no new best individual has been found, the population is deemed to be captured at a local optimum. Consequently, the gene pool of the population is subjected to severe mutation (in the same way the start population is generated (Section 3.4)). It is assumed, that a similar development takes place on earth about every several hundred thousand years [5]. Though, the addition of catastrophes did in general not assist in improving the best results, it improved the lower quality bound of evolved routes, which in turn increased the mean fitness of the best routes found in a number of evolution runs.

## 3.3 Population Size

The choice of the population size is a trade–off between processing time and search space sampling. According to Julstrom (1996), the population size should be roughly equal to $log_2 N$ with $N$ being the number of tours of the TSP instance in order to arrive at the optimal solution with high probability [6]. The results were derived for a basic GA using crossover as the only genetic operator. E.g., above estimation gives a population size of 525 for a TSP instance with 100 cities. As we also employed mutation, we assumed a number of 525 to be an upper bound estimate, and chose the population size $n$ to be 300.

## 3.4 Initial Population

Contrary to the usual practice of generating a random start population, we incorporated the human knowledge of the CNC programmer (Section 2.1) into the system by using his solution as a base for the initial population. Each individual of the start population is a mutation (Section 3.2) of the heuristic solution suggested by the human. However, the number of mutations is not a consequence of the mutation rate, but is randomly chosen from the interval $[10, 1000]$. As a result, some individuals are similar to the heuristic solution, while others differ considerably. This method should enable good exploration of the search space, but still exploit the problem knowledge introduced by the heuristic solution.

### 3.5 Selection Method

We employed a selection method similar to binary tournament selection by drawing two random individuals from the population and replacing the individual of lower fitness by the winner. This procedure is repeated $n/2$ times, and arguably exerts a higher selection pressure than standard binary tournament selection (without replacement). However, we do not claim this selection variant to be superior to the latter.

## 4 Experimental Results

In order to assess the quality of the solutions found by RO3, we compared RO3 to RO2, and to the optimal solutions gathered from the *Concorde* TSP solver freely available for academic research use. Also, we studied the influence of applying the 2–opt procedure to the evolved solutions.

### 4.1 Route Length Comparisons

The results presented here are mean route lengths of ten parts (ten evolution runs each) with different characteristics. This set of problem instances was chosen to cover a wide spectrum of typical problem features, namely, 68–123 holes, 1–2 drill sections, and holes evenly distributed or closely arranged in groups. Evolution runs have been terminated when no more improvement in tour length was noticeable, typically, a single run took 1–2 minutes (about 15–20,000 generations).

All route lengths are expressed in millimeters, which cannot be directly related to machine time, as the total process time (usually 8–12 minutes per part) is the sum of the time needed to position the tool and to drill the holes. We estimate the moving time/drilling time ratio to be about 0.5, therefore, the route length reduction has to be converted appropriately to machine time saving. E.g., a length reduction of 10% corresponds to a time saving of 6.66%.

In Table 2 the evolved route lengths generated by RO2 and RO3 are compared to the optimal length. The fitness standard deviation is about 0.01% (of route length) for RO2 and 0.015% for RO3.

Inspecting Table 2, we notice that on average, the heuristic solution, i.e. the route constructed by human experience, is about 128% the length of the optimal route. Despite the fact that the CNC programmer constructs this route by visual inspection of the problem instance, and has experience in doing so, the route length can still be reduced considerably by the evolutionary approaches RO2 and RO3.

Considering the fact that RO2, as discussed in Section 2.2, has some flaws, it performs

| Holes | Sections | Optimum | Heuristic | % | RO2 | % | RO3 | % |
|---|---|---|---|---|---|---|---|---|
| 105 | 1 | 17776 | 24448 | 138 | 21257 | 120 | 20809 | 117 |
| 79 | 1 | 15558 | 24310 | 156 | 20529 | 132 | 19562 | 126 |
| 123 | 2 | 24818 | 27701 | 112 | 26999 | 109 | 26289 | 106 |
| 123 | 2 | 24822 | 33824 | 136 | 30635 | 123 | 29366 | 118 |
| 116 | 2 | 26641 | 32977 | 124 | 32610 | 122 | 30446 | 114 |
| 84 | 1 | 14745 | 20326 | 138 | 18804 | 128 | 17749 | 120 |
| 108 | 1 | 16044 | 19111 | 119 | 18043 | 112 | 17930 | 112 |
| 68 | 1 | 13843 | 15891 | 115 | 15844 | 114 | 15473 | 112 |
| 90 | 1 | 16067 | 19929 | 124 | 19792 | 123 | 19526 | 122 |
| Optimum % | | 100% | 128% | | 120% | | 116% | |

Table 2: Optimal route lengths (Optimum), human solution (Heuristic), and those evolved by RO2 and RO3 (averaged on ten runs).

surprisingly well. On average, it generates route lengths of about 120% of the optimal route. Route length reduction achieved by RO3 is more effective and produces, on average, values that are 16% above the global optimum. Expressed in machine time the improvement of RO3 over RO2 is about half a minute (over heuristic a good minute) of typically ten minutes drilling time. Thus, RO3 is currently in use at the manufacturing site.

As above results still leave room for improvement, we investigated the impact of heuristics to further hone the evolutionary solutions. EAs incorporating heuristics belong to the class of *hybrid* EAs, hence in a strict sense RO3 is already a hybrid EA, as it uses heuristic solutions for the construction of the start generation (Section 3.4).

## 4.2 The 2–opt Heuristic

A popular local search heuristic for the TSP is 2–opt. It is used frequently when tackling the TSP with hybrid EAs, and has been found to contribute considerably to the solution quality [7]. A TSP route is called *k–optimal*, if after elimination of any $k$ edges, none of all correct reconnections of the $k$ subpaths does decrease the route length. The 2–opt heuristic, then, iteratively performs eliminations and reconnections of two edges (2–opt moves) until route length can no longer be reduced.

In Table 3 we present the resulting route lengths, when applying 2–opt to each best evolutionary solution of all the ten problem instances.

On average, the best routes found by RO3 are about 113% of the length of the optimal route, which after applying 2–opt were only 6% above the minimal length. The distance between the initial heuristic solution and the optimum has been reduced by 83% by applying RO3 and 2–opt. Due to this considerable improvement, we will integrate 2–opt as an additional optimization step in the next release of RO3. The differences between plain RO3 and the algorithm with additional 2–opt can be seen in typical routes depicted in Figure 4.

| Holes | Sections | Optimum | Best RO3 | % | Best RO3 + 2–opt | % |
|---|---|---|---|---|---|---|
| 105 | 1 | 17776 | 20623 | 116 | 18535 | 104 |
| 79 | 1 | 15558 | 18851 | 121 | 17192 | 111 |
| 123 | 2 | 24818 | 26099 | 105 | 25885 | 104 |
| 123 | 2 | 24822 | 28604 | 115 | 25922 | 104 |
| 116 | 2 | 26641 | 30025 | 113 | 28095 | 105 |
| 84 | 1 | 14745 | 17330 | 118 | 15756 | 107 |
| 108 | 1 | 16044 | 17740 | 111 | 17319 | 108 |
| 68 | 1 | 13843 | 15046 | 109 | 14081 | 102 |
| 90 | 1 | 16067 | 18896 | 118 | 17235 | 107 |
| Optimum % | | 100% | 113% | | 106% | |

Table 3: Local optimization of best evolved solutions using 2–opt.

## 5 Conclusions

We presented a real–world problem from the manufacturing industry, scheduling of drilling processes on a CNC machine, being an instance of the traveling salesman problem. We designed the route optimizer RO3 for CNC programs based on an evolutionary algorithm capable of generating solutions with route lengths of 113% of the length of the optimal tour, which corresponds to about one minute of machine time savings per processed part, when compared to a heuristic solution by a human expert. The presented optimizer is currently employed to speed up the drill process for parts assembled to playground equipment.

Assessing the quality of the generated solutions, we found that additional local optimization utilizing the 2–opt heuristic can further decrease the evolved route lengths. Applying 2-opt to the best evolved solutions we arrive at lengths being 6% above the optimum, which we were able to determine using the *Concorde* TSP solver (freely usable for academic purposes only).

In this work we used 2–opt only as a final optimization step, but tighter integration with the evolutionary process can be accomplished easily (at some computational cost). Further improvements of the presented route optimizer RO3 could be gained by more sophisticated creation of the start generation and genetic operators. We believe that introducing a broader genetic diversity into the initial generation may lead to even better results. Candidate methods for the automatized generation of good initial routes are greedy algorithms, or *Christofides'* heuristic [8] (post–processed by 2-opt). Promising alternatives for the genetic operators currently in use are edge assembly crossover [9] or

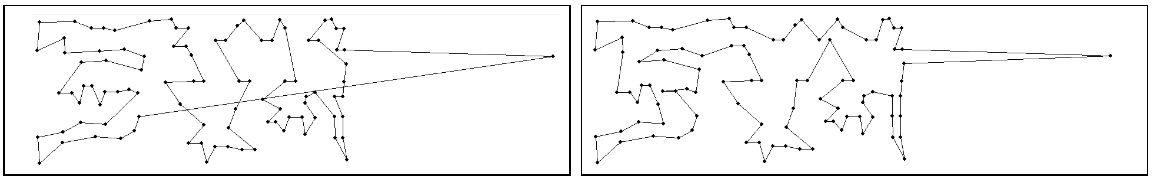

Figure 4: Drill route optimized by RO3 (left) and after additional 2–opt heuristic (right).

displacement [10] as a mutation operator.

## Acknowledgements

We would like to thank Ludwig Schröckeneder[3] for supporting our research. All real–world experiments have been carried out on the prototype CNC machine *CNC–HBZ001* developed by the *Werkschulheim Felbertal*[4], Ebenau, Salzburg, Austria.

## References

[1] Thomas Bäck, *Evolutionary Algorithms in Theory and Practice*, 1996.

[2] J. Linhart, *Diskrete Mathematik*, Abakus, 1996.

[3] D. E. Goldberg & R. Lingle, Alleles, Loci, and the Traveling Salesman Problem, *Proc. of the International Conference on Genetic Algorithms and Their Applications*, 154-159, 1985.

[4] L. Davis, Applying adaptive algorithms to epistatic domains, *Proceedings of IJCAI*, 162-164, 1985.

[5] H. v. Ditfurth, *Kinder des Weltalls*, dtv, 1982.

[6] B. Julstrom, Population Size in GAs for TSP, *Proc. of the Second Nordic Workshop on Genetic Algorithms and their Applications*, 3-13, 1996.

[7] J.-P. Watson, et al., The Traveling Salesrep Problem, Edge Assembly Crossover, and 2-opt, *Parallel Problem Solving from Nature V*, Springer, 1998.

[8] Nicos Christofides, *Worst-case analysis of a new heuristic for the traveling salesman problem*, Technical Report CS-93-13, Carnegie Mellon University, Pittsburgh, 1976.

[9] Yuichi Nagata & Shigenobu Kobayashi, Edge assembly crossover: A high-power genetic algorithm for the traveling salesman problem, *Proceedings of the 7th International Conference on Genetic Algorithms*, 450–457, Morgan Kaufmann, 1997.

[10] Z.Michalewicz, *Genetic algorithms + data structures = evolution programs*, Springer, 1992.

[3]http://www.schroeckeneder.at/start.htm
[4]http://www.werkschulheim.at