MIP Optimization of the Lithium Logistics problem

Christian Fremuth-Paeger, Andreas Hefele, Dieter Jungnickel*
Matthias Leclerc†

Abstract

We investigate the optimization of a real world logistics problem which is concerned with shipping a dangerous chemical substance in various degrees of refinement to several locations and customers. Transport frequencies, inventories and container flows have to be optimized. On the one hand, we discuss the mathematical structure of our problem (one result being its NP-completeness), and on the other hand, we describe our practical approach which achieves nearly optimal solutions.

In spring 2001, Microlog Logistics AG and the chair of Discrete Mathematics, Optimization and Operations Research at the University of Augsburg set themselves the goal to solve the logistics problem of a leading Lithium supplier. The aim was to optimize the business processes of this producer for the given time of 560 days (80 weeks).

1 Description of the problem

The Lithium producer has facilities at several locations (some of them in Germany and the United States), in which different lithium products, for instance lithium-granules, can be produced out of their respective preceeding product which is unambiguously specified in the so called lithium product tree. Apart from incurring costs and the given capacity limits it is

*Lehrstuhl für Diskrete Mathematik, Optimierung und Operations Research, Universität Augsburg, D-86135 Augsburg, Germany

†West Landesbank AG, Herzogstr. 15, D-40217 Düsseldorf, Germany (previously at Microlog Logistics AG)

also known which products can be produced at which location and which amount of the preceeding product is needed to produce one unit of a certain product.

Different products may be produced at one location, and it is also possible that a product can be produced at several locations. There are also locations (so called "sources") at which an "indefinitely large" supply of a certain product is available.

The Lithium producer has a defined set of customers for the several different lithium products. Moreover, it is well known which products are demanded by a certain customer. A storage facility for the products at the individual locations is always given.

If a product transport is scheduled between two locations of the producer or from a location of the company to a customer, the product can be transported at certain costs. Moreover, a customer order is to be satisfied in the time span given by the customer.

For the transport of the products there are one way containers, which are used only once, as well as returnable containers, which are reused, of different types (i.e. Road Tank US or Big Bag) available; some of these can be quite expensive. Container flows are distinguished into empty container flows and dependent (filled) container flows. Variables of the second type are related to product flows by transport constraints. These constraints model the number of containers which are necessary to transport a given amount of product flow.

For the transport of a certain product between two fixed locations mostly only one container type is provided. If a certain amount of a product is to be transported between two locations or to a customer, the number of required containers is determined by the density and amount of the product as well as the volume of the respective container type. Vice versa the number of existing containers of the required type on one day bounds the quantity of a product which can be transported.

For the modeling of the logistics problem, a network node is generated for every product which is available at a certain location and for every product which can be ordered by a certain customer. At these nodes, the storage of the products is modeled via product storage arcs, whereas the production and the transportation of products are modeled by arcs between these nodes. In the same way container flows are modeled. If a container is situated at a certain node at a certain time, it is only available at that time for the product which belongs to the node.

Like product flows, container flows cause costs. Additional costs can arise if containers have to be freighted into shipping containers (oversea

transports) or trucks. This depends on the product which has to be transported, the starting point and the destination of the transport as well as the used container type. As shipping containers and trucks are always only hired if they are needed, they are always sufficiently available.

Naturally, the product flow variables are floating numbers while container flows are integral. Product flows are related to other product flows by product flow balance constraints. In the same way, transport container flows of returnable containers are related to each other by flow balance constraints, but one way containers and shipping containers (including truck transports) are not. The latter are not subject to flow balance constraints, because the Lithium producer is not responsible for the return transport.

Due to the model we now have to distinguish three kinds of variables: Product flows, transport container flows and shipping container flows (including truck transports). All product and container flows within the locations, between them and to the customer as well as the storage of products and returnable containers have to be optimized by minimizing all the costs which are associated with the problem. The mathematical difficulty results from the dependencies of product and container flows as well as of the integrality requirements for container, shipping container and truck flows.

2 The relational database and the optimization tool

All the business processes of the Lithium producer are modeled by a relational database. This data base determines a special kind of flow network with the given product flows and container flows depending on each other together with some time dependent information, namely inventory levels and customer orders.

A mixed integer linear programming (MIP) optimization tool was developed with the purpose of the evaluation of the overall costs and decision making for the Lithium producer's logistics. This optimization tool transforms the dynamic network model into a mixed integer linear programming (MIP) formulation which is then input to a CPLEX LP-Solver, which computes via a Branch & Bound method concrete flow values in a certain time frame $[s_1, t_1]$. The optimization tool delivers the computed flow values via database tables again.

The optimization constraints are the initial inventory levels, the customer orders which must be satisfied in the interval $[s_1, t_1 - 1]$, time invariant product supplies and the production capacity constraints.

The optimization may be restarted with another time frame, say $[s_2, t_2]$ where s_2 must be in the interval $[s_1, t_1]$. In that case, the previous flow values determine the inventory levels at s_2.

The MIP for the total period under consideration of 560 days of the logistics problem would have

- 147,329 constraints
- 226,114 variables, of which 64,655 are integer variables

Due to the limited computing power and capacity of the AIX-workstation at the University of Augsburg on which the CPLEX LP-Solver was running it was only possible to optimize intervals of at most 300 days. Therefore a temporal partitioning of the logistic problem was necessary.

It has emerged that a partitioning of the problem in 5 to maximally 15 intervals makes sense. Then the resulting MIPs for the single intervals possess an average of 25,000 constraints and 40,000 variables, of which in general more than a quarter are integer variables.

3 Labeling policy for problem variables and restrictions

The variable and restriction names of the MIP passed to CPLEX provide some information, but are not too descriptive and hence need some explanation. Variables are associated with the network arcs. A typical variable name looks as follows:

SPR0028T088

The first letter denotes the arc type such as 'S' (Storage) or 'M' (Maintainance). The next two letters denote the entity, that is either 'PR' (Products), 'EC' (Empty container), 'FD' (Filled Container), 'ES', 'FS', 'ET' or 'FT' (Shipping containers and trucks). The four digits which follow determine the ID of the network arc. The next letter 'T' is a constant delimiter. The concluding three digits denote a time stamp, namely the day at which an action was started. In our example, the variable denotes the product flow label associated with the arc with ID 28 in the table PArcs.txt. This arc models the storage of a product from day 88 to day 89 (storage arcs have processing time 1).

The policy for the constraint names is similar: The first part is the general constraint type with the possible alternatives 'BPROD', 'BCONT', 'CTRANS', 'CFSHIP', 'CESHIP', 'CFTRUCK' and 'CETRUCK'. The first two constraint types model the flow balance of products and containers respectively. The other constraint types model the packing of products into transport containers ('CTRANS') and of transport containers into shipping containers ('CFSHIP', 'CESHIP') and trucks ('CFTRUCK', 'CETRUCK'). Note that capacity bounds and integrality requirements are not explicit constraints in the MIP format. Then an arc ID and a time stamp follow which are separated by a letter 'T'. The ID may either denote a node (for flow balances), a product arc (for 'CTRANS' type), an empty container arc ('CESHIP', 'CETRUCK') or an dependent container arc ('CFSHIP', 'CFTRUCK'). For container flow balance constraints, the container type is also specified, separated by a letter 'C'. Two examples for constraints:

```
BCONT0040T310C05:
        1.000 SEC0366T310
-       1.000 SEC0366T309
+       1.000 TEC0446T310
-       1.000 TFC0214T309
=       0.000


CESHIP0446T336:
       48.000 TES0446T336
-       1.000 TEC0446T336
>=           0
```

The first constraint models the balance of incoming and leaving containers of type 5 at day 310 for the network node 40. The second constraint models the packing of empty transport containers into shipping containers at day 336 where the transport containers flow is specified by the arc with ID 466.

4 Mathematical background

4.1 NP-completeness of the problem

The logistics problem under consideration cannot be solved easily due to the integrality requirements. It will even emerge in the following that the problem is NP-complete and therefore difficult to solve.

In order to prove the NP-completeness of our problem we first have to show that the problem belongs to the class NP. This is obviously the case because a given solution can be checked for admissibility (flow balance, capacity compliance, etc.) in polynomial time.

The proof of the NP-completeness of the logistic problem of the Lithium producer is done with the aid of the first proven NP-complete decision problem **SATISFIABILITY** (shortly called **SAT**); cf. [2].

Problem (SAT): Let $X = \{x_1, \ldots, x_n\}$ be a set of Boolean variables. If x_i is a variable in the set X, x_i and the negation $\bar{x}_i$ are called *literals*. A clause C_i is a disjunction ("or"-connection) of some of the literals. Only formulas in *conjunctive normal form* are considered, that means conjunctions ("and"-connections) $C_1 C_2 \ldots C_m$ of clauses. Now the problem **SAT** requires to decide whether there is an assignment of the variables x_i with Boolean variables so that a given formula $C_1 C_2 \ldots C_m$ becomes true.

The problem **SAT** is now transformed as follows to our logistics problem:

With every variable, we associate a network node which is not a customer node and which is called a *variable node* in the following, while the clauses are mapped to customer nodes. At each variable node there is exactly one container available at the beginning. Moreover there are two network nodes for every literal in a clause, where one of the two is always a source node, called the *literal source node*. From the literal source node to the second node, called the *literal target node*, there always exists a product arc. Furthermore there exists a product arc from every literal target node to the customer node which represents the underlying clause. For each of these product arcs there exists a parallel one-way container flow arc. It is always the same product which is assigned to all network nodes.

Up to two container flow arcs for empty containers start from every variable node. The first container flow arc goes to some literal target node which belongs to the variable in non-negative form (if possible). Also there is a container flow arc for filled containers which runs parallel to the product flow arc starting at the literal source node. If the same literal should occur once more, there is a further container flow arc for empty containers from the previous literal target node to the new literal source node. In the same way, a container flow arc for filled containers is generated parallel to the latest product flow arc. This is done until all the positive literals are handled. In an analogous manner container flow arcs for negative literals are generated for every variable node. To all these container flow arcs returnable containers of the same type are assigned. Storage of these containers is possible at all literal nodes.

In order to model the transformation as simply as possible, all execution times are taken to be trivial (0 days). However, a transformation with different execution times would also be possible without a modification of the model. All the customers have ordered for day 0 one barrel of the product. One container is always sufficient for the transport of the product between any of the nodes. The required one-way containers for the transport to the customers are sufficiently supplied at the literal target nodes. All costs are trivial (i.e., 0).

Because of the fact that there is only one container at each variable node there can either flow products on the product flow arcs which belong to the literals at which the variable is not negated or on those at which the variable is negated. This corresponds to the decision of the assignment of the Boolean values to the single variables.

A customer order which belongs to a certain clause can only be accomplished if there is at least one product flow between a literal source node and a literal target node which belongs to the clause. However, there can only be a product flow if there is an associated container flow. Finally, the problem is valid, if all customer orders can be accomplished.

From a valid solution of the transformed problem promptly follows the solution of the given SAT-problem. If the transformed problem is not admissible, the presented formula cannot be satisfied.

As all transformations obviously take place in polynomial time, we have shown that the existence of a polynomial algorithm for our logistics problem would imply a polynomial algorithm for SAT. This proves that the logistics problem of the Lithium producer is NP-complete and therefore cannot be solved easily.

The following example illustrates our transformation. Let:

- $X = \{x_1, x_2, x_3\}$
- $C_1 = \{x_1, \bar{x}_2, x_3\}$, $C_2 = \{x_2, \bar{x}_3\}$, $C_3 = \{\bar{x}_1, \bar{x}_2\}$

A truth assignment of the variables x_i is required so that $C_1C_2C_3$ becomes true. Figure 1 shows a transformation of this problem to our logistics problem. For the sake of clarity all storage arcs as well as the one-way container arcs to the customers are not displayed.

It can easily be seen that this problem is valid: If for example there are container flows on the green container flow arcs which belong to the variable x_1 and on the red container flow arcs which belong to the variables x_2 and x_3, all customer orders can be satisfied. The corresponding truth

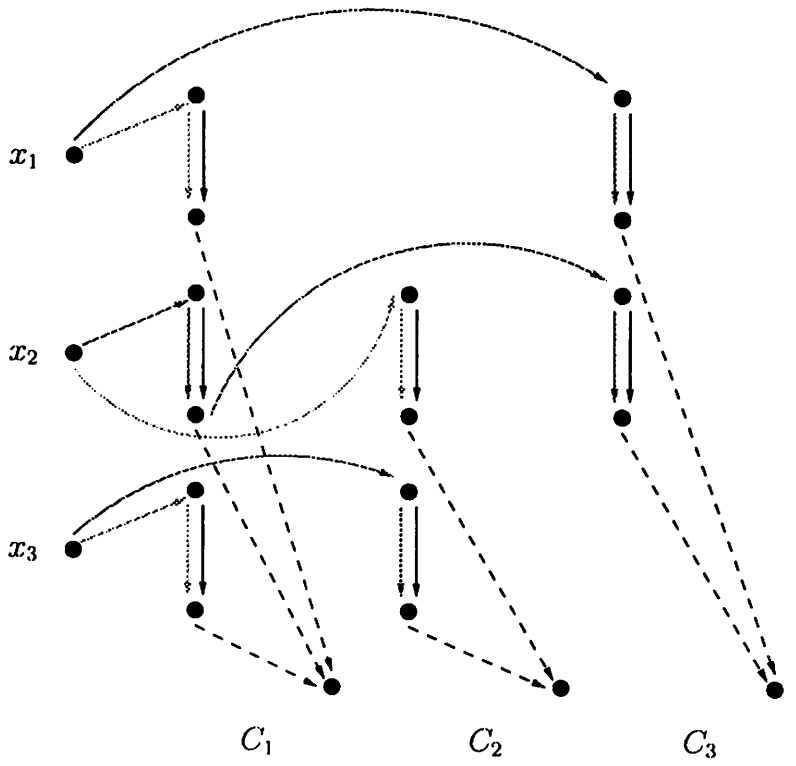

Figure 1: Transformation of the example

assignment of the variables for which the given formula of the SAT-problem becomes true is x_1 = "true", x_2 = "false", x_3 = "false".

4.2 The integrality of empty containers

If all container flows are fixed, the reduced problem is an ordinary linear program and computationally easy. This reduced problem is not a network flow problem in the usual sense since masses change during the production processes.

If all filled container flows and all shipping container flows (including truck transports) are fixed, the problem decomposes into the LP for the product flows and ordinary network flow problems for the various container types, and all of these problems can be solved independently in polynomial time.

It turned out that empty container flows are integral even if these MIP variables are not explicitly required to be integral. This will be proved in a special case which will surely occur in most practical cases.

In this special case we assume that an LP is given at which integer values are assigned to all explicit integer variables by the constraints which got more and more restrictive through the branch & bound method.

We now claim that on the one hand there is an optimal solution of such an LP which fulfills the integrality of the empty container flows and on the other hand that such a solution is calculated by the CPLEX LP-Solver.

As integer values are already assigned to the flow variables of the filled containers, shipping containers and trucks, these variables are replaced by their values at the beginning of the optimization of the LP. As the variables for the empty container flows are now independent from all other variables, the constraints which belong to these variables can be regarded separately for the optimization. These are constraints of the type 'BCONT', 'CESHIP', 'CETRUCK' and the upper capacity bounds for the single variables.

The constraints of the type 'BCONT' result in a system of equations of the form:

$$Ax = b$$

In this system of equations the columns of $A^{k\times n}$ correspond to the n empty container variables and the rows to the k constraints. One should especially take into account that $A_{ij} \in \{-1, 0, 1\}$ for all i and j and that every column has at most one -1 and one 1.

An equation system of the form

$$\tilde{E}x + E\tilde{x} = \tilde{b}$$

describes the constraints of the types 'CESHIP' and 'CETRUCK', where $\tilde{E}^{l\times n}$ is a subset of the rows of the identity matrix $E^{n\times n}$, because shipping containers and trucks are not required for all transports of empty containers. The matrix $E^{l\times l}$ is an identity matrix. The vector $\tilde{x}$ denotes slack variables.

As there is an upper capacity bound for every empty container variable, capacity constraints which are given by a system of equations of the form

$$Ex + E\widehat{x} = \widehat{b}$$

exist. $E^{n\times n}$ is an identity matrix and $\widehat{x}$ is a further vector of slack variables.

Altogether this results in a system of equations of the form

$$\begin{bmatrix} A^{k\times n} & O^{k\times l} & O^{k\times n} \\ \tilde{E}^{l\times n} & E^{l\times l} & O^{l\times n} \\ E^{n\times n} & O^{n\times l} & E^{n\times n} \end{bmatrix} \begin{pmatrix} x \\ \tilde{x} \\ \widehat{x} \end{pmatrix} = \begin{pmatrix} b \\ \tilde{b} \\ \widehat{b} \end{pmatrix}$$

where the matrices 0 are all-zero matrices. In the following the entire restriction matrix is called G.

Now we will show that the matrix G is totally unimodular, which means that every square submatrix has the determinant 0, 1 or -1. This makes sense, as every valid LP with integer data and a totally unimodular restriction matrix has an integer optimal solution (refer to [3], p. 267). Therefore, the total unimodularity of G would imply that there is an integer optimal solution of the problem.

In order to show the total unimodularity of the restriction matrix G we use following result, cf. [1]:

Result 1 *Let $A_{ij} \in \{-1, 0, 1\}$ for every i and every j. If every column has at most one 1 and one -1, then A is totally unimodular.*

Theorem 1 *The restriction matrix G is totally unimodular.*

Proof. Result 1 directly implies the total unimodularity of the matrix A. If a row is added to the matrix A which only has entries 0 but one entry 1, the resulting matrix remains totally unimodular: if a part of the new row is included in a submatrix, the determinant of the submatrix is 0, if the column with the entry 1 of the new row is not included in the submatrix. Otherwise the evaluation of the determinant of the submatrix with respect to the new row gives 1 times the value of the determinant of the reduced submatrix (which is generated by deleting the row and the column which belong to the 1). However, as the determinant of the reduced matrix can only have the values 0, 1 and -1, the total unimodularity of the extended matrix A directly follows. Obviously, it follows that the matrix

$$\begin{bmatrix} A^{k \times n} \\ \tilde{E}^{l \times n} \\ E^{n \times n} \end{bmatrix}$$

is totally unimodular. By a similar argument the total unimodularity is preserved if columns are added to a totally unimodular matrix which have entries 0 except for one entry 1. Thus the matrix G is totally unimodular, as claimed. □

As the restriction matrix G is totally unimodular, an integer optimal solution exists, if the problem is valid. It remains to show that the CPLEX LP-Solver which uses the simplex algorithm delivers such a solution. For that we need a further important result from the theory of integer optimization, cf. [1].

Result 2 *Let A be a totally unimodular matrix of full row rank. The polyhedron $\{x|x \geq 0 : Ax = b\}$ has integral vertices for every integer vector b, if and only if A is totally unimodular.*

It can be assumed that the totally unimodular restriction matrix G has full row rank as the pre-processor of the CPLEX LP-Solver excludes all redundant restrictions of the considered LP before the optimization. Thus it follows from Result 3 that the vertices of the polyhedron $\{x|x \geq 0 : Gx = b\}$ are integral. However, these correspond to the basic solutions of the simplex algorithm. Therefore the simplex algorithm, which the simplex LP-Solver uses, generates only integer optimal solutions.

Thus we have shown for our special case that the explicit demand for the integrality of the empty containers is not necessary. Unfortunately, we could not find a proof in general, but it seems likely that this special case will occur most of the time.

5 The optimization

Due to the necessary partition of the logistics problem into subproblems (temporally seen) again and again there were problems with the validity of the problem during the optimization. Because of this reason the model was on the one hand insofar extended that at the end of every optimization interval a certain minimum stock level at certain storages was demanded. A shortfall was penalized with very high fines (555555/t). Furthermore, there was the possibility that a customer could also receive products from a dummy source at even higher costs (999999/t).

The optimization of the logistics problem finally was done both with the standard settings of the CPLEX LP-Solver and with modified CPLEX-parameters at different reductions of the whole period under consideration (between 5 and 12 intervals). With the modification of the CPLEX-parameters which mainly influence the branch and bound process neither the optimization process could be sped up nor was it possible to achieve any better solution.

It was the use of a self-developed branching strategy which enhanced the results of the optimization as well as sped up the optimization process. In this strategy first those variables are considered during the branch & bound process which model the freight. These variables represent filled containers, shipping containers and trucks. More precisely, the higher the respective costs are the higher the priority of these variables gets. This makes sure that the CPLEX LP-Solver firstly pays attention to the "frame" of the solution.

As the branching direction of the variables is always aimed "upwards", a preferably fast generation of valid solutions is made possible. This is shown above all in the fact that, in contrast to the optimization processes not using the explicit branching specifications, already the first located valid solution of every single time interval had the demanded accuracy of at most 5 per cent error. Here the quality is always measured by the bounds which result from the LP-relaxation of the single intervals. The LP-relaxation is calculated at the beginning of each branch & bound process and is used as a lower bound. Sometimes this lower bound can be improved (i.e., with cutting planes) during the branch & bound process.

Table 1 shows the results which result from a partitioning of the logistics problem in five intervals by using the explicit branching specifications. For every time interval the respective value of the objective function, its quality, the required solution time and the total objective value are given.

time interval	total obj. value	obj. value	accuracy	solution time
0 - 300		1.8815×10^7	1.2%	40 min
250 - 400	4.4654×10^7	3.1790×10^7	0.8%	63 min
350 - 470	6.1044×10^7	2.5052×10^7	0.9%	43 min
420 - 520	7.3197×10^7	2.0852×10^7	0.9%	30 min
470 - 560	8.9360×10^7	2.5849×10^7	0.5%	23 min

Table 1: Results using explicit branching specifications

Table 2 shows the comparison of the results for different partitionings of the problem (5, 7, 12 and 15 intervals). Next to the total values of the objective function and the respective required solution time, the reduced objective value is given. This corresponds to the objective function value reduced by the existing fines.

intervals	5	7	12	15
objective value	8.9360×10^7	9.7578×10^7	8.8928×10^7	9.4914×10^7
red. obj. value	7.9280×10^7	7.9390×10^7	7.9718×10^7	7.9643×10^7
solution time	3h 19min	3h	2h 54min	4h 25min

Table 2: Comparison of the results using explicit branching specifications

Comparing the single values of the objective function with the LP-relaxation (7.8686×10^7) one can see that the solutions of the single in-

terval partitionings have a quality between 12.8 and 24.0 per cent. If one has a closer look at the solution of the LP-relaxation, one observes that it does neither contain any fines caused by shortfalls of certain given minimum stock levels nor any product demands from the dummy source. But exactly these are included with very high costs in the solutions of the single interval partitionings. Therefore it is unrealistic to compare the single objective function values to the LP-relaxation. As the flows causing the fines are only very small, it makes more sense to use the reduced values of every objective function for the comparison. These only differ maximally 1.3 per cent from the LP-relaxation. Due to this reason the optimization of the single interval partitionings provides very good and practically useful solutions.

As it appears from table 2, the different interval partitionings mainly have an impact on the fines. The interval partitioning with five intervals seems to provide the best solution because of the small fines and its smallest reduced value of the objective function. If such a solution data set is used, finally there only has to be decided how the relatively small use of the dummy source is to be eliminated in reality.

As it was still of great interest to optimize over the whole time frame in one step, a conversion of the logistics problem from the day calculation (560 days) to a week calculation (80 weeks) was also made possible. Here special importance was attached to achieving the best possible approximation of the day calculation by the week calculation, as the solutions of the week calculation should serve for conclusions about decisions on single days.

If the logistics problem of the Lithium producer is solved using a week calculation, the solution (7.9235×10^7) has a quality of 0.7 per cent and no flows causing fines. When comparing the week calculation to the day calculation at which only small flows causing fines are existing we again have to compare the solution to the reduced cost of the solutions of the day calculation. Then the week calculation obtains a value at most 0.6 per cent better than that of the day calculation and therefore differs only marginally from the results of the day calculation.

With the week calculation the Lithium producer possesses an instrument which approximates the day calculation very well and with which much larger time spans can be optimized. Therefore the solution of the week calculation can be used as a basis for a heuristic determination of a solution for the day calculation.

Finally, we also developed a programm which executes automatically the whole optimization process using the already mentioned optimization tool and the CPLEX LP-Solver. Besides, a programm for the graphical evaluation of the solutions was provided. With this tool warehouse stocks

can for example be controlled over the course of time or the maintaining of a flow can be verified.

Eventually, the logistics problem of the Lithium producer was not only solved satisfyingly, but the necessary programs for the solution were also provided which allow to master the continuous optimization process in the best possible way.

Acknowledgement: This paper is based on the diploma thesis "Optimierung eines Logistikproblems in der Metallindustrie" written by the second author under the supervision of the third author. This thesis took the results of a research project between the chair of Discrete Mathematics, Optimization and Operations Research at the University of Augsburg and Microlog Logistics AG as a starting point and was financially supported by Microlog.

References

[1] Cook, W.J., Cunningham, W.H., Pulleyblank, W.R. and Schrijver, A. (1998): *Combinatorial Optimization.* John Wiley & Sons, New York.

[2] Garey, M.R. and Johnson, D.S. (1979): *Computers and intractability: A guide to the theory of NP-completeness.* Freeman, New York.

[3] Schrijver, A. (1986): *Theory of integer and linear programming.* John Wiley & Sons, New York.