RUTCOR
RESEARCH
REPORT

Towards the Numerical Solution of a Large and Difficult MILP Problem of Closed Loop Layout

Béla Vizvári[a]

RRR 8-2014,

RUTCOR • Rutgers Center for Operations Research • Rutgers University • P.O. Box 5062 • New Brunswick New Jersey • 08903-5062
Telephone: 908-932-3804
Telefax: 908-932-5472
Email: rrr@rutcor.rutgers.edu

[a]Department of Industrial Engineering, Eastern Mediterranean University, vizvaribela@gmail.com

Rutcor Research Report
RRR 8-2014,

Towards the Numerical Solution of a Large and Difficult MILP Problem of Closed Loop Layout

Béla Vizvári

Abstract. This paper contributes to two different areas: (1) layout problems and (2) integer programming.

(1) One important class of layout problems arises when rectangles are to be placed on the plane. The rectangles represent machines with their close environments or small production cells. There is a flow of the semi-finished products among the rectangles. The problem is to place the rectangles without overlapping such that the total transportation, i.e. the sum of the products of the flows and distances, is minimal. A subcase of this general problem is the closed loop layout problem where the track of the transportation is a rectangle as well. The rectangular layout problem has a set of benchmark problems starting with 4 cells and for each even number there is a problem up to 18 cells. To the best knowledge of the author the optimal solution is known only for 4 and 6 cells. This paper provides the optimal solution of the 8 cell problem and suggest a very good feasible solution for the 10 cell problem. The latter is supposed to be optimal.

(2) The closed loop layout problem is modeled by a mixed 0-1 programming problem. It is a large scale problem, however recent leading solvers can solve even larger problems. In spite of this fact, it is too difficult even for the top solvers. For even 8 or 10 cells, the model cannot be solved in a single run. It means that the structure of the problem is one which makes it extremely difficult. A possible numerical solution can be that the basic problem is cut into several subproblems and the subproblems are solved, more precisely fathomed. This procedure can be characterized as follows: it is a branch and bound structure above the single-run branch and bound procedure of the optimizer. It is described in the paper. It was strong enough to solve exactly the 8 cell problem. The 10 cell problem is still under the procedure. The procedure uncovered some special properties of the integer programming solution methods and suggests some principles to obey in the solution of large scale MILP problems.

1 Introduction

There is a very large variety of problems modeled by Mixed Integer Linear Programming (MILP). In the majority of cases MILP means that the integer variables are binary variables. Binary variables are especially suitable to model *YES/NO* decisions. For example in the layout problem discussed in this paper, a binary decision is whether or not a cell lies within or outside the rectangular shaped track.

Other papers usually report success in connection with the new MILP models. This paper reports only limited success with clear view on the directions of future research.

Very often, a layout problem of industrial origin has the following properties: (i) the objects to be placed have rectangle shapes, (ii) the sizes of the rectangles are different and fixed, (iii) the rectangles may not overlap, however they may touch one another, (iv) there are flows between the pairs of the rectangles, (v) the total transportation is the sum of the products of the flow values and the distances of the rectangles, (vi) the edges of the rectangles must be vertical and horizontal (vii) the rectangles can be rotated by 90, 180, and 270 degrees, (viii) the track of the transportation system may go on the edges of the rectangles, however it may not cross the rectangles, and (ix) the problem is to determine the layout with minimal total transportation.

Notice that the exact shape of the track is not *a priori* determined and its construction is part of the solution.

Further restrictions are possible according to the track of the transportation system. A layout problem is called *open field* if there is no restriction on the track. All constraints of the open field problem was modeled exactly by [Das 1993]. However the model contains an approximate objective function which gives a lower bound of the optimal value only and can mislead the meta-heuristic methods [Niroomand and Vizvári 2013]. For meta-heuristic methods see [Chae and Peters 2006], [Das 1993], [Kim and Kim 2000], [Lasardo and Nazzal 2011], [Rajasekharan *et al.* 1998], [Samarghandi *et al.* 2010], [Tavakkoli-Moghaddam and Panahi 2007], and [Ting, J.-H., Tanchoco 2001]. Misleading means not only avoiding the optimal solution but also the generation of solutions which can be improved in an obvious way.

The challenge of measuring the distances of cells leads to the inexactness of the model. It is also assumed that (x) the input/output point of each rectangle is the middle point of the a well-defined edge. The input/output point is also called the pick-up point. Thus, the distance of the input/output points must be measured. An exact model is obtained only if the distance used in the objective function is the distance that the vehicle passes in the constructed optimal track. This distance is referred below as *distance on the track*. There are solutions where the vehicle must go both up and down or left and right between two input/output points. [Das 1993] did not use the distance on the track but substituted it by the Manhattan distance of the two points. (The Manhattan distance is also called rectilinear distance and in mathematical environment l_1 distance.) Even some solutions provided by [Das 1993] with heuristic methods can be trivially improved because of the misleading effect of the Manhattan distance [Niroomand and Vizvári 2013].

The model by [Das 1993] is very important in spite of the inexactness of the objective function as any restricted version of the problem must satisfy the constraints of [Das 1993].

It should be mentioned that some papers take into consideration even intra-rectangle transportation. However the amount of this type of transportation is always constant and does not affect the optimal solution, therefore it is disregarded in this paper.

A more restricted version of the problem is the *Closed Loop Layout Problem* (CLLP). In the case of CLLP, the shape of the track is also a rectangle but its size is not determined *a priori*. The vehicle can go in both directions. The first exact model of the layout problems with rectangles is given for CLLP by [Niroomand and Vizvári 2013].

Exact models of the open field problem are provided by [Kovács and Vizvári 2014]. There are further versions of the layout problem with different types of track, however they are not directly connected to the topic of this paper, therefore they are not discussed.

2 Philosophical and Subjective Remarks on the Origin of the Research Reported in This Paper

The origin of this paper starts with the numerical experiences carried out with the model of [Niroomand and Vizvári 2013]. There is a set of eight benchmark problems for 4, 6, 8, 10, 12, 14, 16, 18 rectangles. The first five problems were generated by [Das 1993] and the last three problems by [Rajasekharan *et al.* 1998].

One significant problem of this research field is that the published numerical results contain only objective function values belonging to the approximate objective function. However, the structure of the best found solution with one exception are completely unknown. [Das 1993] published the figures of three feasible solutions of the 6 rectangles problem without the numerical descriptions of the solutions. But the solutions are more or less reconstructable from the figures.

From the point of view of the philosophy of science, a result can be considered a scientific result *only if* it is reconstructable. The fact that the structures of the best solutions are unpublished, means that the results are not completely on the level of science. Moreover, this lack of information makes comparison with any further results very difficult.

Another remark is that the misleading property of the approximate objective function is known from the fact that the solutions of [Das 1993] for the 6 rectangles problem are improvable. In this case, the optimal closed loop solution is better than the former best meta-heuristic solution of the open field problem if the exact objective function is taken into consideration.

In [Niroomand and Vizvári 2013] the model was solved completely only for 4 and 6 rectangles. At the same time it became clear that a branch and bound procedure can play the role of a meta-heuristic procedure if it is interrupted and is run again with different parameters of the solver. Different parameters create different course of the enumeration in the branch and bound procedure. Thus, the repeated and interrupted branch and bound was introduced as a new meta-heuristic procedure. In this way several good feasible solutions were generated for all benchmark problems having at least 8 rectangles. These solutions are competitive with the previous solutions having unknown structure and unknown exact objective function values. All the best solutions found in [Niroomand and Vizvári 2013] are published in the same paper.

In spite of the afore-mentioned results one reviewer of [Niroomand and Vizvári 2013] stated that the mathematical model has no value, as only problems with not more than 6 rectangles are solved with it which are easy for human beings. This statement is obviously wrong as an exact model can be the starting point of further research. The original authors or other researchers can use the exact model as raw material of new ideas. Thus for new exact model has an intellectual value on its own right. Another factor is that several different technologies have promised in the recent years providing 1000 times faster computers than the recent ones. If any of them appears on the market then the recent solvers based on the recent models will be able to solve much larger problems. This also shows that the intellectual value of a model is not determined by the recent technological abilities. This remark of the reviewer reflects the sad fact that the quality of the reviewing procedure is very low recently. On the other hand it inspired me to prove that the model is able to solve larger problems even in the recent circumstances.

The first result was the exact solution of the problem with 8 rectangles. The second one is the attempt to solve the problem with 10 rectangles. This procedure gave a very good feasible solution and many lessons for MILP.

3 The Exact Model of the Closed Loop Layout Problem

In the case of the closed loop layout, the model must contain the exact description of three important items: (A) the rectangles must not overlap, (B) the rectangles must be on the rectangle shaped track, (C) the exact objective function. (A) is completely solved by [Das 1993], (B) and (C) are solved in [Niroomand and Vizvári 2013].

In this study, the terms rectangle and cell refer to the same thing and will be used interchangeably for the rest of the paper.

[Das 1993] described the prohibition of the overlap by the requirement that the central points of every pair of rectangles must be far enough from one another. It means that at least one of the horizontal and vertical distances of the two points must be as long as the half of the sum of the lengths of the two horizontal and vertical edges, respectively.

In what follows the input/output point of a cell is called the pick-up point. It is supposed that the first edge of the rectangle is the horizontal one and the other (second) edge is the vertical one in the original position, *i.e.* in 0 degree rotation, of the rectangle.

In what follows the cells are represented by their center. The position of a cell is vertical (horizontal) if the position of its first edge is vertical (horizontal). In the case of a square the tie can be broken arbitrarily.

The discussion of the model is based on [Niroomand and Vizvári 2013].

The notations used in the model are summarized in Tables 1, 2, and 3. The coefficients and variables are not introduced in the text.

Notations

Table 1. Input data and indices

n	the number of cells	parameter
i, j	indices of cells	index
s_i	the length of the second edge of cell i	parameter
t_i	the length of the first edge of cell i	parameter
ω_i	the distance of the pick-up point of cell i from the center of the cell	parameter
φ_{ij}	the flow value between cells i and j	parameter

Table 2. Variables used in the model by [Das 1993]

x_i (y_i)	horizontal (vertical) co-ordinate of the center of cell i
z_i	binary variable; it is 1 (0) if cell i is in vertical (horizontal) position
a_i (b_i)	horizontal (vertical) co-ordinate of the pick-up point of cell i
e_{ij}	$\max\{0, x_i - x_j\}$
f_{ij}	$\max\{0, x_j - x_i\}$
g_{ij}	$\max\{0, y_i - y_j\}$
h_{ij}	$\max\{0, y_j - y_i\}$
α_{ij}	a binary variable; it is 1 if $x_i \geq x_j$
β_{ij}	a binary variable; it is 1 if $y_i \geq y_j$
δ_{ij}	binary variable; if it is 1 then cells i and j are not overlapping vertically and if it is 0 then cells i and j are not overlapping horizontally
M	a great positive number
λ_{1i}, λ_{2i}, λ_{3i}, λ_{4i},	binary variables describing the position of the pick-up point of cell i according to its rotation

3.1 Non-Overlapping Constraints

As it was mentioned before the minimal required horizontal (vertical) distance such that two cells are not overlapping is the half of the sum of length of their edges in horizontal (vertical) position. The sum depends on the rotation of the cells, of course. Notice that $e_{ij} + f_{ij}$ ($g_{ij} + h_{ij}$) is the horizontal (vertical) distance of the centers of the cells i and j. This requirement is described by the following inequalities:

$$\forall i, j, i \neq j: \; e_{ij} + f_{ij} - \frac{1-z_i}{2}t_i - \frac{z_i}{2}s_i - \frac{1-z_j}{2}t_j - \frac{z_j}{2}s_j \geq -M\delta_{ij} \tag{1}$$

and

$$\forall i, j, i \neq j: \; g_{ij} + h_{ij} - \frac{1-z_i}{2}s_i - \frac{z_i}{2}t_i - \frac{1-z_j}{2}s_j - \frac{z_j}{2}t_j \geq -M(1-\delta_{ij}). \tag{2}$$

It is difficult to use the formulae of e_{ij}, f_{ij}, g_{ij}, and h_{ij} explicitly in an optimization problem, therefore they are described implicitly by the following constraints:

$$\begin{aligned}
\forall i, j, i \neq j: \; x_i - x_j &= e_{ij} - f_{ij} && (3)\\
\forall i, j, i \neq j: \; y_i - y_j &= g_{ij} - h_{ij} && (4)\\
\forall i, j, i \neq j: \; e_{ij} &\leq M\alpha_{ij} && (5)\\
\forall i, j, i \neq j: \; f_{ij} &\leq M(1-\alpha_{ij}) && (6)\\
\forall i, j, i \neq j: \; g_{ij} &\leq M\beta_{ij} && (7)\\
\forall i, j, i \neq j: \; h_{ij} &\leq M(1-\beta)_{ij} && (8)\\
\forall\, i: \; x_i,\, y_i,\, e_{ij},\, e_{ij},\, f_{ij},\, g_{ij},\, h_{ij},\, &\geq 0. && (9)
\end{aligned}$$

Notice that the inequalities (1) and (2) handle both overlapping and rotation. Constraints (5) and (6) with nonnegativity constraint (9) ensure that one of e_{ij} and f_{ij} is zero. A similar statement is true for g_{ij} and h_{ij}.

3.2 The Rectangles Must Lay the Track

The next set of constraints determines the positions of the cells from the closed loop point of view. Each cell must be either completely inside, or completely outside of the track. Furthermore the edge of the cell where the vehicle may enter into the cell, must lie on one of the edges of the track.

The co-ordinates of the four corner points of cell i depend on the rotation of the cell described by the binary variable z_i. They are as follows:

$$(x_i - \frac{1-z_i}{2}t_i - \frac{z_i}{2}s_i, y_i - \frac{1-z_i}{2}s_i - \frac{z_i}{2}t_i), \;\; (x_i - \frac{1-z_i}{2}t_i - \frac{z_i}{2}s_i, y_i + \frac{1-z_i}{2}s_i + \frac{z_i}{2}t_i),$$
$$(x_i + \frac{1-z_i}{2}t_i + \frac{z_i}{2}s_i, y_i + \frac{1-z_i}{2}s_i + \frac{z_i}{2}t_i), \;\; (x_i + \frac{1-z_i}{2}t_i + \frac{z_i}{2}s_i, y_i - \frac{1-z_i}{2}s_i - \frac{z_i}{2}t_i).$$

A cell is inside the track if

$$h_1 \;\leq\; x_i - \frac{1-z_i}{2}t_i - \frac{z_i}{2}s_i, \quad h_2 \;\geq\; x_i + \frac{1-z_i}{2}t_i + \frac{z_i}{2}s_i$$

and

$$v_1 \;\leq\; y_i - \frac{1-z_i}{2}s_i - \frac{z_i}{2}t_i, \quad v_2 \;\geq\; y_i + \frac{1-z_i}{2}s_i + \frac{z_i}{2}t_i.$$

Furthermore one of these inequalities must be satisfied with the equation.

A binary variable ϑ_i is introduced to describe if cell i is outside the track or inside. $\vartheta_i = 1$ if cell i is outside.

The cell must satisfy different conditions if it is inside the track than if it is outside.

Inside constraints. A pair of inequalities must be satisfied for each of the four edges the track. The first inequality claims that the cell is inside of the track, and the second one claims that its pick-up point is on the edge. Obviously the first constraint must not claimed if the cell is outside and the cell can be on only one of the edges. It means that the constraints must be satisfied automatically in certain cases and that is ensured by using the binary variables ψ_{ki}'s and ϑ_i's.

$$\text{Edge 1: } y_i + \frac{1-z_i}{2}s_i + \frac{z_i}{2}t_i - M\vartheta_i \le h_2,\ y_i + \frac{1-z_i}{2}s_i + \frac{z_i}{2}t_i + M\vartheta_i + M(1-\psi_{1i}) \ge h_2 \quad (10)$$

$$\text{Edge 2: } x_i + \frac{1-z_i}{2}t_i + \frac{z_i}{2}s_i - M\vartheta_i \le v_2,\ x_i + \frac{1-z_i}{2}t_i + \frac{z_i}{2}s_i + M\vartheta_i + M(1-\psi_{2i}) \ge v_2 \quad (11)$$

$$\text{Edge 3: } y_i - \frac{1-z_i}{2}s_i - \frac{z_i}{2}t_i + M\vartheta_i \ge h_1,\ y_i - \frac{1-z_i}{2}s_i - \frac{z_i}{2}t_i - M\vartheta_i - M(1-\psi_{3i}) \le h_1 \quad (12)$$

$$\text{Edge 4: } x_i - \frac{1-z_i}{2}t_i - \frac{z_i}{2}s_i + M\vartheta_i \ge v_1,\ x_i - \frac{1-z_i}{2}t_i - \frac{z_i}{2}s_i - M\vartheta_i - M(1-\psi_{4i}) \le v_1 \quad (13)$$

$$i = 1, 2, \ldots, n$$

The fact that the pick-up point of cell i must be on exactly one edge is expressed by the equation

$$\psi_{1i} + \psi_{2i} + \psi_{3i} + \psi_{4i} = 1 \quad i = 1, 2, \ldots, n. \quad (14)$$

Notice that the first constraints are automatically satisfied if cell i is outside as $\vartheta_i = 1$, thus "big M" helps to do so. Further on, the two constraints of an edge together claim equation if and only if $\vartheta_i = 0$, *i.e.* the cell is inside, and $\psi_{ki} = 1$, *i.e.* the indicator variable claims that the cell is on edge k.

Outside constraints. The lower edge of a cell cannot be higher than the upper edge of the track, otherwise the cell is not on the track. Similarly its left (upper, right) edge cannot be right (under, left) to the right (lower, left) edge of the track. However the two edges must be on the same line if the indicator variable ψ_{ki} claims it. Moreover if cell i is on a horizontal (vertical) edge of the track then the horizontal (vertical) co-ordinate of its center point must be in the horizontal (vertical) range of the track. Hence two pairs of inequalities must be satisfied for each edge of the track.

Edge 1:

$$\begin{aligned} y_i - \frac{1-z_i}{2}s_i - \frac{z_i}{2}t_i &\le h_2, \\ y_i - \frac{1-z_i}{2}s_i - \frac{z_i}{2}t_i + M(1-\vartheta_i) + M(1-\psi_{1i}) &\ge h_2, \\ v_2 + M(1-\psi_{1i}) \ge x_i &\ge v_1 - M(1-\psi_{1i}) \end{aligned} \quad (15)$$

Edge 2:

$$\begin{aligned} x_i - \frac{1-z_i}{2}t_i - \frac{z_i}{2}s_i &\leq v_2, \\ x_i - \frac{1-z_i}{2}t_i - \frac{z_i}{2}s_i + M(1-\vartheta_i) + M(1-\psi_{2i}) &\geq v_2 \\ h_2 + M(1-\psi_{2i}) \geq y_i &\geq h_1 - M(1-\psi_{2i}) \end{aligned} \tag{16}$$

Edge 3:

$$\begin{aligned} y_i + \frac{1-z_i}{2}s_i + \frac{z_i}{2}t_i &\geq h_1, \\ y_i + \frac{1-z_i}{2}s_i + \frac{z_i}{2}t_i - M(1-\vartheta_i) - M(1-\psi_{3i}) &\leq h_1 \\ v_2 + M(1-\psi_{3i}) \geq x_i &\geq v_1 - M(1-\psi_{3i}) \end{aligned} \tag{17}$$

Edge 4:

$$\begin{aligned} x_i + \frac{1-z_i}{2}t_i + \frac{z_i}{2}s_i &\geq v_1, \\ x_i + \frac{1-z_i}{2}t_i + \frac{z_i}{2}s_i - M(1-\vartheta_i) - M(1-\psi_{4i}) &\leq v_1 \\ h_2 + M(1-\psi_{4i}) \geq y_i &\geq h_1 - M(1-\psi_{4i}) \\ i = 1, 2, \ldots, n & \end{aligned} \tag{18}$$

Notice that the constraints claim that a cell must lay on a certain edge of the track, are satisfied automatically again in all other cases.

For the sake of completeness, the nature of the new variables is claimed again:

$$\forall\, i, j, k, l: \; w_{klij}, \psi_{ki}, \vartheta_i \;=\; 0 \text{ or } 1. \tag{19}$$

3.3 The Exact Distances of the Cells

The next main step is the formulation of the objective function. It is the minimization of the sum of the flow between cells weighted by the distance of the pick-up points of the cells. The first step is to determine the co-ordinates of the pick-up points. It is the middle point of an edge. If the pick-up point is on the second edge then $\omega_i = t_i/2$, and if it is on the first edge then $\omega_i = s_i/2$. The co-ordinates depend (i) on the definition of the position of the pick-up point, *i.e.* if it is on the the first or second edge and (ii) on the rotation of the cell in the layout. The rotation of cell i is described by four binary variables, λ_{1i}, λ_{2i}, λ_{3i}, and λ_{4i} defined as follows:

$$\lambda_{1i} = \begin{cases} 1 & \text{if the pick-up point is on the right side of the center} \\ 0 & \text{otherwise} \end{cases}$$

$$\lambda_{2i} = \begin{cases} 1 & \text{if the pick-up point is below the center} \\ 0 & \text{otherwise} \end{cases}$$

$$\lambda_{3i} = \begin{cases} 1 & \text{if the pick-up point is on the left side of the center} \\ 0 & \text{otherwise} \end{cases}$$

$$\lambda_{4i} = \begin{cases} 1 & \text{if the pick-up point is above the center} \\ 0 & \text{otherwise} \end{cases}$$

The λ and z variables are not independent. Two equations must hold between them. If the pick-up point is the middle point of the first edge then the cell is in vertical position if the pick-up point is below or above the center. Thus

$$z_i = \lambda_{2i} + \lambda_{4i} \tag{20}$$

implying that

$$1 - z_i = \lambda_{1i} + \lambda_{3i}. \tag{21}$$

If the the pick-up point is the middle point of a second edge then the form of the equations is the following using the same equation numbering

$$z_i = \lambda_{1i} + \lambda_{3i} \tag{20}$$

and

$$1 - z_i = \lambda_{2i} + \lambda_{4i}. \tag{21}$$

Finally, the two co-ordinates of the pick-up point of cell i are

$$\forall i: \ a_i = x_i + \omega_i(\lambda_{1i} - \lambda_{3i}) \tag{22}$$

and

$$\forall i: \ b_i = y_i + \omega_i(\lambda_{4i} - \lambda_{2i}). \tag{23}$$

Without loss of generality we may assume that the cells are in the nonnegative quarter of the plane:

$$\forall i: \ a_i, b_i \geq 0. \tag{24}$$

All other variables are binary:

$$\forall i, j: \ z_i, \alpha_{ij}, \beta_{ij}, \delta_{ij}, \lambda_{1i}, \lambda_{2i}, \lambda_{3i}, \lambda_{4i}, = 0 \text{ or } 1. \tag{25}$$

Notations # 2

Table 3. Further notations related to distances

k, l	indices of the edges of the track	index
w_{klij}	binary variable; it is 1 if cell i is on edge k and cell j is on edge l	variable
v_1, v_2	the two vertical co-ordinates of the track ($v_2 \geq v_1$)	variable
h_1, h_2	the two horizontal co-ordinates of the track ($h_2 \geq h_1$)	variable
d^k_{ij}	the distance of cells i and j if both are on edge k; 0 otherwise	variable
d^{kl}_{ij}	the distance of cells i and j if cell i is on edge k and cell j is on edge l ($k \neq l$); 0 otherwise	variable
ψ_{ki}	binary variable; it is 1 if cell i is on edge k	variable
ϑ_i	binary variable; it is 1 if cell i is outside the track	variable
a_i (b_i)	the x (y) co-ordinate of the pick-up point of cell i	variable
u_{13ij}, u_{31ij} u_{24ij}, u_{42ij}	binary variables; they select the minimal path for the vehicle if cells i and j are on opposite edges	variable

The edges of the track have the indices as follows: the upper horizontal edge is 1, the right vertical edge is 2, the lower horizontal edge is 3, and the left vertical edge is 4.

Notice that

$$w_{klij} = \psi_{ki}\psi_{lj}.$$

This relation can be described equivalently by two linear inequalities using an old technique of integer programming:

$$\begin{aligned} 2w_{klij} &\leq \psi_{ki} + \psi_{lj} \\ \psi_{ki} + \psi_{lj} - 1 &\leq w_{klij}. \end{aligned}$$

The equivalence is based on the fact that all three variables are binary. Therefore the first set of constraints of the model is

$$2w_{klij} \leq \psi_{ki} + \psi_{lj} \quad 1 \leq i < j \leq n,\ k, l = 1, 2, 3, 4 \tag{26}$$

$$\psi_{ki} + \psi_{lj} - 1 \leq w_{klij} \quad 1 \leq i < j \leq n,\ k, l = 1, 2, 3, 4. \tag{27}$$

The distances are restricted in the model only from below, *i.e.* by lower bounds. In the optimal solution, the optimality condition forces them to be equal to their maximal lower bound.

There are several cases according to the (potential) position of the two cells.

Case 1: cells i and j are both on edge k. Then one of the co-ordinates of the two pick-up points is the same. If they are on edge 1 or 3 then it is the y co-ordinate, otherwise it is the x co-ordinate. Let d_{ij}^k be the distance of the pick-up points of the two cells. In any other case d_{ij}^k is 0. Then the distance must satisfy the following inequalities.

$$d_{ij}^k + M(1 - w_{kkij}) \geq a_i - a_j \quad 1 \leq i < j \leq n, \; k = 1,3 \tag{28}$$

$$d_{ij}^k + M(1 - w_{kkij}) \geq a_j - a_i \quad 1 \leq i < j \leq n, \; k = 1,3 \tag{29}$$

$$d_{ij}^k + M(1 - w_{kkij}) \geq b_i - b_j \quad 1 \leq i < j \leq n, \; k = 2,4 \tag{30}$$

$$d_{ij}^k + M(1 - w_{kkij}) \geq b_j - b_i \quad 1 \leq i < j \leq n, \; k = 2,4 \tag{31}$$

Notice that the constraints (28)-(31) are not restrictive if not both of cells i and j are on edge k as then $w_{kkij} = 0$ and the constraint is satisfied automatically.

Case 2: cells i and j are on two adjacent edges. Then the vehicle must pass a corner point of the track. For example if cell i is on edge 1 and cell j is on edge 2 then the vehicle must go through the upper right corner of the track. The co-ordinates of this point are (v_2, h_2). The pick-up point of cell i is left from this point, the pick-up point of cell j is under it. Hence the distance d_{ij}^{12} must satisfy the inequality similar to the ones in (28)-(31):

$$d_{ij}^{12} + M(1 - w_{12ij}) \geq v_2 - a_i \; + \; h_2 - b_j. \tag{32}$$

Similarly, the distances of Case 2 must satisfy the following inequalities.

$$d_{ij}^{21} + M(1 - w_{21ij}) \geq v_2 - a_j \; + \; h_2 - b_i \tag{33}$$

$$d_{ij}^{23} + M(1 - w_{23ij}) \geq v_2 - a_j \; + \; b_i - h_1 \tag{34}$$

$$d_{ij}^{32} + M(1 - w_{32ij}) \geq v_2 - a_i \; + \; b_j - h_1 \tag{35}$$

$$d_{ij}^{34} + M(1 - w_{34ij}) \geq a_i - v_1 \; + \; b_j - h_1 \tag{36}$$

$$d_{ij}^{43} + M(1 - w_{43ij}) \geq a_j - v_1 \; + \; b_i - h_1 \tag{37}$$

$$d_{ij}^{14} + M(1 - w_{14ij}) \geq a_i - v_1 \; + \; h_2 - b_j \tag{38}$$

$$d_{ij}^{41} + M(1 - w_{41ij}) \geq a_j - v_1 \; + \; h_2 - b_i \tag{39}$$

Case 3: cells i and j are on two parallel edges. Assume that cell i is on edge 1 and cell j is on edge 3. Any path between them must reach one of the vertical edges of the track first on an horizontal edge, after that it must pass the vertical distance $h_2 - h_1$ and finally it must reach the target cell on the other horizontal edge. If the vehicle starts to move right then the two distances on the horizontal edges are $v_2 - a_i$ and $v_2 - a_j$. If it goes to the opposite direction then the two distances are $a_i - v_1$ and $a_j - v_1$. The vehicle must choose the shorter of the two paths. Thus in that case

$$d_{ij}^{13} = \min\{h_2 - h_1 + 2v_2 - a_i - a_j, h_2 - h_1 - 2v_1 + a_i + a_j\}$$

To describe the minimum function, a new binary variable, say u_{13ij} is introduced which will select the minimum from the two above-mentioned distances, thus d_{ij}^{13} must satisfy the following two inequalities:

$$d_{ij}^{13} + M(1 - w_{13ij}) + Mu_{13ij} \geq h_2 - h_1 + 2v_2 - a_i - a_j \tag{40}$$

and

$$d_{ij}^{13} + M(1 - w_{13ij}) + M(1 - u_{13ij}) \ \geq \ h_2 - h_1 - 2v_1 + a_i + a_j. \tag{41}$$

Notice that if $w_{13ij} = 0$ then none of (40) and (41) is binding. In that case d_{ij}^{13} can be on its lower bound which is 0 as it will be claimed below. As it was mentioned above, the objective function will determine the value of u_{13ij} in a way that d_{ij}^{13} is as small as possible. Formally there are feasible solutions satisfying both (40) and (41) with strict inequality, but they are not optimal. Based on similar analysis the following inequalities must be satisfied:

$$d_{ij}^{31} + M(1 - w_{31ij}) + Mu_{31ij} \ \geq \ h_2 - h_1 + 2v_2 - a_i - a_j \tag{42}$$

$$d_{ij}^{31} + M(1 - w_{31ij}) + M(1 - u_{31ij}) \ \geq \ h_2 - h_1 - 2v_1 + a_i + a_j \tag{43}$$

$$d_{ij}^{24} + M(1 - w_{24ij}) + Mu_{24ij} \ \geq \ v_2 - v_1 + 2h_2 - b_i - b_j \tag{44}$$

$$d_{ij}^{24} + M(1 - w_{24ij}) + M(1 - u_{24ij}) \ \geq \ v_2 - v_1 - 2h_1 + b_i + b_j \tag{45}$$

$$d_{ij}^{42} + M(1 - w_{42ij}) + Mu_{42ij} \ \geq \ v_2 - v_1 + 2h_2 - b_i - b_j \tag{46}$$

$$d_{ij}^{42} + M(1 - w_{42ij}) + M(1 - u_{42ij}) \ \geq \ v_2 - v_1 - 2h_1 + b_i + b_j. \tag{47}$$

and

$$\forall\, i, j, k, l: \ d_{ij}^{k}, d_{ij}^{kl} \geq 0. \tag{48}$$

The objective function is the minimization of the total distance weighted by the flow values, *i.e.* it is

$$\min \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \varphi_{ij} \sum_{k=1}^{4} \left(d_{ij}^{k} + \sum_{l \neq k} d_{ij}^{kl} \right). \tag{49}$$

The model of the closed loop layout with exact distances is the optimization of (49) under the constraints (1)-(48).

4 Preliminary remarks to the numerical solution

It was advantageous to keep the whole layout in the positive quarter of the plane for some practical reasons. Therefore it is claimed that the left lower corner point of the track is the point (40, 40). This point is far enough from the origin that in all benchmark problems all rectangles can lay in the positive orthant. Hence it follows that for all cell i $a_i, b_i \geq 40$.

In all benchmark problems the sizes of all rectangles are integers. If a pick-up point concedes to the (40,40) then the co-ordinates of the corner points of the rectangle are either integers or their fractional part is 0.5. Hence it follows that it is enough to consider the solutions only where the same statement is true for all corner points. Thus the continuous variables of the model have a hidden integrality constraint claiming that the value is either integer, or its fractional part is 0.5. This constraint can be used in the numerical solution at the definition of the branches. It is important to note that claiming integrality in an explicit way makes the problem even harder as some attempts of the solution show.

5 The solution of the problem with 8 rectangles

The computational results related to the 8 cell problem are shortly summarized in this section. All methodological findings are discussed in connection to the 10 cell problem.

The shapes of the rectangles are as follows

Table 4. The rectangles of the 8 cell problem.

Rectangle	The length of the first edge	The length of the second edge	The position of the input/output point (edge)
1	14	10	2
2	10	7	1
3	10	10	2
4	20	15	2
5	18	13	1
6	8	8	2
7	9	6	1
8	15	11	1

The flow is symmetric and has the values as follows:

Table 5. The flow values of the 8 cell problem.

	1	2	3	4	5	6	7	8
1	0	30	12	16	15	0	4	11
2	30	0	6	0	4	3	0	0
3	12	6	0	30	16	85	0	31
4	16	0	30	0	26	60	34	18
5	15	4	16	26	0	30	24	45
6	0	3	85	60	30	0	41	19
7	4	0	0	34	24	41	0	48
8	11	0	31	18	45	19	48	0

The problem was divided into 50 subproblems in a dynamic way. The number of subproblems were not determined *a priori*. The number 50 was obtained by the dynamic process. The expression *dynamic process* means that the subproblems were determined according to the current state of the solution. The best known feasible solution by [Niroomand and Vizvári 2013] is the optimal solution:

Table 6. The optimal closed loop solution of the problems with 8 cells by [Das 1993].

Cell	x_i	y_i	a_i	b_i
1	33	52.5	40	52.5
2	43.5	53	40	53
3	45	35	45	40
4	30	40	40	40
5	69.5	40	63	40
6	44	44	44	40
7	54.5	37	54.5	40
8	55.5	45.5	55.5	40
	h_1	h_2	v_1	v_2
	40	63	40	63

The first branching variable was x_1. If $x_1 < 40$ then rectangle 1 must be on edge 4 of the track and its second (shorter) edge must lie on edge 4. Hence, the only possible value of x_1 is 33 in this case. Now y_1 completely determines the position of cell 1. Therefore y_1 was selected as the second branching variable. The selection of a third branching variable was not necessary. The case $x_1 = 33$ was divided into the subcases as follows: (1) $y_1 \leq 43$, (2) $y_1 = 43.5$, (3) $y_1 = 44$, (4) $44.5 \leq y_1 \leq 46$, (5) $y_1 = 46.5$, (6) $y_1 = 47$, (7) $y_1 = 47.5$, (8) $48 \leq y_1 \leq 48.5$, (9) $49 \leq y_1 \leq 50$, (10) $y_1 = 50.5$, (11) $y_1 = 51$, (12) $y_1 = 51.5$, (13) $y_1 = 52$, (14) $y_1 = 52.5$, (15) $53 \leq y_1 \leq 56$, (16) $y_1 \geq 56.5$.

If $x_1 = 40$ then the only possible value of a_1 is also 40. Edge 4 of the track is on the vertical line having y co-ordinate 40. As the track must not cut into the cell, cell 1 must be either just above, or just under edge 4. Thus the center point of cell 1 is on this vertical line, *i.e.* $a_1 = 40$.

If $x_1 \geq 40.5$ then there are seven possible types of positions of cell 1. It can be inside the track on any of the edges of the track and it can be outside on edges 1, 2, and 3. If cell 1 is on edge 1 or 3 then $a_1 = x_1$. If it is on edge 2 then either $a_1 = x_1 - 7$, or $a_1 = x_1 + 7$. If it is on edge 4 then the only option is $a_1 = x_1 - 7$. The next case which was considered, is $40.5 \leq x_1 \leq 42.5$. Then only $a_1 = x_1$ is possible. The statement is true even for some greater x_1's.

However the case $x_1 = 43$ was divided into the subcases according to x_5 as follows: (i) $x_5 \leq 56$, (ii) $56.5 \leq x_5 \leq 62$, (iii) $x_5 = 62.5$, (iv) $63 \leq x_5 \leq 69$, (v) $x_5 = 69.5$, (vi) $x_5 = 70$, (vii) $x_5 \geq 70.5$. Finally the case $x_{\geq}43.5$ was covered by 13 subcases as follows: (a) $x_1 = 43.5$, (b) $44 \leq x_1 \leq 45$, (c) $x_1 = 45.5$, (d) $x_1 = 46$ and $y_1 = 33$, (e) $x_1 = 46$ and $y_1 \geq 33.5$, (f) $46.5 \leq x_1 \leq 57$ and $x_4 = 30$, (g) $46.5 \leq x_1 \leq 57$ and $x_4 \geq 30.5$, (h) $x_1 = 57.5$, (i) $x_1 = 57.5$, (j) $58 \leq x_1 \leq 58.5$, (k) $59 \leq x_1 \leq 60$, (l) $60.5 \leq x_1 \leq 70$, (m) $x_1 = 70.5$, and (n) $x_1 \geq 71$. Notice that the pick-up point of cell 4 is on the second edge. Hence the center point is away from the track by half of the first edge, *i.e.* by 10. Thus 30 is the smallest possible value of both x_4 and y_4.

All the above mentioned consequences, for example if $x_1 < 40$ then $x_1 = 33$, was proved by adding the opposite constraint to the problem and obtaining a subproblem without feasible solutions.

The hidden integrality property of the continuous variables was also utilized as can be observed from the definition of the subcases.

6 The current state of the solution of the problem with 10 rectangles

6.1 Technical data and details

As the solution of the problem with 8 rectangles was easy, the author hoped that the next problem with 10 cells can be solved also within a reasonable time. The sequence of runs started on March 16, 2013 and is still going on. The number of registered subproblems is more than 2300 (September 4, 2014). The tool of optimization is XPRESS solver in IVE environment. Two laptops were used. One is a Lenovo laptop with an Intel(R) Core(TM) i3 dual core processor @2.1GHz. The other one is a Toshiba laptop with an Intel(R) Core(TM) i5 dual core processor @2.27 GHz. Both computers are x64 architecture and have 4096 Mb memory.

The exact mathematical model discussed in Section 3 has 1135 binary and 944 continuous variables, 2220 linear constraints not including the nonnegativity of the variables. When a subproblem is defined then further constraints are added, however there are no further variables.

The shapes of the rectangles are as follows

Table 7. The rectangles of the 10 cell problem.

Rectangle	The length of the first edge	The length of the second edge	The position of the input/output point (edge)
1	15	10	1
2	12	8	2
3	15	12	2
4	9	9	2
5	9	6	1
6	25	18	2
7	20	16	2
8	8	5	1
9	11	11	2
10	19	13	1

The flow is symmetric and has the values as follows:

Table 8. The flow values of the 10 cell problem.

	1	2	3	4	5	6	7	8	9	10
1	0	10	35	18	25	45	11	0	31	16
2	10	0	0	10	18	0	4	11	30	0
3	35	0	0	16	0	53	0	30	61	8
4	18	10	16	0	18	25	11	8	29	19
5	25	18	0	18	0	63	30	16	0	21
6	45	0	53	25	63	0	0	40	13	14
7	0	11	30	8	16	40	0	0	0	11
8	0	11	30	8	16	40	0	0	0	11
9	31	30	61	29	0	13	17	0	0	35
10	16	0	8	19	21	14	79	11	35	0

Table 9. The best-known closed loop solution of the 10 cell problem with objective function value 12110.5.

Problem	C10			
Cell	x_i	y_i	a_i	b_i
1	54.5	40	49.5	40
2	55.5	51.5	49.5	51.5
3	32.5	46.6	40	46.5
4	44.5	58.5	40	58.5
5	44.5	43	44.5	40
6	40.5	27.5	40.5	40
7	59.5	63.5	49.5	63.5
8	42.5	50	40	50
9	34.5	58	40	58
10	40	70	40	63.5
	h_1	h_2	v_1	v_2
	40	63.5	40	49.5

It is easy to see that if all cells are on a single line and there is no coordinate of the pick-up points on the line less than 40, then all cells can be placed on the line up to 140. Thus it is enough to consider the cases where all coordinates of the pick-up points are between 40 and 140.

In the analysis in this section, the typical phenomena and behaviors are discussed. There is no mathematical theorem that all cases behave like that.

As it is mentioned in the introduction, this project is not the first attempt to solve the 10 cell problem exactly. There were attempts to solve the exact model even earlier and it became clear that the problem in its original form is too difficult. In these attempts, the solution of the exact model was interrupted and started again with different parameters of the XPRESS solver. This method which can be considered as a new kind of metaheuristics, uncovered several good feasible solutions. At the beginning of this project the objective function value of the best known feasible solution was 13984.5 [Niroomand and Vizvári 2013].

As the solution process is not completed yet, all statistical data provided below concern to the first 2300 subproblems. The optimization of the 2300-th subproblem finished on September 4, 2014. The total CPU time is approximately 44,688,000 seconds which is 1 year and 152 days. The longest individual CPU time was 1,559,368 seconds which is 18 days. The number of 0 CPU times is 344, the average is 19,550. 86.78 percent of the subproblems were fathomed in the form as they are defined. These subproblems used the 73.89 percent of the total CPU time. The difference in the percentages is understandable as it takes a relatively long time to recognize that the time needed to reach the target value is hopelessly long.

Cell 6 has the greatest flow value which is 253. Therefore the position of cell 6 is critical. Several parallel branches are under investigation. In all of them the first branching variable is y_6. Some solved cases are below. In all cases only the branching variables are mentioned and the consequences are not.

$$y_6 \leq 39.5,\ y_{10} \leq 39.5$$

$$y_6 \leq 39.5,\ y_{10} = 40,\ y_5 \leq 45$$

$$y_6 \leq 39.5,\ x_6 \geq 40.5,\ y_{10} = 40,\ y_5 \geq 45.5$$

$$y_6 \leq 39.5,\ x_6 \geq 40.5,\ y_{10} \geq 40.5,\ x_{10} \geq 40.5,\quad \text{cell 10 is on track edge 4}$$

$$y_6 \geq 40.5,\ x_6 \leq 39.5,\ y_{10} \leq 40$$

$$y_6 \geq 40.5,\ x_6 \leq 39.5,\ y_{10} \geq 40.5.\ x_{10} \leq 40$$

$$y_6 \geq 40.5,\ x_6 \leq 39.5,\ y_{10} \geq 40.5,\ x_{10} \geq 40.5,\quad \text{cell 10 is on track edge 3 or 4}$$

6.2 Human work

The definition of the subproblems, using of XPRESS solver to solve them, and the evaluation of the state of the solution, *i.e.* to decide if the solution will be completed in reasonable time, was made by a human being, more precisely by me, the author of this paper.

Errors are always inherent in non-automated human work. The most important and most serious error occur if a problem is divided into subproblems, and the defined subproblems do not cover the problem completely. To avoid such cases the administration of the subproblems has two levels. The lower level is a protocol fixing the facts of each attempts of a solution of a subproblem. These facts are as follows:

- the number, *i.e.* the name, of the subproblem,
- the extra constraints defining the subproblem,
- whether or not the solution was interrupted (notice that it can be interrupted without any loss if the lower bound exceeded the target value),
- the achieved upper bound, *i.e.* the objective function value of the best feasible solution,
- the achieved lower bound of the optimal value provided by XPRESS,
- the date when the solution was finished,
- the name of the output file,
- the elapsed CPU time.

On the second level of the administration these data are evaluated and determined:

1. Which larger cases are solved completely? Such a larger case typically consists of tens of successfully solved subproblems. Larger cases can be united to even larger cases.
2. Which subproblems are missing?

Some other errors in human work are: repeating cases, and defining subproblems by contradicting extra constraints.

Human work has advantages as well. It is easy to take into consideration the meaning of the variables and their effect on each other. This is the underlying factor of the above-mentioned high percentage of the successfully fathomed subproblems. A significantly lower percentage can be expected in an automated system.

Another factor of human work is that the allowed CPU times reflect how busy the operator is. Longer times are accepted during nights and working times filled with other jobs. However it can be advantageous if a difficult subproblem is solved even on the expense of a relatively longer CPU time.

As a matter of fact, this work creates a lifestyle. Something always runs and it must be checked.

7 Lessons for MILP

7.1 Properties of Branch and Bound

7.1.1 Sub-branching

The reason why many methods define subproblems, *i.e.* they partition the original problem into smaller parts, is that it is easier to solve the smaller problems. An example of this is subproblem 1068. At that time the best known feasible solution already had been obtained.

Its objective function value, *i.e.* 12110.5, was the *target value*. If the lower bound of the optimal objective function value exceeded the target value then the subproblem was fathomed as it could not contain the optimal solution. Subproblem 1068 is defined by the extra constraints:

$$y_6 \leq 39.5, \quad x_6 \geq 40.5, \quad \lambda_{36} = 1, \quad b_6 = 40, \quad h_2 \geq 55.5, \tag{50}$$

$$y_{10} = 40, \quad y_5 \geq 45.5, b_5 \geq 45.5, \quad h_2 \geq 67.5, \quad 40.5 \leq x_5 \leq 43. \tag{51}$$

Notice, that there are two lower bounds for h_2. Both of them are consequences of previous constraints and/or fathomed subproblems. In this case, consequences are the constraints as follows:

$$\lambda_{36} = 1, \quad b_6 = 40, \quad h_2 \geq 55.5, y_5 \geq 45.5, b_5 \geq 45.5, \quad h_2 \geq 67.5.$$

For example $y_6 \leq 39.5$ means that cell 6 is under the track. If $x_6 \geq 40.5$ then its center point is right from the lower left corner of the track. Hence, it must be on edge 3 of the track which is equivalent to $\lambda_{36} = 1$. Variables in other constraints are free variables and other options of the same variable are still to be investigated. If a consequence has the form of an inequality then it is still possible to define further subproblems based on that variable.

After 48007 seconds of CPU time the lower bound of the optimal objective function value in the subproblem 1068 was still only 11195.05. Even a very rough linear estimation showed that the lower bound will reach the the target value at time 123074. The estimations are discussed below. Linear estimation always underestimates the time still needed. Therefore the solution of the subproblem was stopped and it was partitioned into the subproblems:

Table 10. The sub-branches of branch (50)-(51) according to the values of variable x_5

New extra constraints	CPU time
$x_5 = 40.5$	3890
$41 \leq x_5 \leq 42$	14084
$x_5 = 42.5$	20532
$x_5 = 43$	21478
$x_5 = 43, \ \lambda_{2,10} = 0$	5261
$x_5 = 43, \ \lambda_{2,10} = 1$	14796
Total CPU time	46091

To avoid long CPU times, it is necessary to estimate the time that the process will hit the target value. If the lower bound of the optimal objective function value is investigated as the function of time then its shape is an increasing concave function in a *long time interval*. Thus the linear estimate is on the line of a chord of the function and therefore it underestimates the remaining running time. A much better approximation can be obtained in the form

$$f(t) = pt^q, \tag{52}$$

where t is the time, p is a positive coefficient and q is in the open interval $(0, 1)$. A partial and not complete explanation is that the size of the branch and bound tree is increasing until the very end of the particular run. It is always so if the optimal value of the branch is significantly greater than the target value. Figure 1 shows the increase of the lower bound as the function of time. The figure belongs to subproblem 1596 which was solved completely. This subproblem had the special property that its B&B tree is small and therefore XPRESS executed garbage collection rarely. Garbage collection is discussed in section 7.4. The increase of the best bound in a *short time interval* is discussed also in section 7.4 as they are related to one another.

7.1.2 Consequences

In integer programming, the branches not containing integer feasible solutions are important. Three types of such branches can be observed:

- *Trivially infeasible branch.* In some cases, XPRESS is able to show immediately that the branch does not have any integer feasible solution. The registered CPU time is 0. Most of the cases belonging to this category appear when the extra constraints imply a mandatory fixing of a binary variable. Assume that the variable u must be fixed on value δ. If the extra constraint $u = 1 - \delta$ is added to the problem then there is no feasible solution. In certain sense, this procedure is a mathematical proof for the fact that the $u = \delta$ fixing is mandatory. Why is it necessary to apply this procedure? The human inference can be false from time to time. The procedure either shows that something is wrong or approves the conclusion. Fixing a single variable is not the only case which belongs to this category. It can be shown in the same way that the sum of a couple of not completely independent variables cannot have some values.

- *Almost trivially infeasible branch.* The only difference compared to the previous case is that a very short CPU time is needed to show the non-existence of the integer feasible solutions. In that case the lower bound of the optimal objective function increases very fast.

- *Non-trivially infeasible branch.* The CPU time which is needed to show the infeasibility, is approximately the same as to fathom a feasible branch. An important difference is the behavior of the lower bound. It starts on a lower value than at the feasible branches. However the curvature of the lower bound function is smaller, *i.e.* the speed of the increase becomes higher after a while, see Figure 2 in the Appendix.

All three cases produce a result which is called *consequence* in the implicit enumeration method. A consequence is a statement or constraint on the variables which follows from other constraints. Branch and bound also uses consequences in an explicit or implicit way. In the latter case, B&B need a proof in all cases when the consequence is used. This means that the proof is created in the solution of a problem many times needing a lot of CPU time. It is the reason that there are cases such that by adding superfluous constraints to the problem, the solution time drops. By adding the consequences to the extra constraints, a similar phenomenon can be observed here, too. Notice, that by adding consequences to the extra constraints the branch does not change in the sense that the set of integer feasible solutions remains the same. However its description becomes more accurate.

7.1.3 Time limit

The *time limit* is the time allowed for the solution of a subproblem. Time limits of different subproblems are different and this is not only because a human being cannot always check if the lower bound achieved the target value.

The difficulty of the subproblems is not the same everywhere. If there are plenty of good but not optimal solutions in a region of the set of feasible solutions then the set of subproblems are very likely hard in this region. Thus, longer CPU times are acceptable. If a subproblem contains a new best feasible solution then it must be solved exactly. In that case a long CPU time can be expected. It is true even in the case where the subproblem is still divided into several subproblems. The one which contains the new best solution can need a long solution time.

The restrictions on different variables may amplify each other's effect. For example, the last free variable which is restricted in subproblem 2307 is v_2. All cases with $v_2 \leq 55$ are fathomed under the previous extra constraints of the branch. However it became clear that the further increase of v_2 is not realistic as the case $55 \leq v_1 \leq 60.5$ had to be interrupted after 46545 seconds at lower bound was growing very slowly and was only 11460.4 and the case $v_2 = 55$ alone took 28719 seconds. The next idea was to restrict the number of cells on the second edge of the track. In six steps it was shown that the number of cells on the second edge can be at most 4. After these actions the branch represented by subproblem 2307 was completely fathomed by the following cases: (a) $v_2 = 55.5$, (b) $56 \leq v_2 \leq 60.5$, (c) $61 \leq v_2 \leq 67.5$, (d) $68 \leq v_2 \leq 80.5$, (e) $81 \leq v_2 \leq 140$.

7.2 Possible Actions

The enumeration tree can be more flexible than the usual one. For example the structure of the last example is this:

$$w \geq \alpha \text{ implies } \mathcal{A}, \quad \mathcal{A} \text{ implies } w \geq \beta$$

where $\alpha < \beta$. Although the implication may need some extra effort to eliminate further cases, the procedure is a correct proof for $w \geq \beta$.

The order of the branching variable can be handled in a flexible way. For example the branching variables are given in the branching order for the solved cases mentioned at the end of subsection 6.1. This means that the second branching was made by variable x_6 in the third solved case. To obtain

$$y_6 \leq 39.5, \; y_{10} \leq 40$$

as a solved case it would be enough to solve the case

$$y_6 \leq 39.5,\ y_{10} = 40,\ y_5 \geq 45.5,\ x_6 \leq 40.$$

The advantage is that there are three restricted free variables before the restriction of x_6 which may imply consequences such that they are not true when the restriction of x_6 is in the second position.

7.3 Desired Properties of Solvers

A branch which is not fathomed or branched yet is called *active branch*.

The best bound is the minimal bound on an active branch in the case of minimization. Other branches have higher bounds. If the bound of a branch is higher than the target value then this branch is actually fathomed. Thus it is desirable that the solver have a parameter for the target value. In this way a lot of CPU time can be saved. It is especially true if the subproblem is in a region where the optimal values are significantly above the target value.

The used CPU time depends very much on the size of the B&B tree. Thus, a branching strategy which controls this size is also desirable. One solution could be that if the number of active branches exceeds an *a priori* defined upper limit then the program switches to LIFO strategy and decreases the number of active branches until it falls under another *a priori* defined lower limit.

7.4 Properties Related to XPRESS

The version of XPRESS which are used is 7.1. It is not the latest version of XPRESS. However version 7.6 is significantly slower on this problem. Thus, it was decided that version 7.1 is kept in the project.

Garbage collection is an important function of B&B. It means that if the method runs out the storage capacity, *i.e.* there is a shortage of memory or disc area, then the program deletes the data of branches such that all their subbranches are fathomed, however the description of the branch is still in the B&B tree. XPRESS saves storing area in this way if it is necessary. However the garbage collection is a time consuming operation and optimization is stopped during the process. When the optimization procedure goes ahead, the time spent for garbage collection becomes longer. The horizontal segments on Figure 3 are the garbage collection times.

Figure 4 shows the average time needed to generate 100 new branches. The figure also belongs to subproblem 1596. The vertical peaks are created by garbage collection as their time is also included. The peaks are truncated at 8. One iteration is the generation of 100 new branches. The figure shows the moving average of length 11 of the elapsed time by one iteration. Notice that the function is increasing. This phenomenon can be seen even better on Figure 5. Here, the iteration time is shown as the function of the number of active nodes. This is the reason why the function has two branches; as the number of active nodes starts with zero and after a maximum it goes back to zero as the subproblem was solved completely. The lower branch belongs to the beginning and the upper branch to the second half of the particular run. Thus as one goes towards zero on the second branch, when goes ahead in time. An important question is why is no drop rather it increases? No information is available on what XPRESS does to the B&B tree in the garbage collection. This figure and the underlying computational experiences indicate that the B&B tree obtained after garbage collection is not as simple as it would be if no garbage collection were needed because of the available large storing capacity.

There is an interesting phenomenon which can be observed several times. There are periods of the branch and bound enumeration when the generation of the 100 new branches decreases the number of active branches by 100. The word "period" relates to the fact that it happens not only once but several hundred times consecutively. The generation of 100 branches in such a period takes a short time. The best bound is constant while the number of active nodes decreases.

Another observation is related to finding new, even better integer feasible solutions. XPRESS distinguishes solutions which are the optimal integer solution of a branch from those obtained by heuristic methods. Interestingly, if the optimal integer solution of a branch is obtained, then in almost all cases several even better solutions are found by heuristic methods within a few seconds.

8 Conclusions and future work

The bench mark problem with 8 cells of the rectangular layout problem was solved in the closed loop case. A significant progress was made in the solution of the bench mark problem with 10 cells including the uncovering of a very good feasible solution.

A method for human work was elaborated to solve large scale and difficult problems dividing them into several subproblems. It is necessary to automate this procedure in the future as human beings make error frequently and human work is slow. Such a system may consist of a master computer and several slave computers. Slave computers solve the subproblems assigned to them by the master computer and communicate to the master on the current state of the solution. The master computer

- makes the administration of the solved and unsolved subproblems,
- determines the new subproblems,
- estimates the CPU times needed by the slave computers and if it is necessary then stops the slave computer and defines smaller branches,
- if it has enough time may solve subproblems as well.

This structure has different requirements as the traditional single optimization structure. It is wished that the features of the solvers are adjusted to the new structure.

References

[Chae and Peters 2006] Chae, J. and Peters, B.A., A simulated annealing algorithm based on a closed loop layout for facility layout design in flexible manufacturing systems. Int. J. Prod. Res., 44, 2006, 2561-2572.

[Das 1993] Das, S.K., A facility layout method for feasible manufacturing systems, International Journal of Production Research, 31(1993), No. 2, 279-297.

[Kim and Kim 2000] Kim, J.-G. and Kim, Y.-D., Layout planning for facilities with fixed shapes and input and output points. Int. J. Prod. Res., 38, 2000, 46354653.

[Kovács and Vizvári 2014] Gergely Kovács, Béla Vizvári, Exact models for Open Field Layout Problem with l_2 and l_1 Distances, RUTCOR, Rutgers University, Research Report, 1-2014, http://rutcor.rutgers.edu/pub/rrr/reports2014/01_2014.pdf

[Lasardo and Nazzal 2011] Lasardo, V., Nazzal, D., Design of a manufacturing facility layout with a closed loop conveyor with shortcuts using queuing theory and genetic algorithm. Proceedings of the 2011 Winter Simulation Conference, 2011, 1964-1975.

[Luggen 1991] Luggen, W.W., Flexible Manufacturing Cells and Systems, Prentice-Hall, Englewood Cliffs, 1991.

[Niroomand and Vizvári 2013] Sadegh Niroomand, Béla Vizvári, A mixed integer linear programming formulation of closed loop layout with exact distances, Journal of Industrial and Production Engineering, 2013, Vol. 30, No. 3, 190–201, http://dx.doi.org/10.1080/21681015.2013.805699

[Rajasekharan *et al.* 1998] Rajasekharan, M., Peters, B.A. and Yang, T., A genetic algorithm for facility layout design in flexible manufacturing systems. Int. J. Prod. Res., 36, 1998, 95-110.

[Samarghandi *et al.* 2010] Samarghandi, H., Taabayan, P., Firouzi Jahantigh, F., A practical swarm optimization for the single row facility problem. Computer & Industrial Engineering, 58, 2010, 529-534.

[Tavakkoli-Moghaddam and Panahi 2007] Tavakkoli-Moghaddam, R., Panahi, H., Solving a new mathematical model of a closed-loop layout problem with unequal-sized facilities by a genetic algorithm. Proceedings of the 2007 IEEE IEEM, 2007, 327-331.

[Ting, J.-H., Tanchoco 2001] Ting, J.-H., Tanchoco, J. M. A, Optimal bidirectional spine layout for overhead material handling systems. IEEE Transactions on Semiconductor Manufacturing, 14(1), 2001, 57-64.

Appendix: Figures

Figure 1. The approximation of the increase of the best bound by a power function

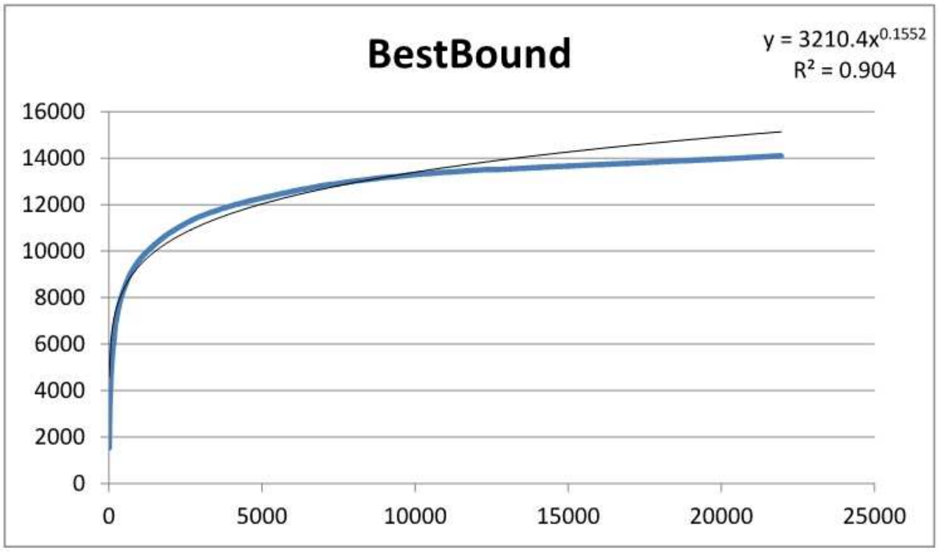

Figure 2. The increase of the best bound in the case of an infeasible subproblem.

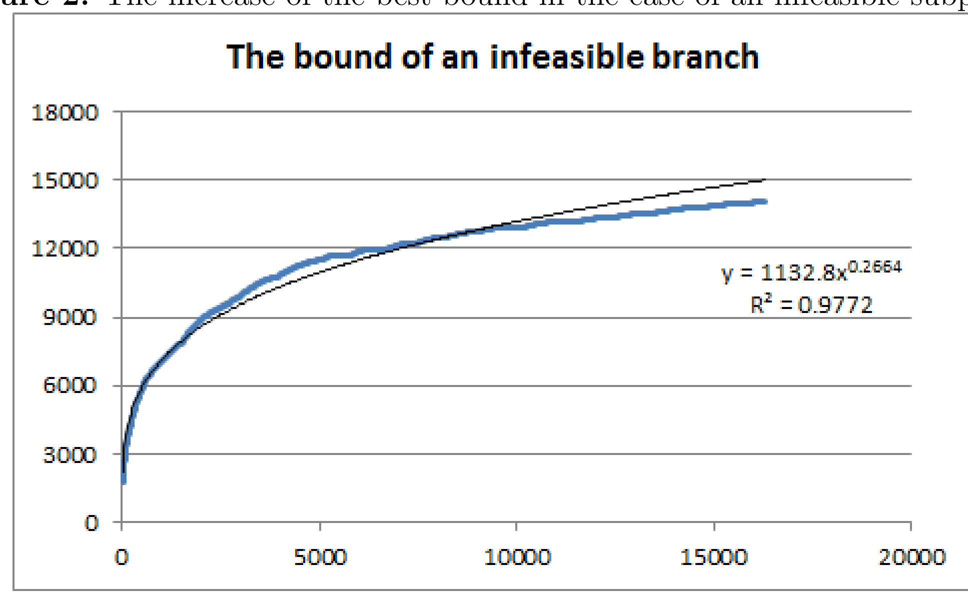

Figure 3. The increase of the best bound in a short time interval. The horizontal lines are the periods of garbage collection.

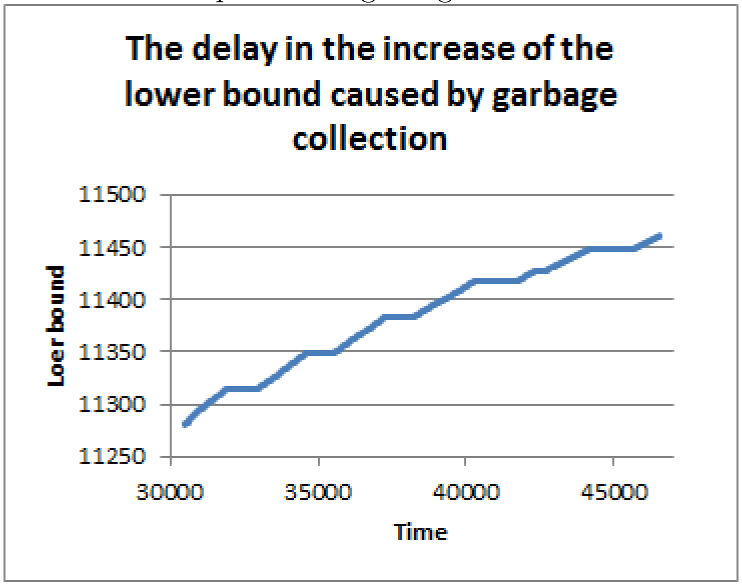

Figure 4. The average time needed to generate 100 new branches as the function of the iteration number. An iteration is the generation of 100 new branches.

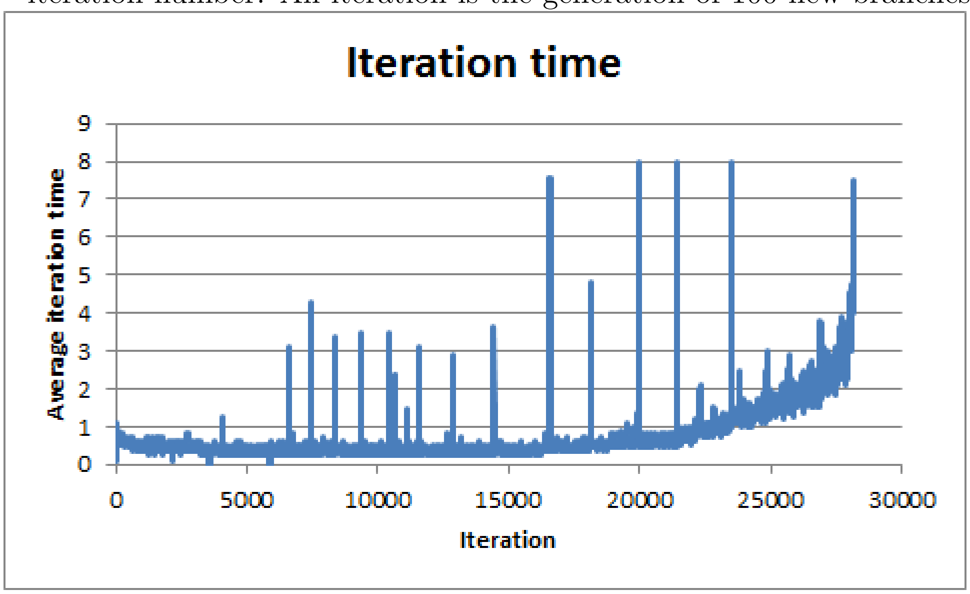

Figure 5. The average time needed to generate 100 new branches as the function of the number of active nodes.