Int j simul model 15 (2016) 2, 275-287
ISSN 1726-4529 Original scientific paper

OPTIMISATION OF MACHINE LAYOUT USING A FORCE GENERATED GRAPH ALGORITHM AND SIMULATED ANNEALING

Kanduc, T.[*,**] & Rodic, B.[*]
[*] Faculty of Information Studies, Ljubljanska 31A, 8000 Novo mesto, Slovenia
[**] National Institute of Advanced Mathematics (INdAM), University of Florence, Italy
E-Mail: tadej.kanduc@unifi.it, blaz.rodic@fis.unm.si

Abstract
This paper presents a novel heuristic method for machine layout optimisation, developed in the course of an internal factory logistics optimisation project. The method is developed from a force-directed graph drawing algorithm, and integrates random permutations using simulated annealing to avoid local minima. The method was verified and validated with a discrete event simulation (DES) model of a furniture development factory consisting of 140 machines. The DES model was developed for manufacturing system analysis as well as design and testing of optimisation methods. The main optimisation goal was reduction of transport costs by minimising the total distance the products travel between the machines. The optimisation problem extends the quadratic assignment problem (QAP) by allowing arbitrary granularity of locations, facility sizes and fixed facilities. The resulting method can be used to solve a wider range of problems by altering the optimisation function or adding new feasibility conditions.
(Received in June 2015, accepted in December 2015. This paper was with the authors 1 month for 2 revisions.)

Key Words: Layout Optimisation, Heuristics, Discrete Event Simulation, Force-Directed Graphs

1. INTRODUCTION

This paper presents a machine layout optimisation method developed within an internal logistics optimisation project at a manufacturer of specialised furniture. The method was developed and tested using a *discrete event simulation* (DES) model of the furniture factory. We examined the influence of layout optimisation on the total product travel distance, total order production time and other factory model statistics.

The machine layout optimisation problem is related to the Quadratic Assignment Problem – a NP-hard optimisation problem, but requires more flexibility in the facility layout. To be able to solve it, we have developed a novel heuristic method that is based on force-directed graph drawing algorithms and simulated annealing. The optimisation method significantly reduces total product travel distance.

The structure of the paper is as follows: In this section we present the manufacturing process in the company, highlight the optimisation problem and our developed optimisation method, and give a review of literature that investigates related problems. In Section 2 we describe the methodology and the resulting novel layout optimisation method. Section 3 contains comparison of related optimisation methods, results of the optimisation project and discussion of the project results and future plans.

1.1 Problem situation

The client company has been manufacturing furniture for more than half a century. During that time, customer demands changed while size of orders and quality requirements grew. New machines were added to the factory as needed and placed within available floor space. Machine placements were determined by experience of foremen and typically never changed. No systematic analysis and optimisation of factory floor machine layout has been done by the company. Section of the layout is shown in Fig. 1 (the figure shows machines, input and output pallets).

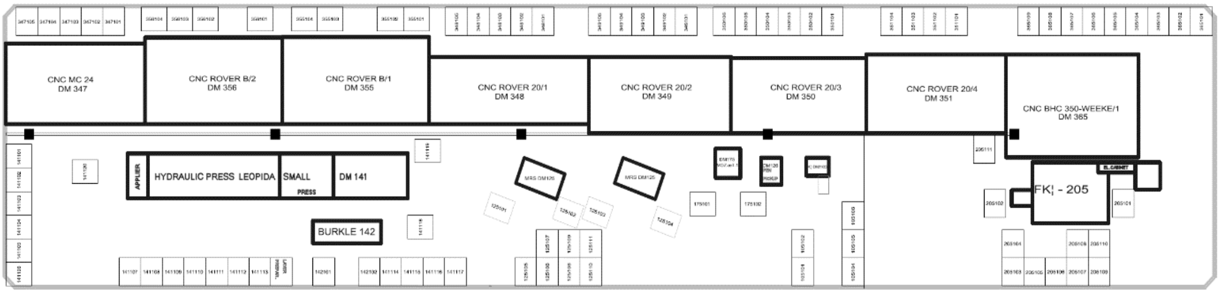

Figure 1: Section of the factory floor.

Our task in the project was to develop a better factory floor machine layout that would fit the current production needs and order projections for the next time period. Repositioning of the machines is to be done during the upcoming renovation of the factory. Most of the machines have no specific location restrictions, allowing us to move them freely.

Approximately 140 machines are located in the factory. The factory makes more than 30,000 products and components with a distinct *bill of materials* (BOM) and manufacturing process referred to as “technical procedure” which specifies appropriate machines for separate operations as well as machine set-up and operation times.

Most machines are multi-purpose (e.g. CNCs), which allows for flexibility in job scheduling, fewer bottlenecks and higher machine utilisation. Our DES model uses the data on equivalent machines (i.e. machine sets) to dynamically select the least utilised machines as jobs (series of products) move from machine to machine.

1.2 Minimising the total travel distance of products

First goal in the project was to reduce the total travel distance of the jobs that travel through a series of machines. This optimisation problem was defined as follows:

$$\min_{\{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N\}} \left(\sum_{i,j=1,\ i \neq j}^{N} f_{ij} \cdot d(\boldsymbol{x}_i, \boldsymbol{x}_j) \right), \tag{1}$$

with f_{ij} representing the volume of products moving from machine m_i to machine m_j, $i, j = 1, \dots N$. Vectors $\boldsymbol{x}_i$, $\boldsymbol{x}_j$ represent positions of machines m_i and m_j, respectively, and d is a distance functional.

By restricting the machine locations to a discrete grid, the optimisation problem can be reduced to the familiar *Quadratic Assignment Problem* (QAP). As the QAP problem is NP-hard, non-heuristic optimisation methods can find the optimum in a reasonable time only if there are very few machines, e.g. $N < 30$. The factory in question however has $N = 140$ and exact optimisation methods are impractical. Therefore a suitable heuristic method that produces sub-optimal but satisfactory results is needed.

As an alternative to QAP algorithms, we have developed a novel method based on force-directed graph drawing algorithms. These algorithms, also referred to as 'spring embedders', are methods for visual representation of graphs (networks) [1]. The basic idea of these methods is to apply attractive and repulsive forces between the nodes of the graph. As forces are applied to the graph, the nodes start to move toward a configuration that has a local minimum of overall energy.

The developed method is more flexible than QAP algorithms as it allows arbitrary granularity of locations, arbitrary facility sizes, custom criteria optimisation function. In our method, nodes of the graph represent facilities in QAP terminology and edges represent overall flow of products between the corresponding machines. By adding attractive forces $f_{ij} \cdot d(\boldsymbol{x}_i, \boldsymbol{x}_j)$ to edges e_{ij} and repulsive forces between the nearby nodes, the overall system tends to minimise energy similar to the sought functional $\sum_{i,j=1,i\neq j}^{N} f_{ij} \cdot d(\boldsymbol{x}_i, \boldsymbol{x}_j)$.

To avoid premature convergence in local minima of the functional, random permutations of neighbouring nodes' positions are applied via simulated annealing technique. 'Cooling' of the system results in reduction of the number of neighbouring candidate nodes for permutations and in smaller number of accepted permutations that increase value of the optimisation functional. Details of the method are presented in Sections 2.5-2.7.

1.3 Review of related research

Simulation and modelling are frequently applied for the scenarios evaluations [2-4]. In the field of factory floor layout optimisation, paper [5] posits that manufacturers with diverse product portfolio need new methods for floor layout development that would allow more flexibility, modularity and easy reconfiguration of layout. Debevec et al. [6] describe a new method (PoVEIR) aimed at optimisation of manufacturing processes in SMEs that is based on a DES simulation model. Floor layout optimisation as a Combinatorial Optimisation Problem is NP-hard, prompting the use of heuristic methods such as evolutionary optimisation and simulated annealing [7]. An original particle swarm optimisation method for intelligent design of unconstrained layouts was proposed by Ficko et al. [8].

Further reading on floor layout optimisation includes [9-11]. Krishan et al. [9] describe a novel facility layout design model aimed at material handling cost minimisation. Solutions based on genetic algorithms (GA) can be used to respond to the changes in product design, product mix and order volume in a dynamically evolving manufacturing system [10][11].

Quadratic assignment problem has been extensively studied in the last few decades (see [12-16] and the references therein). The problem was first introduced in [17] and proven to be NP-hard in [18]. Exact methods are ineffective for larger problems and heuristics that give suboptimal solutions must be used. There exist several open source QAP heuristic method implementations in different programming languages (see [19-21] and QAPLIB website (http://www.opt.math.tu-graz.ac.at/qaplib/codes.html). The drawback of these methods is that they require a rigid definition of problems: no fixed locations of facilities and no different sizes of facilities or alterations to the optimisation function.

Force-directed graph drawing methods are very popular in the mathematical community (see [1, 22, 23] and contained references). An early and well-known method was described by Fruchterman and Reingold in 1991 [23].

Simulated annealing is a widely used metaheuristic optimisation technique where extensive search for the optimal solution is not feasible [24-26]. Premature convergence in poor local minima is prevented by allowing local refinements of the existing solution that increase value of the minimisation function in a specified manner. The method was first described in [27].

2. METHODOLOGY

The optimisation of floor layout was performed by working with the experts at the client, which prepare new layout proposals and evaluated layouts generated by our novel algorithm. Optimisation function was implemented with the DES model, which we used to verify and test new human-generated and algorithm-generated layouts. Fast verification and validation of layouts was made possible by the automated model construction algorithm described in [28].

2.1 Modelling the manufacturing process

Each product has its own *technical procedure* that describes how the product is manufactured and on which machines. The products are divided into *main products* and *semi-finished products* (components). Semi-finished products are integrated onto a main product at some production stage.

For the purpose of optimisation we have modelled the 140 main machines and transport paths. The model parameters included machine station surface area, input and output pallet positions, queue capacity, machine operations and set-up times, sets of equivalent machines and transport cart speed.

Set of equivalent machines contains machines that are equally able to perform a specific operation. Products are manufactured in job lots or series. After the operations for a job lot on a machine are complete, the entire lot/series in moved to the next machine. As series are small, we did not limit the capacity of carts.

2.2 Implementation in modelling and simulation tool

We modelled the manufacturing system with a multi-method modelling and simulation tool AnyLogic that implements discrete event simulation (DES), system dynamics (SD) and agent based modelling (ABM) methodologies. The simulation model allows us to monitor various manufacturing process statistics and to better understand the manufacturing system by discovering rules and connections in the manufacturing system. The model was verified and validated by comparing the simulation results (e.g. manufacturing time, machine utilisation) using synthetic and real historical order data prepared by the company planners with the statistics from the set of orders from the previous year. Model parameters were obtained from the order scheduling database (Preactor) and contain discrete machine set-up and operation times per product type. The model is therefore deterministic and requires only one simulation run per scenario (combination of layout and set of product orders).

Fig. 2 contains the DES implementation of a single machine in AnyLogic. Carts containing job lots enter the machine at the *In* node and are analysed by Java code at node *cartSink* which releases carts and transfers products to the *productSource* node. They proceed to the *productOrSemifinished* node which sorts them according to type. Nodes *sink1* and *sink2* represent input pallets for products and components. Nodes *setUpMachine* and *machineDelay* model machine set-up time and operation. Output pallet is modelled by the *waitForWholeSeries* node. As operations include painting and varnishing the *dryingDelay* node models the time needed for the lot to dry. Whole job lots are transferred to carts at *cartSource* and moved to next machine via node *moveCartTo* which is connected to the *Out* mode (machine exit).

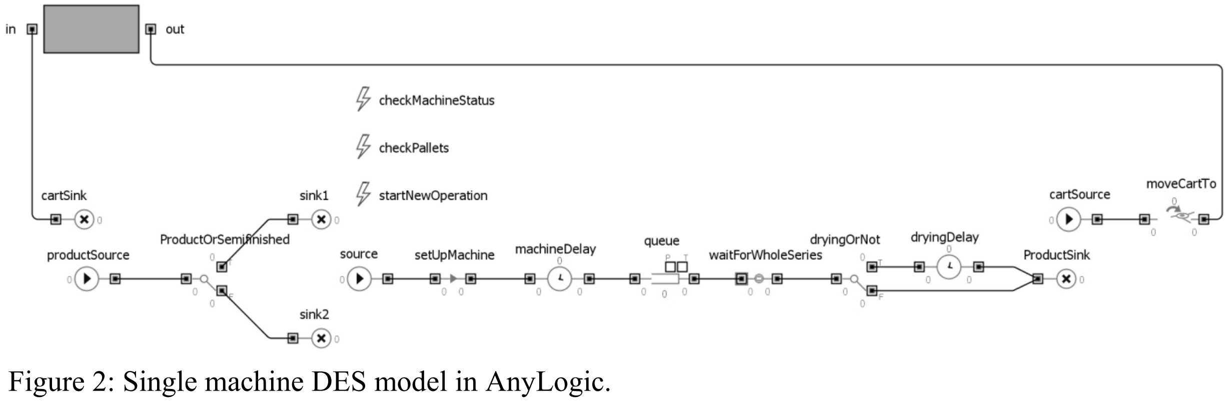

Figure 2: Single machine DES model in AnyLogic.

2.3 Optimisation problem and methods

The optimisation method should generate the machine layout on the factory floor which would minimise total travel distance of the products during the manufacturing. The generated layout is not final as fine adjustment of machine positions to existing infrastructure has to be performed by the client experts.

Formally, the optimisation method manipulates positions of nodes in the network. The nodes represent the machines while the edges are weighted according to product volume flow. Manhattan distances between the machines are used to calculate product travel distance as an acceptable approximation. We tested several layouts and Manhattan distances were consistently shorter than transport routes distances by approximately 20 %. Accurate distance calculation would require designing the transport routes for every layout version, which would significantly prolong the optimisation process.

We have implemented the optimisation method in AnyLogic with a system dynamics (SD) model based on force-directed graph drawing algorithms and simulated annealing approach. The algorithm has produced very good results and will be further developed. When solving QAP, the algorithm returns comparable results to standard QAP heuristic methods, but is far more flexible and suitable for a much wider range of problems.

2.4 Optimisation problem definition

The factory floor is labelled as region Ω in the plane $\mathbb{R}^2$. The problem was reduced by limiting Ω to a rectangle,

$$\Omega = \{(x, y) \in \mathbb{R}^2\colon x_{\min} \leq x \leq x_{\max}, y_{\min} \leq y \leq y_{\max}\}, \tag{2}$$

where $x_{\min}$, $x_{\max}$, $y_{\min}$ and $y_{\max}$ define edges of the shop floor.

The set of machines contains elements m_i, $i = 1,2,\dots,N$. The location of m_i is defined by $\boldsymbol{x}_i := (x_i, y_i) \in \mathbb{R}^2$.

A machine requires a certain area of the floor, defined by a rectangular-like ball $B_{r_i}(\boldsymbol{x}_i)$ with radius r_i and location $\boldsymbol{x}_i$ in ∞-norm L_∞,

$$B_{r_i}(\boldsymbol{x}_i) := \left\{(x, y) \in \mathbb{R}^2\colon \ \|(x, y), \boldsymbol{x}_i\|_\infty := \max\{|x - x_i|, |y - y_i|\} < r_i \right\}. \tag{3}$$

For every pair of machines m_i and m_j, $i, j = 1,2,\dots,N$, we obtain a flow of volume of products $f_{ij} \geq 0$ as a result of the simulation.

Distance $d(m_i, m_j)$ between the pair of machines m_i and m_j is defined as the shortest path between the machines in a predefined network of routes. The total travel distance optimisation problem is defined as:

$$\min_{\{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N\}} \left(\sum_{i,j=1,\ i\neq j}^{N} f_{ij} \cdot d(m_i, m_j) \right), \tag{4}$$

where locations $\boldsymbol{x}_i$ have to meet the conditions

$$B_{r_i}(\boldsymbol{x}_i) \cap B_{r_j}(\boldsymbol{x}_j) = \emptyset \tag{5}$$

for every $i \neq j$ and

$$B_{r_i}(\boldsymbol{x}_i) \subset \Omega \tag{6}$$

for every $i = 1,2,\dots,N$. The conditions (5) and (6) describe physical limitations of machine size and the borders of shop floor states – machines cannot occupy same space and cannot be placed outside of factory floor.

For every layout of machines one would also need to define a suitable network of routes. To simplify the tedious problem of defining the network from the machine positions, we presume the distance between the machines is a *Manhattan distance*,

$$d_{\mathrm{M}}(m_i, m_j) := \left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|_1 = |x_i - x_j| + |y_i - y_j|. \tag{7}$$

Since the original routes in the factory are defined on a rectangular grids, differences in lengths of paths, if we use the functional d_{M} instead of d, are small.

If we presume that all machines take up the same amount of space on the floor (all r_i are the same), we can restrict the positions $\boldsymbol{x}_i$ to discrete points on a predefined grid. Let us denote (fixed) points on the grid by $\boldsymbol{y}_i$, $i = 1,2,\dots,N$. Positions of m_i are determined by

$$\boldsymbol{x}_i := \boldsymbol{y}_{\pi(i)}, \qquad i = 1,2,\dots,N, \tag{8}$$

where π is a permutation on the set $\{1,2,\dots,N\}$ and S_N is a set of all possible permutations on $\{1,2,\dots,N\}$. In that case, the optimisation problem (4) simplifies to well-known quadratic assignment problem:

$$\min_{\pi \in S_N} \left(\sum_{i,j=1,\ i \neq j}^{N} f_{ij} \cdot \left\| \boldsymbol{y}_{\pi(i)} - \boldsymbol{y}_{\pi(j)} \right\|_1 \right). \tag{9}$$

The QAP heuristic methods [19-21] try to solve the problem (9). Although the primary ideas from which the methods were developed are different (i.e. simulated annealing, iterative local search and ant colony algorithm), the basic procedures of the methods are similar. The differences are how the methods search the large space of all possible permutations (for 140 facilities there are $140! \approx 1.3 \cdot 10^{241}$ permutations).

2.5 Force-directed graph drawing based algorithm

A force-directed graph drawing algorithm was used as the basis for the development of the heuristic optimisation algorithm presented in this section. The algorithm assigns the position $\boldsymbol{x}_i$ to every machine m_i, gradually constructing Graph G which finally represents a new machine layout. Machines with higher product volume are placed first to have more freedom of movement. The machines are placed randomly on the floor and change position depending on repulsive and attractive forces between nodes. The machine layout converges to a local minimum of total system energy. The simulation stops after the local minimum is reached.

Graph $G = (V, E)$ contains node sets $V = \{v_i\}_i$ and edges $E = \{e_{ij}\}_{i,j}$. Every machine m_i is presented as a node v_i. Edges e_{ij} represent direct mutual transactions of products between m_i and m_j.

Each node v_i produces a repulsive force F_{ij} affecting other nodes (v_j),

$$F_{ij} := H_{ij}\left(\left\|\boldsymbol{x}_j - \boldsymbol{x}_i\right\|_\infty\right) \cdot \frac{\boldsymbol{x}_j - \boldsymbol{x}_i}{\left\|\boldsymbol{x}_j - \boldsymbol{x}_i\right\|_\infty}, \tag{10}$$

with H_{ij} a positive monotonically decreasing function. Although it is common to define the function as $H_{ij}(r) = r^{-2}$, the drawback is that the function does not have a local support, hence the complexity to compute the repelling forces of the system is $O(|V|^2)$. For our model we chose:

$$H_{ij}(r) = \begin{cases} c_{\mathrm{r}} \cdot (r_i - r)^2, & r \leq r_i \\ 0, & r > r_i \end{cases} \tag{11}$$

and c_{r}, r_i are distance influence parameters. Nodes that are in the influence proximity of node v_i are stored in the set $B_i := \left\{v_j \in V : \left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|_\infty \leq r_i\right\} \setminus \{v_i\}$. The procedure algI($G$, i, r_i) to compute the set is presented as Algorithm I. Since the complexity to compute the neighbourhoods is $O(|V|^2)$, the neighbourhoods are not updated each iteration but only every d_B iterations.

Algorithm I: Update neighbourhood of v_i.

Require: $G = (V, E)$, i, r_i
  $B_i = \{\}$;
  for $j = 1,2, \dots, |V|$ **do**
    if $i \neq j \wedge \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_\infty < r_i$ **then**
      $B_i = B_i \cup \{v_j\}$;
    end if
  end for

Each pair of neighbouring nodes v_i, v_j is assigned a weighted edge e_{ij} with weight $w(e_{ij}) = f_{ij}$. Attractive force between the nodes depends on edge weight and is defined as

$$G_{ij} := -f_{ij} \cdot I_{ij} \left(\|\boldsymbol{x}_j - \boldsymbol{x}_i\|_1\right) \cdot \frac{\boldsymbol{x}_j - \boldsymbol{x}_i}{\|\boldsymbol{x}_j - \boldsymbol{x}_i\|_1}, \tag{12}$$

where I_{ij} is a positive monotonically increasing function, defined as $I_{ij}(r) = \|\boldsymbol{x}_j - \boldsymbol{x}_i\|_1$. The attractive force moves the nodes closer to each other and diminishing the value of $f_{ij} \cdot \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_1$, which represents a part of the overall product distance, in the process.

To keep the nodes within floor limits, forces pull the nodes inside the edges if they are outside of region Ω,

$$J_i := \begin{cases} 0, & \boldsymbol{x}_i \in \Omega \\ \mathrm{dist}(\boldsymbol{x}_i, \Omega), & \boldsymbol{x}_i \notin \Omega \end{cases} \tag{13}$$

and dist is a functional determining the distance between nodes and edges.

Note that the presented method allows straightforward implementations of additional forces on the nodes. This can be useful if we would like to fix or bind positions of certain nodes to reduce searching space or if certain machines cannot be moved. The simulation starts with region Ω empty. Nodes and edges are added to the graph G gradually during simulation. After every $d_e \in \mathbb{N}$ iterations new edge is added to G. New position of nodes are computed every iteration as

$$\boldsymbol{x}_i = \boldsymbol{x}_i + \delta \cdot \left(\sum_{v_j \in B_i} F_{ij} + \sum_{v_j \in N_G(v_i)} G_{ij} + J_i \right), \tag{14}$$

$i = 1,2, \dots, |V|$, where δ is constant parameter and $N_G(v_i)$ is a set of nodes adjacent to v_i. The algorithm algII(G,$\{B_i\}_i$,$\{J_i\}_i$) that updates all the positions is presented as Algorithm II.

Algorithm II: Update positions of nodes.

Require: $G = (V, E)$, $\{B_i\}_{i=1}^{|V|}$, $\{J_i\}_{i=1}^{|V|}$
  for $i = 1,2, \dots, |V|$ **do**
    $F = \sum_{v_j \in B_i} F_{ij}$; // compute repelling forces
    $F = F + \sum_{v_j \in N_G(v_i)} G_{ij}$; // add attractive forces
    $x_i = x_i + \delta \cdot (F + J_i)$;
  end for

Final layout of machines depends on the order in which the edges and nodes are added to the system during simulation. Results from simulation based testing indicated it is better to add edges e_{ij} with larger weights f_{ij} earlier. Let us order the set of edges $\{e_{ij}\}_{i,j=1}^{N}$ in decreasing order with respect to the corresponding flows f_{ij} and let us denote its element as e_k,

$$\{e_k\}_{k=1}^{K} = \{e_{ij}\}_{i,j=1}^{N}, \qquad w(e_k) \geq w(e_{k+1}). \tag{15}$$

Procedure algIII(e_{ij},G,r_i,r_j) to add new edge e_{ij} to graph G is presented as Algorithm III.

Algorithm III: Add a new edge to the graph.

Require: e_{ij}, $G = (V, E)$, r_i, r_j
  if $v_i \notin V$ **then**
    $V = V \cup \{v_i\}$;
    set $\boldsymbol{x}_i$ to random value inside Ω;
    algI(G,i,r_i);
  end if
  if $v_j \notin$ V **then**
    $V = V \cup \{v_j\}$;
    set $\boldsymbol{x}_j$ to random value inside Ω;
    algI(G,j,r_j);
  end if
  $E = E \cup \{e_{ij}\}$;

Strong repulsive forces impede the nodes to move freely, so the system may become stuck in a poor local minimum. To overcome this problem, the starting repulsive forces are set to a much lower value and are increased to the final value at the end of the simulation once all the nodes and edges are inserted into the system. Repulsive force parameter c_r is gradually increased from c_{r_0} to c_{r_1} in d_r iterations.

2.6 Simulated annealing

Performance of the optimisation algorithm is further improved by introducing simple random permutations of the neighbouring node positions during the optimisation process (see Algorithm IV), to move nodes that may be blocked by other nodes to a better location. Probability of permutations and sizes of neighbourhoods are based on simulated annealing.

Algorithm IV: Random permutations with simulated annealing.

Require: i, t
  select random node $v_j \in B_i$;
  $\Delta G := \sum_{v_k \in N_G(v_i)} f_{ik} \cdot \left(\|\boldsymbol{x}_k - \boldsymbol{x}_j\| - \|\boldsymbol{x}_k - \boldsymbol{x}_i\|\right) + \sum_{v_k \in N_G(v_j)} f_{jk} \cdot \left(\|\boldsymbol{x}_k - \boldsymbol{x}_i\| - \|\boldsymbol{x}_k - \boldsymbol{x}_j\|\right)$; // energy difference ΔG (new−old) if permutation is applied
  with probability $p(\Delta G, t)$ exchange $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$;

In our model the probability $p(\Delta G, t)$ to apply a random permutation is defined as

$$p(\Delta G, t) = \min\{e^{-c \cdot t \cdot \Delta G}, 1\}, \tag{16}$$

where $t = 0,1,2, \ldots$ is time step in the simulation and c is a constant parameter. Hence, probability p to apply a permutation that would increase the energy gradually decreases each simulation step t. Simulated annealing temperature is defined as $1/t$. Acceptance of the permutation also depends on energy difference ΔG. Permutation that decreases the energy (i.e. $\Delta G < 0$) is always applied.

In Algorithm IV, candidate node v_j is randomly chosen only inside neighbourhood B_i. In the later stage of the optimisation process, distances between the nodes increase due to increased repelling forces. Hence, number of permutation candidates $|B_i|$ naturally decreases at the end of the simulation as the algorithm converges to a local minimum.

2.7 The algorithm pseudo-code

The whole force-directed graph drawing based optimisation algorithm is presented as Algorithm V.

Algorithm V: Force-directed graph drawing algorithm.

```
Require: Ω, {e_k}_{k=1}^K, {r_k}_{k=1}^K, {r'_k}_{k=1}^K, d_e, d_B, d_r, d_p, c_{r_0}, c_{r_1}, i_max
  V = {}, E = {}, G = (V, E), j = 0, k = 0; c_r = c_{r_0};
  for t = 0,1, ..., t_max do
      if t ≡ 0 (mod d_e) ∧ t < K · d_e then
          algIII(e_k, G, r_k, r'_k); // add new edge (and nodes)
      end if
      if t ≡ 0 (mod d_B) then
          for i = 1,2, ..., |V| do
              algI(G, i, r_i); // update neighbourhoods
          end for
      end if
      if t ≡ 0 (mod d_p) then
          algIV(k, t); // with probability p permute positions of v_k and its neighbouring node
          k = k + 1 (mod |V|);
      end if
      if K · d_e ≤ t ∧ j < d_r then
          c_r = c_r + (c_{r_1} − c_{r_0})/d_r; // increase repulsive forces
          j = j + 1;
      end if
      algII(G, {B_i}_i, {J_i}_i); // update positions of nodes
  end for
```

3. RESULTS AND DISCUSSION

Main results of the project include a DES model in AnyLogic using the client company database to generate simulation scenarios and the novel heuristic machine layout optimisation algorithm, which aims to reduce costs by minimising the travel distance of products within the factory. The DES model is used for verification and validation of generated and manually constructed machine layouts. To accelerate the testing of layouts we have also developed a method for automatic model construction described in [28].

Twenty synthetic customer order sets that represent simplified real historic data were created and their production was simulated in our model. Using the simulated product manufacturing flows the following optimisation methods were tested and compared: QAP Ant colony algorithm [21], QAP Iterative local search (QAP ILS) [20], QAP Simulated annealing (QAP SA) [19] and our Force-directed graph drawing method (FDGD). Algorithm FDGD is based on attractive and repulsive forces only, whereas in FDGD2 additional random permutations with simulated annealing approach are integrated. Both methods search the solution in a continuous space as opposed to QAP's discrete space. To more accurately compare the effectiveness of methods we also projected the FDGD and FDGD2 final layouts onto the discrete grid as used by the QAP methods. This was done by using a simple greedy method that projects the nodes onto the nearest points. Results of this naive projection are labelled as "* grid" in Table I and Table II. These results are slightly worse due to greater granularity of machine positions and simple projections rules but still comparable to QAP results.

Since the methods are stochastic we ran each method ten times. Test results on the simplified historic data are shown in Table I. Additional comparison of the methods was made on standard QAP case: Wil50 by Wilhelm and Ward (http://www.opt.math.tu-graz.ac.at/qaplib/inst.html#WW). Results of the latter case are shown in Table II.

Table I: Test of optimisation methods on simplified problem with historic data.

Method	Best result	Average result
QAP Ant colony	618 km	678 km
QAP ILS	168 km	171 km
QAP SA	164 km	166 km
FDGD	195 km	207 km
FDGD grid	207 km	238 km
FDGD2	176 km	185 km
FDGD2 grid	190 km	215 km

Table II: Test of optimisation methods on standard problem: Wil50.

Method	Best result	Average result
QAP Ant colony	26,531	26,678
QAP ILS	24,420	24,431
QAP SA	24,408	24,411
FDGD	23,296	23,841
FDGD grid	25,305	25,492
FDGD2	22,552	23,098
FDGD2 grid	24,827	25,089

In both tests QAP SA method produced result with the smallest value of the minimisation functional. Method FDGD2 produced slightly better results than the variation without simulation annealing. Both our grid results are comparable to the results of QAP ILS and QAP SA. QAP Ant colony algorithm produced poor results in the first test where problem size is larger.

To obtain the near optimal floor machine layout based on real historic data, our final simulation test included 19 historical sets of customer orders containing 440,000 manufactured items that represented one month's worth of manufacturing. Based on the test results (Table I and Table II) and additional company requirements that would be difficulty to implement in QAP methods, we have decided that the force-directed graph drawing method will be further used in the optimisation process. Results and progress of the optimisation for five main proposed machine layouts are presented in Table III. The second layout '2. 6.' was developed manually by company experts and layout '14. 7.' was generated using simplified synthetic data and algorithm FDGD but slightly modified to satisfy additional company requirements. The last two layouts were obtained by our optimisation method.

Table III: Main optimisation statistics for different machine layouts.

Layout	Time	Total distance	Relative distance
current	30.9 days	690 km	100 %
2. 6.	30.6 days	617 km	89 %
14. 7.	30.2 days	564 km	82 %
3. 10.	30.3 days	506 km	73 %
10. 10.	30.2 days	492 km	72 %

In the final layout proposition we reduced the total product distance for nearly 30 % compared to the current layout. Shorter travel of products means less transport time, fewer carts and fewer workers, which can therefore be transferred to other assignments on the factory floor.

The customer has responded very favourably to these results and is willing to implement the suggested changes. They have also prepared several manually adjusted floor layouts that were based on our generated optimal layout. The manual variations of the layouts slightly differ from our proposal but satisfy additional less formally described layout requirements (e.g. location of installations). All variations were verified with our simulation model. We have also examined the financial aspect of machine relocations by calculating the workforce cost reduction and cost of moving the machines to examine the economic feasibility of new layouts.

Minimising product travel distance only negligibly affected the total manufacturing time for the given set of orders (see 'Time' in Table III). This was predicted since machine operation times are much longer than product transport times.

4. CONCLUSION

The novel force-directed layout optimisation method has generated promising results in the optimisation of the factory floor layout. The optimised layout requires 30 % shorter total product travel distance than the current factory layout.

In the further development of our layout optimisation method we will investigate and test several extensions. These will include different types of optimisation functional (this was partially already tested with the mentioned cost functional to minimise cost reduction and cost of moving the machines). Introducing a more geometrically complex elements such as nonconvex domains and nonrectangular objects (machines) that can also rotate is a challenge that is to be investigated in the future. One of the important extensions of the optimisation method is also to combine it with other heuristic techniques such as genetic algorithms and self-organized clustering [29] or to replace simulated annealing subroutine for permutations of neighbouring nodes' positions with other permutation procedures. Last but not least, optimisation lower bounds and convergence properties should be investigated in the near future.

Several optimisation problems still exist at the client company, e.g. reduction of bottlenecks and reduction of total order production time. To develop optimisation methods for these problems we will need to model additional aspects of machine operation, e.g. machine availability (maintenance, failures) and develop scenarios with varying sequence of job orders. Methods such as genetic algorithms will be used to optimize job scheduling. To accelerate testing of simulation scenarios we will introduce parallel execution of simulation runs at multiple hosts and utilize the HPC (high power computing) infrastructure at the faculty.

Replacement of one or several machines by newer multi-purpose CNC machines, replacement of human operators by robots, and new transport methods such as conveyor belts and automated carts are being considered in the frame of the ongoing factory logistics optimisation project. The effect of new technologies is to be tested in the simulation model before purchase and implementation. This will require significant alterations of the DES model e.g. more detailed modelling of transport unit loading and travel and modelling of operators.

Additional improvement to factory operation would be possible through integration of the simulation model with control points on the factory floor. This would allow real time monitoring of the manufacturing process and prediction of order completion as well as investigation of cause-effect relations in the manufacturing system [30] and use of lean manufacturing methods such as cost-time profiling [31].

ACKNOWLEDGEMENT

Our work has been supported by Creative Core FISNM-3330-13-500033 `Simulations' project funded by the European Union, The European Regional Development Fund. The operation is carried out within the framework of the Operational Programme for Strengthening Regional Development Potentials for the period 2007-2013, Development Priority 1: Competitiveness and research excellence, Priority Guideline 1.1: Improving the competitive skills and research excellence.

The authors wish to thank Uwe Truetsch for valuable information and suggestions on QAP algorithms.

REFERENCES

[1] Kobourov, S. G. (2013). Force-directed drawing algorithms, Tamassia, R. (Ed.), *Handbook of Graph Drawing and Visualization*, CRC Press, 383-408

[2] Kljajić, M.; Bernik, I.; Škraba, A. (2000). Simulation approach to decision assessment in enterprises, *Simulation*, Vol. 75, No. 4, 199-210, doi:10.1177/003754970007500402

[3] Edis, R. S.; Kahraman, B.; Araz, O. U.; Özfirat, M. K. (2011). A facility layout problem in a marble factory via simulation, *Mathematical and Computational Applications*, Vol. 16, No. 1, 97-104

[4] Tearwattanarattikal, P.; Namphacharoen, S.; Chamrasporn, C. (2008). Using ProModel as a simulation tools to assist plant layout design and planning: Case study plastic packaging factory, *Songklanakarin Journal of Science and Technology*, Vol. 30, No. 1, 117-123

[5] Benjaafar, S.; Heragu, S. S.; Irani, S. A. (2002). Next generation factory layouts: Research challenges and recent progress, *Interfaces*, Vol. 32, No. 6, 58-76, doi:10.1287/inte.32.6.58.6473

[6] Debevec, M.; Simic, M.; Herakovic, N. (2014). Virtual factory as an advanced approach for production process optimization, *International Journal of Simulation Modelling*, Vol. 13, No. 1, 66-78, doi:10.2507/IJSIMM13(1)6.260

[7] Sadrzadeh, A. (2012). A genetic algorithm with the heuristic procedure to solve the multi-line layout problem, *Computers & Industrial Engineering*, Vol. 62, No. 4, 1055-1064, doi:10.1016/j.cie.2011.12.033

[8] Ficko, M.; Brezovnik, S.; Klancnik, S.; Balic, J.; Brezocnik, M.; Pahole, I. (2010). Intelligent design of an unconstrained layout for a flexible manufacturing system, *Neurocomputing*, Vol. 73, No. 4-6, 639-647, doi:10.1016/j.neucom.2009.06.019

[9] Krishnan, K. K.; Cheraghi, S. H.; Nayak, C. N. (2008). Facility layout design for multiple production scenarios in a dynamic environment, *International Journal of Industrial and Systems Engineering*, Vol. 3, No. 2, 105-133, doi:10.1504/IJISE.2008.016740

[10] Kochhar, J. S.; Heragu, S. S. (1999). Facility layout design in a changing environment, *International Journal of Production Research*, Vol. 37, No. 11, 2429-2446, doi:10.1080/002075499190590

[11] Enea, M.; Galante, G.; Panascia, E. (2005). The facility layout problem approached using a fuzzy model and a genetic search, *Journal of Intelligent Manufacturing*, Vol. 16, No. 3, 303-316, doi:10.1007/s10845-005-7025-4

[12] Hahn, P. M.; Grant, T.; Hall, N. (1998). A branch-and-bound algorithm for the quadratic assignment problem based on the Hungarian method, *European Journal of Operational Research*, Vol. 108, No. 3, 629-640, doi:10.1016/S0377-2217(97)00063-5

[13] Anstreicher, K. M.; Brixius, N. W. (2001). Solving quadratic assignment problems using convex quadratic programming relaxations, *Optimization Methods and Software*, Vol. 16, No. 1-4, 49-68, doi:10.1080/10556780108805828

[14] Fischetti, M.; Monaci, M.; Salvagnin, D. (2012). Three ideas for the quadratic assignment problem, *Operations Research*, Vol. 60, No. 4, 954-964, doi:10.1287/opre.1120.1073

[15] Povh, J.; Rendl, F. (2009). Copositive and semidefinite relaxations of the quadratic assignment problem, *Discrete Optimization*, Vol. 6, No. 3, 231-241, doi:10.1016/j.disopt.2009.01.002

[16] de Klerk, E.; Nagy, M. E.; Sotirov, R.; Truetsch, U. (2014). Symmetry in RLT-type relaxations for the quadratic assignment and standard quadratic optimization problems, *European Journal of Operational Research*, Vol. 233, No. 3, 488-499, doi:10.1016/j.ejor.2013.10.007
[17] Koopmans, T. C.; Beckmann, M. (1957). Assignment problems and the location of economic activities, *Econometrica*, Vol. 25, No. 1, 53-76, doi:10.2307/1907742
[18] Sahni, S.; Gonzalez, T. (1976). P-complete approximation problems, *Journal of the ACM*, Vol. 23, No. 3, 555-565, doi:10.1145/321958.321975
[19] Taillard, E. (1998). Simulated annealing (SA) procedure for the quadratic assignment problem, from *http://mistic.heig-vd.ch/taillard/codes.dir/sa_qap.cpp*, accessed on 26-05-2015
[20] Shah, S. (n.d.). Implementation of iterative local search (ILS) for the quadratic assignment problem, from *http://shah.freeshell.org/ilsassignment/cILSAssignment.cpp*, accessed on 26-05-2015
[21] Tsourapas, K. (2008). Ant algorithm for the quadratic assignment problem, from *http://www.mathworks.com/matlabcentral/fileexchange/1663-qap*, accessed on 26-05-2015
[22] Bannister, M. J.; Eppstein, D.; Goodrich, M. T.; Trott, L. (2013). Force-directed graph drawing using social gravity and scaling, Didimo, V.; Patrignani, M. (Eds.), *Graph Drawing*, Lecture Notes in Computer Science, Vol. 7704, 414-425
[23] Fruchterman, T. M. J.; Reingold, E. M. (1991). Graph drawing by force-directed placement, *Software: Practice and Experience*, Vol. 21, No. 11, 1129-1164, doi:10.1002/spe.4380211102
[24] Ingber, L. (1993). Simulated annealing: Practice versus theory, *Mathematical and Computer Modelling*, Vol. 18, No. 11, 29-57, doi:10.1016/0895-7177(93)90204-C
[25] Otten, R. H. J. M.; van Ginneken, L. P. P. P. (1998). *The Annealing Algorithm*, Kluwer Academic Publishers, Boston
[26] Bertsimas, D.; Tsitsiklis, J. (1993). Simulated annealing, *Statistical Science*, Vol. 8, No. 1, 10-15, doi:10.1214/ss/1177011077
[27] Kirkpatrick, S.; Gelatt Jr., C. D.; Vecchi, M. P. (1983). Optimization by simulated annealing, *Science*, Vol. 220, No. 4598, 671-680, doi:10.1126/science.220.4598.671
[28] Kanduč, T.; Rodič, B. (2014). Automated simulation model building for a complex furniture manufacturing process, *23rd International Electrotechnical and Computer Science Conference ERK 2014*, Slovenia section IEEE, 159-162
[29] Potocnik, P.; Berlec, T.; Sluga, A.; Govekar, E. (2014). Hybrid self-organization based facility layout planning, *Strojniski vestnik – Journal of Mechanical Engineering*, Vol. 60, No. 12, 789-796, doi:10.5545/sv-jme.2014.1748
[30] Seebacher, G.; Winkler, H.; Oberegger, B. (2015). In-plant logistics efficiency valuation using discrete event simulation, *International Journal of Simulation Modelling*, Vol. 14, No. 1, 60-70, doi:10.2507/IJSIMM14(1)6.289
[31] Gracanin, D.; Lalic, B.; Beker, I.; Lalic, D.; Buchmeister, B. (2013). Cost-time profile simulation for job shop scheduling decisions, *International Journal of Simulation Modelling*, Vol. 12, No. 4, 213-224, doi:10.2507/IJSIMM12(4)1.237