Lancaster University
MANAGEMENT SCHOOL

Integrated facility layout design and flow assignment problem under uncertainty

Yifei Zhao
Department of Management Science
Lancaster University Management School

Stein W. Wallace
Department of Business and Management Science
Norwegian School of Economics

Wroking Paper 2013:5

Integrated facility layout design and flow assignment problem under uncertainty

Yifei Zhao
Department of Management Science
Lancaster University Management School, Lancaster LA1 4YX, UK

Stein W. Wallace
Department of Business and Management Science
Norwegian School of Economics, NO-5045 Bergen, Norway

January 2013

Abstract

The facility layout problem is the problem of assigning facilities to locations. We study the case of limited machine capacity, and hence multiple copies of each machine type. We take into account stochastic demand, described by several types of jobs (i.e. sequences of machine types), each with an uncertain demand level. We develop a heuristic framework allowing us to find good solutions to the stochastic case whenever it is possible to solve the corresponding deterministic Quadratic Assignment Problem, exactly or heuristically. Though the QAP is a very hard problem in its own right, our approach allows randomness (and hence relevance) to be added at only a marginal increase in computational costs.
Keywords: Facility layout; Random demand; Machine capacities

1 Introduction

The facility layout problem (FLP) – also called the plant layout problem – is the problem of assigning physical facilities (such as machine tools, work centres, manufacturing cells, departments, and warehouses) for a production or a delivery system to locations. FLPs arise either in the early design stage of a new facility or during improvement phases of current layouts. Layout design is one of the most fundamental and strategic issue in many manufacturing industries. Any modification or re-arrangement of an existing layout very likely involves substantial financial investment and planning efforts (Singh and Singh 2010). An efficient layout of facilities can reduce operational costs and contribute to overall production efficiency (Tomkins et al 2003).

Given information such as machine dimensions, capacities of machines and volumes transferred between all pairs of machines, the planner's objective is to determine an efficient facility layout according to some design criterion, often the minimization of material handling distance/costs. Such costs are incurred when moving finished parts, work-in-process parts, spares, tools and other supplies between machines or storage locations to meet production demand (Heragu and Kusiak 1988). A significant proportion, about 20% to 50%, of overall product costs can be attributed to material handling (Francis and White 1974), since material handling is involved in almost all manufacturing activities from the arrival of raw materials to the packaging of finished products. In a material handling system, automation is one of the key principles since it can effectively reduce operational costs and enhance manufacturing performance. Unmanned material handling equipment such as conveyors, automatic guided vehicles (AGV), cranes and robots are deployed for automatic delivery activities. In conventional manufacturing, where conveyors are the only available automatic tool, group technology (GT) is applied to take advantage of such automation technology. By using GT, parts are classified into groups with similar process sequences. Within each group, equipments are then connected by a conveyor in a straight line or a work cell which forms a fixed path with no backtracking, jumping or skipping. The primary benefit of GT is reduction in processing time, inventory and tooling. However, GT is not very efficient when facing frequently changing product requirements. As markets have become more unpredictable since the 1990s, flexible manufacturing systems (FMS) are required to adapt to changes in product mix, demand and designs with low-cost solutions. Automatic tools including AGVs and robots are the main types of equipment in FMS. Different from conveyors which can only operate between two fixed locations, AGVs and robots have high degrees of route flexibility since they can travel between any two locations. Our research focuses on planning good layouts for completely automatic FMSs where no human workers are needed. In such a system, the material handling cost is largely affected by the travel distance of the handling tools. Shortening travel distance would help reduce the handling costs and manufacturing lead time.

The FLP was initially formulated as a quadratic assignment problem (QAP) by Koopmans and Beckman (1957). The QAP assigns n facilities to n locations in such a way that each facility is assigned to exactly one location (Constraint (2)) and each location takes exactly one facility (Constraint (3)). This QAP and its augmentations have been widely applied in solving FLPs. The QAP formulation is shown in the following where F_{ij} is the required volume to be moved from machine i to machine j and d_{kl} is the unit flow cost from location k to location l.The binary decision variable x_{ij} is one if machine i is placed in location j, 0 otherwise.

$$\min z = \sum_{i=0}^{n}\sum_{k=0}^{n}\sum_{j=0}^{n}\sum_{l=0}^{n} F_{ij} \times d_{kl} \times x_{ik} \times x_{jl} \tag{1}$$

subject to:

$$\sum_{k=0}^{n} x_{ik} = 1 \qquad i = 1, \ldots, n \tag{2}$$

$$\sum_{i=0}^{n} x_{ik} = 1 \qquad k = 1, \ldots, n \tag{3}$$

$$x_{ik} = \begin{cases} 1, & \text{if machine } i \text{ is assigned to location } k \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

The QAP is NP-hard and is considered as one of the most difficult combinatorial problems around. Extensive research on algorithms to solve the QAP has been done for more than half a century. Exact methods (e.g. branch-and-bound (Salimanpur and Jafari 2008) and cutting planes (Bukard and Bonniger 1983)) are only efficient to layout problems of dimensions up to about 16 (Singh and Sharma 2006). To achieve solutions for larger cases, heuristic methods, based on the QAP, such as simulated annealing (Baykasoglu and Cindy 2001, Misevicius 2003), genetic algorithms (Balakrishnan et al 2003, Lee et al 2003) and tabu search (Chiang and Chiang 1998, Helm and Hadley 2000) have been proposed. The performances of SA and GA have been compared by Heragu and Alfa (2003) and Tavakkoli-Moghaddain and Shanyan (1998). Recently, Zhang et al (2010) reformulated the QAP by reducing both the number of constraints and variables. The reformulations lead to problems that are easier to solve, especially for QAPs with sparse flow matrices F.

In QAPs, the required flows between pairs of machines are given by a flow matrix F. Urban et al (2000) studied a new problem, the integrated facility layout and flow assignment problem (IFLFAP), which is closer to reality by considering capacities of machines. In the IFLFAP, there is a number of machine types and each type has several duplicates with the same capacity.

For these models, F is kept constant throughout the planning period. Since we only know flows between pairs of machine types, how to assign flows to duplicates of each type must be determined by a flow assignment problem. Hence, the IFLFAP is the problem of simultaneously determining the machine layout (how to allocate machines to possible locations) and flow assignment (how to assign flows among machine copies) with the objective of minimizing material handling costs. An extended QAP was used to formulate the IFLFAP which was then solved by using two metaheuristics, GRASP and tabu search (Taghavi and Murat 2011). The resulting layout, as well as the material flows, do not follow any predefined rules, that is, there is full freedom as to where machine copies are placed.

By a *job* we shall understand any *arbitrary* scaling V of the flow matrix F, typically $V = F$, but any positive scaling will do. The point of a job is that it describes a flow *pattern* and relative magnitudes, but without a reference to the actual volumes. So, for example, $V_1 = (1, 2, 0)$ will describe the same job as $V_2 = (2, 4, 0)$.

A stochastic version of IFLFAP was considered by Benjaafar and Sheikhzadeh (2000), who took into account the facts that most facilities must handle many different jobs, and that the volume of a job may also vary (deterministically or stochatically) over time. This variation requires the design of robust facility layouts adaptable to varying jobs and fluctuating volumes. It was assumed that the frequency (probability) by which a job occurs, as well as the probability distribution describing its volume, were known in advance.

Benjaafar and Sheikhzadeh formulated the stochastic IFLFAP as an extended QAP with the objective of minimizing the total expected material handling cost over different production scenarios (describing jobs and their volumes). An iterative heuristic method based on Cooper (1963) was used to solve the problem with sizes up tp 40 by decomposing it into a layout assignment and a flow allocation sub-problem. The resulting distributed layout was then compared with three alternative approaches: functional layouts, random layouts, and maximally-distributed layouts. Functional layouts keep machine copies of the same type adjacent while maximally-distributed layouts allocate

duplicates as uniformly as possible throughout the available plant space. A random layout is generated by randomly assigning machines to locations. Results have shown that distributed layouts outperform all the three others by reducing the total expected costs from 10% to 40%.

In this paper, we study a new greedy heuristic for the stochastic IFLFAP. The formulation of the stochastic IFLFAP is based on the one developed by Benjaafar and Sheikhazadeh. The new approach is both faster and more accurate.

From a stochastic optimization perspective this paper presents an important case: A very hard deterministic problem can be extended to handle a more complex setting (capacitated machines) and, in particular, uncertainty (in demand and job type) without increasing the computational complexity significantly. This is not generally possible, and understanding when (and how) it can be done is crucial for adding uncertainty to discrete models, in particular. This is an important perspective on this paper: What can we do when we want to add uncertainty to models that are already more than difficult enough as deterministic models?

2 Model development

This paper studies the problem of facility layout design with machine duplicates in a stochastic environment. In line with literature, it is assumed that the facility is already there, so that the number of available locations is given, and reasonably balanced, in terms of size, with the needs of the production company. It is also implicitly assumed that the number of copies of each machine type is given: The focus here is the *location* of the machines. So, given the number of machines, and given an overall description of production needs (probably being random on top of a seasonal pattern), the company must decide which job to process in a given period. If the actual need in a period exceeds the capacity, it will process the capacity, while if it is below, it will process the need. So with a given set of machines, and given knowledge of the overall demand patterns, it is possible to calculate probabilities / frequencies of jobs and distribution functions for volumes for each job. In total, this leads to a situation where the number of locations equals the number of machines, and where all machines are needed for at least one job, which is how we shall set the model. (If our heuristic was applied to a case where there were more machines than needed, the unnecessary machine would not be used, but that is a minor point in this paper). So the picture is that these production needs are either internally generated / decided based on production plans, or there is genuine random demand, but the demands are buffered at company level, and then released for production, period by period, and only one job per period. Alternatively, one could assume that demands arrive randomly according to the given distributions, where each request is a combination of job and volume, and such that the demanders (the customers) know the maximal volume they can demand in one order, as that is published by the company.

The assumptions of the stochastic IFLFAP are summarized in the following, where the first two are the same as those in the classical FLP.

- The number of machines equals the number of locations
- The unit flow cost between pairs of machines only depends on their locations.
- Plants can only process one job during a production period, but jobs and volumes fluctuate from period to period.

  If re-layout costs could be ignored, we could reconfigure the layout in each production period according to the required job. However, in this paper, reconfigurations of facility layouts are considered infeasible since the frequency of changes of jobs and the costs of re-layouts are assumed too high for frequent redesigns to be economic and efficient. Hence the objective of this problem is to design a fixed facility layout for all production periods which minimizes the total expected material handling costs.
- Probabilities/frequencies of jobs are known and given.

- Continuous or discrete probability distributions describing volumes (demand levels) for all jobs are known and given.

  The demand volumes will be approximated by a finite number of discrete scenarios with given probabilities in the optimization models, while the full distributions will be used exactly when valuing solutions.

- Machine capacities, which are measured in total available processing time, are given and copies of the same machine type have the same capacity.

- Maximal volume/demand of each job does not exceed the total production capacity.

  This is obtained by the way jobs and volumes are constructed, as explained in Section 1.

- All machine duplicates are needed for the production of at least one job.

  This guarantees that no machine copy is redundant in the plant, and is also obtained by construction as explained in Section 1.

Parameters:

- N = total number of machine types
- N_i = total number of machine copies of type i
- K = total number of locations
- d_{kl} = distance between location k and location l
- C_i = available processing time for each machine copy of type i
- t_{si} = processing time per unit product on machines of type i for job s
- S = total number of jobs
- P_s = probability / frequency of job s being processed in a production period
- V_{sij}^r = volume of flows between machine types i and j of job s in demand scenario r, $r = 1, 2, ..., R$. Demand levels are ordered increasingly from $r = 1$ to $r = R$, so that V_{sij}^R is the maximal volume required by job s between machine types i and j.
- P_s^r = probability of job s having volume scenario r

Decision variables:

- $x_{n_i k} = \begin{cases} 1, & \text{if } n\text{th machine of type } i \text{ is assigned to location } k \\ 0, & \text{otherwise} \end{cases}$
  (the index n_i refers to the nth machine of type i)
- $v_{n_i m_j}$ = flow volume between nth copy of type i and mth copy of type j for a given demand level of the job

The problem is then formulated as follows:

$$\min z = \sum_{s=1}^{S} P_s \sum_{r=1}^{R} P_s^r \pi(x,s,r) \tag{5}$$

subject to:

$$\sum_{k=1}^{K} x_{n_i k} = 1 \qquad n_i = 1,\ldots,N_i \qquad i = 1,\ldots,N \tag{6}$$

$$\sum_{i=1}^{N} \sum_{n_i=1}^{N_i} x_{n_i k} = 1 \qquad k = 1,\ldots,K \tag{7}$$

$$x_{n_i k} = \{0,1\} \qquad n_i = 1,\ldots,N_i \qquad i = 1,\ldots,N \qquad k = 1,\ldots,K \tag{8}$$

where:

$$\pi(x,s,r) = \min\{\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{n_i=1}^{N_i}\sum_{m_j=1}^{N_j}\sum_{k=1}^{K}\sum_{j=1}^{K} v_{n_i m_j} d_{kl} x_{n_i k} x_{m_j l}\} \tag{9}$$

$$\sum_{\substack{i=0 \\ i\neq j}}^{N} \sum_{n_i=1}^{N_i} v_{n_i m_j} t_{sj} \leq C_j \qquad m_j = 1,\ldots,N_j \qquad j = 1,\ldots,N \tag{10}$$

$$\sum_{n_i=1}^{N_i} \sum_{m_j=1}^{N_j} v_{n_i m_j} = V_{sij}^r \qquad i,j = 1,\ldots,N \tag{11}$$

$$\sum_{i=0}^{N} \sum_{n_i=1}^{N_i} v_{n_i m_j} = \sum_{i=0}^{N} \sum_{n_i=1}^{N_i} v_{m_j n_i} \qquad m_j = 1,\ldots,N_j \qquad j = 1,\ldots N \tag{12}$$

$$v_{n_i m_j} \geq 0 \qquad i,j = 0,1,\ldots,N \qquad n_i = 1,\ldots,N_i \qquad m_j = 1,\ldots,N_j \tag{13}$$

The IFLFAP is formulated as a two-stage stochastic problem where the first stage is to decide the allocation of machines (Constraint sets (6) to (8)) and the second stage is, given results from the first stage, to assign flows between machines (Constraint sets (10) to (12)) for a given job and volume. The objective (5) is a cubic function minimizing expected material handling cost over all possible jobs and volumes. Constraint sets (6) and (7) of the first stage problem ensure that one location is allocated to one and only one machine, while one machine is assigned to one and only one location. In the second stage problem, Constraint set (10) guarantees that the capacity of each machine copy is not exceeded. Constraint set (11) ensures that the sum of flow amongst all copies of two machine types equals the required volume. Conservation of material flow in machines is guaranteed by (12).

The stochastic IFLFAP model is NP-hard. In our numerical experiments, computational difficulties were found when applying exact methods to solve problems with more than 10 machines. After that, optimal solutions cannot be obtained within a reasonable amount of time (36 hours in our numerical experiments). Hence heuristics are required to solve larger cases of IFLFAP.

3 The flow-map heuristic

The general idea of our greedy heuristic, which is named the flow-map heuristic, is to reduce the two-stage stochastic problem shown in Section 2 to a standard deterministic QAP by generating a *flow map* in the first step. The QAP can then be solved by any available exact or heuristic method. A flow map refers to a machine-to-machine flow (i.e. the F required by the QAP) rather than a machine type-to-machine type flow (i.e. the F required by the IFLFAP). The flow map is, in principle, obtained by solving a flow assignment problem. However, without knowing the locations of the machines, the ultimate criterion of minimizing flow costs cannot be applied in this flow assignment problem.

We use a two-step procedure to find a flow map. In the first step, only randomness of jobs is used (based on maximal demand for each job), while in the second step we test two procedures, one using also the randomness in demand, the other one not. However, in all cases the quality of the heuristic is measured by expected costs, using randomness over both jobs and demand levels.

In order to explain the basics of this heuristic, observe that each location in the facility has a limited number of nearest neighbours. Both with the Euclidean and taxi (Manhattan) norms, these are the left, right, up and down locations (if they exist, and we have a chess-board like setup). In any case, for any facility and any norm, each location will have a limited number of nearest neighbors, so that transferring materials to them is the cheapest choice. Hence, the overall flow cost will be minimal if machines can be placed such that flows are always between machines that are nearest neighbors. If there are too many non-zeros in a flow map, not all pairs of machines with non-zero flow can be nearest neighbors.

The heuristic idea is therefore to find a flow map with the smallest possible number of non-zeros. The fewer non-zeros in the flow map, the more likely that machines can be placed such that flows occur only between nearest neighbors. So we define a 0/1 variable for each pair of machines, and we open as few pairs as possible such that with the opened pairs, all jobs can be handled with their maximal demands, that is, when $r = R$. Hence the problem is formulated as minimizing the number of connections required by the maximal demand levels. In the formulation, the notation is the same as in Section 2. Note that if a pair is connected, flow will be allowed in both directions.

Though this is simply a model to minimize the number of connections, it is interesting to note the relationship to robust optimization where solutions are often driven by worst-case (which here is maximal demand) analyses. As this heuristic is very efficient, we see a case where a focus on the worst case leads to good solutions measured by average costs.

3.1 Minimize the number of connections

Decision variables:

- $y_{n_i m_j} = \begin{cases} 1, & \text{open the edge from machine } n_i \text{ to } m_j \\ 0, & \text{otherwise} \end{cases}$
- $v'_{sn_i m_j}$ = flows from machine n_i to m_j for job s in the maximal demand scenario R

Formulation:

$$\min z = \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{n_i=1}^{N_i}\sum_{m_j=1}^{N_j} y_{n_i m_j} \tag{14}$$

subject to:

$$v'_{sn_im_j} \le M \times y_{n_im_j} \qquad s=1,\dots,S \qquad i,j=1,\dots,N$$

$$n_i=1,\dots,N_i \qquad m_j=1,\dots,N_j \tag{15}$$

where M = maximal demand over all jobs

$$y_{n_im_j} = y_{m_jn_i} \qquad i,j=1,\dots,N \qquad n_i=1,\dots,N_i \qquad m_j=1,\dots,N_j \tag{16}$$

$$\sum_{\substack{i=0\\ i\neq j}}^{N}\sum_{n_i=1}^{N_i} v'_{sn_im_j}t_j \le C_{sj} \qquad s=1,\dots,S \qquad m_j=1,\dots,N_j \qquad j=1,\dots,N \tag{17}$$

$$\sum_{n_i=1}^{N_i}\sum_{m_j=1}^{N_j} v'_{sn_im_j} = V^R_{sij} \qquad s=1,\dots,S \qquad i,j=1,\dots,N \tag{18}$$

$$\sum_{i=0}^{N}\sum_{n_i=1}^{N_i} v'_{sn_im_j} = \sum_{i=0}^{N}\sum_{n_i=1}^{N_i} v'_{sm_jn_i} \qquad s=1,\dots,S \qquad m_j=1,\dots,N_j \qquad j=1,\dots N \tag{19}$$

$$y_{n_im_j} = 0,1 \qquad n_i=1,\dots,N_i \qquad m_j=1,\dots,N_j \qquad i,j=1,\dots,N \tag{20}$$

$$v'_{sn_im_j} \ge 0 \qquad s=1,\dots,S \qquad n_i=1,\dots,N_i \qquad m_j=1,\dots,N_j \qquad i,j=1,\dots N \tag{21}$$

Constraint (15) guarantees that there is no flow from machine n_i to m_j if these two machines are not connected. Edges are not directed as guaranteed by Constraint (16) so that once the edge between machines n_i and m_j is open, flow is allowed in both directions. Constraints (17) to (19) are used for the assignment of flows and these are the same as Constraints (10) to (12) in Section 2.

3.2 Uneven flow assignment

Using the model in Section 3.1, we end up with a set of open edges, such that each edge has positive flow in at least one direction for at least one job. We denote the set of open edges as

$$E_0 = \{(n_i,m_j) : y_{n_im_j} = 1, i=2,\dots,N; j=1,\dots,i-1; n_i=1,...,N_i; m_j=1,...,N_j\}.$$

Given flows obtained from the model in Section 3.1, flows on each open edge $(n_i,m_j)\in E_0$ can be calculated in different ways by using the probabilities of jobs and demand levels (remember that we do not know the machine locations and hence do not know actual flow costs). It is these flows that later will be passed to the QAP. In particular, flows can be assigned evenly or unevenly amongst copies of a given machine type. For example, assume a volume of 10 units is required from machine type 1 to machine type 2, and there is 1 copy (1_1) of machine type 1 and 2 copies (1_2,2_2) of machine type 2. Flows can be assigned in infinitely many ways, two of which are as follows: A very even flow with

$$v'_{1_1,1_2} = 5 \text{ and } v'_{1_1,2_2} = 5$$

and a very uneven flow with

$$v'_{1_1,1_2} = 10 \text{ and } v'_{1_1,2_2} = 0.$$

If the machines can be placed in such a way that all edges in E_0 connect nearest neighbors, all possible volumes for all jobs can be sent through the plant only by visiting nearest neighbours, hence at minimal costs. In this case, it is unimportant which flows we choose for the QAP. We do, however, not check if this is the case, but proceed as if it is not so. If this happens to be the case though, no harm will be done.

So we now have too many edges to let all edges connect nearest neighbors. So some flow for at least one job will be between non-neighbors. We would like the necessary flow between non-neighbors to be as small as possible. Not wanting to set up a very complicated combinatorial problem, as it would be contrary to the fact that this is a simplified heuristic, we proceed with two related goals in mind: Try to make flow on a subset of edges as small as possible (singling them out as edges that need not connect nearest neighbors, or reversely, try to create flows on a subset of edges as large as possible, so as to have large flows on those edges which will connect nearest neighbors). Our numerical results show that it is best to try to make some flows large (rather than focusing on making some others small). Here we clearly depart from a robust thinking, focusing on expected flows and hence expected costs.

In our small example, if only one of the type-2 machines can be placed close to the single type-1 machine, the uneven flow assignment method indicates that the pair 1_1 and 1_2 should take up the neighboring locations, and that is indeed what the QAP will produce. However, with even flow assignments, since the two machine pairs have the same flow, no indication is given to identify which machine pair should take priority. Therefore machines will be randomly assigned to locations by the QAP and result in increased total costs. Since an uneven flow assignment indicates a lower-cost layout, two methods are applied to assign flows unevenly on open edges $(n_i, m_j) \in E_0$.

3.2.1 The max-max algorithm.

This method assigns as much flow as possible to some of the open edges, using only maximal demands of the different jobs. So randomness in demand levels is not used. The method starts by maximizing the expected flow $z^{(0)}_{(n_i,m_j)}$ on each open edge $(n_i, m_j) \in E_0$ by using the linear model shown in the following. The linear model includes the continuous variables $v'_{sn_im_j}$. The binary $y_{n_im_j}$ are now fixed. The current iteration number is denoted *cur_iter* and is initially set to zero. A new set E_1 is initialized as $\emptyset$.

$$\begin{aligned} &\forall (n_i, m_j) \in E_0 \\ &\max z^{(0)}_{(n_i,m_j)} = \sum_{s=1}^{S} P_s (v'_{sn_im_j} + v'_{sm_jn_i}) \\ &\text{subject to}: (15)(17)(18)(19) \end{aligned} \tag{22}$$

The overall maximal objective value is obtained by

$$z^{(0)} = \max_{(n_i,m_j)\in E_0} \{z^{(0)}_{(n_i,m_j)}\}$$

and the corresponding open edge which has achieved $z^{(0)}$ is denoted by $(n_i^{(0)}, m_j^{(0)})$. The set E_0 is updated by removing $(n_i^{(0)}, m_j^{(0)})$, while E_1 is updated by adding $(n_i^{(0)}, m_j^{(0)})$.

Cur_iter is then increased by 1. In iteration *cur_iter* $= 1$, for edges belonging to the reduced E_0, the same model is applied to maximize the expected flow on each edge with a new constraint (24) added to guarantee that the expected flow on edge $(n_i^{(0)}, m_j^{(0)}) \in E_1$ remains at $z^{(0)}$.

$$\begin{aligned} &\forall (n_i, m_j) \in E_0 \\ &\max z^{(1)}_{(n_i,m_j)} = \sum_{s=1}^{S} P_s (v'_{sn_im_j} + v'_{sm_jn_i}) \\ &\text{subject to}: (15)(17)(18)(19) \end{aligned} \tag{23}$$

$$\text{and } \sum_{s=1}^{S} P_s (v'_{sn_i^{(0)}m_j^{(0)}} + v'_{sm_j^{(0)}n_i^{(0)}}) = z^{(0)} \tag{24}$$

We then proceed to obtain the maximal objective value $z^{(1)}$ and the corresponding edge $(n_i^{(1)}, m_j^{(1)})$. The two sets E_0 and E_1 are then updated by removing and adding the edge $(n_i^{(1)}, m_j^{(1)})$.

In the next iteration ($cur\_iter = 2$) a new constraint for edge $(n_i^{(1)}, m_j^{(1)})$, which has the same function as constraint (24), is added to the model to maximize expected flow on each edge belonging to the updated set E_0. Hence the total number of constraints added to the original model equals to the total number of elements in the set E_1.

The procedure of maximizing expected flow on each edge $(n_i, m_j) \in E_0$ ends when the current iteration number cur_iter exceeds a given limit $limit\_iter$. Our extensive experiments show that $limit\_iter$ around $\left\lceil \frac{1}{10} n \right\rceil$ is sufficient. After that there is hardly any freedom left in chosing flows due to the constraints added in each iteration. The flow map being used in the QAP is then given by flows, $v'_{sn_im_j}$, between each machine pair n_i and m_j from the last iteration. The amount of flow between each machine pair n_i and m_j is then calculated as $V(n_i, m_j) = \sum_{s=1}^{S} P_s v'_{sn_im_j}$. With this flow map, we have finished the uneven flow assignment. The implementation of the max max flow algorithm is briefly summarized as follows:

1. **Step 1** (Initialization)

   1.1. Read solution $y_{n_im_j}$ from model Section 3.1.

   1.2. Initialize edge sets E_0 and E_1.
   $E_0 = \{(n_i, m_j) : y_{n_im_j} = 1, \forall i = 1, ..., j; j = 2, ...N; \forall n = 1, ..., N_i; \forall m = 1, ..., N_j\}$
   $E_1 = \emptyset$

   1.3. Initialize $cur\_iter = 0$. Read $limit\_iter$.

2. **Step 2** (Maximize flow on each edge)

   2.1. For each edge $(n_i, m_j) \in E_0$, solve a linear model

$$\max z_{(n_i,m_j)}^{(cur\_iter)} = \sum_{s=1}^{S} P_s (v'_{sn_im_j} + v'_{sm_jn_i}) \tag{25}$$

$$\text{subject to}: (15)(17)(18)(19)$$

$$\sum_{s=1}^{S} P_s (v'_{sn_i^{(\alpha)}m_j^{(\alpha)}} + v'_{sm_j^{(\alpha)}n_i^{(\alpha)}}) = z^{(\alpha)},$$

$$(n_i^{(\alpha)}, m_j^{(\alpha)}) \in E_1; \alpha = 0, 1, ..., (cur\_iter - 1) \tag{26}$$

3. **Step 3** (update)

   3.1. Find out the maximal objective value $z^{(cur\_iter)} = \max\{z_{(n_i,m_j)}^{(cur\_iter)}, \forall (n_i, m_j) \in E_0\}$.
   Denote the machine pair with maximal flow amount $z^{(cur\_iter)}$ as $(n_i^{(cur\_iter)}, m_j^{(cur\_iter)})$. If there are more than one machine pair with the maximal flow amount, choose any one of them as $(n_i^{(cur\_iter)}, m_j^{(cur\_iter)})$.

   3.2. Update sets E_0 and E_1.
   Remove $(n_i^{(cur\_iter)}, m_j^{(cur\_iter)})$ from E_0.
   Add $(n_i^{(cur\_iter)}, m_j^{(cur\_iter)})$ in E_1.

   3.3. $cur\_iter = cur\_iter + 1$.

4. **Step 4** (Termination)
   If $cur\_iter > limit\_iter$, read results $v'_{sn_im_j}$ from solutions of the linear model with the maximum objective value $z^{(cur\_iter)}$ in Step 2. The flow map (flow between each machine pair) is generated by calculating the expected flow on each machine pair: $V(n_i, m_j) = \sum_{s=1}^{S} P_s v'_{sn_im_j}$; Otherwise, go to Step 2.

3.2.2 The increasing flow algorithm.

This method uses randomness in demand levels as well, and achieves uneven flows by assigning larger flows on those machine pairs which are used also in lower demand scenarios. For each job S, there is a total number of R demand levels ordered increasingly from scenario $r = 1$ to $r = R$. If a positive flow between a machine pair (n_i, m_j) in scenario r is given by $v^r_{sn_im_j}$, then for the next higher demand level $(r+1)$, the algorithm guarantees that $v^{r+1}_{sn_im_j}$ is no less than $v^r_{sn_im_j}$. Since flows stay the same or increase as the demand level increases, machine pairs which are used and have larger positive flows in low demand scenarios are likely to be assigned higher total expected flows over all R scenarios, resulting in an uneven flow assignment. Different from the max-max algorithm (Section 3.2.1) which requires two steps that minimize the number of connections and then assign uneven flows on the resulting open connections, the second algorithm combines the two steps into one model which simultaneously obtains the set of open edges E_0 and uneven flow assignments on the open edges. The formulation is shown as in the following:

$$\min z = \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{n_i=1}^{N_i}\sum_{m_j=1}^{N_j} y_{n_im_j} \tag{27}$$

subject to:

$$v^r_{sn_im_j} \le M \times y_{n_im_j} \qquad s = 1,\ldots,S \qquad r = 1,\ldots,R$$

$$i,j = 1,\ldots,N \qquad n_i = 1,\ldots,N_i \qquad m_j = 1,\ldots,N_j \tag{28}$$

$$y_{n_im_j} = y_{m_jn_i} \qquad i,j = 1,\ldots,N \qquad n_i = 1,\ldots,N_i \qquad m_j = 1,\ldots,N_j \tag{29}$$

$$\sum_{\substack{i=0\\ i\neq j}}^{N}\sum_{n_i=1}^{N_i} v^r_{sn_im_j}t_j \le C_j \qquad s = 1,\ldots,S \qquad r = 1,\ldots,R$$

$$m_j = 1,\ldots,N_j \qquad j = 1,\ldots,N \tag{30}$$

$$\sum_{n_i=1}^{N_i}\sum_{m_j=1}^{N_j} v^r_{sn_im_j} = V^r_{ij} \qquad s = 1,\ldots,S \qquad r = 1,\ldots,R \qquad i,j = 1,\ldots,N \tag{31}$$

$$\sum_{i=0}^{N}\sum_{n_i=1}^{N_i} v^r_{sn_im_j} = \sum_{i=0}^{N}\sum_{n_i=1}^{N_i} v^r_{sm_jn_i} \qquad s = 1,\ldots,S \qquad r = 1,\ldots,R$$

$$m_j = 1,\ldots,N_j \qquad j = 1,\ldots N \tag{32}$$

$$v^{r+1}_{sn_im_j} \ge v^r_{sn_im_j} \qquad s = 1,\ldots,S \qquad r = 1,\ldots,(R-1)$$

$$i,j = 1,\ldots,N \qquad n_i = 1,\ldots,N_i \qquad m_j = 1,\ldots,N_j \tag{33}$$

Constraint (28) guarantees that there is no flow from machine n_i to m_j if these two machines are not connected. Edges are not directed as guaranteed in constraint (29) so that once the edge between machine n_i and m_j is open, flow is allowed in both directions. Constraints (30) to (32) are used to assign flows and these are the same as constraints (10) to (12) in Section 2. Constraint (33) guarantees that flows stay the same or increases, as demand level increases for job s.

A summary of the flow-map heuristic is illustrated in Figure 1.

4 Experimental results

To evaluate the performance of the proposed heuristic, numerical experiments were conducted on examples which were randomly generated with different sizes from ten to forty machines. We applied the iterative heuristic method of Benjaafar and Sheikhzadeh (2000) on the same examples, and used the their results as a benchmark to assess the computational effectiveness of our procedure. We have not been able to find cases from the literature that we could base our tests on.

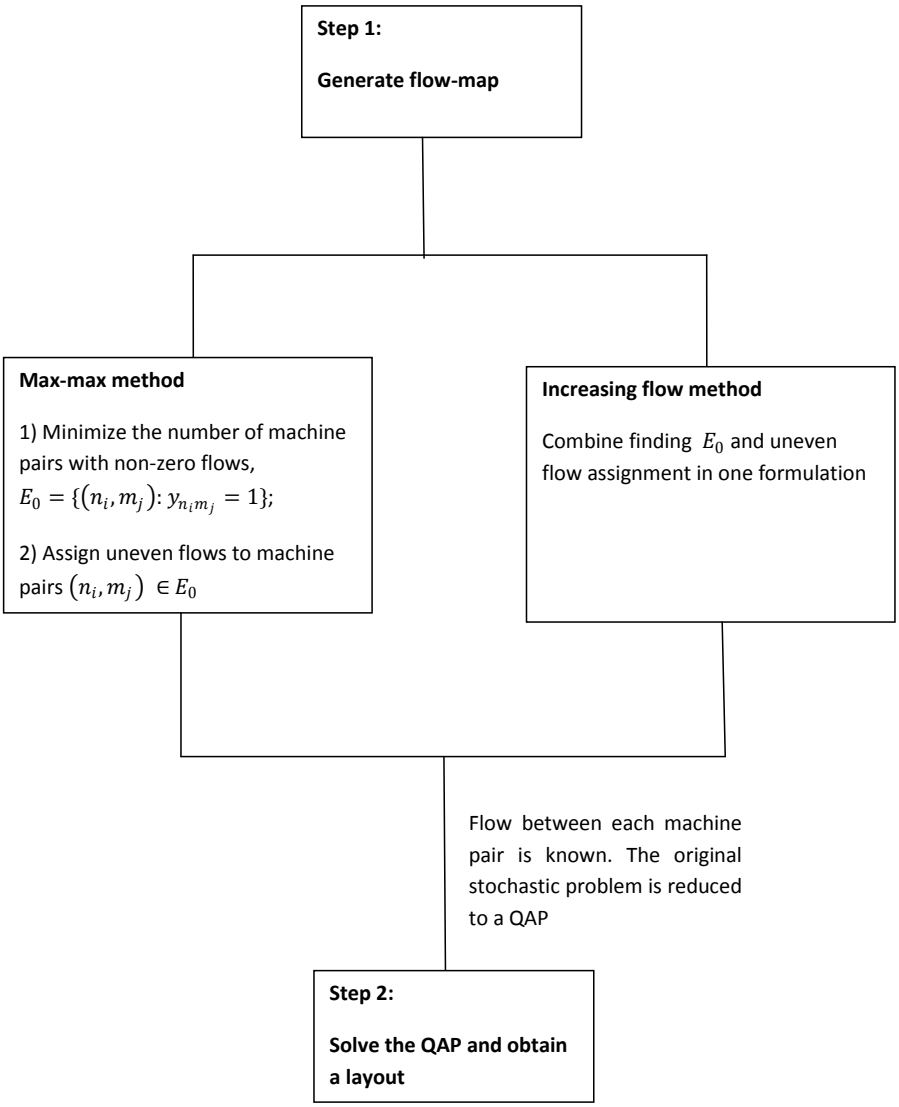

Figure 1: Summary of the flow-map heursitic

- The number of machine types is randomly generated in the range from 3 to $\left\lceil \frac{K}{2} \right\rceil$, where K is the total number of machines (and locations).
- The number of copies of different types is also randomly generated, subject to the constraints that there is at least one copy for each type and each machine is assigned to a type.
- The number of jobs is chosen in the range from 3 to 10. Probabilities for different jobs are assigned randomly between 0 and 1, constrained by the fact that they must sum to 1.
- The number of machine types operated in different jobs is randomly generated in the range from 3 to N, where N is the total number of machine types. These machine types will be randomly picked from all available machine types for each job. For example, assume that the total number of machine types is 5 (indexed as from 1 to 5). A randomly generated job, $Job1$, processes on 4 machine types. The 4 types are randomly picked up from the available 5 types in an order, such as (2,3,1,5). Hence, the operation sequence of $Job1$ is $2 \rightarrow 3 \rightarrow 1 \rightarrow 5$. We do

not consider the case where a machine type is visited several times for the same job.

- The maximal demand levels for the different jobs are randomly generated in the range from 150 to 300 units. The demand level of a job is continuous and uncertain, following a Beta distribution defined from 0 to the job's maximal demand level with random shape parameters. When scenarios are needed, the continuous distribution is then discretized at three demand levels, i.e. $\frac{1}{3}D_{max}$, $\frac{2}{3}D_{max}$ and D_{max} (where D_{max} is the max demand of a job). Probabilities of these levels are obtained according to the beta distribution. For example, the uncertain demand level of a job might be following a beta distribution ranging from 0 to 300 with shape parameters 4 and 5. Three discrete scenarios are then generated for demand level 100, 200 and 300, with probabilities 0.26, 0.65 and 0.09. These probabilities are obtained from the generated beta distribution by calculating the probability of the demand level falling in each of the ranges $[0,100]$, $[100,200]$ and $[200,300]$ because the machine pairs used for the upper demand levels, i.e. 100, 200, 300, are feasible for lower demand volumes in the three ranges as well. Since each job has three possible demand levels, for a case where there are S jobs, the total number of different production scenarios (i.e. different jobs or the same job with different demand levels) is $3S$.
- Processing times for a given job are assumed equal for machine copies of the same type. However, machine copies of different types can have different processing times. The processing times for different machine types and jobs are determined by randomly sampling from $U(5,10)$.
- The capacity available for a machine type, i.e. total available time, will increase proportionally with the number of copies of that machine type. The available capacity is then set equal to the highest workload needed to meet the highest demand level over all jobs, making sure that all problems are feasible.
- The number of available locations is equal to the number of machines. We assume that the facility is rectangular with multiple rows and columns. The taxi (Manhattan) distance between the centroids of the individual locations is used. A door is placed in the corner so for each job the production process starts by moving materials from the door to the starting machine and ends by moving finished parts from the last machine to the door. An example of 19 locations is illustrated in Figure 2.

4.1 Evaluating heuristic results

The outcomes of the heuristics are compared by evaluating the minimum expected material flow costs using formula (34), which sums costs over all jobs taking into account their continuous demand distributions:

$$\sum_{s=1}^{S} P_s \times \int_{\text{min\_demand}}^{\text{max\_demand}} f(s,h)g(s,h)\,d\text{h} \tag{34}$$

where $g(s,h)$ is the density function for the continuous demand distribution of job s, assumed as beta distribution in this paper. The cost function $f(s,h)$ is obtained by the linear model (35):

$$f(s,h) = \min \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{n_i=1}^{N_i}\sum_{m_j=1}^{N_j} v_{n_i m_j} d_{n_i m_j} \tag{35}$$

Subject to: Constraints (10) (12) (13)

$$\text{and} \sum_{n_i=1}^{N_i}\sum_{m_j=1}^{N_j} v_{n_i m_j} = h \qquad i,j = 1,\ldots,N \tag{36}$$

which calculates the total flow costs for commodity s, given demand h when machine locations (and hence costs $d_{n_i m_j}$) are known. As (35) is linear, we solve it parametrically in h, obtaining an explicit piecewise linear function $f(s,h)$ that we then use for the integral in (34). Hence, for evaluation, expected costs are calculated exactly.

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	DOOR

Figure 2: Locations with a door: an example of 19 locations.

4.2 Numerical results

The proposed flow-map heuristic contains two steps: 1) generate a flow map; 2) solve a reduced QAP. The QAP is a well known NP-hard problem and exact methods are only computationally efficient for small-sized problems. We split our test into two parts. We first look at cases where the resulting QAP can be solved to optimality. This has the advantage (from a testing perspective) that all errors come from our heuristic, and hence, it is easier to understand how the heuristics work. This can be done for up to 15 machines. For larger problems, the QAP will be solved heuristically as well, as we have no other choice.

Two flow-map heuristics, using both max-max (Max) and increasing flow (Inc), is benchmarked against the minimum expected cost of an iterative (Iter) method from Benjaafar and Sheikhzadeh (2000), which we start from *five random initial layouts*. It is not possible to compare with optimal solutions, as they are unavailable due to the complexity of the stochastic model. 100 cases are randomly generated for each problem size from 10 to 14. The average cost reductions for the Max and Inc heuristics (named (Iter-Max)% and (Iter-Inc)%) in Table 1) are measured in percentage deviations from the Iter heuristic. Average computational times of the three methods are shown in Table 2. It is seen from these two tables that our flow-map heuristic improves the layout by cutting expected cost, while computing much faster.

Our heuristic and the random generator were written in C++ interfaced with CPLEX 12.1. All reported CPU times are from a personal computer with Intel Core 2 Duo 2.99GHZ CPU and 3.25GB of RAM.

It is shown in Table 1 that the computational efficiency of the flow-map heuristic improves as the problem size grows compared to the iterative method. The reason is that the benchmark iterative method needs more initial random layouts to find a good layout when the number of machines increases. But this will significantly increase the CPU time if starting from more random layouts. So the benchmark iterative method can certainly produce better results, but at the cost of a substantial increase in CPU time.

To assess the performance of our heuristic for larger-sized problems, we carried out numerical

Table 1: Cost reductions (in percent) for the flow-map heuristic for problem size from 10 to 14.

number of machines	(Iter-Max)%		(Iter-Inc)%	
	avg	std	avg	std
10	0.35	0.26	0.32	0.45
11	0.78	0.31	0.71	0.74
12	0.85	0.27	0.83	0.37
13	0.97	1.01	0.96	1.27
14	1.03	1.48	1.53	1.02

Table 2: CPU times (in seconds) of Max, Inc and Iter. The CPU savings relate our heuristic to the iterative approach with five random starts.

Number of machines	Max		Inc		Iter	
	avg (Avg CPU saving%)	std	avg (avg CPU saving%)	std	avg	std
10	2.70 (97.84%)	1.22	0.80(99.36%)	0.58	125	48
11	34.67(98.05%)	15.18	9.69(99.46%)	7.71	1778	695
12	98.28(98.20%)	97.46	60.22(98.90%)	102.66	5461	1319
13	111.81(98.97%)	107.68	69.71(99.36%)	92.88	10887	9300
14	182.96(99.16%)	192.27	158.77(99.27%)	129.26	21716	12884

experiments on cases with 20, 30, 40, 50 and 60 machines. Since the reduced QAPs in these larger cases cannot be solved to optimality within a reasonable amount of time, a heuristic method based on tabu search was applied. The data structure in our tabu search implementation is the same as that of Chiang and Chiang (1998). From Tables 1 and 2, we see that the performances of the max-max and the increasing flow methods are very close. Since these two methods only differ in the ways they generate uneven flows, we will only compare the max-max method to the iterative method for these larger problems.

To avoid that the integer program of minimizing the number of open edges (presented in Section 3.1) becomes a computational bottleneck in the max-max method, a heuristic method introduced by Zhao and Wallace (2013) is applied to obtain the set of open edges E_0. Since the integer program itself represents a heuristic idea (that it is good to have few non-zero flows) not solving it to optimality is not a major issue. Different from the FLPs studied in this paper, in which both jobs and demand levels are uncertain, the heuristic (Zhao and Wallace 2013) is designed for FLPs with a single job but uncertain demand level. This heuristic finds machine pairs connected with non-zero flows by constructing a number of flow paths which follow the visiting sequence of machine types defined by the single job. The heuristic is applied here on one job at a time, and each edge that is opened for at least one job is kept open in our flow map. This may open a few more edges than needed, but our experience is that this is rectified in the uneven flow assignment that follows the determination of which edges to open.

100 cases are randomly generated for each problem size of 20, 30, 40, 50 and 60 machines. The average cost reduction of the max-max method ((Iter-Max)%) is shown in Table 3. CPU times of the max-max and iterative methods are shown in Table 4.

Table 3: Cost reduction (in percent) for the flow-map heuristic for problem size from 20 to 60

number of machines	20	30	40	50	60
avg (Iter-Max)%	1.78	2.26	3.59	4.06	4.74
std (Iter-Max)%	1.24	1.20	2.02	1.78	2.16

As shown in the above results, the flow-map heuristic improves machine layouts further as problem size increases, with an average cost reduction of 4.74% for 60 machines compared to the results from the iterative method. The CPU time of the proposed heuristic is also less than the iterative method. For the case of 60 machines, the flow-map heuristic consumes less than 10 minutes but the iterative

Table 4: CPU times (in seconds) of Max and Iter for problem size from 20 to 60. The CPU savings relate our heuristic to the iterative approach with five random starts.

Number of machines	Max		Iter	
	avg (Avg CPU saving%)	std	avg	std
20	13.02(93.85%)	2.57	212	63
30	22.35(97.92%)	2.79	1074	63
40	56.47(98.06%)	6.49	2924	215
50	239.52(98.16%)	32.15	13022	1100
60	592.42(98.17%)	73.80	32406	2419

method takes nearly 10 hours to find a result. To better illustrate the cost reduction of the flow-map heuristic, a box plot of reduction by the flow-map heuristic ((Iter-max)%) for each problem size from 10 to 60 is shown in Fig 3. The box plot presents minimum, lower quartile, mean, upper quartile and maximum of 100 experiment results for each problem size.

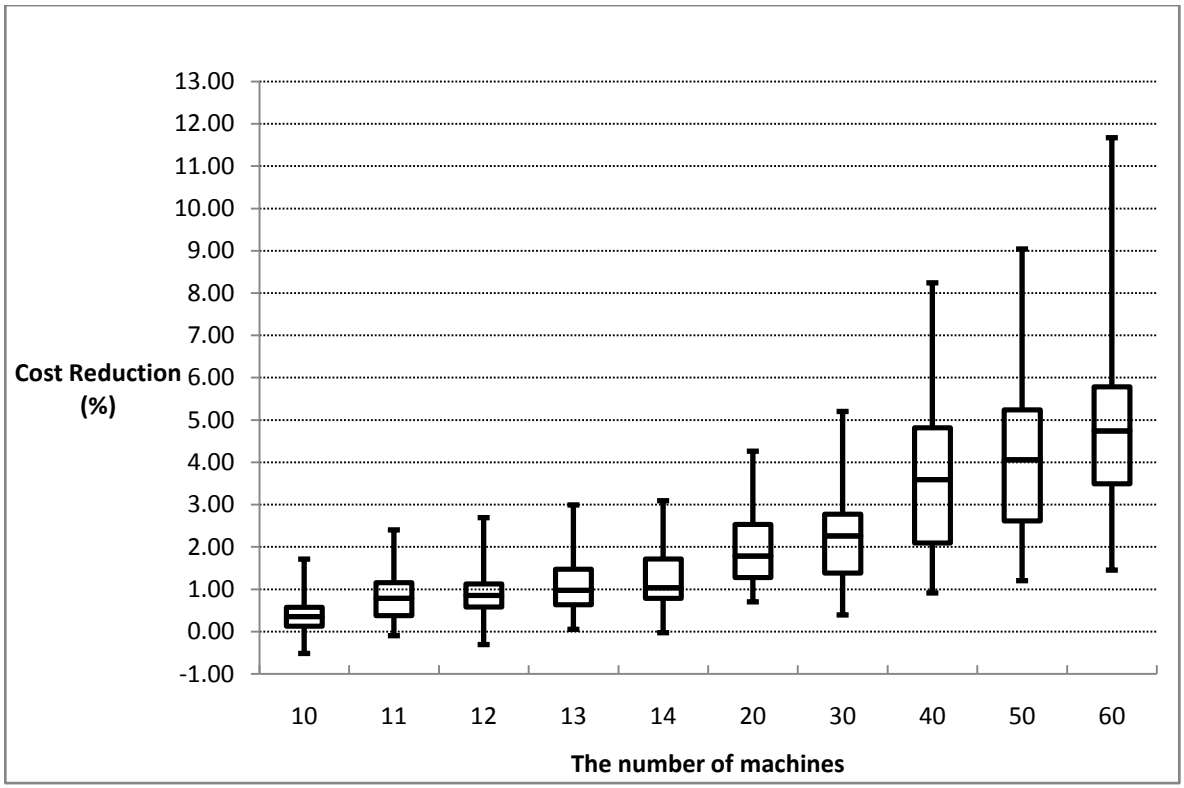

Figure 3: Box plot showing the distribution of cost reductions using the flow-map heuristic.

5 Conclusion

The design of machine layouts is one of the key issues in manufacturing since a good layout can efficiently enhance production performances, such as shortening manufacturing lead time, reducing work-in-process, increasing throughput and manual workers' satisfaction. Minimizing material handling costs is a frequently used criterion when designing efficient layouts. To effectively reduce the material handling cost, automation is normally required. This paper considers a production system

where only automatic handling tools are used to deliver material between machines. Unlike conventional fixed-route conveyors, these automatic handling tools can travel between any two points. AGVs and robots, for example, have already been deployed around the world to form parts of completely automatic production systems. More and more manufacturers start to use automatic tools instead of manned tools in order to reduce labour costs. In this paper, we also discuss uncertainty in the layout design problem such that there exists possible changes of product types and demand volumes from period to period. The problem is formulated as a two-stage stochastic integer program with the objective of minimizing the total expected costs over all scenarios. A new greedy heuristic, named as the flow-map method, is developed. The general idea of the proposed heuristic is to reduce the complex stochastic problem to a classical QAP. All existing algorithms can then be applied to solve this QAP. Compared with the existing iterative heuristic, our flow-map is both faster and more accurate. The proposed heuristic also gives satisfactory performance when problem size increases. In our numerical experiments, the flow-map heuristic on average reduced the expected material flow cost by 4.7% in problems with 60 machines, while running about 55 times faster.

What we have demonstrated here, from a more principal perspective, is that there are important cases where uncertainty can be added to deterministic models, which themselves are numerically very challenging, without increasing the computational burden to any large extent. As almost all applied problems actually face relevant uncertainty about the future, such extensions are crucial from a practical perspective since sensitivity analysis and what-if analysis do not deliver (Wallace 2000). We may ask why it was possible to develop such an approach. We believe the main issue is to understand what optimal solutions must look like – what properties they have. Such knowledge might come from a numerical (statistical) analysis of optimal solutions to small instances of the problem (such as in for example Lium et al (2009)), or, as here, from an explicit understanding of what structure a good solution must have, namely that as few machine-pairs as possible should have positive flows. This way of thinking is common in for example deterministic integer programming, where they are often expressed as valid inequalities or facets, but may also take more explicit forms such as: "Under certain conditions, optimal routes in a VRP do not cross each other". Stochastics is not different in this sense; it combines with the combinatorics to enforce structures in optimal solutions. Looking for them is crucial when understanding how to develop good heuristics.

Future work can include studies of how design factors affect layout of machines in a completely automatic production system. The factors can include the the shape of the factory floor, the number of copies of each machine type, or the number of doors in a factory for receiving and dispatching materials. We may also study cases of redundant machines as that can substantially reduce material handling costs.

References

Balakrishnan J, Cheng CH, Wong KF (2003) Facopt: a user friendly facility layout optimization system. Computers and Operations Research 30(11):1625–1641

Baykasoglu A, Cindy NNZ (2001) A simulated annealing algorithm for dynamic plant layout problem. Computers and Operations Research 28(14):1403–1426

Benjaafar S, Sheikhzadeh M (2000) Design of flexible plant layouts. IIE Transactions 32(4):309–322

Bukard RE, Bonniger T (1983) A heuristic for quadratic boolean programs with applications to quadratic assignemtn problems. European Journal of Operational Research 13(4):347–386

Chiang W, Chiang C (1998) Intelligent local search strategies for solving facility layout problems with the quadratic assignment problem fromulation. European Journal of Operational Research 106(2–3):457–488

Cooper L (1963) Location-allocation problems. Operations Research 11(3):331–343

Francis RL, White JA (1974) Facility layout and location: an analytical approach. Prentice Hall, Englewood Cliffs, NJ

Helm SA, Hadley SW (2000) Tabu search based heuristics for multi floor facility layout. International Journal of Production Research 38(2):365–383

Heragu SS, Alfa AS (2003) Experimental analysis of simulated annealing based algorithms for the layout problem. European Journal of Operational Research 57(2):190–202

Heragu SS, Kusiak A (1988) Machine layout problem in flexible manufacturing systems. Operational Research 36(2):258–268

Koopmans TC, Beckman M (1957) Assignment problems and the location of economic activities. Econometrica 25(1):53–76

Lee KY, Han SN, Roh M (2003) An improved genetic algorithm for facility layout problems having inner structure walls and passage. Computers and Operations Research 30(1):117–138

Lium AG, Crainic TG, Wallace SW (2009) A study of demand stochasticity in stochastic network design. Transportation Science 43(2):144–157, DOI 10.1287/trsc.1090.0265

Misevicius A (2003) A modified simulated annealing algorithm for quadratic assignment problem. Informatica 14(4):497–514

Salimanpur M, Jafari A (2008) Optimal solution for the two-dimensional facility layout problem using a branch-and-bound algorithm. Computers and Industrial Engineering 55(3):609–619

Singh SP, Sharma RRK (2006) A review of different approaches to the facility layout problems. International Journal of Advanced Manufacturing Technology 30(5–6):425–433

Singh SP, Singh VK (2010) An improved heuristic approach for multi-objective facility layout problem. International Journal of Production Research 48(4):1171–1194

Taghavi A, Murat A (2011) A heuristic procedure for the integrated facility layout design and flow assignment problem. Computer and Industrial Engineering 61(1):55–63

Tavakkoli-Moghaddain R, Shanyan E (1998) Facility layout design by genetic algorithms. Computers and Industrial Engineering 35(3):527–530

Tomkins JA, Bozer YA, Frazelle EH, Tanchoco JMA, Trevino J (2003) Facility Planning. John Wiley and Sons, New York

Urban TL, Chiang W, Russell RA (2000) The integrated machine allocation and layout problem. International Journal of Production Research 38(12):2911–2930

Wallace SW (2000) Decision making under uncertainty: Is sensitivity analysis of any use? Operations Research 48(1):20–25

Zhang H, Royo C, Constatino M (2010) Effective formulation reductions for the quadratic assignment problem. Computers and Operationas Research 37(11):2007–2016

Zhao Y, Wallace SW (2013) Facility layout design with random demand and capacitated machines, working paper No. 2012:9, Department of Management Science, Lancaster University Management School