

Large production line optimisation using simulated annealing^{*†}

Diomidis Spinellis[‡]
Chrissoleon Papadopoulos[§]
J. MacGregor Smith[¶]

November 20, 2000

Abstract

We present a robust generalised queuing network algorithm as an evaluative procedure for optimising production line configurations using simulated annealing. We compare the results obtained with our algorithm to those of other studies and find some interesting similarities but also striking differences between them in the allocation of buffers, numbers of servers, and their service rates. While context dependent, these patterns of allocation are one of the most important insights which emerge in solving very long production lines. The patterns, however, are often counter-intuitive, which underscores the difficulty of the problem we address. The most interesting feature of our optimisation procedure is its bounded execution time, which makes it viable for optimising very long production line configurations. Based on the bounded execution time property, we have optimised configurations of up to 60 stations with 120 buffers and servers in less than five hours of CPU time.

Keywords: *Buffer Allocation, Non-linear, Stochastic, Integer, Network Design, Simulated Annealing*

^{*}International Journal of Production Research, 38(3):509–541, February 2000.

[†]This is a machine-readable rendering of a working paper draft that led to a publication. The publication should always be cited in preference to this draft using the reference in the previous footnote. This material is presented to ensure timely dissemination of scholarly and technical work. Copyright and all rights therein are retained by authors or by other copyright holders. All persons copying this information are expected to adhere to the terms and constraints invoked by each author's copyright. In most cases, these works may not be reposted without the explicit permission of the copyright holder.

[‡]Department of Information and Communication Systems, University of the Aegean, 83200 Karlovassi Samos, Greece. (Corresponding author); contact address: Myrsinis 1, GR-145 62 Kifisia, GREECE. Tel. +30 1 2719710, Fax +30 1 2719712, email: dspin@aegean.gr

[§]Department of Business Administration, University of the Aegean, 82100 Chios, Greece. email: hpap@aegean.gr

[¶]Department of Mechanical and Industrial Engineering, University of Massachusetts, Amherst Massachusetts 01003, USA. email: jmsmith@ecs.umass.edu

1 Introduction and literature review

A large amount of research has been devoted to the analysis and design of production lines. A lot of this research concerns the design of manufacturing systems with considerable inherent variability in the processing times at the various stations, a common situation with human operators/assemblers. The literature on the modelling and optimisation of production lines is vast, allowing us to review only the most directly relevant studies. For a systematic classification of the relevant works on the stochastic modelling of these and other types of manufacturing systems (e.g., transfer lines, flexible manufacturing systems (FMS), and flexible assembly systems (FAS)), the interested reader is referred to a review paper by Papadopoulos and Heavey (1996) and some recently published books, such as those by Askin and Standridge (1993), Buzacott and Shanthikumar (1993), Gershwin (1994), Papadopoulos *et al.* (1993), Viswanadham and Narahari (1992), and Altiok (1997).

There are two basic problem classes:

1. the evaluation of production line performance measures such as throughput, mean flow time, workstation mean queue length, and system utilisation, and
2. the optimisation of the decision variables of these lines.

Examples of decision variables that have been considered are:

1. the sizes of the buffers placed between successive workstations of the lines,
2. the number of servers allocated to each workstation, and
3. the amount of workload allocated to each workstation.

The corresponding optimisation problems are named, respectively, (1) *the buffer allocation problem*, (2) *the server allocation problem*, and (3) *the workload allocation problem*, in a production line.

In Papadopoulos *et al.* (1993) both evaluative and generative (optimisation) models are given for modelling the various types of manufacturing systems. This work falls into

the second category. Evaluative and optimisation models can be combined by closing the loop between them; that is, one can use feedback from an evaluative model to modify the decisions taken by the optimisation model.

One of the key questions that the designers face in a serial production line is the buffer allocation problem (BAP), i.e., how much buffer storage to allow and where to place it within the line. This is an important question because buffers can have a great impact on the efficiency of the production line. They compensate for the blocking and the starving of the line's stations. For this reason, the buffer allocation problem has received a lot more attention than the other two design problems. Buffer storage is expensive due both to its direct cost, and to the increase of the work-in-process (WIP) inventories. In addition, the requirement to limit the buffer storage can also be a result of space limitations in the shop floor. The literature on the BAP is extensive. A systematic classification of the research work in this area is given in Singh and MacGregor Smith (1997) and Papadopoulos *et al.* (1993). The works are split according to:

The method used to solve the buffer allocation problem

search methods, (Altiok and Stidham, 1983, Smith and Daskalaki, 1988, Seong *et al.*, 1995, Hillier and So, 1991a, Hillier and So, 1991b), *dynamic programming methods*, (Kubat and Sumita, 1985, Jafari and Shanthikumar, 1989, Yamashita and Altiok, 1997), among others, and *simulation methods*, (Conway *et al.*, 1988, Ho *et al.*, 1979).

The type of production line *balanced/unbalanced*, (Powell (1994) presents a literature review according to this scheme), or *reliable/unreliable*; the majority of papers deal with reliable lines. Only a few algorithms have been developed to calculate the performance measures of unreliable production lines (Glassey and Hong, 1983, Choong and Gershwin, 1987, Gershwin, 1989, Heavey *et al.*, 1993). Hillier and So (1991a) and Seong, Chang, and Hong (1995) have dealt with the buffer allocation problem in unreliable production lines.

The number of machines available at each workstation

Similarly, few researchers have dealt with the buffer allocation problem in production lines with multiple (parallel) machines at each workstation (Hillier and So, 1995, Magazine and Stecke, 1996, Singh and Smith, 1997).

Apart from the buffer allocation problem, the other two interesting design problems have also been considered by some researchers, e.g., the work allocation problem (Hillier and Boling, 1979, Hillier and Boling, 1966, Hillier and Boling, 1977, Ding and Greenberg, 1991, Huang and Weiss, 1990, Shanthikumar *et al.*, 1991, w. Wan and Wolff, 1993,

Yamazaki *et al.*, 1992), among others, and the server allocation problem (Magazine and Stecke, 1996, Hillier and So, 1989). Hillier and So (1995) studied various combinations of these three design problems. Other references may be found therein (Buzacott and Shanthikumar, 1992, Buzacott and Shanthikumar, 1993), among others.

The present work deals with the same design problems (buffer allocation, server allocation, and workload allocation) but for long production lines with multi-machine stations.

As the problem being investigated is combinatorial in nature, traditional Operations Research techniques are not as practical for obtaining optimal solutions for long production lines. We propose a simulated annealing (SA) approach as the *search method* in conjunction with the expansion method developed by Kerbache and MacGregor Smith (1987) as the *evaluative tool*. Simulated annealing is an adaptation of the simulation of physical thermodynamic annealing principles described by Metropolis *et al.* (1953) to the combinatorial optimisation problems (Kirkpatrick *et al.*, 1983, Cerny, 1985). Similar to genetic algorithms (Holland, 1975, Goldberg, 1989) and tabu search techniques (Glover, 1990) it follows the 'local improvement' paradigm for harnessing the exponential complexity of the solution space.

The algorithm is based on randomisation techniques. An overview of algorithms based on such techniques can be found in the survey by Gupta *et al.* (1994). A complete presentation of the method and its applications is described by Van Laarhoven and Aarts (1987) and accessible algorithms for its implementation are presented by Corana *et al.* (1987) and Press *et al.* (1988). A critical evaluation of different approaches to annealing schedules and other method optimisations are given by Ingber (1993). As a tool for operational research SA is presented by Eglese (1990), while Koulamas *et al.* (1994) provide a complete survey of SA applications to operations research problems.

The use of the Simulated Annealing algorithm appears to be a promising approach. We believe that this algorithm could be applied in conjunction with a fast decomposition algorithm to solve efficiently and accurately the aforementioned optimisation problems in much longer production lines.

The remainder of the paper is organised as follows: we first describe the production line model and the problem of our interest followed by the methodology of our approach namely: the performance model, the expansion method used for evaluating the line performance, an overview of the combinatorial optimisation methods, the simulated annealing optimisation method, and the complete enumeration method; we then describe our experimental methodology and present an overview of the numerical and performance results for short and long production lines. In the Appendix following the concluding section, we provide a full tabulated set of the experimental results.

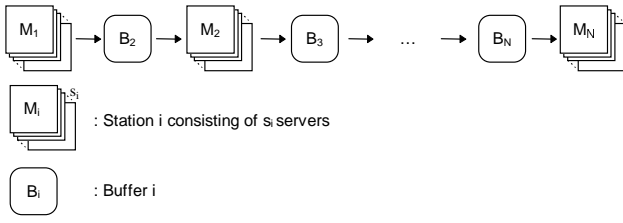


Figure 1: An N -workstation multi-machine production line with $N - 1$ intermediate buffers

2 The production line model and the optimal design problem

We define an asynchronous line as one in which every workstation can pass parts on when its processing is complete as long as buffer space is available. Such a line is subject to manufacturing starving and blocking. We assume that the first station is never starved and the last station is never blocked. Therefore we can say that the line operates in a *push* mode: parts are always available when needed at the first workstation and space is always available at the last workstation to dispose of completed parts.

An N -station line consists of N workstations in series, labelled M_1, M_2, \dots, M_N and $N - 1$ locations for buffers, labelled B_2, B_3, \dots, B_N , is illustrated in figure 1. Each station i has s_i servers operating in parallel. The buffer capacities of the intermediate buffers B_i , $i = 2, \dots, N$, are denoted by q_i , whereas the mean service times of the i stations ($i = 1, \dots, N$) are denoted by w_i .

The main performance measure of the production line is the mean throughput, denoted by $R(\underline{q}, \underline{s}, \underline{w})$, where $\underline{q} = (q_2, q_3, \dots, q_N)$, $\underline{s} = (s_1, s_2, \dots, s_N)$ and $\underline{w} = (w_1, w_2, \dots, w_N)$.

If Q denotes the total number of available buffer slots to be allocated to the $N - 1$ buffers and S the total number of available servers (machines) to be allocated to the N stations then the general version of the optimisation model (first reported by Hillier and So, 1995) is:

$$\max R(\underline{q}, \underline{s}, \underline{w}) \quad (1)$$

subject to:

$$\sum_{i=2}^N q_i = Q, \quad (2)$$

$$\sum_{i=1}^N s_i = S, \quad (3)$$

$$\sum_{i=1}^N w_i = N, \quad (4)$$

$$q_i \geq 0 \text{ and integer } (i = 2, 3, \dots, N),$$

$$s_i \geq 1 \text{ and integer } (i = 1, 2, \dots, N),$$

$$w_i > 0 \ (i = 1, 2, \dots, N),$$

where Q, S and N are fixed constants and \underline{q} , \underline{s} and \underline{w} are the decision vectors. The third constraint indicates that the sum of the expected service times is a fixed constant, which by normalisation can be equal to N .

The objective function of throughput, $R(\underline{q}, \underline{s}, \underline{w})$, is not the only performance measure of interest. The average WIP, the flow time, the cycle time, the system utilisation, the average queue lengths and other measures are equally important performance measures. However, throughput is the most commonly used performance measure in the international literature.

3 Optimal allocation methodology

3.1 Performance models

The queuing model $M/M/C/K$ that we use refers to a queuing system where:

- it is assumed that the arrivals are distributed according to the Poisson distribution (or equivalently that the intermediate times between two successive arrivals are exponentially distributed),
- the service (processing) times follow the exponential distribution,
- there are C parallel servers (machines which are identical at each workstation), and
- the total capacity of the system is finite and equal to K .

While our focus in this paper is on $M/M/C/K$ approximations for open queuing networks of series-parallel topologies, we also briefly discuss some of the available approaches used for modelling $M/M/1/K$ systems since most of the literature has focused on $M/M/1/K$ systems.

Both open and closed systems have been studied by exact analysis although results have been limited. Exact analyses of open two, three, and four node-server models with exponential service are limited by the explosive growth of the Markov chain models for analysing these systems. The analysis of very large Markov chain models has led to effective aggregation techniques for these models (Schweitzer and Altiock, 1989, Takahashi, 1989) but the computation time and power required for these exact results leaves open the need for approximation techniques.

Van Dijk and his co-authors (1988, 1989) have developed some bounding methodologies for both $M/M/1/K$

and $M/M/C/K$ systems and have demonstrated their usefulness in the design of small queuing networks. Of course, when doing optimisation of medium and long queuing networks, bounds can be far off the optimum, so robust approximation techniques close to the optimal performance measures are most desirable.

Most approximation techniques appearing in the literature rely on decomposition/aggregation methods to approximate performance measures. One and two node decompositions of the network have been carried out, all with varying degree of success.

The few approximation approaches available in the literature can be classified as follows: *Isolation methods*, *Repeated Trials*, *Node-by-node decomposition*, and *Expansion* methods. In the Isolation method, the network is subdivided into smaller subnetworks and then studied in isolation (Labetoulle and Pujolle, 1980, Boxma and Konheim, 1981). This method was used by Kuehn (1979) and Gelenbe and Pujolle (1976) but they failed to consider networks with finite capacity.

Closely related to the Isolation method is the Repeated Trials Method, a class of techniques based upon repeatedly attempting to send blocked customers to a queue causing the blocking (Caseau and Pujolle, 1979, Fredericks and Reisner, 1979, Fredericks, 1980).

In Node-by-node decomposition, the network is broken down into single, pairs, and triplets of nodes with augmented service and arrival parameters which are then studied separately (Hillier and Boling, 1967, Takahashi *et al.*, 1980, Altioik, 1982, Altioik and Perros, 1986, Perros and Altioik, 1986, Brandwajn and Jow, 1988). More general service time approximations appear in the work by Gun and Makowski (1989). The Expansion method is the approach argued for in this paper for computing the performance measures of $M/M/C/K$ finite queuing networks (Kerbache, 1984, Kerbache and Smith, 1987, Kerbache and Smith, 1988). It can be characterised conceptually as a combination of Repeated Trials and Node-by-Node Decomposition where the key difference is that a ‘holding’ node is added to the network to register *blocked* customers. The addition of the *holding* node ‘expands’ the network. This approach transforms the queuing network into an equivalent Jackson network which is then decomposed allowing for each node to be solved independently. We have successfully used the Expansion Method to model $M/M/1/K$ (Kerbache and Smith, 1988), $M/M/C/K$ (Jain and Smith, 1994, Han and Smith, 1992), $M/G/1/K$ (Kerbache and Smith, 1987), and most recently $M/G/C/C$ (Cheah and Smith, 1994, Smith, 1991, Smith, 1994) queues and queuing networks. In addition, we have also used our Expansion Methodology to model routing (Daskalaki and Smith, 1989, Daskalaki and Smith, 1986, Gosavi and Smith, 1990) and optimal resource allocation problems (Smith and Daskalaki, 1988, Smith, 1991, Smith and Chikhale, 1995, Singh and

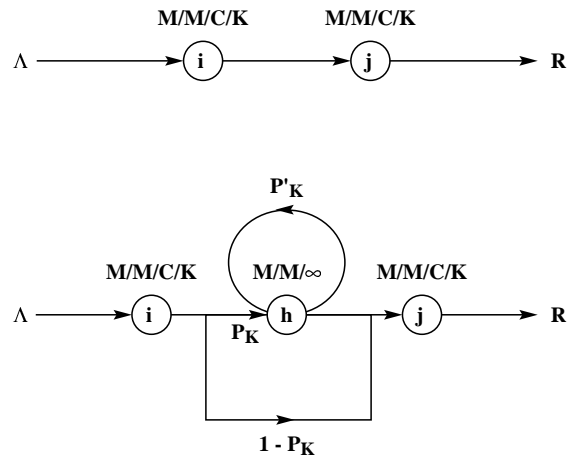


Figure 2: Type I Blocking in Finite Queues

Smith, 1997).

3.2 Expansion method

The Expansion Method is a robust and effective approximation technique developed by Kerbache and Smith (1987). As described in previous papers, this method is characterised as a combination of Repeated Trials and Node-by-node Decomposition solution procedures. Methodologies for computing performance measures for a finite queuing network use primarily the following two kinds of blocking:

Type I: The upstream node i gets blocked if the *service on a customer is completed* but it cannot move downstream due to the queue at the downstream node j being full. This is sometimes referred to as Blocking After Service (BAS) (Onvural, 1990).

Type II: The upstream node is blocked when the downstream node becomes saturated and service must be suspended on the upstream customer regardless of whether service is completed or not. This is sometimes referred to as Blocking Before Service (BBS) (Onvural, 1990).

The Expansion Method uses *Type I* blocking, which is prevalent in most production and manufacturing, transportation and other similar systems.

Consider a single node with finite capacity K (including service). This node essentially oscillates between two states — the saturated phase and the unsaturated phase. In the unsaturated phase, node j has at most $K - 1$ customers (in service or in the queue). On the other hand, when the node is saturated no more customers can join the queue. Refer to figure 2 for a graphical representation of the two scenarios.

The Expansion Method consists of the following three stages :

- *Stage I: Network Reconfiguration.*
- *Stage II: Parameter Estimation.*
- *Stage III: Feedback Elimination.*

The following notation defined by Kerbache and Smith (1987, 1988) shall be used in further discussion regarding this methodology :

h := The holding node established in the Expansion method.

Λ := External Poisson arrival rate to the network.

λ_j := Poisson arrival rate to node j .

$\tilde{\lambda}_j$:= Effective arrival rate to node j .

μ_j := Exponential mean service rate at node j .

$\tilde{\mu}_j$:= Effective service rate at node j due to blocking.

p_K := Blocking probability of finite queue of size K .

p'_K := Feedback blocking probability in the Expansion method.

p_0^j := Unconditional probability that there is no customer in the service channel at node j (either being served or being held after service).

R := Mean throughput rate.

S_j := Service capacity (buffer) at node j , i.e. $S = K$ for a single queue.

3.2.1 Stage I: network reconfiguration

Using the concept of two phases at node j , an artificial node h is added for each finite node in the network to register blocked customers. Figure 2 shows the additional delay, caused to customers trying to join the queue at node j when it is full, with probability p_K . The customers successfully join queue j with a probability $(1 - p_K)$. Introduction of an artificial node also dictates the addition of new arcs with p_K and $(1 - p_K)$ as the routing probabilities.

The blocked customer proceeds to the finite queue with probability $(1 - p'_K)$ once again after incurring a delay at the artificial node. If the queue is still full, it is re-routed with probability p'_K to the artificial node where it incurs another delay. This process continues till it finds a space in the finite queue. A feedback arc is used to model the repeated delays. The artificial node is modelled as an $M/M/\infty$ queue. The infinite number of servers is used simply to serve the blocked customer a delay time without queuing.

3.2.2 Stage II: parameter estimation

This stage essentially estimates the parameters p_K , p'_K and μ_h utilising known results for the $M/M/C/K$ model.

- p_K : Analytical results from the $M/M/C/K$ model provide the following expression for p_K :

$$p_K = \frac{1}{c^{K-c}c!} \left(\frac{\lambda}{\mu}\right)^K p_0 \quad (5)$$

where for $(\lambda/c\mu \neq 1)$

$$p_0 = \left[\sum_{n=0}^{c-1} \frac{1}{n!} \left(\frac{\lambda}{\mu}\right)^n + \frac{(\lambda/\mu)^c}{c!} \frac{1 - (\lambda/c\mu)^{K-c+1}}{1 - \lambda/c\mu} \right]^{-1} \quad (6)$$

and for $(\lambda/c\mu = 1)$,

$$p_0 = \left[\sum_{n=0}^{c-1} \frac{1}{n!} \left(\frac{\lambda}{\mu}\right)^n + \frac{(\lambda/\mu)^c}{c!} (K - c + 1) \right]^{-1} \quad (7)$$

- p'_K : Since there is no closed form solution for this quantity an approximation is used given by Labetoulle and Pujolle obtained using diffusion techniques (Labetoulle and Pujolle, 1980).

$$p'_K = \left[\frac{\mu_j + \mu_h}{\mu_h} - \frac{\lambda[(r_2^K - r_1^K) - (r_2^{K-1} - r_1^{K-1})]}{\mu_h[(r_2^{K+1} - r_1^{K+1}) - (r_2^K - r_1^K)]} \right]^{-1} \quad (8)$$

where r_1 and r_2 are the roots to the polynomial:

$$\lambda - (\lambda + \mu_h + \mu_j)x + \mu_h x^2 = 0 \quad (9)$$

while, $\lambda = \lambda_j - \lambda_h(1 - p'_K)$ and λ_j and λ_h are the actual arrival rates to the finite and artificial holding nodes respectively.

In fact, λ_j the arrival rate to the finite node is given by:

$$\lambda_j = \tilde{\lambda}_i(1 - p_K) = \tilde{\lambda}_i - \lambda_h \quad (10)$$

Let us examine the following argument to determine the service time at the artificial node. If an arriving customer is blocked, the queue is full and thus a customer is being serviced, so the arriving customer to the holding node has to remain in service at the artificial holding node for the remaining service time interval of the customer in service. The delay distribution of a blocked customer at the holding node has the same distribution as the remaining service time of the customer being serviced at the node doing the blocking. Using renewal theory, one can show that the remaining service time distribution has the following rate μ_h :

$$\mu_h = \frac{2\mu_j}{1 + \sigma^2\mu_j^2} \quad (11)$$

where, σ^2 is the service time variance given by Kleinrock (1975). Notice that if the service time distribution at the finite queue doing the blocking is exponential with rate μ_j , then:

$$\mu_h = \mu_j$$

the service time at the artificial node is also exponentially distributed with rate μ_j .

3.2.3 Stage III: feedback elimination

Due to the feedback loop around the holding node, there are strong dependencies in the arrival processes. Elimination of these dependencies requires reconfiguration of the holding node which is accomplished by recomputing the service time at the node and removing the feedback arc. The new service rate is given by:

$$\mu'_h = (1 - p'_K)\mu_h \quad (12)$$

The probabilities of being in any of the two phases (saturated or unsaturated) are p_K and $(1 - p_K)$. The mean service time at a node i preceding the finite node is μ_i^{-1} when in the unsaturated phase and $(\mu_i^{-1} + \mu_h'^{-1})$ in the saturated phase. Thus, on an average, the mean service time at the node i preceding a finite node is given by:

$$\mu_i^{\sim -1} = \mu_i^{-1} + p_K \mu_h'^{-1} \quad (13)$$

Similar equations can be established with respect to each of the finite nodes. Ultimately, we have simultaneous non-linear equations in variables p_K, p'_K, μ_h^{-1} along with auxiliary variables such as μ_j and $\tilde{\lambda}_i$. Solving these equations simultaneously we can compute all the performance measures of the network.

$$\lambda = \lambda_j - \lambda_h(1 - p'_K) \quad (14)$$

$$\lambda_j = \tilde{\lambda}_i(1 - p_K) \quad (15)$$

$$\lambda_j = \tilde{\lambda}_i - \lambda_h \quad (16)$$

$$p'_K = \left[\frac{\mu_j + \mu_h}{\mu_h} - \frac{\lambda[(r_2^K - r_1^K) - (r_2^{K-1} - r_1^{K-1})]}{\mu_h[(r_2^{K+1} - r_1^{K+1}) - (r_2^K - r_1^K)]} \right]^{-1} \quad (17)$$

$$z = (\lambda + 2\mu_h)^2 - 4\lambda\mu_h \quad (18)$$

$$r_1 = \frac{[(\lambda + 2\mu_h) - z^{\frac{1}{2}}]}{2\mu_h} \quad (19)$$

$$r_2 = \frac{[(\lambda + 2\mu_h) + z^{\frac{1}{2}}]}{2\mu_h} \quad (20)$$

$$p_K = \frac{1}{c^{K-c}c!} \left(\frac{\lambda}{\mu} \right)^K p_0 \quad (21)$$

Equations 14 to 17 are related to the arrivals and feedback in the *holding* node. The equations 18 to 20 are used

for solving equation 17 with z used as a dummy parameter for simplicity of the solution. Lastly, equation 21 gives the approximation to the blocking probability derived from the exact model for the $M/M/C/K$ queue. Hence, we essentially have five equations to solve, viz. 14 to 17 and 21. To recapitulate, we first expand the network with an artificial holding node; this stage is then followed by the approximation of the routing probabilities, due to blocking, and the service delay in the holding node; and, finally, the feedback arc at the holding node is eliminated. Once these three stages are complete, we have an expanded network which can then be used to compute the performance measures for the original network. As a decomposition technique this approach allows successive addition of a holding node for every finite node, estimation of the parameters and subsequent elimination of the holding node.

3.3 Combinatorial optimisation models

The Buffer Allocation Problem (BAP) is perhaps best formulated as a non-linear multiple-objective programming problem where the decision variables are the integers. Not only is the BAP a difficult \mathcal{NP} -hard combinatorial optimisation problem, it is made all the more difficult by the fact that the objective function is not obtainable in closed form to interrelate the integer decision variables \bar{x} and the performance measures such as throughput R , work-in-process \bar{L} , total buffers allocated $\sum_i x_i$, and other system performance measures such as system utilisation $\bar{\rho}$ for any but the most trivial situations. Combinatorial Optimisation approaches for solving problems like the BAP are generally classified as either exact optimal approaches or heuristic ones.

Exact approaches are appropriate for solving small problem instances or for problems with special structure *e.g.* the Travelling Salesman Problem, which admit optimal solutions. Classical approaches for achieving an optimal solution include Branch-and-Bound, Branch-and-Cut, Dynamic Programming, Exhaustive Search, and related implicit and explicit enumeration methods. The difficulty with utilising these exact approaches for the BAP such as Branch-and-Bound is that the subproblems for which one seeks to compute upper and lower bounds on the objective function are stochastic, non-linear programming problems which are as difficult as the original problem so little is gained by these exact problem decomposition methods.

This dilemma implies that heuristic approaches are the only reasonable methodology for large scale problem instances of the BAP problem. Heuristic approaches can be classified as either classical Non-linear Programming search methods or Metaheuristics.

Non-linear Programming (derivative-free) search (Himmelblau, 1972) methods such as, to name a few, Hooke-Jeeves, Nelder-Mead simplex methods, PARTAN, Powell's Conjugate Direction methods, Flexible Tolerance, the Com-

plex Method of Box, and other related techniques have met with varied levels of success in the BAP literature and are viable means of dealing with the BAP because of the non-closed form nature of the non-linear objective function. While many researchers feel that the objective function is concave or pseudo-concave in the decision variables, the discrete nature of the decision variables makes the problem discontinuous and so no derivative information is available.

Metaheuristic methods such as Simulated Annealing, Tabu Search, Genetic Algorithms, and related techniques have not historically been utilised to solve the BAP; in this paper we shall explore the use of Simulated Annealing.

3.4 Simulated annealing

Simulated annealing is an optimisation method suitable for combinatorial optimisation problems. Such problems exhibit a discrete, factorially large configuration space. In common with all paradigms based on ‘local improvements’ the simulated annealing method starts with a non-optimal initial configuration (which may be chosen at random) and works on improving it by selecting a new configuration using a suitable mechanism (at random in the simulated annealing case) and calculating the corresponding cost differential (ΔR_K). If the cost is reduced, then the new configuration is accepted and the process repeats until a termination criterion is satisfied. Unfortunately, such methods can become ‘trapped’ in a local optimum that is far from the global optimum. Simulated annealing avoids this problem by allowing ‘uphill’ moves based on a model of the annealing process in the physical world.

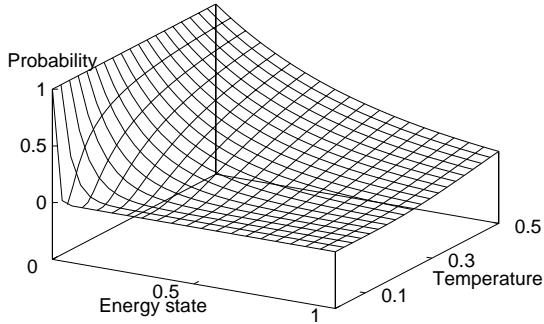


Figure 3: Probability distribution $w \sim \exp(\frac{-E}{kT})$ of energy states according to temperature.

Physical matter can be brought into a low-temperature ground state by careful annealing. First the substance is melted, then it is gradually cooled with a lot of time spent

at temperatures near the freezing point. If this procedure is not followed the resulting substance may form a glass with not crystalline order and only metastable, consisting of locally optimal structures (Kirkpatrick *et al.*, 1983). During the cooling process the system can escape local minima by moving to a thermal equilibrium of a higher energy potential based on the probabilistic distribution w of entropy S :

$$S = k \ln w \quad (22)$$

where k is Boltzmann’s constant and w the probability that the system will exist in the state it is in relative to all possible states it could be in. Thus given entropy’s relation to energy E and temperature T :

$$dS = \frac{dE}{T} \quad (23)$$

we arrive at the probabilistic expression w of energy distribution for a temperature T :

$$w \sim \exp(\frac{-E}{kT}) \quad (24)$$

This so-called Boltzmann probability distribution is illustrated in figure 3. The probabilistic ‘uphill’ energy movement that is made possible avoids the entrapment in a local minimum and can provide a globally optimal solution.

The application of the annealing optimisation method to other processes works by repeatedly changing the problem configuration and gradually lowering the temperature until a minimum is reached.

The correspondence between annealing in the physical world and simulated annealing as used for production line optimisation is outlined in table 1.

3.5 Complete enumeration

As a base-rate benchmark for our approach, we used complete enumeration (CE) of all possible buffer and server allocation possibilities. The number of feasible allocations of Q buffer slots among the $N-1$ intermediate buffer locations and S servers among the N stations increases dramatically with Q , S , and N and is given by the formula:

$$\frac{\binom{Q+N-2}{N-2}}{(Q+1)(Q+2)\dots(Q+N-2)} \frac{\binom{S+N-1}{N-1}}{(S+1)(S+2)\dots(S+N-1)} = \quad (25)$$

All buffer and server combinations can be methodically enumerated by considering a vector \underline{p} denoting the *position* within the production line of each buffer or server. Given the vector \underline{p} we can then easily map \underline{p} to \underline{q} or \underline{s} using the following equation (for the case of \underline{q}):

$$q_i = \sum_{j=1}^Q \begin{cases} 1 & \text{if } p_j = i \\ 0 & \text{otherwise} \end{cases} \quad (26)$$

Physical World		Production Optimisation	Line
Atom placement		Line configuration	
Random	atom	Buffer space, server,	
movements		service rate movement	
Energy E		Throughput R	
Energy differential ΔE		Configuration through- put differential ΔR	
Energy state probability distribution		Changes according to the Metropolis crite- rion, $\exp(\frac{-\Delta E}{T}) >$ $\text{rand}(0 \dots 1)$, imple- menting the Boltzmann probability distribution	
Temperature		Variable for establish- ing configuration ac- ceptance termination	

Table 1: Correspondence between annealing in the physical world and simulated annealing used for production line optimisation.

Given then a buffer or server configuration \underline{p} we advance to the next configuration \underline{p}' using the function g :

$$\underline{p}' = g(\underline{p}, 1) \quad (27)$$

where g is recursively defined (for Q buffers) as:

$$g(\underline{p}, l) = \begin{cases} (p_1, \dots, p_l + 1, \dots, p_Q) & \text{if } p_l + 1 \leq N \\ (p'_1, \dots, p'_{l-1}, p'_{l+1}, \dots, p'_{l+1}) & \text{otherwise} \\ \text{where } \underline{p}' = g(\underline{p}, l + 1) \end{cases} \quad (28)$$

Essentially, g maps the vector of positions to a new one representing another line configuration. When the position p_i of buffer or server resource i , as it is incremented, reaches the last place in the line N ($p_i + 1 = N$) g is recursively applied to the buffer or server in position p_{i+1} setting $p_i - p_Q$ to the new value of p_{i+1} . The complete enumeration terminates when all buffers or servers reach the line position N . To enumerate all buffer and server combinations one complete server enumeration is performed for each line buffer configuration.

4 Experimental design

The main generative procedure we used was Simulated Annealing (Kirkpatrick *et al.*, 1983, Laarhoven and Aarts, 1987, Spinellis and Papadopoulos, 1997) with complete enumeration used where practical in order to validate the SA results. In order to test the approach, 339 cases of small and large production lines from a wide combination of allocation options were searched using SA or CE.

The generalised queuing network throughput evaluation algorithm was initially ported from a VAX VMS operating system, to a PC-based Intel architecture. Most changes involved the adjustment of numerical constants according to the IEEE 488 floating point representation used on the Intel platforms. Subsequently, the algorithm was rewritten in a pure subroutine form so that it could be repeatedly called from within a program and semi-automatically converted from FORTRAN to ANSI C.

The SA algorithm used for the production line resource optimisation is given in figure 4.

The SA procedure was run with the following characteristics based on the number of stations N :

Maximum trials at given temperature $100 \times N$

Maximum successes at given temperature $10 \times N$

Initial temperature 0.5

Cooling schedule Exponential: $T_{K+1} = 0.9T_K$

Initial line configuration Equal division of buffers and servers among stations with any remaining resources placed on the station in the middle.

Reported time Elapsed wall clock time in seconds.

The following facts clarify the use of the evaluation algorithm:

- the throughput used as the line metric and presented in the tabulated results is the throughput at the last station,
- the line topology graph is a linear series of stations allowing for parallel servers, and
- the initial and the effective arrival rate at the first server are set to 1.5.

Three batches of tests were planned and executed. One, presented in tables 2–13, was planned in order to compare the results of our approach with those of Hillier and So (1995). A second set, presented in tables 14–17, was planned in order to compare the approach to the results obtained using a different evaluative procedure (Spinellis and Papadopoulos, 1997). Finally, the third batch of tests, presented in tables 18–27, aimed at establishing our method's efficacy in determining optimal configurations of large production lines. Tables 14–27 contain an additional column, T , presenting the program execution time (s).

5 Experimental results

The following paragraphs outline the results we obtained by utilising the described methodology on a variety of problems. Where applicable we compare the obtained results

1. [Set initial line configuration.] Set $q_i \leftarrow \lfloor Q/N \rfloor$, set $q_{N/2} \leftarrow q_{N/2} + Q - \sum_{i=2}^N \lfloor Q/N \rfloor$, set $s_i \leftarrow \lfloor S/N \rfloor$, set $s_{N/2} \leftarrow s_{N/2} + S - \sum_{i=1}^N \lfloor S/N \rfloor$, set $w_i \leftarrow 1/N$.
2. [Set initial temperature T_0 .] Set $T \leftarrow 0.5$.
3. [Initialize step and success count.] Set $S \leftarrow 0$, set $U \leftarrow 0$.
4. [Create new line with a random redistribution of buffer space, servers, or service rate.]
 - (a) [Create a copy of the configuration vectors.] Set $\underline{q}' \leftarrow \underline{q}$, set $\underline{s}' \leftarrow \underline{s}$, set $\underline{w}' \leftarrow \underline{w}$.
 - (b) [Determine which vector to modify.] Set $R_v \leftarrow \lfloor \text{rand}[0 \dots 2] \rfloor$.
 - (c) if $R_v = 0$ [Create new line with a random redistribution of buffer space.] Move R_n space from a source buffer q_{R_s} to a destination buffer q_{R_d} : set $R_s \leftarrow \lfloor \text{rand}[2 \dots N] \rfloor$, set $R_d \leftarrow \lfloor \text{rand}[2 \dots N] \rfloor$, set $R_n \leftarrow \lfloor \text{rand}[1 \dots q_{R_s} + 1] \rfloor$, set $q_{R_s} \leftarrow q_{R_s} - R_n$, set $q_{R_d} \leftarrow q_{R_d} + R_n$.
 - (d) if $R_v = 1$ [Create new line with a random redistribution of server allocation.] Move R_n servers from source station s_{R_s} to a destination station s_{R_d} : set $R_s \leftarrow \lfloor \text{rand}[1 \dots N] \rfloor$, set $R_d \leftarrow \lfloor \text{rand}[1 \dots N] \rfloor$, set $R_n \leftarrow \lfloor \text{rand}[1 \dots s_{R_s} + 1] \rfloor$, set $s_{R_s} \leftarrow s_{R_s} - R_n$, set $s_{R_d} \leftarrow s_{R_d} + R_n$.
 - (e) if $R_v = 2$ [Create new line with a random redistribution of service rate.] Move R_n service rate from source station s_{R_s} to a destination station s_{R_d} : set $R_s \leftarrow \lfloor \text{rand}[1 \dots N] \rfloor$, set $R_d \leftarrow \lfloor \text{rand}[1 \dots N] \rfloor$, set $R_n \leftarrow \text{rand}(0 \dots w_{R_s})$, set $w_{R_s} \leftarrow w_{R_s} - R_n$, set $w_{R_d} \leftarrow w_{R_d} + R_n$.
5. [Calculate energy differential.] Set $\Delta E \leftarrow R(\underline{q}, \underline{s}, \underline{w}) - R(\underline{q}', \underline{s}', \underline{w}')$.
6. [Decide acceptance of new configuration.] Accept all new configurations that are more efficient and, following the Boltzmann probability distribution, some that are less efficient: if $\Delta E < 0$ or $\exp(-\frac{\Delta E}{T}) > \text{rand}(0 \dots 1)$, set $\underline{q} \leftarrow \underline{q}'$, set $\underline{s} \leftarrow \underline{s}'$, set $\underline{w} \leftarrow \underline{w}'$, set $U \leftarrow U + 1$.
7. [Repeat for current temperature.] Set $S \leftarrow S + 1$. If $S < \text{maximum number of steps}$, go to step 4.
8. [Lower the annealing temperature.] Set $T \leftarrow cT$ ($0 < c < 1$).
9. [Check if progress has been made.] If $U > 0$, go to step 3; otherwise the algorithm terminates. \square

Figure 4: Simulated annealing algorithm for distributing Q buffer space, S servers, and N service rate in an N -station g line.

with other published data. In our description of the results we sometimes refer to the phenomena of the *bowl* and the *L*-shape.

The bowl phenomenon occurs when for $j = 1, 2, \dots, N$,

$$w_j^* = w_{N+1-j}^* \quad (29)$$

$$w_j^* > w_{j+1}^* \quad \left(1 \leq j \leq \frac{N-1}{2}\right) \quad (30)$$

$$w_j^* < w_{j+1}^* \quad \left(\frac{N-1}{2} \leq j \leq N+1\right) \quad (31)$$

The name arises from the fact that the value of $w_1^* = w_N^*$ is considerably larger than the other w_j^* , whereas the other w_j^* are relatively close, so that plotting the w_j^* versus j gives the shape of the cross-section of the interior of a bowl.

Correspondingly, the *L*-phenomenon is the allocation of extra servers to the first station of the production line. This leads to almost the maximum throughput of the line, which is attained when the extra servers are allocated to the last station of the line, having the advantage that it leads to less work-in-process inventory.

5.1 Short lines

Tables 2–13 present results for the optimal allocation using SA for lines similar to the ones examined by Hillier and So (1995). In all cases CE results have been calculated for comparison purposes. In comparing the results obtained with those presented by Hillier and So the following important observations can be made:

1. The service rate allocation (table 2) does not follow the bowl phenomenon, but diminishes towards the end of the line.
2. The buffer allocation (table 3) does not follow the bowl phenomenon, but increases monotonically across the line.
3. The server allocation (table 4) for a small number of servers follows a pattern strikingly similar to the one presented in Hillier and So (1995). However, as the number of servers increases, servers tend to accumulate towards the beginning of the line.
4. The server and service rate allocation (table 5) do not exhibit the *L*-phenomenon.
5. The buffer and service rate allocation (table 6) results are very similar to the results presented in Hillier and So (1995). As far as the buffer allocation is concerned, buffers tend to accumulate towards the end of the line. The allocation of work however, does not exhibit the bowl phenomenon presented in the other study and follows the usual descending rate across the line.

6. The buffer and server allocation (tables 7, 8) results are roughly similar to those presented in Hillier and So (1995) in both the server and the buffer allocation vectors. Servers tend to accumulate towards the beginning and middle of the line, whereas buffers tend to accumulate towards the line ends.
7. Finally, the buffer, server, and service rate allocation (table 9) roughly follows the shape of service rate allocation presented in Hillier and So (1995), but the allocation of buffers and servers is quite dissimilar.

Concerning the behaviour of SA compared to CE the results of the two methods are as follows:

1. Buffer allocation (tables 3, 10) is the same using SA and CE.
2. The server allocation of a large number of servers (tables 4, 11) using SA differs from the allocation using CE but only in the last two experiments, and this is probably due to incorrect results returned by the evaluation algorithm.
3. For buffer and server allocation for up to 4 stations (tables 7, 12) SA and CE give the same results.
4. For buffer and server allocation of 5 stations (tables 8, 13) SA and CE differ in the allocated vectors of some configurations, but with only a slight impact on the resulting throughput.

5.2 Long lines

The tables (16–27) present results for optimal allocation using SA for lines of $N = 4 - 70$ stations with $Q = 2N$ buffers and $S = 2N$ servers. Where practical (i.e. for small lines) CE results have been calculated for comparison purposes. The optimal allocation of a variable number of buffers to a fixed number of servers (9 servers in table 14; 15 servers table 16) follows a regular pattern: buffers are added to the line from the right to the left. The results of the CE (table 15; table 17) confirm the SA results. However, this placement strategy is different from the optimal strategy obtained using the decomposition method (Dallery and Frein, 1993) as the evaluative procedure (Spinellis and Papadopoulos, 1997).

5.3 Performance analysis

The execution time of the SA optimisation algorithm increased exponentially depending on the number of stations (figure 5). This increase was a lot better than the combinatorial explosion of CE, and allowed us to produce near-optimal configurations for relatively large production lines in reasonable time. At the extreme end for a 60 station line SA produced a solution for the allocation of 120 buffers,

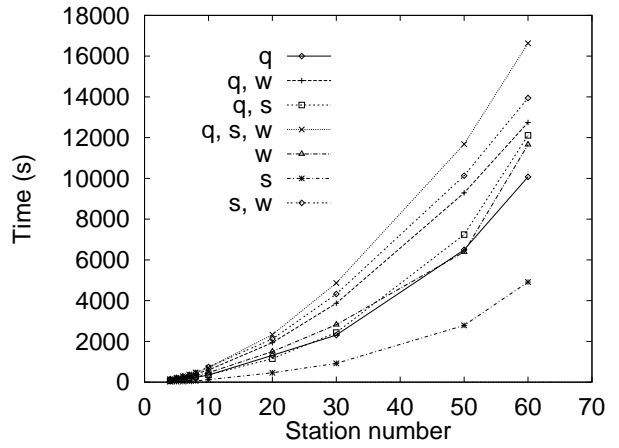


Figure 5: Execution time for optimal configuration calculations using SA.

120 servers, and the service rate configuration in 4.5 hours on a 120MHz Pentium-processor system. This solution using SA required the testing of 238,248 different configurations. The dominant factor of the algorithm is the line configuration evaluation taking on our system about 70ms for every configuration test. Multiplying this time factor by the number of all possible buffer and server allocation configurations for the above case — according to equation 25 — gives us an approximate evaluation time using CE of $\binom{120+60-2}{60-2} \binom{120+60-1}{60-1} 0.07s = 10^{68}s$ or 10^{62} years.

6 Summary and conclusions

Our approach of using the expansion method for evaluating the production line configurations generated by simulated annealing optimisation allowed us to explore small and large line configurations in bounded execution time. The results obtained with our approach, compared to those of other studies exhibit some interesting similarities but also striking differences between them in the allocation of buffers, numbers of servers, and their service rates. While context dependent, these patterns of allocation are one of the most important insights which emerge in solving very long production lines. The patterns, however, are often counter-intuitive, which underscores the difficulty of the problem we addressed.

Having demonstrated the viability of using an algorithm based on randomisation techniques for optimising large production lines we now plan to explore other optimisation methods based on similar principles such as genetic algorithms to find how they compare to our approach. The sim-

ulated annealing algorithm can be fine-tuned in a number of ways. Results for long production lines from different approaches will allow us to tune the algorithm for optimal performance in terms of execution time and derivation of optimal results and propose prescriptive guidelines for implementing production line optimisation systems.

Appendix: tabulated experimental results

N	S	Q	w	R
3	3	3	(1.04 1.01 0.951)	0.5478
4	4	4	(1.06 1.01 0.976 0.95)	0.4913
5	5	5	(1.06 1.02 0.99 0.975 0.947)	0.4493
6	6	6	(1.07 1.03 1.01 0.975 0.962 0.955)	0.4164
7	7	7	(1.08 1.05 1.01 0.966 0.968 0.969 0.965)	0.3898
8	8	8	(1.09 1.04 0.98 0.998 0.973 0.983 0.97 0.962)	0.3675
9	9	9	(1.07 1.06 0.995 0.989 0.994 0.98 0.976 0.964 0.978)	0.3487

Table 2: Optimum service rate allocation using SA.

N	S	Q	q	R
8	9	8	(1 1 1 1 1 1 1 2)	0.3809
8	10	8	(1 1 1 1 1 1 2 2)	0.3961
8	11	8	(1 1 1 1 1 2 2 2)	0.4122
8	12	8	(1 1 1 1 2 2 2 2)	0.4292
8	13	8	(1 1 1 2 2 2 2 2)	0.4469
8	14	8	(1 1 2 2 2 2 2 2)	0.4648
8	15	8	(1 2 2 2 2 2 2 2)	0.4831
6	12	6	(2 2 2 2 2 2 2)	0.5429
6	13	6	(2 2 2 2 2 3)	0.5578
4	25	4	(5 6 7 7)	0.8246
4	28	4	(5 7 8 8)	0.8415
4	29	4	(5 8 8 8)	0.8462
4	30	4	(5 8 8 9)	0.8509

Table 3: Optimum buffer allocation using SA.

N	S	Q	s	R
7	7	8	(1 2 1 1 1 1 1)	0.421
7	7	9	(1 2 1 1 1 2 1)	0.4614
7	7	10	(1 2 1 2 1 2 1)	0.5132
7	7	11	(2 2 1 2 1 2 1)	0.5753
5	5	6	(1 2 1 1 1)	0.4992
5	5	7	(1 2 1 2 1)	0.5683
5	5	8	(2 2 1 2 1)	0.6594
5	5	9	(2 2 2 2 1)	0.7573
3	3	5	(2 2 1)	0.8051
3	3	47	(27 8 12)	1
3	3	92	(89 1 2)	1

Table 4: Optimum server allocation using SA.

N	S	Q	s	w	R
5	5	6	(1 1 1 2 1)	(1.16 1.11 1.05 0.795 0.878)	0.512
5	5	8	(2 2 1 2 1)	(0.995 0.789 1.16 0.948 1.11)	0.678
5	5	10	(2 2 2 2 2)	(1.31 0.917 0.934 0.923 0.918)	0.8651

Table 5: Optimum server and service rate allocation using SA.

N	S	Q	q	w	R
3	4	3	(1 1 2)	(1.08 1.04 0.885)	0.5928
3	5	3	(1 2 2)	(1.08 0.972 0.943)	0.6371
3	6	3	(2 2 2)	(1.04 0.998 0.959)	0.6685
3	7	3	(2 2 3)	(1.07 1.0931)	0.6986
3	8	3	(2 3 3)	(1.07 0.978 0.953)	0.7258
4	5	4	(1 1 1 2)	(1.09 1.04 1 0.862)	0.5258
4	6	4	(1 1 2 2)	(1.12 1.06 0.912 0.908)	0.5598
4	7	4	(1 2 2 2)	(1.09 0.988 0.957 0.961)	0.5926
4	8	4	(2 2 2 2)	(1.06 1.0977 0.966)	0.6157
4	9	4	(2 2 2 3)	(1.07 1.02 0.998 0.914)	0.6397
4	10	4	(2 2 3 3)	(1.09 1.02 0.945 0.938)	0.6621
5	6	5	(1 1 1 1 2)	(1.1 1.06 1.02 0.986 0.834)	0.477
5	7	5	(1 1 1 2 2)	(1.12 1.08 1.02 0.895 0.877)	0.5049
5	8	5	(1 1 2 2 2)	(1.14 1.07 0.944 0.925 0.917)	0.5319
5	9	5	(1 2 2 2 2)	(1.12 1.0965 0.968 0.944)	0.5577
5	10	5	(1 2 2 2 3)	(1.14 1.01 0.991 0.97 0.889)	0.5764
5	11	5	(2 2 2 2 3)	(1.08 1.03 1.01 0.985 0.899)	0.5956
5	12	5	(2 2 2 3 3)	(1.09 1.03 1.01 0.932 0.927)	0.6147
5	13	5	(2 2 3 3 3)	(1.1 1.04 0.956 0.962 0.947)	0.6324

Table 6: Optimum buffer and service rate allocation using SA.

N	S	Q	q	s	R
3	3	4	(1 1 1)	(1 2 1)	0.6487
3	4	4	(2 1 1)	(1 2 1)	0.6918
3	5	4	(2 2 1)	(1 2 1)	0.7329
3	6	4	(2 2 2)	(1 2 1)	0.7647
3	7	4	(3 2 2)	(1 2 1)	0.7906
3	3	5	(1 1 1)	(2 2 1)	0.8051
3	4	5	(1 1 2)	(2 2 1)	0.8447
3	5	5	(1 1 3)	(2 2 1)	0.8698
3	6	5	(1 1 4)	(2 2 1)	0.8872
3	7	5	(1 1 5)	(2 2 1)	0.8999
3	4	6	(2 1 1)	(2 2 2)	1.037
3	5	6	(3 1 1)	(2 2 2)	1.09
3	6	6	(3 1 2)	(2 2 2)	1.134
3	7	6	(3 2 2)	(2 2 2)	1.181
4	5	5	(1 1 1 2)	(1 2 1 1)	0.6008
4	6	5	(2 1 1 2)	(1 2 1 1)	0.6323
4	7	5	(2 2 1 2)	(1 2 1 1)	0.6607
4	8	5	(2 2 1 3)	(1 2 1 1)	0.687
4	5	6	(1 1 1 2)	(2 2 1 1)	0.7065
4	6	6	(1 1 1 3)	(2 2 1 1)	0.7385
4	7	6	(1 1 2 3)	(2 2 1 1)	0.7648
4	8	6	(1 1 2 4)	(2 2 1 1)	0.786
4	5	7	(1 1 1 2)	(2 2 2 1)	0.817
4	6	7	(1 1 1 3)	(2 2 2 1)	0.8403
4	7	7	(1 1 2 3)	(2 2 2 1)	0.8569
4	8	7	(1 2 2 3)	(2 2 2 1)	0.8737
4	5	8	(2 1 1 1)	(2 2 2 2)	0.9762
4	6	8	(3 1 1 1)	(2 2 2 2)	1.019
4	7	8	(3 1 1 2)	(2 2 2 2)	1.055
4	8	8	(3 1 2 2)	(2 2 2 2)	1.093

Table 7: Optimum buffer and server allocation (3–4 stations) using SA.

N	S	Q	q	s	R
5	6	6	(1 1 1 1 2)	(1 2 1 1 1)	0.5304
5	7	6	(1 2 1 1 2)	(1 1 2 1 1)	0.5603
5	8	6	(2 1 1 2 2)	(1 2 1 1 1)	0.5884
5	9	6	(2 2 1 2 2)	(1 2 1 1 1)	0.6097
5	6	7	(1 1 1 1 2)	(2 2 1 1 1)	0.5984
5	7	7	(1 1 1 2 2)	(2 2 1 1 1)	0.6426
5	8	7	(1 1 1 2 3)	(2 2 1 1 1)	0.6669
5	9	7	(1 1 1 3 3)	(2 2 1 1 1)	0.6913
5	6	8	(1 1 1 2 1)	(2 2 1 2 1)	0.6939
5	7	8	(1 1 1 1 3)	(2 2 2 1 1)	0.7213
5	8	8	(1 1 1 2 3)	(2 2 2 1 1)	0.7468
5	9	8	(1 1 2 2 3)	(2 2 1 2 1)	0.7649
5	6	9	(2 1 1 1 1)	(2 2 2 2 1)	0.8065
5	7	9	(2 1 1 1 2)	(2 2 2 2 1)	0.8463
5	8	9	(2 1 1 1 3)	(2 2 2 2 1)	0.8716
5	9	9	(2 1 1 1 4)	(2 2 2 2 1)	0.889
5	6	10	(2 1 1 1 1)	(2 2 2 2 2)	0.9262
5	7	10	(3 1 1 1 1)	(2 2 2 2 2)	0.9611
5	8	10	(3 1 1 1 2)	(2 2 2 2 2)	0.9916
5	9	10	(3 1 1 2 2)	(2 2 2 2 2)	1.024

Table 8: Optimum buffer and server allocation (5 stations) using SA.

N	S	Q	q	s	w	R
3	4	4	(1 2 1)	(1 2 1)	(1.2 0.778 1.02)	0.7228
3	5	4	(2 2 1)	(1 2 1)	(1.16 0.789 1.05)	0.7686
3	6	4	(2 2 2)	(1 2 1)	(1.2 0.767 1.04)	0.8052
3	7	4	(2 3 2)	(1 2 1)	(1.19 0.773 1.04)	0.8317
3	4	5	(2 1 1)	(2 2 1)	(0.913 0.823 1.26)	0.9071
3	5	5	(2 1 2)	(2 2 1)	(0.93 0.836 1.23)	0.9481
3	6	5	(3 1 2)	(2 2 1)	(0.874 0.859 1.27)	0.9881
3	7	5	(3 2 2)	(2 2 1)	(0.883 0.802 1.31)	1.027
3	4	6	(2 1 1)	(2 2 2)	(1.09 0.964 0.948)	1.041
3	5	6	(3 1 1)	(2 2 2)	(1.02 0.994 0.987)	1.091
3	6	6	(3 1 2)	(2 2 2)	(1.06 1.03 0.914)	1.139
3	7	6	(3 2 2)	(2 2 2)	(1.09 0.96 0.954)	1.185
4	5	5	(1 1 1 2)	(1 2 1 1)	(1.22 0.804 0.989 0.983)	0.6161
4	6	5	(1 2 2 1)	(1 1 2 1)	(1.19 1.04 0.778 0.99)	0.6515
4	7	5	(2 2 2 1)	(1 1 2 1)	(1.15 1.07 0.767 1.02)	0.6833
4	8	5	(2 2 2 2)	(1 1 2 1)	(1.16 1.09 0.772 0.971)	0.7103
4	5	6	(1 1 2 1)	(2 1 2 1)	(0.95 1.15 0.797 1.11)	0.7299
4	6	6	(2 2 1 1)	(1 2 2 1)	(1.27 0.794 0.797 1.14)	0.7681
4	7	6	(2 2 1 2)	(1 2 2 1)	(1.31 0.798 0.791 1.1)	0.7997
4	8	6	(2 1 3 2)	(2 1 2 1)	(0.86 1.21 0.798 1.13)	0.8367
4	5	7	(2 1 1 1)	(2 2 2 1)	(0.965 0.853 0.862 1.32)	0.8766
4	6	7	(2 1 1 2)	(2 2 2 1)	(0.988 0.883 0.869 1.26)	0.9121
4	7	7	(3 1 1 2)	(2 2 2 1)	(0.923 0.897 0.894 1.29)	0.9449
4	8	7	(2 1 3 2)	(2 1 2 2)	(1.01 1.39 0.803 0.801)	0.9404
5	6	6	(1 1 1 1 2)	(1 1 2 1 1)	(1.19 1.14 0.796 0.932 0.94)	0.5438
5	7	6	(1 1 1 2 2)	(1 2 1 1 1)	(1.23 0.809 0.986 0.99 0.983)	0.5759
5	6	7	(1 1 2 1 1)	(1 1 2 2 1)	(1.3 1.19 0.743 0.748 1.02)	0.6226
5	7	7	(1 2 2 1 1)	(1 1 2 2 1)	(1.27 1.1 0.788 0.786 1.06)	0.6556
5	6	8	(1 2 1 1 1)	(1 2 2 2 1)	(1.43 0.803 0.796 0.8 1.17)	0.7202
5	7	8	(2 1 1 1 2)	(2 2 2 1 1)	(0.908 0.809 0.801 1.24 1.24)	0.769

Table 9: Optimum buffer, server, and service rate allocation using SA.

N	S	Q	q	R
8	9	8	(11111112)	0.3809
8	10	8	(11111122)	0.3961
8	11	8	(11112222)	0.4122
8	12	8	(11122222)	0.4292
8	13	8	(11222222)	0.4469
8	14	8	(12222222)	0.4648
8	15	8	(22222222)	0.4831
6	12	6	(222222)	0.5429
6	13	6	(222223)	0.5578
4	25	4	(5677)	0.8246
4	28	4	(5788)	0.8415
4	29	4	(5888)	0.8462
4	30	4	(5889)	0.8509

Table 10: Optimum buffer allocation using CE.

N	S	Q	g	R
7	7	8	(12111111)	0.421
7	7	9	(1211121)	0.4614
7	7	10	(1212121)	0.5132
7	7	11	(2212121)	0.5753
5	5	6	(121111)	0.4992
5	5	7	(12121)	0.5683
5	5	8	(22121)	0.6594
5	5	9	(22221)	0.7573
3	3	5	(221)	0.8051
3	3	47	(151517)	4.109e+004
3	3	92	(293231)	41.89

Table 11: Optimum server allocation using CE.

N	S	Q	q	g	R
3	4	4	(211)	(121)	0.6918
3	5	4	(221)	(121)	0.7329
3	6	4	(222)	(121)	0.7647
3	7	4	(322)	(121)	0.7906
3	4	5	(112)	(221)	0.8447
3	5	5	(113)	(221)	0.8698
3	6	5	(114)	(221)	0.8872
3	7	5	(115)	(221)	0.8999
3	4	6	(211)	(222)	1.037
3	5	6	(311)	(222)	1.09
3	6	6	(312)	(222)	1.134
3	7	6	(322)	(222)	1.181
4	5	5	(1112)	(1211)	0.6008
4	6	5	(2112)	(1211)	0.6323
4	7	5	(2212)	(1211)	0.6607
4	8	5	(2213)	(1211)	0.687
4	5	6	(1112)	(2211)	0.7065
4	6	6	(1113)	(2211)	0.7385
4	7	6	(1123)	(2211)	0.7648
4	8	6	(1124)	(2211)	0.786
4	5	7	(1112)	(2221)	0.817
4	6	7	(1113)	(2221)	0.8403
4	7	7	(1123)	(2221)	0.8569
4	8	7	(1223)	(2221)	0.8737
4	5	8	(2111)	(2222)	0.9762
4	6	8	(3111)	(2222)	1.019
4	7	8	(3112)	(2222)	1.055
4	8	8	(3122)	(2222)	1.093

Table 12: Optimum buffer and server (3–4 stations) allocation using CE.

N	S	Q	q	g	R
5	6	6	(11112)	(12111)	0.5304
5	7	6	(11122)	(12111)	0.5639
5	8	6	(21122)	(12111)	0.5884
5	9	6	(22122)	(12111)	0.6097
5	6	7	(11112)	(22111)	0.5984
5	7	7	(11122)	(22111)	0.6426
5	8	7	(11123)	(22111)	0.6669
5	9	7	(11133)	(22111)	0.6913
5	6	8	(11121)	(22121)	0.6939
5	7	8	(11122)	(22121)	0.7218
5	8	8	(11123)	(22211)	0.7468
5	9	8	(11124)	(22211)	0.7665
5	6	9	(21111)	(22221)	0.8065
5	7	9	(21112)	(22221)	0.8463
5	8	9	(21113)	(22221)	0.8716
5	9	9	(21114)	(22221)	0.889
5	6	10	(21111)	(22222)	0.9262
5	7	10	(31111)	(22222)	0.9611
5	8	10	(31112)	(22222)	0.9916
5	9	10	(31122)	(22222)	1.024

Table 13: Optimum buffer and server allocation (5 stations) using CE.

Q	q	R	T
11	111111122	0.3735	311
12	111111222	0.3876	299
13	111112222	0.4024	282
14	111122222	0.4179	291
15	111222222	0.4337	287
16	112222222	0.4496	491
17	122222222	0.4658	135
18	222222222	0.4761	338
19	222222223	0.4861	429
20	222222233	0.4964	374
21	222222333	0.5071	342

Table 14: Optimum buffer allocation using SA. Nine stations, 11–21 buffers, nine servers, nine service rate units.

Q	q	R	T
11	111111122	0.3735	1
12	111111222	0.3876	2
13	111112222	0.4024	6
14	111122222	0.4179	15
15	111222222	0.4337	37
16	112222222	0.4496	78
17	122222222	0.4658	155
18	222222222	0.4761	294

Table 15: Optimum buffer allocation using CE. Nine stations, 11–18 buffers, nine servers, nine service rate units.

Q	q	R	T
16	111111111111112	0.281	843
17	111111111111122	0.288	817
18	111111111111222	0.2954	880
19	111111111112222	0.3031	849
20	111111111122222	0.3113	870
21	111111111222222	0.3198	836
22	111111112222222	0.3287	705
23	111111122222222	0.338	748
24	111111222222222	0.3475	763
25	111112222222222	0.3572	862
26	111122222222222	0.3669	819
27	111222222222222	0.3765	740
28	112222222222222	0.3858	1326
29	122222222222222	0.3947	425
30	122222222222223	0.4003	1290
31	222222222222223	0.406	1442
32	222222222222233	0.412	1196
33	222222222222333	0.4182	1155
34	222222222223333	0.4246	1059
35	222222222233333	0.4311	1000
36	222222222333333	0.4378	1009
37	222222223333333	0.4446	881
38	222222233333333	0.4514	869
39	222223333333333	0.4582	815
40	222233333333333	0.465	785
41	222333333333333	0.4716	830
42	223333333333333	0.4779	852
43	233333333333333	0.4837	748
44	233333333333333	0.489	803
45	233333333333334	0.4937	1051

Table 16: Optimum buffer allocation using SA. 15 stations, 16–45 buffers, 15 servers, 15 service rate units.

Q	q	R	T
16	111111111111112	0.281	0
17	111111111111122	0.288	2
18	111111111111222	0.2954	12
19	111111111112222	0.3031	54
20	111111111122222	0.3113	208

Table 17: Optimum buffer allocation using CE. 15 stations, 16–20 buffers, 15 servers, 15 service rate units.

N	q	R	T
4	3122	1.093	55
5	31222	1.057	100
6	312222	1.025	132
7	3122222	0.9976	170
8	31222222	0.9728	235
10	3122222222	0.93	347
20	31222222222222222222	0.7932	1315
30	312222222222222222222222222222	0.7152	2313
40	312222222222222222222222222222222222	0.6626	4174
50	3122	0.6237	6495
60	31222	0.5933	10075

Table 18: Optimum buffer allocation using SA. $N = 4 - 60$ stations, $2 \times N$ buffers and servers, N service rate units.

[illegible]

Table 24: Optimum buffer (\underline{q}) and server (\underline{s}) allocation using SA. $N = 4 - 60$ stations, $2 \times N$ buffers and servers, N service rate units.

N	$\underline{q}, \underline{s}$	R	T
4	$\underline{q}: 3\ 1\ 2\ 2$ $\underline{s}: 2\ 2\ 2\ 2$	1.093	25
5	$\underline{q}: 3\ 1\ 2\ 2\ 2$ $\underline{s}: 2\ 2\ 2\ 2\ 2$	1.057	828

Table 25: Optimum buffer (\underline{q}) and server (\underline{s}) allocation using CE. $N = 4 - 5$ stations, $2 \times N$ buffers and servers, N service rate units.

[illegible]

Table 26: Optimum server (s) and service rate (w) allocation using SA. $N = 4 - 60$ stations, $2 \times N$ buffers and servers, N service rate units.

Table 27: Optimum buffer (\underline{q}), server (\underline{s}) and service rate (\underline{w}) allocation using SA. $N = 4 - 60$ stations, $2 \times N$ buffers and servers, N service rate units.

References

- Altioik, T., 1982, Approximate analysis of exponential tandem queue with blocking. *European journal of operations research*, **11**, 390–398.
- Altioik, T., 1997, *Performance analysis of manufacturing systems*. (New York: Springer-Verlag).
- Altioik, T., and Perros, H. G., 1986, Open networks of queues with blocking: Split and merge configurations. *IIE transactions*, 251–261.
- Altioik, T., and Stidham, S. Jr., 1983, The allocation of inter-stage buffer capacities in production lines. *IIE transactions*, **15**(4), 292–299.
- Askin, R. G., and Standridge, C. R., 1993, *Modeling and analysis of manufacturing systems*. (New York: John Wiley).
- Boxma, O., and Konheim, A., 1981, Approximate analysis of exponential queueing systems with blocking. *Acta informatica*, **15**, 19–26.
- Brandwajn, A., and Jow, Y., 1988, An approximation method for tandem queues with blocking. *Operations research*, **36**(1), 73–83.
- Buzacott, J. A., and Shanthikumar, J. G., 1992, Design of manufacturing systems using queueing models. *Queueing systems*, **12**, 135–214.
- Buzacott, J. A., and Shanthikumar, J. G., 1993, *Stochastic models of manufacturing systems*. (New Jersey: Prentice Hall).
- Caseau, P., and Pujolle, G., 1979, Throughput capacity of a sequence of queues with blocking due to finite waiting room. *IEEE transactions on software engineering*, **5**, 631–642.
- Cerny, V., 1985, Thermodynamical approach to the traveling salesman problem: an efficient simulation algorithm. *Journal of optimization theory and applications*, **45**, 41–51.
- Cheah, Jenyeng, and Smith, J. MacGregor., 1994, Generalized M/G/C/C state dependent queueing models and pedestrian traffic flows. *Queueing systems and their applications (QUESTA)*, **15**, 365–386.
- Choong, Y. F., and Gershwin, S. B., 1987, A decomposition method for the appropriate evaluation of capacitated transfer lines with unreliable machines and random processing times. *IIE transactions*, **19**(2), 150–159.
- Conway, R., Maxwell, W., McClain, J. O., and Thomas, L. J., 1988, The role of work in process inventory in serial production lines. *Operations research*, **36**(2), 229–241.
- Corana, A., Marchesi, M., Martini, C., and Ridella, S., 1987, Minimizing multimodal functions of continuous variables with the “simulated annealing” algorithm. *ACM transactions on mathematical software*, **13**(3), 262–280.
- Dallery, Y., and Frein, Y., 1993, On decomposition methods for tandem queueing networks with blocking. *Operations research*, **41**(2), 386–399.
- Daskalaki, S., and Smith, J. MacGregor., 1986, *Static routing in open finite queueing networks*. Working paper.
- Daskalaki, S., and Smith, J. MacGregor., 1989, *Optimal routing and buffer space allocation in series-parallel queueing networks*. Working paper.
- Ding, J., and Greenberg, B. S., 1991, Bowl shapes are better with buffer – sometimes. *Prob. eng. inf. sci.*, **5**, 159–169.
- Eglese, R. W., 1990, Simulated annealing: A tool for operational research. *European journal of operational research*, **46**, 271–281.
- Fredericks, A. A., 1980, Congestion in blocking systems – a simple approximation technique. *The Bell system technical journal*, **59**(6), 805–827.
- Fredericks, A. A., and Reisner, G. A., 1979, Approximations to stochastic service systems with an application to a retrial model. *The Bell system technical journal*, **58**(3), 557–576.
- Gelenbe, E., and Pujolle, G., 1976, The behavior of a single queue in a general queueing network. *Acta informatica*, **7**, 123–136.
- Gershwin, S. B., 1989, An efficient decomposition algorithm for unreliable tandem queueing systems with finite buffers. In: Perros, H. G., and Altioik, T. (eds), *Queueing networks with blocking*. North Holland.
- Gershwin, S. B., 1994, *Manufacturing systems engineering*. (New Jersey: Prentice Hall).
- Glassey, C. R., and Hong, Y., 1983, Analysis of behaviour of an unreliable n -stage transfer line with $(n-1)$ inter-stage storage buffers. *International journal of production research*, **31**(3), 519–530.
- Glover, F., 1990, Tabu search — part I. *ORSA journal on computing*, **1**, 190–206.

- Goldberg, David E., 1989, *Genetic algorithms: In search of optimization & machine learning*. (Addison-Wesley).
- Gosavi, Hemant, and Smith, J. MacGregor., 1990, *Heavy traffic multi-commodity routing in open finite queueing networks*. Working paper.
- Gun, L., and Makowski, A. M., 1989, An approximation method for general tandem queueing systems subject to blocking. *Pages 147–171 of*: Perros, Harry G., and Altioik, Tayfur (eds), *Queueing networks with blocking: First international workshop*. Raleigh, North Carolina, USA: North-Holland.
- Gupta, R., Smolka, S. A., and Bhaskar, S., 1994, On randomization in sequential and distributed algorithms. *ACM computing surveys*, **26**(1), 7–86.
- Han, Y., and Smith, J. MacGregor., 1992, Approximate analysis of open $M/M/C/K$ queueing networks. *Pages 113–126 of*: Onvural, Raif O., and Akyildiz, Ian F. (eds), *Queueing networks with finite capacity: Second international conference on queueing networks with finite capacity*. North-Holland.
- Heavey, C., Papadopoulos, H. T., and Browne, J., 1993, The throughput rate of multistation unreliable production lines. *European journal of operational research*, **68**, 69–89.
- Hillier, F. S., and Boling, R. W., 1966, The effect of some design factors on the efficiency of production lines with variable operation times. *J. ind. eng.*, **17**, 651–658.
- Hillier, F. S., and Boling, R. W., 1967, Finite queues in series, with exponential or erlang service times – a numerical approach. *Operations research*, **15**, 286–303.
- Hillier, F. S., and Boling, R. W., 1977, Toward characterizing the optimal allocation of work in production line systems with variable operation times. *In*: Roubens, M. (ed), *Advances in operations research, proc. Euro II*. Amsterdam: North-Holland.
- Hillier, F. S., and Boling, R. W., 1979, On the optimal allocation of work in symmetrically unbalanced production line systems with variable operation times. *Management science*, **25**, 721–728.
- Hillier, F. S., and So, K. C., 1989, The assignment of extra servers to stations in tandem queueing systems with small or no buffers. *Performance evaluation*, **10**, 219–231.
- Hillier, F. S., and So, K. C., 1991a, The effect of machine breakdowns and interstage storage on the performance of production line systems. *International journal of production research*, **29**(10), 2043–2055.
- Hillier, F. S., and So, K. C., 1991b, The effect of the coefficient of variation of operation times on the allocation of storage space in production line systems. *IIE transactions*, **23**(2), 198–206.
- Hillier, Frederick S., and So, Kut C., 1995, On the optimal design of tandem queueing systems with finite buffers. *Queueing systems*, **21**, 245–266.
- Himmelblau, D. M., 1972, *Applied nonlinear programming*. (Mc-Graw-Hill).
- Ho, Y. C., Eyler, M. A., and Chien, T. T., 1979, A gradient technique for general buffer storage design in a production line. *International journal of production research*, **17**(2), 557–580.
- Holland, J. H., 1975, *Adaptation in natural and artificial systems*. (Ann Arbor, Michigan: University of Michigan Press).
- Huang, C. C., and Weiss, G., 1990, On the optimal order of m machines in tandem. *Operations research letters*, **9**, 299–303.
- Ingber, L., 1993, Simulated annealing: Practice versus theory. *Journal of mathematical computation modelling*, **18**(11), 29–57.
- Jafari, M. A., and Shanthikumar, J. G., 1989, Determination of optimal buffer storage capacities and optimal allocation in multistage automatic transfer lines. *IIE transactions*, **21**(2), 130–135.
- Jain, Sushant, and Smith, J. MacGregor., 1994, Open finite queueing networks with $M/M/C/K$ parallel servers. *Computers and operations research*, **21**(3), 297–317.
- Kerbache, L., 1984, *Analysis of open finite queueing networks*. Ph.D. thesis, Department of Industrial Engineering and Operations Research, University of Massachusetts, Amherst, MA 01003, USA.
- Kerbache, L., and Smith, J. MacGregor., 1987, The generalized expansion method for open finite queueing networks. *European journal of operational research*, **32**, 448–461.
- Kerbache, L., and Smith, J. MacGregor., 1988, Asymptotic behaviour of the expansion method for open finite queueing networks. *Computers and operations research*, **15**(2), 157–169.
- Kirkpatrick, S., Jr., C. D. Gelatt, and Vecchi, M. P., 1983, Optimization by simulated annealing. *Science*, **220**, 671–679.
- Kleinrock, Leonard., 1975, *Queueing systems*. Vol. I: Theory. (John Wiley and Sons).

- Koulamas, C., Antony, S. R., and Jaen, R., 1994, A survey of simulated annealing applications to operations research problems. *Omega international journal of management science*, **22**(1), 41–56.
- Kubat, P., and Sumita, U., 1985, Buffers and backup machines in automatic transfer lines. *International journal of production research*, **23**(6), 1259–1270.
- Kuehn, P. J., 1979, Approximate analysis of general queueing networks by decomposition. *IEEE trans. on COMM*, **27**, 113–126.
- Laarhoven, P. J. M. Van, and Aarts, E. H. L., 1987, *Simulated annealing: Theory and applications*. (Dordrecht, The Netherlands: D. Reidel).
- Labetoulle, J., and Pujolle, G., 1980, Isolation method in a network of queues. *IEEE transactions on software engineering*, **6**(4), 373–380.
- Magazine, M. J., and Stecke, K. E., 1996, Throughput for production lines with serial work stations and parallel service facilities. *Performance evaluation*, **25**, 211–232.
- Metropolis, N., Rosenbluth, A. N., Rosenbluth, M. N., Teller, A. H., and Teller, E., 1953, Equation of state calculation by fast computing machines. *Journal of chemical physics*, **21**(6), 1087–1092.
- Onvural, Raif., 1990, Survey of closed queueing networks with blocking. *ACM computing surveys*, **22**(2), 83–121.
- Papadopoulos, H. T., and Heavey, C., 1996, Queueing theory in manufacturing systems analysis and design: A classification of models for production and transfer lines. *European journal of operational research*, **92**, 1–27.
- Papadopoulos, H. T., Heavey, C., and Browne, J., 1993, *Queueing theory in manufacturing systems analysis and design*. (London: Chapman and Hall).
- Perros, H. G., and Altiok, T., 1986, Approximate analysis of open networks of queues with blocking: Tandem configurations. *IEEE transactions on software engineering*, **12**(3), 450–461.
- Powell, S. G., 1994, Buffer allocation in unbalanced three-station serial lines. *International journal of production research*, **32**(9), 2201–2217.
- Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling, W. T., 1988, *Numerical recipes in C*. (Cambridge University Press). Pages 343–352.
- Schweitzer, P., and Altiok, T., 1989, Aggregate modelling of tandem queues without intermediate buffers. *Pages 33–46 of: Perros, Harry G., and Altiok, Tayfur (eds), Queueing networks with blocking: First international workshop*. Raleigh, North Carolina, USA: North-Holland.
- Seong, D., Chang, S. Y., and Hong, Y., 1995, Heuristic algorithms for buffer allocation in a production line with unreliable machines. *International journal of production research*, **33**(7), 1989–2005.
- Shanthikumar, J. G., Yamazaki, G., and Sakasegawa, H., 1991, Characterization of optimal order of servers in tandem queue with blocking. *Operations research letters*, **10**, 17–22.
- Singh, A., and Smith, J. MacGregor., 1997, Buffer allocation for an integer nonlinear network design problem. *Computers & operations research*, **24**(5), 453–472.
- Smith, J. MacGregor., 1991, State dependent queueing models in emergency evacuation networks. *Transportation res-b*, **25B**(6), 373–389.
- Smith, J. MacGregor., 1994, Applications of state dependent queues to pedestrian/vehicular network design. *Operations research*, **42**(3), 414–427.
- Smith, J. MacGregor, and Chikhale, Nikhil., 1995, Buffer allocation for a class of nonlinear stochastic knapsack problems. *Annals of operations research*, **58**, 323–360.
- Smith, J. MacGregor, and Daskalaki, S., 1988, Buffer space allocation in automated assembly lines. *Operations research*, **36**(2), 343–358.
- Spinellis, Diomidis, and Papadopoulos, Chrisoleon T., 1997 (May), A simulated annealing approach for buffer allocation in reliable production lines. *Pages 365–375 of: International workshop on performance evaluation and optimization of production lines*. University of the Aegean, Department of Mathematics, Samos, Greece.
- Takahashi, Y., 1989, Aggregate approximation for acyclic queueing networks with communication blocking. *Pages 33–46 of: Perros, Harry G., and Altiok, Tayfur (eds), Queueing networks with blocking: First international workshop*. Raleigh, North Carolina, USA: North-Holland.
- Takahashi, Y., Miyahara, H., and Hasegawa, T., 1980, An approximation method for open restricted queueing networks. *Operations research*, **28**(3), 594–602. Part I.
- VanDijk, N., and Lamond, B., 1988, Simple bounds for finite single server exponential tandem queues. *Operations research*, **36**(3), 470–477.

- VanDijk, N., and van der Wal, J., 1989, Simple bounds and monotonicity results for finite multi-server exponential tandem queues. *Queueing systems*, **4**(1), 1–16.
- Viswanadham, N., and Narahari, Y., 1992, *Performance modeling of automated manufacturing systems*. (New Jersey: Prentice Hall).
- w. Wan, Y., and Wolff, R. W., 1993, Bounds for different arrangements of tandem queues with nonoverlapping service times. *Management science*, **39**, 1173–1178.
- Yamashita, H., and Altiok, T., 1997 (May), Buffer capacity allocation for a desired throughput in production lines. *Pages 1–24 of: Samos international workshop on performance evaluation and optimization of production lines*. University of the Aegean, Samos, Greece.
- Yamazaki, G., Sakasegawa, H., and Shanthikumar, J. G., 1992, On optimal arrangement of stations in a tandem queueing system with blocking. *Management science*, **38**, 137–153.