European Journal of Industrial Engineering

Volume 1, No. 4, 2007

Publisher's website: www.inderscience.com

E-mail: ejie@inderscience.com

ISSN (Print) 1751-5254
ISSN (Online) 1751-5262

Published and typeset in the UK by Inderscience Enterprises Ltd

080207

The *European Journal of Industrial Engineering* (*EJIE*) is an international quarterly journal aimed at disseminating the latest developments in all areas of industrial engineering, including information and service industries, ergonomics and safety, quality management as well as business and strategy, and at bridging the gap between theory and practice.

The main goal of EJIE is to present state of-the-art, high quality, research developments in all areas of industrial engineering, including applications in industry and services, to a broad audience of academics and professionals. Target areas include manufacturing, operations management, product and process design, information systems, business and strategy, and quality management amongst others.

In order to bridge the gap between theory and practice, applications and case studies are particularly welcome. As regards theoretical papers, originality and research contributions will be regarded as key during the evaluation process. The Editorial Board is committed to a quick and swift process of less than 10 weeks (from submission to initial decision) ensuring not only that authors know the outcome of the evaluation process as soon as possible but also that the latest developments and advances are made available without delay to professionals, academics, researchers, and practitioners.

Subject coverage

EJIE covers a broad range of topics including, but not limited to, the following:

- Business and strategy
- Case studies in industry and services
- Decision analysis
- Engineering economy and cost estimation
- Environmental issues
- Facility location, layout, design and materials handling
- Human factors, ergonomics and safety
- Industrial Engineering Education
- Information and communication technology and systems
- Innovation, knowledge management and organizational learning
- Inventory, logistics and transportation
- Manufacturing, control and automation
- Operations management
- Performance analysis
- Product and process design and management
- Forecasting, production planning and control
- Project management
- Reliability and maintenance engineering
- Service systems and service management
- Scheduling in industry and service
- Systems and service modelling and simulation
- Supply chain management
- Total quality management and quality engineering

Submission of papers

Papers in the areas covered by *EJIE* are invited for submission. Notes for intending authors can be found at: https://www.inderscience.com/papers Authors of accepted papers will receive a PDF file of their published paper.
Authors are invited to submit their papers to one of the following editors:

Ali Allahverdi
Kuwait University
Department of Industrial and Management Systems Engineering, College of Engineering and Petroleum, PO Box. 5969, 13060 Safat, Kuwait
Email: allahverdi@kuc01.kuniv.edu.kw
or
Rubén Ruiz
Polytechnic University of Valencia
Department of Applied Statistics, Operations Research and Quality, Camino de Vera, s/n
46022 Valencia, Spain
Email: rruiz@eio.upv.es
or
Jose M. Framinan
University of Seville
School of Engineering
Camino de los Descubrimientos s/n
41092 Sevilla, Spain
Email: jose@esi.us.es

A copy of the submitted paper and submission letter should also be sent via email to the IEL Editorial Office at:
E-mail: ejie@inderscience.com
Fax: (UK) +44 1234-240515
Website: www.inderscience.com
Neither the editors nor the publisher can accept responsibility for opinions expressed in the *European Journal of Industrial Engineering* nor in any of its special publications.

Subscription orders

EJIE is published in four issues per volume.
A Subscription Order Form is provided in this issue.
Payment with order should be made to:
Inderscience Enterprises Ltd. (Order Dept.),
World Trade Center Building,
29 Route de Pre-Bois, Case Postale 896,
CH-1215 Genève 15, Switzerland.
You may also FAX to:
(UK) +44 1234 240 515
or Email to subs@inderscience.com

Electronic PDF files

EJIE papers are available to download from website: www.inderscience.com
Online payment by credit card.

Advertisements

Please address enquiries to the above-mentioned Geneva address or
Email: adverts@inderscience.com

Contents

EJIE SUBSCRIPTION ORDER FORM

Volume 1, 2007

(THIS FORM MAY BE PHOTOCOPIED)

Subscription price and ordering information:

The ***European Journal of Industrial Engineering*** is published four times a year (in one volume of four issues), in English.

Subscription for hard copy OR online format (one simultaneous user only) **€ 450** per annum (including postage and handling).

Subscription for hard copy AND online format (one simultaneous user only) **€ 610**

Airmail option € 40 per volume extra.

Prices for multi-simultaneous users are available on request.

Subscription orders should be addressed to the publishers:

Inderscience Enterprises Ltd (Order Dept.), World Centre Building, 29 route de Pre-Bois, Case Postale 896, CH-1215 Genève 15, Switzerland.

- ***Payment with order:***
  Cheques or bankers drafts should be sent with order, made payable to:
  Inderscience Enterprises Ltd.
  Credit card payments will be accepted and will be converted to £ Sterling at the prevailing rates.
  For rush orders, contact:
  Fax: (UK) +44 1234 240 515
  Website: www.inderscience.com or Email to subs@inderscience.com

---

- ***Please enter my subscription to the European Journal of Industrial Engineering***
  ☐ subscriptions to Volume 1, 2007 **€**.....
- ***Please dispatch my order by air mail (add € 40 per Volume): €***.....
- ***I enclose total payment of***..... **€**
- ***Name of Subscriber***.....
- ***Position***.....
- ***Company/Institution***.....
- ***Address***.....
  .....
  .....
- ***Fax*** .....***E-mail*** .....
- ***Date***.....***Signature*** .....

I wish to pay by credit card……..…….……..

- ***I authorise you to debit my account with the amount in GBP sterling equivalent to € .....……***
- ***Three digit security number (on reverse of card)……***
- ***Card No.*** .....***Expiry Date***.....
  Signature.....***Date***.....

Please tick if you would like details of other Inderscience publications ☐

European Journal of Industrial Engineering (EJIE)

Analysing inaccurate judgemental sales forecasts

Annastiina Kerkkänen* and Janne Huiskonen

Department of Industrial Engineering and Management,
Lappeenranta University of Technology,
P.O. Box 20, Lappeenranta 53851, Finland
Fax: +358-5-621-2699
E-mail: annastiina.kerkkanen@lut.fi
E-mail: Janne.huiskonen@lut.fi
*Corresponding author

Abstract: This paper deals with categorising errors that exist in qualitative sales forecasts, so that it can be defined what kind of development is needed to improve forecast accuracy. A framework for pointing out different types of sales forecast errors is presented. The framework includes analysing demand profiles of customers and the continuity of under-/over-forecast errors. The error types are named as random error, positive bidirectional error, negative bidirectional error, systematic under/over estimation error and unforecasted sales. The differences between the approaches for reducing each type of error are explained. The use of the framework is illustrated with sales and forecast data of a large process industry company. The analysis steps are illustrated and actions for reducing different types of sales forecast errors are suggested.
[Received 15 November 2006; Accepted 11 May 2007]

Keywords: demand forecasting; supply chain management; decision making; forecast errors; judgemental forecasting; uncertainty; forecasting systems; biases; forecasting management; case study.

Reference to this paper should be made as follows: Kerkkänen, A. and Huiskonen, J. (2007) 'Analysing inaccurate judgemental sales forecasts', *European J. Industrial Engineering*, Vol. 1, No. 4, pp.355–369.

Biographical notes: Annastiina Kerkkänen received a Masters degree at Lappeenranta University of Technology in 2002. Currently, she is working for her doctoral dissertation. Her current research theme is forecasting and inventory management.

Janne Huiskonen received a Master and PhD in Lappeenranta University of Technology, Finland. Currently, he works as Professor of Supply Chain and Operations Management at the Lappeenranta University of Technology, Finland.

1 Introduction

Forecasting means estimating a future event or condition which is outside an organisation's control and that which provides a basis for managerial planning. Forecasting techniques range from simple to complex, and include the use of executive

judgement, surveys, time-series analysis, correlation methods and market tests. Many companies do not know their future demands and have to rely on demand forecasts to make production planning decisions.

There is evidence that in the industrial markets, companies rely commonly on judgemental demand forecasting (Mentzer and Moon, 2005). The contribution of human judgement in forecasting has also started to gain wider academic acceptance since the 1980s (Lawrence et al., 2006; Webby and O'Connor, 1996). Despite the great efforts that have been put on forecasting research, many companies still face a situation where forecasts do not meet the targets that have been set.

In industrial markets, demand patterns are often such that it is difficult to forecast demand based only on history and applying technical methods to data. A larger part of demand errors are due to judgemental part of forecasting and therefore due to behavioural elements of forecasting.

In former literature, the importance of information flow has been emphasised as a remedy for the whole supply chain management (Chen, 1998; Lin et al., 2002). The lack of information technology is nowadays hardly the reason for poor forecasting performance. In particular, new information technology has made real-time, online communications among parties within the supply chain possible (Hanfield and Nicholas, 1999). However, sharing information more efficiently does not guarantee efficiency in the supply chain management if the information that is shared is not valid. When the forecasts are produced by sales people, which is common in the industrial markets, it is difficult to control the quality of the data that enters the forecasting system. The weakness of the forecasting process is then rather managerial than technical or mathematical in nature. Managing such a complex, cross-functional process as demand forecasting requires tools for illustrating and communicating the problems of sales forecasting in the organisation.

It is important to identify that there are different motives behind forecasting in the organisation that cause errors in the forecasts. If the behavioural elements are substantial in forecasting, it is reasonable to focus development into managerial, not only technical aspects of forecasting process. It is impossible to detect the deepest underlying causes to forecast errors from data, but it is possible to identify regularities to which development acts can be directed to. This paper presents a framework for analysing sales and forecast data produced by the sales people. The idea is to categorise forecast errors so that it is possible to point out distinct policies for reducing forecast errors in separate categories. The approach presented in this paper is only a first, but still a very important step in order to find a way into better sales forecasting management.

In Section 1, we consider the literature on judgemental forecasting and explanations and remedies for poor forecasting performance. In Section 2, we explain the use of the suggested research method. Section 3 presents the framework for categorising forecast errors. In Section 4, we illustrate the use of the framework in the case company. Section 5 discusses the managerial implications in the case company and Section 6 offers some concluding comments.

2 Literature review

In former literature, there are two main approaches to forecasting; quantitative techniques meaning that forecasts are produced based on historical data only, and qualitative techniques meaning that forecasts aim at anticipating future demand. In this paper,

the focus is on the latter approach, and to be more specific, on forecasts that are the responsibility of sales people. This, forecasting method is usually called 'salesforce composite method' or 'salesforce forecasting'.

Salesforce forecasting is seen as a possibility to estimate future demand, as estimating intermittent/lumpy demand based on historical data has been noticed to be difficult. Anyhow, it has been known for long that the forecasts that are on the responsibility of sales force tend to bias (Lines, 1996; Moon and Mentzer, 1999). In this section, theories for the poor forecasting accuracy are presented and the suggested approaches for overcoming the weaknesses are reviewed.

2.1 Causes and approaches for overcoming poor forecasting performance

The poor performance of the forecasts produced by sales force is usually explained by the conflicts between the roles of forecaster and a salesman. The reasons for poor forecasting performance can be divided into three categories:

1 game playing
2 low motivation
3 lack of ability.

Game playing means that the salesman uses forecasting to serve his own purposes. The forecasts reflect the salesman's optimism about the future sales, as he seeks to guarantee the availability of the products to the customers. *Low motivation* means that the salesman does not see any point in forecasting, as he does not benefit from forecasting accurately. *Lack of ability* means that the salesman lacks tools and/or abilities to produce reliable forecasts. This includes also lack of information from the customers.

Lines (1996) emphasises the game playing reason:

> "Because a saleman's *raison d'être* is to improve the level of sales over what has been seen in the past, however, if entrusted with forecasting he may be tempted to alter the value of any forecast produced by extrapolation techniques to reflect his optimism."

Lines stresses that proper control is needed in order to reduce the forecast error, though he admits that it is still difficult to get the timing right although the general message might be correct.

Moon and Mentzer (1999) emphasise the lack of motivation most. In an in-depth study of the sales-forecasting management practices of 33 companies, they found some resistance from the salespeople concerning their forecasting responsibilities in almost all the studied companies. Many salespersons felt that it was not their job to forecast and the time spent on forecasting was time taken away from their real job of managing customer relationships and selling products and services.

The suggestions for overcoming the problem by Mentzer and Moon (1999) aim at facilitation of forecasting. They suggest the following:

1 make forecasting part of the sales people's job by including forecasting as a part of their job descriptions, by creating incentives for high performance in forecasting, and by providing feedback and training
2 minimise game playing by making forecasting accuracy an important outcome for the sales people and by clearly separating sales quotas from forecasts

3 keep it simple by asking the sales people only to adjust statistically generated forecasts rather than produce forecasts from scratch and by providing them with adequate tools that enable them to complete their forecasting work as efficiently as possible

4 keep it focused by having the sales people deal only with the products and customers that are truly important and where their input can significantly affect the company's supply chain.

Some approaches that remind the suggestions of Mentzer and Moon have been developed. For example, Holmström (1998) has presented an approach called 'assortment forecasting'. The approach focuses on reducing the time spent on forecasting by working with an entire assortment at a time instead of producing a forecast for each product individually. The approach has been tested by Småros and Hellström (2004) with a case company that provides supermarkets, video rentals and the like with pick-and-mix sweets.

Mentzer and Kahn (1997) describe a problem they call 'islands of analysis', which is a consequence of game playing:

> "Islands of analysis are systems phenomena where one individual or group develops a sales forecast based upon their own information and needs, and does not share that information of forecast with others in the company. The resultant sales forecasts may be significantly different than forecasts developed elsewhere in the company (other islands) and these differences lead to conflicting plans."

Helms et al. (2000) emphasise the poor management of forecasting. They claim that forecasting is often the most maligned department in any company. They claim that the 'islands of analysis' problem could be tackled with better management of forecasting. The approach they suggest is titled as Collaborative forecasting, and it emphasises the cooperation between sales units and production units, and creating a consensus forecast. According to Helms et al. the process should include analysis of the actual sales versus the forecast and the creation of a baseline forecast based on historical information.

Some approaches even expand the collaboration across trading partners. Since the mid-1990s, academics have emphasised the importance of creating a seamless supply chain, using concepts like Vendor Managed Inventory (VMI), Collaborative forecasting and Replenishment (CPFR) and Continuous Replenishment (CR). Yet, mainstream implementation of these concepts has been less prominent as expected, despite the benefits that have been claimed (Holweg et al., 2005).

As a contrast to Collaborative forecasting, there is an anecdote reported in the study of Helms et al. (2000): "In the past, marketing folks would put together a forecast, but production personnel would put together what they considered to be a more accurate forecast". The abovementioned anecdote calls to mind another approach for improving forecast accuracy, splitting the forecasting responsibility between sales units and production unit. This approach has been presented by Zotteri and Verganti (2001).

Zotteri and Verganti see the exaggerated sales forecasts provided by sales force as an approach to manage demand uncertainty, and call this approach 'order overplanning'. This method was originally introduced in the study of Bartezzaghi and Verganti (1995). This is just another example of game playing.

In their study, Zotteri and Verganti (2001) examine whether the manufacturing or the sales units should manage demand uncertainty. The first option is a decentralised approach, where slack is incorporated into the overstated forecasts provided by the sales

units. The second option is a centralised approach, where slack is set by the manufacturing unit on the basis of information gathered by the sales units. In the centralised approach, each sales unit must specify not only the production level they believe to be optimal, but also the probability associated with each single potential customer order. In this approach, the management of demand uncertainty is split between the manufacturing (where the slack is defined) and sales (where the forecasts are made).

Even though many authors emphasise organisational issues (game playing and low motivation) when it comes to salesforce forecasting, it must be kept in mind that forecasts produced by sales force may error from same reasons as the forecasts in general; internal factors like unsuitable time horizon and external factors like lumpiness of the demand affect the forecasting accuracy.

It is very likely that different causes for forecast errors coexist, and therefore it is important to analyse how the errors mainly build up, before implementing a solution to reduce forecast errors. In the next section, an approach is presented, which helps pointing out the magnitude of different types of forecast errors so that the suitability of different kinds of corrective actions can be estimated.

3 Methodological viewpoints

Especially in the case of qualitative forecasting, managing the forecasting process is a complex issue, including for example target setting, forming basic assumptions about the nature of the demand and leading the practical work of the forecasters. The practical problem the case company is facing is how to operate in a situation where qualitative forecasts do not meet the targets that have been set. At the same time as forecast errors are examined, also the targets must be kept under critical assessment; are they relevant and realistic? Therefore, improving the performance of the forecasting process is a multidimensional managerial problem, and solving it requires broader methodological view than attempting to create a generic solution algorithm.

In these kind of situations, another type of approach is often suggested, in which the decision-maker is offered supporting tools to recognise the type of the problem and some potentially effective actions (e.g. heuristic rules). The decision-maker is left with the task of integrating the context-dependent information and also all the intuitive (tacit) knowledge which he may possess and find relevant in the situation. This kind of approach belongs to the paradigm of design sciences (e.g. Van Aken, 2004), of which purpose is to provide knowledge to support the design of interventions that managers need to do in various decision-making situations. It is based on the claim that in complex situations only the design knowledge (on the potential types of actions) can be general (i.e. valid for classes of cases), but the problem of the manager is always unique and specific (Van Aken, 2004).

4 A framework for analysing demand forecasts

The framework presented in this paper is designed for a situation where forecasts are developed for each customer separately, but aggregated from several sales units into a higher level, for example, product family level. This is a common situation in the industrial markets. Though in the industrial markets, the number of customers is lower

than in the consumer markets, it is still time consuming to go through the buying behaviour of each single customer, and in that way, trying to find out the deepest causes of forecast inaccuracy, and finding out possible remedies for them. Therefore, a framework is needed that points out where the greatest improvement potential lies, so that development actions can be rationally focused. Demand and forecast data is easily available, so it is economical to begin the development work by analysing that data and thereby identifying and evaluating the further steps of development work. This kind of analysis tool is not meant for regular use, but rather to be performed once a year to illustrate the performance of the forecasting process.

The idea of the analysis framework is to sort out different types of forecast errors, because different types of errors call for different approaches to improve the forecast accuracy. The categorisation of errors that we suggest is presented in Table 1. Firstly, under-forecasts (demand exceeds the forecast) and over-forecasts (forecast exceeds the demand) must be separated from each other. Examining under-forecasts and over-forecasts separately enables deeper analysis of forecast bias. For example, then it is possible to detect, how bias (negative or positive) relates to individual forecasters.

Table 1 Definitions of the error types

Error type	*Definition*
Unforecasted sales	The sales of the customer are not forecasted in the analysis period
Systematic over-estimation	The sales of the customer are always over-forecasted in the analysis period
Systematic under-estimation	The sales of the customer are always under-forecasted in the analysis period
Positive bidirectional error	The sales of the customer are both under-forecasted and over-forecasted, but the net error in the analysis period is over-forecasting
Negative bidirectional error	The sales of the customer are both under-forecasted and over-forecasted, but the net error in the analysis period is under-forecasting
Random error	The sales of the customer are both under-forecasted and over-forecasted, but the under- and over forecast errors cancel each other in the analysis period

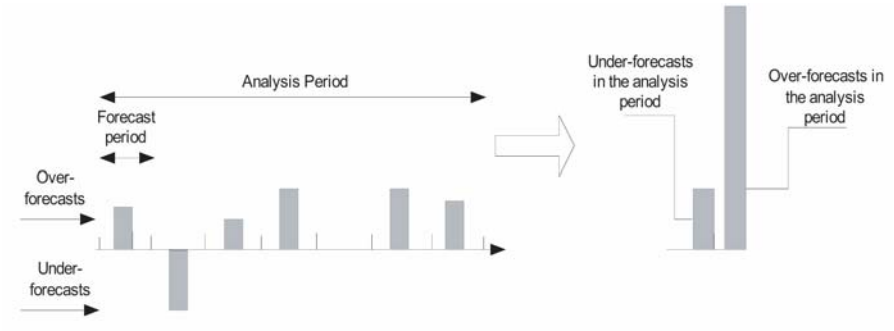
Secondly, the regularity of over/under-forecasting must be analysed. Systematic over-forecasting errors are assumed to be easier to reduce than the irregular errors, because forecasts could be trimmed down/up to some extent without increasing forecast errors on any period. If bias is not systematic, it indicates that there is an exceptional period in demand, for example, the so-called demand pike. In these cases, reducing the forecast errors requires finding out the reason for unpredicted exception in the demand, in addition with the reduction of bias.

In addition, it is distinguished which part of the under-forecasting errors occur because the forecasts are not made, and which part of the forecast errors can be considered random.

Figure 1 clarifies the way the forecast errors in one customer's demand during analysis period are summed. For example, if forecasts are produced on a monthly basis, the 'forecasting period' is one month. The forecast errors are counted up from a chosen

analysis period that is a multiple of the forecasting period (seven months in the example). Counting up the negative and positive errors separately gives a picture if the forecast has been biased in the long run.

Figure 1 Forecast errors on the forecast horizon and in the analysis period (example of a positive bidirectional error)



It is presumable that if the demand is continuous, there are better premises to forecast more accurately than if the demand is intermittent. Also, the effectiveness of the actions aiming at improving the forecast accuracy depends on the continuity of demand. If the forecast period is extended, for example from one month to two months, the forecast accuracy improves in all demand categories, but in the category of intermittent demand, there is relatively more potential to accuracy improvement. That is why forecast errors should be examined separately in different demand profile categories. Hence, demand profiles are categorised roughly into 0-demand (no demand), intermittent demand and continuous demand.

In summary, we suggest that the categorisation follows the three-step analysis framework described below:

Step 1 Set the parameters that are needed for categorising the forecast errors. These parameters include the length of proper analysis period and the bounds of error categories.

Step 2 Divide forecast errors into over- and under-forecasts and classify the data according to demand profile.

Step 3 Divide forecast errors into error types that were introduced in Table 1.

Full categorisation of forecast errors is presented in Table 2. After categorising forecast errors, possible corrective actions for different error categories can be suggested.

For some error categories, it is relatively easy to define the corrective actions. If there is no existing demand, but forecasts are still continuously produced, it is rational to calculate if the forecast accuracy is better if the forecast was 0, and then guide the forecasters accordingly.

Unforecasted sales are more understandable in the intermittent demand category than in the continuous demand category. If the demand is continuous, but forecasts do not exist, it is likely that forecaster has just forgotten to produce forecasts for that customer. If demand is intermittent, so that in some periods demand does not exist and in some periods it exists, neglecting the forecasting can be rational behaviour from the forecaster. If the forecaster is unsure of the periods when demand will exist, entering guesses to the

forecasts easily results in more forecast errors than if making the forecasts is fully neglected. In this kind of situation, the forecaster is still able to make a forecast, but only on a longer forecasting period.

Table 2 Categorisation of forecast errors and actions needed to reduce the forecasts

Unforecasted sales	–	Find out reasons for neglecting forecasting. Consider using longer forecast period	Find out reasons for neglecting forecasting
Systematic over-forecasting	Consider refraining from making forecasts	Trim the forecast down. Consider using longer forecast period	Trim the forecast down
Systematic under-forecasting	–	Trim the forecast up. Consider using longer forecast period	Trim the forecast up
Positive bidirectional errors	–	Find out the reason for the exception in forecast errors	Find out the reason for the exception in forecast errors
Negative bidirectional errors	–		
Random errors	–	No universal rules for reducing errors, longer forecasting periods should be considered	No universal rules for reducing errors
	0-demand	Intermittent demand	Continuous demand

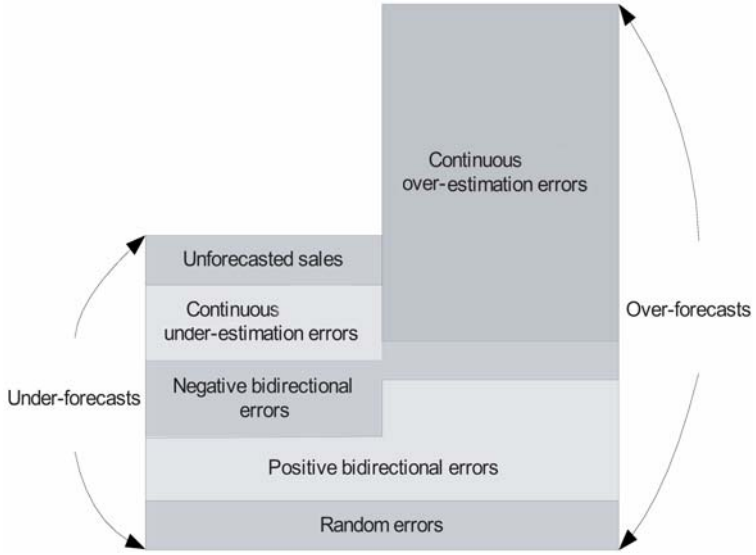
Systematic over-forecasting is typical for the forecasts that are produced by sales force, since sales people seek to minimise the possibility of under-forecasting. In this kind of situation, forecasts can be cut down to some extent without increasing forecast errors on any period. If the demand is intermittent, systematic over-forecasting can be rational behaviour from the forecaster. If the forecaster is unsure of the periods when demand will exist, entering guesses to the forecasts easily results in more forecast errors than if forecasts were evenly divided to each period. In this kind of situation, the forecaster may still be able to make a reasonable forecast, but only on a longer forecasting period.

In the case of bidirectional errors, demand is almost systematically under- or over-forecasted, but there is an exception that turns the situation the other way around for a while. Reducing the forecast errors in this category requires checking if there is an exception in the demand pattern and then finding out the reason for the exception. One possible reason for that kind of exception is a seasonal pattern that is either not recognised or reacted with a delay. If the irregularity can be explained, it is possible to give relevant feedback to the people in question.

Random errors mean that forecast is not positively or negatively biased in the long run. Some random errors will always exist, and there is no universal rule how to reduce it. Still, it is important to tell which part of the forecast errors is ‘natural’ or ‘noise’. Using longer forecasting period will smooth out a part of these errors.

Figure 2 illustrates the different types of errors from which the overall over- and under-forecasting errors (introduced in Figure 1) consist of. This analysis can be performed both on factory level and at sales unit level. Figure 2 illustrates that the net error may be the same, though the weight of separate error categories is different.

Figure 2 Categorisation of different types of forecast errors



5 Application of the framework in the case company

The framework was designed to improve the forecasting process in a large international process industry company that has several sales units and several production units. In the case company, the forecasts are produced by salespeople and then aggregated from several sales units into product family level. The company uses demand forecast for capacity planning. Capacity is allocated to forecast, but the actual production is made to order. It is possible to allocate the production between separate production units. The problem that the inaccurate demand forecasts cause is the inability to see the real capacity situation and to react to it in advance. Because the forecasts are used for capacity planning, it is justified to study absolute forecast errors, because largest absolute errors cause largest absolute problems. Also, in this situation, it is essential to know if the error is negative or positive.

The management of the company also wishes to enhance the control over the inventory policy decisions of the sales units, and this is the other reason why the forecast accuracy is under inspection. It is known that forecasts are in error, on the factory level, the net error compared to sales was 24%, but for single customers, the errors were greater, so the sum of over-forecast errors relative to actual sales was 60% and the sum of under-forecast errors was 36%. There are strong assumptions that forecasting accuracy should be better, but it is not yet identified where exactly the improvement potential lies and how it could be realised. In the beginning, there was an assumption that forecast errors are caused mostly by the sales people who are not active enough to update the demand forecasts.

To test the analysis framework, one machine from one of the production units was chosen to be the source of case data. At this machine, the forecast errors were higher than average. The data set included data on monthly level from 33 sales units and over 600 customers. The product is a bulk product, but finished to customer orders. The forecast errors were calculated as a distinction between actual sales and forecast

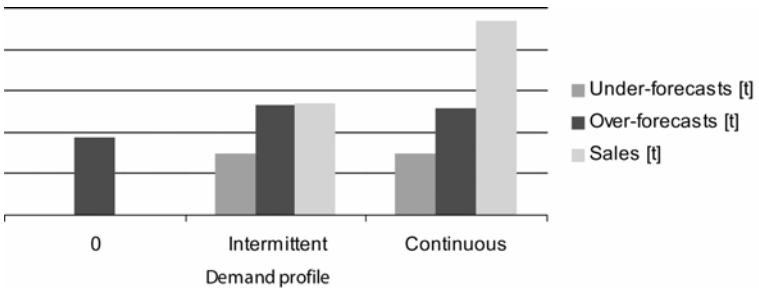
sales two months before the sales month. The data needed for the analysis was available for a seven months' time period, which was considered to be sufficiently long. Table 3 illustrates the data that was needed for the analysis.

Table 3 A piece of input data for the analysis; one product, factory level

Time	*Sales unit*	*Customer*	*Forecast*	*Sales*	*Under-forecast*	*Over-forecast*
200X/April	Sales unit 33	Customer 1	70	70		
200X/April	Sales unit 33	Customer 500	176	110		66
200X/April	Sales unit 33	Customer 13	0	50	50	
200X/May	Sales unit 1	Customer 83	72	48		24
200X/May	Sales unit 1	Customer 7	25	20		5
200X/May	Sales unit 2	Customer 606	44	0		44
200X/May	Sales unit 2	Customer 69	34	0		34

Figure 3 shows the results of categorising the forecast errors by demand profile. The demand profiles where categorised roughly into 3 categories: *0-demand:* No actual orders during the seven-month period (179 customers), *Intermittent demand*: 1–4 ordering months during the seven-month period (355 customers) and *Continuous demand*: 5–7 ordering months during the seven-month period (91 customers). The amount of sales in each demand category is added into Figure 3 in order to illustrate the percentage value of the forecast errors relative to sales.

Figure 3 Results of the analysis step 2: the magnitude of factory level forecast errors and sales by demand profile



It can be seen in Figure 3 that 25% of the over-forecast errors were caused by the 0-demand customers, and from the net error, this is as high as 50%. The results of the analysis strengthened the presumption that those sales units that have no settled, continuously ordering customers, give more inaccurate forecasts than the sales units with continuously ordering customers. Table 4 shows some comparisons between four example sales units. For example, at sales unit S4, even 89% of the over-forecast errors appeared in the 0-demand category, whereas in S2, only 3% appeared in that category. It can be concluded that this is one of the key figures that can be used when choosing different approaches for improving the forecast accuracy in different sales units. For example, the forecasts of S4 would possibly improve from adding a probability factor to the forecasts, whereas in S2, that would probably be just a waste of time. The differences in forecasting performance between the sales units that have similar demand profiles also give basis for benchmarking.

Table 4 Sales and forecast errors in tons categorised by demand profile: comparisons between four sales units

	Sales unit S1	*Sales unit S2*	*Sales unit S3*	*Sales unit S4*
Sales volume				
Total	29,967	18,847	6225	189
Intermittent demand	3900 (13%)	1297 (7%)	298 (5%)	189 (100%)
Continuous demand	26,067 (87%)	17,550 (93%)	5927 (95%)	0 (0%)
Over-forecasts				
Total	16,955	14,562	593	981
Relation to the sales	57%	77%	10%	519%
0-demand	3050 (18%)	384 (3%)	148 (25%)	870 (89%)
Intermittent demand	3628 (21%)	2010 (14%)	77 (13%)	111 (11%)
Continuous demand	10,277 (61%)	12,168 (84%)	368 (62 %)	0 (0%)
Under-forecasts				
Total	12,043	5837	2091	129
Relation to the sales	40%	31%	34%	68%
Intermittent demand	3155 (26%)	662 (11%)	251(12%)	129 (100%)
Continuous demand	8888 (74%)	5175 (87%)	1840 (88%)	0 (0%)

It can be seen in Table 4 that in S1, S2 and S3, most of the absolute forecast errors (as well as sales) appear in the category of continuous demand, so that is why that category is studied further. Figure 4 shows the results of step 3, categorising the continuous demand forecast errors more specifically on factory level. It is easy to separate forecasts that were systematically overestimated or underestimated. Separating the random errors from bidirectional errors is more complicated. In this case, the following simplification was made: When the forecast error in the analysis period was less than ±20%, the errors were categorised as random errors, and when the forecast error in the analysis period was more than ±20%, the errors were categorised as bidirectional errors. Using this parameter is discretionary, however. Table 5 shows comparisons between three sales units, analysing the continuous demand category. The analysis step 3 can also be performed for the intermittent demand category.

Figure 4 Results of the analysis step 3 on factory level, continuous demand

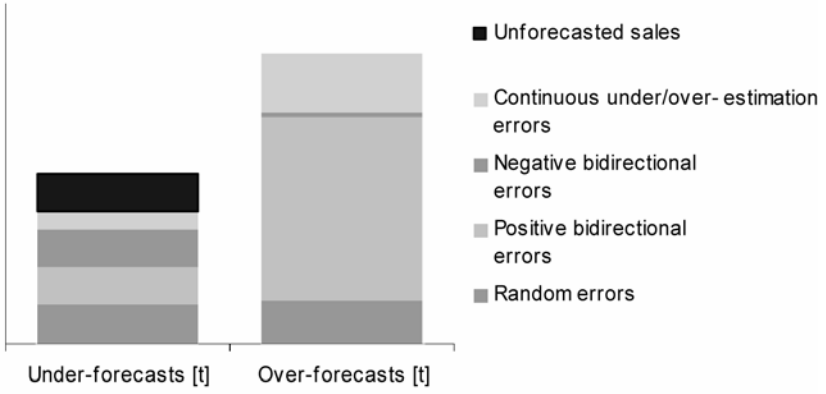


Table 5 Forecast errors of the continuous demand category divided into different types: comparisons between three sales units, S1, S2 and S3

Errors [t]	*S1*		*S2*		*S3*	
	UF	*OF*	*UF*	*OF*	*UF*	*OF*
Total	8888	10,277	5175	12,168	1840	368
R	2671 (30%)	3125 (30%)	1698 (33%)	1735 (14%)	0	0
PBE	1077 (12%)	6353 (62%)	2033 (39%)	7225 (59%)	82 (4%)	368 (100%)
NBE	794(9%)	235 (2%)	1097 (21%)	26 (0%)	0	0
S	0	564 (5%)	0	3182 (26%)	1758 (96%)	
U	4346 (49%)		347 (7%)		0	

Note: UF: Under-forecasts; NBE: Negative bidirectional errors; OF: Over-forecasts; S: Systematic over/under – estimation errors; R: Random errors; U: unforecasted sales; PBE: Positive bidirectional error.

Compared with traditional error measures, this analysis gives more information about the roots of forecast errors. In the sales unit 1, the MAPE was 86%, and in the sales unit 2, the MAPE was as high as 448%. When MAPE gives such high numbers, it is questionable if the error measure is applicable at all, and at least, it is worth studying further what causes these high numbers. Even if the error numbers were lower, MAPE does not tell if the accuracy of aggregate forecast is the result of forecasting accurately for individual customers or a lucky combination of bad forecasts that dampen each other. Therefore, using only MAPE does not provide enough basis for evaluating the potential benefits of corrective actions on forecasting performance. The other common error measures such as MAD or MSE have the same insufficiency.

The results show that there are differences between sales units in how the forecast errors are divided in different types of errors. According to this analysis, some suggestions can be made for the sales units. Sales unit S3 is doing relatively well, but has improvement potential with reducing systematic under-forecasting. Sales unit S1 has many unforecasted sales, so the reason to this should be found out. Sales unit S2 has improvement potential in reducing positive bidirectional errors, which cause 59% of the over-forecast errors in this demand category.

Examining positive bidirectional errors further revealed that usually in these cases, the demand followed a pattern where demand dropped down for a month or two, but then rose higher than average for the next one or two months. Forecast did not notice this pattern or reacted to it with a delay and this lead to bidirectional errors. Reducing bidirectional errors requires examining these demand patterns further. An analysis like this does not reveal the reasons behind these demand patterns, but points out the customers whose demand patterns should be studied further.

The analysis framework provides better understanding on how the forecast errors build up in different sales units, and where the improvement potential lies. Secondly, the differences in the forecasting performance between the sales units may enable benchmarking, which could be performed in different fields of forecasting, like dealing with new customers, communication procedures with the long-term customers, and using historical data as a basis for doing the forecast. However, the analysis results should be used only as a tool to help the managers to find out better where the problem lies in the forecasting.

6 Managerial implications

Since many kinds of development work is carried out in the case company the same time, it is impossible to reliably and quantitatively measure the benefits of this analysis framework. In addition, resulting benefits depend on the implementation of corrective actions which can take time. Still, a short description about the actions that resulted from doing this analysis in the case company will give an idea how the managerial implications were like.

First of all, taking a closer look at the forecasts revealed that the forecast errors cannot be explained by forecasters being lazy, that was the assumption in the beginning. Obviously, forecasters need more specific instructions about how, when and why forecasts should be made. It was noticed that all the forecasters had not been following the instructions that had been given, but entered very unlikely orders to forecasts. From the net errors, 50% was related to this behaviour. Feedback was decided to be given to the salespeople in question.

When assessing forecast accuracy, it turned out that the exact timing of demand was not critical for the capacity planning. So one month inaccuracy in timing was meaningless. Still, the one month forecasting period was maintained, but a new way to measure the forecast error was developed. The measure screens out the errors, where timing errs only with one month.

One development action that is under construction is making forecasts separately for the businesses that operate on stable markets and for the businesses that operate on volatile markets. Therefore, the predictability of demand in different customer groups is under evaluation to set realistic targets for forecast accuracy in different customer groups.

In the beginning, all the forecasts were the responsibility of sales people. One area of further research is finding out which customer groups or product groups should be taken out from the sales people's responsibility. This will be accomplished by evaluating, when the salespeople have true access to such demand information that enables better forecasting than mathematical models.

7 Conclusions

Forecasts that are produced by sales people are in jeopardy to err for different reasons. Practice shows that managing judgemental sales forecasting can turn out so challenging that traditional error measures are not sufficient in controlling the forecasting process. Because of the heterogeneity of customer base and heterogeneity of people who are producing the forecasts, it is difficult to generate a general view about what are the most substantive weaknesses of the forecasting process. Therefore, a proper analysis framework is needed. Analysing historical demand and forecasting data is an economical way to begin the development work, because such data is easily available.

The analysis framework presented in this paper enables categorising forecast errors into different types, and thereby helps to point out proper approaches for reducing the sales forecast errors. The results of the case company show that performing this kind of analysis is of use before a specific forecasting technique or incentive system is implemented.

The possible benefits of this kind of analysis are the following:

1 enhancing the understanding and challenging the assumptions about the achievable forecasting performance

2 enabling internal benchmarking in making the forecasts for different market areas and customer groups

3 finding out the most significant sources or forecast errors, and in this way, focusing the development resources

4 clarifying the feedback given to the forecasters about their forecasting performance.

It must be kept in mind that the approach presented here is only one step towards better forecasting performance. The benefit of this kind of analysis is that it is quick and easy to repeat, but it has its restrictions. Considerable amount of demand information is in qualitative form, and therefore it impossible to finally solve all the problems of forecasting with categorising quantitative information. Anyhow, the descriptive statistics that result can be used as an objective basis for forming hypotheses about the reasons of errors for a qualitative study, for example interview study.

References

Bartezzaghi, E. and Verganti, R. (1995) 'Managing demand uncertainty through order overplanning', *International Journal of Production Economics*, Vol. 40, No. 1, pp.107–120.

Chen, F. (1998) 'Echelon reorder points, installation reorder points, and the value of centralised demand information', *Management Science*, Vol. 44, No. 12, p.S221.

Hanfield, R.B. and Nicholas, E.L. (Eds) (1999) *Introduction to Supply Chain Management*, 1st edition, Upper Saddle River, NJ: Prentice Hall.

Helms, M.M., Ettkin, L.P. and Chapman, S. (2000) 'Supply chain forecasting; collaborative forecasting supports supply chain management', *Business Process Management Journal*, Vol. 6, No. 5, pp.392–407.

Holmström, J. (1998) 'Handling product range complexity: a case study on re-engineering demand forecasts', *Business Process Management Journal*, Vol. 4, No. 3, pp.241–258.

Holweg, M., Disney, S., Holmström, J. and Småros, J. (2005) 'Supply chain collaboration: making sense of the strategy continuum', *European Management Journal*, Vol. 23, No. 2, pp.170–181.

Lawrence, M., Goodwin, P., O'Connor, M. and Önkal, D. (2006) 'Judgmental forecasting: a review of progress over the last 25 years', *International Journal of Forecasting*, Vol. 22, No. 3, pp.493–518.

Lin, F., Huang, S. and Lin, S. (2002) 'Effects of information sharing on supply chain performance in electronic commerce', *IEEE Transactions on Engineering Management*, Vol. 49, No. 3, p.258.

Lines, A.H. (1996) 'Forecasting - key to good service at low cost', *Logistics Information Management*, Vol. 9, No. 4, pp.24–27.

Mentzer, J.T. and Kahn, K.B. (1997) 'State of sales forecasting systems in corporate America', *Journal of Business Forecasting Methods and Systems*, Vol. 16, No. 1, p.6.

Moon, M.A. and Mentzer, J.T. (1999) 'Improving salesforce forecasting', *The Journal of Business Forecasting*, Vol. 18, No. 2, pp.7–12.

Mentzer, J.T. and Moon, M.A. (Eds) (2005) *Sales Forecasting Management: A Demand Management Approach*, 2nd edition, Thousand Oaks, CA: Sage Publications, Inc.

Småros, J. and Hellström, M. (2004) 'Using the assortment forecasting method to enable sales force involvement in forecasting: a case study', *International Journal of Physical Distribution and Logistics Management*, Vol. 34, No. 2, pp.140–157.

Van Aken, J.E. (2004) 'Management research based on the paradigm of the design sciences: the quest for field-tested and grounded technological rules', *Journal of Management Studies*, Vol. 41, No. 2, pp.219–246.

Webby, R. and O'Connor, M. (1996) 'Judgmental and statistical time series forecasting: a review of the literature', *International Journal of Forecasting*, Vol. 12, pp.91–118.

Zotteri, G. and Verganti, R. (2001) 'Multi-level approaches to demand management in complex environments: an analytical model', *International Journal of Production Economics*, Vol. 71, No. 1, pp.221–233.

A Lagrangian Relaxation approach for production planning with demand uncertainty

Haoxun Chen

Industrial Systems Optimisation Laboratory,
Charles Delaunay Institute (FRE CNRS 2848),
University of Technology of Troyes,
Troyes 10010, France
Fax: +0033-325-715649
E-mail: haoxun.chen@utt.fr

Abstract: A production planning problem with stochastic demands is considered in this paper. The problem is to determine over a given time horizon the production quantity of each intermediate/final product at each facility of finite capacity so that a system-wide total cost is minimised while meeting given service level requirements for the final products. After reformulating the stochastic decision problem as a multiitem, multistage capacitated lot-sizing problem with a non-linear cost function using deterministic equivalence, it is solved by using a Lagrangian Relaxation (LR) approach enhanced with a local search method based on a modified simplex algorithm. Numerical experiments show that the approach can find high quality near-optimal solutions for randomly generated problems of realistic sizes in a computation time much shorter than that of an exact algorithm.
[Received on 2 February 2007; Revised 28 May 2007; Accepted 7 June 2007]

Keywords: production planning; lot sizing; demand uncertainty; lagrangian relaxation; LR; local search.

Reference to this paper should be made as follows: Chen, H. (2007) 'A Lagrangian Relaxation approach for production planning with demand uncertainty', *European J. Industrial Engineering*, Vol. 1, No. 4, pp.370–390.

Biographical notes: Haoxun Chen is Professor at Industrial Systems Optimisation Laboratory, Charles Delaunay Institute (FRE CNRS 2848), University of Technology of Troyes, France. He received his PhD in Systems Engineering from Xi'an Jiaotong University, China, in 1990. His research interests include supply chain management, production planning and scheduling and discrete event systems. He has published more than 70 papers in technical journals and conference proceedings and received the 1998 King-Sun Fu Memorial Best Transactions Paper Award from IEEE Robotics and Control Society.

1 Introduction

Effective production planning is critical for manufacturers to reduce production and inventory costs while improving services to customers. With recent advances of Mathematical Programming (MP) techniques, commercial Advanced Planning and

Scheduling (APS) systems can now solve practical production planning problems in a reasonable computation time. However, one difficulty for the application of MP and rolling schedule-based APS systems is that they ignore demand uncertainty. Since customer demands cannot be precisely predicted in advance in most situations, a good planning tool should take account of demand uncertainty.

In this paper, we consider a supply/production network with multiple levels (stages) consisting of multiple facilities of finite capacity where multiple final products are produced through the network. The demand of each final product in each period is stochastic with a known probability distribution. As in most studies on production planning, we assume that the production decisions of all the facilities are made centrally based on a central model. This assumption is realistic in the case when all facilities in the supply/production network are owned by a single enterprise. The problem is to determine the production quantity for each intermediate/final product at each facility over a given time horizon to minimise a system-wide total cost subject to given service level constraints for the final products, where the service level for a product is defined as the probability of non-stockout of the product.

The literature relating to our present work includes plenty of papers on deterministic lot-sizing and a few papers on stochastic lot-sizing for production/distribution planning. Existing methods for deterministic lot-sizing with general production structure fall into four categories: decomposition approaches (Tempelmeier and Helber, 1994), meta-heuristic search approaches (Salomon, 1991), linear programming-based approaches (Harrison and Lewis, 1995; Katok et al., 1998) and Lagrangian Relaxation (LR)-based approaches (Billington et al., 1986; Tempelmeier and Derstroff, 1996). A recent review of the deterministic lot-sizing literature is given by Drexl and Kimms (1997).

The research on stochastic lot-sizing is relatively recent. Most studies consider either a single product problem or a single stage problem (Sox et al., 1999). For a single product and single stage dynamic lot sizing problem with random demand and non-stationary costs, an optimal algorithm is developed, which resembles the Wagner-Whitin algorithm but with some additional feasibility constraints (Sox, 1997). A multiperiod single-item inventory lot-sizing problem with stochastic demands is studied under a 'static-dynamic uncertainty' strategy (Tarim and Kingsman, 2004). In the strategy, the replenishment periods are fixed at the beginning of the planning horizon, but the actual orders are determined only at those replenishment periods and will depend upon that demand realised. The authors present a mixed integer programming formulation that determines both the replenishment periods and the actual orders in a single step. Multiple items procurement planning in the consumer goods wholesaling and retailing industry is studied under stochastic demand settings (Martel et al., 1995). By computing procurement plans over rolling planning horizons, a difficult sequential stochastic planning problem is transformed into a multiple-period static decision problem under risk. A branch and bound algorithm is developed to exactly solve the equivalent deterministic decision problem, and a piecewise concave approximation method is proposed to reduce this problem to a linear program with 0–1 variables.

There have been few papers addressing multiple stage stochastic lot sizing problems. A model for managing demand uncertainty in supply chain planning is proposed based on stochastic programming (Gupta and Maranas, 2003; Gupta et al., 2000). In the proposed model, the manufacturing decisions are modelled as 'here-and-now' decisions, which are made before demand realisation, while the logistics decisions are postponed

in a 'wait-and-see' mode after demand realisation. A chance constraint programming approach in conjunction with a two-stage stochastic programming methodology is utilised for capturing the trade-off between customer satisfaction level and production costs. A model of a multilevel capacity-constrained system with external stochastic demand is presented for production-inventory planning (Grubbstrom and Wang, 2003). Unlike the total cost objective adopted in the majority of traditional production-inventory models, the expected net present value is used as the objective function. Dynamic programming is chosen as the solution procedure of the model. A production planning problem with randomness in the estimate of future demands is studied under two demand settings (Albritton et al., 2000). The problem is solved by using two variants of Monte Carlo sampling-based optimisation techniques. Managing material flows in a multiechelon supply-distribution network is considered under probabilistic time-varying demand settings (Martel, 2003). The problem is formulated as a stochastic programme with recourse, and its deterministic equivalent programme is approximated by a multilevel lot-sizing model based on 'risk inflated effective demands'. Heuristic planning policies based on a Distribution Resources Planning (DRP)-decomposition of the approximate model and planning time fences or allocation algorithms are then developed, which outperform a classical DRP approach using safety stocks.

The existing methods for production planning with multiple stages and stochastic demands are either exact ones such as stochastic programming and dynamic programming algorithms that cannot solve problems of realistic sizes in a reasonable computation time or heuristics that cannot assure the quality of the solution. In this paper, we try to develop an effective method for large production planning problems with stochastic demands. Because of the complexity of the problem, we restrict ourselves to search for an open-loop solution, for which the rolling schedule scheme can be applied. The difference between our model and a MP-based APS model is that we treat the demands of each final product as a stochastic process.

We show that the open-loop decision problem can be equivalently transformed into a deterministic problem. The new problem has similar constraints as in classical multi-item, multistage lot-sizing models but with a non-linear cost function. The non-linearity makes the deterministic problem very difficult to solve. We extend our previously developed LR approach for supply chain planning with deterministic demand (Chen and Chu, 2003) to the problem. Different from other existing LR approaches that relax capacity constraints and/or inventory balance constraints for deterministic lot-sizing problems, our approach only relaxes the technical constraints that each 0–1 setup variable must take value 1 if its corresponding continuous variable is positive. The relaxed problem, which is a non-linear programme, is approximately solved by using a partial linearisation procedure, while the Lagrange multipliers are updated by using a Surrogate Subgradient (SSG) that ensures the convergence of the dual problem. To obtain a high quality solution of the original problem, at each iteration of the dual maximisation, a feasible solution of the original problem is constructed from the solution of the relaxed problem. The feasible solution is further improved by using a local search based on a modified simplex algorithm that takes account of setup costs and the non-linearity of the objective function. Numerical testing for randomly generated medium size problems shows that our approach can obtain near-optimal solutions in a short computation time.

The remainder of the paper is organised as follows: the planning problem is described and its mathematical model is established in Section 2. Section 3 is dedicated to the

solution methodology of the model with four subsections presenting LR, partial linearisation for the relaxed problem, SSG method for the dual problem, and construction of a feasible solution and its local search improvement, respectively. Computational results of the numerical testing are presented in Section 4. Concluding remarks are given in Section 5.

2 Problem description and formulation

We consider a production network consisting of multiple facilities. Each facility has a finite capacity and produces multiple intermediate or final products (also called items hereafter). Its capacity in each period is measured in time units where the production of each unit of an item consumes a given amount of time. To produce an intermediate or final product, it may require multiple units of several other intermediate products and/or raw materials. This requirement is defined through the Bill-of-Materials (BOM) of the product. The production of an item in a facility has a lead time that represents the minimum time from the release of a production order for the item to its availability in the facility. If production capacity is not available or the required raw materials or intermediate products are not available, the real production lead time for the item may be longer than the minimum time. The demands of final products are stochastic but with known probability distributions. It is assumed that external demands only incur at the final products level. The demand for each item at any intermediate facility is only generated by the demands coming from its downstream facilities. A service level, which defines the minimum probability of meeting customer demands on-time, is specified for each final product. The cost of operating the system is the sum of the costs of all its facilities, where the cost of each facility consists of inventory holding costs and production setup costs for all items involved. For simplicity, production setup times are not considered so that the capacity utilisation of each facility in each period is a linear function of production quantities, and as in most papers on production planning, setups for producing the same product in subsequent periods cannot be merged into a single setup. In the following, a mathematical model is established for the planning problem. The indices, the parameters and the variables of the model are first introduced.

Indices

$i = 1, \ldots, M$	index of final products
$i = M + 1, \ldots, N$	index of intermediate products
$t = 1, \ldots, T$	index of planning periods
$k = 1, \ldots, K$	index of facilities.

Parameters

h_i: inventory holding cost per unit of item i per period, $i = 1, \ldots, N$

cs_i: setup cost per setup for item i, $i = 1, \ldots, N$

L_i: lead time for item i, $i = 1, \ldots, N$

a_{ij}: number of units of item i required for the production of one unit of item j, $i = 1, \ldots, N, j = 1, \ldots, N$

d_{it}: random demand for final product i in period t, $i = 1, \ldots, M, t = 1, \ldots, T$

$G_{it}(.)$: probability distribution function of random variable $S_{it} = d_{i1} + \ldots + d_{it-1} + d_{it}$ which is bijective so that G_{it}^{-1} exists, $i = 1, \ldots, M, t = 1, \ldots, T$

p_{ik}: capacity consumption of facility k for producing each unit of item i (in units of time), $i = 1, \ldots, N, k = 1, \ldots, K$

C_{kt}: capacity of facility k in period t (in units of time), $k = 1, \ldots, K, t = 1, \ldots, T$

α_{it}: required service level for final product i in period t, $i = 1, \ldots, M, t = 1, \ldots, T$.

Variables

I_{it}: inventory level of item i at the end of period t, $i = 1, \ldots, N, t = 1, \ldots, T$

I_{it}^{+}: on-hand inventory of item i at the end of period t, $I_{it}^{+} = \max(0, I_{it})$, $i = 1, \ldots, M$, $t = 1, \ldots, T$

δ_{it}: setup variables. $\delta_{it} = 1$ if item i is produced in period t, 0 otherwise

x_{it}: production quantity of item i in period t.

We look for an open-loop solution of the planning problem. In this case, x_{it} and δ_{it} do not depend on the state $I_{i,\,t-1}$ observed at the beginning of period t and consequently are not random variables. Thus, the planning problem can be formulated as:

Problem *SP*:

$$\text{Min } J = \mathrm{E}\left\{\sum_{i=1}^{M}\sum_{t=1}^{T} h_i I_{it}^{+}\right\} + \sum_{i=1}^{M}\sum_{t=1}^{T} \mathrm{cs}_i \delta_{it} + \sum_{i=M+1}^{N}\sum_{t=1}^{T} h_i I_{i,t} + \sum_{i=M+1}^{N}\sum_{t=1}^{T} \mathrm{cs}_i \delta_{it}$$

s.t.

$$I_{i,t-1} + x_{i,t-Li} - I_{i,t} = d_{it}, i = 1,\ldots,M, t = 1,\ldots,T \tag{1a}$$

$$I_{i,t-1} + x_{i,t-Li} - I_{i,t} - \sum_{j=1}^{N} a_{ij} x_{jt} = 0,\ i = M+1,\ldots,N,\ t = 1,\ldots,T \tag{1b}$$

$$\sum_{i=1}^{N} p_{ik} x_{it} \le C_{kt},\ k = 1,\ldots,K,\ t = 1,\ldots,T \tag{2}$$

$$\mathrm{Prob}\left(I_{it} \ge 0\right) \ge \alpha_{it},\ i = 1,\ldots,M, t = 1,\ldots,T \tag{3}$$

$$I_{it} \ge 0,\ i = M+1,\ldots N, t = 1,\ldots,T \tag{4}$$

$$\delta_{it} = 1 \text{ if } x_{it} > 0,\ i = 1,\ldots N, t = 1,\ldots,T \tag{5}$$

$$x_{it} \ge 0,\ i = 1,\ldots N, t = 1,\ldots,T \tag{6}$$

$$\delta_{it} = 0 \text{ or } 1\ \ i = 1,\ldots N, t = 1,\ldots,T \tag{7}$$

where the four terms of the cost function J are the expected inventory holding cost for final products, setup cost for final products, inventory holding cost for intermediate products, setup cost for intermediate products, respectively. Constraints (1a) and (1b) are inventory balance equations for final products and intermediate products, respectively. Constraints (2) are facility capacity constraints. Constraints (3) are service level constraints for final products. Constraints (4) imply that the stockout of any intermediate product is not allowed. Decision variables x_{it} are coupled with variables δ_{it} through Constraints (5). Constraints (6) and (7) define the domain of variable x_{it} and δ_{it}, respectively.

For final product i, Constraints (1a) can be reformulated as

$$I_{i,t} = I_{i,0} + \sum_{j=1}^{t} x_{i,j-Li} - \sum_{j=1}^{t} d_{ij}, t = 1,\ldots,T$$

Consequently, Constraints (3) can be reformulated as

$$\mathrm{Prob}\left(I_{i,0} + \sum_{j=1}^{t} x_{i,j-Li} - \sum_{j=1}^{t} d_{ij,} \geq 0 \right) \geq \alpha_{it,} \quad t = 1,\ldots,T$$

or equivalently as

$$\mathrm{Prob}\left(\sum_{j=1}^{t} d_{ij} \leq I_{i,0} + \sum_{j=1}^{t} x_{i,j-Li} \right) \geq \alpha_{it,} \quad t = 1,\ldots,T$$

That is,

$$G_{it}\left(I_{i,0} + \sum_{j=1}^{t} x_{i,j-Li} \right) \geq \alpha_{it,} \quad t = 1,\ldots,T$$

or equivalently,

$$I_{i,0} + \sum_{j=1}^{t} x_{i,j-Li} \geq G_{it}^{-1}(\alpha_{it}), \quad t = 1,\ldots,T$$

Let

$$v_{it} = \mathrm{E}\left\{ h_i I_{it}^{+} \right\}, \quad i = 1,\ldots,M, \quad t = 1,\ldots,T$$

We have

$$v_{it} = h_i \int_0^{I_{i,0}+\sum_{j=1}^{t} x_{i,j-Li}} \left(I_{i,0} + \sum_{j=1}^{t} x_{i,j-Li} - s \right) \mathrm{d}G_{it}(s), \; i = 1,\ldots,M, \quad t = 1,\ldots,T$$

The cost function of problem *SP* becomes:

$$J = \sum_{i=1}^{M}\sum_{t=1}^{T} v_{it} + \sum_{i=M+1}^{N}\sum_{t=1}^{T} h_i I_{it} + \sum_{i=1}^{N}\sum_{t=1}^{T} cs_i \delta_{it}$$

Since the demand of product i is always non-negative, we have $G_{i0}^{-1}(\alpha_{i0}) = 0$. Define y_{it} as:

$$y_{it} = I_{i,0} + \sum_{j=1}^{t} x_{i,j-Li} - G_{it}^{-1}(\alpha_{it}),\ i = 1,\ldots,M,\ t = 0,1,\ldots,T$$

$$y_{it} = I_{i,t},\ i = M+1,\ldots,N,\ t = 0,1,\ldots,T$$

We have

$$y_{it} = y_{i,t-1} + x_{i,t-Li} - \left(G_{it}^{-1}(\alpha_{it}) - G_{it-1}^{-1}(\alpha_{it-1})\right),\ i = 1,\ldots,M,\ t = 1,\ldots,T$$

$$y_{i,t-1} + x_{i,t-Li} - y_{i,t} - \sum_{j=1}^{N} a_{ij} x_{jt} = 0,\ i = M+1,\ldots,N,\ t = 1,\ldots,T$$

$$y_{it} \geq 0,\ i = 1,\ldots,N,\ t = 1,\ldots,T$$

and v_{it} becomes:

$$v_{it} = h_i \int_0^{y_{it}+G_{it}^{-1}(\alpha_{it})} \left(y_{it} + G_{it}^{-1}(\alpha_{it}) - s\right) dG_{it}(s),\ i = 1,\ldots,M,\ t = 1,\ldots,T \tag{8}$$

Note that $y_{it} + G_{it}^{-1}(\alpha_{it}) = I_{i,0} + \sum_{j=1}^{t} x_{i,j-Li}$ is the initial inventory of product i plus its cumulative production quantity that can be used to fill the demand of the product until period t and that G_{it} is the probability distribution function of the cumulative demand of the product until the period. The function v_{it} is thus the same as the inventory holding cost function of the classical newsboy model.

Problem *SP* can therefore be reformulated as

Problem *P*:

$$\text{Min } J = \sum_{i=1}^{M}\sum_{t=1}^{T} v_{it} + \sum_{i=M+1}^{N}\sum_{t=1}^{T} h_i y_{it} + \sum_{i=1}^{N}\sum_{t=1}^{T} cs_i \delta_{it}$$

s.t.

$$y_{it} = y_{it-1} + x_{i,t-Li} - \left(G_{it}^{-1}(\alpha_{it}) - G_{it-1}^{-1}(\alpha_{it-1})\right),\ i = 1,\ldots,M,\ t = 1,\ldots,T \tag{9a}$$

$$y_{i,t-1} + x_{i,t-Li} - y_{i,t} - \sum_{j=1}^{N} a_{ij} x_{jt} = 0,\ i = M+1,\ldots,N,\ \ t = 1,\ldots,T \tag{9b}$$

$$\sum_{i=1}^{N} p_{ik} x_{it} \leq C_{kt},\ k = 1,\ldots,K,\ t = 1,\ldots,Ti = 1 \tag{10}$$

$$\delta_{it} = 1 \text{ if } x_{it} > 0, i = 1,\ldots,N,\ t = 1,\ldots,T \tag{11}$$

$$x_{it} \geq 0, i = 1,\ldots,N,\ t = 1,\ldots,T \tag{12a}$$

$$y_{it} \geq 0, i = 1,\ldots,N,\ t = 1,\ldots,T \tag{12b}$$

$$\delta_{it} = 0 \text{ or } 1, i = 1,\ldots,N,\ t = 1,\ldots,T \tag{13}$$

where $y_{i,0} = I_{i,0},\ G_{i0}^{-1}(\alpha_{i0}) = 0,\ i = 1,\ \ldots,\ M$

The above model is a multiitem, multistage capacitated lot-sizing model with a non-linear cost function. The objective function becomes non-linear because negative inventories in the original stochastic programming model do not bring a contribution to the inventory holding cost.

3 Solution methodology

Problem *P* is more difficult than the (deterministic) capacitated multi-item, multistage lot sizing problem with setup costs which is NP-hard (Drexl and Kimms, 1997) because of the non-linearity of its cost function. This inspires us to seek for an approximate approach to solve the problem. LR has been used to solve multiitem, multistage lot sizing problems (Billington et al., 1986; Tempelmeier and Derstroff, 1996). However, those LR approaches relax capacity constraints and/or inventory balance constraints, which makes the construction of a high-quality feasible solution from the solution of the relaxed problem difficult because of the excessive relaxation of important constraints. In our previous work, we developed an effective LR approach for a supply chain planning problem with deterministic demands (Chen and Chu, 2003). In the approach, only the technical constraints that each 0–1 setup variable must take value 1 if its corresponding continuous production variable is positive are relaxed. In this section, we extend the approach to the non-linear multiitem, multistage lot sizing problem *P*. In order to do so, we first reformulate the Constraints (11) (and (5)) coupling x_{it} and δ_{it} as a set of bilinear equality constraints (14):

$$x_{it}\left(1-\delta_{it}\right)=0, i=1,\ldots,N,\ t=1,\ldots,T \tag{14}$$

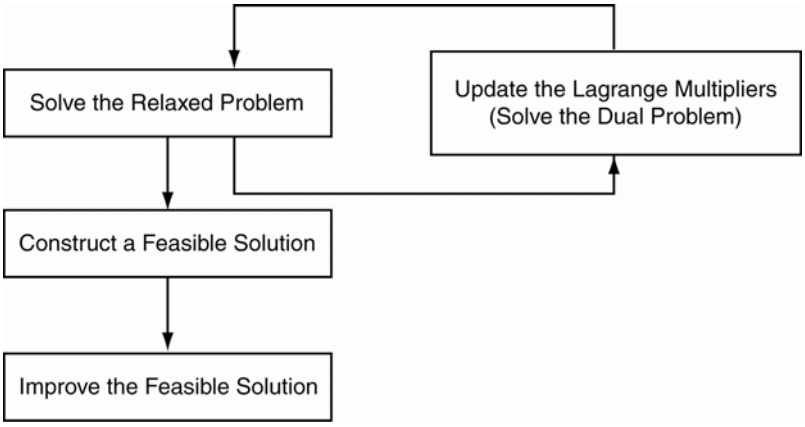
Note that in the literature and textbooks, the coupling constraints are usually formulated as a set of linear inequality constraints by introducing a big positive number *B* (or multiple big positive numbers B_{it}) as:

$$x_{it} \le B\delta_{it}\ \left(\text{or } x_{it} \le B_{it}\delta_{it}\right),\ i=1,\ldots,N,\ t=1,\ldots,T \tag{15}$$

However, if we relax the Constraints (15) by using Lagrange multipliers, the solution obtained by using the corresponding LR approach is very sensitive to the parameter *B* (parameters B_{it}). Because a good estimation of the parameter (the parameters) is hard to be obtained, a poor performance of the LR approach was frequently observed in our preliminary tests. For this reason, we use the Constraints (14) rather than the Constraints (15) to formulate the technical constraints between x_{it} and δ_{it} although the former exhibits an undesirable non-linear feature. Fortunately, by using a recently developed Surrogate Subgradient (SSG) method for the lagrangian dual problem, the inconvenience caused by this non-linearity can be well overcome.

The algorithmic scheme of our LR approach is given in Figure 1. Roughly explained, the approach relaxes the Constraints (14) that replace the Constraints (11). The relaxed problem of *P* is approximately solved by applying a Gauss Seidel iteration-like method and a partial linearisation method to be presented in Sections 3.1 and 3.2, respectively. The dual problem is solved in Section 3.3 by using the SSG method which ensures the convergence of the solution of the dual problem to its optimal solution in case of the approximate resolution of the relaxed problem. The construction of a feasible solution of *P* and its improvement using local search based on a modified simplex algorithm is presented in Section 3.4.

Figure 1 Algorithmic Scheme of the LR approach



3.1 Lagrangian relaxation

By introducing Lagrange multipliers $\{\lambda_{it}\}$ to relax the Constraints (14), we get a relaxed problem of *P* as:

RP(λ):

$$\underset{x,y,\delta}{\text{Min}}\, L(x,y,\delta) = \sum_{i=1}^{M}\sum_{t=1}^{T} v_{it} + \sum_{i=M+1}^{N}\sum_{t=1}^{T} h_i y_{it} + \sum_{i=1}^{N}\sum_{t=1}^{T} cs_i \delta_{it} + \sum_{i=1}^{N}\sum_{t=1}^{T} \lambda_{it} x_{it}\left(1-\delta_{it}\right) \tag{16}$$

s.t. (9)–(10), (12)–(13).

where Constraints (12) are Constraints (12a) plus Constraints (12b).

The problem cannot be decomposed into independent subproblems because of the coupling terms $x_{it}\delta_{it}$ in its objective function. However, it can be approximately solved by using a Gauss-Seidel iteration-like method that alternately solves a subproblem with given $\{\delta_{it}\}$ and a subproblem with given $\{x_{it}, y_{it}\}$. The two subproblems are:

RP^1($\lambda, \hat{\delta}$):

$$\underset{x,y}{\text{Min}}\, L^1(x,y) = \sum_{i=1}^{M}\sum_{t=1}^{T} v_{it} + \sum_{i=M+1}^{N}\sum_{t=1}^{T} h_i y_{it} + \sum_{i=1}^{N}\sum_{t=1}^{T} cs_i \hat{\delta}_{it} + \sum_{i=1}^{N}\sum_{t=1}^{T} \lambda_{it} x_{it}\left(1-\hat{\delta}_{it}\right) \tag{17}$$

s.t. (9)–(10), (12).

RP^2($\lambda, \hat{x}, \hat{y}$):

$$\underset{\delta}{\text{Min}}\, L^2(\delta) = \sum_{i=1}^{M}\sum_{t=1}^{T} \hat{v}_{it} + \sum_{i=M+1}^{N}\sum_{t=1}^{T} h_i \hat{y}_{it} + \sum_{i=1}^{N}\sum_{t=1}^{T} cs_i \delta_{it} + \sum_{i=1}^{N}\sum_{t=1}^{T} \lambda_{it} \hat{x}_{it}\left(1-\delta_{it}\right) \tag{18}$$

s.t. (13).

Problem RP^1 can be reduced to:

$$\underset{x,y}{\text{Min}}\, \hat{L}^1(x,y) = \sum_{i=1}^{M}\sum_{t=1}^{T} v_{it} + \sum_{i=M+1}^{N}\sum_{t=1}^{T} h_i y_{it} + \sum_{i=1}^{N}\sum_{t=1}^{T}\left(\lambda_{it} - \lambda_{it}\hat{\delta}_{it}\right) x_{it}$$

s.t. (9)–(10), (12).

Problem RP[1] has linear constraints and a non-linear objective function (v_{it} is non-linear).

Problem RP[2] can be reduced to:

$$\operatorname{Min}_{\delta} \hat{L}^2(\delta) = \sum_{i=1}^{N}\sum_{t=1}^{T} \mathrm{cs}_i \delta_{it} + \sum_{i=1}^{N}\sum_{t=1}^{T} \lambda_{it}\hat{x}_{it}\left(1-\delta_{it}\right) = \sum_{i=1}^{N}\sum_{t=1}^{T}\left(\mathrm{cs}_i - \lambda_{it}\hat{x}_{it}\right)\delta_{it} + \sum_{i=1}^{N}\sum_{t=1}^{T}\lambda_{it}\hat{x}_{it}$$

s.t. (13).

Its optimal solution is

$$\delta_{it} = 0, \text{ if } \mathrm{cs}_i - \lambda_{it}\hat{x}_{it} > 0 \tag{19}$$

$$\delta_{it} = 1, \text{ otherwise.}$$

In each iteration of the LR approach, the relaxed problem is approximately solved by using a Gauss Seidel iteration-like method. That is, problem RP[1] and RP[2] are solved alternately until a certain condition holds (see Section 3.3 for the condition) or the solution of RP1 or RP2 keeps unchanged in two iterations of the method.

Let $D(\lambda)$ be the optimal objective value of RP for any Lagrange multiplier vector $\lambda = \{\lambda_{it}, i = 1, \ldots, N, t = 1, \ldots, T\}$. The lagrangian dual problem of RP is:

$$\mathrm{DP}: \operatorname{Max}_{\lambda} D(\lambda) \tag{20}$$

where $D(\lambda) = \operatorname{Min}_{x,y,\delta} \{L(\lambda, x, y, \delta) \mid \text{s.t. (9)–(10), (12)–(13)}\}$.

3.2 Partial linearisation for the relaxed problem

Defining $F_{it}(x) = h_i \int_0^x (x-s)\mathrm{d}G_{it}(s)$, we have

$$v_{it} = F_{it}\left(y_{it} + G_{it}^{-1}\left(\alpha_{it}\right)\right)$$

$$\frac{dv_{it}}{\mathrm{d}y_{it}} = h_i \int_0^{y_{it}+G_{it}^{-1}(\alpha_{it})} \mathrm{d}G_{it}(s)$$

$$= h_i G_{it}\left(y_{it} + G_{it}^{-1}\left(\alpha_{it}\right)\right) - h_i G_{it}(0) = h_i G_{it}\left(y_{it} + G_{it}^{-1}\left(\alpha_{it}\right)\right)$$

$\mathrm{d}^2 v_{it}/\mathrm{d}y_{it}^2 = h_i g_{it}(y_{it} + G_{it}^{-1}(\alpha_{it})) \geq 0$, where g_{it} is probability density function of random variable S_t.

Thus, the non-linear term $\sum_{i=1}^{M}\sum_{t=1}^{T} v_{it}$ of the objective function of RP[1] is convex, and so is the function itself. Since all constraints of RP[1] are linear, the relaxed subproblem can therefore be solved by using a partial linearisation method proposed by Patriksson (1993).

At any point $(\{x_{it}^k\}, \{y_{it}^k\})$, the linearisation of v_{it} is:

$$F_{it}\left(y_{it}^k + G_{it}^{-1}\left(\alpha_{it}\right)\right) - h_i G_{it}\left(y_{it}^k + G_{it}^{-1}\left(\alpha_{it}\right)\right)\left(y_{it} - y_{it}^k\right)$$

The linearisation of $\hat{L}^1$ at the point, denoted by $\bar{L}^1((x, y), (x^k, y^k))$, is thus:

$$\begin{aligned}\bar{L}^{-1}\left(\left(x,y\right),\left(x^k,y^k\right)\right) = &\sum_{i=1}^{M}\sum_{t=1}^{T}\left\{F_{it}\left(y_{it}^k+G_{it}^{-1}\left(\alpha_{it}\right)\right)\right\}\\ &\left\{+h_iG_{it}\left(y_{it}^k+G_{it}^{-1}\left(\alpha_{it}\right)\right)\left(y_{it}-y_{it}^k\right)\right\}\\ &+\sum_{i=M+1}^{N}\sum_{t=1}^{T}h_iy_{it}+\sum_{i=1}^{N}\sum_{t=1}^{T}\left(\lambda_{it}-\lambda_{it}\hat{\delta}_{it}\right)x_{it}\end{aligned} \tag{21}$$

The partial linearisation method solves a linear programming problem and performs a line search at each iteration. For our problem, at each iteration k, the method solves the following linear programming problem:

RP^{1k}:

$$\underset{x,y}{\text{Min}}\ \bar{L}^{-1}\left(\left(x,y\right),\left(x^k,y^k\right)\right) \tag{22}$$

s.t. (9)–(10), (12)

and performs a line search to minimise $\hat{L}^1$ for problem RP^1, that is,

$$\underset{\rho}{Min}\left\{\hat{L}^1(x,y)\middle|(x,y)=\rho\left(x^k,y^k\right)+(1-\rho)\left(\bar{x}^k,\bar{y}^k\right),0\le\rho\le1\right\} \tag{23}$$

where $\hat{L}^1(x,\ y)=\sum_{i=1}^{M}\sum_{t=1}^{T}v_{it}+\sum_{i=M+1}^{N}\sum_{t=1}^{T}h_iy_{it}+\sum_{i=1}^{N}\sum_{t=1}^{T}(\lambda_{it}-\lambda_{it}\hat{\delta}_{it})_{x_{it}}$, and $(\bar{x}^k,\bar{y}^k)$ is an optimal solution of RP^{1k}. The starting point (x^{k+1}, y^{k+1}) of iteration $k+1$ is the solution of the line search at iteration k.

The iterative procedure continues until (x^k, y^k) also solves RP^{1k}. That is, $\bar{L}^1((x^k,\ y^k),\ (x^k,\ y^k)) = \bar{L}^{-1}\ ((\bar{x}^k,\bar{y}^k),\ (x^k,\ y^k))$. Initially, ($x^0$, y^0) is taken as a feasible solution of RP^1.

3.3 Surrogate subgradient method for the dual problem

Since the relaxed problem is only approximately solved, the well-known Subgradient (SG) method cannot be used for the dual problem. Instead, we use a recently developed SG-like method called Surrogate Subgradient (SSG) method (Zhao et al., 1999) to solve the problem. The method ensures that the solution of the dual problem converges to its optimal solution in case of an approximate resolution of the relaxed problem under some conditions.

SSG is similar to SG except for the definition of surrogate subgradient different from the definition of subgradient and the step sizing scheme for the update of Lagrange multipliers. For our problem, the Surrogate Subgradient is defined as

$$\tilde{g}^k=\tilde{g}\left(x^k,\delta^k\right)=\left\{x_{it}^k\left(1-\delta_{it}^k\right)\right\} \tag{24}$$

where (x, δ) is the approximate solution of RP obtained by using the method presented in the last two subsections.

Given the Lagrange multiplier vector λ^k and the approximate solution (x^k,δ^k) of the relaxed problem at the kth iteration, the SSG method updates the vector according to

$$\lambda^{k+1}=\lambda^k+s^k\tilde{g}^k \tag{25}$$

where $\tilde{g}^k$ is the surrogate subgradient at the kth iteration, given by

$$\tilde{g}^k = \tilde{g}\left(x^k, \delta^k\right) = \left\{ x_{it}^k \left(1 - \delta_{it}^k\right) \right\} \tag{26}$$

with stepsize s^k chosen to satisfy

$$0 < s^k < \frac{\left(D^* - \tilde{L}^k\right)}{\left\|\tilde{g}^k\right\|} \tag{27}$$

where $D^* = \max_{\lambda} D(\lambda)$ is the optimal objective value of the dual problem *DP* and $\tilde{L}^k = L(\lambda^k, x^k, \delta^k)$ is the surrogate dual at the *k*th iteration.

The conditions for SSG to converge towards an optimal solution of the dual problem are:

1 at the initial multiplier vector λ^0, the solution $\{x^0, \delta^0\}$ of the relaxed problem satisfies

$$L\left(\lambda^0, x^0, \delta^0\right) < D^* \tag{28}$$

2 at each iteration *k* with multiplier vector λ^k ($k \geq 1$), the solution $\{x^k, \delta^k\}$ of the relaxed problem satisfies

$$L\left(\lambda^k, x^k, \delta^k\right) < L\left(\lambda^k, x^{k-1}, \delta^{k-1}\right) \tag{29}$$

where (x^{k-1}, δ^{k-1}) is the solution of the relaxed problem obtained at the last iteration $k-1$ with multiplier vector λ^{k-1}.

In our implementation of SSG, the multiplier vector λ is initiated at zero, that is, $\lambda^0 = 0$, then Condition 1) holds. Similar to what had been done in our previous work (Chen and Chu, 2003), we can demonstrate that $L(\lambda^k, x^k, \delta^k) \leq L(\lambda^k, x^{k-1}, \delta^{k-1})$. It implies that Condition 2) is almost satisfied. In a few exceptions, $L(\lambda^k, x^k, \delta^k) = L(\lambda^k, x^{k-1}, \delta^{k-1})$ may happen. In this case, RP^1 and RP^2 will be solved once again with the current solution (x^k, δ^k) as the starting point. This procedure repeats until $L(\lambda^k, x^k, \delta^k) < L(\lambda^k, x^{k-1}, \delta^{k-1})$ holds or the solution of RP^1 or RP^2 is unchanged in two consecutive iterations of this Gauss Seidel iteration-like procedure.

Since the surrogate dual is not a Lagrangian dual in strict sense, its value may exceed the optimal (minimum) objective value of the original problem. For this reason, step sizing is critical for ensuring a good performance of SSG. An adaptive step sizing scheme has been proved effective to the method. The scheme first estimates the optimal objective value of the original problem based on its surrogate dual rather than its best objective value obtained so far and then sets the step size s^k according to the following formula:

$$s^k = \frac{\left(\hat{D}^* - \tilde{L}^k\right)}{\left\|\tilde{g}^k\right\|^2} \tag{30}$$

where β is a parameter with $0 < \beta < 1$, $\hat{D}^* = (1 + \frac{\omega}{\theta^\rho}) \times \tilde{L}^{[k]}$ is an estimate of the optimal dual value D^* with $\tilde{L}^{[k]}$ being the best surrogate dual obtained prior to iteration *k*.

Parameters ω and ρ are chosen as $\omega \in [0.1, 1.0]$ and $\rho \in [1.1, 1.5]$, respectively. Parameter θ is adaptively adjusted with $\theta = \max(1, \theta - 1)$ if $\tilde{L}^k > \tilde{L}^{[k]}$, and $\theta = \theta + 1$ otherwise.

For the planning problem considered, in some cases, the SSG method may converge too early to obtain a near-optimal solution of the dual problem because of the approximate resolution of the relaxed problem. This will happen when the coupling Constraints (14) are all satisfied for the solution of the relaxed problem, leading to a null surrogate subgradient and the immobilisation of the Lagrange multiplier vector. To prevent the premature convergence, one strategy is to reformulate the Constraints (14) by the constraints:

$$x_{it}\left(1-\delta_{it}\right) \le 1-\varepsilon,\ i=1,\ldots,N,\ t=1,\ldots,T \tag{14}$$

where $0 < \varepsilon < 1$ is taken as a small positive number.

Constraints (14') are equivalent to Constraints (14), because:

1 the holding of constraints (14) implies the holding of constraints (14')

2 if constraints (14') hold, two cases may happen: $\delta_{it} = 1$ or $\delta_{it} = 0$. If $\delta_{it} = 1$, Constraints (14) hold. Otherwise, we have $x_{it} \le 1 - \varepsilon < 1$.

With the integral assumption of x_{it}, this implies that $x_{it} = 0$ and Constraints (14) also hold.

The LR approach described above can still be applied to the problem with the reformulation of Constraints (14'). For the reformulated problem, the SSG of the corresponding dual problem becomes $\tilde{g}(x,\delta) \equiv \{\tilde{g}_{it}(x,\delta)\} = \{x_{it}(1-\delta_{it}) - (1-\varepsilon)\}$ and the Lagrange multipliers of the problem take non-negative values.

3.4 Construction of feasible solutions and local search improvement

When an optimal solution (x, y) of subproblem RP^l is found at each iteration of the LR approach, a feasible solution (x, y, δ) to the original problem P can be obtained by setting each setup variable δ_{it} corresponding to positive x_{it} of the optimal solution of RP^l to 1 and all other setup variables to 0. We say the feasible solution (x, y, δ) of P is derived from the solution (x, y) of RP^l hereafter. This feasible solution can be further improved by a local search, which is performed in the neighbourhood obtained by changing at most two setup variables from the current solution. Benefiting from the polyhedral structure of the constraint set of the original problem, the local search can be implemented using a modified simplex algorithm, which drastically reduces the computational time. In the following, we present the basic idea and framework of the local search.

Suppose that at each iteration of the LR approach, an optimal solution of RP^l is found as (x, y) and the feasible solution of P derived from (x, y) is (x, y, δ). Consider the following non-linear programming problem derived from P by excluding all its setup variables $\{\delta_{it}\}$, their associated constraints and cost terms in its objective function:

$$NLP: \operatorname{Min} J_L = \sum_{i=1}^{M}\sum_{t=1}^{T} v_{it} + \sum_{i=M+1}^{N}\sum_{t=1}^{T} h_i y_{it} \tag{31}$$

s.t. (9)–(10), (12).

Since the constraint set of NLP is the same as that of RP^l, the solution (x, y), an extreme point (feasible basic solution) of RP^l, is also an extreme point (feasible basic solution) of NLP. For NLP, starting from the extreme point (x, y), the local search

attempts to find another extreme point (x', y') adjacent to (x, y) such that the feasible solution (x', y', δ') of P derived from (x', y') is better than the feasible solution (x, y, δ) of P derived from (x, y).

According to MP theory, each of such adjacent extreme points can be obtained from its current extreme point by performing a pivot operation that exchanges a basic variable and a non-basic variable. Note that the basic variable is determined by the non-basic variable. As soon as the non-basic variable to enter the basis is given, the basic variable to leave the basis is determined by the variable whose value is first driven to its lower bound or upper bound when the non-basic variable increases from its lower bound or decreases from its upper bound.

In the local search, all non-basic variables are examined. For each non-basic variable to enter the basis, its corresponding leaving basic variable is determined. Suppose that the current solution of NLP is (x, y) and the new solution of NLP obtained from (x, y) by a pivot operation that changes a non-basic variable to a basic variable is (x', y'). If the feasible solution (x', y', δ') of the original problem P derived from (x', y') is better than the feasible solution (x, y, δ) of P derived from (x, y), the pivot operation is performed and the current solution of NLP is updated to (x', y'). This procedure repeats until no such non-basic variable leading to an improved solution of P can be found.

For large problems, it may be time consuming for performing the above described local search because of the evaluation of the non-linear integral function J_L in each iteration. In this case, J_L can be approximated by the linear function $\sum_{i=1}^{N}\sum_{t=1}^{T} h_i y_{it}$.

4 Computational results

In this section, we evaluate the performance of our approach in terms of its solution quality and computation time. The algorithm of the approach is coded using C++ based on GNU Linear Programming Kit – version 4.4 (GNU, 2003). The GLPK package is an open source free software for solving large-scale linear programming and mixed integer programming. The choice of GLPK rather than a more powerful commercial LP/MIP solver such as CPLEX for the numerical testing of our approach is because GLPK provides the source codes of all its callable routines so that we can easily code the modified simplex algorithm for the local search improvement of the approach. First, we test our approach on a set of small-size randomly generated problems to see how good its solutions are in comparison with the optimal solutions of the problems. We then compare the approach with a time-truncated branch and bound algorithm on a set of medium-size randomly generated problems to demonstrate the applicability of our approach to real problems.

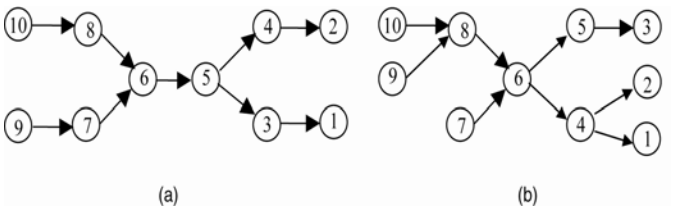
4.1 Small-size random instances

Two manufacturing systems with their BOM structures given by Figures 2 (a) and (b), respectively are considered in this testing. For each system, ten sets of instances with 10 problems for each set are generated.

For the system with the BOM given by Figure 2(a), referred to as system (a), 8 intermediate products/components and 2 final products are produced by 3 facilities. Manufacturing stages represented by nodes 7 to 10 share the capacity of facility 1.

The assembly stage and the post-processing stage represented by nodes 5 and 6, respectively are performed in facility 2. The stages represented by the nodes 1 to 4 are differentiation stages for the two final products and share the capacity of facility 3. The manufacturing/assembly lead time for each stage L_i (i = 1, 2, …, 8) is simply assumed to be zero. The planning horizon is taken as 6 periods (months).

Figure 2 BOM structure for small problems



The demand of each product follows a seasonal pattern or a geometric trend. In our study, the demand of product i (i = 1, 2) in period t is given by

$$d_{it} = \left\{1 + A_i \sin\left(\frac{2t\pi}{6}\right)\right\} N\left(\mu_i, \sigma_i^2\right) \tag{32}$$

for the seasonal case, and given by

$$d_{it} = r_i^t N\left(\mu_i, \sigma_i^2\right) \tag{33}$$

for the geometric trend pattern. The parameters in d_{it} are varied as follows:

$$\mu_1 = 100, \mu_2 = 200$$

$$\frac{\sigma_i}{\mu_i} = 0.1, 0.3$$

$$A_i = 0.0, 0.25, 0.50 \,(\text{for seasonal demand pattern})$$

r_i = 1.016 (10% growth per six periods) and 0.9826 (10% decline per six periods).

The service level α_{it} for each final product in each period is taken as 95%. The demand parameters for the ten sets of problems are listed in Table 1.

Table 1 Demand patterns for small problems

Problem set No.	*Demand pattern*	A_i *(or* r_i*)*	σ_i/μ_i
1	Seasonal	0.00	0.1
2	Seasonal	0.25	0.1
3	Seasonal	0.50	0.1
4	Seasonal	0.00	0.3
5	Seasonal	0.25	0.3
6	Seasonal	0.50	0.3
7	Trend	1.016	0.1
8	Trend	0.9826	0.1
9	Trend	1.016	0.3
10	Trend	0.9826	0.3

Other data of the problems are randomly generated in the following way. For each item, its inventory holding cost rate and setup cost are assumed to be constant over time. They are randomly generated from uniform distribution U[1, 10] and U[500, 1500], respectively. The unit resource consumption p_{ik} is randomly generated from uniform distribution U[1, 5]. The facility capacity C_{kt} is set based on the average workload per period wl_k for each facility k, which is calculated based on final product demands, the BOM, and p_{ik}, with a resource utilisation rate ur randomly generated from uniform distribution U[0.7, 0.9]. That is, $C_{kt} = wl_k/ur$ for any facility k and any period t. The initial inventory $I_{i,0}$ of each stage i (i = 1, 2, …, 10) is set to be its average lead time demand plus a safety stock with z-value 1.65 corresponding to the service level 95%.

Each problem contains 120 variables and 150 constraints where 60 variables are binary. We compare the solutions found by our algorithm with the optimal solutions obtained by an algorithm that combines a branch and bound algorithm of GNU LPK 4.4 for mixed integer programming and the partial linearisation introduced in Section 3.2 (the algorithm is simply referred to as branch and bound algorithm hereafter) for each set of problems. The parameters β, ω, ρ of our algorithm are taken as 0.9, 0.6 and 1.2 for all problems and the algorithm is terminated after 100 iterations. The computational results are given in Table 2, where the 2nd to 4th columns give, respectively the mean percentage deviation, the minimum percentage deviation, and the maximum percentage deviation of the solutions of our algorithm compared with the optimal solutions for each set. Each percentage deviation is evaluated based on the average cost obtained by our algorithm and the average optimal cost obtained by the branch and bound algorithm for each set of problems. The 5th column indicates the number of problems for which our algorithm finds an optimal solution among 10 problems of each set. CPU_{LR} and CPU_{OPT} are the average computation time (in sec) of our algorithm and the average computation time (in sec) of the branch and bound algorithm, respectively.

From the table, we can see that our algorithm can find a near-optimal solution with the average deviation 0.43 to 1.31% of the optimal solution for each set of the problems with the average computation time less than 1% of that of the branch and bound algorithm.

Table 2 Computational results for small problems (a)

Prob. set No.	*Mean dev. %*	*Min dev. %*	*Max dev. %*	*Num. opt. sol.*	CPU_{LR}	CPU_{OPT}
1	1.04	0	3.65	2	3.51	664.77
2	0.76	0	1.43	2	3.82	1143.90
3	0.79	0	4.54	5	3.79	495.78
4	1.31	0	2.70	2	1.62	354.52
5	0.98	0	2.66	2	2.29	809.19
6	0.62	0	1.72	3	1.78	438.45
7	1.06	0	4.55	4	4.55	885.23
8	0.96	0	2.16	2	3.59	416.37
9	0.96	0	2.25	2	1.92	474.22
10	0.43	0	1.65	3	1.82	474.22

For the system with the BOM given by Figure 2(b), referred to as system (b), 7 intermediate products/components and 3 final products are produced by 6 facilities. Manufacturing stages represented by nodes 9 and 10 share the capacity of facility 1, whereas the stage represented by node 7 is realised by facility 2. The assembly stages represented by nodes 8 and 6 are performed in facility 3. The differentiation stages represented by the nodes 4 and 5 share the capacity of facility 4, whereas the stages represented by nodes 1 and 2 share the capacity of facility 5. Finally, the post-processing stage represented by node 3 is realised by facility 6. The demand of each product also follows a seasonal pattern or a geometric trend with the demand of product i (i = 1, 2, 3) in period t is given by (32) and (33), respectively, where the parameters $\mu(i = 1, 2, 3)$ are taken as $\mu_1 = 100$, $\mu_2 = 200$ and $\mu_3 = 300$. All other parameters of the testing problems for the system are generated in the same way as for system (a). The computational results, given in Table 3, are similar to what obtained for system (a), except that the computation time of the branch and bound algorithm is much shorter than that for system (a), although the time is still much longer than that of the LR approach.

Table 3 Computational results for small problems (b)

Prob. set no.	*Mean dev. %*	*Min dev. %*	*Max dev. %*	*Num. opt. sol.*	CPU_{LR}	CPU_{OPT}
1	0.3	0	1.6	7	3.12	33.24
2	0.4	0	3.25	8	3.03	21.85
3	0.7	0	4	7	2.87	26.42
4	0.26	0	1.9	7	2.76	32.73
5	0.19	0	1.4	8	2.23	22.05
6	0.9	0	2.2	5	1.85	11.02
7	0.7	0	5.8	7	2.69	34.01
8	0.65	0	3.3	6	2.61	79.44
9	0	0	0	10	2.50	7.56
10	0.57	0	4.8	8	2.54	23.17

As explained before, the non-linearity in the objective function of our deterministic equivalent model P comes from no contribution of negative inventories. In order to evaluate the impact of the non-linearity on the solution quality of the planning problem considered, our LR approach is also applied to the approximate model of the problem with a linear cost function given by $\bar{J} = \sum_{i=1}^{N}\sum_{t=1}^{T} h_i y_{it} + \sum_{i=1}^{N}\sum_{t=1}^{T} \mathrm{cs}_i \delta_{it}$ and the constraints given by (9a)–(13). The total expected cost $\bar{J}$ of the approximate model, the total expected cost J of model P and the total expected cost $\hat{J}$ of model P at the solution of the approximate model, all obtained by the LR approach, are compared in Table 4, where the second row gives the results for the 100 instances of system (a) and the third row gives the results for the 100 instances of system (b). In the table, Mean Dev. %, Min Dev. % and Max Dev. % represents the mean percentage deviation, the minimum percentage deviation, and the maximum percentage deviation over the 100 instances, respectively.

Table 4 Comparison of model *P* with its linear approximate model

Prob. set no	*Mean. Dev. .% of* $(\hat{J}-\overline{J})/\hat{J}$	*Min dev. % of* $(\hat{J}-\overline{J})/\hat{J}$	*Max dev. % of* $(\hat{J}-\overline{J})/\hat{J}$	*Mean dev. % of* $(\hat{J}-J)/J$	*Min dev. % of* $(\hat{J}-J)/J$	*Max dev. % of* $(\hat{J}-J)/J$
(a)	13.14	3.29	28.80	1.70	–1.71	11.22
(b)	39.47	11.24	73.58	0.63	–0.88	16.77

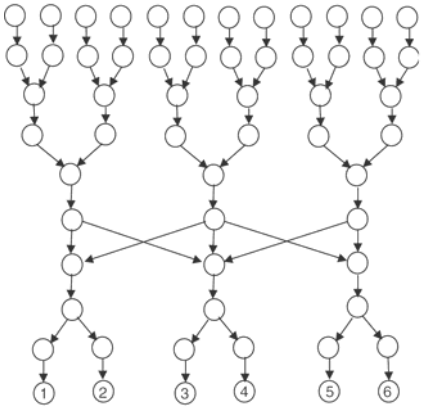
From the table, we can see the difference between the total expected cost of the approximate model that employs the linear function with the one with the non-linear cost function, both obtained by the LR approach may be quite large, although, on the average, applying the LR approach to the approximate model can obtain a solution close to what we obtain by applying the approach to the non-linear model. The direct application of the LR approach to the non-linear model is necessary if the computation time is permitted because in some cases, it can generate a solution 16.77% better than the solution obtained based on the linear approximate model in terms of the total expected cost. The necessity of using a non-linear inventory cost function in such a model was also addressed in Martel et al. (1995). Note that the minimum percentage of $(\hat{J}-J)/J$ may be negative because the LR approach can only find suboptimal solutions for both models.

4.2 Medium-size random instances

The instances are generated from a manufacturing system with its BOM structure shown in Figure 3. In the system, 54 intermediate products and 6 final products are produced by 10 facilities. The BOM has 10 levels, each level corresponds to a facility which produces all intermediate or final products at the level. The manufacturing/assembly lead time for each stage L_i (i = 1, 2, …, 8) is also assumed to be zero. The planning horizon is taken as 12 periods (months).

Similarly, 10 sets of instances with 10 problems for each set are randomly generated with the demand pattern and the parameters of each set identical to those given in Table 1, except that μ_i is simply set to 100 for all final products, i = 1, 2, …, 6. The service level α_{it} for each final product in each period is also set to 95%.

Figure 3 BOM structure for medium problems



Other data of the medium-size problems are randomly generated in the same way as for the small problems except for the setting of resource utilisation rate *ur* that affects the setting of facility capacities. The rate *ur* is now randomly generated from uniform distribution U[0.5, 0.7]. The decrease of the resource utilisation rate is only to ensure that the randomly generated problems with more BOM levels have feasible solutions.

Each problem of the ten sets contains 1440 variables and 2040 constraints where 720 variables are binary. For all the problem sets, we compare the performance of our algorithm with a time-truncated version of the branch and bound algorithm mentioned above for the small problems. The time-truncated algorithm is terminated after 1 hr (3600 sec) of execution. The parameters and the iteration number of our algorithm are taken the same as those for the small problems. The computational results are given in Table 5, where the 2nd to 4th columns give respectively the mean percentage deviation, the minimum percentage deviation, and the maximum percentage deviation of the solutions of our algorithm compared with the solutions of the time-truncated algorithm for each set. A negative value in any entity (cell) of the columns implies that our algorithm finds a better solution on average. The 5th column indicates the number of problems for which our algorithm finds a better solution among 10 problems of each set. CPU_{LR} is the average computation time (in sec) of our algorithm.

Table 5 Computational results for medium problems

Prob. set no.	*Mean dev. %*	*Min dev. %*	*Max dev. %*	*Num. better sol.*	CPU_{LR}
1	–7.23	–15.12	–1.90	10	296.06
2	–5.69	–18.81	1.03	9	281.71
3	–7.36	–11.97	–2.52	10	272.07
4	–8.72	–17.45	–0.38	10	262.94
5	–4.78	–9.72	–0.59	10	246.94
6	–6.12	–11.27	–0.71	10	240.06
7	–16.44	–23.03	–6.42	10	321.50
8	–11.88	–20.95	0.03	9	309.50
9	–10.12	–23.70	–1.71	10	266.44
10	–4.79	–9.88	–0.92	10	269.89

From the table, we can see that our algorithm can find a better solution for 98 over 100 problems with the average improvement 4.78–16.44% of the solution obtained by the time-truncated branch and bound algorithm for each set of the problems with the average computation time less than 10% of that of the time-truncated algorithm. The results show that our approach can solve production planning problems of realistic sizes and is promising for for industrial applications.

5 Conclusion

In this paper, we have developed a model and a solution methodology for a production planning problem with stochastic demands. The model, which is a mixed integer

programme with non-linear objective function, is established based on deterministic equivalence, whereas the solution methodology is based on LR, partial linearisation, and local search. By enhancing the LR approach with an efficient local search method implemented using a modified simplex algorithm, our approach can find high quality near-optimal solutions in short time for medium-size randomly generated instances and is thus promising for practical application. Further research following this work will consider setup times in the planning model and develop its effective solution methods.

References

Albritton, M., Shapiro, A. and Spearman, M. (2000) 'Finite capacity production planning with random demand and limited information', *Stochastic Programming E-Print Series*, Available at: *http://www.speps.info*.

Billington, P.J., McClain, J.O. and Thomas, L.J. (1986) 'Heuristics for multilevel lot-sizing with a bottleneck', *Management Science*, Vol. 32, No. 8, pp.989–1006.

Chen, H. and Chu, C. (2003) 'A lagrangian relaxation approach for supply chain planning with order/setup costs and capacity constraints', *Journal of Systems Science and Systems Engineering*, Vol. 12, No. 1, pp.98–110.

Drexl, A. and Kimms, A. (1997) 'Lot sizing and scheduling –survey and extensions', *European Journal of Operational Research*, Vol. 99, pp.221–235.

GNU (2003)'GLPK: GNU linear programming kit', Available at: *http://www.gnu.org /software/ glpk/glpk.html*.

Grubbstrom, R.W. and Wang, Z. (2003) 'A stochastic model of multi-level/multi-stage capacity-constrained production–inventory systems', *Int. J. Production Economics*, Vols. 81–82, pp.483–494

Gupta, A. and Maranas, C.D. (2003) 'Managing demand uncertainty in supply chain planning', *Computers and Chemical Engineering*, Vol. 27, pp.1219-1227.

Gupta, A., Maranas, C.D. and McDonald, C.M. (2000) 'Mid-term supply chain planning under demand uncertainty: customer demand satisfaction and inventory management', *Computers and Chemical Engineering*, Vol. 24, pp.2613–2621.

Harrison, T.P. and Lewis, H.S. (1995) 'Lot sizing in serial assembly systems with multiple constrained resources', *Management Science*, Vol. 41,No. 11.

Katok, E., Lewis, H.S. and Harrison, T.P. (1998) 'Lot sizing in general assembly systems with setup costs, setup times and multiple constrained resources', *Management Science*, Vol. 44, No. 6.

Martel, A. (2003) 'Planning policies for multi-echelon supply systems with probabilistic time-varying demands', *INFOR*, Vol. 41, No. 1, pp.71–91.

Martel, A., Diaby, M. and Boctor, F. (1995) 'Multiple items procurement under stochastic nonstationary demands', *European Journal of Operational Research*, Vol. 87, pp.74–92.

Patriksson, M. (1993) 'Partial linearization methods in nonlinear programming', *Journal of Optimization Theory and Applications*, Vol. 78, pp.227–246.

Salomon, M. (1991) 'Determining lotsizing models for production planning', *Lecture Notes in Economics and Mathematical Systems*, Vol. 355, Springer Verlag, Heidelberg.

Sox, C.R. (1997) 'Dynamic lot sizing with random demand and non-stationary costs', *Operations Research Letters* 20, pp.155–164.

Sox, C.R., Jackson, P.L., Bowman, A. and Muckstadt, J.A. (1999) 'A review of the stochastic lot scheduling problem', *Int. J. Production Economics*, Vol. 62, pp.181–200.

Tarim, S.A. and Kingsman, B.G. (2004) 'The stochastic dynamic production/inventory lot-sizing problem with service-level constraints', *Int. J. Production Economics*, Vol. 88, pp.105–119.

Tempelmeier, H. and Derstroff, M. (1996) 'A lagrangean-based heuristics for dynamic multi-level multi-item constrained lotsizing with setup times', *Management Science*, Vol. 42, pp.738–757.

Tempelmeier, H. and Helber, S. (1994) 'A heuristic for dynamic multi-item multi-level capacitated lotsizing for general product structure', *European Journal of Operational Research*, Vol. 75, pp.296–311.

Zhao, X., Luh, P.B. and Wang, J. (1999) 'The surrogate gradient algorithm for lagrangian relaxation method', *Journal of Optimization Theory and Applications*, Vol. 100, No. 3, pp.699–712.

Bi-criteria scheduling of a flowshop manufacturing cell with sequence dependent setup times

S. Hamed Hendizadeh, Tarek Y. ElMekkawy* and G. Gary Wang

Department of Mechanical and Manufacturing Engineering,
University of Manitoba,
Winnipeg, MB, Canada R3T 5V6
Fax: +204-275-7507
E-mail: umhendiz@cc.umanitoba.ca
E-mail: tmekkawy@cc.umanitoba.ca
E-mail: gary_wang@umanitoba.ca
*Corresponding author

Abstract: The paper considers a flowshop scheduling problem of a manufacturing cell that contains families of jobs whose setup times are dependent on the manufacturing sequence of the families. Two objectives, namely the makespan and total flow time, have been considered simultaneously in this work. Since minimisation of each of these two objectives is an Np-Hard problem, a Multi-Objective Genetic Algorithm (MOGA) has been proposed to deal with the bi-criteria optimisation problem. The performance of the proposed MOGA is compared with the makespan and total flow time lower bounds. The proposed MOGA obtained solutions that only deviate by an average of 1% from the lower bounds. Future research will develop more efficient lower bounds for the total flow time and also compare the proposed method with other multiobjective meta-heuristics.
[Received on 6 February 2007; Revised 6 June 2007; Accepted 16 June 2007]

Keywords: bi-criteria scheduling; cellular manufacturing; sequence-dependent setups; flowshop; pareto-optimal frontier; multi-objective genetic algorithm; MOGA; makespan; total flow time; lower bound; branch and bound; B&B.

Reference to this paper should be made as follows: Hendizadeh, S.H., ElMekkawy, T.Y. and Wang, G.G. (2007) 'Bi-criteria scheduling of a flowshop manufacturing cell with sequence dependent setup times', *European J. Industrial Engineering*, Vol. 1, No. 4, pp.391–413.

Biographical notes: S. Hamed Hendizadeh is currently a Master of Science student in the Department of Mechanical and Manufacturing Engineering, University of Manitoba, Canada. He received a BSc from Sharif University of Technology, Tehran, Iran in 2003 and worked in the field of scheduling optimisation in industry. His current research is in the field of scheduling optimisation by meta-heuristics and exact algorithms. He has also published a paper in the *International Journal of Production Economics*.

Tarek Y. ElMekkawy received a BSc and an MSc from the Department of Mechanical Design and Production, Cairo University, Egypt in 1990 and 1994, respectively. He received a PhD from the Department of Industrial and Manufacturing Engineering, University of Windsor, Canada, in 2001. He has joined the University of Manitoba (UM) in 2003 after working in the Canadian

Automotive Industry between 2001 and 2003. His research interest is in the area of scheduling optimisation. He has published many papers in international journals such as *IJPR*, *IJCIM*, *IJAMT*, *CIRP Annals* and *IJOR*.

G. Gary Wang joined the University of Manitoba (UM) in 1999 right after receiving a PhD in Mechanical Engineering from the University of Victoria. He has been active in research on design optimisation, design for manufacturing, advanced manufacturing and rapid prototyping. He is the recipient of the 2005 National I. W. Smith award for creative engineering from Canadian Society of Mechanical Engineers (CSME), as well as the 2007 Rh Award from UM for outstanding research contribution in the Applied Science category.

1 Introduction

Batch manufacturing accounts for 60 to 80% of all manufacturing activities in the world. The high level of variety and the small lot sizes of products have been major difficulties in this type of manufacturing. Cellular manufacturing addresses some of the problems and helps to gain economic advantage of batch manufacturing. Cellular manufacturing helps companies to build a variety of products for their customers with as little waste as possible. Here waste refers to elements of a manufacturing process that add cost instead of value to the product, such as scrap, rework, stock-out, overproduction, unnecessary human and material movement and excessive raw material. Cellular manufacturing is an application of group technology, which can be defined as a management philosophy that attempts to group products with similar design or manufacturing characteristics. In a cellular manufacturing environment, machines are grouped into cells. Each cell is dedicated to the production of a specific part family. A cell consists of people and machines or workstations, with the machines arranged in the processing sequence.

A cell based flowshop scheduling problem is considered in this paper. A regular flowshop problem consists of two main elements, which are a group of M machines and a set of N jobs to be processed on this group of machines. These N jobs have the same processing sequence on the M machines. Each job can be processed only on one machine at a time and only once on each machine. Moreover, each machine can process only one job at a time. In a cellular flowshop problem, the set of N jobs are grouped into different families according to their similar attributes or production techniques. The sequence of families and the sequence of jobs within each family are the same on all machines. The objective of such flowshop scheduling problem is to schedule the families and jobs within each family to minimise some performance criteria such as the makespan, total flow time, mean flow time or total tardiness/earliness.

The makespan in a flowshop problem can be defined as the completion time, which is the total time of processing all jobs. The makespan criterion is usually used to weigh the utilisations of machines. As long as the makespan is minimised, the production efficiency can be improved.

The flow time refers to the time that a part spends from its entrance to the system to completion of all the operations. The total flow time is the sum of flow times of all the parts in the system. By minimising the total flow time, the work-in-process inventory can be reduced.

Minimising either of the makespan or total flow time in flowshop manufacturing cell problems is time consuming as they are Np-hard problems. Therefore, a Multi-Objective Genetic Algorithm (MOGA) has been developed in this paper which minimises both criteria in a reasonable time.

The rest of this paper is organised as follows: Section 2 gives a comprehensive literature review of the related problems. Section 3 describes the considered problem and illustrates it by a simple example. Section 4 describes the lower bound for total flow time. The steps of proposed MOGA and the computational results have been explained in Sections 5 and 6. Finally, Section 7 describes future avenues for solving this problem.

2 Literature review

Many researchers have considered flowshop problems with multiobjectives. Nagar et al. (1995) proposed a Branch and Bound (B&B) approach to solve the two-machine flowshop problem with objectives of minimising makespan and total flow time. Murata et al. (1996) proposed the MOGA to solve scheduling problems with makespan and total tardiness as a bi-objective problem and with makespan, total flow time and total tardiness as a tri-objective problem. Jin et al. (2001) investigated the problem of multiobjective evolution strategies by adapting weighted aggregation.

Allahverdi (2004) considered the scheduling of flowshop problem with the objectives of minimising a weighed sum of makespan and maximum tardiness. Varadharajan and Rajendran (2005) developed a multiobjective simulated annealing algorithm with the objectives of minimising the makespan and the total flow time of jobs.

One of the important factors that need to be discussed in flowshop scheduling problems is the setup time, which can be defined as the time required to shift from one job to another on a given machine. For instance, consider a painting department where parts are grouped according to their colour. The setup time to change from yellow to green is the same as the setup time to change from green to yellow when the problem is sequence independent. On the contrary, the problem is sequence dependent if the required setup time of switching from yellow to green is shorter than switching from green to yellow. In many real life flowshop problems, the setup times of jobs are sequence dependent such as the Printed Circuit Board (PCB) manufacturing environment.

In a cellular manufacturing flowshop, since jobs are assigned to families based on tooling and setup requirements, usually a negligible or minor setup is needed to change from one job to another within a family and hence can be included in the processing times of each job. However, a major setup is needed to change processing from one family to another.

Considering sequence independent setup times of the cellular manufacturing flowshop problems, Skorin-Kapov and Vakharia (1993) developed a Tabu Search (TS) approach to minimise the makespan in a flowshop that outperformed an existing simulated annealing approach proposed by Vakharia and Chang (1990). Schaller (2001) developed a new lower bound for the flow shop group scheduling problem with a makespan criterion. Allahverdy et al. (in press) showed that most prior research on manufacturing cell scheduling assumed sequence independent setup times. Therefore it is worthy to pay more attention to problems with sequence dependent setup times.

Some cellular manufacturing flowshop problems with sequence dependent setup times have been considered as well. Schaller et al. (2000) solved a flowline manufacturing cell scheduling problem with sequence dependent family setup times and developed 11 new heuristics to minimise the makespan. Franca et al. (2005) considered the problem of scheduling a flowshop manufacturing cell with sequence dependent family setup times and developed an evolutionary algorithm to find minimum makespan permutation schedules. The heuristic algorithms Memetic Algorithm (MA), Genetic Algorithm (GA) and multistart strategy were implemented. Hendizadeh et al. (2007) also considered the same problem and developed a TS method and hybrid algorithms of TS and Simulated Annealing (TS/SA). The results of both MA and TS/SA algorithms showed that they were very effective in minimising the makespan. Gupta and Schaller (2006) considered the problem of scheduling a flowshop manufacturing cell with sequence independent family setup times and developed a B&B algorithm and several heuristic algorithms to find permutation schedules with minimum total flow time for medium size problems. The B&B algorithm was found to be able to solve small problems quickly. The proposed heuristic algorithms consistently generated solutions that were better than those solutions generated by the GA developed by Sridhar and Rajendran (1996).

Although some research has been done to optimise families of sequence-dependent setup times, they are mainly concerned with one criterion. This research attempts to address this problem with two criteria and a MOGA is then developed to solve such a problem.

3 Problem description

In this paper, the cellular manufacturing flowshop scheduling problem is optimised with the objectives of minimising both the makespan and total flow time. Sequence dependent setup times are considered in this problem and a MOGA is used to solve this problem. Several assumptions need to be clarified before the problem is described.

In this flowshop manufacturing system, the layout of machines has been set and the processing sequences are the same for all the jobs of each family and there is no preemption. The families of jobs have been identified-based on their attributes or operation requirements. All of the jobs and families are processed with the same order on each machine for this permutation flowshop manufacturing system. As stated before, since the setup times of individual jobs are negligible, they are considered to be included in the processing times of jobs in each family. However, the setup times for changeover are required between families and these setup times are sequence dependent.

For an M-machine, N-job cellular manufacturing flowshop scheduling problem, the number of families is F. N_f represents the number of jobs in each family $f = 1, 2,\ldots, F$. The starting time and processing time of each job j in family f on machine m can be expressed as $st_{f,j,m}$ and $p_{f,j,m}$, respectively. The setup time of family f on machine m is $se_{f,f',m}$ when family f' immediately precedes family f. It is evident that $f' = f$ if family f is the first family in the sequence. The starting time of the first job in the first family on the first machine is equal to the family setup time: $st_{1,1,1} = se_{1,1,1}$.

The starting time of the first job in the first family on machine $m = 2,\ldots, M$ is either the family setup time or the time that the operation of this job is done on the previous machine, depending on which event happens later than the other: $st_{1,1,m} = \max\{se_{1,1,m};$

$st_{1,1,m-1} + p_{1,1,m-1}\}$. The starting time of job j in the first family on the first machine only depends on the previous job: $st_{1,j,1} = st_{1,j-1,1} + p_{1,j-1,1}$. However, for $f = 1$; $j = 2,\ldots, N1$; $m = 2,\ldots, M$: $st_{1,j,m} = \max\{st_{1,j-1,m} + p_{1,j-1,m}; st_{1,j,m-1} + p_{1,j,m-1}\}$.

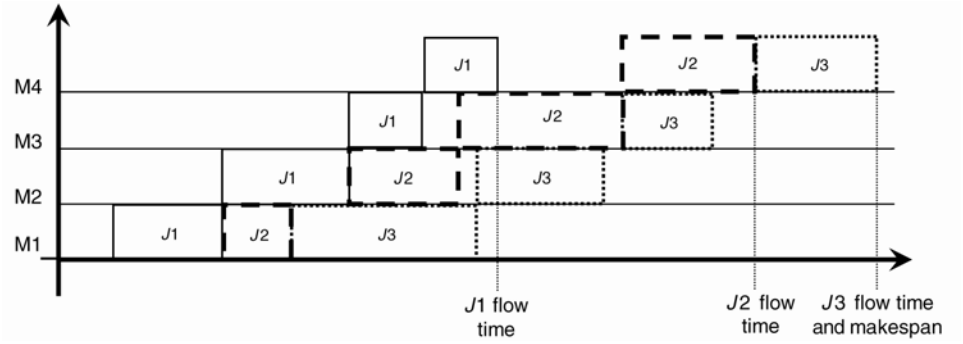
After processing the first family, additional setups are required. Therefore, for the first job in family $f = 2,\ldots, F$ on the first machine, the starting time can be shown as follows: $st_{f,1,1} = st_{f-1,N(f-1),1} + p_{f-1,N(f-1),1} + se_{f,f-1,1}$. The starting time of the job j in family $f = 2,\ldots, F$ on the first machine is: $st_{f,j,1} = st_{f,j-1,1} + p_{f,j-1,1}$.

Similarly, for the first job in family $f = 2,\ldots, F$ on machine $m = 2,\ldots, M$: $st_{f,1,m} = \max\{st_{f-1,N(f-1),m} + p_{f-1,N(f-1),m} + se_{f,f-1,m}; st_{f,1,m-1} + p_{f,1,m-1}\}$. Accordingly the starting time of job j in family f on the mth machine lies on the previous job in family f as well as the previous machine on which this job j is processed: $st_{f,j,m} = \max\{st_{f,j-1,m} + p_{f,j-1,m}; st_{f,j,m-1} + p_{f,j,m-1}\}$.

From the above recursive formula, the makespan can be obtained by adding the processing time of the last job (N) of the last family (F) on the last machine (M) to the starting time of this job ($st_{F,N,M}$). Likewise, the total flow time can be calculated by adding the completion time of all jobs. The completion time of each job is the sum of its starting time on the last machine ($st_{f,j,M}$) and its corresponding processing time on that machine.

The Figure 1 shows a permutation flowshop manufacturing system with four machines for only one family that has three jobs. The makespan and flow time of each job has been shown.

Figure 1 Permutation flowshop with four machines, one family and three jobs



Since all jobs have to be processed on machines one to four, respectively, this is a flowshop production system. Also since all machines have to process jobs one to three, respectively, this is a permutation flowshop production system. A job cannot be processed on a machine unless it has been done on the previous machine and also this current machine is free, not processing or being set up. The next family is added to each machine with above considerations and with a needed setup time between two different families.

4 A lower bound for total flow time

One of the strictest methods to verify the performance of the heuristic algorithms is to compare their solution with the optimal solution or a lower bound. An efficient lower

bound for makespan has been proposed in Schaller et al. (2000) and has been used in this paper. For the sake of total flow time, a lower bound has been proposed in Gupta and Schaller (2006) but it is for sequence independent problems. In this paper this lower bound has been modified to be applicable for sequence dependent problems.

To minimise the total flow time a B&B procedure is developed to obtain a family sequence and the sequences for jobs within families. The proposed B&B method modifies the procedure for scheduling families of jobs in a flowshop environment developed by Ham et al. (1985).

4.1 Branching strategy

To map the idea of jobs and family in the B&B tree, two types of nodes are introduced in the proposed procedure: family nodes and job nodes. The first level of search tree begins with branching from root on family nodes. Procedure works down the tree by branching on each family node to determine arrangement of the jobs within the family. The second level of tree includes job nodes. The tree goes down the levels to do branching on jobs within the family until a complete job schedule for the family is obtained. When all jobs of a family are scheduled, nodes are created for all remaining unscheduled families. This process will continue until all the jobs are scheduled.

To determine if branching should continue on a node, a lower bound for optimal total flow time is calculated. Then it is compared with an incumbent value which is the total flow time of the best feasible solution that has been found so far. If the obtained value for the lower bound is less than the incumbent value, branching on the remaining family or job nodes continues unless a complete solution was achieved. The lower bound is increased by coming down the tree. Therefore, if a node's lower bound is greater than the incumbent value, it is not promising and must be fathomed.

This B&B procedure is based on Depth-First Search (DFS). This search structure starts from the root and search as far as possible along each branch before back-tracking. Formally, DFS is known by expanding the first node of the tree that appears and thus going deeper and deeper until a complete solution is found, or it hits a node that is fathomed. Then the search starts back-tracking returning to the most recent level and looks for the next open node to branch. The criterion for selecting a node to branch is having the minimum value for the lower bound among a set of qualified nodes. The same procedure will pursue to reach a complete solution or fathomed node. Again, search back-tracks to the previous level until the whole tree is covered. As acquiring an incumbent value has crucial effect on fathoming open nodes, DFS was chosen to find the first incumbent value as soon as possible.

4.2 Lower bound

The lower bound estimates the sum of completion times in a flow shop to schedule families of jobs. Consider partial schedule $\rho = \rho(1),\ldots,\rho(k)$ of k jobs out of n with k belongs to the family r, let $\overline{\rho}$ be the subset of all jobs which are not scheduled in the ρ. Suppose that the family r has q job not scheduled in the ρ. Let the number of families not included in the ρ be ω. With the above definition, it is determined that the earliest time to start any job $i \in \overline{\rho}$ at machine j, EST(i, j) satisfies the following condition:

$$\mathrm{ETS}(i,j) \geq \max_{1\leq x\leq j}\left\{c(\rho(k),x)+\min_{y\in\bar{\rho}}\sum_{z=x}^{j-1}p_{yz}\right\} \tag{1}$$

For each family f, let p_{fj} denotes the effective processing time on machine j that is determined by the following formula:

$$P_f^j = s_f^j + \sum_{i=1}^{n_f} p_{ij} \tag{2}$$

where n_f is the number of jobs in family f. s_f^j is the minimum required setup time for family f on machine j with respect to the other families that can be processed before this family.

The proposed lower bound contains four major components. The first component considers the total flow time for the scheduled jobs. This is equal to the sum of completion time of jobs included in the partial schedule ρ.

$$\sum_{y\in\rho} C(y,j) \tag{3}$$

The second component is the earliest time that a job i can start processing on machine j which is represented by EST(i , j).

This condition specifies that to start a new job on the current machine, two conditions should be satisfied. Firstly, the previously scheduled jobs in the sequence should have been processed at the current machine. Secondly, the current job which is going to be processed should be ready. That is, the process of current job on machines prior to this machine must be finished.

$$(n-k)\max_{1\leq x\leq j}\left\{c(\rho(k),x)+\min_{y\in\bar{\rho}}\sum_{z=x}^{j-1}p_{yz}\right\} \tag{4}$$

The third component presents a lower bound for the flow time of unscheduled jobs including the unscheduled jobs in the family r and the jobs in the unscheduled families on machine j. This is obtained by the following three parts.

Part 1: schedule remaining unscheduled jobs in the family r.

Part 2: determine the arrangement of unscheduled families.

Part 3: calculate the total flow time of all jobs on the machine j.

Part 1 starts with sequencing of the unscheduled jobs in each family according to the Shortest Processing Time (SPT) rule. The SPT rule stipulates that the jobs with the SPT appear earlier in the sequence. The SPT rule provides the minimum total flow time for single machine problems.

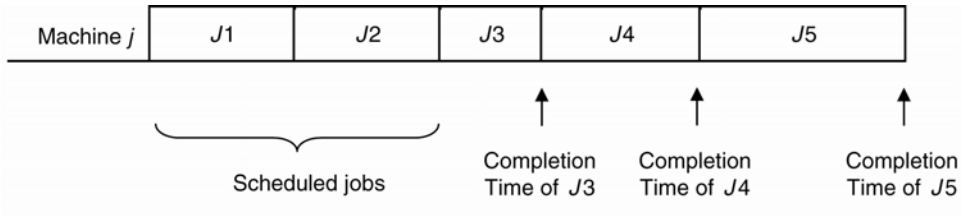
Part 2 concerns the sequencing of unscheduled families on machine j. Let $p_j(1)$, $p_j(2),\ldots, p_j(\omega)$ be the Shortest Weighted Effective Processing Time (SWEPT) schedule of all remaining families, that is, $P^j_{\pi_{j(1)j}} / n_{\pi_{j(1)}} \leq P^j_{\pi_{j(2)j}} / n_{\pi_{j(2)}} \leq \cdots \leq P^j_{\pi_{j(\omega)j}} / n_{\pi_{j(\omega)}}$.

Formula (5) shows the first part of component three. This part accounts for calculation of minimum total flow time for unscheduled jobs in the family r. Coefficient $(n-k-x+1)$, is the number of unconsidered jobs and P concerns the processing time of jobs which are arranged according to the SPT rule.

$$\sum_{x=1}^{q}(n-k-x+1)p_{\delta_j(x)j} \tag{5}$$

To make this part clearer, suppose that family r has five jobs and 2 of them have been scheduled. The remaining three jobs have been scheduled according to SPT rule. Figure 2 shows this sequence and completion time for each job.

Figure 2 Calculation of flow time for a sequence of jobs



Completion time of $J3$, $J4$ and $J5$ are $P3$, $P3 + P4$ and $P3 + P4 + P5$, respectively. Because the total flow time for these unconsidered three jobs is the sum of their completion times, the total flow time is: $3 \times P3 + 2 \times P4 + 1 \times P5$. The coefficient of the processing times, are the mentioned coefficient in (5). The same idea has been used in the part two and three of this component. Instead of processing times in (5), setup times of unscheduled families have been used in (6). Also, in (7) the processing times of (5) have been replaced by the processing times of the jobs within unscheduled families.

The second part of component three considers the setup times of unscheduled families which are scheduled by SWEPT. The coefficient shows the number of unconsidered jobs.

$$\sum_{y=1}^{n_{\pi_j(\alpha)}}\left\{(n-k-q)-\sum_{f=1}^{\alpha-1}n_{\pi_j(f)}+1\right\}s_{\pi_j(\alpha)} \tag{6}$$

The third part of component three concerns the total flow time of unscheduled jobs on machine j. Again, coefficient accounts for unconsidered jobs.

$$\sum_{\alpha=1}^{\omega}\left\{\sum_{y=1}^{n_{\pi_j(\alpha)}}\left\{(n-k-q)-\sum_{f=1}^{\alpha-1}n_{\pi_j(f)}-y+1\right\}p_{\omega\pi_j(\alpha)(y)j}\right\} \tag{7}$$

The fourth component of this lower bound concerns the processing time of unscheduled jobs on the rest of the machines (i.e. the machines that follow machine j).

$$\sum_{i\in\bar{\rho}}\sum_{x=j+1}^{m}P_{ix} \tag{8}$$

The summation of these four components is the lower bound for ρ on machine j ($\mathrm{LB}_j(\rho, j)$). The lower bound for the ρ is calculated according to the following formula:

$$\mathrm{LB}(\rho)=\max_{1\le j\le m}\mathrm{LB}_j(\rho,j) \tag{9}$$

Since the B&B is based on Depth First Search, it spends significant time on the last levels. Suppose that it has started at the first level from a family that is not the one in the optimal solution. There would be a huge gap between the acquired incumbent value from this branch and the optimal objective value. Moreover in most of the times the B&B can find the optimal solution pretty sooner than when it is stopped by itself, therefore a limit of CPU time can be useful to prevent spending significant time on computational experiments. Considering these two facts, the B&B is stopped when there is no improvement in incumbent value after an hour.

5 The proposed multi-objective genetic algorithm

According to the literature review, GA has been successfully applied for scheduling production systems when multiple objectives are considered simultaneously. For comprehensive details on the theory of multiobjective optimisation by means of GA, see Coello et al. (2002) and Deb (2001).

The proposed MOGA starts from a random initial *Elite Set* as starting points and then tries to evolve the *Elite Set* over successive generations. In each generation, at first the individuals are ranked according to an assigned fitness value which specifies the chance that an individual would be selected for the next generation. The genetic operations are done on selected individuals to generate new and diverse individuals. The *Elite Set* is updated after comparing with a new population of individuals. The pseudo code of the proposed MOGA is given in Figure 3. A few steps of the algorithm are discussed in more detail in the following subsections.

Figure 3 Pseudo code of the proposed MOGA

- Let time counter t=0;
- Initialise search parameters;
- Generate initial {Elite Set};
- **do**
  - Assign dummy fitness to individuals using non-dominated sorting and niching;
  - Select individuals for next generation;
  - Generate offspring using crossover and mutation;
  - Let current generation = new generation;
  - Identify non-dominated frontier of the current generation;
  - Update {Elite Set} by inclusion of non-dominated frontier of current generation;
- **while** ($t < t_{max}$)
- report result {Elite Set};

5.1 Non-dominated sorting

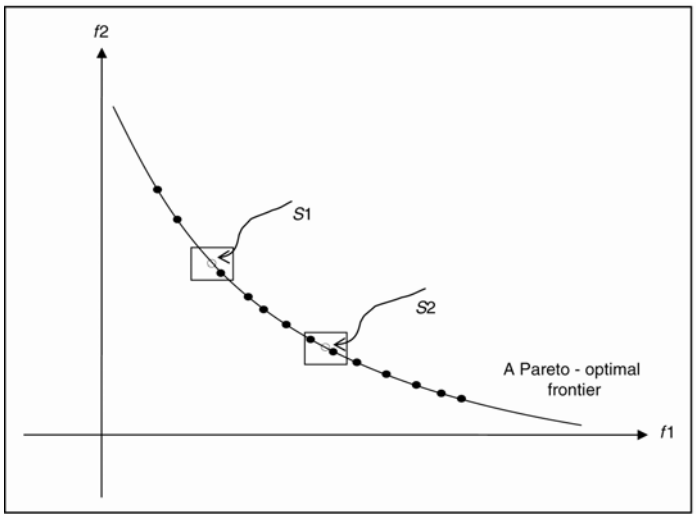
To assign appropriate fitness to individuals in a population when taking into account both objectives, the non-dominated sorting method proposed by Srinivas and Deb (1994) was selected. The non-dominated individuals present in the population are first identified from the current population. Then, all these individuals are assumed to constitute the first non-dominated frontier in the population and assigned a same large *dummy fitness value*. The same fitness value is to give an equal reproductive potential to all these non-dominated individuals. The individuals of the first non-dominated frontier are ignored temporarily to process the rest of the population in the same way to identify individuals for the second non-dominated frontier. These non-dominated solutions are then assigned a new dummy fitness value that is kept smaller than the minimum dummy fitness of the previous frontier. This process is continued until the entire population is classified into several frontiers.

To maintain diversity in the population, the first assigned dummy fitness value of these classified individuals is revised. By introducing the concept of niche cubicle proposed by Hyun et al. (1998), the first assigned fitness value of an individual is divided by a quantity proportional to the number of individuals exist in its niche. As Figure 4 shows, a niche cubicle for an individual is a rectangular region whose centre is the individual. This causes diversified Pareto-optimal solutions to coexist in the population. Dimensions of the niche cubicle in a problem having m objectives are computed as follows:

$$\sigma_{lg} = \frac{\mathrm{Max}_{lg} - \mathrm{Min}_{lg}}{\sqrt[m]{\mathrm{PopSize}}}, \quad l = 1, 2, \ldots, m \tag{10}$$

where Max_{lg} and Min_{lg} are the maximum and the minimum of the lth objective function at generation g. The niche size is calculated at every generation. A solution located in a less dense cubicle is allowed to have a higher probability to survive in the next generation. For an illustrative example for the non-dominated sorting with niching the readers may refer to Mansouri (2005).

Figure 4 Niche cubicles for solutions S1 and S2



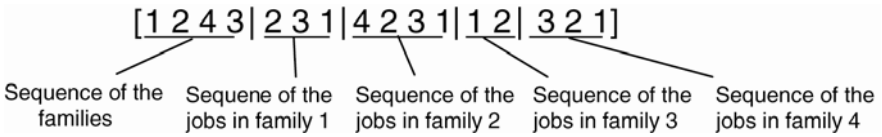
5.2 Selection

Individuals at each generation are selected according to the integer part of their associated *Fitness Value.* This is carried out using the so-called *remainder stochastic sampling without replacement* (Goldberg and Lingle, 1985). The remaining individuals are randomly selected from the {*Elite Set*}. For instance, consider a population of four individuals: 1, 2, 3 and 4 whose associated *fitness value* are: 2.6, 1.4, 0.9 and 0.7, respectively. According to the selection scheme, two copies of individual 1 and one copy of individual 2 will be selected. The third and the fourth individuals will be selected from the {*Elite Set*}.

5.3 Crossover

The crossover operator that has been used in this paper is a variant of the well known order crossover (OX) from Franca et al. (2005). To illustrate how it works, consider the following representation of the problem. The flowshop scheduling problem of this paper can be considered as a two-stage scheduling problem that the sequence of families must be determined at first and then the sequence of jobs within each family is determined. Thus the first part of the chromosome represents the sequence of families and the other parts represents the sequence of jobs in family number one and two, respectively. As an example suppose that there are four families and family number one has three jobs, family number two has four jobs, family number three has two jobs and family number four has three jobs. A random solution of this problem can be represented as (Figure 5).

Figure 5 A random solution of the flowshop scheduling problem



To construct a new offspring from two parents of the population, each pair of two respective parents are selected in such a way that the first parent is considered as parent one and the second parent is considered as parent two. A segment of each part of parent one is selected randomly and copied in the same position in the new offspring. The remainder positions are filled from the second parent with respect to the sequence of the member of each part. As an example suppose that there are two selected parents from the population (Figure 6).

Figure 6 A new offspring constructed from two parents

Parent1: [1 2 4 3 | 2 3 1 | 4 2 3 1 | 1 2 | 3 2 1]

Parent2: [3 4 2 1 | 1 3 2 | 1 3 4 2 | 2 1 | 3 1 2]

Offspring: [3 **2 4** 1 | 1 **3** 2 | **4 2** 1 3 | **1 2** | **3** 1 2]

The bold numbers represent the members that have been copied from the parent one and positioned in the same position that they had in parent one. The other members have

been selected from parent two according to their sequence. The crossover could transfer important characteristics of the parents into the new offspring.

5.4 Mutation

Mutation is an operator that generates offspring from a single parent by mutating part of it. It first chooses two random positions in each part of a parent and then swaps the members of these two positions. *Mutation* is implemented to all individuals for at least one time. The Times of Mutation (TM) is determined in a parameter setting procedure to do mutation for more than one time.

The offspring produced by *mutation* operators are then compared to its corresponding parent. The parent will be replaced by its offspring if the parent individual turned out to be dominated. However, a dominated offspring is still given a chance to replace its parent. The probability for accepting a dominated offspring, starting from 1.0, is decreased exponentially over the generations. The probability for accepting a dominated offspring resulted from the *mutation* operator at a given time t is denoted by $P(A)$ and determined using the following formula:

$$P(A) = \exp\left(-\frac{t}{t_{\max} - t}\right), \quad t = 0,\ldots,t_{\max} \tag{11}$$

where $t_{\max}$ is the maximum execution (CPU) time. The above formula is inspired by the annealing process of simulated annealing. The algorithm tries to be converged by accepting only new offspring that cannot be dominated by their parents, when the algorithm reaches to the maximum execution time.

5.5 Parameter setting

From the literature, it seems that parameter setting for metaheuristic algorithms presents one of the most difficult problems. In this work, comprehensive experiments have been done to determine the parameters' values of the proposed algorithm. The final parameter values are set as follows: population size of 30, crossover probability of 90%, TM of 2, and maximum CPU time of 60 sec.

6 Computational results

6.1 Comparison method

One of the most important issues in multiobjective problems is how to verify the obtained Pareto set. In the literature there are different kinds of quality indices. One of the most restricted quality indices is to compare each solution of the Pareto set with its corresponding lower bound or optimal solution. This kind of quality index has been used in Mansouri (2005) which tried to find the optimal solution for each of the considered objectives and simultaneously make them as close as possible to the optimal one. As the problem is a multiobjective problem, the objective functions must be conflicting, which means by making one of them better, the other one will get worse. Therefore, this kind of quality index can be used for the multiobjective problems which are not that much

conflicting. The problem that has been considered in Mansouri (2005) is one of these kinds and in many problems the true Pareto set has just one solution.

Consequently, some papers compared their Pareto set with the previous Pareto set that has been reported in the literature. Varadharajan and Rajendran (2005) have combined their Pareto set with the one that has been obtained from the literature and after deleting the dominated solutions, they have made this new Pareto set as a reference one. In this way, they have compared their Pareto set and the one from the literature with this reference Pareto set. The number of common solutions between the proposed algorithms and the reference set shows how promising are the proposed algorithms. This kind of quality index is not applicable for this study, because this problem has not been considered before and there is no existing Pareto set available from the literature.

One of the other kinds of quality index is using the Schaumann et al. (1998) method which is based on obtaining the true-pareto set by total enumeration. In this study, it has been tried to follow the same approach by performing total enumeration for some small problems. For example, a problem with 3 families and 3, 6 and 4 jobs within each family, which has 622,080 different solutions, it takes about 180 sec to find the true-pareto set. But suppose that instead of 3 jobs in family one, there are 8 jobs. This problem has 4,180,377,600 solutions and takes about 1,209,600 sec (14 days) to find the true-optimal solution. This dramatic increase in the number of solutions is the result of the combinatorial nature of these kinds of problems that made them hard to find the optimal solution even for single objective. These kinds of problems are Np-Hard problems and therefore many heuristics have been developed to find their near optimal solutions.

The other kind of quality index in the literature is to consider the best obtained objective value of Pareto set for each of the considered objectives and compare it to the lower bound or optimal objective value (Varadharajan and Rajendran, 2005). This kind of quality index has been used in this paper.

6.2 Comparative results

The proposed MOGA and B&B algorithm were coded in C++ and run on a Pentium IV processor of 3.0 GHz with 1.0 GB RAM. Table 1 shows the obtained makespan and total flow time for small test problems. These problems are exactly the same ones that have been used in Schaller et al. (2000) with at most 15 jobs. The corresponding lower bound for each of those problems has also been reported according to the proposed lower bound of the makespan from Schaller et al. (2000) and the lower bound of the total flow time that was presented in Section 4. The first column denotes the setup time of a family. Small setup time is a random number from $U[1, 10]$ probability distribution function, $U[1, 50]$ for medium and $U[1, 100]$ for large ones. The second column shows the problem size which includes the number of families (F), the number of machines (M) and the number of jobs in each family (J). As each problem has been run for ten times, the next six columns show the minimum, average and maximum obtained value of the makespan and total flow time by MOGA. It should also be mentioned that for each run, the best obtained makespan and total flow time of the pareto set have been reported. The next three columns show the lower bound of makespan, total flow time and the required CPU time to obtain the lower bound of total flow time. The deviation has been calculated according to the following formula:

$$\text{Deviation} = \frac{\text{ObjectiveFunction}_{\text{MOGA}} - \text{LowerBound}}{\text{LowerBound}} \times 100 \tag{12}$$

Table 1 MOGA performance for small size problems

Setup time	*Problem size*			*Makespan*			*Total flow time*			*Makespan lower bound*	*Total flow time lower bound*	*CPU time(sec)*	*Deviation for makespan (%)*	*Deviation for total flow time (%)*
	F	*M*	*J*	*Min*	*Ave*	*Max*	*Min*	*Ave*	*Max*					
Small	3	3	4	66	66.1	67	172	172.3	175	66	172	0	0.15	0.17
	3	3	9	86	86.0	86	484	485.0	486	86	484	3	0.00	0.21
	3	3	9	100	100.0	100	527	527.0	527	100	524	2	0.00	0.57
	3	3	5	75	75.0	75	249	249.0	249	75	249	0	0.00	0.00
	3	3	13	117	118.2	120	870	880.0	895	117	870	386	1.03	1.15
	3	3	12	124	124.0	124	826	827.2	829	124	826	1582	0.00	0.15
	3	3	15	125	125.0	125	1107	1107.0	1107	122	1107	3081	2.46	0.00
	3	3	15	125	125.2	127	1126	1128.0	1131	125	1196	5736	0.16	−5.69
	3	4	9	103	103.2	104	637	639.4	650	101	637	0	2.18	0.38
	3	4	15	151	151.0	151	1376	1377.1	1387	151	1376	4221	0.00	0.08
	3	4	13	133	133.6	135	1063	1063.6	1065	130	1063	167	2.77	0.06
	3	4	13	125	125.0	125	1044	1045.8	1047	123	1044	168	1.63	0.17
	3	4	9	116	116.4	118	674	674.9	683	115	674	0	1.22	0.13
	3	4	7	96	96.7	98	402	405.7	413	96	402	0	0.73	0.92
	3	4	14	128	128.0	128	1068	1068.5	1069	128	1068	4465	0.00	0.05
	3	4	14	120	120.0	120	1093	1093.0	1093	118	1093	4038	1.69	0.00
	3	4	15	140	140.5	145	1273	1279.6	1281	140	1491	3602	0.36	−14.18
	3	4	7	80	80.4	84	403	404.7	412	80	403	0	0.50	0.42

Table 1 MOGA performance for small size problems (continued)

Setup time	*Problem size*			*Makespan*			*Total flow time*			*Makespan lower bound*	*Total flow time lower bound*	*CPU time(sec)*	*Deviation for makespan (%)*	*Deviation for total flow time (%)*
	F	*M*	*J*	*Min*	*Ave*	*F*	*Min*	*Ave*	*Max*					
	3	4	14	131	131.1	132	1042	1042.4	1046	129	1042	2311	1.63	0.04
	3	4	12	111	111.0	111	760	760.0	760	109	760	48	1.83	0.00
	4	4	8	104	104.0	104	563	563.8	565	104	563	0	0.00	0.14
	4	4	14	138	138.0	138	1181	1181.0	1181	138	1181	2142	0.00	0.00
	4	4	13	132	132.0	132	961	961.0	961	132	961	2385	0.00	0.00
	5	5	15	154	154.0	154	1342	1343.8	1345	153	1342	1677	0.65	0.13
	5	5	15	151	151.2	152	1233	1235.6	1244	149	1233	3605	1.48	0.21
	5	6	14	157	157.0	157	1389	1389.2	1390	154	1389	3935	1.95	0.01
	6	5	15	167	167.0	167	1517	1521.8	1524	164	1517	5473	1.83	0.32
												Average	0.90	0.21
Medium	3	3	4	116	116.0	116	329	329.0	329	116	329	0	0.00	0.00
	3	3	9	150	150.0	150	882	882.6	885	149	882	3	0.67	0.07
	3	3	9	152	152.0	152	755	755.4	759	152	755	2	0.00	0.05
	3	3	5	117	117.0	117	405	406.0	410	114	405	0	2.63	0.25
	3	3	13	177	177.0	177	1073	1073.0	1073	177	1073	387	0.00	0.00
	3	3	12	200	200.0	200	1367	1367.4	1368	199	1367	1585	0.50	0.03
	3	3	15	161	161.0	161	1461	1461.0	1461	161	1461	3054	0.00	0.00
	3	3	15	164	164.0	164	1395	1395.4	1399	164	1551	6426	0.00	−10.03
	3	4	9	150	150.8	154	755	756.0	760	150	755	0	0.53	0.13
	3	4	15	211	211.0	211	1905	1907.4	1911	207	1905	6431	1.93	0.13

Table 1 MOGA performance for small size problems (continued)

Setup time	Problem size			Makespan			Total flow time			Makespan lower bound	Total flow time lower bound	CPU time(sec)	Deviation for makespan (%)	Deviation for total flow time (%)
	F	M	J	Min	Ave	F	Min	Ave	Max					
	3	4	13	192	192.0	192	1626	1627.6	1630	192	1626	169	0.00	0.10
	3	4	13	186	186.0	186	1560	1561.4	1574	186	1560	167	0.00	0.09
	3	4	9	178	178.0	178	1118	1118.6	1120	174	1118	0	2.30	0.05
	3	4	7	156	156.4	157	633	639.4	648	156	633	0	0.26	1.01
	3	4	14	179	179.0	179	1390	1390.0	1390	179	1390	7492	0.00	0.00
	3	4	14	187	187.0	187	1699	1699.0	1699	186	1699	4450	0.54	0.00
	3	4	15	172	172.0	172	1613	1616.0	1619	172	1913	6334	0.00	−15.53
	3	4	7	119	119.0	119	593	596.6	605	119	593	0	0.00	0.61
	3	4	14	199	199.0	199	1697	1697.4	1701	199	1697	2355	0.00	0.02
	3	4	12	161	161.0	161	1335	1336.0	1337	160	1335	48	0.63	0.07
	4	4	8	160	160.0	160	851	853.0	857	160	851	0	0.00	0.24
	4	4	14	196	196.0	196	1455	1455.0	1455	196	1455	2188	0.00	0.00
	4	4	13	201	201.0	201	1355	1355.0	1355	199	1355	2408	1.01	0.00
	5	5	15	233	233.0	233	1914	1914.6	1915	233	1914	1717	0.00	0.03
	5	5	15	211	211.0	211	1843	1844.2	1846	211	1843	4141	0.00	0.07
	5	6	14	254	254.0	254	2215	2216.2	2217	254	2215	3978	0.00	0.05
	6	5	15	254	254.6	256	2352	2358.8	2371	251	2361	4947	1.43	−0.09
												Average	0.46	0.12

Table 1 MOGA performance for small size problems (continued)

Setup time	*Problem size*			*Makespan*			*Total flow time*			*Makespan lower bound*	*Total flow time lower bound*	*CPU time(sec)*	*Deviation for makespan (%)*	*Deviation for total flow time (%)*
	F	*M*	*J*	*Min*	*Ave*	*F*	*Min*	*F*	*M*					
Large	3	3	4	162	162.0	162	439	439.0	439	162	439	0	0.00	0.00
	3	3	9	263	263.0	263	1467	1468.2	1469	263	1467	2	0.00	0.08
	3	3	9	257	257.0	257	1336	1336.4	1340	257	1336	3	0.00	0.03
	3	3	5	254	254.8	258	900	901.0	905	254	900	0	0.31	0.11
	3	3	13	278	278.0	278	2073	2073.0	2073	278	2073	390	0.00	0.00
	3	3	12	251	251.0	251	1844	1844.0	1844	251	1844	1588	0.00	0.00
	3	3	15	253	253.0	253	2477	2477.0	2477	253	2477	3095	0.00	0.00
	3	3	15	269	269.0	269	2385	2388.0	2391	266	2385	4179	1.13	0.13
	3	4	9	237	237.0	237	1318	1318.8	1320	236	1318	0	0.42	0.06
	3	4	15	290	290.0	290	2791	2792.3	2796	290	2923	3602	0.00	−14.47
	3	4	13	268	268.0	268	2053	2053.0	2053	266	2053	169	0.75	0.00
	3	4	13	263	263.0	263	1828	1829.0	1833	261	1828	170	0.77	0.05
	3	4	9	264	264.0	264	1503	1503.8	1505	264	1503	0	0.00	0.05
	3	4	7	251	251.0	251	1002	1002.7	1005	251	1002	0	0.00	0.07
	3	4	14	284	284.0	284	2089	2089.0	2089	284	2093	3603	0.00	−0.19
	3	4	14	291	291.0	291	2084	2084.0	2084	291	2084	6654	0.00	0.00
	3	4	15	286	286.4	287	2884	2891.1	2894	286	2884	4421	0.14	0.25
	3	4	7	202	202.0	202	988	990.3	996	201	988	0	0.50	0.23

Table 1 MOGA performance for small size problems (continued)

Setup time	*Problem size*			*Makespan*			*Total flow time*			*Makespan lower bound*	*Total flow time lower bound*	*CPU time(sec)*	*Deviation for makespan (%)*	*Deviation for total flow time (%)*
	F	*M*	*J*	*Min*	*Ave*	*F*	*Min*	*F*	*M*					
	3	4	14	309	309.0	309	2506	2507.4	2508	309	2506	2298	0.00	0.06
	3	4	12	249	249.0	249	1393	1393.0	1393	246	1393	48	1.22	0.00
	4	4	8	284	284.0	284	1351	1352.2	1354	284	1351	0	0.00	0.09
	4	4	14	323	323.0	323	2849	2849.0	2849	323	2849	2142	0.00	0.00
	4	4	13	331	331.0	331	2284	2284.0	2284	331	2284	2397	0.00	0.00
	5	5	15	395	395.0	395	3307	3308.0	3309	392	3307	1688	0.77	0.03
	5	5	15	372	372.6	375	3226	3229.6	3240	372	3640	4216	0.16	−11.27
	5	6	14	417	417.0	417	2655	2657.5	2660	412	2655	3992	1.21	0.09
	6	5	15	438	439.5	441	4003	4003.9	4012	436	4003	3743	0.80	0.02
												Average	0.30	0.06

The following points can be drawn from the Table 1:

- The exponential nature of the problem can be seen from the required CPU time for B&B procedure. That can be noticed by comparing the CPU time of a problem with 13 jobs and a problem with 15 jobs.

- Although the CPU time of B&B is in the matter of thousands, the MOGA could find a solution with a deviation of 0.13% in average from B&B in just 60 sec.

- According to the mentioned stopping criterion for B&B, for some cases it could not find a near optimal solution as its value is much larger than the one obtained by MOGA. The negative in the last columns shows these cases. Also because of this fact the maximum of 15 jobs was chosen as the largest problem in small size problems.

- The average of deviation has increased by changing the setup time from large to small. This fact also verifies the Gupta and Schaller (2006) claim that the smaller setup time in comparison with processing time the more complex the problem becomes.

- MOGA could find near optimal solution for both makespan and total flow time objective functions. This verifies Liu and Reeves (2001) claim that if it is difficult to solve a problem with the makespan objective function, it is also difficult to be solved with total flow time objective function. This means that if the proposed algorithm can find a near optimal solution with the first objective function, it is also likely to be capable of finding a near optimal solution with the second objective function.

- The distribution of jobs between the families affects the required CPU time for B&B procedure. Although two problems have the same number of jobs, but the one that has larger maximum number of jobs in the families is more time-consuming.

The average deviation over all problems is 0.54% for makespan and 0.13% for total flow time. Therefore, the proposed MOGA is more capable of finding near optimal solutions for total flow time rather than makespan. According to our knowledge, there is no optimisation method to solve large problems with total flow time objective function (Gupta and Schaller, 2006). Therefore, with keeping these two matters in mind, if the MOGA can find near optimal solution for large problems with makespan objective function, it would be also promising to find the near optimal solution with total flow time objective function.

Table 2 shows the performance of MOGA for large problems. The average deviation for the makespan objective function is 1.41% in average. This shows that the proposed MOGA is capable of finding near optimal solution even for large problems.

Table 2 OGA performance for large size problems

Setup time	*Problem*			*Makespan*			*Total flow time*			*Makespan lower bound*	*Deviation for makespan (%)*
	F	*M*	*J*	*Min*	*Ave.*	*Max.*	*Min*	*Ave.*	*Max.*		
Small	3	3	17	144	144.0	144	1428	1428.0	1428	144	0.00
	4	4	21	183	183.0	183	2200	2201.1	2203	180	1.67
	5	5	22	276	276.7	279	4802	4809.4	4816	271	2.10
	6	5	19	195	197.0	199	2201	2204.1	2208	192	2.60
	5	6	39	301	301.9	304	6535	6559.0	6595	301	0.30
	6	6	39	331	331.5	332	7139	7173.5	7220	321	3.27
	8	8	37	357	357.9	359	7675	7686.2	7697	349	2.55
	8	10	41	415	418.9	424	10,325	10395.6	10,474	405	3.43
	10	10	59	511	515.0	522	17,045	17135.3	17,253	491	4.89
Medium	3	3	17	207	207.3	208	2199	2199.0	2199	207	0.14
	4	4	21	270	270.0	270	3012	3014.9	3017	269	0.37
	5	5	22	355	355.0	355	6457	6464.4	6477	355	0.00
	6	5	19	287	287.0	287	2980	2983.4	2987	283	1.41
	5	6	39	378	378.0	378	8021	8041.3	8058	378	0.00
	6	6	39	417	417.3	418	9083	9110.8	9166	416	0.31
	8	8	37	467	472.7	481	9577	9595.9	9643	465	1.66
	8	10	41	564	569.1	573	13,060	13118.6	13,168	554	2.73
	10	10	59	654	664.8	676	21,492	21651.6	21,782	647	2.75

Table 2 OGA performance for large size problems (continued)

Setup time	*Problem*			*Makespan*			*Total flow time*			*Makespan lower bound*	*Deviation for makespan (%)*
	F	*M*	*J*	*Min*	*Ave.*	*Max.*	*Min*	*Ave.*	*Max.*		
Large	3	3	17	311	311.0	311	2975	2975.0	2975	311	0.00
	4	4	21	367	367.0	367	4611	4611.8	4614	366	0.27
	5	5	22	492	492.0	492	8405	8413.0	8423	492	0.00
	6	5	19	414	414.5	415	4952	4956.4	4963	410	1.10
	5	6	39	539	539.0	539	12,559	12591.2	12,641	539	0.00
	6	6	39	592	592.0	592	13,572	13583.2	13,592	584	1.37
	8	8	37	702	710.4	720	13,872	13889.6	13,933	696	2.07
	8	10	41	864	865.4	875	19,540	19564.2	19,579	843	2.66
	10	10	59	939	941.3	952	30,791	31094.0	31,414	937	0.46
										Average	1.41

7 Conclusion and future research

In this paper, a MOGA has been proposed for bi-criteria scheduling of a flowshop manufacturing cell with sequence dependent setup times. The makespan and total flow time have been considered as two conflicting objectives. Each of those objectives has been considered in the literature as a single objective problem and has been proved that they are Np-Hard problems. The performance of the proposed MOGA has been evaluated according to the makespan and total flow time lower bounds. It has been showed that the average overall deviation from the lower bounds is less than 1% for small problems and about 1% for large ones.

For future research, it is recommended to develop a more efficient lower bound for the total flow time objective function. In addition, the performance of the MOGA can be compared with other multiobjective meta-heuristics such as Multiobjective Simulated Annealing. According to the literature, simulated annealing is promising to find a near optimal solution in less CPU time than GA with similar quality of the obtained solution. There is also some room to make the proposed MOGA more efficient for example by using a structured population and also some other genetic operators.

References

Allahverdi, A. (2004) 'A new heuristic for m-machine flowshop-scheduling problem with bicriteria of makespan and maximum tardiness', *Computer and Operation Research*, Vol. 31, No. 2, pp.157–180.

Allahverdi, A., Ng, C.T., Cheng, T.C.E., and Kovalyov, M.Y. (in press) 'A survey of scheduling problems with setup times or costs', *European Journal of Operational Research*.

Coello, C.A.C., Van Veldhuizen, D.A. and Lamont, G.B. (2002) *Evolutionary Algorithms for Solving Multi-objective Problems*, New York: Kluwer.

Deb, K. (2001) *Multi-Objective Optimization using Evolutionary Algorithms*, Chichester: Wiley.

Franca, P.M., Gupta, J.N.D., Mendes, A.S., Moscato, P. and Veltink, K.J. (2005) 'Evolutionary algorithms for scheduling a flowshop manufacturing cell with sequence dependent family setups', *Computer and Industrial Engineering*, Vol. 48, No. 3, pp.1–16.

Goldberg, D.E. and Lingle, R. (1985) 'Alleles, loci, and the TSP', *Proceedings of the First International Conference on Genetic Algorithms*, pp.154–159.

Gupta, J.N.D. and Schaller, J. (2006) 'Minimizing flow time in a flow-line manufacturing cell with family setup times', *Journal of the Operational Research Society*, Vol. 57, No. 2, pp.163–176.

Ham, I., Hitomi, K. and Yoshida, T. (1985) *Group Technology*, Boston: Kluwer-Nijhoff Publishing.

Hendizadeh, S.H., Faramrzi, H., Mansouri, S.A., Gupta, J.N.D. and ElMekkawy, T. (2007) 'Meta-heuristics for scheduling a flowline manufacturing cell with sequence dependent family setup times', *International Journal of Production Economics*, doi:10.1016/j.ijpe.2007.02.031.

Hyun, C.J., Kim, Y. and Kim, Y.K. (1998) 'A genetic algorithm for multiple objective sequencing problems in mixed model assembly lines', *Computer and Operations Research*, Vol. 25, Nos. 7/8, pp.675–690.

Jin, Y., Okabe, T. and Sendhoff, B. (2001) 'Adapting weighted aggregation for multiobjective evolutionary strategies', *Proceedings of the First Conference on Evolutionary Multi-Criterion Optimization*, pp.96–110.

Liu, J. and Reeves, C.R. (2001) 'Constructive and composite heuristic solutions to the P//$\sum$ Ci scheduling problem', *European Journal of Operational Research*, Vol. 132, No. 2, pp.439–452.

Mansouri, S.A. (2005) 'Coordination of set-ups between two stages of a supply chain using multi-objective genetic algorithms', *International Journal of Production Research*, Vol. 43, No. 5, pp.3163–3180.

Murata, T., Ishibuchi, H. and Tanaka, H. (1996) 'Multi-objective genetic algorithm and its applications to flowshop scheduling', *Computers and Industrial Engineering*, Vol. 130, No. 4, pp.957–968.

Nagar, A., Heragu, S.S. and Haddock, J. (1995) 'A branch and bound approach for a two-machine flowshop scheduling problem', *Journal of the Operational Research Society*, Vol. 46, No. 6, pp.721–734.

Schaller, J. (2001) 'A new lower bound for the flow shop group scheduling problem', *Computer and Industrial Engineering*, Vol. 41, No. 2, pp.151–161.

Schaller, J., Gupta, J.N.D. and Vakharia, A.J. (2000) 'Scheduling a flowline manufacturing cell with sequence dependent family setup times', *European Journal of Operational Research*, Vol. 125, No. 2, pp.324–339.

Schaumann, E.J., Balling, R.J. and Day, K. (1998) 'Genetic algorithms with multiple objectives', *Seventh AIAA/USAF/NASA/ISSMO Symposium on Multi-disciplinary Analysis and Optimization*, St. Louis, MO, AIAA, Vols. 3 and 99, pp.2114–2123.

Skorin-Kapov, J. and Vakharia, A.J. (1993) 'Scheduling a flow-line manufacturing cell: a tabu search approach', *International Journal of Production Research*, Vol. 31, No. 5, pp.1721–1734.

Srinivas, N. and Deb, K. (1994) 'Multiobjective optimization using nondominated sorting in genetic algorithms', *Evolution Computation*, Vol. 2, pp.221–248.

Vakharia, A.J. and Chang, Y.L. (1990) 'A simulation annealing approach to scheduling a manufacturing cell', *Naval Research Logistics*, Vol. 37, No. 6, pp.559–577.

Varadharajan, T.K. and Rajendran, C. (2005) 'A multi-objective simulated-annealing algorithm for scheduling in flowshops to minimize the makespan and total flowtime of jobs', *European Journal of Operational Research*, Vol. 167, No. 3, pp.772–795.

Operator staffing and scheduling for an IT-help call centre

Hesham K. Alfares

Department of Systems Engineering,
King Fahd University of Petroleum and Minerals,
P.O. Box 5067,
Dhahran 31261, Saudi Arabia
Fax: +9663-860-2965
E-mail: alfares@kfupm.edu.sa

Abstract: This paper describes the staffing and scheduling of IT help desk operators for a large petrochemical company. The objective is to reduce the labour cost by determining the best staffing level and employee weekly tour schedules required to meet the workload that varies over a 24-hr operating period. Several steps are taken for the staffing and tour scheduling of an IT help desk agents. First, data on the number of calls is used to estimate hourly labour requirements. Next, new scheduling options are proposed to better match these requirements. An Integer Programming (IP) model is then formulated and solved to determine tour scheduling assignments. Finally, alternative schedules are evaluated in terms of tradeoffs between workforce size and cost, service level and employee utilisation. The chosen tour schedules provide better service with a lower cost and a smaller number of employees.
[Received 10 November 2006; Revised 23 April 2007; Accepted 18 June 2007]

Keywords: employee scheduling; staffing; queueing models; stochastic demand; call centre scheduling; integer programming; IP.

Reference to this paper should be made as follows: Alfares, H.K. (2007) 'Operator staffing and scheduling for an IT-help call centre', *European J. Industrial Engineering*, Vol. 1, No. 4, pp.414–430.

Biographical notes: Hesham K. Alfares is a Professor in the Department of Systems Engineering of King Fahd University of Petroleum and Minerals in Dhahran, Saudi Arabia. He received a BS in Electrical and Computer Engineering from the University of California, Santa Barbara, in 1982. He received an MS in Industrial Engineering from the University of Pittsburgh in 1984 and a PhD in Industrial Engineering from Arizona State University in 1991. His research interests include employee scheduling, production and inventory control, petrochemical industry modelling and maintenance modelling and simulation. He has 30 papers in these areas in various international journals, in addition to many conferences papers, technical reports and funded research projects.

1 Introduction

This paper presents the modelling and solution of an actual tour scheduling problem at the Information Technology Help Desk (ITHD) of a large petrochemical company in Saudi Arabia. The objective is to minimise the labour cost by determining the optimum number of agents and their schedules to meet the fluctuating demand. ITHD agents provide IT services and support 24 hr a day, seven days a week to all the company's employees. They receive and respond to employees' technical enquiries, problems and complaints. ITHD has 47 agents, in three groups with different pay scales and different work schedules:

1 one company employee on regular day work

2 19 company employees on shifts

3 27 contractors.

In Saudi Arabia, it must be noted that the regular workdays are Saturday to Wednesday and the weekends are Thursday and Friday.

ITHD management noticed callers' frustration with being placed on hold for several minutes during peak periods, although ITHD uses the First-Come First-Served (FCFS) rule to queue incoming calls. According to ITHD management, many users complained because of long delays and gave the management the impression that many calls were lost. Unscheduled 1-hr breaks created another problem for ITHD; any agent who deserves a 1-hr break was allowed to take it randomly any time during a predetermined 2-hr interval. This policy gave the agents the flexibility to go whenever they wanted to go but, on the other hand, it made it too difficult to constantly maintain an adequate staffing level.

Prior to this study, hourly labour demands were estimated and employees were assigned to different schedules-based only on management observations and the shifts leaders' opinions; no systematic or scientific methods were used. In order to systematically determine the optimum tour schedule, several steps were taken. First, extensive data on the number of and duration of incoming calls for each hour of the day and each day of the week was collected and analysed. Using queuing theory, this data was converted into hourly labour demands for a typical work week. Next, an Integer Programming (IP) model that represents the various restrictions and alternative schedules was formulated and solved. Based on the IP solution, detailed weekly tour schedules were constructed that defined work hours, meal break times and off days for each employee.

The remainder of this paper is organised as follows. Related literature is surveyed in Section 2, and then the problem and context are introduced in Section 3. Hourly staffing demands are calculated in Section 4, and the assumptions and alternatives are presented in Section 5. The IP model is formulated and solved in Section 6. Finally, some conclusions are drawn in Section 7.

2 Literature survey

The focus of this literature review is on recent techniques for call centre staffing and telephone operator scheduling. One of the most popular techniques is Linear Programming (LP) and its variations, including IP and Goal Programming (GP).

Willis and Huxford (1991) applied IP to generate telephone operator rosters for Telecom Australia. Thompson (1997) used a GP-like heuristic to assign New Brunswick Telephone Company operators to shifts, aiming to meet all demands and satisfy employee preferences in the order of seniority. Brusco and Jacobs (2000) formulated a compact implicit IP model for flexible tour scheduling of employees at a Motorola call centre. Çezik et al. (2001) developed an IP model to schedule agents in a call centre by combining days-off and shift scheduling constraints into a network flow structure.

Many LP-based approaches combine LP with other techniques. Caprara et al. (2003) used IP, dynamic programming and heuristics to determine the minimum-workforce days-off schedule of emergency call centre employees. Lin et al. (2000) combined regression and simulation models to determine the hourly staffing levels for a 24-hr hotline service. They then used a mixed IP model to develop equitable daily and monthly schedules of senior and junior operators. Bard (2004) used IP followed by several post-processors to determine the full-time staffing level for a service facility with a highly variable demand. Harrison and Zeevi (2005) used stochastic fluid models to reduce the call centre staffing problem to a multidimensional newsboy problem, which they numerically solved by LP and simulation.

Simulation has been widely used in scheduling call centre operators. Saltzman and Mehrotra (2001) used a simulation model for staffing a technical support call centre in a software company. As mentioned earlier, Harrison and Zeevi (2005) combined simulation with LP for call centre staffing. Atlason et al. (2004) also combined simulation and IP in an iterative cutting plane algorithm to determine the minimum-cost schedule of a call centre's employees. AbdelMalek and Allahverdi (2005) used simulation to analyse the (G/G/c) queuing model in order to determine the optimum number of technicians for the help desk of a telecommunications company.

Queuing models have been primarily used to determine call centre staffing levels to satisfy specific service-level criteria. Agnihothri and Taylor (1991) determined hourly staffing requirements by the (M/M/c) queuing model and rearranged the work shifts to schedule a hospital phone appointment workforce. Andrews and Parsons (1993) used economic optimisation instead of service-level criteria for determining staffing levels. Fromm (1997) discussed Erlang queuing formulas for calculating the required number of operators. Estimating staffing costs to account for over half of a call centre's total operating cost, Duder and Rosenwein (2001) used queuing-based 'rule-of-thumb' formulas to estimate the cost of abandonment and to determine the optimum number of agents. Green et al. (2003) used queuing models and heuristic rules to determine staffing levels for a call centre. Whitt (2005a) used approximations of Markovian queuing models to estimate staffing levels needed to achieve abandonment rate and waiting time targets. Whitt (2005b) also uses queuing analysis for staffing a call centre, modelling employee absenteeism by considering the proportion of servers present as a random variable.

Call centre employee scheduling techniques include meta-heuristics and local search algorithms. Yamada et al. (1999) used a genetic algorithm approach to schedule a variable number of operators in a telephone information centre, aiming to maintain service quality and reduce labour costs. Koole and van der Sluis (2003) utilised local search for call centre shift scheduling with a service level objective, using the problem's multimodularity property to ensure convergence to a globally optimal solution.

Several software systems are available for staffing and scheduling call centre employees. Fromm (1997) compared the use of several software products for call centre scheduling, both to determine hourly staffing demands and to construct optimum

operator schedules. Brazier et al. (1999) described a multiagent system architecture to schedule call centre agents, using the compositional development method for multiagent systems DESIRE. Fukunaga et al. (2002) used DIRECTOR, a constraint-based staff scheduling system that utilises artificial-intelligence search methods, to produce optimum call centre schedules. Yang et al. (2003) described a scheduling algorithm implemented on SANet software system to balance quality, efficiency and cost of a call centre for a cable-TV company.

3 The problem and context

The ITHD's main function is to provide high quality, timely and cost-effective services and support to all of its customers. The 'customers' of ITHD are all employees of this large petrochemical company (several thousand employees), who may call ITHD for IT related services and support. The ITHD is staffed by agents 24 hr a day, seven days a week. Agents receive customers' questions and complaints and try to solve them immediately, that is, within approximately 5–10 min, as much as they can. Otherwise, they escalate these issues to another level by booking a trouble ticket. Since ITHD was created to assist customers, courtesy and good communications skills are necessary to keep the help desk running well, no matter how busy the day is or how many problems have arisen.

The IT Help Desk has 47 agents, classified into three categories:

1 company employees who work in shifts

2 non-shift company employees who work in regular day schedules

3 contractors who work in shifts.

The current tour schedules of agents in these three categories, shown in Table 1, are described as follows.

Table 1 The current assignment of 47 employees to 11 tours

No.	*Agent type*	*Shift time*	*Agents*	*Break*	*Off days*
1	Employees on shifts	3 shifts, 24 hr	19	None	Varying
2	Day employees	7:00 am–4:00 pm	1	12:00 pm–1:00 pm	Thu–Fri
3	Contractors	6:00 am–3:00 pm	16	11:00 am–1:00 pm	Thu–Fri
4	Contractors	9:00 am–6:00 pm	4	12:00 pm–2:00 pm	Tue–Wed
5	Contractors	3:00 pm–12:00 pm	1	7:00 pm–9:00 pm	Mon–Tue
6	Contractors	3:00 pm–12:00 pm	1	7:00 pm–9:00 pm	Wed–Thu
7	Contractors	3:00 pm–12:00 pm	1	7:00 pm–9:00 pm	Fri–Sat
8	Contractors	3:00 pm–12:00 pm	1	7:00 pm–9:00 pm	Tue–Wed
9	Contractors	3:00 pm–12:00 pm	1	7:00 pm–9:00 pm	Sun–Mon
10	Contractors	3:00 pm–12:00 pm	1	7:00 pm–9:00 pm	Thu–Fri
11	Contractors	3:00 pm–12:00 pm	1	7:00 pm–9:00 pm	Sat–Sun

3.1 Company employees on shift schedules

This employee category has 19 agents divided into four groups. Groups A, B and C consist of five agents each, while group D consists of four agents. The agents work in three different shifts:

1 from 7:00 am to 3:00 pm

2 from 3:00 pm to 11:00 pm

3 from 11:00 pm to 7:00 am.

These three 8-hr work shifts do not have meal breaks. Over a four-week cycle, employees in this category work seven consecutive days on each of the three shifts, using the days-on/days-off pattern 7/2-7/2-7/3. Ideally, this schedule is carried out by four equally-sized groups of employees, such that on any given day, three groups are working (one on each shift), while one group is off.

3.2 Company employees on day schedule

Only one agent is assigned to the regular day schedule who works five days a week, Saturday to Wednesday, taking the weekend off on Thursday and Friday. This agent begins at 7:00 a.m. and finishes at 4:00 p.m., taking a 1-hr lunch break from 12:00 noon to 1:00 pm.

3.3 Contractors

Currently, there are 27 contractors assigned to three shift types as follows:

1 *6:00 am to 3:00 pm*: 16 contractors are assigned to this shift, taking 1-hr lunch breaks any time from 11:00 am to 1:00 pm. These agents work from Saturday to Wednesday and take Thursday and Friday off.

2 *9:00 am to 6:00 pm*: in this shift, there are only four contractors, who work from Thursday to Monday and take Tuesday and Wednesday off. Each agent can take a 1-hr lunch break anytime from 12:00 noon to 2:00 pm.

3 *3:00 pm to 12:00 midnight*: during this time period, seven contractors are assigned to seven different tours differing only in the pair of consecutive off days. Each of these contractors takes a 1-hr dinner break any time between 7:00 pm and 9:00 pm.

The management of the ITHD observed an overall moderate satisfaction with the service. However, callers expressed serious frustration with being placed on hold for several minutes during peak periods, although the FCFS rule is used to queue incoming calls. Management's initial response to these complaints was to specify a minimum number of agents needed to meet the demand over different intervals as shown in Table 2.

The minimum staffing values shown in Table 2 are not reliable. In estimating these values, ITHD management depended on its casual observations and the shifts leaders' opinions. Management did not use any systematic or scientific method to estimate the workload and translate it into number of agents. Therefore, although the management

reports that working with the values in Table 2 had decreased the number of lost calls and users' complaints. According to management, the problems still persisted. many calls continued to be lost and many users (callers) complained.

Table 2 Minimum no. of agents needed for time intervals specified by management

Workdays		*Weekend*	
Time interval	*No. of agents*	*Time interval*	*No. of agents*
6:00 am–11:00 am	20	9:00 am–3:00 pm	8
11:00 am–1:00 pm	16	3:00 pm–12:00 midnight	6
1:00 pm–3:00 pm	20	12:00 midnight–9:00 am	3
3:00 pm–6:00 pm	10		
6:00 pm–12:00 am	9		
12:00 midnight–6:00 am	3		

Another problem with the current schedule is the unscheduled 1-hr breaks. Any contractor agent who deserved a 1-hr break was free to take it randomly any time during a predetermined 2-hr interval as given in Table 1. This policy gives the agents some flexibility in taking their meal breaks. However, it makes it too difficult for the shift leader to control the staffing levels in order to satisfy the varying workload.

Based on the above problem description, the IT Help Desk management aimed to scientifically find the minimum-cost staffing levels and tour schedules of the agents subject to meeting labour demands in each work period. The tour schedules must specify the details of different days off, work hours and meal breaks for each employee. Moreover, for managerial reasons, the number of company employees on shift schedules must be between 5 and 20 agents in any proposed schedule. In order to achieve these objectives, the work was divided into the following stages: data collection and analysis to determine hourly labour requirements, model formulation and model solution.

4 Data collection and analysis to estimate labour demands

In administering telephone-agent operations, the minimum number of agents must be determined for short-term staffing of the call centre. The ITHD management needed to forecast the workload on an hourly basis. This hourly workload is a function of both the number and the duration of calls received in each hour. Thus, two types of data were collected: the number of calls per hour and the durations of answered calls. One fact concerning data collection must be noted here. The actual call arrival (caller dialling) time was not kept in the ITHD's database. Naturally, some calls that were made to the ITHD were never answered. Records were kept, only for *answered* calls, of the call's start (agent answering) time, which is different from the call arrival (caller dialling) time. Thus, no data was available on inter-arrival times, that is, times between caller dialling of all successive calls, whether answered or not.

4.1 Call frequency and duration data

In order to build a reliable history of the pattern of the demand (number of calls), historical data of the number of calls received during each hour was collected for five typical months. The data was used to calculate the average number of received calls for every hour of the day and every day of the week. The average number of calls during each of the 168 hr of the week, shown in Figure 1, depends mostly on the type of day (whether workday or weekend). Figure 1 clearly shows that the workdays (1–120 hr) have a higher number of calls than the weekend (121–168 hr). The average of calls per hour is 79.21 during workdays, but the hourly average drops to 29.10 during weekends. In workdays, call frequency is higher during the official work hours of the company (7:00 am to 4:00 pm).

Figure 1 Average number of arriving calls in one week using the historical data

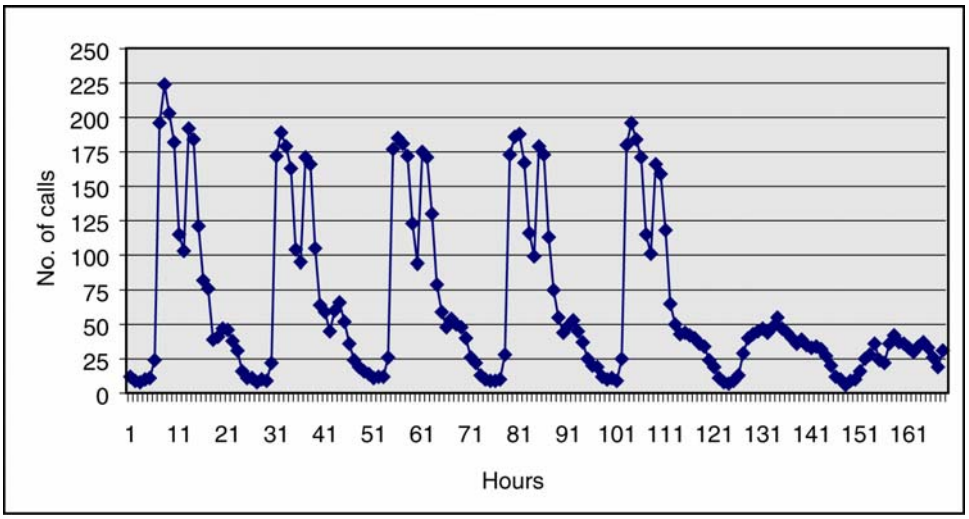


Figure 1 shows that the call frequency depends on both the time of the day and the day of the week. It is apparent that call reception pattern for the workdays (Saturday to Wednesday) differs significantly from that of the weekend (Thursday and Friday). All the workdays have very similar patterns. For example, all workdays have two peak periods, one from 6:00 am to 11:00 am and the other from 12:00 noon to 3:00 pm. We could also make a similar observation about the weekends. Therefore, the average number of hourly calls was calculated separately for workdays and weekends. Table 3 displays the average number of calls received per hour during each workday and each weekend day.

The next step in data collection and analysis had to do with service time or call duration. Service time is defined as the time duration of each successfully-completed call to the ITHD, from the time the agent answers the call to the time the call ends. In order to find the average service time, historical data was collected and statistically analysed using STATISTICA. The service time (call duration) has a mean of 4.033 min and a standard deviation of 1.741 min. The Chi-square goodness-of-fit test, at significance level $\alpha = 0.05$, could not confirm the hypothesis that service time (call duration) has an exponential distribution.

Table 3 Average and maximum number of calls for workdays and weekends

Time of day	*Average calls*		*Maximum calls*	
	Workdays	*Weekend*	*Workdays*	*Weekend*
12:00 midnight–1:00 am	15	8	19	20
1:00 am–2:00 am	12	12	14	12
2:00 am–3:00 am	10	9	11	10
3:00 am–4:00 am	11	7	12	7
4:00 am–5:00 am	11	9	12	9
5:00 am–6:00 am	25*	12*	28	13
6:00 am–7:00 am	180	23	196	29
7:00 am–8:00 am	196*	33	224	40
8:00 am–9:00 am	187	36	203	43
9:00 am–10:00 am	171	41*	182	45
10:00 am–11:00 am	115*	36*	123	47
11:00 am–12:00 noon	99	33	103	44
12:00 noon–1:00 pm	177*	42	192	48
1:00 pm–2:00 pm	171	49*	184	55
2:00 pm–3:00 pm	118	42	130	47
3:00 pm–4:00 pm	73*	40*	82	44
4:00 pm–5:00 pm	60	37	76	40
5:00 pm–6:00 pm	44	33	48	36
6:00 pm–7:00 pm	50	37	60	39
7:00 pm–8:00 pm	52	36	66	37
8:00 pm–9:00 pm	47*	33*	52	33
9:00 pm–10:00 pm	38	30	40	34
10:00 pm–11:00 pm	28	26	34	32
11:00 pm–12:00 midnight	21	12	24	31

*Average call arrival rate λ_i per hour for the given time interval.

4.2 Using queuing to determine staffing requirements

The staffing requirements in call centres are usually determined by either queuing or simulation models. Simulation models are more flexible, but they require complete specification of all relevant distributions. Since the inter-arrival distribution is not available for building a simulation model, a queuing model was used to estimate hourly staffing demands. Although the service time distribution is not exponential, a Markovian queuing model was used to determine the hourly labour demands. This approximation is justified by the following facts:

1 Markovian queuing models are robust to the assumptions made.

2 Non-Markovian queuing model are difficult to solve analytically.

3 Although call inter-arrival time data is not available, it can be assumed exponentially distributed because call arrivals are completely random for each hour.

4 Similar results were obtained with the Allen-Cunneen approximation (Willis and Huxford, 1991) of the *G*/*G*/*c* queuing model for general service and inter-arrival time distributions.

FCFS rule is applied in answering incoming calls as a policy of the IT help desk. Calls that cannot be answered immediately because all agents are busy join a queue and wait for agents to be available. Based on this description, using Kendall's notation as in Thompson (1997), an (*M*/*M*/*c*):(*GD*/∞/∞) queuing model was used to represent the system and determine hourly staffing requirements. For both workdays and weekends, the intervals with similar arrival rates were grouped and analysed together using a separate queuing model with the same arrival rate. Since ITHD data includes only answered calls, it may underestimate the actual call arrival rate. In order to account for abandoned (unanswered) calls, the largest average hourly call rate was taken as the arrival rate λ_i for the given time interval. For example, we assumed a rate of 196 calls per hour for the whole interval (6:00 am–10:00 am) during weekdays. The time intervals and corresponding arrival rates are given in Table 3.

Since the day of the week has no effect on the service time (call duration), a uniform service time distribution with a mean μ was used for every day of the week. This mean service rate represents the average number of calls processed per time unit. In order to calculate the effective service rate in terms of calls per hour, we must consider only the number of productive work minutes per hour. Out of each 8-hr work shift, approximately 1 hr is lost due to unscheduled breaks, personal needs and so on. Therefore, the net work time averages (7/8) × 60 = 52.5 min per hour. Since the average call duration is 4.033 min, the effective service rate is given by:

$$\mu = \frac{52.5}{4.033} \cong 13 \,\text{calls per hour} \tag{1}$$

Various service measures can be used to evaluate a call centre's performance, including probability of waiting, average waiting time and abandonment rate (proportion of lost calls). Since the IT Help desk serves the company's own employees, abandoned calls are not equivalent to lost sales and thus are not considered as a serious problem. In fact, management was mostly concerned with minimising excessive waiting which has led to user dissatisfaction and complaints. Therefore, the IT Help desk management wanted to limit the average waiting time.

For the (*M*/*M*/*c*):(*GD*/∞/∞) queue, the expected waiting time $W_{q,i}$ if c_i operators are assigned during hour *i* is given by Thompson (1997):

$$p_{0,i} = \left\{ \sum_{n=0}^{c_i-1} \frac{\rho^n}{n!} + \frac{\rho^{c_i}}{c_i!} \left(\frac{1}{1-\rho/c_i} \right) \right\}^{-1}, \quad \frac{\rho}{c_i} < 1 \tag{2}$$

$$W_{q,i} = \frac{\rho^{c_i+1} p_0}{\lambda_i (c_i - 1)! (c_i - \rho)^2} \tag{3}$$

where $\rho = \lambda_i/\mu$; $p_{0,i}$ is the probability of no waiting (zero queue length) during hour i; λ_i is the arrival rate (number of calls received during hour i); μ is the service rate (number of calls processed per hour); c_i is the number of operators working during hour i.

By setting service-level target values on the maximum expected waiting time ($W_{q,i} \le W_{\max}$), the minimum staffing level required for each hour c_i can be determined by Equations (2) and (3). The IT Help Desk management wanted to have alternative solutions that could be satisfied by the existing workforce size. Therefore, management specified two values of $W_{\max}$, aiming to choose the most appropriate resulting feasible schedule. Specifically, the two following cases were proposed by the management:

Case I $W_{\max} = 2$ min.

Case II $W_{\max} = 5$ min.

The call centre industry best practices for average waiting time range from 20 to 50 sec [10, 16]. However, the ITHD management could not adopt the common standard of $W_{\max} = 0.5$ min because it would require hiring more agents than currently available, at a higher cost than the current schedule. Applying Equations (2) and (3), the minimum number of the agents needed for each hour c_i was calculated using two values of $W_{\max}$ (2 and 5 min). Thus, two initial estimates of the number of agents required for each hour (c_i) were obtained. Assuming exponential call inter-arrival time, comparison of the results obtained from (2) and (3) with the Allen-Cunneen (Willis and Huxford, 1991) approximation of $W_{q,i}$ confirmed the validity of using the (M/M/c):(GD/∞/∞) queuing model.

In real life, staffing levels must include allowances for employee absenteeism caused by vacations, training assignments, sick and emergency leaves and so on. The ITHD management uses the company's standard 10% allowance to account for employee absenteeism, which is based on one-month annual vacations (1/12 = 8.3%) plus training and sick leaves (1.7%). Including this allowance, the minimum staffing requirement for each hour i is calculated by:

$$r_i = \lceil 1.1\, c_i \rceil \tag{4}$$

where [s] is the smallest integer greater than or equal to s.

It should be noted here that the adjustment in the service rate defined by (1) accounts for lost times of working employees, while the adjustment in staffing levels defined by (4) accounts for lost times of absent employees. The days of the week were divided into two groups (workdays and weekends), and the days in each group were divided into intervals with similar call arrival rates λ_i. The staffing requirements for the two values of $W_{\max}$ are given in Table 4.

5 Scheduling assumptions and alternatives

The following inputs and assumptions are needed to completely describe the agent tour scheduling problem, as a step towards building the IP model.

Table 4 Minimum number of agents needed in hour i, r_i, as a function of W_q

Time interval	*Range of hours i*	*Case I**		*Case II***	
		Workdays	*Weekends*	*Workdays*	*Weekends*
12:00 midnight–6:00 am	1,…, 6	4	3	4	3
6:00 am–10:00 am	7,…, 10	19	6	18	5
10:00 am–12:00 noon	11,…, 12	13	5	11	5
12:00 noon–3:00 pm	13,…, 15	18	6	17	6
3:00 pm–8:00 am	16,…, 20	8	6	8	5
8 pm–12:00 midnight	21,…, 24	6	5	6	5

*$W_q \leq 2$ min.

**$W_q \leq 5$ min.

5.1 Assumptions

1 Shifts start and finish only on the hour.

2 Contractors' shift length is 9 hr with 1 hr (lunch or dinner) break. Company employees' shift length is 8 hr without a 1-hr break.

3 Normal interruptions and short breaks do not need to be scheduled.

4 One-hour meal breaks are scheduled over 2-hr intervals as described in Table 1.

5 The number of company employees on shift schedule must be between 5 and 20. Since this schedule applies to four equally-sized groups of agents, each group must have 1–5 agents. Letting x_1 denote the group size, then x_1 must be between 1 and 5 and the cost of x_1 must be multiplied by 4 in the objective function.

6 The total monthly cost per employee, including benefits and overhead, is given for the three employee categories as:

   a Shift employees: \$5781.39.
   b Non-shift (day) employees: \$4680.19.
   c Contractors on any shift: \$7978.31.

The current total number of agents is 47 agents (19 shift employees, 1 day employee and 27 contractors. Based on the above costs, the total monthly cost is = 19 × 5781.39 + 1 × 4680.19 + 27 × 7978.31 = \$329,940.97.

5.2 New scheduling options

In addition to the 11 existing tour types described in Table 1, new tours were introduced to provide more flexibility in order to better match the varying demand pattern. According to management instructions, only the new tours that would cause the least disruption to the current work practices could be considered. New tours were created by introducing the following combinations of days off, shift start/finish times and lunch (dinner) hours.

1 Non-shift (day) employees work hours will be from 6:00 am to 3:00 pm instead of 7:00 am to 4:00 pm. It is better to start at 6:00 am because the peak period starts at 6:00 am. The 1-hr lunch break is still taken from 12:00 noon to 1:00 pm.

2 For contractors who work from 6:00 am to 3:00 pm with Thursday and Friday off, 14 variations were introduced by considering all seven pairs of consecutive days-off and two possible lunch hours (either 11:00 am–12:00 noon or 12:00 noon–1:00 pm).

3 For contractors who work from 9:00 am to 6:00 pm with Tuesday and Wednesday off, the work hours were delayed to 10:00 am–7:00 pm in order to avoid overlapping their lunch hours with those of the 6:00 am shift. The lunch break will be from 1:00 pm to 3:00 pm instead of 11:00 am to 1:00 pm. Moreover, 14 variations were introduced by considering all seven pairs of consecutive days-off and two possible lunch hours (either 1:00 pm–2:00 pm or 2:00 pm–3:00 pm).

4 For contractors who work from 3:00 pm to midnight with different pairs of days off, 14 variations were introduced by considering all seven pairs of consecutive days-off and two possible dinner hours (either 7:00 pm–8:00 pm or 8:00 pm–9:00 pm).

6 Model construction and solution

After determining labour requirements for each hour of the week and developing new tours to better match these requirements, an IP model is constructed to optimise employee tour schedules. The objective of the IP model shown below is to minimise labour cost, subject to meeting the hourly staffing demands and satisfying all applicable scheduling rules and constraints.

$$\text{Minimise } Z = \sum_{j=1}^{44} k_j x_j \tag{5}$$

s.t.

$$\sum_{j=1}^{44} a_{ij} x_j \geq r_i, \quad i = 1,2,\ldots,168 \tag{6}$$

$$1 \leq x_1 \leq 5 \tag{7}$$

$$x_j \geq 0 \text{ and integer}, \quad j = 1,2,\ldots,44 \tag{8}$$

where Z: labour cost, that is, total cost of employees assigned to all tours; x_j: number of employees assigned to weekly tour j; k_j: cost of weekly tour j; a_{ij}: 1 if hour i is a work period for tour j, 0 otherwise; r_i: minimum number of employees required in hour i (for Cases I or II).

6.1 Alternative solutions

Using the above inputs and assumptions, the integer programme was solved for two different values of the right hand side vector r_i, corresponding to the two cases. Using LINDO (1994) to solve the two resulting IP problems, two alternative solutions were obtained. Table 5 compares the features of the two proposed solutions with the current schedule.

Table 5 Comparison of alternative solutions

Solution property	*Current schedule*	*Case I solution*	*Case II solution*
Shift employees: $4x_1$	19	20	20
Non-shift employees: x_2	1	5	4
Contractors: $x_3 + x_4 + \ldots + x_{44}$	27	19	17
Total workforce	47	44	41
Total monthly cost ($)	329,941	290,617	269,980
Workforce utilisation (%)	76.1	81.1	86.9
Agent occupancy (%)	25.7	27.3	29.3

It must be noted that workforce utilisation is defined as the ratio of required man-hours per week obtained from the queuing model, R, to assigned man-hours per week, S. On the other hand, average agent occupancy, is the ratio of actual weekly workload (call volume) in minutes, L, to assigned man-minutes per week, $60S$. Average required man-hours per week can be calculated from Table 4 (Case I) as follows:

$$R = \sum_{i=1}^{168} r_i = 1460$$

Since each assigned shift contains eight work hours, the assigned man-hours per week are calculated for each case as follows:

$$S = 7 \times 24 \times x_1 + 5 \times 8 \sum_{j=2}^{44} x_j$$

Thus

$$\text{Workforce utilisation} = \frac{R}{S} = \frac{1460}{168x_1 + 40 \sum_{j=2}^{44} x_j} \tag{9}$$

In order to calculate the required weekly call volume, we first use Table 3 to calculate the average number of total calls per week:

$$\text{Average number of weekly calls} = \sum_{i=1}^{168} \lambda_i = 7320$$

Next, we multiply the average number of calls per week by the average call duration in order to calculate the average actual weekly workload:

$$L = 4.033 \times 7320 = 29{,}521.56 \,\text{min/week}$$

Finally

$$\text{Agent occupancy} = \frac{L}{60S} = \frac{29{,}521.56}{10{,}080x_1 + 2400\sum_{j=2}^{44} x_j} \tag{10}$$

6.2 Comparing solutions

The choice between the two alternative solutions will depend primarily on the waiting time, the total cost and the total number of agents needed. The choice will also be affected by the other advantages and disadvantages of each solution, according to the preferences of the IT help desk management.

Case I: By setting $W_q \leq 2$ min, this solution significantly reduces the waiting time and improves the service quality. Moreover, the workforce size and cost are both reduced. The workforce size is reduced to 44 agents, producing a fairly high (81%) utilisation of the agents. The IT help desk management liked this solution because:

1 it decreases labour cost by \$39,324 per month

2 it reduces the workforce size by three agents

3 it increases agent occupancy by almost 2%

4 it minimises the waiting time in comparison to Case II.

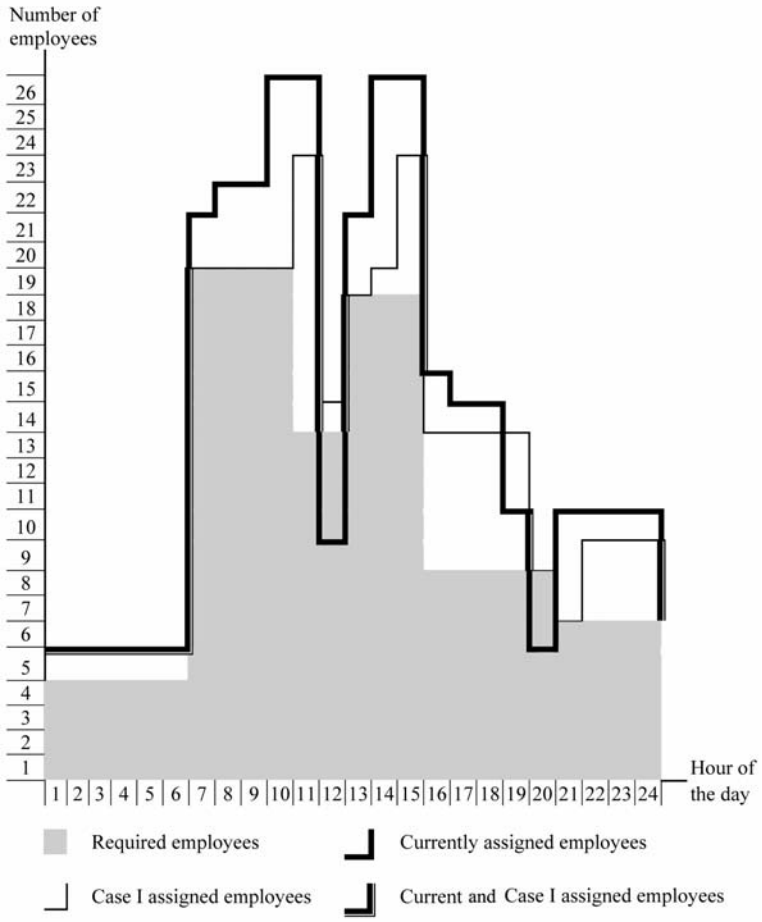
Case II: By setting $W_q \leq 5$ min, Case II solution improves the existing level of customer service. The workforce size is reduced to 41 agents, leading to the maximum workforce utilisation (87%) and agent occupancy (29%). This solution reduces labour cost by \$59,961 per month and the number of agents by six. However, agents could be under pressure especially if there is excessive absence (for medical reasons or annual leave). In that case, the problems of long queues and lost calls may occasionally resurface during peak periods.

After evaluating and analysing all alternative solutions, the ITHD management chose the Case I solution. Compared to the current schedule, this solution reduces the number of agents by 6.4%, saves 11.9% of the labour cost and increases the workforce utilisation by 5% points. Another advantage of Case I solution is the fact that the 44 agents are assigned to only 10 different tours instead of 11 tours in the current schedule, as described in Table 6. A smaller set of active tours is a desirable feature from a managerial point of view, because it simplifies the administration and supervision responsibilities. Figure 2 shows why the proposed solution succeeds in meeting all hourly demands, although it has a smaller workforce than the current schedule.

Table 6 Details of tour assignments obtained from the two proposed solutions

No.	*Agent type*	*Shift time*	*Break*	*Off days*	*Case I*	*Case II*
1	Shift employees	3 shifts, 24 hr	None	Varying	20	20
2	Day employees	6:00 am–3:00pm	12:00 pm–1:00 pm	Thu–Fri	5	4
3	Contractors	6:00 am–3:00 pm	11:00 am–12:00 pm	Sun–Mon	1	
4	Contractors	6:00 am–3:00 pm	11:00 am–12:00 pm	Thu–Fri	8	9
5	Contractors	6:00 am–3:00 pm	11:00 am–12:00 pm	Fri–Sat	1	
6	Contractors	10:00 am–7:00 pm	1:00 pm–2:00 pm	Sun–Mon		1
7	Contractors	10:00 am–7:00 pm	1:00 pm–2:00 pm	Tue–Wed		1
8	Contractors	10:00 am–7:00 pm	1:00 pm–2:00 pm	Thu–Fri	4	2
9	Contractors	3:00 pm–12:00 pm	7:00 pm–8:00 pm	Thu–Fri	1	1
10	Contractors	3:00 pm–12:00 pm	8:00 pm–9:00 pm	Tue–Wed	1	
11	Contractors	3:00 pm–12:00 pm	8:00 pm–9:00 pm	Thu–Fri	2	3
12	Contractors	3:00 pm–12:00 pm	8:00 pm–9:00 pm	Fri–Sat	1	

Figure 2 Assigned and required employees for each hour on Saturday



7 Conclusions

This paper described an employee staffing and tour scheduling approach for the ITHD of a large petrochemical company. Extensive data was collected on the number of calls received by ITHD per hour for each hour of the day and each day of the week. Data was also collected on the ITHD service time (call duration). Statistical techniques and queuing models were used to analyse the data in order to determine the minimum number of agents required for each hour of the week. New tour schedules were proposed to provide more flexibility to better meet labour demands while minimising disruption to the current scheduling system. An IP model was constructed to find the optimum employee tour schedules that satisfy labour requirements with the minimum cost.

Using two service-level targets to set labour demands, two tour scheduling solutions were obtained and evaluated. The proposed tour schedules define for every employee the starting and finishing work hours, the lunch or dinner break hour (if applicable) and the days off. The two proposed solutions were compared in order for the management to choose the best schedule. The chosen solution is expected to save $470,000 a year while satisfying customer demands and management objectives.

Acknowledgement

The author wishes to express gratitude to King Fahd University of Petroleum and Minerals for providing support and research facilities.

References

AbdelMalek, F. and Allahverdi, A. (2005) 'Optimizing a help desk performance at a telecommunication company', *Proceedings of the 1st International Conference on Modelling, Simulation, and Applied Optimization*, Sharjah, United Arab Emirates, 1–3 February.

Agnihothri, S.R. and Taylor, P.F. (1991) 'Staffing a centralized appointment scheduling department in Lourdes hospital', *Interfaces*, Vol. 21, No. 5, pp.1–11.

Andrews, B.H. and Parsons, H.L. (1993) 'Establishing telephone agent staffing levels through economic optimization', *Interfaces*, Vol. 23, No. 2, pp.14–20.

Atlason, J., Epelman, M. and Henderson, S.G. (2004) 'Call centre staffing with simulation and cutting plane methods', *Annals of Operations Research*, Vol. 127, Nos. 1–4, pp.333–358.

Bard, J.F. (2004) 'Selecting the appropriate input data set when configuring a permanent workforce', *Computers and Industrial Engineering*, Vol. 47, No. 4, pp.371–389.

Brazier, F.M.T., Jonker, C.M., Jungen, F.J. and Treur, J. (1999) 'Distributed scheduling to support a call centre: a cooperative multiagent approach', *Applied Artificial Intelligence*, Vol. 13, Nos. 1/2, pp.65–90.

Brusco, M.J. and Jacobs, L.W. (2000) 'Optimal models for meal-break and start-time flexibility in continuous tour scheduling', *Management Science*, Vol. 46, No. 12, pp.1630–1641.

Caprara, A., Monaci, M. and Toth, P. (2003) 'Models and algorithms for a staff scheduling problem', *Mathematical Programming*, Vol. 98, Nos. 1–3, pp.445–476.

Çezik, T., Günlük, O. and Luss, H. (2001) 'An integer programming model for the weekly tour scheduling problem', *Naval Research Logistics*, Vol. 48, No. 7, pp.607–624.

Duder, J.C. and Rosenwein, M.B. (2001) 'Towards "zero abandonments" in call centre performance', *European Journal of Operational Research*, Vol. 135, No. 1, pp.50–56.

Fromm, A. (1997) 'How to achieve optimum operator staffing', *Answer Magazine*, Spring, Vol. 97, ATSI Online, Available at: http://www.atsi.org/publications/answer/1997/1997_2.html.

Fukunaga, A., Hamilton, E., Fama, J., Andre, D., Matan, O. and Nourbakhsh, I. (2002) 'Staff scheduling for inbound call centres and customer contact centres', *AI Magazine*, Vol. 23, No. 4, pp.30–40.

Green, L.V., Kolesar, P.J. and Soares, J. (2003) 'An improved heuristic for staffing telephone call centres with limited operating hours', *Production and Operations Management*, Vol. 12, No. 1, pp.46–61.

Harrison, J.M. and Zeevi, A. (2005) 'A method for staffing large call centres based on stochastic fluid models', *Manufacturing and Service Operations Management*, Vol. 7, No. 1, pp.20–36.

Koole, G. and van der Sluis, E. (2003) 'Optimal shift scheduling with a global service level constraint', *IIE Transactions*, Vol. 35, No. 11, pp.1049–1055.

Lin, C.K.Y., Lai, K.F. and Hung, S.L. (2000) 'Development of a workforce management system for a customer hotline service', *Computers and Operations Research*, Vol. 27. No. 10, pp.987–1004.

LINDO Systems Inc. (1994) *LINDO/386 5.3*, Chicago, IL, USA, Available at: www.lindo.com.

Saltzman, R.M. and Mehrotra, V. (2001) 'A call centre uses simulation to drive strategic change', *Interfaces*, Vol. 31, No. 3, pp.87–101.

StatSoft, Inc. (2002) *STATISTICA 6.1*, Tulsa, OK, USA, Available at: www.statsoft.com.

Taha, H. (2003) *Operation Research: An Introduction*, 7th edition, USA: Prentice-Hall, pp.611–613.

Tanner, M. (1995) *Practical Queueing Analysis*, London: McGraw-Hill, pp.218–220.

Thompson, G.M. (1997) 'Assigning telephone operators to shifts at New Brunswick Telephone Company', *Interfaces*, Vol. 27, No. 4, pp.1–11.

Whitt, W. (2005a) 'Engineering solution of a basic call-centre model', *Management Science*, Vol. 51, No. 2, pp.221–235.

Whitt, W. (2005b) 'Staffing a call centre with uncertain arrival rate and absenteeism', *Working Paper*, Columbia University, Available at: http://www.columbia.edu/~ww2040/FluidStaff2.pdf.

Willis, R.J. and Huxford, S.B. (1991) 'Staffing rosters with breaks – a case study', *Journal of the Operational Research Society*, Vol. 42, No. 9, pp.727–731.

Yamada, T., Yoshimura, K. and Nakano, R. (1999) 'Information operator scheduling by genetic algorithms. Simulated Evolution and Learning', *Lecture Notes in Artificial Intelligence*, Vol. 1585, pp.50–57.

Yang, Q., Wang, Y. and Zhang, Z. (2003) 'Sanet: a service-agent network for call-centre scheduling', *IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans*, Vol. 33, No. 3, pp.396–406.

Heuristics for the single machine scheduling problem with early and quadratic tardy penalties

Jorge M.S. Valente

LIAAD, Faculdade de Economia
Universidade do Porto,
Rua Dr. Roberto Frias,
Porto 4200-464, Portugal
Fax: +351-22-550-50-50
E-mail: jvalente@fep.up.pt

Abstract: This paper considers the single machine scheduling problem with linear earliness and quadratic tardiness costs, and no machine idle time. Several dispatching heuristics are proposed, and their performance is analysed on a wide range of instances. The heuristics include simple scheduling rules, as well as a procedure that takes advantage of the strengths of these rules. Linear early/quadratic tardy dispatching rules are also considered, as well as a greedy-type procedure. Extensive experiments are performed to determine appropriate values for the parameters required by some of the heuristics. The computational tests show that the best results are given by the linear early/quadratic tardy dispatching rule. This procedure is also quite efficient, and can quickly solve even very large instances.
[Received 15 December 2006; Revised 20 July 2007; Accepted 24 July 2007]

Keywords: heuristics; scheduling; single machine; early penalties; quadratic tardy penalties; no machine idle time; dispatching rules.

Reference to this paper should be made as follows: Valente, J.M.S. (2007) 'Heuristics for the single machine scheduling problem with early and quadratic tardy penalties', *European J. Industrial Engineering*, Vol. 1, No. 4, pp.431–448.

Biographical notes: Jorge M.S. Valente is an Assistant Professor of Operations Research at the Faculty of Economics, University of Porto, Portugal. He received a PhD in Management Science and an MS in Economics from the University of Porto. His current research interests include production scheduling, combinatorial optimisation, heuristic techniques and agent-based computational economics.

1 Introduction

This paper considers a single machine scheduling problem with linear earliness and quadratic tardiness costs, and no machine idle time. Formally, the problem can be stated as follows. A set of n independent jobs $\{J_1, J_2, \ldots, J_n\}$ has to be scheduled on a single machine that can handle at most one job at a time. The machine is assumed to be continuously available from time zero onwards, and preemptions are not allowed. Job J_j, $j = 1, 2, \ldots, n$, requires a processing time p_j and should ideally be completed on its due date d_j. For a given schedule, the earliness and tardiness of J_j are defined as $E_j = \max\{0, d_j - C_j\}$ and

$T_j = \max\{0, C_j - d_j\}$, respectively, where C_j is the completion time of J_j. The objective is to find a schedule that minimises the sum of linear earliness and quadratic tardiness costs $\sum_{j=1}^{n}\left(E_j + T_j^2\right)$, subject to the constraint that no machine idle time is allowed.

Scheduling models with a single processor may appear to arise infrequently in practice. However, this scheduling environment does indeed occur in several activities (for a recent example in the chemical industry, see Wagner et al. (2002)). Moreover, the performance of many production systems is quite often dictated by the quality of the schedules for a single bottleneck machine. Models with a single processor are then most useful in practice for scheduling such a machine. Also, the analysis of single machine problems provides insights that prove valuable for scheduling more complex systems. In fact, multiple processor systems can sometimes be relaxed to a single machine problem, or a sequence of such problems. Furthermore, the solution procedures for some complex systems, such as job shop environments, often require solving single machine subproblems.

Scheduling models with both earliness and tardiness penalties are compatible with the philosophy of Just-In-Time (JIT) production. The JIT production philosophy emphasises producing goods only when they are needed, and therefore takes up the view that both earliness and tardiness should be discouraged. Therefore, an ideal schedule is one in which all jobs are completed exactly on their due dates. Earliness/tardiness problems are also compatible with a recent trend in industry, namely supply chain management. This approach seeks to integrate the flow of materials from the suppliers to the customers, in order to improve the efficiency of the supply chain and to provide a better service to the end-user. The adoption of this approach has caused organisations to view early deliveries, in addition to tardy deliveries, as undesirable.

Linear earliness and quadratic tardiness costs are considered in this paper. On the one hand, early deliveries or early completions of jobs result in unnecessary inventory that ties up cash, as well as space and resources required to maintain and manage the inventory. These costs tend to be proportional to the quantity of inventory held, and therefore a linear penalty is used for early jobs.

On the other hand, late deliveries can result in lost sales and loss of goodwill, as well as disruptions and delays in stages further down the supply chain or production line. A quadratic penalty is considered for the tardy jobs, instead of the more usual linear tardiness or maximum tardiness functions. As described in Sun et al. (1999), the quadratic penalty may be preferable to these two other tardiness measures for the following reasons.

Firstly, the maximum tardiness measure does not distinguish between schedules where tardiness occurs for all jobs, or only one, as long as the maximum tardiness is the same. Secondly, when a linear tardiness is used, it is possible that a single or only a few jobs contribute the majority of the cost, without regard to how the overall tardiness is distributed. In fact, the linear tardiness criterion does not differentiate between sequences where all jobs are only a little tardy, or a single job is extremely late, as long as the total cost is equal. The quadratic penalty overcomes these problems, and provides a more robust performance measure. Moreover, a quadratic tardiness penalty is also appropriate in practice. Indeed, the tardiness of a job is an important attribute of service quality. Also, a customer's dissatisfaction tends to increase quadratically with the tardiness, as proposed in the loss function of Taguchi (1986).

In this paper, it is assumed that no machine idle time is allowed. This assumption is appropriate for many production settings. Indeed, when the capacity of the machine is limited when compared with the demand, the machine must be kept running to meet customers' orders. Idle time must also be avoided for machines with high operating costs,

since the cost of keeping the machine running is then higher than the earliness cost incurred by completing a job before its due date. Furthermore, the assumption of no idle time is also justified when starting a new production run involves high set-up costs or times. Some specific examples of production settings where no idle time assumption is appropriate have been given by Korman (1994) and Landis (1993). More specifically, Korman considers the Pioneer Video Manufacturing (now Deluxe Video Services) disc factory at Carson, California, while Landis analyses the Westvaco envelope plant at Los Angeles.

This problem has been previously considered by Valente (to appear). He proposed a lower bounding procedure based on a relaxation of the job completion times, as well as a branch-and-bound algorithm. The corresponding problem with inserted idle time was studied by Schaller (2004), who presented a timetabling procedure to optimally insert idle time in a given sequence, as well as a branch-and-bound procedure and simple and efficient heuristic algorithms.

The single machine early/tardy problem with linear earliness and tardiness costs $\sum_{j=1}^{n}\left(E_j + T_j\right)$ has also been previously considered by Garey et al. (1988), Kim and Yano (1994) and Schaller (2007). Garey et al. (1988) show that the problem is NP-hard, and propose a timetabling procedure. Kim and Yano (1994) present some properties of optimal solutions, and use them to develop both optimal and heuristic algorithms. Schaller (2007) develops a new lower bound and a new dominance condition, and also shows how to strengthen the lower bounds proposed by Kim and Yano (1994). The computational tests show that the new lower bounds improve the efficiency of a branch-and-bound algorithm.

Valente and Alves (to appear) presented several heuristics for the problem with quadratic earliness and tardiness costs and job-dependent penalties $\sum_{j=1}^{n}\left(h_j E_j^2 + w_j T_j^2\right)$, and no machine idle time. The minimisation of the quadratic lateness $\sum_{j=1}^{n} L_j^2$, where the lateness of J_j is defined as $L_j = C_j - d_j$, has also been previously considered. Gupta and Sen (1983) presented a branch-and-bound algorithm and a heuristic rule for the problem with no idle time. Su and Chang (1998) and Schaller (2002) considered the insertion of idle time, and proposed timetabling procedures and heuristic algorithms. Sen et al. (1995) presented a branch-and-bound algorithm for the weighted problem $\sum_{j=1}^{n} w_j L_j^2$ where idle time is allowed only prior to the start of the first job.

Baker and Scudder (1990) provide an excellent survey of scheduling problems with earliness and tardiness penalties, while Kanet and Sridharan (2000) give a review of scheduling models with inserted idle time that complements our focus on a problem with no machine idle time. Also, a recent survey of multicriteria scheduling problems is given in Hoogeveen (2005). This survey also considers and reviews problems with earliness and tardiness penalties.

In this paper, several dispatching heuristics are proposed, and their performance is analysed on a large set of instances. Three simple but widely used scheduling rules are considered, and an adaptation of one of those rules to a quadratic tardiness objective function is proposed. A heuristic that tries to take advantage of the strengths of the best-performing of these simple rules is also developed. Modified versions of early/tardy dispatching procedures originally proposed for the weighted problem with fully linear costs are also presented. These heuristics have been suitably adapted, in order to take into account the quadratic tardiness cost, as well as the non-weighted nature of the considered problem. Finally, a greedy-type heuristic procedure is also presented. Extensive computational experiments are performed in order to determine appropriate values for the parameters required by some of the heuristics.

The remainder of this paper is organised as follows. The heuristics are described in Section 2. In Section 3, the computational results are presented. Finally, some concluding remarks are provided in Section 4.

2 The heuristics

2.1 Simple linear dispatching rules

Three simple scheduling rules are considered, namely the Longest Processing Time (LPT), Earliest Due Date (EDD) and Shortest Processing Time (SPT) heuristics. The LPT (SPT) rule schedules the jobs in non-increasing (non-decreasing) order of their processing times, while the EDD heuristic sequences the jobs in non-decreasing order of their due dates. These rules only require sorting, and their time complexity is then $O(n \log n)$.

These heuristics are considered for two major reasons. On the one hand, these rules are quite well-known and widely used in many production settings. Therefore, it seems reasonable to include them for comparison purposes.

On the other hand, these rules have some interesting properties for the related problem with a fully linear objective function $\sum_{j=1}^{n} \left(E_j + T_j\right)$. Indeed, the LPT heuristic is particularly adequate to problems where most jobs will be completed early. In fact, the LPT sequence is optimal if it does not contain any tardy jobs. Conversely, the SPT rule is optimal if it generates a schedule with no early jobs. Therefore, this rule is appropriate for problems where most jobs will be tardy. Finally, the EDD heuristic usually performs better than either the LPT or SPT rules when the number of early and tardy jobs is relatively balanced.

Therefore, each one of these simple rules can perform quite well, under the appropriate circumstances, for the problem with a completely linear objective function. For this reason, it seems appropriate to analyse their performance for the problem with a quadratic tardiness cost.

2.2 Simple quadratic dispatching rule

The SPT rule is locally optimal, under the appropriate conditions, for the linear total tardiness problem $\sum_{j=1}^{n} T_j$. In fact, if two adjacent jobs are always tardy, regardless of their order, it is optimal to schedule those jobs in SPT order. In this section, a dispatching rule derived from a local optimality condition for the quadratic tardiness problem $\sum_{j=1}^{n} T_j^2$ is presented. Therefore, this heuristic is an adaptation of the SPT rule to a quadratic objective function. This procedure can also be seen as an adaptation of the WPT_s_j_T dispatching rule proposed by Valente and Alves (to appear) for the problem with quadratic early/tardy costs and job-dependent penalties.

Theorem 1: *Consider any two adjacent jobs J_i and J_j that are always tardy, regardless of their order. In an optimal sequence, all such adjacent pairs of jobs must satisfy the following condition:*

$$\left(\frac{1}{p_i}\right)\left[p_j + 2\left(t + p_i - d_i\right)\right] \geq \left(\frac{1}{p_j}\right)\left[p_i + 2\left(t + p_j - d_j\right)\right]$$

where job J_i immediately precedes job J_j and t is the start time of job J_i.

Proof: The condition can be established using simple interchange arguments. For the sake of brevity, the details are omitted.

Theorem 1 provides a local optimality condition for two adjacent jobs that are always tardy, regardless of their order. The left (right) side of this expression can be interpreted as the priority of job J_i with respect to job J_j (job J_j with respect to job J_i) at time t. A dispatching rule priority index can then be derived by comparing the priority of each job with an average job with processing time $\overline{p}$, where $\overline{p}$ is the average processing time of the remaining unscheduled jobs. Therefore, the priority index of job J_j at time t, denoted as $I_j(t)$, can be calculated as:

$$I_j(t) = \left(\frac{1}{p_j}\right)\left[\overline{p} + 2\max\left(t + p_j - d_j, 0\right)\right]$$

At each iteration, the SPT_s_j dispatching rule selects the unscheduled job with the largest priority. The priority index of the SPT_s_j heuristic includes both a SPT component and a slack (s_j) related component (the slack of job J_j is defined as $s_j = d_j - t - p_j$). When a job is early, the SPT_s_j heuristic is equivalent to the SPT rule, since the priority of job J_j is then equal to $\left(1/p_j\right)\overline{p}$. When a job is tardy, however, the SPT ratio $\left(1/p_j\right)$ is modified by a slack-related component, and the priority increases with the job's tardiness.

The SPT_s_j dispatching heuristic is particularly suited to problems where most jobs will be completed after their due dates, since it is derived from a local optimality condition for tardy jobs. Therefore, the SPT_s_j rule is essentially an adaptation of the SPT heuristic to a quadratic tardiness objective. The time complexity of the SPT_s_j heuristic is $O\left(n^2\right)$.

The following numerical example will be used to illustrate the proposed heuristics. Consider an instance with six jobs, with processing times 8, 10, 6, 4, 3 and 5, and due dates 15, 10, 9, 2, 12 and 17, respectively. In the first iteration, at time $t = 0$, the average processing time $\overline{p}$ of the remaining unscheduled jobs is 6.3333. The priorities of the six available jobs are 0.7917, 0.6333, 1.0556, 2.8533, 2.1111 and 0.9048. Job 4 has the largest priority, and is then selected for processing. After the subsequent iterations are performed, the final sequence 4-5-3-2-1-6, with objective function value 891, is then obtained.

2.3 The CS heuristic

Early computational tests were performed with the LPT, EDD, SPT and SPT_s_j dispatching rules. These tests showed that the SPT_s_j heuristic performed better than the SPT rule. Moreover, the preliminary tests also showed that the best results were given by the EDD (SPT_s_j) heuristic for problems where most jobs were early (tardy). The LPT heuristic was outperformed by the other procedures, even for instances where most jobs were early. In fact, the LPT heuristic focuses on minimising the earliness costs, and completely disregards the tardiness component of the objective function. This means that the LPT sequence may contain a few jobs that are quite tardy, even for instances where most jobs will indeed be early. Since the objective function penalty for tardiness is much higher than the penalty for earliness, the LPT sequence will have a large cost, even though it minimises the earliness component.

In this section, a heuristic (denoted as Critical Slack (CS)) that tries to take advantage of the strengths of the EDD and SPT_s_j rules is presented. At each iteration, the CS heuristic uses one of these two rules to choose the next job. Indeed, the CS procedure selects the rule that is expected to provide the best performance, given the characteristics of the current workload.

The CS heuristic classifies the current workload as non-tardy or tardy. When most jobs have large slacks, the current workload is classified as non-tardy. Conversely, a tardy load consists mainly of jobs with low slacks. At each iteration, the CS heuristic analyses the characteristics of the current set of unscheduled jobs, and classifies the workload as either non-tardy or tardy. Then, the CS procedure selects the EDD (SPT_s_j) rule when the load is non-tardy (tardy).

Two versions of the CS heuristic are considered. These versions share the same basic framework, and differ only in the criterion used to classify the workload as non-tardy or tardy. In both versions, a CS value crit_slack is first calculated. This critical value is calculated as crit_slack = slack_prop $\times$ n_U $\times$ $\overline{p}$, where n_U is the number of unscheduled jobs, and $0 \leq$ slack_prop < 1 is a user-defined parameter. Therefore, the critical slack value is then a proportion slack_prop of the total processing time of the currently unscheduled jobs.

The CS_AS version calculates the average slack $\overline{s}$ of the remaining unscheduled jobs. The workload is then classified as non-tardy (tardy) if $\overline{s} >$ crit_slack ($\overline{s} \leq$ crit_slack). In the CS_LP version, on the other hand, each job is first classified as non-tardy or tardy. A job is said to be non-tardy (tardy) if $s_j >$ crit_slack ($s_j \leq$ crit_slack). The proportion of non-tardy and tardy jobs is then calculated, and the current workload is classified as non-tardy (tardy) if the percentage of non-tardy (tardy) jobs is the largest. The time complexity of both versions of the CS heuristic is $O\left(n^2\right)$.

Consider the numerical example that was previously presented, and assume slack_prop = 0.2. In the first iteration, the critical slack value is equal to 7.6. In the CS_AS version, the average slack $\overline{s}$ of the remaining unscheduled jobs is 4.5. Since $\overline{s} <$ crit_slack, the load is classified as critical, and the SPT_s_j rule is used to select the next job. The priorities of the six unscheduled jobs are 0.7917, 0.6333, 1.0556, 2.8533, 2.1111 and 0.9048. Job 4 is selected for processing, since it has the largest priority. After the subsequent iterations are performed, the final sequence 4-5-3-2-1-6, with objective function value 891, is then obtained. The same final sequence is also generated by the CS_LP version.

2.4 *Linear early/quadratic tardy dispatching rules*

Ow and Morton (1989) developed two early/tardy dispatching rules, denoted as LINET and EXPET, for the fully linear problem with job-dependent earliness and tardiness penalties $\sum_{j=1}^{n}\left(h_j E_j + w_j T_j\right)$ (where h_j and w_j are the job-specific earliness and tardiness penalties, respectively). In this section, adaptations of these rules to the linear earliness and quadratic tardiness problem are proposed. Therefore, the heuristics proposed by Ow and Morton have been suitably modified, to take into account the quadratic tardiness cost, as well as the fact that $h_j = w_j = 1$. The proposed heuristics are denoted by EQTP_LIN and EQTP_EXP, where EQTP stands for Earliness and Quadratic Tardiness Penalties.

Both versions of the EQTP dispatching rule calculate a priority index for each remaining job every time the machine becomes available, and the job with the highest priority is selected to be processed next. Let $I_j(t)$ denote the priority index of job J_j at time t. The EQTP_LIN version uses the following priority index $I_j(t)$:

$$I_j(t) = \begin{cases} \left(1/p_j\right)\left[\overline{p} + 2\left(t + p_j - d_j\right)\right] & \text{if} \quad s_j \leq 0 \\ \left(\overline{p}/p_j\right) - \left(1/p_j\right)\left(\overline{p} + 1\right) s_j / k\overline{p} & \text{if} \quad 0 < s_j < k\overline{p} \\ -\left(1/p_j\right) & \text{otherwise} \end{cases}$$

where k is a lookahead parameter and s_j and $\overline{p}$ are as previously defined.

The EQTP_EXP rule instead uses the following priority index:

$$I_j(t) = \begin{cases} \left(1/p_j\right)\left[\overline{p} + 2\left(t + p_j - d_j\right)\right] & \text{if} \quad s_j \leq 0 \\ \left(\overline{p}/p_j\right)\exp\left[-\left(\overline{p}+1\right)s_j/k\overline{p}\right] & \text{if} \quad 0 < s_j < \left[\overline{p}/\left(\overline{p}+1\right)\right]k\overline{p} \\ \left(1/p_j\right)^{-2}\left[\left(\overline{p}/p_j\right) - \left(1/p_j\right)\left(\overline{p}+1\right)s_j/k\overline{p}\right]^3 & \text{if} \quad \left[\overline{p}/\left(\overline{p}+1\right)\right]k\overline{p} \leq s_j < k\overline{p} \\ -\left(1/p_j\right) & \text{otherwise} \end{cases}$$

where s_j, $\overline{p}$ and k are as previously defined.

The EQTP_LIN and EQTP_EXP dispatching rules assign a priority value of $-\left(1/p_j\right)$ to jobs that are in no danger of becoming tardy ($s_j \geq k\overline{p}$). This assures that two jobs that have large slacks will be scheduled in LPT order. Conversely, the SPT_s_j rule is used to calculate the priority value when a job is on time or late ($s_j \leq 0$). The EQTP_LIN and EQTP_EXP heuristics differ in the calculation of the job priorities for the intermediate values of the job slack. The priority decreases linearly as the job slack increases in the EQTP_LIN dispatching rule, while exponential and cubic functions are instead used in the EQTP_EXP heuristic.

The effectiveness of the EQTP_LIN and EQTP_EXP heuristics depends on the value of the lookahead parameter k. This parameter should reflect the number of competing critical jobs, that is, the number of jobs that may clash each time a sequencing decision is to be made (for details, see Ow and Morton, 1989). In the proposed implementation, the value of k is calculated dynamically at each iteration. Therefore, each time a scheduling decision has to be made, the characteristics of the current workload are used to determine an appropriate value for the lookahead parameter.

The following procedure is used to compute the value of the lookahead parameter k at each iteration. First, a critical slack value crit_slack is calculated, just as previously described for the CS heuristics. Then, each job is classified as critical if $0 < s_j \leq$ crit_slack, and non-critical otherwise. Therefore, a job is considered critical if it is not already tardy ($s_j > 0$), but is about to become tardy ($s_j \leq$ crit_slack). Finally, the lookahead parameter k is set equal to the number of critical jobs. The time complexity of the EQTP_LIN and EQTP_EXP dispatching rules is $O\left(n^2\right)$.

Again, consider the previous numerical example, and assume slack_prop $= 0.25$. In the first iteration, at time $t = 0$, the critical value crit_slack is equal to 9.5. Three jobs have a slack $0 < s_j \leq 9.5$, and the lookahead parameter is then set at $k = 3$. In the EQTP_LIN version, the priorities of the six available jobs are 0.4539, 0.6333, 0.8626, 2.5833, 0.9532 and 0.3534. Job 4 is selected for processing, since it has the largest priority. Once the remaining iterations are performed, the final sequence 4-5-3-2-1-6 (with an objective function value of 891) is obtained. The EQTP_EXP version, on the other hand, generates the sequence 4-2-5-3-1-6, with objective function value 938.

2.5 Greedy heuristic

In this section, a greedy-type procedure, denoted by Greedy, is presented. This heuristic is an adaptation of a procedure originally introduced by Fadlalla et al. (1994) for the mean tardiness problem, and later adapted to other problems (see, for instance, Valente and Alves, 2005; Volgenant and Teerhuis, 1999).

Two different versions of the Greedy heuristic are considered. These versions share the basic framework, and differ only slightly in the calculation of the job priorities. Let c_{xy}, with $x \neq y$, be the combined cost of scheduling jobs J_x and J_y, in this order, in the next two positions in the sequence, that is, c_{xy} is the sum of the costs of J_x and J_y when they are

completed at times $t + p_x$ and $t + p_x + p_y$, respectively. Also, let L be a list with the indexes of the yet unscheduled jobs and $P(j)$ the priority of job J_j. The steps of the Greedy_v1 version are:

Step 1. Initialisation:

Set $t = 0$ and $L = \{1, 2, \ldots, n\}$.

Step 2. Calculate the job priorities:

Set $P(j) = 0$, for all $j \in L$;

For all pairs of jobs $(i, j) \in L$, with $i < j$, do:

Calculate c_{ij} and c_{ji};

If $c_{ij} < c_{ji}$, set $P(i) = P(i) + 1$;

If $c_{ij} > c_{ji}$, set $P(j) = P(j) + 1$;

If $c_{ij} = c_{ji}$, set $P(i) = P(i) + 1$ and $P(j) = P(j) + 1$.

Step 3. Select the next job:

Schedule job l for which $P(l) = \max\left\{P_j;\ j \in L\right\}$;

Set $t = t + p_l$ and $L = L \setminus \{l\}$.

Step 4. Stopping condition:

If $|L| = 1$, stop;

Else, go to step 2.

In the Greedy_v2 version, Step 2 is instead given by:

Step 2. Calculate the job priorities:

Set $P(j) = 0$, for all $j \in L$;

For all pairs of jobs $(i, j) \in L$, with $i < j$, do:

Calculate c_{ij}, c_{ji} and $|c_{ij} - c_{ji}|$;

If $c_{ij} < c_{ji}$

set $P(i) = P(i) + |c_{ij} - c_{ji}|$;

set $P(j) = P(j) - |c_{ij} - c_{ji}|$.

Else

set $P(i) = P(i) - |c_{ij} - c_{ji}|$;

set $P(j) = P(j) + |c_{ij} - c_{ji}|$.

If $c_{ij} < c_{ji}$, it seems better to schedule job J_i in the next position rather than job J_j. Conversely, it seems preferable to schedule job J_j next when $c_{ij} > c_{ji}$. In the Greedy_v1 version, the priority $P(j)$ of job J_j is therefore the number of times job J_j is the preferred job for the next position when it is compared with all the other unscheduled jobs. In the

Greedy_v2 version, for all pairs of jobs (i, j), with $i < j$, the priority of the preferred job is instead increased by $|c_{ij} - c_{ji}|$, while the priority of the other job is decreased by that same value. The time complexity of both versions of the Greedy heuristic is $O(n^3)$.

Again, consider the numerical example. In the first iteration, the priorities of the six unscheduled jobs are equal to 2, 3, 4, 5, 0 and 1, in the Greedy_v1 version. In the Greedy_v2 version, these priorities are instead equal to −184, 4, −2, 392, −54 and −156. In both versions, job 4 has the largest priority, and is selected for processing. After the subsequent iterations are performed, the final sequence 4-3-5-2-1-6 (with objective function value 872) is then obtained, for both versions.

3 Computational results

In this section, the set of test problems used in the computational tests is first presented, and the preliminary computational experiments are described. These experiments are performed to determine appropriate values for the parameters required by the CS and EQTP heuristics. Moreover, the performance of the alternative versions of the CS, EQTP and Greedy heuristics is also analysed in these initial experiments in order to select the best-performing. Finally, the computational results are presented. The heuristic procedures are first compared, and the heuristic results are evaluated against optimum objective function values for some instance sizes.

The instances used in the computational tests are available online at http://www.fep.up.pt/docentes/jvalente/benchmarks.html. The objective function value provided by the EQTP_EXP heuristic, as well as the optimum objective function value (when available), can also be obtained at this address. Throughout this section, and in order to avoid excessively large tables, results will sometimes be presented only for some representative cases.

3.1 Experimental design

The computational tests are performed on a set of problems with 10, 15, 20, 25, 30, 40, 50, 75, 100, 250, 500, 750, 1000, 1500 and 2000 jobs. These problems were randomly generated as follows. For each job J_j, an integer processing time p_j was generated from one of the two uniform distributions [45, 55] and [1, 100], in order to obtain low (L) and high (H) variability, respectively, for the processing time values. For each job J_j, an integer due date d_j was generated from the uniform distribution $[P\,(1 - T - R/2)\,,\, P\,(1 - T + R/2)]$, where P is the sum of the processing times of all jobs, T is the tardiness factor, set at 0.0, 0.2, 0.4, 0.6, 0.8 and 1.0 and R is the range of due dates, set at 0.2, 0.4, 0.6 and 0.8.

For each combination of problem size n, processing time variability (var), T and R, 50 instances were randomly generated. Therefore, a total of 1200 instances were generated for each combination of problem size and processing time variability. All the algorithms were coded in Visual C++ 6.0, and executed on a Pentium IV – 2.8 GHz personal computer. Due to the large computational times that would be required, the Greedy heuristic was only applied to instances with up to 500 jobs.

3.2 Parameter adjustment tests

In this section, the preliminary computational experiments are described. These initial experiments were performed to determine appropriate values for the parameters required

by the CS_AS, CS_LP, EQTP_LIN and EQTP_EXP dispatching rules. The performance of the alternative versions of the CS, EQTP and Greedy heuristics was also analysed, in order to select the best-performing versions. A separate problem set was used to conduct these preliminary experiments. This test set included instances with 25, 50, 100, 250, 500, 1000 and 2000 jobs, and contained five instances for each combination of instance size, processing time variability, T and R. The instances in this smaller test set were generated randomly just as previously described for the full problem set.

Extensive computational tests were performed to determine an appropriate value for the slack_prop parameter used by the CS_AS, CS_LP, EQTP_LIN and EQTP_EXP heuristics. The values $\{0.00, 0.05, 0.10, \ldots, 0.95\}$ were considered, and the objective function value was computed for each slack_prop value and each instance. An analysis of these results showed that a value of slack_prop$= 0.15$ provided the best performance for the CS_AS and CS_LP heuristics. For the EQTP_LIN and EQTP_EXP dispatching rules, the best results were given by slack_prop values in the range $[0.55, 0.95]$. The slack_prop parameter was then set at 0.60 for both the EQTP_LIN and the EQTP_EXP heuristics, since this value consistently provided good results for all instance types.

The slack_prop parameter is instance-dependent, so the best results can be achieved with different values when several instances are considered. Consequently, the values recommended above for this parameter will not provide the best possible performance for all instances. Nevertheless, the computational tests that were performed showed that the chosen values do consistently provide good results across all the instance types.

The slack_prop parameter is also problem- and shop-dependent. Therefore, other parameter values may be more appropriate for problems or shops whose characteristics are different from those of the considered test instances. For instance, production environments that use due date setting methods such as CON, SLACK or TWK may require different slack_prop values. In such environments, experiments should be performed to determine adequate values for the slack_prop parameter. These experiments are likely to be costly and/or difficult to perform, although this task is simplified by the fact that only one parameter has to be fine-tuned.

The performance of the alternative versions of the CS, EQTP and Greedy heuristics was also analysed in these preliminary computational experiments, in order to select the best-performing versions. Therefore, the following three ($h1$ versus $h2$) pairs of alternative heuristic versions were compared: (CS_AS versus CS_LP), (EQTP_EXP versus EQTP_LIN) and (Greedy_v1 versus Greedy_v2).

Table 1 presents the average relative improvement in objective function value provided by the $h1$ heuristic over its $h2$ counterpart (%imp), as well as the percentage number of times version $h1$ performs better ($<$), equal ($=$) or worse ($>$) than version $h2$. The relative improvement given by version $h1$ is calculated as $(h2\_\text{ofv} - h1\_\text{ofv}) / h2\_\text{ofv} \times 100$, where $h2$_ofv and $h1$_ofv are the objective function values of the appropriate heuristic versions.

The performance of the alternative versions of the CS heuristic is quite similar. In fact, the objective function values provided by these alternative versions is generally quite close, particularly for the medium and large size instances. The CS_AS version, however, usually provides better results than its CS_LP counterpart for a slightly larger number of instances.

The EQTP_EXP heuristic performs better than its EQTP_LIN alternative, particularly for instances with a high processing time variability. Indeed, the EQTP_EXP version provides on average a relative improvement in the objective function value of over 3% for instances with a high variability. For low variability instances, however, this improvement is under 1%. Also, the EQTP_EXP version gives better results for a larger number of the test instances.

Table 1 Heuristic version comparison

	n	*Low var*				*High var*			
		%imp	<	=	>	*%imp*	<	=	>
CS_AS	25	0.97	8.33	91.67	0.00	0.33	11.67	81.67	6.67
versus	50	0.12	13.33	82.50	4.17	0.02	16.67	67.50	15.83
CS_LP	100	0.03	9.17	83.33	7.50	−0.03	25.00	62.50	12.50
	250	0.07	6.67	76.67	16.67	0.13	15.83	62.50	21.67
	500	0.00	13.33	70.83	15.83	0.03	28.33	54.17	17.50
	1000	0.00	15.83	73.33	10.83	0.03	21.67	59.17	19.17
	2000	0.00	15.00	67.50	17.50	0.00	25.00	51.67	23.33
EQTP_EXP	25	0.41	43.33	55.83	0.83	3.81	65.83	25.00	9.17
versus	50	0.15	45.00	53.33	1.67	3.86	65.00	25.00	10.00
EQTP_LIN	100	0.10	48.33	49.17	2.50	3.73	65.00	23.33	11.67
	250	0.13	43.33	48.33	8.33	3.51	52.50	23.33	24.17
	500	0.08	40.83	45.83	13.33	3.37	49.17	25.00	25.83
	1000	0.07	42.50	45.83	11.67	3.03	49.17	25.00	25.83
	2000	0.07	40.83	45.00	14.17	2.81	49.17	25.00	25.83
Greedy_v1	25	0.05	54.17	45.83	0.00	−0.50	80.00	9.17	10.83
versus	50	0.24	64.17	34.17	1.67	4.19	85.00	0.83	14.17
Greedy_v2	100	0.01	78.33	20.83	0.83	5.70	83.33	0.00	16.67
	250	0.25	83.33	15.83	0.83	4.51	82.50	0.00	17.50
	500	0.10	87.50	12.50	0.00	5.51	82.50	0.00	17.50

The Greedy_v1 version clearly outperforms its Greedy_v2 alternative when the processing time variability is high. In fact, the Greedy_v1 heuristic provides a relative improvement in the objective function value of about 4–5% (with the exception of the instances with 25 jobs). For instances with low variability, the relative improvement given by the Greedy_v1 heuristic is below 1%. Also, for both low and high variability settings, the Greedy_v1 version gives better results for around 80% of the test instances. In the following sections, results will only be presented for the CS_AS, EQTP_EXP and Greedy_v1 versions.

3.3 Heuristic results

In this section, the computational results for the heuristic procedures are presented. Table 2 gives the average objective function value (ofv) for each heuristic, as well as the percentage number of times a heuristic provides the best result when compared with the other heuristics (%best). The average objective function values are calculated relative to the EQTP_EXP heuristic, and are therefore presented as index numbers. More precisely, these values are calculated as heur_ofv/eqtp_exp_ofv × 100, where heur_ofv and eqtp_exp_ofv are the average objective function values of the appropriate heuristic and the EQTP_EXP dispatching rule, respectively.

The best results are given by the EQTP_EXP dispatching rule, closely followed by the CS_AS procedure. In fact, the EQTP_EXP heuristic not only provides the lowest average objective function value, but also gives the best results for a large percentage of the instances

(particularly for the largest instances, or when the variability of the processing times is low). The CS_AS procedure also performs quite well, providing an average objective function value that is quite close to the results given by the EQTP_EXP heuristic.

Table 2 Heuristic results

Var	*Heur*	*n = 25*		*n = 100*		*n = 500*		*n = 2000*	
		ofv	*%best*	*ofv*	*%best*	*ofv*	*%best*	*ofv*	*%best*
L	LPT	156.92	4.50	162.79	1.33	163.93	0.83	164.47	0.00
	EDD	100.81	8.67	100.93	3.25	100.95	2.75	100.95	3.67
	SPT	140.70	0.00	144.86	0.00	145.82	0.00	145.92	0.00
	SPT_sj	102.05	15.92	101.52	15.50	101.31	29.08	101.25	31.83
	CS_AS	100.10	32.67	100.05	23.42	100.05	27.83	100.04	24.58
	EQTP_EXP	100.00	65.83	100.00	71.25	100.00	86.58	100.00	91.75
	Greedy_v1	101.57	49.25	101.89	38.92	101.96	36.25	–	–
H	LPT	325.02	0.08	364.43	0.00	377.41	0.00	380.17	0.00
	EDD	134.93	4.83	139.77	0.75	141.89	1.67	142.08	1.83
	SPT	129.45	0.00	133.08	0.00	134.78	0.00	134.82	0.00
	SPT_sj	102.36	5.17	101.39	0.92	101.08	12.75	100.99	25.00
	CS_AS	100.13	20.92	100.25	10.42	100.29	18.08	100.29	23.50
	EQTP_EXP	100.00	43.75	100.00	55.25	100.00	53.92	100.00	91.83
	Greedy_v1	102.44	45.33	103.63	35.67	103.92	49.92	–	–

The SPT_s_j and the Greedy_v1 heuristics also provide good results. The Greedy_v1 procedure gives an average objective function value that is about 2–3% worse than the EQTP_EXP heuristic, but it nevertheless provides the best results for a significant number of instances. The SPT_s_j procedure performs well for medium and large instances, since it provides an average objective function value that is about 1% worse than the results given by the EQTP_EXP dispatching rule.

The simple LPT and SPT rules perform rather poorly, giving results that are substantially worse than those of the other heuristics. The linear SPT rule was clearly outperformed by its SPT_s_j quadratic counterpart, meaning that the modifications that were introduced in this linear rule, in order to adapt it to a quadratic objective function, have indeed significantly improved its performance. Therefore, it is certainly important to address the quadratic tardiness component of the cost function and develop a specific procedure, instead of simply using a heuristic appropriate for a linear tardiness objective function.

The EDD rule does provide an average objective function value that is quite close to the EQTP_EXP results for instances with a low processing time variability, but its performance is quite poor when the variability is high. The CS_AS heuristic performs considerably better than either of the EDD and SPT_s_j rules, particularly when the variability of the processing times is high. Consequently, a considerable performance improvement can be achieved by selectively using these two simple heuristics (i.e. by choosing at each iteration the rule that is expected to perform better, given the characteristics of the current job load).

Table 3 presents the effect of the T and R parameters on the average objective function value (calculated relative to the EQTP_EXP heuristic). This Table gives results for the best heuristics (the EQTP_EXP is omitted, since its values would all be equal

to 100) and for instances with 100 jobs. The SPT_s_j heuristic provides an average objective function value that is quite close to the results given by the EQTP_EXP dispatching rule for instances with a large tardiness factor T. The SPT_s_j rule, however, performs considerably worse when the tardiness factor is low. This result is to be expected, since the SPT_s_j heuristic is particularly well suited to problems where most jobs will be completed after their due dates, since it is derived from a local optimality condition for tardy jobs. When the tardiness factor T is high, most jobs will be tardy and the SPT_s_j rule indeed performs well. For low values of T, on the other hand, the proportion of tardy jobs is lower, and the performance of the SPT_s_j heuristic correspondingly deteriorates.

Table 3 Objective function value, relative to the EQTP_EXP heuristic, for instances with 100 jobs

Heur	*T*	*Low var*				*High var*			
		R = 0.2	*R = 0.4*	*R = 0.6*	*R = 0.8*	*R = 0.2*	*R = 0.4*	*R = 0.6*	*R = 0.8*
SPT_s_j	0.0	108.50	111.19	114.96	116.03	204.42	202.17	215.09	207.42
	0.2	160.11	376.49	315.94	184.66	131.46	794.73	565.84	467.52
	0.4	123.33	169.45	277.28	1325.44	119.35	156.01	280.71	1876.76
	0.6	108.34	115.34	115.49	101.34	104.68	112.71	115.49	109.19
	0.8	101.59	100.00	100.00	100.00	100.53	100.49	100.05	100.02
	1.0	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
CS_AS	0.0	102.82	102.78	102.73	102.68	135.53	133.37	131.02	126.46
	0.2	79.42	100.73	104.17	103.68	62.01	126.35	135.65	130.47
	0.4	100.57	100.04	100.01	100.13	99.97	100.41	102.19	109.45
	0.6	100.47	100.00	100.01	100.07	100.15	100.19	101.06	106.77
	0.8	100.42	100.00	100.01	100.02	100.15	100.11	100.60	101.93
	1.0	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
Greedy_v1	0.0	100.35	100.61	100.85	101.82	100.35	100.98	101.82	103.93
	0.2	179.73	331.87	160.98	125.95	242.03	487.37	254.17	169.94
	0.4	138.45	175.98	236.52	986.42	183.55	217.47	337.38	1351.64
	0.6	114.44	117.69	114.65	100.50	132.66	132.32	125.99	101.88
	0.8	102.45	100.00	100.00	100.00	105.71	99.98	99.99	99.99
	1.0	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00

The CS_AS heuristic is quite close to the EQTP_EXP dispatching rule for $T \geq 0.4$. However, the CS_AS heuristic is clearly outperformed when most jobs are early, particularly when the variability of the processing times is high. In fact, when the processing time variability is high (low), the CS_AS heuristic provides an average objective function value that is about 30% (3%) worse than the results given by the EQTP_EXP procedure when $T = 0.0$ or $T = 0.2$ and $R \geq 0.4$. The Greedy_v1 heuristic performs well for instances where most jobs are early ($T = 0.0$) or tardy ($T = 1.0$). The performance of the Greedy_v1 procedure deteriorates when the tardiness factor T takes on intermediate values. Therefore,

the Greedy_v1 procedure is less effective when there is a greater balance between the number of early and tardy jobs.

The heuristic runtimes (in sec) are presented in Table 4. The Greedy_v1 heuristic is computationally demanding, and therefore can only be used for small and medium size instances. The other heuristic procedures are quite fast, even for large instances. The simple LPT, EDD and SPT rules are the most efficient, since they only require sorting, which can be performed in $O(n \log n)$ time. The SPT_s_j and CS_AS procedures are also quite efficient, even with their higher $O(n^2)$ time complexity. The EQTP_EXP dispatching rule, even though it also requires $O(n^2)$ time, is more computationally demanding. Nevertheless, this heuristic is still extremely fast, being capable of solving even quite large instances with 2000 jobs in less than 0.3 sec on a personal computer. The EQTP_EXP is then the heuristic procedure of choice, since it not only provides the best results, but is also computationally efficient.

Table 4 Heuristic runtimes (in sec)

Var	*Heur*	*n = 100*	*n = 250*	*n = 500*	*n = 1000*	*n = 1500*	*n = 2000*
L	LPT	0.0000	0.0001	0.0002	0.0004	0.0006	0.0006
	EDD	0.0000	0.0000	0.0001	0.0002	0.0004	0.0004
	SPT	0.0000	0.0001	0.0002	0.0003	0.0004	0.0006
	SPT_sj	0.0001	0.0009	0.0030	0.0123	0.0267	0.0477
	CS_AS	0.0002	0.0009	0.0036	0.0139	0.0312	0.0544
	EQTP_EXP	0.0005	0.0041	0.0153	0.0600	0.1360	0.2412
	Greedy_v1	0.1887	2.9302	23.3965	–	–	–
H	LPT	0.0000	0.0001	0.0001	0.0004	0.0006	0.0008
	EDD	0.0000	0.0001	0.0001	0.0002	0.0003	0.0006
	SPT	0.0001	0.0001	0.0001	0.0003	0.0004	0.0007
	SPT_sj	0.0001	0.0007	0.0030	0.0123	0.0264	0.0464
	CS_AS	0.0001	0.0010	0.0037	0.0144	0.0326	0.0593
	EQTP_EXP	0.0007	0.0042	0.0160	0.0639	0.1433	0.2548
	Greedy_v1	0.1858	2.8757	22.9192	–	–	–

3.4 Comparison with optimum results

In this section, the heuristic results are compared with the optimum objective function values for instances with up to 20 jobs. The optimum results were obtained using the branch-and-bound algorithm proposed by Valente (to appear). Table 5 presents the average of the relative deviations from the optimum (%dev), calculated as $(H - O)/O \times 100$, where H and O are the heuristic and the optimum objective function values, respectively. The percentage number of times each heuristic generates an optimum schedule (%opt) is also given.

From Table 5, it can be seen that the heuristics are much closer to the optimum for instances with a low processing time variability, with the exception of the SPT procedure. The EQTP_EXP and CS_AS heuristics perform quite well for instances with a low variability, giving results that are 1–2% above the optimum. The other heuristics are far from the optimum, with the exception of the EDD rule. When the variability of the processing times is high, however, even the best-performing EQTP_EXP and CS_AS

heuristics are 10–20% above the optimum (though the deviation from the optimum decreases with the instance size for the EQTP_EXP procedure). For the Greedy_v1 heuristic, the average deviation from the optimum is quite large for instances with a high processing time range. However, this heuristic provides an optimum solution for a large number of instances. In fact, for some problem sizes, particularly when the variability is low, the Greedy_v1 procedure generates an optimum solution for over 50% of the instances.

Table 5 Comparison with optimum objective function values

Var	*Heur*	*n = 10*		*n = 15*		*n = 20*	
		%dev	*%opt*	*%dev*	*%opt*	*%dev*	*%opt*
L	LPT	641.48	7.08	1047.29	6.00	1404.28	4.33
	EDD	1.50	9.17	1.71	2.00	1.86	0.92
	SPT	586.77	0.00	835.35	0.00	1088.79	0.00
	SPT_sj	128.88	23.67	119.50	21.17	98.89	18.17
	CS_AS	1.61	30.33	1.53	24.25	1.66	20.58
	EQTP_EXP	1.78	45.58	2.14	34.50	1.83	28.17
	Greedy_v1	38.79	62.17	55.41	54.67	54.59	48.17
H	LPT	1659.80	2.17	2786.84	0.50	3755.46	0.50
	EDD	32.07	0.33	36.33	0.00	37.32	0.00
	SPT	589.29	0.00	810.08	0.00	1051.87	0.00
	SPT_sj	195.66	7.17	230.05	5.25	224.85	3.50
	CS_AS	13.48	8.33	14.47	5.75	14.84	3.42
	EQTP_EXP	22.14	22.25	16.45	11.92	11.96	8.67
	Greedy_v1	40.18	52.67	68.90	35.83	92.85	30.33

The effect of the T and R parameters on the relative deviation from the optimum is presented in Table 6. This table gives results for the best heuristics and for instances with 20 jobs. The SPT_s_j heuristic is quite close to the optimum for instances with a large tardiness factor T. The average relative deviation from the optimum, however, is substantially higher when the tardiness factor is low. This is to be expected, since the SPT_s_j heuristic is particularly suited to problems where most jobs will be tardy. When the tardiness factor T is high, most jobs will indeed be tardy and the SPT_s_j rule is then quite close to optimal. For low values of T, on the other hand, the number of tardy jobs is lower, and the average deviation of the SPT_s_j heuristic from the optimum correspondingly increases.

The CS_AS dispatching rule is quite close to the optimum when the tardiness factor is greater than or equal to 0.6. The performance of the CS_AS heuristic, however, is clearly inferior when most jobs are early, particularly when the variability is high. In fact, the CS_AS heuristic is about 30–50% above the optimum for high variability instances with $T = 0.0$ or $T = 0.2$. The effect of the T and R parameters on the average relative deviation from the optimum is similar for the EQTP_EXP and the Greedy_v1 heuristics. These procedures are much closer to the optimum when nearly all jobs are early ($T = 0.0$) or when there is a larger proportion of tardy jobs ($T \geq 0.6$). The performance of these dispatching rules deteriorates, particularly for the Greedy_v1 heuristic, when $T = 0.2$ or $T = 0.4$.

Table 6 Relative deviation from the optimum for instances with 20 jobs

Heur	*T*	*Low var*				*High var*			
		R = 0.2	*R = 0.4*	*R = 0.6*	*R = 0.8*	*R = 0.2*	*R = 0.4*	*R = 0.6*	*R = 0.8*
SPT_s_j	0.0	15.57	24.92	22.35	70.52	131.89	141.19	191.15	194.88
	0.2	103.34	312.48	261.33	382.97	125.65	620.08	1546.85	729.72
	0.4	22.38	62.02	168.47	873.41	22.86	64.63	185.59	1377.21
	0.6	8.82	16.02	17.08	9.92	5.60	12.68	17.66	22.90
	0.8	1.50	0.14	0.00	0.00	1.45	1.81	1.10	1.08
	1.0	0.00	0.00	0.00	0.00	0.05	0.07	0.09	0.11
CS_AS	0.0	2.90	2.98	3.00	2.57	37.15	33.52	36.63	31.75
	0.2	3.59	4.47	5.13	4.67	13.20	54.29	53.44	46.79
	0.4	1.35	0.66	1.30	4.43	2.73	3.51	8.13	19.82
	0.6	1.11	0.33	0.01	0.10	1.94	1.45	1.75	5.18
	0.8	1.10	0.08	0.00	0.01	1.22	1.31	0.91	1.20
	1.0	0.00	0.00	0.00	0.00	0.05	0.07	0.09	0.11
EQTP_EXP	0.0	0.19	0.09	0.08	0.11	0.66	1.64	2.65	2.64
	0.2	17.05	13.56	3.98	2.23	60.07	91.36	26.80	20.54
	0.4	0.10	0.13	0.42	5.92	6.34	4.57	13.49	44.53
	0.6	0.02	0.02	0.01	0.01	3.97	1.38	1.23	1.88
	0.8	0.01	0.01	0.00	0.00	1.68	0.82	0.30	0.25
	1.0	0.00	0.00	0.00	0.00	0.03	0.05	0.05	0.06
Greedy_v1	0.0	0.75	1.41	2.90	5.44	3.33	8.85	16.18	19.08
	0.2	74.84	173.11	170.09	198.99	127.37	265.48	287.64	371.61
	0.4	28.41	61.53	108.97	453.53	60.78	70.64	139.77	814.20
	0.6	9.88	10.13	8.51	0.91	18.66	17.10	5.51	1.05
	0.8	0.68	0.00	0.00	0.00	0.96	0.13	0.06	0.04
	1.0	0.00	0.00	0.00	0.00	0.00	0.00	0.02	0.01

4 Conclusion

This paper considered the single machine scheduling problem with linear earliness and quadratic tardiness costs, and no machine idle time. Several dispatching heuristics were proposed, and their performance was analysed on a wide range of instances. The heuristics included simple scheduling rules, as well as a procedure that takes advantage of the strengths of these rules. Linear early/quadratic tardy dispatching rules were also considered, as well as a greedy-type procedure. Extensive computational experiments were performed to determine adequate values for the parameters required by some of the heuristics.

Dispatching heuristics are widely used in practice and, in fact, most real scheduling systems are either based on dispatching rules, or at least use them to some degree. Also, dispatching rules are often the only heuristic approach capable of generating solutions, within reasonable computation times, for large instances. Additionally, dispatching rules

are used by other heuristic procedures, for example, they are often used to generate the initial sequence required by local search or metaheuristic algorithms.

The best results were given by the EQTP_EXP dispatching rule. This heuristic provided the lowest average objective function values, and also obtained the best results for a large percentage of the instances. The performance of the EQTP_EXP procedure was quite good for instances with a low processing time range, since it provided results that are about 1–2% above the optimum. For instances with a high range, however, the deviation from the optimum exceeded 10%, though it decreased as the instance size increased.

The Greedy_v1 heuristic is computationally demanding, and therefore can only be used for small and medium size instances. The other heuristic procedures, however, were quite fast, and are capable of solving even very large instances in less than one second on a personal computer. The EQTP_EXP dispatching rule is then the heuristic procedure of choice, since it not only provided the best results, but is also computationally efficient.

The best of the proposed heuristics perform quite adequately for the problem with no idle time. These heuristics, however, might also be useful for the problem with inserted idle time. Indeed, the procedure developed by Schaller (2004) can be applied to optimally insert idle time in the sequences generated by the proposed heuristics. Therefore, investigating the performance of these heuristics for the problem with inserted idle time certainly seems an interesting possibility for future research.

Acknowledgements

The author would like to thank the anonymous referees for several, and most useful, comments and suggestions that have been used to improve this paper.

References

Baker, K.R. and Scudder, G.D. (1990) 'Sequencing with earliness and tardiness penalties: a review', *Operations Research*, Vol. 38, pp.22–36.

Fadlalla, A., Evans, J.R. and Levy, M.S. (1994) 'A greedy heuristic for the mean tardiness sequencing problem', *Computers and Operations Research*, Vol. 21, pp.329–336.

Garey, M.R., Tarjan, R.E. and Wilfong, G.T. (1988) 'One-processor scheduling with symmetric earliness and tardiness penalties', *Mathematics of Operations Research*, Vol. 13, pp.330–348.

Gupta, S.K. and Sen, T. (1983) 'Minimizing a quadratic function of job lateness on a single machine', *Engineering Costs and Production Economics*, Vol. 7, pp.187–194.

Hoogeveen, H. (2005) 'Multicriteria scheduling', *European Journal of Operational Research*, Vol. 167, pp.592–623.

Kanet, J.J. and Sridharan, V. (2000) 'Scheduling with inserted idle time: problem taxonomy and literature review', *Operations Research*, Vol. 48, pp.99–110.

Kim, Y.D. and Yano, C.A. (1994) 'Minimizing mean tardiness and earliness in single-machine scheduling problems with unequal due dates', *Naval Research Logistics*, Vol. 41, pp.913–933.

Korman, K. (1994) 'A pressing matter', *Video*, pp.46–50.

Landis, K. (1993) *Group Technology and Cellular Manufacturing in the Westvaco Los Angeles VH Department*, Project report in IOM 581, School of Business, University of Southern California.

Ow, P.S. and Morton, T.E. (1989) 'The single machine early/tardy problem', *Management Science*, Vol. 35, pp.177–191.

Schaller, J. (2002) 'Minimizing the sum of squares lateness on a single machine', *European Journal of Operational Research*, Vol. 143, pp.64–79.

Schaller, J. (2004) 'Single machine scheduling with early and quadratic tardy penalties', *Computers and Industrial Engineering*, Vol. 46, pp.511–532.

Schaller, J. (2007) 'A comparison of lower bounds for the single-machine early/tardy problem', *Computers and Operations Research*, Vol. 34, pp.2279–2292.

Sen, T., Dileepan, P. and Lind, M.R. (1995) 'Minimizing a weighted quadratic function of job lateness in the single machine system', *International Journal of Production Economics*, Vol. 42, pp.237–243.

Su, L-H. and Chang, P-C. (1998) 'A heuristic to minimize a quadratic function of job lateness on a single machine', *International Journal of Production Economics*, Vol. 55, pp.169–175.

Sun, X., Noble, J.S. and Klein, C.M. (1999) 'Single-machine scheduling with sequence dependent setup to minimize total weighted squared tardiness', *IIE Transactions*, Vol. 31, pp.113–124.

Taguchi, G. (1986) *Introduction to Quality Engineering*, Asian Productivity Organization, Tokyo, Japan.

Valente, J.M.S. (to appear) 'An exact approach for the single machine scheduling problem with linear early and quadratic tardy penalties', *Asia-Pacific Journal of Operational Research*.

Valente, J.M.S. and Alves, R.A.F.S. (2005) 'Improved heuristics for the early/tardy scheduling problem with no idle time', *Computers and Operations Research*, Vol. 32, pp.557–569.

Valente, J.M.S. and Alves, R.A.F.S. (to appear) 'Heuristics for the single machine scheduling problem with quadratic earliness and tardiness penalties', *Computers and Operations Research*.

Volgenant, A. and Teerhuis, E. (1999) 'Improved heuristics for the n-job single-machine weighted tardiness problem', *Computers and Operations Research*, Vol. 26, pp.35–44.

Wagner, B.J., Davis, D.J. and Kher, H. (2002) 'The production of several items in a single facility with linearly changing demand rates', *Decision Sciences*, Vol. 33, pp.317–346.

An analysis of the stochastic behaviour for shift conversion system

K. Senthamarai Kannan*

Department of Statistics,
Manonmaniam Sundaranar University,
Tirunelveli, Tamilnadu, India
E-mail: senkannan2002@yahoo.com
*Corresponding author

C. Vijayalakshmi

Department of Mathematics,
Sathyabama University,
Chennai, Tamilnadu, India
E-mail: vijusesha2002@yahoo.co.in

Abstract: This paper deals with the analysis of the stochastic behaviour and maintenance planning of shift conversion system with four subsystems A, B, C, D with possible states good and failure. Kumar and Pandey have discussed the process design for crystallisation system along with the maintenance planning and resource allocation. Subsequently, Kumar et al. have discussed about the steady state behaviour, maintenance planning along with the probabilistic analysis of desulphurisation system. Based on the Markov graph, differential equations are derived and the shift conversion system is analysed. The steady states of repair in each subsystem are obtained for the improvement in the process design which results to minimum failure.
[Received on 11 January 2007; Revised 26 April 2007; Accepted 24 July 2007]

Keywords: shift conversion; failure rate; repair rate; Markov graph; availability; desulphurisation system.

Reference to this paper should be made as follows: Kannan, K.S. and Vijayalakshmi, C. (2007) 'An analysis of the stochastic behaviour for shift conversion system', *European J. Industrial Engineering*, Vol. 1, No. 4, pp.449–461.

Biographical notes: K. Senthamarai Kannan is currently working as a Professor in Statistics, at the Manonmaniam Sundaranar University, Tirunelveli, Tamilnadu. He has more than 18 years of teaching experience at post-graduate level. He has published more than 30 research papers in international and national journals and authored four books. He has visited Turkey, Singapore and Malaysia. He has been awarded TNSCST Young Scientist Fellowship and SERC Visiting Fellowship. His area of specialisation is 'Stochastic Processes and their Applications'. His other research interests include stochastic modelling in the analysis of birth intervals in human fertility, bio-informatics, data mining and precipitation analysis.

C. Vijayalakshmi is currently working as an Assistant Professor in Mathematics, at the Sathyabama University, Chennai, Tamilnadu. She has more than ten years of teaching experience at graduate and post-graduate level. She has published more than ten research papers in international and national journals and authored three books. She had received Vijayshree Award and Best Teachers award for the academic excellence. Her area of specialisation is 'Stochastic Processes and their Applications'. Her other research interests include bio-informatics, data mining.

1 Introduction

A probabilistic analysis of the system, under the given operative conditions, is useful in modifying the design for the system, so that the system runs with minimum failure. The main functionary part of the ammonia production process is defined as the shift conversion process. Process design for reliable operation has been already discussed by Lieberman (1973). Kumar et al. (1991) have discussed about the behaviour analysis of urea decomposition system with general repair policy in fertiliser industry and have analysed about the process design for crystallisation system in the Urea Fertiliser Industry. Kumar et al. (1993) have discussed the process design for crystallisation system along with the maintenance planning and resource allocation. Subsequently, Sunand Kumar et al. (1996,1997) have discussed about the steady state behaviour, maintenance planning along with the probabilistic analysis of desulphurisation system. In a urea fertiliser industry, liquid ammonia and carbon dioxide (CO_2) are two by- products obtained from ammonia production plant that produces urea. The ammonia production process consists of various processes namely, air separation, shell gasification, carbon recovery, desulphurisation, nitrogen washing and ammonia synthesis process along with the refrigeration process.

This system consists of four subsystems A, B, C and D arranged in series and they are as follows:

1 *Subsystem (A)*: this consists of two units in a series (humidifier and dehumidifier pump). A cold standby pump is provided with each unit. Complete failure occurs only when the standby pump of a unit fails.

2 *Subsystem (B)*: this subsystem B consists of four condensate pumps, two working in parallel while two remain in cold standby. Complete failure occurs only when three pumps or more are in failed state and this is due to either simultaneous failure or delay in repair.

3 *Subsystem (C)*: four feed gas heater (two working in series and the other two in cold standby). Complete failure occurs only when three heaters or more are in failed state (due to either simultaneous failure or delay in repair).

4 *Subsystem (D)*: consists of nine units in series (humidifier, shift converter, dehumidifier, feed water heater, absorption refrigerator, separators, low pressure boiler, economiser and cooler). Failure of any unit will cause complete failure of the system.

In Section 2, the differential equations are obtained based on the Markov graph. The failure and repair rates are considered to be constant and formulation of the problem is carried out by using probability considerations in Section 3. In Section 4 the

operational measures are being discussed. The states of repair in each subsystem are obtained and the effect of each working unit on the system reliability analysis on availability is computed for minimum failure in Sections 5 and 6.

2 System description

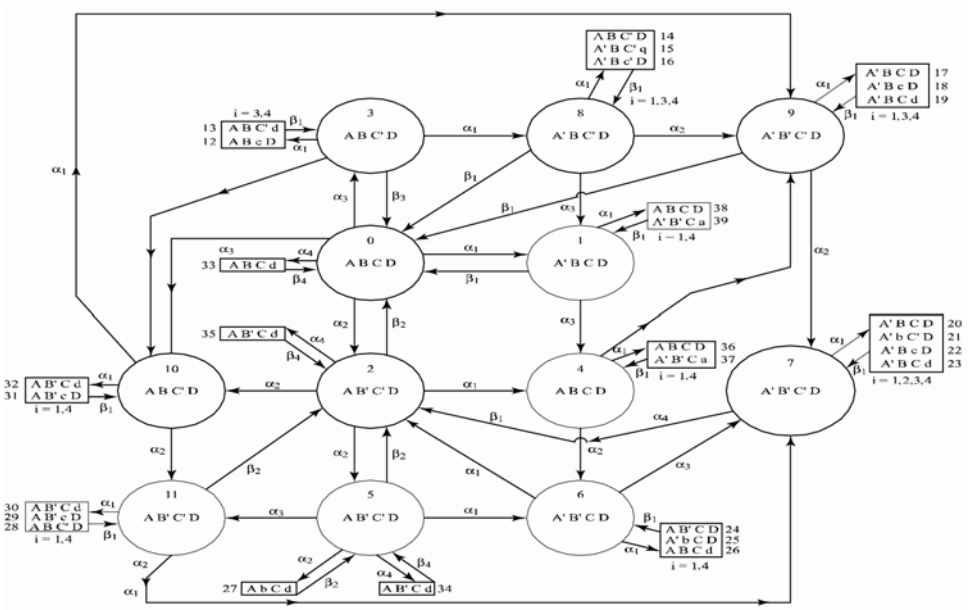
The various states of the subsystems are given in Table 1 and the Markov Graph of the system is shown in Figure 1.

Table 1 Steady states of the subsystem

State	*Subsystem*			
	A	*B*	*C*	*D*
Full capacity working (without standby unit)	A	B	C	D
Full capacity working (with standby unit)	A′	B′B″	C′	–
Failed state	a	b	c	D
Failure rate	α_1	α_2	α_3	α_4
Repair rate	β_1	β_2	β_3	β_4

Let P_0 be the probability of full capacity working without any standby unit and P_i, $i = 1 - 11$, be the probability of full capacity working with standby unit. $P_j, j = 12 - 39$ is considered as the probability of failed state.

Figure 1 Markov Graph of the system



3 Nature and behaviour of each subsystems

Lenz (1970) has created the reliability design for the process plant and the reliability in a larger refinery has been briefly discussed by Mcfalter (1972). Lieberman (1973) has made the process design for reliable operation. Kumar et al. (1999, 2000) have analysed the maintenance management for ammonia synthesis system in a urea fertiliser plant and a probabilistic analysis of desulphurisation system in a urea fertiliser plant was discussed. Vijayalakshmi and Senthamarai Kannan (2005) have discussed about the bulk queue G/M/1 in inventory control and a detailed analysis was made on Stochastic behaviour for the reliability and cost benefit analysis in a petro-chemical industry by Vijayalakshmi et al. (2007). However, in this section the operating behaviour of each unit is depicted and the remedial measures are obtained (Table 2).

Table 2 Behaviour and remedial measures of the subsystem

Behaviour	*Effect/Occurrence*	*Remedy*
Subsystems (A, B)		
Major failure	Rare	Replacement is done within short time
Minor failure	Often	Unskilled worker rectifies/input of raw parameters
Loss in operating capacity	Small	Bypass the system for short duration
Performance degradation	Rate	Replace the part
Significant performance	Yes	Check the pump timing
Inconvenience in operation	Yes	Control leakage and lubricate the bearings
Subsystem (C)		
Major failure	Rate	Replacement is the only solution
Minor failure	Small	Unskilled worker rectifies
Loss in operating capacity	Yes	Bypass the unit
Performance degradation	Rate	Bypass the unit
Significant performance	Yes	Check the heating unit
Inconvenience in operation	Rare	Replace the unit
Subsystem (D)		
Major failure	Rate	Shut the plant and maintain it by the skilled worker
Minor failure	Often	Highly skilled worker rectifies
Loss in operating capacity	None	–
Performance degradation	Possible	Highly skilled workers repair the parts
Significant performance	Yes	Highly skilled workers repair the parts
Inconvenience in operation	Yes	Control the water level, temperature and pressure

4 Operational measures for subsystems

For the operation of the subsystem with the maximum availability under the existing facilities, the following measures are adopted, so that the failure rate of each unit is minimised.

1 reducing the delay in communicating information to the section manager

2 providing the instruction manuals and support software

3 reducing the delay in starting the process repair

4 proper training to the operator in order to operate the equipment

5 availability of the required tools

6 studying the following points about failure and repair

   a cause of rapid failure of a particular part of each equipment

   b studying whether failure reduces the availability below an acceptable level and whether failure function is understandable by the operator during routine operation

   c verifying the hidden function during routine checkout

   d assessing the feasibility of repair during operation

   e correcting the failure by adjusting or aligning any one part

   f replacing the part or unit so that the failure is corrected

   g feasibility and effectiveness of the repair unit.

Based on the above operational measures the following decisions are made, such as fixing the priority, planning the maintenance strategy and reducing the delays in starting the maintenance work.

5 Mathematical modelling

Priel (1974) has briefly discussed about the systematic maintenance in which the failure, repair rates are constant and statistically independent.

The following assumptions are made for constructing the mathematical model.

1 Failure and repair rates are constant and statistically independent.

2 Based on the performance, a repaired unit is as good as new one.

3 Repairmen are always available and their costs are not considered.

4 Each subsystem has a separate repair facility. There is no waiting time for repair of any unit.

5 Service includes repair or replacement of the unit.

6 The standby units are of the same nature and capacity is as that of the active unit.

7 There is no simultaneous failure among the subsystems.

8 Repair rate of subsystem A is more than the repair rate of subsystem B ($\beta_1 > \beta_2$).

9 Repair rate of subsystem C is more than the repair rate of subsystem A ($\beta_3 > \beta_1$).

Based on the Markov graph (Figure 1) the differential equation is given as,

$$P'_X(t) + Y(r)P_X(t) = Z(t) \tag{1}$$

with initial condition $P_0\ (0) = 1$, zero otherwise

For Equation (1), when

$$X = 1, Z(t) = \beta_1 P_{38}(t) + \beta_4 P_{39}(t) + \alpha_1 P_0(t)$$

$$Y(r) = \sum_{i=1}^{4} \alpha_j + \beta_1$$

$$X = 2, Z(t) = \sum_{i=2}^{3} \alpha_i P_{i-5}(t) + \beta_1 P_5(t) + \beta_3 P_{11}(t) + \beta_4 P_{35}(t) + \alpha_2 P_0(t)$$

$$Y(r) = \sum_{i=1}^{4} \alpha_i + \beta_2$$

$$X = 3, Z(t) = \alpha_3 P_0(t) + \sum_{i=3}^{4} \beta_i P_i(t)$$

$$Y(r) = \sum_{i=1}^{4} \alpha_i + \beta_3$$

$$X = 4, Z(t) = \beta_1 P_{36}(t) + \beta_4 P_{37}(t) + \alpha_1 P_2(t) + \alpha_2 P_1(t)$$

$$Y(r) = \sum_{i=1}^{4} \alpha_i + \beta_1$$

$$X = 5, Z(t) = \sum_{i=1}^{2} \beta_i P_{i+14}(t) + \beta_4 P_{26}(t) + \alpha_1 P_6(t) + \alpha_2 P_4(t)$$

$$Y(r) = \sum_{i=1}^{4} \alpha_i + \beta_1$$

$$X = 6, Z(t) = \beta_2 P_{27}(t) + \beta_4 P_{34}(t) + \alpha_2 P_2(t)$$

$$Y(r) = \sum_{i=1}^{2} \alpha_i + \beta_2$$

$$X = 7, Z(t) = \sum_{i=1}^{4} \beta_i P_{i+10}(t) + \alpha_1 P_{11}(t) + \alpha_2 P_9(t) + \alpha_3 P_5(t)$$

$$Y(r) = \sum_{i=1}^{4} \alpha_i + \beta_3$$

$$X = 8, Z(t) = \beta_1 P_{14}(t) + \sum_{i=3}^{4} \beta_i P_{i+3}(t) + \alpha_3 P_1(t) + \alpha_1 P_3(t)$$

$$Y(r) = \sum_{i=1}^{4} \alpha_i + \beta_3$$

$$X = 9, Z(t) = \beta_1 P_{17}(t) + \sum_{i=3}^{4} \beta_i P_{i+6}(t) + \alpha_3 P_4(t) + \alpha_2 P_8(t) + \alpha_1 P_{10}(t)$$

$$Y(r) = \sum_{i=1}^{4} \alpha_i + \beta_3$$

$$X = 10, Z(t) = \sum_{i=3}^{4} \beta_i P_{i+19}(t) + \alpha_2 P_3(t) + \alpha_3 P_2(t)$$

$$Y(r) = \sum_{i=1}^{4} \alpha_i + \beta_3$$

$$X = 11, Z(t) = \sum_{i=2}^{4} \beta_i P_{i+17}(t) + \alpha_3 P_6(t) + \alpha_2 P_{10}(t)$$

$$Y(r) = \sum_{i=1}^{4} \alpha_i + \beta_3$$

$$X = j, \quad Z(t) = \alpha_i P_k(t) \tag{2}$$

$$Y(r) = \beta_i S \tag{3}$$

For

$i = 1$ $\quad j = 18, k = 8;\ j = 17, k = 9;\ j = 20, k = 7;\ j = 24, k = 5;$
$j = 36, k = 4;\ j = 38, k = 1$

$i = 2$ $\quad j = 21, k = 7;\ j = 25, k = 5;\ j = 2, k = 6;\ j = 28, k = 11$

$i = 3, 4$ $\quad j = i, k = 3;\ j = i + 3, k = 8;\ j = i + 6, k = 9;\ j = i + 10, k = 7;$
$j = 26, k = 5;\ j = 34, k = 6;\ j = i + 17, k = 11;\ j = i + 19, k = 10;$
$j = 35, k = 2;\ j = 33, k = 0;\ j = 37, k = 4;\ j = 39, k = 1$

6 Steady state reliability analysis

Applebaum (1965) has discussed about the steady state reliability of systems which are mutually independent. The concept of reliability engineering for the process plant industries have been discussed by Freshwater and Buffham (1969). Norman (1988) has analysed the design changes in availability forecasts. Since the fertiliser industry is a process industry, its every unit should be available for a longer period. Therefore, the long run availability of the system is computed by substituting $\mathrm{d}/\mathrm{d}t = 0$ as $t \to \infty$ in Equation (1). By solving Equation (1), the various steady state probabilities are obtained.

Steady state probabilities for a shift conversion system are given as follows:

$$P_j = \left(\frac{\alpha_i}{\beta_i}\right) P_k \tag{4}$$

where, if

$i = 1 \quad j = 18, k = 8;\ j = 17, k = 9;\ j = 20, k = 7;\ j = 24, k = 5;$
$j = 36, k = 4;\ j = 38, k = 1$

$i = 2 \quad j = 21, k = 7;\ j = 25, k = 5;\ j = 27, k = 6;\ j = 28, k = 11$

$i = 3, 4 \quad j = 1, k = 3;\ j = i + 3, k = 8;\ j = i + 6, k = 9;\ j = i + 10, k = 7;$
$j = 26, k = 5;\ j = 34, k = 6;\ j = i + 17, k = 11;\ j = i + 19, k = 10;$
$j = 35, k = 2;\ j = 33, k = 0;\ j = 37, k = 4;\ j = 39, k = 1$

$P_1 = X_{24} P_0;\ P_2 = QP_0;\ P_3 = X_{18} P_0;\ P_4 = SP_0;\ P_5 = TP_0;$
$P_6 = WP_0;\ P_7 = UP_0;\ P_8 = XP_0;\ P_9 = VP_0;\ P_{10} = RP_0;\ P_{11} = NP_0$

where 'P_0' is the probability of full working which is computed using a normalising condition.

That is,

$$\sum_{i=0}^{39} P_i = 1, \text{ it gives } P_0 = \left[N_3\right]^{-1}$$

$$X_1 = \left[\frac{\alpha_1}{\alpha_2 + \beta_3}\right];\quad X_2 = \left[\frac{\alpha_3}{\alpha_1 + \beta_3}\right];\quad X_3 = \left[\frac{\alpha_2}{\alpha_1 + \alpha_2 + \beta_3}\right];$$

$$X_4 = \left[\frac{\alpha_4}{\alpha_1 + \alpha_2 + \beta_3}\right];\quad X_5 = X_8 = \left[\frac{\alpha_3}{\alpha_2 + \beta_3}\right];\quad X_6 = \left[\frac{\alpha_2}{\alpha_2 + \beta_3}\right];$$

$$X_7 = X_9 = \left[\frac{\alpha_1}{\alpha_2 + \beta_3}\right];\quad X_{10} = \left[\frac{\alpha_1}{\beta_3}\right];\quad X_{11} = \left[\frac{\alpha_3}{\beta_3}\right];\quad X_{12} = \left[\frac{\alpha_2}{\beta_3}\right];$$

$$X_{13} = \left[\frac{\alpha_2}{\alpha_1 + \alpha_3 + \beta_3}\right];\quad X_{14} = \left[\frac{\alpha_1}{\alpha_3 + \alpha_1}\right];\quad X_{15} = \left[\frac{\alpha_2}{\alpha_3 + \beta_1}\right];$$

$$X_{16} = \left[\frac{\alpha_1}{\alpha_2 + \alpha_3 + \beta_1}\right];\quad X_{17} = \left[\frac{\alpha_2}{\alpha_2 + \alpha_3 + \beta_1}\right];$$

$$X_{18} = \left[\frac{\alpha_3}{\alpha_1 + \alpha_2 + \beta_3}\right];\quad X_{19} = \left[\frac{\beta_3}{M}\right];\quad X_{20} = \left[\frac{\beta_1}{M}\right];\quad X_{21} = \left[\frac{\beta_2}{M}\right];$$

$$X_{22} = \left[\frac{\beta_3}{M}\right];\quad X_{23} = \left[\frac{\beta_2}{M}\right]$$

The values of the steady state availability function have been computed from Equation (1), by assuming suitable repair and failure rates and the effect of failure and repair rates of subsystem are tabulated in Tables 3 and 4.

Table 3 Effect of failure rates of subsystems

			Availability [AV]				
α_1	α_2	$\alpha_3 = 0.01$	*0.002*	*0.004*	*0.006*	*0.008*	*0.01*
0.000	0.000	1.0000	0.9016	0.8108	0.7303	0.6603	0.6001
0.000	0.005	0.9987	0.9002	0.8094	0.7290	0.6591	0.5989
0.000	0.010	0.9952	0.8963	0.8055	0.7254	0.6658	0.5959
0.005	0.000	0.9976	0.8990	0.8082	0.7279	0.6582	0.5980
0.005	0.005	0.9964	0.8976	0.8069	0.7266	0.6570	0.5970
0.005	0.010	0.9929	0.8938	0.8031	0.7231	0.6538	0.5941
0.010	0.000	0.9909	0.8922	0.8018	0.7220	0.6529	0.5934
0.010	0.005	0.9898	0.8909	0.8004	0.7208	0.6518	0.5923
0.010	0.010	0.9865	0.8871	0.7968	0.7173	0.6486	0.5895

Note: Taking $\alpha_2 = \beta_1$, $\beta_3 = \beta_4 = 0.02$, $\alpha_3 = \alpha_4 = 0.01$, $\beta_2 = 0.0001$.

Table 4 Effect of repair rates of subsystems

			Availability [AV]				
β_1	β_3	$\beta_4 = 0.01$	*0.02*	*0.04*	*0.06*	*0.08*	*0.1*
0.1	0.01	0.5904	0.6927	0.7583	0.7831	0.7961	0.8041
0.1	0.02	0.6384	0.7597	0.8394	0.8699	0.8859	0.8958
0.1	0.04	0.6555	0.7839	0.8691	0.9018	0.9191	0.9297
0.1	0.08	0.6606	0.7913	0.8781	0.9115	0.9291	0.9400
0.1	0.10	0.6612	0.7922	0.8793	0.9127	0.9304	0.9414
0.1	0.50	0.6624	0.7939	0.8814	0.9115	0.9328	0.9438
0.2	0.01	0.5958	0.7001	0.7672	0.7926	0.8059	0.8141
0.2	0.02	0.6426	0.7656	0.8466	0.8776	0.8940	0.9041
0.2	0.04	0.6589	0.7889	0.8752	0.9083	0.9258	0.9367
0.2	0.08	0.6637	0.7958	0.8837	0.9175	0.9354	0.9464
0.2	0.10	0.6643	0.7967	0.8848	0.9187	0.9366	0.9477
0.2	0.50	0.6655	0.7983	0.8868	0.9208	0.9388	0.9500
0.5	0.01	0.5985	0.7038	0.7717	0.7973	0.8108	0.8191
0.5	0.02	0.6443	0.7681	0.8497	0.8808	0.8973	0.9075
0.5	0.04	0.6601	0.7906	0.8773	0.9106	0.9282	0.9391
0.5	0.08	0.6647	0.7372	0.8855	0.9194	0.9374	0.9485
0.5	0.10	0.6653	0.7981	0.8866	0.9206	0.936	0.9497
0.5	0.50	0.664	0.7996	0.884	0.9226	0.9407	0.9519

Note: Taking $\alpha_1 = 0.01$, $\alpha_2 = 0.04$, $\alpha_3 = \alpha_4 = 0.005$, $\beta_1 = \beta_2$.

$$X_{28} = \left[\frac{\alpha_1}{\alpha_2 + \alpha_3 + \beta_1}\right]; \quad M = \left[\alpha_2 + \alpha_3 + \beta_2\right];$$
$$X = \left[X_{18}X_{24} + X_9X_{18}\right]$$

$$M_1 = \left[X_1X_3X_{18}\right]; \quad M_2 = \left[X_1X_4 + X_2X_{13}\right]; \quad M_3 = \left[X_5X_{16} + X_4X_7\right];$$
$$M_4 = \left[X_5X_{17}X_{24} + XX_6 + X_3X_7X_{18}\right]; \quad M_5 = \left[X_{13}X_{14} + X_{15}X_{16}\right];$$
$$M_6 = \left[X_{15}X_{17}X_{24}\right]$$

$$Y_1 = \left[X_{10}M_1 + X_{11}M_{16} + X_{12}M_4\right]; \quad Y_2 = \left[X_{10}M_2 + X_{11}X_5 + X_{12}M_3\right];$$
$$Y_3 = \left[X_{19}Y_1 + X_{20}M_6 + X_{22}M_1 + X_{23}\right]; \quad Y_4 = \left[X_{19}Y_2 + X_{20}M_5 + X_{21}X_{13} + Y_2M_2\right]$$

$$N_3 = G + H$$

$$G = \left[1 + X_{24}\left(1 + B_5\right) + X_{18}(1 + B) + X\left(1 + B_1\right) + R(1 + B) + N\left(1 + B_4\right)\right]$$

$$H = \left[T\left(1 + B_3\right) + W\left(1 + B_6\right) + U\left(1 + B_2\right) + V\left(1 + B_1\right) + R(1 + B) + N\left(1 + B_4\right)\right]$$

$$Q = \left[\frac{Y_3}{1 - Y_4}\right]; \quad T = \left[\left(QM_5 + M_6\right)\right]; \quad U = \left[QY_2 + Y_1\right]; \quad V = \left[QM_3 + M_4\right]$$

$$N = \left[QM_2 + M_1\right]; \quad W = \left[QX_{13}\right]; \quad R = \left[X_3X_{13} + QX_4\right]$$

$$S = \left[X_{17}X_{24} + QX_{16}\right]$$

$$B = \left[\left(\frac{\alpha_3}{\beta_3}\right) + \left(\frac{\alpha_4}{\beta_4}\right)\right]; \quad B_1 = \left[\left(\frac{\alpha_1}{\beta_1}\right) + B\right];$$
$$B_2 = \left[B_1 + \left(\frac{\alpha_2}{\beta_2}\right)\right]; \quad B_3 = \left[\left(\frac{\alpha_1}{\beta_1}\right) + \left(\frac{\alpha_2}{\beta_2}\right) + \left(\frac{\alpha_4}{\beta_4}\right)\right];$$
$$B_4 = \left[\left(\frac{\alpha_2}{\beta_2}\right) + \left(\frac{\alpha_3}{\beta_3}\right) + \left(\frac{\alpha_4}{\beta_4}\right)\right]; \quad B_5 = \left[\left(\frac{\alpha_1}{\beta_1}\right) + \left(\frac{\alpha_4}{\beta_4}\right)\right];$$
$$B_6 = \left[\left(\frac{\alpha_2}{\beta_2}\right) + \left(\frac{\alpha_4}{\beta_4}\right)\right]; \quad B_7 = \left[\frac{\alpha_4}{\beta_4}\right]$$

$$Z_1 = \left[X_{18}X_{24}\right]; \quad Z_2[N + Q + R + S + T + U + V + W + X]$$

The long run availability for the system is given as:

$$\mathrm{AV} = \sum_{i=0}^{11} P_i = \left[1 + Z_1 + Z_2\right]\left[N_3\right]^{-1}$$

7 Results and discussion

Bennet et al. (1977) have discussed about the failure prevention methods and reliability of the system. By taking the relevant values of failure and repair rates of each unit in the system the effects of these parameters on system availability are tabulated in Tables 3 and 4. Table 3 shows that if the failure rate of a feed gas heater (C) and the subsystem (D) increases, the system availability decreases by 9% from 500 to 250 hr.

Further increase in failure rates, decreases the availability further by 15%. Also an increase in the failure rate of the condensate pump (B) from once in 200 to 100 hr decreases the availability marginally to 0.3%. This is because the condensate pumps (2NO) are provided as standby units. Similarly increase in failure rate in subsystem. A decreases the availability by 0.7% from 200 to 100 hr, because the subsystem (A) also has pumps in standby for the units.

Table 4 shows the effect of decrease in repair time on availability. For subsystem (D) the reduction in repair time from 100 to 50 hr improves the system availability by 10% while for subsystem (C) the system availability improves by 2–3%, respectively. The effect of decrease in repair time for subsystem A and B on availability is negligible, because both subsystems have units in standby and there is no waiting for repair to maintain the system operative. Moreover the above analysis indicates that the repair in the subsystems can be undertaken in the order of preference as follows:

D	C	A	B

From Table 5, the repair priority is obtained based on the increase in failure rate, reduce in failure time.

Table 5 Availability and Repair priority of the system

Subsystem	*Increase in failure rate*	*Reduce in availability %*	*Reduce in failure time*	*Improve in availability*	*Repair priority*
A	200–100	0.7	100–50	Negligible	III
B	200–100	0.3	100–50	Negligible	IV
C	500–250	9.0	100–50	4–5%	II
D	500–250	9.0	100–50	10%	I

White (1979) has briefly discussed about the maintenance planning, control and documentation. Williom (1981) has done a case study about the shut down of ammonia plants. In this paper the maintenance schedule for the system is given in Table 6 which leads to effective production.

Table 6 Maintenance schedule

Schedule	*Component*	*Check/remarks*
Daily	All rotary and stationary equipments	Regular checking by unskilled workers and reported to maintenance manager for any abnormality
Weekly	Pumps	Vibration monitoring (check for unbalance, misalignment, looseness and distortion)
	Gear boxes	Check for temperature of oil, oil level
	Bearings	Check the condition and lubricate if required
	Humidifier, dehumidifier, refrigerator, boiler, economiser and cooler	Check the leakage
	Heater	Check the temperature difference
Fortnightly	Pumps	Check the vibration level and bearings temperature
Quarterly	Pumps	Coupling inspection, greasing, sleeve inspection
Annual	(22 days shutdown)	Thorough checking of all the components of each unit. Complete overhauling of rotating and stationary equipments through planned maintenance schedule is done

8 Conclusion

The stochastic behaviour and nature of the conversion system is analysed based on the Markov graph. Under the existing facilities, the system is operated to its maximum availability and the operational measures are carried out to attain minimum failure. A mathematical model is designed with the probabilistic considerations and the steady state reliability analysis is made for the long run availability of the system. Based on the above results, it is observed that for any level of system availability, various combinations of failure or repair time are possible and in practical, feasible combinations have to be restricted. The results are obtained for improving the design of the system, thereby obtaining the minimum failure.

References

Applebaum, S.P. (1965) 'Steady state reliability of systems of mutually independent subsystems', *IEEE Transactions on Reliability*, Vol. R14, No. 1.

Bennet, S.B., Ross, A.L. and Zamanick, P.Z. (1977) *Failure Prevention and Reliability, American Society of Mechanical Engineers*, New York.

Freshwater, D.C. and Buffham, B.A. (1969) 'Reliability engineering for the process plant industries', *The Chemical Engineer*, pp.367–369.

Kumar, D. and Pandey, P.C. (1993) 'Maintenance planning and resource allocation in a urea fertilizer industry', *Quality and Reliability Engineering International Journal*, Vol. 9, pp.411–423.

Kumar, D., Singh, J. and Pandey, P.C. (1991a) 'Behaviour analysis of urea decomposition system with general repair policy in fertilizer industry', *Micro Electron Reliability, International Journal*, Vol. 31, No. 5, pp.851–854.

Kumar, D., Singh, J. and Pandey, P.C. (1991b) Process design for crystallization system in the urea fertilizer industry', *Micro electron Reliability, International Journal*, Vol. 31, No. 5, pp.855–859.

Kumar, S., Kumar, D. and Mehta, N.P. (1996) 'Behavioural analysis of shell gasification and Carbon recovery process in urea fertilizer plant', *Micro Electron Reliability, International Journal*, Vol. 36, No. 5, pp.671–673.

Kumar, S., Kumar, D. and Mehta, N.P. (1997) 'Steady state behavioural and maintenance planning for desulphurization system in urea fertilizer plant', *Microelectron Reliability, International Journal*, Vol. 37, No. 6, pp.949–953.

Kumar, S., Kumar, D. and Mehta, N.P. (1999) 'Maintenance management for ammonia synthesis system in a urea fertilizer plant', *International Journal of Management and System (IJOMAS)*, Vol. 15, No. 3, pp.211–214.

Kumar, S., Kumar, D. and Mehta, N.P. (2000) 'Probabilistic analysis of desulphurization system in a urea fertilizer plant', *Journal of Institution of Engineers*, Vol. 80, pp.135–139.

Lenz, R.E. (1970) 'Reliability design in process plant', *Chemical Engineering Progress*, Vol. 66, pp.42–44.

Lieberman, N.P. (1973) *Process Design for Reliable Operation*, Houston, TX: Gulf publishing company.

Mcfalter, W.E. (1972) 'Reliability experiences in a larger refinery', *Chemical Engineering Progress*, Vol. 68, pp.52–55.

Norman, D. (1988) 'Incorporating operational experience and design changes in availability forecasts', *Reliability Engineering and System Safety*, Vol. 20, pp.245–261.

Priel, V.Z. (1974) *Systematic Maintenance Organization*, London McDonald and Sons.

Vijayalakshmi, C. and Senthamarai Kannan, K. (2005) *A Bulk Queue G/M/1 in Inventory Control, Computational Mathematics*, New Delhi: Narosa Publishing House, pp.239–246.

Vijayalakshmi, C. and Senthamarai Kannan, K. (2007) 'A study on stochastic behaviour for the reliability and cost benefit analysis in a Petro-Chemical Industry', *Applied Science Periodical*, Vol. X1, No. 3.

White, E.N. (1979) *Maintenance Planning, Control and Documentation*, England: Gower.

Williom, G.P. (1981) 'Case of ammonia plants shutdown', *Chemical Engineering Progress*, Vol. 74, pp.88–93.

EJIE Referees 2006

The *European Journal of Industrial Engineering* (*EJIE*) was launched in the June of 2006. We would like to express our thanks to those listed below who reviewed papers submitted to the *EJIE* in the year 2006. Each submitted paper is sent to three independent referees, and the editors' decisions depend heavily on referees' reliable assessment of submitted papers. We appreciate the time dedicated by the referees in assessing the first or revised versions of the papers. Of course, we also appreciate the help of our editorial board members whose names are not listed below.

A. Allahverdi, R. Ruiz, J.M. Framinan

Editors

Aggoune, R.
Álvarez-Valdés Olaguíbel, R.
Aparisi, F.
Baker, K.
Bermúdez, J.D.
Bertolini, M.
Billaut, J-C.
Bullington, S.
Buyurgan, N.
Camarinha-Matos, L.M.
Carrión García, A.
Chatfield, D.C.
Chen, J.S.
Chu, P.
Cuatrecasas Arbos, L.
De Almeida, J.
DePuy, G.W.
De Ron, A.J.
Doerner, K.
Engin, O.
Ferreira-Gomes, C.
Finke, G.
García Díaz, J.C.
Gökçen, H.
Gonzalez, P.L.
Greco, G.
Gu, J.
Gutiérrez Expósito, J.M.
Hayya, J.
Ho, J.C.
Kahraman, C.
Kalchschmidt, M.
Karabuk, S.
Khouja, M.
Kim, Y.K.
Lambrecht, M.
Le, T.
Leisten, R.
Liaw, C.F.
Lin, C.K.Y.
Lin, S-W.
Martin-Andino, R.
Matsumoto, T.
Matson, J.O.
Mendonça, D.
Mönch, L.
Moon, I.
Ndlela, L.
Neron, E.
Oechsner, R.
Pastor, R.
Poksinska, B.
Rajendran, C.
Rossi, R.
Ruiz-Usano, R.
Sakazume, Y.
Scarcello, F.
Schaller, J.
Scholl, A.
Schrage, L.
Smith, A.
Soares, J.
Soman, C.
Soroush, H.M.
Stecke, K.E.
Tarantilis, C.D.
Tasgetiren, M.F.
Taylor, G.D.
Trienekens, J.
Urban, T.L.
Villa, G.
Whitt, W.
Wirth, A.
Wysk, R.A.
Zhang, X.
Zolfaghari, S.

CONTENTS, KEYWORDS AND AUTHOR INDEXES FOR VOLUME 1

Contents Index Volume 1, 2007

Issue No. 1

Issue No. 2

Issue No. 3

Issue No. 4

Keywords Index Volume 1, 2007

Indexing is based on the keywords and phrases, title and abstract on the first page of each paper. Page references are to the first page of the paper or report.

Author Index Volume 1, 2007

CALL FOR PAPERS

European Journal of Industrial Engineering (EJIE)

Website: www.inderscience.com

ISSN (Online): 1751-5262 ISSN (Print): 1751-5254

Special Issue on: '*Emergent Computing for Service Management*'

The global economy is becoming increasingly service-oriented due to the key role of services, where the service industry produces over 80% of GNP and total employment in developed countries, with emerging and inspiring figures in developing countries, too. As a result, service management has received growing interest in recent years. The importance of studying service systems and finding "robust solutions" for the problems encountered in service management (design, strategy, quality, deployment and configuration of services, service operations management, service pricing, service reliability, etc.) has also been increasing. The main aim is to find "robust and acceptable solutions" for the problems within an affordable time period. However, many service industry problems still remain within this reasonable time due to the complexity and dynamic nature of the service systems. Emergent Computing (EC) studies offer nature-inspired problem solving systems, for this purpose. Examples typically cited as EC applications include (but are not limited to) agent-based systems, swarm intelligence (ant-colony, bee-colony, particle swarm algorithms etc.), cellular automata, chaos theory, evolutionary algorithms, artificial immune systems, neural and fuzzy systems etc. We believe that the use of EC for solving service management problems can improve not only "service intelligence" but also quality and performance of the service systems.

The main goal of this special issue is to increase awareness of the service sector on the effectiveness and power of emergent computing technology, through high quality research papers. We are inviting people from both academia and industry to submit papers on their recent research experience considering emergent computing to be applied to service management problems.

Subject Coverage

Suitable topics include but are not limited to:

Service Management

- Service design and development processes
- Supply chain management and logistics
- Service project management
- Service quality management
- Service operations design, development and management
- Service delivery, deployment and maintenance
- Human resource management in services

Emergent Computing

- Evolutionary algorithms and metaheuristcs
- Neural and fuzzy systems
- Swarm intelligence (ant-colonies, bee-colonies and particle swarm optimisation etc.)
- Agent-based systems
- Cellular automata
- Chaos theory
- Artificial immune systems

Important Dates

Submission Deadline:	30 January 2008
Notification of the Initial Decision:	30 April 2008
Notification of Acceptance:	30 July 2008

Submission

Authors should follow the instructions available at the journal website: (http://www.inderscience.com/ejie). Papers should be submitted in electronic form (in PDF format). Your submission should be original, unpublished, and not currently under consideration for publication elsewhere. Each submission will be evaluated by three independent referees. Quality and originality of the contribution are the main acceptance criteria. *EJIE* follows the double blind refereeing process. Therefore, there should be a separate title page giving the name and addresses of the authors. Any references that reveal the identity of the authors should be removed.

The manuscript should be emailed to:

Guest Editor: Türkay Dereli
Department of Industrial Engineering
Faculty of Engineering, University of Gaziantep
27310 Sehitkamil, Gaziantep, Turkey
Email: Turkay.Dereli@gantep.edu.tr

Guest Editor: M. Emin Aydın
Department of Computing and Information Systems
University of Bedfordshire, Luton, Beds., UK
Email: Mehmet.Aydin@beds.ac.uk