Integrating system dynamics and multi-objective optimisation for manufacturing supply chain analysis

Tehseen Aslam*, Amos H.C. Ng and Ingemar Karlsson

Virtual Systems Research Centre, University of Skövde, SE-54128, Skövde, Sweden
E-mail: Tehseen.Aslam@his.se
*Corresponding author

Abstract: The aim of this paper is to address the dilemma of supply chain management (SCM) within a truly Pareto-based multi-objective context. This is done by introducing an integration of system dynamics and multi-objective optimisation. An extended version of the well-known pedagogical SCM problem, the Beer Game, originally developed at MIT since the 1960s, has been used as the illustrative example. As will be discussed in the paper, the integrated multi-objective optimisation and system dynamics model has been shown to be very useful for revealing how the parameters in the Beer Game affect the optimality of the three common SCM objectives, namely, the minimisation of inventory cost, backlog cost, and the bullwhip effect.

[Received 22 April 2013; Revised 8 July 2013; Accepted 31 August 2013]

Keywords: multi-objective optimisation; MOO; system dynamics; supply chain.


Biographical notes: Tehseen Aslam is a PhD candidate within Industrial Informatics at the University of Skövde, Sweden. He holds a MSc degree in Manufacturing Management from Loughborough University, UK. His research interests include simulation modelling and simulation-based multi-objective optimisation for the design and analysis of supply chains.

Amos H.C. Ng is an Associate Professor at the University of Skövde, Sweden. He holds a BEng degree and an MPhil degree both in Manufacturing Engineering from the City University of Hong Kong and a PhD degree in Computing Sciences and Engineering from De Montfort University, Leicester, UK. He is a member of the IET and a Chartered Engineer in the UK. His research interests include modelling, simulation and multi-objective optimisation for production systems design and analysis.

Ingemar Karlsson is a PhD candidate at the University of Skövde. He holds a BSc in Computer Science and an MSc in Automation Engineering. His research interests include simulation-based optimisation, cloud computing technologies and data visualisation.

Copyright © 2014 Inderscience Enterprises Ltd.
This paper is a revised and expanded version of a paper entitled ‘Integrating system dynamics and multi-objective optimization for manufacturing supply chain analysis’ presented at the Swedish Production Symposium, Linköping, 6–8 November 2012.

1 Introduction

Modelling is an effective way of designing, understanding or analysing real-world processes and systems. A model enables a decision maker to gain a better understanding of the complexity of a process/system and evaluate/predict its performance under various circumstances. A supply chain incorporates the integrated processes in which products are transformed from raw material, or components, e.g., from the suppliers, to finished products delivered to end customers. Typically, these processes include different business functions in a company, e.g., procurement, production, logistics, etc., which need to collaborate, coordinate and interact with each other, in order to produce the commodity of the supply chain (Kim et al., 2004). Hence, a supply chain can be seen as a good example of a complex system which requires the modelling of processes in the presence of multiple, autonomous entities (i.e., suppliers, manufacturers, distributors, retailers, etc.), multiple performance measures and multiple objectives, both local and global, which together constitute very complex interaction effects (Keramati, 2010). Li et al. (2002) point out that supply chain modelling is more or less a prerequisite for supply chain integration and present four incentives for supply chain modelling:

1 capturing supply chain complexities, e.g., interaction effects between supply chain members, through a better understanding and uniform representation of the supply chain

2 designing the supply chain management process in order to manage supply chain interdependencies

3 establishing the visions to be shared by supply chains members and providing a basis for supply chain coordination and integration

4 facilitating the reduction of supply chain dynamics at supply chain design phases.

The modelling method for supply chains proposed in this paper is based on system dynamics (SD). The use of SD for modelling supply chains started in the late 1950s and early 1960s with the work of Jay W. Forrester, who constructed a model of a production and distribution system consisting of a four-echelon supply chain. By using such a SD model, he exposed the existence of up-stream demand amplification in a supply chain and further elaborated that this could be tackled through reducing or eliminating delays together with the suitable design of feedback loops (Forrester, 1961). However, since the development of Forrester’s production-distribution model, the research community shifted its focus from SD modelling to other approaches, such as discrete event modelling. As Angerhofer and Angelides (2000) point out, the use of SD modelling within the supply chain domain has just recently re-emerged after a slack period. Despite the increase in research within the domain of utilising SD for SCM issues, Aslam et al. (2011) have found that in the literature there are only a few articles related to the
Integration of simulation-based optimisation (SBO) with SD models. In particular, they conclude that the integration of multi-objective optimisation (MOO) and SD models can provide a novel method that can be used for not only for the optimisation, but also the analysis in SCM studies. The aim of this paper is therefore to introduce such an integration of MOO and SD for manufacturing supply chain analysis. This is done by implementing a SBO application for the well-known Beer Game (BG), which in its original form is a role-playing simulation game that aims to replicate a multi-echelon supply chain, with the overall objective to illustrate the system dynamic aspects of SCM. The remainder of this paper is as follows: Section 2 is an introduction of the concept of applying MOO for SCM studies, supported by a review of related work. Section 3 introduces the BG, including its background and the SD model developed for the specific study in this paper. The integrated MOO and SD study for the BG is presented in Section 4. MOO results and analysis, particularly using data visualisation methods like Parallel coordinates are included in Section 5, which is followed by conclusions and future work.

2 MOO for supply chain management

In a general MOO problem, there is no single best solution with respect to all objectives: improving one objective of an optimal solution would deteriorate the other (Deb, 2004). In an MOO problem, a decision maker is presented with the Pareto-optimal, or non-dominated solutions, which belong to a set of ‘optimal’ trade-offs, among the multiple conflicting objectives. It is important to note that an MOO problem can easily be converted into a single-objective optimisation problem by formulating a weighted-sum objective function which is composed of the multiple objectives, so that a single trade-off optimal solution can be sought effectively. However, the major drawback is that the trade-off solution obtained by using this procedure is very sensitive to the relative preference vector. Therefore, the choice of the preference weights and thus the obtained trade-off solution is highly subjective to the particular decision maker. At the same time, it is also argued that using weighted-sum objective function to obtain a single ‘global’ optimal solution for a multi-tier system like a supply chain is not desirable if the ‘global’ optimum suggests a set of decision variable values that may sacrifice the performance of the sub-system level. For example, the optimal solution found by the SBO may be optimal when considering the overall supply chain, but totally unacceptable to the company that plays the role of the manufacturer. Consequently, for a decision maker, it would be useful for the posterior Pareto front to be generated by using an MOO algorithm, so that further analysis can be done on the Pareto-optimal solution set, in order to acquire more knowledge about the system before making any decision.

When any supply chain system is examined more closely, it becomes very clear that it is a complex system comprising multiple entities (e.g., suppliers, manufacturers, distributors and retailers, as mentioned earlier) which individually have their own performance measures and objectives to optimise their internal process. However, optimising these individual entities is not adequate when optimising a supply chain, as it is a dynamic network comprising multiple transaction points with complex transportations, information processes as well as financial arrangements between entities. Hence, optimising the supply chain as a whole is as crucial as the optimisation of the individual entities, with the aim of SCM being to align and combine all these objectives,
individually, as well as in the entire supply chain, so that they work towards a common goal – increasing the efficiency and profitability of the overall supply chain. SCM is thus multi-objective in nature and involves several conflicting objectives, both at the individual entity level and at the supply chain level.

A comprehensive literature survey was carried out by Aslam et al. (2011), which showed that most of the research conducted on MOO for supply chain management is based on mathematical approaches, e.g., linear programming, mixed integer programming, mixed integer linear programming, and so on. In comparison to the large number of publications on applying simulation approaches to SCM problems, it seems that exploring the use of SBO, especially in the context of MOO, is far from adequate.

3 The BG

The BG is a role playing simulation game which aims to replicate a multi-echelon supply chain that incorporates four entities, namely: Factory (F), Wholesaler (W), Distributor (D), and Retailer (R). These entities together build a beer production and distribution supply chain with the overall objective of fulfilling customer demand. The BG was originally developed at the MIT Sloan School of Management in the 1960s (Sterman, 1989). The main purpose of the game is to demonstrate the existence of the so-called bullwhip effect, which refers to a phenomenon in supply chains where the demand variability of incoming orders is amplified as they move upstream in the supply chain. The most well-known analytical study on the bullwhip effect was carried out by Lee et al. (1997), who point out the important role of collaboration, coordination, information management, inventory and production control, order management, purchasing, etc., in order to manage the bullwhip effect. In terms of structure, the BG is actually simple and easy to understand. Nonetheless, it incorporates a rather intrinsic dynamic complexity. Sterman (2000) points out that most people might think of complexity in the form of the number of components in a system or the amount of various combinations of inputs/information a decision maker must consider when making a decision. But he argues that in contrast to the combinatorial complexity or detail complexity of a problem or a system, the dynamic complexity can arise from very simple structures with low combinatorial complexity, especially in the presence of feedback. The BG exemplifies such a structure in which the dynamic complexity arises from the interaction between the entities in the supply chain represented in the BG.

The original BG is played on a board with physical chips/markers moving around, for instructions of the rules and regulation of this BG, the reader is referred to Sterman (1989). However, in this paper, we present a replica of a SD model presented in Joshi (2001), which in turn is a modified version of the original system dynamics beer game (SD-BG) model built by Kirkwood (2008). The overall purpose of the models presented in Joshi (2001) and Kirkwood (2008), as well as the BG played on a board, is to show the existence of the bullwhip effect and how to manage it by implementing different information sharing policies. However, our main intention is to implement MOO into SD within the supply chain context and investigate the Pareto-optimal trade-off solutions when minimising three conflicting objectives commonly found in many supply chains, namely, inventory level, backlog level, and the bullwhip effect.
Each entity in the SD-BG model follows the generic structure presented in Figure 1, the thicker lines/arrows in the structure represent elements in the system where delay occurs. As in the case of the original BG presented in Sterman (1989), there is no collaborative interaction between the entities in the simulation model. Each entity places orders with its supplier according to the observed demand pattern from its downstream customer. The two exceptions are the raw material supplier, which only delivers what is ordered by \( F \), and the end customer, who only places orders with \( R \). Before continuing with the model description and default settings of the BG, it is essential to describe the notions and definitions of the main variables in the generic structure. For a complete variables list and description of the BG model, readers are referred to Joshi (2001). The main variables of most interest in this study are the ordering policy, total supply chain inventory cost, total supply chain backlog cost, and total supply chain cost. Equations for these variables in the generic SD-BG entity model are presented below.

The inventory cost of the entire supply chain is given by:

\[
CINV_{i}^{SC} = CINV_{i-1}^{SC} + \left(INV_{i}^{SC} \times cf^{INV}\right)
\]

where \( cf^{INV} = 0.5 \)

\[
(1)
\]

where

\[
INV_{i}^{SC} = INV_{i-1}^{SC} + \sum_{i} INV_{i}^{i}
\]

and

\[
INV_{i}^{i} = INV_{i-1}^{i} + IU_{i}^{i} - SU_{i}^{i}
\]

\[
(2)
\]

\[
(3)
\]
Inventory of the entire supply chain, in period $t$.
- Indicates the aggregated inventory of the supply chain.

Inventory of entity $i$, in period $t$.
- The on-hand inventory at each entity $i$.

Incoming units to entity $i$ from entity $i-1$ in period $t$.
- Entity $F$, which does not have an upstream entity, receives units from a raw material supplier who just sends the number of units ordered by $F$.
- Incoming units are received with a delay.

Sold units from entity $i$ to entity $i+1$, in period $t$.
- Entity $R$, which does not have a downstream entity, sends the sold units to a sink and the units are consumed by the model.

Cost of supply chain inventory, in period $t$.
- Indicates the holding cost of aggregated inventory of the supply chain.

Cost factor for inventory.
- Penalty factor for holding inventory.

The backlog cost of the entire supply chain is given by:

$$CB^{\text{SC}}_t = CB^{\text{SC}}_{t-1} + \left( B^{\text{SC}}_t * cf^B \right)$$  \hspace{1cm} (4)

where

$$B^{\text{SC}}_t = B^{\text{SC}}_{t-1} + \sum_i B^i_t$$  \hspace{1cm} (5)

and

$$B^i_t = B^i_{t-1} + BR^i_t$$  \hspace{1cm} (6)

where

$$BR^i_t = D^{i+1}_t - SU^i_t$$ for each entity $i$  \hspace{1cm} (7)

Except entity $R$ where $D^{i+1}_t = D^{\text{sup}}_t$

Backlog of entire supply chain, in period $t$.
- Indicates the aggregated backlog of the supply chain.
- Backlog refers to orders that have been received but not yet fulfilled.

Backlog of entity $i$, in period $t$.
- Indicates the aggregated backlog of entity $i$. 

Integrating system dynamics and multi-objective optimisation

\( BR_i^t \)  
Backlog rate of entity \( i \), in period \( t \).
- Is the accumulation of backlog in entity \( i \).

\( D_i^t \)  
Orders placed by entity \( i \) to entity \( i-1 \), in period \( t \).
- when orders are placed to entity \( i-1 \) then \( D_i^{t+1} = D_i^t \)
- entity \( R \), which does not have a downstream entity, receives orders placed by the end customer
- orders are placed with a delay.

\( D_{\text{out}} \)  
End customer demand.

\( CB_i^{SC} \)  
Cost of supply chain backlog, in period \( t \).
- Indicates the cost of the aggregated backlog orders of the supply chain.

\( cf^B \)  
Cost factor for backlog
- Penalty factor for having backlog.

The cumulative cost of the entire supply chain is given by:

\[
C_t^{SC} = C_{t-1}^{SC} + CI_t^{SC}
\]  

where

\[
CI_t^{SC} = CINV_t^{SC} + CB_t^{SC}
\]

\( C_t^{SC} \)  
Cost of entire supply chain, in period \( t \).
- Indicates the aggregated cost of the supply chain.

\( CI_t^{SC} \)  
Rate of cost increase of the entire supply chain, in period \( t \).
- Represents the accumulation of the cost in the supply chain.

The governing variable in each entity is the ordering policy, which is based on an anchoring and adjustment heuristic first presented in Tversky and Kahneman (1974). The anchoring and adjustment heuristic is used to estimate an unknown quantity by first setting or defining an anchor, e.g., recalling a known reference point, and then adjusting factors, e.g., additional or updated information, to estimate the unknown quantity (Sterman, 1989). As mentioned earlier, there is no collaborative interaction between the entities in the simulation model, thus the ordering policy heuristics aim to utilise the local information available to the entities in the supply chain, such as local inventory, backlog, and supply line information. The ordering policy is defined on the basis of four conditions:

1. the order must be non-negative
2. order enough to cover the expected inventory losses
3. reduce the discrepancy between desired and actual inventory
4. reduce the discrepancy between desired and actual supply line.
The ordering policy at each entity $i$ is given by:

$$OP_i^t = \text{MAX} \left( 0, DOP_i^t \right)$$  \hspace{1cm} (10)

where

$$DOP_i^t = DF_i^t + \alpha^t \left( DI^t - \left( INV_i^t - B_i^t \right) \right) + \beta^t \left( DSL_i^t - SL_i^t \right)$$  \hspace{1cm} (11)

where

$$DF_i^t = \text{SMOOTH} \left( D_i^{*+1}, \gamma^t \right)$$  \hspace{1cm} (12)

where $D_i^{*+1} = D_i^{*\text{last}}$ in entity $R$

$$\gamma^t = 1$$

and

$$SL_i^t = SL_{i-1}^t + SLR_i^t$$  \hspace{1cm} (13)

where

$$SLR_i^t = OP_i^t - IU_i^t$$  \hspace{1cm} (14)

$OP_i^t$ Ordering policy of entity $i$, in period $t$.
- represents the actual amount of the order to be placed
- this is defined once a week.

$DOP_i^t$ Desired ordering policy of entity $i$, in period $t$.
- Represents the desired amount of the order, calculated with the anchoring and adjustment heuristic.

$DF_i^t$ Demand forecast by entity $i$, in period $t$.
- use the Vensim® SMOOTH function to make a demand forecast
- SMOOTH() function is based on exponential smoothing technique.

$SL_i^t$ Supply line of entity $i$, in period $t$.
- Indicates the aggregated amount of units to be received for entity $i$.

$SLR_i^t$ Supply line rate of entity $i$, in period $t$.
- Represents the accumulation of the orders estimated by the ordering policy less than those which have been delivered.

$DI^t$ Desired inventory at entity $i$.
- Preferred amount of units in the inventory.
Integrating system dynamics and multi-objective optimisation

\( DSL^i \) Desired supply line at entity \( i \).
- Preferred amount of units in the supply line.

\( \alpha^i \) Forecasting parameter at entity \( i \).
- forecasting parameter for the inventory
- the parameter is usually represented in the range of \( \alpha^i = [0, 1] \).

\( \beta^i \) Forecasting parameter for supply line at entity \( i \).
- forecasting parameter for the supply line
- the parameter is usually represented in the range of \( \beta^i = [0, 1] \).

\( \gamma^i \) Smoothing time parameter at entity \( i \).
- Time input to the Vensim® SMOOTH function.

The first condition of the ordering policy, i.e., the order must be non-negative, is applied by equation (10) where the MAX() function in Vensim prevents \( OP^i \) from placing order values less than 0. The other three conditions refer to the anchoring and adjustment heuristic in equation (11), where the depletion or surplus of inventory and supply line will require adjustment towards the desired inventory and supply line levels, which is done by \( \alpha^i (DI^i - (INV^i - B^i)) \) and \( \beta^i (DSL^i - SL^i) \) respectively, in equation (11). Here, the \( \alpha \) and \( \beta \) parameters represent the discrepancy of the amount of units needed in the inventory in the case of \( \alpha \) and \( \beta \), which represents the fraction of supply line taken into account when determining \( OP^i \). Thus, a high \( \alpha \) value would indicate that the majority of the required units for the inventory will be ordered or, e.g., that the manager at entity \( i \) will implement an aggressive policy/effort in order to adjust the inventory towards the desired inventory level. In the case of \( \beta \), a value of \( \beta = 1 \) would indicate that all the orders in the supply line have been taken into account when deciding the amount of orders to place with the supplier, whereas \( \beta = 0 \) would indicate that no order in the supply line has been taken into account.

Figure 2 illustrates the flow of information, i.e., orders, moving upstream in the supply chain and the flow of material, i.e., beer crates, moving downstream through the supply chain. The simulated BG begins at \( t = 0 \) and every incremental week each entity of the supply chain has to make a decision about the number of crates that need to be ordered from the supplier, which is calculated in \( OP^i \) for each entity. Similarly, the number of crates that can be shipped downstream to its customer is done by calculating \( SU^i \). Thus, in the default simulation settings, the model starts out in equilibrium, i.e., \( t = 0 \), with no oscillating effect in the supply chain.

The end customer demand, i.e., \( D_{ cust}^i \), starts by ordering four crates of beer during the first four weeks and then suddenly, in week 5, the end customer increases its demand to eight crates a week for the rest of the simulation to simulate a ‘surprise demand increase’ (Morecroft, 2007). The default initial values for variables at each entity at \( t = 0 \) are: \( INV^i_0 = 12 \), \( IU^i_0 = 4 \), \( D^i_0 = 4 \), \( SL^i_0 = 8 \), \( DI^i = 12 \), \( DSL^i = 14.7 \), \( \alpha^i = 0.26 \), \( \beta^i = 0.088 \), \( \gamma^i = 1 \). Note that only variables with an initial value \( > 0 \) are presented here.
Figure 3 clearly shows the existence of the bullwhip effect from the output of the SD-BG model. This oscillating effect is captured using the default settings of the simulation model and it shows how an increase in end customer demand, from four crates to eight in week 5, has led to a huge oscillating effect at the final entity, $F$, where the demand ranges from zero to nearly 27 crates.

4 MOO for the BG

The experiment in this paper is conducted through two optimisation scenarios, both of which use the same function to minimise the inventory cost and backlog cost values. However, the two scenarios differ with regard to how the BWE is minimised. In scenario 1 (S1), the BWE is minimised through an approach first presented in Dudas et al. (2011) where the authors intend to minimise the BWE by minimising the peak (highest) ordering value. However, in scenario 2 (S2) an approach is implemented, presented by Chen et al. (2000), in which the BWE is quantified through the ratio between the variance of orders and the variance of demand. The same method of quantifying the bullwhip effect can find its practical use, e.g., in a recent study for examining the so-called internal bullwhip effect in automotive manufacturing (Klug, 2013). Two key factors that enable a successful MOO experiment are:
Integrating system dynamics and multi-objective optimisation

1. The ability to formulate a business problem as an optimisation problem that can be processed by the optimisation algorithm and the simulation model.

2. The ability to define a clear interface between the simulation model and the optimisation engine.

Hence, the three business goals considered in this paper, i.e., minimise inventory cost, backlog cost, and the BWE of the entire supply chain, are formulated as three objective functions that produce the quantitative values for the evaluations used in the SBO cycles.

It should be noted that the BWE in this paper is minimised by implementing the two above-mentioned BWE approaches at entity \( F \), as this entity is the final one when moving upstream in the supply chain. As shown in Figure 3, it is also this entity which experiences the highest demand fluctuations compared to other supply chain members. Hence, minimising the BWE at entity \( F \) is crucial. Furthermore, by limiting or minimising the demand fluctuation at entity \( F \), it would be interesting to know whether the demand amplification from the entity \( R \) to entity \( D \) can also be reduced.

The objective functions for the optimisation scenarios are denoted as:

\[
O_F^{S1} \begin{cases} 
\text{Min } O_F^{S1}(CINV_{SC}) = \text{Min } \mu_{CINV}\text{SC} \\
\text{Min } O_F^{S1}(CB_{SC}) = \text{Min } \mu_{CB}\text{SC} \\
\text{Min } O_F^{S1}(BWE_{max}^{F}) = \text{Min } \max{\mu_{BWE}^{F}}
\end{cases}
\]

\[
O_F^{S2} \begin{cases} 
\text{Min } O_F^{S2}(CINV_{SC}) = \text{Min } \mu_{CINV}\text{SC} \\
\text{Min } O_F^{S2}(CB_{SC}) = \text{Min } \mu_{CB}\text{SC} \\
\text{Min } O_F^{S2}(BWE_{var}^{F}) = \text{Min } \frac{\sigma_{\mu_{BWE}^{F}}^2}{\sigma_{\mu_{BWE}^{F}}^2}
\end{cases}
\]

where

\[
I = Dl^i, DSL^i, \alpha^i, \beta^i
\]

and

\[
O = \mu_{INV}^i, \mu_B^i, \mu_{SL}^i, \mu_C^i, \mu_{INV}\text{SC}, \mu_{CB}\text{SC}, \mu_{BWE}\text{SC}
\]

where

\[
\mu_{INV}^i = \frac{\sum_{t=0}^{T} INV_{t}^i}{T},
\]

\[
\mu_B^i = \frac{\sum_{t=0}^{T} B_{t}^i}{T},
\]

\[
\mu_{SL}^i = \frac{\sum_{t=0}^{T} SL_{t}^i}{T},
\]

\[
\mu_{BWE}^i = \frac{\sum_{t=0}^{T} BWE_{t}^i}{T}
\]
\[ \mu_{Ct} = \frac{\sum_{t=0}^{T} C_i^t}{T}, \]  
(22)

\[ \mu_{CINV} = \frac{\sum_{t=0}^{T} INV_{i}^S}{T}, \]  
(23)

\[ \mu_{CB} = \frac{\sum_{t=0}^{T} B_{i}^S}{T}, \]  
(24)

\[ \mu_{CSC} = \frac{\sum_{t=0}^{T} C_{i}^{SC}}{T} \]  
(25)

subject to

\[ CS_{i} = OP_{i} \geq 0, \alpha_{i} \leq 0 \leq D_{i} \leq 12, 0 \leq DSL_{i} \leq 12, \]  
\[ 0 \leq \alpha_{i} \leq 1, 0 \leq \beta_{i} \leq 1. \]  
(26)

**O_{F}^{S1} and O_{F}^{S2}**

- All optimisation objectives
- Represent all objective functions for the optimisation of respective scenario.

**O_{F}^{S1} and O_{F}^{S2}**

- First objective function
- Each represents the optimisation function to minimise the supply chain inventory of respective scenario.

**O_{F}^{S1} and O_{F}^{S2}**

- Second objective function
- Each represents the optimisation function to minimise the supply chain backlog of respective scenario.

**O_{F}^{S1} and O_{F}^{S2}**

- Third objective function
- represents the optimisation function to minimise the BWE of respective scenario
- where in \( O_{F}^{S1} \) the method \( BWE_{max}^F \) is utilised
- and, in \( O_{F}^{S2} \) the method \( BWE_{min}^F \) is utilised.

**BWE_{max}^F**

- Highest order value of entity \( F \)
- represents the highest value of order placed (\( max_{OF} \)) by entity \( F \)
- for details regarding this approach, readers are referred to Dudas et al. (2011).
### Integrating system dynamics and multi-objective optimisation

**BWE\textsubscript{var}** Ratio of order and demand variance
- states the ratio between order variance ($\sigma\textsuperscript{2}_{Oi}$) at entity $F$ and demand variance ($\sigma\textsuperscript{2}_{Di}$) of entity $D$
- for details regarding this approach, readers are referred to Chen et al. (2000).

**I** Input variables
- Indicates all the input variables utilised in the optimisation of both scenarios.

**O** Output variables
- Indicates all the output variables utilised in the optimisation of both scenarios.
- Each output represents a performance measure considered in the optimisation evaluation.

**CS\textsuperscript{i}** Default model constraints of entity $i$.
- Indicates all the constraints in the model for entity $i$.
- Defined in most cases as the upper and lower bounds of decision variables, e.g., $I$.

**$\mu\textsubscript{INV\textsuperscript{i}}$** Mean inventory of entity $i$.
- Gives the mean value of the inventory at entity $i$.

**$\mu\textsubscript{B\textsuperscript{i}}$** Mean backlog of entity $i$.
- Gives the mean value of the backlog at entity $i$.

**$\mu\textsubscript{SL\textsuperscript{i}}$** Mean supply line of entity $i$.
- Gives the mean value of the supply line at entity $i$.

**$\mu\textsubscript{C\textsuperscript{i}}$** Mean cost of entity $i$.
- Gives the mean value of the overall cost at entity $i$.

**$\mu\textsubscript{CINV\textsuperscript{SC}}$** Mean inventory cost of the supply chain.
- Gives the mean value of the inventory holding costs for the entire supply chain.

**$\mu\textsubscript{CB\textsuperscript{SC}}$** Mean backlog cost of the supply chain.
- Gives the mean value of the backlog costs for the entire supply chain.

**$\mu\textsubscript{C\textsuperscript{SC}}$** Mean cost of the supply chain.
- Gives the mean value of the overall cost for the entire supply chain.
End of output data collection period.

- states the end time of the output data collection period
- $T = 130$ weeks is used throughout this study.

5 SD-MOO interface

The optimisation software utilised for the MOO is called modeFrontier® (mF) (ESTECO, 2013), a Java-based application with a graphical user interface in which users can easily define input and output variables together with other required information of the optimisation as object nodes drawn on a canvas. These nodes can be connected to each other by links to construct a workflow. Figure 4 illustrates the iterative process of the SBO cycle using an integration of Vensim and mF. A user can select an MOO algorithm provided by mF, which generates a set of input values for the Vensim simulation model to evaluate. The mF has built-in support for the integration with a list of engineering software, e.g., MATLAB, for model evaluations, but no SD software such as Vensim used in this study. To remedy this, a small interface application has been written as a Windows console application in C#. This in-house developed application, called VensimInterface (VI), loads the Vensim model with help of the Vensim Application Programming Interface (API). The input values from a text file written by mF together with a settings file are used to set corresponding variables in the Vensim model.

As the figure also illustrates, in a SBO cycle, VI reads in input values from mF via a text file (Input.txt) and then writes the simulation evaluation to a text file (Output.txt) and sends it back to mF.

Figure 4 Integrating optimisation and SD for running MOO (see online version for colours)
The VI application is called by mF with four arguments, as seen in Figure 5. The first argument, which is the path to the (1) settings file, contains the names of the input and output variables in Vensim. The values in the input and output text files have to be in the same order as in the settings file, because mF reads and writes line by line to these text files. Argument (2) is the path to input text file and (3) is the path to output text file. The final argument, i.e., (4) is the design Id, as Vensim® stores all results from an evaluation in a data file which is stored together with the model file. Since it is crucial not to use the same filename for these files when running concurrent evaluations, VI makes use of the design Id to keep the filenames unique.

When the VI is called, it reads the settings file together with the input file. These settings and the input values are used to set the correct variables in Vensim after the model file is loaded. Thereafter, VI runs the model and retrieves the results from variables in the model based on the structure defined in the settings file. Vensim, a SD software, is based on a time period and therefore presents the results as a series of time-based values for each variable. The VI retrieves the series of values and, based on the settings file, calculates either the mean, the maximum, the minimum, or the standard deviation for each variable. These values are then written to the output text file.

The optimisation in this paper utilises nine different nodes in mF, which are shown in Figure 6. The **DOE node** is where the workflow begins; in this node the user specifies or generates a set of initial designs for the optimisation. The DOE node is connected to the **scheduler node**. This node is always connected to the DOE node and defines the optimisation algorithm and the settings for the algorithm. In this study, the DOE setting used is a Latin Hypercube Design (LHD) and the optimisation algorithm used is the widely-used NSGA-II algorithm (Deb et al., 2002). NSGA-II has been used to generate all the results presented in this paper. There are three major techniques that render the outstanding performance of NSGA-II (Ding et al., 2008):

1. a ‘fast’ non-dominated sorting approach that reduces the $O(mN^3)$ complexity of MOGA to $O(mN^2)$ (Babbar et al., 2003)
2. a $\lambda + \mu$ elitism selection procedure
3. the use of crowding distance, as a measure for comparison and selection after the non-dominated sorting, to preserve the diversity of the solutions in the population.

On the other hand, instead of started from random initial solutions, LHD can assist the optimisation algorithm, in this case NSGA-II, to sample and explore the decision space more efficiently. The **input node** defines an input variable and its corresponding lower and upper bounds, together with the step size of the incremental increase or decrease. The **input file node**, on the other hand, writes the values from one or more input nodes to a text file, which is later read by the VI.
The DOS batch script node, named VI, starts the VI console application with the necessary arguments, i.e., settings file, input file, and output file. The output file node reads a text file written by the VI application and loads the result, in form of output data, to output nodes, which represent the result from the model evaluation. A design objective node, MinInvSC or MinBacSC, defines an objective for the optimisation, by setting an expression formulating the objective function for one or more output nodes. The Logic end, i.e., exit in Figure 6, indicates the end of the workflow. An evaluation of the SD model in Vensim is fast (< 0.2 sec.) compared to a discrete-event simulation model developed for identical purpose, but an even better performance is possible by running model evaluations in parallel computers. This is supported in mF by starting more than one instance of the batch script, in this case the VI application. The user can modify how many evaluations to run concurrently by setting the property Num. of Concurrent Design Evaluations in the scheduler node.
Figure 7 illustrates the internal process of the VI, which is initiated by mF and the DOS batch script node that starts the VI console application. By executing the VI.exe, the application loads the simulation model setting, together with the input and output variable settings defined in the `settings.xml` file. The VI application then loads the input text file which contains the input parameter values set by the input nodes in mF for the current evaluation run. After loading input values, the application opens the Vensim model and sets input values to the corresponding variables in Vensim. When the simulation run is finished, the application obtains the performance measure values from Vensim and transcribes the values to an output text file, which is consequently read into the output nodes in mF, as presented in Figure 6. The input and output parameter values, together with the optimisation objective, results are stored in mF. Identical to most of the SBO applications, the Vensim model and mF simply treat each other as a black box, which is made possible with such an in-house developed integration.
6 Results and analysis

A total of 300,000 evaluations were run for each scenario in the experiment, which generated around 2500 Pareto optimal solutions for S1 and 3800 Pareto solutions for S2. The results of the two scenarios are shown in Figure 8. Apparently, running the optimisation for 300,000 evaluations might seem to be excessive at a first glance. Nevertheless, it is essential to remember that 8 of the 16 design variables are continuous variables with an accuracy of $10^{-6}$ decimal (in the original SD model developed by MIT). By considering only these eight continuous variables would already generate a search space of $10^{48}$ possible solutions. At the same time, as mentioned, a single evaluation of the SD model took only approximately 0.2 seconds to run. Therefore, the number of evaluations run was actually not a real issue from a computation perspective and in order to obtain a Pareto front that could meet our preferences in terms of quality, accuracy, and intensification of the Pareto solutions, it is argued that 300,000 evaluations for generating the results are justified.

Figure 8 Pareto-optimal solutions for S1 and S2 (see online version for colours)

With the data sets containing Pareto optimal solution sets obtained from S1 and S2, the problem lies in how to extract valuable or useful information about the BG problem. One way of handling high-volume and high-dimensional data sets is to utilise a divide-and-conquer approach, i.e., dividing a larger problem into smaller sub-problems, where the results of the partitioning of the problem are expected to obtain data samples which have common properties and characteristics within the same sub-group or cluster (Xiao and Yu, 2012). Clustering or cluster analysis is one such approach that aims to discover concealed structures in the data set and identify homogenous clusters, or sets of groups, from the samples in the data set, by minimising the variation of samples within a group as well as maximising the variation between the partitioned subgroups (Di Stefano, 2009). Jain et al. (1999) describe cluster analysis as an unsupervised learning method which in contrast to a supervised learning method, e.g., classification techniques for data analysis, does not require pre-classified, i.e., labelled, patterns. Cluster analysis is very useful for pattern analysis, grouping, as well as decision making situations, by providing
the ability to explore the interrelationships between the data samples in the high volume data sets. It is important to note that the cluster analysis in this paper is used to identify patterns and common properties among the Pareto solutions gained from the optimisation, which is a unique concept introduced by Bandaru and Deb (2011).

Figure 9 Partitive clustering 3D plot of S1 and S2 Pareto solutions (see online version for colours)

In order to identify any patterns or common properties among the Pareto solutions, the Partitive Clustering Analysis (PCA) tool in mF was used, in which the three optimisation objectives were chosen as the clustering criterion. By evaluating the clustering analysis results, as shown in Figure 9, five clusters were identified by the clustering tool for respective scenario. For instance, the S2 clustering plot in Figure 9 clearly illustrates that the Pareto solutions in cluster S2-C2 have a much lower value for the \( CB^E \) objective than Pareto solutions in cluster S2-C0. On the other hand, most of the solutions in S2-C0 have a much lower value for the \( FC_{fen} \) objective than S2-C2. Similarly, the S1 cluster plot shows that solutions in S1-C3 have significantly lower values for the \( BWE_{max} \) objective than S1-C2, whereas S1-C2 outperforms S1-C3 in terms of the remaining two optimisation objectives, namely \( CINV^H \) and \( CB^H \), by having a lower value for respective objective. While the clustering analysis in Figure 9 presents an understandable overview of the partitioning of solutions in the objective space, it does not provide the decision maker with the hidden interrelationships between the Pareto solutions as well as the common properties and characteristics of the Pareto solutions within a specific cluster in the decision space. In this case, the clustering results from PCA can be visualised with a parallel coordinate (PC) plot, which is one of the most efficient approaches for visualising high-volume and high-dimensional data to assist decision makers in finding relations between the design space variables and the optimisation objectives (Savoska and Loskovska, 2009). PC plots can also be used to understand cluster properties as well as to explore common features and characteristics of the solutions within a specific cluster or group (Theus and Urbanek, 2009).
Figure 10  PC plot of S1 clustered Pareto solutions (see online version for colours)
Figure 11  PC plot of S2 clustered Pareto solutions (see online version for colours)
Figure 10 is a PC plot of the clustered Pareto solutions of S1, showing the mapping or relationship between the objective space values and the design space variables and their values. By comparing the clusters obtained by PCA for S1, it can be seen, for instance, that the aforementioned cluster S1-C3 contains Pareto solutions with the lowest $BWE_{\text{max}}$ objective values and the highest $CINV^{\text{SC}}$ objective value, compared to the other clusters found in S1. Solutions in cluster S1-C0, on the other hand, generate the highest $BWE_{\text{max}}$ objective values and the lowest $CINV^{\text{SC}}$ values, whereas S1-C3 solutions have a slightly higher $CB^{\text{SC}}$ objective value than S1-C0. It is interesting to note that clusters S1-C1 and S1-C4 contain the best trade-off solutions concerning the $BWE_{\text{max}}$ and $CINV^{\text{SC}}$ objectives, where the values for these objectives are slightly higher for S1-C4 than S1-C1. However, the S1-C1 cluster has the highest $CB^{\text{SC}}$ objective value, whereas the S1-C4 value is almost in the middle, in comparison to the other clusters. Examining the design space variables for the S1 clusters reveals that in cluster S1-C3 the design variable $\beta$ has the highest value compared to other clusters. This indicates that for these Pareto solutions the manager of the $R$ must take the orders in the supply line into account to a much higher degree than for all other clusters, when deciding the number of orders to place with the supplier. The S1-C1 cluster, on the other hand, has the highest $\alpha$ values, compared to all S1 clusters. This indicates that all entity managers in the supply chain are required to implement an aggressive policy to adjust the inventory towards the desired levels, in order to minimise the discrepancy of the amount of units needed in the inventory.

In contrast, while the solutions in cluster S1-C2 have lower $\alpha$ and $\beta$ values, these solutions have instead the highest DSL values for all members except the $R$. On the other hand, solutions in S1-C1 have lower DSL values but higher $\alpha$ and $\beta$ values compared to other S1 clusters.

The PC plot in Figure 11 presents the clusters obtained in S2 and similar to the clustering results presented in Figure 10, the trade-offs between the three optimisation objectives are clearly revealed. Figure 11 illustrates that clusters S2-C1 and S2-C2 have very high $CINV^{\text{SC}}$ values compared to other identified clusters in S2. However, solutions in S2-C1 generate the lowest $BWE_{\text{cum}}$ values and the highest $CB^{\text{SC}}$ values for S2, whereas S2-C2 solutions generate the highest $BWE_{\text{cum}}$ and the lowest $CB^{\text{SC}}$ values. By examining the design variables of these two clusters, S2-C2 has much higher DSL and DI values than S2-C1, due to the considerable bullwhip effect, i.e., the high $BWE_{\text{cum}}$ value. Through a comparison of the clusters obtained from the S2-C1 and S2-C2 solutions, it can therefore be concluded that by decreasing the DSL and DI values, the bullwhip effect can also be decreased. However, it is interesting to note that both clusters have more or less the same $CINV^{\text{SC}}$ values, whereas S2-C1 has a high $CB^{\text{SC}}$ value and S2-C2 has a very considerable low value. These results could be explained by the fact that S2-C2 solutions have higher $\alpha$ and $\beta$ values for all supply chain members, except $\beta$, indicating that managers at the different supply chain entities implement a very aggressive, engaging, and considerate policy regarding their inventory and supply line. Additionally, as Figure 11 shows, the solutions in S2-C2 have the highest DSL and DI values for all entities, except $DF$, which indicates that the inventory levels are high throughout the supply chain, due to the high DSL and DI values. The fact that the values for S2-C2 vary from entity to entity is an indication that some supply chain members are required to have
Integrating system dynamics and multi-objective optimisation

higher inventory levels than the others. Hence, the matching $CINV^{SC}$ values of the two clusters would indicate that the inventory for cluster S2-C2 is more or less evenly distributed among the supply chain entities, whereas the inventory for cluster S2-C1 is unevenly distributed. The huge difference in the $CB^{SC}$ value also indicates that the solutions in S2-C1 are not able to meet customer demand. Hence there is a high backlog, as wrong entities hold the higher and lower inventory levels due to the uneven inventory distribution, i.e., the inventory is not placed at the right entity and at the right levels.

Looking at cluster S2-C3, which contains the best trade-off solutions, it can be seen that these solutions have generated lower $CB^{SC}$ and $CINV^{SC}$ values, despite having a higher $BWE_{chen}$ value than cluster S2-C1. By evaluating the design variables of the three clusters, it can be observed that solutions in cluster S2-C3 also have higher $\alpha$ and $\beta$ values than S2-C1. However, instead of having very high DSL and DI at all supply chain entities, as in the case of cluster S2-C2, the solutions in cluster S2-C3 show that only high values for specific parameters are required, which in this case are $DI$, $DSL^I$ and $DSLD$, whereas the values for $DI^W$ and $DSLR$ can be significantly lower.

A comparison of the clusters for the two scenarios through normalising the two data sets, Figure 12, shows that S2 clusters in general achieve much lower demand amplification than S1. S2 clusters also obtain a lower total backlog cost than S1 clusters, with the exception of cluster S1-C2, which outperforms two S2 clusters, namely, clusters S2-C0 and S2-C1. On the other hand, only cluster S1-C0 has a lower total inventory cost than the clusters in S2. By an evaluation of the design variables for the two scenarios, Figure 12, it shows that in general the S2 clusters have greater DSL and DI values as well as greater $\alpha$. However, the $\beta$ values are greater in S1 clusters with the exception of $\beta^F$, which is significantly larger for clusters in S2.

Despite all the knowledge regarding relationships between the design space variables and the optimisation objectives that could be gained by using the combined PCA and PC plotting method, the decision maker still faces the question of which cluster should be chosen. If precedence is given to the $CB^{SC}$ objective, then cluster S2-C2 should be chosen. On the other hand, if the $CINV^{SC}$ objective is to be prioritised, then the chosen cluster should be S1-C2. In the case of giving preference to the $BWE^{F}$ objective, S2-C1 is the most preferred cluster. However, if no preference is given to either of the objectives, then cluster S2-C4 has the best trade-off solutions obtained from the optimisation, as shown in Figure 12.

A deeper analysis can be made by evaluating cluster S2-C4 in terms of total supply chain cost, i.e., $C^{SC}$. Figure 13 shows that this cluster is actually the second best in terms of total supply chain cost, together with S2-C3, which obtains the same $C^{SC}$ cluster values. However, both these clusters are outperformed by cluster S2-C2, which has the highest inventory cost levels, but also the lowest backlog cost. At this point, it might be argued that it would not be possible for the solutions in S2-C2 to achieve the lowest total supply chain cost, since they generate the greatest inventory holding costs for the supply chain, whereas solutions in cluster S2-C4 can achieve the lowest inventory holding costs and a slightly higher backlog cost. As a matter of fact, with regard to the design of this BG, it should be remembered that in accordance with equation (1) and equation (4) in the simulation model, it is twice as expensive to have backlog than to hold inventory. This can easily explain why the solutions in S2-C2 can achieve the lowest total supply chain cost compared to S2-C4.
Figure 12  Comparison of S1 and S2 clustered Pareto solutions (see online version for colours)
Figure 13  Cost comparison of S1 and S2 clustered Pareto solutions (see online version for colours)
Figure 14  All Pareto solutions in cluster S2-C2 (see online version for colours)
Figure 15  Best solutions in terms of $C^{sc}$ (see online version for colours)
When a preferred region has been identified in the objective space, in this case cluster S2-C2, the question arises regarding the design variable properties and characteristics of the Pareto solutions within the selected cluster; Figure 14 displays all the Pareto solutions in the S2-C2 cluster. Figure 14 illustrates that the manager at entity $F$ is required to consider their supply line when placing an order with the supplier, due to the high $\beta$ values which range from $\beta^F[0.6850–1]$. Other interesting observations in Figure 14 are that the solutions require high values of $DIF[11–12]$ and $DSLF[12]$, whereas the solutions require a desired inventory value of $DIR[10]$ or $DIR[12]$ for entity $R$. On the other hand, the $DSL^D[0–12]$ and $DSL^W[0–12]$ parameters seem to have no influencing effect on the solutions, as they vary between the lower and upper boundaries of the parameters. It can therefore be identified that the lowest values of $DIW[3–12]$ do not have any effect on the solution, whereas values between $[0–2]$ and $[5–6]$ of $DID$ also have no effect on the solution, as the values range between $DF^D[7–12]$ and $DF^D[3]$ for the parameter. All upper and lower bounds of the design parameters found for the Pareto solutions in cluster S2-C2 are presented in Table 1.

Table 1  Design and Objective parameter boundaries for S2-C2

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Lower Bound</th>
<th>Upper Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha^F$</td>
<td>$[0.1901–0.4320]$</td>
<td>$[0.2847–0.6904]$</td>
</tr>
<tr>
<td>$\alpha^D$</td>
<td>$[0.0769–0.3207]$</td>
<td>$[0.0963–0.4051]$</td>
</tr>
<tr>
<td>$\alpha^W$</td>
<td>$[0.0199–0.1599]$</td>
<td>$[0.0534–0.2826]$</td>
</tr>
<tr>
<td>$\alpha^R$</td>
<td>$[0.0540–0.2826]$</td>
<td>$[0.1190–0.4676]$</td>
</tr>
<tr>
<td>$\beta^F$</td>
<td>$[0.6850–1]$</td>
<td>$[0.0199–0.1599]$</td>
</tr>
<tr>
<td>$\beta^D$</td>
<td>$[0.0769–0.3207]$</td>
<td>$[0.0963–0.4051]$</td>
</tr>
<tr>
<td>$\beta^W$</td>
<td>$[0.0769–0.3207]$</td>
<td>$[0.0963–0.4051]$</td>
</tr>
<tr>
<td>$\beta^R$</td>
<td>$[0.0199–0.1599]$</td>
<td>$[0.0534–0.2826]$</td>
</tr>
</tbody>
</table>

For a decision maker to seek a single ‘best’ solution, in terms of the lowest $C^SC$ value presented in Figure 15, which in this case is a selected solution with the following $\alpha$ and $\beta$ values: $\alpha^F = 0.3782$, $\alpha^D = 0.4531$, $\alpha^W = 0.2250$, $\alpha^R = 0.1482$, $\beta^F = 0.7759$, $\beta^D = 0.0676$, $\beta^W = 0.21$ and $\beta^R = 0.2138$, the desired inventory and supply line values at the different supply chain entities are as follows: $DIF = 12$, $DID = 8$, $DIW = 10$, $DIR = 12$, $DSLF = 12$, $DSL^D = 11$, $DSL^W = 10$ and $DSL^R = 12$. The presented design variables subsequently generate the following values for the three optimisation objectives and the total supply chain cost parameter respectively: $CB^SC = 5.9395$, $CINV^SC = 5.2303$, $BWE^S_{cli}$ = 0.5984 and $C^SC = 857.93$.

A very interesting question that may arise is: Has this particular chosen solution and its parameter setting outperformed the original settings of the BG? Figure 16 compares the results gained from the original BG settings and from the parameter settings of the selected solution of our MOO analysis. As the figure shows, the selected solution of the MOO is able to substantially decrease the total supply cost, i.e., $C^SC$, as well as reduce the costs for the individual members in the supply chain. The inventory has also been significantly reduced throughout the supply chain, whereas the backlog is reduced only for entities $D$ and $W$. However, due to the increase of the backlog of the entity $F$, shown
in Figure 16, the overall backlog has been increased. Hence, the original BG settings achieve lower total supply chain backlog than the final solution of the MOO analysis. As Figure 16 illustrates, the magnitude of the bullwhip effect, in terms of the peak of oscillations, is also reduced. Nevertheless, as the figure reveals, the $BWE_{val}$ value for entities $F$ and $D$ in the selected solution continuously fluctuates somewhat until the end of the simulation horizon, whereas the oscillation has levelled out around week 50 or 60 with the original BG setting.

**Figure 16** Comparing BG default run with the final chosen solution of the moo analysis (EV. 101209) (see online version for colours)

The interesting aspect of this comparison is that the chosen solution (Evaluation 101209) is able to reduce the total cost despite having a higher backlog. This phenomenon can be explained by the fact that compared to the original BG inventory the MOO has found solutions that can achieve a significantly lower inventory throughout the supply chain, which in turn is able to reduce the costs both for the individual entities, as well as the whole supply chain.
7 Conclusions and future work

This paper has proposed an integrated SD and MOO approach to find and investigate the Pareto-optimal solutions of the pedagogical SCM model, namely, the BG. In contrast to other related studies where inventory and backlog are formulated into a single optimisation objective, in order to minimise the total cumulative cost, the current study has clearly shown that the minimisation of total backlog cost and the minimisation of total inventory cost are two conflicting objectives, when they are optimised within a truly MOO context. In addition to minimising the total inventory and backlog costs, this paper has considered a third optimisation objective, which is to minimise the demand fluctuations, i.e., the bullwhip effect, in the supply chain. The results gained from the integration of SD and MOO for the BG described in this paper have clearly shown that the three optimisation objectives are in conflict with each other, in the sense that a SC manager cannot minimise the bullwhip effect without increasing the total inventory and total backlog levels. Through the results of this study, the integrated SD and MOO method is believed to provide an innovative approach for generating the design and analysis of manufacturing supply chain systems. Future work includes further MOO runs that include additional optimisation parameters and investigating other ordering policies, as well as comparing other methods of quantifying the bullwhip effect. Applying the method to SD models developed for real-life industrial problems is also underway.

References


Integrating system dynamics and multi-objective optimisation


