Production Planning and Control via Service-oriented Simulation Integration Platform

Dong Xu, Young-Jun Son
Systems and Industrial Engineering
The University of Arizona
Tucson, AZ 85721, USA

Abstract

In this paper, a service-oriented simulation integration platform is proposed to support manufacturing production planning and control of a complex manufacturing system. In particular, a multi-level service composition structure is considered, where simulation models at different levels of the hierarchy (e.g. equipment, shop, and enterprise) can be seamlessly and efficiently integrated. The Service oriented architecture Modeling Language (SoaML) is then employed to specify the service capabilities, service interfaces, service data model and choreography related to production planning and control. Furthermore, the proposed approach is demonstrated through single-period and multi-period inventory management. For the single-period inventory control, the optimal product price is estimated under different demand variability. For the multi-period inventory control, the convergence of a multi-agent reinforcement learning algorithm is demonstrated considering the eligibility trace. The proposed platform has been successfully deployed for integrating various different simulation models (e.g. discrete-event, agent-based, systems dynamics, process simulation). In addition, experiments illustrate the impact of demand variability on the product price, and the learning results of the optimal decision policy.

Keywords
Service-oriented architecture, simulation, inventory control, reinforcement learning

1. Introduction

Traditional manufacturing planning and control have gradually evolved from process-oriented and object-oriented to service-oriented architecture (SOA) in recent years. There are several drivers for adopting SOA in manufacturing industry, where the top 3 have been identified as 1) competitive advantage, 2) internal operational/process efficiency, and 3) dynamic business environment support [1]. Service-oriented development of applications (SODA) helps to reduce the total IT expenses and increase ROI over the long run, when compared with the traditional client-server development approaches (e.g. rapid application development (RAD), architected model-driven (AMD), architected rapid application development (ARAD)). In addition, the saving amount tends to be exponential over time as the business service library enriches [2]. As a result, SOA provides a solution to integrate legacy systems for reusing them without much re-development over and over again as a new technology becomes available. Before applying the SOA, various services (e.g. production planning, scheduling, and inventory control) and their interfaces should be considered and formulated explicitly first. Through such clearly designed services, the shop level operations have to be performed not only in a timely and cost-saving manner, but also in a coordinated way considering other functionalities. Failures in doing so will result in delay in the delivery of the final product and consequently, large penalties and even loss of contracts.

In this work, the SOA is applied firstly to an exemplary manufacturing production enterprise using Service oriented architecture Modeling Language (SoaML), followed by the discussion of inventory management to illustrate the usage of proposed SOA. In the first portion of this work, a service composition is constructed firstly by breaking down the enterprise level services into shop and equipment levels. Then, service capabilities, service interfaces, service data model, and choreography related to order placement and associated operations are discussed. For the service implementation, web service technology is used for integrating various real as well as simulated operations. Here, the considered simulation models can be used for the planning purpose, when various alternatives are evaluated before making a decision. The same simulation models, however, can also be used for the control purpose,
when deployed in real time to generate control commands for driving real operations in the shop. The second portion of this work discusses inventory management using the proposed SOA, where determining the inventory quantities and estimating its related cost are inherently a production planning problem [3]. In this work, the problem is studied under two different circumstances: 1) single-period and 2) multi-period inventory control. The single-period inventory control problem (based on the newsvendor problem) is introduced firstly involving a product retail price and its competitor consideration, where a closed-form analytic solution is only available under the assumption of a certain statistical distribution (e.g. uniform) on demand. Furthermore, such a single-period problem is extended into multi-period inventory control, and the competitive Markov decision processes (also known as stochastic game) and multi-agent reinforcement learning are used to model and resolve the problem, respectively. Experiments are conducted to demonstrate the proposed service oriented simulation integration framework by integrating various distributed simulation models via web service. Furthermore, given the proposed service oriented simulation integration framework, the experiments also illustrate the product price estimation under different values of salvage gain and opportunity loss given the high and low demand variability, and the convergence of proposed learning algorithm for inventory management application.

The remainder of this paper is organized as follows: In Section 2, the literature work related to the proposed platform is summarized. In Section 3, the details of different modules that constitute the proposed service-oriented simulation integration platform are discussed, followed by the application for inventory management utilizing the proposed SOA. In Section 4, experimental scenarios and results are presented. Finally, the conclusions and future extensions are discussed in Section 5.

2. Background and Literature Review

SOA in production planning and control has been studied by several researchers. Karnouskos et al. [4] proposed a web service-based integration between enterprise systems and shop floor activities. Shen et al. [5] proposed an agent-based SOA for dynamic resource scheduling services in a collaborative manufacturing environment. Savio et al. [6] demonstrated how SOA can provide the flexibility for optimizing the production plan dynamically. Jung [7] applied SOA for service chain-based business alliance formation, where social network analysis was employed for studying the social relationships between services. However, a systematic modeling framework is needed, which will facilitate service orchestration, dynamic resource provisioning, information security, and service management. Furthermore, it will provide a coherent platform to support evaluations of various decisions (e.g. scheduling, inventory control) that arise in a manufacturing enterprise.

To enable the SOA, web service technology is employed in this work. According to World Wide Web Consortium (W3C), web service is “a software system designed to support interoperable machine-to-machine interaction over a network”, where the communication/interaction occurs between digital simulation models and physical machines connected via Internet in this paper. The web service follows the prototype of standard protocol stack including 1) HTTP serving as data transport media, 2) SOAP XML for message format, 3) WSDL for web service function description, and 4) UDDI for publishing the service online. Web service technology was demonstrated in [8] and [9] to possess both the flexibility for customizing time synchronization methods needed to integrate multiple simulations in a distributed computing environment, and its simplicity required for use.

Regarding inventory control, two major types of problems have been studied in the literature: 1) single-period inventory control and 2) multi-period inventory control. As mentioned before, the single inventory control problem in this paper is built upon the newsvendor problem with price and competitor extensions. Several seminal works on the newsvendor problem and its extension can be found in [10, 11]. In [10], customer demand, pricing policy, and risk preference are emphasized as the key ongoing research themes. For the multi-period inventory control, traditional approaches use dynamic programming to solve Markov decision processes (MDPs), which require the full knowledge of environmental dynamics (e.g. state transition probability, reward function). As the full knowledge of environmental dynamics is difficult to obtain, model-free based approach (e.g. reinforcement learning technique) is usually applied to resolve such problems. Giannoccaro et al. [12] studied general inventory management via reinforcement learning approach, in which the efficiency of the algorithm was proved especially when the size of MDP is large. Ravulapati et al. [13] applied reinforcement learning approach in a business stochastic game environment, in which the performance (upper bound) of the proposed approach was demonstrated. Chinthalapati et al. [14] applied reinforcement learning approach to study the dynamic prices in a competitive electronic retail market under the stochastic game framework. Various decision variables considered in [14] include order quantity, stochastic demand and sensitive customers. In this work, a multi-agent reinforcement learning algorithm is
Xu and Son

considered for addressing the competition, and simulation is used to mimic the product flow (from upstream to downstream) and market environments.

3. Production Planning and Control via Service-oriented Simulation Integration Platform

3.1 SOA and its Modeling Framework

The service-oriented approach can be applied following the procedure of service identification, service specification, service realization, service composition, and service implementation. Among these procedures, this paper focuses on 1) service identification, where different levels of services, service capabilities and structures are established; 2) service specification, where different service interfaces and the service data model are discussed; and 3) service composition, where the service choreography of different participants is emphasized.

As the first step, it is important to explicitly identify the service items at different levels of the hierarchy and their relationships. Figure 1 demonstrates a hierarchical service decomposition structure from the enterprise level to the shop floor level, and furthermore to the manufacturing cell level. The service interface layer can be used to represent the interrelationship among different service levels. The application layers that include applications of .NET, J2EE, and other legacy systems can be embedded into the service interface layer to fulfill any service operations. It is worthy to note that simulations (e.g. discrete-event, systems dynamics, and process simulation) can be used to replace certain application programs for generating the same outputs in order to fulfill the service request. An exemplary shop level service is production management service, which can be further decomposed into services such as production planning, scheduling and quality assurance. To fulfill the scheduling service, a scheduling program or a simulation model can be employed.

Figure 1: Structure of hierarchical service decomposition in an exemplary manufacturing enterprise

Service identification is the process to decide which capabilities should be exposed as services and how one particular capability may use other capabilities to fulfill its service. Figure 2(a) describes various capabilities needed for processing an order requested by customers. These service capabilities involve inventory management service, production management service, and invoice management. The production management service further uses various
Xu and Son

other services including production planning, scheduling, and quality assurance services. Figure 2(b) shows three exemplary service interfaces related with process planning, scheduling and quality assurance.

Figure 2: (a) Service capabilities for order placement service and (b) exemplary service interfaces

Figure 3(a) represents the service data model, which is the model to represent information exchanged between service providers and consumers. As shown in Figure 3(a), the ProductionOrder object consists of various data entities such as ItemList entity, ProcessPlan entity, Schedule entity, and QualityAssurance entity. Each data entity includes various instances, and a <<id>> notation is used for uniquely identifying different instances of that class. Figure 3(b) defines a multi-party service choreography, which is a behavior diagram for identifying what and when the interactions between different service parties (e.g. providers and consumers) happen. The subjects of Figure 3(b) are order quotation and order processing, in which various scenarios are considered including order placement, production request, and invoice management. While simulations are used to fulfill the service request (e.g. price quote), various benefits can be achieved including low operational cost and what-if analysis capability. Next section discusses inventory management application using the proposed service oriented simulation integration platform.

Figure 3: (a) Service data model and (b) multi-party service choreography
3.2 Inventory Management via Service-oriented Simulation Integration Platform

As discussed in the previous section, various operations can be involved when invoking an enterprise level or shop level service, which includes production planning, scheduling, inventory control, and invoice management operations as examples. It is noted that one type of service (e.g. inventory management) interacts with other services and performs various functionalities as a whole, but the focus of this section is only on the inventory control for demonstration purposes when the related service operations are invoked. Other related operations (e.g. scheduling) are left for future works due to the page limit of this paper. Table 1 summarizes the nomenclature used throughout this section.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Explanation</th>
<th>Notation</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product sale price from company $i$</td>
<td>$p_i$</td>
<td>Product unit underage cost of company $i$</td>
<td>$c^{(u)}_i$</td>
</tr>
<tr>
<td>Product wholesale price (purchasing cost) of company $i$</td>
<td>$w_i$</td>
<td>Product unit salvage income (overstock) of company $i$</td>
<td>$G_i$</td>
</tr>
<tr>
<td>Product unit shortage loss (understock)</td>
<td>$S_i$</td>
<td>Coefficients of demand function of company $i$</td>
<td>$k_i, b_i, c_i$</td>
</tr>
<tr>
<td>Order quantity of company $i$</td>
<td>$q_{i}$</td>
<td>Profit function of company $i$</td>
<td>$q_i$</td>
</tr>
<tr>
<td>Product demand function of company $i$</td>
<td>$x_i$</td>
<td>Cumulative density function (cdf) of demand error</td>
<td>$F(e)$</td>
</tr>
<tr>
<td>Demand error term</td>
<td>$e$</td>
<td>Demand variability term</td>
<td>$z$</td>
</tr>
<tr>
<td>State index ($s \in S$)</td>
<td>$s$</td>
<td>Action index of company $i$ ($a_i \in A_i$)</td>
<td>$a_i$</td>
</tr>
<tr>
<td>Number of times for choosing action $a_{-i}$ while visiting at state $s$</td>
<td>$n(s)$</td>
<td>Number of times for visiting state $s$</td>
<td>$\gamma$</td>
</tr>
<tr>
<td>Value function for choosing action $a_i$ while visiting at state $s$</td>
<td>$Q(s, a_i)$</td>
<td>Discount factor of learning</td>
<td>$\gamma$</td>
</tr>
<tr>
<td>Learning rate</td>
<td>$\alpha_i$</td>
<td>Trace-decay parameter</td>
<td>$\lambda$</td>
</tr>
</tbody>
</table>

### 3.2.1 Single-period Inventory Control

In this subsection, we focus on using the service of single-period inventory control to estimate 1) the product order quantity, and 2) the product price, given the product procurement cost, uncertainty in demand, and the competitor. The service can be requested by operation manager or higher level executive personnel, and the service inputs include the forecasted product demand and various product cost items (i.e. salvage gain, opportunity loss). As the product demand and cost items involve uncertainties, Monte Carlo simulation is employed for processing the service request. Finally, the output is the product price (selected for illustration purposes) under different values of salvage gain and opportunity loss given the high and low demand variability. Other types of outputs are left for future work.

The inventory decision is considered to be influenced by 1) the product price and 2) its competitor decision in this paper. We employed the formulation of newsvendor problem as our basic structure of inventory control model. By adding the above mentioned two items into the basic newsvendor setting, the expected profit function can be formulated as follows:

$$E(\varphi_1(p_1, q_1)) = (p_1 - w_1)E(x_1(p_1, p_2)) - c^{(o)}_1 \int_{0}^{q_1}[q_1 - x_1(p_1, p_2)]f(x_1)dx_1 - c^{(u)}_1 \int_{q_1}^{\infty}[x_1(p_1, p_2) - q_1]f(x_1)dx_1$$

(1)

Note that $E(q - x)^{+}$ and $E(x - q)^{+}$ can be used to represent $\int_{0}^{q}(q - x)f(x)dx$ and $\int_{q}^{\infty}(x - q)f(x)dx$ inside Equation (1), respectively. Equation (1) can be rewritten as the following one.

$$E(\varphi_1(p_1, q_1)) = (p_1 - w_1)E(x_1(p_1, p_2)) - c^{(o)}_1 E(q_1 - x_1(p_1, p_2))^{+} - c^{(u)}_1 E(x_1(p_1, p_2) - q_1)^{+}$$

(2)

Where, $x_1(p_1, p_2)$ is the product demand function considering additive demand form (vs. multiplicative demand form) between the two competitive companies (see Equation (3)). It is noted that the assumed demand form conforms to the various properties identified in [15]. Furthermore, it is assumed that $k_i, b_i, c_i > 0$, and $E(e) = 0$. The product unit underage and underage cost can be represented by Equations (4) and (5), where product unit salvage income and shortage loss are assumed to be the same between the competitive companies.

$$x_1(p_1, p_2) = k_1 - b_1 p_1 + c_1 p_2 + e$$

(3)

$$c^{(o)}_1 = w_1 - g$$

(4)
Then after plug-in and manipulation, it is straightforward to show that Equations (6) and (7) hold. Similar relationship for single player newsvendor problem was found in [16].

\[
E(q_1 - x_1(p_1, p_2))^+ = \int_0^{q_1 + b_1 p_1 - c_1 p_2} f(e) \, de
\]

\[
E(x_1(p_1, p_2) - q_1)^+ = \int_{q_1 - b_1 p_1 + c_1 p_2}^{+\infty} [1 - F(e)] \, de
\]

In order to find the optimum order quantity and product price, the concave property of the expected profit function is derived below. Firstly, the Hessian matrix of the expected profit function is calculated as follows:

\[
H(E(\varphi_1(p_1, q_1))) = \begin{pmatrix}
\frac{\partial^2 E(\varphi_1(p_1, q_1))}{\partial p_1^2} & \frac{\partial^2 E(\varphi_1(p_1, q_1))}{\partial p_1 \partial q_1} \\
\frac{\partial^2 E(\varphi_1(p_1, q_1))}{\partial q_1 \partial p_1} & \frac{\partial^2 E(\varphi_1(p_1, q_1))}{\partial q_1^2}
\end{pmatrix}
\]

It is not difficult to find that

\[
\frac{\partial^2 E(\varphi_1(p_1, q_1))}{\partial p_1^2} < 0, \quad \frac{\partial^2 E(\varphi_1(p_1, q_1))}{\partial q_1^2} < 0, \quad \text{and} \quad \det |H(E(\varphi_1(p_1, q_1)))| > 0,
\]

which concludes that the expected profit is concave with respect to the order quantity and product price. Then, by letting

\[
\frac{\partial E(\varphi_1(p_1, q_1))}{\partial q_1} = 0,
\]

Equation (8) can be derived to represent the order quantity that involves the term of product price. Finally, after plugging Equation (8) into the expected profit function, and letting

\[
\frac{\partial E(\varphi_1(p_1, q_1))}{\partial p_1} = 0,
\]

the product price can be derived as shown in Equation (9). Similar relationship of order quantity and product price can be derived for the competitive company.

\[
q_1(p_1) = F^{-1}\left(\frac{c_1}{z \sqrt{\pi}} + \frac{c_1^2}{2z^2}ight) + k_1 - b_1 p_1 + c_1 p_2
\]

\[
p_1 = \left((3b_1 - 1)p_2 + \int_{-\infty}^{-1}\frac{e}{(p_2 - g^s) + e} \, [1 - F(e)] \, de + (b_1 - 1)s + w_1 - k_1\right) / c_1
\]

Note that for the exact analytic solution to be derived, the above equations can only be resolved with the assumption of certain statistic distribution of demand error. The following part focuses on resolving such problems by assuming the uniform distribution of demand error term. Equations (10) and (11) provide the generic representation of probability density function and cumulative density function of demand error distribution, with variable \( z \) for adjusting the demand variability. By knowing the PDF and CDF of the demand error, Equation (12) can be derived to replace the integral part in Equation (9). Then, the corresponding product price and order quantity can be calculated. In the experiment section, different values of demand variability (\( z \) value) are chosen for studying its impact to the product price. Due to the context limit of this paper, other studies (e.g. impact to the order quantity, impacts of other parameters) are left for future works.

\[
f(e) = \begin{cases} 
\frac{1}{2z} & e \in (-z, z) \\
0 & \text{otherwise} 
\end{cases}
\]

\[
F(e) = \begin{cases} 
\frac{e + z}{2z} & e \in (-z, z) \\
1 & e \geq -z
\end{cases}
\]

\[
\int_{-1}^{+\infty}\left(\frac{p_2 - w_1 + s}{p_2 - g^s}\right) [1 - F(e)] \, de = z^2 \left(\frac{w_1 - g^s}{p_1 - g^s}\right)^2
\]

### 3.2.2 Multi-period Inventory Control

In this subsection, we discuss the service of multi-period inventory control to 1) decide the optimal strategy of product order quantity, and 2) estimate the properties (e.g. stability, convergence) of long-run profit, given that the market environment varies over time (e.g. expanding or shrinking market size, changing demand pattern, and inventory substitution consideration). In this case, the service inputs involve not only the forecasted product demand and various product cost items, but also the potential market condition changes over time. When the input request is submitted, various data can be exchanged through the integration of multiple simulation models via web service technology. A reinforcement learning algorithm can be incorporated together with service oriented simulation integration platform for deciding the optimal ordering strategy and estimating properties of long-run profit as outputs.
After extending the above problem into multiple periods, several issues are to be addressed such as whether the decision (derived in the previous section) is stationary for selection, and when the market environment varies over time (e.g. expanding or shrinking market size, changing demand pattern, inventory substitution consideration). It’s not straightforward to handle such issues via the above formulated mathematical equations as various approximations (e.g. demand distribution, uncertainty) are required for that, and the obtained results are likely to be unrealistic as well.

To explicitly consider the environment uncertainties over time, the framework of MDPs is employed. As the competition is considered between two companies, the competitive MDPs – also known as stochastic game, is considered. In our problem, the decision is made after every fixed period of time according to the inventory control and price setting frequency. The combination of inventory and price is considered as a state (i.e. state \( k \) can be represented as \((20, 6)\) with the inventory level equal to 20 and the product price equal to 6), and the action covers two dimensions with inventory replenishment and price determination. The reward is set as the net profit for each company. The difficulties of dealing with such problems include: 1) the full knowledge of environmental dynamics is not available, and 2) the same action selected by one company may result in a different profit and market share due to its competitor’s behavior. To overcome these difficulties, simulation has been applied in this work to model the potential changes in the product flow (from upstream to downstream) and market environments (e.g. expanding or shrinking market size, changing demand pattern), and competing action sets of both companies.

To resolve MDPs, two streams of approaches are available in the literature: 1) model-based approaches, and 2) model-free approaches. Model-based approaches mainly include dynamic programing method (e.g. policy iteration, value iteration), which usually requires the full knowledge of environmental dynamics to construct a state transition probability matrix and a reward matrix. As discussed before, lack of full knowledge of environmental dynamics and competitor’s behavior make such an approach not feasible for our problem. Instead, the model-free approach, which mainly comprises of the reinforcement learning method (e.g. temporal-difference learning), is employed in this work. Furthermore, to clearly address the competition between companies, multi-agent reinforcement learning technique [17] is used. Figure 4 demonstrates a general procedure for a two-agent reinforcement learning, in which the action selections, state and reward updates of two agents are illustrated.

![Figure 4: Procedure for multi-agent reinforcement learning algorithm (a two-agent example)](image)

Out of various multi-agent reinforcement learning algorithms for resolving the stochastic game, our goal is to find the best response (or equilibrium) learning policy, which can find the optimal state-action pairs under various different circumstances. To this end, the opponent modeling \( Q(\lambda) \)-learning algorithm is adopted for its capability of modeling the opponent’s competitive behaviors explicitly. Note that the eligibility trace is added into the original opponent modeling \( Q \)-learning algorithm to achieve the optimum balance between exploration and exploitation. Figure 5 represents the algorithmic details for opponent modeling \( Q(\lambda) \) learning algorithm. Inside the algorithm, it is observed that the counters (i.e. \( C(s, a) \), \( n(s) \)) for state visiting and action selection were designed to record the opponent’s behavior and visiting count of all system states. The information is used to guide the further action selection of each player (company). The eligibility trace (i.e. \( \epsilon(s, a) \)) is also used and updated every step. In the experiments conducted in this work, the trace-decay parameter (i.e. \( \lambda \)) has been adjusted for achieving the optimum convergence results.
4. Experiments and Results

The service-oriented simulation integration platform is run on a homogeneous windows-based platform with various workstations and a Windows Server involved. All the workstations and the server are connected by the 100 Mbps Ethernet. The web service has been developed using Visual C# in the ASP.NET framework and deployed at the Windows Internet Information Services (IIS 6.0). Figure 6 demonstrates the web service based simulation integration platform for integrating various simulation models (e.g. systems dynamics, agent-based, discrete-event, process simulator). During the simulation run, time synchronization needs to be maintained, and data interactions can be achieved.

Figure 6: Service-oriented simulation integration platform implemented via web services

Considering the single-period inventory control scenario, Table 2 provides the parameter values used in the experiments. As seen from the table, two extreme values of $z$ were considered with 50 for a low demand variability and 1000 for a high demand variability. The salvage income and opportunity loss were assumed to have a normal distribution with mean 2.5 and variance 0.1. Monte Carlo (MC) simulation was conducted via MATLAB, and Figure 7 demonstrates the usage of service-oriented simulation integration platform (one MC simulation with
Xu and Son

multiple input data sources) for estimating the product price given the product salvage gain and opportunity loss under low and high demand variability. From the results, it can be concluded that 1) the product price with a high demand variability was suggested to be higher than that with a low demand variability, and 2) given the same value of opportunity loss, the price with the low demand variability has much higher uncertainty than that with a high demand variability. The variability and dynamics of supply, demand, and price are found common in practice, which can be incorporated into the proposed platform by firstly requesting service through corresponding service interface. Then, various scenarios generated from these dynamics (e.g. supply, demand and price) are evaluated through service capability (i.e. integrated simulation platform). Lastly, the results are shared with the service consumer (e.g. production manager).

<table>
<thead>
<tr>
<th>Parameters</th>
<th>( k_1, k_2 )</th>
<th>( b_1, b_2 )</th>
<th>( c_1, c_2 )</th>
<th>( w_1, w_2 )</th>
<th>( z )</th>
<th>( s )</th>
<th>( g )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Values</td>
<td>1000</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>50 (low variability); 1000 (high variability)</td>
<td>( N(2.5, 0.1) )</td>
<td>( N(2.5, 0.1) )</td>
</tr>
</tbody>
</table>

Figure 7: plots of product price with respect to salvage gain and opportunity loss under a) low demand variability and b) high demand variability

When the multi-period inventory control problem is concerned, the opponent modeling \( Q(\lambda) \) learning algorithm is applied. The length of one simulation step is set as 7 days, and roughly 500 simulation steps were executed (the number has been selected to guarantee the algorithm converging). The entire model was partitioned into 9 different states with 9 actions available for each state (a total of 81 state-action pairs). Through the integration of two systems dynamic models and one agent-based model (using web service), Figure 8(a) demonstrates that convergence results of the average profits of the two competitive companies. From the results, we can observe that 1) the average profits of the two competitive companies converge eventually, and 2) although the final average profits are not the same, the numbers of steps needed to converge are similar between two companies (e.g. about 250 steps). Figure 8(b) shows the number of steps needed to converge under different trace-decay parameter (\( \lambda \)) values. From the figure, it was found that the algorithms with \( \lambda = 0.3 \) and 0.8 involve the least and the most number of simulation steps needed to converge, respectively.

Figure 8: a) convergence results of MARL learning over simulation steps with \( \lambda = 0 \), b) convergence results of total simulation steps needed under different \( \lambda \) values
5. Conclusions and Future Works

In this work, the SOA and modeling language were firstly applied to an exemplary manufacturing enterprise, followed by the inventory management discussion to demonstrate the usage of the proposed approach under two different cases (i.e. single-period and multi-period inventory control). To this end, the service composition was discussed, followed by the service capabilities, service interfaces, service data model and choreography related to order placement and associated operations. For the service implementation, web service technology was used for integrating various real operations and simulation models. For the second portion of this work, the impact of demand variably on the product price was demonstrated for the single-period inventory control via the proposed service-oriented simulation integration platform. For the multi-period inventory control, the convergence of learning algorithm and the optimal trace-decay parameter values were shown. Future works will concern: 1) other applications for the proposed service-oriented simulation integration platform, and 2) other impacts (variability of salvage income, opportunity loss) in the inventory management problem on the product price and order quantity, and sensitivity of the proposed multi-agent reinforcement learning algorithm.

References