Proceedings of the 2015 Winter Simulation Conference
L. Yilmaz, W. K. V. Chan, I. Moon, T. M. K. Roeder, C. Macal, and M. D. Rossetti, eds.

TRADEOFFS BETWEEN OBJECTIVE MEASURES AND EXECUTION SPEED IN ITERATIVE OPTIMIZATION-BASED SIMULATION (IOS)

Mohammad Dehghanimohammadabadi
Thomas K. Keyser

Department of Industrial Engineering and
Engineering Management
Western New England University
Springfield, MA, USA

ABSTRACT

In this paper an Iterative Optimization-based Simulation (IOS) framework is designed, developed and examined. This framework includes a threefold contribution of Simulation, Optimization and Database Manager. In this IOS model, optimization takes place repeatedly at the model's operational level to optimize the combination of system's state variables during the simulation run. Predefined trigger events momentarily pause the simulation and activate optimization in order to optimize the system's configuration. This cycle replicates until the simulation reaches its timespan. By deploying this promising IOS model, practitioners can take advantage of a long-term simulation run of their system, while it optimizing several times according to the occurrence of predefined incidents. The main concern here is the trade-off between simulation and optimization which is examined in this study. The results prove a positive impact of the IOS approach on the system's performance measures, although it takes longer to execute compared to the Non-IOS approaches.

1 INTRODUCTION

The dichotomy between simulation and optimization is fading gradually, as researchers are applying a balanced use of simulation and optimization Figueira and Almada-Lobo (2014). Since its inception, the Simulation-based Optimization (SO) approach has gained popularity quickly. SO provides a structured approach to optimize parameter values, where optimization is performed on a function of the output variables (steady state or transient) associated with a simulation model Swisher et al. (2004). Typically, SO methods have been widely applied in various industries and with computer advances, integration of these methods has seen remarkable advances. Figueira and Almada-Lobo (2014) has overviewed variety of well-known SO approaches and has proposed a taxonomy. According to this study, one major class of SO models is Iterative Optimization-based Simulation (IOS) which an optimization manager is embedded in a simulation agent. In the IOS approach, optimization would be called during simulation execution.

IOS models are not extensively studied and less than a handful of studies can be found in the literature. The aim of this paper is to review the existing studies and propose a generalize IOS framework. The proposed framework could be integrated with simulation software engines to effectively handle stochastic events in a simulation run. Therefore the main contributions of this paper are listed as follow:

- develop an Iterative Optimization-based Simulation (IOS) framework which can be applied to a wide range of complex systems, e.g. Manufacturing, Healthcare, etc.
- include the ability to call optimization with predictable and unpredictable trigger events in simulated time

- analyze the trade-off between execution time and cost of the model vs. Non-IOS approaches

Additionally, most of the optimization-simulation models are designed for short term planning with limited stochasticity. While, this novel IOS model, enables practitioners to take advantage of a long-term simulation run of their system while it has been optimized several times according to predefined incidents without simplifying assumptions.

As opposed to other papers that call optimization at predictable events during simulation run, this framework is able to trigger optimizer by occurrence of predictable and unpredictable events. For instance, in a manufacturing system, predictable event could be *job completion*, and unpredictable event could be *machine failure* or *new job arrival*.

The trial version of this framework has been coded in C# and has been integrated with SIMIO as simulation manager, MATLAB as optimizer and MySQL as database manager. Although most of the examples in this paper are referred to a manufacturing system, however, this promising approach can be applied in any stochastics system, where uncertainty prevails, e.g. healthcare systems, supply chain system, etc.

The remainder of this paper is organized as follows. In order to clarify the aim of this study and the gap that it is addressing, the literature review of the most recent IOS approaches are shown in Section 2. Section 3 is dedicated to the details of the framework and structure of different components of the model. The results of the proposed method are shown in Section 4 while Section 5 is devoted to summary and concluding remarks. The recommendations for future research of this study are stated in Section 6.

2 LITERATURE REVIEW

A large number of studies have been conducted on the subject of simulation optimization during last two decades. Figueira and Almada-Lobo (2014) recently categorized these methods into two different classes including *Solution Evaluation* and *Solution Generation* techniques. Solution evaluation approaches compare a set of solutions in solution space and will come up with the best or set of the best solutions. Some of the main methods of this class include statistical selection methods, metaheuristics, random search, stochastic approximation and reverse simulation technique.

The other type of SO models are called "*Solution Generation*" (SG) methods. SG approaches do not compare the solutions' advantages, but simply compute some variables which would be part of the whole solution generation. According to Figueira & Almada-Lobo, Solution Generation methods are subdivided into two categories: "*Solution Completion by Simulation*" (SCS) and "*Iterative Optimization-based Simulation*" (IOS). In the SCS, the optimizer discovers initial solutions under ideal conditions. Then, tries to achieve a better and more accurate value for a subset of initial solution's variables. While in the IOS techniques, optimization functions in a simulated system. The optimization takes place frequently at the model's operational level to optimize the combination of system configuration during the simulation timespan. In this method, the simulation module may either face a need for optimization or use a trigger event which calls for optimization. In fact, simulation momentarily halts itself and transfers the state of the system to the optimization manager, which solves an analytical problem that is accordingly formulated to account for the current system's state.

Very little, if any, has been reported in the literature relating to Interactive Optimization-based Simulation approaches. Jeong (2000) introduced the idea of condition-based events IOS which is called optimized simulation-based scheduling system (OSBSS). OSBSS seeks to find an optimized schedule with aid of the simulation optimizer by using Artificial Intelligence (AI) search techniques. In this framework the optimizer module interactively communicates with a simulation module to identify the improved dispatching rules. Subramanian et al. (2000) used an IOS approach to solve a stochastic optimization related to the management of a Research and Development (R&D) pipeline problem (cross-project management of all tasks associated with a particular set of objectives with a limited pool of resources). Subramanian et al. extended their earlier study by developing a twin-loop computational

architecture, which integrates mathematical programming and discrete event simulation Subramanian et al. (2003). The inner loop of this computational architecture does simulation-optimization of timelines, which later can be fitted in the outer loop, called risk-control loop. The latter, obtains improvements in the solutions to the underlying stochastic optimization problem.

Gupta and Sivakumar (2005) presented a Conjunctive Simulated Scheduling (CSS) scheme which combines the concepts of simulation and scheduling of a shop. Here, as soon as a resource is released, a new job has to be selected as the next operation on this resource. At this decision point, scheduling of jobs is done and the simulated clock is progressed forward. In this study, simulation-based scheduling signals optimization just by occurrence of predictable events in simulated time which are job completions.

Mejtsky (2007) described a metaheuristic algorithm for simulation optimization which carries out simulation runs simultaneously and evaluates different simulation runs during their execution before the end is reached. This algorithm leverages a branching approach which mirrors the decision tree of an optimization problem. Kulkarni and Venkateswaran (2014) developed an iterative simulation and optimization approach where a discrete-event simulation model is interfaced with a meta-heuristic based solver. Sivakumar (2001) designed, developed and implemented a discrete event simulation-based "on-line near-realtime" dynamic multiobjective scheduling system. This model dynamically generates schedules in a manner to achieve Pareto optimal cycle time distribution and machine utilization. In their model, for short–term analysis, simulation is initiated by using a deterministic approach with defined rules and policies.

3 METHODOLOGY

The IOS approach that is designed, developed and implemented in this research includes a threefold contribution of Simulation, Optimization and Database Managers. These system managers operate in harmony to achieve the goal of selecting the best system configurations (e.g. scheduling, resource allocation, etc.) for different states of the system. In this approach, the simulation manager (Simio ^{TM}) marches through the time until any of the predefined trigger events occur and halts the simulation momentarily. When a trigger event occurs, a snapshot of the manufacturing system will be taken in order to store the current state parameters of the manufacturing system into the database (MySQL).

This approach yields an optimized status of the entire environment upon a stochastic event occurrence. The main concern here is the trade-off between simulation and optimization which can occur either moment-by-moment, in periodic scheme or on an event-driven basis. In moment-by-moment IOS, glimpses of the simulation's run constantly transfer to the optimizer for every single event within the simulation run, while in periodic scheme this could occur on a regular basis. However, the most applicable and realistic type of these techniques is event-driven basis or trigger-based IOS. These events will be activated during the simulation run once the status of the system changes and needs optimization. Therefore, simulation will momentarily pause itself and transfer the state of the system and variable attributes to the optimization manager. The optimization manager solves an analytical problem and sends the results back to the simulation. In order to have better understanding of the proposed framework, all of examples are provided here are drawn for a manufacturing system. However, this does not limit the model's practical usefulness for manufacturing system and could be applicable to any kinds stochastic environment.

This framework is capable to signal the optimization with either predictable or unpredictable events with simulated time. The predictable events involve a certain amount of predictability whose are usually controllable in real world systems by users, e.g. shift schedule optimization or schedule optimization after each job completion. Unpredictable events in the system are those random events that are not necessarily under the user's control and require quick response in a real-world system e.g. machine failure, machine repair or new job arrival. Therefore, this framework is designed in such a way that could mimic any type of events that require system optimization in the real world environment.

Two different types of trigger events are considered in this study. The first type of trigger events are limit-based events, e.g. jobs' tardiness or machines' utilizations. These monitor the performance of the system and trigger the optimization manager upon limit being exceeded circumstances. The second type of triggers are event-based, e.g. machine breakdown or new demand, which upon their occurrence cause the simulation to pause and call for optimization. Several possible events are listed in Table 1. The details of these three managers and their integration are described as follows.

3.1 Simulation Manager

In any organization, leadership is the most important factor and its support is crucial for sustaining continuous improvement in organizations (Dehghanimohammadabadi and Keyser 2014). Simulation modeling is pivotal element in any improvement changes because it empowers researchers to reduce the risk and cost of the changes. Simulation determines the ideal state before touching the real-world system which could be a sufficient justification to convince managers to adopt improvements.

Table 1: List of possible events which act as an optimization trigger for a manufacturing system.

Type	Event	Parameters	Description
Event-driven basis	Machine Breakdowns	MTBF MTTR	If a machine breaks down or repairs, the number of available resources changes and the manufacturing system needs to be reconfigured.
	Preventive Maintenance	Maintenance Schedule	Due to the Preventive Maintenance (PM) plans, machines can be unavailable during certain periods.
	Demand Pattern	Demand's rate	Demand's fluctuation signals the optimizer to adopt the manufacturing system configuration accordingly.
Limit based events	Unexpected Tardiness	USL LSL	The average tardiness is monitored upon the most recent finished entities and, if it is beyond limits, the optimizer will be called.
	Machine queue length	Threshold	A threshold limit could be set up to alarm the simulation once average queue length of the server(s) intensifies.
	Machine Utilization	MnMU MxMU	dropping the machine utilization below Minimum Machine Utilization (MnMU) or above Maximum

In this methodology Simio™ is utilized as the simulation manager. Simio stands for "SIMulation modeling framework based on Intelligent Objects" (Pegden 2007). The main reason that Simio is considered for this study is that the software is programmed in C# language and is compatible with any other C# modules. This *"Application Programmers Interface"* (API) allows the users to customize or extend their designed model properly. The extension could be adding new steps, elements and rules, importing and exporting data, enhancing experimentation with external algorithms, or interfacing from external programs. For this study, a few APIs are utilized or programmed in order to integrate the simulation manager with other components of the model. A few of these steps are listed as follows:

- DbExecute, DbRead and DbWrite step instances are used in order to store the simulation attributes to the MySQL database. The frequency of data transfer can be adjusted by the user.
- A "user defined" step instance called "OptTrig" is programmed to launch the optimization manager whenever needed. In this step, the occurrence of any predefined event along the simulation run, will halt the simulation temporarily and execute the optimization manager. It needs to be stated that, in addition to the previously mentioned steps, several other steps are required to be embedded into the simulation modeling in order to detect the trigger events.

The conceptual algorithm of this framework is explicitly described in Table 2 and includes two main phases. In phase 1, prior to simulation initialization, substantial supporting information such as resources, jobs, processing times, setup time, due dates etc. will be stored to the database manager. This information is usually available from Enterprise Resource Planning (ERP) systems. Then, the optimizer calculates the initial optimal scheduling and updates the second layer of the database (L2). At this point, the simulation has the first optimal solution and is ready to run. The real IOS scheme occurs through the second phase of the model. Any pre-defined event will trigger the framework to dynamically update the scheduling of jobs in the system. The details of the framework system and relationship between managers are described as following.

3.2 Database Manager

The use of a database is easily motivated, especially with large amounts of data and supports the convenient preservation and management of data (Syberfeldt et al. 2013). A database manager is utilized in this paper in order to store and retrieve optimization results and simulation system attributes. The Database manager in this framework is the single point of interaction between simulation and optimization manager and plays a "connecting chain" role in this model. In case of implementing this IOS framework for real-world systems, the ERP data could exported to the database. Moreover, if simulation runs for long-term strategic planning, database could store huge amount of data regarding system attributes and all of the achieved optimal solutions during simulation run.

This database is composed of two layers, L1 and L2. The first layer (L1) is devoted to relevant data from the simulation manager, which would be utilized later as inputs for the mathematical modeling. In the real world systems, the required data may either derive from system sensors or the output of a high-fidelity numerical simulation model (Shi and Zhou 2009). In this model, an abundance of instant data will be readily available for further analysis by embedding proper Simio steps (DbWrite, DbRread and DbExecute) during the simulation runs. This data is composed of several attributes and variables from different states of the system, which will be taken by snapshotting the manufacturing system along different states. This data will be used for the current manufacturing state assessment via optimization. The L2 stores the optimization solutions to be fed into the simulation model and updates the manufacturing system setting. All of the aforementioned data persist in database until simulation completes. The MySQL database is integrated to the model because it is fast enough for our framework, it is reliable free of charge and equally important, has an available API for the C# programming language.

Table 2: IOS framework algorithm: A manufacturing system example.

Initial Phase	1: Update the first layer of database (jobs, machines, stages, processing and setup times, due dates, etc.) 2: Run optimization model. 3: Derive initial scheduling solution. 4: Update the second layer of database (Assign jobs to available machines and priorities). 5: Feed the solution to the simulation model. 6: Run simulation model.
IOS Phase	7: **while** unprocessed-jobs != 0 **do** 7.1: **If** trigger-event-occurs **then** 7.1.1:Pause the simulation. 7.1.2:Take snapshot of the simulation 7.1.3:Update database (unprocessed jobs, available machines). 7.1.4:Form new mathematical modeling based on unprocessed jobs and

available machines.
7.1.5:Run optimization model.
7.1.6:Derive new scheduling solution.
7.1.7:Update the L2, the second layer of database (Assign jobs to available machines and priorities).
7.1.8:Feed the solution to the simulation model.
7.1.9:Run simulation model.
7.2: **end if**
8: **end while**
9: Return the simulation results (Cmax, Machine utilizations, WIP, etc.).

3.3 Optimization Manager

One of the advantages of this research is that the optimization manager is independent from a specific optimization method. Varieties of analytical tools can be plugged into the model and function as mathematical solvers. Once the trigger event signals the optimization manager, the required data will be taken from first layer of the database and will be fed to the optimization manager.

A Parallel Machine Scheduling (PMS) problem has been modeled. In the classical PMS problem, there are n number of jobs which can be operated on any of m available machines with specific processing time and without preemption. The objective is to find the optimal schedule in terms of certain performance measures of the system. MATLAB is used to develop the mathematical modeling and optimization algorithm of a single-stage parallel machine scheduling. The first algorithm that is used is Simulated Annealing (SA) which provides near to optimal solution. SA was developed by Kirkpatrick, Gelatt, and Vecchi (1983) and its name is inspired by annealing from metallurgy. SA has shown successful applications in a wide range of combinatorial optimization problems, and this fact has motivated researchers to use simulated annealing in many simulation-optimization problems (Rai and Ettam 2013). This algorithm starts with an initial feasible solution, then a cooling schedule will be utilized to move from one solution to another for identifying the optimal solution (Rai and Ettam 2013). The mathematical modeling of the current problem is presented follows:

3.3.1 Assumptions

Several assumptions are assumed in this case which are stated as follows:

- All jobs and available machines are ready to be scheduled in time zero.
- Preemption of operations of each job is not allowed.
- The setup times are sequence-dependent, which means the setup time varies from one job to other job on each machine.
- Different jobs have different processing time on each machine.
- Each machine can process only one operation at a time.
- Each job has a distinct due date and must be processed only one time.
- Each job can be processed only by one free machine.
- All machines are unrelated.

In this method, once the triggers take place, the simulation is paused and the manufacturing system configuration is controlled.

3.3.2 Notation

Subscripts

N	The number of Jobs
M	The number of Machines
i, j	Index for job $(i, j = 1,2, \dots, N)$
m	Index for machine $(m = 1,2, \dots, M)$

Input parameters

P_{im}	Processing time of job i on machine m
α_i	The earliness unit penalty of job i
β_i	The tardiness penalty of job i
d_i	Due date of job i
λ	Factory cost per time unit (including variable and invariable costs)
S_{jim}	Setup time for assigning job i after job k on machine m
w_1	Normalized weight of makespan
w_2	Normalized weight of earliness/tardiness

Decision variables

C_i	Completion time of job i
C_{max}	Total completion time or makespan
E_i	Earliness of job i; $E_i = \max\{0, d_i - C_i\}$
T_i	Tardiness of job i; $T_i = \max\{0, C_i - d_i\}$
y_{ijm}	1 if job i on machine m precedes job j; otherwise, it is zero.

3.3.3 The mathematical model

$$\min Z = \min\left(w_1 \times \lambda C_{max} + w_2 \times \sum_{i=1}^{N} \alpha_i E_i + \beta_i T_i\right) \quad (1)$$

$$M \times y_{ijm} + \left(C_i - C_j\right) + S_{ijm} \leq p_{im} \qquad \forall\, i, j \in N;\ \forall m \in M; \quad (2)$$

$$M \times (1 - y_{ijm}) + \left(C_j - C_i\right) + S_{jim} \leq p_{jm} \qquad \forall\, i, j \in N;\ \forall m \in M; \quad (3)$$

$$C_i - d_i \leq T_i \qquad \forall i \in N; \quad (4)$$

$$d_i - C_i \leq E_i \qquad \forall i \in N; \quad (5)$$

$$C_{max} \geq C_i \qquad \forall i \in N; \quad (6)$$

$$y_{ijm} \in \{0,1\} \qquad \forall\, i, j \in N;\ \forall m \in M; \quad (7)$$

$$T_i, E_i, C_i \geq 0 \qquad \forall i \in N; \quad (8)$$

Equation (1) is the objective function which aims to minimize makespan (c_{max}) and the total jobs' earliness or tardiness cost at the same time. Inequality (2) and (3) impose the restriction that job i precedes the job j or job j precedes job I on machine m. Earliness and tardiness of job i are indicated at equations (4) and (5). Equation (6) is used to calculate makespan which is the max of all of machines completion time. Last two equations, constraints (7) and (8) are used to identify the binary variables and non-negativity of decision variables.

4 RESULTS

In this study the tradeoff between objective measures of system performance and execution speed is explored. As is shown in Figure 1, these experiments are implemented on a simulated single-stage manufacturing system with four parallel non-identical machines. In this simulation model, jobs are processed in priority basis with different processing times and sequenced-dependent setup time.

A generalized factorial experiment with two factors and up to three levels is designed. The first factor is "Type of Simulation" which includes IOS compared with two Non-IOS techniques, Shortest Processing Time (SPT) and Earliest Due Date (EDD), which are common heuristic dispatching rules. In the Non-IOS approaches, simulation marches through the time until reaching the model timespan. In SPT, whenever a machine is available, the shortest job for that particular machine is handled first and completed, while in EDD, upon machine availability, the job with earliest due date is selected. In these two Non-IOS simulation types, no trigger event has been embedded into the system which makes simulation run faster. While, for IOS technique, in each optimization iteration, the SA algorithm solves a scheduling problem where the objective function is the weighted summation of makespan and earliness/tardiness cost. In this experiment, event-driven basis IOS approach is used with two triggers. These triggers are machines failures and repairs which both change the number of available machines in the system. By changing the number of available machines, the simulation manager initiates the optimization and reschedules all of the unprocessed jobs according to the fresh optimal solution. For instance, as is depicted in Figure 1, if Machine 1 fails, other machines take over processing for it.

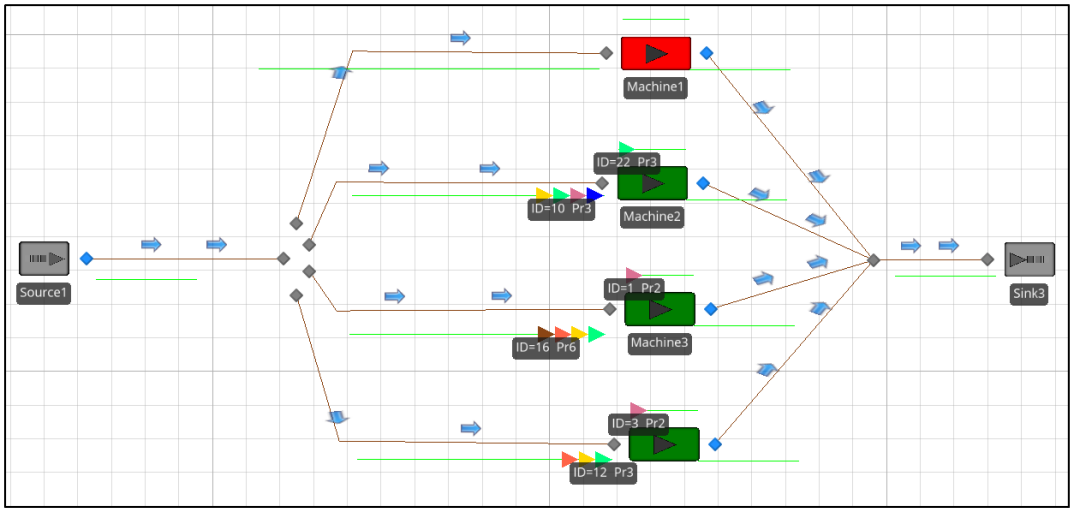

Figure 1: Layout of the simulation model.

Another factor considered is the "Number of Jobs" in the system with three levels: 20, 40 and 60 jobs. In this experiment, the responses of interest are "Execution Time" of simulation and maximum completion time or Cmax, Work In Process (WIP), and cost of earliness/tardiness of the jobs. The original design in Table 3, shows one replication of the generalized factorial experiment design. The first two columns of data, correspond to the factors of experiment with three types of simulation and three levels of Number of Jobs. The yields of experiment are displayed in last four columns of the table.

The results of this experiment indicate that, the IOS framework clearly will affect the "Execution Time" of simulation run. Implementing the IOS approach, as was expected, makes the simulation engine slower since multiple optimizations occur within each simulation iteration. As shown in Figure 2, the difference of Execution Time between the approaches becomes more apparent when number of jobs in

the system increases. When the Number of jobs is 20, this difference is negligible, while this gap becomes larger when the Number of Jobs increases to 60.

The impact of the IOS model on the system's performance measures is examined and a few insights are obtained. The first is that, as it appears in Figure 3(a) and 3(b), the horizontal reference lines indicate the positive impact of the IOS on WIP and Cmax compared with SPT and EDD. The reason is, in IOS, optimization manager provides an optimal schedules by considering sequence-based setup time and processing time, while SPT takes into account just the shortest processing time. As is shown in Figure 3(c), there is a visible improvement in tardiness/earliness cost of the jobs as well. The reason that the IOS outperforms the EDD is that, the optimization manager takes cost-effectiveness into account while calculating both tardiness cost and earliness cost of each job.

Table 3: The generalized factorial design of the experiment.

Factors		**Responses of Interest**			
Simulation Type	Number of Jobs	Execution Time (Min)	Cmax (Min)	WIP	Earliness / Tardiness Cost
EDD	20	0.3	140.1	1322.1	7503.5
EDD	40	0.9	233.4	4444.4	28821.6
EDD	60	1.9	342.4	10282.7	79508.7
IOS	20	2.1	74.9	849.7	4760.8
IOS	40	13.1	130.2	2723.4	14360.1
IOS	60	27.3	177.0	5435.7	34652.4
SPT	20	0.4	107.1	1084.2	7670.7
SPT	40	1.4	234.2	4415.3	30740.1
SPT	60	2.1	352.0	10328.1	79184.5

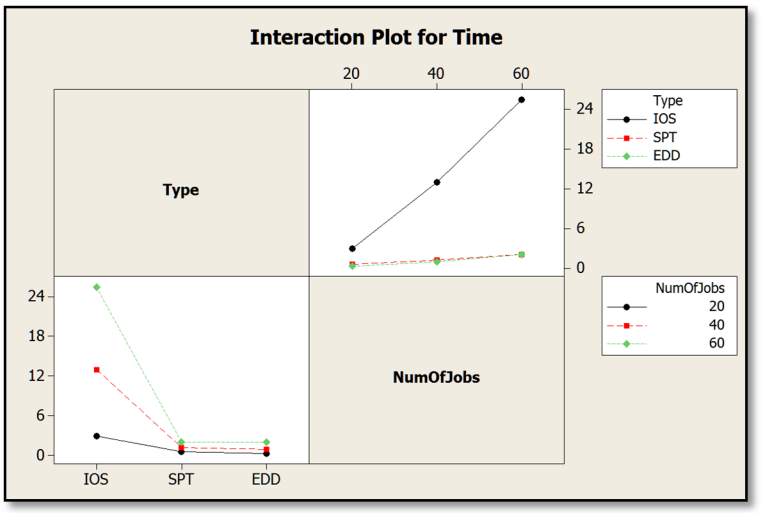

Figure 2: Effects of experiment factors on Execution Time of the simulation.

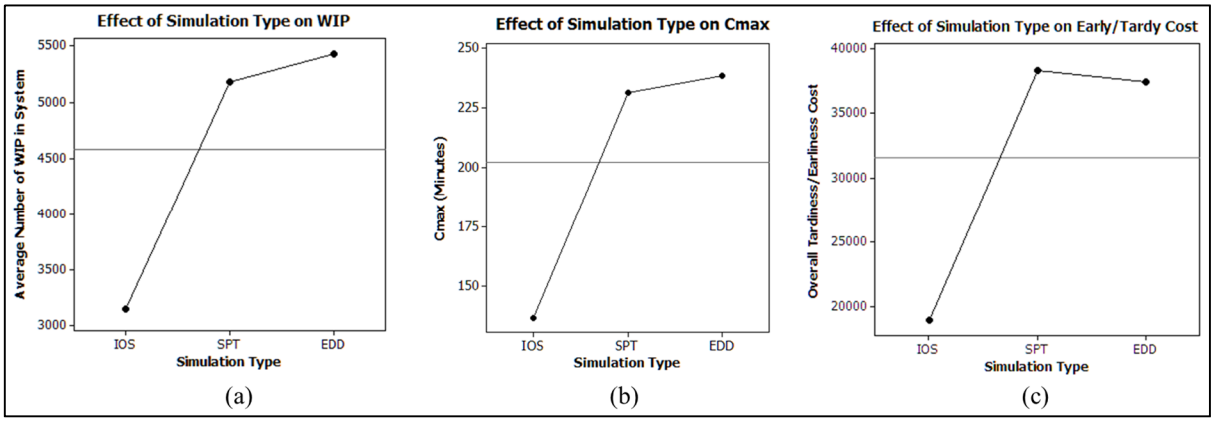

Figure 3: Effect of IOS and Non-IOS simulation on manufacturing metrics.

The impact of the second factor of this study, Number of Jobs in the system, is illustrated in Figure 4, as expected, the results prove that, by increasing the Number of Jobs, all of the system performance metrics decline. By changing Number of Job from 20 to 60, not only Execution Time rises, but also WIP, Cmax and earliness/tardiness cost remarkably increase.

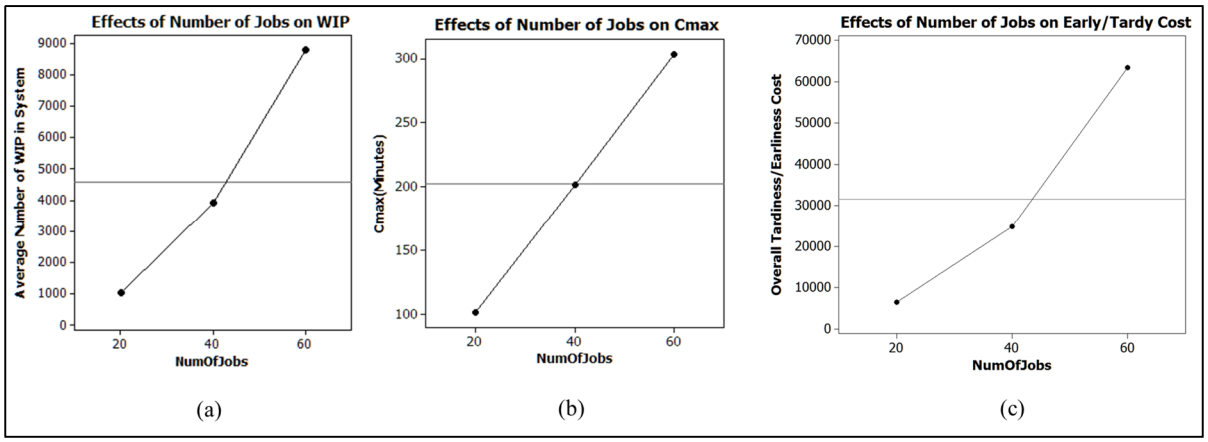

Figure 4: Effect of Number of Jobs on manufacturing metrics.

5 DISCUSSION AND CONCLUSION

The proposed IOS framework is novel from many perspectives. First, the introduced method leverages the capability of a *Data Base Management System* (DBMS) for transferring the data for the both simulation and optimization managers. This model is able to consider parameters such as: the state of the system at upstream and downstream locations by reading from a continually updated database. Another benefit of the model is independency of the simulation and optimization managers, which are performing separately across a local network over multiple CPUs.

This promising approach can be applied in any stochastic system, where uncertainty prevails. According to Rogers and Flanagan, in this kind of environment, it may be beneficial to change the way the shop is controlled at certain points in time (Rogers and Flanagan 1991). Therefore, this method forms a simulation-optimization model which is consistent with the real incidents that occur in the stochastic systems in real-time. Most of the examples provided in this study refer to manufacturing system, while this model is applicable to variety of stochastic systems. For instance, one can apply this framework to

simulate a healthcare system while the user uses optimization module to get an optimal scheduling of the patients for physicians. the possible trigger events could be new patient arrival, call in sick of two physicians or even out of limit patients waiting time.

Deploying this favorable IOS model, improves the accuracy of the system's simulation analysis for long-term planning, while it optimizes several times without simplifying the assumptions. One can verify the accuracy of the IOS model by comparing the model's results such as Cmax, tardiness cost, WIP, etc., with the actual manufacturing system performance.

One concern associated with this method is the potential for slow execution. In this study, the tradeoffs between objective measures and execution speed are examined. The experiment results indicate that, the IOS model compared to the Non-IOS approaches (SPT and EDD) has longer "Execution Time" but progressively improves the system's performance. It needs to be mentioned that, the Execution Time difference between these two approaches becomes more apparent when the Number of Jobs in the system increases. In the case of 20 jobs, almost no difference exists between the IOS and the Non-IOS in terms of Execution Time, but the IOS yields better performance measures. Whereas, this difference gap becomes more evident when the Number of Jobs increases to 60.

6 FUTURE WORKS

This approach is speculative and needs further evaluation before implementation in a real-world system. The data in the current database can be substituted with real data from ERP systems. The suggested framework can be used as the basis for future studies to enhance the discrete event simulation engine. As a suggestion for future research, the optimization manager can also utilize the optimization historical results to boost the optimization process, by mining common patterns in the data. Also, multiple optimization algorithms could be embedded into the platform in order to evaluate different mathematical problems. One can investigate the efficiency of various optimizers and select the most appropriate one.

REFERENCES

Dehghanimohammadabadi, M., and Thomas, K. 2014. "Does the Iranian National Productivity and Excellence Award Get Leadership Buy-In." *In Proceeding of 2014 Annual IIE Conference*, Montreal, QC. http://www.xcdsystem.com/iie2014/abstract/finalpapers/I994.pdf.

Figueira, G., and Bernardo, A. 2014. "Hybrid Simulation–optimization Methods: A Taxonomy and Discussion." *Simulation Modelling Practice and Theory*. http://www.sciencedirect.com/science/article/pii/S1569190X14000458.

Gupta, A., and Appa I. S. 2005. "Conjunctive Simulated Scheduling." *The International Journal of Advanced Manufacturing Technology* 26 (11-12): 1409–13.

Jeong, K. 2000. "Conceptual Frame for Development of Optimized Simulation-Based Scheduling Systems." *Expert Systems with Applications* 18 (4): 299–306.

Kirkpatrick, S., C. D. Gelatt, M. P. Vecchi, and others. 1983. "Optimization by Simmulated Annealing." *Science* 220 (4598): 671–80.

Kulkarni, K., and Jayendran V. 2014. "Iterative Simulation and Optimization Approach for Job Shop Scheduling." *In Proceedings of the 2014 Winter Simulation Conference*, edited by A. Tolk, S. Y. Diallo, I. O. Ryzhov, L. Yilmaz, S. Buckley, and J. A. Miller, 1620–31. Savannah, Georgia: IEEE Press. http://dl.acm.org/citation.cfm?id=2694054.

Mejtsky, G. J.. 2007. "A Metaheuristic Algorithm for Simultaneous Simulation Optimization and Applications to Traveling Salesman and Job Shop Scheduling with Due Dates." *In Proceedings of the 39th Conference on Winter Simulation: 40 Years! The Best Is yet to Come*, edited by S. G. Henderson, B. Biller, M. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, 1835–43. Washington, D.C., IEEE Press. http://dl.acm.org/citation.cfm?id=1351871.

Pegden, C. D. 2007. "SIMIO: A New Simulation System Based on Intelligent Objects." *In Proceedings of the 39th Conference on Winter Simulation: 40 Years! The Best Is yet to Come*, edited by S. G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, 2293–2300. Washington, D.C., IEEE Press. http://dl.acm.org/citation.cfm?id=1351948.

Rai, S., and Ranjit K. E. 2013. "Simulation-Based Optimization Using Simulated Annealing for Optimal Equipment Selection within Print Production Environments" *In Proceedings of the 2013 Winter Simulation Conference*, edited by R. Pasupathy, S.-H. Kim, A. Tolk, R. Hill, and M. E. Kuhl, 1097–1108. Washington, D.C., IEEE Press. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6721499.

Rogers, P., and Maureen, T. F. 1991. "Online Simulation for Real-Time Scheduling of Manufacturing Systems." *Industrial Engineering* 23 (12): 37–40.

Shi, J., and Shiyu, Z. 2009. "Quality Control and Improvement for Multistage Systems: A Survey." *IIE Transactions* 41 (9): 744–53.

Sivakumar, A. I. 2001. "Multiobjective Dynamic Scheduling Using Discrete Event Simulation." *International Journal of Computer Integrated Manufacturing* 14 (2): 154–67.

Subramanian, D., Joseph F. P., and Gintaras V. R. 2000. "A Simulation—optimization Framework for Addressing Combinatorial and Stochastic Aspects of an R&D Pipeline Management Problem." *Computers & Chemical Engineering* 24 (2): 1005–11.

Subramanian, D., Joseph F. P., Gintaras V. R., and Gary E. B.. 2003. "Simulation-Optimization Framework for Stochastic Optimization of R&D Pipeline Management." *AIChE Journal* 49 (1): 96–112.

Swisher, J. R., Paul D. H., Sheldon, H. J., and Lee W. S. 2004. "A Survey of Recent Advances in Discrete Input Parameter Discrete-Event Simulation Optimization." *IIE Transactions* 36 (6): 591–600.

Syberfeldt, A., Ingemar, K., Amos, N., Joakim S., and Torgny A. 2013. "A Web-Based Platform for the Simulation–optimization of Industrial Problems." *Computers & Industrial Engineering* 64 (4): 987–98.

AUTHOR BIOGRAPHIES

Mohammad Dehghanimohammadabadi is a PhD candidate in Engineering Management, Western New England University, MA, USA. This article is part of his PhD dissertation which is related to develop a new Iterative Optimization-based simulation method. In this research, he is seeking to add a new feature to the simulation software packages by running an optimizer manager through a simulation run. His Email address is mohammad.dehghani@wne.edu.

Thomas K. Keyser is a Professor of Industrial Engineering and Engineering Management, Western New England University, MA, USA. He has published over 30 works in multiple journals and conference proceeding and has done several projects related to the simulation. He has received funding from organizations such as National Science Foundation, National Institute of Standards and Technology, GTE, and General Electric Aircraft Engines. His Email address is thomas.keyser@wne.edu.