Manual for the Make-To-Stock Manufacturing Systems Simulator

Christos Panayiotou
Dept. of Electrical and Computer Engineering,
University of Cyprus,
Cyprus.
christosp@ucy.ac.cy

April 26, 2005

Abstract

This Java simulator includes a discrete-event model for the manufacturing system shown in Fig. 1. In addition, it includes optimization modules that run Infinitesimal Perturbation Analysis to get sensitivity estimates which are then used together with stochastic approximation to solve constrained and unconstrained optimization problems.

1 Model

We consider the make-to-stock (MTS) system shown in Fig. 1. The upstream machine $M2$ can produce parts at a rate $\mu_2(t)$ and is assumed to have an infinite supply. The output of $M2$ is placed in the work-in-process buffer W which has capacity θ_W. Machine $M1$ processes parts from W at a rate $\mu_1(t)$ and fills the finished part inventory buffer F. Finally, F is drained by the demand at a rate $\rho(t)$. Functions $\mu_1(t)$, $\mu_2(t)$ and $\rho(t)$ are assumed piecewise constant.

Let $(x(t), y(t))$ denote the state of the system, where $x(t) \leq \theta_F$, $0 \leq y(t) \leq \theta_W$ correspond to the number of parts present in F and W respectively. Note that $x(t)$ can take negative values and these correspond to the amount of backlogged demand.

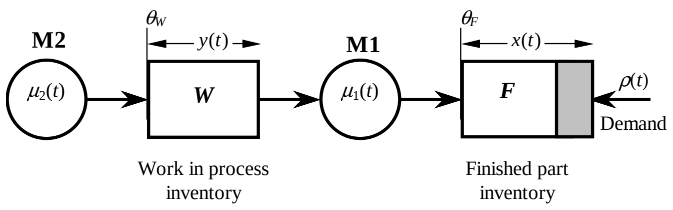

Figure 1: System Model

1.1 Objective

In the context of MTS manufacturing systems we consider the sample function $L_T(\theta_F, \theta_W)$ which in general has the form

$$L_T(\theta_F, \theta_W) = \frac{1}{T}\int_0^T \left(c_F[x(t)]^+ + c_W y(t) + c_D[-x(t)]^+\right) dt. \tag{1}$$

The first two terms correspond to the average inventory cost in F and W respectively and the third one corresponds to the average cost of backlogged demand. Furthermore, $[x]^+ = \max\{0, x\}$ and c_F, c_W, c_D are non-negative constants. c_F and c_W correspond to the inventory cost per part per unit time in the buffers F and W respectively. c_D is the backlog demand cost per part per unit time.

2 Installing and Running the Simulator

In a command line window type

```
D:\Temp\mts>java -jar MTSSimulator.jar mtssimulator.mts
```

where `D:\Temp\mts>` is the location where the MTSSimulator.jar file is saved.

3 Software Options

From the *Model* menu item you can select the model to be used. The two possible models are a detailed *discrete event simulation* (DES) model or a *discrete time* (DT) model. The default is a DT model.

The cost parameters c_F, c_W, and c_D can be changed from the *Cost* menu item. The default values are $c_F = 1$, $c_W = 1$, and $c_D = 0$.

3.1 Run Menu Item

From the *Run* menu item it is possible to run simulations and optimization algorithms for minimizing the $L_T(\theta_F, \theta_W)$ given in (1).

From the *Simulator Test* you can run a single simulation for a given buffer capacity. *Derivative Test* is similar to Simulator Test but it also provides derivative estimates with respect to the buffer capacities.

Plot cost surface runs simulations for various buffer capacities in a given range. The recorded cost and stockout probability is the average of the selected number of simulation runs. The results are saved in a file that can be easily plotted using Excel.

From the *Optimization* menu item you can select various optimization algorithms that minimize $L_T(\theta_F, \theta_W)$.

No Constraint: There is no explicit QoS constraint. In this case $c_D > 0$ otherwise the optimal is at the origin.

$$L_T^U(\theta_F, \theta_W) = L_T(\theta_F, \theta_W) \tag{2}$$

Penalty Method: Minimizes the unconstraint function

$$L_T^P(\theta_F, \theta_W) = L_T(\theta_F, \theta_W) + \frac{p}{2}\| \Pr[x(t) < 0] - \epsilon\|^2 \tag{3}$$

Lagrangian Relaxation: Minimizes the unconstraint function

$$L_T^L(\theta_F, \theta_W) = L_T(\theta_F, \theta_W) + \lambda \max\{0, \ \Pr[x(t) < 0] - \epsilon\} \tag{4}$$

Multiplier Method: Minimizes the unconstraint function

$$L_T^M(\theta_F, \theta_W) = L_T(\theta_F, \theta_W) + \lambda\left(\Pr[x(t) < 0] - \epsilon\right) + \frac{p}{2}\left\|\Pr[x(t) < 0] - \epsilon\right\|^2 \tag{5}$$

For all algorithms, the buffer thresholds are updated using gradient descent:

$$\begin{aligned} \theta_F^{k+1} &= \theta_F^k - \alpha\frac{\partial L_T^i(\theta_F^k, \theta_W^k)}{\partial \theta_F} \quad (6) \\ \theta_W^{k+1} &= \theta_W^k - \alpha\frac{\partial L_T^i(\theta_F^k, \theta_W^k)}{\partial \theta_W} \quad (7) \end{aligned}$$

where $i = U, P, L, M$, and α is a constant step size. For the *Penalty* and *Multiplier* methods the penalty is increased linearly

$$p^{k+1} = p^k + \delta p \tag{8}$$

where $\delta p > 0$ is a given constant selected by the user. For the Multiplier method, λ is updated using the following equation:

$$\lambda^{k+1} = \lambda^k + p^k\left(\Pr[x(t) < 0] - \epsilon\right) \tag{9}$$

All optimization algorithm end when the derivative magnitude becomes less than a user selected value (minimum derivative).

4 Distributions

The user can also change the demand and machine processing distributions. The default is the one from Yong's report. Other options are the following

Exponential: Parameter: mean event lifetime (NOT rate=1/mean).

Erlang: Parameters: mean and Erlang order.

Uniform: Parameter: Lifetime from the interval [min, max].

2-State MMPP: Returns random variates from 2 exponential distributions. Each exponential distribution has a given mean. Furthermore, the exponential variate to be generated is determined from the state of a 2-state Markov chain. State changes occur with certain rates that are also parameters of the distribution.

2-State MMCP: Returns either of 2 (deterministic) event lifetimes. The returned variate to be generated is determined from the state of a 2-state Markov chain. State changes occur with certain rates that are also parameters of the distribution.

2-State DT-MMPP: Discrete time MMPP. At every time slot (of length 1) it generates a number of events that is Poisson distributed with a rate that depends on the state of the Markov chain. The returned lifetimes are such that the events are uniformly distributed in the time slot. State changes can occur at the beginning of the time slot with a given state transition probabilities (parameters of the distribution).

2-State DT-MMPP-E: Similar to the 2-State DT-MMPP above, except that all lifetimes are such that all events occur at the end of the time slot.

2-State DT-MMCP: Discrete time MMCP. At every time slot (of length 1) it generates a deterministic number of events which depends on the state of the Markov chain. The returned lifetimes are such that the events are uniformly distributed in the time slot. State changes can occur at the beginning of the time slot with a given state transition probabilities (parameters of the distribution).

2-State DT-MMCP-E: Similar to the 2-State DT-MMCP above, except that all lifetimes are such that all events occur at the end of the time slot.

Deterministic: Returns a deterministic lifetime.

Hyperexponential: Returns an exponentially distributed lifetime with either of 2 means, each with a given probability.

Gaussian: Normally distributed lifetimes with a given mean and standard deviation