8th International IFAC Symposium on
Dynamics and Control of Process Systems
Preprints Vol.3, June 6-8, 2007, Cancún, Mexico

COMPLEX DECISION MAKING AT A RE-ENTRANT FLOW STATION UNDER UNCERTAINTY

Rakshita Agrawal *, Jay H. Lee *, Matthew J. Realff *

* *Department of Chemical and Biomolecular Engineering,*
Georgia Institute of Technology,
Atlanta, GA 30332-0100, USA

Abstract: This work is aimed at studying the complex relationships between vital decisions in manufacturing environments facing uncertainty. Maintenance, testing and job scheduling decisions are inter-related and their relative costs govern the decision making. Presented here is a formal mathematical representation of the integrated decision making problem and a discussion on the computational complexity. Furthermore, a systematic decomposition/simplification of the problem is proposed, whenever applicable. An exemplary system of re-entrant flow station is chosen for illustrations. Rigorous probability theories, Approximate Dynamic Programming and Heuristic rules are combined, to come up with good decision making. *Cpyright © 2007 IFAC*

Keywords: Markov Decision Processes, Dynamic Programming, Scheduling, Uncertainty

1. INTRODUCTION

Most manufacturing environments are faced with decision making under uncertainty. Moreover, most of the crucial decisions like machine maintenance scheduling, job scheduling, testing, etc. are interdependent. In complex manufacturing environments like semiconductor wafer fabs, delays due to testing and shutdowns translate directly into huge loss in revenue. Despite the obvious inter-dependence, these decisions are mostly treated independent of one another, in the existing literature.

General machine maintenance problems have been formulated and solved as Markov Decision Processes (*MDP*). In case of preventive maintenance, the states are not fully observable. This problem is discussed in (Smallwood and Sondik, 1973). Past researches in wafer-fabrication scheduling are mostly aimed at global job-shop scheduling. Gupta and Shivakumar (2006) provide a comprehensive overview of scheduling techniques in a semiconductor manufacturing process. Attempts have also been made in the past to capture, specifically, the re-entrant flow structure of the problem. Choi and Reveliotis (2003) provide an analytical formulation for the scheduling problem in re-entrant lines, using a generalized stochastic Petri Net model. Shen and Leachman (2003) have proposed a stochastic dynamic programming model for scheduling new releases. They have captured the re-entrant flow structure characteristic of wafer manufacturing by a stochastic linear quadratic (SLQ) model. To address the fact that, not every intermediate can be tested during manufacturing, many statistical control studies have been conducted for optimal sampling policies. One such work is by Nurani et al. (1994). They have suggested ways to develop an optimal sampling strategy for defect inspection in semi-conductor wafers using real life data from different fabs. Their work gives useful insights into modeling of process drifts and defect-yield relationship. With the rich literature available on independent studies of these decisions, we adopt the objective of studying how these decisions affect one another. By means of a hypothetical one machine problem with re-entrant flow, we analyze how combining the decisions would result in better overall performance. We also propose a new adaptive grid based method for solving Partially Observable Markov decision processes (*POMDPs*) in order to improve upon the performance of existing heuristic based methods.

The rest of the paper is organized as follows. Section 2 contains a detailed description of the system.

Section 3 presents the mathematical formulation and the computational complexity of the resulting problem. In section 4 we talk about the proposed solution methods and present the conclusions in section 5.

2. PROBLEM DESCRIPTION

2.1 System details

A re-entrant flow shop is characterized by a job going through the same operation more than once. Thus, jobs at various stages of operation compete for the same resources. For the present work (please see Fig. 1), a machine is selected that is capable of depositing identical layers, one layer at a time. It operates on a single wafer at a time. The product of interest is a wafer with 3 identical layers deposited. So the wafers are fed one at a time. While one wafer is being operated on, the others are waiting at the entry point called 'queue'. The queue at any time can contain 3 types of intermediate/unprocessed jobs
a_0 = bare wafer or unprocessed job
a_1 = wafer with 1 layer
a_2 = wafer with 2 layers

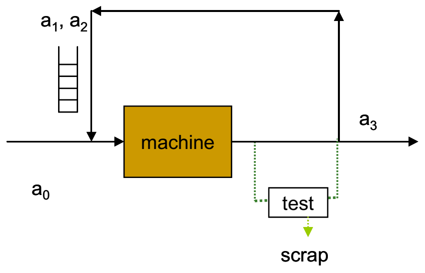

Fig. 1. Schematic representation of a single work-station facilitating re-entrant flow

The cost of machine operation is substantial and hence, a_2 is a more expensive intermediate than a_1, which is more expensive than a_0. We assume that there exists unlimited raw material supply, all final products are tested, and negligible time is spent on test and maintenance, for which some costs are incurred.

2.2 Process drift

The machine is prone to degradation with time and hence produces bad layers once in a while. The defect generation is random and machine in good working condition may also produce defects but with much lower frequency as compared to when it is mal-functioning. In essence, the good condition differs from bad on the basis of rate of defect generation.

In their statistical studies, Nurani et al. (1994) show that the number of defects per unit time fluctuates about a constant mean at the beginning and then keeps increasing with time. The different states of the machine health are not directly observable and the only information that rests with the operator is whether a defect has occurred or not. To model the time dependency of the defect rate, the machine performance is approximated with a Markov switching model containing 3 regimes as in Figure 2. The transition probabilities of the Markov Chain are as shown in Figure 3. The 3 regimes differ in terms of their defect rate. Regime 1 is the best possible machine state with lowest defect rate, while regime 3 is the absorbing state, i.e., the system remains at regime 3 until a maintenance job is performed to bring it back to the best state.

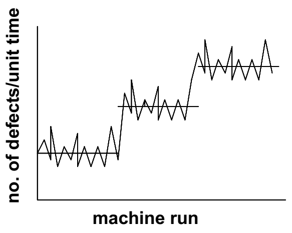

Fig. 2. 3-regime machine model with Markov switching

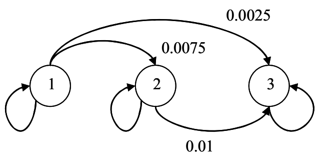

Fig. 3. Underlying Markov Chain at the work-station

In terms of 3 layer product of our interest, we assume that a defect in any layer renders the entire product defective. No distinction is made between a defect in the 1st layer and a defect in the 3rd layer in this study. Since the cost of maintenance is high, defect generation is inevitable. Intuitively, it should be best to run expensive job (a_2) when the machine is close to regime 1 (with a lower chance of making a defect). Cheaper intermediates (a_0 and a_1) should be processed when the defect rate is high. To keep defects from propagating, these cheaper intermediates must be tested and reworked/scrapped when found defective. Keeping this intuition in mind, we present the mathematical formulation of the problem in the following section.

3. PROBLEM FORMULATION

3.1 Mathematical representation

System State
The system at any time is fully characterized by the following
$x = [n_1\ n_2\ d_1\ d_2\ regime]$

n_1 - number of a_1 in the queue
n_2 - number of a_2 in the queue
d_1 - number of defective a_1's in the queue
d_2 - number of defective a_2's in the queue

regime - regime of machine operation

The state space would consist of all possible combinations of the above parameters. For instance, if the queue length is limited to 5 for n_1 and 5 for n_2, then n_1 can have 6 possible values (0, 1, 2..,5). Similarly, n_2 can have 6 possible values. For a particular value of n_1 say 3, d_1 can hold 4 possible values from 0,1,2,3. For the given queue lengths then, the number of possible combinations for [n_1 n_2 d_1 d_2] is 441. With 3 regimes for machines, the size of state space is 1323 (441 x 3). If there is a common queue with length 10, the size of the state space becomes 3003 states (1001 x 3).

Action/ decisions
u = [*schedule test maintain*]
schedule - job scheduling decision, whether to admit a_0 ,a_1 or a_2
test – binary, test (1) or not (0)
maintain – binary, maintain the machine(1) or not (0)
Size of action space = 3 x 2 x 2 = 12

Uncertainty
a). Machine regime switching - As shown in Figure 3, the machine can switch between regimes with certain probabilities in a non-deterministic manner.

b). Defect generation - Defect generation is probabilistic and the defect probability is set by the regime in which the machine is operating.

c). Error propagation- Since all the intermediates are not tested, the queue might contain defective intermediates, designated as d_1 and d_2 in the state description. Probability that a defective intermediate is picked and operated upon is given by q:

$$q = \frac{d_1}{n_1} \qquad \text{For } a_1 \text{ being operated}$$

$$q = \frac{d_2}{n_2} \qquad \text{For } a_2 \text{ being operated}$$

Objective
A profit measure is maximized. In this study we look to maximize infinite horizon discounted profit/ reward:

$$profit = \max_{I_{mt}, I_{Tt}, I_{a0t}, I_{a2t}} \sum_{t=1}^{\infty} \gamma^t (C_p I_{pt} - C_m I_{mt} - C_T I_{Tt} - C_{a_0} I_{a_0 t} - C_R) \tag{1}$$

where,

γ = discounting factor	(0.99)
C_p = product price	(1000)
C_m = cost of maintenance	($20{*}C_p$)
C_T = cost of test	($0.05{*}C_p$)
C_{a0} = cost of raw material	($0.01{*}C_p$)
C_R = cost of a run all	($0.1{*}C_p$)

The values indicated in the parenthesis are parameter values for all simulation purposes throughout this work. These values are reasonably chosen to represent the trade-offs between different cost heads in a typical manufacturing environment.
All I's (I_{pt}, I_{mt}, I_{Tt}, I_{a0t}) are binary and are equal to 1 when a non-defective product is produced, when a maintenance job is run, when a job is tested and when a_0 is run at time t, respectively. Since scheduling, testing, and maintenance are part of the overall decision making, they should be chosen so that the overall profit is maximized.

The above mentioned problem falls under the vast domain of Markov Decision Processes and can be solved as a discounted infinite horizon problem using value iteration proposed by Bellman (Bertsekas, 1995), for reasonable size problems. Problem size is governed by queue length. The *Bellman equation* is as shown below,

$$V(s) = \max_{a \in A}\{r(s,a) + \gamma \sum_{s' \in S} p(s' \mid s,a) * V(s')\} \tag{2}$$

For finite queue sizes, the analysis would give an optimal stationary policy for each state. The biggest challenge arises from the fact that a part of the state is not exactly known to the decision-maker. This is discussed in the following section.

3.2 Partial Observability

As mentioned earlier the decision-maker does not see the regime that the machine is in, at any time other than when the machine is just serviced. Also, a part of the state is not known to the decision-maker since d_1 and d_2 are the undetected defects accumulated in the system. To sum up, the elements that are the part of observation are n_1, n_2 and defect status of each tested product (1 if a defect occurred, 0 otherwise). This partial observability led us to look into the literature for Partially Observable Markov Decision Processes, which are briefly described here.

POMDP description
A partially observable Markov decision process (*POMDP*) describes a stochastic control process with partially observable (hidden) states. Formally, it corresponds to a tuple (*S, A, Θ, T, O, R, Π*):

- *S* – set of Markov states / state space
- *A* – set of actions / action space
- *Θ* – set of observations / observation space
- *T – p(s'|s,a)* transition probability
  Probability of being in *s'* at *t+1*, when action a is taken from state *s* at time *t*
- *O- p(o|s',a)* observation probability
  Probability of getting observation o at time *t+1*, when action a is taken at time *t* and state s' is reached at time *t+1*
- *R – r(s,a,s')*
  Reward when action a is taken in state s at time *t* to reach state s' at time *t+1*
- *Π – p(s)* Initial probability distribution at *t=0*

Concept of Information state MDP

The *POMDP* problem shown above can be represented as an equivalent information state *MDP* where the information state contains
- a prior belief b_0 on states *S* at time 0
- a complete history of actions and observations starting from time 0

Using Baye's rule, it turns out that a belief state *b(s)*, i.e., the conditional probability of being in state *s* at time *t*, is a sufficient information state for our problem. The conditions for a sufficient informationstate can be found in (Hauskrecht, 2000). Therefore, the Bellman equation (2) of section 3.1 becomes

$$V(b) = \max_{a\in A}\{\sum_{s\in S} r(s,a)b(s) + \gamma \sum_{o\in\theta}\sum_{s\in S} p(o \mid s,a)b(s) * V(b')\} \quad (3)$$

Belief-states can be updated via the following recursive Bayesian approach:

$$b'(s') = \frac{p(o \mid s',a)\sum_{s\in S} p(s' \mid s,a)b(s)}{p(o \mid b,a)} \quad (4)$$

The partial observability, thus converts the original problem into a continuous state Fully Observable *MDP* (*FOMDP*), where the state dimension is one less than the size of the state space. The belief-states are continuous since they contain the probability values, which are continuous numbers between 0 and 1. This characteristic of the problem poses computational challenge as discussed further.

Computational complexity

- For the problem described in section 3.1, the continuous state MDP formed would have a state dimension of 1322 with 12 actions. A problem of this magnitude is intractable, even with approximate solution methods for POMDPs. Furthermore, it would increase exponentially with increase in queue length and so will the complexity.
- To work around the partial observability, state estimation methods are in place. But in this case, state estimation techniques are equally unwieldy, since the number of hidden states corresponding to a unique observation state is large. Simulation based crude estimates perform hardly better than the periodic decisions based on experience.

4. DECOMPOSING THE PROBLEM

Given the large computational complexity, we propose to decompose the problem by exploiting the basic problem structure. The machine maintenance decision is affected by testing, since the results of testing give information about machine regime, which is not directly observable. Also, testing and maintenance decisions directly affect job scheduling, while the reverse is not true. Job scheduling does not have a direct impact on machine maintenance in this particular setup. But test decisions are weakly dependent on the job being processed, owing to the need for picking out defective intermediates. Therefore, we aim at optimizing the maintenance and test decisions independent of job scheduling decision. The optimal policy thus obtained may then be used for good job release schedules. Therefore, we discuss the basic machine maintenance problem without considering the test decision, as a start.

4.1 Machine maintenance

As described in section 2.2, the proposed workstation can operate in any of the 3 regimes, transitions among which are governed by the underlying Markov Chain. First, let us assume that we test the product every time. Then this 'sub-system' is also a POMDP with the following parameters:

State *s* – machine state *S*= {1,2,3}
Action *a* – binary; maintain (1), do not maintain (0) *A*= {0, 1}
Observation – binary; defect (1) , no defect (0)
Θ ={0,1}

$$p(s' \mid s,0) = \begin{bmatrix} 0.99 & 0.0075 & 0.0025 \\ 0 & 0.99 & 0.01 \\ 0 & 0 & 1 \end{bmatrix} \qquad p(s' \mid s,1) = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$

$$p(o \mid s',a) = \begin{bmatrix} 0.05 & 0.1 & 0.5 \end{bmatrix}$$

Exact solution methods for this 2-dimensional continuous state *MDP* can be found in existing literature (Sondik, 1978). However, the exact methods soon become computationally challenging with increase in the problem size. Since we wanted to extend the analysis to bigger problems, in this work, we focus our attention on approximate methods for solution of the *POMDPs*.

Approximate methods
There is a slew of value function approximation methods reported in the *POMDP* literature. One of the latest reviews in this area is by Hauskrecht, (2000). We chose to use two of the available approximate methods and propose a refinement of these heuristic methods for machine maintenance problems as a special case of *POMDPs*, called adaptive grid method. The approximate methods employed in this study are as follows:

- Fully observable MDP *(FO-MDP)* value function approximation
- Grid based approximation with K-nearest neighbors interpolation
- Adaptive grid method – an approximate dynamic programming approach

FO-MDP value function approximation
This method is based on approximating the belief state value function using the *FOMD's* optimal value function.

$$\hat{V}(b) = \sum_{s\in S} b(s)V^{*}_{MDP}(s) \quad (5)$$

where $\hat{V}(b)$ is the value function approximation for the *POMDP* and $V^{*}_{MDP}(s)$ is the optimal value function for the 3 state *FOMDP*, obtained using value iteration (equation (2)). The decision policy can be extracted in real time by the following equation

$$a_t = \arg\max_{a}\{\sum_{s\in S} r(s,a)b_t(s) + \gamma \sum_{o\in\theta} (b' \mid o,b_t,a)\hat{V}(b')\} \quad (6)$$

where *b'* is the future belief state based on present belief state b_t and all possible observations.

Grid-based approximation with K-nearest neighbors interpolation

In this method, the entire state space is divided into equidistant grid points in order to discretize it. These grid points form the state space of the *MDP* problem, which can be solved as an FOMDP using standard methods. When states that do not belong to the discrete state space are visited during the Bellman iteration, the value function is approximated using K-nearest neighbors interpolation scheme (Hauskrecht, 2000), based on the Euclidean distance norm. For this problem, 4 nearest neighbors were used for interpolation.

Adaptive grid based approximation

The value function for *POMDPs* is proven to be piecewise linear and convex, (Sondik, 1978). The new adaptive grid method proposed here makes use of this property coupled with the problem structure that one state is better than the other.

In the continuous state domain, the point where the decision making changes is crucial and several mathematical programming approaches have been used to locate the break point for optimal solution (Smallwood and Sondik, 1973). The adaptive grid method is aimed at improving an approximate solution by selectively meshing the 'sensitive region' around the break point. Fig. 4 shows a 2 states (1,2), 2 actions (α_1, α_2) problem with 1 dimensional continuous state. The optimal value function (max over α_1 and α_2) is plotted against the belief state.

Let us say that the optimal break point lies at a belief state value of B^o as shown. For states between p_1 and B^o, optimal action is α_1, while α_2 is optimal for the rest of the state space. Also, state 1 is the desirable state and state 2 is the absorbing state with high defect rate. After action α_2, the system returns to state 1. Since an approximate method/heuristic would produce a sub-optimal policy, let us say that the sub-optimal solution gives a break point at B^{so}. Depending upon the method used, we can generate a *precision band* as shown by the shaded region. This region must include the optimal break point B^o (yet unknown). This shaded region, marked by l_1 and l_2 now forms our new state space. Since the system returns to the state 1 upon action α_2, p_1 is a part of the state space. For all the rest of the points, actions are fixed. For states lying between p_1 and l_1, action α_1 is the optimal decision. For states between l_2 and p_2 , action α_2 is the optimal decision. Now, this reduced state space can be discretized to run the value iteration. During the value iteration, certain actions can take us to states that lie outside the reduced state space. Due to the problem structure, these states would lie in the vicinity of the reduced state space, called 'neighborhood' (depicted in the fig. 4 by dashed lines). This is when the piecewise linear structure of the value function proves helpful. The value function corresponding to states in this neighbourhood can be linearly interpolated between 2 extreme points

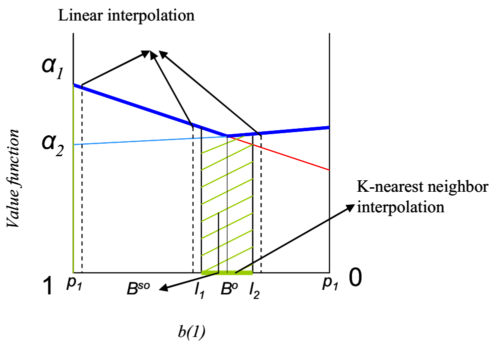

Fig. 4.Adaptive grid illustration for 2 states-2 actions

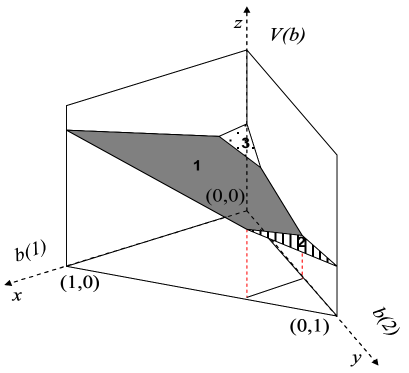

Fig. 5. Value function representation for 3-state problem

Figure 5 shows the 3-state case, described earlier in this section. The probability of being in state 1 and 2 is plotted on *x* and *y* axes respectively. Value function is on *z*-axis. The value function is a combination of 3 planes (1,2,3 in figure 5), and the break lines signify the change in the decision. For this 3 state problem, the *FOMDP* optimal solution sets an upper bound on the performance. The decision rule in *FOMDP* recommends running a maintenance job every time we are in regime 2 or 3 with the optimal value function of

$$V^*_{MDP} = [73606 \quad 57630 \quad 57630]$$

We establish the precision band for adaptive grid method based on FOMDP value function approximation, since it requires the least computational effort. The plot in Fig. 6 sums up the performance of the approximate methods. The Adaptive Grid method performs best while the number of states is very small. This suggests better performance with significantly less computational effort. This analysis is easily extendable to more machine regimes and more actions. The profit values, in Fig. 6 are shown for an average over 100 experiments for each sub-optimal policy pertaining to a particular method. Values in parenthesis show the size of state space which is a measure of the calculations were carried out in MATLAB 7.0.4 environment on a Pentium(R) 3.00 GHZ machine.

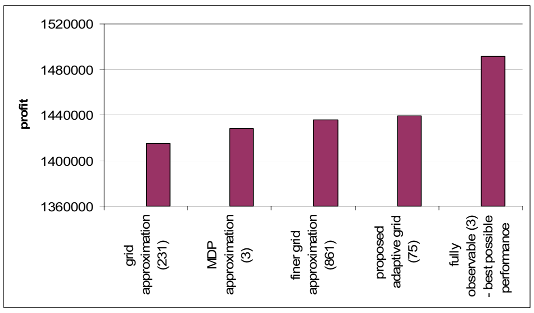

Fig. 6. Results obtained by using the POMDP solution methods for machine maintenance.

4.2 Incorporating the test decision

The test decision can be incorporated in the maintenance problem of the previous section, in a straightforward manner. The belief state update rule and the action space changes as shown below

Belief state update

$$b'(s') = \frac{p(o \mid s',a)\sum_{s\in S} p(s' \mid s,a)b(s)}{p(o \mid b,a)} \qquad \textit{if tested}$$

$$b'(s') = \sum_{s\in S} p(s' \mid s,a)b(s) \qquad \textit{if not tested}$$

Action space = {(0,0), (0,1), (1,0), (1,1)}
where {0,0} mean no test , no maintenance
{1,1} mean test and maintain and so on

Figure 7 shows the difference in performance, when we decide to test all the time, and when we make testing a decision. It is seen that testing all the time is not the best policy.

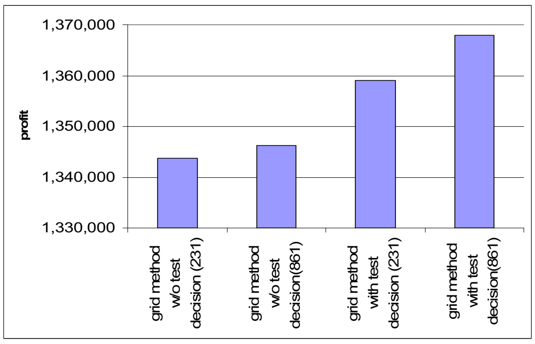

Fig. 7. Comparison of objective values obtained before and after incorporating the test decision.

4.3 Job Scheduling

After obtaining a sub-optimal maintenance and test schedule independent of job scheduling, a heuristic rule can be employed for the job scheduling. As mentioned earlier in section 1.1, the expensive intermediates must be processed, when the machine is close to a good regime

5. CONCLUSIONS

A complex decision making problem in an uncertain manufacturing environment has been addressed by formalizing it as a *POMDP*. A systematic decomposition scheme is proposed by taking advantage of the one way interaction among the decisions. Approximate solution methods exploiting the inherent problem structure are employed to generate sub-optimal policies to work around the computational complexity. However, most manufacturing facilities have multiple machines operating in series or parallel. It is highly unlikely then, that the maintenance and job scheduling decisions would be made on the basis of one work-station. Therefore, extending this analysis to multiple machines is an obvious candidate for future work in this area. Furthermore, we have ignored the time of operation considerations in this study, which is where scheduling, testing and maintenance decisions directly affect one another. An iterative scheme that successively improves the decision making appears to be one feasible direction toward solution.

Acknowledgment: The authors gratefully acknowledge the financial support from the National Science Foundation (CTS--\#0301993).

REFERENCES

Bertsekas, D. P. (1995). *Dynamic Programming and Optimal Control*, Athena Sceineticfic, Belmont, Massachusetts

Choi, J. Y. and S. A. Reveliotis (2003). "A generalized stochastic Petri net model for performance analysis and control of capacitated reentrant lines.*" IEEE transactions on robotics and automation* **19**(3): 474-480

Gupta, A. K. and A. I. Sivakumar (2006). "Job shop scheduling techniques in semiconductor manufacturing." *The International Journal of Advanced Manufacturing Technology* **27**(11-12).

Hauskretch, M. (2000). "Value-function approximations for partially observable Markov decision processes." *Journal of Artificial Intelligence Research* **13**: 33-94.

Nurani, R. K., R. Akella, A.J. Strojwas, R. Wallace, M.G. McIntyre, J. Shields and I. Emami (1994). " Development Of An Optimal Sampling Strategy For Wafer Inspection." *Extended Abstracts of 1994 International Symposium on Semiconductor Manufacturing.*

Shen, Y. and R. C. Leachman (2003). "Stochastic Wafer Fabrication Scheduling." *IEEE transactions on semiconductor manufacturing* **16**(1): 2-14.

Smallwood, R. D. and E. J. Sondik (1973). "The Optimal Control of Partially Observable Markov Processes over a Finite Horizon.*" Operations Research* **21**(5): 1071-1088.

Sondik, E. J. (1978). "The optimal control of Partially Observable Markov Processes over the Infinite Horizon: Discounted Costs." *Operations Research* **26**(2): 282-304