行政院國家科學委員會專題研究計畫　期中進度報告

子計畫三:結合模擬與排序佳化法的生產排程及其應用於 12 吋晶圓製造之研究(1/3)

計畫類別： 整合型計畫
計畫編號： NSC92-2213-E-002-099-
執行期間： 92 年 08 月 01 日至 93 年 07 月 31 日
執行單位： 國立臺灣大學電機工程學系暨研究所

計畫主持人： 張時中

計畫參與人員： 張銘辰、何元祥、李豐凡、朱紹儀、陳俊宏教授、謝博偉博士

報告類型： 精簡報告

處理方式： 本計畫可公開查詢

中　華　民　國 93 年 6 月 28 日

結合模擬與排序佳化法的生產排程及其應用於 12 吋晶圓製造之研究(1/3)

Research on Simulation-based Ordinal Optimization Methods with Applications to Production Scheduling of 300mm Foundry Fabs (1/3)

計 畫 編 號： 92-2213-E-002-099
主 持 人： 張時中教授
計畫參與人員： 張銘辰、何元祥、李豐凡、朱紹儀
陳俊宏教授、謝博偉博士

執行期限：92 年 8 月 1 日至 93 年 7 月 31 日
執行單位：國立臺灣大學電機工程學系暨研究所

一、中文摘要

本三年計畫的目標如下：(1) 為排序佳化模擬設計削減搜尋空間的新方法，(2) 將上述方法開發成工具模組，作為整合系統佳化平台的一部份，(3) 將模擬進行排序佳化應用在次世代半導體廠的有效生產排程。為達成這些目標，第一年中我們發展一個以模擬為基礎的排序策略疊代(OOBPI)法, 用來處理平穩(Stationary)馬可夫決策。我們利用策略疊代的架構，以模擬為基礎的排序佳化來估算每一狀態的 cost-to-go 函數值與與最佳決策。初步模擬結果顯示這個方法較傳統模擬為基礎的策略疊代法的計算效能可快百倍。另外正進以合約演算法的觀念架構來設計對狀態的最佳計算資源分配，以進一步提昇 OOBPI 法的計算效能。

Abstract

In this proposed three-year research project, objectives are (1) to design search space reduction method for simulation-based OO, (2) to develop these methods into tool modules as part of an integrated system optimization platform, and (3) to apply simulation-based OO to effective production scheduling of 300mm foundry fabs. In the first year, we have designed an *OO-Based Policy Iteration (OOBPI)* method to handle the combinatorial complexity of decisions over the time axis for Stationary Markov decision problems. Utilizing the framework of policy iteration, we approximate the optimal cost-to-go and optimal decision of each state by simulation-based OO. The *OOBPI* method demonstrates, in preliminary numerical studies, two orders of speed-up in than policy iteration using traditional simulation for evaluating cost-to-go values. To efficiently handle the large state space under computing processor capacity and run time limits, we have been investigating the notion of *contract* algorithms in general and ordinal computing budget allocation in specific to further speed up the convergence of OOBPI.

Keywords: ordinal optimization, policy iteration,, simulation-based policy iteration, Markov decision process, contract algorithm, computing budget allocation

二、緣由與目的

Flexibility and speed in operating a complicated semiconductor fab are essential to competitiveness in this e-business era. Externally, markets change rapidly with diversified consumer demands and short life cycle of electronic products. Internally, infrastructure for operations quickly evolves with advancement in process and tool technologies and the emergence of new information technology. As a result, fab

designs for highly modular and easily re-configurable operations have been a trend for foundry manufacturing, which lead to many options for responding to market and fab uncertainties. However, for a given change, how a good option can be quickly selected from a large amount of options remains a critical challenge. With the sky rocketing capital investment of 300mm fabs, adopting a good option can result in significant savings of cost.

In this project, we will address such a critical need particularly for problems of production scheduling, which is a significant class of resource allocation problems in fab operations. Major fab scheduling problems include how wafers should be released into a fab and how they should be dispatched among machines for processing. A popular practitioners' approach for scheduling the production in a fab is to select from the many empirical scheduling rules available for IC fabs. As rule options differ by tool groups, there is a combinatorial number of rule options for the complete fab operations. Further, operation objectives of a fab dynamically change among throughput, cycle time, on time delivery, etc. Dynamic selection of scheduling rule based on fab objective and state changes is therefore needed for competitive operations, which requires that a good option be selected in a timely manner.

Another major concern is how to analyze the performance of a semiconductor fab for considered options, given its tremendous complexity. Computer simulation technology has matured over the past decade and is now commonly used in industry. Simulation allows one to more accurately specify fab operations through the use of logically complex, and often non-algebraic, variables and constraints. This capability compliments the inherent limitation of traditional optimization. However, the added flexibility often creates models that are computationally intractable. Furthermore, many alternative options (the number could be combinatorial in many cases) must be simulated in order to find a good design. The total computation cost for typical simulation-based approaches is usually too expensive.

Central to our solution methodology is a new control-theoretic approach to simulation experiments. In particular, *ordinal optimization* (OO)[HSV92], i.e., determining *relative*, rather than *absolute,* merits of candidate designs, show that a much faster convergence can be achieved when compared to traditional approach. Our proposed research capitalizes on the principles of *ordinal optimization* while further improving upon its efficiency through an optimal computing budget allocation (OCBA) technique [CDC98]. The great potential of developing an efficient simulation methodology for semiconductor fab based on OO and OCBA techniques has been supported by the exciting results of the PI's preliminary investigation. The PI and his colleague designed a OO- and OCBA-based simulation system as shown in Figure 1 and studied the problem of dispatching rule selection over a much simplified fab model [HCC01]. Results demonstrate that two orders of computation time reduction over traditional simulations can be achieved and that such a speed gain can be exploited for dynamic rule selection to effectively handle uncertainties.

In spite of its superior computational efficiency over the traditional simulations and the insights generated in this application study, the OO-based simulation needs to cross a few hurdles before full-scale application to dynamic selection of scheduling rules for fab operations. First, there are many more release policies and dispatching rules desirable for evaluation although the frequently used rules are just a few. Second, the number of rule combinations grows in a combinatorial way over time. For example, m wafer release policies and n dispatching rules over a p-unit-time horizon constitute $(m \times n)^p$ rule

options when the rule may change over individual time units. Third, distinct rule libraries are adopted for tool groups of different characteristics. This leads to a combinatorial growth of rule options over the number of tool groups. Brute-force application of OO-based simulation is therefore infeasible. Further research on option search method exploiting the OO-based simulation is thus needed.

To tackle the combinatorial rule options over the number of tool groups, Hsieh et al [HCC03] have designed a fast simulation methodology by an innovative combination of the notions of *ordinal optimization* (OO) and *design of experiments* (DOE) to efficiently select a good scheduling policy for fab operation. The DOE method is exploited to largely reduce the number of scheduling policies to be evaluated by OO-based simulation. Preliminary simulation results of applications to scheduling wafer fabrications show that most of the OO-based DOE simulations require 2 to 3 orders of magnitude less computation time than those of a traditional approach, and the speedup is up to 7,000 times in certain cases.

This proposed three-year research will advance the design of new simulation methods for full-scale semiconductor fabs based on the OO principles and the insights we have obtained [HCC01, HCC03]. There are three objectives (1) to design search space reduction method for simulation-based OO, (2) to develop these methods into tool modules as part of an integrated system optimization platform, and (3) to apply simulation-based OO to effective production scheduling of 300mm foundry fabs.

三、結果與討論

In production scheduling problems, it is advantageous that the scheduling rules are changed from time to time, considering the rapid change of demand and machine availability. The decision for each time period could be highly dependent with those in other time periods. A major challenge is that the number of alternative options grows very quickly as the planning time horizon becomes longer. For example, if there are M alternative scheduling rules for consideration in each time period and if we want to look ahead for T periods. Then the total number of alternative rule options is $(M)^T$, which can be very large if M or T is not too small.

Dynamic programming is an optimization approach that transforms a complex problem into a sequence of simpler problems; its essential characteristic is the multi-stage nature of the optimization procedure. In this task, an ordinal optimization-based dynamic programming will be developed to handle the combinatorial complexity over the time axis. At the higher level, dynamic programming transforms the complex multi-stage scheduling problem into a sequence of single-stage optimization problems. At the level of single-stage optimization problem, the objective function (also called cost-to-go) must be evaluated using discrete-event simulation and several alternative options must be simulated. A key issue here is how to solve the single-stage optimization problem very efficiently, since many of such single-stage problems must be solved in the framework of dynamic programming. We propose to use our OO-based simulation tool for solving such single-stage problems by taking advantage of exponential convergence of OO and intelligence of OCBA.

In the past 10 month of research, this subproject has been focused on the design of *ordinal optimization-Based Dynamic Programming (OODP)* methods to handle the combinatorial complexity of scheduling policies over the time axis. Progresses have been made in the design of a simulation-based ordinal policy iteration algorithm and the design of contract algorithm for dynamic allocation of

computation budget.

I. Design of Ordinal Optimization-based Policy Iteration Algorithm (OOBPI)

To explore the basic ideas, we first consider a special class of dynamic optimization problem [Ber76], the Markov Decision Process (MDP) with finite state and action spaces, by defining appropriate states, controls, transition probabilities, time horizon, and stage-wise cost function. The stage-wise cost function is so complicated that it can only be evaluated by simulation.

One basic algorithm for solving a stationary MDP is policy iteration (PI). The PI algorithm consists of a *policy evaluation* step in which the cost function value of the current policy is computed, and a *policy improvement* step where, if possible, the current policy is improved upon. These two steps are repeated iteratively until some stopping requirements are met. The evaluation step of policy iteration consists of solving a set of linear equations called the *average evaluation equations* (AEE). Using the solution to the AEE, the improvement step then employs a one-step analysis to decide if the current policy can be improved.

In the case that stage-wise cost function is so complicated that it can only be evaluated by simulation, a simulation-based policy iteration (SBPI), has been proposed in [??]. Rather than exactly solving the AEE in the policy evaluation step, SBPI estimates solutions of the AEE via simulation and uses them in the policy improvement step. Note that this procedure does not require the solution of the large linear system. However, when the dimension of states is large, SBPI may spent lots of time doing simulation to evaluate the cost-to-go of individual states under a given policy. When the dimension of control space is large, the policy improvement step will also require time-consuming simulations.

To speed up the computation, we develop, in this research, a fast simulation methodology by an innovative combination of the notions of ordinal optimization (OO) and SBPI. The key idea is that For every state, instead of finding the exact expected cost-to-go among admissible controls, our approach compares relative orders of expected cost-to-go among admissible controls to a specified level of confidence [Che96]. It thus finds a good enough rule with a much reduced simulation time requirement. The schematic diagram of the method is shown in Figure 2 and the OOBPI algorithm given in the Appendix. Preliminary simulation result over a 10-state and 4-control per state MDP problem shows that OOBPI is more than 200 times faster than SBPI.

II. Design of Contract Algorithm-based Computation Budget Allocation (CABCBA)

To efficiently search over the large state space under processor capacity and run time limits, there needs a method to properly allocate computing budget. We have been investigating the notion of *contract* algorithms (Zilberstein, Charpillet, and Chassaing, 2003), where the solution quality of an algorithm improves as the allocated amount of computation time increases. Exploiting the design of SBOPIA, we are establishing a stochastic performance profile $P_A(q|t)$ denoting the probability of getting an optimal-cost-to-go estimate of quality q with budget time t by the OOBPI algorithm. Dynamic computing budget allocation will then be designed based on such profile.

III. Publications

1. B.-W. Hsieh, S.-C. Chang, C.-H. Chen, M.-C. Chang, “Efficient Composition of Good Enough Dispatching Policies for Semiconductor Manufacturing,” *submitted to ISSM 2003*, Santa Clara, Sept. 30 – Oct. 2, 2003.
2. T.K. Hwang, S.-C. Chang, “Properties of

Iterative Proportional Capacity Allocation for Re-entrant Line Operations," to appear in *Proceedings of APIEMS2004, Australia, Dec. 2004.*

四、參考文獻

[Ber76] D. P. Bertsekas, *Dynamic Programming and Stochastic Control.* New York: Academic Press, 1976.

[CDC98] C. H. Chen, L. Dai, H. C. Chen, and E. Yucesan, "Efficient Computation of Optimal Budget Allocation for Discrete-event Simulation Experiment, " *IIE Transactions on Operational Researches*, 1998.

[Che96] C.-H. Chen, "A lower bound for the correct subset selection probability and its application to discrete-event system simulations," *IEEE Transactions on Automatic Control*, vol. 41, no. 8, pp. 1227-1231, Aug. 1996.

[HCC01] B.-W. Hsieh, C.-H. Chen, S.-C. Chang, "Scheduling Semiconductor Wafer Fabrication by Using Ordinal Optimization-based Simulation," *IEEE Trans. on Robotics and Automation*, Vol. 17, Issue: 5 , Oct. 2001, pp. 599 -608.

[HCC03] B.-W. Hsieh, S.-C. Chang, C.-H. Chen, "Efficient Selection of Scheduling Rule Combination by Combining Design of Experiment and Ordinal Optimization-based Simulation," *Proceedings of ICRA2003*, Taipei, Sept., 2003.

[HSV92] Y. C. Ho, R. S. Sreenivas, and P. Vakili, "Ordinal Optimization of DEDS," *Journal of Discrete-event Dynamic Systems*, Vol.2, No.2, pp. 61-88, 1992.

[ZCC2003] S. Zilberstein, F. Charpillet, and P. Chassaing, "Optimal Sequencing of Contract Algorithms," *Annals of Mathematics and Artificial Intelligence*, Vol. 39, No. 1-2, 2003, pp. 1-18.

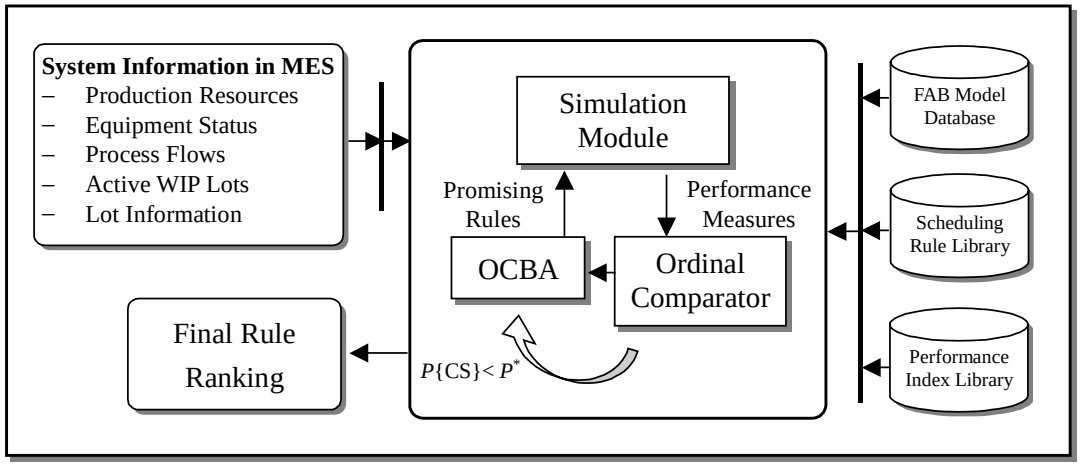

Figure 1. An Ordinal Optimization-Based Simulation Framework

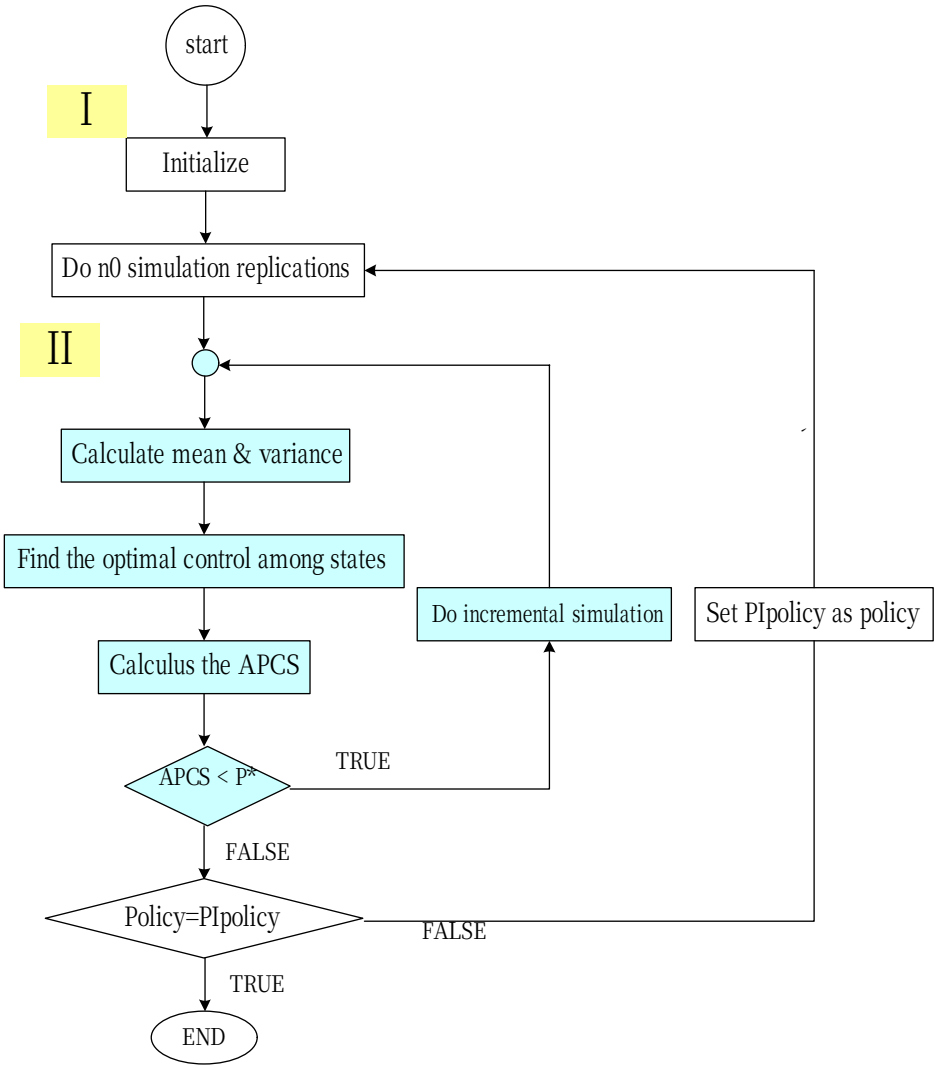

Figure 2. Flowchart of the OO-based PI algorithm

Appendix: OOBPI Algorithm

π^k	Policy of the k^{th} iteration
b	the simulated best control $\equiv \arg\left\{\underset{i \in U(i)}{Min}\, E\left(\tilde{J}_{\pi^k}(i)\right)\right\}$
i	State index
i_t	State at time t
S	Set of state space
t	Time epoch

r	Simulation replications
r_0	Number of initial simulation replications
τ	Number of incremental simulation replications
$\tilde{J}_{\pi^k}(i_w)$	the w^{th} simulation with policy π^k from state i $\equiv g(i_0, i_1)+\alpha \cdot g(i_1, i_2)+\Lambda\ +\alpha^t \cdot g(i_t, i_{t+1})+\Lambda$
$\tilde{J}_{\pi^k}(i)$	$\equiv \frac{1}{r}\sum_{w=1}^{r} \tilde{J}_{\pi^k}(i_w)$
$J_{\pi^k}(i)$	The real optimal cost-to-go while applying policy π^k in state i $\equiv \lim_{r\to\infty}\left\{\frac{1}{r}\sum_{w=1}^{r} \tilde{J}_{\pi^k}(i_w)\right\}$
$\tilde{J}(i,u_w)$	The w^{th} simulated cost-to-go with control $u \in U(i)$ from state i $\equiv g(i,u)+\sum_{j=1}^{n} p_{ij}(u)\cdot \tilde{J}_{\pi^k}(j_w)$
$\sigma^2(i,u)$	$\equiv \frac{1}{r-1}\sum_{w=1}^{r}\left\{\tilde{J}(i,u_w)-\left[\frac{1}{n}\sum_{s=1}^{r}\tilde{J}(i,u_w)\right]\right\}^2$
α	Discount factor$\in [0,1]$
P^*	Satisfactory confidence level of correct control selections
$U(i)$	Admissible controls of state i
OO	Ordinal optimization
$P\{\text{CS}\}$	Probability of correct selection $\equiv P\{$the current top-ranking rule is actually the best one$\}$
$\Pr(i,u)$	The transition probability matrix in state i with control u
$APCS(i,u)$	the probability of correction selection of control u at state i
u_m	The m^{th} admissible control
$p_{ij}(u)$	The transition probability from state i to state j with control u
$g(i,j)$	Transition cost from state i to state j $\equiv g(i,\pi^k(i))$

OOBPI Algorithm

Step 0:

Set a satisfactory confidence level of correct control selection P^*

Step 1: (Initialization)

Set $k = 0$, select an arbitrary stationary policy π^k

Step 2: (Policy evaluation)

Simulate the cost-to-go $\tilde{J}_{\pi^k}(i)$ of every state $i \in S$ by

$$\tilde{J}_{\pi^k}(i) \equiv \frac{1}{r}\sum_{l=1}^{r} \tilde{J}_{\pi^k}(i,w)$$

with r replications$^{\psi}$ (r_0 replications in initial)

where $\tilde{J}_{\pi^k}(i,w)$ denotes the w^{th} simulation with policy π^k from state i

$$\tilde{J}_{\pi^k}(i,w) \equiv g(i_0,i_1) + \alpha \cdot g(i_1,i_2) + \Lambda + \alpha^t \cdot g(i_t,i_{t+1}) + \Lambda$$

★ i_t denote the state in time t.

★ α is the discount factor;

★ $g(i,j) \equiv g(i,\pi^k(i))$ is the transition cost from state i to j

Step 3: (Policy improvement)

For every state $i \in S$, find the minimal simulated state-wise cost-to-go among all admissible controls $u \in U(i)$,

i.e. $\underset{u\in U(i)}{Min} E(\tilde{J}_{\pi^k}(i)) = \underset{u\in U(i)}{Min}\left[g(i,u) + \sum_{j=0}^{n-1} p_{ij}(u)\cdot \tilde{J}_{\pi^k}(j)\right]$

denote the simulated best control $b \equiv \arg\left\{\underset{u\in U(i)}{Min} E(\tilde{J}_{\pi^k}(i))\right\}$

calculate the APCS of control b at state i. i.e. calculate $APCS(i,b)^{\Delta}$

if $APCS(i,b) > P^*$, then stop, $\pi^{k+1}(i) = b$

else add τ simulation replications;

goto Step 2;

where $APCS(i,b)$ is the probability of correction selection of control b at state i.

Step 4:

If $\pi^{k+1} = \pi^k$, then stop and set $\pi^* = \pi^k$; otherwise increment k by 1 and return to Step 2.

ψ : a replication is defined as doing 1 simulation run$^{\Delta}$ per state with policy π^k

A : A simulation run is defined as

(a) arrive the terminal state

(b) the simulation time horizon t is greater than $\dfrac{-2}{\log\alpha}$.

$\Delta : APCS(i,b)$

$$= \prod_{u=1,u\neq b}^{m} P\{\tilde{J}_{\pi^k}(i,b) < \tilde{J}_{\pi^k}(i,u)\} = \prod_{u=1,u\neq b}^{m} \Phi\left(\frac{\frac{1}{r}\sum_{w=1}^{r}\tilde{J}(i,u_w) - \frac{1}{r}\sum_{w=1}^{r}\tilde{J}(i,b_w)}{\sqrt{\frac{\sigma^2(i,u)}{r} + \frac{\sigma^2(i,b)}{r}}} \right)$$

where Φ is the standard normal cumulative distribution.