Journal of the Korean Institute of Industrial Engineers
Vol. 32, No. 1, pp. 9-17, March 2006.

# 단일 실행의 빠른 근사해 기법과 반복 실행의 최적화 기법을 이용한 이산형 시스템의 시뮬레이션 연구

박경종[1†] · 이영해[2]

[1]광주대학교 경영학과 / [2]한양대학교 산업공학과

# Simulation Study of Discrete Event Systems using Fast Approximation Method of Single Run and Optimization Method of Multiple Run

Kyoung Jong Park[1] · Young Hae Lee[2]

[1]Department of Business Administration, Gwangju University, Gwangju, 503-703
[2]Department of Industrial Engineering, Hanyang University, Ansan, 426-791

This paper deals with a discrete simulation optimization method for designing a complex probabilistic discrete event simulation. The developed algorithm uses the configuration algorithm that can change decision variables and the stopping algorithm that can end simulation in order to satisfy the given objective value during single run. It tries to estimate an auto-regressive model for evaluating correctly the objective function obtained by a small amount of output data. We apply the proposed algorithm to M/M/s model, (s, S) inventory model, and known-function problem.

The proposed algorithm can't always guarantee the optimal solution but the method gives an approximate feasible solution in a relatively short time period. We, therefore, show the proposed algorithm can be used as an initial feasible solution of existing optimization methods that need multiple simulation run to search an optimal solution.

***Keywords*:** single run, discrete simulation optimization, auto-regressive model

## 1. 서 론

복잡한 확률적 이산사건 시스템의 설계도구로서 이산사건 시뮬레이션을 사용하기 위해서는 극복해야 할 몇 가지 문제가 있다. 첫째, 시스템의 성능을 평가할 목적함수와 일부 제약식의 값은 알려진 함수를 이용한 단순한 계산이 아니라 오직 시뮬레이션을 수행하여서만 얻을 수 있다. 둘째, 시뮬레이션을 수행하여 얻은 결과는 확률적 이산사건 시스템의 경우 반드시 확률적 요소가 포함되어 있기 때문에 효과적인 통계적 분석이 요구되고, 최적화 문제도 결정적(deterministic) 최적화가 아닌 확률적 최적화 문제가 된다. 셋째, 수행하는 시뮬레이션의 형태가 안정상태(steady state) 시뮬레이션인 경우, 하나의 대안에 대하여 안정상태에서 시스템의 평가척도를 구할 때는 일반적으로 출력 데이터들이 서로 자기상관(auto-correlation)을 포함하고 있기 때문에 많은 출력 데이터들이 필요하며 긴 실행시간이 요구된다. 넷째, 시뮬레이션을 이용하여 하나의 대안을 평가하는 데도 많은 시간이 걸리는데, 해 발견을 위한 탐색공간이 클 경우에는 더욱 많은 시간이 걸리게 된다(Azadivar 1992).

그러므로 본 연구에서는 위에서 설명한 문제점을 해결하기 위해 이산사건 시뮬레이션 기법을 이용하여 이산형 변수를 가진 복잡한 확률적 이산사건 시스템을 설계할 경우에 요구되는

† 연락저자 : 박경종 교수, 503-703 광주광역시 남구 진월동 광주대학교 경영학과, Fax : 062-670-2631, E-mail : kjpark@gwangju.ac.kr
2005년 1월 접수; 2005년 7월 수정본 접수; 2006년 1월 게재 확정

시스템 평가척도와 제약식을 충족하며 신뢰할 수 있는 대안을 효율적으로 빠른 시간 내에 온-라인으로 발견해 내는 알고리듬을 여러 유형의 문제에 적용하여 일반화시킨다. 또한, 본 연구에서 사용되는 기법이 항상 최적해를 보장하지 못하기 때문에 최적해를 구하는 경우에 반복적인 시뮬레이션이 필요한 이산형 최적화 기법과의 연동을 통해 시간을 줄이면서도 최적해를 얻는 방안을 제시한다.

## 2. 기존 연구 고찰

시뮬레이션 최적화에 대한 연구는 일반적으로 연속형 결정변수에 대하여 주로 발전되었고, 본 연구에서 다루고자 하는 이산형 결정변수에 대한 시뮬레이션 최적화 연구는 상대적으로 연구가 많이 진행되지 못했다. 그러나 시스템의 버퍼 최적화나 자원 최적화 등에서 많은 관심과 필요성이 증대되어 최근에 와서 관심이 높아지고 있다. 시뮬레이션 최적화 문제에서 이산형 결정변수를 가진 이산형 확률적 최적화(discrete stochastic optimization) 문제를 풀기 위해서 simulated annealing(Lee and Iwata, 1991; Ahmed *et al*., 1997), stochastic ruler method(Yan and Mukai, 1992), stochastic comparison method(Gong *et al*., 1992), random walk(Andradottir, 1992, 1995, 1996), nested partitions method(Shi and Sigurdur, 1997), evolutionary(genetic) algorithm (Pierreval and Tautou, 1997), multi-armed bandit method(Barry and Fristedt, 1985), learning automata(Yakowitz and Lugosi, 1990) 등의 방법들이 사용되었으며, 이 방법들은 시뮬레이션 과정을 반복 수행하면서 최적화를 수행한다.

기존 연구들 중에서 특히 Chen(1994)은 $g(x)=\gamma$라고 표현되는 stochastic root finding 문제에 대해 Monte Carlo sampling을 이용한 알고리듬과 retrospective approximation(Fu and Healy, 1997) 방법을 이용한 알고리듬을 제시하였다. Chen이 제시한 방법은 매우 많은 시뮬레이션의 반복 실행을 요구하므로 간단한 수치 시뮬레이션을 목적함수의 평가도구로 이용하고, 목적함수가 $g(x) =\gamma$라고 표현되는 경우에는 이용이 가능하나, 복잡하고 한 번의 시뮬레이션 수행 동안 긴 실행시간을 요구하는 안정상태 이산사건 시뮬레이션에는 적합하지 않다.

Wild and Pignatiello(1994)는 안정상태 이산사건 시뮬레이션을 이용하여 시스템을 설계할 때 주어진 목표치($g(x) \leq \gamma$)를 만족하는 대안을 발견하기 위해 시뮬레이션 도중에 결정변수 값을 변경해 가는 reverse-simulation이라고 부르는 방법을 제시하였다. 그러나 이 방법은 결정변수 값을 변경할 때, 분석하는 시스템의 목적함수 값이 진행되는 방향을 알 수 있는 M/M/s로 모델링 될 수 있는 아주 간단한 경우에만 적용이 가능하다. 이 방법은 각 개체가 시스템을 떠날 때마다 결정변수 값을 변경하므로 안정되고 신뢰할 수 있는 대안을 얻기가 어렵고, 다양한 형태의 목표치가 주어진 경우에 사용하기가 불가능하다. 또한, 지금까지의 연구들은 시뮬레이션 기법을 이용한 이산사건 시스템의 최적 설계 시 반드시 고려해야 할 긴 시뮬레이션 수행 시간을 간과한 경우가 많았고, 시뮬레이션을 기초로 한 확률적 최적화 알고리듬에 국한하는 경우가 많았다. 기존 연구에 대한 보다 자세한 설명은 Park and Lee(1999)의 연구를 참조한다.

그러므로 본 연구에서는 이산사건 시뮬레이션 기법을 이용하여 이산형 변수를 가진 복잡한 확률적 이산사건 시스템을 설계할 경우에 단일 실행으로 빠른 시간 내에 요구되는 시스템 평가척도와 주어진 시스템의 목표치를 충족하는 대안을 발견해내는 문제를 대상으로 한다.

본 연구의 1장에서는 시뮬레이션 기법의 일반적인 문제점들을 다루고 2장에서는 이산형 시뮬레이션 최적화 기법의 기존 연구에 대하여 설명한다.

3장에서는 적용되는 알고리듬의 해 탐색과정을 설명하는데, 알고리듬에 대한 기본적인 설명과 알고리듬의 핵심이 되는 결정변수 값의 조정을 위한 세부 알고리듬 및 알고리듬의 종료조건들을 기술한다. 또한, 안정상태 시뮬레이션을 수행할 때 짧은 시뮬레이션 수행으로 얻은 적은 수의 출력 데이터로 주어진 목적함수의 평가를 정확하게 할 수 있는 방법에 대해서도 설명한다. 이를 위해 적은 수의 출력 데이터로 모델링 할 수 있는 자동회귀 모델을 사용한다. 3장에서 설명되는 알고리듬에 대한 자세한 내용은 Park and Lee(1999)의 연구를 참조하도록 하며, 본 연구에서는 4장 및 5장에서 제시되는 내용에 초점을 맞춘다.

4장에서는 지금까지 제시된 방법의 유용성과 효율성을 평가하기 위하여 서두에서 제시한 것처럼 M/M/s 모델, (s, S) 재고 모델 및 알려진 함수(known function) 문제를 이용하여 실험 및 분석을 행한다.

5장에서는 본 연구에서 제시되는 기법이 빠른 시간에 적은 양의 데이터를 가지고 단일 시뮬레이션 과정을 통해서 해를 제공하지만 최적해를 보장하지 못하기 때문에 반복 시뮬레이션 과정을 통해서 최적해를 보장하는 기법의 초기해로 이용될 수 있음을 보인다. 마지막으로, 6장에서는 본 연구의 결과를 설명하고 추후 연구 방향을 제시한다.

## 3. 단일 실행 시뮬레이션 기법

일반적으로 제조 시스템 분야에서 제기되는 문제의 해를 단일 시뮬레이션 실행(single simulation run) 환경에서 찾기 위한 방법은 알고리듬 3.1과 같이 정리된다.

본 연구에서는 결정변수($x_j$)가 이산형인 경우를 가정하므로, 각 변수에 대한 $\Delta x_j$는 각 변수의 최소 증감 값(resolution)을 나타내는데, 정수값을 갖는 경우는 일반적으로 1을 가지나 탐색공간이 큰 경우에는 사용자가 크기를 적절하게 조정할 수 있다.

목적함수를 평가하고 결정변수 값을 조정할 시간간격($\Delta t$)의 단위는 바로 전 단계에서 결정변수 값의 조정 행위가 일어

[알고리듬 3.1]

| | |
|---|---|
| 단계 1 | 목적함수($f_i(X)$), 결정변수($x_j, j=1,...,n$), 목표치($A_i$), 결정변수 $x_j, j=1,...,n$에 대한 증감($\Delta x_j$), 목적함수를 평가하고 결정변수 값을 조정할 시간 간격($\Delta t$)을 설정하고, 시뮬레이션을 시작한다. |
| 단계 2 | 시뮬레이션 도중 $\Delta t$마다 목적함수와 목표치를 비교하여 결과에 따라 바람직한 방향으로 $\Delta x_j$만큼 결정변수 값을 조정하고, 시뮬레이션을 계속한다(Park and Lee(1999)의 알고리듬 3.2 참조). |
| 단계 3 | 알고리듬 종료조건을 검사하고 만족하면, 최근 시점까지 가장 많이 방문했던 결정변수 값들의 조합을 최종해 $x_j^*, j=1,...,n$으로 정하고 시뮬레이션을 종료한다(알고리듬 3.3 참조). |
| 단계 4 | 얻어진 최종해를 가지고 충분한 길이의 검증 시뮬레이션을 수행하고 필요한 자료를 출력한다. |

난 뒤로부터의 처리 횟수 또는 시뮬레이션에서 처리된 사건(event) 수 등으로 설정이 가능하다. 예를 들어, $\Delta t$가 10인 경우는 10번의 트랜잭션(transaction)에 대한 정보를 수집하여 다음 단계로 진행할 방향을 결정한다는 의미이다. 수행하는 시뮬레이션의 형태가 종료형(terminating) 시뮬레이션인 경우는 $\Delta t$가 수행하는 시뮬레이션의 총 수행길이가 될 수 있다.

그러나 안정상태 시뮬레이션인 경우는 주어진 변수 값들의 조합인 하나의 대안에 대하여 안정상태에서의 출력을 얻을 때까지는 장시간의 시뮬레이션 수행시간이 필요하므로 요구되는 최종해를 얻을 때까지 필요한 여러 대안들의 평가에 걸리는 총 시간은 컴퓨터로 주어진 시간 내에 처리할 수 없는 경우가 발생할 수 있다. 따라서 이를 줄이기 위한 노력이 절대적으로 필요하다. 일반적으로 $\Delta t$가 작을수록 자주 목적함수를 평가하고 결정변수 값을 조정하므로 빨리 목표치를 만족하는 결정변수 값으로 수렴할 확률이 높을 수도 있으나, $\Delta t$가 작은 만큼 얻은 데이터의 수가 적어 목적함수 값을 평가하는 데 오차가 커질 수 있다. 그리고 목적함수를 평가하고 결정변수 값을 조정하는 데 많은 계산시간을 소비하게 된다. 이를 위해 짧은 시뮬레이션의 수행으로 안정상태에서의 목적함수 값을 효율적으로 추정할 수 있는 방법을 사용하며 보다 자세한 설명은 Park and Lee(1999)의 논문을 참조한다.

앞의 알고리듬 3.1의 단계 2에서, 시뮬레이션 도중 $\Delta t$마다 목적함수와 목표치를 비교하여 결과에 따라 바람직한 방향으로 $\Delta x_j$만큼 결정변수 $x_j$의 값을 조정한다고 하였는데, 이에 대한 설명은 다음의 3.1절에서 자세히 기술한다.

### 3.1 탐색 알고리듬

알고리듬 3.1의 단계 2에서, 시뮬레이션 도중 $\Delta t$마다 목적함수와 목표치를 비교하여 결과에 따라 바람직한 방향으로 $\Delta x_j$만큼 결정변수 $x_j$의 값을 조정한다고 하였는데, $x_j$의 값의 증감에 따라 목적함수 값의 변화가 쉽게 파악되는 경우와 그렇지 않는 경우로 나눌 수 있다.

전자의 경우의 예로 단일 작업장 문제에서는 기계 대수(결정변수)의 증가(감소)는 작업물의 평균 대기시간(목적함수)을 감소(증가)시키며, 잡샵(Job Shop) 문제에서는 작업장 i에 있는 기계 대수의 증가(감소)는 버퍼에서 대기하는 부품의 수를 감소(증가)시키며, 기계의 평균 이용률의 감소(증가)를 초래한다. 또한, 작업장의 수리공 문제에서는 가능한 기계 대수의 증가(감소)는 기계의 평균 고장상태 시간의 증가(감소)를 초래하고, 수리공 수의 증가(감소)는 기계의 평균 고장상태 시간의 감소(증가)를 초래한다.

후자의 경우와 같이 결정변수 값의 증감에 따라 목적함수 값의 변화가 바로 감지되지 않는 경우의 예로 (s, S) 재고관리 시스템에서 결정변수인 (s, S) 값의 증감이 목적함수인 단위기간당 발생하는 평균 재고관리 비용의 변화가 예측되지 않는데, 이러한 경우도 알고리듬에서 고려하는 것이 필요하다.

시뮬레이션 도중 $\Delta t$마다 목적함수와 목표치를 비교하여 결과에 따라 결정변수 $x_j$의 값을 조정할 방법의 기본적인 설명은 다음과 같다. 각 변수에 대하여 시뮬레이션 출력으로 얻는 목적함수의 값이 단조증가함수(monotonic increasing function) 또는 단조감소함수(monotonic decreasing function)인지를 파악하여 다음의 조건에 따라 방향을 결정한다.

즉, ① 주어진 $A_i$의 형태 [$\{f_i = c\}$, $\{f_i > c\}$, 또는 $\{f_i < c\}$], ② t-1 시점과 t 시점의 결정변수 j의 값 $x_{j,t-1}$와 $x_{j,t}$, ③ t-1 시점과 t 시점에서 시뮬레이션 모델로부터 얻어진 목적함수 i의 값 $y_{i,t-1}$와 $y_{i,t}$, ④ t 시점에서 시뮬레이션 모델로부터 얻어진 목적함수 i의 값 $y_i$와 주어진 목표치의 값 c의 관계에 따라 결정한다. 각 변수 값의 증가, 감소, 그리고 불변의 횟수를 누적하여 가장 많은 횟수를 기록한 방향으로 각 변수 값을 $\Delta x_j$만큼씩 변화시킨다. 이로 인해 변수 값의 변화 방향이 목적함수에 따라 충돌(conflict)이 발생하는 경우도 동시에 고려할 수 있다. 결정변수 값의 조정을 위해 사용되는 용어에 대한 자세한 설명은 Park and Lee(1999)의 논문을 참조한다.

시스템을 평가할 때는 본 연구에서처럼 목적함수의 값이 움직이는 방향만 평가하는 경우와 목적함수의 값이 움직이는 방향과 값의 변동 크기를 동시에 고려하는 경우로 나눌 수 있다. 확률적 시뮬레이션 모델에서 목적함수의 값이 변하는 크기를 고려하면 시스템이 확률적이기 때문에 변동되는 크기도 고려되는 시스템에 매우 의존적일 수밖에 없다. 그러므로 목적함수의 값이 변동되는 크기에 따라 시스템의 파라미터를 조정하는 작업은 무의미해질 수 있기 때문에 목적함수의 방향만 고려한다.

### 3.2 종료 알고리듬

목적식 또는 대안을 만족할 때 임의의 시점에서 시뮬레이션

을 종료해야 하는데, 각 시점 t에서 수행하게 될 알고리듬의 종료조건을 검사하기 위해 사용되는 용어 및 알고리듬을 설명하면 다음과 같다.

$x_j^*$: 현시점 t에서 최근 K번의 시점에서 결정된 변수의 값들 중에서 빈도수가 가장 높은 값

K: 종료조건을 검사하기 위해 기준이 될 최근의 시점수(K = 1, 2, ..., n)

η: 종료조건을 위해 결정변수 $x_j, j = 1, ..., n$에 대하여 허용치($x_j^* \pm \eta \Delta x_j$)를 산정하는 데 사용되는 정수(η= 0, 1, 2, ..., m)

[알고리듬 3.3]

| | |
|---|---|
| 단계 0 | K와 η값을 부여한다. |
| 단계 1 | 결정변수 $x_j, j = 1, ..., n$에 대하여 다음 과정을 수행하고, 모든 결정변수에 대하여 만족하면 단계 2로 간다.<br>(알고리듬 3.1)에 의해 $x_j^*$를 구하고, 최근 K 시점에서 결정변수 $x_j, j = 1, ..., n$의 모든 값이 $[x_j^* \pm \eta \Delta x_j]$ 범위에 들어오면 단계 2로 간다.<br>그렇지 않으면, 본 알고리듬을 중단하고 다음 시점 $t + \Delta t$까지 시뮬레이션을 계속한다. |
| 단계 2 | 목표치 c에 대하여 만족 여부를 확인하여, 만족하면 시뮬레이션을 종료한다.<br>그렇지 않은 경우,<br>η = 0이면, 본 알고리듬 및 시뮬레이션을 중단하고, '목표치를 만족하는 대안이 존재하지 않음'으로 결론을 내리고,<br>η > 0이면, η를 η = max[0,η-1]로 수정한 뒤 다음 시점 $t + \Delta t$까지 시뮬레이션을 계속한다. |

단계 0에서의 K 값( K = 1, 2, ..., n)은 시뮬레이션을 통해 얻는 시스템 수행도의 확률성 크기에 종속되며, 시스템 설계자에게 주어진 컴퓨터 종류와 할당된 시간을 고려하여 시스템 설계자가 결정해야 한다. η값(η= 0, 1, ..., m)은 정의된 문제 및 변수의 성질에 따라 달라져야 하는데, 보통 0 또는 1의 값이면 적당하다.

단계 2에서는 모든 결정변수 값들이 종료를 위한 허용범위 내에 포함되어도 목표치를 만족하지 못하면 허용범위를 줄인다. 더 이상 허용치가 없는 경우(η= 0)는 '목표치를 만족하는 대안이 존재하지 않음'으로 결론을 내리고, 알고리듬 및 시뮬레이션을 중단하고, 허용치가 있는 경우(η> 0)는 시뮬레이션을 계속 수행한다. 허용범위가 넓으면 결정변수의 변화폭이 그만큼 크다는 것을 내포한다.

따라서 허용범위가 클수록 빨리 수렴하지만 선택되는 목표치의 변화폭이 커질 수가 있기 때문에 목표치가 만족되지 못하면 허용치를 줄여서 다시 검사한다.

## 4. 수치 예제

본 장에서는 3장에서 설명한 알고리듬의 다양한 평가를 위해 3개의 모델에 대한 수치실험을 수행한다. 첫째, 시뮬레이션 시스템에 대한 이론치를 구할 수 있고, 결정변수의 값을 변경함으로서 목적함수의 방향을 알 수 있고M/M/s 대기모형으로 표현될 수 있는 제조 시스템에 적용한다. 둘째, 결정변수의 값을 변경해도 목적함수 값의 진행 방향을 알 수 없는(s, S) 재고 문제에 적용한다. 셋째, 시뮬레이션 모델이 아닌 해석적인 함수로 표현될 수 있는 문제에 적용하고 평가한다.

### 4.1 M/M/s 문제

작업물이 λ=18인 포아송 과정으로 도착하여 대기 버퍼에 대기하다가 μ=2인 지수분포로 작업을 수행한 후 시스템을 빠져나가는 M/M/s 단일 작업장 모델을 대상으로 실험한다. 대기 버퍼의 용량은 무한대이며, 기계의 작업물 선택규칙은 선입선출방법을 따른다고 가정한다. 지금까지 설명한 모델을 대기행렬 모델로 나타내면 다음의 <Figure 1>과 같이 표현할 수 있다.

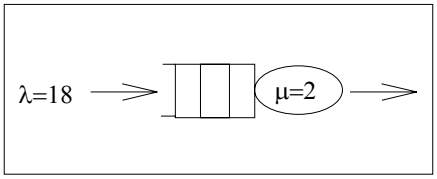


**Figure 1.** M/M/s queueing model.

본 수치예제의 목표치는 대기 버퍼에서 기다리는 작업물의 평균 대기시간($W_q$)이 10분 이하가 되도록 하는 최소 기계의 대수를 찾는 것이다. 이를 수리식으로 표현하면 다음과 같다.

$$\min_x \; arg\{E[f(X)] \in A\}$$

$E[f(X)]$= 버퍼에서 기다리는 작업물의 평균 대기시간

A: {작업물의 평균 대기시간 ≤ 10}

x: 기계 대수

M/M/s 대기행렬 모델의 평균 대기고객수($L_q$), 평균 고객수($L$), 평균 대기시간($W_q$), 그리고 평균 체재시간($W$)에 대한 공식은 Hillier and Lieberman(1990)과 Little(1961)의 식을 사용한다. Hillier and Lieberman과 Little의 공식을 사용하여 목표치를 만족하는 안정상태에서의 이론치를 구하면 기계의 수가 최소 11대가 되면 작업물의 대기시간이 10 이하를 만족하는 결과가 나온다.

본 절에서 대상으로 하는 M/M/s 문제를 Park and Lee(1999)에서 설명한 탐색 알고리듬에 적용하기 위한 변수들의 실험치를 다음과 같이 설정한다. 기계 대수 $x_1$의 증감값($\Delta x_1$)을 1로

하고, 목적함수를 평가하고 결정변수 값을 조정할 시간간격($\Delta t$)은 작업을 마친 작업물의 수가 30, 50, 100, 200, 그리고 300일 때로 가정하여 결과를 비교한다. 시뮬레이션을 시작할 때는 동일한 조건으로 실험하기 위해 같은 시드(seed) 번호를 사용하고, 시스템은 빈 상태로 시작하며, 그 때까지의 누적 평균값을 사용한다. 또한 가용한 기계 대수의 범위를 [$a_1$= 1, $b_1$= 20]로 설정하며, 실험하는 단일작업장 문제의 결정변수는 기계 대수로 한다. 기계 대수가 증가(감소)하면 작업물이 대기하는 대기시간이 감소(증가)하기 때문에 결정변수는 $L_1^{+-}$={$x_1$}에 속하게 된다.

알고리듬 3.3에서 시스템을 만족하는 기계 대수의 변화구간을 검사하는 횟수를 K = 10으로 하고 기계 대수의 허용되는 변화폭 η를 0으로 한다. 또한, 적은 수의 데이터를 가지고 분석하기 위해서 Park and Lee(1999)의 알고리듬 3.4에서 $p_{\max}$의 값을 20으로 설정한다.

위에서 설명한 조건을 기준으로 각각의 $\Delta t$에 따라 목적함수($W_q$ <= 10) 조건을 만족할 때까지 시뮬레이션을 수행한다. 대기열에서 대기하는 평균 대기시간의 추정치 $W_q$의 수렴 방향은 $\Delta t$를 30, 50, 100, 200 및 300 으로 변경하면서 관찰하면 다음의 <Figure 2>와 같이 나타난다.

<Table 1>은 탐색 알고리듬이 주어진 조건을 만족하여 종료되었을 때 최종적으로 선택된 결정변수의 값과 종료 알고리듬이 적용되어 시뮬레이션이 완료된 시간을 보여준다.

<Figure 2>와 <Table 1>의 결과를 분석해 보면 시스템을 검사하는 시간간격인 $\Delta t$가 커질수록 시뮬레이션 종료시간은 증가하지만 목적함수를 만족하는 기계 대수는 이론치인 값 11에 수렴하며, 이론치 근처에서 가능해 영역을 설정함을 알 수 있다. 또한 $\Delta t$가 커질수록 초기에 출발하는 $W_q$의 값이 크지만 시간이 지날수록 급격하게 수렴값에 근접해 감을 알 수 있다. <Table 1>에서 $\Delta t$가 30인 경우를 보면 시뮬레이션 종료시간이 5227.28인데 선택된 결정변수의 값은 20으로 이론치를 만족하지 못한다. 그러나 $\Delta t$가 50 이상인 경우에는 시뮬레이션 종료시간이 8246.61 이상으로 길어지지만 선택된 결정변수의 값이 이론치 11을 만족시킨다. 결국 $\Delta t$가 너무 작은 경우(본 실험의 경우에는 $\Delta t$< 50)는 빠른 시간에 시뮬레이션이 종료되지만 목적함수를 만족하는 결정변수의 값이 이론치에서 많이 벗어나기 때문에 적절한 $\Delta t$를 선택하는 것이 필요하다.

**Table 1.** The results of the M/M/s model

| $\Delta t$ | # of machines | Ending time |
|---|---|---|
| 30 | 20 | 5227.28 |
| 50 | 11 | 8246.61 |
| 100 | 11 | 15807.87 |
| 200 | 11 | 30381.43 |
| 300 | 11 | 45323.06 |

### 4.2 (s, S) 재고문제

(s, S) 재고문제는 결정변수의 변화에 대한 목적함수 값의 방향을 알 수 없는 결정변수를 갖는 문제로 분류되며, 결정변수가 발주 검사점 (s)와 최대 발주점 (S)로 표현되는 복수 개의 결정변수문제이다. 실험대상이 되는 (s, S) 재고문제는 Low and Kelton(1995)의 모델을 사용하며 자세한 설명은 Park and Lee(1999)와 Low and Kelton(1995)을 참조한다.

고려되는 (s, S) 재고모델에서 발생하는 비용요소는 주문비용, 유지비용 및 결품비용만 있다고 가정한다. 따라서 본 실험에서는 고려되는 3가지 비용요소의 평균 총 비용을 목적식으로 설정한다. 실험치를 계산하기 위해서 결정변수 (s, S)를 (40, 60)으로 가정하고 충분히 긴 10,000개월 동안 시뮬레이션하여 실험치의 값 125.74를 얻었다. 본 실험의 진행 방향은 $\Delta t$의 값을 변화시키면서 실험치로 설정된 목표치 125.74를 만족하는 결정변수의 값 (40, 60)을 제시된 알고리듬이 만족시키는지를 알아보는 것이다. 지금까지 설명된 (s, S) 재고문제의 수치 모델은 다음과 같이 구성된다.

$$\arg\{E[f(X)] \in A\}$$

$E[f(X)]$: 각 단위기간 동안에 발생하는 평균 재고비용

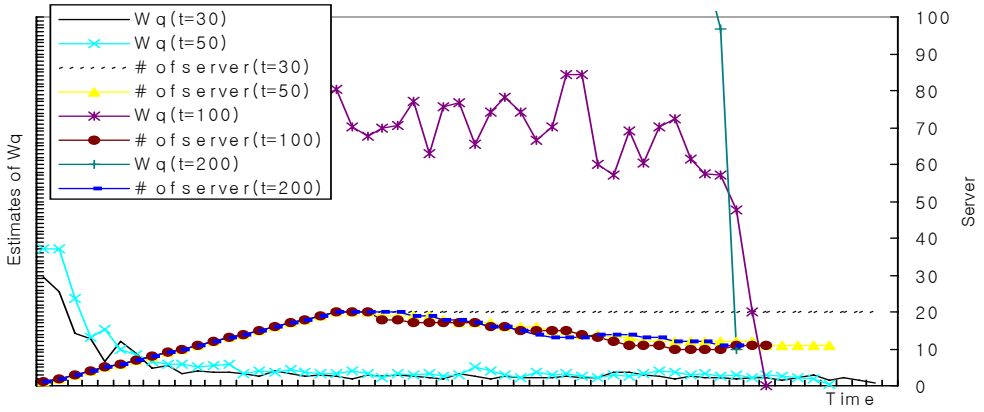


**Figure 2.** The estimates of $W_q$ and the number of server in a single workstation.

$c-\alpha \le A \le c+\alpha$
$[x_1, x_2] = [s, S]$

실험에 필요한 추가적인 조건으로서 재발주점s의 범위를 [10, 50]으로 설정하고 발주량S의 범위는 [20, 70]으로 설정한다. 시작되는 결정변수 (s, S)의 초기값은 (10, 30)으로 가정한다. 또한 종료를 위한 조건K의 값은 10으로 하고, 실험치로 주어진 c의 값을 만족하는 $E[f(x)]$의 값을 찾기 위한 c의 허용치 범위 $\alpha$의 값은 5로 가정한다. 결정변수 (s, S)의 단위변동 값 $(\Delta s, \Delta S)$는 (0, 0), (1, 0), (0, 1) 및 (1, 1)에서 선택된다고 가정한다.

본 실험에서 적용되는 자동회귀 모델은 데이터들이 상관(correlation)된 경우를 가정하기 때문에 예제에서 적용되는 Park and Lee(1999)의 알고리듬 3.2의 $y_{i,t}$의 값들은 개체들의 누적평균(accumulate average)을 사용하여 데이터의 상관 관계가 유지된다.

<Table 2>는 $\Delta t$를 10, 30, 50 및 100일 때 제시된 탐색 알고리듬과 종료 알고리듬을 만족하는 (s, S) 재고문제의 결정변수 및 평균 재고비용의 결과 값을 보여준다.

**Table 2.** The results of the (s, S) inventory model

| $\Delta t$ | (s, S) | Average inventory cost |
|---|---|---|
| 10 | (29, 49) | 121.05 |
| 30 | (32, 52) | 122.12 |
| 50 | (39, 59) | 125.20 |
| 100 | (39, 59) | 125.62 |

<Table 2>에서 $\Delta t$가 10인 경우와 30인 경우에는 종료되었을 때의 평균 재고비용이 121.05와 122.12가 되어 목표치 값 $125.74 \pm 5$를 만족하지만 선택된 결정변수는 (29, 49) 및 (32, 52)가 되어 10,000개월 동안의 실험치인 (40, 60)과는 많은 차이가 발생한다. $\Delta t$가 50과 100인 경우에는 얻어진 평균 재고비용이 125.20과 125.62로 목표치 조건 $125.74 \pm 5$를 만족하고 이 때의 결정변수의 값은 (39, 59)가 되어 실험치 (40, 60)에 근사함을 알 수 있다.

(s, S) 재고문제는 결정변수의 변화에 대한 목적함수 값의 방향을 알 수 없는 결정변수를 갖는 문제로 분류되며, 결정변수가 발주 검사점 (s)와 최대 발주점 (S)로 표현되는 복수 개의 결정변수문제이다. 실험에서 나타난 것처럼 (s, S) 재고문제에서도 $\Delta t$가 작은 경우에는 빠른 시간에 시뮬레이션이 종료되지만 목적함수를 만족하는 결정변수의 값이 안정상태 시뮬레이션을 실행하여 얻은 실험치에서 많이 벗어남을 알 수 있다.

### 4.3 Known-function 문제

4.1절과 4.2절에서는 시뮬레이션 모델로만 구성되는 문제들을 대상으로 실험하였다. 본 절에서는 알려진 함수에 회귀 모델로 표현되는 오차항목이 포함된 문제를 가지고 실험하며, 실험을 위해 사용되는 문제는 다음과 같다(Azadivar and Lee, 1988).

$$\frac{1}{18}\{(x_1+10x_2)^2+5(x_3-x_4)^2+(x_2-2x_3)^4+10(x_1-x_4)^4\}+y_t$$
$$-10 \le x_j \le 15, j=1,2,3,4$$

오차항목 $y_t$가 없는 결정적 함수(deterministic function)인 경우에 최적해는 결정변수 $x_j, j=1,2,3,4$에 대해서 $x_j=0$, $j=1,2,3,4$ 일 때 0이 얻어진다.

본 실험에서는 오차항목을 포함시켜서 시뮬레이션 모델을 구성하기 위해 $y_t = \phi_0 + \sum_{i=1}^{p}\phi_i y_{t-i} + \epsilon_t, t=1,2,...$인 함수를 포함시킨다. 이 때, $\epsilon_t$은 평균이 0이고 분산이 $\sigma_\epsilon^2$인 정규 확률변수이다. 오차항목의 실험치를 결정하기 위해, 0=1, p=1, $y_t$의 평균을 $\mu=2$라 가정하면, 2 = 1/(1-$\phi_1$)로 계산되어 $\phi_1$=0.5가 된다.

따라서 t > 1인 경우에는 $y_t = 1 + 0.5y_{t-1} + \epsilon_t$이고, t = 1인 경우에는 $y_t = 1 + \epsilon_t$로 구성되며, 분산은 $\sigma_y^2 = \frac{\sigma_\epsilon^2}{(1-0.5)^2}$로 설정된다. 이 때, $\sigma_\epsilon^2$는 평균이 0이고 분산이 10인 정규분포를 가정한다. 오차항목을 포함시키면 최적해는 결정변수 $x_j, j=1,2,3,4$에 대해서 $x_j=0, j=1,2,3,4$일 때 1이 된다. 지금까지 설명된 내용을 중심으로 본 실험에서는 $\Delta t$를 변경시키면서 목적함수를 만족하는 결정변수를 발견하는지 살펴보고 나타나는 결과를 분석한다.

또한, 알고리듬에서 사용되는 실험치로서 결정변수X의 시작점 $(x_1, x_2, x_3, x_4)$를 (12, 12, 12, 12)로 가정하고 결정변수의 허용범위 $\eta$는 1로 가정한다. 시뮬레이션 모델로부터 얻어지는 목적함수의 허용오차범위 $\varepsilon$는 30인 경우와 5인 경우를 실험한다. 실험을 할 때는 $\Delta t$의 값을 10, 50, 100, 500 및 1000으로 변화시키면서 각각의 결과를 관찰한다. 실험을 통해서 선택된 결정변수와 목적함수 값의 결과를 정리하면 다음의 <Table 3>과 같다.

<Table 3>의 결과를 분석하면 목적함수의 허용오차범위 $\varepsilon$가 30인 경우에는 $\Delta t$ > 50이면 목적함수가 허용범위 안에 들어오는 결정변수 값을 얻을 수 있다. 목적함수의 허용오차 범위 $\varepsilon$를 5로 설정하면, $\Delta t$가 1000 미만이면 조건을 만족하는 결정변수를 발견할 수 없고, $\Delta t$가 1000이면 목적함수의 허용범위를 만족하는 f = 2.892258을 얻을 수 있다. 이 때 얻어진 목적함수 값은 허용범위 가 30인 경우의 f값 14.767364보다 작고 결정변수 $(x_1, x_2, x_3, x_4)$의 값은 (2, 1, 2, 1)에서 (1, 0, 1, 0)으로 변화됨을 알 수 있다.

Known-function 문제는 결정변수가 4개이며, 결정변수의 값을 변경하면 목적함수 값의 방향을 알 수 없는 $L_i^0$ 문제에 속하

**Table 3.** The results of the known-function problem at ε=30 and ε=5

| $\Delta t$ / $\epsilon$ | 30 | | 5 | |
|---|---|---|---|---|
| | f | $(x_1,x_2,x_3,x_4)$ | f | $(x_1,x_2,x_3,x_4)$ |
| 10 | infeasible | - | infeasible | - |
| 50 | 29.055216 | (-2, -2, -2, -2) | infeasible | - |
| 100 | 28.064962 | (5, -2, -2, 5) | infeasible | - |
| 500 | 25.429712 | (2, 0, 2, 0) | infeasible | - |
| 1000 | 14.767364 | (2, 1, 2, 1) | 2.892258 | (1, 0, 1, 0) |

기 때문에 $L_i^{++}$나 $L_i^{+-}$에 속하는 결정변수를 고려하는 문제보다 탐색 방향을 찾기가 어렵다. 또한 결정변수의 수가 단일작업장 문제나 (s, S) 재고관리 시스템 문제의 결정변수의 수보다 많기 때문에 만족되는 값을 얻기 위해서 고려되는 $\Delta t$의 값도 증가됨을 알 수 있다.

## 5. 최적화 기법으로의 확장

4장에서는 개발된 알고리듬을 단일작업장 문제, (s, S) 재고관리 시스템 문제 및 Known-function 문제와 같은 다양한 경우에 적용하여 알고리듬의 일반화 가능성에 대해서 살펴보았다. 그러나 앞에서도 설명한 바와 같이 제시되는 알고리듬이 항상 최적해를 보장하지 못하기 때문에 본 장에서는 반복적인 시뮬레이션 과정을 통해서 이산형 결정변수의 최적해를 보장해 주는 기법의 초기해로 입력하여 빠른 시간에 최적해를 찾는 방안을 제시한다.

### 5.1 적용방법

기존에 사용되던 이산형 결정변수를 고려하여 시뮬레이션 최적화를 수행하는 방법들은 가능해 영역을 설정하여 가능해 영역 내에서 최적화 과정을 수행하기 때문에 가능해 영역이 설정되지 않은 경우에는 적용하기가 어렵고, 이론적으로 무한 가능해 영역(infinite feasible area)을 설정하는 경우에도 가능해 영역을 설정한 후 문제를 해결하는 데 그 초점을 두고 있다.

그러나, Andradottir(1992, 1995, 1996)의 방법에서는 가능해 영역이 주어진 경우와 주어지지 않는 경우를 고려할 수 있는 알고리듬을 제안하였는데, 이는 다음의 식 (1)과 같다.

$$\min_{n \in N} f(n) = E[X_n] \qquad (1)$$
$$where, N = \{1, 2, \ldots\}$$

식 (1)과 같은 형태로 표현되는 최적화 문제를 고려할 경우에는 이산형 변수 $n \in N$에 대해서 랜덤 변수 $Y_n$과 $Y_n^+$이 $P\{Y_n^+ > Y_n\} > 0$과 $P\{Y_n^+ < Y_n\} > 0$을 만족하고 다음의 식 (2)가 성립한다는 가정하에 기본 알고리듬을 구성한다.

$$\begin{aligned} f(n+1) > f(n) &\rightarrow P\{Y_n^+ > Y_n\} > P\{Y_n^+ < Y_n\} \\ f(n+1) < f(n) &\rightarrow P\{Y_n^+ > Y_n\} < P\{Y_n^+ < Y_n\} \\ f(n+1) = f(n) &\rightarrow P\{Y_n^+ > Y_n\} = P\{Y_n^+ < Y_n\} \end{aligned} \qquad (2)$$

식 (2)에서는 랜덤 변수 $Y_n$과 $Y_n^+$을 상태 n과 상태 n+1을 비교하는 데 사용한다. 만약 $Y_n < Y_n^+$이면, 식 (2)에 의해 $f(n) < f(n+1)$이 성립함을 알 수 있다. 이 예는 목적식을 최소화하는 문제이므로 상태 n이 상태 n+1보다 좋다고 말할 수 있다. 비슷한 이유로 $Y_n > Y_n^+$일 경우는 위와 반대로 생각할 수 있고, $Y_n = Y_n^+$인 경우는 상태 n과 상태 n+1이 목적식 $f(n)$을 최소화하는 데 미치는 영향이 없다고 말할 수 있다. 보다 자세한 설명은 Andradottir(1992, 1995, 1996)의 논문을 참조한다.

지금까지 설명된 알고리듬과 단일 실행으로 빠른 시간에 근사해에 도달할 수 있는 알고리듬 3.1과 시뮬레이션 모델을 연결하는 방법은 다음의 <Figure 3>과 같이 구성한다.

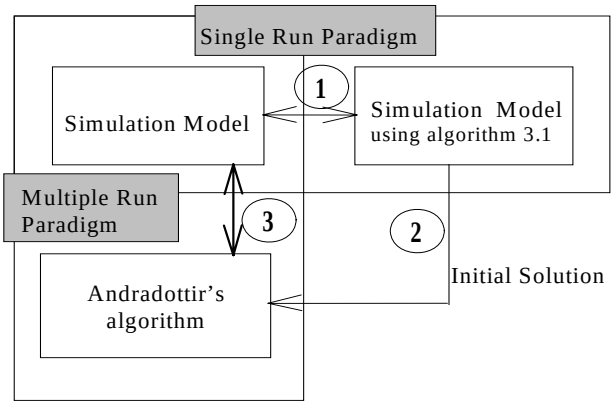


**Figure 3.** Connection of a single run model and a multiple run model.

<Figure 3>은 단일실행을 통해 빠른 시간에 얻어진 근사해를 초기치로 입력하여 최적해를 반복 실행을 통해서 얻어내는 과정이다. <Figure 3>의 ①에서는 시뮬레이션 모델과 알고리듬 3.1을 사용하여 한 번의 시뮬레이션을 실행하는 동안 적은 양의 데이터를 사용하여 목적식을 만족하는 해(근사해)를 빠른 시간에 제시한다. ②에서는 ①의 과정을 통해서 얻어진 해를 반복 실행 모델의 초기치로 입력하여 최적해를 얻도록 한다.

③에서는 Andradottir의 알고리듬과 같은 최적해를 보장하는 알고리듬을 사용하여 시뮬레이션을 반복적으로 수행함으로써 최적해를 도출한다.

### 5.2 수행도 비교

5.1절에서는 초기해의 가능해 영역이 정해지지 않을 때, 제시된 알고리듬을 사용하여 얻어진 해를 초기해로 간주하여 Andradottir의 기법과 연결하여 빠른 시간에 최적해를 찾는 알고리듬에 대해서 설명하였다.

그렇다면 5.1절에서 제안한 기법을 이용하여 최적해를 찾는 기법과 Andradottir의 기법만을 사용하여 최적해를 얻는 방법에 대한 수행도에 차이가 있는지를 살펴보는 것이 필요하다. 수행도 비교를 위해서 4.1절의 단일작업장 문제를 가지고 실험하도록 한다. 결정변수의 초기값을 0으로 하고 $\Delta t$를 200으로 설정하면 약 9200개의 부품이 작업장을 빠져나갈 때 목적함수를 만족하는 기계의 수가 12로 얻어진다. 본 장에서는 보다 정확한 해를 얻기 위해서 Andradottir의 알고리듬을 적용할 때 기계 대수 12를 초기해로 가진 경우에 수렴속도의 개선 효과를 검토한다. 여기서 가장 중요한 수행도 척도는 최적해에 도달하는 수렴시간이다. 위에서 설명한 실험 예제의 일반적인 수렴속도 경향을 살펴보면 다음의 <Figure 4>와 같다.

<Figure 4>를 살펴보면 혼합 알고리듬(Mixed algorithm)에서는 3장에서 제안된 단일 실행 알고리듬을 사용하여 짧은 동요(fluctuation)를 거친 후 단시간 내에 초기해 12를 도출해 낸다. 그 후에는 Andradottir의 방법을 이용해서 최적해인 11에 수렴한다. 그러나 Andradottir의 방법만을 사용한 경우에는 초기해를 임의로 설정해서 알고리듬을 수행하기 때문에 최적해에서 거리가 먼 값에서 많은 반복실행이 필요하게 된다. 그러므로 일반적으로 혼합 알고리듬이 Andradottir 알고리듬만을 사용한 경우보다 빠른 시간에 최적해에 수렴한다는 것을 알 수 있다.

## 6. 결론 및 추후 연구과제

본 연구에서는 이산형 결정변수를 갖는 확률적 시스템을 대상으로 단일실행으로 빠른 시간에 온라인으로 최적 시스템의 설계대안을 발견하고, 얻어진 설계대안을 안정상태에서 분석할 때 데이터의 손실을 방지하기 위해 짧은 시뮬레이션 수행으로 적은 수의 출력 데이터로 주어진 목적함수를 평가하기 위해 자동회귀 모델링 방법을 사용한 알고리듬을 다양한 실험 예에 적용하였다.

또한 제시된 알고리듬을 사용하여 최적의 시스템 설계대안을 선정할 때 시뮬레이션 모델의 확률적 특성에 의해 주어진 시스템 목표치를 만족하는 대안이 최적값을 보장하지 못할 때, 제안된 알고리듬에서 얻어진 결정변수의 값을 초기해로 간주하여 기존의 최적화 기법을 사용해 수렴속도가 향상됨을 보였다.

본 논문에서 제시된 알고리듬은 최적화 알고리듬이 아니고 단일 실행으로 목적함수를 만족하는 결정변수 값을 빠른 시간에 찾아내는 시스템 설계 문제이기 때문에 많은 반복을 통하여 최적값을 만족하는 결정변수를 찾아내는 기존 방법보다 효율이 좋다는 보장을 할 수 없다. 즉, 최적의 결정변수를 찾기 위해 목적함수 값의 허용범위 $\epsilon$를 아주 작게 주는 경우(예를 들어, $\epsilon$=0.05)에는 만족하는 설계대안을 찾아내지 못하기 때문에 시뮬레이션이 종료되지 못한다. 그러므로 본 논문에서 제시되는 알고리듬은 최적화 알고리듬이 아니고 목적식 값을 만족하는 적절한 대안을 단일실행으로 빠른 시간에 찾아내는

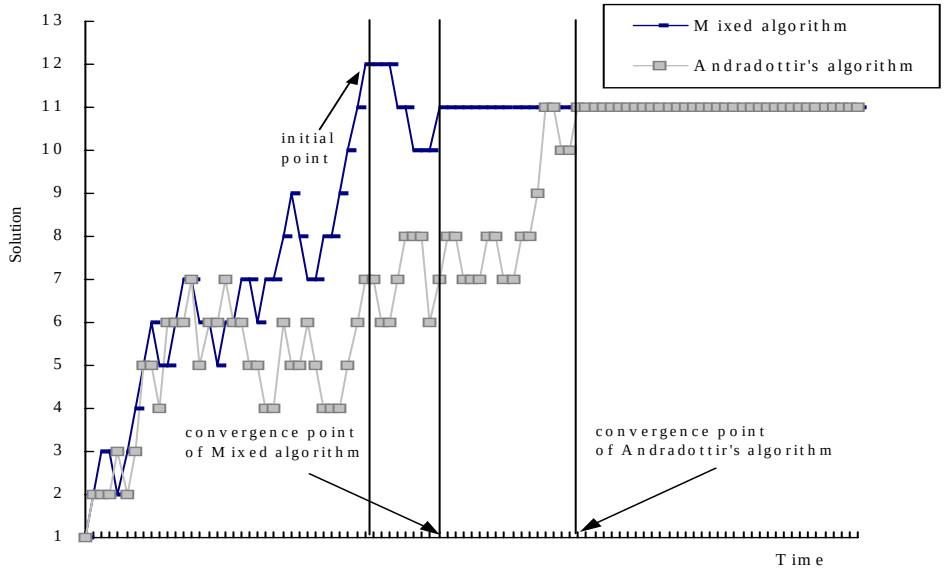


**Figure 4.** Convergence trend by Andradottir's and Mixed algorithms.

시스템 설계변수를 찾는 문제로 분류되지만 짧은 시간에 적은 양을 데이터를 사용하여 단일실행으로 해를 찾아낼 수 있다는 점에서 그 의의가 크다고 본다.

M/M/s 문제, (s, S) 재고문제 및 Known-function 문제의 결과에서 나타나 것처럼 $\Delta t$의 값은 문제의 유형에 매우 의존적이다. 그러므로 추후 연구과제에서는 시뮬레이션 모델에 탐색 알고리듬과 종료 알고리듬을 적용할 때 선택되는 $\Delta t$의 적절한 크기에 대한 종합적인 연구가 필요할 것으로 판단된다.

## 참고문헌


Ahmed, M. A., T. M. Alkhamis, and M. Hasan (1997), Optimizing discrete stochastic systems using simulated annealing and simulation, *Computers & Industrial Engineering*, **32**(4), 823-836.

Andradottir, S. (1996), A global search method for discrete stochastic optimization, *SIAM Journalon Optimization*, **6**(2), 513-530.

Andradottir, S. (1995), A method for discrete stochastic optimization, *Management Science*, **41**(12), 1946-1961.

Andradottir, S. (1992), Discrete optimization in simulation: a method and applications, *Proceedings of the 1992 Winter Simulation Conference*, 483-486.

Azadivar, F. (1992), A tutorial on simulation optimization, *Proceedings of 1992 Winter Simulation Conference*, 198-204.

Azadivar, F. and Y. H. Lee (1988), Optimization of discrete variable stochastic systems by computer simulation, *Journal of Mathematics & Computers in Simulation*, **30**, 331-345.

Barry, D. A. and B. Fristedt (1985), *Bandit problems*, Chapman and Hall, London.

Chen, H. (1994), Stochastic root finding in system design, working paper SMS 94-8, School of Industrial Engineering, Purdue University, U.S.A.

Fu, M. C. and K. J. Healy (1997), Techniques for optimization via simulation: an experimental study on (s,S) inventory system, *IIE Transactions*, **29**(3), 191-200.

Gong, W. B., Y. C. Ho, and W. Zhai (1992), Stochastic comparison algorithm for discrete optimization with estimation, *Proceedings of the 31stIEEE Conference on Decision and Control*, 795-800.

Hillier, F. S. and G. J. Lieberman (1990), *Introduction to Operations Research*, 5th ed. McGraw-Hill.

Law, A. M. and W. D. Kelton (1995), *Simulation Modeling and Analysis*, McGraw-Hill.

Lee, Y. H. and K. Iwata (1991), Part ordering through simulation-optimization in a FMS, *International Journal of Production Research*, **29**(7), 1309-1323.

Park, K. J. and Lee, Y. H. (1999), A Method for Design of Discrete Variable Stochastic Systems using Simulation, *Journal of the Korea Society for Simulation*, **8**(3), 1-16.

Pierreval, H. and L. Tautou (1997), Using evolutionary algorithms and simulation for the optimization of manufacturing systems, *IIE Transactions*, **29**, 181-189.

Shi, L. and O. Sigurdur (1997), Nested partitions method for stochastic optimization, Technical Report, Dept. of I. E., University of Wisconsin-Madison.

Wild, R. H. and J. J. Pignatiello (1994), Finding stable system designs: a reverse simulation technique, *Communications of the ACM*, **35**(10), 87-98.

Yakowitz, S. and E. Lugosi (1990), Random search in the presence of noise with application to machine learning, *SIAM Journalon Scientific Statistical Computing*, **11**, 702-712.

Yan, D. and H. Mukai (1992), Stochastic discrete optimization, *SIAM Journal on Control and Optimization*, **30**, 594-612.