論文 한국정밀공학회지 제11권 제4호 1994. 8
Journal of the Korean Society of Precision Engineering Vol. 11 No. 4 August. 1994.

컴퓨터 통합 생산을 위한 통신망의 성능 관리
제 2 부 - 의사 결정

이 석*

Performance Management of Communication Networks for Computer Integrated Manufacturing
Part II: Decision Making

Suk Lee

ABSTRACT

Performance management of computer networks is intended to improve a given network performance in order for more efficient information exchange between subsystems of an integrated large-scale system. Importance of performance management is growing as many functions of the large-scale system depend on the quality of communication services provided by the network. The role of performance management is to manipulate the adjustable protocol parameters on line so that the network can adapt itself to a dynamic environment. This can be divided into two subtasks : performance evaluation to find how changes in protocol parameters affect the network performance and decision making to determine the magnitude and direction of parameter adjustment. This paper is the second part of the two papers focusing on conceptual design, development, and evaluation of performance management for token bus networks. This paper specifically deals with the task of decision making which utilizes the principles of stochastic optimization and learning automata. The developed algorithm can adjust four timer settings of a token bus protocol based on the result of performance evaluation. The overall performance management has been evaluated for its efficacy on a network testbed.

Key Words : Performance Management, Communication Network, Computer Integrated Manufacturing, Token Bus Protocol, Stochastic Optimization, Learning Automata

1. 서 론

컴퓨터와 그 부속 장치들을 사용한 자동화의 광범위한 응용과 이로 말미암은 자동화의 섬(islands of automation)을 통합하여 보다 큰 생산성의 향상을 추구하는 추세로 인하여 컴퓨터 네트워크는 정보의 원할하고 유기적인 교환을 위하여 없어서는 안될 통합 시스템의 한 구성 요소가 되었다. 이러한 컴퓨터 네트워크는 컴퓨터 통합 생산과 항공기의 fly-by-wire를 포함한 여러 종류의 통합 시스템에 사용되고 있는데 다양한 응용 분야의 특정한 요구 사항을 만족시킬 수 있는 네트워크가 선정되어야 하고 변화하는 상황에 적절히 대처하는 능력을 가져야 한다.

* 부산대학교 생산기계공학과, 기계기술연구소

이같은 대처 능력을 통하여 네트워크의 신뢰성, 재구성에 대한 유연성, 그리고 운영의 효율성을 향상시킬 수 있다. 이러한 기능들은 네트워크의 관리(network management) 분야로 구분되는데 International Standard Organization (ISO)에 따른 주요한 세부 기능은 아래와 같다. [1, 2]

오류관리 (fault management) : 네트워크의 비정상적인 작동을 검출하고, 고립시키며, 검출된 오류로부터 회복하는 기능이다. 비정상적인 작동의 원인에는 네트워크 구성 요소의 고장이나 잘못 지정된 네트워크 변수 등이 포함된다.

구성관리(configuration management) : 사용되는 주소의 길이, 각각의 스테이션에 부여된 주소, 그리고 통신 규약에서 사용하는 시간 제한의 길이 등 네트워크의 운영에 필수적인 변수들을 초기화하고, 감시하며, 값을 변화시켜주는 기능의 집합이다.

성능관리(performance management) : 네트워크의 성능을 감시하고 이를 개선하기 위하여 조절이 가능한 통신 규약의 변수들을 변화시킴으로써 네트워크의 용량을 동적으로 할당하는 기능을 한다. 네트워크의 성능은 주로 전송 지연과 처리 능력을 포함하고 있는데 이의 개선을 위하여 시간 제한 같은 변수를 조절한다.

위에 설명한 주요 관리 기능 중 성능 관리는 네트워크의 통신 부하와 통신자원의 가용성이 시간에 따라 변화하는 피할 수 없는 사실 때문에 그 중요성이 오래 전부터 인식되어 왔다. 더구나, 광섬유의 사용으로 놀라울 정도로 증대된 통신 용량과 네트워크가 지원하는 여러 기능 간의 보다 긴밀한 통합에 대한 요구로 말미암아 하나의 네트워크로 다양한 기능을 수행하는 기구들을 접속시키는 추세는 여러 종류의 메시지를 각각의 요구 조건에 맞게 처리할 수 있는 성능 관리를 요구하고 있다. 그 일례로 그림 1에 나타난 컴퓨터 통합 생산 (computer integrated manufacturing, CIM)을 위한 네트워크를 들 수 있다. 재무와 시장 분석 등을 포함하는 기업 관리(business administration), 연구와 설계를 담당하는 공학 부서(engineering)와 제품의 생산과 그 관리를 책임지는 생산 부서(manufacturing)의 긴밀한 통합 체제를 위하여 하나의 대용량 네트워크를 사용한 예가 그림에 나타나 있다. 이러한 네트워크는 기업 관리를 위한 크지 않은 문서 file, 공학 부서에서 생성되는 상당히 큰 CAD file, 그리고 생산 부서에서 발생하는 아주 짧지만 자주 전송되어야 하는 sensor data 등 다양한 종류의 메시지를 각각의 요구 조건에 맞도록 처리하여야 하며 이 같은 기능은 새로운 생산 주문이나 생산 장비의 고장 등의 혼한 사건으로 인하여 통신 부하가 변화하는 동적인 환경에서 수행되어야 한다.

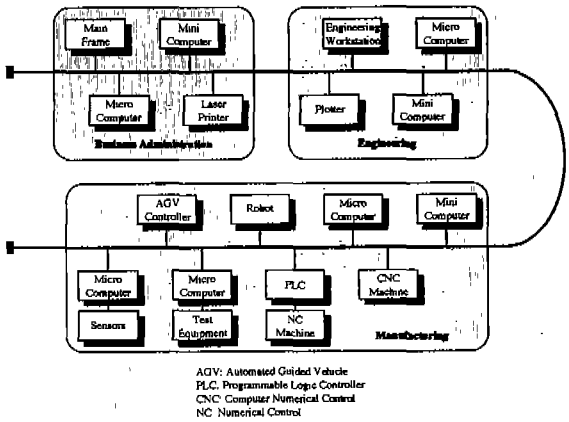


그림 1. 컴퓨터 통합 생산을 위한 네트워킹

성능 관리의 기능은 조절이 가능한 통신 규약의 변수들을 실시간에 변화시켜서 네트워크가 동적인 환경에 적응하게 하는 것이다. 성능 관리는 크게 성능 평가(performance evaluation)와 의사 결정(dicision making)의 두 부분으로 분류할 수 있는데 성능 평가는 네트워크의 성능과 통신 규약의 변수 사이의 관계를 파악하는 것이고 의사 결정은 성능 평가의 결과를 토대로 새로이 취해야 할 변수의 값을 결정하는 기능이다.

성능 관리를 위한 의사 결정에 수치적인 최적화 알고리듬을 사용할 수 있는데 선택된 네트워크의 성능 지표의 경사(gradient)를 유한 차분법(finite difference method)을 이용하여 단순한 언덕 오르기(hill climbing) 등으로 변수들을 조절하면 실시간에 네트워크의 성능을 개선할 수 있다. 이러한 최적화 알고리듬은 네트워크의 성능이 한정된 시간 동안의 관찰로 얻어지기 때문에 항상 잡음을 포함하고 있는 점을 유념하여 선택되어야 한다. 곧, 이 의사 결정의 문제는 확률적인 최적화(stochastic optimization) 문제이므로 stochastic approximation 이나 Monte Carlo method 같은 기법을 사용하여야 한다 [3, 4].

확률적인 최적화 문제에서는 최적화하려는 성능 지표가 알려지지 않은 확률 분포를 갖는 임의의 측정에 대한 기대값으로 표현된다. stochastic approximation 기법은 성능

지표의 독립 변수에 대한 유한 차분 quotient를 구하기 위하여 임의의 측정값을 사용한다. 그리고, 이러한 quotient는 어떠한 독립 변수의 값에서도 0보다 큰 분산을 가지므로 step size를 점차 감소시켜서 최적값에 수렴하도록 구성되어 있다.

많은 확률적인 최적화 문제에서 한 종류의 최적화 알고리듬이 어느 특정한 영역에서만 효과적인 경우가 있기 때문에 하나 이상의 알고리듬이 필요할 수도 있다. 이러한 경우에는 현재의 통신 규약 변수의 값과 네트워크의 통신 부하 특성 등에 근거하여 어떠한 알고리듬을 사용할 것인가를 결정하는 한 차원 높은 의사 결정이 필요하다. 이 같은 구성은 성능 관리의 효과적인 수행과 신뢰도를 향상시킬 수 있는 효과가 있고 한 차원 높은 의사 결정에 learning automaton [5] 같은 기법을 사용할 수 있다.

learning automaton은 선택할 수 있는 행동들의 집합(set of actions), 각각의 행동을 선택할 확률(action probabilities), 그리고 강화 알고리듬(reinforcement scheme)으로 구성되어 있다. 각각의 행동에 대한 확률은 그 행동이 선택되었을 때 learning automaton과 상호 작용하는 환경이 나타낸 반응에 근거한 강화 알고리듬의 작용에 의하여 계속적으로 조절된다. 한 예로 어떤 행동을 선택하여 실행했을 때 주위의 환경이 바람직한 반응을 나타냈다면 다음 행동을 선택하기 전에 강화 알고리듬은 그 행동을 선택할 확률을 증가시키고 다른 행동에 대한 확률을 감소시킨다. 이와 같은 과정을 반복하는 과정에서 가장 바람직한 행동의 확률은 점차 증가하고 그 반대의 행동에 대한 확률은 감소하게 되어서 learning auto- maton은 현재 처한 상황에 있어서 최상의 행동을 취하게 된다.

그림 2에 성능 관리의 전반적인 구조가 나타나 있다. 그림의 윗 부분에 성능관리의 대상인 네트워크가 주어진 통신 규약의 변수값을 사용하여 운영되고 이와 병행해서 변수값이 변화되었을 때의 성능을 추정하기 위하여 perturbation analysis가 수행되는 것을 나타낸다. 충분한 관찰이 이루어진 후 기준과 변화된 성능 (nominal and perturbed performance)을 추정하여 성능 관리의 의사 결정 부분에 전달한다. 이 때 바로 직전에 사용되었던 최적화 알고리듬의 유효성을 평가하기 위하여 기준의 성능이 effectiveness evaluator에 입력되고 이를 바탕으로 learning automaton은 선택할 수 있는 알고리듬에 대한 확률을 개정하고 새로운 알고리듬을 선택하여 switch에 전달한다. switch의 상태에 따라 관찰된 기준과 변화된 성능은 한 최적화 알고리듬으로 입력되고 새로운 변수값과 perturbation analysis를 위한 변화량이 결정되어 네트워크의 운영이 계속된다.

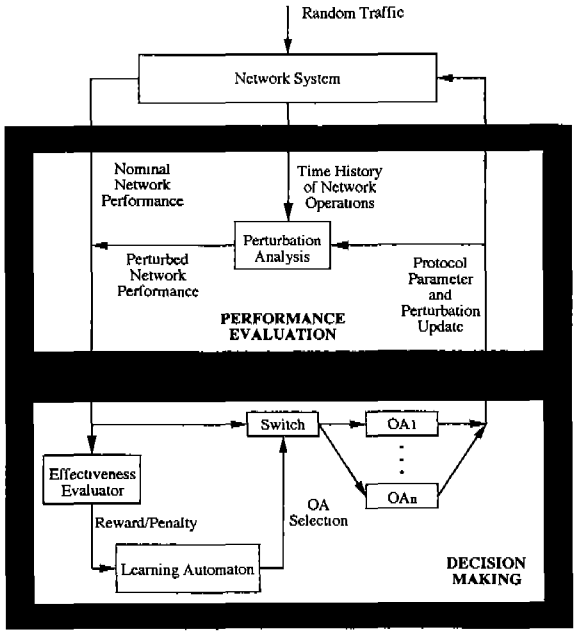


그림 2. 성능 관리의 전체적인 구조

본 논문은 복잡한 대규모의 시스템을 통합하는데 중요한 요소인 컴퓨터 네트워크의 성능 관리에 대한 연구 결과를 정리한 두 논문 중에 그 두 번째 부분으로서 의사 결정과 전체적인 성능 관리를 구현하는데 대한 연구 결과를 담고 있고 본 논문의 제 1부에서는 성능 평가에 대한 연구 결과를 기술하고 있다. 본 논문은 이 서론을 포함하여 다섯 절로 구성되어 있다. 제2절은 의사 결정 알고리듬에 대하여 기술하고 제3절에서는 이 연구에서 고찰한 토큰 버스 통신 규약을 한 네트워크에서 이뮬레이션한 과정과 성능 관리의 구현에 관하여 설명하였다. 토큰 버스 통신 규약의 성능 관리 실험을 통하여 얻어진 결과와 그에 대한 토론이 제4절에 포함되어 있다. 마지막으로 제5절은 본 연구에서 얻은 결론을 요약하고 있다.

2. 의사 결정 알고리듬

확률적인 최적화 문제는

$$\min_{x} J(x) = \min_{x} E[g(x, \omega)]$$

로 표현할 수 있는데 여기서 $J(x)$는 시스템의 변수 $x \in R^n$의 함수인 scalar의 성능 지표이다. 성능 지표의 정확한 표현은 $g: R^n \times \Omega \rightarrow R$의 확률 분포 함수가 알려져 있지 않기 때문에 불가능하고 단지 $J(x)$에 대한 임의의 측정인 $g(x, \omega)$만이 특정한 변수값 x와 표본점 $\omega \in \Omega$에 대하여 주어진다. 만일 $g(x, \omega)$의 확률 분포 함수가 알려져 있다면 문제는 결정적인 최적화 (deterministic optimization) 문제로 축소되어 일반적인 최적화 문제의 해법을 사용할 수 있게 된다. 확률적인 최적화 문제는 알려지지 않았거나 일부분만이 알려진 확률 분포 함수를 갖는 확률적인 과정(stochastic process)을 다루는 많은 공학 문제를 포함한다.

2.1 Modified Stochastic Approximation

널리 알려진 stochastic approximation [3, 4]이 어떤 확률적인 의미에서 극한값에 수렴한다는 것이 증명되어 있지만 이 알고리듬에서 step size를 k번째 iteration에서의 변수값 $x[k]$에 관계없이 균일하게 감소시키는 것 때문에 많은 실제적인 상황에서는 수렴하는 속도가 늦은 것 또한 알려져 있다. 이 같은 어려움을 극복하기 위해 modified stochastic approximation (MSA)이라고 이름 붙여진 알고리듬이 이와 유사한 문제를 다루었던 경우에 사용되었던 알고리듬 [6]의 발상을 연장하여 구성되었다. MSA는 아래와 같이 나타낼 수 있다.

$$x[k+1] = x[k] - \Gamma[k]\, f(q^{\pm}[k], \cdots, q^{\pm}[k-m_w+1])$$

여기서 $\Gamma[k]$는 step size를 나타내는 diagonal matrix이며 대각선 상의 0이 아닌 요소들은 상응하는 전방 또는 후방 유한 차분(forward or backward difference) quotient $q_i^{\pm}[k]$의 부호 전환 회수에 의하여 결정된다. 즉, $\Gamma[k]$의 i번째 대각선 요소는

$$\gamma_i[k] = \max\left(cir^{e_i[k]}, l \right)$$

으로 표현되는데 c_i는 최초의 변화를 나타내는 양의 수이고 $r \in (0, 1)$은 감소 비율, $e_i[k]$는 i번째 quotient $q_i^{\pm}[k]$의 부호 전환 회수, 그리고 l은 모든 대각선 요소에 공통인 step size의 최소 크기이다. 유한 차분 quotient $q_i^{\pm}[k]$는

$$q_i^{\pm}[k] = \frac{g(x[k] + \Delta x_i[k] u_i, \omega[k]) - g(x[k], \omega[k])}{\Delta x_i[k]}$$

로 나타낼 수 있는데 u_i는 i번째 단위 벡터이고 $\Delta x_i[k]$가 양의 부호를 취하면 전방 유한 차분, 그 반대의 경우에는 후방 유한 차분 quotient이다. 그리고 MSA 알고리듬은 오차가 포함된 quotient에 의한 변수값 $x[k]$의 지나친 변동을 줄이기 위하여 최근 m_w 회의 iteration에서 얻어진 quotient에 서로 다른 비중을 부여하여 평균을 산출하여 사용한다. 이같은 기능을 함수 f가 수행하는데 i번째 요소를 수식으로 나타내면

$$f_i(q^{\pm}[k], \cdots, q^{\pm}[k-m_w+1]) = \frac{g(x[k], w[k])}{x_i[k]} \sum_{j=1}^{m_n} \frac{w_j x_i[k-j+1]\, q_i^{\mp}[k-j+1]}{g(x[k-j+1], \omega[k-j+1])}$$

와 같고 여기서 $q^{\pm}[k] = \sum_{j=1}^{n} q_i^{\pm}[k] u_i$이고 $\sum_{j=1}^{m_n} w_j = 1$ $(w_j \geq 0 \forall j)$를 만족시킨다.

이와 같은 MSA 알고리듬의 기본적인 발상은 $x[k]$가 최적값에 접근할수록 quotient의 기대값은 0에 가까와지는 반면 한정된 시간의 관찰과 기타 다른 임의적인 요소로 인한 확률적인 변동은 그대로 남아 있기 때문에 quotient 부호의 변화는 더욱 자주 일어난다는 점에 착안한 것이다. 이와 더불어 약간 수정된 Matyas의 Monte Carlo optimization 알고리듬 [4]을 다섯번째 iteration마다 수행하여 수렴 속도를 더욱 향상시키도록 구성되었다.

2.2 Learning Automaton

learning automaton은 한정된 수의 행동중에서 하나를 선택하고 주위 환경 (environment)은 선택된 행동이 주어진 기준에 비하여 어느 정도 바람직한지를 임의적으로 learning automaton에게 알린다. 이러한 환경의 응답과 선택되었던 행동에 근거해서 automaton은 다음의 선택이 보다 낮은 응답을 받기 위하여 자기 자신을 조정한다. 이 같은 automaton의 일종인 가변 구조의 확률적 automaton (variable-structure stochastic automaton, VSSA) [5]은 간단한 구조와 다양한 기능을 수용할 수 있는 유연성으로 인하여 최적화 알고리듬을 선택하는 한 차원 높은 의

사결정을 의하여 선택되었다.

단순화된 VSSA는 세쌍 $\{\overline{\alpha}, \overline{\beta}, A\}$ 으로 표현되는데 $\overline{\alpha}$는 선택 가능한 행동의 집합, $\overline{\alpha}=\{\alpha_1, \alpha_2, \cdots, \alpha_a\}$, $\overline{\beta}$는 환경이 표현할 수 있는 반응의 집합, $\overline{\beta}=\{\beta_1, \beta_2, \cdots, \beta_s\}$, 그리고 함수 A: $\overline{\alpha}\times\overline{\beta}\rightarrow\overline{\alpha}$는 강화 알고리듬으로 선택된 행동과 그에 대한 환경의 반응을 기준하여 각 행동을 선택할 확률을 조정한다. 성능 관리를 위한 의사 결정을 위한 관점에서 살펴 보면 automaton의 행동은 최적화 알고리듬에 해당하고 환경의 반응은 선택된 알고리듬의 유효성에 해당된다.

discrete reward/penalty (DRP) [7]의 개념을 이용하여 성능 관리를 위한 강화 알고리듬이 두 가지의 행동과 다섯 가지의 반응을 위하여 개발되었다. 환경의 반응은 0, 0.25, 0.5, 0.75, 1의 값을 취할 수 있는데 이 중에 0이 가장 바람직한 반응이고 값이 증가함에 따라 바람직하지 못한 정도가 증가한다. 일반적인 경우에는 0과 1의 두 값만을 사용하는데 본 연구에서는 한정된 시간 동안의 관찰을 통하여 알고리듬을 선택에 대한 평가를 내려야 하므로 다섯 값을 이용하여 보다 신중한 평가를 얻으려고 했다. 그리고, 두 가지 행동에 대한 확률은 $M+1$ 개의 값을 취할 수 있는데 M은 2보다 큰 짝수이다. $p_1[k]$이 k 번째 iteration에서 첫번째 최적화 알고리듬을 선택할 확률을, $\alpha[k]$가 그 선택에 대한 반응을 나타낸다면 강화 알고리듬은 아래와 같이 표현될 수 있다.

$$\Delta p_1[k]=\begin{cases}\dfrac{2}{M} & \text{if } \alpha[k]=1 \text{ and } \beta[k]=0,\\ & \text{or if } \alpha[k]=2 \text{ and } \beta[k]=1\\ \dfrac{1}{M} & \text{if } \alpha[k]=1 \text{ and } \beta[k]=0.25,\\ & \text{or if } \alpha[k]=2 \text{ and } \beta[k]=0.75\\ 0 & \text{if } \beta[k]=0.5\\ \dfrac{-1}{M} & \text{if } \alpha[k]=1 \text{ and } \beta[k]=0.75,\\ & \text{or if } \alpha[k]=2 \text{ and } \beta[k]=0.25\\ \dfrac{-2}{M} & \text{if } \alpha[k]=1 \text{ and } \beta[k]=1,\\ & \text{or if } \alpha[k]=2 \text{ and } \beta[k]=0\end{cases}$$

$$p_1[k+1]=\begin{cases}\min(p_1[k]+\Delta p_1[k],\ 1) & \text{if } \Delta p_1[k]>0\\ p_1[k] & \text{if } \Delta p_1[k]=0\\ \max(p_1[k]+\Delta p_1[k],\ 0) & \text{if } \Delta p_1[k]<0\end{cases}$$

그리고 두 번째 알고리듬을 택할 확률은 $p_2[k]=1-p_1[k]$ $\forall k$를 이용하여 구할 수 있다.

3. 성능 관리의 구현과 관련된 세부 사항

3.1 성능 관리의 구현

제1부에서 기술한 성능 평가와 본 논문에서 다룬 의사 결정을 구현하는데 있어서 가장 우선적으로 고려해야 할 것은 어느 곳에 이러한 기능들을 배치할 것인가에 대한 문제이다. 성능 평가를 위한 perturbation analysis(PA)를 집중식으로 하나의 스테이션에 배치하게 되면 스테이션에 들어오는 메시지의 생성 시간과 길이 등 모든 스테이션의 상태를 완전히 기술할 수 있는 정보가 PA 알고리듬의 입력으로 필요하므로 이 같은 정보의 전송 자체가 네트워크의 성능을 저하시킬 가능성이 있다. 그러므로, 성능 평가의 중앙 집중식 구현 대신에 분산식 구현을 통하여 추가적인 전송 부하를 줄일 수 있다. 네트워크의 스테이션들은 자신이 보유하고 있는 메시지에 관한 정보를 전송할 필요가 없고 거의 독자적으로 PA 알고리듬을 수행할 수 있다. 이를 위한 외부 입력으로 제1부의 4절에서 설명된 $\Delta C(6)$가 필요한데 logical ring의 선행 스테이션이 이 정보를 토큰에 포함시켜서 전송하도록 하여 추가적인 전송 부하를 상대적으로 작게 유지하면서 성능 평가를 구현할 수 있다.

성능 관리의 의사 결정 기능을 분산식으로 구현하면 각 스테이션들이 자신이 전송한 메시지 만을 토대로 한 지역적인 성능을 향상시키려는 경향을 보일 수 있기 때문에 서로 상충되는 결정을 내리기 쉽다. 더구나, 분산식 구현은 스테이션 마다 서로 다른 변수값이 사용되므로 우선 순위의 개념이 모호해지는 단점이 있어서 일관성있는 네트워크의 운영이 어렵게 된다. 이러한 단점으로 인하여 의사 결정 기능은 성능 관리 스테이션(performance manager)이라고 불리우는 하나의 특정한 스테이션에 집중식으로 구현되는 것이 바람직하다.

이와 같이 분산식의 성능 평가와 집중식의 의사 결정이 구현된 네트워크에서는 초기에 성능 관리 스테이션이 네가지 시간 제한(THT, TRT4, TRT2, TRT0)과 PA 알고리듬의 입력인 시간 제한의 변화량을 동시에 모든 스테이션에게 전송한다(broadcast). 이 때부터 스테이션은 정상적인 작동을 시작하여 접속된 기구로부터 들어오는 메시지들을 그들의 우선 순위와 통신 규약에 따라 전송한다. 이러한 메시지의 전송은 성능 관리 스테이션에 의하여 그 회수가 기록되고 통계적으로 충분한 자료가 축적되었다고 판단되고 토큰이 성능 관리 스테이션에 주어졌을 때 각 스테이

션에 수집된 성능 보고를 요청하는 메시지를 broadcast 하고 토큰을 다음 스테이션에게 넘긴다. 다른 스테이션들은 정상적인 작동을 중지하고 자신의 성능보고를 준비하여 토큰이 주어졌을 때 다른 일반 메시지의 전송 없이 이를 성능 관리 스테이션에게 전송한다. 이러한 과정을 통하여 토큰이 다시 성능 관리 스테이션으로 돌아왔을 때 모든 스테이션으로부터의 성능 보고가 수집되어 있는 상태이고 이를 바탕으로 성능 관리 스테이션에 설치된 의사 결정 알고리듬이 새로운 시간 제한과 PA 알고리듬에 사용될 변화량을 결정하고 이 것이 다시 모든 스테이션에 전송된 후 정상적인 작동을 재개한다.

3. 2 MAP 네트워크와 토큰 버스 통신 규약의 이뮬레이션

본 연구에서 개발된 성능 관리는 Manufacturing Automation Protocol(MAP)[8]으로 운영되는 네트워크에서 구현되었다. 이 네트워크는 head-end remodulator와 coaxial cable로 구성되어 있고 PC1과 PC2로 불리우는 IBM-PC/AT호환 컴퓨터와 PC3로 이름 붙은 IBM-PC/XT호환 컴퓨터가 접속되어 있다. 각 컴퓨터에는 Industrial Networking Incorporated(INI)[9]에서 제작된 확장 카드가 내장되어 있는데 이는 INTEL사의 80186 microprocessor, 토큰 버스용 chip set, 스테이션 주소를 저장하는 PROM과 256KB의 RAM등으로 구성되어 있다. 확장 카드와 PC와의 통신은 확장 카드에 포함된 RAM에 쓰거나 그 내용을 읽음으로써 이루어지는데 이 같은 과정은 INI사의 독자적인 개발인 Data Exchange Protocol (DXP)에 의하여 통제된다. 네트워크의 초기화를 위하여 PC3는 네트워크 관리 console로 사용되는데 네트워크가 시동될 때 각 종 통신 규약과 관련된 변수값들을 PC1와 PC2로 download한다.

네트워크 관리 console에 사용되는 software는 본 연구의 초점인 토큰 버스 통신 규약의 네가지 시간 제한을 정의할 수 있지만 source code가 공개되어 있지 않아서 수동으로만 조작되어야 하기 때문에 개발된 성능 관리 알고리듬과 접속하여 사용할 수 없다. 이와 더불어 이 네트워크에는 오직 두 개의 스테이션(PC1 과 PC2)이 있기 때문에 충분한 통계를 얻기 위해서 지나치게 긴 시간이 필요하고 다양한 통신 요구에 대한 네트워크의 성능을 측정하는 것이 매우 어렵다. 이 같은 어려움을 피하기 위하여 PC1과 PC2에 네트워크를 통하여 상호 협동하는 application을 작동시켜서 짝수의 스테이션을 갖는 네트워크를 이뮬레이션하였다. PC1에는 SENDER라는 application이 작동하고 PC2에는 RECEIVER라는 application이 작동하는데 이들은 MAP의 application층에 채용된 통신 규약의 하나인 Association Control Service Element(ACSE)에 의하여 서로 정보를 교환한다. 토큰 버스 이뮬레이션은 홀수 번호와 스테이션을 이뮬레이트하는 SENDER가 짝수 번호의 스테이션을 이뮬레이트하는 RECEIVER로 Association Request를 보내는 것으로 시작된다. 모든 초기화가 완료되면 SENDER가 토큰 버스 통신 규약에 따라 1번 스테이션의 메시지 전송을 하고 전송이 완료되면 이를 RECEIVER에게 emulated token을 전송하여 알리면 RECEIVER는 2번 스테이션을 이뮬레이트한다. 이 같은 과정을 반복하여 토큰 버스 네트워크를 이뮬레이트하면서 성능 평가를 위한 PA 알고리듬을 수행하고 홀수 번호의 스테이션 중에 지정된 성능 관리 스테이션이 네 가지의 시간 제한을 조절한다. 이와 같은 이뮬레이션의 구조가 그림 3에 나타나 있다.

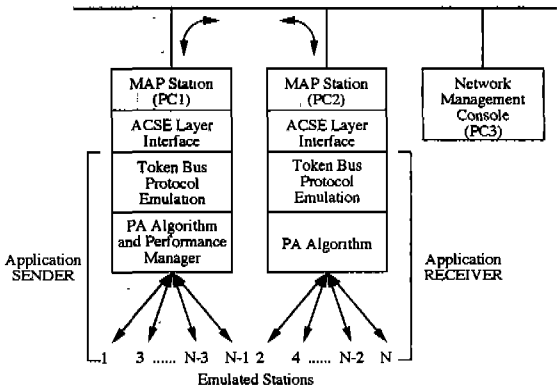


그림3. 토큰 버스 네트워크와 성능 관리 이뮬레이션구조

3. 3 성능 관리 구현의 세부 사항

네트워크의 성능 지표는 제 1부와 동일한 형태를 사용하였는데 모든 메시지의 전송 지연의 평균의 제곱을 나타내는 항을 $\rho=1$로 취하여 무시하였다. 이 같은 단순화는 성능 지표의 두항이 서로 밀접하게 correlate되어 있는 것이 관찰된 시뮬레이션의 결과에 근거한 것이다. 그리고, 시

간의 단위로는 microsecond(μsec)대신에 second(sec)가 사용되었다.

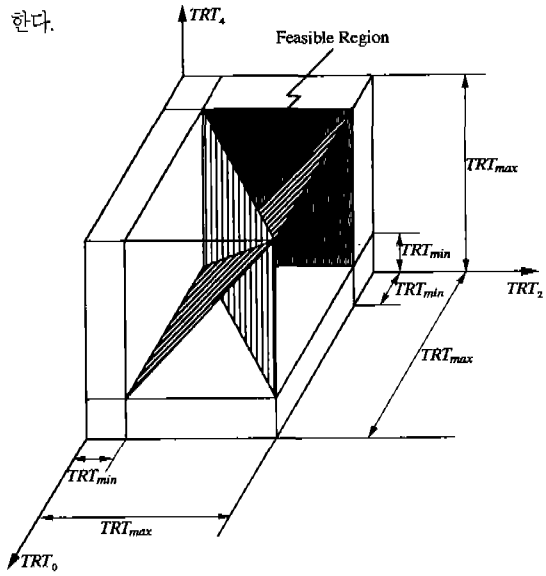
제 1부에서 설명된 PA 알고리듬은 네개의 변화된 진행(perturbed path)을 동시에 추적할 수 있고 각각의 변화된 진행에서는 네 가지의 시간 제한이 동시에 변화될 수 있다. 그리고, 제 1부의 시뮬레이션과는 달리 PA 알고리듬이 수행되는 시간 이후에 생성되는 메시지에 대한 아무런 정보없이 변화된 진행을 추적한다. SENDER와 RECEIVER가 주고 받는 emulated token에는 송신 스테이션과 수신 스테이션의 주소와 함께 토큰 수신 순간의 변화량인 $\Delta C(6)$ 가 네개의 변화된 진행에 대하여 각각 하나씩 포함되어 있다. MSA 알고리듬이 사용될 때에는 각각의 변화된 진행에서 하나의 시간 제한만을 변화시켜서 그 영향을 계산한다. 즉, 첫번째의 변화된 진행에서는 THT만을, 두번째에서는 TRT4만을, 세번째에서는 TRT2만을, 네번째에서는 TRT0만을 변화시킨다. 그 변화의 방향은 직전의 iteration에서 결정된 값의 변화와 같은 방향을 갖도록 조정하여 진행 방향과 일치하는 전방위 또는 후방위 유한 차분 quotient를 구하도록 구현되었다. 변화량은 그 바로 전 iteration에서 기준과 변화된 성능 지표가 동일했으면 두 배로 증가시키고 차이가 있을 때에는 반으로 감소시켜서 변화량이 너무 작아서 아무런 변화가 없는 경우를 피하게 하였다.

MSA 알고리듬은 앞에서 설명된 것과 같이 전방위 또는 후방위 quotient를 사용하는데 바로 직전의 quotient와 이번 iteration의 quotient를 비중이 다른 평균을 취하여 사용한다. 즉, 2. 1 절의 m_w가 2의 값을 갖는다. 조절 대상인 시간 제한의 값들은 한정된 범위 내에서 조절되어야 하는데 그 한계는 아래와 같이 나타낼 수 있다.

$$THT_{\min} \le THT \le THT_{\max}$$

$$TRT_{\min} \le TRT_0 \le TRT_2 \le TRT_4 \le TRT_{\max}$$

여기서 TRT간의 제한은 우선 순위 4가 우선 순위 2보다 더 많은 전송 기회를 갖도록, 그리고 우선 순위 2가 0보다 더 많은 기회를 갖도록 필요한 제한이다. 이러한 제한을 만족시키는 TRT의 영역이 그림 4에 나타나있다. 의사 결정 알고리듬이 결정한 새로운 값들이 이 영역을 벗어나는 경우에는 현재의 시간 제한과 새로운 시간 제한을 연결하는 선이 영역을 형성하는 면을 만나는 점을 계산하여 이를 다음 시간 제한으로 삼는다. 현재의 시간 제한이 경계면 위에 있고 새로운 시간 제한이 영역 밖에 있을 때에는 그 변화량의 벡터를 경계면에 투영하여 새로운 시간 제한을 결정한다.



Feasible region is surrounded by shaded planes and top plane of the cube

그림4. 사용가능한 TRT의 영역

2. 2절에 소개된 DRP learning automaton은 두가지의 행동을 취할 수 있다. 그러므로, 이 같은 automaton으로 두가지의 최적화 알고리듬 중에 처해있는 상황에 따라 보다 더 적합한 것을 선택할 수 있다. 선택의 기준은 선택된 알고리듬이 얼마나 적합한가를 평가하는 effectiveness evaluator의 결과에 의존한다. 이 연구에서 사용된 effectiveness evaluator는 지난 다섯 iteration동안 관찰된 네트워크의 성능과 시간 제한의 변화를 보관하여 이를 평가의 기준으로 사용한다. 선택된 알고리듬의 평가는 네 가지 부분으로 나누어지는데 네트워크의 성능, 시간 제한이 변화한 크기, 그리고 시간 제한이 변화한 방향을 과거 (지난 다섯 iteration)의 평균에 대하여 비교하고 마지막으로 시간 제한이 변화한 방향을 다음 iteration의 변화 방향과 비교한다. 이와 같은 평가를 통하여 effectiveness evaluator는 $\beta[k]$를 계산하여 DRP learning automaton이 두 알고리듬을 선택할 확률을 조정하도록 한다.

4. 결과와 토의

성능 관리의 유효성을 밝히기 위하여 10개의 스테이션으로 이루어진 토큰 버스 네트워크를 이뮬레이션하였다.

이 네트워크가 처리하는 통신 부하는 표 1에 요약된 것과 같은 통계적 특성을 갖고 있다. 우선 순의 6과 4의 메시지 생성기간(massage generation period, MGP)은 센서와 제어 신호나 화상 신호 같이 전송지연에 민감하고 어느 정도 일정한 간격으로 생성되는 메시지를 모사하기 위하여 균일 분포 (uniform distribution)로 가정되었다. 이와 같은 균일 분포에 의한 통신 부하는 본 논문의 제 1부에서 사용되였던 지수 분포보다 공장 자동화를 위한 네트워크의 통신 부하를 보다 가깝게 표현하리라고 기대된다. 이와 마찬가지로 메시지의 길이도 상당히 일정한 경향이 있다고 가정하여 균일 분포를 사용하였다. 나머지 두 하위 순위에는 지수 분포를 사용하여 메시지 생성 기간과 메시지 길이를 나타내었는데 이는 제 1부에서 설명된 것과 같이 전송지연에 그다지 민감하지 않은 file 전송 같은 통신 부하를 나타내려는 의도이다. 각 우선 순위에 대한 메시지 생성 기간과 길이의 평균값이 표 1에 주어져 있다. 이 같은 통신 부하에서 각 우선 순위는 이뮬레이션된 네트워크 용량의 약 15%를 차지한다.

Priority Level	Message Generation Period (sec)		Message Length (bytes)		Ratio to Network Capacity
	Distribution	Average	Distribution	Average	
6	Uniform	10 [0, 20]	Uniform	150 [0, 300]	0.15
4	Uniform	20 [0, 40]	Uniform	300 [0, 600]	0.15
2	Exponential	30	Exponential	450	0.15
0	Exponential	40	Exponential	600	0.15

표 1. 이뮬레이션에서 사용된 통신 부하의 특성

이뮬레이션이 수행되는 동안 각각의 시간 제한이 취할 수 있는 값은 THT의 경우 0.25에서 2초까지, 그리고, TRT4, TRT2, TRT0의 경우에는 2초에서 10초까지로 제한되었다. 네트워크의 성능 지표는 앞 절에서 언급된 것과 같이 벌칙 함수만을 고려하였고 3.5초를 초과하는 우선 순위 6의 메시지에 초과 시간이 4초 이내일 때 그 초과량의 제곱에 해당하는 법칙을 부과하였다. 즉, θ_6 는 3.5초이고 b_6는 4초이다. 우선 순위 4, 2, 0에 대해서는 θ_i는 각각 5.5, 7.5, 9.5 초로 선택되었고 b_i는 4초로 같은 값을 사용하였다. 이같은 성능 지표는 iteration마다 계산되어 저장되는데 한 iteration의 길이는 시험적인 이뮬레이션을 통하여 약 3000개의 메시지 전송으로 선택되었다.

아무런 성능 관리 없이 이뮬레이션을 수행하였을 때 시간이 지남에 따라 네트워크의 성능이 악화되는 것이 관찰되었는데 그 정확한 원인은 알려지지 않았다. 아마도 네트워크 접속 board에서 기억 장소를 부여하는데 점진적으로 더 많은 시간이 소요되는 것이 아닌가 추정된다. 성능 관리의 효과를 살펴보기 위한 실험은 100회의 iteration으로 구성되었다. 그 중 첫 25회의 iteration에서는 네트워크의 성능이 안정되기를 기다리기 위하여 아무런 성능 관리의 작동없이 이뮬레이션이 수행되었고 그 후의 75회의 iteration 동안 성능 관리를 통하여 네가지의 시간 제한을 조절하였다.

의사 결정을 위하여 두 가지의 MSA 알고리듬을 사용하였다. 그 하나는 보수적 알고리듬이라고 이름붙여졌고 나머지 하나는 진보적이라고 이름붙여졌다. 진보적 알고리듬에 비하여 보수적 알고리듬은 더 작은 초기 변화량 c_i, 더 작은 변화량의 감소 비율r, 그리고 더 작은 비중 w_1을 갖는다. 바꾸어 말하면 보수적인 알고리듬은 진보적 알고리듬 보다 초기에 시간 제한을 더 작게 변화시키고 그 변화의 크기는 iteration이 계속되는 과정에서 더 빨리 감소한다. 이와 더불어 보수적 알고리듬은 유한 차분 quotient를 계산할 때 현재의 iteration에 더 작은 비중을 부여한다. 이 두가지의 알고리듬을 이용하여 세 번의 에뮬레이션을 수행하였는데 처음의 두 에뮬레이션에서는 보수적 또는 진보적 알고리듬 하나만을 사용하여 단수 알고리듬 성능 관리(single-algorithm performance management, SPM)를 수행하였다. 그리고 마지막의 실험에서는 두 종류의 알고리듬 중 하나를 DRP learning automaton이 iteration마다 선택하는 이중 알고리듬 성능 관리 (dual-algorithm performance management, DPM)를 수행하였다.

그림 5는 보수적 SPM을 수행하였을 때 얻은 네트워크의 성능을 나타낸다. 그림에서 점선은 각 iteration에서 얻은 성능 지표의 값을 나타내고 추세를 조금 더 쉽게 알아보기 위하여 다섯 iteration씩 평균을 낸 값이 실선으로 표시되었다. 그림에서 나타난 바와 같이 보수적 SPM 이 네트워크의 성능을 향상시키는 것을 확실히 알 수 있다. 이와 동일한 실험을 진보적 SPM을 이용하여 수행하였고 네트워크의 성능이 역시 유사한 정도로 향상되는 것을 관찰할 수 있었다. 이뮬레이션이 종료될 때까지의 네트워크 성능 지표의 누진 평균(cumulative average)을 비교하면 보수적 SPM이 약간 우수한 효과를 보였다.

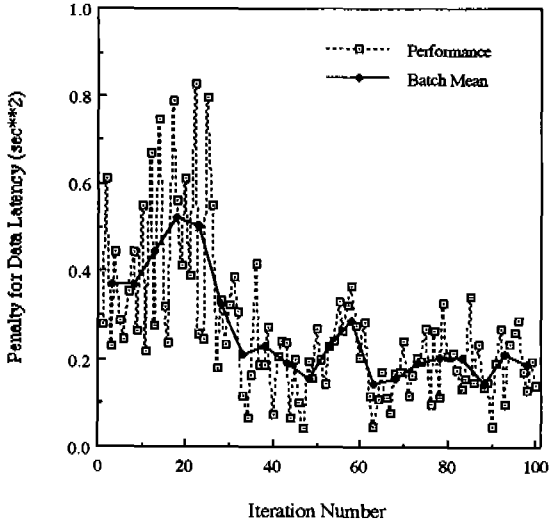


그림 5. 보수적 SPM에 의한 네트워크 성능

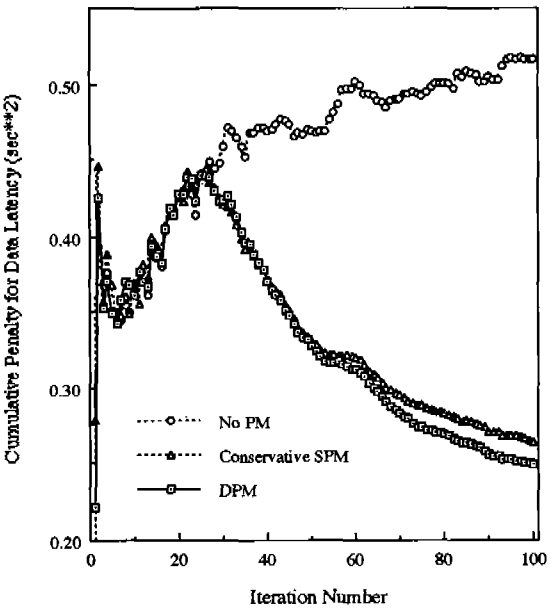


그림 6. 네트워크 누진 평균 성능의 비교

DPM을 이용하였을 때의 네트워크 성능 지표의 누진 평균이 그림 6에 나타나있다. 비교를 위하여 성능 관리가 수행되지 않았을 때와 보수적 SPM이 수행되었을 때의 네트워크 성능을 같이 나타내었다. 그림에 나타난 것 같이 DPM과 보수적 SPM 모두 성능 관리가 수행되지 않은 경우보다 월등히 우수한 성능을 얻고 있다. 이 두 가지의 성능 관리는 약 50번 째 iteration까지는 별다른 차이를 보이지 않다가 그 이후부터는 DPM이 약간 우수한 결과를 보여주고 있다. 이는 실험의 후반부에 들어서면서 보수적 SPM의 step size가 너무 빨리 감소하여 시간제한을 충분히 변화시킬 수 없는 반면 DPM은 진보적 알고리듬을 사용하여 이 같은 어려움을 극복할 수 있기 때문이다. 이와 같은 현상은 DRP learning automaton이 보수적 알고리듬을 선택할 확률의 추이를 살펴봄으로써 알 수 있는데 추가적으로 수행된 한 실험으로부터 기록된 것이 그림 7에 나타나 있다. 0.5로부터 시작된 확률이 약 50번째 iteration까지는 증가하는 추세를 보이다가 그 이후부터는 급격히 감소하는 것이 보수적 알고리듬이 시간이 지남에 따라 진보적 알고리듬에 대하여 상대적으로 비효율적인 것을 나타낸다.

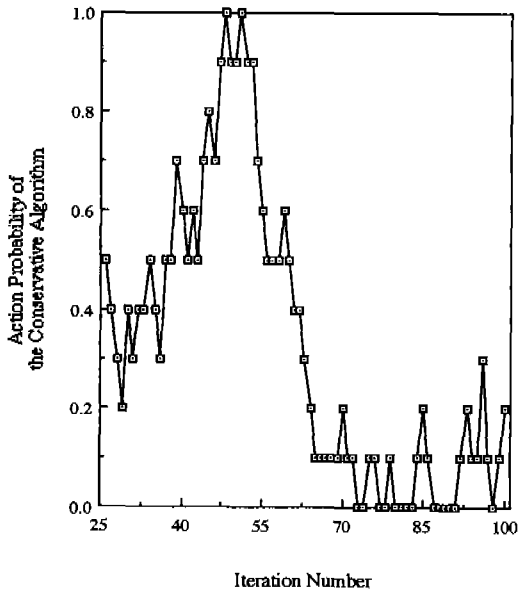


그림 7. 보수적인 알고리듬을 선택할 확률의 변화

5. 결 론

본 논문은 토큰 버스 네트워크의 성능 관리를 위하여 제 1부의 성능 평가와 함께 사용된 의사 결정 알고리듬의 개발과 총체적인 성능 관리의 평가를 수행하였다. 의사 결정 알고리듬은 확률적인 최적화의 원리와 learning automaton의 원리를 결합하여 개발되었고 토큰 버스 네트워크의 이뮬레이션을 통하여 그 효과가 입증되었다. 보다 구체적으로는 네트워크의 통신 부하에 관한 아무런 지

식이 없이 온 라인으로 네 가지 시간 제한 변수를 조절하여 네트워크의 성능을 개선할 수 있음을 보여주었다. 또한, discrete reward/penalty learning automaton을 이용하여 두 종류의 의사 결정 알고리듬을 상황에 따라 선택하여 사용함으로써 성능 관리의 효과를 더욱 향상시킬 수 있었다.

본 연구에서 개발된 성능 관리는 대규모 시스템 안에 산재한 여러 부분 사이의 효과적인 정보 교환을 가능하게 한다. 원활한 정보 교환은 변화하는 외부 환경에 적시에 유기적인 적응을 할 수 있는 컴퓨터 통합 생산(computer integrated manufacturing)같은 통합 시스템에 없어서는 안 될 요소이다.

참 고 문 헌

1. D. M. Thompson, "A Management Standard for Local Area Networks," IEEE 1985 Phoenix Conference on Computers and Communications, pp.390-396.
2. D. M. Thompson, "LAN Management Standards-Architecture and Protocols," IEEE INFOCOM 1986, pp.355-363
3. Ya. Z. Tsypkin, *Adaptation and Learning in Automatic Systems*, Academic Press, 1971.
4. R. Y. Rubinstein, *Monte Carlo Optimization, Simulation and Sensitivity of Queueing Networks*, John Wiley & Sons, 1986.
5. K. S. Narendra and M. A. L. Thatachar, *Learning Automata*, Prentice-Hall, 1989.
6. Y. C. Ho and X. Cao, "Perturbation Analysis and Optimization of Queueing Networks," Journal of Optimization Theory and Applications, Vol. 40, No. 4, August 1983, pp.559-582
7. B. J. Oommen and J. P. Christensen, "ε-optimal Discretized Linear Reward-Penlty Learning Automata," IEEE Transactions on System, Man, and Cybernetics, Vol. SMC-18, No. 3, May/June 1988, pp. 451-458.
8. *Manufacturing Automation Protocol(MAP) 3.0 Implementation Release*, Available through MAP/TOP Users Group, One SME Drive, P.O. Box 930, Dearborn, MI 48121.
9. MP-400 Programming Reference Manual, Industrial Networking Incorporated, July 1987.