論文 한국정밀공학회지 제14권 제4호 (1997년 4월)
Journal of the Korean Society of Precision Engineering Vol. 14. No. 4. April. 1997.

본 연구는 1995년도 교육부 학술연구조성비(기계공학 : ME95- E - 05)에 의하여 연구되었음

인공지능에 의한 MAP 네트워크의 성능관리기 개발

손 준 우*, 이 석**

Development of MAP Network Performance Manager Using Artificial Intelligence Techniques

Joon-Woo Son*, Suk Lee**

ABSTRACT

This paper presents the development of intelligent performance management of computer communication networks for large-scale integrated systems and the demonstration of its efficacy using computer simulation. The innermost core of the performance management is based on fuzzy set theory. This fuzzy performance manager has learning ability by using principles of neuro-fuzzy model, neural network, genetic algorithm(GA). Two types of performance managers are described in this paper. One is the Neuro-Fuzzy Performance Manager(NFPM) of which learning ability is based on the conventional gradient method, and the other is GA-based Neuro-Fuzzy Performance Manager(GNFPM) with its learning ability based on a genetic algorithm. These performance managers have been evaluated via discrete event simulation of a computer network.

Key Words : Manufacturing Automation Protocol(공장자동화용 프로토콜), Computer Integrated Manufacturing (컴퓨터 통합생산), Local Area Network(근거리 통신망), Data Latency(전송지연), Token Circulation Time(토큰 순환 시간), Priority Mechanism(우선 순위 도구), Neuro-Fuzzy Performance Manager(뉴로-퍼지 성능관리기), GA-based Neuro-Fuzzy Performance Manager(유전자 알고리즘에 기초한 뉴로-퍼지 성능관리기)

1. 서 론

오늘 날 급속히 변화하는 시장의 요구와 다품종 소량 생산 중심의 생산체제는 유연 생산 시스템(Flexible Manufacturing System, FMS)이나 컴퓨터 통합 생산(Computer Integrated Manufacturing, CIM)과 같이 보다 발전된 형태의 생산방식을 요구하게 되었다. 이러한 생산 시스템에서는 컴퓨터를 이용하여 단위 공정의 자동화를 이루고, 이를 다시 수평적, 수직적으로 통합하여 전체 공정을 일관되게 관리함으로써 생산성 향상, 생산비용 절감, 생산 공정의 유연성 제공 등의 효과를 목표로 하고 있다. 이와 같이 기업 전체를 하나의 유기적인 시

* 대우정밀공업주식회사 기술연구소
** 부산대학교 기계공학부 및 기계기술연구소

스템으로 통합하는데 있어 중추신경과 같은 역할을 수행하는 것이 컴퓨터 네트워크이다[1]. 특히, 공장자동화용 네트워크에 적합한 근거리 통신망(Local Area Network, LAN)은 제한된 공간 내에 분산되어 있는 여러 컴퓨터와 생산 장비들을 공통된 전송 매체로 연결하여 상호 간의 정보 교환을 가능케 하는 수단을 제공한다. 따라서, 시스템의 점진적인 재구성과 장비의 효율적인 사용이 가능하도록 해줄 뿐만 아니라, 시스템의 전체적인 신뢰도를 향상시켜 준다[2]. 그러나, 근거리 통신망에서는 실시간 데이터(real-time data)와 비실시간 데이터(non-real-time data)를 하나의 전송 매체를 이용하여 모두 전송해야 하므로, 컴퓨터 통합생산을 위한 네트워크를 설계하거나 관리하고자 할 때, 시간에 대한 긴급성을 요구하는 메세지들이 전송지연에 관한 요구 조건을 만족할 수 있도록 설계하여야 한다[1].

공장자동화용 네트워크 프로토콜인 MAP(Manufacturing Automation Protocol)[2-3]에서는 매체 접속 제어(Medium Access Control, MAC)의 표준으로 IEEE 802.4 토큰버스 프로토콜[4]을 선정하였으며, 여기에는 다양한 메세지들을 네 가지 우선순위로 분류하여 우선순위가 높은 메세지를 우선적으로 전송할 수 있는 기능을 제공한다. 이러한 우선순위는 네 개의 타이머, 즉 THT(Token Holding Timer)와 TRTi(Token Rotation Timer i, i=4,2,0)로 구성된다. 이러한 네 가지 타이머 값의 설정은 전송지연, throughput 등과 같은 토큰버스 네트워크 성능에 직접적인 영향을 끼친다. 따라서, 네트워크의 설계시 프로토콜 변수의 값을 결정하는 것은 매우 중요한 과제이며, 아직 체계적인 방법이 정립되어 있지 않아서 어려운 작업이라고 할 수 있다. 그리고, 네트워크의 통신부하와 통신자원이 시간에 따라 변하기 때문에, 프로토콜 변수의 값을 동적인 환경에 맞추어 자동적으로 조정할 수 있는 기능도 요구되고 있다. 이와 같이 통신규약의 조절 가능한 변수들을 실시간에 변화시킴으로써 네트워크가 동적인 환경에 적절히 대응하여 요구되는 성능을 유지하도록 하는 기능을 성능 관리 기능이라 한다. 네트워크 성능 관리는 국제 표준화 기구(ISO)에서 제안하는 네트워크 관리(network management) 기능들 중 하나이며, 이러한 네트워크 관리는 오류관리(fault management), 구성관리(configuration management), 성능관리(performance management) 등의 세부 기능으로 구분된다[5]. 그러나, 네트워크 관리기능에 관한 표준은 주로 관리의 기능과 구조를 정의할 뿐이며, 네트워크 운용시 발생하는 문제를 실시간에 관리하는 방법에 관해서는 언급하지 않는데, 이는 네트워크 관리를 위한 구체적인 방법의 개발은 표준화의 범위를 벗어나기 때문이다.

네트워크 관리기능 중에서, 성능관리를 구현하기 위한 연구가 시작된 것은 최근의 일이며, 한 연구에서 Perturbation Analysis(PA), Stochastic Approximation(SA), Learning Automata(LA)등의 이론을 복합적으로 사용하여 그 유효성을 입증한 바가 있다[6]. 그러나, 그 구조가 상당히 복잡하며, 통신규약을 일부 개정해야 할 필요도 있었다. 이러한 어려움을 극복하기 위하여 인간의 추론기능을 모방한 퍼지논리를 도입하였고 그 효용성을 시뮬레이션을 통하여 입증하였다[7-9]. 하지만, 퍼지논리에서 사용되는 퍼지집합들의 소속함수를 결정하기 위한 체계적인 방법이 없기 때문에 적절한 관리기능을 구현하기까지는 상당한 시행착오를 겪었다.

본 논문에서는 이러한 시행착오를 신경망이론과 유전자 알고리즘을 이용하여 보다 효율적으로 수행하도록 이전의 연구를 보완한 결과를 제시하고 있다. 신경망이론과 퍼지논리를 결합한 뉴로-퍼지 성능관리기(Neuro-Fuzzy Performance Manager, NFPM)는 소속 함수를 자동 조절하는 기능과 인간의 지식을 신경망의 구조에 주입할 수 있는 능력을 가지고 있기 때문에[10], 온-라인 학습을 통하여 적절한 퍼지언어 변수영역을 찾아낼 수 있음을 보였다[11]. 또한, 신경망이론의 최적값 탐색기능을 제거하고 유전자 알고리즘으로 대체한 유전자 알고리즘[12]에 기초한 뉴로-퍼지 성능관리기(GA-based Neuro-Fuzzy Performance Manager, GNFPM)를 제안하여 비선형이고 불연속적이며 다수의 극점을 갖는 탐색공간에서 효율적으로 전역 최적값으로 수렴할 수 있는 기능을 부여하였다.

본 논문은 모두 다섯 절로 이루어져 있다. 2절에서는 gradient method에 바탕을 두고 학습을 수행하는 뉴로-퍼지 성능관리기(NFPM)의, 그리고 3절에서는 유전자 알고리즘에 기초한 뉴로-퍼지 성능관리기(GNFPM)의 구성과 특징을 설명하였다. 4절에서는 NFPM과 GNFPM을 이용하여 시뮬레이션을 수행한 결과를 고찰하고, 각 모델을 평가하였다. 끝으로 5절에서는 4절에서 얻어진 결과를 토대로 한 결론을 수록하였다.

2. 뉴로-퍼지 성능관리기 (Neuro-Fuzzy Performance Manager)

공장자동화용 네트워크와 같이 실시간 데이터와 비실시간 데이터를 모두 전송해야 하는 네트워크에서는 실시간 데이터의 전송을 보장할 수 있도록 네트워크의 용량을 적절히 할당해야 하기 때문에 성능관리는 매우 중요한 기능이다.

MAP에서 사용된 IEEE 802.4 프로토콜을 위한 성능관리기는 프로토콜에서 정의된 네 가지 타이머를 조절하여 네 가지 우선순위에 속하는 메시지들이 각각 요구되는 전송지연의 한계를 넘지 않도록 네트워크의 성능을 관리하는 것이 그 목적이다. 이러한 목적을 위하여, 신경망(Neural Networks)이 갖는 장점인 학습에 의한 적응성과 병렬 분산 처리에 의한 빠른 계산 능력 등의 특성을 이용하여 네트워크의 어드미션 제어(admission control)와 스위치 제어(switch control)를 수행하는 신경망 제어기도 개발된 바 있다[13]. 그러나, 신경망의 구조와 크기를 결정하는 정량적인 방법이 존재하지 않기 때문에, 은닉층의 개수와 각 은닉층에 필요한 뉴론의 개수 등을 명확히 계산해 낼 수 없으며, 신경망의 각 연결 가중치가 의미하는 바를 알 수 없기 때문에, 설계자가 사전 지식을 가지고 있어도 이를 신경망에 주입할 수 없다는 점이 보다 광범위한 응용에 걸림돌이 되고 있다.

본 연구에서 개발된 뉴로-퍼지 성능관리기(Neuro-Fuzzy Performance Manager, NFPM)는 Fig. 1과 같이 입력층을 포함하여 모두 다섯 층으로 이루어진다. 이는 선행 연구[8-9]에서 얻어진 퍼지규칙인 Table 1을 병렬 분산 구조로 나타낸 것이다. NFPM에서 각 층 사이의

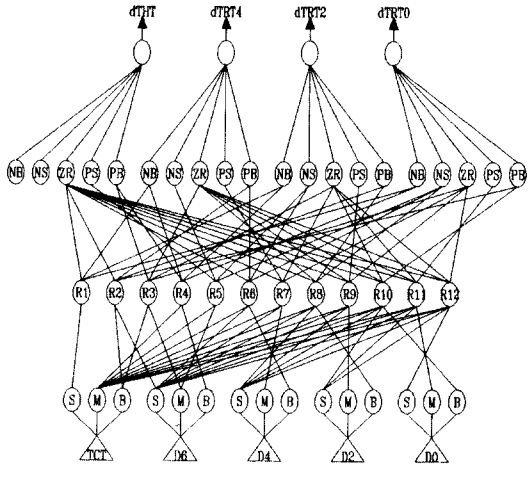

Fig. 1 Architecture of NFPM

가중치는 0 또는 1로 고정되는 것으로 가정하였으며, Fig. 1에서의 가중치에 대한 표현은 가중치가 1인 경우와 0인 경우는 각각 실선과 무연결로 나타내었다. 따라서, 학습을 통해 조절되는 파라미터는 가중치가 아니라, 비퍼지화층의 각 뉴론이 갖는 소속 함수의 중심점과 폭이다. 다음에서 각 층의 기능을 구체적으로 기술한다.

Table 1. TCT-based Fuzzy Rules

	TCT	D6	D4	D2	D0	ΔTHT	ΔTRT4	ΔTRT2	ΔTRT0
1	S					ZR	NB	NB	NB
2	L	S				ZR	PB	PB	PB
3	L	M				PB	ZR	ZR	ZR
4	M	B				PB	NB	NB	NB
5	M	M				PS	NS	NS	NS
6	M	S	B			ZR	PB	NS	NS
7	M	S	M			ZR	PS	ZR	ZR
8	M	S	S	B		ZR	ZR	PB	NS
9	M	S	S	M		ZR	ZR	PS	ZR
10	M	S	S	S	B	ZR	ZR	ZR	PB
11	M	S	S	S	M	ZR	ZR	ZR	PS
12	M	S	S	S	S	ZR	ZR	ZR	ZR

Layer 1 : 입력층

입력 값들을 Layer 2로 전달하는 역할을 수행하며, 이를 식으로 나타내면 다음과 같다.

$$O_i^1 = u_i^1 \tag{1}$$

여기서, u는 입력, O는 노드의 출력을 나타내며, 아래 첨자는 입력층에서의 노드 번호를 나타내고, 윗첨자는 층을 나타낸다.

NFPM에서 입력은 평균 토큰 순환 시간(Token Circulation Time, TCT : 토큰이 논리적 링을 한 바퀴 돌고 처음 위치로 돌아오는 데 걸린 시간)과 각 우선순위에 대한 평균 전송지연(D6, D4, D2, D0)에 3배의 표준편차를 더한 값이다. 이러한 방법을 취한 것은 대부분의 메세지가 전송지연 요구 조건을 만족시킬 수 있도록 하기 위한 것이다.

Layer 2 : 퍼지화층

퍼지화층은 각 입력에 대해, 퍼지 언어 변수의 소속값을 계산하며, 종 모양의 소속 함수를 이용하여 퍼지화를 수행한다. 다음의 식에서 c는 소속 함수의 중심점, s는 소속함수의 퍼진정도를 나타낸다.

$$O_i^2 = e^{-(\frac{u_i^2 - c_i}{s_i})^2} \quad (2)$$

Layer 3 : 퍼지 AND 연산층

퍼지 규칙에서 조건부에 부합하는 정도 즉, 규칙의 fire strength를 계산하는 층으로 퍼지 AND 연산을 수행하며 p는 layer 3의 i번째 뉴론에 연결된 입력의 수이다.

$$O_i^3 = \min(u_{i1}^3, u_{i2}^3, \cdots, u_{ip}^3) \quad (3)$$

위의 식에서 보이는 것과 같이 AND 연산자로 min함수를 선택하였다. 이는 Mamdani에 의해 제안된 추론법을 따르는 퍼지 제어기에서 일반적으로 사용되는 것으로 기존의 퍼지시스템을 그대로 뉴로-퍼지로 변환할 수 있도록 하기 위해 사용하였다.

Layer 4 : 퍼지 OR 연산층

여러 가지 규칙들이 결론부의 동일한 언어 변수에 대해 서로 다른 추론 결과를 제시할 경우 이를 통합하는 역할로서 퍼지 OR연산을 수행하는 층이다. 본 연구에서는 일반적으로 많이 이용되는 max함수를 OR연산자로 사용하였고 아래 식에서의 m은 layer 4의 i번째 뉴론에 연결된 입력의 수이다.

$$O_i^4 = \max(u_{i1}^4, u_{i2}^4, \cdots, u_{im}^4) \quad (4)$$

Layer 5 : 출력층

출력층은 소속 정도에 따라 가중평균을 취하여, 비퍼지화된 최종 출력을 구하는 층으로 최종 출력은 각 우선순위에 대한 타이머 값의 변화이다. 타이머의 변화량을 계산하기 위하여 면적중심법과 유사한 방법을 사용하였다. 출력층의 소속함수를 밑변의 폭이 s_j, 높이가 u_{ij}^5, 밑변의 중심이 c_j에 놓인 이등변 삼각형으로 제한되고, 여러 입력에 의해 중첩되는 부분을 중복해서 계산하여 Fig. 2에 나타난 것과 같이 비퍼지화를 수행하였다. 이를 식으로 나타내면 다음과 같고 n은 layer 5의 i번째 노드에 연결된 입력의 수이다.

$$O_i^5 = \frac{\frac{1}{2}\sum_j^n c_j(s_j u_{ij}^5)}{\frac{1}{2}\sum_j^n s_j u_{ij}^5} \quad (5)$$

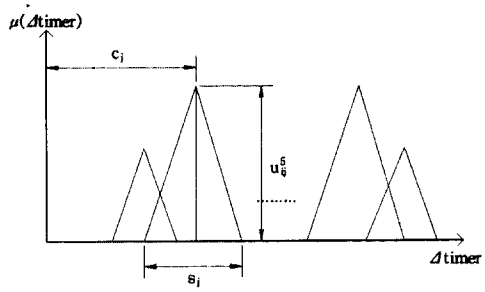

Fig. 2 Defuzzification Method

NFPM은 gradient method에 의해 출력층의 소속 함수의 중심과 폭을 조절하는 데, 성능 지표는 식(6)과 같이 정의하였다. 이러한 에러값은 IEEE 802.4 네트워크에서 전송된 각 메세지의 지연에 근거하여 계산된다.

$$E(\delta_j^i) = \frac{1}{m_a}\sum_{i=6,4,2,0}\sum_{j=1}^{m_i} F_i(\delta_j^i) \quad (6)$$

여기서, m_a는 주어진 관찰주기에서 관찰된 메세지의 개수, i는 우선순위, m_i는 우선순위 i의 메세지 개수, δ_j^i는 우선순위 i의 j번째 메세지의 전송지연, $F_i(\cdot)$는 우선순위 i의 벌칙 함수로서 식(7)과 같이 정의된다.

$$F_i(\delta_j^i) = \begin{cases} 0 & if \quad \delta_j^i \le \theta_i \\ (\delta_j^i - \theta_i)^2 & if \quad \theta_i < \delta_j^i \le \theta_i + b_i \\ b_i^2 & if \quad \delta_j^i > \theta_i + b_i \end{cases} \quad (7)$$

여기서, θ_i, b_i는 각각 우선순위가 i인 메세지의 벌칙한계와 벌칙폭이다. 즉, 한 메세지가 벌칙한계 이하의 전송지연을 가지면, 그 메세지에 대한 벌칙은 없고, 벌칙한계를 넘는 양이 벌칙폭보다 작을 때까지는 그 초과량의 자승에 해당하는 벌칙을 받는다. 초과량이 벌칙폭보다 커지면 일정하게 벌칙폭의 자승의 값을 벌칙으로 부과받는다. 이러한 벌칙의 평균이 바로 $E(\delta_j^i)$이다. 이는 주어진 제한시간을 조금 넘기는 경우에 초과된 시간에 따라 벌칙을 부과하여 상당수의 메시지가 제한시간을 넘기지 않도록 하는 반면에 벌칙폭보다 더 크게 초과한 경우에는 그 초과량에 상관없이 전송된 정보의 가치가 거의 없으므로 일정한 벌칙을 부과하는 것이다.

각 파라미터에 대한 에러 함수의 구배는 연쇄 법칙

(chain rule)을 이용하여 구할 수 있다. NFPM의 출력(타이머 변화량)이 에러 함수에 미치는 영향은 각 타이머에 약간의 섭동(perturbation)을 주어 차분값으로 얻는다. 이는 성능 지표를 해석적인 방법에 의해 구해 낼 수 없기 때문이다. 따라서, 다음의 식(8), (9)에서 이용되는 $\partial E / \partial O_i^5$ 는 차분식으로 대체하였다.

$$\frac{\partial E}{\partial c_j} = \frac{\partial E}{\partial O_i^5} \cdot \frac{\partial O_i^5}{\partial c_j} = \frac{\partial E}{\partial O_i^5} \cdot \frac{s_j u_{ij}^5}{\sum s_j u_{ij}^5} \tag{8}$$

$$\frac{\partial E}{\partial s_j} = \frac{\partial E}{\partial O_i^5} \cdot \frac{\partial O_i^5}{\partial s_j}$$

$$= \frac{\partial E}{\partial O_i^5} \cdot \frac{c_j u_{ij}^5 (\sum s_j u_{ij}^5) - (\sum c_j s_j u_{ij}^5) u_{ij}^5}{(\sum s_j u_{ij}^5)^2} \tag{9}$$

이렇게 얻어진 구배는 성능지표를 감소시키기 위하여 다음과 같이 출력층의 소속함수를 조절한다.

$$c_i(k+1) = c_i(k) - \eta \frac{\Delta E}{\Delta O_i^5} \frac{s_j u_i}{\sum s_j u_i} \tag{10}$$

$$s_i(k+1) = s_i(k) - \eta \frac{\Delta E}{\Delta O_i^5} \frac{c_j u_i (\sum s_j u_i^5) - \sum c_j s_j u_i^5 (u_i^5)}{(\sum s_j u_i)^2}$$

조건부의 파라미터를 조절하기 위해서 연쇄 규칙을 계속 적용할 수 있으나 NFPM의 AND와 OR연산에 대한 미분이 불가능하므로 이를 위해서는 AND와 OR연산 대신에 product나 bounded sum과 같은 연산으로 교체해야 할 필요가 있다. 따라서 본 연구에서는 결론부의 소속함수만을 조절하였다.

3. 유전 알고리즘에 기초한 뉴로-퍼지 성능관리기 (Genetic-Neuro-Fuzzy Performance Manager, GNFPM)

GNFPM은 컴퓨터 네트워크의 신경망 모델, 뉴로-퍼지 성능관리기, 유전 알고리즘 최적화기(GA optimizer)로 구성되며, Fig. 3과 같은 구조로 모듈간에 상호 작용하게 된다. 여기서, NFPM은 프로토콜 파라미터를 조절하여 컴퓨터 네트워크의 성능관리를 수행한다. 이때, NFPM의 자체 학습기능은 배제되며, GA 최적화기가 신경망에 의한 컴퓨터 네트워크 모델을 이용하여 NFPM의 소속 함수를 변화하는 환경에서 최적으로 유지하기 위해 학습을 수행한다.

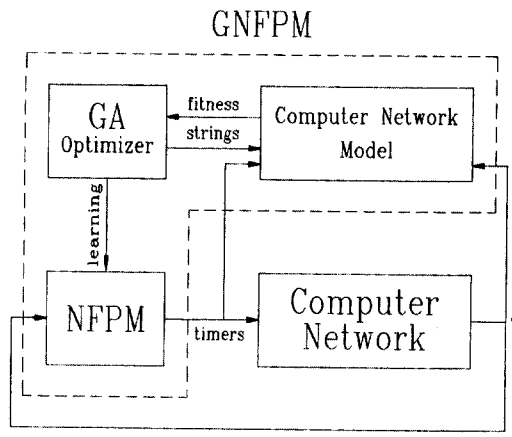

Fig. 3 Configuration of GNFPM

유전 알고리즘이란, 생물의 진화 이론과 자연의 유전 현상에 기초하여 만들어진 최적화(optimization) 또는 탐색(search) 알고리즘을 말한다. GA는 탐색 공간상에서 다양한 의미를 갖는 염색체로서 동시에 탐색을 수행하므로 전역 최적값으로 수렴할 가능성이 크며 미분 가능이나 연속 조건과 같은 제약 조건에 대한 가정이 필요 없으므로 많은 파라미터를 갖는 시스템의 최적화를 위한 기법으로 매우 효과적이다.

이러한 유전 알고리즘의 진화 과정은 염색체 스트링에 대한 재생산(reproduction), 교배(crossover), 그리고 돌연변이(mutation)등의 연산자에 의해 이루어지며, 이러한 유전 연산(genetic operation)을 수행하기 위해서는 적합도(fitness)를 평가하여야 한다[14].

적합도는 주어진 문제에 대한 스트링의 성능 척도(performance measure)가 되며, 적합도 평가는 각 스트링에 대한 목적 함수(objective function)를 계산하는 것으로 이루어진다. 일반적으로 컴퓨터 네트워크에는 특정한 조건하에서의 전송지연을 계산할 수 있는 해석적인 방법이 존재하지 않기 때문에, 시뮬레이션을 통해 목적함수를 계산할 수 있다. 그러나, 컴퓨터 시뮬레이션을 통해 유전 알고리즘의 모든 개체군에 대한 적합도를 계산할 경우, 관리기의 계산 부담이 매우 커질 뿐만 아니라, 시뮬레이션에 소요되는 시간으로 인해 실시간 관리가 불가능해진다.

따라서, GNFPM에는 신경망의 함수 근사 능력을 이용하여, 컴퓨터 네트워크를 신경망이 동조할 수 있도록 학습시킨 신경망 모델이 포함되었다. 컴퓨터 네트워크의 동

조를 위해 사용된 신경망은 Fig. 4와 같이, 입력층을 포함하여 세 개의 층으로 구성되었다.

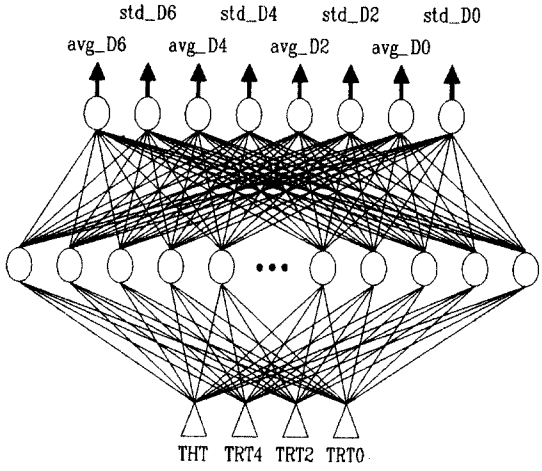

Fig. 4 Neural Network for Computer Network Identification

이 신경망 모델의 입력은 토큰버스 프로토콜의 네 가지 타이머이고, 출력은 네 개의 우선순위에 대한 전송지연(data latency)의 평균과 표준편차이다. 신경망 모델은 현재의 토큰버스 네트워크를 정의해 주는 함수라고 생각할 수 있다. 즉, 네 개의 타이머 값을 대입하면, 각 우선순위의 평균 지연과 표준편차를 얻을 수 있게 된다. 이러한 신경망 모델은 GA 최적화기에서 목적함수(objective function)의 기능을 수행한다.

입력층의 뉴론 개수는 네 개이며, 이는 각각 네 가지 타이머에 대응한다. 그리고, 은닉층의 뉴론 개수를 결정하는 정량적인 방법이 존재하지 않기 때문에, 몇 가지 선택에 대하여 근사 능력을 비교 평가하여 결정하였다. 출력층의 뉴론 개수는 8개 즉, 우선순위 6, 4, 2, 0에 대한 전송지연의 평균과 표준편차이다. 신경망 모델의 학습 데이터는 시뮬레이션 모델을 이용하여 여러 가지 조건에 대한 토큰버스 네트워크의 거동을 관찰하여 얻어졌다.

본 GNFPM의 유전 알고리즘 최적화기가 갖는 각각의 스트링은 64개의 파라미터를 포함하고 있다. 조건부의 각 우선순위에 해당하는 전송지연을 위한 퍼지언어변수 Small, Medium, Big의 중심점과 폭을 표현한 24개와 결론부 ΔTHT, ΔTRT4, ΔTRT2, ΔTRT0에 대한 퍼지언어변수 NB, NS, ZR, PS, PB의 중심점과 폭을 표현하기 위한 40개이다. 여기서, 조건부의 학습까지도 고려한 것은, 유전알고리즘의 장점으로 인해 사용자의 제약조건을 쉽게 포함시킬 수 있고, 비교적 넓은 영역에 대해 학습을 수행할 수 있기 때문이다. 또한 각 스트링에서 하나의 파라미터는 16bit의 2진수에 의해 표현되었는데, 이진화를 위해 각각의 변수가 가질 수 있는 최소값과 최대값을 이용하여 선형 scaling하고, 이를 2진수로 나타내었다.

각 스트링의 적합도는 식(11)을 이용하여 계산되는데, s_j는 j번째 스트링이고, K_T는 적합도의 표현영역을 조절하기 위한 상수이며, E는 성능지표로서 식(12)에서 정의되었다.

$$F(s_j) = \frac{K_T}{E} \tag{11}$$

$$E = E_6 + E_4 + E_2 + E_0 \tag{12}$$

$$\text{단 } E_j = \begin{cases} K_i(y_i - d_i)^2 & \text{if } y_i \geq d_i \\ 0 & \text{if } y_i < d_i \end{cases}, \quad i = 6,4,2,0$$

여기서, K_i는 상수이고, d_i는 전송지연에 대한 threshold이고, y_i는 신경망 모델에 의해 얻어진 우선순위 i에 대한 평균전송지연과 세 배의 표준편차의 합이다. 식(12)에서 각 우선순위에 대한 성능지표는 전송지연이 threshold를 넘는 경우에 대해 그 정도를 기록하기 위한 것이며, 각 우선순위별로 가중치를 부여하기 위해, K_i를 곱하는데, 그 값은 시뮬레이션을 통해 시행착오적으로 적절히 조절되었다.

GNFPM이 네트워크 성능관리와 학습기능을 수행하는 절차는 Fig. 5에 주어진 순서도와 같다. 즉, 네트워크의 성능을 동조하는 신경망 모델의 off-line학습이 완료된 후 주어진 시간동안 네트워크의 성능을 관찰한 후 GNFPM이 성능관리를 시작한다. 초기 population에 포함된 스트링(소속함수)들을 사용하여 새로운 타이머 값들을 결정하고 이 타이머 값을 신경망 모델로 입력한다. 신경망 모델은 각 스트링에 대한 네트워크 성능을 산출하고 이를 이용하여 적합도를 계산한다. 이 적합도를 이용하여 스트링을 재생산(reproduction)하고 얻어진 스트링에 대하여 교배와 돌연변이를 일으켜서 다음 세대를 위한 population을 구성한다. 이때, 어느 세대이상 진화가 이루어 지거나, 현 세대의 평균적합도와 전 세대의 평균적합도의 비가 1보다는 크지만 그 정도가 미미할 경우 학

습을 멈춘다. 그리고, 새롭게 학습된 스트링 중 가장 적합도가 높은 것을 이용하여 GNFPM은 새로운 타이머 값을 추론하고, 이 값으로 네 개의 타이머를 세트함으로써 하나의 사이클을 완성하게 된다.

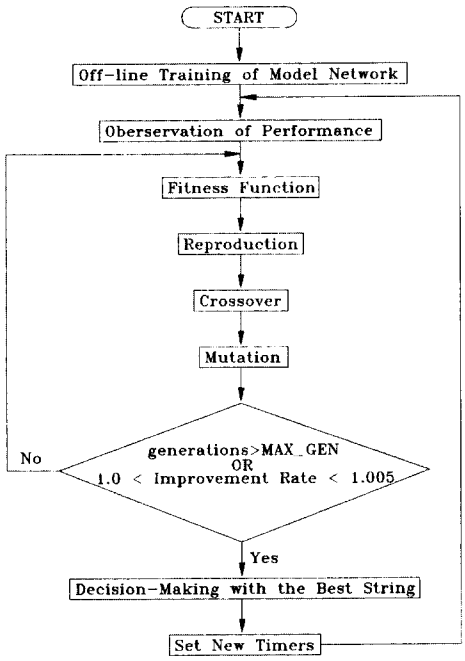

Fig. 5 Procedure of GNFPM Learning Using GA-Optimizer

4. 결과와 고찰

4.1 NFPM의 평가

NFPM의 유효성을 입증하기 위하여 NFPM에 학습을 수행하지 않는 경우와 온-라인 학습을 수행하는 경우, 그리고 학습이 완료된 경우에 대한 성능 지수를 비교하였다. 네트워크의 전체 트래픽은 80%이며, 각 우선순위들이 모두 20%씩 차지하고 있는 것으로 가정하였다. 이는 상위 우선순위에 대한 요구 조건을 만족시키기 어렵게 하여, NFPM의 학습 능력을 보다 효과적으로 입증하기 위한 것이다.

본 실험에서 학습률은 1로 고정되었는데, 이는 비교적 적은 반복 횟수를 통해서도 확연한 학습 정도를 보이기 위해 큰 값을 취한 것이다. Table 2는 네트워크 시뮬레이션에 사용된 조건들을 나타내며, Table 3에서 실험의 초기 조건을 보이고 있다.

Table 2. Simulation Parameters

	THT	TRT4	TRT2	TRT0
No. of Station	10			
No. of Queue per Station	4			
Type of Probability Distribution for Message Generation Interval	uniform	uniform	expon.	expon.
Average Message Generation Interval(μsec)	10^4	5×10^4	10^5	5×10^5
Type of Probability Distribution for Message Length	uniform	uniform	expon.	expon.
Average Message Length(bit)	2×10^3	10^4	2×10^4	10^5

Table 3. Initial Conditions for Simulation Experiments

		Priority 6	Priority4	Priority 2	Priority 0
초기 타이머 (μsec)		150	3500	3500	3500
성능 지수	threshold(θ_i)	8000	20000	100000	1600000
	band (b_i)	500			

Fig. 6은 성능지표의 누적 평균을 나타내고 있는데, 이는 서로 다른 종자값에 따른 네트워크의 특성이 큰 편차를 발생시켜, 시각적인 비교에 어려움을 주기 때문에 취해진 것이다. Fig. 6에서 볼 수 있듯이, 학습하기 전의 NFPM은 네트워크의 성능을 향상시키기는 하지만, 그 속도가 느리고, 성능의 향상에도 한계를 안고 있다. 그러나, NFPM의 온-라인 학습이 진행됨에 따라, 성능이 크게 향상됨을 볼 수 있으며, 그 속도 또한 기존의 것보다 향상되었음을 알 수 있다. 학습이 완료된 파라미터로 NFPM이 성능관리를 수행할 경우, 속도가 가장 빠름을 알 수 있다. 그러나, 온-라인 학습이 50번밖에 수행되지 않았기 때문에, 모든 파라미터에 대해 학습이 이루어지

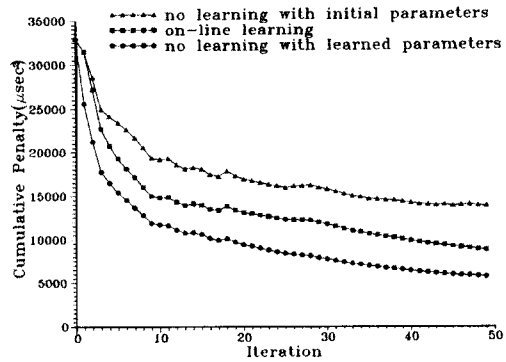

Fig. 6 Comparison of Performance Using NFPM

지는 못했으며, 학습 조건도 약간 편중된 경우이기 때문에, THT를 증가시키거나, TRT들을 감소시키는 규칙이 주로 적용되었다.

4.2 GNFPM의 평가

NFPM과의 비교를 위해 동일한 시뮬레이션 조건으로 수행되었으며, GNFPM의 전체적인 성능을 시뮬레이션을 통하여 검증하였다.

4.2.1 신경망의 근사 능력 평가

신경망에서 은닉층의 뉴론 개수를 결정할 수 있는 체계적인 기법이 알려져 있지 않기 때문에, 시행착오적으로 적절한 수를 결정하는 것이 일반적이다. 따라서 은닉층의 뉴론 개수가 컴퓨터 네트워크를 동조하는 신경망의 근사 능력에 미치는 영향을 평가하기 위하여, 은닉층의 뉴론 개수를 각각 100, 250, 400인 경우에 대해 학습을 수행하고, 그 정확도를 평가하였다.

신경망 학습에 사용된 학습 방법은 모멘텀 항을 갖는 일반적인 역전파 알고리즘(back-propagation algorithm)이며, 학습률(η)은 0.3이고, 모멘텀 상수(α)는 0.4로 두었다. 또한 학습 데이터는 0에서 1사이로 정규화되었으며, 학습 완료 조건은 정규화된 평균 에러가 0.0006, 즉 실제 에러가 약 16.5(sec)보다 작을 때로 하였다.

Table 4는 각 경우에 대한 평균 에러율을 나타내며, 학습된 데이터의 재생(recall)능력과 보간(interpolation) 능력을 비교 평가하였다. Table 4의 total average percent error와 maximum percent error의 결과치를 비교해 볼 때, 재생 능력은 100개의 은닉 뉴론을 갖는 경우가 가장 우수하며, 보간 능력은 400개의 은닉 뉴론을 갖는 경우가 가장 우수하다. 또한, 모든 경우에 있어서, 우선순위 6과 0에 대한 학습은 잘 이루어지는 반면에, 우선순위 4와 2에 대한 학습은 상대적으로 잘 이루어지지 않는 현상을 관찰할 수 있다. 이는 토큰버스 네트워크에서 우선순위 4와 2의 전송지연이 타이머의 변화에 매우 민감한 반응을 보이기 때문이다.

은닉 뉴론의 개수가 400개인 경우나 100개인 경우에 대해서, 재생 능력이나 보간 능력의 차이가 크게 나지 않기 때문에, 현재의 학습 데이터를 근사하는데 필요한 은닉 뉴론의 수를 100으로 선정하였다.

Table 4 Average Percent Error of Neural Networks with Different Number of Neurons in the Hidden Layer

No. of neuron in the hidden layer		avg. percent error for D6	avg. percent error for D4	avg. percent error for D2	avg. percent error for D0	total average percent error	max. percent error
100	trained data	0.025	0.042	0.042	0.021	0.032	0.305
	interpolation	0.027	0.053	0.060	0.021	0.040	1.042
250	trained data	0.025	0.041	0.042	0.021	0.032	0.373
	interpolation	0.028	0.053	0.062	0.021	0.041	1.018
400	trained data	0.027	0.039	0.043	0.021	0.033	0.330
	interpolation	0.027	0.047	0.061	0.021	0.039	0.944

4.2.2 GNFPM의 성능관리 능력 평가

GNFPM의 평가를 위해, 두 가지의 모의실험을 수행하였다. 첫 번째는 유전 알고리즘에 사용되는 각종 파라미터를 적절히 선정하기 위하여 동일한 시뮬레이션을 반복 수행한 경우이다. 여기서, GNFPM이 관찰하는 네트워크의 상태는 난수 발생기의 동일한 종자값(seed)에 의해 생성되기 때문에 각각의 관찰 주기마다 동일하게 유지된다. 두 번째 경우는 이렇게 선정된 파라미터를 이용하여 50회의 서로 다른 종자값을 사용하여 보다 현실에 가까운 시뮬레이션을 수행한 경우이다.

동일한 종자값을 이용한 모의실험을 통한 평가

유전 알고리즘을 이용하여 학습할 경우에, 유전 알고리즘 자체가 갖는 파라미터의 선택에 따라 다른 결과를 얻을 수 있다. 따라서, 유전 알고리즘의 파라미터를 적절히 선택하여야 할 필요가 있으며, 이를 위해 교배 확률(crossover probability, Pc)과 개체군 크기(population size)의 변화에 따른 결과들을 비교, 검토하였다.

본 모의실험은 교배 확률이 0.6과 0.9인 경우와 개체군 크기가 20개와 40개인 경우의 조합으로 구성된다. 모의실험에 사용된 교배확률, 개체군 크기, 적합도를 정의하는 상수들은 Table 5에 나타나 있고, 이 밖에 초기 타이머값, threshold, 그리고 통신망 시뮬레이션 조건은 Table 2, 3에 나타나 있다. 네 경우에 모두 돌연변이 확률은 0.0001로 고정하였다.

Table 5 GA Parameters for Simulation Experiments Using GNFPM

parameters / experiments	P_c	population size	K_6	K_4	K_2	K_0	K_T
EXP1	0.6	20	10^9	100	10	0.1	0.001
EXP2	0.6	40					
EXP3	0.9	20					
EXP4	0.9	40					

또한 스트링을 초기화할 때, 1개의 스트링은 이전에 가장 우수한 적합도를 얻은 스트링을 사용하였고, 9개는 스트링의 비트(bit)중 30%가 변할 수 있도록 돌연변이 연산을 수행하여 만들었다. 그리고, 남은 30개(EXP2와 EXP4) 또는 10개(EXP1과 EXP3)의 초기 스트링은 랜덤값으로 초기화하였다.

Fig. 7의 (a), (b)는 GA 최적화기의 네 가지 조건에 대한 모의실험 결과와 gradient method를 사용한 경우의 결과를 비교하고 있다. Fig. 7에서 볼 수 있는 바와 같이, GA 최적화기로 on-line 학습하는 GNFPM의 성능이 gradient method에 의해 온-라인 학습하는 NFPM의 경우보다 비교적 우수한 성능을 보인다. 이는 유전 알고리즘이 보다 넓은 영역을 빠르게 탐색할 수 있는 능력을 가지고 있기 때문이라고 추정된다.

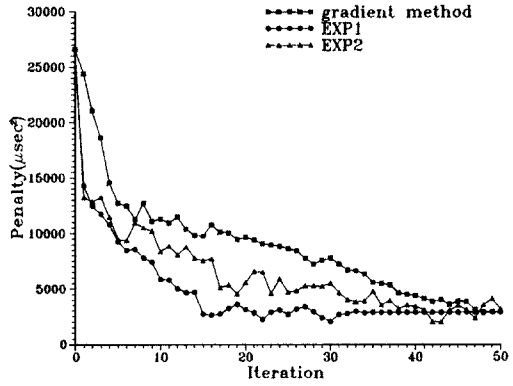

Fig. 7 (a)Comparison of Performance of GNFPM and NFPM with Different GA Parameters (Experiment 1 and Experiment 2)

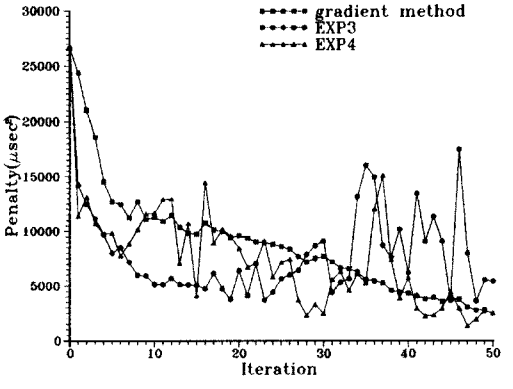

Fig. 7 (b)Comparison of Performance of GNFPM and NFPM with Different GA Parameters (Experiment 3 and Experiment 4)

Fig. 7(a)와 같이 교배 확률이 0.6일 때는 사용자의 스트링을 보존하면서 유전 연산을 수행하기 때문에 비교적 학습이 잘 이루어졌다. 여기서, EXP1은 20개의 개체군으로, EXP2는 40개의 개체군으로 모의실험을 수행하였는데, EXP1은 사용자의 정보에 지나치게 의존하여 학습을 수행한 경향이 있다. 즉, EXP1은 타이머를 너무 빨리 변화시켜 35번째 이후부터는 타이머의 제한치에 도달하여 타이머를 변화시키지 못하고 있다. EXP2는 약간의 교란이 있기는 하지만 만족할 만한 성능을 보이며, 지속적으로 성능을 향상시키고 있는 것으로 보아 EXP1보다 더 나은 학습 능력을 제공할 수 있는 가능성을 보이고 있다.

이와는 대조적으로 Fig.7(b)의 EXP3과 EXP4의 경우에는 gradient method에 비해 성능의 변화가 크고, 시스템이 불안정해 질 수 있는 가능성을 보여준다. 이 경우는 교배 확률을 0.9로 두었기 때문에 EXP1이나 EXP2에 비해 많은 교배 연산을 수행하고, 이로 인해 일정한 지점에 머무르지 못하고 지속적으로 탐색 공간을 이동함으로써 성능이 오히려 나빠지는 경향을 보이고 있다. 또한, 교배 확률이 높을 때에는 관리자가 제공한 초기의 스트링을 잘 보존할 수 없다는 것도 성능 저하의 한 요인이 될 수 있을 것이다.

이상의 동일한 종자값을 이용한 모의실험에 근거하여 교배 확률로는 0.6, 개체군의 크기로는 40을 선택하였고, 이를 바탕으로 50개의 종자값을 이용하여 모의 실험을 수행하였다.

상이한 종자값을 이용한 모의실험을 통한 평가

Fig. 8은 50개의 서로 다른 종자값을 사용하여 통신망 시뮬레이션을 수행하면서, 성능지표를 관찰한 결과이다. Fig. 8은 성능지표의 누적 평균을 나타내고 있는 데,

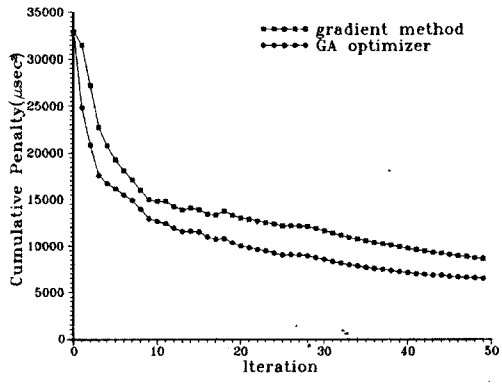

Fig. 8 Comparison of Cumulative Performance of GNFPM and NFPM with Multiple Seeds

GNFPM의 온-라인 학습 능력이 NFPM의 온-라인 학습 능력보다 우수함을 알 수 있다. 이와 같은 결과는 GA 기법이 여러 스트링을 사용하여 동시에 파라메타 공간을 탐색하며 NFPM에서 조절하지 못하는 조건부의 소속함수들을 조절하기 때문이다.

5. 결 론

본 논문에서는 공장자동화용 표준 프로토콜인 MAP의 매체 접속 제어(MAC)로 선정된 IEEE 802.4 토큰버스 네트워크를 위한 성능관리기를 퍼지논리, 신경망이론, 그리고 유전자 알고리즘의 원리를 이용하여 개발하였다. 개발된 성능관리기들의 효용성을 시뮬레이션을 통해 검증하였으며, 그 결과 다음과 같은 결론을 얻을 수 있었다.

1) NFPM과 GNFPM은 인공지능 기법에 의하여 네트워크의 실시간 성능관리를 효과적으로 수행할 수 있었다. 이러한 성능관리기들은 네트워크 프로토콜의 어떠한 개정도 필요로 하지 않으며, 메세지의 발생 간격과 메세지의 길이에 대한 확률 분포와 같은 네트워크 트래픽에 관한 통계적인 정보 없이도 그 기능을 수행할 수 있었다. 또한, Perturbation Analysis(PA), Stochastic Approximation(SA), Learning Automata(LA)와 같은 복잡한 기법을 사용하지 않고도 그에 상응하는 효과를 얻을 수 있었다.

2) NFPM과 GNFPM은 퍼지 집합들의 소속함수를 시행착오를 통하여 조절할 필요없이 신경망이론과 유전자 알고리즘을 이용하여 자동적으로 조절하는 기능을 효율적으로 수행함을 보였다.

3) GNFPM에서 사용된 신경망은 학습 데이터를 바탕으로 컴퓨터 네트워크를 동조할 수 있었으며, 학습되지 않은 영역에 대해서도 적절히 보간값을 제시할 수 있었다.

4) 시뮬레이션을 이용한 평가에 따르면 적절하게 조절된 GA 최적화기를 통해 학습한 GNFPM은 gradient method에 의해 학습하는 NFPM보다 우수한 성능을 보였다.

참 고 문 헌

1. A. Ray, "Networking for Computer-Integrated Manufacturing," IEEE Network, Vol.2, No.3, pp.40-47, 1988.
2. A. Valenzano, et al., MAP and TOP Communications Standards and Applications, Addison-Wesley, 1992.
3. 홍승호, "MAP : 공장자동화를 위한 네트워크의 표준," 대한기계학회, 제35권, 제5호, pp.427-441, 1995.
4. IEEE Computer Society,"Information processing systems-local area networks-Part4," IEEE Inc., 1990.
5. S.M. Klerer, "The OSI Management Architecture:An Overview," IEEE Network, Vol.2, No.2, pp.20-29, 1988.
6. Suk Lee and Asok Ray,"Performance Management of Multiple Access Communication Networks," IEEE Jour.on SELECTED AREAS IN COMMUNICATION, Vol.11, No.9, pp.1426-1437, 1993.
7. 이상호, 손준우, 이석, "토큰버스 프로토콜의 우선순위 시간 할당에 관한 Fuzzy Algorithm의 개발," '94 한국자동제어 학술회의 논문집(I), pp.547-552, Oct. 1994.
8. S.H. Lee, J.W. Son and S. Lee, "Fuzzy Performance Management of IEEE 802.4 Token Bus Networks," American Control Conference, pp.3254-3258, June 1995.
9. 이상호, 이석, "컴퓨터 통합생산을 위한 토큰버스 네트워크의 성능관리," 한국정밀공학회 논문집 제13권, 제6호, 1996년 6월.
10. C.T. Lin, Neural Fuzzy Control Systems with Structure and Parameter Learning, World Scientific Pub., 1994.
11. 손준우, 이상호, 이석, "뉴로-퍼지 추론 알고리즘에 의한 토큰버스 네트워크의 성능관리," '95 한국자동제어 학술회의 논문집, pp.1101-1104, Oct. 1995.
12. J.W. Kim, Y.K. Moon and B.P. Zeigler, "Designing Fuzzy Net Controllers Using Genetic Algorithms," IEEE Control Systems, pp.66-72, June 1995.
13. R.J.T. Morris and B. Samadi, "Neural Network Control of Communications Systems," IEEE Trans. on Neural Networks, Vol.5, No.4, pp.639-650, 1994.
14. D.E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, 1989.