韓國컴퓨터情報學會 論文誌
第14卷 第12號, 2009. 12.

2009-14-12-3-3

시스템 상태의 대수적 성질을 이용한 다중모드 네트워크 시스템 성능 근사계산 알고리즘

오 대 호*

An Algorithm For Approximating The Performance of Multi-mode Network System Using Algebraic Property of System States

Dae-Ho Oh*

요 약

본 논문에서는 다중모드를 갖는 장치 구성요소들의 네트워크 구조를 갖는 시스템 혹은 통신 네트워크에서 성능 품질 혹은 신뢰도 근사계산을 위해서 네트워크 시스템 상태들의 확률크기에 관해 최고 가능 상태들만을 생성하는 알고리즘을 제안한다. 대부분의 네트워크 시스템 같은 복잡한 시스템들은 장치들의 수나 장치들의 다중 모드의 수가 증가함에 따른 기하급수적 상태 공간의 증가로 인해 그 성능이나 신뢰도를 직접 계산하는 것이 매우 어려운 경우가 많다. 따라서 근사계산의 해법이 더욱 타당한 접근법이라 할 수 있다. 이때, 시스템의 기대 성능이나 신뢰도 계산을 위해서는 시스템 상태와 그 상태 확률을 통해 기댓값을 산출하여야 한다. 이때, 근사계산을 통한 접근 방법에서 사전에 수행되어야 하는 것은 네트워크 시스템 상태들의 발생 가능 순서로 나열해야 하는 것이 필요하다. 이에 본 논문에서는 시스템 상태들 중 가장 발생 가능성이 큰 상태들을 찾고 이로부터 네트워크 성능이나 신뢰도의 근사값을 구하는데 활용할 수 있는 방법을 제안한다. 제안된 알고리즘은 기존의 방법과 자원의 효율성 측면 중에서 메모리 효율성을 고려하여 실험을 통해 예시하고 장·단점을 논의하고자 한다.

Abstract

A practical algorithm of generating most probable states in decreasing order of probability of the network system state is suggested for approximating the performance of multi-mode network system using algebraic structure of the system states. Most complex system having network structure with multi-mode unit state is difficult to evaluate the performance or reliability due to exponentially increasing size of state space. Hence not an exact computing method but an approximated one is reasonable approach to solve the problem. To achieve the goal we should enumerate the network system states in order as a pre-processing step. In this paper, we suggest

• 제1저자 : 오대호
• 투고일 : 2009. 12. 04, 심사일 : 2009. 12. 15, 게재확정일 : 2009. 12. 24.
* 한림성심대학 의료기기정보과 교수

an improved algorithm of generating most probable multi-mode states to get the ordered system states efficiently. The method is compared with the previous algorithms in respective to memory requirement and empirical computing time. From the experiment proposed method has some advantages with regard to the criterion of algorithm performance. We investigate the advantages and disadvantage by illustrating experiment examples.

▸ Keyword :다중모드(multi-mode), 최고 가능 상태(most probable state), 네트워크 성능 및 신뢰도(network performance and reliability), 대수적 성질(algebraic property), 알고리즘(algorithm)

Ⅰ. 서 론

통신망, 다중 프로세서의 상호연결망등과 같은 복잡한 시스템들은 주로 네트워크 구조를 갖는다[1][2]. 네트워크 시스템의 성능 품질이나 신뢰도는 네트워크 시스템의 유효성을 평가하는데 있어 중요한 요소이다. 특정 네트워크 신뢰성(혹은 성능)측도가 선택되고 각 상태 확률이 주어지면 네트워크 신뢰도는 각 상태에 대한 신뢰성의 기대 값으로 계산된다[8]. 그러나 정확한 시스템 신뢰도의 계산은 일반적으로 매우 복잡하며, 주로 수리적으로 다루기 힘든 경우가 대부분이다. 특히, 시스템 구성 장치들의 모드 수가 증가할수록 네트워크 시스템의 상태 공간(state space)의 크기가 지수적(exponentially)으로 증가하기 때문에 모든 상태를 나열함으로써 평가하는 경우 많은 계산 시간이 요구된다[8][9]. 따라서 정확한 신뢰도를 계산하는 것보다 그 근사 값을 도출하는 방법들이 요구된다. 네트워크 구성요소들의 작동 확률(혹은 신뢰도)는 일반적으로 매우 크며, 따라서 적은 수의 상태들만으로도 전체 상태 확률 중에서 그 비율은 상당히 높다[14][15][16]. 이에, 네트워크 시스템의 작동 환경에서 발생 가능성(혹은 확률)이 낮은 시스템 상태들을 계산에서 제외하고, 그 중에서 발생 가능성이 큰 시스템 상태 들 만을 통해 그 상 • 하계나 근사 값 등을 계산하는 것은 매우 타당한 근사 해법이 될 수 있다. 일반적으로 네트워크 장치들이 고 신뢰도 일 때 상태공간의 10% 이내의 상태들만으로도 약 95%이상의 상태 공간 포함 정도(coverage)를 만족한다고 알려져 있다[3][9]. 이러한 관점에서, Li 와 Silvester[4]는 망 유닛들이 이진 상태일 때 망 신뢰도의 근사 값을 계산하기 위하여 발생 가능성이 큰 상태(most probable state, 이하 MPS)의 개념을 도입하였으며 이들 상태들을 생성하는 알고리즘을 제시하고, 특정 네트워크 신뢰도 측도에 대해서 MPS에 의해 상•하계를 계산함으로써 네트워크 신뢰도(성능 신뢰도)의 근사 값으로서 유용한 방법임을 보였다. 그러나 네트워크 구조를 갖는 복잡한 시스템들은 그 장치들이 이진 상태로서 작동이거나 고장인 상태가 되기보다는 점차적으로 그 성능의 수준이 여러 단계로 감소하는 형태로 작동하는 경우가 많다. 품질이나 신뢰성의 단계가 아니라 통신 매체가 음성이거나 혹은 문자의 전송일수 있는 것과 같이, 여러 작동 모드를 갖는 경우도 있다[11][14].

이에 본 논문에서는 다중모드의 장치들로 구성된 네트워크 시스템에 대해서 최대 가능 상태를 그 상태확률의 크기에 따라 내림차순으로 생성하는 개선된 방법을 제안하고 실험 예제를 통해 알고리즘의 자원 효율성 측면 중 매모리 소요량에 관하여 비교하며 그 장단점을 예시 하고자 한다. 2장에서는 다중 모드 시스템 신뢰도 모형과 기존의 방법을 논의하고, 3장의 1,2절에서는 본 논문에서 제안하는 알고리즘을 기술하고자 하며, 3장의 3절에서 실험을 통해 그 효율성을 검토하고자 한다.

Ⅱ. 관련 연구

1. 다중 모드 네트워크 시스템 신뢰도 모형

Chiou 와 Li[5]는 시스템 장치들이 다중 모드의 구성요소로 구성된 네트워크의 신뢰도 평가에 있어서, Li 와 Silvester의 이진 모드의 알고리즘을 수정하여, MPS의 생성 알고리즘을 처음 제안하고 성능 측도로는 메시지 지연에 대하여 망 신뢰도에 대한 상•하계를 예시하였다. 또한, Gaebler와 Chen[6] 그리고 Shier[7]등이 더욱 개선된 방법들을 제안하였다. 그러나 이후 이와 관련한 문제에 대해 개선 된 방법보다는 성능 평가 측도의 문제로 많은 문헌들이 발표되어 왔으며 이러한 문제에 관하여 많은 연구가 이루어 지지 않았다. 이에 본 논문에서는 MPS의 생성 알고리즘의 기존 방법을 재검토하고 개선된 방법을 제안하고자 한다. 이를 위해 기존 방법과 공통적인 기본 적인 가정은 다음과 같다.

가정

(1) 네트워크 장치는 노드 혹은 링크가 될 수 있다.

(2) 장치 i 는 D_i 가지 상태(모드) 중 하나의 상태에서 작동될 수 있으며 그 작동 상태는$0,1,\cdots,D_i-1$ 이다.

(3) 장치i의 상태j는 주어진 확률 $p_{ij}>0$, $i=1,\cdots,n$, $j=0,\cdots,D_i-1$에 관하여$p_{i0}\geq p_{i1}\geq\cdots\geq p_{iD_i-1}$, $\sum_{j=0}^{D_i-1}p_{ij}=1$, 으로 작동하며 시스템의 다른 장치에 관하여 독립적으로 작동한다. (단, $p_{i0}\geq 0.5$)

위 가정과 함께 네트워크 시스템의 상태공간의 크기가 매우 큼으로 어떤 종료 기준(stopping rule)을 만족할 만한 수준에서 전체 시스템 상태보다는 발생 가능성이 큰 혹은 확률이 큰 시스템 상태들만을 고려하여 신뢰도의 근사 값이나 그 상·하 한계를 계산 하는 것은 타당한 접근 방법이다 할 것이다. 네트워크 시스템 장치들의 상태별 작동 확률이 주어지고 특정 신뢰도 혹은 성능에 대한 측도가 주어졌을 때 이를 평가하기 위해서는 상태 확률의 내림차순으로 네트워크 상태 $S_1,S_2,\cdots,$들을 특정한 종료기준을 만족할 때 까지 각 확률의 계산을 줄이거나 그러한 과정을 생략하여 효율적으로 생성하는 것이 요구된다. 적절한 m개의 MPS를 얻는다면, 네트워크 시스템의 신뢰도의 상한계와 하한계는 다음과 같이 각각 근사적으로 도출할 수 있다[8][12].

$$\overline{R_U}=\sum_{k=1}^{m}R(S_k)P(S_k)+(1-\sum_{k=1}^{m}P(S_k))\alpha$$

$$\overline{R_L}=\sum_{k=1}^{m}R(S_k)P(S_k)+(1-\sum_{k=1}^{m}P(S_k))\beta$$

이때, m은 최고 가능 상태의 수 이며, $R(\cdot)$은 시스템 상태 S_k에서의 성능 혹은 신뢰도 측도이며, α와 β는 각각 모든 상태에서의 성능 측도의 최대값, 최소값이다. $\overline{R_U}$, $\overline{R_L}$은 따라서 시스템 기대 성능 혹은 신뢰도의 상한계와 하한계가 된다.

네트워크의 장치의 수는 n이고 각 장치$i\in\{1,\cdots,n\}$의 작동 상태에서 장치 j의 모드 혹은 상태, x_i는 편의상 0의 상태 값으로 시작하여$0,1,\cdots,D_i-1$을 갖는 경우 장치 i의 작동 상태 확률은 위 가정의 (3)과 같다 하자. 보통의 네트워크 장치들의 상태가 다중 모드일 경우 각 장치의 최고 성능인(혹은 일반적 작동 모드)상태를 0으로 표현하기로 하며 이때의 작동할 확률 혹은 신뢰도는 가장 큰 것이 일반적이다.

이때, 원(original) 네트워크 시스템의 상태는 n벡터 형태로 $X=(x_1,\cdots,x_n)$과 같이 표현된다. 이때, x_k는 각 장치 k의 상태 값이다. 위에서 언급한 상·하한계나 근사 값을 도출하기 위해서 모든 시스템 상태의 확률을 계산하고 단지 나열하고 생성하면 상태공간의 기하급수적 증가로 많은 계산 자원을 소모하게 된다. 따라서 모든 상태의 확률을 계산하지 않고도 효율적인 근사 값을 계산하기 위한 사전 처리가 요구된다. 이때, 각 장치의 작동 혹은 성능 확률이 가장 큰 것과 그 다음 크기의 확률의 비율을 각 시스템 장치 i에 관하여 $w_i=\frac{p_{i1}}{p_{i0}}$와 같이 정의 한다. 이러한 정의와 함께 w_i의 크기에 따라서 모든 시스템의 장치$i\in\{1,\cdots,n\}$를 다음 식(1)을 만족하도록 재 순서화 하도록 한다.

장치$i<j$에 대해서

$$w_i\geq w_j, i,j=1,\cdots,n \quad\cdots\cdots(1)$$

이러한 재 순서화(re-ordering)방법은 각 장치들의 작동 상태에서 상태 0을 제외한 것 중 두 번째 작동확률 크기의 작동 상태 1의 상태 확률만을 기준으로 모든 네트워크 장치들을 다시 레이블링(labeling) 한 것이 된다. 이러한 재 순서화 후, 장치 i는 식(1)에 의한 재 순서화된 장치 인덱스를 나타내는 것으로 재 정의한다. 이때, 네트워크 시스템의 각 상태들을 그 확률 크기의 내림차순으로 찾는 것이 목적이다. 식(1)에 의해서 레이블링 될 때, 원 네트워크 상태,$X=(x_1,\cdots,x_n)$는 $S_i=(y_1,y_2,\cdots,y_n)$(단,$i=1,\cdots,\prod_{i=1}^{n}D_i$)로 재 표현할 때, y_k는 순서화된 장치 k의 상태값, 다음을 만족하는 내림차순으로 순서화된 네트워크 시스템 상태로 정의된다. 즉,

$$P(S_1)\geq P(S_2)\geq\cdots\geq P(S_{\prod_{i=1}^{n}D_i}).$$

이러한 재 순서화에 의해서 가장 가능성이 높은 즉, 확률이 가장 큰 네트워크 상태는 각 장치 i의 작동 상태 중 확률, $\Pr\{y_i=0\}$가 가장 크므로$y_1=y_2=\cdots=y_n=0$인 경우로서 $S_1=(0,\cdots,0)$이고 다음으로 작동 확률이 큰 시스템 상태는 p_{i1}/p_{i0}가 가장 큰 장치를 1로 레이블링을 한 것이므로$S_2=(1,0,\cdots,0)$이며 가장 확률이 작은 시스템 상태는 각 장치의 작동 상태의 확률이 가장 작은$y_i=D_i-1$, $i=1,\cdots,n$인 경우의 시스템 상태인 ,

$S_{\prod_{i=1}^{n}D_i}=(D_1-1,D_2-1,\cdots,D_n-1)$이 된다. 이러한 재 순서화와 상태 표현에 의해서 각 네트워크 상태 S

$=(y_1,\cdots,y_n)$의 상태 확률 $P(S)$는, $P(S)=\prod_{i=1}^{n}p_{ij}$ $j=0,\cdots,D_i-1$이며 만일 S의 i 번째 장치의 상태가 j에서 k로 변경될 때 시스템 상태, $S'=(y_1,\cdots,y_{i-1},k,y_{i+1},\cdots,y_n)$의 확률은, $p(S')=p(S)\cdot(p_{ik}/p_{ij})$으로 쉽게 재계산될 수 있다.

위에서 기술한 가정과 재 레이블링 방법과 함께 Gaebler 외의 알고리즘 개요는 다음과 같다.

$S(i)$를 시스템상태 S의 i번째 장치의 작동 상태라 하고 최근 반복 단계에서 얻은 MPS가 $S_c=(y_1\cdots y_n)$이며 $h=\max(i|S(i)\neq 0)$, $j=S(h)$라 정의 되어

첫째, 만일 $j<D_h-1$ 이면 h번째 원소를 1 증가하여 생성 하고, 둘째, $h<n$이면 $h+1$번째 원소에 1을 증가하여 생성하며, 셋째, $h<n$이고 $j=1$ 이면 h번째 원소를 0으로 취하고 $h+1$번째 원소를 1로 대체하여 생성 하는 절차를 취한다. 위와 같은 방법에 대하여 Shier는 다소 개선된 방법을 제안하였으며 그 알고리즘의 개요는 다음과 같다.

$f=\min(i|S(i)\neq 0)$, $j=S(f)$라 정의 되어

첫째, $j<D_f-1$ 이면 f 번째 장치를 1 증가하여 생성하고, 둘째, $f<n, j=1$ 이고 $f+1$번째 장치의 상태가 0 이면 $f+1$번째 원소를 1, f 번째를 0으로 대체하여 생성 하며, 셋째, $f>1$ 이면 첫 번째 장치의 상태를 1로 대체하여 생성한다.

본 논문에서는 위에서 언급한 레이블링 절차와 함께 최고 가능 상태들을 효율적으로 생성하여 메모리 자원을 효율적으로 활용하고 궁극적으로 네트워크 시스템의 성능 혹은 신뢰도의 근사 값을 계산하는데 활용하기 위한 개선된 방법에 관하여 논의하고자 하고자 며 3장에서는 제안된 방법에 대해 기술하고자 한다. 또한, 위의 기존의 방법과 효율성에 관하여 실험예제를 통해 비교 검토하고자 한다.

III. 본 론

1. 다중 모드 시스템 상태의 대수적 성질

본 장에서는 2장에서 논의한 기존의 방법 및 가정과 함께 기존의 방법 중 시스템 상태의 재 순서화 방법은 그대로 따르되 시스템의 상태들을 상호 대수적인 연관성 및 규칙성을 고려하여 개선된 방법을 논의하고자 한다.

네트워크 시스템 장치들을 벡터 형태로 표현한 결과는 네트워크 장치들의 모든 상태들이 세 가지를 갖고 세 가지 장치로 구성된 경우를 검토하여 보자. 이때의 모든 상태들의 관계는 다음 그림 1과 같이 표현된다. 그림 1에서 각 원의 내부에는 네트워크 시스템의 상태 벡터를 표기 하였으며 각 각의 화살표는 각 네트워크 상태 확률에 관한 대수적 크기 관계를 표시한 것이다. 또한, 네트워크 상태 확률들 중에서 직접 확률 계산을 통하지 않고서도 서로 대수적으로 대소 관계를 알 수 있는 경우의 상태들을 화살표로 연결하였다. 화살표 연결이 없는 경우는 상호간 확률을 직접 계산하여 비교한 후에야 대소 관계가 식별이 되는 것을 의미한다. 이때의 시스템 전체의 상태 공간 Ω 의 크기는 27이며, 시스템 상태 중에서 확률의 계산 수행이 없이도 네트워크 작동 상태(0,0,0), (1,0,0)은 장치들의 작동 상태 확률이 큰 네트워크 상태들이다.

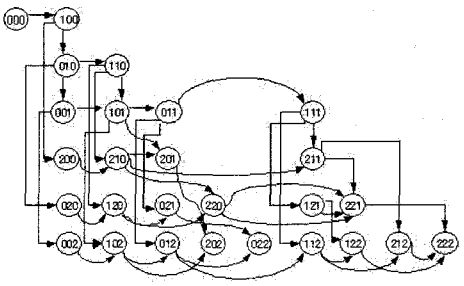

그림 1. 네트워크 시스템 상태의 확률 크기 관계
($n=3, D_i=D=3,\ i=1,2,3$)
Fig. 1. Relation of the Network System State's Probability Size

또한, (1,0,0) 다음으로 확률이 큰 상태는 (0,1,0) 혹은 (2,0,0)이 될 수 있으며 이 크기의 결정은 확률의 계산 없이는 알 수가 없다. (1,0,0), (0,1,0), (0,0,1)와 같이 장치 작동 상태가 1 인 시스템 상태들은 2장에서 논의한 식(1)에 의해 재 순서화한 결과로부터 알 수 있으나 (2,0,0), (0,2,0), (0,0,2) 등의 시스템 상태들은 서로 간의 대소 관계를 확률 계산 없이는 대수적으로 알 수가 없다. 이러한 결과는 네트워크 장치들의 수, $n\geq 3$인 경우에도 성립하며 장치들의 수가 증가한다면 매우 증가하게 된다. 시스템상태들 간의 대소 관계는 작동 상태 가 $y_i=0,1$로 구성된 시스템 상태들인 경우 확률 크기가 대수적으로 식별이 가능한 경우가 많으나 상태 수준이 2 이상을 포함하는 경우에는 대수적으로 그 크기가 식별되는 것이 적은 비율을 갖기 때문에 그 식별 여부가 더욱 어렵게 된다. 따라서 각 장치들의 상태 수, D_i가 증가할수록 대소 관계가 확률 계산 없이 비교 가능하지 않는(non-comparable) 네트워크 상태들이 전체 네트워크 상태들 중 그 비율이 더욱 증가한다. 이러한 네트워크 상태들 중에서 식별이 용이하며 가장 확률이

큰 상태들인 $S_1=(0,0,0)$, $S_2=(1,0,0)$으로부터 그 다음으로 확률이 클 수 있는 후보 상태를 효율적으로 확률의 계산 없이 정확히 생성하고 가장 큰 확률의 상태를 찾는 것이 중요한 문제가 된다. 따라서 $S_2=(1,0,0)$을 기준으로 하여 확률의 계산 없이 알 수 있는 확률의 대소 관계로부터 다음으로 확률이 클 것으로 예상되는 상태를 후보 상태로서 생성하고 이 후보상태들만의 확률의 계산한 후 그 다음으로 큰 상태, S_3를 얻어 다시 반복적인 절차를 통해서 모든 상태를 생성하는 효율적 방법을 통해 실행 시간의 단축과 기억 자원 낭비를 줄이고자 한다. 이러한 목적을 위에서는, 대수적 관계를 효과적으로 이용하여 MPS 후보 상태를 적게 생성하고도 정확히 MPS를 생성하는 알고리즘이 요구된다 할 수 있다. 따라서 후보 상태들을 생성하는 방법에 있어서 본 논문에서는 가능하면 장치 상태의 수준 0, 1이 네트워크 상태에 포함되어 있을 때 그 대수적 관계를 최대한 이용하고 2 이상인 것들을 포함하는 시스템 상태는 그 크기가 직접 비교 가능한 상태들만을 후보 상태로서 생성하는 방식을 취하고자 한다. 그림 1의 대수적 크기 관계의 관찰로부터 전체 네트워크 상태들을 다음 표 1과 같은 시스템 상태의 분할을 고려하자.

표 1은 그림 1의 네트워크 시스템의 상태들 마지막 장치 번호가 0이 아닌 것 중 1인 상태벡터들을 한 쌍의 우측에 그렇지 않은 것 들을 좌측에 하나의 쌍으로 재구성한 것이다. 즉, (1,0,0)과 (2,0,0) 그리고 (0,1,0)과 (0,2,0)등과 같이 대응 되어 나타낸 것이다. 제 1행은 항상 가장 상태 확률이 큰 (0,0,0)의 경우이고 나머지 상태들을 제 2행에 쌍으로 나타내었다. 제2행의 각각의 시스템 상태벡터들은 우측의 것 들이 항상 작은 확률의 시스템 상태들로 분할된 것임 알 수 있다.

표 1. 네트워크 시스템 상태 분할
Table 1. Network System State Partition

구분	시스템 상태
1	(0,0,0)
2	(1,0,0)-(2,0,0),(0,1,0)-(0,2,0),(0,0,1)-(0,0,2) (1,1,0)-(1,2,0),(1,0,1)-(1,0,2),(0,1,1)-(0,1,2) (1,1,1)-(1,1,2),(0,2,1)-(0,2,2),(2,0,1)-(2,0,2) (2,1,0)-(2,2,0),(1,2,1)-(1,2,2),(2,1,1)-(2,1,2) (2,2,1)-(2,2,2)

$S(h)$를 시스템 상태 S의 h번째 장치의 작동 상태라 하고 h는 임의 네트워크 상태 $S=(y_1,\cdots,y_n)$에서 장치 상태가 0이 아닌 가장 마지막 장치 인덱스 즉,

$h=\max(i|S(i)\neq 0)$이라 정의하도록 하자. 예로써, $S=(0,2,2,0,0)$이면 $h=3$이며, $S(h)=2$가 된다. 표 1에서 가장 확률이 큰 네트워크 상태(0,0,0)을 제외하고 나머지 시스템 상태들은 $S(h)=1$, $h=1,\cdots,n$을 만족하는 시스템 상태와 이보다 확률 계산 없이 항상 대수적으로 그 크기가 작은(algebraically comparable) $S(h)=2$등의 상태들의 집합으로 구성될 수 있음을 알 수 있으며 $D_i=D\geq 3$인 일반적인 경우도 일반화 될 수 있다. 이러한 대수적 크기 관계와 표 1의 상태들의 분할에 관한 관찰로부터 다음과 같은 정리를 얻을 수 있다.

정리 1. 재 순서화된 임의의 시스템 상태 $S=(y_1,\cdots,y_n)$에 대하여, $S(h)$는 상태 S의 h번째 장치의 작동 상태라 하고 $h=\max(i|S(i)\neq 0)$일 때, 전체 네트워크 상태는 $S_1=(0,\cdots,0)$과 $S(h)=1$, $h=1,\cdots,n$을 만족하는 $1+\sum_{h=1}^{n-1}\prod_{i=1}^{h}D_i$ 가지의 네트워크 상태와 이 보다 항상 확률이 작은 대수적 비교 가능한(algebraically comparabe) D_h-2의 네트워크 상태 집합들로 구성된다.

증명: $S(h)=1$을 만족하는 것은 네트워크 시스템 상태는 $S=(y_1,\cdots,y_{h-1},1,0,0,\cdots,0)$으로 표현 될 수 있으며 가정에 의해서, $\Pr(y_1,\cdots,1,0,\cdots,0)\geq\Pr(y_1,\cdots,2,0,\cdots,0)\geqq\cdots\geq\Pr(y_1,\cdots,D_h-1,0,\cdots,0)$을 만족하는 D_h-2 상태들이 존재하며, $S(h)=1$을 만족하는 상태는, $h=1,\cdots,n$에 대하여, h가 1일 때는 1가지 경우와 h가 2인 경우는 상태벡터의 첫 원소가 될 수 있는 D_1가지 경우 그리고 h가 3인 경우는 상태벡터의 첫 번째 두 번째 원소가 될 수 있는 D_1D_2의 경우가 존재하며 따라서 $h=1,\cdots,n$에 대해서는 $1+D_1+D_1D_2+\cdots+D_1D_2D_3\cdots D_{n-1}$가지의 상태들이 존재한다.

따름정리 1. 각 작동 상태가 $D_i=D$, $i=1,\cdots,n$ 일 때 네트워크의 상태에서 $S(h)=1$을 만족하는 상태와 대수적 비교가능 상태들의 집합으로 분할되며$(D^n-1/D-1)$가지의 $S(h)=1$의 네트워크 상태들과 이와 비교 가능한 $(D-2)(D^n-1/D-1)$의 상태가 존재한다.

증명: 정리 1로부터 $h=1,\cdots,n$에 대하여 $S(h)=1$을 만족하는 상태는,

$$1+D+D^2+\cdots+D^{n-1}=\frac{D^n-1}{D-1} \quad \cdots\cdots (2)$$

의 경우가 존재하며 이와 비교 가능한 상태 집합들은 식 (2)의 좌변의 각 항에 대해서 $D-2$가지의 경우가 있으므로

성립한다. 또한, $S(h)=1$을 만족하는 상태와 이와 비교 가능한 상태들의 합은, D^n-1 가지가 이 되며 전체 시스템 상태 중 $(0,0,\cdots,0)$을 제외한 모든 상태들의 수가 된다.

위 정리와 같이 네트워크 상태들을 분할한 상태 공간을 고려하고, 임의의 최고 가능 확률 상태의 후보 상태들을 확률의 계산과정 없이 생성할 때 표 1의 2행에서 각 우측 상태들은 좌측 상태가 최근 단계의 MPS로 결정된 후의 시점에서 이 MPS로부터 차례로 생성 될 수 있다면 좌측의 $S(h)=1$을 만족하는 상태들과 이와 확률 크기가 비교 가능하지 않은 $S(h)\neq 1$의 상태들을 어떻게 효율적으로 생성하는가의 후보 생성 알고리즘을 찾는 문제로 귀결된다.

2. 최고 가능 상태 후보 생성 알고리즘

MPS의 후보 상태들의 생성 알고리즘의 전체과정은 최근 반복 단계의 MPS를 기준으로 그 다음 순위의 MPS가 될 수 있는 후보 상태를 생성하여 힙 구조에 삽입하고 이러한 후보 상태들로부터 가장 확률이 큰 네트워크 상태를 힙 추출하며 이때 찾은 MPS를 기준으로 그 다음의 후보 상태들을 힙 삽입하는 절차로 구성된다. 이러한 과정은 $n=3, D_i=D=3$ 일 때 그림 2와 같이 트리 구조로 표현될 수 있다. 이때, $S_1=(0,0,0)$를 제외한 $S_2=(1,0,0)$을 초기 MPS로 취할 수 있으며 S_2를 부모 상태로 하고 트리 구조에서 자식상태를 생성하는 형태가 된다. 따라서 자식 상태들을 어떠한 방법으로 생성할 것인가가 관건이 된다. 후보 상태 생성 방법 및 알고리즘의 기술을 위해 필요한 몇 가지 함수 들을 다음과 같이 정의한다.

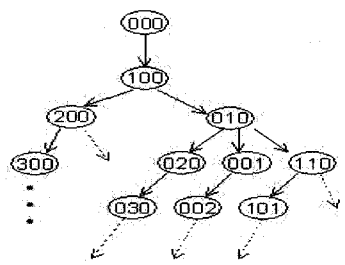

그림 2. 다중모드의 MPS 및 후보 상태 생성 과정
Fig. 2. Generation Process of Multi-mode MPS and the Candidate States

임의의 네트워크 상태 $S=(y_1,\cdots,y_n)$에 대해서

(1) $g_1(k)$는 S의 k번째 장치의 작동 상태 수준을 1 증가하는 함수,

(2) $g_2(k)$는 k번째 장치의 작동 상태 수준을 0, $k+1$ 번째 장치 상태 수준을 1로 변경하는 함수,

(3) $g_3(k)$는 k번째 장치 상태를 1로 취하는 함수,

(4) $c(k)$는 네트워크 상태 S의 $k-1$번째 장치로부터 장치 1까지의 역방향(backward)으로 장치 상태 수준이 1인 횟수 이다. 이러한 함수와 함께 초기 MPS로부터 후보 상태들을 생성 규칙은 다음 정의 1과 같다.

정의 1. 임의 네트워크 상태 $S=(y_1,y_2,\cdots,y_n)$에 관하여, $S(h)$는 상태 S의 h번째 장치의 작동 상태이며 $s=S(h)$ 이고 $h=\max(i\mid S(i)\neq 0)$일 때,

(1) $s<D_h-1$ 이면, $g_1(h)$는 S의 자식 상태,

(2) $s=1, h<n$이면, $g_2(h)$는 S의 자식 상태,

(3) $c(h)\geq 1$ 이면, $g_1(h-j), j=1,\cdots,c(h)$는 S의 자식 상태,

(4) $S(h-(c(h)+1))=0$이고, $S(h-(c(h)+2))=1$이면, $g_2(h-(c(h)+2))$는 S의 자식 상태,

(5) $S(h-(c(h)+1))=0$, $h-c(h)=2$ 이면 $g_3(1)$은 S의 자식 상태이다.

1절의 그림1과 유사한 시스템인 경우 Gaebler외, Shier 외 그리고 이 절의 마지막 부분에서 전개되는 알고리즘의 기본 규칙인 정의 1에 의한 최고 가능 후보 상태들의 생성과 MPS 결정에 관한 검토를 위해 다음 표 2와 같은 시스템에 관하여 논의하여 보자.

표 2. 상태, 확률, 확률비, 재순서 순위
($n=3$, $D_i=3$, $i=1,2,3$)
Table 2. State, Probability, Ratio, Reordered Rank

		상태 j			w_i (p_{i1}/p_{i0})	순위
		0	1	2		
장치 i	1	0.90	0.05	0.05	0.055	3
	2	0.80	0.15	0.05	0.188	2
	3	0.70	0.20	0.10	0.286	1

표 2에는 2장에서 논의된 확률비값에 의해 재순서 화된 순위를 나타내었다. 확률비는 소수점 3자리까지 반올림하여 표시하였다. 평가된 순위에 의해 장치 1은 3, 2는 2 그리고 3은 장치1로 되며 시스템의 상태는 재순서 화된 형태를 따르게

된다. 이러한 재순서화된 시스템의 n백터 표현과 함께 후보 상태들을 생성하는 기존의 두 가지 방법과 정의1에 의한 규칙 적용 과정을 상호 비교하여 표 3에 나타내었다.

표 3. 최고 가능 후보상태 및 MPS, S_c 생성과정
Table 3. Generation Process of Most Probable Candidate states and MPS

반복	S_c	S_c 확률	Gaebler 외	Shier외	제안된 방법 (정의1)
0	(000) (100)	0.504 0.144	-	-	-
1	(010)	0.094	(200),(110)	(200)	(200)
2	(200)	0.072	(110),(020) (011),(001)	(020), (001) (110)	(020), (001) (110)
3	(020)	0.032	(210),(110), (011),(001)	(001), (110)	(001), (110)
4	(001)	0.028	(210),(110) (021),(011)	(120), (110)	(110)

표 3에는 계산과정이 없이도 가정 시스템 상태 확률이 큰 (0,0,0)과 (1,0,0)을 초기 값으로 하여 기존의 방법의 과정을 총 4회 반복할 경우 6개의 MPS(S_c)를 찾는 과정을 표시하였다. Gaebler외의 경우 반복 1회에서 초기 상태, (1,0,0)으로부터 2장에서 논의한 첫 번째 규칙에 의해 (2,0,0)을 두 번째 규칙에 의해 (1,1,0)을 세 번째 규칙에 의해서 (0,1,0)을 후보상태들로 각각 생성하고 각 시스템 확률을 계산한 후 (0,1,0)을 확률 0.094로 가장 큰 1회째의 MPS로 결정하게 된다. Shier외의 경우는 (1,0,0)상태로부터 첫 번째 규칙에 의해 (2,0,0)을 두 번째 규칙에 의해 (0,1,0)을 생성하고 세 번째 규칙에서는 적용되지 않아 생성하지 않게 되어 결국 적은 수의 후보 상태들을 생성하게 된다. 이때, 본 논문에서 제안되는 정의 1의 규칙 적용을 보면 (1,0,0)으로부터 정의1의 (1)에 의해 (2,0,0)을 생성하고 정의1의 (2)에 의해 (0,1,0)을 생성하고 (3),(4),(5)에서는 생성되지 않아 1회째의 경우 Shier외의 방법과 같은 결과를 얻게 된다. 이러한 반복 과정의 최근 단계의 MPS로부터 다른 후보 상태들을 계속 생성하여 그 후 시스템 확률을 계산하는 과정을 갖게 되는데 반복 2, 3회의 경우 Gaebler외의 경우는 많은 후보 상태들을 생성하게 되는 결과를 낳게 된다. Shier외의 경우는 반복3회 째까지 정의 1에서의 자식상태 생성 방법과 같은 결과를 보이나 반복 3회째 얻은 MPS, (0,2,0)으로부터 규칙 3에 의해 (1,2,0)을 생성한다. 그러나 이 상태는 (1,1,0)보다 발생가능성이 작은, 대수적으로 비교가능 상태로서 (1,1,0)이 MPS로 결정된 후 생성되는 것이 더 효율적이라 할 수 있다. 제안하는 방법의 경우 정의 1의 (5)에 의해서 같은 후보 상태를 생성할 수 있으나 본 논문에서는 장치의 작동 상태가 1인 경우에 규칙 대수적 비교가능 상태를 최대한 활용하기 위해 (3), (4)에 의해서 생성되도록 이러한 경우를 회피하도록 알고리즘에 적용하도록 함으로 결국 반복 4회째와 같이 Shier외의 방법보다도 적은 후보 상태들을 생성하게 된다.

위에서 언급한 동기와 함께 본 논문에서 제안되는 MPS의 생성 알고리즘은 MPS 후보 상태의 생성 방법과 더불어 개략적으로 다음과 같은 절차로 구성된다.

단계1: 원 네트워크의 고장가능한 장치들을 2장의 식 (1)에 의해서 재 순서화 한다.

단계2: 상태 $(0,0,\cdots,0)$, $(1,0,\cdots,0)$을 초기 MPS S_1, S_2로 각각 취한다.

단계3: S_2를 현 단계의 MPS로 하고 이 절의 정의 1에 의해서 후보상태들을 생성 한다.

단계4: 단계3으로부터 생성된 후보 상태들의 상태 확률을 계산하고 상태벡터와 확률을 노드로 하여 메모리 구조인 힙에 저장 및 재 힙화(heapify)한다.

단계5: 힙으로부터 가장 확률이 큰 상태를 추출하여 취하고 다음 단계의 MPS, S_c로 한다.

단계6: 종료 기준에 따라 원하는 MPS가 선택될 때까지 단계 3 로부터 반복 한다.

정리 1과 정의 1 그리고 위의 단계 1~6의 논의한 절차로부터 MPS를 생성하는 알고리즘은 다음 그림 3과 같다.

알고리즘은 식(1)에 의해서 원 네트워크의 장치들이 레이블링 되었다 가정하며, 세 가지 유형의 후보 상태들을 생성하고 메모리인 힙에 삽입, 정렬하여 확률의 크기로 정렬된 힙으로부터 가장 확률이 큰 하나의 상태를 취하는 과정으로 구성된다.

세 유형의 후보 상태들을 생성하는 단계에 있어서는 그 특성상 나누어진 것으로 첫째, 정의 1의 (1)에 의한 후보 상태를 생성하고 둘째, 정의 1의 (2)에 의한 방법에 의해 생성하며 셋째, 정의 1의 (3),(4),(5)에 의한 후보 상태를 생성하는 과정으로 구성된다. 세 번째 유형의 후보 상태의 생성방법 및 힙으로의 삽입은 그림 4에 나타내었다.

그림3의 알고리즘은 초기화 과정에서 네트워크 상태, $(1,0,\cdots,0)$를 반복 알고리즘의 현 단계 MPS, S_c로

취하며 힙(H)에는 어떤 상태도 없도록 한다. S_c의 원소 중 0보다 큰 마지막 원소의 상태 수준 $h = \max(i|S(i)\neq 0)$는 1 및 s는 1로 각각 초기화한다. 알고리즘의 1회 반복에서 위의 세 가지 유형별로 MPS 후보 상태를 생성하고 그 다음 순위의 MPS를 얻으며 그 세부 과정 및 관련 함수는 다음과 같다.

(알고리즘 세부과정 1) 최근 단계의 MPS, S_c로부터 h번째 장치의 작동 상태가 $D_h - 1$보다 작을 때 정의 1의 (1)의 함수 $g_1(h)$함수와 $y_h \leftarrow y_h + 1$를 통해 후보 상태를 생성하여 그 확률을 계산하며 상태 벡터와 확률을 노드로 하여 힙에 삽입한다.

(알고리즘 세부과정 2) h 번째 장치의 작동 상태 수준이 1이고 네트워크 구성요소인 장치의 수, n보다 작을 때 정의 1의 (2)의 $g_2(h)$에 의해 S_c로부터 $y_h \leftarrow 0, y_{h+1} \leftarrow 1$을 통해 후보 상태를 생성한다.

(알고리즘 세부과정 3) S_c의 h번째 장치의 작동 상태 수준이 1일 때 $c(h)$의 값을 계산하고 h와 $c(h)$를 인수로 하여 Candidate_2 함수를 호출하고 세 번째 유형의 후보 상태들을 조건에 따라 생성한다. 이를 위해 정의1의 (3),(4),(5)를 이용한다.

```
Algorithm
/* INPUT: number of units, n, probability of unit's state,
p_ij, i=1,...,n, j=0,...,D_i-1, */
/* OUTPUT: m MPS, S_i=(y_1,...,y_n), i=1,...,m */
 Initialize:  S_1←(0,0,...,0),  S_2←(1,0,...,0),
H=Φ
        i←2, S_c←S_2, h←1, s=S_c(h)←1;
Do
  /* 세부과정 1 */
  If s < D_h - 1 Then
   generate candidate state, S' by g_1(h);
   compute probability of candidate state, S', P(S');
   Call INSERT( S', P(S'), H);
  End If
  /* 세부과정 2 */
  If s=1 and h < N Then
   generate candidate state, S' by g_2(h);
   compute probability of candidate state, S', P(S');
   Call INSERT( S', P(S'), H);
  End If
  /* 세부과정 3 */
  If s=1 Then
     find c(h);
     Call Candidate_2( h, c(h));
  End If
  i←i+1
  S_i ← EXTRACT(H);
  S_c←S_i; s← S_c(h);
 Until enough states are selected
End Algorithm/* end of algorithm */
```

그림 3. 최고 기능 상태(MPS) 생성 알고리즘
Fig. 3. MPS Generation Algorithm

그림 4의 알고리즘 Candidate_2 함수에서, $h = c(h)+1$의 조건은 시스템 상태 S의 h번째까지의 장치 상태 수준이 모두 1일 경우에 장치 1부터 $h-1$까지 모두 $y_k \leftarrow y_k + 1$, $k = 1,\cdots,h-1$에 의해서 최고 가능 후보 상태를 생성한다.

$h \neq c(h) + 1$일 때 $S(h - c(h) - 1) = 0$이고,

$h - c(h) = 2$이면 $g_3(1)$에 의해서 최고 가능 후보 상태들을 생성하고 모두 $c(h)$개의 후보상태 들을

$g_1(h-k)$, $k = 1,\cdots,c(h)$에 의해 생성 한다.

$S(h-c(h)-1) = 0, h - c(h) > 2$이고 ,

$S(h-c(h)-2) = 1$을 모두 만족하면

$g_2(h-c(h)-2)$에 의해 후보상태를 생성한다. 또한, $c(h) \geq 1$을 만족하면 $g_1(h-k), k = 1,\cdots,c(h)$에 의해 $c(h)$ 후보 상태를 추가로 생성하여 모두 힙에 삽입한다.

위 알고리즘에서 공통적으로 사용하는 INSERT 함수는 후보 상태들의 확률을 계산하고 힙에 삽입하는 함수이며 EXTRACT 함수는 힙의 근 노드로부터 다음 순위의 최고 가능 상태를 취득하는 함수이다.

```
Candidate_2( h, c(h))
/* INPUT: last index of state, h,
 number of state level 1 from h-1, c(h) */
/* OUTPUT: generation of 3rd type of candidate states
and Insert into heap, H */
If h = c(h) +1 Then /* when y_i=1 ∀ i */
 For k=1 to h-1 Do
  generate candidate state, S' by g_1(k);
  INSERT( S', P(S'), H);
 End For
Else
 If S( h - c(h) -1) = 0 Then
  If h - c(h) =2 Then
     generate candidate state, S' by g_3(1);
     Call INSERT( S', P(S'), H);
     For k=1 to c(h) Do
        generate candidate state, S' by g_1(h-k);
```

```
        Call INSERT( S', P(S'), H);
      End For
  Else
      If h − c(h) > 2 Then
        If S( h − c(h) − 2) = 1 Then
          generate candidate state S'
            by g2(h − c(h) − 2);
          Call INSERT( S', P(S'), H);
        End If
        If c(h) ≥ 1 Then
          For k = 1 to c(h) Do
            generate candidate state S'
              by g1(h − k);
            Call INSERT( S', P(S'), H);
          End For
        End If
      End If
  End If
 End If
End If
End Candidate_2/* end of Candidate_2()*/
```

그림 4. 최고 가능 상태(MPS) 생성 알고리즘(세 번째 유형)
Fig. 4. MPS Generation Algorithm(3rd type)

3. 실험 및 알고리즘 평가

본 절에서는 1, 2절에서 제안된 최고 가능 네트워크 상태 생성 알고리즘을 통해 알고리즘의 효율성 평가에 있어서 메모리 측면에서 고려하고자 한다. 기존 방법과의 비교는 네트워크의 위상 구조 없이 네트워크 시스템의 장치 수와 각 성능 혹은 작동 확률만을 명시하여 Gabler외 및 Shier외의 방법과 비교 검토하고 그 장·단점에 관하여 논의 하고자 한다. 프로그래밍 언어는 C언어를 사용하여 구현하였으며 같은 조건하에서 실행하여 결과를 얻고 이를 비교, 검토하였다.

실험예제 1. 네트워크 시스템의 장치의 수 는 $n=3$이고 $D_i=D=4$인 경우 각 작동 상태의 확률이 표 4와 같을 때 알고리즘의 효율성을 메모리 측면에서 고려해 보기로 한다. 표 2에서 $w_i=p_{i1}/p_{i0}$에 의한 순위는 원 네트워크 시스템의 장치 레이블과 같은 경우이다. 네트워크 상태 공간의 크기는, $|\Omega|=64$으로 그리 크지 않은 경우이나 알고리즘을 평가 하는 데는 큰 무리는 없는 수준이다. MPS의 수를 10부터 전체 네트워크 상태를 생성할 때까지 일정 간격(10개)로 생성하여 표 5와 같은 결과를 얻었다.

상태공간의 포함정도는 약 84%부터 100%까지이다. 최고 가능 네트워크 상태의 개수를 m으로 표기하였으며 최대 힙 크기는 m개의 네트워크 상태를 생성할 때까지 최대 힙의 크기를 나타내었다. 또한, 기억공간의 효율성을 평가하기 위해서는 m개의 네트워크 상태를 추출할 때까지의 힙에 남아 있는 노드들이 총수의 평균값을 나타내었다.

표 4. 상태, 확률, 확률비, 재 순서 순위
($n=3$, D_i=4, i=1,2,3)
Table 4. State, Probability, Ratio, Reordered Rank

		상태 j				w_i (p_{i1}/p_{i0})	순위
		0	1	2	3		
장치 i	1	0.65	0.20	0.10	0.05	0.3076	1
	2	0.75	0.15	0.07	0.03	0.2	2
	3	0.80	0.10	0.07	0.03	0.125	3

표 5. 최대 가능 상태 수, 힙의 최대 크기, 평균 힙 잔여 노드 수
Table 5. Number of Most Probable states, Maximal Heap Size, Average Number of Residual Nodes in Heap

m	Gaebler 외		Shier외		제안된 방법		포함도 (%)
	최대 힙크기	평균 잔여 노드	최대 힙크기	평균 잔여 노드	최대 힙크기	평균 잔여 노드	
10	8	5.25	6	4.00	4	2.75	83.69
20	12	7.56	9	5.61	8	4.22	93.59
30	13	9.25	11	6.93	9	5.46	97.33
40	13	9.68	11	7.52	9	5.95	98.99
50	13	9.33	11	7.46	9	5.97	99.69
64	13	8.03	11	6.53	9	5.39	100

이러한 평가 방법에 의해 최대 힙 크기에 관하여는 제안된 방법이 Gaebler외의 방법에 비해 최고 약 50% 적게 소모 되었으며 Shier외의 방법에 비하여는 약 30%정도 적게 소요됨을 알 수 있다. m개의 MPS를 생성할 때까지의 평균적인 힙의 사용량인 잔여 노드 수는 Gaebler외의 방법에 비해 최고 48%적게 사용하며, Shier외의 방법에 비해 최고 약 30%적게 사용함을 알 수 있다. 포함정도가 98.99%인 경우를 기준으로 할 때, 평균 힙의 사용량은 Gaebler와 Chen에 비해 약 39% 적게 사용하며, Shier외의 방법에 대해서는 약 21% 적게 사용함을 알 수 있다.

실험 예제 2. 두 번째 예제에서는 장치의 수가 다소 증가된, $n=5$이고 각 작동 상태, $D_i=D=4$으로 구성된 네트워크에 대해서 알고리즘의 성능을 검토하고자 한다. 네트워크 시스템의 각 성능 혹은 작동 확률은 표 6과 같고 각 장치의 작동 수준 0의 확률, p_{i0}는 그다지 크지 않은 경우이다.

전체 네트워크 상태 공간의 크기는, $|\Omega| = 1024$이며 MPS의 수 m의 크기를 50부터 $|\Omega|$까지 적당 간격으로 생성할 때의 결과를 표 7에 나타내었다. $|\Omega|$의 약 20%인 $m = 200$일 때 95%의 포함정도를 초과한다.

표 6. 상태, 확률, 확률비, 재 순서 순위
($n = 5$, M_i =4, i=1,2,3)
Table 6. State, Probability, Ratio, Reordered Rank

		상태 j				w_i (p_{i1}/p_{i0})	순위
		0	1	2	3		
장치 i	1	0.7	0.2	0.05	0.05	0.286	2
	2	0.65	0.2	0.1	0.05	0.31	1
	3	0.75	0.2	0.03	0.02	0.267	3
	4	0.85	0.08	0.04	0.03	0.0941	5
	5	0.8	0.1	0.07	0.03	0.125	4

표 7. 최고가능상태 수, 힙의 최대크기, 평균 힙 잔여노드 수
Table 7. Number of Most Probable states, Maximal Heap Size, Average Number of Residual Nodes in Heap

m	Gaebler의		Shier외		제안된 방법		포함도 (%)
	최대 힙크기	평균 잔여 노드	최대 힙크기	평균 잔여 노드	최대 힙크기	평균 잔여 노드	
50	49	27.8	34	18.9	29	15.9	84.42
100	77	46.4	55	38.4	42	25.7	92.22
200	116	72.6	85	50.7	63	39.6	97.09
300	137	89.9	100	65.0	79	49.6	98.70
400	144	101.8	118	76.8	86	56.6	99.38
500	146	109.8	118	83.0	91	62.0	99.70
600	146	113.2	118	86.9	91	64.3	99.86
800	146	112.3	118	89.3	91	65.9	99.98
900	146	107.6	118	86.8	91	63.9	99.99
1024	146	97.8	118	80.1	91	58.8	100

m 최고 가능 상태를 생성할 때까지 이용되는 힙의 최대 크기에 관하여 비교한 결과, Gaebler외의 방법에 비해 Shier외의 방법은 최저 20%에서 최고 30%까지 감소되었으며, 제안된 방법은 최저 약 37%에서 최고 46%까지 감소되었음을 알 수 있다. Shier외의 방법에 비해서는 최대 힙 크기 측면에서 최저 15%에서 최고 28%까지 감소된 결과를 보였다. 평균 잔여 노드 수에 관하여는 최저 약 12%에서 최고 약 27%까지 Shier의 방법에 비해 적게 메모리를 사용함을 알 수 있다. 실행 시간에 관한 경험적 결과는 보이지 않았으나 제안된 방법은 힙의 크기가 실행시간에 영향을 미치는 주요인이 되며 상당히 그 크기가 감소되기 때문에 m 의 크기에 따른 전체 범위에서 항상 실행시간이 적게 소요되리라 본다.

IV. 결 론

본 논문에서는 네트워크 구조를 갖는 복잡한 시스템의 성능 품질이나 신뢰도 평가에 대한 근사값 계산 방법에 대해서 논의하였다. 네트워크의 상태 공간 Ω 의 크기가 기하급수적으로 증가 하는 문제들에서 기대 성능(Expected Performance)이나 신뢰도의 근사계산을 위해서 발생 가능성이 높은 최고 가능 상태(MPS)들 만을 이용하는 방법을 논의하였으며, 이때 선행되어야 하는 전체 네트워크 시스템 상태들 중에서 임의의 m 개의 상태들을 확률 크기의 내림차순으로 생성하는 효율적인 알고리즘을 제안하였다. 기존의 다중 모드 알고리즘과 유사한 절차를 취하되 계산 소요시간을 축소하기 위해 대수적 비교가능의 개념을 통해 이로부터 비교적 적은 수의 비교 가능한 후보 상태들을 생성하는 개선된 알고리즘을 제안하였다. 실험예제를 통해 메모리의 사용 측면에서 비교 검토하였으며 장치의 수나 각 장치의 작동 상태의 수의 변화에 따라 메모리의 효율성이 개선되었음을 확인할 수 있었다. 대부분의 복잡한 네트워크 시스템에서 각 장치의 첫 번째 상태(모드)로 자동할 확률이 다른 상태에 비해 주로 매우 크며 적은 수의 MPS에 의해서도 약 90%의 포함정도를 보이기 때문에 제안된 알고리즘의 활용성이 있다 할 수 있다. 본 논문에서는 성능 품질 및 신뢰도의 측도들에 관하여는 고려하지 않았으나 여러 다양한 네트워크 구조를 갖는 복잡한 시스템에서 특정 목적의 성능 및 신뢰도 측도(measure)의 개발과 이에 대한 제안된 MPS생성 알고리즘을 적용 및 활용하는 영역들이 향후의 과제로서 남아 있다할 수 있다.

참고문헌

[1] 권오흠, 김숙연, "무선 센서 네트워크에서 두 노드간 거리 추정 기법," 한국컴퓨터정보학회논문지, 제10권, 5호, 209-216쪽, 2005년 11월.

[2] 노경택, 정수목, "이동망에서 확장된 HMIP를 이용한 경로 최적화", 한국컴퓨터정보학회논문지, 제12권, 6호, 235-242쪽, 2007년 12월.

[3] K.K. Aggarwal, Suresh Rai, "Reliability Evaluation in Computer-Communication Networks," IEEE Transaction on Reliability, Vol. R-30, No. 1, pp. 32-35, 1981.

[4] V.O.K. Li and J.A. Silvester, "Performance

analysis networks with unreliable components," IEEE Transactions on Communications, Vol. COM-32, pp. 1105-1110, 1984.

[5] S.N. Chiou and V.O.K. Li, "Reliability Analysis of a Communication Network with Multimode Components," IEEE Journal on Selected Areas in Communications, Vol. SAC-4, No. y, pp 1156-1161, 1986.

[6] Gaebler, R.F. and Chen, R.J., "An Efficient Algorithms for Enumerating States of a System with Multimode Unreliable Components," ORSA/TIMS, St. Louis, MO.,1987.

[7] Shier, D.R., Bibelnieks, E., Jarvis, J.P., Lakin, P.J., "Algorithms for approximating the Performance of Mulimode Systems," IEEE Proceedings:INFOCOM '90, Ninth Annual Joint Conference of the IEEE Computer and Communication Societies, pp. 741-748, 1990.

[8] Yang, C.L, Kubat, P., "An Algorithm for Network Reliability bounds", ORSA Journal on Computing, Vol. 2, No. 4, pp. 336-345, 1990.

[9] M.O. Ball, "Complexity of network reliability computations," Networks, Vol. 10, No. 2, pp 153-165, 1980.

[10] Michael Ball, Richard M. Van Slyke, Israel Gitman and Howard Frank, "Reliability of Packet Switching Broadcast Radio Networks," IEEE Transactions on CirCuits and Systems, Vol. CAS-23, No. 12, 1976.

[11] F. Boesh, D. Gross, C. Suffel, "A Coherent Model for Reliability of Multiprocessor Networks," IEEE Transactions on Reliability, Vol. 45, No. 4, pp. 678-684, 1996.

[12] Daryl D. Harms, M. Kraetzl, C.J. Colbourn, J.S. Devitt, "*Network Reliability*," CRC Press, 1995.

[13] E. Horowitz and S. Sahni, "Fundamentals of data structure," Pitman, 1976.

[14] John F. Meyer, "On Evaluating the Performability of Degradable Computing Systems," IEEE Transactions on Computers, Vol. C-29, No. 8, pp. 720-730, 1980.

[15] Wei_Jenn Ke, Sheng-De Wang, "Reliability Evaluation for Distributed Computing Networks with Imperfect Nodes," IEEE Transantions on Reliability, Vol. 46, No. 3, pp. 342-349, 1997.

[16] B. Sanso and F. Soumis, "Communication & Transportation Network Reliability Using Routing Models," IEEE Transactions on Reliability, Vol. 40, No. 1, pp. 29-38, 1991.

저 자 소 개

오 대 호

1991: 동국대학교 이학석사.
2002: 동국대학교 이학박사.
1997 - 현재: 한림성심대학 의료기기정보과 교수
관심분야: 성능평가, 네트워크, 생체신호처리, 영상처리, 데이터마이닝