▩ 연구논문

네트워크 신뢰도를 추정하기 위한 SMCS/SMPS 시뮬레이션 기법

SMCS/SMPS Simulation Algorithms for Estimating Network Reliability

서 재 준*
Suh, Jae Joon

Abstract

To estimate the reliability of a large and complex network with a small variance, we propose two dynamic Monte Carlo sampling methods: the sequential minimal cut set (SMCS) and the sequential minimal path set (SMPS) methods. These methods do not require all minimal cut sets or path sets to be given in advance and do not simulate all arcs at each trial, which can decrease the variance of network reliability. Based on the proposed methods, we develop the importance sampling estimators, the total hazard (or safety) estimator and the hazard (or safety) importance sampling estimator, and compare the performance of these simulation estimators. It is found that these estimators can significantly reduce the variance of the raw simulation estimator and the usual importance sampling estimator. Especially, the SMCS algorithm is very effective in case that the failure probabilities of arcs are low. On the contrary, the SMPS algorithm is effective in case that the success probabilities of arcs are low.

1. 서론

해석적인 방법으로 대규모의 복잡한 네트워크의 신뢰도를 계산하려면 그 계산량이 지수적으로 증가하여 거의 불가능하게 된다. 따라서 시뮬레이션에 의한 추정방법을 많이 이용하고 있으며 그 중에서 Monte Carlo 시뮬레이션[1,2,4,6,7]이 대표적인 방법이다. 그러나 raw Monte Carlo 방법으로 상당히 정밀한 신뢰도 추정치를 얻기 위해서는 매우 많은 시뮬레이션 반복 시행이 필요하다. 그러므로 상대적으로 적은 시뮬레이션 시행으로 정밀한 신뢰도를 추정하기 위해 추정치의 분산을 줄이기 위한 연구[3,5,7,9]가 진행되어 왔다.

Mazumdar[7]는 Monte Carlo 추정치의 분산을 줄이기 위해 네트워크를 구성하는 각 아크(arc)의 원래 고장률 대신 적절히 선택한 다른 고장률을 이용하는 임포턴스 샘플링(Importance Sampling) 방법을 제안하였다. 이 방법은 항상 추정치의 분산을 감소한다는 보장이 없으며 또한 네트워크를 구성하는 모든 아크들을 매번 시뮬레이션해야 하므로 상대적으로 분산 감소 효과를 줄일 수 있다. Ross[9]는 random hazard를 네트워크 신뢰도의 추정치로 이용하는 분산 감소 기법을 제시했는데 매우 훌륭한 분산 감소 효과를 보여주었으나 시뮬레이션을 시행하기 전에 모든 MCS(Minimal Cut Set)을 미리 구해야 한다. 네트워크가 크고 복잡하면 모든 MCS를 구하는 것은 또다른 어려운 문제로 나타난다.

---

† 이 논문은 2000년도 대전산업대학교 교내학술연구비 지원을 받았음

* 한밭대학교 산업공학과

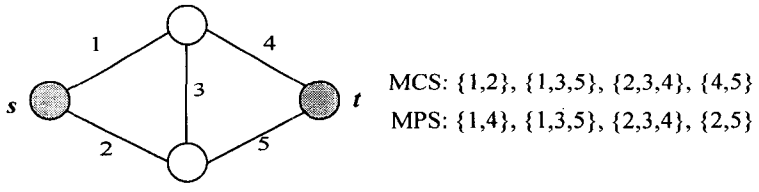
본 논문에서는 매 시행마다 모든 아크들을 시뮬레이션할 필요가 없고 시뮬레이션 시행마다 모든 MCS나 모든 MPS(Minimal Path Set)를 미리 구할 필요가 없으며 또한 추정량의 분산을 줄일 수 있는 두가지 동적 몬테칼로(dynamic Monte Carlo) 시뮬레이션 방법, 즉 SMCS (Sequential Minimal Cut Set) 알고리듬과 SMPS(Sequential Minimal Path Set) 알고리듬을 제안하였다. SMCS 알고리듬은 소스노드(source node)와 연결된 아크들로 구성된 하나의 MCS를 찾아 시뮬레이션하여 그 결과에 따라 네트워크를 축소시켜가면서 네트워크의 연결 상태가 확인될 때까지 일련의 과정을 반복한다. 최종적인 네트워크 신뢰도는 일단 네트워크 상태가 확인되면 SMCS 알고리듬에 따라 유도된 신뢰도 추정량(estimator)에 의해 계산된다. SMPS 알고리듬도 SMCS 알고리듬과 유사한 방법으로 진행되며 단지 MCS 대신 MPS를 이용하여 시뮬레이션을 수행하는 점이 SMCS 알고리듬과 다르다.

본 논문은 서론에 이어 2장에서 제안한 두 알고리듬의 개념과 절차를 설명하고 3장에서는 두 알고리듬에 따라 네트워크 신뢰도를 추정할 수 있는 몇가지 신뢰도 추정량들을 유도하였다. 4장에서 간단한 브릿지 네트워크의 예를 들어 제안한 알고리듬에 따라 시뮬레이션을 수행하고 유도한 추정량들의 분산을 비교분석하였다. 마지막으로 5장에 결론을 내렸다.

2. SMCS 및 SMPS 알고리듬

2.1 네트워크 연결성(Network Connectivity)

노드(node)와 아크(arc)로 구성된 무방향 네트워크(undirected network)에서 모든 노드들은 고장이 나지 않으며 아크들은 서로 독립적으로 고장이 발생한다고 할 때, 임의의 소스노드 (source node) s와 터미널노드(terminal node) t가 연결될 확률, 즉 네트워크의 신뢰도를 계산하는 데 관심이 있다고 하자. 이런 네트워크를 소스-터미널 연결망(s-t connection network)이라고 하는데 이런 네트워크에서 시스템이 정상상태인지 고장상태인지를 판단함에 있어서 모든 아크들의 상태를 알 필요가 없다. 다시 말해서 하나의 MCS를 구성하는 아크들이 모두 고장이면 시스템은 고장상태이며, 또한 하나의 MPS를 구성하는 아크들이 모두 정상이면 소스노드와 터미널노드는 연결될 수 있음을 알 수 있다. 각각 네개의 MCS와 MPS를 가진 다음 (그림 1)과 같은 간단한 브릿지 네트워크를 예로 들어 보자.



(그림 1) 브릿지 네트워크

MCS(또는 MPS)를 이용해 네트워크의 연결 여부를 결정하는 방법을 알아보기 위해 우선 첫번째 MCS {1,2}를 시뮬레이션한 후 아크들의 상태를 살펴보자. 두 아크가 모두 고장이면 네트워크는 연결될 수 없다고 판단할 수 있으므로 더 이상 다른 아크들의 상태를 알아 볼 필요가 없다.

두 아크중 하나가 살아 있다면 네트워크의 연결 여부를 판단할 수 없으므로 계속해서 다른 아크들의 상태를 살펴보아야 한다. 만일 아크 1이 살아 있고 아크 2가 고장이라면 첫번째 MCS {1,2}와 두번째 MCS {1,3,5}는 살아있는 아크 1을 포함하고 있으므로 더 이상 MCS가 될 수 없다. 세번째 MCS {2,3,4}는 아크 2가 고장임이 밝혀졌으므로 네트워크의 연결여부를 판단

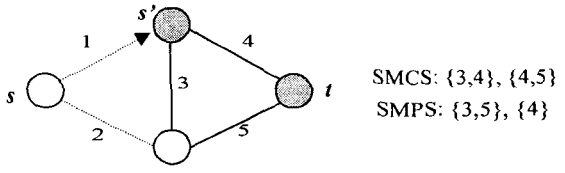
하는데 아크 3과 4의 상태만 알면 된다. 네번째 MCS {4,5}는 두 아크에 대한 아무런 정보가 없으므로 그대로 MCS의 조건을 가지게 된다. 즉 MCS {1,2}를 시뮬레이션 했을 때 아크 1이 살아있고 아크 2가 고장난 경우, 최종적으로 다시 구한 MCS는 {3,4}와 {4,5}가 되는데 일련의 시뮬레이션 실행 결과에 따라 구해진 MCS라 해서 SMCS(Sequential MCS)라 부르기로 한다.

아크 1이 살아있고 아크 2가 고장난 경우, MPS에 대해서도 같은 방법으로 SMPS (Sequential MPS)를 구해보자. 세번째 MPS {2,3,4}와 네번째 MPS {2,5}는 고장난 아크 2를 포함하고 있으므로 더 이상 MPS의 요건을 갖추지 못하므로 버리고 첫번째 MPS {1,4}와 두번째 MPS {1,3,5}는 살아있는 아크 1을 제외하면 최종적인 SMPS는 {3,5}와 {4}를 얻게 된다. <표 1>은 첫번째 MCS {1,2}의 시뮬레이션 결과에 따라 앞에서 설명한 방법으로 구한 SMCS와 SMPS를 보여주고 있다.

<표 1> 아크 {1,2}의 시뮬레이션 결과에 따른 SMCS와 SMPS

아크 {1,2}의 상태	SMCS	SMPS
1 정상, 2 정상	{4,5}	{4}, {5}
1 정상, 2 고장	{3,4}, {4,5}	{3,5}, {4}
1 고장, 2 정상	{3,5}, {4,5}	{3,4}, {5}
1 고장, 2 고장	네트워크 고장	네트워크 고장

<표 1>에서 두번째 경우의SMCS {3,4}, {4,5}와 SMPS {3,5}, {4}는 각각, (그림 2)와 같이 살아 있는 아크 1을 따라 소스노드 s를 소스노드 s' 로 옮기고 시뮬레이션한 아크들을 모두 없앤 후 새로 구성한 축소된 네트워크의 MCS, MPS와 정확하게 일치한다. 여기서 축소된 네트워크의 새로운 소스노드 s' 를 sequential 소스노드로 부르기로 한다.



(그림 2) 축소된 브릿지 네트워크

계속해서 축소된 브릿지 네트워크의 SMCS {3,4}를 시뮬레이션해 보자. 두 아크 모두 고장이면 (그림 1)의 원래 브릿지 네트워크의 세번째 MCS {2,3,4}가 고장이므로 소스노드와 터미널노드는 연결될 수 없다고 판단할 수 있다. 만일 아크 4가 정상이면 원래 브릿지 네트워크의 아크 1과 4를 따라 하나의 경로가 존재하므로, 즉 MPS {1,4}가 정상이므로 소스노드와 터미널노드는 연결될 수 있음을 알 수 있다. 지금까지 설명한 방법으로 MCS와 MPS를 이용해서 소스노드와 터미널노드의 연결 여부를 판단할 수 있다.

2.2 SMCS 알고리듬

2.1절에서 예를 든 브릿지 네트워크는 간단한 네트워크라 모든 MCS를 쉽게 찾을 수 있지만 네트워크가 크고 복잡하면 매우 어려운 작업이 된다. 그러나 하나의 MCS는 간단하게 구할 수 있다. 즉, 소스노드와 직접 연결된 아크들중에서 터미널노드까지 연결될 수 있는 아크들은

하나의 MCS를 구성한다. SMCS 알고리듬은 이런 기본원리를 바탕으로 소스노드와 직접 연결된 아크들로 구성된 하나의 MCS를 구하여 이의 상태를 시뮬레이션하는 것으로부터 시작된다. 시뮬레이션해 본 아크들의 상태를 바탕으로 앞의 예에서 설명한 바와 같이 소스노드와 터미널노드의 연결 여부가 판단되면 시뮬레이션 시행을 끝내고 3.1절의 SMCS 알고리듬에 근거한 추정량으로 네트워크 신뢰도를 추정할 수 있다. 시스템의 상태가 파악되지 않으면 아래에 기술한 네트워크 축소방법에 따라 새로운 네트워크를 구성하여 축소된 네트워크의 소스노드로부터 나온 아크들로 구성된 SMCS를 구해 시스템의 상태가 파악될 때까지 계속적인 반복시행을 하면 된다. 지금까지 설명한 SMCS 알고리듬의 절차를 요약해서 정리하면 다음과 같다.

단계 1. (새로운) 소스노드와 직접 연결된 아크들로 구성된 하나의 (sequential) MCS를 찾아 그 아크들을 시뮬레이션한다.

단계 2. 시뮬레이션된 아크들의 상태가 아래의 중지요건들 중 어느 하나를 만족하는가? 만족하지 않으면 단계3으로 가고 만족하면 단계4로 간다.

중지요건

① 시뮬레이션한 현 (sequential) MCS의 모든 아크들이 고장이면 소스노드와 터미널노드는 연결될 수 없으므로 시스템 고장으로 판단하고 중지한다.

② 시뮬레이션된 현 (sequential) MCS의 아크들 중 살아있는 아크의 또 다른 종단노드(ending node)가 터미널노드이면 원래의 소스노드로부터 하나의 경로가 확인되었으므로 네트워크 연결로 판단하고 중지한다.

단계 3. 아래에 기술한 네트워크 축소 방법에 따라 네트워크를 축소하고 단계 1로 간다.

네트워크 축소 방법

① 현재의 소스노드를 없애고 살아있는 아크들의 또 다른 종단노드들을 하나로 묶어 새로운 (sequential) 소스노드로 만든다.

② 시뮬레이션하여 그 상태가 파악된 현 (sequential) MCS의 모든 아크들을 제거한다.

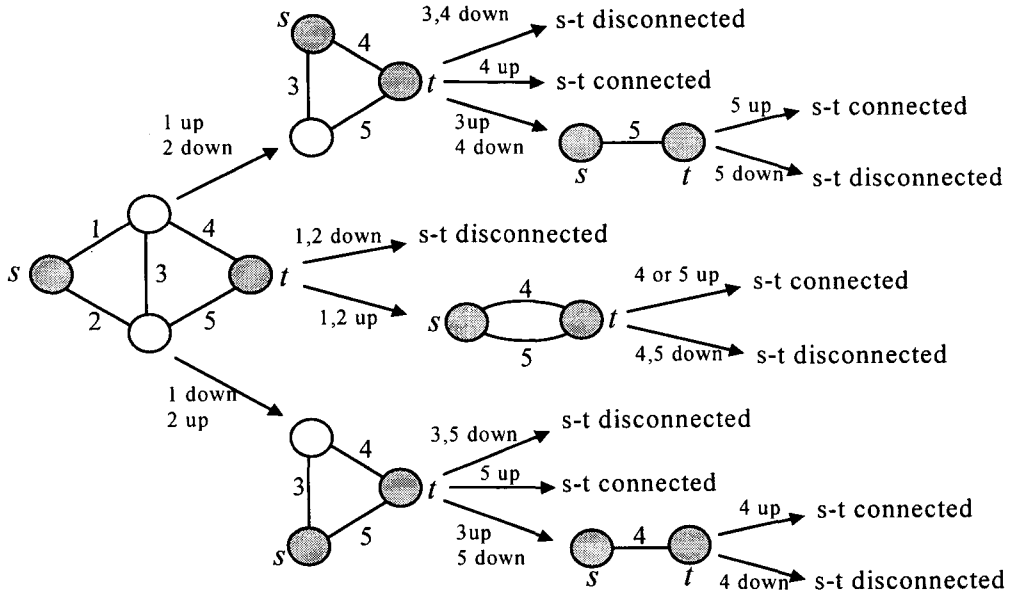
③ 현 네트워크의 (sequential) 소스노드와 직접 연결되었으나 (sequential) MCS에 속하지 않은 아크들과 그 아크들에 연결된 모든 아크 및 노드들, 즉 irrelevant subnetwork는 더 이상 소스-터미널 연결 여부와 무관하므로 모두 제거한다.

④ 살아있는 아크들의 또 다른 종단노드들을 연결하는 아크들은 아래의 proposition에 의해 축소된 네트워크에서 self-cycling 아크가 되어 더 이상 SMCS의 구성요건이 되지 않으므로 모두 제거한다.

Proposition) 소스노드와 연결된 두 아크들의 또 다른 종단노드들을 연결하는 아크가 어떤 MCS에 속해 있다면 그 MCS는 반드시 소스와 연결된 두 아크중 어느 하나를 반드시 포함하고 있다.

단계 4. 시스템의 상태가 파악되었으므로 시뮬레이션을 중지하고 그 결과에 따라 네트워크 신뢰도를 추정한다. (SMCS 알고리듬에 의한 네트워크 신뢰도 추정량은 3.2절 참조)

SMCS 알고리듬에 따라 (그림 1)의 브릿지 네트워크를 시뮬레이션하는 과정을 도식화해보면 (그림 3)과 같다.



(그림 3) SMCS 알고리듬에 의한 브릿지 네트워크의 시뮬레이션 과정

2.3 SMPS 알고리듬

SMCS 알고리듬과 마찬가지로 SMPS 알고리듬도 소스노드와 터미널노드를 연결하는 하나의 MPS를 구해서 그 아크들을 시뮬레이션하는 것으로 시작된다.

시뮬레이션한 MPS를 구성하는 아크들의 상태에 따라 소스노드와 터미널노드의 연결 여부가 판단되면 시뮬레이션 시행을 끝내고 3.2절에서 기술한 SMPS 알고리듬에 근거한 추정량으로 네트워크 신뢰도를 추정한다. 시스템의 상태가 파악되지 않으면 마찬가지로 네트워크 축소방법에 따라 새로운 네트워크를 구성하여 소스노드와 터미널노드를 연결하는 하나의 SMPS를 구해 시스템의 상태가 파악될 때까지 시뮬레이션 시행을 반복한다. SMPS 알고리듬의 절차도 SMCS 알고리듬과 유사하며 다음과 같다.

단계 1. 하나의 소스노드와 터미널 노드를 연결하는 하나의 (sequential) MPS를 구해 그 아크들을 시뮬레이션 한다.

단계 2. 시뮬레이션된 아크들의 상태가 아래의 중지요건들 중 어느 하나를 만족하는가? 만족하지 않으면 단계3으로 가고 만족하면 단계4로 간다.

중지요건

① 시뮬레이션한 현 (sequential) MPS의 모든 아크들이 정상이면 소스노드와 터미널노드 간 하나의 경로가 존재하므로 시스템 정상으로 판단하고 중지한다.

② 더 이상 (sequential) MPS가 발견되지 않으면 시스템 고장으로 판단하고 중지한다.

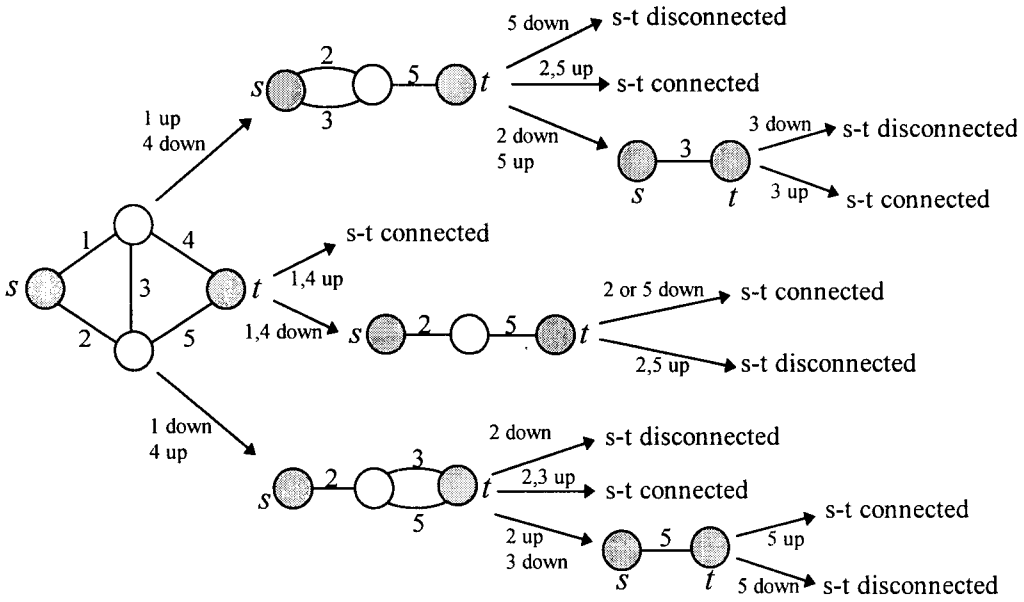
단계 3. 아래에 기술한 네트워크 축소방법에 따라 네트워크를 축소하고 단계 1로 간다.

네트워크축소방법

① 현재의 소스노드를 없애고 살아있는 아크들의 또 다른 종단노드들을 하나로 묶어 새로운 (sequential) 소스노드로 만든다.
② 시뮬레이션한 현 (sequential) MPS의 모든 아크들을 제거한다.
③ self-cycling 아크와 irrelevant subnetwork가 있으면 모두 제거한다.

단계 4. 시스템의 상태가 파악되었으므로 시뮬레이션을 중지하고 그 결과에 따라 네트워크 신뢰도를 추정한다. (SMPS 알고리듬에 의한 네트워크 신뢰도 추정량은 3.3절 참조)

SMPS 알고리듬에 따라 (그림 1)의 브릿지 네트워크를 시뮬레이션하는 과정을 도식화해보면 (그림 4)와 같다.



(그림 4) SMPS 알고리듬에 의한 브릿지 네트워크의 시뮬레이션 과정

3. 네트워크 신뢰도 추정량

3.1 용어 설명

노드와 m개의 아크로 구성된 무방향 소스-터미널 연결망에서 모든 노드들은 고장이 나지 않으며 각각의 아크들은 서로 독립적으로 정상 또는 고장의 두 상태로만 작동한다고 하자. 소스-터미널 연결망에서는 소스노드와 터미널노드가 연결될 확률(네트워크 신뢰도(network reliability)) 또는 연결되지 않을 확률(네트워크 불신뢰도(network unreliability))에 관심이 있다. 본 장에서는 2장에서 설명한 SMCS 또는 SMPS 알고리듬을 적용하여 네트워크 신뢰도(또는 불신뢰도)를 추정할 수 있는 추정량들을 유도한다. 네트워크 신뢰도(또는 불신뢰도) 추정량을 유도하기 위해 필요한 용어들을 다음과 같이 정의하자.

X_i : 아크 i가 정상인지 고장인지를 나타내는 상태 확률변수로 0 또는 1의 값을 가지는데 네트워크 신뢰도 추정량에서는 아크 i가 정상일 때 1의 값을, 네트워크 불신뢰도 추정

량에서는 아크 i가 고장일 때 1의 값을 가진다.

p_i : 아크 i가 정상일 확률로 네트워크 신뢰도 추정량에서 $Pr\{X_i=1\}$

q_i : 아크 i가 고장일 확률로 네트워크 불신뢰도 추정량에서 $Pr\{X_i=1\}$

$\underline{X}$: 아크들의 상태를 나타내는 상태벡터, $(X_1, \cdots, X_m)$

$\Phi(\underline{X})$: 상태벡터 $\underline{X}$하에서 소스노드와 터미널노드의 연결 여부를 나타내는 함수로, 네트워크 신뢰도 추정량에서는 상태벡터 $\underline{X}$하에서 소스노드와 터미널노드가 연결될 때 1의 값을, 네트워크 불신뢰도 추정량에서는 상태벡터 $\underline{X}$하에서 소스노드와 터미널노드가 연결되지 못할 때 1의 값을 가지며 나머지 경우는 0의 값을 가진다.

P_S : 소스노드와 터미널노드가 연결될 확률, 즉 네트워크 신뢰도

P_F : 소스노드와 터미널노드의 연결되지 못할 확률, 즉 네트워크 불신뢰도

위와 같은 정의하에서 네트워크 신뢰도 P_S와 네트워크 불신뢰도 P_F는 모두 $Pr\{\Phi(\underline{X})=1\}$로 표현될 것이다. SMCS 또는 SMPS 알고리듬에 따라 시뮬레이션할 때, 소스-터미널 연결여부가 알려질 때까지 시뮬레이션된 SMCS와 SMPS를 각각 $C_1, C_2, \cdots, C_i$와 $S_1, S_2 \cdots, S_k$라 하자. 3.2절과 3.3절에서는 지금까지 설명한 용어와 가정하에서 SMCS 또는 SMPS 알고리듬을 이용하여 네트워크 신뢰도 또는 불신뢰도를 추정할 수 있는 몇가지 추정량들을 유도하였는데 각각 아래첨자 r과 u로 표현하였다. 여기서 기술한 추정량들은 기존의 분산감소 기법을 이용하여 2장에서 설명한 알고리듬을 적용할 수 있도록 변형시킨 추정량들과 일부는 새로 유도한 추정량들로 모두 불편 추정량들(unbiased estimators)이다.

3.2 SMCS 알고리듬에 따른 신뢰도 추정량

임포턴스 샘플링 추정량

임포턴스 샘플링 방법은 아크 i의 고장률 q_i대신 적절히 정한 다른 확률 β_i로 확률변수 X_i를 시뮬레이션함으로써 네트워크 불신뢰도 P_F를 추정하는 방법으로 임포턴스 샘플링 추정량(IMP_u)은 다음과 같이 표현되며 이는 불편추정량(unbiased estimator)임이 증명되어 있다[8].

$$IMPu = \Phi(\underline{X})\prod_{i=1}^{m}\frac{q_i^{X_i}(1-q_i)^{1-X_i}}{\beta_i^{X_i}(1-\beta_i)^{1-X_i}}$$

위의 식에서 β_i는 $\Phi(\underline{X})=1$이 되는 확률이 상당히 큰 값(0.5정도)이 되도록 선택하는 게 보통이다. 참고문헌[3]에는 분산을 줄일 수 있는 β_i값을 선택하는 한 아이디어를 제안하고 있는데 이에 따르면 모든 q_i가 같을 경우 MCS의 아크 수가 가장 작은 값을 전체 아크 수로 나눈 값이 임포턴스 샘플링 추정량의 분산을 작게 하는 것으로 기술하고 있다.

SMCS 알고리듬으로 j번째 SMCS를 시뮬레이션한 후 네트워크 상태가 파악되었다면, 나머지 아크들은 네트워크 상태와 무관하므로 P_F와 P_S의 임포턴스 샘플링 추정량은 시뮬레이션된 아크들만을 대상으로 다음과 같이 변형시킬 수 있으며, 이는 기존의 임포턴스 샘플링 추정량과 마찬가지로 불편 추정량임을 증명할 수 있다.

$$CIMPu = \Phi(\underline{X})\prod_{i\in C}\frac{q_i^{X_i}(1-q_i)^{1-X_i}}{\beta_i^{X_i}(1-\beta_i)^{1-X_i}},\quad C = C_1 \cup C_2 \cup \cdots \cup C_j$$

$$CIMPr = \Phi(\underline{X})\prod_{i\in C}\frac{p_i^{X_i}(1-p_i)^{1-X_i}}{\beta_i^{X_i}(1-\beta_i)^{1-X_i}},\quad C = C_1 \cup C_2 \cup \cdots \cup C_j$$

토탈 해저드 추정량

랜덤 해저드(random hazard)의 개념은 참고문헌 [1,9]에 나와 있으며 Ross[9]는 이를 이용하여 시스템 신뢰도를 추정하는 방법을 제안하였다. 이것을 SMCS 알고리듬에 따라 토탈 해저드(total hazard) 추정량을 유도해보자.

$\{\underline{X}_n, n\geq 0\}$을 SMCS 알고리듬에 의해 n번째 SMCS를 시뮬레이션한 후의 네트워크 상태를 나타내는 마코프 프로세스(Markov Process)라 하고 $\underline{X}_F$를 네트워크 고장상태를 나타낸다고 할 때, 확률변수 N을

$$\mathbf{N} = \min(n > 0; \underline{X}_n \in \underline{X}_F)$$

라 두자. 랜덤 해저드 $h_n(n\geq 0)$을 n번째 SMCS를 시뮬레이션할 때 네트워크가 처음 고장상태로 들어갈 확률이라고 정의하면

$$h_n = \Pr\{N = n \mid \underline{X}_1, \cdots, \underline{X}_{n-1}\} = \prod_{i\in C_n} q_i$$

로 표현된다. j번째 SMCS를 시뮬레이션한 후 네트워크의 상태여부가 파악되었다고 할 때, 그 때까지의 랜덤 해저드의 총합을 토탈 해저드라 하며 토탈 해저드 추정량 THAZu는 네트워크 불신뢰도 P_F의 불편추정량이 된다.

$$\boldsymbol{THAZu} = \sum_{i=1}^{j} h_i$$

해저드 임포턴스 샘플링 추정량

$\Phi(\underline{X})$와 THAZu의 기대치는 모두 P_F가 되므로 임포턴스 샘플링 방법에서 $\Phi(\underline{X})$ 대신에 THAZu를 이용하여 P_F를 추정할 수 있다. 이 추정량을 해저드 임포턴스 샘플링 추정량이라 할 때 다음 식과 같이 표현되며 임포턴스 샘플링 추정량과 마찬가지로 P_F의 불편추정량이다.

$$\boldsymbol{HIMPu} = \left[\sum_{k=1}^{j}\prod_{i\in C_k} q_i\right]\left[\prod_{i\in C}\frac{q_i^{X_i}(1-q_i)^{1-X_i}}{\beta_i^{X_i}(1-\beta_i)^{1-X_i}}\right], C = C_1 \cup C_2 \cup \cdots \cup C_j$$

3.3 SMPS 알고리듬에 따른 신뢰도 추정량

임포턴스 샘플링 추정량

k번째 SMPS를 시뮬레이션한 후 네트워크 상태가 파악되었다면, SMPS 알고리듬에 의한 P_S와 P_F의 임포턴스 샘플링 추정량은 3.2절에서 설명한 바와 마찬가지로 각각 다음과 같이 변형시킬 수 있으며 이는 기존의 임포턴스 샘플링 추정량과 마찬가지로 불편 추정량이다.

$$\boldsymbol{PIMPr} = \Phi(\underline{X})\prod_{i\in S}\frac{p_i^{X_i}(1-p_i)^{1-X_i}}{\beta_i^{X_i}(1-\beta_i)^{1-X_i}},\quad S = S_1 \cup S_2 \cup \cdots \cup S_k$$

$$\boldsymbol{PIMPu} = \Phi(\underline{X})\prod_{i\in S}\frac{q_i^{X_i}(1-q_i)^{1-X_i}}{\beta_i^{X_i}(1-\beta_i)^{1-X_i}},\quad S = S_1 \cup s_2 \cup \cdots \cup S_k$$

토탈 세이프티 추정량

토탈 세이프티(Total Safety)는 3.2절의 토탈 해저드(Total Hazard)의 반대 개념으로 토탈 해저드와 같은 방법으로 네트워크 신뢰도 P_S를 추정하기 위한 토탈 세이프티 추정량을 유도할 수 있다.

{$\underline{Y}_n$, n≥0}을 SMPS 알고리듬에 의해 n번째 SMPS를 시뮬레이션한 후의 네트워크 상태를 나타내는 마코프 프로세스라 하고 $\underline{Y}_S$를 네트워크 정상상태를 나타낸다고 할 때, 확률변수 N을

$$\mathbf{N} = \min(n > 0; \underline{Y}_n \in \underline{Y}_S)$$

라 두자. 랜덤 세이프티 t_n(n≥0)을 n번째 SMPS를 시뮬레이션할 때 네트워크가 처음 정상상태로 판명될 확률이라고 정의하면

$$t_n = \Pr\{N = n | \underline{Y}_1, \cdots, \underline{Y}_{n-1}\} = \prod_{i \in S_n} p_i$$

로 표현된다. k번째 SMPS를 시뮬레이션한 후 네트워크의 상태가 파악되었다고 할 때, 그때까지의 랜덤 세이프티의 총합을 토탈 세이프티라 하며 토탈 세이프티 추정량 TSAFr은 네트워크 신뢰도 P_S의 불편추정량이 된다.

$$TSAFr = \sum_{j=1}^{k} t_j = \sum_{j=1}^{k} \prod_{i \in S_j} p_i$$

세이프티 임포턴스 샘플링 추정량

F($\underline{X}$)와 TSAFr의 기대치는 모두 P_S가 되므로 임포턴스 샘플링 방법에서 F($\underline{X}$) 대신에 TSAFr을 이용하여 P_S를 추정할 수 있다. 이 추정량을 세이프티 임포턴스 샘플링 추정량이라 할 때 다음 식과 같이 표현되며 마찬가지로 P_S의 불편추정량이다.

$$SIMPr = \left[\sum_{j=1}^{k} \prod_{i \in S_j} p_i\right]\left[\prod_{i \in S} \frac{p_i^{X_i}(1-p_i)^{1-X_i}}{\beta_i^{X_i}(1-\beta_i)^{1-X_i}}\right], S = S_1 \cup s_2 \cup \cdots \cup S_k$$

4. 추정량 분산 분석

시뮬레이션을 이용하여 네트워크 신뢰도를 추정하고자 할 때 그 추정치가 얼마나 정밀한가는 그 분산이 추정치에 비해 상대적으로 얼마나 적은가로 판단할 수 있으며, 이는 곧 적은 횟수의 시뮬레이션으로 상당히 정밀한 추정치를 얻을 수 있음을 나타낸다. 이 장에서는 (그림 1)의 브릿지 네트워크를 대상으로 SMCS 알고리듬과 SMPS 알고리듬으로 시뮬레이션 하였을 때 3장에서 기술한 추정량들에 대한 분산을 추정해보고, 그 결과를 토대로 SMCS 알고리듬과 SMPS 알고리듬을 비교분석해 보았다. <표 2>는 (그림 1)의 브릿지 네트워크를 대상으로 SMCS 알고리듬과 SMPS 알고리듬을 이용하여 각각 100만번의 시뮬레이션을 시행한 결과 얻어진 각 추정량들에 대한 분산을 보여주고 있는데, 여기서 모든 아크들의 고장확률은 같다고 가정하였으며 임포턴스 샘플링 방법에 근거한 추정량에서 β 값은 SMCS 알고리듬에서는 0.25를, SMPS 알고리듬에서는 0.75를 사용했다. <표 2>에서 보여준 raw 몬테칼로 시뮬레이션에 의한 네트워크 불신뢰도 추정량 RAWu의 분산은 해석적으로 구한 값이다.

우선 SMCS 알고리듬과 SMPS 알고리듬을 이용한 네트워크 불신뢰도 추정량들의 분산과 raw 몬테칼로 시뮬레이션에 의한 네트워크 불신뢰도 추정량 RAWu의 분산과 비교해 보면 <표 2>에서 보는 것처럼 두 알고리듬을 이용한 추정량들의 분산이 RAWu보다 훨씬 더 낮은 값을 보여주고 있다. 따라서 본 연구에서 제안한 시뮬레이션 기법을 이용하면 훨씬 적은 횟수의 시행으로 정밀한 네트워크 신뢰도를 얻을 수 있을 것이다. 두 알고리듬내에서 살펴보면, 네트워크 신뢰도 추정량들은 아크들의 신뢰도가 낮을 때, 네트워크 불신뢰도 추정량들은 아크들의 신뢰도가 높을 때 매우 낮은 분산값을 보여주고 있다. 그러므로 아크의 고장률에 따라 네트워크 신뢰도 추정량 또는 네트워크 불신뢰도 추정량을 적절히 선택하는 것이 바람직할 것이다. 한편, 기존의 임포턴스 샘플링 추정량 IMPu와 SMCS 또는 SMPS 알고리듬을 이용한 임포턴

스 샘플링 추정량 CIMPu와 PIMPu의 분산을 비교해보면, 본 연구에서 제안한 알고리듬을 이용한 추정량이 상대적으로 더 낮은 분산감소 효과를 보여주고 있는데 이것은 CIMPu와 PIMPu가 각각 IMPu와 그 평균이 1인 독립변수와의 곱의 형태로 표현되기 때문이다. 전체적으로, 네트워크 불신뢰도를 추정할 경우 SMCS 알고리듬에 의한 토탈 해저드 추정량 THAZu가, 네트워크 신뢰도를 추정할 경우 SMPS 알고리듬에 의한 토탈 세이프티 추정량 TSAFr이 가장 낮은 분산을 보여준다. 특히, 각각 아크의 고장률이 낮을수록 또는 아크의 고장률이 높을수록 좋은 분산감소 효과를 보여준다.

<표 2> 브릿지 네트워크의 신뢰도/불신뢰도 추정량의 분산
(β=0.25(SMCS), 0.75(SMPS))

추정량 / 아크 신뢰도	RAW_u	IMP_u	SMCS 알고리듬				SMPS 알고리듬			
			$CIMP_u$	$CIMP_r$	$THAZ_u$	$HIMP_u$	$PIMP_r$	$PIMP_u$	$TSAF_r$	$SIMP_r$
0.001	1.0	5.07e+4	1.50e+1	2.99e-16	2.00e-3	1.49e+1	7.30e-16	9.82e+2	1.00e 16	5.54e 15
0.01	0.999	4.39e+4	1.44e+1	2.83e-10	1.97e-2	1.38e+1	6.81e-10	8.98e+2	1.08e 8	5.00e 9
0.1	0.997	9.98e+3	1.05e+1	1.64e-4	0.178	8.559	3.39e 4	3.69e+2	1.32e 4	1.72e 3
0.2	0.979	1.80e+3	7.800	5.76e-3	0.276	6.227	9.84e-3	1.35e+2	2.46e-3	3.34e-2
0.3	0.924	3.09e+2	5.694	3.52e-2	0.275	4.033	4.96e-2	4.68e+1	1.22e-2	0.120
0.4	0.823	5.21e+1	3.807	9.97e-2	0.206	2.062	0.119	1.53e+1	3.27e-2	0.222
0.5	0.672	9.040	2.183	0.171	0.123	0.789	0.179	4.701	5.88e-2	0.282
0.6	0.487	1.709	0.99998	0.192	5.88e-2	0.211	0.187	1.341	8.54e 2	0.264
0.7	0.296	0.357	0.336	0.159	2.22e 2	3.51e-2	0.156	0.347	0.121	0.184
0.8	0.136	7.51e-2	7.09e-2	0.248	5.66e 3	4.23e-3	0.240	7.33e+2	0.181	0.135
0.9	3.30e-2	8.78e-3	5.42e 3	1.049	4.70e-4	5.53e-4	0.857	6.92e-3	0.218	0.325
0.99	3.04e-4	2.30e-6	8.11e-7	4.001	5.92e-8	1.71e-7	2.823	1.32e 6	4.62e-2	2.479
0.999	3.01e-6	2.57e-10	8.56e-11	4.563	7.00e-12	1.92e-11	3.176	1.43e 10	5.02e-3	3.135

5. 결론

시뮬레이션을 이용하여 크고 복잡한 대규모의 네트워크 신뢰도를 추정하고자 할 때, 분산감소기법을 이용하면 훨씬 적은 시행으로 정밀한 추정치를 얻을 수 있다. 본 연구에서 제안한 SMCS 알고리듬과 SMPS 알고리듬에 의한 추정량들이 모두 raw 몬테칼로 시뮬레이션보다 훨씬 적은 분산값을 보여 주었다. 또한 제안한 알고리듬을 이용할 경우 기존의 임포턴스 샘플링 추정량보다 상대적으로 더 나은 분산감소 효과를 보여 주었다. 토탈 해저드와 토탈 세이프티 추정량이 가장 좋은 분산감소 효과를 보여주고 있는데, 랜덤 해저드 또는 랜덤 세이프티 개념을 이용하여 네트워크 신뢰도를 추정하려면 기존의 방법은 모든 MCS를 미리 알아야 하는데, 네트워크가 크고 복잡하면 이것은 거의 불가능하므로 SMCS 알고리듬과 SMPS 알고리듬은 이를 해결해 줄 수 있는 시뮬레이션 기법이다. 결론적으로 본 연구에서 제안한 두 알고리듬은 시뮬레이션을 이용한 네트워크 신뢰도 추정량의 분산을 감소시킬 수 있을 뿐만 아니라 통신망과 같은 매우 복잡한 네트워크에 대해서도 쉽고 간편하게 적용할 수 있는 방법이라 할 수 있다.

본 연구에서는 네트워크를 구성하는 아크들의 용량(Capacity)은 모두 같다고 전제한 상태에서 단지 소스노드와 터미널노드가 연결될 확률만을 고려하였는데, 일반적으로 정보통신망이나

수송망은 각 아크들의 용량은 서로 다르다. 따라서 아크들의 용량을 고려한 네트워크의 성능신뢰도(performance reliability)를 추정하는 데에도 본 논문에서 제안한 알고리듬을 적용하는 연구가 수행되어야 할 것이다. 또한 네트워크의 상황에 따라 제안한 두 알고리듬을 혼합하여 적용하는 방법도 고려해 봄 직하다.

참고문헌

[1] Brown, M., and Ross, S.M.; "The Observed Hazard and Multicomponent Systems," *Naval Research Logistics Quarterly*, 29(3), 679-683, 1982.

[2] Fishman, G.S.; "A Monte Carlo Sampling Plan for Estimating Network Reliability," *Operations Research*, 34(4), 581-594, 1986.

[3] Jun, C.H., and Ross, S.M.; "System Reliability by Simulation : Random Hazards versus Importance Sampling," *Probability in the Engineering and Informational Sciences*, 6, 119-126, 1992.

[4] Kamat, S.J., and Riley, M.W.; "Determination of Reliability Using Event-Based Monte Carlo Simulation," *IEEE Trans. Reliability*, R-24, 73-75, 1975.

[5] Kumamoto, H., Tanaka, K., and K. Inoue; "Efficient Evaluation of System Reliability by Monte Carlo Method," *IEEE Trans. Reliability*, R-26(5), 311-315, 1977.

[6] Levy, L.L., and Moore, A.H.; "A Monte Carlo Techniques for Obtaining System Reliability Confidence Limits from Component Test Data," *IEEE Trans. Reliability*, R-16, 69-72, 1967.

[7] Mazumdar, M.; "Importance Sampling in Reliability Estimation," *Reliability and Fault Tree Analysis*, R. Barlow, J. Fussel, N. Singpurwalla (eds), SIAM, Philadelphia, 153-165, 1975.

[8] Ross, S.M.; *A Course in Simulation*, MacMillan, New York, p. 136, 1990.

[9] Ross, S.M.; "Variance Reduction in Simulation Via Random Hazards," *Probability in the Engineering and Informational Sciences*, 4, 299-309, 1990.