

Probing 알고리즘을 이용한 베이지안 네트워크 기반 적응형 미들웨어의 행동 예측

이승수⁰, 김경중, 조성배
연세대학교 컴퓨터과학과

{leess⁰, uribyul, sbcho}@scslab.yonsei.ac.kr

Behavior Prediction of Adaptive Middleware based on Bayesian Networks using Probing Algorithm

Seung-Soo Lee⁰, Kyung-Joong Kim, Sung-Bae Cho
Dept. of Computer Science, Yonsei University

요약

실시간으로 변화하는 컴퓨터 통신 환경에서 멀티미디어 응용 프로그램은 QoS를 만족하기 위해 안정적으로 튜닝 되고 재구성되는 것이 필요하다. 그러나 안정적으로 QoS를 보장하는 것은 응용 프로그램의 자원 예약이나 실시간 보장과 같은 메커니즘을 제공하지 않는 일반적인 목적의 시스템 상에서 수행될 때 많은 어려움을 가지게 된다. 특히, 예측 불가능한 개방형 환경에서 최우선 자원 할당에 의해 발생하는 자원의 유효성에 대응하기 위해 QoS 적응은 수행되어야 한다. 그러나 적응을 위해, 어떻게 조정해야 하고 폭 넓은 범위에서 응용 프로그램에 어떻게 적용시킬지를 알기 위해 일반적인 알고리즘을 제시해야 할 필요가 있다. 이러한 목적을 위해, 본 논문에서는 멀티미디어 어플리케이션의 파라미터를 모델링하고, 파라미터 간의 관계를 정량적으로 얻기 위해 계층적 QoS 프로빙 알고리즘을 적용한다. 이것을 기반으로 설계된 베이지안 네트워크를 이용하여 불확실한 정보를 확률 값으로 처리함으로써 적응 행동을 예측하도록 한다. 마지막으로 실제 실험을 통해 제안된 미들웨어의 유용성을 확인한다.

1. 서론

인터넷의 보편화와 네트워크 기술의 발달로 서로 이질적인 컴퓨팅 환경에서 시스템 간에 데이터를 공유하고 관리하는 분산 시스템에 대한 연구가 활발하게 진행되고 있다. 이러한 분산 환경에서 실시간으로 멀티미디어 서비스를 제공받는 응용 프로그램들은 수행시 서비스의 저하를 막기 위해 처리 능력, 전송 지연, 정확성 및 신뢰성 등과 같은 속성으로 구성된 서비스 품질(QoS: Quality of Services)을 보장하도록 요구하고 있다. 그러나 예측할 수 없는 트래픽이 발생하는 환경에 안정적으로 대처하기 위해서는 적절한 적응 행동을 결정하는 것이 중요하다. 단순히 시스템 레벨이나 어플리케이션 레벨 등의 단일 레벨에서 적응 행동을 결정하기 보다는 두 레벨 간의 상호 협력을 통해 적응 행동을 선택하는 것이 가장 효과적인 방법이다.

본 논문에서는 어플리케이션 레벨의 QoS 파라미터를 안정적으로 유지하기 위해 프로빙 알고리즘(Probing Algorithm)을 이용한 베이지안 기반 미들웨어 시스템을 제안한다. 베이지안 네트워크는 확률 기반의 모델로써 불확실하고 변화하는 환경에 적응적이고 유연한 동작을 제공한다. 본 논문에서 제안한 미들웨어는 시스템 레벨의 CPU 및 네트워크 대역폭 정보와 응용 프로그램 레벨의 트래킹 정밀도, 프레임 전송률과 같은 파라미터 정보를 사용하여 불확실한 환경에서 적응 행동을 예측한다. 특히, 응용 프로그램에서 사용된 각 파라미터 간의 관계는 정량적으로 분석됨으로써 베이지안 네트워크 설계시 이를 적용하여 시스템의 최적화된 성능을 유도한다. 본 논문에서는 각 파라미터의 상태를 관찰하여 정량적인 관계를 얻기 위해 계층적 프로빙 알고리즘을 사용한다[1]. 프로빙 알고리즘은 각 파라미터에 대한 상태를 관찰하기 위한 방법으로 많은 연구에서 제안되었다. 그림 1은 본 논문에서 제안한 미들웨어 구조를 보여주고 있다.

2. 관련 연구

컴퓨터 통신 네트워크 환경에서 QoS를 보장하기 위한 기존의 연구들은 결정적 자원 예약 메커니즘이나 스케줄링 알고리즘을 기반으로 수행되기 때문에 응용 프로그램 레벨에서 요구하는 속성에 대한 사전 지식을 필요로 한다. 이러한 기존 방법들은 대역폭의 예약 방법과 유무선 네트워크상에서 패킷 스케

줄링 알고리즘, CPU 예약 메커니즘 등이 있다[1]. 이러한 연구의 목적은 시스템 레벨에서 통계적인 방법으로 QoS를 보장하는 것이다. 그러나 이질적인 네트워크 환경에서 발생하는 문제를 해결하기 위해 시스템 운영체제와 네트워크 프로토콜에 대한 수정은 부가적으로 요구된다.

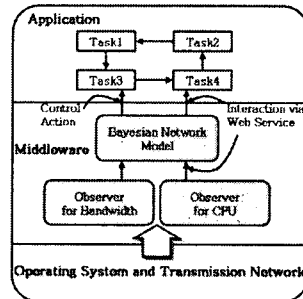


그림 1. 미들웨어 컨트롤 프레임워크 컴포넌트

2.1 자원 관리

시스템 레벨의 자원 관리 메커니즘은 변화하는 환경에 적응하기 위해 개발되었다. 특히, 무선 네트워크와 모바일 컴퓨팅 연구에서 자원의 부족이나 네트워크 오류로 인한 문제를 해결하기 위한 방법이 사용되었다[2]. 이 연구에서 서비스는 여러 개의 서비스 클래스로 분류되고, 각 자원의 예약을 예측하며 네트워크상에서 최소한의 손실을 갖는 적응 행동을 선택하기 위한 적응 개념을 제안하였다. 다른 연구에서는 모바일 환경의 응용 프로그램을 적용시키기 위해 시스템 레벨과 응용 프로그램 레벨을 분류하여 구축함으로써, 데이터의 신뢰성과 시스템 성능을 tradeoff하여 QoS의 균형을 유지하는 방법을 제안하였다[3].

2.2 프로빙 알고리즘

응용 프로그램 레벨에서 안정적으로 QoS를 유지하기 위해 내부 파라미터를 감시하는 프로빙 메커니즘을 이용한 연구들이 진행되고 있다. Al-Shaer et al.[4]는 분산 멀티미디어 시스템을 감

시하기 위한 모니터링 구조를 제안하였으며, Abdelzahr[5]는 선형 실행 모델을 기반으로 QoS-인식 웹 서버에서 시스템 파라미터를 평가하기 위해 제곱추정량(Square Estimator)기법을 사용하였다. 본 논문에서는 어플리케이션 레벨로부터 적용 행동 과정을 분리하고 오프라인 알고리즘을 통해 어플리케이션 레벨의 내부 파라미터를 검사하기 위해 프로빙 알고리즘을 사용한다. 또한, 오프라인 프로빙 알고리즘의 결과로서 획득된 데이터를 이용하여 베이지안 네트워크의 학습 데이터로 이용한다.

2.3 베이지안 네트워크

베이지안 네트워크는 변수들 간의 원인과 결과 관계를 확률적으로 모델링하기 위한 도구로서 불확실한 환경에서 좀 더 신뢰성 있는 결과를 추론하기 위해 쓰이는 대표적인 방법이다. 변수의 확률적 인과관계를 네트워크로 구성한 다음 특정 조건이나 증거가 주어진 경우의 확률, 즉 조건부 확률을 복합적으로 계산하여 결과를 추론한다.

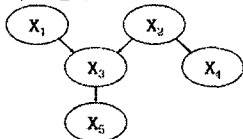


그림 2. 간단한 베이지안 네트워크 구조의 예

그림 2는 간단한 베이지안 네트워크의 예를 표현하고 있다. 베이지안 네트워크는 노드와 에지, 그리고 조건부 확률 테이블로 구성된다. 노드는 변수를, 방향성 에지는 부모에서 자식으로의 인과관계를 표현한다.

$$P(x_i | x_j) = \frac{P(x_i, x_j)P(x_j)}{P(x_i)} \quad (1)$$

$$P(x_1, x_2, x_3, x_4, x_5) = P(x_1)P(x_2)P(x_3 | x_1, x_2)P(x_4 | x_3)P(x_5 | x_3, x_4) \quad (2)$$

노드의 부모와 확률 값이 정해지면 조건부 확률 테이블(CPT: Conditional Probability Table)을 가지고 식 (1)과 같은 베이지안 규칙(Bayes's rule)을 이용해 계산을 하게 되는데, 관측 정보가 불완전하다더라도 안정적 추론이 가능하고 증거변수를 토대로 원인을 역추론 할 수 있다. 그림 2에서 결합 확률 값인 $P(x_1, x_2, x_3, x_4, x_5)$ 는 변수 독립성 가정과 체인 규칙을 이용하여 식 (2)와 같이 계산된다.

3. 프로빙 알고리즘을 이용한 데이터 획득

3.1 QoS 파라미터의 정의 및 분류

그림 3은 시스템 자원과 QoS 파라미터 간의 관계를 설명하고 있다. 각각의 컴포넌트는 입력 파라미터로써 Q_i^{in} 을 받으며, 출력 파라미터 Q_i^{out} 을 생성한다. 두 개의 파라미터는 어플리케이션 수행 시 요구되는 QoS 파라미터를 의미한다. 또한, 입력을 처리하고 출력을 생성하기 위해서는 시스템으로부터 자원 R을 제공 받아야 하며, 이러한 파라미터들은 벡터로서 표현된다.

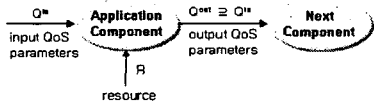


그림 3. 어플리케이션 컴포넌트 간의 관계

각 파라미터는 다음과 같이 표현할 수 있다.

$$R = [R_1, R_2, \dots, R_m]^T$$

$$Q^{in} = [p_1^{in}, p_2^{in}, \dots, p_n^{in}]^T$$

$$Q^{out} = [p_1^{out}, p_2^{out}, \dots, p_k^{out}]^T$$

그림 3에서 설명한바와 같이 입력 QoS 파라미터를 제어할 경우 그에 대한 결과로서 출력 QoS 파라미터가 변경될 것이다. 즉, 출력 QoS 파라미터는 입력 QoS 파라미터에 종속적이며, 자원 또한, 입력 QoS 파라미터로써 정의될 수 있다.

3.2 QoS 파라미터 기반 그래프 모델 설계

프로빙 알고리즘을 수행하기 전에 어플리케이션 레벨에서 필요한 QoS 파라미터들의 관계를 정의한다. 정의된 QoS 파라미터의

관계는 방향성 비순환 그래프(DAG: Directed Acyclic Graph) 형태로 구성된다. 이러한 구성은 연산의 복잡도를 줄이고 두 파라미터 사이의 정량적인 데이터를 얻기 위한 전처리 과정이다. 그림 4는 본 논문에서 구현한 트래킹 시뮬레이터에 대해 각 파라미터 간의 관계를 정의한 그래프 모델이다.

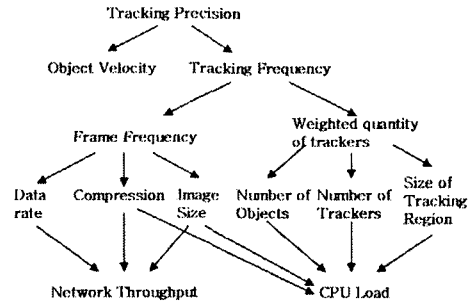


그림 4. QoS 파라미터를 위한 그래프 모델

3.3 계층적 QoS 프로빙 알고리즘

QoS 프로빙 알고리즘의 목적은 어플리케이션의 성능과 안정성을 향상시키기 위해 입력 파라미터의 값을 조절함으로써 출력 파라미터의 변화되는 값을 관찰하여, 각 QoS 파라미터 간의 관계를 고려한 데이터를 추출하는 것이다. 본 논문에서는 계층적 QoS 프로빙 알고리즘을 이용하여 QoS 파라미터의 데이터를 추출하고 추출된 데이터를 학습하여 베이지안 네트워크 모델을 생성한다. 그림 5는 계층적 QoS 프로빙 알고리즘에 대한 처리 과정을 보여주고 있다.

```

for each critical parameter  $p_i^{out}$  and its associated Tree( $p_i^{out}$ )
for level  $j=1, j \leq \text{depth}(\text{Tree}(p_i^{out}))$ ,  $j++$  (from bottom to top level)
for each node  $p_k^j$  at level  $j$ 
 $d := d(p_k^j)$  //outdegree
assume its children are  $p_{k1}^{j+1}, p_{k2}^{j+1}, \dots, p_{kd}^{j+1}$ 
list :=  $\{p_k^j\}$  // observation list
for each child  $p_{kw}^{j+1}, 1 \leq w \leq d$ 
if  $p_{kw}^{j+1}$  is a leaf and is not a resource parameter then
Value( $p_{kw}^{j+1}$ ) iterates from  $(p_{kw}^{j+1})_{\min}$  to  $(p_{kw}^{j+1})_{\max}$ , step  $(p_{kw}^{j+1})_{\text{increment}}$ 
else if  $p_{kw}^{j+1}$  is a leaf and is a resource parameter then
if  $p_k^j \in \text{list}$  then
list := list -  $\{p_k^j\}$ 
list := list -  $\{p_{kw}^{j+1}\}$ 
Value( $p_k^j$ ) iterates from  $(p_k^j)_{\min}$  to  $(p_k^j)_{\max}$ , step  $(p_k^j)_{\text{increment}}$ 
else
list := list -  $\{p_{kw}^{j+1}\}$ 
else //non-leaf node
search the dependency log and find the stored value set for  $p_{kw}^{j+1}$ 
Value( $p_{kw}^{j+1}$ ) iterates element in the stored value set for  $p_{kw}^{j+1}$ 
log observed values for parameters in list
    
```

* Tree(p_i^{out}): Subtree with root as p_i^{out}

그림 5. 계층적 QoS 프로빙 알고리즘

4. 실험 및 결과

4.1 실험 환경

본 논문에서 제안한 미들웨어의 성능을 평가하기 위해 서버-클라이언트 기반의 트래킹 시뮬레이터를 구현하였다. 서버는 이동하는 물체 이미지를 클라이언트로 전송하고, 이를 받은 클라이언트는 움직이는 물체에 대해 여러 개의 트래커를 이용하여 트래킹 작업을 수행한다. 사각형 모양의 움직이는 물체는 총 4개로, 트래커는 각 물체의 꼭지점 및 중심을 트래킹하며 한 물체에 대해 최대 7개의 트래커가 트래킹을 수행한다. 이와 같은 환경에서 계층적 QoS 프로빙 알고리즘을 이용하여 학습 데이터를 수집한다. 그림 6은 수집한 데이터를 이용하여 학습된 베이지안 네트워크 모델의 한 예를 보여주고 있다.

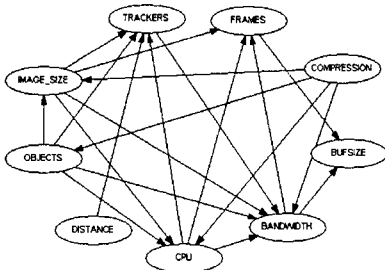


그림 6. 학습된 베이지안 네트워크 예측 모델의 예

4.2 실험 및 결과

본 논문에서 제안한 미들웨어의 성능 평가를 위해 적응 행동 모델을 적용하지 않은 어플리케이션과 퍼지 및 전문가에 의해 설계된 베이지안 네트워크, 그리고 프로빙 알고리즘을 이용해 학습된 베이지안 네트워크 등 적응 행동 모델을 적용한 어플리케이션을 비교 실험한다. 각각의 어플리케이션이 수행하는 동안 스트리밍 서비스를 이용한 동영상 프로그램을 함께 실행하여 시스템 자원(CPU, 대역폭)의 변화를 중심으로 환경의 변화에 적응하고 안정성을 유지하는 실험을 수행한다.

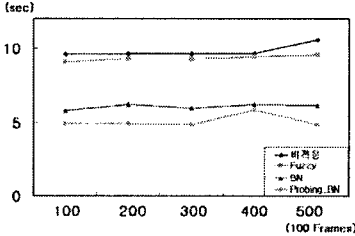


그림 7. 100~500 Frames 당 평균 전송 시간(sec)

그림 7은 비교된 실험 결과를 보여준다. 결과와 같이 전문가에 의해 설계된 베이지안 네트워크를 적용한 적응 행동 모델보다 학습을 통해 설계된 베이지안 네트워크 적응 행동 모델의 성능이 나은 것을 볼 수 있다. 이것은 대량의 학습 데이터를 통해 변화하는 상황을 종합적으로 분석한 베이지안 네트워크가 보다 안정적임으로 보여준다.

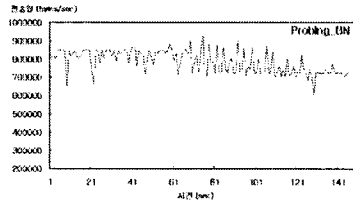
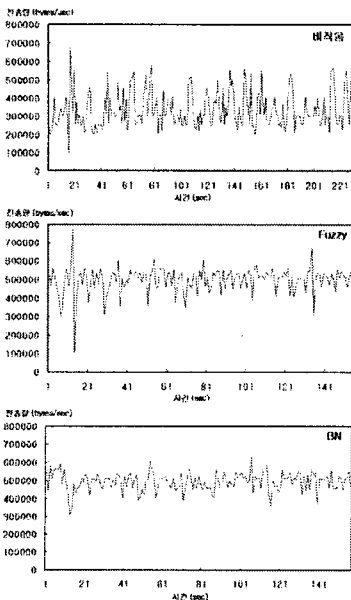


그림 8. 각 미들웨어를 적용한 어플리케이션의 데이터 전송량

그림 8은 네 가지 형태의 적응 모델을 이용하여 약 7000개의 프레임 전송하였을 때 측정된 데이터 전송량을 보여주며, 표 1은 측정된 전송량에 대해 평균 및 표준편차를 설명하고 있다. 그림 8과 표 1을 분석한 결과, 본 논문에서 제안한 적응 모델은 일정한 양의 전송량을 유지하지만, 표준편차가 다른 모델보다 높게 나타난 것은 초기에 최대 전송량으로 프레임 전송한 후 CPU의 부하 및 다른 환경적 요인에 적응하기 위하여 미들웨어의 적응 행동 모델이 동작되기 때문이다. 이것은 제안한 모델이 다른 모델에 비해 많은 양의 프레임을 비교적 안정적으로 전송하고 변화하는 환경에서 적응하는데 효과적인 방법임을 알 수 있다.

표 1. 데이터 전송량의 평균 및 표준편차 (10회 실험, 단위:KB)

	Probing_BN	BN	Fuzzy	비적용
평균	783.5	499.3	492.8	338.3
표준편차	62.9	41.8	51.9	83.9

5. 결론

본 논문의 목적은 불확실한 환경 및 한정된 자원을 이용하는 어플리케이션에서 보다 나은 서비스를 제공받기 위해 미들웨어 레벨에서 적응 행동을 선택함으로써 안정성을 유지하는 것이다. 본 논문에서는 어플리케이션의 파라미터 및 각 파라미터 간의 관계를 정의하였으며, 이를 통해 프로빙 알고리즘을 이용하여 데이터를 수집하였다. 이렇게 수집된 데이터를 기반으로 베이지안 네트워크 적응 모델을 구축하였으며, 트래킹 시뮬레이터를 이용한 비교 실험을 통해 프레임 전송량이 보다 안정적으로 유지되는 것을 보였다. 향후 연구로는 강화학습과 같은 알고리즘을 이용하여 실시간으로 새로운 데이터가 입력될 경우 베이지안 네트워크의 노드를 업데이트함으로써 불확실성을 해결하고 모델의 정확성을 높일 수 있는 연구가 필요하다.

감사의 글

본 연구는 정보통신부 및 정보통신연구진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음, IITA-2005-(C1090-0501-0019).

참고 문헌

- [1] B. Li, W. Kalter, and K. Nahrstedt, "A hierarchical quality of service control architecture for configurable multimedia applications," *In Journal of High-Speed Networks, Special Issue on QoS for Multimedia on the Internet*, IOS Press, vol. 9, pp. 153-174, 2000.
- [2] V. Bharghavan, K.-W. Lee, S. Lu, S. Ha, J. Li, and D. Dwyer, "The TIMELY adaptive resource management architecture," *IEEE Personal Communications Magazine*, vol. 5, pp. 20-31, August 1998.
- [3] B. Noble, M. Satyanarayanan, D. Narayanan, J. Tilton, J. Flinn, and K. Walker, "Agile application-Aware adaptation for mobility," *In Proceedings of the 16th ACM Symposium on Operating Systems and Principles*, vol. 31, pp. 276-287, October 1997.
- [4] E. Al-Shaer, "Active management framework for distributed multimedia systems," *Journal of Network and Systems Management*, vol. 8, pp. 49-54, March 2000.
- [5] T. Abdelzaher, "An automated profiling subsystem for QoS-aware services," *In Proceedings of Second IEEE Real-Time Technology and Applications Symposium*, pp. 208-217, 2000.