인공지능의 유전알고리즘을 이용한 네트워크의 혼잡제어 연구

송자영° 최병석*
동국대학교 영상정보통신대학원 네트워크관리학과°
동국대학교 정보통신공학과*
comnetwork°@dongguk.edu, bchoe@dongguk.edu

A Study of Congestion Control of Network using Genetic Algorithm of Artificial Intelligence

Ja-Young Song° Byeong-Seog Choe*
Dept. of Network Management, Graduate School of Image & information Technology, Dongguk University°
Dept. of Information Communication Engineering, Dongguk University*

1. 서 론

혼잡을 해결하기 위하여 라우터의 큐 상에서 패킷을 확률에 의해 무작위로 능동적으로 폐기하여 사전에 전역동기화와 같은 문제를 해결하는 기법을 통틀어 일반적으로 동적 큐 관리(AQM: Active Queue Management)라 한다. AQM의 아이디어는 혼잡상황의 발생으로 인하여 큐 오버플로우를 통해 지속적인 패킷 폐기가 일어나기 전에 그 상황을 전송 TCP에 알려서 재전송을 줄이는 것이다. AQM의 예로 RED(Random Early Detection)가 있다. RED는 그 구현의 단순함으로 인하여 매우 널리 사용되고 있으나 다양한 환경에 적용하기 위해서는 많은 어려움을 가지고 있다. 환경에 적합한 RED 매개변수에 대한 적절한 설정을 하기가 쉽지 않기 때문이다. 때문에 환경의 변화에 적응하는 변형된 형태의 RED가 많이 제안되었으며 대표적인 것으로 ARED가 있다. ARED의 경우 순간적인 트래픽의 변화에 적응하는 능력을 가지고 있으나 ARED를 운영하는데 필요한 매개변수(α, β)를 설정하는 기준이 없다.[1] 두 값에 대한 결정이 ARED의 성능에 결정적인 영향을 준다고 볼 때 두 값에 대한 설정은 역시 환경의 변화를 인지하는 관리자의 경험에 의해 수동으로 처리해야 하는 문제가 발생한다. 이것은 기타의 RED를 변형한 대부분의 기법에서 발생하는 문제라고 할 수 있다.

본 논문에서는 라우터에서 인공지능을 운영할 수 있는 환경이 마련되어 있다는 가정 하에 라우터의 출력 큐에 인공지능의 유전 알고리즘(Genetic Algorithm)을 도입하여 스스로 환경에 적응하는 AI RED를 제안하고 시뮬레이션을 통하여 일반적인 RED 운영방법과 비교함으로써 그 성능을 검증한다. AI RED의 적용을 위해서는 하나의 전제조건이 필요하다. 출력 포트에 멀티코어 마이크로프로세서(이하 CPU)와 대용량의 메모리가 탑재되어 있다는 전제조건이다. 현재 CPU 구조의 진화 패러다임은 다중스레드(Multi-thread)의 성능 향상에 초점을 맞추고 있다.[2] 2009년이면 8~32코어의 시대가 열릴 것으로 예상된다. 멀티코어 마이크로프로세서의 발전과 고성능 대용량 메모리의 발전은 지속적인 가격의 하락 속에 컴퓨팅 파워의 향상을 가지고 오고 있으며 이러한 현상을 잘 이용할 수 있는 소프트웨어의 출현은 필연적인 것이라고 할 수 있다. AI RED는 네트워크 관리와 성능의 최적화 측면에서 많은 효율성의 향상을 관리자에게 제공하여 줄 수 있다.

2. 본 론

AI RED는 다음과 같은 개념으로 동작 한다. 출력 큐에 코어의 개수가 N개인 멀티코어 CPU가 존재하며, N개인 멀티코어에서 두 개의 코어는 실제(Real)의 출력 큐에서의 처리를 담당한다. 한 개의 코어는 큐에 대한 처리를 담당하고 다른 하나의 코어는 입력 트래픽에 대한 로그 파일 작성과 가상 큐에 대한 랜덤 넘버 작성을 담당한다. AI RED의 활용 정책에 의해 실제 큐의 RED 값을 조절해야 한다는 결정이 내려지면, 나머지 코어들은 각각에 할당된 가상(Virtual)의 큐에 로그 파일을 이용하여 관리자에 의해 주어진 서비스변수를 기준으로 그 서비스변수에 적합한 RED 매개변수 값을 찾기 위해 시뮬레이션을 한다. 여기서 RED 매개변수 값(본 논문에서는 max_p, min_{th}, max_{th})의 모임은 인공지능에서 언급한 개체라 하며, max_p, min_{th}, max_{th} 각각은 유전자라 한다. 그리고 max_p와 같은 값에 존재하는 한 비트를 우리는 염색체라 한다. AI RED에서 사용할 수 있는 서비스 변수에는 패킷의 폐기율, CPU의 이용율, 평균 큐 길이 등이 있으며 본 논문에서는 논의의 단순화를 위해 CPU의 이용율과 패킷의 폐기율 두 가지를 이용한다.

랜덤 넘버에 의해 RED의 초기 값을 설정 후 그 값에 의해 시뮬레이션을 하게 되며 각각의 가상 큐에서 나온 결과와 서비스 변수를 이용하여 적합도 테스트를 한다. 그 적합도(fitness)에 순위를 매겨 우성 유전자를 가진 개체들을 찾아낸 후 우성유전자를 가지고 교배와 돌연변이를 거쳐 몇 개의 새로운 유전자를 가진 개체들을 만들어 낸다. 우성유전자를 가지는 개체와 새로운 개체 이외의 기존의 열성유전자를 가지는 개체는 모두 새로운 랜덤 값으로 설정한다. 이후 다시 시뮬레이션을 시작하며 이 과정이 M번의 세대를 거치거나 아니면 M세대 안

에 기준(기준 적합도, Standard fitness)에 어울리는 적합한 값(유전자)을 가지는 개체를 찾게 되면 실제 큐에 그 값을 적용 시킨다. 적합도를 판정하는 기준은 특별히 없다. M세대 동안 실행되는, AI RED 루프에서 실행종료의 조건이 되는, 기준 적합도는 관리자가 입력하는 서비스 변수 중 하나이며, 적합도는 폐기율이나 이용율과 같은 관리자가 입력한 서비스 변수들과 한 AI RED 개체의 실험결과에 대한 분산의 합으로 정의할 수 있을 뿐이다. AI RED에서 사용되는 유전알고리즘의 정책은 선택 연산자의 경우 순위 선택(Ranking selection)과 엘리트 보존 선택(Elitist preserving selection)을 사용하며, 교배 연산자는 일점교배(One-point crossover)를 사용한다.[3] 돌연변이율(Mutation probability)은 일반적인 기준(5%)보다 높은 20% 정도를 사용한다.

AI RED의 활용 정책은 두 가지의 정책을 사용할 수 있다. 첫째, 완전 자동화 정책(Full-automation policy)이다. 완전자동화 정책은 관리자가 설정해 놓은 서비스 변수를 기반으로 스스로 알아서 작동하는 정책을 의미한다. 즉 항상 주어진 매개변수를 기반으로 환경에 맞추어 RED 매개변수를 최대한 최적화 시켜 동작하는 방법이다. 관리자는 라우터를 설정할 때에 자신이 완전자동화 정책을 사용한다고 선택할 경우 두 가지 모드 중 하나를 선택할 수 있다. 하나는 보통모드(Normal mode)이고 다른 하나는 혼잡모드(Congestion mode)이다.

[표 1] Floyd의 RED 매개변수 권고치

max_p	0.1
min_{th}	5 packet
max_{th}	15 packet
w_q	0.002
Queue Length	25 packet

보통모드는 [표 1]의 Floyd가 제안한 RED 매개변수 값으로 동작하는 모드를 의미하고 혼잡모드는 AI RED 알고리즘이 동작하여 RED 매개변수 값을 세팅하고 그 값으로 실제 RED 큐가 동작하는 모드를 의미한다. 보통모드에서 폐기율이 X보다 커지면 AI RED 동작모드로 진입하게 된다. 이후 AI RED에서 적용한 값으로 동작하다가 폐기율이 Y보다 작아지면 [표 1]의 매개변수 값으로 다시 설정하여 동작하는 보통 모드로 동작하게 된다. 이 폐기율에 관한 통계적 계산은 라우터에서 로그 기록을 이용하여 관리자의 설정에 따라 주기적으로(예를 들면 1시간, 1일, 또는 1주일) 실행 한다. 관리자는 혼잡모드로 진입하는 폐기율 기준인 X와 혼잡모드에서 보통모드로 진입하게 되는 폐기율 Y 값에 대한 기준을 설정한다. (예: X를 0.135 Y를 0.03이라 설정) 이 값에 따라 혼잡모드에서 동작할 서비스 변수 값(예: CPU 이용율 0.6, 폐기율 0.05, 적합도 0.6)을 설정해 놓은 후 라우터를 동작시킨다. 이후에는 자동으로 RED 값이 갱신되어 동작하게 된다. 완전 자동화 정책에서 큐는 관리자가 설정한 혼잡상황(예: X>0.135)에서 주어진 부하에 최대한 적응하려 노력하게 된다.

둘째, 반자동화 정책(Semi-automation policy)이다. 반자동화 정책은 네트워크 환경의 변화에 따라 관리자가 인터넷을 이용하여 원격으로 라우터의 출력 큐에 대한 로그 정보 확인하고, 로그 정보에서 출력 큐 CPU의 이용율과 패킷의 폐기율 등을 확인한 후, 그 확인된 결과를 기반으로 관리자가 라우터의 서비스 변수를 설정 하는 방법이다. 서비스 변수가 설정이 되면 그 값을 최대한 만족하도록 AI RED가 작동하여 자동으로 RED 매개변수를 설정하게 된다.

제안한 AI RED의 시뮬레이션 결과에 따라 [표2]와 같은 RED 개체의 유전자 데이터를 얻을 수 있었으며 [표 3]과 같은 실험 결과를 얻을 수 있었다. G의 자료를 기준으로 CPU 이용율은 14.5% 향상되었으며, 패킷의 폐기율은 11.2% 더 낮아져 있다. 어떠한 경우이던지 AI RED가 자동으로 찾아낸 RED 매개변수는 표준 RED 매개변수보다 관리자가 원하는 더 효율적인 서비스를 제공하여 준다는 것을 확인 할 수 있다. 더 중요한 것으로 [표 5]을 통하여 관리자가 주어진 작업부하 환경에서 확인 할 수 있는 것은 주어진 작업부하의 특성으로 인하여 CPU의 이용율을 일정 수준으로 유지하면서 폐기율을 낮추는 데는 한계가 있다는 점일 것이다. 이 경우 관리자는 폐기율이 자신이 생각하고 있는 관리의 기준에 적합하지 않다고 판단될 경우 네트워크에 대한 재설계에 근거 자료 등으로 참고 할 수 있을 것이다.

[표 2] AI RED 매개변수 산출 결과(100세대)

AI Object per Generation / item	5	10	20	40
	E	F	G	H
fitness	0.684994	0.576042	0.555189	0.593067
minth	11	10	10	13
maxth	19	14	14	15
maxp	0.611816	0.011719	0.013428	0.947266
generation for find to fitness	55	58	14	38
termination	100	58	14	38

[표 3] RED와 AI RED 실험 결과 비교

	RED Queue : standard parameter	AI RED		
		F	G	H
Average delay in queue	5.642	7.048	7.042	6.845
Average number in queue	4.874	7.019	7.115	6.801
CPU utilization	0.518	0.598	0.606	0.596
Drop percent	0.139	0.126	0.125	0.127
Time of a simulation ended	1157453.234	1004107.331	989882.314	1006461.648

3. 결 론

AI RED는 RED 성능의 최적화를 가능하게 한다. AI RED의 가장 큰 장점은 바로 자동화를 구현하고 있다는 것에 있다. 기존의 다른 알고리즘은 어떤 경우이던 매개변수에 대한 수동적 설정이 필요한 반면에 AI RED의 경우는 초기 서비스 변수 값만 설정을 해주면 이후 어떠한 설정도 필요치 않은 방법을 제공한다.

또한 AI RED의 경우 굳이 자동으로 부하에 적응하는 RED의 의미가 아니라 하더라도 네트워크의 관리자에게 제공할 수 있는 자동화 된 도구로써의 의미도 가진다.

참고문헌

[1] W. Feng, "A Self-Configuring RED Gateway", IEEE INFOCOM99, March, 1999.

[2] Jon Stokes, "Inside machine", No starch press, 2007.

[3] Alex J. Champandard, "AI Game Development", New Riders, 2004.