논문 2018-2-8

3차원 스위치 연결구조를 위한 시뮬레이터 개발

기장근*†

Development of Simulator for 3D Switch Connection Architecture

Jang-Geun Ki*†

요 약

최근 데이터 센터의 다양한 컴퓨팅 및 네트워크 자원들을 효율적으로 구성하는 방안에 대한 연구들이 활발히 이루어지고 있다. 본 논문에서는 4포트 또는 6포트를 갖는 스위치를 모델링하고 컴퓨팅 자원들을 2차원 또는 3차원으로 연결하는 시뮬레이터를 개발하고, 이에 따른 성능분석을 수행하였다. 시뮬레이터는 Mininet과 Ryu 프레임워크를 기반으로 Python 언어를 사용해 개발되었으며, 시뮬레이션을 통한 성능분석 결과 4x4x4 3차원 연결 구조가 동일한 스위치 개수를 가지는 8x8 2차원 구조에 비해 1:n 연결의 성공 개수가 최대 약 2.5배에서 1.5배 정도 많음을 보였다.

Abstract

Recently, there have been many studies on how to efficiently organize computing and network resources in data centers. In this paper, the simulator that models switches with four or six ports and connects computing resources in two or three dimensions has been developed and performed performance analysis accordingly. Python language was used to develop the simulator on the basis of the Mininet and the Ryu frameworks. Performance analysis with simulation shows that the 4x4x4 3D connection has maximum 2.5 times more successful ratio of 1:n connections than the 8x8 2D connection architecture.

한글키워드 : 스위치 3차원 연결 구조, 성능분석, 시뮬레이터

keywords : switch 3-dimensional connection architecture, performance evaluation, simulator

1. 서 론

데이터 센터(DC : Data Center)[1,2]는 서버 컴퓨터들과 네트워크 회선 등을 집중된 장소에 모아 안정적인 인터넷 또는 클라우드 서비스를 제공할 목적으로 구축되며, 최근 점점 더 많은 일반 기업 사용자들이 컴퓨팅 자원과 네트워크 자원을 직접 구축하는 것 대신에 비용절감과 운영의 효율성 등을 위해 데이터 센터로부터 필요한 만큼의 서버 및 네트워크 용량을 동적으로 임대해 사용하고 있다. 따라서 데이터 센터를 효율적으로 운영하기 위해서는 사용자의 요구에 따라 서버와 저장장소 등의 용량을 가변적이면서 또한

* 공주대학교 전기전자제어공학부
† 교신저자: 기장근(email: kjg@kongju.ac.kr)
접수일자: 2018.11.28. 심사완료: 2018.12.07.
게재확정: 2018.12.21.

동적으로 제공할 수 있는 능력을 가져야 한다.

최근 데이터 센터를 구성하는 다양한 컴퓨팅 자원들을 동적으로 구성하는 효율적 방안에 대한 연구들이 이루어지고 있으며, HPE Synergy 시스템[3], CISCO UCS M 시리즈 서버[4] 등이 일부 제한적 서비스들을 제공하고 있다. 데이터 센터의 컴퓨팅 자원들을 동적으로 연결하기 위해 Nirmal Kumbhare가 발표한 연구 결과[5]인 JITA(Just in Time Architecture) 구조에서는 광 크로스바 스위치 구조를 제안하고 이를 이용해 네 방향의 포트를 연결하는 방안을 제시하고 있으나, 1:1 연결만을 지원하여 많은 자원들을 효율적으로 연결하기가 용이하지 않다. 따라서 이와 같은 제한사항을 해결하기 위한 방안의 일환으로 본 논문의 저자는 참고문헌[6]에서 일대다 연결이 가능한 4포트 스위치를 모델링하고 이에 따른 시뮬레이션을 수행하여 성능분석을 수행한 바 있다.

본 논문에서는 위의 선행연구를 기반으로 6포트를 갖는 스위치를 3차원 적으로 연결하는 스위칭 구조를 모델링할 수 있는 시뮬레이터를 개발하고 이에 따른 시뮬레이션을 통해 성능분석을 수행하여, 2차원 구조에 비해 3차원 연결구조가 수용할 수 있는 호스트의 수가 더 많고 또한 지원할 수 있는 1:n 연결 개수가 더 많음을 보였다.

2. 스위치 모델링

참고문헌 [6]에서 제안된 스위치는 4포트를 가지면서 1:1부터 1:3까지의 연결을 지원한다. 본 연구에서는 1:5까지의 연결을 지원하는 6포트 스위치를 모델링하고, 이들을 입체적으로 연결하여 3차원 구조를 갖는 모델로 구성하였다. 각 스위치들은 중앙제어센터의 실시간 제어에 따라 경로 설정을 위한 적절한 1:n 연결기능을 각 포트마다 동적으로 제공하게 된다.

그림 1은 본 논문에서 제한하는 3차원 스위치 구조의 예로 4x4x4 스위치 연결 구조를 보여주고 있다. 그림에서 원은 연결하고자 하는 컴퓨팅 자원을 의미하고, 선은 링크를 의미하며, 선과 선이 만나는 곳에 그림 2의 (b)와 같은 스위치가 위치한다. 그림 2의 (a)는 본 연구의 선행연구로 수행된 연구[6]에서 사용되었던 2차원 스위치 구조를 나타낸다.

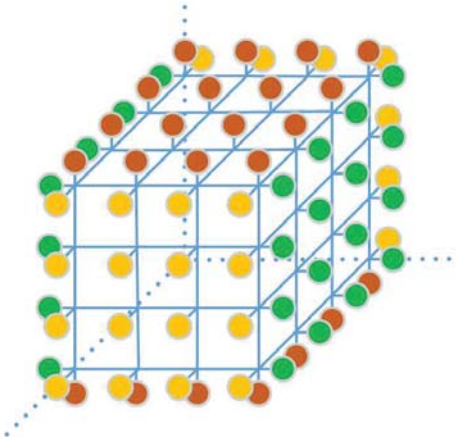
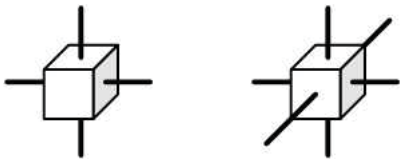


그림 1. 4x4x4 3차원 스위치 연결 구조
Fig. 1. 4x4x4 3-dimensional switch connection architecture



(a) 2-dimential switch (b) 3-dimensional switch

그림 2. 스위치 구조
Fig. 2. switch architecture

일반적으로 m*n개의 스위치로 구성된 2차원 연결구조의 경우 수용가능한 호스트 수 H_2와 전체 링크 수 L_2는 아래와 같이 계산할 수 있다.

$$H_2 = (n+m)\times 2$$

$$L_2 = (n-1)\times m + n\times(m-1) + (n+m)\times 2$$

m*n*d개의 스위치로 구성된 3차원 연결구조의 경우에 수용가능한 호스트 수 H_3와 사용되는 전체 링크 수 L_3는 다음 식으로 구할 수 있다.

$$H_3 = (m\times n + n\times d + d\times m)\times 2$$

$$L_3 = \{(n-1)\times m + n\times(m-1)\}\times d + m\times n\times(d-1) + (m\times n + n\times d + d\times m)\times 2$$

표 1에는 64개의 4포트 스위치를 사용하는 2차원 구조 경우와 64개의 6포트 스위치를 사용하는 3차원 구조 경우를 비교하여 나타내었다. 4포트 스위치 64개를 이용한 2차원 연결구조는 총 32개의 호스트(컴퓨팅 자원)를 수용할 수 있는 반면 6포트 스위치 64개를 3차원으로 연결하는 구조는 96개의 호스트를 수용할 수 있어 수용 용량이 크게 증가함을 알 수 있다.

그림 3에는 연결구조에 사용되는 스위치 개수에 따른 수용 가능한 호스트 개수를 비교하여 나타내었다. 그림에서 알 수 있듯이 스위치들을 3차원 구조로 연결하는 경우가 2차원 구조로 연결하는 경우에 비해 수용 가능한 호스트 개수가 훨씬 많음을 알 수 있다.

표 1. 스위치 구조 비교
Table 1. switch architecture comparison

구분	구조	총 스위치 수	수용 가능한 호스트 수
4포트 스위치	2차원	8x8 = 64	8x4 = 32
6포트 스위치	3차원	4x4x4 = 64	4x4x6 = 96

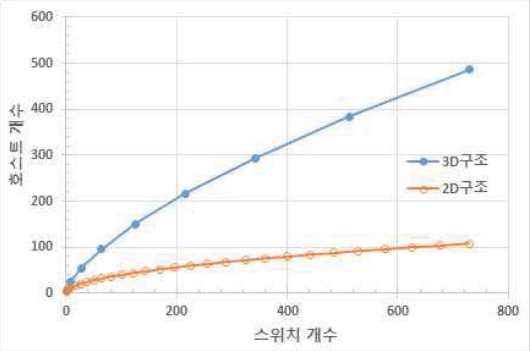


그림 3. 스위치 구조별 수용가능 호스트 수
Fig. 3. Number of affordable hosts in 3D/2D architecture

3. 시뮬레이터 개발 및 성능 분석

데이터 센터 구성 요소인 컴퓨팅 자원과 네트워크 자원의 효율적 3차원 연결구조를 시뮬레이션 하기 위해 본 연구에서 개발된 시뮬레이터 프로그램은 최근 소프트웨어 정의 망(SDN: Software Defined Network)[7] 연구분야에서 많이 사용되고 있는 Mininet[8]을 기본으로 해서 개발되었다. Mininet은 하나의 머신상에서 실제와 같은 가상 네트워크 환경을 에뮬레이션 할 수 있는 기능을 제공하고 있어서 OpenFlow와 SDN 시스템 연구분야에 많이 사용되고 있다.

본 연구에서 개발된 3차원 스위치 연결구조 시뮬레이터는 Mininet을 이용해 m*n*d의 3차원 스위치 네트워크를 생성하는 기능 뿐만 아니라 이들 스위치들의 동작을 제어하는 컨트롤러 기능을 제공한다.

시뮬레이터는 컴퓨팅 자원을 가상 호스트로 모델링하고, 이들을 연결하는 스위치와 링크들을 3차원 구조로 연결시키는 네트워크를 형성하고, 컴포넌트 기반 SDN 골격구조를 제공하는 Ryu OpenFlow 컨트롤러를 기반으로 본 논문에서 Python 언어로 개발된 스위치 제어 기능을 가지

고 있다. Ryu[9]는 잘 정의된 API를 통해 소프트웨어 컴포넌트들을 제공하여 개발자들이 새로운 네트워크 관리 및 제어 응용 프로그램을 개발할 수 있도록 지원하고 있으며, OpenFlow를 비롯해 Netconf, OF-config 등 다양한 프로토콜들을 지원하고 있다.

그림 4에 본 연구에서 개발된 3차원 스위치 연결구조를 위한 시뮬레이터에서 스위치들을 제어하는 컨트롤러 노드에 대한 주요 코드를 의사코드(pseudo code) 형태로 나타내었다. 그림에서 알 수 있듯이 3차원 스위치 연결 구조는 3차원 그래프 형태로 모델링 되었으며 그래프 이론을 이용한 최단경로 알고리즘을 이용해 마스터 호스트와 슬레이브 호스트들 사이의 최단 경로를 찾고 이 정보를 이용해 스위치들을 제어하는 형태로 구성되어 있다.

시뮬레이터 컨트롤러 노드 소프트웨어는 먼저 그래프 객체 g를 생성한 후 호스트 노드들과 스위치 노드들을 추가한다. 예를 들어 그림 1에 나타내었던 4x4x4 스위치 구조의 경우, 원으로 표시된 컴퓨팅 자원(호스트)들을 3차원 육면체의 각 면 방향에 4x4개씩 총 6면에 96개의 호스트 노드를 추가하고, 4x4x4개의 스위치 노드를 추가한다.

다음에 스위치들끼리 연결하는 링크와 스위치와 호스트 노드를 연결하는 링크들을 추가한다.

다음에는 1:n 연결구조에 대한 시뮬레이션을 수행하기 위해 서로 연결할 하나의 마스터 호스트와 n개의 슬레이브 호스트들을 랜덤하게 선택하는 과정을 수행한다. 선택과정의 효율성을 위해 호스트들의 리스트에서 호스트 순서를 랜덤하게 섞은 후 마스터 호스트 1개, 슬레이브 호스트 n개를 차례로 선택하는 과정을 반복하도록 프로그램 되었다.

마스터 호스트와 슬레이브 호스트들의 각 조합에 대해서는 최단경로 알고리즘을 수행해 최적의 경로를 설정하고 결과에 따른 통계치를 갱신한다. 마스터 호스트로부터 슬레이브 호스트들로 향하는 경로상의 모든 스위치들은 경로 설정을 위해 입력포트별로 연결되는 출력포트 리스트들을 유지 관리하는데, 새로운 경로가 설정될 때마다 그림 4의 프로그램 코드 중 33번 줄의 코드에 의해 해당 입력포트의 출력포트 리스트들을 갱신하게 된다. 그림 5는 1:3 연결이 설정된 스위치의 한 예를 보여주고 있다. 그림에서 5번 포트로 입력되는 정보는 1,2,3번 포트로 전달되며, 1,2,3번 포트로 입력된 정보는 5번 포트로 전달된다.

```
1   // 그래프 객체 g 생성
2   g = Graph(row, col, dep);
3
4   // 그래프에 6방향에 연결되는 호스트 추가
5   g.add_vertex(hostname);
6
7   // 그래프에 row*col*dep 개수의 스위치 추가
8   g.add_vertex(swname);
9
10  // 스위치 사이의 링크 추가 및
11  // 호스트와 스위치 사이의 링크 추가
12  g.add_edge(swname1, swname2, 1);
13  g.add_edge(hostname, swname2, 1);
14
15  // 전체 호스트들의 리스트 구한 후
16  // 호스트 리스트를 랜덤하게 섞음
17  host_list = g.get_host_list();
18  random.shuffle(host_list);
19
20  // 임의로 선택된 마스터 호스트와
21  // 임의로 선택된 슬레이브 호스트들 사이 연결 설정
22  loop
23  {   // 마스터 호스트 랜덤 선택
24      host_from = host_list.get_next_host(1);
25      // 선택된 마스터에 연결될 n개의 슬레이브 호스트들을 선택
26      host_to_list = host_list.get_next_host(n);
27
28      // 마스터 호스트로부터 슬레이브 호스트들로의 최단 경로 찾기
29      from_to_path = g.search_shortest_path(host_from, host_to_list);
30      if (len(from_to_path) > 0)          // 최단경로 찾았으면
31      {   // 그래프 연결정보, 스위치 상태, 통계치 갱신
32          g.update_path(from_to_path);
33          g.update_switch_status(from_to_path);
34          g.update_statistics();
35      }
36  }
```

그림 4. 3차원 연결구조 시뮬레이터 의사코드
Fig. 4. Pseudo code for 3D connection simulator

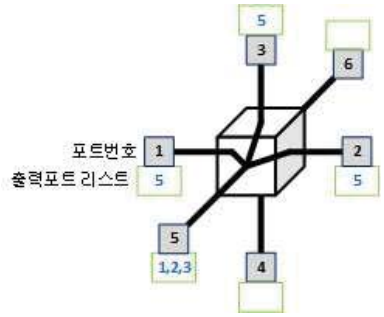


그림 5. 3차원 스위치 동작 예(1:3 연결)
Fig. 5. 3D switch operation(1:3 connection)

본 논문에서 제안한 3차원 스위치 구조와 기존의 2차원스위치 구조의 성능 분석 결과의 예로 그림 6에 1:n 연결에 따른 성공 연결 개수를 비교하여 나타내었다. 3차원 구조에서는 x, y, z 각 방향으로 4개씩 총 4x4x4=64개의 스위치를 연결하였고, 2차원 구조에서는 x, y 방향으로 각 8개씩 총 8x8=64개의 스위치를 연결하여, 비교되는 두 가지 구조 모두 동일한 64개의 스위치를 사용하도록 설정하였다. 그림 7에 나타낸 성공연결개수 비율에서 알 수 있듯이 1:n 연결의 성공 개수는 3차원 구조가 2차원 구조에 비해 최대 약 2.5배에서 1.5배 정도 큰 것을 알 수 있다.

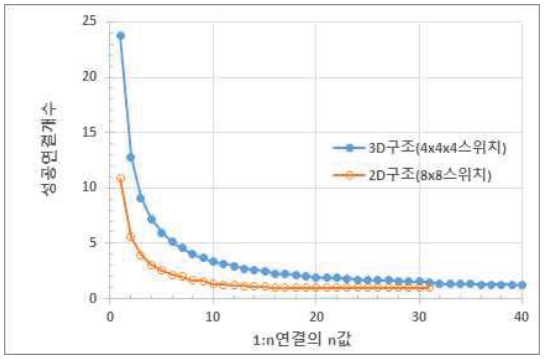


그림 6. 성공연결개수 비교
Fig. 6. Comparison of the number of successful connection

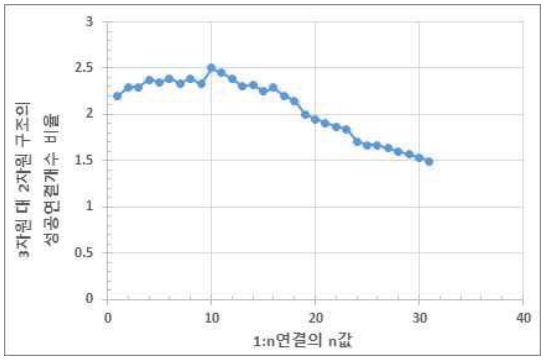


그림 7. 3차원 대 2차원 구조의 성공연결개수 비율
Fig. 7. Ratio of the number of successful connection in 3D/2D architecture

4. 결 론

최근 데이터 센터의 다양한 컴퓨팅 및 네트워크 자원들을 효율적으로 구성하는 방안에 대한 연구들이 활발히 이루어지고 있다.

본 논문에서는 선행연구[6]의 결과를 확장하여 6포트를 갖는 스위치를 모델링하고 이들을 효율적으로 연결하기 위해 3차원 연결구조를 갖도록 하는 시뮬레이션 소프트웨어를 개발하여 성능분석을 수행하였다.

본 논문에서 사용된 시뮬레이터는 SDN 연구 분야에 많이 사용되고 있는 Mininet과 컴포넌트 기반 API를 제공하는 Ryu 프레임워크를 기반으로 Python 언어를 사용해 개발되었다.

성능분석 결과의 한 예로 4x4x4 3차원 연결구조는 동일한 스위치 개수를 가지는 8x8 2차원 구조에 비해 1:n 연결의 성공 개수가 최대 약 2.5배에서 1.5배 정도 많음을 시뮬레이션을 통해 보였다.

앞으로 개발된 시뮬레이터를 이용해 보다 다양한 스위치 연결구조에 대한 심도있는 분석을 수행하고 이에 따른 성능 향상 방안을 도출할 예정이다.

참 고 문 헌

[1] Cynthia Harvey, “Data Center”, Datamation, https://www.datamation.com/data-center/what-is-data-center.html, July 10, 2017.
[2] Hwaiyu Geng, “Data Center Handbook”, Wiley & Sons, Inc., ISBN: 978-1-118-43663-9, Dec., 2014.
[3] Hewlett Packard Enterprise, “HPE Synergy”, https://www.hpe.com/us/en/integrated-systems/synergy.html, 2018.
[4] Cisco, “Cisco UCS M-Series Modular Servers datasheet”, https://www.cisco.com/c/en/us/products/collateral/servers-unified-computing/ucs-m-series-modular-servers/datasheet-c78-732667.html, 2018.
[5] Nirmal Kumbhare, Cihan Tunc, Salim Hariri, Ivan Djordjevic, Ali Akoglu, Howard Jay Siegel, “Just In Time Architecture (JITA) for Dynamically Composable Data Centers”, IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA), Nov.29-Dec.2, 2016.
[6] 기장근, 권기영, “스위치 연결구조 시뮬레이션 소프트웨어 개발”, 한국소프트웨어감정평가학회 논문지, 14권 1호, pp.41-46, ISSN 2092-8114, Jun.. 2018.
[7] Kamal Benzekki, Abdeslam El Fergougui Abdelbaki Elbelrhiti Elalaoui, “Software defined networking (SDN): a survey”, Security and Communication Networks, Vol.9, No.18, pp.5803-5833, https://doi.org/10.1002/sec.1737, Dec., 2016.
[8] Mininet, http://mininet.org/, 2018.
[9] Ryu, “Component-based software defined networking framework”, https://osrg.github.io/ryu/, 2018.

저 자 소 개

기장근(Jang-Geun Ki)

1986.2 고려대학교 전자공학과 졸업
1988.2 고려대학교 전자공학과 석사
1992.2 고려대학교 전자공학과 박사
2002.6-2003.6 Univ. of Arizona 방문교수
2010.6-2011.8 Univ. of Arizona 방문교수
2016.8-2017.8 Univ. of Arizona 방문교수
1992.3-현재 : 공주대학교 공과대학 전기전자제어공학부 교수
<주관심분야>통신프로토콜,이동통신시스템