(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2022-0071895
(43) 공개일자 2022년05월31일

(51) 국제특허분류(Int. Cl.)
G06N 3/08 (2006.01) *G06N 3/04* (2006.01)
(52) CPC특허분류
G06N 3/082 (2013.01)
G06N 3/04 (2013.01)
(21) 출원번호 10-2021-0142448
(22) 출원일자 2021년10월25일
심사청구일자 2021년10월25일
(30) 우선권주장
1020200158750 2020년11월24일 대한민국(KR)

(71) 출원인
포항공과대학교 산학협력단
경상북도 포항시 남구 청암로 77 (지곡동)
(72) 발명자
홍원기
경상북도 포항시 남구 지곡로 319, 328동 304호
유재형
서울특별시 송파구 올림픽로 135, 211동 1303호
이도영
경상북도 포항시 남구 청암로 77, 19동 509호
(74) 대리인
특허법인이상

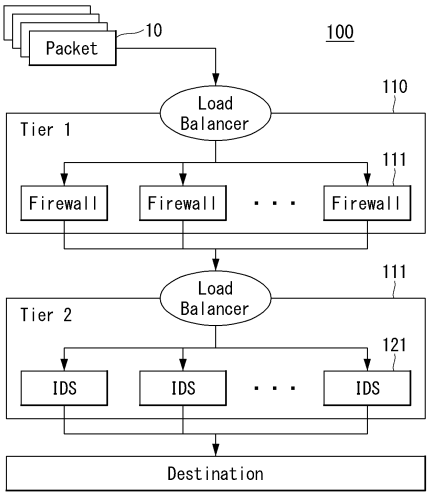
전체 청구항 수 : 총 20 항

(54) 발명의 명칭 **오토 스케일링 방법, 장치 및 시스템**

(57) 요 약

본 발명의 오토 스케일링 방법은 심층 Q 네트워크(Deep Q-networks, DQN) 장치를 통해 SFC(Service Function Chaining)를 구성하는 VNF(Virtual Network Functions) 인스턴스들의 스케일 인아웃(Scale-in/out)을 주기적으로 수행하는 오토 스케일링 방법에 있어서, 심층 Q 네트워크(Deep Q-networks, DQN) 장치를 통해 SFC(Service Function Chaining)를 구성하는 계층(Tier)들의 상황(Status)을 강화학습의 상태(State)로 정의하여 입력 값으로 받아들인 후, 어떤 물리 서버에서 스케일 인아웃(Scale-in/out)을 수행할지 또는 현재 VNF(Virtual Network Functions) 인스턴스들을 유지(Maintain)할지를 행동으로 출력하는 단계 및 스케일 인아웃(Scale-in/out)을 수행할 때, 스케일링이 필요한 SFC(Service Function Chaining)의 계층을 선택하여 해당 계층에 스케일링을 적용하는 단계를 포함한다.

대 표 도 - 도1



이 발명을 지원한 국가연구개발사업

과제고유번호	1711102868
과제번호	2018-0-00749-003
부처명	과학기술정보통신부
과제관리(전문)기관명	정보통신기획평가원
연구사업명	방송통신산업기술개발
연구과제명	인공지능 기반 가상 네트워크 관리기술 개발
기 여 율	1/1
과제수행기관명	포항공과대학교 산학협력단
연구기간	2020.01.01 ~ 2020.12.31

명 세 서

청구범위

청구항 1

오픈 스택(OpenStack)으로 구성된 NFV(Network Function Virtualization) 환경을 구현하는 네트워크 기능 가상화 서버; 및

NFV 환경에서 SFC의 오토 스케일링을 강화학습 알고리즘 중 하나인 심층 Q 네트워크(Deep Q-networks, DQN)로 수행하는 심층 Q 네트워크(Deep Q-networks, DQN) 장치; 를 포함하는,

오토 스케일링 시스템.

청구항 2

청구항 1에 있어서, 심층 Q 네트워크(Deep Q-networks, DQN) 장치는,

SFC(Service Function Chaining)의 오토 스케일링 문제를 해결하기 위한 Q-network, Target Q-network와 Replay Memory; 을 포함하는,

오토 스케일링 시스템.

청구항 3

청구항 1에 있어서, 상기 시스템은,

SFC(Service Function Chaining)를 구성하는 각 계층의 상황(Tier Status) 정보를 입력으로 받아 어느 위치(Place)에서 스케일링을 수행할지 결정하는,

오토 스케일링 시스템.

청구항 4

청구항 1에 있어서, 상기 시스템은,

심층 Q 네트워크(Deep Q-networks, DQN) 장치를 통해 특정 상태에서 출력한 스케일링 행동을 수행할 때 얻게 되는 보상 값의 모델을 수학식 2를 통해 SFC(Service Function Chaining)의 SLO(Service Level Objectives) 값 대비 트래픽 평균 응답 시간, 가상 네트워크 기능(Virtual Network Functions) 인스턴스 분포도, SFC(Service Function Chaining)을 구성하는 가상 네트워크 기능(Virtual Network Functions) 인스턴스들이 배치된 물리 서버 개수에 따라 정의하고,

SFC(Service Function Chaining)를 구성하는 VNF(Virtual Network Functions) 인스턴스들이 적은 물리 서버 위에 밀집된 형태로 배치되어 있는지 여부와 SFC(Service Function Chaining)의 성능(트래픽 응답 시간)을 고려하여 보상 값을 정의하는,

오토 스케일링 시스템.

(수학식 2)

$$\mathbf{Node}_{\mathbf{Util.}} = \frac{Node_{used}}{Node_{total}}, \mathbf{Dist}_{\mathbf{VNF}} = \prod_{i=1}^{n} \frac{VNF_{node_i}}{VNF_{total}}$$

$\mathbf{node_i}$

단, 는 최소한 1개 이상의 VNF 인스턴스가 배치된 물리 서버이고,

$$\mathbf{R_{t+1}} = -\frac{\boldsymbol{resTime}}{\boldsymbol{SLO}} + \boldsymbol{\alpha}\mathbf{Dist_{VNF}} \times \boldsymbol{e^{-\beta Node_{Util.}}}$$

청구항 5

청구항 4에 있어서, 상기 시스템은,

SFC(Service Function Chaining)를 구성하는 가상 네트워크 기능(Virtual Network Functions) 인스턴스들이 적은 물리 서버 위에 밀집된 형태로 배치되어 있는지 여부와 SFC(Service Function Chaining)의 성능인 트래픽 응답 시간을 고려하여 효과적으로 보상 값을 정의하는,

오토 스케일링 시스템.

청구항 6

청구항 1에 있어서, 상기 시스템은,

수학식 3을 통해 오토 스케일링 대상이 되는 SFC(Service Function Chaining)의 계층을 선택하는,

오토 스케일링 시스템.

(수학식 3)

$$\mathbf{score} = \mathbf{mask} \times \mathbf{f(Tier_i)}$$

$$\mathbf{resUtil} = \boldsymbol{\alpha}\mathbf{CPU_{Util.}} + \boldsymbol{\beta MEM_{Util.}}$$

$$\mathbf{f(Tier_i)} = \begin{cases} \dfrac{\boldsymbol{Node_{tier}}}{\boldsymbol{Node_{used}}} \times \boldsymbol{e^{-resUtil}}, & \boldsymbol{if\ scale-in} \\ \dfrac{\boldsymbol{Node_{used}}}{\boldsymbol{Node_{tier}}} \times \boldsymbol{e^{resUtil}}, & \boldsymbol{else\ if\ scale-out} \end{cases}$$

청구항 7

심층 Q 네트워크(Deep Q-networks, DQN) 장치를 통해 SFC(Service Function Chaining)를 구성하는 VNF(Virtual Network Functions) 인스턴스들의 스케일 인아웃(Scale-in/out)을 주기적으로 수행하는 오토 스케일링 방법에 있어서,

심층 Q 네트워크(Deep Q-networks, DQN) 장치를 통해 SFC(Service Function Chaining)를 구성하는 계층(Tier)들의 상황(Status)을 강화학습의 상태(State)로 정의하여 입력 값으로 받아들인 후, 어떤 물리 서버에서 스케일 인아웃(Scale-in/out)을 수행할지 또는 현재 VNF(Virtual Network Functions) 인스턴스들을 유지(Maintain)할지를 행동으로 출력하는 단계; 및

스케일 인아웃(Scale-in/out)을 수행할 때, 스케일링이 필요한 SFC(Service Function Chaining)의 계층을 선택하여 해당 계층에 스케일링을 적용하는 단계; 를 포함하는,

오토 스케일링 방법.

청구항 8

청구항 7에 있어서, 상기 방법은,

에이전트가 심층 Q 네트워크(Deep Q-networks, DQN) 장치의 안정적인 학습을 위해 큐 네트워크(Q-network) 및 타겟 큐 네트워크(Target Q-network), 리플레이 메모리(Replay Memory)를 활용하는,

오토 스케일링 방법.

청구항 9

청구항 7에 있어서, 상기 방법은,

심층 Q 네트워크(Deep Q-networks, DQN) 장치가 큐-러닝(Q-learning)의 특정 상태에서 행동을 수행할 때 얻을 수 있는 보상을 예측하는 지표인 큐 밸류(Q-value)를 반복적으로 학습하고,

학습된 큐 밸류(Q-value)는 특정 상태에서 어떤 행동을 수행할지 결정하는 정책으로 사용하는,

오토 스케일링 방법.

청구항 10

청구항 7에 있어서, 상기 방법은,

심층 Q 네트워크(Deep Q-networks, DQN) 장치가 학습을 통해 특정 상태에서 수행할 행동을 출력하는 큐 네트워크(Q-network)의 네트워크 파라미터를 갱신하고,

학습된 큐 네트워크(Q-network)는 최적의 스케일링 행동을 수행하는 최적 정책으로 사용되는,

오토 스케일링 방법.

청구항 11

청구항 7에 있어서, 상기 방법은,

심층 Q 네트워크(Deep Q-networks, DQN) 장치가 큐 네트워크(Q-network)와 타겟 큐 네트워크(Target Q-network)를 생성하고,

수학식 1의 손실 함수(Loss function) 값을 최소화하는 형태로 네트워크 파라미터를 학습하고,

수학식 1은 심층 Q 네트워크(Deep Q-networks, DQN) 장치에서 학습을 위해 일반적으로 사용되는 손실 함수이며

$E_{s,a,r,s'}$

리플레이 메모리(Replay Memory)에 저장된 데이터 를 학습 데이터로 입력받고,

θ_i^-

타겟 큐 네트워크(Target Q-network)(네트워크 파라미터)에서 얻을 수 있는 최대 큐 밸류(Q-value)와 큐 네

θ_i

트워크(Q-network)(네트워크 파라미터)의 Q-value의 차이를 줄이는 방향으로 Q-network를 학습하고,

일정 횟수 이상 큐 네트워크(Q-network)를 학습하면 큐 네트워크(Q-network)의 네트워크 파라미터를 타겟 큐 네트워크(Target Q-network)로 복사하는,

오토 스케일링 방법.

(수학식 1)

$$L_i(\theta_i) = E_{(s,a,r,s')}[(r + \gamma max Q(s', a'; \theta_i^-) - Q(s, a; \theta_i))^2]$$

청구항 12

청구항 7에 있어서, 상기 방법은,

심층 Q 네트워크(Deep Q-networks, DQN) 장치가 학습 과정에서 수행한 행동에 대한 보상 값(r)이 반영되는 수학식 2의 보상 모델로 정의되는,

오토 스케일링 방법.

(수학식 2)

$$\mathbf{Node_{Util.}} = \frac{Node_{used}}{Node_{total}}, \mathbf{Dist_{VNF}} = \prod_{i=1}^{n} \frac{VNF_{node_i}}{VNF_{total}}$$

$\mathbf{node_i}$

단, 는 최소한 1개 이상의 VNF 인스턴스가 배치된 물리 서버이고,

$$\mathbf{R_{t+1}} = -\frac{resTime}{SLO} + \alpha \mathbf{Dist_{VNF}} \times e^{-\beta Node_{Util.}}$$

청구항 13

청구항 12에 있어서, 상기 방법은,

$resTime$

수학식 2에서 은 스케일링을 수행한 SFC(Service Function Chaining)를 통해 트래픽을 전송하고, 응답을 받을 때까지 소요되는 응답 시간(Response time)을 의미하고,

DQN 기반 오토 스케일링 방법에서는 SLO(Service Level Objectives)로 트래픽의 응답 시간을 활용하고,

$resTime$

측정된 응답 시간인 을 미리 정의된 SLO(Service Level Objectives) 대비 응답 시간이 얼마나 되는지를 비율로 환산하여 보상 값에 반영하는,

오토 스케일링 방법.

(수학식 2)

$$\mathbf{Node_{Util.}} = \frac{Node_{used}}{Node_{total}}, \mathbf{Dist_{VNF}} = \prod_{i=1}^{n} \frac{VNF_{node_i}}{VNF_{total}}$$

$\mathbf{node_i}$

단, 는 최소한 1개 이상의 VNF 인스턴스가 배치된 물리 서버이고,

$$\mathbf{R_{t+1}} = -\frac{resTime}{SLO} + \alpha \mathbf{Dist_{VNF}} \times e^{-\beta Node_{Util.}}$$

청구항 14

청구항 13에 있어서, 상기 방법은,

$Node_{Util.}$ $Node_{total}$

는 NFV 환경에서 가용할 수 있는 총 물리 서버 개수( ) 대비 SFC를 구성하는 VNF(Virtual

$Node_{used}$

Network Functions) 인스턴스가 배치되어 있는 물리 서버 개수( )의 비율이며,

$Dist_{VNF}$

는 SFC를 구성하는 VNF 인스턴스들의 분포도를 나타내는 값이며,

$Node_{Util.}$ $Dist_{VNF}$

수학식 2의 보상모델은 과 를 보상(r)에 반영하고,

$Node_{Util.}$ $Node_{total}$

여기서, 는 NFV 환경에서 가용할 수 있는 총 물리 서버 개수( ) 대비 SFC(Service Function

$Node_{used}$

Chaining)를 구성하는 VNF 인스턴스가 배치되어 있는 물리 서버 개수( )의 비율을 의미하고,

$Dist_{VNF}$

는 SFC(Service Function Chaining)를 구성하는 VNF 인스턴스들의 분포도를 나타내는 값이며,

VNF_{total}

SFC(Service Function Chaining)를 구성하는 전체 VNF 인스턴스 개수( ) 대비 VNF(Virtual Network Functions) 인스턴스가 배치된 각 물리 서버에서 실행되는 VNF(Virtual Network Functions) 인스턴스 개수

VNF_{node_i}

( )의 비율을 곱을 의미하는,

오토 스케일링 방법.

청구항 15

청구항 12에 있어서, 상기 방법은,

$Node_{Util.}$ $Dist_{VNF}$ α β

수학식 2의 보상 모델은 지수 함수에 와 를 가중치 와 로 보정하여 반영하고,

SFC(Service Function Chaining)를 흐르는 트래픽의 응답 시간을 보상(r)에 고려한,

오토 스케일링 방법.

청구항 16

청구항 7에 있어서, 상기 방법은,

현재 상태에서 스케일링이 필요하다고 에이전트가 판단했을 경우, 수학식 3에 의해 스케일링을 적용할 계층을 선택하는,

오토 스케일링 방법.

(수학식 3)

$$\mathbf{score} = \mathbf{mask} \times \mathbf{f}(\mathbf{Tier_i})$$

$$\mathbf{resUtil} = \alpha \mathbf{CPU_{Util.}} + \boldsymbol{\beta MEM_{Util.}}$$

$$\mathbf{f}(\mathbf{Tier_i}) = \begin{cases} \dfrac{Node_{tier}}{Node_{used}} \times e^{-resUtil}, & if\ scale-in \\ \dfrac{Node_{used}}{Node_{tier}} \times e^{resUtil}, & else\ if\ scale-out \end{cases}$$

청구항 17

청구항 7에 있어서, 상기 방법은,

수학식 3은 각 계층마다 점수(Score)를 계산하여, 가장 높은 점수를 가지는 계층을 스케일링 할 계층으로 선택하고,

$mask$ $f(Tier_i)$

각 계층의 점수는 와 함수 결과 값의 곱으로 계산되며,

$mask$

이 중, 는 해당 계층 내에서 스케일링이 불가능한 경우에 0, 가능한 경우에는 1을 할당하여 점수를 보정하고,

$f(Tier_i)$

는 각 계층의 함수가 스케일 인아웃(Scale-in/out)에 적합한 정도를 나타내는 함수인,

오토 스케일링 방법.

청구항 18

청구항 17에 있어서, 상기 방법은,

스케일링을 적용할 계층은 각 계층의 CPU 사용량($CPU_{Util.}$)과 메모리 사용량($MEM_{Util.}$)을 기반으로 $resUtil$ 로 정의하며, 각각 가중치 α 와 β 로 보정되고,

$f(Tier_i)$

는 현재 SFC(Service Function Chaining)를 구성하는 VNF 인스턴스들이 배치된 물리 서버의 개수($Node_{used}$)와 선택된 계층의 VNF 인스턴스들이 배치된 물리 서버의 개수($Node_{tier}$)의 비율을 고려하고,

수학식 3은 $resUtil$, $Node_{used}$, $Node_{tier}$ 와 지수 함수를 활용하여 각 계층이 스케일링에 적합한지 점수로 나타내는,

오토 스케일링 방법.

청구항 19

청구항 7에 있어서, 상기 방법은,

보상 정의에 활용되는 데이터를 가져오기 위한 모니터링 기능이 존재한다고 가정하고,

SFC(Service Function Chaining) 데이터, VNF(Virtual Network Functions) 인스턴스 설치 위치 데이터와 물리 서버 데이터는 VNF가 운영되는 NFV환경에서 제공하는 모니터링 도구를 활용하여 가져올 수 있으며,

각 VNF(Virtual Network Functions) 인스턴스의 자원 활용률은 오픈소스 모니터링 에이전트인 컬렉트(Collectd)를 설치해 주기적으로 모니터링한 후, 시계열 데이터베이스에 저장하는 것으로 확보할 수 있는,

오토 스케일링 방법.

청구항 20

청구항 7에 있어서, 상기 방법은,

성능은 임계값(Threshold) 기반 오토 스케일링 방법과 대비하여 수행한 오토 스케일링이 더 좋은 성능을 가진다는 것을 보여서 검증하고,

성능 지표로 오토 스케일링 되는 SFC(Service Function Chaining)의 SLO 위반 비율을 측정하여 활용하는,

오토 스케일링 방법.

발명의 설명

기 술 분 야

[0001] 본 발명은 오토 스케일링 방법, 장치 및 시스템에 관한 것으로, 상세하게는 다계층 구조를 갖는 서비스 펑션 체이닝(Service Function Chaining)의 스케일 인아웃(Scale-in/out)을 위한 심층 Q 네트워크(Deep Q-networks, DQN)에 기반한 오토 스케일링 방법, 장치 및 시스템에 관한 것이다.

배 경 기 술

[0002] 강화학습은 기계학습 방법 중 하나로, 에이전트(Agent)가 시행착오(Trial-and-error)를 거치며 주어진 환경

(Environment)의 현재 상태(State)에서 어떤 행동(Action)을 수행할 지 결정하는 최적의 정책(Policy)을 찾는 학습 방법이다. 이 때, 최적의 정책은 에이전트가 각 상태에서 수행하는 행동들로 인해 최대의 누적 보상(Reward)을 받을 수 있도록 한다. 강화학습은 입력 값과 그에 대한 정답 데이터가 주어지지 않더라도 입력 값과 보상 값만으로 학습을 수행할 수 있기 때문에, 동적으로 변화하는 네트워크 환경에서 효과적으로 관리 정책을 찾기 위한 방법으로 적합하다. 특히, 강화학습은 수 많은 가상 네트워크와 자원들로 구성되는 복잡한 네트워크 기능 가상화(NFV, Network Function Virtuzliation) 환경에서 VNF(Virtual Network Function)의 라이프사이클(Life-cycle) 관리 기술에 활용될 수 있다.

[0003] VNF 라이프사이클 관리 기술 중, 오토 스케일링(Auto-scaling)은 트래픽 변화에 대응하여 VNF 인스턴스(VNF이 동작하는 가상 머신 또는 컨테이너)의 자원을 동적으로 할당하는 기술이다. 오토 스케일링의 종류로는 인스턴스의 개수를 증감하는 스케일 인아웃(Scale-in/out)과 인스턴스에 할당한 컴퓨팅 자원(CPU, 메모리 등)을 조절하는 Scale-up/down이 존재하며, 오토 스케일링에서는 서비스 요구사항을 충족시킬 수 있도록 알맞은 자원을 동적으로 VNF 인스턴스에 할당하는 것이 중요하다.

[0004] 일반적으로, NFV 환경에서는 서비스 펑션 체이닝(SFC, Service Function Chaining)을 통해 일련의 네트워크 기능들을 트래픽에 적용한다. 따라서 NFV 환경에서 VNF 인스턴스를 위한 오토 스케일링은 단일 VNF 종류만 고려하는 것이 아니라 SFC를 구성하는 VNF들의 종류와 개수(SFC를 구성하는 각 계층의 상황)를 함께 고려할 필요가 있다. 따라서 NFV 환경에서의 오토 스케일링은 SFC의 오토 스케일링 문제로 정의할 수 있다. SFC의 오토 스케일링에 강화학습을 활용하기 위해서는 오토 스케일링을 적용할 SFC 계층을 선택하고, 상황에 맞는 스케일링(VNF 인스턴스의 추가 또는 제거) 을 수행하도록 상태와 행동, 보상을 정의해야 한다.

[0005] 네트워크 기능 가상화(NFV, Network Function Virtualization) 기술은 네트워크의 구성요소인 하드웨어와 소프트웨어를 분리하고, 범용 클라우드 컴퓨팅 환경에서 네트워크 기능을 가상화하여 제공하는 기술이다. 즉, 물리적인 네트워크 장치의 기능을 소프트웨어로 구현하고, VNF가 동작하는 가상 머신 또는 컨테이너와 가상 스토리지 및 가상 네트워크를 이용하여 실행하는 방식이다. NFV는 네트워크 장비 투자비와 운용비용을 절감하고, 서비스 대응 및 트래픽 변화에 신속하게 대처할 수 있는 장점이 있다. 이런 장점들로 인해 NFV는 소프트웨어 정의 네트워킹(SDN, Software-Defined Networking) 기술과 함께 5G 네트워크의 핵심 기술로 활용되고 있다.

[0006] 기계학습은 사람의 도움 없이 컴퓨터 소프트웨어가 주어진 환경을 스스로 학습하여 문제를 해결하는 방법을 말하며, 크게 지도학습(Supervised Learning), 비지도학습(Unsupervised Learning), 강화학습(Reinforcement Learning)으로 구분한다. 그 중, 강화학습은 입력 값은 주어지지만 정답에 해당하는 값은 없고, 대신 보상 값(Reward)만 주어지는 경우에 사용할 수 있는 학습 방법이다. 따라서, 강화학습은 환경에 대한 사전지식이 없어도 학습을 수행할 수 있는 장점이 있으며 순차적으로 현재 상태에서 특정 행동을 선택하는 마르코프 의사결정과정(MDP, Markov Decision Process) 문제를 해결하는 데 활용할 수 있다.

[0007] 기존의 하드웨어 기반 네트워크 장치 및 미들박스 운용환경과 달리, NFV 환경에서는 가상화된 서버와 가상 네트워크 및 스토리지를 기반으로 운용이 이루어지므로 라이프사이클 관리가 매우 복잡해진다. 즉, 트래픽이나 장애 상태에 따라 가상 서버가 생성되거나 위치를 이동하고 이에 따라 가상 네트워크의 구성도 수시로 변경되는 등 매우 복잡한 운용관리 기능을 필요로 한다. 특히, 통신사업자나 대규모 데이터센터에 모든 미들박스(Middlebox)를 NFV로 대체하는 시점에서는 수만 개 이상의 가상 서버가 수시로 위치를 변경하거나 트래픽 변화에 따라 가상 서버 자원의 수와 가상 네트워크 대역폭을 증감하는 등 실시간으로 동적인 변화가 이루어지므로 사람의 판단에 의한 운용관리가 한계에 이를 것으로 예상된다.

[0008] 복잡한 네트워크 환경의 운용 관리 문제를 해결하기 위해 기계학습 기술을 도입하여 네트워크 운용을 자동화하는 방법이 존재한다. 이를 실현하기 위해서는 기계학습을 통한 네트워크 상태 학습이 전제되어야 하며, 학습을 위한 많은 데이터가 요구된다. 네트워크에서는 네트워크를 구성하는 장치들의 자원 정보, 트래픽 정보 등 수집할 수 있는 대용량의 데이터가 존재하지만, 기계학습을 적용하는데 한계가 있다. 그 예로 기계학습 기법에 적용하기에 적합하게 표준화되거나 라벨링(Labeling) 된 데이터가 부족하며, 데이터를 수집하는 방법들은 기계학습으로 처리하기 어려운 형태로 데이터를 제공한다. 특히, 현재의 하드웨어 기반 통신장비에서 사용되는 프로토콜 및 입출력 데이터는 그 원리 및 데이터 구조가 모두 달라 기계학습 적용에 부적합한 상태이다. 따라서 기계학습 적용이 가능한 형태의 데이터 수집 및 전처리 기능이 요구된다. 뿐만 아니라, NFV 환경에서는 물리 자원뿐만 아니라 가상 자원, 네트워크 상태 정보, 트래픽 정보 등이 수집되어야 하는 등 상태 모니터링 및 분석에 대한 연구가 필요하다.

[0009] 기존의 기계학습을 활용한 대부분의 네트워크 관리 연구에서는 학습을 위한 대량의 데이터가 존재한다는 전제

조건 하에, 지도학습과 비지도학습을 이용하여 트래픽 분류(Traffic classification), 비정상 징후 탐지(Anomaly detection), 침입 탐지(Intrusion detection) 등을 주로 수행하였다. 하지만, 많은 학습 데이터를 필요로 하는 지도학습이나 비지도학습은 동적으로 변화하는 네트워크 환경에 빠르게 대응하기 어렵다는 한계가 있다. 이에 반해, 강화학습은 상태, 행동, 보상 값의 정의를 통해 네트워크 환경에서 수집되는 데이터를 즉각적으로 활용하여 최적의 관리 정책을 결정할 수 있다. 하지만, 강화학습을 네트워크 관리 자동화에 활용하려는 대부분의 연구는 미니넷(Mininet)과 같은 제한된 시뮬레이션 환경에서 SDN 어플리케이션을 개발하거나 프레임워크 구조를 제안하는 등, 강화학습 모델에 대한 논의보다는 네트워크 관리 기능 구현에 집중되어 왔다. 따라서, 강화학습을 VNF 라이프사이클 관리 기술에 효과적으로 적용하기 위해서는 상태, 행동, 보상 값을 어떻게 정의해야 할 지에 대한 연구가 필요하다.

[0010] VNF 라이프사이클 관리 기능은 특정 사건이나 서비스 요청 발생 시 이에 대한 대응행위를 표준 및 자동화 된 절차에 따라 수행해야 한다. 관리자의 편의를 위해 스케일링을 포함한 일부 VNF 라이프사이클 관리 기능을 제공하는 오픈소스 소프트웨어(ex. OpenStack)들이 이미 존재하지만, SFC의 스케일링을 위해서는 관리자가 수동으로 VNF 인스턴스들의 개수를 조절하고 SFC를 재설정해야 하는 불편함이 존재한다. 이는 동적으로 변화하는 네트워크 상황에 유연하게 대응하여 스케일링을 적용하는 것을 어렵게 하며, 비효율적인 네트워크 관리의 원인이 된다.

발명의 내용

해결하려는 과제

[0011] 상기와 같은 문제점을 해결하기 위한 본 발명의 목적은 강화학습 기반으로 SFC의 오토 스케일링을 수행하는 문제를 정의하여, 이를 위한 보상 모델(Reward model)을 제안하고 스케일링을 적용할 SFC계층을 선택하는 방법을 구현하는 것이다. 이 때, 보상 모델에 이용되는 요소들은 SFC를 통과하는 트래픽의 평균 응답 시간(Response time), SFC를 구성하는 VNF 인스턴스들의 분포도, 전체 가용 서버 대비 VNF 인스턴스를 배치하기 위해 사용된 물리 서버의 개수 비율이다. 또한, 스케일링을 적용할 SFC 계층을 선택하는 것(어떤 종류의 VNF 인스턴스를 스케일링 할지)은 각 계층 내 VNF 인스턴스들의 평균 자원 사용량(CPU, 메모리), 해당 계층의 VNF 인스턴스들을 배치하는데 활용된 물리 서버의 개수를 고려한다. 본 발명은 강화학습 알고리즘 중 하나인 심층 Q 네트워크(Deep Q-networks, DQN)을 활용하며, SFC를 구성하는 각 계층의 상태 정보를 바탕으로 SFC에 어떤 스케일 인아웃(Scale-in/out)을 수행할지 결정한다. 이 때, 어떤 물리 서버에서 스케일링을 수행할지도 결정한다.

과제의 해결 수단

[0012] 상기 목적을 달성하기 위한 본 발명의 일 실시예에 따른 오토 스케일링 방법, 장치 및 시스템은, 강화학습 알고리즘 중 하나인 심층 Q 네트워크(Deep Q-networks, DQN)을 통해 SFC를 구성하는 VNF 인스턴스들의 스케일 인아웃(Scale-in/out)을 주기적으로 수행하는 오토 스케일링을 목표로 한다. 제안하는 방법은 DQN을 통해 SFC를 구성하는 계층(Tier)들의 상황(Status)을 강화학습의 상태(State)로 정의하여 입력 값으로 받아들인 후, 어떤 물리 서버에서 스케일 인아웃(Scale-in/out)을 수행할지 또는 현재 VNF 인스턴스들을 유지(Maintain)할지를 행동으로 출력한다. 또한, 스케일 인아웃(Scale-in/out)을 수행할 때, 스케일링이 필요한 SFC의 계층을 선택하여 해당 계층에 스케일링을 적용한다. 에이전트에서는 DQN의 안정적인 학습을 위해 Q-network및 Target Q-network, Replay Memory를 활용한다.

[0013] DQN은 일반적인 Q-learning과 마찬가지로, 특정 상태에서 행동을 수행할 때 얻을 수 있는 보상을 예측하는 지표인 Q-value를 반복적으로 학습한다. 학습된 Q-value는 특정 상태에서 어떤 행동을 수행할지 결정하는 정책으로 사용한다(예를 들어, 특정 상태에서 Q-value가 가장 큰 행동을 선택). DQN은 학습을 통해 특정 상태에서 수행할 행동을 출력하는 Q-network의 네트워크 파라미터를 갱신하는데, 학습된 Q-network는 최적의 스케일링 행동을 수행하는 최적 정책으로 사용된다. 본 발명의 DQN은 Q-network와 Target Q-network를 생성하고, 수학식 1의 손실 함수(Loss function) 값을 최소화하는 형태로 네트워크 파라미터를 학습한다. 수학식 1은 DQN에서 학습을 위해 일반적으로 사용되는 손실 함수이며, Replay Memory에 저장된 데이터 $E_{s,a,r,s'}$ 를 학습 데이터로 입력받는다. 이 때, Target Q-network(네트워크 파라미터 θ_i^-)에서 얻을 수 있는 최대 Q-value와 Q-network(네트워크 파라미터

θ_i

)의 Q-value의 차이를 줄이는 방향으로 Q-network를 학습한다. 또한, 일정 횟수 이상 Q-network를 학습하면 Q-network의 네트워크 파라미터를 Target Q-network로 복사하는데, 이는 매 학습마다 Target Q-network도 같이 갱신하면 Q-network이 학습을 제대로 수행되지 않고 발산하기 때문이다.

[0014] DQN의 학습 과정에서는 수행한 행동에 대한 보상 값 r이 반영되어야 하기 때문에 수학식 2와 같이 보상 모델을 $resTime$ 정의하였다. 수학식 2에서 은 스케일링을 수행한 SFC를 통해 트래픽을 전송하고, 응답을 받을 때까지 소요되는 응답 시간(Response time)을 의미한다. 또한, DQN 기반 오토 스케일링 방법에서는 SLO(Service Level Objectives)로 트래픽의 응답 시간을 활용한다. SFC 경로를 통해 측정되는 응답 시간은 편차가 클 수 있기 때문에, 측정된 결과 값을 그대로 활용하면 보상 값에도 큰 편차가 생길 수 있다. 따라서 본 발명에서는 측정된 응답 시간인 $resTime$ 을 그대로 사용하는 것이 아니라, 미리 정의된 SLO 대비 응답 시간이 얼마나 되는 지를 비율로 환산하여 보상 값에 반영한다. 예를 들어, SLO로 50ms가 설정되어 있고, 실제 측정된 $resTime$ 가 25ms일 경우, 0.5가 반영된다.

[0015] 트래픽 응답 시간 외에도 수학식 2에서는 $Node_{Util.}$ 과 $Dist._{VNF}$ 를 보상 r에 반영한다. $Node_{Util.}$ 는 NFV 환경에서 가용할 수 있는 총 물리 서버 개수($Node_{total}$) 대비 SFC를 구성하는 VNF 인스턴스가 배치되어 있는 물리 서버 개수($Node_{used}$)의 비율을 의미한다. 반면, $Dist._{VNF}$ 는 SFC를 구성하는 VNF 인스턴스들의 분포도를 나타내는 값이다. SFC를 구성하는 전체 VNF 인스턴스 개수(VNF_{total}) 대비 VNF 인스턴스가 배치된 각 물리 서버에서 실행되는 VNF 인스턴스 개수(VNF_{node_i})의 비율을 곱하여 계산한다. $Dist._{VNF}$ 는 VNF 인스턴스들이 적은 수의 물리 서버에 밀집되어 배치되면 높은 값을 가지게 되고, 많은 서버들에 분산 배치되어 있을 경우 작은 값을 가지게 된다. SFC의 각 계층 내 VNF 인스턴스들은 로드 밸런서(Load-balancer)를 통해 트래픽을 분산 받기 때문에, SFC를 구성하는 VNF 인스턴스들이 많은 물리 서버에 분산 배치되어 있을 경우 트래픽 또한 해당 물리 서버들로 분산된다. 결국, 각 계층에 속한 VNF 인스턴스들이 크게 분산되어 있다면, 동일한 SFC가 적용되는 트래픽의 패킷 전달 시간과 응답 시간의 편차가 클 수 있다.

[0016] 본 발명에서 제안하는 수학식 2의 보상 모델은 지수 함수에 $Node_{Util.}$ 와 $Dist._{VNF}$ 를 가중치 α 와 β 로 보정하여 반영하고, SFC를 흐르는 트래픽의 응답 시간을 보상 r에 고려한다. 따라서 수학식 2로 계산되는 보상 값은 스케일링 행동으로 갱신된 SFC를 통과하는 트래픽의 응답 시간이 짧고, SFC를 구성하는 VNF 인스턴스들이 적은 물리 서버에 밀집된 형태로 배치되었을 경우 큰 값을 가지게 된다.

[0017] 본 발명의 오토 스케일링 방법은 다계층으로 이루어 진 SFC를 대상으로 하기 때문에 스케일링을 적용할 특정 계층을 선택하는 것이 필요하다. 따라서 현재 상태에서 스케일링이 필요하다고 에이전트가 판단했을 경우, 수학식 3에 의해 스케일링을 적용할 계층을 선택한다. 본 발명에서 제안하는 수학식 3은 각 계층마다 점수(Score)를 계산하여, 가장 높은 점수를 가지는 계층을 스케일링 할 계층으로 선택한다. 각 계층의 점수는 $mask$ 와 함수 $f(Tier_i)$ 결과 값의 곱으로 계산된다. 이 중, $mask$ 는 해당 계층 내에서 스케일링이 불가능한 경우에 0, 가능한 경우에는 1을 할당하여 점수를 보정한다. $f(Tier_i)$ 는 각 계층의 함수가 스케일 인아웃(Scale-in/out)에 얼마나 적합한지를 나타내는 함수이다. 스케일링을 적용할 계층은 각 계층의 CPU 사용량($CPU_{Util.}$)과 메모리 사용

$MEM_{Util.}$ $resUtil$ α β $f(Tier_i)$
량( )을 기반으로 로 정의하며, 각각 가중치 와 로 보정된다. 는 현재 SFC를 구성

$Node_{used}$
하는 VNF 인스턴스들이 배치된 물리 서버의 개수( )와 선택된 계층의 VNF 인스턴스들이 배치된 물리 서

$Node_{tier}$
버의 개수( )의 비율을 고려한다.

$resUtil$ $Node_{used}$ $Node_{tier}$
[0018] 수학식 3은 상기 정의한 , , 와 지수 함수를 활용하여 각 계층이 스케일링에 적합한 지 점수로 나타낸다. 즉, 수학식 3에서 Scale-in의 경우에는 자원 사용량이 낮고, VNF 인스턴스들이 여러 물리 서버에 분산되어 있는 계층에 큰 점수를 부여한다. 반면, Scale-out의 경우에는 자원 사용량이 높고, VNF 인스턴스들이 적은 수의 물리 서버에 밀집해 있는 계층에 큰 점수를 부여한다. 이는 Scale-in의 경우 VNF 인스턴스들이 분산되어 있는 계층에서 불필요한 VNF 인스턴스를 제거하고, Scale-out에서는 VNF 인스턴스들이 밀집해 있는 계층에 가용 VNF 인스턴스를 추가하기 위해서이다.

[0019] 본 발명에서는 보상 정의에 활용되는 데이터를 가져오기 위한 모니터링 기능이 존재한다고 가정한다. SFC 데이터, VNF 인스턴스 설치 위치 데이터와 물리 서버 데이터는 VNF가 운영되는 NFV환경에서 제공하는 모니터링 도구(예를 들어, OpenStack의 경우 Ceilometer)를 활용하여 가져올 수 있으며, 각 VNF 인스턴스의 자원 활용률은 오픈소스 모니터링 에이전트인 Collectd를 설치해 주기적으로 모니터링한 후, 시계열 데이터베이스에 저장하는 것으로 확보할 수 있다.

[0020] 본 발명의 성능은 임계값(Threshold) 기반 오토 스케일링 방법보다 제안하는 방법으로 수행한 오토 스케일링이 더 좋은 성능을 가진다는 것을 보여서 검증할 수 있다. 이 때, 성능 지표로는 오토 스케일링 되는 SFC의 SLO 위반 비율을 측정하여 활용할 수 있다.

발명의 효과

[0021] 본 발명의 일 실시예에 따르면, 사람이 수동으로 SFC의 스케일링을 결정하고 설정하는 것이 아닌, 강화학습 알고리즘 중 하나인 DQN을 활용하여 오토 스케일링을 수행하는 방법을 제시한다.

[0022] 본 발명의 결과물은 모듈 형태로 구현되어 실제 NFV 환경(예를 들면, OpenStack 등)에서 동작할 수 있으며, 오토 스케일링을 적용할 SFC를 정하면 해당 SFC를 구성하는 각 계층의 상황(Status) 정보를 주기적으로 상태로 받아들여 스케일링 행동을 결정한다.

[0023] 이러한 방법은 SFC의 오토 스케일링을 위한 편의성을 제공하고, SFC의 성능(트래픽 응답 시간), SFC를 구성하는 VNF 인스턴스들의 분포도, 물리 서버 개수 등을 고려하기 때문에 SFC를 통한 패킷 처리 안정성 측면에서 임의로 스케일링을 수행할 때보다 좋은 성능을 보일 수 있다.

도면의 간단한 설명

[0024] 도 1은 본 발명의 일실시예의 오토 스케일링 장치의 오토 스케일링 대상이 되는 다계층(Multi-tier) 구조를 가지는 SFC의 예이다.

도 2은 본 발명의 일실시예의 오토 스케일링 장치의 오토 스케일링 문제를 본 발명에서 제안하는 방법으로 해결할 때, 각 구성요소들이 동작하는 과정을 도식화한 것이다.

도 3은 본 발명의 일실시예의 오토 스케일링 장치의 SFC의 각 계층 상황 정보가 상태로 주어졌을 때, DQN을 통해 스케일링 행동을 출력하는 과정을 도식화한 것이다.

도 4는 본 발명의 일실시예의 오토 스케일링 장치의 DQN을 활용하여 오토 스케일링을 수행하는 과정을 표현하고 있다.

도 5는 본 발명의 일실시예의 오토 스케일링 장치의 구성도이다.

발명을 실시하기 위한 구체적인 내용

[0025] 본 발명은 다양한 변경을 가할 수 있고 여러 가지 실시예를 가질 수 있는 바, 특정 실시예들을 도면에 예시하고 상세한 설명에 상세하게 설명하고자 한다. 그러나, 이는 본 발명을 특정한 실시 형태에 대해 한정하려는 것이

아니며, 본 발명의 사상 및 기술 범위에 포함되는 모든 변경, 균등물 내지 대체물을 포함하는 것으로 이해되어야 한다. 각 도면을 설명하면서 유사한 참조부호를 유사한 구성요소에 대해 사용하였다.

[0026] 제1, 제2, A, B 등의 용어는 다양한 구성요소들을 설명하는 데 사용될 수 있지만, 상기 구성요소들은 상기 용어들에 의해 한정되어서는 안 된다. 상기 용어들은 하나의 구성요소를 다른 구성요소로부터 구별하는 목적으로만 사용된다. 예를 들어, 본 발명의 권리 범위를 벗어나지 않으면서 제1 구성요소는 제2 구성요소로 명명될 수 있고, 유사하게 제2 구성요소도 제1 구성요소로 명명될 수 있다. "및/또는"이라는 용어는 복수의 관련된 기재된 항목들의 조합 또는 복수의 관련된 기재된 항목들 중의 어느 항목을 포함한다.

[0027] 어떤 구성요소가 다른 구성요소에 "연결되어" 있다거나 "접속되어" 있다고 언급된 때에는, 그 다른 구성요소에 직접적으로 연결되어 있거나 또는 접속되어 있을 수도 있지만, 중간에 다른 구성요소가 존재할 수도 있다고 이해되어야 할 것이다. 반면에, 어떤 구성요소가 다른 구성요소에 "직접 연결되어" 있다거나 "직접 접속되어" 있다고 언급된 때에는, 중간에 다른 구성요소가 존재하지 않는 것으로 이해되어야 할 것이다.

[0028] 본 출원에서 사용한 용어는 단지 특정한 실시예를 설명하기 위해 사용된 것으로, 본 발명을 한정하려는 의도가 아니다. 단수의 표현은 문맥상 명백하게 다르게 뜻하지 않는 한, 복수의 표현을 포함한다. 본 출원에서, "포함하다" 또는 "가지다" 등의 용어는 명세서상에 기재된 특징, 숫자, 단계, 동작, 구성요소, 부품 또는 이들을 조합한 것이 존재함을 지정하려는 것이지, 하나 또는 그 이상의 다른 특징들이나 숫자, 단계, 동작, 구성요소, 부품 또는 이들을 조합한 것들의 존재 또는 부가 가능성을 미리 배제하지 않는 것으로 이해되어야 한다.

[0029] 다르게 정의되지 않는 한, 기술적이거나 과학적인 용어를 포함해서 여기서 사용되는 모든 용어들은 본 발명이 속하는 기술 분야에서 통상의 지식을 가진 자에 의해 일반적으로 이해되는 것과 동일한 의미를 가지고 있다. 일반적으로 사용되는 사전에 정의되어 있는 것과 같은 용어들은 관련 기술의 문맥 상 가지는 의미와 일치하는 의미를 가지는 것으로 해석되어야 하며, 본 출원에서 명백하게 정의하지 않는 한, 이상적이거나 과도하게 형식적인 의미로 해석되지 않는다.

[0031] 본 발명은 강화학습 기술을 활용하여 NFV 환경 내 SFC의 오토 스케일링을 수행하는 방법에 관한 것이다. 제안하는 방법은 SFC를 통과하는 트래픽의 변동으로 인해 발생할 수 있는 SFC의 과부하, 성능 저하 등의 문제에 대응하기 위하여, SFC를 구성하는 VNF가 동작하는 가상 머신(VM, Virtual Machine) 또는 컨테이너(Container)를 스케일링 하는 강화학습 문제로 정의한다. 오토 스케일링 문제를 해결하기 위해 강화학습 알고리즘 중 하나인 심층 Q 네트워크(Deep Q-networks, DQN)을 이용하며, 오토 스케일링으로 갱신된 SFC를 통해 측정되는 트래픽의 응답 시간(Response time)과 SFC를 구성하는 VNF 인스턴스들의 분포 상태, 가용 물리 서버 대비 VNF 인스턴스를 배치하는데 사용한 물리 서버 개수의 비를 보상 값으로 고려한다.

[0032] 본 발명은 강화학습 알고리즘 중 하나인 심층 Q 네트워크(Deep Q-networks, DQN)를 통해 SFC를 구성하는 VNF 인스턴스들의 스케일 인아웃(Scale-in/out)을 주기적으로 수행하는 오토 스케일링을 목표로 한다. 제안하는 방법은 DQN을 통해 SFC를 구성하는 계층(Tier)들의 상황(Status)을 강화학습의 상태(State)로 정의하여 입력 값으로 받아들인 후, 어떤 물리 서버에서 스케일 인아웃(Scale-in/out)을 수행할지 또는 현재 VNF 인스턴스들을 유지(Maintain)할지를 행동으로 출력한다. 또한, 스케일 인아웃(Scale-in/out)을 수행할 때, 스케일링이 필요한 SFC의 계층을 선택하여 해당 계층에 스케일링을 적용한다. 에이전트에서는 DQN의 안정적인 학습을 위해 Q-network및 Target Q-network, Replay Memory를 활용한다.

[0033] DQN은 일반적인 Q-learning과 마찬가지로, 특정 상태에서 행동을 수행할 때 얻을 수 있는 보상을 예측하는 지표인 Q-value를 반복적으로 학습한다. 학습된 Q-value는 특정 상태에서 어떤 행동을 수행할지 결정하는 정책으로 사용한다(예를 들어, 특정 상태에서 Q-value가 가장 큰 행동을 선택). DQN은 학습을 통해 특정 상태에서 수행할 행동을 출력하는 Q-network의 네트워크 파라미터를 갱신하는데, 학습된 Q-network는 최적의 스케일링 행동을 수행하는 최적 정책으로 사용된다.

[0035] (수학식 1)

$$L_i(\theta_i) = E_{(s,a,r,s')}[(r + \gamma max Q(s', a'; \theta_i^-) - Q(s, a; \theta_i))^2]$$

[0036]

[0038] 본 발명의 DQN은 Q-network와 Target Q-network를 생성하고, 수학식 1의 손실 함수(Loss function) 값을 최소화하는 형태로 네트워크 파라미터를 학습한다. 수학식 1은 DQN에서 학습을 위해 일반적으로 사용되는 손실 함수이며, Replay Memory에 저장된 데이터 $E_{s,a,r,s'}$ 를 학습 데이터로 입력받는다. 이 때, Target Q-network(네트워크 파라미터 θ_i^-)에서 얻을 수 있는 최대 Q-value와 Q-network(네트워크 파라미터 θ_i)의 Q-value의 차이를 줄이는 방향으로 Q-network를 학습한다. 또한, 일정 횟수 이상 Q-network를 학습하면 Q-network의 네트워크 파라미터를 Target Q-network로 복사하는데, 이는 매 학습마다 Target Q-network도 같이 갱신하면 Q-network이 학습을 제대로 수행되지 않고 발산하기 때문이다.

[0040] (수학식 2)

$$\mathbf{Node}_{\mathbf{Util.}} = \frac{Node_{used}}{Node_{total}}, \mathbf{Dist}_{\mathbf{VNF}} = \prod_{i=1}^{n} \frac{VNF_{node_i}}{VNF_{total}}$$

[0041]

[0042] 단, $\mathbf{node}_i$ 는 최소한 1개 이상의 VNF 인스턴스가 배치된 물리 서버이고,

$$\mathbf{R}_{t+1} = -\frac{resTime}{SLO} + \alpha \mathbf{Dist}_{\mathbf{VNF}} \times e^{-\beta Node_{Util.}}$$

[0043]

[0045] DQN의 학습 과정에서는 수행한 행동에 대한 보상 값 r이 반영되어야 하기 때문에 수학식 2와 같이 보상 모델을 정의하였다. 수학식 2에서 $resTime$ 은 스케일링을 수행한 SFC를 통해 트래픽을 전송하고, 응답을 받을 때까지 소요되는 응답 시간(Response time)을 의미한다. 또한, DQN 기반 오토 스케일링 방법에서는 SLO(Service Level Objectives)로 트래픽의 응답 시간을 활용한다. SFC 경로를 통해 측정되는 응답 시간은 편차가 클 수 있기 때문에, 측정된 결과 값을 그대로 활용하면 보상 값에도 큰 편차가 생길 수 있다. 따라서 본 발명에서는 측정된 응답 시간인 $resTime$ 을 그대로 사용하는 것이 아니라, 미리 정의된 SLO 대비 응답 시간이 얼마나 되는 지를 비율로 환산하여 보상 값에 반영한다. 예를 들어, SLO로 50ms가 설정되어 있고, 실제 측정된 $resTime$ 가 25ms일 경우, 0.5가 반영된다.

[0046] 트래픽 응답 시간 외에도 수학식 2에서는 $Node_{Util.}$ 과 $Dist_{VNF}$ 를 보상 r에 반영한다. $Node_{Util.}$ 는 NFV 환경에서 가용할 수 있는 총 물리 서버 개수($Node_{total}$) 대비 SFC를 구성하는 VNF 인스턴스가 배치되어 있는 물리 서버 개수($Node_{used}$)의 비율을 의미한다. 반면, $Dist_{VNF}$ 는 SFC를 구성하는 VNF 인스턴스들의 분포도를 나타내는 값이다. SFC를 구성하는 전체 VNF 인스턴스 개수(VNF_{total}) 대비 VNF 인스턴스가 배치된 각 물리 서버에서

VNF_{node_i} $Dist_{VNF}$

실행되는 VNF 인스턴스 개수( )의 비율을 곱하여 계산한다. 는 VNF 인스턴스들이 적은 수의 물리 서버에 밀집되어 배치되면 높은 값을 가지게 되고, 많은 서버들에 분산 배치되어 있을 경우 작은 값을 가지게 된다. SFC의 각 계층 내 VNF 인스턴스들은 로드 밸런서(Load-balancer)를 통해 트래픽을 분산 받기 때문에, SFC를 구성하는 VNF 인스턴스들이 많은 물리 서버에 분산 배치되어 있을 경우 트래픽 또한 해당 물리 서버들로 분산된다. 결국, 각 계층에 속한 VNF 인스턴스들이 크게 분산되어 있다면, 동일한 SFC가 적용되는 트래픽의 패킷 전달 시간과 응답 시간의 편차가 클 수 있다.

$Node_{Util.}$ $Dist_{VNF}$ α β

[0047] 본 발명에서 제안하는 수학식 2의 보상 모델은 지수 함수에 와 를 가중치 와 로 보정하여 반영하고, SFC를 흐르는 트래픽의 응답 시간을 보상 r에 고려한다. 따라서 수학식 2로 계산되는 보상 값은 스케일링 행동으로 갱신된 SFC를 통과하는 트래픽의 응답 시간이 짧고, SFC를 구성하는 VNF 인스턴스들이 적은 물리 서버에 밀집된 형태로 배치되었을 경우 큰 값을 가지게 된다.

[0048] **(수학식 3)**

$$\mathbf{score} = \mathbf{mask} \times \mathbf{f}(\mathbf{Tier_i})$$

[0049]

$$\mathbf{resUtil} = \alpha \mathbf{CPU}_{\mathbf{Util.}} + \boldsymbol{\beta} \boldsymbol{MEM}_{\boldsymbol{Util.}}$$

[0050]

$$\mathbf{f}(\mathbf{Tier_i}) = \begin{cases} \dfrac{Node_{tier}}{Node_{used}} \times e^{-resUtil}, & if\ scale-in \\ \dfrac{Node_{used}}{Node_{tier}} \times e^{resUtil}, & else\ if\ scale-out \end{cases}$$

[0051]

[0052] 본 발명의 오토 스케일링 방법은 다계층으로 이루어 진 SFC를 대상으로 하기 때문에 스케일링을 적용할 특정 계층을 선택하는 것이 필요하다. 따라서 현재 상태에서 스케일링이 필요하다고 에이전트가 판단했을 경우, 수학식 3에 의해 스케일링을 적용할 계층을 선택한다. 본 발명에서 제안하는 수학식 3은 각 계층마다 점수(Score)를 계산하여, 가장 높은 점수를 가지는 계층을 스케일링 할 계층으로 선택한다. 각 계층의 점수는 $mask$ 와 함수 $f(Tier_i)$ 결과 값의 곱으로 계산된다. 이 중, $mask$ 는 해당 계층 내에서 스케일링이 불가능한 경우에 0, 가능한 경우에는 1을 할당하여 점수를 보정한다. $f(Tier_i)$ 는 각 계층의 함수가 스케일 인아웃(Scale-in/out)에 얼마나 적합한지를 나타내는 함수이다. 스케일링을 적용할 계층은 각 계층의 CPU 사용량($CPU_{Util.}$)과 메모리 사용량($MEM_{Util.}$)을 기반으로 $resUtil$ 로 정의하며, 각각 가중치 α 와 β 로 보정된다. $f(Tier_i)$ 는 현재 SFC를 구성하는 VNF 인스턴스들이 배치된 물리 서버의 개수($Node_{used}$)와 선택된 계층의 VNF 인스턴스들이 배치된 물리 서버의 개수($Node_{tier}$)의 비율을 고려한다.

$resUtil$ $Node_{used}$ $Node_{tier}$

[0053] 수학식 3은 상기 정의한 , , 와 지수 함수를 활용하여 각 계층이 스케일링에 적합한지 점수로 나타낸다. 즉, 수학식 3에서 Scale-in의 경우에는 자원 사용량이 낮고, VNF 인스턴스들이 여러 물리 서버에 분산되어 있는 계층에 큰 점수를 부여한다. 반면, Scale-out의 경우에는 자원 사용량이 높고, VNF 인스턴스들이 적은 수의 물리 서버에 밀집해 있는 계층에 큰 점수를 부여한다. 이는 Scale-in의 경우 VNF 인스턴스들이 분산되어 있는 계층에서 불필요한 VNF 인스턴스를 제거하고, Scale-out에서는 VNF 인스턴스들이 밀집해 있

는 계층에 가용 VNF 인스턴스를 추가하기 위해서이다.

[0054] 본 발명에서는 보상 정의에 활용되는 데이터를 가져오기 위한 모니터링 기능이 존재한다고 가정한다. SFC 데이터, VNF 인스턴스 설치 위치 데이터와 물리 서버 데이터는 VNF가 운영되는 NFV환경에서 제공하는 모니터링 도구(예를 들어, OpenStack의 경우 Ceilometer)를 활용하여 가져올 수 있으며, 각 VNF 인스턴스의 자원 활용률은 오픈소스 모니터링 에이전트인 Collectd를 설치해 주기적으로 모니터링한 후, 시계열 데이터베이스에 저장하는 것으로 확보할 수 있다.

[0055] 본 발명의 성능은 임계값(Threshold) 기반 오토 스케일링 방법보다 제안하는 방법으로 수행한 오토 스케일링이 더 좋은 성능을 가진다는 것을 보여서 검증할 수 있다. 이 때, 성능 지표로는 오토 스케일링 되는 SFC의 SLO 위반 비율을 측정하여 활용할 수 있다.

[0057] 이하, 본 발명에 따른 바람직한 실시예를 첨부된 도면을 참조하여 상세하게 설명한다.

[0058] 도 1은 오토 스케일링 대상이 되는 다계층(Multi-tier) 구조를 가지는 SFC(100)의 예를 도시한 구성도이다.

[0059] 도 1을 참조하면, 본 발명에서 제안하는 DQN 기반 오토 스케일링 방법은 SFC(100)를 구성하는 각 계층(Tier)(110,120)의 VNF 인스턴스 개수를 조절하는 스케일 인아웃(Scale-in/out) 문제로 정의한다. 또한, SFC는 여러 계층으로 구성될 수 있기 때문에 스케일링을 적용할 계층을 선택하는 방법도 포함한다. 도 1의 오토 스케일링 대상인 되는 다계층(Multi-tier) 구조를 가진 SFC(100) 는 Firewall(111), IDS(121) 두 종류로 이루어진 2-계층(110,120) SFC(100)를 보이고 있으며, 오토 스케일링이 필요할 때 Fireall 계층(110)과 IDS 계층(120) 중 어느 계층에 스케일링을 수행할지 결정해야 한다. DQN 기반 오토 스케일링 문제에서는 SFC를 구성하는 각 계층의 상황을 강화학습 문제의 상태(State)로 정의하고, 이를 입력으로 받아들여 DQN이 특정 상태에서 수행해야 하는 스케일링을 행동으로 출력한다. 스케일링을 수행한 후에는 갱신된 SFC에서 측정된 트래픽 응답 시간, SFC를 구성하는 VNF 인스턴스들의 분포도 및 인스턴스를 배치하는데 사용된 물리 서버의 개수 등을 고려하여 보상을 부여한다.

[0060]

[0061] 도 2는 오토 스케일링 문제를 본 발명에서 제안하는 방법으로 해결할 때, 각 구성요소들이 동작하는 과정을 도식화한 구성도이다.

[0062] 도 2를 참조하면, 본 발명의 DQN은 안정적인 학습을 위해 에이전트(200)에서 두 개의 심층 네트워크인 Q-network(210)와 Target Q-network(220)를 사용한다. 이는 Q-value를 학습하는 과정에서 최적 값으로 수렴하지 않고 발산하는 것을 방지한다. 또한, 특정 상태에서 스케일링 행동을 수행했을 때 받게 되는 보상과 다음 상태, $(\mathbf{S}_t, \mathbf{A}_t, R_{t+1}, S_{t+1}, Mask)$ 그리고 스케일링이 성공했는지 여부를 Replay Memory(400)에 형태로 저장한다. $\mathbf{S}_t, \mathbf{A}_t, R_{t+1}$ $Mask$ 은 각각 상태, 행동, 보상이며, 는 선택된 스케일링 행동이 정상적으로 수행되었는지를 나타내는 값이다. 예를 들어, 물리 서버에 가용 자원이 없이 VNF 인스턴스를 추가할 수 없는 경우는 Scale-out에 실패 $Mask$ 하여 값이 0으로 할당되며, 성공할 경우에는 1이 할당된다. Replay Memory(400)에 저장된 데이터들을 Mini-batch 방식으로 학습을 하면, 데이터 간 상관관계로 인해 잘못된 네트워크 파라미터를 학습하는 것을 방지할 수 있다.

[0063] 본 발명은 NFV 환경(300)에서 SFC(100)의 오토 스케일링을 강화학습 알고리즘 중 하나인 심층 Q 네트워크(Deep Q-networks, DQN)(200)로 수행하는 방법을 개발하고자 한다. 제안하는 방법에서는 오토 스케일링을 적용할 SFC(100)가 정해지면, 주기적으로 SFC(100)를 구성하는 각 계층(110, 120)의 상황 정보를 입력 값으로 받아 어떤 위치(예를 들면, 물리 서버)에서 어떤 스케일링 행동이 수행되어야 하는지를 출력한다. 이 때, 수행할 행동이 스케일 인아웃(Scale-in/out)일 경우, 스케일링을 적용할 적합한 SFC 계층(110,120)을 선택한다. 본 발명에서는 DQN을 사용하여 오토 스케일링을 수행하는 방법과 함께 OpenStack 환경(300)에서 실제 시스템 형태로 구현할 수 있는 방법도 제시하고 있다.

[0065] 도 3은 SFC의 각 계층 상황 정보가 상태로 주어졌을 때, DQN을 통해 스케일링 행동을 출력하는 과정을 도식화한 모식도이다.

[0066] 도 3을 참조하면, SFC의 각 계층의 상황(Tier Status)이 DQN의 상태로 주어졌을 때, 수행할 스케일링 행동을 출력하는 과정이 도식화된다. 이 때, 스케일링 행동은 VNF 인스턴스를 추가(Add)하는 Scale-out, VNF 인스턴스를 제거(Remove)하는 Scale-in, 현재 VNF 인스턴스를 유지(Maintain)하는 경우로 나뉘며, 스케일 인아웃(Scale-in/out)의 경우 어떤 위치에 VNF 인스턴스를 추가/제거할지도 고려한다. 각 계층의 상황 정보는 5개의 데이터로 구성이 되며, 데이터 종류는 계층에 존재하는 VNF 인스턴스들의 평균 CPU 사용량, 평균 메모리 사용량, 평균 디스크 작업 수행 횟수, 계층 내 VNF 인스턴스 개수, VNF 인스턴스의 분포도이다. 계층 상황에서 CPU와 메모리를 고려하는 이유는 해당 자원들이 충분치 않으면 패킷(10) 처리가 지연(Delay)되거나 이로 인한 패킷 손실(Packet loss)를 발생시키는 등, 패킷(10) 처리 성능에 영향을 크게 미치는 요소들이기 때문이다. 또한, 디스크 작업 수행 횟수는 디스크를 읽거나 쓰는 작업 횟수를 의미하는데, 메모리 자원이 과도하게 사용될 경우 Swap 작업이 발생하여 디스크 작업 횟수가 높게 측정될 수 있다. Swap 작업은 메모리에 저장할 데이터 일부를 디스크에 저장하는 것인데, 메모리 작업에 비해 디스크 작업은 속도가 느리기 때문에 병목 현상을 발생시켜 간접적으로 패킷(10) 처리 성능에 영향을 미친다. 그 외에는 각 계층에 속한 VNF 인스턴스 개수와 VNF 인스턴스 분포도를 계층 상황으로 고려한다. VNF 인스턴스 분포도는 NFV 환경 내에서 VNF 인스턴스를 생성할 수 있는 총 가용 물리 서버 개수 대비 실제 VNF 인스턴스가 배치 된 물리 서버 개수로 계산 된 값이다. 예를 들어, 10개의 가용서버가 있는데, 그 중 3개의 서버에 현재 계층의 VNF 인스턴스가 배치되어 있을 경우, 분포도 값은 0.3이 된다.

[0067] 본 발명에서는 사람이 수동으로 SFC의 스케일링을 결정하고 설정하는 것이 아닌, 강화학습 알고리즘 중 하나인 DQN을 활용하여 오토 스케일링을 수행하는 방법을 제시한다. 본 발명의 결과물은 모듈 형태로 구현되어 실제 NFV 환경(예를 들면, OpenStack 등)에서 동작할 수 있으며, 오토 스케일링을 적용할 SFC를 정하면 해당 SFC를 구성하는 각 계층의 상황(Status) 정보를 주기적으로 상태로 받아들여 스케일링 행동을 결정한다. 이러한 방법은 SFC의 오토 스케일링을 위한 편의성을 제공하고, SFC의 성능(트래픽 응답 시간), SFC를 구성하는 VNF 인스턴스들의 분포도, 물리 서버 개수 등을 고려하기 때문에 SFC를 통한 패킷(10) 처리 안정성 측면에서 임의로 스케일링을 수행할 때보다 좋은 성능을 보일 수 있다.

[0069] 도 4는 DQN을 활용하여 오토 스케일링을 수행하는 과정을 표현한 순서도이다.

[0070] 도 4를 참조하면, DQN 기반 오토 스케일링을 요청했을 때, 오토 스케일링 기능을 수행하는 순서를 보인다(S401). 본 발명의 결과물은 실제 NFV 환경(ex. OpenStack)에서 동작할 수 있는 모듈 형태로 구현되며, 오토 스케일링 모듈이 오토 스케일링을 적용할 SFC의 이름과 오토 스케일링을 수행하는데 필요한 파라미터(Parameter)가 포함된 요청 메시지를 수신하면 오토 스케일링 프로세스를 실행한다. 이 때, 오토 스케일링 프로세스는 쓰레드(Thread)로 동작하여 여러 오토 스케일링 프로세스가 동시에 수행될 수 있도록 한다.

[0071] 프로세스가 실행되면 오토 스케일링을 적용 할 SFC의 데이터와 물리 서버 정보 등, 오토 스케일링에서 필요한 데이터를 모니터링 모듈로 요청하여 받아온다(S402).

[0072] 이후, DQN의 하이퍼파라미터(Hyperparameter) 값을 설정하고(S403), Q-network(210)와 Target Q-network(220)를 생성한다(S404).

[0073] 다음으로는 Replay Memory(400)를 생성하는데(S405), 만약 Replay Memory(400)로 읽어올 학습용 데이터(Dataset)이 미리 파일 형태로 존재한다면(S406), 해당 데이터를 읽어서 Replay Memory(400)에 저장한다(S407). Replay Memory(400) 생성까지 완료된 후에는 본격적인 오토 스케일링을 수행한다.

[0074] 먼저, 오토 스케일링 대상이 되는 SFC 내 각 계층 상황(Tier Status)를 가져온 후, 현재 상태(State)를 DQN에 입력할 수 있는 형태인 텐서(Tensor)로 변환한다(S408).

[0075] 텐서로 변환된 상태는 DQN에 입력되고, 어떤 물리 서버에서 어떤 스케일링을 수행할 것인지를 나타내는 행동이 출력된다(S409).

[0076] 출력된 결과가 스케일 인아웃(Scale-in/out) 일 경우(S410), 스케일링을 적용할 계층(Tier)를 선택한다(S411).

[0077] 계층까지 결정된 후에는 DQN의 결과로 선택된 스케일링 행동을 수행한다(S412).

[0078] 행동을 수행하고 나서, 에이전트(200)는 스케일링으로 인한 보상 값을 계산하고(S413), 새롭게 추가 또는 제거

된 VNF 인스턴스를 반영하여 SFC를 갱신한다(S414).

[0079] SFC 갱신을 완료한 후에는 SFC 내 각 계층 상황을 다시 가져온 후, 새로운 상태를 텐서로 변환한다(S415).

[0080] 새로운 상태까지 텐서로 반환한 후에는 현재 상태, 행동, 보상, 새로운 상태, 행동 성공 여부로 이루어 진 데이터를 Replay Memory(400)에 저장한다(S416).

[0081] Replay Memory(400)에 최소 개수 N개 이상의 데이터가 쌓이면(S417), 이를 Mini-batch로 샘플링(Sampling)하여 Q-network(210)를 학습한다(S418).

[0082] 본 발명의 DQN은 Q-network(210)와 Target Q-network(220)로 구성되어 있기 때문에 스케일링 행동을 일정 횟수 반복(S419)할 때마다 주기적으로 Q-network(210)의 학습된 네트워크 파라미터를 Target Q-network(220)로 복사 및 갱신한다(S420).

[0083] 매번 스케일링 행동을 수행하고 나서 오토 스케일링 모듈은 오토 스케일링 프로세스가 현재까지 실행된 시간을 계산한다(S421). 미리 정의된 수행 시간 한도(Duration)보다 실제 실행된 시간이 길면(S422) 오토 스케일링 프로세스를 종료한다(S423). 반면, 오토 스케일링 프로세스의 만료까지 시간이 남아있을 경우, 다시 현재 SFC의 상태를 계산(S408)한 후 스케일링 과정을 반복한다.

[0085] 도 5는 본 발명의 일 실시예의 오토 스케일링 장치(1000)의 구성도이다.

[0086] 도 5를 참조하면, 본 발명의 일 실시예의 오토 스케일링 장치(1000)는, 프로세서(1100), 메모리(1200), 송수신 장치(transceiver, 1300), 입력 인터페이스 장치(1400), 출력 인터페이스 장치(1500), 저장 장치(1600) 및 버스(bus)(1700)를 포함하여 구성될 수 있다.

[0087] 본 발명의 오토 스케일링 장치(1000)는, 프로세서(processor)(1100) 및 프로세서(1100)를 통해 실행되는 적어도 하나의 명령이 저장된 메모리(memory)(1200)를 포함하되, 적어도 하나의 명령은 상기 프로세서(1100)가,

[0088] 를 수행하도록 구성된다.

[0089] 프로세서(1100)는 중앙 처리 장치(central processing unit, CPU), 그래픽 처리 장치(graphics processing unit, GPU), 또는 본 발명의 실시예들에 따른 방법들이 수행되는 전용의 프로세서를 의미할 수 있다.

[0090] 메모리(1200) 및 저장 장치(1600) 각각은 휘발성 저장 매체 및 비휘발성 저장 매체 중에서 적어도 하나로 구성될 수 있다. 예를 들어, 메모리(1200)는 읽기 전용 메모리(read only memory, ROM) 및 랜덤 액세스 메모리(random access memory, RAM) 중에서 적어도 하나로 구성될 수 있다.

[0091] 또한, 오토 스케일링 장치(1000)는 무선 네트워크를 통해 통신을 수행하는 송수신 장치(transceiver)(1300)를 포함할 수 있다.

[0092] 또한, 오토 스케일링 장치(1000)는 입력 인터페이스 장치(1400), 출력 인터페이스 장치(1500), 저장 장치(1600) 등을 더 포함할 수 있다.

[0093] 또한, 오토 스케일링 장치(1000)에 포함된 각각의 구성 요소들은 버스(bus)(1700)에 의해 연결되어 서로 통신을 수행할 수 있다.

[0094] 본 발명의 오토 스케일링 장치(1000)의 예를 들면, 통신 가능한 데스크탑 컴퓨터(desktop computer), 랩탑 컴퓨터(laptop computer), 노트북(notebook), 스마트폰(smart phone), 태블릿 PC(tablet PC), 모바일폰(mobile phone), 스마트 워치(smart watch), 스마트 글래스(smart glass), e-book 리더기, PMP(portable multimedia player), 휴대용 게임기, 네비게이션(navigation) 장치, 디지털 카메라(digital camera), DMB(digital multimedia broadcasting) 재생기, 디지털 음성 녹음기(digital audio recorder), 디지털 음성 재생기(digital audio player), 디지털 동영상 녹화기(digital video recorder), 디지털 동영상 재생기(digital video player), PDA(Personal Digital Assistant) 등일 수 있다.

[0096] 본 발명의 실시예에 따른 방법의 동작은 컴퓨터로 읽을 수 있는 기록매체에 컴퓨터가 읽을 수 있는 프로그램 또는 코드로서 구현하는 것이 가능하다. 컴퓨터가 읽을 수 있는 기록매체는 컴퓨터 시스템에 의해 읽혀질 수 있는 정보가 저장되는 모든 종류의 기록장치를 포함한다. 또한 컴퓨터가 읽을 수 있는 기록매체는 네트워크로 연결된

컴퓨터 시스템에 분산되어 분산 방식으로 컴퓨터로 읽을 수 있는 프로그램 또는 코드가 저장되고 실행될 수 있다.

[0097] 또한, 컴퓨터가 읽을 수 있는 기록매체는 롬(rom), 램(ram), 플래시 메모리(flash memory) 등과 같이 프로그램 명령을 저장하고 수행하도록 특별히 구성된 하드웨어 장치를 포함할 수 있다. 프로그램 명령은 컴파일러(compiler)에 의해 만들어지는 것과 같은 기계어 코드뿐만 아니라 인터프리터(interpreter) 등을 사용해서 컴퓨터에 의해 실행될 수 있는 고급 언어 코드를 포함할 수 있다.

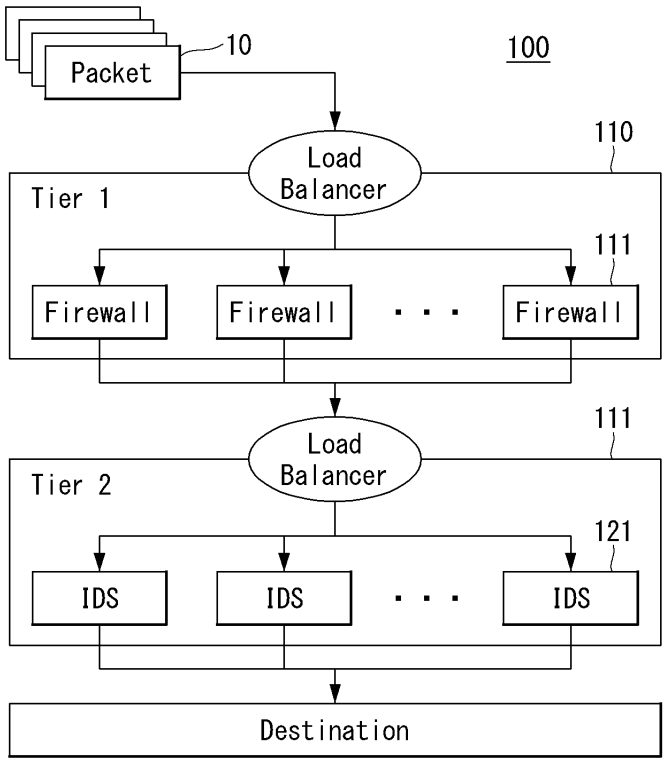
[0098] 본 발명의 일부 측면들은 장치의 문맥에서 설명되었으나, 그것은 상응하는 방법에 따른 설명 또한 나타낼 수 있고, 여기서 블록 또는 장치는 방법 단계 또는 방법 단계의 특징에 상응한다. 유사하게, 방법의 문맥에서 설명된 측면들은 또한 상응하는 블록 또는 아이템 또는 상응하는 장치의 특징으로 나타낼 수 있다. 방법 단계들의 몇몇 또는 전부는 예를 들어, 마이크로프로세서, 프로그램 가능한 컴퓨터 또는 전자 회로와 같은 하드웨어 장치에 의해(또는 이용하여) 수행될 수 있다. 몇몇의 실시예에서, 가장 중요한 방법 단계들의 하나 이상은 이와 같은 장치에 의해 수행될 수 있다.

[0099] 실시예들에서, 프로그램 가능한 로직 장치(예를 들어, 필드 프로그래머블 게이트 어레이)가 여기서 설명된 방법들의 기능의 일부 또는 전부를 수행하기 위해 사용될 수 있다. 실시예들에서, 필드 프로그머블 게이트 어레이는 여기서 설명된 방법들 중 하나를 수행하기 위한 마이크로프로세서와 함께 작동할 수 있다. 일반적으로, 방법들은 어떤 하드웨어 장치에 의해 수행되는 것이 바람직하다.

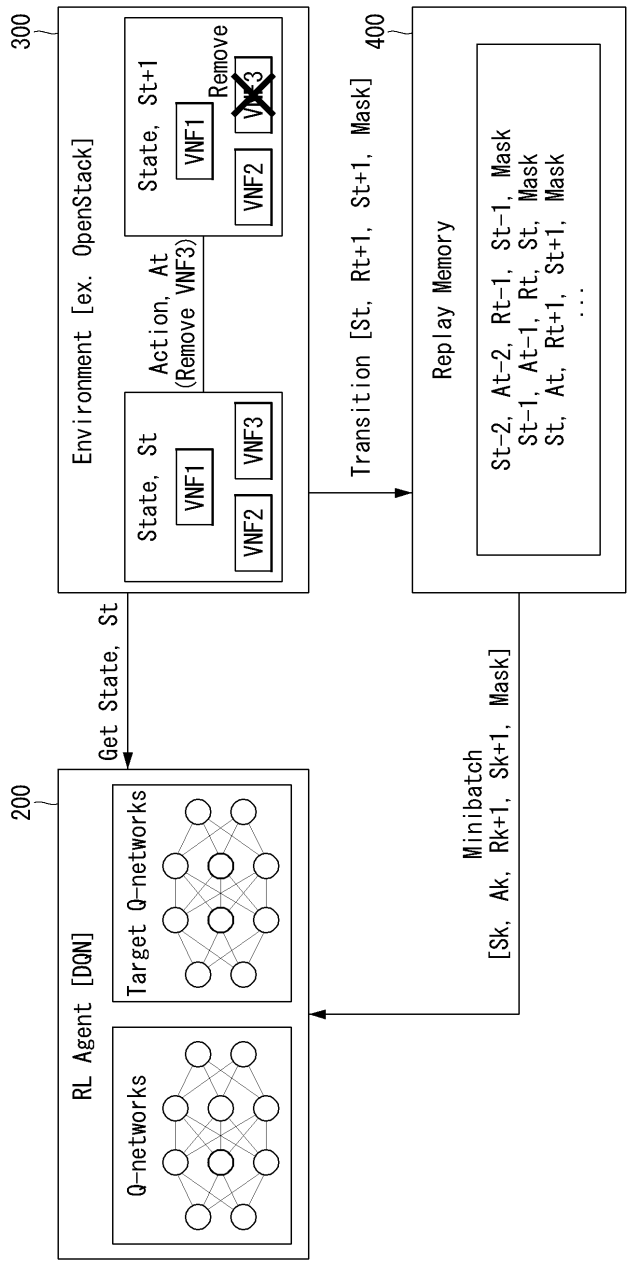
[0100] 이상 본 발명의 바람직한 실시예를 참조하여 설명하였지만, 해당 기술 분야의 숙련된 당업자는 하기의 특허 청구의 범위에 기재된 본 발명의 사상 및 영역으로부터 벗어나지 않는 범위 내에서 본 발명을 다양하게 수정 및 변경시킬 수 있음을 이해할 수 있을 것이다.

도면

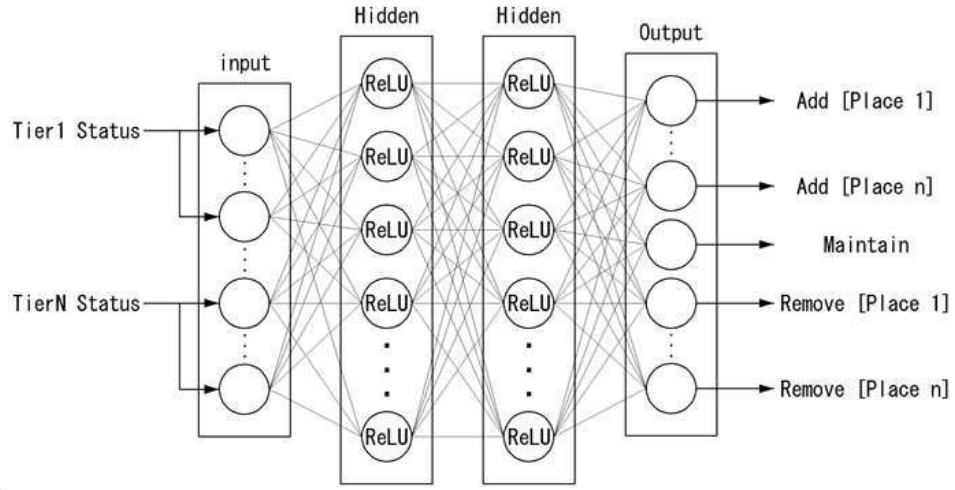
도면1



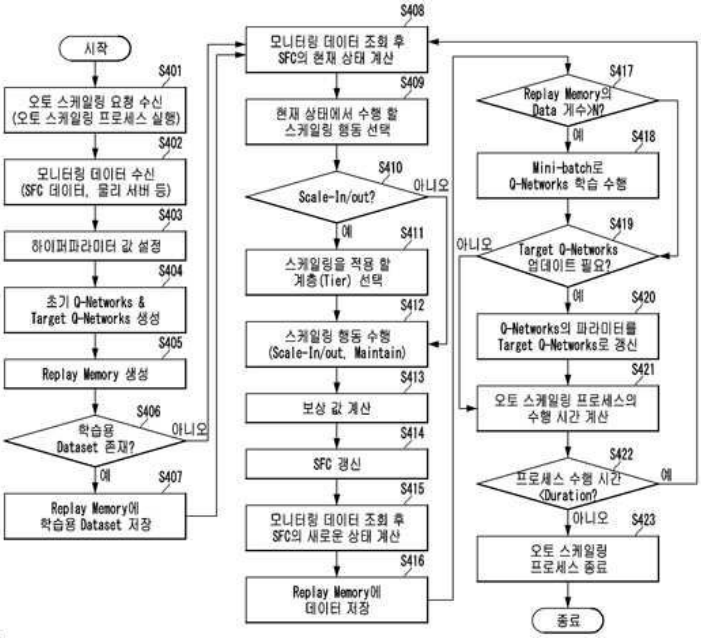
도면2


300
Environment [ex. OpenStack]
State, St
VNF1
VNF2
VNF3
Action, At
(Remove VNF3)
State, St+1
VNF1
VNF2
VNF3
Remove
400
Transition [St, Rt+1, St+1, Mask]
Replay Memory
St-2, At-2, Rt-1, St-1, Mask
St-1, At-1, Rt, St, Mask
St, At, Rt+1, St+1, Mask
...
Get State, St
Minibatch
[Sk, Ak, Rk+1, Sk+1, Mask]
200
RL Agent [DQN]
Target Q-networks
Q-networks

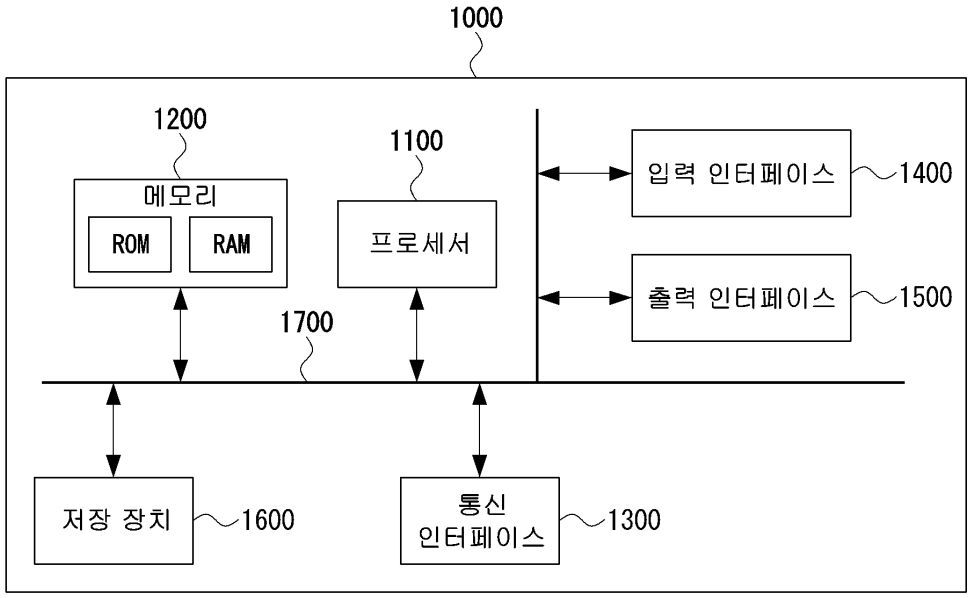
도면3


input
Hidden
Hidden
Output
Tier1 Status
TierN Status
ReLU
Add [Place 1]
Add [Place n]
Maintain
Remove [Place 1]
Remove [Place n]

도면4


시작
S401
오토 스케일링 요청 수신
(오토 스케일링 프로세스 실행)
S402
모니터링 데이터 수신
(SFC 데이터, 물리 서버 등)
S403
하이퍼파라미터 값 설정
S404
초기 Q-Networks &
Target Q-Networks 생성
S405
Replay Memory 생성
S406
학습용
Dataset 존재?
아니오
예
S407
Replay Memory에
학습용 Dataset 저장
S408
모니터링 데이터 조회 후
SFC의 현재 상태 계산
S409
현재 상태에서 수행 할
스케일링 행동 선택
S410
Scale-In/out?
아니오
예
S411
스케일링을 적용 할
계층(Tier) 선택
S412
스케일링 행동 수행
(Scale-In/out, Maintain)
S413
보상 값 계산
S414
SFC 갱신
S415
모니터링 데이터 조회 후
SFC의 새로운 상태 계산
S416
Replay Memory에
데이터 저장
S417
Replay Memory의
Data 개수>N?
예
S418
Mini-batch로
Q-Networks 학습 수행
S419
Target Q-Networks
업데이트 필요?
아니오
예
S420
Q-Networks의 파라미터를
Target Q-Networks로 갱신
S421
오토 스케일링 프로세스의
수행 시간 계산
S422
프로세스 수행 시간
<Duration?
예
아니오
S423
오토 스케일링
프로세스 종료
종료

도면5


1000
1200
메모리
ROM
RAM
1100
프로세서
입력 인터페이스
1400
출력 인터페이스
1500
1700
저장 장치
1600
통신
인터페이스
1300