(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2022-0006904
(43) 공개일자 2022년01월18일

(51) 국제특허분류(Int. Cl.)
G06F 16/901 (2019.01) *G06F 16/174* (2019.01)
G06F 16/903 (2019.01) *G06F 16/9032* (2019.01)
G06F 16/9038 (2019.01) *G06F 17/18* (2006.01)
(52) CPC특허분류
G06F 16/9024 (2019.01)
G06F 16/1744 (2019.01)
(21) 출원번호 10-2020-0084900
(22) 출원일자 2020년07월09일
심사청구일자 2020년07월09일

(71) 출원인
한국과학기술원
대전광역시 유성구 대학로 291(구성동)
충남대학교산학협력단
대전광역시 유성구 대학로 99 (궁동, 충남대학교)
(72) 발명자
신기정
대전광역시 유성구 대학로 291 (구성동, 한국과학기술원)
이규한
경기도 성남시 분당구 양현로 192(이매동, 이매촌 삼성아파트)
(뒷면에 계속)
(74) 대리인
양성보

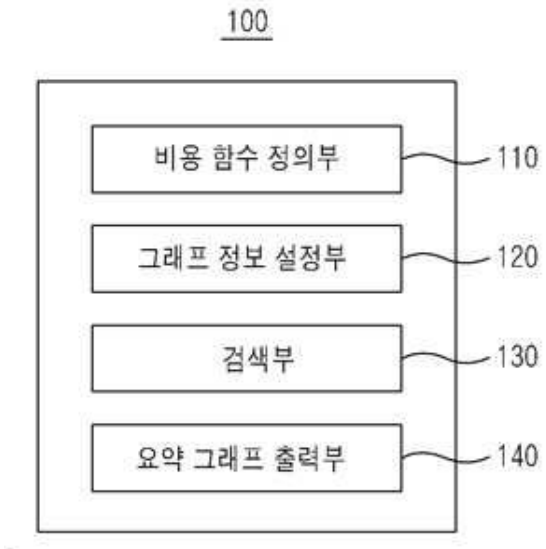
전체 청구항 수 : 총 10 항

(54) 발명의 명칭 **대용량 그래프의 희소 요약 방법 및 시스템**

(57) 요 약

대용량 그래프의 희소 요약 방법 및 시스템이 개시된다. 일 실시예에 따른 그래프 요약 방법은, 대용량 그래프에 대한 요약 그래프의 품질을 측정하기 위하여 비용 함수를 정의하는 단계; 상기 정의된 비용 함수를 이용하여 비트 단위의 요약 그래프 정보를 설정하는 단계; 상기 설정된 비트 단위의 요약 그래프 정보에 기초하여 생성된 후보 요약 그래프에 대한 무작위 검색을 수행하는 단계; 및 상기 수행된 무작위 검색을 통하여 획득된 요약 그래프를 결과로서 출력하는 단계를 포함할 수 있다.

대 표 도 - 도1



(52) CPC특허분류
G06F 16/90328 (2019.01)
G06F 16/90335 (2019.01)
G06F 16/9038 (2019.01)
G06F 17/18 (2013.01)

(72) 발명자
조현수
대전광역시 유성구 문화원로146번길 2(봉명동)
고지훈
제주특별자치도 제주시 도남로 104(도남동, 백록아파트)
임성수
대전광역시 유성구 대학로 99, 충남대학교 공학5호관 612호(궁동)

이 발명을 지원한 국가연구개발사업
과제고유번호 1711096050
과제번호 2019059755
부처명 과학기술정보통신부
과제관리(전문)기관명 한국연구재단
연구사업명 이공분야기초연구사업
연구과제명 (EZBARO)대용량의 동적인 그래프를 위한 점진적 압축 알고리즘 설계(2019)
기 여 율 1/3
과제수행기관명 한국과학기술원
연구기간 2019.06.01 ~ 2020.05.31
이 발명을 지원한 국가연구개발사업
과제고유번호 1711093607
과제번호 20190000750011001
부처명 과학기술정보통신부
과제관리(전문)기관명 정보통신기획평가원
연구사업명 정보통신.방송 연구개발사업
연구과제명 (EZBARO)인공지능대학원지원(한국과학기술원)(2019)
기 여 율 1/3
과제수행기관명 한국과학기술원
연구기간 2019.04.01 ~ 2019.12.31
이 발명을 지원한 국가연구개발사업
과제고유번호 1711100269
과제번호 G04190013
부처명 과학기술정보통신부
과제관리(전문)기관명 한국과학기술원
연구사업명 KAIST자체연구사업
연구과제명 대용량의 동적인 데이터를 분석하기 위한 고확장성 알고리즘 연구
기 여 율 1/3
과제수행기관명 한국과학기술원
연구기간 2019.02.26 ~ 2022.12.31

명 세 서

청구범위

청구항 1

그래프 요약 방법에 있어서,

대용량 그래프에 대한 요약 그래프의 품질을 측정하기 위하여 비용 함수를 정의하는 단계;

상기 정의된 비용 함수를 이용하여 비트 단위의 요약 그래프 정보를 설정하는 단계;

상기 설정된 비트 단위의 요약 그래프 정보에 기초하여 생성된 후보 요약 그래프에 대한 무작위 검색을 수행하는 단계; 및

상기 수행된 무작위 검색을 통하여 획득된 요약 그래프를 결과로서 출력하는 단계

를 포함하는 그래프 요약 방법.

청구항 2

제1항에 있어서,

상기 무작위 검색을 수행하는 단계는,

상기 대용량 그래프를 구성하는 수퍼 노드들을 클러스터링하여 후보군을 생성하는 단계

를 포함하는 그래프 요약 방법.

청구항 3

제2항에 있어서,

상기 무작위 검색을 수행하는 단계는,

상기 생성된 후보군에서 검색된 비용 함수가 감소하는 복수 개의 수퍼 노드를 반복적으로 병합함과 동시에 상기 병합을 통해 새로 생성된 수퍼 노드에 인접한 수퍼 엣지를 선택적으로 생성하여 요약 그래프를 희소화하는 단계

를 포함하는 그래프 요약 방법.

청구항 4

제3항에 있어서,

상기 무작위 검색을 수행하는 단계는,

상기 희소화된 요약 그래프의 크기가 목표 크기보다 큰 경우, 상기 희소화된 요약 그래프의 크기가 요약 그래프 정보에 포함된 목표 크기에 도달할 때까지 상기 희소화된 요약 그래프에 대한 추가 희소화를 통해 요약 그래프를 획득하는 단계

를 포함하는 그래프 요약 방법.

청구항 5

제2항에 있어서,

상기 무작위 검색을 수행하는 단계는,

상기 대용량 그래프를 구성하는 각각의 수퍼 노드에서 노드 간 거리가 기 설정된 거리 이내에 존재하는 수퍼 노드들을 클러스터링하는 단계

를 포함하는 그래프 요약 방법.

청구항 6

그래프 요약 시스템에 있어서,

대용량 그래프에 대한 요약 그래프의 품질을 측정하기 위하여 비용 함수를 정의하는 비용 함수 정의부;

상기 정의된 비용 함수를 이용하여 비트 단위의 요약 그래프 정보를 설정하는 그래프 정보 설정부;

상기 설정된 비트 단위의 요약 그래프 정보에 기초하여 생성된 후보 요약 그래프에 대한 무작위 검색을 수행하는 검색부; 및

상기 수행된 무작위 검색을 통하여 획득된 요약 그래프를 결과로서 출력하는 요약 그래프 출력부

를 포함하는 그래프 요약 시스템.

청구항 7

제6항에 있어서,

상기 검색부는,

상기 대용량 그래프를 구성하는 수퍼 노드들을 클러스터링하여 후보군을 생성하는

것을 특징으로 하는 그래프 요약 시스템.

청구항 8

제7항에 있어서,

상기 검색부는,

상기 생성된 후보군에서 검색된 비용 함수가 감소하는 복수 개의 수퍼 노드를 반복적으로 병합함과 동시에 상기 병합을 통해 새로 생성된 수퍼 노드에 인접한 수퍼 엣지를 선택적으로 생성하여 요약 그래프를 희소화하는

것을 특징으로 하는 그래프 요약 시스템.

청구항 9

제8항에 있어서,

상기 검색부는,

상기 희소화된 요약 그래프의 크기가 목표 크기보다 큰 경우, 상기 희소화된 요약 그래프의 크기가 요약 그래프 정보에 포함된 목표 크기에 도달할 때까지 상기 희소화된 요약 그래프에 대한 추가 희소화를 통해 요약 그래프를 획득하는

것을 특징으로 하는 그래프 요약 시스템.

청구항 10

제7항에 있어서,

상기 검색부는,

상기 대용량 그래프를 구성하는 각각의 수퍼 노드에서 노드 간 거리가 기 설정된 거리 이내에 존재하는 수퍼 노드들을 클러스터링하는

것을 특징으로 하는 그래프 요약 시스템.

발명의 설명

기 술 분 야

[0001] 아래의 설명은 그래프를 요약하는 기술에 관한 것이다.

배 경 기 술

[0003] 그래프(graph)란 임의의 대상과 그 대상들 간의 관계를 나타내는 자료구조로서 다양하고 수많은 데이터들을 표현할 수 있다. 대표적인 예시로, 온라인 소셜네트워크(사용자간 연결), 월드와이드웹(웹페이지간의 연결), E-commerce에서의 상품 구매 내역(고객과 상품간의 연결), 심층학습 모델(뉴런간 연결) 등이 있다. 최근, 빅데이터 시대가 도래함에 따라, 대용량의 그래프 데이터들이 등장하고 있다. 예를 들어, 월드와이드웹은 55 억개 이상의 웹페이지를 연결하며, 페이스북은 24억명 이상의 사용자를 연결하고 있다.

[0004] 이러한 대용량 데이터는 컴퓨터의 메인 메모리나 캐시 메모리의 용량을 넘어서기 때문에 분석에 많은 어려움이 있다. 이는 대부분 그래프 분석 알고리즘이 그래프 전부를 메인 메모리에 저장하는 상황을 가정하기 때문이다. 이러한 문제를 해결하는 한 가지 방법은, 그래프 데이터가 메인 메모리에 저장될 수 있도록 압축하여 저장하는 것이다.

발명의 내용

해결하려는 과제

[0006] 대용량 그래프의 희소화 과정을 통해 요약 그래프를 생성하는 방법 및 시스템을 제공할 수 있다.

[0007] 비용 함수를 통해 요약 그래프를 비트 단위로 크기를 제한하면서 재구성 오류를 최소화하는 그래프 요약 방법 및 시스템을 제공할 수 있다.

과제의 해결 수단

[0009] 그래프 요약 방법은, 대용량 그래프에 대한 요약 그래프의 품질을 측정하기 위하여 비용 함수를 정의하는 단계; 상기 정의된 비용 함수를 이용하여 비트 단위의 요약 그래프 정보를 설정하는 단계; 상기 설정된 비트 단위의 요약 그래프 정보에 기초하여 생성된 후보 요약 그래프에 대한 무작위 검색을 수행하는 단계; 및 상기 수행된 무작위 검색을 통하여 획득된 요약 그래프를 결과로서 출력하는 단계를 포함할 수 있다.

[0010] 상기 무작위 검색을 수행하는 단계는, 상기 대용량 그래프를 구성하는 수퍼 노드들을 클러스터링하여 후보군을 생성하는 단계를 포함할 수 있다.

[0011] 상기 무작위 검색을 수행하는 단계는, 상기 생성된 후보군에서 검색된 비용 함수가 감소하는 복수 개의 수퍼 노드를 반복적으로 병합함과 동시에 상기 병합을 통해 새로 생성된 수퍼 노드에 인접한 수퍼 엣지를 선택적으로 생성하여 요약 그래프를 희소화하는 단계를 포함할 수 있다.

[0012] 상기 무작위 검색을 수행하는 단계는, 상기 희소화된 요약 그래프의 크기가 요약 그래프 정보에 포함된 목표 크기보다 큰 경우, 상기 희소화된 요약 그래프의 크기가 목표 크기에 도달할 때까지 상기 희소화된 요약 그래프에 대한 추가 희소화를 통해 요약 그래프를 획득하는 단계를 포함할 수 있다.

[0013] 상기 무작위 검색을 수행하는 단계는, 상기 대용량 그래프를 구성하는 각각의 수퍼 노드에서 노드 간 거리가 기설정된 거리 이내에 존재하는 수퍼 노드들을 클러스터링하는 단계를 포함할 수 있다.

[0014] 그래프 요약 시스템은, 대용량 그래프에 대한 요약 그래프의 품질을 측정하기 위하여 비용 함수를 정의하는 비용 함수 정의부; 상기 정의된 비용 함수를 이용하여 비트 단위의 요약 그래프 정보를 설정하는 그래프 정보 설정부; 상기 설정된 비트 단위의 요약 그래프 정보에 기초하여 생성된 후보 요약 그래프에 대한 무작위 검색을 수행하는 검색부; 및 상기 수행된 무작위 검색을 통하여 획득된 요약 그래프를 결과로서 출력하는 요약 그래프 출력부를 포함할 수 있다.

[0015] 상기 검색부는, 상기 대용량 그래프를 구성하는 수퍼 노드들을 클러스터링하여 후보군을 생성할 수 있다.

[0016] 상기 검색부는, 상기 생성된 후보군에서 검색된 비용 함수가 감소하는 복수 개의 수퍼 노드를 반복적으로 병합

함과 동시에 상기 병합을 통해 새로 생성된 수퍼 노드에 인접한 수퍼 엣지를 선택적으로 생성하여 요약 그래프를 희소화할 수 있다.

[0017] 상기 검색부는, 상기 희소화된 요약 그래프의 크기가 목표 크기보다 큰 경우, 상기 희소화된 요약 그래프의 크기가 요약 그래프 정보에 포함된 목표 크기에 도달할 때까지 상기 희소화된 요약 그래프에 대한 추가 희소화를 통해 요약 그래프를 획득할 수 있다.

[0018] 상기 검색부는, 상기 대용량 그래프를 구성하는 각각의 수퍼 노드에서 노드 간 거리가 기 설정된 거리 이내에 존재하는 수퍼 노드들을 클러스터링할 수 있다.

발명의 효과

[0020] 희소화 과정을 통해 요약 그래프를 생성함으로써 압축 속도와 압축률을 향상시킬 수 있다.

[0021] 그래프 압축 및 요약이라는 목적에 부합하는 비트(bit) 수 기반의 비용 함수를 제시하여, 가용한 메모리의 크기(bit 수)가 주어졌을 때, 해당 메모리에 정확하게 맞는 요약 그래프를 생성할 수 있다.

[0022] 대용량 그래프의 데이터를 적은 정보의 손실을 통하여 제한된 크기의 메인 메모리에 저장함으로써 그래프를 실시간으로 분석할 수 있다. 또한, 메모리가 제한된 엣지 컴퓨팅 환경에서도 대용량 그래프(학습된 심층 학습 모델 등)을 처리할 수 있다.

도면의 간단한 설명

[0024] 도 1은 일 실시예에 따른 그래프 요약 시스템의 구성을 설명하기 위한 블록도이다.

도 2는 일 실시예에 따른 그래프 요약 시스템에서 그래프 요약 방법을 설명하기 위한 흐름도이다.

도 3은 일 실시예에 따른 그래프 요약 시스템에서 그래프를 설명하기 위한 예이다.

도 4는 일 실시예에 따른 그래프 요약 시스템에서 요약 그래프 검색을 수행하는 방법을 설명하기 위한 흐름도이다.

도 5는 일 실시예에 따른 그래프 요약 시스템의 개괄적인 동작을 설명하기 위한 예이다.

도 6은 일 실시예에 따른 그래프 요약 시스템에서 요약 그래프를 검색하는 동작을 설명하기 위한 예이다.

발명을 실시하기 위한 구체적인 내용

[0025] 이하, 실시예를 첨부한 도면을 참조하여 상세히 설명한다.

[0027] 도 1은 일 실시예에 따른 그래프 요약 시스템의 구성을 설명하기 위한 블록도이고, 도 2는 일 실시예에 따른 그래프 요약 시스템에서 그래프 요약 방법을 설명하기 위한 흐름도이다.

[0028] 그래프 요약 시스템(100)은 비용 함수 정의부(110), 그래프 정보 설정부(120), 검색부(130) 및 요약 그래프 출력부(140)를 포함할 수 있다. 이러한 그래프 요약 시스템(100)의 구성요소들은 그래프 요약 시스템(100)에 저장된 프로그램 코드가 제공하는 명령에 따라 프로세서에 의해 수행되는 서로 다른 기능들(different functions)의 표현들일 수 있다. 그래프 요약 시스템(100) 및 그래프 요약 시스템(100)의 구성요소들은 도 2의 그래프 요약 방법이 포함하는 단계들(210 내지 240)을 수행하도록 그래프 요약 시스템(100)을 제어할 수 있다. 이때, 그래프 요약 시스템(100) 및 그래프 요약 시스템(100)의 구성요소들은 메모리가 포함하는 운영체제의 코드와 적어도 하나의 프로그램의 코드에 따른 명령(instruction)을 실행하도록 구현될 수 있다.

[0029] 그래프 요약 시스템(100)은 그래프 요약 방법을 위한 프로그램의 파일에 저장된 프로그램 코드를 메모리에 로딩할 수 있다. 예를 들면, 그래프 요약 시스템(100)에서 프로그램이 실행되면, 프로세서는 운영체제의 제어에 따라 프로그램의 파일로부터 프로그램 코드를 메모리에 로딩하도록 그래프 요약 시스템을 제어할 수 있다. 이때, 그래프 요약 시스템(100) 및 그래프 요약 시스템(100)이 포함하는 비용 함수 정의부(110), 그래프 정보 설정부(120), 검색부(130) 및 요약 그래프 출력부(140) 각각은 메모리에 로딩된 프로그램 코드 중 대응하는 부분의 명

령을 실행하여 이후 단계들(210 내지 240)을 실행하기 위한 그래프 요약 시스템(100)의 서로 다른 기능적 표현들일 수 있다.

[0030] 단계(210)에서 비용 함수 정의부(110)는 대용량 그래프에 대한 요약 그래프의 품질을 측정하기 위하여 비용 함수를 정의할 수 있다. 이때, 대용량 그래프는 용량이 큰 입력 그래프 또는 요약 그래프를 의미할 수 있다.

[0031] 단계(220)에서 그래프 정보 설정부(120)는 정의된 비용 함수를 이용하여 비트 단위의 요약 그래프 정보를 설정할 수 있다.

[0032] 단계(230)에서 검색부(130)는 설정된 비트 단위의 요약 그래프 정보에 기초하여 생성된 후보 요약 그래프에 대한 무작위 검색을 수행할 수 있다. 도 4를 참고하면, 요약 그래프를 검색하는 방법을 설명하기 위한 흐름도이다. 단계(410)에서 검색부(130)는 대용량 그래프를 구성하는 각각의 수퍼 노드에서 기 설정된 거리 이내에 존재하는 수퍼 노드들을 클러스터링하여 후보군을 생성할 수 있다. 단계(420)에서 검색부(130)는 생성된 후보군에서 검색된 비용 함수가 감소하는 복수 개의 수퍼 노드를 반복적으로 병합함과 동시에 병합을 통해 새로 생성된 수퍼 노드에 인접한 수퍼 엣지를 선택적으로 생성하여 요약 그래프를 희소화(Sparsification)할 수 있다. 이때, 희소화란, 수퍼 노드에 인접한 수퍼 엣지를 선택적으로 연결 또는 삭제함으로써 드문드문 노드들이 연결되도록 하는 것을 의미할 수 있다. 각 수퍼 엣지는 비용 함수를 감소시킬 때만 생성될 수 있다. 단계(430)에서 검색부(130)는 희소화된 요약 그래프의 크기가 목표 크기보다 큰 지 여부를 판단할 수 있다. 검색부(130)는 희소화된 요약 그래프의 크기가 목표 크기보다 크지 않다면(또는 작다면), 프로세스를 종료할 수 있다. 단계(440)에서 검색부(130)는 희소화된 요약 그래프의 크기가 목표 크기보다 클 경우, 희소화된 요약 그래프에 대한 추가 희소화를 수행할 수 있다. 또한, 검색부(130)는 추가 희소화를 수행함에 따라 단계(430)를 재수행할 수 있다. 검색부(130)는 희소화된 요약 그래프의 크기가 목표 크기에 도달할 때까지 희소화된 요약 그래프에 대한 추가 희소화를 통해 요약 그래프를 획득할 수 있다. 이와 같이, 수퍼 엣지를 반복적으로 제거함으로써 비용 함수가 최소화될 수 있다.

[0033] 단계(240)에서 요약 그래프 출력부(140)는 수행된 무작위 검색을 통하여 획득된 요약 그래프를 결과로서 출력할 수 있다.

[0034] 도 3은 일 실시예에 따른 그래프 요약 시스템에서 그래프를 설명하기 위한 예이다.

[0035] 우선적으로, 그래프에 대하여 설명하기로 한다. 입력 그래프는 노드 V와 엣지 E가 있는 비방향 그래프 G = (V, E)일 수 있다. 각 엣지 {u, v} ∈ E는 두 개의 서로 다른 노드인 u, v ∈ V에 결합할 수 있다. G가 단순성을 위해 셀프 루프(self-loop) 없이 방향을 바꾸지 않는 반면, 실시예에서 제안되는 방법은 자체 루프와 함께 지시된 그래프로 쉽게 확장될 수 있다고 가정하기로 한다. 아래에서 설명되는 요약 그래프와 구별하기 위해 노드와 엣지를 서브 노드와 서브 엣지라고 부르기로 한다.

[0036] 그래프 G = (V, E)의 요약 그래프 $\bar{G}$ = (S, P, ω)는 수퍼 노드의 집합 S, 수퍼 엣지의 집합 P, 가중치 함수 ω로 구성될 수 있다. 수퍼 노드 S는 서브 노드 V와 구별되고 철저한 부분 집합이다. 즉, 모든 $A \neq B \in S$에서 $\cup_{A \in S} A = V$ 및 $A \cap B = \emptyset$ 이다. V의 모든 서브 노드는 정확히 하나의 S의 수퍼 노드에 포함되어 있으며, 각 서브 노드 $v \in V$가 $V_v \in S$로 속해 있는 수퍼 노드를 나타낸다. 각 수퍼 엣지 $\{A, B\} \in P$는 두 개의 수퍼 노드 $A, B \in S$를 연결하며, 만약 A = B이면 {A, B} = {A, A}는 수퍼 노드 $A \in S$에서 셀프 루프를 나타낸다. 무질서한 모든 수퍼 노드 쌍을 나타내기 위해 $\Pi_S := \binom{S}{2} \cup \{\{A, A\} : A \in S\}$ 를 활용하면, $P \subseteq \Pi_S$이다. 가중치 함수 $\omega : P \to \mathbb{Z}^+$는 각각 A ∈ S와 B ∈ S에 속하는 두 서브 노드 사이의 서브 엣지의 수를 각 수퍼 엣지 {A, B} ∈ P에 할당한다. 각 서브 엣지의 집합을 $E_{AB} := \{\{u, v\} \in E : u \in A, v \in B\}$로 기재하기로 한다. 그 다음 $\{\{u, v\} \in E : u \in A, v \in B\}$와 같은 서브 엣지 집합을 두면, 각 수퍼 엣지 $\{A, B\} \in P$에서 $\omega(\{A, B\}) := |E_{AB}|$ 이다.

[0037] 요약 그래프 $\bar{G}$ = (S, P, ω)를 고려하여, 재구성된 그래프 $\hat{G} = (V, \hat{E}, \hat{\omega})$가 획득될 수 있다. 서브 노드의 집

합 V는 S의 모든 수퍼 노드의 조합에 의해 복구될 수 있다. 서브 엣지의 설정 $\hat{E}$는 P에서 수퍼 엣지에 의해 연결된 두 개의 수퍼 노드에 속하는 모든 쌍의 뚜렷한 서브 노드의 집합으로 정의된다. 즉, $\hat{E} := \{\{u,v\} \in V \times V : u \neq v, \{V_u, V_v\} \in P\}$ 이다. 가중치 함수 $\hat{\omega} : \hat{E} \rightarrow \mathbb{R}^+$ 는 다음과 같이 정의될 수 있다.

[0038] 수학식 1:

$$\hat{\omega}(\{u,v\}) := \frac{\omega(\{V_u, V_v\})}{|\Pi_{V_u V_v}|} = \frac{|E_{V_u V_v}|}{|\Pi_{V_u V_v}|}$$

[0039]

[0040] 여기서 $\Pi_{V_u V_v} := \{\{u,v\} : u \neq v, u \in V_u, v \in V_v\}$ 는 두 수퍼 노드 각각에 포함되는 두 서브 노드 사이에서 가능한 모든 서브 엣지의 집합이다. 수학식 1에서, 분모는 두 개의 수퍼 노드 사이의 최대 서브 엣지의 수이며, 각 분자는 두 개의 수퍼 노드 사이의 실제 서브 엣지의 수이다. 이때, $\overline{G}$에서 재구성된 그래프 $\hat{G}$가 원래 그래프 G와 반드시 같은 것은 아니다.

[0041] 원래 그래프 G의 인접 행렬을 나타내기 위해 A를 활용하고, 재구성된 그래프 $\hat{G}$의 가중된 인접 행렬을 나타내기 위해 $\hat{A}$를 활용할 수 있다.

[0042] 도 3을 참고하면, (a)는 입력 그래프 G, (b)는 인접 행렬 A, (c)는 요약 그래프 $\overline{G}$, (d)는 복원된 인접 행렬 $\hat{A}$를 나타낸 것이다. (a)에서의 그래프 G는 (b)에서 인접 행렬 A를 갖는다. (c)의 요약 그래프 $\overline{G}$에서 가중 인접 행렬 $\hat{A}$이 (d) 에 있는 그래프 $\hat{G}$를 복원할 수 있다. 그래프 G의 각 서브 노드는 그래프 $\hat{G}$의 하나의 수퍼 노드에 속하며, 각 수퍼 엣지의 가중치는 두 개의 수퍼 노드 사이의 서브 엣지 수에 대응될 수 있다. 예를 들면, 수퍼 노드 V1과 V3 사이에 3 개의 서브 엣지(즉, {1, 6}, {2, 3}, {2, 6})가 있으므로, 수퍼 에지{V1, V3}의 가중치 ω ({V1, V3})는 3이다. 두 개의 수퍼 노드 사이에 서브 엣지가 있는 경우에도 수퍼 엣지에 의해 연결될 필요가 없다(V3 및 V7 참조). 수학식 1을 통해 $\hat{G}$에서 수퍼 엣지의 가중치를 도출할 수 있다. 예를 들어, $\hat{G}$에서 수퍼 엣지 {1, 3}의 가중치 ω ({1, 3})는 $\frac{\omega(\{V_1, V_3\})}{|\Pi_{V_1 V_3}|} = \frac{3}{8}$ 이다.

[0043] 도 5는 일 실시예에 따른 그래프 요약 시스템의 개괄적인 동작을 설명하기 위한 예이다.

[0044] 도 5는 대용량 그래프의 희소화 요약 동작을 설명하기 위한 알고리즘을 도시한 예이다. 그래프 요약 시스템에도 5의 희소화 동작을 설명하기 위한 알고리즘이 탑재될 수 있다. 그래프 요약 시스템은 세 가지의 특징이 탑재된 무작위 그리디(greedy) 검색을 수행할 수 있다.

[0045] 그래프 요약 시스템은 검색의 각 단계에서 재구성 오류의 변화와 요약 그래프의 크기를 신중하게 균형을 맞추기 위하여, 최소 설명 길이 원칙(MDL 원칙)을 적용하여 재구성 오류와 비트의 개수에 공통적으로 발생하는 크기를 모두 측정할 수 있다. 그런 다음, 총 비트 수를 최소화하기 위하여 무작위 검색을 수행할 수 있다.

[0046] 그래프 요약 시스템은 요약에 대한 두 가지 전략을 긴밀하게 결합할 수 있다. 다시 말해서, 그래프 요약 시스템은 수퍼 노드를 단일 수퍼 노드로 병합하는 것과 요약 그래프를 희소화하는 것이다. 그래프 요약 시스템은 가중치가 0이 아닌 한, 가능한 모든 수퍼 엣지를 생성하는 대신에, 선택적으로 수퍼 엣지를 생성하여 비용 함수가 최소화되도록 한다. 그래프 요약 시스템은 수퍼 노드를 병합하기로 결정할 때, 선택적인 수퍼 엣지 생성을 고려할 수 있다.

[0047] 그래프 요약 시스템은 통합될 유망한 수퍼 노드 후보 쌍을 신속하고 효과적으로 찾아 선형적 확장성을 달성할 수 있다.

[0048] 우선적으로, 검색의 각 단계에서 사용되는 비용 함수에 대하여 설명하기로 한다. 비용 함수는 크기와 재구성 오류의 균형을 맞춰 후보 요약 그래프의 품질을 측정하는 것으로, 재구성 오류를 가능한 적게 증가시키면서 출

력 요약 그래프의 크기를 줄이는 것을 목표로 하기 때문에 중요하다. 여기서, 재구성 오류란 그래프 G와 요약 그래프 $\overline{G}$ 에서 재구성된 그래프 $\hat{G}$ 사이의 차이를 의미한다.

[0049] 크기와 재구성 오류의 균형을 맞추기 위해서 직접 비교할 수 있어야 한다. 그래프 요약 시스템은 최소 설명 길이 원칙을 적용하여 비트 수 측면에서 크기와 재구성 오류를 포함하는 두 가지를 모두 측정할 수 있다. 그래프 요약 시스템은 입력 그래프 G이고, 요약 그래프 $\overline{G}$ 에서 비용($\overline{G}$, G)를 최소화하는 모델을 구성할 수 있다. 이와 같이 비용을 최소화하는 모델을 입력 그래프 G의 비트로 정의될 수 있다.

[0050] 수학식 2:

[0051] $$Cost(\overline{G}, G) := Cost(\overline{G}) + Cost(G|\overline{G})$$

[0052] 여기서 설명 길이는 모델 비용 Cost(G)와 데이터 비용 $Cost(G|\overline{G})$로 나눈다. 모델 비용은 G를 설명하는 데 필요한 비트 수를 측정할 수 있다. 데이터 비용은 G가 주어지는 $\overline{G}$ 를 설명하는 데 필요한 비트 수를 측정하거나 G에서 재구성되는 G와 $\hat{G}$의 차이를 동등하게 설명하는 데 필요한 비트 수를 측정할 수 있다. 따라서 데이터 비용은 당연히 G의 재구성 오류(비트 단위)로 해석될 수 있다. 재구성의 오류 없이 G = $\hat{G}$이면 $Cost(G|\overline{G})$은 0이 된다는 점에 유의한다.

[0053] 수학식 2는 그래프 요약 시스템이 크기와 재구성 오류를 균형 있게 조정하여 후보 요약 그래프의 품질을 측정하기 위해 사용하는 비용 함수이다. 이하에서는 수학식 2에 대한 각 용어를 상세히 설명한 다음, 각 수퍼 노드에 대한 비용으로 나누어 설명하기로 한다.

[0054] 모델 비용은 수학식 3을 사용할 수 있다.

[0055] 수학식 3:

[0056] $$Cost(\overline{G}) := |P|(2\log_2|V| + \log_2|E|) + |V|\log_2|V|$$

[0057] 수학식 3은 요약 그래프의 크기를 비트 단위로 측정하는 식에 기반한 것으로, 수학식 3에서 $\log_2|V|\ (\geq \log_2|S|)$ 와 $\log_2|E|\ (\geq \log_2\omega_{max})$ 비트는 각각 수퍼 노드와 수퍼 엣지를 구별하기 위하여 사용될 수 있다. 총 모델 비용을 각 수퍼 노드 쌍에 대한 모델 비용으로 다음과 같이 나눌 수 있다.

[0058] 수학식 4:

[0059] $$Cost(\overline{G}) = |V|\log_2|V| + \sum_{\{A,B\}\in\Pi_S} Cost(\{A,B\}|\overline{G})$$

[0060] 여기서, $Cost(\{A,B\}|G) := \mathbb{1}(\{A,B\} \in P) \times (2\log_2|V| + \log_2|E|)$ 는 각 수퍼 노드 쌍 $\{A,B\} \in \Pi_S$에서의 모델 비용이다.

[0061] 데이터 비용 $Cost(G|\overline{G})$은 G를 정확히 설명하기 위해 필요한 비트 수 또는 G로 주어진 $\overline{G}$ 로부터 G를 나타내기 위해 추가적으로 필요한 비트 수를 의미한다. 앞서 설명한 것과 같이, $Cost(G|\overline{G})$는 비트로 된 $\overline{G}$ 의 재구성 오류로 해석될 수 있다. 총 데이터 비용을 각 수퍼 노드 쌍에 대한 데이터 비용으로 다음과 같이 나눌 수 있다.

[0062] 수학식 5:

[0063] $$Cost(G|\overline{G}) = \sum_{\{A,B\}\in\Pi_S} Cost(E_{AB}|\overline{G})$$

[0064] 여기서, $Cost(E_{AB}|\overline{G})$ 는 수퍼 노드 A와 B(즉, E_{AB}) 사이의 서브 엣지를 설명하는데 필요한 비트 수이다. 각 $Cost(E_{AB}|\overline{G})$ 에 대해, 수퍼 엣지 {A, B}가 존재하거나 존재하지 않을 경우를 모두 고려하기 위하여 이중 인코딩 방법을 가정하기로 한다. 구체적으로, 수퍼 엣지 {A, B}가 $\overline{G}$ 에 존재하는 경우, 첫 번째 인코딩 방법이 사용되며 Π_{AB} 에서 가능한 각 서브 엣지가 존재하는지 여부를 나타내기 위해 비트를 최적으로 할당할 수 있다. 그런 다음, 필요한 비트 수는 Shannon 엔트로피에 의해 엄격하게 제한될 수 있다. 이에 따라, $Cost(E_{AB}|\overline{G})$ 를 다음과 같이 정의할 수 있다.

[0065] 수학식 6:

[0066] $$Cost_{(1)}(E_{AB}|\overline{G}) := -|\Pi_{AB}|(\sigma \log_2 \sigma + (1-\sigma)\log_2(1-\sigma))$$

[0067] 여기서, $\sigma := \frac{|E_{AB}|}{|\Pi_{AB}|}$ 는 Π_{AB} 에서 존재하는 서브 엣지의 비율이다. σ 를 계산하려면 수퍼 엣지 {A, B}와 가중치 ω ({A, B})를 $\overline{G}$ 로 유지해야 한다. 수퍼 엣지 {A, B}가 $\overline{G}$ 에 존재하지 않을 경우, 두 번째 인코딩 방법을 사용하며, E_{AB}에 기존의 모든 서브 엣지만 나열할 수 있다. 그러면 필요한 비트 수는 다음과 같다.

[0068] 수학식 7:

[0069] $$Cost_{(2)}(E_{AB}|\overline{G}) := 2|E_{AB}|\log_2|V|$$

[0070] 여기서, $2\log_2|V|$는 서브 엣지를 인코딩하는데 필요한 숫자 비트이다. 이 인코딩 방법의 경우, 수퍼 엣지 {A, B}와 가중치 ω ({A, B})를 $\overline{G}$ 에 유지할 필요가 없다. 그러면, E_{AB}를 설명하는데 필요한 최종 비트 수는 다음과 같이 나타낼 수 있다.

[0071] 수학식 8:

[0072] $$Cost(E_{AB}|\overline{G}) := \begin{cases} Cost_{(1)}(E_{AB}|\overline{G}) & \text{if } \{A,B\} \in P \\ Cost_{(2)}(E_{AB}|\overline{G}) & \text{otherwise.} \end{cases}$$

[0073] 비용 분해는 수학식 2와 수학식 8을 결합함으로써 총 기술 비용 $Cost(\overline{G}, G)$ 은 다음과 같이 각 수퍼 노드 쌍에 대한 비용으로 나눌 수 있다.

[0074] $$Cost(\overline{G}, G) = |V|\log_2|V| + \sum_{\{A,B\}\in\Pi_S} Cost_{AB}(\overline{G}, G)$$

[0075] 여기서, 각 수퍼 노드 쌍 {A, B}의 $Cost(\overline{G}, G)$, 총 설명 비용은 다음과 같이 정의될 수 있다.

[0076] 수학식 9:

[0077] $$Cost_{AB}(\overline{G}, G) := Cost(\{A,B\}|\overline{G}) + Cost(E_{AB}|\overline{G})$$

[0078] 또한, 설명 비용을 바탕으로 수퍼 노드 A를 포함하는 쌍에 대한 비용을 합산하여 각 수퍼 노드 A의 총 설명 비용을 정의할 수 있다.

[0079] 수학식 10:

[0080] $$Cost_A(\overline{G}, G) := \sum_{B\in S} Cost_{AB}(\overline{G}, G)$$

[0081] 수학식 10은 병합할 수퍼 노드를 결정할 때 사용될 수 있다.

[0082] 수퍼 노드 집합이 지정된 최적의 인코딩 방법을 설명하기로 한다. 일단 수퍼 노드의 집합 S가 고정되면, 수퍼 노드의 각 쌍 {A, B}$\in \Pi_S$에 대하여 수학식 9를 최소화함으로써 수학식 2를 최소화하는 수퍼 엣지의 집합 P를 획득할 수 있다. 다시 말해서, 각 쌍 {A, B} 사이의 수퍼 엣지는 수학식 9를 줄이는 경우에만 생성될 수 있다. P(S)를 주어진 S에 따라 수학식 2를 최소화하는 수퍼 엣지 집합으로 하고 $\overline{G}$ (S) = (S, P(S), ω)를 S와 P(S)로 구성된 요약 그래프로 한다. 수학식 2를 최소화하는 것은 최소화하는 S를 찾는 것과 같다.

[0083] 수학식 11:

[0084] $$Cost^{\star}(S) := Cost(\bar{G}^{\star}(S)) + Cost(G|\bar{G}^{\star}(S))$$

[0085] 마찬가지로 수학식 9 및 수학식 10에서와 같이, $\overline{G}$ (S)에서 각 수퍼 노드 쌍 {A, B} $\in \Pi_S$과 수퍼 노드 A$\in$ S의 설명 비용을 다음과 같다고 한다.

[0086] 수학식 12:

[0087] $$Cost^{\star}_{AB}(S) := Cost_{AB}(\bar{G}^{\star}(S), G),$$

[0088] 수학식 13:

[0089] $$Cost^{\star}_{A}(S) := \sum_{B \in S} Cost^{\star}_{AB}(S).$$

[0090] 그래프 요약 시스템은 후보 요약 그래프의 품질을 측정하기 위한 비용 함수(수학식 11)을 정의하였으므로, 후보 요약 그래프에 대해 신속하고 효과적인 무작위 검색을 수행하는 방법을 제공할 수 있다.

[0091] 입력 그래프 G = (V, E), 요약 그래프 비트의 원하는 크기 k, 반복 횟수 T가 주어지면, 그래프 요약 시스템은 요약 그래프 $\overline{G}$ = (S, P, ω)를 생성할 수 있다. 그래프 요약 시스템은 먼저 $\overline{G}$에서 G로 초기화할 수 있다. 다시 말해서, 도5의 1-2 행을 참고하면 S = {{u} : u $\in$ V} 및 P = {{Vu,Vv} $\in$ S×S : {u,v} $\in$ E}이다. 그런 다음, 요약 그래프의 크기가 k에 도달하거나 반복 횟수가 T에 도달할 때까지 두 과정을 번갈아 실행하여 수퍼 노드 쌍을 반복적으로 병합하고 요약 그래프를 희소화할 수 있다.

[0092] 그래프 요약 시스템은 후보 요약 그래프를 생성할 수 있다(5행). 그래프 요약 시스템은 병합으로 비용 함수가 현저히 감소하는 유망한 수퍼 노드 쌍을 신속하고 효과적으로 검색하기 위해 먼저 S를 후보군(후보 집합) S_t로 나누며 각각 기 설정된 거리(예를 들면, 2홉(hop)) 이내의 수퍼 노드를 구성할 수 있다. 더 많은 쌍의 수퍼 노드를 고려하기 위해 각 반복 t 에서 S_t를 확률적으로 변경할 수 있다.

[0093] 그래프 요약 시스템은 요약 그래프를 병합 및 희소화할 수 있다(6-7행). 그래프 요약 시스템은 이전 단계에서 획득한 각 후보군 내에서 병합으로 비용 함수가 가장 많이 감소하는 두 개의 수퍼 노드를 반복적으로 병합할 수 있다. 동시에, 그래프 요약 시스템은 새로 생성된 수퍼 노드에 인접한 수퍼 엣지를 선택적으로 생성하여 요약 그래프를 희소화할 수 있다. 이때, 각각의 수퍼 엣지는 비용 함수를 줄일 때만 생성될 수 있다. 그 후, 요약 그래프의 크기가 지정된 목표 크기 k보다 여전히 크면, 다음 단계가 실행될 수 있다.

[0094] 그래프 요약 시스템은 추가 희소화를 수행할 수 있다(11-12행). 그래프 요약 시스템은 요약 그래프의 크기가 지정된 목표 크기 k에 도달할 때까지 더 희소화할 수 있다. 특히, 그래프 요약 시스템은 비용 함수를 최소화하기 위해 수퍼 엣지를 반복적으로 제거할 수 있다.

[0095] 최종적으로, 그래프 요약 시스템은 요약 그래프를 출력으로 반환할 수 있다. 이때, 요약 그래프는 가중 그래프일 수 있다.

[0096] 도 6은 일 실시예에 따른 그래프 요약 시스템에서 요약 그래프를 검색하는 동작을 설명하기 위한 예이다.

[0097] 그래프 요약 시스템은 수퍼 노드의 후보군(후보 요약 그래프)을 찾을 수 있다. 그래프 요약 시스템은 병합 및

희소화를 통해 후보군을 사용하고, 각 후보군 내에서 병합할 수퍼 노드 쌍을 검색할 수 있다. 신속하고 효과적인 검색을 위하여 후보군은 작아야 하며 동시에 비용 함수의 대폭적인 감소로 이어지는 많은 수퍼 노드 쌍(예를 들면, 수학식 11)을 포함해야 한다.

[0098] 그래프 요약 시스템은 후보군을 찾기 위하여 각각의 수퍼 노드들 간 거리가 기 설정된 기준(예를 들면, 2홉) 이내에서 수퍼 노드를 클러스터링할 수 있다. 또는, 두 개의 수퍼 노드 사이의 거리를 각 수퍼 노드에 속하는 서브 노드 사이의 최소 거리로 정의한다면, 2 홉 거리보다 먼 수퍼 노드를 병합하는 것보다 (2 홉 내에서) 비용 함수를 더 감소시킬 수 있다.

[0099] 수학식 14:

[0100]
$$Reduction(A,B) := Cost^{\star}_{A}(S) + Cost^{\star}_{B}(S) - Cost^{\star}_{AB}(S) - Cost^{\star}_{A\cup B}(S \cup \{A \cup B\} \setminus \{A,B\})$$

[0101] 수학식 14는 두 개의 수퍼 노드 $A \neq B \in S$ 가 병합되었을 때 수학식 11의 비용 함수를 감소시키는 정도를 나타낸다.

[0102] 그래프 요약 시스템은 2개의 수퍼 노드 A ∈ S와 B ∈ S가 2홉 이내에 있으면, 수학식 15이고, 해당 부등식은 엄격하다.

[0103] 수학식 15:

[0104]
$$Reduction(A,B) \leq \min(Cost^{\star}_{A}(S), Cost^{\star}_{B}(S))$$

[0105] 그래프 요약 시스템은 2개의 수퍼 노드 A ∈ S와 B ∈ S가 서로 3홉 이상 떨어져 있을 경우, 수학식 16이다.

[0106] 수학식 16:

[0107]
$$Reduction(A,B) \leq 2\log_2|V| + \log_2|E|$$

[0108] 선택된 2홉 이내의 $A \neq B \in S$ 의 경우, $\min(Cost^{\star}_{A}(S), Cost^{\star}_{B}(S))$ 및 Reduction(A, B)는 $2\log_2|V| + \log_2|E|$ 보다 훨씬 크다.

[0109] 그래프 요약 시스템은 수퍼 노드를 서로 2홉 이내에 신속하게 그룹화하기 위해 수퍼 노드를 동일한 shingle을 가진 것으로 나눈다. 각 수퍼 노드 A ∈ S의 shingle을 다음과 같이 정의하면 무작위 일대일 대응함수 h에 대해 V *{1, ...,|V|}이다.

[0110]
$$f(A) := \min_{u\in A}\left(\min\left(h(u), \min_{(u,v)\in E} h(v)\right)\right)$$

[0111] 두 개의 수퍼 노드 $A \neq B \in S$ 는 A와 B가 2홉 이내에 있는 경우에만 동일한 shingle(즉, f(A) = f(B))를 갖는다. 구체적으로는 각 후보군이 최대 상수(사양, 500) 노드 수로 구성될 때까지 shingle을 사용하여 최대 상수(사양, 10) 번 반복적으로 수퍼 노드를 나눈다. 모든 수퍼 노드의 shingle을 계산하는 데는 O(|V| + |E|) 시간이 걸리며 무작위 해시함수 h를 생성하면 O(|V|) 시간이 소요되고, 모든 서브 노드 u ∈ V에 대해 O(|V| + |E|) 시간이 걸리는 $\min(h(u), \min_{(u,v)\in E} h(v))$를 계산하고 저장하며, 모든 수퍼 노드 A ∈ S에 대해 f(A)를 계산하고 이는 O(|V|) 시간이 걸린다는 점에 유의한다.

[0112] 그래프 요약 시스템은 동일한 후보군 내의 두 개의 수퍼 노드를 단일 수 퍼 노드로 반복적으로 병합하고, 그리디(greedy)하게 인접한 수퍼 엣지를 희소화하여 간결하고 정확한 요약 그래프를 검색할 수 있다. 이를 위해, 이전 단계에서 획득된 각 후보군 집합 C를 도 6에서 설명한 대로 처리할 수 있다. 그래프 요약 시스템은 먼저 $\log_2|C|$ 중 무작위로 선택한 C의 수퍼 노드 쌍 두 개 $A \neq B \in S$ 를 찾아낼 수 있다.

[0113] 수학식 17:

$$Relative_Reduction(A,B) := 1 - \frac{Cost^{\star}_{A\cup B}(S \cup \{A \cup B\} \setminus \{A,B\})}{Cost^{\star}_{A}(S) + Cost^{\star}_{B}(S) - Cost^{\star}_{AB}(S)}$$

[0114]

[0115] 다시 말해서, A와 B에 인접한 서브 엣지를 설명하는 현재의 비용에 대해 A와 B의 병합으로 인한 비용 함수(수학식 10)의 감소이다. 수학식 17이 임계값(4행)을 초과하면, A와 B가 단일 수퍼 노드 A ∪ B(5행)로 병합될 수 있다. 그래프 요약 시스템은 예를 들면, 시뮬레이션을 통해 반복하여 임계값을 감소시킬 수 있다.

[0116] 수학식 18:

$$\theta(t) := \begin{cases} (1+t)^{-1} & \text{if } t < T \\ 0 & \text{if } t = T \end{cases}$$

[0117]

[0118] 여기서 t는 현재 반복 번호를 나타낸다. A와 B를 A ∪ B로 병합하면 A나 B에 인접한 모든 수퍼 엣지가 제거(6행)되고, A ∪ B에 인접한 수퍼 엣지는 선택적으로 생성(또는 균등하게 희소화)되어 S에 주어진 비용 함수(즉, 식 (13)에 정의된 $Cost_{A\cup B}(\overline{G}, G)$)가 최소화될 수 있다. 후보군 C에서 두 개의 수퍼 노드를 병합하는 것은 상대적 감소(예: 16 (20))가 임계값 Θ(t)를 초과하지 않을 때까지, (1, 2, 8)행에서 max($\log_2|C|$, 1)번 반복될 수 있다. 그런 다음, 다른 후보군도 각각 동일한 방식으로 처리될 수 있다.

[0119] 그래프 요약 시스템은 각 후보군에서 소수의 수퍼 노드 쌍으로 주의를 제한함으로써 검색 공간을 크게 줄이고 선형 확장성을 달성할 수 있다. 그러나, 이러한 감소는 후보군의 신중한 형성, 적응 임계 값 Θ(t) 및 max($\log_2|C|$,1) 확률의 강력한 종료 때문에 출력 요약 그래프의 품질을 크게 해치지 않는다.

[0120] 그래프 요약 시스템은 병합 및 희소화 과정을 T 시간 동안 반복한 후 요약 그래프의 크기가 지정된 목표 크기 k(알고리즘 1의 11-12행)를 여전히 초과하는 경우, 추가 희소화 과정을 실행할 수 있다. 추가 희소화 과정에서 그래프 요약 시스템은 다음과 같이 Size($\overline{G}$)가 k에 도달할 때까지 요약 그래프를 희소화할 수 있다.

[0121] (1)각 수퍼 엣지를 P에서 떨어뜨린 후 재구성 오류 RE_P의 증가를 계산할 수 있다. 비용 함수 대신 RE_P가 직접 사용된다는 점에 유의한다. 왜냐하면 각 수퍼 엣지를 떨어뜨린 후의 Size(G)의 감소는 가중치 ω_{max}를 제외한 상수(사양., $2\log_2|S| + \log_2\omega_{max}$)이기 때문이다.

[0122] (2) RE_P에서 $\xi := \lceil \frac{Size(\overline{G}) - k}{2\log_2|S| + \log_2\omega_{max}} \rceil$ 번째로 작은 증가를 찾고 Δ_ξ 이 되게 한다.

[0123] (3) P의 각 수퍼 엣지에 대해 REP의 증가가 Δ_ξ 이하라면, 수퍼 엣지를 떨어뜨린다.

[0124] 일 실시예에 따른 그래프 요약 시스템은 요약 그래프를 저장하기 위해 필요한 비트(bit) 수를 비용 함수로 사용할 수 있다. 이에, 그래프 압축 및 요약이라는 목적에 부합하는 비트 수 기반 비용 함수를 제시함으로써, 가용한 메모리의 크기(비트 수)가 주어졌을 때, 해당 메모리에 정확하게 맞는 요약 그래프를 생성할 수 있다.

[0125] 일 실시예에 따른 그래프 요약 시스템은 노드들을 병합하는 것과 동시에 중요도가 낮은 엣지(간선)들을 제거하는 희소화 과정을 사용함으로써 정보 손실 양 대비 요약 그래프의 크기(비트 수 기준)를 개선할 수 있다.

[0126] 일 실시예에 따른 그래프 요약 시스템은 하나로 병합할 노드들을 결정할 때, 탐색 공간을 거리가 2이하인 노드들의 쌍으로 제한하는 방식을 통해, 속도를 개선할 수 있다. 또한, 선형 시간 복잡도를 가지게 됨으로써 수 많은(예를 들면, 8억 개 이상의) 간선을 포함한 그래프를 요약할 수 있다.

[0127] 이상에서 설명된 장치는 하드웨어 구성요소, 소프트웨어 구성요소, 및/또는 하드웨어 구성요소 및 소프트웨어 구성요소의 조합으로 구현될 수 있다. 예를 들어, 실시예들에서 설명된 장치 및 구성요소는, 예를 들어, 프로

세서, 콘트롤러, ALU(arithmetic logic unit), 디지털 신호 프로세서(digital signal processor), 마이크로컴퓨터, FPGA(field programmable gate array), PLU(programmable logic unit), 마이크로프로세서, 또는 명령(instruction)을 실행하고 응답할 수 있는 다른 어떠한 장치와 같이, 하나 이상의 범용 컴퓨터 또는 특수 목적 컴퓨터를 이용하여 구현될 수 있다. 처리 장치는 운영 체제(OS) 및 상기 운영 체제 상에서 수행되는 하나 이상의 소프트웨어 애플리케이션을 수행할 수 있다. 또한, 처리 장치는 소프트웨어의 실행에 응답하여, 데이터를 접근, 저장, 조작, 처리 및 생성할 수도 있다. 이해의 편의를 위하여, 처리 장치는 하나가 사용되는 것으로 설명된 경우도 있지만, 해당 기술분야에서 통상의 지식을 가진 자는, 처리 장치가 복수 개의 처리 요소(processing element) 및/또는 복수 유형의 처리 요소를 포함할 수 있음을 알 수 있다. 예를 들어, 처리 장치는 복수 개의 프로세서 또는 하나의 프로세서 및 하나의 콘트롤러를 포함할 수 있다. 또한, 병렬 프로세서(parallel processor)와 같은, 다른 처리 구성(processing configuration)도 가능하다.

[0128] 소프트웨어는 컴퓨터 프로그램(computer program), 코드(code), 명령(instruction), 또는 이들 중 하나 이상의 조합을 포함할 수 있으며, 원하는 대로 동작하도록 처리 장치를 구성하거나 독립적으로 또는 결합적으로(collectively) 처리 장치를 명령할 수 있다. 소프트웨어 및/또는 데이터는, 처리 장치에 의하여 해석되거나 처리 장치에 명령 또는 데이터를 제공하기 위하여, 어떤 유형의 기계, 구성요소(component), 물리적 장치, 가상 장치(virtual equipment), 컴퓨터 저장 매체 또는 장치에 구체화(embody)될 수 있다. 소프트웨어는 네트워크로 연결된 컴퓨터 시스템 상에 분산되어서, 분산된 방법으로 저장되거나 실행될 수도 있다. 소프트웨어 및 데이터는 하나 이상의 컴퓨터 판독 가능 기록 매체에 저장될 수 있다.

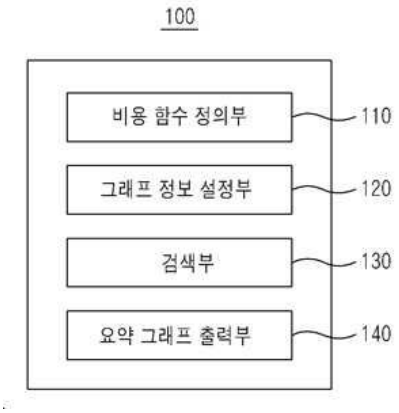
[0129] 실시예에 따른 방법은 다양한 컴퓨터 수단을 통하여 수행될 수 있는 프로그램 명령 형태로 구현되어 컴퓨터 판독 가능 매체에 기록될 수 있다. 상기 컴퓨터 판독 가능 매체는 프로그램 명령, 데이터 파일, 데이터 구조 등을 단독으로 또는 조합하여 포함할 수 있다. 상기 매체에 기록되는 프로그램 명령은 실시예를 위하여 특별히 설계되고 구성된 것들이거나 컴퓨터 소프트웨어 당업자에게 공지되어 사용 가능한 것일 수도 있다. 컴퓨터 판독 가능 기록 매체의 예에는 하드 디스크, 플로피 디스크 및 자기 테이프와 같은 자기 매체(magnetic media), CD-ROM, DVD와 같은 광기록 매체(optical media), 플롭티컬 디스크(floptical disk)와 같은 자기-광 매체(magneto-optical media), 및 롬(ROM), 램(RAM), 플래시 메모리 등과 같은 프로그램 명령을 저장하고 수행하도록 특별히 구성된 하드웨어 장치가 포함된다. 프로그램 명령의 예에는 컴파일러에 의해 만들어지는 것과 같은 기계어 코드뿐만 아니라 인터프리터 등을 사용해서 컴퓨터에 의해서 실행될 수 있는 고급 언어 코드를 포함한다.

[0130] 이상과 같이 실시예들이 비록 한정된 실시예와 도면에 의해 설명되었으나, 해당 기술분야에서 통상의 지식을 가진 자라면 상기의 기재로부터 다양한 수정 및 변형이 가능하다. 예를 들어, 설명된 기술들이 설명된 방법과 다른 순서로 수행되거나, 및/또는 설명된 시스템, 구조, 장치, 회로 등의 구성요소들이 설명된 방법과 다른 형태로 결합 또는 조합되거나, 다른 구성요소 또는 균등물에 의하여 대치되거나 치환되더라도 적절한 결과가 달성될 수 있다.

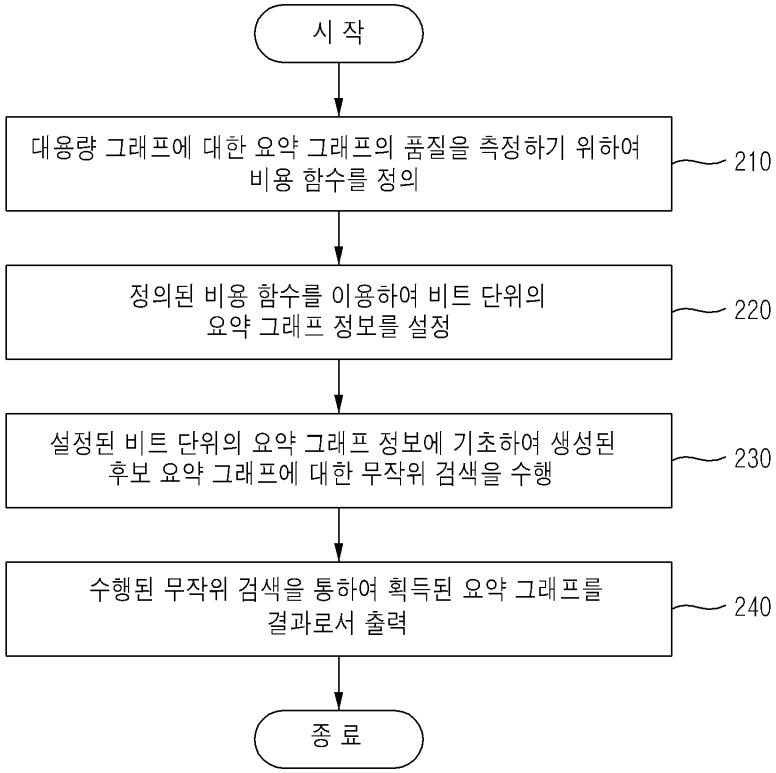
[0131] 그러므로, 다른 구현들, 다른 실시예들 및 특허청구범위와 균등한 것들도 후술하는 특허청구범위의 범위에 속한다.

도면

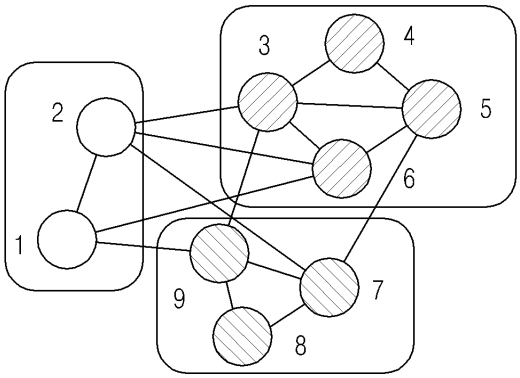
도면1


100
비용 함수 정의부
110
그래프 정보 설정부
120
검색부
130
요약 그래프 출력부
140

도면2


시 작
대용량 그래프에 대한 요약 그래프의 품질을 측정하기 위하여 비용 함수를 정의
210
정의된 비용 함수를 이용하여 비트 단위의 요약 그래프 정보를 설정
220
설정된 비트 단위의 요약 그래프 정보에 기초하여 생성된 후보 요약 그래프에 대한 무작위 검색을 수행
230
수행된 무작위 검색을 통하여 획득된 요약 그래프를 결과로서 출력
240
종 료

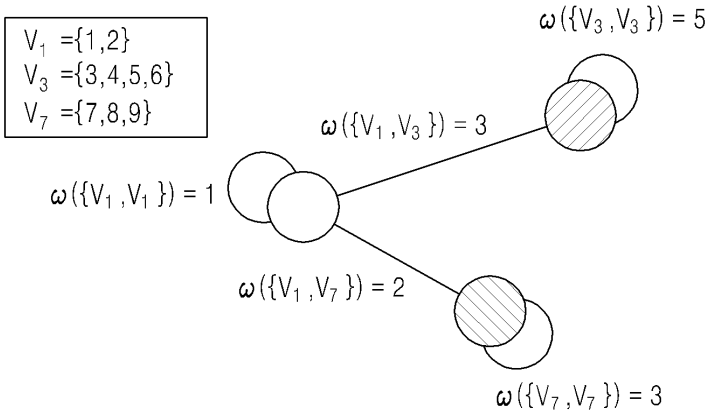
도면3a



도면3b

	1	2	3	4	5	6	7	8	9
1	0	1	0	0	0	1	0	0	1
2	1	0	1	0	0	1	1	0	0
3	0	1	0	1	1	1	0	0	1
4	0	0	1	0	1	0	0	0	0
5	0	0	1	1	0	1	1	0	0
6	1	1	1	0	1	0	0	0	0
7	0	1	0	0	1	0	0	1	1
8	0	0	0	0	0	0	1	0	1
9	1	0	1	0	0	0	1	1	0

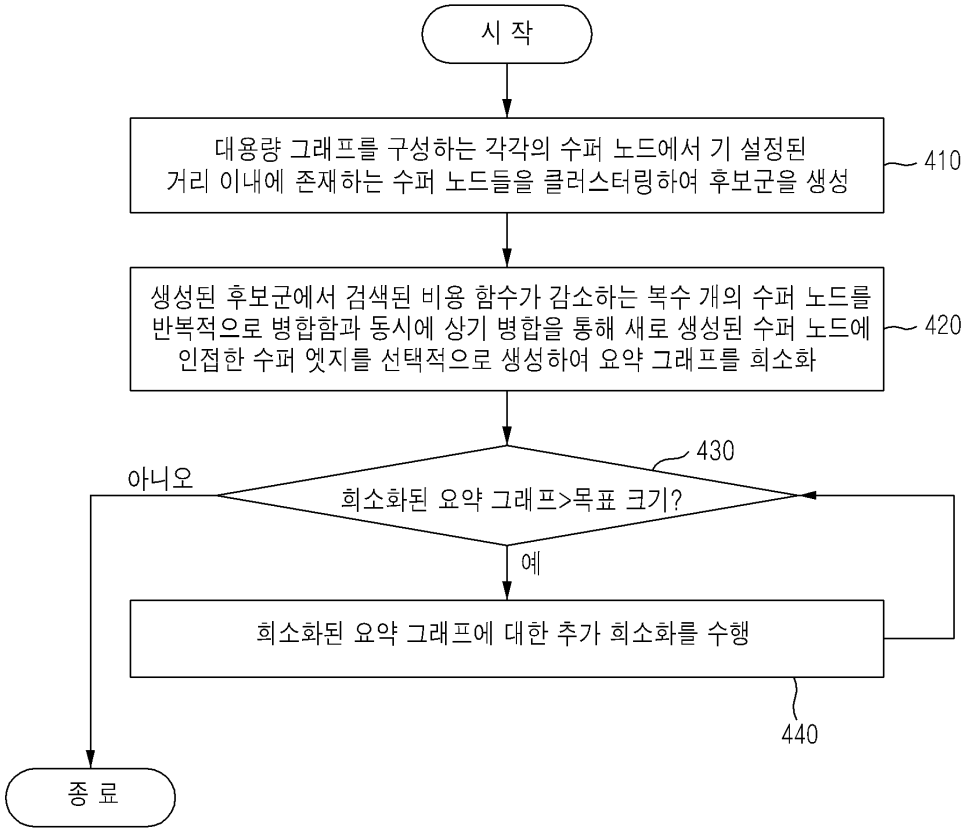
도면3c



도면3d

	1	2	3	4	5	6	7	8	9
1	0	1	3/8	3/8	3/8	3/8	1/3	1/3	1/3
2	1	0	3/8	3/8	3/8	3/8	1/3	1/3	1/3
3	3/8	3/8	0	5/6	5/6	5/6	0	0	0
4	3/8	3/8	5/6	0	5/6	5/6	0	0	0
5	3/8	3/8	5/6	5/6	0	5/6	0	0	0
6	3/8	3/8	5/6	5/6	5/6	0	0	0	0
7	1/3	1/3	0	0	0	0	0	1	1
8	1/3	1/3	0	0	0	0	1	0	1
9	1/3	1/3	0	0	0	0	1	1	0

도면4



도면5

Algorithm 1: Overview of $SS_{UM}M$

Input: (a) input graph: G = (V,E)
(b) the number of iterations: T
(c) desired size of $\overline{G}$: k
Output: summary graph: $G = (S,P,\omega)$

```
S ← {{u} : u∈V} ;                          ▷ initialize Ḡ to G
P ← {{V_u, V_v} ∈ Π_S : {u, v}∈E};
t ← 1 ;                                    ▷ t : iteration
while t ≤ T and k < Size(Ḡ) do
    generate candidate sets S_t ⊆ 2^S ;
    for each candidate set C∈S_t do
        greedily merges supernodes within C⊆S and adds
          new superedges selectively ;
    end
    t ← t + 1;
end
if Size(Ḡ) > k then
    greedily drops superedges from P so that Size(Ḡ) ≤ k ;
end
return Ḡ = (S,P,ω)
```

도면6

Algorithm 2: Merging & Sparsification in a Candidate Set

Input: (a) input graph G = (V, E)
(b) current summary graph $\overline{G} = (S, P, \omega)$
(C) current iteration number t
(d) a candidate supernode set C
Output: updated summary graph $\overline{G} = (S, P, \omega)$

```
num_skips ←0 ;
while num_skips < max (log_2 |C| ,1) do
    find a pair {A, B} that maximizes Eq. (17) among log_2 |C|
      random pairs of supernodes in C ;
    if Relative_Reduction(A, B) > θ(t) then
        merge A, B into A∪B both in S and C ;          ▷ merge
        remove the superedges adjacent to A or B from P ;
        add the superedges adjacent to A∪B to p selectively
          so that Cost_{A∪B}(Ḡ, G) is minimized ;      ▷ sparsify
        num_skips←0 ;
    end
    else
        num_skips← num_skips + 1 ;
    end
end
```