



(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2012년06월04일
(11) 등록번호 10-1153113
(24) 등록일자 2012년05월29일

(51) 국제특허분류(Int. Cl.)
G06F 11/00 (2006.01) G06F 12/00 (2006.01)
(21) 출원번호 10-2005-0062656
(22) 출원일자 2005년07월12일
심사청구일자 2010년07월05일
(65) 공개번호 10-2006-0050069
(43) 공개일자 2006년05월19일
(30) 우선권주장
10/929,514 2004년08월30일 미국(US)
(56) 선행기술조사문헌
US5966702 A
KR1020010047696 A
US20020124214 A1
JP04180127 A

(73) 특허권자
마이크로소프트 코포레이션
미국 워싱턴주 (우편번호 : 98052) 레드몬드 원
마이크로소프트 웨이
(72) 발명자
모트와니, 라지브
미국 98052 워싱턴주 레드몬드 원 마이크로소프
트 웨이마이크로소프트 코포레이션 내
차우두리, 수라지트
미국 98052 워싱턴주 레드몬드 원 마이크로소프
트 웨이마이크로소프트 코포레이션 내
간디, 벤카데시
미국 98052 워싱턴주 레드몬드 원 마이크로소프
트 웨이마이크로소프트 코포레이션 내
(74) 대리인
제일특허법인

전체 청구항 수 : 총 14 항

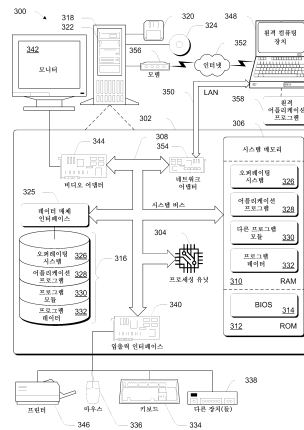
심사관 : 이정은

(54) 발명의 명칭 퍼지 중복 삭제 시스템 및 방법

(57) 요약

여기에 설명된 적어도 하나의 구현은 퍼지 중복(fuzzy duplicate)들을 검출하고 이러한 중복들을 삭제한다. 퍼지 중복은 동일한 실세계 실체(entity) 또는 현상을 표현하는, 데이터베이스 내의 겹보기에 별개인 다수의 튜플(tuple)(즉, 레코드)이다.

대표도 - 도3



특허청구의 범위

청구항 1

프로세서 실행가능 명령어들을 갖는 하나 이상의 프로세서 판독가능 기록 매체로서, 상기 프로세서 실행가능 명령어들은, 프로세서에 의해 실행될 때,

데이터베이스로부터 복수의 튜플들(tuples)을 포함하는 데이터세트를 획득하는 동작,

상기 데이터세트의 상기 복수의 튜플들의 각각에 대해, 상기 데이터세트의 상기 튜플들 상에서 정의된 거리 함수(distance function)를 사용하는 것에 의해 상기 데이터세트의 정의된 개수의 최상위 인접 이웃 튜플들을 계산하는 동작,

각각의 튜플에 대하여, 소정 개수의 최상위 인접 이웃 튜플들의 세트를 이웃(neighborhood)으로 정의하는 동작,

각각의 이웃에 대하여, 이웃 성장 비율(neighborhood growth ratio)을 계산하는 동작 - 상기 이웃 성장 비율은 상기 이웃이 정의될 상기 튜플 주위의 구(sphere) 안의 튜플들의 개수(a number of tuples)이며, 상기 구의 반지름은 상기 이웃이 정의될 상기 튜플과 상기 튜플의 최인접 이웃 사이의 거리보다 큼 -,

컴팩트 세트(compact set) 및 희박한 이웃(sparse neighborhood)인 각각의 파티션에 기초하여, 복수의 튜플들의 복수의 분해 파티션들(disjoint partitions)을 정의하는 동작 - 상기 데이터세트 내의 한 세트의 튜플들은, 상기 한 세트의 모든 튜플들에 대하여 상기 한 세트의 튜플들 중의 해당 튜플과 상기 한 세트의 튜플들 중의 임의의 다른 튜플 사이의 거리가 상기 한 세트의 튜플들 중의 해당 튜플과 상기 한 세트의 튜플들의 외부의 상기 데이터세트 내의 임의의 다른 튜플 사이의 거리보다 작으면, 컴팩트 세트이고, 상기 한 세트의 튜플들 중의 모든 튜플들의 이웃 성장 비율들의 총계값(aggregated value)이 임계값보다 작으면 상기 한 세트의 튜플들은 희박한 이웃 그룹이고, 각각의 파티션의 튜플들은 서로의 퍼지 중복들을 포함하여 파티션의 각각의 퍼지 중복 튜플(fuzzy duplicate tuple)이 동일한 실세계의 실체(same real world entity) 또는 현상을 표현함 -, 및

복수의 퍼지 중복 튜플들의 상기 복수의 파티션들 내의 중복들을 제거하는 동작 - 이로써 상기 파티션의 각각에 하나의 중복되지 않은 튜플만 남아있도록 함 - 을 수행하는, 하나 이상의 프로세서 판독가능 기록 매체.

청구항 2

제1항에 있어서,

상기 튜플들의 세트의 카디널리티(cardinality)가 1과 같으면 상기 튜플들의 세트는 희박한 이웃 그룹인, 하나 이상의 프로세서 판독가능 기록 매체.

청구항 3

제1항에 있어서,

상기 계산하는 동작은 상기 데이터세트의 상기 튜플들의 인덱스를 통해 최상위 인접 이웃들을 룩업(look up)하는 동작을 더 포함하는, 하나 이상의 프로세서 판독가능 기록 매체.

청구항 4

제1항에 있어서,

상기 정의하는 동작은 컴팩트 세트, 희박한 이웃 및 작은 카디널리티(small cardinality)인 각각의 파티션에 기초하여 복수의 튜플들의 복수의 파티션들을 정의하는 동작을 더 포함하고, 상기 파티션의 상기 퍼지 중복들의 그룹의 카디널리티가 사전-정의된 정수보다 작거나 또는 같으면 파티션의 상기 카디널리티는 작은(small) 것인, 하나 이상의 프로세서 판독가능 기록 매체.

청구항 5

제1항에 있어서,

각각의 파티션 내의 상기 튜플들은 서로에 대한 정확한 중복들을 더 포함하는, 하나 이상의 프로세서 판독가

능 기록 매체.

청구항 6

퍼지 중복 제거 컴퓨터 시스템으로서,

데이터베이스로부터 복수의 튜플들을 포함하는 데이터세트를 획득하기 위한 데이터세트 획득 수단;

상기 데이터세트의 상기 튜플들 상에서 정의된 거리 함수를 사용하여, 상기 데이터세트의 상기 복수의 튜플들의 각각에 대한 상기 데이터세트의 정의된 개수의 최상위 인접 이웃 튜플들을 계산하고, 각각의 튜플에 대하여, 소정 개수의 최상위 인접 이웃 튜플들의 세트를 이웃(neighborhood)으로 정의하고, 각각의 이웃에 대하여 이웃 성장 비율을 계산하기 위한 계산 수단 - 상기 이웃 성장 비율은 상기 이웃이 정의될 상기 튜플 주위의 구 안의 튜플들의 개수이며, 상기 구의 반지름은 상기 이웃이 정의될 상기 튜플과 상기 튜플의 최인접 이웃 사이의 거리보다 큼 -;

컴팩트 세트 및 희박한 이웃인 각각의 파티션에 기초하여 복수의 튜플들의 복수의 분해 파티션들을 정의하기 위한 파티셔닝 수단 - 상기 데이터세트 내의 한 세트의 튜플들은, 상기 한 세트의 모든 튜플들에 대하여 상기 한 세트의 튜플들 중의 해당 튜플과 상기 한 세트의 튜플들 중의 임의의 다른 튜플 사이의 거리가 상기 한 세트의 튜플들 중의 해당 튜플과 상기 한 세트의 튜플들의 외부의 상기 데이터세트 내의 임의의 다른 튜플 사이의 거리보다 작으면, 컴팩트 세트이고, 상기 한 세트의 튜플들 중의 모든 튜플들의 이웃 성장 비율들의 총계 값이 임계값보다 작으면 상기 한 세트의 튜플들은 희박한 이웃 그룹이고, 각각의 파티션의 겹보기에 별개인 (seemingly distinct) 튜플들은 서로의 퍼지 중복들을 포함하여 파티션의 각각의 퍼지 중복 튜플이 동일한 실세계의 실체 또는 현상을 표현함 -, 및

상기 파티션들의 각각에 하나의 중복되지 않은 튜플만 남아있도록, 상기 복수의 파티션들 내의 복수의 퍼지 중복 튜플들의 중복들을 제거하기 위한 중복 제거 수단

을 포함하는 퍼지 중복 제거 컴퓨터 시스템.

청구항 7

제6항에 있어서,

상기 튜플들의 세트의 카디널리티가 1과 같으면 상기 튜플들의 세트는 희박한 이웃 그룹인, 퍼지 중복 제거 컴퓨터 시스템.

청구항 8

제6항에 있어서,

각각의 파티션 내의 상기 튜플들은 서로에 대한 정확한 중복들을 더 포함하는, 퍼지 중복 제거 컴퓨터 시스템.

청구항 9

제6항에 있어서,

상기 계산하는 수단은 또한 상기 데이터세트의 상기 튜플들의 인덱스를 통해 최상위 인접 이웃들을 룩업하기 위한 것인, 퍼지 중복 제거 컴퓨터 시스템.

청구항 10

제6항에 있어서,

상기 파티셔닝 수단은 컴팩트 세트, 희박한 이웃 및 작은 카디널리티인 각각의 파티션에 기초하여 복수의 튜플들의 복수의 파티션들을 정의하기 위한 것이고, 상기 파티션의 상기 퍼지 중복들의 그룹의 카디널리티가 사전-정의된 정수보다 작거나 또는 같으면 파티션의 상기 카디널리티는 작은 것인, 퍼지 중복 제거 컴퓨터 시스템.

청구항 11

데이터세트의 퍼지 중복 튜플들을 제거하기 위한 컴퓨터-구현 방법으로서,

데이터베이스로부터 획득한 데이터세트의 복수의 튜플들의 각각에 대해, 상기 데이터세트의 상기 튜플들 상에서 정의된 거리 함수를 사용하는 것에 의해 정의된 개수의 최상위 인접 이웃 튜플들을 계산하는 단계;

각각의 튜플에 대하여, 소정 개수의 최상위 인접 이웃 튜플들의 세트를 이웃(neighborhood)으로 정의하는 단계;

각각의 이웃에 대한 이웃 성장 비율을 계산하는 단계 - 상기 이웃 성장 비율은 상기 이웃이 정의될 상기 튜플 주위의 구에서의 튜플들의 개수이며, 상기 구의 반지름은 상기 이웃이 정의될 상기 튜플과 상기 튜플의 최인접 이웃 사이의 거리보다 큼 -;

컴팩트 세트 및 희박한 이웃인 각각의 파티션에 기초하여 복수의 튜플들의 복수의 분해 파티션들을 정의하는 단계 - 상기 데이터세트 내의 한 세트의 튜플들은, 상기 한 세트의 모든 튜플들에 대하여 상기 한 세트의 튜플들 중의 해당 튜플과 상기 한 세트의 튜플들 중의 임의의 다른 튜플 사이의 거리가 상기 한 세트의 튜플들 중의 해당 튜플과 상기 한 세트의 튜플들의 외부의 상기 데이터 세트 내의 임의의 다른 튜플 사이의 거리보다 작으면, 컴팩트 세트이고, 상기 한 세트의 튜플들 중의 모든 튜플들의 이웃 성장 비율들의 총계값이 임계값보다 작으면 상기 한 세트의 튜플들은 희박한 이웃 그룹이고, 각각의 파티션의 겹보기에 별개인 튜플들은 서로의 퍼지 중복들을 포함하여 파티션의 각각의 퍼지 중복 튜플이 동일한 실세계의 실체 또는 현상을 표현함 - ,

복수의 퍼지 중복 튜플들의 상기 복수의 파티션들 내의 중복들을 제거하는 단계 - 이로써 상기 파티션들의 각각에 하나의 중복되지 않은 튜플만 남아있도록 함 - 를 포함하는 컴퓨터-구현 방법.

청구항 12

제11항에 있어서,

상기 튜플들의 세트의 카디널리티가 1과 같으면 상기 튜플들의 세트는 희박한 이웃 그룹인, 컴퓨터-구현 방법.

청구항 13

제11항에 있어서,

상기 계산하는 단계는 상기 데이터세트의 상기 튜플들의 인덱스를 통해 최상위 인접 이웃들을 탐색하는 단계를 포함하는, 컴퓨터-구현 방법.

청구항 14

제11항에 있어서,

상기 정의하는 단계는 컴팩트 세트, 희박한 이웃 및 작은 카디널리티인 각각의 파티션에 기초하여 복수의 튜플들의 복수의 파티션들을 정의하는 단계를 포함하고, 상기 파티션의 상기 퍼지 중복들의 그룹의 카디널리티가 사전-정의된 정수보다 작거나 또는 같으면 파티션의 상기 카디널리티는 작은 것인, 컴퓨터-구현 방법.

청구항 15

삭제

청구항 16

삭제

청구항 17

삭제

청구항 18

삭제

청구항 19

삭제

청구항 20

삭제

청구항 21

삭제

명세서

발명의 상세한 설명

발명의 목적

발명이 속하는 기술 및 그 분야의 종래기술

- [0010] 본 발명은 일반적으로 데이터베이스 및 데이터 웨어하우스(data warehouse)에 관련된 기술에 관한 것이다.
- [0011] 데이터 웨어하우스 상의 결정 지원 분석이 중요한 비즈니스 결정에 영향을 주므로, 이러한 분석의 정확도는 중요하다. 그러나 데이터 웨어하우스에 수신되는 외부 소스(source)로부터의 데이터는 보통 에러(예를 들어, 잘못된 스펠링, 데이터 소스들에의 모순적인 규약)를 포함한다. 이러한 에러들은 종종 튜플들의 중복 엔트리를 발생시킨다. 이에 따라, 데이터 클리닝(data cleaning), 즉 데이터 내의 에러들을 검출하고 정정하는 태스크에 상당한 시간 및 비용이 소요된다.
- [0012] 데이터베이스 내의 중복된 튜플들을 검출 및 삭제하는 것에 대한 문제는 광범위한 영역의 데이터 클리닝 및 데이터 품질의 중요 문제들 중 하나이다. 동일한 논리적 실세계 실체가 데이터 웨어하우스 내에서 다수의 표현을 가질 수 있는 경우가 종종 있다.
- [0013] 예를 들어, 이름이 Isabel인 한 고객이 슈퍼메가마켓(SuperMegaMarket)에서 제품을 두 번 구매했을 때, 그녀의 이름은 두개의 상이한 레코드, 즉, [Isabel Christie, Seattle, WA, USA, 98025] 및 [Christy Isabel, Seattle, WA, United States, 98025]로 나타내질 수 있다. 이러한 불일치는 데이터 엔트리 에러 및/또는 데이터를 입력하는 판매원의 선호(preference) 때문일 수 있다.
- [0014] Isabel과 같은 몇몇 고객들이 중복된 카탈로그를 수신할 수 있으므로, 이러한 중복된 정보는 다이렉트 메일링(direct mailing) 비용을 상당히 증가시킬 수 있다. 엄격한 예산 제약을 갖는 다이렉트 메일링 캠페인에서, 이러한 에러들은 캠페인의 성공과 실패 사이의 차이를 낳을 수 있다. 게다가, 이러한 에러들은 오류가 많은 분석 모델 생성뿐만 아니라 잘못된 쿼리 결과(예를 들어, 시애틀에는 얼마나 많은 슈퍼메가마켓 고객들이 있는가?)들을 생성할 수 있다.
- [0015] 겹보기에는 별개이지만 실제로는 중복되는 엔트리들을 데이터베이스에서 제거하는 것은 퍼지 중복 삭제 문제이다. 여기서, "퍼지 중복"은 정확하게 일치하지는 않지만 동일한 실세계 실체 또는 현상을 표현하는, 겹보기에 개별적인 튜플들(즉, 레코드들)이다.
- [0016] 이러한 문제는 두개의 튜플의 모든 속성들이 정확하게 일치할 때만 그들이 중복된다고 간주되는 표준의 정확한 중복 삭제 문제와는 다르다. 문맥이 명확하게 다르게 나타내지지 않는 한, 이제부터는 중복 검출 및 삭제에 대한 언급은 퍼지 중복 삭제 문제에 초점을 맞춘다고 가정하자.
- [0017] 퍼지 중복 삭제에 대한 종래 해결법들은 감독(supervised) 및 무감독(unsupervised) 접근법으로 분류될 수 있다. 감독 접근법은 알려진 중복들로 구성된 트레이닝 데이터(training data)로부터 중복들의 쌍들을 특성화하는 규칙들을 학습한다. 또한, 이러한 접근법들은 트레이닝 데이터가 실제로 관찰되는 에러의 분포 및 다양성을 나타낸다고 가정한다. 상호작용적인 수동 안내(interactive manual guidance)를 필요로 하는 결점을 갖는 활성 학습 접근법에 의하여, 논점이었던 이러한 포괄적인 트레이닝 데이터를 한정된 범위까지 획득하는 것은 불가능하지는 않더라도 어려운 일이다. 많은 실제의 데이터 통합 시나리오들에서, 양호한 트레이닝 데이터 또는 상호작용적 사용자 안내를 획득하는 것은 불가능하다.
- [0018] 무감독 중복 삭제와 클러스터링(clustering) 둘 다 데이터세트를 분해(disjoint) 그룹으로 파티션(partition)하려고 한다는 점에서, 무감독 중복 삭제의 문제는 클러스터링(clustering)의 문제들과 유사하다. 그러나,

표준의 클러스터링 공식화와 중복 삭제 문제 사이에는 몇몇의 뚜렷한 차이점이 있다. 이러한 차이점은 후에 설명될 것이다.

[0019] 현재의 무감독 접근법들은 이러한 차이점들은 무시하고, 그 대신, 다수의 속성을 갖는 튜플들과 임계치 기반 제약 사이의 표준의 텍스트적인 유사성 함수[예를 들어, 편집 거리 및 코사인 메트릭(cosine metric)과 같은 잘 알려진 단일 연계 클러스터링 알고리즘(single-linkage clustering algorithm)]에 의존하여 중복 쌍들을 검출하려는 경향이 있다. 그러나, 이러한 임계치 기반 접근법들은 다수의 거짓 긍정(실제로 중복이 아니지만 그렇다고 예상된 튜플들) 또는 다수의 거짓 부정(실제로 중복이지만 그렇다고 인식되지 않는 튜플)을 유발할 수 있다.

발명이 이루고자 하는 기술적 과제

[0020] 여기에 설명된 적어도 하나의 구현은 퍼지 중복을 검출하고, 이러한 중복들을 삭제한다. 퍼지 중복들은 동일한 실세계 실체 또는 현상들을 표현하는, 데이터 베이스 내의 겹보기에 별개인 다수의 튜플이다.

발명의 구성 및 작용

[0021] 유사한 구성요소들 및 특징들을 참조하기 위해 도면에서 동일한 숫자들이 사용된다.

[0022] 다음은 데이터베이스 내의 퍼지 중복 튜플들의 검출 및 삭제를 용이하게 하는 기술에 대해 설명한다. 이러한 기술들은 프로그램 모듈, 범용 및 특수 목적 컴퓨팅 시스템, 전용 전자 기기 및 하나 이상의 컴퓨터 네트워크들의 일부를 포함하는 (이에 한정되지 않음) 다양한 것에서 구현될 수 있다.

[0023] 이러한 기술에 대한 예시적인 구현은 "예시적인 퍼지 중복 검출기"로 언급될 수 있으며, 다음에 설명된다.

[0024] 예시적인 퍼지 중복 검출기는 퍼지 중복 삭제 문제를 다룬다. 여기서, "퍼지 중복"은 동일한 실세계 실체 또는 현상을 표현하지만 정확하게 일치하지 않는, 겹보기에 별개인 튜플들(즉, 레코드들)이다. 퍼지 중복들을 검출 및 삭제하는 것은 퍼지 중복 삭제 문제이다.

[0025] 기준 특성화 중복

[0026] 퍼지 중복들을 검출할 때, 예시적인 퍼지 중복 검출기는 종래 접근법이 사용하지 않는 적어도 두개의 새로운 제약들을 사용한다. 특히, 이러한 두개의 새로운 기준은 CS(compact set) 및 SN(sparse neighborhood)로 불린다. 이러한 기준은 데이터의 국부적 구조 속성(local structural property)들을 명시적으로 캡처하여, 중복 튜플들의 그룹을 특성화한다.

[0027] CS 및 SN 기준은 다음과 같은 속성들을 캡처한다.

[0028] * 그룹 내의 중복들은 다른 것들보다 서로 근접함; 및

[0029] * 중복 튜플들의 "국부적 이웃"은 비었거나 희박함(sparse).

[0030] 이러한 기준을 만족하지 못하는 튜플들은 근접해있어도 함께 그룹화될 수 없는 반면, 이러한 기준을 만족시키는 튜플들은 그들이 서로 멀리 떨어져 있더라도 중복으로서 함께 그룹화될 수 있다. 이러한 국부화된 구조 속성들은 중복 삭제 문제를 표준 클러스터링 공식화로부터 구별짓는다.

표 1

[0031]

ID	아티스트이름	트랙이름
1*	The Doors	LA Woman
2*	Doors	LA Woman
3*	The Beatles	A Little Help from My Friends
4*	Beatles, The	With A Little Help From My Friend
5*	Shania Twain	Im Holding on to Love
6*	Twain, Shania	I'm Holding On To Love
7	4th Elemynt	Ears/Eyes
8	4th Elemynt	Ears/Eyes - Part
9	4th Elemynt	Ears/Eyes - Part III
10	4 th Elemynt	Ears/Eyes - Part IV
11	Aalivah	Are You Ready

12	AC DC	Are You Ready
13	Bob Dylan	Are You Ready
14	Creed	Are You Ready

- [0032] <미디어 데이터베이스로부터의 예들. 별표로 표시된 투플들은 중복 투플임>
- [0033] 표 1은 전형적인 음악 데이터베이스의 예를 제공한다. 처음 6개의 투플들(별표 "*"로 표시됨)은 중복 투플들이고, 나머지 투플들(7-14)은 고유하다.
- [0034] 컴팩트 세트 기준(compact set criterion)
- [0035] 컴팩트 세트(CS) 기준은 중복들의 세트가 서로 최인접 이웃들의 컴팩트 세트라는 것이다. 이러한 기준은 중복 투플들이 다른 별개의 투플들보다 서로 근접해있다는 것을 전제로 하고 있다. 즉, 중복 투플들은 주로 서로 최인접 이웃들이다. 예를 들어, 표 1에서 투플 1은 투플 2의 최인접 이웃이고, 그 역도 마찬가지이다. 대조적으로, 투플 8은 투플 7의 최인접 이웃이고, 투플 9도 투플 8에게 그럴 수 있다.
- [0036] 대조적으로, 단일 연계 클러스터링에 기초한 종래 임계치 기반 접근법은 이행성(transitivity)을 가정하고(즉, 'a'가 'b'의 중복이고, 'b'가 'c'의 중복이면, 'a'는 'c'의 중복임), 임계치 그래프 내의 연결된 컴포넌트들을 식별한다. 이에 따라, 그들은 다수의 거짓 긍정을 생성할 가능성이 있다.
- [0037] 희박한 이웃 기준
- [0038] 희박한 이웃(SN) 기준은 중복들의 그룹의 국부적인 이웃이 희박하다는 것에 전제를 두고 있다. 예를 들어, 이러한 기준은 표 1 내의 고유한 투플(7-14)들 주위에서 만족되지 않으며, 중복들의 세트보다 대규모의(이 예에서는 4) 그룹들 내에서 발생한다.
- [0039] 한 관점에서 볼 때, 투플들의 그룹의 국부적인 이웃은 투플들의 국부적인 분포에 의존하는 크기의 주위 영역의 관점에서 정의된 직접적인 근방이다. 예를 들어, 그것은 반경 $2nn(v)$ 의 구일 수 있으며, 여기서 $nn(v)$ 는 투플 v 의 최인접 이웃 거리이다.
- [0040] 투플 주위의 성장 속도- 외부 구에 있는 투플들의 개수 -가 낮으면, 국부적인 이웃은 "희박"하다고 불린다. 이러한 표현은 투플들의 그룹으로 확장되며, 투플의 개별적인 성장 속도의 총계가 작으면(예를 들어, 임계치 c 보다 작을 경우), 그들의 결합(joint) 국부적인 이웃은 "희박하다"고 불린다. 예를 들어, 총계 함수의 최대는 그룹 내의 모든 투플들의 이웃 값이 임계치보다 작을 것을 요구하며, 반면 함수 평균은 모든 성장 속도들의 평균이 작을 것만을 요구한다. 최대 함수는 평균 함수보다 제약을 많이 받는다.
- [0041] 기준의 공식화
- [0042] 다음의 정의에서, R 이 관계(즉, 데이터세트)이고, $d: R \times R \rightarrow [0, 1]$ 은 R 내의 투플들에 대한 대칭적 거리 함수라고 하자. 명확히 설명하기 위해, 이제부터는 (i) 2개의 투플 사이의 거리는 투플들이 정확히 일치할 때만 제로(zero)이고; (ii) R 내의 어느 두개의 투플도 서로 완전히 동일하지 않음을 가정한다. 이러한 가정의 타당성은 투플들이 정확히 동일할 때는 0을 리턴하고, 그렇지 않으면 소정의 작은 $\varepsilon > 0$ 에 대하여 $d(v_1, v_2) + \varepsilon$ 를 리턴하도록 d 를 수정함으로써 보증된다.
- [0043] CS 기준: R 로부터의 투플들의 세트 S 내의 모든 투플 v 에 대해서, S 내의 v 와 임의의 다른 투플 v' 사이의 거리 $d(v, v')$ 가 $R-S$ 내의 임의의 다른 v'' 와 v 사이의 거리 $d(v, v'')$ 보다 짧으면, S 는 컴팩트 세트이다.
- [0044] SN 기준: 투플 v 에 대해서, 2개의 동심구를 고려해보면, 보다 작은 구는 반경 $nn(v)$ (v 와 그것의 최인접 이웃 사이의 거리)를 가지며, 보다 큰 구는 $g(nn(v))(>nn(v))$ 의 반경을 갖는다. 여기에서는, $g(x)=2x$ 가 사용된다. 이웃 성장 비율 $ng(v)$ 는 v 주위의 보다 큰 구 내의 포인트들의 개수이다.
- [0045] $AGG: 2^R \rightarrow R$ 이 총계 함수이고, $c(>0)$ 가 상수라 하자. (i) $|S|=1$ 또는 (ii) S 내의 모든 투플들의 이웃 성장 비율의 총값이 c 보다 작으면(즉, $AGG(\{ng(v): v \text{ in } S\}) < c$), 투플들의 세트 S 는 $SN(AGG, c)$ 그룹이라고 한다.
- [0046] 작은 그룹(Small Group, SG) 기준: 고려될 수 있는 중복들의 그룹들의 또다른 특성은 그들이 주로 매우 작다는 것이다. 소정의 미리정의된 상수 $K>1$ 에 대해서, $|G| \leq K$ 이면, 중복들의 그룹 G 는 작다. 그룹의 카디널리티(cardinality)(즉, 원소 개수)가 작으므로, 이것은 "작은 카디널리티"로 불릴 수도 있다.
- [0047] 예시적인 퍼지 중복 검출기

- [0048] 일반적으로, 예시적인 퍼지 중복 검출기는 입력 관계 R(예를 들어, 데이터베이스의 데이터세트)을 최소 개수의 "유효" 그룹들로 파티셔닝하는데, 여기서 그룹이 작고 CS 및 SN 기준을 만족하면, 이 그룹은 유효하다.
- [0049] 예시적인 퍼지 중복 검출기의 문맥에서, 중복 삭제(DE) 문제는 다음과 같다: 관계 R, 거리 함수 d, 양의 정수 $K(>1)$, 총계 함수 AGG 및 양의 실수 c가 주어지면, 모든 $1 \leq i \leq m$ 에 대해서, 예시적인 퍼지 중복 검출기는 R을 최소 개수의 그룹들 $\{G_1, \dots, G_m\}$ 로 파티셔닝한다.
- [0050] * $|G_i| \leq K$
- [0051] * G_i 는 컴팩트 세트, 및
- [0052] * G_i 는 SN(AGG, c) 그룹
- [0053] 도 1은 퍼지 중복 검출 및 삭제(FDDE) 시스템(100)을 예시한다. 예시적인 퍼지 중복 검출기는 여기에 설명되는 FDDE 시스템(100)으로 구현될 수 있다. FDDE 시스템(100)의 컴포넌트들은 소프트웨어, 하드웨어 또는 양자의 혼합으로 구성될 수 있다. 일반적으로, FDDE 시스템(100)은, 동등한 최인접 이웃 세트들을 갖는 튜플 쌍들을 결정한 후, 쌍 동등성을 튜플들의 그룹으로 확장한다.
- [0054] FDDE 시스템(100)은 서버 또는 클라이언트의 일부일 수 있다. 이 FDDE 시스템(100)은 범용 컴퓨터, 특수 목적 컴퓨터, 휴대용 컴퓨터, PDA, 전기 제품, 전용 전자기기, 또는 데이터베이스를 관리할 수 있는 다른 이러한 컴퓨팅 및 전자기기들 상에서 구현될 수 있다.
- [0055] FDDE 시스템(100)의 컴포넌트가 도 1의 점선 내에 도시된다. 도시된 시스템(100)은 데이터세트 획득 서브시스템(110), 최인접 이웃 계산 서브시스템(120), 파티셔닝 서브시스템(130) 및 중복 삭제 서브시스템(140)을 포함한다.
- [0056] 시스템(100)은 데이터 웨어하우스 서버(160)에 연결된다(예를 들어, 방향성 접속을 통해 또는 인터넷과 같은 네트워크를 통해). 이러한 서버(160) 상에는 데이터베이스(170)가 있다. 시스템(100) 및 데이터베이스(170)는 동일한 컴퓨팅 장치(서버(160)와 같은) 내에 또는 다수의 장치들에 걸쳐 상주할 수 있다.
- [0057] 이 시스템(100)에서, 획득기(110)는 데이터베이스(170)로부터 데이터세트를 획득한다. 이 데이터세트는 시스템(100)의 동작 대상이다. 이 데이터세트는 또한 "관계" 및/또는 단순히 "R"이라고 불릴 수 있다. 이러한 획득의 예는 튜플들의 세트를 가진 데이터 패킷을 수신하는 것, 튜플들을 포함하는 저장된 파일에 액세스하는 것 및/또는 쿼리를 제출하고 그 쿼리에 대한 결과를 수신하는 것을 포함한다.
- [0058] 최인접 이웃 계산 서브시스템(120)은 획득된 관계 내의 모든 튜플의 K개의 최인접 이웃들을 결정한다. 서브시스템(120)은 다음의 쿼리- R 내의 임의의 주어진 튜플 v에 대해서, K개의 최인접 이웃을 인출 -에 대해 효과적으로 대답하도록 하는 기존 색인을 사용하거나 색인을 생성한다.
- [0059] 관계 R 내의 각각의 튜플에 대해, 최인접 이웃 계산 서브시스템(120)은 해당 튜플의 K개의 최인접 이웃 및 이웃 성장 비율을 결정한다. 이 서브시스템의 출력은 관계 $NN_Reln[ID, ID1, \dots, IDK, NG]$ 이며, 여기서 $ID1, \dots, IDK$ 는 식별자 ID 및 이웃 성장 비율 NG를 갖는 튜플 v의 K개의 최인접 이웃의 식별자이다. K개의 최인접 이웃을 인출하고 이웃 성장 비율을 계산하는 데 사용될 수 있는 색인이 주어지면, 이 서브시스템은 입력 관계 R을 스캔하고, R 내의 각각의 튜플에 대해서, 색인을 룩업(look up)하고, 출력에 튜플 $[v, v1, \dots, vK, ng(v)]$ 을 기록한다.
- [0060] 파티셔닝 서브시스템(130)은 관계를 최소 개수의 유효 중복 그룹으로 파티셔닝한다. 적어도 하나의 구현에서, 이 서브시스템은 표준 데이터베이스 (예를 들어, SQL) 쿼리들을 사용한다. 이 방식으로, 파티셔닝 서브시스템(130)은 데이터베이스 시스템의 효율적인 쿼리 프로세싱 능력을 사용하여 대용량의 데이터가 클라이언트와 서버 사이에 이동하는 것을 방지할 수 있다.
- [0061] 파티셔닝 서브시스템(130)은 제1 페이즈(phase)의 출력을 사용하여, 입력 관계를 최소 개수의 컴팩트 SN 세트 로 파티셔닝한다. 그에 대한 파티션은 DE 문제의 해결법이다.
- [0062] 파티셔닝 서브시스템에 의한 퍼지 중복의 식별에 기초하여, 중복 삭제 서브시스템(140)은 불필요한 중복 튜플들을 제거하여 데이터베이스를 클리닝한다. 이 서브시스템은 데이터베이스(170)를 수정하여 이러한 식별된 중복들을 제거한다.

[0063] **색인 록업 순서**

[0064] 여기서, 최인접 이웃 계산 서브시스템(120)의 동작에 관한 추가적인 세부사항들이 설명된다. 특히, 여기서는 색인이 액세스되는 순서에 초점이 맞춰져있다.

[0065] K개의 최인접 이웃을 인출하는 데 전형적으로 사용되는 색인 구조들은 디스크 기반이다. 즉, 그들은 고속으로 액세스가능한 주메모리(예를 들어, RAM) 내에 저장된 데이터보다 저속인 2차 저장 시스템(하드 디스크와 같은) 상에 저장된 데이터에 액세스한다.

[0066] 일반적으로 사용되는 디스크 기반의 최인접 이웃 색인들로 록업되는 연속적인 튜플들이 서로 인접해있으면, 록업 절차는 색인의 동일한 부분에 액세스할 가능성이 있다. 따라서, 제2 록업은 제1 록업으로부터 이익을 얻는다. 이는 버퍼 히트 비율(buffer hit ratio) 및 전체 실행 시간을 상당히 개선시킨다. 예시적인 퍼지 중복 검출기는 효율적으로 구현될 수 있는 록업 순서를 사용한다.

[0067] 표 1 내의 예시적인 튜플들을 고려해보자. 시스템이 R 내의 튜플들의 최인접 이웃들을 록업하는 순서가 1, 12, 5 등이라고 가정하자. 튜플 1("The Doors, LA Woman")의 최인접 이웃들을 인출하기 위해, 색인 절차는 색인의 한 부분에 액세스하고, 프로세스 동안, 그것을 데이터베이스 버퍼 내에 캐시한다. 튜플 12는 튜플 1과 매우 멀리 떨어져 있기 때문에, 튜플 12("Aliyah, Are you ready")의 최인접 이웃들에 대한 유사한 록업은 색인의 완전히 다른 일부에 액세스할 것이다. 대안적으로, 튜플 1을 처리한 후 튜플 2("Doors, LA Woman")의 최인접 이웃들을 록업하면, 시스템은 색인의 거의 동일한 부분을 사용할 것이다. 따라서, 시스템은 그 부분이 데이터베이스 버퍼에 이미 존재한다는 점을 유리하게 활용할 수 있다.

[0068] 예시적인 퍼지 중복 검출기는 양호한 록업 순서의 두가지 속성을 만족시키는 BF(breadth first) 순서를 수용한다: 첫번째로, 순서에 있어서 임의의 튜플 바로 앞에 있는 튜플들은 반드시 그 임의의 튜플에 인접해야 한다. 두번째로, 입력 튜플들을 순서화하기 위한 절차는 반드시 효율적이어야 한다.

[0069] 순서는, 임의의 입력 튜플이 선택되어 트리의 루트가 되도록 구성된 트리 T를 트래버스(traversal)하는 BF에 대응한다. 트리 내의 임의의 노드의 자식은 이전에 트리에 삽입된 적이 없는, 그 노드의 K개의 최인접 이웃들이다. 예시적인 퍼지 중복 검출기는 실제로 이러한 트리를 구축할 필요는 없지만, 단지 입력 튜플들을 올바른 순서로 인출하는 것은 필요함을 명심해야 한다.

[0070] BF 순서의 각 튜플(루트는 제외)은 항상 자신의 형제들 또는 자신의 부모들의 형제들의 자식들에 후속한다. 이러한 튜플들은 임의적인 튜플 쌍들보다 서로에 훨씬 더 인접해 있다. 그러므로, 록업 순서의 모든 튜플들은 그들에 매우 인접한 튜플들에 후속한다. 결론적으로, 록업 알고리즘은 국부화된 색인 액세스를 생성한다.

[0071] 입력 튜플의 각각의 록업에 대해서, 그것의 K개의 최인접 이웃 튜플들이 인출된다. 그러므로, 한 튜플에 마주쳤을 경우, BF 순서에서 시스템은 그 튜플의 부모 튜플이 록업될 때, 그 튜플을 이미 인출했을 것이다. 그러므로, 데이터베이스 버퍼는 이미 그 튜플을 캐시했을 것이다. 시스템은 이러한 튜플들을 명시적으로 캐시하여, 메모리 허가하거나, 최근 액세스들을 버퍼링하기 위해 데이터베이스 시스템에 의존할 수 있다.

[0072] 여기서, 최인접 이웃 색인들에의 참조는 정확하거나 가능성 있는 최인접 이웃 색인들의 사용을 고려한다. 당업자들은 표준 거리 함수(예를 들어, 코사인 매트릭, 편집 거리 및 퍼지 일치 유사성)에 기초한 근사치 및 확률을 사용하는 많은 종래의 색인들에 익숙하다.

[0073] **방법론적 구현**

[0074] 도 2는 FDDE 시스템(100)의 방법론적 구현을 도시한다. 이 방법론적 구현은 소프트웨어, 하드웨어 또는 이들의 조합으로 행해질 수 있다. 용이한 이해를 위해, 방법 단계들을 개별적인 단계들로 나타냈지만, 이러한 개별적으로 나타내진 단계들이 수행시에 반드시 그 순서대로 구성되지 않아도 된다.

[0075] 설명을 위해, 이러한 방법론적 구현에 대한 설명은 특정한 예에 대한 적용에 대해 설명한다. 이 예에서, 그룹 {10, 50, 100, 150}은 콤팩트 SN 세트를 형성한다. 각각의 튜플의 이웃 성장(NG) 값 이외에, 쌍 {10, 50}, {10, 100}, {10, 150}의 4개의 최인접 이웃 세트들이 모두 동일하다는 것을 알면 충분하다. 그룹{10, 50, 100, 150}이 콤팩트 세트라는 것은, 쌍 동등성 및 인 세트 동등성의 이행성으로부터 추론될 수 있다.

[0076] 도 2의 블럭(212)에서, FDDE 시스템(100)은 데이터베이스(170)로부터 데이터세트(즉, 관계)를 획득한다.

[0077] 블럭(214)에서, FDDE 시스템(100)은 획득된 관계 내의 모든 튜플의 K개의 최인접 이웃들을 계산한다. 이를 수행하기 위해, FDDE 시스템(100)은 기존 색인을 사용하거나, 자기 자신만의 색인을 생성할 수 있다.

[0078] 블럭(214)의 부분으로서, FDDE 시스템(100)은 튜플 쌍들 사이의 다양한 크기의 이웃 세트들의 동등성을 계산한다. 즉, 튜플 쌍(10, 100)에 대해서, 시스템은 2-최인접 이웃 세트들, 3-최인접 이웃 세트들, ... K-최인접 이웃 세트들이 동등한지를 결정한다. 튜플 v1과 v2 사이의 이러한 비교는 그들의 이웃 성장 비율 $ng(v1)$ 과 $ng(v2)$ 과 함께 부울(Boolean) 결과 CS2, ..., CSK를 생성한다. 값 $CS_i (2 \leq i \leq K)$ 는 v1과 v2의 i-이웃 세트가 동등한지를 나타낸다. 이 결과는 임시적인 관계 CSPairs 내에 저장될 수 있다.

[0079] 여전히 블럭(214)의 부분으로서, FDDE 시스템(100)은 제1 페이지의 결과(NN_ReIn)에 대해 데이터베이스 쿼리(예를 들어, SQL(select into) 쿼리)를 발행하여, CSPairs 관계를 계산할 수 있다. 이 쿼리는 튜플 NN_ReIn1.ID가 NN_ReIn2.ID 보다 작고, 그것이 NN_ReIn2.ID의 K-최인접 이웃 세트 내에 있을 때, NN_ReIn의 셀프 조인(self-join)(예를 들어, FROM NN_ReIn1, NN_ReIn2)에 관련되며, 그 역도 마찬가지이다.

[0080] 선택된 열 리스트는 (i) NN_ReIn1.ID, NN_ReIn2.ID, NN_ReIn1.NG, NN_ReIn2.NG 및 (ii) 2와 K 사이의 각각의 j에 대해서, NN_ReIn1.ID의 j개의 최인접 이웃들의 세트가 NN_ReIn2.ID의 j개의 최인접 이웃들의 세트와 동일하면, 1을 리턴하는 케이스 스테이트먼트(case statement)의 두개의 부분을 갖는다. 이 쿼리는 표준 SQL을 사용하여 표현될 수 있으며, 데이터베이스 서버에서 모든 계산을 수행할 수 있음을 명심하자.

[0081] 블럭(216)에서, FDDE 시스템(100)은 관계를 최소 개수의 유효 중복 그룹들로 파티셔닝한다. 그에 의해 생성되는 유효 중복 그룹들은 다음 기준 각각을 만족한다:

[0082] * $|G_i| \leq K$

[0083] * G_i 는 컴팩트 세트, 및

[0084] * G_i 는 SN(AGG, c) 그룹

[0085] 블럭(216)의 부분으로서, FDDE 시스템(100)은 튜플 쌍들의 이웃 세트들 사이의 동등성을 튜플들의 세트들로 확장하고, 이웃들의 세트가 컴팩트하고 SN 기준을 만족하는지를 결정한다. 표 2에서, 튜플 쌍들 (10, 50), (10, 100) 및 (10, 150)의 4-이웃 세트가 동일하므로, 크기 4의 컴팩트 세트를 형성한다. 여기서 (i) 이 그룹의 총계 SN 값이 임계치 c 보다 작고, (ii) 이것이 보다 큰 컴팩트 SN 그룹으로 확장될 수 없다면, 세트 {10, 50, 100, 150}는 중복들의 그룹으로서 출력될 수 있다. 세트 동등성이 이행을 가지므로, 쌍 (50, 100), (50, 150) 등의 4-이웃 세트가 동일한지 아닌지가 명시적으로 확인되지 않는다.

표 2

ID	: NN1, NN2, NN3, ..., NG(TID)
10	: 100, 50, 150, ..., 2.0
50	: 10, 150, 100, ..., 2.0
100	: 50, 10, 150, ..., 3.0
150	: 10, 100, 100, ..., 2.0
...	
ID1, ID2 : CS2, CS3, CS4, ..., NG(ID1), NG(ID2)	
10, 50	: 0, 0, 1, ..., 2.0, 2.0
10, 100	: 0, 1, 1, ..., 2.0, 3.0
10, 150	: 0, 0, 1, ..., 2.0, 2.0
...	
ID	: Compact SN Group
10	: 50, 100, 150
...	

[0086]

[0087] <파티셔닝을 설명하는 예>

[0088] FDDE 시스템(100)은 "select * from CSPairs order by ID"와 같은 CS 그룹 쿼리를 발행함으로써 CSPairs 관계 [블럭(214)의 출력]를 처리하여, v.ID < v'.ID인 경우에 튜플 v와 그것의 이웃 v' 사이의 모든 이웃 세트 비교 결과를 그룹화한다. CS 그룹 쿼리의 결과에서, 각각의 컴팩트 SN 세트 G는 G 내의 최소 ID를 갖는 튜플 아래로 함께 그룹화될 것임을 주의하자.

[0089] FDDE 시스템(100)은 CS 그룹 쿼리의 결과 내의 튜플들의 그룹 Q[ID=v](동일한 ID로 식별됨) 각각을 처리한다. 튜플의 그룹 Q[ID=v]에 대해서, v가 컴팩트 SN 세트에 속한 것으로 미리 식별되지 않았으면, 이 시스템은 v가 속할 수 있는 가장 큰 비자명(non-trivial) (크기가 1 보다 큰) 컴팩트 SN 세트 Gv를 결정한다. 이 세트는 그룹 Q[ID=v]로부터만 식별될 수 있다. 시스템은 세트 Gv를 출력하고, Gv 내의 각각의 튜플에 마킹(mark)하여 그것의 정렬을 컴팩트 세트에 나타낸다. 예를 들어, 튜플 10에 대한 세트는 {10, 50, 100, 150}이다. 세트 {10, 50, 100, 150}를 출력하고, 튜플 식별자 10, 50, 100, 150를 유효 컴팩트 SN 세트에 속하는 것으로

마킹하여, 그들이 또다시 처리되지 않도록 한다.

[0090] 블록(218)에서, FDDE 시스템(100)은 불필요한 중복 튜플들을 제거함으로써 데이터베이스(170)를 클리닝하는데, 여기서 이러한 중복 튜플들의 식별은 블록(216)의 결과에 기초한다.

[0091] 다른 구현 세부사항

[0092] 예시적인 퍼지 중복 검출기가 제공하는 퍼지 중복 삭제 문제에 대한 해결책은 현존의 종래 접근법들보다 정확하고 효과적이다. 그것은 스케일 불변이므로, 거리 함수의 스케일이 튜플들의 국부적인 구조 속성들에 영향을 준다. 중복들의 그룹 내의 튜플들 사이의 거리를 줄이는 것 및 그룹들에 있는 튜플들 사이의 거리를 확장시키는 것은 제한된 방식으로만 파티션을 수정할 수 있다는 점에서, 분리/병합은 일관적이다. 그것은 제한된 풍부함을 가지므로, 중복 삭제 함수의 범위는 구현에서 유용할 수 있는 그룹화 모드를 허용한다.

[0093] 적어도 일 구현에서, 사용자는 희박한 이웃 임계치 c 를 지정해야 한다. 그러나, 그 계산은 자동적으로 행해질 수 있거나 임계치 c 를 계산할 때 사용자에게 도움을 줄 수 있다.

[0094] 예시적인 컴퓨팅 시스템 및 환경

[0095] 도 3은 여기에 설명된 바와 같은 예시적인 퍼지 중복 검출기가 (전체적으로 또는 부분적으로) 구현될 수 있는 적절한 컴퓨팅 환경(300)의 예를 예시한다. 컴퓨팅 환경(300)은 여기 설명된 컴퓨터 및 네트워크 아키텍처들 내에서 사용될 수 있다.

[0096] 예시적인 컴퓨팅 환경(300)은 컴퓨팅 환경의 일 예일 뿐이며, 컴퓨터 및 네트워크 아키텍처의 사용 또는 기능의 영역에 어떤 제한을 두기 위한 것은 아니다. 컴퓨팅 환경(300)은 예시적인 컴퓨팅 환경(300)에 예시된 컴포넌트들 중 임의의 하나 또는 그것들의 조합에 관련된 어떤 종속성 또는 요구사항을 갖는 것으로 해석되어서는 안된다.

[0097] 예시적인 퍼지 중복 검출기는 수많은 다른 범용 또는 특수 목적 컴퓨팅 시스템 환경 또는 구성으로 구현될 수 있다. 사용하기에 적절할 수 있는 잘 알려진 컴퓨팅 시스템, 환경 및/또는 구성의 예는 개인용 컴퓨터, 서버 컴퓨터, 썬 클라이언트(thin client), 씩 클라이언트(thick client), 핸드헬드 또는 랩탑 장치, 멀티프로세서 시스템, 마이크로프로세서 기반 시스템, 셋톱 박스, PDA, 가전 제품, 특수 목적 가전 제품(예를 들어, DVD 플레이어), 프로그램가능 가전 제품, 네트워크 PC, 미니컴퓨터, 메인프레임 컴퓨터, 상기 시스템 또는 장치들 중 임의의 것을 포함하는 분산 컴퓨팅 환경 등을 포함하지만, 이에 한정된 것은 아니다.

[0098] 예시적인 퍼지 중복 검출기는 컴퓨터에 의해 실행되는, 프로그램 모듈과 같은 프로세서 실행가능 명령어의 일반적인 문맥으로 설명될 수 있다. 일반적으로, 프로그램 모듈은 특정 태스크를 수행하거나 특정 추상 데이터 유형을 구현하는 루틴, 프로그램, 오브젝트, 컴포넌트, 데이터 구조 등을 포함한다. 예시적인 퍼지 중복 검출기는 또한 태스크가 통신 네트워크를 통해 링크된 원격 프로세싱 장치에 의해 수행되는 분산 컴퓨팅 환경에서 구현될 수 있다. 분산 컴퓨팅 환경에서, 프로그램 모듈은 메모리 저장 장치를 포함하는 로컬 및 원격 컴퓨터 저장 매체 내에 위치될 수 있다.

[0099] 컴퓨팅 환경(300)은 컴퓨터(302)의 형태인 범용 컴퓨팅 장치를 포함한다. 컴퓨터(302)의 컴포넌트들은 하나 이상의 프로세서 또는 프로세싱 유닛(304), 시스템 메모리(306) 및 프로세서(304)를 포함하는 다양한 시스템 컴포넌트들을 시스템 메모리(306)에 연결시키는 시스템 버스(308)를 포함할 수 있지만, 이에 한정되는 것은 아니다.

[0100] 시스템 버스(308)는 메모리 버스 또는 메모리 제어기, 주변 버스, 고속 그래픽 포트 및 다양한 버스 아키텍처 중 임의의 것을 사용하는 프로세서 또는 로컬 버스를 포함하는 몇몇 유형의 버스 구조 중의 하나 이상의 임의의 것을 나타낸다. 예를 들어, 이러한 아키텍처는 CardBus, 개인용 컴퓨터 메모리 카드 국제 연합(PCMCIA), 고속 그래픽 포트(AGP), 소형 컴퓨터 시스템 인터페이스(SCSI), USB, IEEE 1394, 비디오 일렉트로닉스 표준 어소시에이션(VESA) 로컬 버스, 및 메자닌(Mezzanine) 버스로도 알려진 PCI 버스를 포함할 수 있다.

[0101] 컴퓨터(302)는 전형적으로 다양한 프로세서 판독가능 매체를 포함한다. 이러한 매체는 컴퓨터(302)에 의해 액세스될 수 있는 임의의 사용가능 매체일 수 있으며, 휘발성 및 비휘발성 매체, 분리형 및 비분리형 매체를 둘다 포함한다.

[0102] 시스템 메모리(306)는 RAM(310)과 같은 휘발성 메모리 및/또는 ROM(312)과 같은 비휘발성 메모리의 형태인 프로세서 판독가능 매체를 포함한다. 시동중과 같은 때에 컴퓨터(302) 내의 구성요소들 사이에 정보를 전송하는 것을 돕는 기본 루틴을 포함하는 기본 입출력 시스템(314; BIOS)은 ROM(312)에 저장된다. RAM(310)은 전

형적으로 프로세싱 유닛(304)에 즉시 액세스될 수 있고/있거나 그것에 의해 현재 동작되고 있는 데이터 및/또는 프로그램 모듈을 포함한다.

[0103] 컴퓨터(302)는 다른 분리형/비분리형, 휘발성/비휘발성 컴퓨터 저장 매체들도 포함할 수 있다. 예를 들어, 도 3은 비분리형 비휘발성 자기 매체(도시되지 않음)로부터 판독하거나 그 자기 매체에 기록하는 하드 디스크 드라이브(316), 분리형 비휘발성 자기 디스크(320)(예를 들어, "플로피 디스크")로부터 판독하거나 그 자기 디스크에 기록하는 자기 디스크 드라이브(318), 및 CD-ROM, DVD-ROM 또는 다른 광학 매체 등의 분리형 비휘발성 광학 디스크(324)로부터 판독하거나 그 광학 디스크에 기록하는 광학 디스크 드라이브(322)를 예시한다. 하드 디스크 드라이브(316), 자기 디스크 드라이브(318) 및 광학 디스크 드라이브(322) 각각은 하나 이상의 데이터 매체 인터페이스(325)에 의해 시스템 버스(308)에 접속된다. 대안적으로, 하드 디스크 드라이브(316), 자기 디스크 드라이브(318) 및 광학 디스크 드라이브(322)는 하나 이상의 인터페이스(도시되지 않음)에 의해 시스템 버스(308)에 접속될 수 있다.

[0104] 디스크 드라이브 및 그 관련 프로세서 판독가능 매체는 컴퓨터(302)에 대한 컴퓨터 판독가능 명령어, 데이터 구조, 프로그램 모듈 또는 다른 데이터의 비휘발성 저장소를 제공한다. 예가 하드 디스크(316), 분리형 자기 디스크(320) 및 분리형 광학 디스크(324)를 예시하지만, 자기 카세트 또는 다른 자기 저장 장치, 플래시 메모리 카드, CD-ROM, DVD 또는 다른 광학 저장소, RAM, ROM, EEPROM 등의 컴퓨터가 액세스할 수 있는 데이터를 저장할 수 있는 프로세서 판독가능 매체의 다른 유형이 예시적인 컴퓨팅 시스템 및 환경을 구현하는 데에도 사용될 수 있음이 이해될 것이다.

[0105] 예를 들어, 오퍼레이팅 시스템(326), 하나 이상의 어플리케이션 프로그램(328), 다른 프로그램 모듈(330) 및 프로그램 데이터(332)를 포함하는 임의의 수의 프로그램 모듈들은 하드 디스크(316), 자기 디스크(320), 광학 디스크(324), ROM(312) 및/또는 ROM(310) 상에 저장될 수 있다.

[0106] 사용자는 키보드(334) 및 포인팅 장치(336)(예를 들어, "마우스")와 같은 입력 장치를 통해 컴퓨터(302)에 코맨드 및 정보를 입력할 수 있다. 다른 입력 장치(338)(구체적으로 도시되지 않음)는 마이크로폰, 조이스틱, 게임 패드, 위성 안테나, 직렬 포트, 스캐너 등을 포함할 수 있다. 이런 및 다른 입력 장치는 시스템 버스(308)에 연결된 입출력 인터페이스(340)를 통해 프로세싱 유닛(304)에 접속되지만, 병렬 포트, 게임 포트 또는 USB와 같은 다른 인터페이스 및 버스 구조에 의해 접속될 수도 있다.

[0107] 모니터(342) 또는 다른 유형의 디스플레이 장치는 또한 비디오 어댑터(344)와 같은 인터페이스를 통해 시스템 버스(308)에 접속될 수 있다. 모니터(342) 이외에, 다른 출력 주변 장치는 입출력 인터페이스(340)를 통해 컴퓨터(302)에 접속될 수 있는 스피커(도시되지 않음) 및 프린터(346)와 같은 컴포넌트들을 포함할 수 있다.

[0108] 컴퓨터(302)는 원격 컴퓨팅 장치(348)와 같은 하나 이상의 원격 컴퓨터로의 논리적 접속을 사용하는 네트워크 환경에서 동작할 수 있다. 예를 들어, 원격 컴퓨팅 장치(348)는 개인용 컴퓨터, 휴대용 컴퓨터, 서버, 라우터, 네트워크 컴퓨터, 피어(peer) 장치 또는 다른 일반 네트워크 노드 등일 수 있다. 원격 컴퓨팅 장치(348)는, 컴퓨터(302)에 관련하여 여기에 설명된 구성요소 및 특징들 중 다수 또는 모두를 포함할 수 있는 휴대용 컴퓨터로 예시된다.

[0109] 컴퓨터(302)와 원격 컴퓨터(348) 사이의 논리적 접속은 근거리 통신망(LAN; 350) 및 일반적인 원격 통신망(WAN; 352)으로 표현된다. 이러한 네트워크 환경은 사무실, 기업 광역 컴퓨터 네트워크, 인트라넷 및 인터넷에서 일반적이다. 이러한 네트워크 환경은 유선 또는 무선일 수 있다.

[0110] LAN 네트워크 환경에서 구현될 때, 컴퓨터(302)는 네트워크 인터페이스 또는 어댑터(354)를 통해 LAN(350)에 접속된다. WAN 네트워크 환경에서 구현될 때, 컴퓨터(302)는 전형적으로 WAN(352)에 통신을 구축하기 위한 모뎀(356) 또는 다른 수단을 포함한다. 컴퓨터(302)에 대해서 내장형 또는 외장형일 수 있는 모뎀(356)은 입출력 인터페이스(340) 또는 다른 적절한 메커니즘을 통해 시스템 버스(308)에 접속될 수 있다. 예시된 네트워크 접속은 예시적인 것이며, 컴퓨터들(302와 348) 사이의 통신 링크(들)를 구축하는 다른 수단이 사용될 수 있음이 인식된다.

[0111] 네트워크 환경에서, 컴퓨팅 환경(300)에 예시된 바와 같이, 컴퓨터(302) 또는 그것의 일부에 관련해 설명된 프로그램 모듈들은 원격 메모리 저장 장치에 저장될 수 있다. 예를 들어, 원격 어플리케이션 프로그램(358)은 원격 컴퓨터(348)의 메모리 장치 상에 상주한다. 예시를 위해, 이러한 프로그램 및 컴포넌트들이 다양한 시간에 컴퓨팅 장치(302)의 상이한 저장 컴포넌트에 상주하고 컴퓨터의 데이터 프로세서(들)에 의해 실행되는 것으로 인식되지만, 어플리케이션 프로그램 및 오퍼레이팅 시스템과 같은 다른 실행 프로그램 컴포넌트는 여

기에 별개의 블록들로 예시된다.

[0112] 프로세서 실행가능 명령

[0113] 예시적인 퍼지 중복 검출기의 구현은 프로그램 모듈과 같은 하나 이상의 컴퓨터 또는 다른 장치들에 의해 실행되는 프로세서 실행가능 명령어의 일반적인 문맥으로 설명될 수 있다. 일반적으로, 프로그램 모듈은 특정한 태스크를 수행하거나 특정한 추상 데이터 유형을 구현하는 루틴, 프로그램, 오브젝트, 컴포넌트, 데이터 구조 등을 포함한다. 전형적으로, 프로그램 모듈의 기능은 다양한 실시예들에 설명된 바와 같이 조합 또는 분산될 수 있다.

[0114] 예시적인 운영 환경

[0115] 도 3은 예시적인 퍼지 중복 검출기가 구현될 수 있는 적절한 운영 환경(300)의 예를 예시한다. 특히, 여기에 설명된 예시적인 퍼지 중복 검출기(들)는 도 3 내의 프로그램 모듈들(328-330) 및/또는 운영 시스템(326) 또는 그것의 일부로 (전체적으로 또는 부분적으로) 구현될 수 있다.

[0116] 이 운영 환경은 적절한 운영 환경에 대한 예일 뿐이며, 여기에 설명된 예시적인 퍼지 중복 검출기(들)의 기능의 영역 또는 사용에 어떠한 제한을 두기 위한 것은 아니다. 사용하기에 적절한 다른 잘 알려진 컴퓨팅 시스템, 환경 및/또는 구성은 개인용 컴퓨터, 서버 컴퓨터, 핸드헬드 또는 랩탑 장치, 멀티프로세서 시스템, 마이크로프로세서 기반 시스템, 프로그램가능 가전 제품, 무선 전화 및 장치, 범용 및 특수 목적 가전 기기, ASIC, 네트워크 PC, 미니컴퓨터, 메인프레임 컴퓨터, 상기 시스템 또는 장치들 중 임의의 것을 포함하는 분산 컴퓨팅 환경을 포함하지만, 이에 한정된 것은 아니다.

[0117] 프로세서 판독가능 매체

[0118] 예시적인 퍼지 중복 검출기의 구현은 몇몇 형태의 프로세서 판독가능 매체에 저장되거나 그것을 통해 전달될 수 있다. 프로세서 판독가능 매체는 컴퓨터에 의해 액세스될 수 있는 임의의 사용가능 매체일 수 있다. 예를 들어, 프로세서 판독가능 매체는 "컴퓨터 저장 매체" 및 "통신 매체"를 포함할 수 있지만, 이에 한정된 것은 아니다.

[0119] "컴퓨터 저장 매체"는 컴퓨터 판독가능 명령어, 데이터 구조, 프로그램 모듈 또는 다른 데이터와 같은 정보를 저장하기 위한 임의의 방법 또는 기술로 구현된 휘발성 및 비휘발성, 분리형 및 비분리형 매체를 포함한다. 컴퓨터 저장 매체는 RAM, ROM, EEPROM, 플래시 메모리 또는 다른 메모리 기술, CD-ROM, DVD, 다른 광학 저장소, 자기 카세트, 자기 테이프, 자기 디스크 저장소 또는 다른 자기 저장 장치, 또는 원하는 정보를 저장하는데 사용될 수 있고 컴퓨터가 액세스할 수 있는 임의의 다른 매체들을 포함하지만, 이에 한정된 것은 아니다.

[0120] "통신 매체"는 전형적으로 프로세서 판독가능 명령어, 데이터 구조, 프로그램 모듈 또는 다른 데이터를 반송파 또는 다른 전송 메커니즘과 같은 변조된 데이터 신호로 구현한다. 통신 매체는 임의의 정보 전달 매체도 포함한다.

[0121] "변조된 데이터 신호"라는 용어는 신호 내에 정보를 인코딩하는 방식으로 설정 또는 변환된 특성을 하나 또는 그 이상을 갖는 신호를 의미한다. 예를 들어, 통신 매체는 유선 네트워크 또는 직접 유선 접속과 같은 유선 매체와, 음향, RF, 적외선 및 다른 무선 매체와 같은 무선 매체를 포함하지만, 이에 한정된 것은 아니다. 상술한 것들 중의 임의의 조합도 프로세서 판독가능 매체의 영역 내에 포함된다.

[0122] 결론

[0123] 하나 이상의 상술된 구현이 구조적 특징들 및/또는 방법론적 단계들에 특정한 언어로 설명되었지만, 설명된 특정한 특징들 또는 단계들 없이도 다른 구현들이 구현될 수 있음이 이해된다. 오히려, 특정한 특징들 또는 단계들은 하나 이상의 구현들에 대한 바람직한 형태로 나타내진 것이다.

발명의 효과

[0124] 본 발명은 퍼지 중복을 검출하고 이러한 중복들을 제거하는 방법을 제공한다.

도면의 간단한 설명

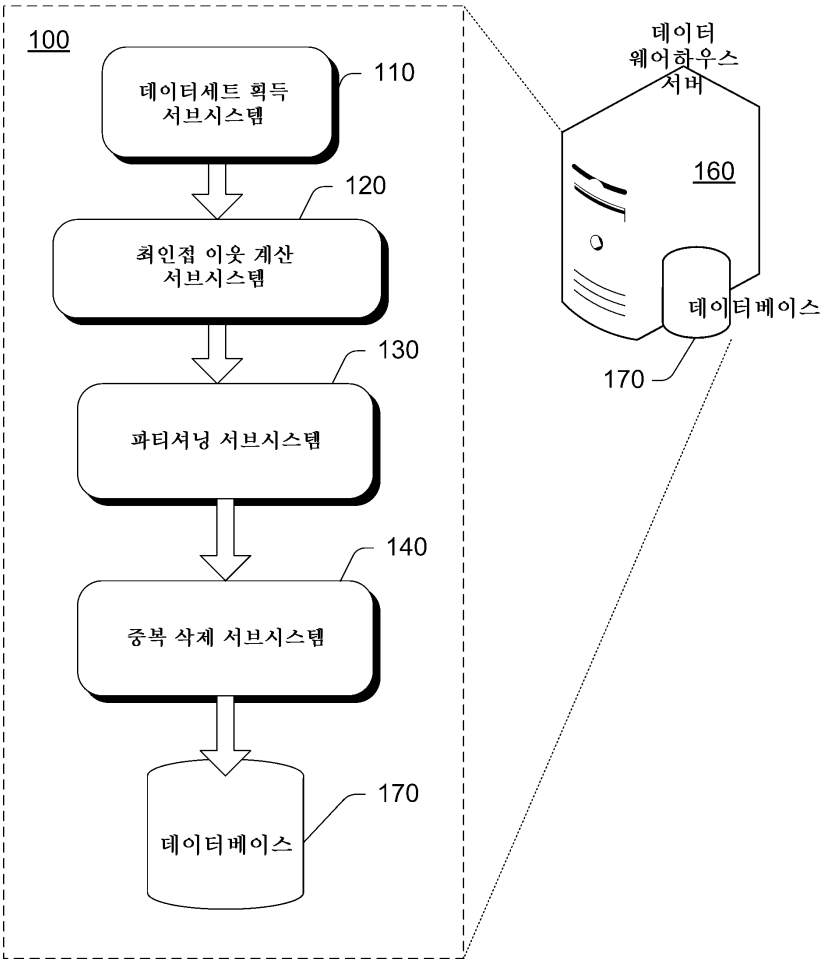
[0001] 도 1은 여기에 설명된 구현에 대한 블록도.

[0002] 도 2는 여기에 설명된 방법론적 구현을 나타내는 순서도.

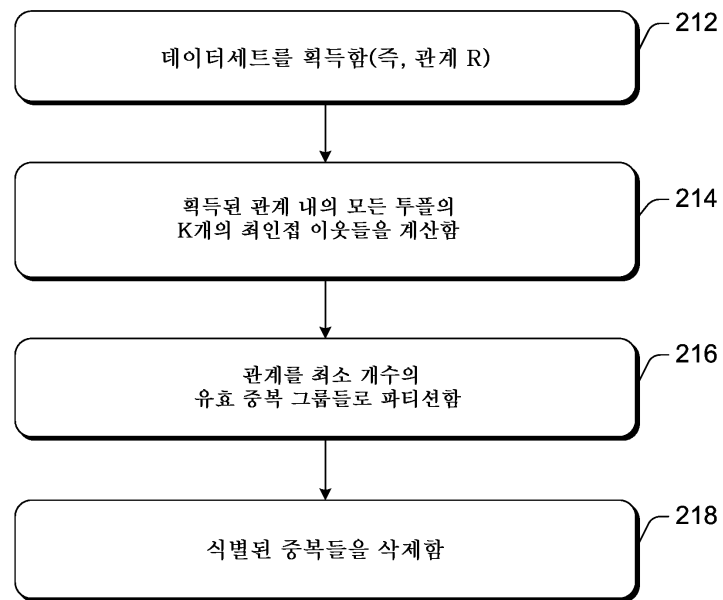
- [0003] 도 3은 여기에 설명된 적어도 하나의 실시예를 (전체적으로 또는 부분적으로) 구현할 수 있는 컴퓨팅 동작 환경의 예를 나타내는 도면.
- [0004] <도면의 주요 부분에 대한 부호의 설명>
- [0005] 110: 데이터세트 획득 서브시스템
- [0006] 120: 확인점 이웃 계산 서브시스템
- [0007] 130: 파티셔닝 서브시스템
- [0008] 160: 데이터 웨어하우스 서버
- [0009] 170: 데이터베이스

도면

도면1



도면2



도면3

