(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2015-0123823
(43) 공개일자 2015년11월04일

(51) 국제특허분류(Int. Cl.)
H04L 9/00 (2006.01)
(52) CPC특허분류
H04L 9/008 (2013.01)
H04L 2209/46 (2013.01)
(21) 출원번호 10-2015-7023956
(22) 출원일자(국제) 2013년09월25일
심사청구일자 없음
(85) 번역문제출일자 2015년09월02일
(86) 국제출원번호 PCT/US2013/061696
(87) 국제공개번호 WO 2014/137393
국제공개일자 2014년09월12일
(30) 우선권주장
61/772,404 2013년03월04일 미국(US)

(71) 출원인
톰슨 라이센싱
프랑스 92130 이씨레뮬리노 루 잔다르크 1-5
(72) 발명자
니콜랜코 발레리아
미국, 캘리포니아 94305, 스탠포드, 에이피티. 800, 아브람스 씨티. 63
바인스베르그 우디
미국, 캘리포니아 94025, 멘로 파크, 아비 애비뉴 2135
(뒷면에 계속)
(74) 대리인
문경진, 안문환

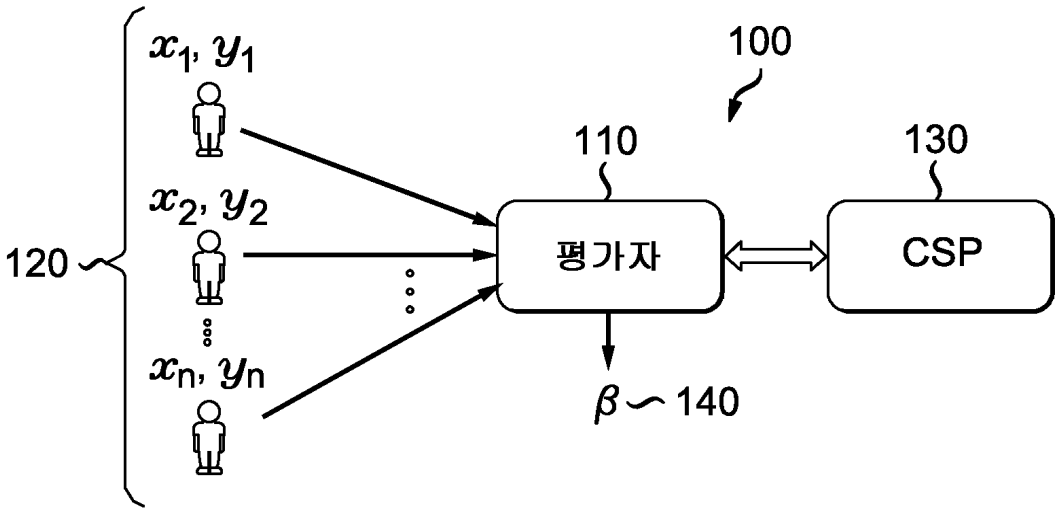
전체 청구항 수 : 총 15 항

(54) 발명의 명칭 **마스크들을 사용한 프라이버시-보존 리지 회귀**

(57) 요 약

마스크들을 사용한 프라이버시-보존 리지 회귀를 위한 방법 및 시스템이 제공된다. 본 방법은 크립토 서비스 제공자로부터 왜곡된 회로를 요청하는 단계, 포맷팅되고 준동형 암호화를 사용하여 암호화된 다수의 사용자들로부터의 데이터를 수집하는 단계, 포맷팅되고 준동형 암호화를 사용하여 암호화된 데이터를 합산하는 단계, 준비된 마스크들을 합산된 데이터에 적용하는 단계, 불확정 전송을 사용하여 크립토 서비스 제공자로부터, 준비된 마스크에 대응하는 왜곡된 입력들을 수신하는 단계, 및 왜곡된 입력들 및 마스킹된 데이터를 사용하여 크립토 서비스 제공자로부터의 왜곡된 회로를 평가하는 단계를 포함한다.

대 표 도 - 도1



(52) CPC특허분류
H04L 2209/50 (2013.01)
(72) 발명자
이온니디스 스트라티스
미국, 캘리포니아 94133, 샌 프란시스코, 테일러 스트리트 1961
조예 마르크
프랑스, 푸게레스 에프-35300, 루 두 제너럴 차지 22
타프트 니나
미국, 캘리포니아 94114, 샌 프란시스코, 24번 스트리트 4265

명 세 서

청구범위

청구항 1

프라이버시-보존 리지 회귀(privacy-preserving ridge regression)를 제공하기 위한 방법으로서,

크립토 서비스 제공자로부터 왜곡된 회로(garbled circuit)를 요청하는 단계;

포맷팅되고 준동형 암호화를 사용하여 암호화된 다수의 사용자들로부터의 데이터를 수집하는 단계;

포맷팅되고 준동형 암호화를 사용하여 암호화된 데이터를 합산하는 단계;

준비된 마스크들을 합산된 데이터에 적용하는 단계;

불확정 전송(oblivious transfer)을 사용하여 크립토 서비스 제공자로부터, 준비된 마스크에 대응하는 왜곡된 입력들을 수신하는 단계; 및

왜곡된 입력들 및 마스킹된 데이터를 사용하여 크립토 서비스 제공자로부터의 왜곡된 회로를 평가하는 단계를

포함하는, 프라이버시-보존 리지 회귀를 제공하기 위한 방법.

청구항 2

제1항에 있어서, 크립토 서비스 제공자로부터 왜곡된 회로를 요청하는 단계는

왜곡된 회로에 대한 입력 변수들의 차원을 제공하는 단계; 및

입력 변수들의 값 범위를 제공하는 단계를

포함하는, 프라이버시-보존 리지 회귀를 제공하기 위한 방법.

청구항 3

제1항에 있어서, 컴퓨팅 디바이스 상에 구현된 평가자는 상기 방법을 수행하는, 프라이버시-보존 리지 회귀를 제공하기 위한 방법.

청구항 4

제3항에 있어서, 크립토 서비스 제공자는, 평가자가 구현되는 컴퓨팅 디바이스로부터 떨어져 있는 컴퓨팅 디바이스 상에 구현되는, 프라이버시-보존 리지 회귀를 제공하기 위한 방법.

청구항 5

제1항에 있어서, 다수의 사용자들로부터의 데이터를 암호화하기 위한 암호화 키를 제공하는 단계를 더 포함하는, 프라이버시-보존 리지 회귀를 제공하기 위한 방법.

청구항 6

제5항에 있어서, 다수의 사용자들로부터의 데이터는 크립토 서비스 제공자에 의해 제공된 암호화 키를 통해 더 암호화되는, 프라이버시-보존 리지 회귀를 제공하기 위한 방법.

청구항 7

제1항에 있어서, 왜곡된 회로를 평가하는 단계는

합산된 데이터로부터 준비된 마스크를 제거하는 단계; 및

왜곡된 회로에 의해 구체화된 리지 선형 방정식을 해답하는 단계를

더 포함하는, 프라이버시-보존 리지 회귀를 제공하기 위한 방법.

청구항 8

제1항에 있어서, 다수의 사용자들로부터 데이터를 수집하는 단계는 컴퓨팅 디바이스를 통해 다수의 사용자들 각각으로부터 전송된 데이터를 수신하는 단계를 포함하는, 프라이버시-보존 리지 회귀를 제공하기 위한 방법.

청구항 9

프라이버시-보존 리지 회귀를 제공하기 위한 컴퓨팅 디바이스로서,

사용자 데이터를 저장하기 위한 저장소;

프로세싱을 위한 데이터를 저장하기 위한 메모리; 및

프로세서로서, 크립토 서비스 제공자로부터 왜곡된 회로를 요청하고, 포맷팅되고 준동형 암호화를 사용하여 암호화된 다수의 사용자들로부터의 데이터를 수집하고, 포맷팅되고 준동형 암호화를 사용하여 암호화된 데이터를 합산하고, 준비된 마스크들을 합산된 데이터에 적용하고, 불확정 전송을 사용하여 크립토 서비스 제공자로부터, 마스킹된 데이터에 대응하는 왜곡된 입력들을 수신하고, 왜곡된 입력들 및 마스킹된 데이터를 사용하여 크립토 서비스 제공자로부터의 왜곡된 회로를 평가하도록 구성된, 프로세서를

포함하는, 컴퓨팅 디바이스.

청구항 10

제9항에 있어서, 네트워크에 연결하기 위한 네트워크 연결부를 더 포함하는, 컴퓨팅 디바이스.

청구항 11

제9항에 있어서, 크립토 서비스 제공자는 분리된 컴퓨팅 디바이스 상에 구현되는, 컴퓨팅 디바이스.

청구항 12

제9항에 있어서, 크립토 서비스 제공자로부터 왜곡된 회로를 요청하는 단계는

왜곡된 회로에 대한 입력 변수들의 차원을 제공하는 단계; 및

입력 변수들의 값 범위를 제공하는 단계를

포함하는, 컴퓨팅 디바이스.

청구항 13

제9항에 있어서, 왜곡된 회로를 평가하는 단계는

합산된 데이터로부터 준비된 마스크를 제거하는 단계; 및

왜곡된 회로에 의해 구체화된 리지 선형 방정식을 해답하는 단계를

더 포함하는, 컴퓨팅 디바이스.

청구항 14

제9항에 있어서, 다수의 사용자들로부터의 데이터는 크립토 서비스 제공자에 의해 제공된 암호화 키를 통해 암호화되고, 컴퓨팅 디바이스에 의한 암호화 키를 통해 암호화되는, 컴퓨팅 디바이스.

청구항 15

기계 판독 가능한 매체로서,

실행될 때,

크립토 서비스 제공자로부터 왜곡된 회로를 요청하는 단계;

포맷팅되고 준동형 암호화를 사용하여 암호화된 다수의 사용자들로부터의 데이터를 수집하는 단계;

포맷팅되고 준동형 암호화를 사용하여 암호화된 데이터를 합산하는 단계;

준비된 마스크들을 합산된 데이터에 적용하는 단계;

불확정 전송을 사용하여 크립토 서비스 제공자로부터, 준비된 마스크에 대응하는 왜곡된 입력들을 수신하는 단계; 및

왜곡된 입력들 및 마스킹된 데이터를 사용하여 크립토 서비스 제공자로부터의 왜곡된 회로를 평가하는 단계를

포함하는 단계들을 수행하는 명령어들을 포함하는, 기계 판독 가능한 매체.

발명의 설명

기 술 분 야

[0001] 본 출원은 2013년 03월 04일에 제출된 미국 가특허 출원번호 제61/772,404호의 이익을 주장하고, 이는 그 전체가 본 명세서에서 참조로서 병합된다.

[0002] 본 출원은 또한, 동시에 제출되었으며 그 전체가 본 명세서에서 참조로서 병합되는 "PRIVACY-PRESERVING RIDGE REGRESSION" 및 "PRIVACY-PRESERVING RIDGE REGRESSION USING PARTIALLY HOMOMORPHIC ENCRYPTION AND MASKS"라는 명칭이 부여된 출원들에 관련된다.

[0003] 본 발명은 일반적으로 데이터 마이닝(data mining)에 관한 것이고, 보다 구체적으로 리지 회귀(ridge regression)를 사용한 데이터 마이닝 동안에 프라이버시를 보호하는 것에 관한 것이다.

배 경 기 술

[0004] 추천 시스템들은, 상이한 항목들에 대한 많은 사용자들의 선호도들 및 등급들을 수집하고 데이터에 대한 학습 알고리즘을 실행함으로써 연산한다. 학습 알고리즘은 새로운 사용자가 특정 항목들에 어떻게 등급을 매길 것인지를 예측하도록 사용될 수 있는 모델을 생성한다. 특히, 사용자가 특정 항목들에 대해 제공하는 등급들이 주어지는 경우, 모델은 사용자가 다른 항목들에 어떻게 등급을 매길 것인지를 예측할 수 있다. 이러한 예측 모델들을 생성하기 위한 방대한 알고리즘들이 존재하고, 많은 알고리즘들이 아마존 및 넷플릭스와 같은 대형 사이트들에서 실제로 사용된다. 학습 알고리즘들은 또한 대량의 의학 데이터베이스들, 재무 데이터, 및 많은 다른 도메인들에서 사용된다.

[0005] 현재의 구현들에서, 학습 알고리즘은 예측 모델을 조성하기 위해 명문으로(in the clear) 모든 사용자 데이터를 확인해야 한다. 이러한 개시사항에서, 학습 알고리즘이 명문의 데이터 없이 연산할 수 있는지의 여부가 결정되고, 이로써 사용자들이 그들의 데이터의 제어를 유보하는 것을 허용한다. 의학 데이터에 대해, 이는 모델이 사용자 프라이버시에 영향을 주지 않고도 조성되는 것을 허용한다. 책 및 영화 선호도들에 대해, 사용자들이 이들의 데이터의 제어를 유지하게 하는 것은 서비스 제공자에 있어서의 데이터 침해(data breach)의 경우 미래의 예상치 못한 곤란한 상황의 위험을 감소시킨다. 대충 이야기하면, 비공개 사용자 데이터를 데이터-마이닝하는 것에 대한 기존의 세 가지의 접근법들이 존재한다. 제1 접근법은 사용자들이 비밀 공유를 사용하여 다수의 서버 사이에서 이들의 데이터를 분할하게 한다. 그러고 나서, 이들 서버들은 배포된 프로토콜을 사용하여 학습 알고리즘을 실행하고, 대다수의 서버가 공모하지 않는 한 프라이버시가 보장된다. 제2 접근법은 완전한 준동형 암호화에 기초하는데, 여기서 학습 알고리즘은 암호화된 데이터에 걸쳐 실시되고, 신임을 받은 제3 파티는 최종의 암호화된 모델만을 단지 암호 해독하도록 신임을 받는다. 제3 접근법에서, Yao의 왜곡된 회로 구축은, 그 밖의 사용자 데이터에 관한 어떤 것도 학습하지 않고도 암호화된 데이터를 컴퓨팅하고 최종 모델을 획득하도록 사용될 수 있다. 하지만, Yao에 기초한 접근법은 이전에 알고리즘들의 회귀 클래스에 전혀 적용되지 않았다.

발명의 내용

해결하려는 과제

[0006] 본 개시사항의 초점은 많은 학습 알고리즘들, 즉 리지 회귀에서 사용된 기본적인 메커니즘에 있다. 높은 차원에서의 많은 수의 지점들이 주어지면, 회귀 알고리즘은 이들 지점들을 통하는 최적합 곡선을 산출한다. 목표는 사용자 데이터나 사용자 데이터에 관한 임의의 다른 정보를 노출하지 않고 컴퓨테이션을 수행하는 것이다. 이는 도 1에 도시된 바와 같이 시스템을 사용함으로써 성취된다.

과제의 해결 수단

[0007] 준동형 암호화 및 Yao 왜곡된 회로들 모두를 사용하는 프라이버시-보존 리지 회귀에 대한 하이브리드 접근법이 제시된다. 본 시스템에서의 사용자들은 Paillier 또는 Regev와 같은 선형적 준동형 암호화 시스템 하에서 암호화된 이들의 데이터를 제출한다. 평가자는 오로지 선형 연산들만을 요구하는 알고리즘의 제1 국면을 수행하기 위해 선형 준동형(linear homomorphism)을 사용한다. 이 국면은 암호화된 데이터를 생성한다. 이러한 제1 국면에서, 본 시스템은 (시스템 n에서의 사용자들의 수에 비례하는) 기록들의 큰 수를 프로세싱하도록 요구받는다. 이러한 제1 국면에서의 프로세싱은 알고리즘의 제2 국면이 n에 독립적이도록 하기 위한 데이터를 준비한다. 제2 국면에서, 평가자는, 먼저 준동형 암호 해독을 구현한 뒤, 그러고 나서 회귀 알고리즘의 나머지를 수행하는 Yao 왜곡된 회로를 평가한다(보여진 바와 같이, 최적화된 실현은 왜곡된 회로에서의 암호 해독을 회피할 수 있다). 이러한 회귀 알고리즘의 단계는 빠른 선형 시스템 해답기를 요구하고, 매우 비-선형적이다. 이 단계에 대해, Yao 왜곡된 회로 접근법은 현재의 완전한 준동형 암호화 스킴들보다 훨씬 더 빠르다. 따라서, 대량의 데이터 집합을 다루기 위해 선형 준동형을 사용하고, 컴퓨테이션의 막대한(heavy) 비-선형 부분을 위해 왜곡된 회로들을 사용함으로써 일거양득의 기회가 획득된다. 제2 국면은 또한 컴퓨테이션이 두 개의 국면으로 분할되는 방식으로 인해 n에 독립적이다.

[0008] 한 실시예에서, 프라이버시-보존 리지 회귀를 위한 방법이 제공된다. 본 방법은 크립토 서비스 제공자로부터 왜곡된 회로를 요청하는 단계, 포맷팅되고 준동형 암호화를 사용하여 암호화된 다수의 사용자들로부터의 데이터를 수집하는 단계, 포맷팅되고 준동형 암호화를 사용하여 암호화된 데이터를 합산하는 단계, 준비된 마스크들을 합산된 데이터에 적용하는 단계, 불확정 전송(oblivious transfer)을 사용하여 크립토 서비스 제공자로부터, 준비된 마스크에 대응하는 왜곡된 입력들을 수신하는 단계, 및 왜곡된 입력들 및 마스킹된 데이터를 사용하여 크립토 서비스 제공자로부터의 왜곡된 회로를 평가하는 단계를 포함한다.

[0009] 또 다른 실시예에서, 프라이버시-보존 리지 회귀를 위한 컴퓨팅 디바이스로가 제공된다. 컴퓨팅 디바이스는 저장소, 메모리, 및 프로세서를 포함한다. 저장소는 사용자 데이터를 저장하기 위한 것이다. 메모리는 프로세싱을 위한 데이터를 저장하기 위한 것이다. 프로세서는, 크립토 서비스 제공자로부터 왜곡된 회로를 요청하고, 포맷팅되고 준동형 암호화를 사용하여 암호화된 다수의 사용자들로부터의 데이터를 수집하고, 포맷팅되고 준동형 암호화를 사용하여 암호화된 데이터를 합산하고, 준비된 마스크들을 합산된 데이터에 적용하고, 불확정 전송을 사용하여 크립토 서비스 제공자로부터, 준비된 마스크에 대응하는 왜곡된 입력들을 수신하고, 왜곡된 입력들 및 마스킹된 데이터를 사용하여 크립토 서비스 제공자로부터의 왜곡된 회로를 평가하도록 구성된다.

[0010] 청구항들에서 특별히 지적된 요소들 및 결합들(couplings)에 의해 목적들 및 장점들이 실현 및 달성될 것이다. 개시된 실시예들은 본 명세서에서의 혁신적인 가르침들의 많은 유리한 사용들에 대한 예시들일 뿐이라는 점에 주목하는 것이 중요하다. 주장된 바와 같이, 전술한 일반적인 설명 및 다음의 상세한 설명 모두는 예시적 및 설명적이고 본 발명을 제한하지 않는다는 점이 이해될 것이다. 또한, 일부 진술들은 일부 발명의 특징들에는 적용될 수 있지만, 다른 것들에는 적용되지 않을 수도 있다. 일반적으로, 달리 표시되지 않는다면, 일반성의 손실 없이, 단일 요소가 복수로 되어 있을 수도 있고, 복수 요소가 단수로 되어 있을 수도 있다. 도면들에서, 유사 부호들은 여러 도면들에 걸쳐 유사한 부분들을 지칭한다.

발명의 효과

[0011] 본 개시사항의 초점은 많은 학습 알고리즘들, 즉 리지 회귀에서 사용된 기본적인 메커니즘에 있다. 높은 차원에서의 많은 수의 지점들이 주어지면, 회귀 알고리즘은 이들 지점들을 통하는 최적합 곡선을 산출한다. 목표는 사용자 데이터나 사용자 데이터에 관한 임의의 다른 정보를 노출하지 않고 컴퓨테이션을 수행하는 것이다. 이는 도 1에 도시된 바와 같이 시스템을 사용함으로써 성취된다.

도면의 간단한 설명

[0012] 도 1은 한 실시예에 따른 프라이버시-보존 리지 회귀 시스템의 개략적인 블록도를 도시하는 도면.

도 2는 한 실시예에 따른 컴퓨팅 디바이스의 개략적인 블록도를 도시하는 도면.

도 3은 한 실시예에 따른 예시적인 왜곡된 회로를 도시하는 도면.

도 4는 실시예에 따른 프라이버시-보존 리지 회귀를 제공하기 위한 방법론의 하이 레벨 흐름도를 도시하는 도면.

도 5는 실시예에 따른 프라이버시-보존 리지 회귀를 제공하기 위한 제1 프로토콜의 연산을 도시하는 도면.

도 6은 실시예에 따른 프라이버시-보존 리지 회귀를 제공하기 위한 제1 프로토콜의 연산을 도시하는 도면.

도 7은 실시예에 따른 Cholesky 분해를 위한 알고리즘의 한 예시적인 실시예를 도시하는 도면.

발명을 실시하기 위한 구체적인 내용

[0013] 본 개시사항의 초점은 많은 학습 알고리즘들, 즉 리지 회귀에서 사용된 기본적인 메커니즘에 있다. 높은 차원에서의 많은 수의 지점들이 주어지면, 회귀 알고리즘은 이들 지점들을 통하는 최적합 곡선을 산출한다. 목표는 사용자 데이터나 사용자 데이터에 관한 임의의 다른 정보를 노출하지 않고 컴퓨테이션을 수행하는 것이다. 이는 도 1에 도시된 바와 같이 시스템을 사용함으로써 성취된다.

[0014] 도 1에서, 프라이버시-보존 리지 회귀를 구현하기 위한 시스템(100)의 한 실시예의 블록도가 제공된다. 시스템은 평가자(110), 하나 이상의 사용자들(120) 및 크립토 서비스 제공자(CSP)(130)를 포함하는데, 이들은 서로 통신한다. 평가자(110)는 서버 또는 개인 컴퓨터(PC)와 같은 컴퓨팅 디바이스 상에 구현된다. CSP(130)는 유사하게도 서버 또는 개인 컴퓨터와 같은 컴퓨팅 디바이스 상에 구현되고, 이더넷 또는 와이-파이 네트워크와 같은 네트워크를 통해 평가자(110)와 통신한다. 하나 이상의 사용자들(120)은 개인 컴퓨터들, 태블릿들, 또는 스마트폰들 등과 같은 컴퓨팅 디바이스들을 통해 평가자(110) 및 CSP(130)와 통신한다.

[0015] 사용자들(120)은 학습 알고리즘을 실행하는 (예를 들어, 서버상의) 평가자(110)에 (예를 들어, PC로부터의) 암호화된 데이터를 전송한다. 특정 지점들에서, 평가자는, 평가자(110)와 공모하지 않는 것으로 신임을 받는 (또 다른 서버상의) 크립토 서비스 제공자(130)와 상호작용할 수 있다. 최종적인 결과는 평문 예측 모델(β 140)이다.

[0016] 도 2는 프라이버시-보호 리지 회귀를 위한 다양한 방법론 및 시스템 요소들을 구현하도록 사용될 수 있는 서버, PC, 태블릿, 또는 스마트폰과 같은 한 예시적인 컴퓨팅 디바이스(200)를 도시한다. 컴퓨팅 디바이스(200)는 하나 이상의 프로세서(210), 메모리(220), 저장소(230), 및 네트워크 인터페이스(240)를 포함한다. 이들 요소들 각각은 아래에 보다 상세하게 논의될 것이다.

[0017] 프로세서(210)는 전자 서버(200)의 연산을 제어한다. 프로세서(200)는, 서버를 연산시키고 콜드 스타트 추천들(cold start recommendations)의 기능을 제공하는 소프트웨어를 실행한다. 프로세서(210)는 메모리(220), 저장소(230), 및 네트워크 인터페이스(240)에 연결되고, 이는 이들 요소들 사이에서의 정보의 송신 및 프로세싱을 다룬다. 프로세서(210)는 일반적인 프로세서 또는 특정 기능을 위해 전용된 프로세서일 수 있다. 특정 예시들에서 다수의 프로세서들이 존재할 수 있다.

[0018] 메모리(220)는 프로세서에 의해 실행될 명령어들 및 데이터가 저장되는 장소이다. 메모리(210)는 휘발성 메모리(RAM), 비-휘발성 메모리(EEPROM), 또는 다른 적절한 미디어를 포함할 수 있다.

[0019] 저장소(230)는 본 발명의 콜드 저장 추천 방법론을 실행하는 프로세서에서 사용 및 산출된 데이터가 저장되는 장소이다. 저장소는 자기 미디어(하드 드라이브), 광학 미디어(CD/DVD-롬), 또는 플래쉬 기반의 저장소일 수 있다.

[0020] 네트워크 인터페이스(240)는 네트워크를 통한 서버(200)와 다른 디바이스들의 통신을 다룬다. 적절한 네트워크의 한 예시는 이더넷 네트워크이다. 적절한 홈 네트워크들의 다른 유형들은 본 개시사항의 이점이 주어질 때 당업자에게 명백할 것이다.

[0021] 도 2에 열거된 요소들은 예시적이라는 점이 이해될 것이다. 서버(200)는 임의의 수의 요소들을 포함할 수 있고, 특정 요소들은 다른 요소들의 기능의 부분 또는 전부를 제공할 수 있다. 다른 가능한 구현은 본 개시사항의 이점이 주어질 때 당업자에게 명백할 것이다.

[0022] 설정 및 위협 모델

[0023] A. 체계들 및 엔티티들

[0024] 도 1을 다시 언급하면, 많은 사용자들(120)이 평가자(110)라고 부르는 중앙 서버에 데이터를 기여하는 시스템(100)이 설계되었다. 평가자(110)는 기여된 데이터를 통한 회귀를 수행하고, 모델(β 140)을 산출하는데, 이는 예측 또는 추천 과제들을 위해 나중에 사용될 수 있다. 보다 구체적으로, 각 사용자 i=1;:::;n은 두 개의 변수들 $x_i \in \mathbb{R}^d$ 및 $y_i \in \mathbb{R}$를 포함하는 비공개 기록을 갖고, 평가자는 $y_i \approx \beta^T x_i$이도록 - 모델 - $\beta \in \mathbb{R}^d$를 계산하기를 원한다. 목표는 회귀 알고리즘의 최종 결과인 β 140에 의해 드러나는 것 이상으로 사용자의 기록들에 관

한 어떤 것도 학습하지 않는다는 점을 보장하는 것이다. 시스템을 초기화하기 위해, 제3 파티가 요구되는데, 이는 본 명세서에서 오프라인 작업의 대부분을 수행하는 "크립토 서비스 제공자(Crypto Service Provider)"라고 지칭된다.

[0025] 보다 정확하게, 시스템 내의 파티들은 도 1에 도시된 대로 다음과 같다.

[0026] ● 사용자들(120): 각 사용자 i는 비공개 데이터 x_i, y_i를 가지며, 암호화된 데이터를 평가자(110)에 전송한다.

[0027] ● 평가자(110): 암호화된 데이터에 대한 회귀 알고리즘을 실행하고, 명문의 학습되 모델(β 140)을 획득한다.

[0028] ● 크립토 서비스 제공자(CSP)(130): 평가자(110) 및 사용자들(120)에게 설정 파라미터들을 제공함으로써 시스템(100)을 초기화한다.

[0029] CSP(130)는 사용자들(120)이 이들의 데이터를 평가자(110)에 기여하기 전에 작업의 대부분을 오프라인으로 장기간 수행한다. 대부부의 효율적인 설계에 있어서, 평가자(110)가 모델(β 140)을 컴퓨팅할 때 짧은 일-라운드 온라인 단계 동안에 CSP(130)가 또한 요구된다.

[0030] B. 위협 모델

[0031] 목표는 평가자(110) 및 CSP(130)가 학습 알고리즘의 최종 결과들에 의해 드러나는 것 이상으로 사용자들(120)에 의해 기여된 데이터에 관한 어떤 것도 수행할 수 없다는 점을 보장하는 것이다. 평가자(110)가 사용자들(120)의 일부와 공모하는 경우, 사용자들(120)은 학습 알고리즘의 결과들에 의해 드러나는 것 이상으로 다른 사용자들(120)에 의해 기여된 데이터에 관한 어떤 것도 학습하지 못할 것이다.

[0032] 이러한 예시에서, 정확한 모델(β 140)을 산출하는 것에 가장 관심이 있는 것이 평가자(110)라는 것이 가정된다. 이에 따라, 이 실시예는 부정확한 결과를 산출한다는 희망으로 컴퓨테이션을 부패시키려고 시도하는 악의 있는 평가자(110)에 관한 것은 아니다. 하지만, 평가자(110)는, 부정한 짓을 하고 사용자들(120)에 의해 기여된 비공개 데이터에 관한 정보를 학습하도록 동기가 부여되는데, 이는 이러한 데이터가 잠재적으로 다른 파티들, 예컨대 광조주들에게 매수될 수 있기 때문이다. 따라서, 심지어 악의 있는 평가자(110)는 학습 알고리즘의 결과들에 의해 드러나는 것 이상으로 사용자 데이터에 관한 어떤 것도 학습할 수 없을 것이다. 정직하지만-호기심이 있는 평가자에 대해서만 안전한 기본적인 프로토콜이 본 명세서에 열거된다.

[0033] 비-위협들: 본 시스템은 다음의 공격들을 방어하도록 설계되지 않는다.

[0034] ● 평가자(110) 및 CSP(130)가 공모하지 않는다는 것이 가정된다. 각각은 앞서 논의된 바와 같이 시스템을 파괴하려고 시도할 수 있지만, 이들은 독립적으로 그렇게 한다. 보다 정확하게, 보안을 논의하면, 이들 두 개의 파티들 중 최대 하나가 악의 있는 것(이는 어떤 보안도 그것 없이는 성취될 수 없는 본래의 필수조건이라는 것)임이 가정된다.

[0035] ● 설정이 정확하게 동작한다는 것, 즉 모든 사용자들(120)이 CSP(130)로부터 정확한 비공개 키를 획득한다는 것이 가정된다. 이는 인증 기관들(Certificate Authorities)의 적절한 사용으로 실제로 시행될 수 있다.

[0036] 배경

[0037] A. 선형 모델 학습

[0038] 간략하게 리지 회귀를 검토하면, 평가자(110)가 β 140을 학습하기 위해 시스템(110)에서 수행하는 알고리즘이 제공된다. 아래에 논의된 모든 결과들은 전형적(classic)이고, 이는 대부분의 통계학 및 기계 학습 텍스트북들에서 확인될 수 있다.

[0039] *선형 회귀*: n 개의 입력 변수들의 집합 $x_i \in \mathbb{R}^d$, 및 출력 변수들의 집합 $y_i \in \mathbb{R}$을 가정하면, $y_i \simeq f(x_i)$이기 위한 함수 f: $\mathbb{R}^d \rightarrow \mathbb{R}$을 학습하는 것의 문제는 *회귀*로서 알려진다. 예를 들어, 입력 변수들은 개인의 나이, 체중, 체질량 지수 등일 수 있지만, 출력은 질병에 걸릴 가능성일 수 있다.

[0040] 실제 데이터로부터 이러한 함수를 학습하는 것은 회귀가 데이터 마이닝, 통계학, 및 기계 학습에 편재하게 만드는 많은 흥미로운 애플리케이션들을 갖는다. 한 편에서, 함수 그 자체는 예측을 위해, 즉 새로운 입력 $x \in \mathbb{R}^d$의 출력 값 y를 예측하기 위해 사용될 수 있다. 또한, f의 구조는 상이한 입력들이 출력에 어떻게 영향을 주는지를 식별하는 것 - 예컨대, 나이보다는 오히려 체중이 질병에 더 강력하게 상관된다는 것을 확립하는 것을 지

원할 수 있다.

[0041] *선형 회귀*는 일부 $\beta \in \mathbb{R}^d$에 대해 f가 선형 맵, 즉 $y_i \backsimeq \beta^T x_i$, $i \in [n] \equiv 1,...,n$에 의해 잘 근사화된다는 전제에 기초한다. 선형 회귀는 과학에 있어서 추론 및 통계학적 분석을 위해 가장 널리 사용된 방법들 중 하나이다. 또한, 이것은 커널 방법들과 같은 통계학적 분석 및 기계 학습에 있어서의 보다 진보된 여러 방법들에 대한 기본적인 조성 블록(building block)이다. 예를 들어, 2차 다항식인 함수를 학습하는 것은 $1 \leq k, k' \leq d$에 대해 $x_{ik}x_{ik'}$에 걸쳐 선형 회귀로 감소하고; 기본 함수들의 유한 집합에 의해 걸쳐진(spanned) 임의의 함수를 학습하기 위해 동일한 원리가 일반화될 수 있다.

[0042] 앞서 언급된 바와 같이, 예측을 위한 명백한 사용 이상으로, 벡터 $\beta=(\beta_k)_{k=1,...,d}$는 y가 입력 변수들에 어떻게 의존하는지를 드러내기 때문에 흥미롭다. 특히, 계수 β_k의 부호는 출력에 대한 양 또는 음 상관을 표시하지만, 크기(magnitude)는 상대적인 중요성을 획득한다. 이들 계수들이 필적한다는 것을 보장하기 위해, 뿐만 아니라 수치적인 안정성을 위해, 입력들 x_i는 동일하고 유한한 도메인(예컨대, [-1; 1])으로 재 스케일링된다.

[0043] *계수들을 컴퓨팅*: 벡터 $\beta \in \mathbb{R}^d$를 컴퓨팅하기 위해, 후자(latter)는 $\mathbb{R}^d$에 걸쳐 다음의 2차 함수를 최소화함으로써 데이터에 맞추어진다:

수학식 1

[0044] $$F(\beta) = \sum_{i=1}^{n} (y_i - \beta^T x_i)^2 + \lambda \|\beta\|_2^2$$

[0045] 수학식 1을 최소화하는 절차는 *리지 회귀*라고 불리우고; 오브젝티브 $F(\beta)$는 페널티 항(penalty term) $\lambda \| \beta \|_2^2$을 병합하는데, 이는 인색한 해답을 선호한다. 직관적으로, $\lambda=0$에 대해, 수학식 1을 최소화하는 것은 간단한 최소 자승(least squares) 문제를 해답하는 것에 대응한다. 양의 $\lambda>0$에 대해, 항 $\lambda \| \beta \|_2^2$은 높은 놈(norm)으로 해답에 페널티를 가하는데: 데이터를 동등하게 맞추는 두 개의 해답들 사이에서, 소수의(fewer) 큰 계수들을 갖는 하나가 선호될 수 있다. β의 계수들이, 입력이 출력에 어떻게 영향을 주는지에 대한 표시기들임을 회상하면, 이는 "Occam의 레이저(razor)"의 형태의 역할을 하는데: 소수의 큰 계수들을 갖는 더 간단한 해답들이 선호될 수 있다. 실제로, $\lambda>0$은 최소 자승 해답 기반보다는 새로운 입력들에 걸친 더 양호한 예측들을 실제로 제공한다. $y \in \mathbb{R}^n$이 출력들의 벡터이고, $x \in \mathbb{R}^{n \times d}$이 각 행에 하나씩 입력 벡터들을 포함하는 행렬이라고 하되; 즉

[0046] $$y = (y_i)_{i=1,...,n} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$$

[0047] 및

[0048] $$X = (x_i^T)_{i=1,\ldots,n} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} \end{pmatrix}$$ 라고 해본다.

[0049] 수학식 1의 최소화기는 선형 시스템인

수학식 2

[0050] $$A\beta = b$$

[0051] 을 해답함으로써 계산될 수 있는데, 여기서 $A=X^TX+\lambda I$이고, $b=X^Ty$이다. $\lambda>0$에 대해, 행렬 A는 대칭 양 정의(symmetric positive definite)이고, 효율적인 해답은 아래에 개요가 서술된 바와 같이 Cholesky 분해를 사용하여 확인될 수 있다.

[0052] B. Yao의 왜곡된 회로들

[0053] 그것의 기본적인 버전에서, Yao의 프로토콜(별칭은 왜곡된 회로들)은 반-정직 대항자들의 존재 시에 함수 $f(x_1;x_2)$의 두 개의-파티 평가를 허용한다. 프로토콜은 입력 소유자들 사이에서 실행된다(α_i는 사용자 i의 비공개 입력을 지칭한다). 프로토콜의 끝에서, $f(\alpha_1;\alpha_2)$의 값이 획득되지만, 어떤 파티도 이 출력 값으로부터 드러나는 것보다 더 많이 학습하지는 않는다.

[0054] 프로토콜은 다음과 같이 진행한다. *왜곡자(garbler)*라고 부르는 제1 파티는 f를 컴퓨팅하는 회로의 "왜곡된" 버전을 조성한다. 그러고 나서, 왜곡자는 평가자라고 부르는 제2 파티에, 왜곡된 회로, 및 α_1에 대응하는 왜곡된-회로 입력 값들(그리고 오로지 이러한 것들)을 제공한다. 표기법 $GI(\alpha_1)$는 이들 입력 값들을 지칭하도록 사용된다. 왜곡자는 또한 왜곡된-회로 출력 값들과 실제 비트 값들 사이의 맵핑을 제공한다. 회로를 수신할 때, 평가자는 비공개 입력 α_2에 대응하는 왜곡된-회로 입력 값들 $GI(\alpha_2)$을 불확정적으로 획득하기 위해 왜곡자를 통한 2-중(out-of)-1의 불확정 전송 프로토콜에 관여하고, 선택자의 역할을 한다. $GI(\alpha_1)$ 및 $GI(\alpha_2)$로부터, 평가자는 이에 따라 $f(\alpha_1;\alpha_2)$를 계산할 수 있다.

[0055] 좀더 상세하게, 프로토콜은 도 3에 도시된 바와 같이 불리언 회로(300)를 통해 함수 f를 평가한다. 회로의 w_i(310, 320)를 각각 와이어링하기 위해, 왜곡자는 비트-값들 $b_i=0$ 및 $b_i=1$에 각각 대응하는 두 개의 랜덤 크립토그래픽 키들인 $K_{w_i}^0$ 및 $K_{w_i}^1$을 연관시킨다. 다음으로, 입력 와이어들(w_i,w_j)(310, 320) 및 출력 와이어 w_k(330)을 갖는 각각의 이진 게이트 g(예컨대, OR-게이트)에 대해, 왜곡자는 4개의 사이퍼텍스트들(ciphertexts)을 컴퓨팅한다. bi,bj∈{0,1}에 대해, $\mathrm{Enc}_{(K_{w_i}^{b_i},K_{w_j}^{b_j})}(K_{w_k}^{g(b_i,b_j)})$ for $b_i,b_j\in\{0,1\}$ 이다. 이들 4개의 랜덤하게 순서가 배열된 사이퍼텍스트들의 집합은 왜곡된 게이트를 정의한다.

[0056] 키들의 쌍에 의해 키잉되는(keyed) 대칭 암호화 알고리즘 Enc는 선택된-평문 공격들 하에서 구분이 되지 않는 암호화를 갖는다는 것이 요구된다. 또한 키들의 쌍 $(K_{w_i}^{b_i}, K_{w_j}^{b_j})$을 가정하면, 대응하는 암호 해독 프로세스는 왜곡된 게이트를 구성하는 4개의 사이퍼텍스트로부터 $K_{w_k}^{g(b_i,b_j)}$의 값을 애매모호하지 않게 복구한다는 것이

요구된다. $(K_{w_i}^{b_i}, K_{w_j}^{b_j})$의 지식이 오로지 $K_{w_k}^{g(b_i,b_j)}$ 의 값만을 산출한다는 것과, 어떤 다른 출력 값들도 이 게이트를 위해 복구될 수 없다는 것에 주목할 만한 가치가 있다. 따라서, 평가자는, 중간의 컴퓨테이션들에 관한 어떤 추가적인 정보도 유출되지 않도록, 전체의 왜곡된 회로를 게이트 단위로 평가할 수 있다.

[0057] 하이브리드 접근법

[0058] 이러한 설정에서, 각 입력 및 각 출력 변수 $x_i,y_i,i\in[n]$이 비공개이고, 상이한 사용자에 의해 유지된다는 것을 상기해본다. 평가자(110)는 주어진 $\lambda>0$을 갖는 리지 회귀를 통해 획득되는 바와 같이, 입력 및 출력 변수들 사이의 선형 관계를 결정하는 β를 학습하기를 원한다.

[0059] 앞서 기재된 바와 같이, β를 획득하기 위해, 수학식 2에 정의된 바와 같이, 행렬 $A\in\mathbb{R}^{d\times d}$ 및 벡터 $b\in\mathbb{R}^{d}$를 요구한다. 이들 값들이 일단 획득되면, 평가자(110)는 수학식 2의 선형 시스템을 해답할 수 있고, β를 추출할 수 있다. 프라이버시-보존 방식으로 이러한 문제와 씨름하기 위한 여러 방식들이 존재한다. 하나는 예를 들어 비밀 공유에 의존할 수 있거나, 완전한 준동형 암호화에 의존할 수 있다. 현재, 이들 기술들은 현재 설정에 대해서는 적절하지 않은 것으로 보여지는데, 이는 이들이 상당한 (온-라인) 통신 또는 컴퓨테이션 오버헤드를 야기하기 때문이다. 결과적으로, Yao의 접근법은 앞서 개요가 서술된 바와 같이 탐구된다.

[0060] Yao의 접근법에 대한 한 가지 간단한 방식은, 행렬들 A 및 b를 컴퓨팅하고 후속적으로 $A\beta=b$를 해답하는 $i\in[n]$ 및 $\lambda>0$에 대한 입력들 x_i,y_i를 갖는 단일 회로를 설계하는 것이다. 이러한 접근법은 경매의 승자와 같은 다수의 사용자들로부터 유래하는 입력들의 간단한 함수들의 컴퓨테이션을 위해 과거에 사용되었다. (선형 시스템을 해답하는 회로를 설계하는 방법과 같은) 구현 사안들을 무시하기 때문에, 이러한 해답의 주요 결점은 결과적인 왜곡된 회로가 사용자들의 수 n과, 입력 변수들 및 β의 차원 d 모두에 의존한다는 것이다. 실제적인 애플리케이션들에서, n이 크고, 수백만명의 사용자의 순서로 있을 수 있다는 것이 공통적이다. 대조적으로, d는 상대적으로 작고, 수십(10s)의 순서에 있다. 이에 따라, 스케일링 가능한 해답을 얻기 위해 n의 왜곡된 회로의 의존성을 감소시키거나, 심지어 제거하는 것이 선호될 수 있다. 이를 위해, 아래에 논의된 바와 같이 문제가 재 공식화되었다.

[0061] A. 문제 재 공식화

[0062] 행렬 A 및 벡터 b가 아래와 같이 반복적인 방식으로 컴퓨팅될 수 있다는 것에 주목한다. 각 x_i 및 대응하는 y_i이 상이한 사용자들에 의해 유지된다는 것을 가정하면, 각 사용자 i는 행렬 $A_i = x_i x_i^T$ 및 벡터 $b_i=y_ix_i$를 국부적으로 컴퓨팅할 수 있다. 그러고 나서, 부분적인 기여를 합산하는 것이 다음을 산출한다는 것이 용이하게 검증된다:

수학식 3

[0063]
$$A=\sum_{i=1}^{n}A_i+\lambda I \quad \text{and} \quad b=\sum_{i=1}^{n}b_i$$

[0064] 수학식 3은 중요하게도 A 및 b가 일련의 덧셈들의 결과라는 것을 보여준다. 평가자의 회귀 과제는 이에 따라 두 개의 서브 과제들: (a) 행렬 A 및 벡터 b를 구축하기 위해 A_i의 것 및 b_i의 것을 수집하는 것과, (b) 선형 시스템인 수학식 2의 해답을 통해 β를 획득하기 위해 이것들을 사용하는 것으로 분리될 수 있다.

[0065] 물론 사용자들은 명문으로 평가자에게 그들의 국부적 셰어들(local shares) $(A_i;b_i)$을 전송할 수 없다. 하지만, 후자(latter)가 공개-키 덧셈(additive) 준동형 암호화 스킴을 사용하여 암호화되는 경우, 평가자(110)는 $(A_i;b_i)$의 것의 암호화로부터 A 및 b의 암호화를 재 구축할 수 있다. 나머지 도전은, β와는 다른 어떤 추가적인 정보를 {평가자(110) 또는 CSP(130)에게} 드러내지 않고도, CSP(130)의 도움으로 수학식 2를 해답하는 것이고; Yao의 왜곡된 회로들의 사용을 통해 그렇게 행하는 것에 대한 두 개의 구별된 방식들은 아래에 기재된다.

[0066] 보다 명백하게, $\mathfrak{E}_{pk} : (A_i;b_i) \in \mathcal{M} \mapsto c_i = \mathfrak{E}_{pk}(A_i;b_i)$ 가, 메시지 공간 $\mathcal{M}$에서의 쌍($A_i;b_i$)을 입력으로 취하고 pk,c_i 하에서 ($A_i;b_i$)의 암호화를 반환하는 공개 키 pk에 의해 색인화된 의미론적으로 안전한 암호화 스킴이라고 해본다. 그리고 나서, 일부 공개 이진 연산자에 대해 $\mathfrak{E}_{pk}(A_i;b_i) \otimes \mathfrak{E}_{pk}(A_j;b_j) = \mathfrak{E}_{pk}(A_i + A_j;b_i + b_j)$ 라는 것을 임의의 pk 및 임의의 두 개의 쌍 ($A_i;b_i$),($A_j;b_j$)을 위해 유지해야 한다. 이러한 암호화 스킴은 A_i 및 b_i의 엔트리들을 구성요소 단위로(component-wise) 암호화함으로써 임의의 의미론적으로 안전한 덧셈 준동형 암호화 스킴으로부터 구축될 수 있다. 예시들은 Regev의 스킴 및 Paillier의 스킴을 포함한다.

[0067] 프로토콜들은 이제 제공될 준비가 되어 있다. 하이-레벨 흐름도(400)가 도 4에 제공된다. 흐름도(400)는 준비 국면(410), 제1 국면(국면 1)(420), 및 제2 국면(국면 2)(430)을 포함한다. 사용자 셰어들을 종합하는 국면은 *국면 1*(420)으로 지칭되고, 그것이 수반하는 덧셈이 n에 선형으로 의존한다는 것에 주목한다. A 및 b의 암호화된 값들로부터 수학식 2에 대한 해답을 컴퓨팅하는 것에 이르는 후속적인 국면은 *국면 2*(430)로 지칭된다. 국면 2(430)가 n에 대한 의존성을 갖지 않는다는 점에 주목한다. 이들 국면들은 특정 프로토콜들과 연계되어 아래에 논의될 것이다. 그것은 시스템 Aβ=b을 해답할 수 있는 회로의 존재 하에서 가정되고; 이러한 회로가 효율적으로 구현될 수 있는 방법이 본 명세서에 논의된다는 점에 주목한다.

[0068] B. 제1 프로토콜

[0069] 제1 프로토콜의 연산의 하이 레벨 도면(500)은 도 5에서 확인될 수 있다. 제1 프로토콜은 다음과 같이 연산한다. 앞서 열거된 바와 같이, 제1 프로토콜은 3개의 국면들: 준비 국면(510), 국면 1(520), 및 국면 2(530)를 포함한다. 명백하듯이, 오로지 국면 2(530)만이 실제로 온-라인 처리(treatment)를 요구한다.

[0070] *준비 국면*(510). 평가자(110)는 입력 변수들의 차원(즉, 파라미터 d) 및 이들 값의 범위와 같은 사양서들을 CSP(130)에 제공한다. CSP(130)는 국면 2(530)에서 기재된 회로를 위해 Yao 왜곡된 회로를 준비하고, 왜곡된 회로가 평가자(110)에게 이용 가능하게 한다. CSP(130)는 또한 준동형 암호화 스킴 $\mathfrak{E}$에 대한 공개 키 pk_{csp} 및 비공개 키 sk_{csp}를 생성하지만, 평가자(110)는 (준동형일 필요가 없는) 암호화 스킴 $\mathcal{E}$에 대한 공개 키 pk_{ev} 및 비공개 키 sk_{ev}를 생성한다.

[0071] *국면 1*(520). 각 사용자 i는 그의 부분적 행렬 A_i 및 벡터 b_i를 국부적으로 컴퓨팅한다. 이들 값들은 그러고 나서 CSP(130)의 공개 암호화 키 pk_{csp} 하에서 덧셈 준동형 암호화 스킴 $\mathfrak{E}$을 사용하여 암호화되고; 즉 $c_i = \mathfrak{E}_{pk_{csp}}(A_i;b_i)$ 이다.

[0072] CSP(130)이 이 값에 액세스를 얻지 못하게 하기 위해, 사용자 i는 평가자(110)의 공개 암호화 키 pk_{ev} 하에서 c_i의 값을 재-암호화(super-encrypt)하는데; 즉 $C_i = \mathcal{E}_{pk_{ev}}(c_i)$ 이고, 평가자(110)에게 C_i를 전송한다.

[0073] 평가자(110)는 $c_\lambda = \mathfrak{E}_{pk_{csp}}(\lambda I;0)$ 를 컴퓨팅한다. 그것은 후속적으로 모든 수신된 C_i의 것을 수집하고, c_i의 것을 복구하기 위해 그것의 비공개 암호 해독 키 sk_{ev}를 사용하여 그것들을 암호 해독하는데; 즉

[0074] $1 \le i \le n$에 대해, $c_i = \mathcal{D}_{sk_{ev}}(C_i)$ 이다.

[0075] 그러고 나서 그렇게-획득된 값들을 종합하고, 다음을 얻는다:

수학식 4

[0076] $$c = \left(\bigotimes_{i=1}^{n} c_i\right) \otimes c_\lambda = \mathcal{E}_{pk_{csp}}\left(\sum_{i=1}^{n} A_i + \lambda I; \sum_{i=1}^{n} b_i\right)$$
$$= \mathcal{E}_{pk_{csp}}(A; b)$$

[0077] *국면 2*(530), 준비 국면(510)에서 CSP(130)에 의해 제공된 왜곡된 회로는 입력으로서 GI(c)를 취하는 회로의 왜곡(garbling)이고, 다음의 두 개의 단계를 수행한다:

[0078] 1) A 및 b를 복구하기 위해 sk_{csp}로 c를 암호 해독하는 단계(여기서 sk_{csp}는 왜곡된 회로에 내장됨); 및

[0079] 2) 수학식 2를 해답하고 β를 반환하는 단계.

[0080] 이 국면 2(530)에서, 평가자(110)는 단지 c에 대응하는 왜곡된-회로 입력 값들; 즉, GI(c)를 획득할 필요가 있다. 이들은 평가자(110)와 CSP(130) 사이의 표준 불확정 전송(OT: Oblivious Transfer)을 사용하여 획득된다.

[0081] 앞의 하이브리드 컴퓨테이션은 왜곡된 회로 내에서 암호화된 입력들의 암호 해독을 수행한다. 이것은 부담이 클 수 있으므로, 예를 들어 $\mathcal{E}$에 대한 조성 블록으로서 Regev 준동형 암호화 스킴을 사용하는 것이 제안되는데, 이는 Regev 스킴이 매우 간단한 암호 해독 회로를 갖기 때문이다.

[0082] C. 제2 프로토콜

[0083] 제2 프로토콜의 연산의 하이 레벨 도면(600)은 도 6에서 확인될 수 있다. 제2 프로토콜은 랜덤 마스크들을 사용하여 왜곡된 회로 내에서의 암호 해독(A;b)을 회피하는 수정을 제시한다. 국면 1(610)은 동일하게 폭넓게 유지된다. 따라서 국면 2는 하이라이팅될 것이다(그리고 대응하는 준비 국면). 사상은 추가적인 마스크로 입력들을 모호하게 하는 준동형 특성을 활용하는 것이다. (μ_A; μ_B)가 $\mathcal{M}$(즉, 준동형 암호화 $\mathcal{E}$의 메시지 공간)에서의 한 요소를 지칭하는 경우, 수학식 4로부터,

[0084] $c \otimes \mathcal{E}_{pk_{csp}}(\mu_A; \mu_b) = \mathcal{E}_{pk_{csp}}(A + \mu_A; b + \mu_b)$ 이 뒤따른다는 것에 주목한다. 따라서 평가자(110)가 $\mathcal{M}$에서의 랜덤 마스크 (μ_A; μ_B)를 선택하고, 전술한 바와 같이 c를 모호하게 하고, 결과 값을 CSP(130)에 전송하는 것을 가정한다. 그러고 나서, CSP(130)는 그것의 암호 해독 키를 적용하고 마스킹된 값들

[0085] $\hat{A} = A + \mu_A$ 및 $\hat{b} = b + \mu_b$을 복구할 수 있다. 결과로서, 암호 해독이 마스크의 제거로 대체되는 이전 섹션의 프토토콜을 적용할 수 있다. 좀 더 상세하게, 다음을 수반한다:

[0086] *준비 국면*(610): 이전과 같이, 평가자(110)는 평가를 설정한다. 평가자(110)는 그것의 평가를 지원하는 왜곡된 회로를 조성하기 위해 사양서들을 CSP(130)에 제공한다. CSP(130)는 회로를 준비하고, 그것을 평가자(110)에게 이용 가능하게 하고, 공기 및 비공개 키들을 모두 생성한다. 평가자(110)는 랜덤 마스크 (μ_A; μ_B)$\in \mathcal{M}$를 선택하고, (μ_A; μ_B)에 대응하는 왜곡된-회로 입력 값들; 즉 GI μ_A; μ_B)를 얻기 위해 CSP(130)와 함께 불확정 전송(OT) 프로토콜에 참여한다.

[0087] *국면 1*(620): 이는 제1 프로토콜과 유사하다. 추가로, 평가자(110)는

[0088] $\hat{c} = c \otimes \mathcal{E}_{pk_{csp}}(\mu_A; \mu_b)$ 로서 c를 마스킹한다.

[0089] *국면 2*(630). 평가자(110)는 $\hat{c}$를 CSP(130)에 전송하고, CSP(130)는 명문으로 $(\hat{A}; \hat{b})$을 획득하기 위해 그것을 암호 해독한다. CSP(130)는 그러고 나서 왜곡된 입력 값들 GI $(\hat{A}; \hat{b})$을 다시 평가자(110)에게

전송한다. 준비 국면에서 CSP(130)에 의해 제공된 왜곡된 회로는, GI $(\hat{A};\hat{b})$ 및 GI($\mu_A;\mu_B$) 입력으로 취하고 다음의 두 단계들을 수행하는 회로의 왜곡이다:

[0090] 1) A 및 b를 복구하기 위해 $(\hat{A};\hat{b})$ 로부터 마스크 ($\mu_A;\mu_B$)를 빼는 단계; 및

[0091] 2) 수학식 2를 해답하고 β를 반환하는 단계.

[0092] 왜곡된 회로 뿐만 아니라 ($\mu_A;\mu_B$)에 대응하는 왜곡된-회로 입력 값들 GI $(\hat{A};\hat{b})$ 은 준비 국면(610) 동안 획득되었다. 이 국면에서, 평가자(110)는 $(\hat{A};\hat{b})$ 에 대응하는 왜곡된 회로 입력 값들 GI $(\hat{A};\hat{b})$ 를 단지 CSP(130)로부터 수신할 필요가 있다. 이 국면에서 불확정 전송(OT)이 존재하지 않다는 점에 주목한다.

[0093] 이러한 제2 실현에 대해, 암호 해독은 회로의 부분으로서 실행되지 않는다. 따라서 회로로서 효율적으로 구현될 수 있는 준동형 암호화 스킴을 선택하는 것으로 제한되지 않는다. Regev의 스킴 대신에, Paillier의 스킴이나, 또는 $\mathfrak{E}$ 에 대한 조성 블록으로서 Damgård 및 Jurik에 의한 그것의 일반화를 사용하는 것이 제안된다. 이들 스킴들은 Regev보다 더 짧은 사이퍼텍스트 확장을 갖고, 보다 더 작은 키들을 요구한다.

[0094] D. 제3 프로토콜

[0095] 일부 애플리케이션들에 대해, 관련된 사상은 준동형 암호화 스킴이 부분적 준동형 특성만을 가질 때 적용된다. 이러한 개념은 다음 정의에서 명백해진다.

[0096] *정의 1: 부분적 준동형 암호화 스킴*은, 비공개 암호화 키를 필요로 하지 않고도 암호화된 평문에 상수를 더하는 것(부분적 준동형이 덧셈인 경우)이나 곱하는 것(부분적 준동형이 곱셈인 경우)이 가능하게 하기 위한 암호화 스킴이다.

[0097] 여기 일부 예시들이 있다.

[0098] ● $\mathbb{F}_p$ 를 프라임 필드로 지칭하고, G=<g>를 g에 의해 생성된 곱셈 그룹 $\mathbb{F}_p^*$ 의 순환하는 서브 그룹이라고 해본다. q를 G의 순서(order)라고 해본다. 분명한(plain) ElGamal 암호화에 대해, 메시지 공간은 $\mathcal{M}$=G이다. 공개 암호화 키는 $y=g^x$이지만, 비공개 키는 x이다. $\mathcal{M}$ 에서의 공간 m의 암호화는 (R;c)로 주어지는데, 여기서 일부 랜덤 $r\in\mathbb{Z}/q\mathbb{Z}$ 에 대해 $R=g^r$이고, $c=my^r$이다. 평문 m은 이후 비밀 키 x를 사용하여 $m=c/R^x$로서 복구된다.

[0099] - 전술한 시스템은 $\mathbb{F}_p^*$ 에 있어서 곱셈에 대해 부분적 준동형이다: 임의의 상수 $K\in\mathcal{M}$ 에 대해, C'=(R;Kc)는 메시지 m'=Km의 암호화이다.

[0100] ● 소위 해싱된(hashed) ElGamal 크립토시스템은 추가로 해시 함수 H를 요구하고, 일부 파라미터 k에 대해 그룹 요소들을 G로부터 $\mathbb{F}_2^k$ 으로 맵핑한다. 메시지 공간은 $\mathcal{M}=\mathbb{F}_2^k$ 이다. 키 생성은 분명한(plain) ElGamal에 대한 것이다. 메시지 $m\in\mathcal{M}$ 의 암호화는 (R;c)로 주어지는데, 여기서 일부 랜덤 $r\in\mathbb{Z}/q\mathbb{Z}$ 에 대해, $R=g^r$이고, $c=m+H(y^r)$이다. 평문 m은 이후 비밀 키를 사용하여 $m=c+H(R^x)$로서 복구된다. '+'는 $\mathbb{F}_2^k$ 에서의 덧셈에 대응한다는 것(즉, 그것은 등가적으로 k-비트 스트링에서의 XOR로서 확인될 수 있음)에 주목한다.

[0101] - 전술한 시스템은 XOR에 대해 부분적 준동형이다: 임의의 상수 $K\in\mathcal{M}$ 에 대해, C'=(R;K+c)는 메시지 m'=K+m의 암호화이다.

[0102] 비-제한적인 예시를 위해, 이제 c는 부분적 준동형 암호화 스킴, 다시 말해 $\mathfrak{E}$ 하에서의 (A;b)의 암호화임을 가

정해보는데, (μ_A;μ_B)가 $\mathcal{M}$(즉, 부분적 준동형암호화 $\mathcal{E}$의 메시지 공간)에서의 요소를 지칭하는 경우, 일부 연산자 $\oplus$에 대해, 수학식 4로부터 다음이 뒤따른다:

[0103] $c \oplus \mathcal{E}_{pk_{csp}}(\mu_A;\mu_b) = \mathcal{E}_{pk_{csp}}(A+\mu_A;b+\mu_b)$. {전술한 기재에서, 준동형은 덧셈으로 주목되고; 동일한 것은 곱셈으로 쓰여진 준동형에 대해 참(true)으로 유지된다.}

[0104] 이에 따라, 평가자(110)는 $\mathcal{M}$에서의 랜덤 마스크 (μ_A;μ_B)를 선택하고, 전술한 바와 같이 c를 모호하게 하고, 결과 값을 CSP(130)에 전송한다는 것을 가정한다. 그러고 나서, CSP(130)는 그것의 암호 해독 키를 적용하고, 마스킹된 값들 $\hat{A}=A+\mu_A$ 및 $\hat{b}=b+\mu_b$을 복구할 수 있다. 결과로서, 이전 섹션의 프로토콜은 암호 해독이 마스크의 제거로 대체되는 경우에 적용될 수 있다.

[0105] 최종적으로, 제2 또는 제3 프로토콜에 대해 마스크를 사용하는 트릭은 리지 회귀의 경우로 제한되지 않는다는 점에 주목한다. 그것은 왜곡된 회로를 통해 준동형 암호화(즉, 부분적 준동형 암호화)를 하이브리드 방식으로 결합하는 임의의 애플리케이션에서 사용될 수 있다.

[0106] E. 논의

[0107] 제안된 프로토콜들은, 이들을 효율적이게 하고 실제-세계 시나리오들에서 실용적이게 하는 여러 강점들을 갖는다. 첫째로, 프로세스 동안 사용자들이 온-라인을 유지해야할 필요가 없다. 국면 1(420)은 증가적(incremental)이기 때문에, 각 사용자는 이들의 암호화된 입력들을 제출하고, 시스템을 떠날 수 있다.

[0108] 게다가, 시스템(100)은 리지 회귀를 여러번 수행하는 것에 용이하게 적용될 수 있다. 평가자(110)가 ℓ 추정(estimations)을 수행하기를 원한다고 가정하면, 그것은 준비 국면(410) 동안 CSP(130)로부터 ℓ 왜곡된 회로들을 검색할 수 있다. 다수의 추정들은 새로운 사용자들(120)의 도착을 수용하도록 사용될 수 있다. 특히, 공개 키들이 오래가기 때문에, 이들은 너무 자주 리프레쉬될 필요가 없는데, 이는 새로운 사용자들이 보다 많은 쌍들 (A_i;b_i)을 평가자(110)에게 제출할 때, 평가자(110)가 이것들을 이전 값들로 합산하고, 갱신된 β를 컴퓨팅할 수 있다는 것을 의미한다. 이러한 프로세스가 새로운 왜곡된 회로를 사용하는 것을 요구할지라도, 입력들을 이미 제출한 사용자들은 그것들을 다시 제출할 필요가 없다.

[0109] 최종적으로, 요구된 통신의 양은 비밀 공유 스킴에서보다 상당히 더 작고, 오로지 평가자(110) 및 CSP(130)만이 불확정 전송(OT)을 사용해 통신한다. 또한, 국면 1(420)에서 공개 키 암호화 스킴 $\mathcal{E}$을 사용하는 것보다 오히려, 사용자들이, 예컨대 SSL과 같은 평가자(110)와의 안전한 통신을 확립하기 위한 임의의 수단을 사용할 수 있다는 것에 주목한다.

[0110] F. 추가적인 최적화들

[0111] 행렬 A가 $\mathbb{R}^{d\times d}$에 있고 벡터 b가 $\mathbb{R}^{d}$에 있다는 것을 상기해본다. 이에 따라, k가 실수들을 인코딩하도록 사용된 비트-사이즈를 지칭한다고 하면, 행렬 A 및 벡터 b는 이들의 표현을 위해 d^2k 비트들 및 dk 비트들을 각각 필요로 한다. 제2 프로토콜은 $\mathcal{M}$에서의 랜덤 마스크 (μ_A;μ_B)를 요구한다. 준동형 암호화 스킴 $\mathcal{E}$이, 모든 A의 엔트리와 b의 엔트리가 개별적으로 Paillier 암호화되는 Paillier의 위에 조성되었다는 것을 가정해본다. 이러한 경우, $\mathcal{E}$의 메시지 공간 $\mathcal{M}$은 일부 RSA 모듈러스 N에 대해, $\mathbb{Z}_N$에서의 (d^2+d) 요소들로 이루어진다. 하지만, 이들 요소들이 k-비트 값들이기 때문에, 전체 범위 $\mathbb{Z}_N$에서의 대응하는 마스킹 값들을 가져올 필요가 없다. 일부 (상대적으로 짧은) 보안 길이 l에 대한 임의의 (k+1) 비트 값들은 이들이 통계학적으로 대응하는 엔트리를 숨기는 한 수행할 것이다. 실제로, 이는 준비 국면에서의 소수의 불확정 전송을 야기하고 보다 더 작은 왜곡된 회로를 야기한다.

[0112] 효율성을 개선시키기 위한 또 다른 방식은 표준 배칭(batching) 기술, 즉 A 및 b의 다수의 평문 엔트리들을 단일 Paillier 사이퍼텍스트로 팩킹(packing)하는 것을 통해서이다. 예를 들어, 평문 값들을 단일 Paillier 사이

퍼텍스트(충분히 많은 0으로 분리됨)로 팩킹(20)하는 것은 20의 인수(factor)만큼 국면 1의 실행 시간을 감소시킬 것이다.

[0113] 구현

[0114] 프라이버시-보존 시스템의 실용성을 평가(assess)하기 위해, 본 시스템은 종합적 및 실제 데이터 집합들 모두에 대해 구현 및 테스트되었다. 앞서 제안된 제2 프로토콜이 구현되었고, 이는 왜곡된 회로 내에서 암호 해독을 요구하지 않고, (단지 합산만을 수반하는) 국면 1을 위해 효율적인 준동형 암호화의 사용을 허용한다.

[0115] A. 국면 1 구현

[0116] 앞서 논의된 바와 같이, 준동형 암호화에 대해, Paillier의 스킴은 80-비트들 보안 레벨에 대응하는 1024 비트들 길이 모듈러스를 사용하였다. 국면 1을 진전시키기 위해, 앞서 개요가 서술된 바와 같이 배칭이 또한 구현되었다. 입력들을 기여하는 n명의 사용자들을 고려해볼 때, 1024 비트들의 하나의 Paillier 사이퍼텍스트로 배칭될 수 있는 요소들의 수는 $1024=b(b+\log_2 n)$이고, 여기서 b는 수들을 표현하기 위한 비트들의 총 수이다. 나중에 논의되겠지만, b는 원하는 정확성의 함수로서 결정되어, 이에 따라 이러한 실험에서 15 및 30 사이에서의 요소들이 배칭되었다.

[0117] B. 회로 왜곡 프레임워크

[0118] 본 시스템은 FastGC 이외에, 개발자들이 기초적인 XOR, OR 및 AND 게이트들을 사용하여 임의적인 회로들을 정의하는 것을 가능하게 하는 Java-기반의 개방-소스 프레임워크 하에서 조성되었다. 일단 회로들이 구축되면, 프레임워크는 왜곡, 불확정 전송 및 왜곡된 회로의 완전한 평가를 다룬다. FastGC는 여러 최적화들을 포함한다. 첫째로, 회로에서의 XOR 게이트들에 대한 통신 및 컴퓨테이션 비용은 "프리(free) XOR" 기술을 사용하여 현저하게 감소된다. 둘째로, 왜곡된-행 감소 기술을 사용하여, FastGC는 $1=2^k$만큼 k-팬(fan)-인(in) 비-XOR 게이트들에 대한 통신 비용을 감소시키고, 이는 25%의 통신 절약을 제공하는데, 이는 오로지 2-팬-인 게이트들만이 프레임워크에서 정의되기 때문이다. 셋째로, FastGC는 추가적인 OT당 여러 대칭-키 연산들 및 k OT들의 비용으로 실질적으로 제한된 수의 전송을 실행할 수 있는 OT 확장을 구현한다. 최종적으로, 마지막 최적화는 간결한 "3 비트들의 덧셈" 회로인데, 이는 4개의 XOR 게이트(이들 모두는 통신 및 컴퓨테이션에 있어서 "프리"임) 및 단 하나의 AND 게이트를 갖는 회로를 정의한다. FastGC는 왜곡 및 평가가 동시에 발생하는 것을 가능하게 한다. 보다 구체적으로, CSP(130)는 평가자(110)에게 왜곡된 테이블들을 송신하는데, 이들은 회로 구조에 의해 정의된 순서로 산출되기 때문이다. 평가자(110)는 그러고 나서 이용 가능한 출력 값들 및 테이블들에 기초하여 다음에 어떤 게이트를 평가할지를 결정한다. 일단 게이트가 평가되었다면, 그것의 대응하는 테이블은 즉시 폐기된다. 이는 모든 왜곡된 회로들을 오프-라인으로 사전-컴퓨팅하는 것으로서 동일한 컴퓨테이션 및 통신에 이르게 하지만, 메모리 소비를 일정하게(constant) 한다.

[0119] C. 회로에서의 선형 시스템을 해답

[0120] 본 접근법의 주요 도전들 중 하나는 수학식 2에 정의된 바와 같이, 선형 시스템 $A\beta=b$를 해답하는 회로를 설계하는 것이다. 함수를 왜곡된 회로로서 구현할 때, 데이터-불가지론적(data-agnostic)인 연산들을 사용하는 것이 바람직할 수 있는데, 즉 이것의 실행 경로는 입력에 의존하지 않는다. 예를 들어, 입력들이 왜곡될 때, 평가자(110)는 if-then-else 문장의 모든 가능한 경로들을 실행할 필요가 있는데, 이는 내포된(nested) 조건문들의 존재 시 회로 사이즈와 실행 시간 모두의 기하급수적인 성장을 야기한다. 이는, 예컨대 가우시안 소거(Gaussian elimination)와 같은 피벗팅(pivoting)을 요구하는 선형 시스템들을 해답하기 위한 종래의 알고리즘들 중 실질적이지 않은 임의의 것을 렌더링한다.

[0121] 간단함을 위해, 이 시스템은 아래에 제시된 표준 Cholesky 알고리즘을 구현했다. 하지만, 그것의 복잡성은 유사한 기술들을 사용하여 블록 단위 반전(inversion)으로서 동일한 복잡성으로 더 감소될 수 있다.

[0122] 선형 시스템들을 해답하기 위한 여러 가능한 분해 방법들이 존재한다. Cholesky 분해는 행렬 A가 대칭 양 정의일 때에만 적용 가능한 선형 시스템을 해답하기 위한 데이터-불가지론적 방법이다. Cholesky의 주요 장점은 그것이 피벗팅을 위한 필요 없이 수치적으로 강력하다는 점이다. 특히, 고정 소수점 표현을 위해 매우 적절하다.

[0123] $\Lambda = \lambda I + \sum_i^n x_i x_i^T$ 가 실제로 $\lambda>0$에 대해 양 정의 행렬이기 때문에, Cholesky는 이 구현에서 $A\beta=b$를 해답하는 방법으로서 선택되었다.

[0124] Cholesky 분해의 주요 단계들은 아래에 간략하게 개요가 서술된다. 알고리즘은 $A=L^{T}L$이기 위한 낮은(lower)-삼각형 행렬 L을 구축하고: 시스템 Aβ=b을 해답하는 것은 다음의 두 개의 시스템들을 해답하는 것으로 감소된다:

[0125] $L^{T}y=b$; 및

[0126] Lβ=y. 행렬들 L 및 LT가 삼각형이기 때문에, 이들 시스템들은 대입(substitution)을 다시 사용하여 용이하게 해답될 수 있다. 또한, 행렬 A가

[0127] 양 정의이기 때문에, 행렬 L은 반드시 대각선 상에 영이 아닌 값들을 가지며, 이에 따라 어떤 피벗팅도 필요없다.

[0128] 분해 $A=L^{T}L$은 도 7에 도시된 알고리즘 1에 기재된다. 이는 $\Theta(d^3)$ 덧셈들, $\Theta(d^3)$ 곱셈들, $\Theta(d^2)$ 나눗셈들 및 $\Theta(d)$ 제곱근 연산들을 수반한다. 또한, 두 개의 백워드 소거(backwards elimination)를 통한 위의 두 시스템들의 해답은 $\Theta(d^2)$ 덧셈들, $\Theta(d^2)$ 곱셈들 및 $\Theta(d)$ 나눗셈들을 수반한다. 이들 연산들의 구현은 아래에 논의된다.

[0129] D. 실수들의 표현

[0130] 선형 시스템 수학식 2를 해답하기 위해, 이진 형태의 실수들을 정확하게 표현할 필요가 있다. 실수를 표현하기 위한 두 개의 가능한 접근법이 고려되었다: 부동 소숫점 및 고정 소숫점. 실수 α의 부동 소숫점 표현은 공식으로 주어진다:

[0131] [a]=[m;p]; 여기서 $a \approx 1.m \cdot 2^{p}$ 이다.

[0132] 부동 소숫점 표현은 실제로 임의적인 크기의 수를 수용한다는 장점이 있다. 하지만, 덧셈과 같은 부동 소숫점 표현에 대한 기초적인 연산들은 데이터-불가지론적인 방식으로 구현하기에 어렵다. 중요하게도, Cholesky를 사용하는 것은 구현하기에 훨씬 더 간단한 고정 소숫점 표현을 사용하는 것을 보장한다. 실수 α를 고려해보면, 그것의 고정 소숫점 표현은 $[a] = \lfloor a \cdot 2^{p} \rfloor$로 주어지고, 여기서 지수 p는 고정된다.

[0133] 본 명세서에서 논의된 바와 같이, 수행되도록 요구된 많은 연산들은 고정 소숫점 수들에 걸쳐 데이터-불가지론적 방식으로 구현될 수 있다. 또한, 고정 소숫점 표현을 위해 생성된 회로들은 훨씬 더 작다. 또한, β의 계수들이 필적하다는 것을 보장하기 위해, 그리고 수치적인 안정성을 위해, 리지 회귀의 입력 변수들 xi가 전형적으로 동일한 도메인에서 (-1과 1 사이에서) 재 스케일링된다는 점을 상기해본다. 이러한 설정에서, Cholesky 분해가 오버플로우를 야기하지 않고도 고정 소숫점 수들을 갖는 A에 대해 수행될 수 있다. 또한, 행렬 A의 조건 수 및 y_i에 대한 경계들(bounds)을 고려할 때, 오버플로우를 방지하기 위해 필요한 비트들은 본 방법에서의 마지막 두 개의 삼각형 시스템들을 해답하는 동안 컴퓨팅될 수 있다. 따라서, 본 시스템은 고정 소숫점 표현을 사용하여 구현되었다. 소수 부분(fractional part)의 비트들 p의 수는 시스템 파라미터로서 선택될 수 있으며, 시스템의 정확성과 생성된 회로들의 사이즈 사이의 트레이드-오프를 생성한다. 하지만, p를 선택하는 것은 원하는 정확성에 기초하여 원리적인 방식으로 수행될 수 있다. 음수들은 표준 2(two)의 보수 표현을 사용하여 표현된다.

[0134] 본 명세서에 개시된 다양한 실시예들은 하드웨어, 펌웨어, 소프트웨어, 또는 이들의 임의의 결합으로서 구현될 수 있다. 또한, 소프트웨어는 프로그램 저장 유닛 또는 컴퓨터 판독 가능한 매체 상에서 실제로 구체화된 애플리케이션 프로그램으로서 바람직하게 구현된다. 애플리케이션 프로그램은, 임의의 적절한 구조를 포함하는 기계로 업로딩되거나, 그러한 기계에 의해 실행될 수도 있다. 바람직하게, 그러한 기계는 하나 이상의 중앙 처리 유닛("CPU"), 메모리, 및 입력/출력 인터페이스와 같은 하드웨어를 갖는 컴퓨터 플랫폼상에서 구현된다. 컴퓨터 플랫폼은 또한 운영 체제 및 마이크로명령어(microinstruction) 코드를 포함할 수도 있다. 본 명세서에 기술된 다양한 프로세스 및 기능은 CPU에 의해 실행될 수 있는 마이크로명령어 코드의 부분, 또는 애플리케이션 프로그램의 부분일 수도 있으며, 또는 그들의 임의의 조합일 수도 있는데, 이는 이러한 컴퓨터 또는 프로세서가 명백하게 도시되었는지의 여부와는 상관없다. 또한, 추가적인 데이터 저장 유닛 및 프린팅 유닛과 같은 다양한 다른 주변 유닛이 컴퓨터 플랫폼으로 연결될 수도 있다.

[0135] 본 명세서에 상술된 모든 예시들 및 조건부 언어는, 발명자에 의해 종래의 기술을 진전시키는데 기여되는 실시예들의 원리들 및 개념들을 독자가 이해하는 것에 도움을 줄 교육적인 목적을 위해 의도되었으며, 이러한 구체적으로 상술된 예시들 및 조건들에 대해 제한하지 않는 것으로 해석될 것이다. 또한, 본 발명의 원리들, 양상들, 및 다양한 실시예들뿐만 아니라 이들의 특정 예시들을 상술하는 본 명세서의 모든 문장들은 이들의 구조적 및 기능적 등가물을 모두 포함하는 것으로 의도된다. 추가적으로, 이러한 등가물들은 현재 알려진 등가물들뿐만 아니라 미래에 개발될 등가물들, 즉 구조에 상관없이 동일한 기능을 수행하는 임의의 개발될 요소들도 포함하는 것이 의도된다.

부호의 설명

[0136] 110: 평가자 130: 크립토 서비스 제공자(CSP)

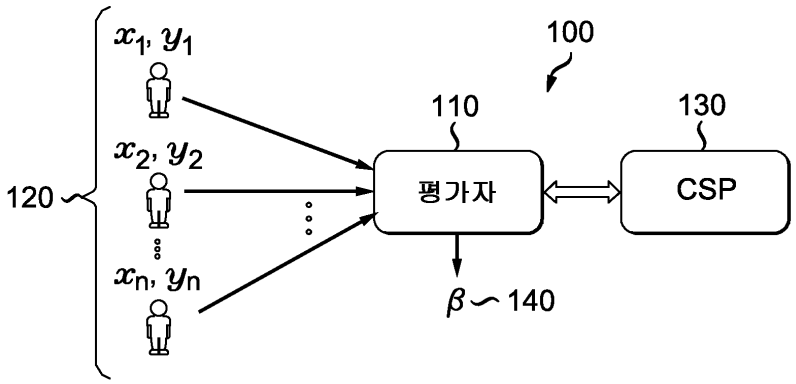
200: 서버 210: 프로세서

220: 메모리 230: 저장소

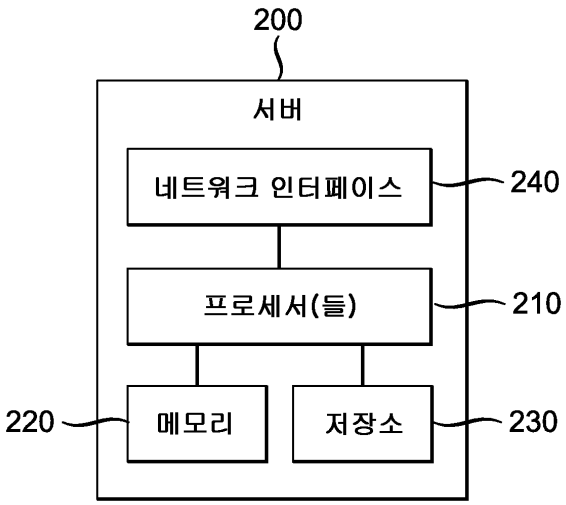
240: 네트워크 인터페이스

도면

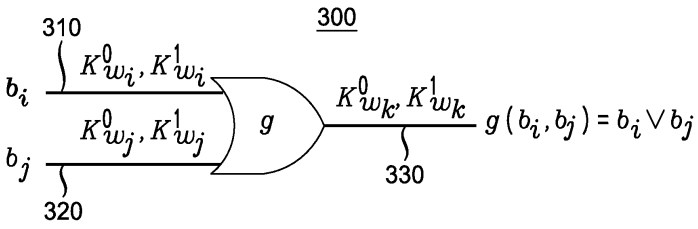
도면1



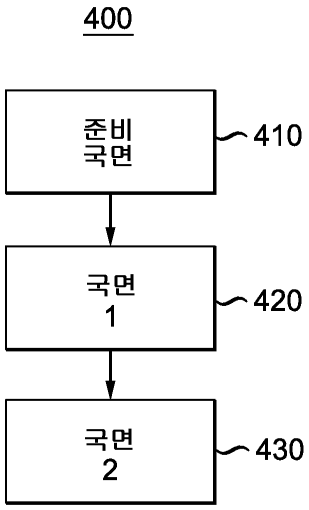
도면2



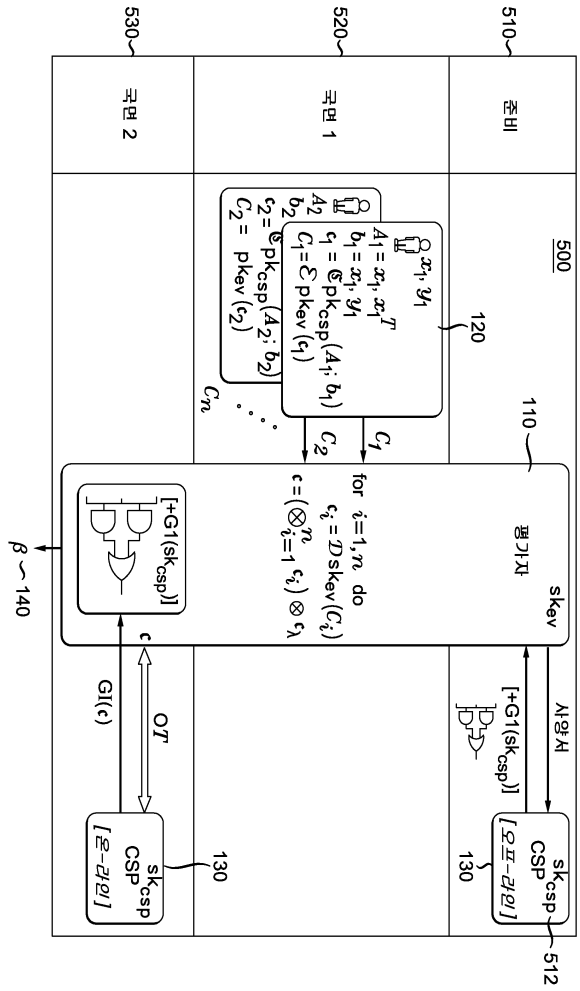
도면3


300
310
b_i
$K^0_{w_i}, K^1_{w_i}$
$K^0_{w_j}, K^1_{w_j}$
b_j
320
g
$K^0_{w_k}, K^1_{w_k}$
330
$g(b_i, b_j) = b_i \vee b_j$

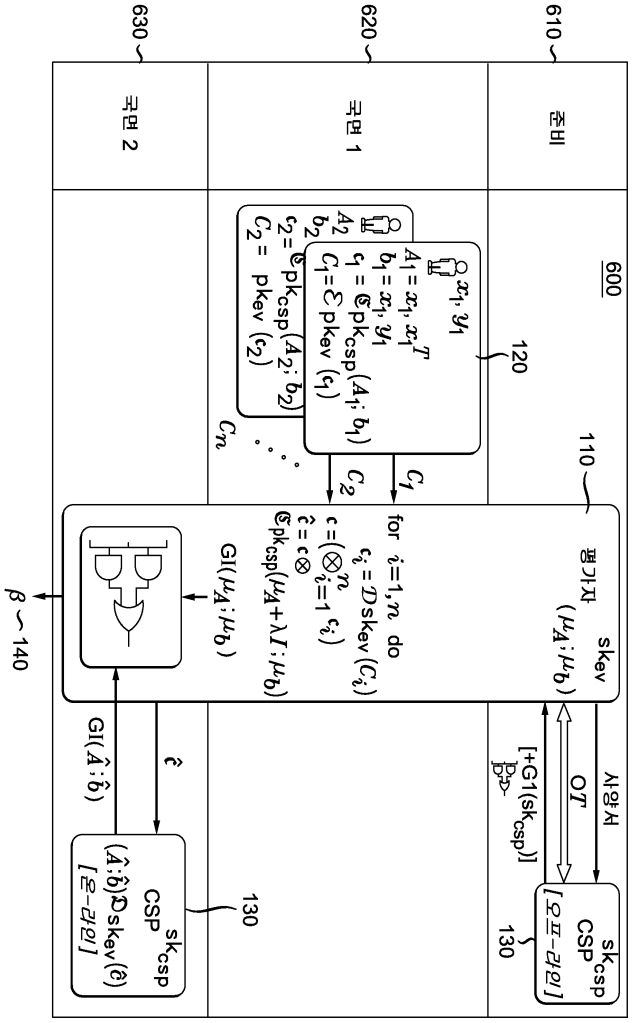
도면4


400
준비 국면
410
국면 1
420
국면 2
430

도면5


510
520
530
준비
국면 1
국면 2
500
110
120
130
512
140
평가자
sk_{ev}
x_1, y_1
$A_1 = x_1, x_1^T$
$b_1 = x_1, y_1$
$c_1 = \mathcal{E}pk_{csp}(A_1; b_1)$
$C_1 = \mathcal{E}pk_{ev}(c_1)$
A_2
b_2
$c_2 = \mathcal{E}pk_{csp}(A_2; b_2)$
$C_2 = pk_{ev}(c_2)$
C_n
C_1
C_2
for $i=1,n$ do
$c_i = \mathcal{D}sk_{ev}(C_i)$
$c = (\otimes_{i=1}^{n} c_i) \otimes c_\lambda$
[+G1(sk_{csp})]
사양서
sk_{csp}
CSP
[오프-라인]
[+G1(sk_{csp})]
GI(c)
OT
sk_{csp}
CSP
[온-라인]
β

도면6



도면7

알고리즘 1 $A \in \mathbb{R}^{d \times d}$ 의 Cholesky 분해
Input: $A = (a_{ij})_{i,j \in \{1,\ldots,d\}}$
Output: L, s.t., $A = L^T L$
1: **for** j = 1, d **do**
2: **for** k = 1, $j - 1$ **do**
3: **for** $i = j$, d **do**
4: $a_{ij} -= a_{ik} a_{jk}$
5: $a_{jj} = \sqrt{a_{jj}}$
6: **for** $k = j + 1$, d **do**
7: $akj \ /= \ ajj$
8: **return** $L := A$