(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2021년09월30일
(11) 등록번호 10-2307111
(24) 등록일자 2021년09월24일

(51) 국제특허분류(Int. Cl.)
G06F 21/60 (2013.01) *G06F 17/16* (2006.01)
H04L 9/08 (2006.01)
(52) CPC특허분류
G06F 21/602 (2013.01)
G06F 17/16 (2013.01)
(21) 출원번호 10-2020-0127971
(22) 출원일자 2020년10월05일
심사청구일자 2020년10월05일
(56) 선행기술조사문헌
Mitsuru Ambai 외 3인, 'TERNARY WEIGHT DECOMPOSITION AND BINARY ACTIVATION ENCODING FOR FAST AND COMPACT NEURAL NETWORK', Under review as a conference paper at ICLR 2017.
Dmitry Efanov 'Ternary Sum Codes', 2020 IEEE East-West Design & Test Symposium (EWDTS), 2020.09.04.
subhash kak, 'On ternary coding and three-valued logic', july 2018.

(73) 특허권자
인하대학교 산학협력단
인천광역시 미추홀구 인하로 100(용현동, 인하대학교)
(72) 발명자
이문규
인천광역시 연수구 컨벤시아대로42번길 95, 1005동 1403호(송도동, 더샵 엑스포)
투라벡
인천광역시 미추홀구 인하로 100(용현동, 인하대학교)
김강욱
대구광역시 북구 구암서로 41, 310동 601호(구암동, 그린빌3차)
(74) 대리인
양성보

전체 청구항 수 : 총 9 항 심사관 : 구대성

(54) 발명의 명칭 **삼진 벡터 인코딩을 이용한 정수 대소비교 방법**

(57) 요 약

삼진 벡터 인코딩을 이용한 정수 대소비교 방법이 개시된다. 일 실시예에 따른 컴퓨터로 구현되는 대소비교 시스템에 의해 수행되는 대소비교 방법은, 복수 개의 정수 각각에 서로 다른 인코딩 방법을 적용함에 따라 각각의 벡터 정보를 획득하는 단계; 및 상기 획득된 각각의 벡터 정보를 이용하여 벡터 내적을 통한 대소비교를 수행하는 단계를 포함할 수 있다.

대 표 도 - 도5

Algorithm 5 Comparison of two values encoded in two input vectors

Input: Array of vectors $X[N]$ and $Y[N]$ for encoded values, vector size D
Ensure: $D \geq 3$
Output: Result of comparison r

$N \leftarrow D - 2$ {Number of vectors}
$n \leftarrow 0$
while $n < N$ **do** $\{0 \leq n < N\}$
  $r \leftarrow \langle X_n, Y_n \rangle$
  if $r \neq 1$ **then**
    break
  end if
  $n \leftarrow n + 1$
end while
return r

(52) CPC특허분류
H04L 9/0816 (2013.01)
H04L 2209/34 (2013.01)

이 발명을 지원한 국가연구개발사업
과제고유번호 1711108725
과제번호 2020R1A2C2013089
부처명 과학기술정보통신부
과제관리(전문)기관명 한국연구재단
연구사업명 중견연구
연구과제명 블록체인 상의 기밀성 제공을 위한 암호 최적화 및 응용 기술 개발
기 여 율 1/1
과제수행기관명 인하대학교
연구기간 2020.03.01 ~ 2023.02.28
공지예외적용 : 있음

명 세 서

청구범위

청구항 1

컴퓨터로 구현되는 대소비교 시스템에 의해 수행되는 대소비교 방법에 있어서,

복수 개의 정수 각각에 서로 다른 인코딩 방법을 적용함에 따라 각각의 벡터 정보를 획득하는 단계; 및

상기 획득된 각각의 벡터 정보를 이용하여 벡터 내적을 통한 대소비교를 수행하는 단계

를 포함하는 대소비교 방법.

청구항 2

제1항에 있어서,

상기 각각의 벡터 정보를 획득하는 단계는,

상기 복수 개의 정수 각각을 기 설정된 값에 대한 거듭제곱의 합으로 표현하고, 상기 표현된 기 설정된 값에 대한 거듭제곱의 합을 차수 정보에 기초하여 정렬하고, 상기 정렬된 거듭제곱의 합으로 표현된 각 항에 대한 인코딩의 결과로서 특정 차원의 벡터를 획득하는 단계

를 포함하고,

상기 특정 차원의 벡터는, 0, 1, 2 중 하나의 성분으로 구성된

것을 특징으로 하는 대소비교 방법.

청구항 3

제2항에 있어서,

상기 각각의 벡터 정보를 획득하는 단계는,

인코딩 규칙에 기초하여 상기 복수 개의 정수 각각에 대하여 0 또는 2의 거듭제곱의 합으로 표현된 각 항에 대한 인코딩을 수행함에 따라 인코딩의 결과를 산출하는 단계

를 포함하고,

상기 인코딩 규칙은,

입력이 0일 경우, 목표 위치 인덱스 p=0으로 설정하고, 입력이 0이 아닐 경우(즉, 2^i일 경우), p=i+1로 설정하는 출력 벡터의 p번째 원소를 1로 설정하고

출력 벡터의 원소들 중 p보다 인덱스가 작은 원소들을 0으로 설정하고

출력 벡터의 원소들 중 p보다 인덱스가 큰 원소들을 2로 설정하는

것을 특징으로 하는 대소비교 방법.

청구항 4

제2항에 있어서,

상기 각각의 벡터 정보를 획득하는 단계는,

특정 차원의 벡터의 개수를 이용하여 기 설정된 범위의 정수에 대하여 0 또는 2의 거듭제곱의 합으로 표현된 각 항에 복수 번의 인코딩을 적용하는 단계

를 포함하는 대소비교 방법.

청구항 5

제2항에 있어서,

상기 각각의 벡터 정보를 획득하는 단계는,

상기 복수 개의 정수 중 어느 하나의 정수로부터 표현된 2의 거듭제곱수 또는 0을 대상으로 원-핫(one-hot) 인코딩을 적용하는 단계

를 포함하는 대소비교 방법.

청구항 6

제1항에 있어서,

상기 대소비교를 수행하는 단계는,

상기 획득된 각각의 벡터 정보의 내적을 통해 0, 1 또는 2를 포함하는 내적값을 반환하는 단계

를 포함하는 대소비교 방법.

청구항 7

제6항에 있어서,

상기 획득된 각각의 벡터 정보는, 제1 벡터 및 제2 벡터를 포함하고,

상기 대소비교를 수행하는 단계는,

상기 제1 벡터가 인코딩하는 값이 제2 벡터가 인코딩하는 값보다 클 경우, 0을 반환하고, 상기 제1 벡터가 인코딩하는 값과 제2 벡터가 인코딩하는 값이 동일할 경우, 1을 반환하고, 상기 제1 벡터가 인코딩하는 값이 제2 벡터가 인코딩하는 값보다 작을 경우, 2를 반환하는 단계

를 포함하는 대소비교 방법.

청구항 8

제6항에 있어서,

상기 대소비교를 수행하는 단계는,

상기 획득된 각각의 벡터 정보의 내적을 통해 반환된 내적값이 1이 아닌 값이 나올 때까지 내적을 반복하는 단계

를 포함하는 대소비교 방법.

청구항 9

컴퓨터로 구현되는 대소비교 시스템에 있어서,

복수 개의 정수 각각에 서로 다른 인코딩 방법을 적용함에 따라 각각의 벡터 정보를 획득하는 인코딩부; 및

상기 획득된 각각의 벡터 정보를 이용하여 벡터 내적을 통한 대소비교를 수행하는 대소 비교부

를 포함하는 대소비교 시스템.

발명의 설명

기 술 분 야

[0001] 아래의 설명은 정수의 대소를 비교하는 기술에 관한 것이다.

배 경 기 술

[0003] 데이터에 대한 기밀성을 확보하기 위해 많은 응용에서 암호화 방법이 널리 사용되고 있으며, 특히 암호문 자체에 대한 기밀성은 유지하면서 암호문들 간의 연산을 통해 유용한 결과를 도출하는 데에는 함수암호가 사용되고 있다. 여기서 함수암호란, 평문 a와 평문 b를 암호화하여 암호문 A와 암호문 B를 생성하였을 때, 암호화에 이용한 키나 평문 a, b를 모르더라도 암호문 A와 B에 대한 연산만으로 평문 a, b에 대한 특정 함수값이 계산되도록 하는 암호화 기법이다. 특히, 평문 a, b가 벡터 형태일 때 A, B를 입력으로 a, b 벡터의 내적을 계산하되 a, b 자체에 대한 정보는 숨길 수 있는 내적 함수 암호 기법이 가장 대표적이며, 이는 본질적으로 데이터의 무결성은 보장되지만 다수에게 데이터가 전면 공개되는 문제가 있는 블록체인과 같은 환경에서 데이터를 보호하면서도 데이터에 대한 유용한 연산을 수행하는 데 필수적인 기술이다. 예를 들어 블록체인을 이용하는 개인 대 개인(peer to peer) 거래는 블록체인의 특성상 거래의 무결성은 보장되나 판매 및 구매자의 정보와 가격이 블록체인 상에 공개되는 문제가 있으나, 내적 함수 암호를 이용하면 거래 가격을 숨긴 상태에서 대소비교가 이루어지게 할 수 있으므로 가격 정보는 보호하면서 정상적인 거래를 수행할 수 있는 장점이 있다. 그러나 기존의 내적 함수 암호를 이용한 대소비교는 인코딩 대상 값의 범위를 1부터 N이라 할 때 N 차원 벡터가 필요하므로 대상 값의 범위가 넓을 경우 벡터의 차원이 증가하여 성능이 비효율적이 되는 문제가 있었다.

발명의 내용

해결하려는 과제

[0005] 인코딩 대상 값의 범위를 1부터 N이라고 할 때, N 대신 $(\log N)^2$ 차원의 벡터를 이용하여 기존의 대소비교 방법과 동일한 기능을 수행하게 하는 방법 및 시스템을 제공할 수 있다.

[0006] 삼진 벡터 인코딩을 이용한 정수 대소비교 방법 및 시스템을 제공할 수 있다.

과제의 해결 수단

[0008] 컴퓨터로 구현되는 대소비교 시스템에 의해 수행되는 대소비교 방법은, 복수 개의 정수 각각에 서로 다른 인코딩 방법을 적용함에 따라 각각의 벡터 정보를 획득하는 단계; 및 상기 획득된 각각의 벡터 정보를 이용하여 벡터 내적을 통한 대소비교를 수행하는 단계를 포함할 수 있다.

[0009] 상기 각각의 벡터 정보를 획득하는 단계는, 상기 복수 개의 정수 각각을 기 설정된 값에 대한 거듭제곱의 합으로 표현하고, 상기 표현된 기 설정된 값에 대한 거듭제곱의 합을 차수 정보에 기초하여 정렬하고, 상기 정렬된 거듭제곱의 합으로 표현된 각 항에 대한 인코딩의 결과로서 특정 차원의 벡터를 획득하는 단계를 포함하고, 상기 특정 차원의 벡터는, 0, 1, 2 중 하나의 성분으로 구성될 수 있다.

[0010] 상기 각각의 벡터 정보를 획득하는 단계는, 인코딩 규칙에 기초하여 상기 복수 개의 정수 각각에 대하여 0 또는 2의 거듭제곱의 합으로 표현된 각 항에 대한 인코딩을 수행함에 따라 인코딩의 결과를 산출하는 단계를 포함하고, 상기 인코딩 규칙은, 입력이 0일 경우, 목표 위치 인덱스 p=0으로 설정하고, 입력이 0이 아닐 경우(즉, 2^i 일 경우), p=i+1로 설정하는 출력 벡터의 p번째 원소를 1로 설정하고 출력 벡터의 원소들 중 p보다 인덱스가 작은 원소들을 0으로 설정하고 출력 벡터의 원소들 중 p보다 인덱스가 큰 원소들을 2로 설정할 수 있다.

[0011] 상기 각각의 벡터 정보를 획득하는 단계는, 특정 차원의 벡터의 개수를 이용하여 기 설정된 범위의 정수에 대하여 0 또는 2의 거듭제곱의 합으로 표현된 각 항에 복수 번의 인코딩을 적용하는 단계를 포함할 수 있다.

[0012] 상기 각각의 벡터 정보를 획득하는 단계는, 상기 복수 개의 정수 중 어느 하나의 정수로부터 표현된 2의 거듭제곱수 또는 0을 대상으로 원-핫(one-hot) 인코딩을 적용하는 단계를 포함할 수 있다.

[0013] 상기 대소비교를 수행하는 단계는, 상기 획득된 각각의 벡터 정보의 내적을 통해 0, 1 또는 2를 포함하는 내적값을 반환하는 단계를 포함할 수 있다.

[0014] 상기 획득된 각각의 벡터 정보는, 제1 벡터 및 제2 벡터를 포함하고, 상기 대소비교를 수행하는 단계는, 상기 제1 벡터가 인코딩하는 값이 제2 벡터가 인코딩하는 값보다 클 경우, 0을 반환하고, 상기 제1 벡터가 인코딩하는 값과 제2 벡터가 인코딩하는 값이 동일할 경우, 1을 반환하고, 상기 제1 벡터가 인코딩하는 값이 제2 벡터가 인코딩하는 값보다 작을 경우, 2를 반환하는 단계를 포함할 수 있다.

[0015] 상기 대소비교를 수행하는 단계는, 상기 획득된 각각의 벡터 정보의 내적을 통해 반환된 내적값이 1이 아닌 값이 나올 때까지 내적을 반복하는 단계를 포함할 수 있다.

[0016] 컴퓨터로 구현되는 대소비교 시스템은, 복수 개의 정수 각각에 서로 다른 인코딩 방법을 적용함에 따라 각각의 벡터 정보를 획득하는 인코딩부; 및 상기 획득된 각각의 벡터 정보를 이용하여 벡터 내적을 통한 대소비교를 수행하는 대소 비교부를 포함할 수 있다.

발명의 효과

[0018] 인코딩 대상 값의 범위를 1부터 N이라고 할 때, N 대신 $(\log N)^2$ 차원의 벡터를 이용하여 기존의 대소비교 방법과 같은 기능을 수행하게 함으로써, 메모리 사용량, 통신량, 계산량을 현저히 감소시킬 수 있다.

도면의 간단한 설명

[0020] 도 1 내지 도 4는 일 실시예에 따른 대소비교 시스템에서 각각의 정수에 대한 인코딩 동작을 설명하기 위한 도면이다.

도 5는 일 실시예에 따른 대소비교 시스템에서 대소비교를 설명하기 위한 도면이다.

도 6은 일 실시예에 따른 대소비교 시스템의 구성을 설명하기 위한 블록도이다.

도 7은 일 실시예에 따른 대소비교 시스템에서 삼진 벡터 인코딩을 이용한 정수의 대소비교 방법을 설명하기 위한 흐름도이다.

발명을 실시하기 위한 구체적인 내용

[0021] 이하, 실시예를 첨부한 도면을 참조하여 상세히 설명한다.

[0023] 실시예에서는 데이터의 무결성은 보장되지만 다수에게 데이터가 전면 공개되는 블록체인과 같은 환경에서 데이터 값을 공개하지 않으면서도 데이터 간의 대소비교가 가능하게 하는 내적 함수 암호 응용에 있어, 기존의 기술에 비해 현저히 적은 메모리 사용 및 현저히 적은 연산만으로 같은 범위의 숫자 간의 비교를 가능하게 하는 새로운 인코딩 방법을 설명하기로 한다.

[0024] 도 1 내지 도 4는 일 실시예에 따른 대소비교 시스템에서 각각의 정수에 대한 인코딩 동작을 설명하기 위한 도면이다.

[0025] 비교 대상 두 정수를 V_x, V_y라고 하자. 실시예에서는 두 정수 V_x, V_y에 대하여 서로 다른 인코딩 방법을 적용하는데, 두 방법 모두 인코딩 대상 정수를 2의 거듭제곱의 합으로 표현한 뒤, 높은 차수의 항부터 정렬하는 것으로 시작한다. 예를 들어, V_x=23이라 하면, 23=16+4+2+1로 표현될 수 있으며, 16, 4, 2, 1의 각 항들은 독립적으로 인코딩될 수 있다. 각 자리수(항)에 대한 인코딩을 X_0, X_1, ... 으로 표기하고, 전체에 대한 인코딩을 X로 표기하면 23에 대한 인코딩은 다음과 같이 표현될 수 있다.

[0026] $X(V_x=23=16+4+2+1) = X_0(16)+ X_1(4)+ X_2(2)+ X_3(1) \rightarrow [X_0(16), X_1(4), X_2(2), X_3(1)]$

[0027] 여기서, 각 항에 대한 인코딩 X_i은 다음과 같이 구현될 수 있다. 경우에 따라 0에 대한 인코딩이 필요할 수 있으므로, 각 항의 인코딩 X_i 대상은 2의 거듭제곱 또는 0이다. 각 항의 인코딩 X_i의 결과는 D차원 벡터이다. D=8일 경우, 벡터는 다음과 같은 8가지 값을 인코딩할 수 있다.

[0028] $\{0, 2^0, 2^1, 2^2, 2^3, 2^4, 2^5, 2^6\}=\{0, 1, 2, 4, 8, 16, 32, 64\}$

[0029] 각 항에 대한 인코딩 X_i의 결과로 산출되는 D 차원 벡터의 각 성분은 {0, 1, 2} 중 하나이며, 인코딩 규칙은 다음과 같다.

[0030] 1. 입력이 0일 경우, 목표 위치 인덱스 p=0으로 설정할 수 있다. 입력이 0 이 아닐 경우(즉, 2^i일 경우) p=i+1로 설정한다. 출력 벡터의 p번째 원소를 1로 설정한다.

[0031] 2. 출력 벡터의 원소들 중 p보다 인덱스가 작은 원소들은 0으로 설정한다.

[0032] 3. 출력 벡터의 원소들 중 p보다 인덱스가 큰 원소들은 2로 설정한다.

[0033] 예를 들어, 16(2^4)은 아래와 같이 인코딩될 수 있다.

[0034]

0	1	2	4	8	16	32	64
0	0	0	0	0	1	2	2

[0035] 8가지 가능한 모든 값들에 대한 인코딩 결과는 다음과 같이 나타낼 수 있다.

[0036]

	0	1	2	4	8	16	32	64
$V_x = 0$	1	2	2	2	2	2	2	2
$V_x = 1$	0	1	2	2	2	2	2	2
$V_x = 2$	0	0	1	2	2	2	2	2
$V_x = 4$	0	0	0	1	2	2	2	2
$V_x = 8$	0	0	0	0	1	2	2	2
$V_x = 16$	0	0	0	0	0	1	2	2
$V_x = 32$	0	0	0	0	0	0	1	2
$V_x = 64$	0	0	0	0	0	0	0	1

[0037] 이러한 과정을 알고리즘으로 표현하면 도 1과 같이 나타낼 수 있다.

[0038] 2의 거듭제곱수 또는 0에 대한 인코딩을 여러 번 적용하면, 일반적인 정수에 대한 인코딩이 가능하다.

[0039] 예를 들어 상기의 $X(V_x=23=16+4+2+1)= X_0(16)+ X_1(4)+ X_2(2)+ X_3(1)$ -> $[X_0(16), X_1(4), X_2(2), X_3(1)]$와 같은 예에서,

[0040] $X_0(16)=\{0, 0, 0, 0, 0, 1, 2, 2\}$

[0041] $X_1(4)=\{0, 0, 0, 1, 2, 2, 2, 2\}$

[0042] $X_2(2)=\{0, 0, 1, 2, 2, 2, 2, 2\}$

[0043] $X_3(1)=\{0, 1, 2, 2, 2, 2, 2, 2\}$

[0044] 이 되어 4개의 벡터가 필요하다.

[0045] 상기와 같은 조합에 의해 8차원 벡터들로 표현될 수 있는 숫자의 범위는 0부터 127(=64+32+16+8+4+2+1)이며, 127은 인코딩을 위해 최소 7개의 벡터가 필요하다. 단, 7비트 이진수(0부터 127 사이의 정수) 중 7개의 벡터가 필요한 유일한 조합은 127뿐이므로, 표현 가능한 숫자의 범위를 0부터 126으로 하면, 6개의 벡터만으로 표현이 가능하다. 다시 말해서, 표현 가능 숫자 범위와 사용되는 벡터의 수 사이에 조절이 가능하나, 아래에서는 일례로서 D 차원의 벡터 N(=D-2)개를 이용하여 R=[0, 2^{N+1}-2] 범위의 정수를 표현하는 방법을 설명하기로 한다. 단, 인코딩 대상 정수 V_x의 값에 따라 인코딩된 벡터의 개수가 달라지면 안전성에 문제가 발생할 수 있으므로, 인코딩 대상 정수 V_x값과 무관하게 결과 벡터의 개수를 일정하게 유지할 수 있다. 이것이 0의 인코딩이 필요한 근거이다.

[0046] 예를 들면, 96을 6개의 벡터로 인코딩한 결과(즉, D=8, N=6)는 다음과 같이 나타낼 수 있다.

[0047] V_x=96:

[0048] $X_0(64)=\{0, 0, 0, 0, 0, 0, 0, 1\}$

[0049] $X_1(32)=\{0, 0, 0, 0, 0, 0, 1, 2\}$

[0050] $X_2(0)= X_3(0)= X_4(0)= X_5(0)=\{1, 2, 2, 2, 2, 2, 2, 2\}$

[0051] 상기의 과정을 일반화하면 도 2와 같은 알고리즘이 된다.

[0052] 정수 V_y에 대한 인코딩에 대하여 설명하기로 한다. V_y에 대한 인코딩은 인코딩 대상 정수 V_y를 2의 거듭제곱의 합으로 표현한 뒤, 높은 차수의 항부터 정렬할 수 있다. 단, V_y에 대한 인코딩은 2의 거듭제곱수 또는 0을 대

상으로 원-핫(one-hot) 인코딩을 적용할 수 있다. 다시 말해서, D=8이라면,

[0053] {0, 2^0, 2^1, 2^2, 2^3, 2^4, 2^5, 2^6}={0, 1, 2, 4, 8, 16, 32, 64}

[0054] 값들은 다음과 같이 인코딩될 수 있다.

	0	1	2	4	8	16	32	64
$V_y = 0$	1	0	0	0	0	0	0	0
$V_y = 1$	0	1	0	0	0	0	0	0
$V_y = 2$	0	0	1	0	0	0	0	0
$V_y = 4$	0	0	0	1	0	0	0	0
$V_y = 8$	0	0	0	0	1	0	0	0
$V_y = 16$	0	0	0	0	0	1	0	0
$V_y = 32$	0	0	0	0	0	0	1	0
$V_y = 64$	0	0	0	0	0	0	0	1

[0055]

[0056] 이러한 과정을 알고리즘으로 표현하면 도 3과 같다.

[0057] V_x에서와 같이, 0 또는 2의 거듭제곱이 아닌 값을 나타낼 때에도 동일한 속성을 사용할 수 있다. 예를 들어 17을 인코딩하면 다음과 같이 나타낼 수 있다.

[0058] $Y(V_y=17=16+1)=Y_0(16)+Y_1(1)$ -> $[Y_0(16), Y_1(1)]$

[0059] $Y_0(16)$={0, 0, 0, 0, 0, 1, 0, 0}

[0060] $Y_1(1)$={0, 1, 0, 0, 0, 0, 0, 0}

[0061] $Y_2(0)=Y_3(0)=Y_4(0)=Y_5(0)$={1, 0, 0, 0, 0, 0, 0, 0}

[0062] 상기의 과정을 일반화하면 도 4와 같은 알고리즘이 된다.

[0063] 도 5는 일 실시예에 따른 대소비교 시스템에서 대소비교를 설명하기 위한 도면이다.

[0064] 인코딩된 V_x, V_y에 대해 대소비교를 수행하는 과정은 도 5의 알고리즘과 같다. 대소비교를 수행하는 알고리즘의 입력은 각각 V_x, V_y를 인코딩한 벡터들의 배열인 X[N]과 Y[N]이며, 알고리즘의 반환값은 0, 1 또는 2이다. 4행의 X_n, Y_n은 X[N]과 Y[N] 배열의 n번째 원소들인 X_n와 Y_n 간의 벡터 내적을 수행하는 것을 의미한다. 개별 벡터에 대한 내적 X_n Y_n의 결과는 다음과 같다.

[0065] 내적 0: X_n이 인코딩하는 값 > Y_n이 인코딩하는 값

[0066] 내적 1: X_n이 인코딩하는 값 = Y_n이 인코딩하는 값

[0067] 내적 2: X_n이 인코딩하는 값 < Y_n이 인코딩하는 값

[0068] 이에 따라, while 루프는 내적 $r_i = X_i \bullet Y_i \text{ for } i \in [0, N-1]$이 첫 번째로 1이 아닌 값이 나올 때까지, 즉, 비교 대상 V_x, V_y의 자리수 값이 처음으로 달라지는 데까지 반복되며, 해당 시점에서의 개별 벡터 내적값에 따라 전체 비교 결과가 결정될 수 있다. 다시 말해서, 해당 시점에서의 내적값이 도 5의 알고리즘의 반환값이 될 수 있다. 만약, while 루프가 종료될 때까지 내적값이 계속 1이면 도 5의 알고리즘의 반환값도 1이 되며 이는 V_x=V_y를 의미한다.

[0069] 다음 예제는 알고리즘의 반환값이 각각 0, 1, 2인 경우를 나타낸 것이다.

[0070] V_x=103, V_y=93인 경우 (0 반환)

X	0	1	2	4	8	16	32	64
$X_0(64)$	0	0	0	0	0	0	0	1
$X_1(32)$	0	0	0	0	0	0	1	2
$X_2(4)$	0	0	0	1	2	2	2	2
$X_3(2)$	0	0	1	2	2	2	2	2
$X_4(1)$	0	1	2	2	2	2	2	2
$X_5(0)$	1	2	2	2	2	2	2	2

r_i
1
0
2
2
1
1

Y	0	1	2	4	8	16	32	64
$Y_0(64)$	0	0	0	0	0	0	0	1
$Y_1(16)$	0	0	0	0	0	1	0	0
$Y_2(8)$	0	0	0	0	1	0	0	0
$Y_3(4)$	0	0	0	1	0	0	0	0
$Y_4(1)$	0	1	0	0	0	0	0	0
$Y_5(0)$	1	0	0	0	0	0	0	0

[0071]

[0072] V_x=93, V_y=93인 경우 (1 반환)

X	0	1	2	4	8	16	32	64
$X_0(64)$	0	0	0	0	0	0	0	1
$X_1(16)$	0	0	0	0	0	1	2	2
$X_2(8)$	0	0	0	0	1	2	2	2
$X_3(4)$	0	0	0	1	2	2	2	2
$X_4(1)$	0	1	2	2	2	2	2	2
$X_5(0)$	1	2	2	2	2	2	2	2

r_i
1
1
1
1
1
1

Y	0	1	2	4	8	16	32	64
$Y_0(64)$	0	0	0	0	0	0	0	1
$Y_1(16)$	0	0	0	0	0	1	0	0
$Y_2(8)$	0	0	0	0	1	0	0	0
$Y_3(4)$	0	0	0	1	0	0	0	0
$Y_4(1)$	0	1	0	0	0	0	0	0
$Y_5(0)$	1	0	0	0	0	0	0	0

[0073]

[0074] V_x=95, V_y=96인 경우 (2 반환)

X	0	1	2	4	8	16	32	64
$X_0(64)$	0	0	0	0	0	0	0	1
$X_1(16)$	0	0	0	0	0	1	2	2
$X_2(8)$	0	0	0	0	1	2	2	2
$X_3(4)$	0	0	0	1	2	2	2	2
$X_4(2)$	0	0	1	2	2	2	2	2
$X_5(1)$	0	1	2	2	2	2	2	2

r_i
1
2
0
0
0
0

Y	0	1	2	4	8	16	32	64
$Y_0(64)$	0	0	0	0	0	0	0	1
$Y_1(32)$	0	0	0	0	0	0	1	0
$Y_2(0)$	1	0	0	0	0	0	0	0
$Y_3(0)$	1	0	0	0	0	0	0	0
$Y_4(0)$	1	0	0	0	0	0	0	0
$Y_5(0)$	1	0	0	0	0	0	0	0

[0075]

[0076] 도 6은 일 실시예에 따른 대소비교 시스템의 구성을 설명하기 위한 블록도이고, 도 7은 일 실시예에 따른 대소비교 시스템에서 삼진 벡터 인코딩을 이용한 정수의 대소비교 방법을 설명하기 위한 흐름도이다.

[0077] 대소비교 시스템(100)의 프로세서는 인코딩부(610) 및 대소 비교부(620)를 포함할 수 있다. 이러한 프로세서의 구성요소들은 대소비교 시스템에 저장된 프로그램 코드가 제공하는 제어 명령에 따라 프로세서에 의해 수행되는 서로 다른 기능들(different functions)의 표현들일 수 있다. 프로세서 및 프로세서의 구성요소들은 도 7의 삼진 벡터 인코딩을 이용한 정수의 대소비교 방법이 포함하는 단계들(710 내지 720)을 수행하도록 대소비교 시스템을 제어할 수 있다. 이때, 프로세서 및 프로세서의 구성요소들은 메모리가 포함하는 운영체제의 코드와 적어도 하나의 프로그램의 코드에 따른 명령(instruction)을 실행하도록 구현될 수 있다.

[0078] 프로세서는 삼진 벡터 인코딩을 이용한 정수의 대소비교 방법을 위한 프로그램의 파일에 저장된 프로그램 코드를 메모리에 로딩할 수 있다. 예를 들면, 대소비교 시스템에서 프로그램이 실행되면, 프로세서는 운영체제의 제어에 따라 프로그램의 파일로부터 프로그램 코드를 메모리에 로딩하도록 대소비교 시스템을 제어할 수 있다. 이때, 프로세서 및 프로세서가 포함하는 인코딩부(610) 및 대소 비교부(620) 각각은 메모리에 로딩된 프로그램 코드 중 대응하는 부분의 명령을 실행하여 이후 단계들(710 내지 720)을 실행하기 위한 프로세서의 서로 다른 기능적 표현들일 수 있다.

[0079] 단계(710)에서 인코딩부(610)는 복수 개의 정수 각각에 서로 다른 인코딩 방법을 적용함에 따라 각각의 벡터 정보를 획득할 수 있다. 인코딩부(610)는 복수 개의 정수 각각을 기 설정된 값에 대한 거듭제곱의 합으로 표현하고, 표현된 기 설정된 값에 대한 거듭제곱의 합을 차수 정보(예를 들면, 내림차순, 오름차순)에 기초하여 정렬하고, 정렬된 거듭제곱의 합으로 표현된 각 항에 대한 인코딩의 결과로서 특정 차원의 벡터를 획득하는 단계를 포함할 수 있다. 이때, 특정 차원의 벡터는, 0, 1, 2 중 하나의 성분으로 구성될 수 있다. 인코딩부(610)는 인코딩 규칙에 기초하여 복수 개의 정수 각각에 대하여 0 또는 2의 거듭제곱의 합으로 표현된 각 항에 대한 인코딩을 수행함에 따라 인코딩의 결과를 산출할 수 있다. 이때, 인코딩 규칙은, 입력이 0일 경우, 목표 위치 인덱스 p=0으로 설정하고, 입력이 0이 아닐 경우(즉, 2^i일 경우), p=i+1로 설정하는 출력 벡터의 p번째 원소를 1로 설정하고 출력 벡터의 원소들 중 p보다 인덱스가 작은 원소들을 0으로 설정하고 출력 벡터의 원소들 중 p보

다 인덱스가 큰 원소들을 2로 설정할 수 있다. 인코딩부(610)는 특정 차원의 벡터의 개수를 이용하여 기 설정된 범위의 정수에 대하여 0 또는 2의 거듭제곱의 합으로 표현된 각 항에 복수 번의 인코딩을 적용할 수 있다. 인코딩부(610)는 복수 개의 정수 중 어느 하나의 정수로부터 표현된 2의 거듭제곱수 또는 0을 대상으로 원-핫(one-hot) 인코딩을 적용할 수 있다.

[0080] 단계(720)에서 대소 비교부(620)는 획득된 각각의 벡터 정보를 이용하여 벡터 내적을 통한 대소비교를 수행할 수 있다. 대소 비교부(620)는 획득된 각각의 벡터 정보의 내적을 통해 0, 1 또는 2를 포함하는 내적값을 반환할 수 있다. 대소 비교부(620)는 벡터 정보 중 제1 벡터가 인코딩하는 값이 제2 벡터가 인코딩하는 값보다 클 경우, 0을 반환하고, 제1 벡터가 인코딩하는 값과 제2 벡터가 인코딩하는 값이 동일할 경우, 1을 반환하고, 제1 벡터가 인코딩하는 값이 제2 벡터가 인코딩하는 값보다 작을 경우, 2를 반환할 수 있다. 대소 비교부(620)는 획득된 각각의 벡터 정보의 내적을 통해 반환된 내적값이 1이 아닌 값이 나올 때까지 내적을 반복할 수 있다.

[0081] 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, 프로세서, 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.

[0082] 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.

[0083] 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.

[0084] 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.

[0085] 그러므로, 다른 구현들, 다른 실시예들 및 특허청구범위와 균등한 것들도 후술하는 특허청구범위의 범위에 속한다.

도면

도면1

Algorithm $X(V, D)$ method

Input: Value to be encoded V, size of a vector D
Ensure: $D \geq 3$, V is 0 or power of 2
Output: Vector with encoded value x of size D

if $V = 0$ **then**
  $p \leftarrow 0$ {p - Position for 1}
else
  $p \leftarrow \log_2 V + 1$
end if
for $i = 0$ to D **do** $\{0 \leq i < D\}$
  if $i < p$ **then**
    $x_i \leftarrow 0$
  else if $i = p$ **then**
    $x_i \leftarrow 1$
  else
    $x_i \leftarrow 2$
  end if
end for
return x

도면2

Algorithm Encoding a value with X method

Input: Value to be encoded V, vector size D
Ensure: $V \in R$ and $D \geq 3$
Output: Array of vectors $X_0, X_1, ...X_{N-1}$

$N \leftarrow D - 2$ {Number of vectors}
$i \leftarrow D - 2$ {Highest power of two}
$n \leftarrow 0$ {Current vector}
while $n < N$ **do** $\{0 \leq n < N\}$
  if $V \geq 2^i$ **then**
    $X_n \leftarrow X(2^i, D)$
    $n \leftarrow n + 1$
    $V \leftarrow V - 2^i$
  else if $V = 0$ **then**
    $X_n \leftarrow X(0, D)$
    $n \leftarrow n + 1$
  else
    $i \leftarrow i - 1$
  end if
end while
return $[X_0, X_1, ..., X_{N-1}]$

도면3

Algorithm $Y(V, D)$ method

Input: Value to be encoded V, size of a vector D
Ensure: $D \geq 3$, V is 0 or power of 2
Output: Vector with encoded value y of size D

if V=0 **then**
  $p \leftarrow 0$ {p - Position for 1}
else
  $p \leftarrow \log_2 V + 1$
end if
for $i = 0$ to D **do** $\{0 \leq i < D\}$
  if $i = p$ **then**
    $y_i \leftarrow 1$
  else
    $y_i \leftarrow 0$
  end if
end for
return y

도면4

Algorithm Encoding a value with Y method

Input: Value to be encoded V, vector size D
Ensure: $V \in R$ and $D \geq 3$
Output: Array of vectors $Y_0, Y_1, ...Y_{N-1}$

$N \leftarrow D - 2$ {Number of vectors}
$i \leftarrow D - 2$ {Highest power of two}
$n \leftarrow 0$ {Current vector}
while $n < N$ **do** $\{0 \leq n < N\}$
  if $V \geq 2^i$ **then**
    $Y_n \leftarrow Y(2^i, D)$
    $n \leftarrow n + 1$
    $V \leftarrow V - 2^i$
  else if $V = 0$ **then**
    $Y_n \leftarrow Y(0, D)$
    $n \leftarrow n + 1$
  else
    $i \leftarrow i - 1$
  end if
end while
return $[Y_0, Y_1, ..., Y_{N-1}]$

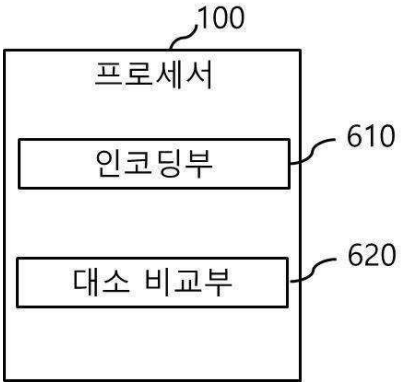
도면5

Algorithm 5 Comparison of two values encoded in two input vectors

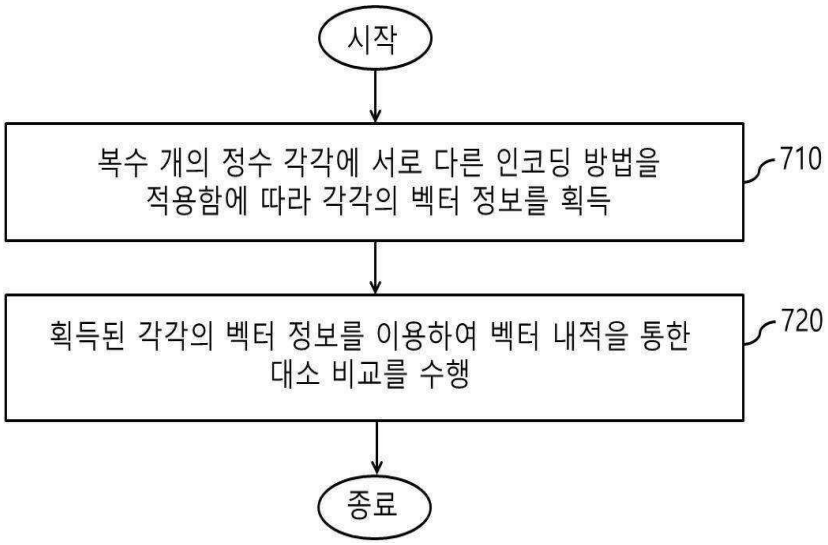
Input: Array of vectors $X[N]$ and $Y[N]$ for encoded values, vector size D
Ensure: $D \geq 3$
Output: Result of comparison r

$N \leftarrow D - 2$ {Number of vectors}
$n \leftarrow 0$
while $n < N$ **do** $\{0 \leq n < N\}$
  $r \leftarrow \langle X_n, Y_n \rangle$
  if $r \neq 1$ **then**
    break
  end if
  $n \leftarrow n + 1$
end while
return r

도면6



도면7


시작
복수 개의 정수 각각에 서로 다른 인코딩 방법을 적용함에 따라 각각의 벡터 정보를 획득
710
획득된 각각의 벡터 정보를 이용하여 벡터 내적을 통한 대소 비교를 수행
720
종료