(19)대한민국특허청(KR)
(12) 공개특허공보(A)

(51) 。Int. Cl.
H03M 7/30 (2006.01)

(11) 공개번호 10-2006-0106835
(43) 공개일자 2006년10월12일

(21) 출원번호 10-2006-7009513
(22) 출원일자 2006년05월16일
번역문 제출일자 2006년05월16일
(86) 국제출원번호 PCT/AU2004/001406
국제출원일자 2004년10월15일
(87) 국제공개번호 WO 2005/039057
국제공개일자 2005년04월28일

(30) 우선권주장 2003905688 2003년10월17일 오스트레일리아(AU)

(71) 출원인 팩바이트 소프트웨어 피티와이 리미티드
오스트렐리아 뉴 사우스 웨일즈 2121 에핑 랑스톤 플레이스 36에이

(72) 발명자 파커 브루스
오스트레일리아 뉴 사우스 웨일즈 마운트 콜라 베릴 애비뉴 23

(74) 대리인 김진환
송승필

심사청구 : 없음

(54) 데이터 압축 시스템 및 방법

요약

본 발명은 본 발명은 소정의 길이 보다 크거나 또는 동일한 길이의 바이트들의 시퀀스를 포함하는 데이터 파일을 압축하는 방법을 제공하며, 이 방법은, 보조 기억 장치로 부터 상기 데이터 파일을 검색(retrieve)하는 단계; 그 데이터 파일을 직접 액세스 메모리에 저장하는 단계; 그 데이터 파일의 서브 시퀀스 내의 고유의 바이트 값들의 주파수를 계산하는 단계로서, 이 서브 시퀀스는 소정의 길이를 초과하지 않는 길이를 가지는 것인, 주파수 계산 단계; 이 서브 시퀀스에 대한 인덱스를 생성하는 단계로서, 이러한 인덱스는 계산된 서브 시퀀스 내의 고유의 바이트 값들의 주파수를 나타내는 데이터 값을 포함하는 것인, 인덱스 생성 단계; 미리 결정된 임계값 미만의 고유의 바이트 값들의 주파수를 가지는 서브 시퀀스 상에서, 이 서브 시퀀스 내의 고유의 바이트 값들의 주파수를 증가시키기 위하여 이 서브 시퀀스에 데이터 변환을 적용하는 단계, 및 이러한 인덱스에 데이터 변환을 나타내는 데이터 값을 더하는 단계; 소정의 임계값 이상의 고유의 바이트 값들의 주파수를 가지는 서브 시퀀스 상에서, 이러한 인덱스에 이 서브 시퀀스 내의 하나 이상의 고유의 값의 위치를 나타내는 데이터 값을 더하는 단계; 파일 타입 식별자를 가지는 출력 데이터 파일을 생성하는 단계; 및 이 출력 데이터 파일에 이러한 인덱스를 더하는 단계를 포함한다.

대표도

도 1

색인어

컴퓨팅 장치

명세서

기술분야

본 발명은 데이터 압축 분야에 관한 것으로, 특히 팩토리얼 반복 무손실 압축에 기초한 데이터 압축 시스템 및 방법에 관한 것이다.

배경기술

전자 2 진 파일은 많은 서로 다른 사용을 위하여 많은 서로 다른 포맷으로 표현된다. 이 포맷들은 이미지, 사운드, 텍스트, 데이터, 실행가능한 파일 등의 저장 장치에 대하여 적합한 포맷들을 포함한다.

데이터를 포함하는 2 진 파일은, 만일 암호화되지 않으면, 구조화된 포맷으로 되는 경향이 있다. 통상적으로 헤더 정보, 텍스트, 반복, 및 다른 구성요소들에 대한 위치결정이 존재한다. 일반적으로, 2 진 파일 내의 처음 일부 바이트는 파일 타입의 표시자 및 2 진 파일이 호환될 수 있는 애플케이션을 포함한다. 실행가능한 파일 또는 임의의 타입의 기능을 수행하는데 사용되는 파일들은 상당히 덜 구조화된 포맷을 가진다. 그러나, 이러한 파일들이 동작 시스템과 상호작용하여 기능을 수행하거나 또는 동작 시스템의 일부가 되기 때문에 구조의 엘리먼트가 존재한다.

압축 및 암호화된 파일들은, 설계에 의해 이들이 파일 내의 반복 값들을 제거하기 때문에 최소한의 구조를 가진다. 암호화의 경우에, 키는 대체된 값들을 규정하는데 사용된다. 압축을 위하여, "속기(shorthand)" 가 반복적인 구조들에 대하여 사용된다. 암호화되거나 또는 압축된 파일의 경우에, 그 파일은 변경된 내부 구조를 가질 뿐만 아니라 특히 압축의 경우에 파일의 크기를 가진다.

수학적으로, 1,048,576 바이트(1Mb)의 크기의 2 진 파일에 대하여, 허용가능한 바이트들의 배열의 $256^{1,048,576}$ 개의 허용가능한 구조가 존재한다. 실제 사용시에, 이러한 수치의 일부만이 사용된다. 실제로 사용되는 번호는 복수의 서로 다른 파일 타입의 추정값, 실행가능하거나 또는 동작가능한 파일들의 기능성, 그리고 이용가능한 압축 및 암호화 루틴에 기초하여 단지 근사화될 수 있다.

데이터 파일상에 데이터 압축을 수행하기 위하여 존재하는 많은 기술들이 존재한다. 일부 데이터 압축 알고리즘은 인덱싱 기술에 기초하며, 데이터 파일 내의 고유의 값의 계산 및 인덱싱을 포함한다. 대부분의 압축된 데이터 파일들에 있어서, 각각의 256 바이트 코드 세그먼트 내의 데이터 값들의 반복이 일부 존재한다. 평균 파일에 있어서, 코드의 256 바이트 세그먼트 당 160 내지 170 개의 고유의 비반복된 값만이 존재한다. 팩토리얼 계산에 기초한 데이터 압축 기술은 이러한 값들의 수와 매우 잘 동작하지 못 한다.

발명의 상세한 설명

일 양태에서, 본 발명은 소정의 길이 보다 크거나 또는 동일한 길이의 바이트들의 시퀀스를 포함하는 데이터 파일을 압축하는 방법을 제공하며, 이 방법은, 보조 기억 장치로 부터 상기 데이터 파일을 검색(retrieve)하는 단계; 그 데이터 파일을 직접 액세스 메모리에 저장하는 단계; 그 데이터 파일의 서브 시퀀스 내의 고유의 바이트 값들의 주파수를 계산하는 단계로서, 이 서브 시퀀스는 소정의 길이를 초과하지 않는 길이를 가지는 것인, 주파수 계산 단계; 이 서브 시퀀스에 대한 인덱스를 생성하는 단계로서, 이러한 인덱스는 계산된 서브 시퀀스 내의 고유의 바이트 값들의 주파수를 나타내는 데이터 값을 포함하는 것인, 인덱스 생성 단계; 미리 결정된 임계값 미만의 고유의 바이트 값들의 주파수를 가지는 서브 시퀀스 상에서, 이 서브 시퀀스 내의 고유의 바이트 값들의 주파수를 증가시키기 위하여 이 서브 시퀀스에 데이터 변환을 적용하는 단계, 및 이러한 인덱스에 데이터 변환을 나타내는 데이터 값을 더하는 단계; 소정의 임계값 이상의 고유의 바이트 값들의 주파수를 가지는 서브 시퀀스 상에서, 이러한 인덱스에 이 서브 시퀀스 내의 하나 이상의 고유의 값의 위치를 나타내는 데이터 값을 더하는 단계; 파일 타입 식별자를 가지는 출력 데이터 파일을 생성하는 단계; 및 이 출력 데이터 파일에 이러한 인덱스를 더하는 단계를 포함한다.

도면의 간단한 설명

이하, 본 발명의 데이터 압축 시스템 및 방법의 바람직한 형태들을 첨부된 도면들을 참조하여 설명한다.

도 1 은 본 발명의 시스템의 바람직한 형태를 나타낸다.

도 2, 3 및 4 는 본 발명의 바람직한 형태의 압축 프로세스의 흐름도를 나타낸다.

도 5 는 본 발명의 바람직한 실시형태들에 대하여 예상된 압축 결과들의 테이블을 나타낸다.

도 6 은 다중-반복 바이트 압축 향상에 관한 본 발명의 추가적인 양태를 나타낸다.

도 7 은 다중-반복 바이트 압축 향상에 관한 본 발명의 추가적인 양태를 낱낸다.

실시예

본 발명은 데이터 파일(5)에 적용되는 데이터 압축 시스템 및 방법을 제공한다. 데이터 파일(5)은 BMP, WAV, DOC, XLS, MDB, ZIP, SIT, ARJ, ZOO, TIF, JPG, GIF, MP3, MP4 등을 포함하는 임의의 적절한 데이터 포맷일 수 있다. 데이터 파일(5)은 컴퓨팅 장치(15)의 일부를 형성하거나 또는 컴퓨팅 장치(15)와 적어도 인터페이싱하는 보조 기억장치(10)에 기억될 수 있다. 컴퓨팅 장치(15) 는 적어도 직접 액세스 메모리(25)와 인터페이싱하는 프로세서(20)와 디스플레이(30)를 포함한다. 컴퓨팅 장치는 예를 들어 데이터 입력 장치(미도시)와 출력 장치(미도시) 인 다른 구성요소들을 포함하거나 또는 이들과 인터페이싱될 수 있다.

데이터 파일(5)이 소정의 길이 보다 크거나 또는 동일한 길이의 바이트들의 시퀀스를 포함한다. 본 발명의 바람직한 형태에서, 소정의 길이는 300 바이트이다.

동작시에, 컴퓨팅 장치(15)의 프로세서(20)는 보조 기억장치(10)로부터 데이터 파일(5)의 전체 또는 일부를 검색한다. 그 검색된 데이터 파일의 전체 또는 일부를 직접 액세스 메모리(25)에 기억한다. 여러가지 동작들을 직접 액세스 메모리에 기억된 상기 데이터 파일의 전체 또는 일부에서 수행한다. 결과적인 출력 데이터 파일(35)은 직접 액세스 메모리(25)에서 생성되고, 보조 기억 장치(10) 또는 다른 보조 기억 장치에 기억된다. 많은 경우에서, 출력 데이터 파일(35)은 데이터 파일(5) 보다 작은 크기를 가질 수 있다.

데이터 파일(5)의 서브 시퀀스를 먼저 조사한다. 서브 시퀀스의 길이는 바람직하기로는 소정의 길이인 300 바이트를 초과하지 않는다. 식별된 고유의 값들의 개수가 임계값 아래로 떨어지는 경우, 서브 시퀀스 내의 고유의 바이트 값의 주파수를 증가시키기 위하여 데이터 변환의 시퀀스를 서브 시퀀스에 적용할 수 있다.

복수의 데이터 변환 구성요소(40)가 직접 액세스 메모리(25) 또는 보조 기억 장치에 기억된다. 데이터 변환 구성요소(40)는 복수의 랜덤하게 생성된 바이트 값들의 시퀀스 또는 소정의 바이트 값들의 시퀀스를 포함할 수 있다. 이 시퀀스는 마스크 구조(45)로서 기억된다. 또한, 대안으로서 또는 바람직하기로는, 데이터 변환 구성요소들은 또한 부가적인 마스크 구조(45)들을 생성하는데 사용될 수 있는 복수의 마스크 규격(formulae)(50)을 포함한다. 이하, 데이터 변환 구성요소들의 애플리케이션을 추가적으로 설명한다.

또한, 시스템은 복수의 인덱싱 구성요소(60)를 포함한다. 데이터 파일(5)의 서브 시퀀스의 프로세싱 동안에, 출력 데이터 파일(35)에 후속하여 기록되는 인덱스(65)를 생성한다. 또한, 인덱싱 구성요소(60)는 임시 위치 인덱스(70), 위치 인덱스(75) 및 순열 인덱스(80)를 포함할 수 있다. 일부 경우에서 위치 인덱스(75) 및 순열 인덱스(80)의 컨텐츠를 인덱스(65)에 더 할 수 있다. 이하에서는 여러가지 인덱싱 구성요소(60)의 동작을 추가적으로 설명한다.

또한, 시스템은 직접 액세스 메모리(25) 또는 보조 기억장치에 기억되는 데이터 어레이(90)를 포함할 수도 있다. 데이터 어레이(90)는, 데이터 어레이(90)의 컨텐츠를 출력 데이터 파일(35)에 기록하기 이전에, 여러가지 인덱싱 구성요소(60) 및 압축되는 데이터 파일(5)의 서브 시퀀스의 일부를 기억하는데 사용될 수 있다.

도 2 내지 도 4 는 본 발명의 바람직한 형태의 동작을 나타낸다. 2 진 데이터 파일(5)은 바람직하기로는 복수의 데이터 그룹으로 분할된다. 본 발명의 하나의 바람직한 형태에서, 각각의 데이터 그룹은 바람직하기로는 300 바이트 또는 그 미만이다. 그러나, 압축되는 데이터 그룹의 크기는 5 비트가 넘는 임의의 크기를 가질 수 있음을 이해해야 한다. 데이터 파일은 먼저 데이터 파일이 소정의 길이 보다 크거나 또는 동일한 길이를 가지는 지를 규정하기 위하여 조사된다(단계 200). 본 발명

의 하나의 바람직한 형태에서, 초기의 소정 길이는 300 바이트이다. 하나의 형태에서, 전체 데이터 파일은 보조 기억 장치로부터 검색될 수 있고, 이 전체 데이터 파일은 데이터 어레이(90) 및 직접 액세스 메모리(25)에 기억될 수 있다. 다른 방법으로, 데이터 파일(5)의 일부는 데이터 스트림으로서 보조 기억장치(10) 로부터 검색될 수 있다.

데이터 그룹은 데이터 그룹 내의 고유의 데이터 값들의 주파수를 계산하기 위하여 카운트된다(단계 205). 고유의 데이터 값들의 주파수를 소정의 임계값과 비교한다(단계 210). 하나의 바람직한 형태에서, 소정의 임계값은 256 이다. 300 바이트의 서브 시퀀스내에 256 개의 고유값보다 작은 값이 있다면, 서브 시퀀스 내의 고유의 바이트 값들의 주파수를 증가시키기 위하여 상기 서브 시퀀스에 하나 이상의 데이터 변환을 적용할 수 있다.

고유의 바이트 값들의 주파수가 300 바이트 내의 256 개의 값의 소정의 임계값 아래로 떨어지는 경우에, 데이터 변환 "마스크"를 서브 시퀀스에 적용할 수 있는지 여부를 식별하기 위하여 서브 시퀀스를 테스트한다(단계 215). 본 발명의 하나의 바람직한 형태에서, 구조 라이브러리는 예를 들어 직접 액세스 메모리(25)인 컴퓨터 메모리에 유지된다. 이러한 라이브러리는 바람직하기로는 랜덤하게 생성된 복수의 데이터 세트들을 포함한다. 이러한 데이터 세트들은 데이터 세트 식별자에 의해 각각 식별될 수 있으며, 이러한 데이터 식별자들은 컴퓨터 메모리에 기억되고, 각각의 램덤하게 생성된 데이터 세트들과 연관된다.

하나의 형태에서, 랜덤하게 생성된 데이터 세트들 중 하나 이상은 데이터 파일의 서브 시퀀스의 길이와 실질적으로 동일한 길이를 가진다. 즉, 서브 시퀀스 내의 바이트 수는 송신 데이터 세트 또는 마스크 내의 바이트 수와 동일하다. 이러한 마스크는 대응하는 바이트 값들 및 검색된 송신 데이터 세트에 기초하여 서브 시퀀스 내의 각각의 바이트 값들에 데이터 변환을 적용함으로써 서브 시퀀스에 적용될 수 있다.

데이터 변환의 일례는 모듈러스(modulus) 가산이다. 서브 시퀀스의 제 1 바이트 값과 데이터 세트의 제 1 바이트 값을 함께 더하고, 총 256 개의 모듈러스를 계산한다. 예를 들어, 서브 시퀀스의 제 1 이진 값이 168 이고, 식별된 데이터 세트의 제 1 이진 값 203 이면, 조합된 총계는 371 이다. 371 MOD 256 에 의해 계 산된, 변환된 값이 115 이다. 그 후, 상기 시퀀스 내의 2 바이트를 데이터 세트 내의 2 바이트에 의해 동일한 방식으로 변환한다. 그 후, 서브 시퀀스 내의 3 바이트를 데이터 세트 내의 3 바이트 등에 기초하여 변환한다.

이러한 방식으로, 마스크를 서브 시퀀스에 적용한다(단계 220).

하나의 형태에서, 컴퓨터 메모리에 기억된 65,536 개의 마스크 구조가 있으며, 각각의 마스크에는 0 과 65,536 사이의 인덱스 넘버의 형태로 데이터 세트 식별자가 제공된다. 이러한 인덱스는 단순히 관련 데이터 세트 식별자에 포인팅되는 14 비트 세그먼트일 수 있다.

데이터 변환 구성요소(40)는 예를 들어 마스크 규격을 포함할 수 있다.

- 선행하는 300 바이트 또는 그 미만의 데이터 파일의 시퀀스들의 표준 편차. 이러한 규격은 서브 시퀀스를 선행하지 않는 시간에 있기 때문에 데이터 파일의 제 1 시퀀스 상에서 이용하는데 도움이 되지 않음을 이해해야 한다.

- 이전의 서브 시퀀스 또는 표준 편차에 기초한 서브 시퀀스 내의 값들의 반전

- 서브 시퀀스의 구조에 기초하여 계산된 적용가능한 구조들

- 관련 서브 시퀀스에 더하거나 또는 감산하는, 파일 구조에 기초하여 랜덤하게 생성된 세그먼트들

상기 규격은 일련의 마스크 구조들을 생성하기 앞서 적용될 수 있다. 다른 방법으로, 관련 바이트 값들은 데이터 송신 동안에 계산될 수 있다. 하나의 형태에서, 512 개의 랜덤하게 생성된 구조들 또는 마스크 구조들을 직접 액세스 메모리(25)에 기억한다. 이러한 구조들을 300 바이트 시퀀스 내에 256 또는 이 보다 더 많은 널(null) 값을 가질 수도 있는 데이터 파일들의 시퀀스들에 적용할 수 있다. 이는 많은 소프트웨어 애플리케이션들의 이진 파일들의 일부를 형성하는 헤더들에서 공통적이다. 또한, 이러한 랜덤하게 생성된 구조들을 높은 반복 레벨들을을 가지는 다른 포맷들에 적용할 수 있다.

서브 시퀀스 상의 데이터 변환에 후속하여, 서브 시퀀스를 다시 테스트하여, 300 바이트 시퀀스 내에 256 고유의 값들이 있는지 여부를 식별한다(단계 210). 만일 256 개의 고유의 값들이 있지 않으면, 서브 시퀀스에 추가적인 마스크를 적용할 필요가 없으며, 300 바이트의 임계값이 낮아지고 프로세스가 더 작은 서브 시퀀스 상에 반복된다. 하나의 바람직한 실시형

태에서, 이 임계값은 300 8 비트 값(바이트) 보다 작게 조사하기 위하여 152 7 비트 값 또는 77 6 비트 값으로 일시적으로 낮아질 수 있다. 그 후, 임계값은 다음 서브 시퀀스에 대하여 300 바이트로 상승될 수 있다. 이하에서는, 이를 더 상세히 설명한다.

랜덤 파일의 부가는 256 바이트 세그먼트 내에 256 개의 고유의 값을 생성하지만 이것이 약 10% 의 경우에서 발생할 수도 있다. 일단 적절한 랜덤 파일 구조를 적용하면 300 바이트를 초과하지 않는 데이터 세그먼트 내에 256 개의 고유의 값이 있음을 예상할 수 있다. 임의의 경우에, 데이터 변환의 취지는 데이터 그룹 내의 고유의 데이터 값들의 주파수를 증가시키려는 것이다.

본 발명은 데이터 그룹 내의 300 개의 데이터 값의 인덱스를 계산한다.

이 인덱스는 바람직하기로는 직접 액세스 메모리(25)내의 데이터 어레이(90)에 기억된다. 300 데이터 값의 인덱스를 먼저 2 비트를 이용하여 생성한다. 256 개의 고유의 값들이 300 바이트 데이터 그룹 내에서 식별되면, 비트값 "01"이 인덱스에 기록된다(단계 225).

마스크를 서브 시퀀스에 적용한 경우에, 마스크 또는 데이터 세트 식별자는 인덱스에 기록된다(단계 230). 이 마스크 식별자는 바람직하기로는 0 과 65,536 사이의 마스크 값을 식별하는 16 비트 값일 수 있다. 마스크 식별자 내의 값 0 은, 서브 시퀀스에 마스크를 적용하지 않았거나 또는 널 마스크를 서브 시퀀스에 적용하였다는 사실을 나타낸다. 널 데이터 세트를 서브 시퀀스에 적용하는 경우에, 데이터 변환이 후속하는 서브 시퀀스는 데이터 변환 이전에 서브 시퀀스에 실질적으로 동일하다.

본 발명의 방법에서의 다음 단계는 235 개의 임시 위치 임덱스를 생성하는 것이다.

임시 위치 인덱스 생성 방법은, 256 개의 고유값들이 300 바이트 데이터 그룹으로부터 추출되는 경우에, 데이터 그룹 내의 제 1 바이트로서 개시하며, 256 개의 고유의 값들을 식별할 때 까지 데이터 그룹 내의 후속 바이트들을 조사한다. 조사되는 특정 값은 데이터 그룹 또는 이전의 데이터 그룹 내의 데이터 값이 먼저 발생하면, "1" 비트 값을 임시 인덱스에 부가한다. 한편, 조사되는 데이터 값이 초기의 데이터 값의 반복인 경우에, "0" 비트 값을 인덱스에 기록한다. 인덱싱 방법은 256 개의 "1" 비트가 인덱스에 기록 될 때 보다 빨리 종료한다.

임시 인덱스는 결과적인 압축 비트 스트림내의 데이터 그룹의 각각의 데이터 값의 배치 및 식별을 용이하게 한다. 이러한 인덱스 내의 "1" 값들의 개수는 얼마나 많은 비트 값들이 사용되는 지를 나타낸다. 예를 들어, 256 개의 "1" 값들이 임시 인덱스 내의 283 개의 입력 이후에 임시 인덱스에서 발생하는 경우, 이는 서브 시퀀스의 283 바이트 내에 256 개의 고유의 바이트 값들이 존재함을 나타낸다.

만일 300 바이트 데이터 그룹 내에 256 이상의 값이 있으면 인덱스의 첫번째 2 비트는 이미 "01"로 설정된다. 임시 인덱스를 간단히 메인 인덱스에 더할 수 있지만, 이러한 정보를 기억하는 효율적인 방식이 더 많이 존재한다. 서브 시퀀스에 나타나는 "1" 값의 개수는 이미 알려져 있다. "1" 값이 나타나는 순서를 무시하면, 고유의 바이트 값들의 경우의 수를 기록하기만 하면 된다.

임시 인덱스 자체를 기록하기 보다는, 240 개의 위치 인덱스를 생성하고, 이 위치 인덱스를 메인 인덱스에 기록하는 것이 바람직하다. 이 임시 인덱스는 300 바이트 서브 시퀀스에 대하여 44 "0" 값이 후속하는 256 "1" 값을 포함하면, 이것에 위치 인덱스 "0" 이 할당될 수 있다. 44 "0" 값 및 256 "1" 값을 300 바이트 데이터 그룹 내에서 배열할 수 있는 방법들의 개수는 ${}^{n}C_{r}$ 이다. 이는 300 개의 값 내에서 300!/256!.44! 의 허용가능한 조합이 1.34×10^{53} 과 동일함을 의미하며, 여기서 256 "1" 값과 44 "0" 값이 존재한다.

1.34×10^{53} 의 최대 위치 인덱스 값은 값 2^{177} 보다 작고, 그 값은 표현을 위하여 177 비트를 요구한다.

이는 300 비트의 실제 임시 인덱스를 기억하기 보다는, 임시 인덱스에서 적어도 256 "1" 값이 있다는 사실의 이점을 취함으로써, 대신에 위치 인덱스를 177 비트 또는 22.125 바이트에 기록할 수 있음을 의미한다.

압축 뿐만 아니라 압축 해제를 가능하게 하기 위하여 데이터 그룹 내의 데이터 값들의 순서를 기록하는 것이 중요하다. 이는 245 순열 인덱스를 생성하고, 이러한 순열 인덱스를 메인 인덱스에 기록함으로써 달성된다.

순열 인덱스 계산은 256 개의 고유값들을 순서화할 수 있는 방법들의 개수 또는 반복 없이 256 값들의 순열에 기초한다. 제 1 값에 대하여, 256 개의 가능성이 존재하며, 제 2 값에 대하여는, 255 개의 가능성이 존재하며, 제 3 값에 대하여, 254 개의 가능성이 존재한다. 이는 "256 팩토리얼" 로서 지칭되는 256! 로서 표현된다. 따라서, 256 개의 고유의 값의 허용가능한 순열 개수는 8.57×10^{506} 이다. 이 값은 2^{1684} 가 8.57×10^{506} 보다 큰 8.6×10^{506} 과 동일하므로, 1,584 비트로표현될 수 있다. 1,684 비트는 210.5 바이트와 동일하다.

시퀀스 0, 1, 2, 3, 4 ..., 254, 255 는 순열 번호 1 로서 표현되며, 시퀀스 255, 254, 253 ... 3, 2, 1, 0 는 순열 번호 8.57×10^{506} 에 의해 표현된다.

순열 인덱스는 메인 인덱스에 기록된다. 다음으로, 메인 인덱스는 계산된 서브 시퀀스 내의 고유의 비트 값들을 나타내는 데이터 값을 포함한다. 이는 적용된 마스크를 나타내는 16 비트에 의해 후속하며, 순열 인덱스를 나타내는 1,684 비트가 후속하며, 위치 인덱스를 나타내는 177 비트가 후속하는, 비트 값 "01" 일 수 있다.

충분한 길이의 서브 시퀀스를 획득하기 위하여 데이터 파일에 남겨 있는 충분한 비트들의 없거나, 또는 남겨진 충분한 고유의 값들이 없다는 포인트에 도달하는 경우에, 인덱스는 출력 파일에 기록된다(단계 250).

출력 파일은 바람직하기로는 파일 타입을 식별하기 위하여 3 개의 초기 바이트를 포함한다. 파일 타입 식별자에 후속하는 추가적인 2 개의 바이트는 본 발명의 방법이 65,536 회 반복의 최대값으로 특정 데이터 파일에 걸쳐서 동작하는 회수를 나타낸다.

이러한 5 바이트에 후속하여, 데이터 어레이(90)에 기억되는 인덱스를 출력 파일에 더한다. 그 후, 인덱스에 인덱스들로부터 두드러지거나, 또는 데이터 파일에 남아있는 충분한 비트 값 또는 고유의 값들의 부족으로 인하여 임의의 값들을 더한다.

대부분의 경우에, 바디가 후속하는 5 헤더 바이트가 존재하고 출력 파일의 말단에 비압축된 형태로 전부 기록된 63 또는 그 미만의 비트 값이 있음을 예상한다. 출력 파일의 바디(body)는 스트리밍 방식으로 추출을 용이하게 하기 위하여 연속적으로 기록되는 인덱스들의 집합인 것이 바람직하다.

도 2 에서 상술한 바와 같이, 본 발명의 방법의 복수의 반복에 후속하는 데이터 파일에 남아있는 바이트가 300 바이트가 아닌 경우가 존재할 수 있거나, 또는 300 바이트의 서브 시퀀스가 존재할 수 있으며, 여기서는 256 개의 고유의 값이 존재하지 않으며 추가적인 마스크를 적용할 수 없다. 도면 부호 260 에서 나타낸 바와 같이, 하나의 바람직한 형태에서는, 데이터 파일로부터 검색되는 서브 시퀀스의 크기를 감소시킬 수 있다.

도 3 을 참조하면, 데이터 파일은 그 데이터 파일에 적어도 152 바이트가 남아있는 지를 식별하기 위하여 검사된다(단계 315).

만일 데이터 파일에 남아있는 152 7 비트 값을 포함하는 적어도 133 바이트가 존재하는 경우에, 152 7 비트 값내의 고유의 값의 개수가 카운트된다(단계 310). 그 후, 고유의 값들의 개수를 예를 들어 128 의 임계 수에 대하여 검사한다(단계 315). 133 바이트의 서브 시퀀스 내에 고유값들이 충분하지 않은 경우에, 적용가능한 마스크는 도 2 의 단계 215 및 220 와 유사한 방식으로 식별되고(단계 340), 적용된다(단계 345).

일단 고유값들의 임계 개수가 데이터 파일 내의 152 7 비트 값에서 식별되면, 비트 시퀀스 "10" 는 인덱스에 기록되고(단계 350), 그 방법은 도 2 의 단계 230 에서 전방으로 표시된 단계들로 진행한다.

데이터 파일에서 처리될 152 7 비트 값이 남아 있지 않거나 또는 128 고유값들이 152 7 비트 서브 시퀀스 내에 배치될 수 없고 단계 355 에서 표시된 바와 같이 추가적인 마스크들을 적용할 수 없는 경우에, 그 방법은 도 4 에 나타낸 단계로 진행한다. 도 4 에 나타낸 바와 같이, 조사 중인 데이터 파일의 비트 그룹들의 개수는 77 6 비트 값으로 감소된다. 데이터 파일에 77 6 비트 값이 남아 있는 경우에(단계 405), 77 6 비트 값들내의 고유의 값들의 개수가 카운트된다(단계 410).

고유값들의 개수는 64 개의 임계값에 대하여 조사된다(단계 415). 만일 77 6 비트 값 내의 64 개의 고유값 보다 작으면, 이 방법은 마스크가 적용가능한지 여부를 설정한다(단계 420). 만일 마스크를 적용할 수 있다면, 그 마스크를 적용한다(단계 425). 이러한 마지막 2 개의 단계 420 및 425 는 도 2 의 단계 215 및 220 와 도 3 의 단계 340 및 345 와 유사하다.

77 6 비트 서브 시퀀스에 64 개의 고유값들이 존재하는 경우에, 값 "11" 이 인덱스에 기록된다(단계 430). 그 후, 제어는 도 2 에서 단계 230 이후로 리턴한다.

만일 데이터 파일에서 처리할 77 6 비트 값들이 남아 있지 않거나, 또는 77 6 비트 값들의 시퀀스 내에 64 개의 고유 값들이 존재하지 않는 경우에, 비트 값 "00" 은 인덱스에 기록되며(단계 435), 인덱스는 도 2 에 나타낸 단계 250 와 동일한 방식으로 출력 파일에 기록되며, 데이터 파일 내의 나머지 바이트는 출력 파일에 기록된다.

조사 중인 바이트 수에 따라서, 도 2 의 단계 245 에 나타낸 순열 인덱스에 약간의 변경이 요구됨을 이해하여야 한다. 152 7 비트 그룹 내에 128 개의 고유의 데이터 값들이 존재하는 경우에, 위치 인덱스는 5.48×10^{27} 과 동일한 152!/128!.24! 일 수 있다. 이는 $2^{93} = 9.9 \times 10^{27}$ 로서 93 비트로 표현될 수 있다.

77 6 비트 그룹에 걸쳐서 64 개의 고유의 값이 존재하는 경우에, 인덱스는 77!/64!/.13! 이 된다. 이는 1.84×10^{14} 인 전자의 값 보다 큰 $2^{48} = 2.81 \times 10^{14}$ 로서 42 비트로 표현될 수 있다.

이와 유사하게, 조사 중인 바이트 수에 따라서, 도 2 의 단계 245 에 나타낸 순열 인덱스에 약간의 변경이 요구된다. 128 개의 값들에 대한 순열은 128! 또는 3.86×10^{215} 이다. 이는 $2^{717} = 6.89 \times 10^{215}$ 로서 표현하기 위하여 717비트를 요구한다.

64 개의 값들에 대한 순열은 64! 또는 1.27×10^{89} 이다. 이는 $2^{296} = 1.27 \times 10^{89}$ 로서 296 비트에 의해 표현될 수 있다.

도 5 는 377(8 비트 그룹) 바이트, 350 바이트, 320 바이트, 300 바이트, 152 7 비트 그룹 및 77 6 비트 그룹의 데이터 그룹 크기에 대한 예상 결과의 테이블을 나타낸다. 이 테이블에는 편차 포함 효과의 표시가 포함된다. 이를 아래에서 설명한다.

압축해제(decompression)는 상기 절차를 반전시키는 간단한 작업이다. 인덱스 값은 첫 번째로부터 마지막 (256 번째)까지의 각각의 값의 범위를 나타낸다. 이 범위는 관련된 값을 제공한다. 인덱스는 헤더와 함께 재구성을 위해 사용될 수 있다. 모든 구성요소들이 함께 패키지 되기 때문에, 스트리밍을 사용할 수 있음을 생각할 수 있다.

반복된 값들의 배치의 인덱싱은 효과적인 방법이 더 많이 있는 경우들에서 세그먼트들에 대하여 "0" 및 "1" 값의 스트링으로부터 변경될 수 있다. 예를 들어, 만일 단지 하나 또는 2 개의 반복된 값이 존재하는 경우에, 바이트 수는 257 또는 258 개일 수 있다. 257 번째 및 258 번째 비트를 사용하기 보다, 최초 및 마지막 바이트가 그 세그먼트에 대하여 고유한 것임이 알려져 있다. 따라서, 257 값의 경우에, 8 비트는 단일 반복된 값의 위치를 제공하고, 16 비트는 258 바이트 세그먼트의 경우에 양자의 반복된 값들의 위치를 제공한다.

이 방법은 모든 파일 타입 및 구조에 적용될 수 있다. PKWare 의 ZIP 프로덕트와 같은 툴에 의해 상당량 만큼 압축된 파일 타입 또는 구조에 대하여, 본 발명의 방법은 단일 경로 상에서 동일 레벨을 획득할 수 없다. 그러나, 이 방법은 동일한 파일에 걸쳐서 반복적으로 적용될 수 있고, 매번 이것의 크기를 감소시킨다. 회수 또는 반복 회수는 하드웨어 프로세싱 및/또는 사용자 요청 회수에 따른다.

압축해제는 전체 구성요소들이 알려져 있으므로 매우 빠르다. 압축해제는 압축이 랜덤한 데이터 구조들을 매칭할 것을 요구하므로 압축보다 더 빠르게 될 수 있다.

전체 인덱싱은 그 자체로 실제 데이터 내에 포함되기 때문에, 복수의 압축해제 루틴들은 동시에 수행될 수 있다.

다른 애플리케이션들은 소프트웨어 압축, 데이터 압축, 소니 플레이스테이션 2 및 마이크로소프트사의 X-박스 등과 같은 콘솔들 간의 온라인 게임, VoIP(Voice over IP) 및/또는 비디오 온 디멘드를 포함할 수 있다. 본 발명은 데이터 또는 2 진 정보가 기억, 송신되거나 또는 임의의 포맷으로 사용되는 어느 곳이든지 애플리케이션을 갖는다.

상기 설명은 코드의 300 바이트 세그먼트 내의 256 개의 고유의 값 또는 이 보다 작은 값에 기초한다. 이 선택된 크기는 단지 예시적인 목적을 위해서임을 이해해야 한다.

5 비트 이상의 데이터 그룹 또는 0 과 31 사이의 값은 이 방법을 이용하여 재구성될 수 있다. 랜덤하게 생성된 데이터 세트 또는 오버레이 파일들의 개수를 감소시키는 것은 3 및 4 비트 값을 또한 사용할 수 있음을 의미한다.

더 큰 절약은 설명한 8 비트(256 개의 값) 보다 더 큰 비트 값을 이용하여 행해질 수 있다. 예를 들어, 9 비트 값들이 압축되는 경우에, 압축시에 8 비트 압축에 의해 달성되는 추가적 이득이 존재한다.

절약 또는 압축은 값에 대하여 사용되는 비트 수에 따라서 증가한다. 256 개의 값(300 바이트 세그먼트)은 512 개의 값(600 바이트 세그먼트)과 같이 많이 압축되지 않는다. 차례로 512 값의 데이터는 1024 값과 같이 많이 압축되지 않는다. 계산이 파일 크기에 기초하여야 하므로, 상부의 결정가능 레벨이 존재하지 않는다.

상술한 300 바이트 방법을 이용하여, 이는 377 바이트 그룹으로 확장될 수 있다. 이는 이 명세서에서 설명하고 도 5 에 나타낸 바와 같이 바람직한 실시형태에 대한 최적의 레벨인 300 바이트에 대하여, 효과적인 범위가 256 내지 377 바이트 그룹임을 의미한다.

300 8 비트 그룹(바이트) 상의 변동, 152 7 비트 그룹 및 77 6 비트 그룹은 압축된 파일의 헤더에 표시될 수도 있다. 그 변동은 2 개의 부분으로 구성될 수도 있다. 이들은,

1. 세그먼트 크기 당 관련 비트 그룹의 개수의 표시. 8 비트 그룹에 대한 크기의 범위는 256 내지 377 이며, 이는 7 비트로 표현될 수도 있다. 7 비트 그룹에 대하여, 그 범위는 5 비트로 표시될 수 있고, 6 비트 그룹에 대하여, 그 범위는 4 비트로 표시될 수도 있다.

2. 추가적인 비트는 비트 그룹들 각각에서 편차가 발생하는 경우를 표시하기 위하여 상술한 것의 각각의 말단에 더해질 수도 있다. "0"은 아니오(No)를 나타내며, "1"은 예(Yes)를 나타낸다.

그 후, 헤더는 상기 값들을 나타내는 부가적인 19 비트를 포함할 수도 있다.

헤더에 대하여 편차 값이 허용되면, 그룹 기초에 의한 그룹 상에, 편차 값이 인덱스에 기록될 수도 있다.

예를 들어, 8 비트 그룹의 하나의 그룹 상의 디폴트는 300 개의 값을 가질 수 있지만, 각각의 세그먼트는 포함된 편차 값에 의해 표시된 바와 같이 256 과 377 값 사이에서 변할 수도 있다.

본 발명의 추가적인 실시형태들은 다중-반복 바이트 압축 향상을 포함할 수도 있으며, 이를 도 6 및 도 7 을 참조하여 설명한다.

기능적인 전자 파일들은 바이트 구조의 복수의 다른 카테고리에 포함될 수 있다. 이들은 샘플 2 칼라 비트맵으로부터 현재의 이용가능한 무손실 압축 알고리즘들 중 임의의 것을 이용하여 압축된 파일들로 변할 수 있다.

헤더 정보가 후속하는 2 칼라 비트맵에 대하여, 하나의 비트 값은 검은색을 의미하며, 또 다른 비트 값은 하얀색을 의미한다. 복수회 반복되기 때문에, 무손실 방식에서의 이러한 파일들의 압축은 간단하게 된다.

24 비트의 비트맵으로 이동하면, 패턴들을 식별하기가 더 어려워 지므로, 무손실 압축율은 더 간단한 비트맵 구조들 상에 있을 때와 같이 현재의 알고리즘을 이용하여 크게 되지 않는다.

여기서 설명된 프로세스는 간단한 패턴들을 24 비트 비트맵에 도입하고, 이는 표준 포토 타입 이미지에 대하여 현재 이용가능한 무손실 압축 알고리즘들 중 임의의 압축 알고리즘을 이용하여 무손실 압축을 허용하여 압축량을 현저하게 증가시킬 수 있다.

이를 달성하기 위하여, 오리지날 알고리즘은 도 6 의 610 에 나타낸 바와 같이 3 개의 구성요소로 분할되며, 결합되는 크기는 오리지날 이미지 보다 현저하게 크다.

그 후, 전체 3 바이트(24 비트)그룹은 620 에 나타낸 바와 같이 오름차순의 10진 값 순서으로 배열된다. 예를 들어, 236, 217, 67 은 67, 217, 236 으로 재배열된다. 바이트들의 배열의 변동은 호프만 구조를 이용하여 인덱스에 기록된다.

6 개의 허용가능한 오리지날 구조만이 존재하므로, 이들은 이하의 비트 인덱스를 이용하여 기록된다.

00 = 123

01 = 132

100 = 213

101 = 231

110 = 312

111 = 321

상기 숫자들 각각은, 이들의 정렬된 위치와 비교하는 경우에 바이트들의 오리지날 위치를 나타낸다.

이 인덱스는 일단 이미지가 625 에 도시된 바와 같이 완전하게 스캐닝되면 파일(파일 A) 에 기록된다.

가장 낮은 것의 전부 또는 각각의 그룹으로부터의 제 1 바이트 값들은 630 에 나타낸 바와 같이 별도의 파일(파일)에 기록된다.

바이트 값들이 정렬되어 있으므로, 제 2 바이트 값에서 제 1 바이트 값을 뺀 값은, 단계 635 에 나타낸 바와 같이 제 3 바이트 값에서 제 2 바이트 값을 뺀 값이 바로 후속하는 파일(파일 C)에 기록된다.

이는 3 개의 파일 즉, 파일 A, 파일 B 및 파일 C 를 생성한다. 파일 B 및 파일 C 의 결합된 전체는 오리지날 24 비트의 비트맵과 동일할 수 있다. 파일 A 는 이것인 바이트들의 인덱싱을 나타내므로 여분의 오버헤드 크기를 가진다.

그 후, 전체 3 개의 파일(A, B 및 C)이 무손실 알고리즘 또는 WINZIP(640)과 같은 프로덕트를 이용하여 하나의 파일로 압축되면, 결과적인 파일은 변경되지 않은 이미지 파일에 걸쳐서 이러한 툴들을 간단히 적용함으로써 달성되는 파일보다 평균적으로 25% 작다.

테스팅은 2.5% 하락인 최악의 경우의 시나리오 및 최상의 경우는 트루(true) 24 비트 비트맵의 화상 품질의 82% 를 가짐을 나타낸다. 동일한 이득은 JPEG 의 무손실 압축 모드를 이용하여 획득될 수 있다.

3 바이트 그룹핑을 이용하여 이 프로세스를 임의의 파일 구조에 적용하여 데이터를 유지한다. 또한, 이는 무손실 압축의 레벨을 더 높게 하기 위하여 4, 5, 6, 7, 8 등의 바이트 구조를 커버하도록 확대될 수도 있다.

비트맵 파일들이 이미지들을 디스플레이하는데 사용되기 때문에, Wave(.wav) 파일들은 소리를 내는데 사용된다. 다음으로, 압축 향상 프로세스의 추가적인 일례를 도 7 을 참조하여 Wave 포맷 파일에 관하여 설명한다. 비트맵 파일(2 비트, 4 비트, 8 비트, 10 비트, 12 비트, 16 비트, 24 비트 및 30 비트)의 서로 다른 레벨이 존재하므로, 각각은 더 많은 칼라 또는 품질을 제공하며, 동일한 것이 웨이브 파일들에 대하여 발생한다.

웨이브 파일들은 복수의 구성요소들을 이용하여 생성되고, 이들은 평균 샘플링 레이트, 샘플 레이트, 오디오 샘플 크기 및 채널 개수이다.

더 낮은 샘플링 레이트는 더 작은 파일을 가질수 있지만 더 작은 품질을 가진다. 또한, 모노 파일은 스테레오 파일보다 더 작다.

여기서 논의되는 웨이프 포맷은, 상업용 CD 에 최고의 품질의 스테레오 음악을 저장하는 경우에 사용되는 포맷이다. 이 포맷은 웨이프 포맷으로부터 CD 포맷으로 변환된다.

176.4 Kb/초의 평균 데이터 레이트, 44.1 ㎑의 샘플 레이트, 16 비트의 오디오 샘플 크기 및 2(스테레오) 채널을 가지는 웨이프 파일에 대하여, 이하의 것을 적용할 수도 있다.

파일 내의 전체 바이트 값들은 1 내지 n 으로 넘버링되는 것으로 표현되고 여기서 n 이 파일(노멀 오디어 파일에 대하여, 이는 50,000,000 의 순서를 가짐) 내의 최종 바이트이면, 도면 부호 725 에 나타낸 바와 같이 전체 우수 위치 바이트 값들은 하나의 파일(파일 A)에 기록되고, 도면 부호 730 에 나타낸 바와 같이 전체 기수 위치 바이트 값들은 별도의 파일(파일 B)에 기록된다. 예를 들어,

바이트 값	255	167	33	0	0	24	24	167	167
순서 값	1	2	3	4	5	6	7	8	9
파일 1(기수)	255	33	0	24	167				
파일 2(우수)	167	0	24	167					

도면 부호 240 에 나타낸 바와 같이 또 한번 2 개의 파일(파일 1 및 파일 2)을 무손실 알고리즘 또는 WINZIP 과 같은 프로덕트를 이용하여 하나의 파일로 압축하면, 결과적인 파일(도면 부호 650)은, 변경되지 않은 이미지 파일에 걸쳐서 간단히 이들 툴들에 적용함으로써 달성되는 파일보다 평균적으로 20% 작다.

테스팅은 압축된 파일의 크기가 10 % 추가적으로 저하한다는 최악의 경우의 시나리오, 및 최적의 경우에는 크기가 43% 저하한다고 제시한다.

추출/압축 해제가 간단하면, 파일 2 로 부터의 바이트들은, 2 개의 파일들의 관련 무손실 툴을 이용하여 압축해제된 이후에 파일 1 내의 바이트들 각각에 재삽입된다.

이하에서는 본 발명의 바람직한 형태들을 포함하여 본 발명을 설명하였다. 당업자에게 명백하게 되는 변화 및 변경은 본 발명의 범위 내에 포함되도록 의도되며, 첨부된 청구항들에 의해 규정된다.

(57) 청구의 범위

청구항 1.

소정의 길이 보다 크거나 또는 동일한 길이의 바이트들의 시퀀스를 포함하는 데이터 파일을 압축하는 방법으로서,

보조 기억 장치로 부터 상기 데이터 파일을 검색(retrieve)하는 단계;

상기 데이터 파일을 직접 액세스 메모리에 저장하는 단계;

상기 데이터 파일의 서브 시퀀스 내의 고유의 바이트 값들의 주파수를 계산하는 단계로서, 상기 서브 시퀀스는 상기 소정의 길이를 초과하지 않는 길이를 가지는 것인, 주파수 계산 단계;

상기 서브 시퀀스에 대한 인덱스를 생성하는 단계로서, 상기 인덱스는 상기 계산된 상기 서브 시퀀스 내의 고유의 바이트 값들의 주파수를 나타내는 데이터 값을 포함하는 것인, 인덱스 생성 단계;

미리 결정된 임계값 미만의 고유의 바이트 값들의 주파수를 가지는 상기 서브 시퀀스 상에서, 상기 서브 시퀀스 내의 고유의 바이트 값들의 주파수를 증가시키기 위하여 상기 서브 시퀀스에 데이터 변환을 적용하는 단계, 및 상기 인덱스에 상기 데이터 변환을 나타내는 데이터 값을 더하는 단계;

소정의 임계값 이상의 고유의 바이트 값들의 주파수를 가지는 상기 서브 시퀀스 상에서, 상기 인덱스에 상기 서브 시퀀스 내의 하나 이상의 고유의 값의 위치를 나타내는 데이터 값을 더하는 단계;

파일 타입 식별자를 가지는 출력 데이터 파일을 생성하는 단계; 및

상기 출력 데이터 파일에 상기 인덱스를 더하는 단계를 포함하는 데이터 파일의 압축 방법.

청구항 2.

제 1 항에 있어서, 상기 서브 시퀀스에 데이터 변환을 적용하는 단계는,

컴퓨터 메모리에서, 바이트 값들의 시퀀스를 가지며 변환 데이터 세트 식별자에 의해 식별된 복수의 변환 데이터 세트를 유지하는 단계;

컴퓨터 메모리로부터 상기 변환 데이터 세트들 중 하나의 변환 데이터 세트를 검색하는 단계로서, 상기 검색된 변환 데이터 세트는 상기 데이터 파일의 상기 서브 시퀀스의 길이와 실질적으로 동일한 길이를 가지는 것인, 변환 데이터 세트의 검색 단계; 및

상기 검색된 데이터 세트 내의 대응하는 바이트 값들에 기초하여 상기 서브 시퀀스 내의 각각의 바이트 값들에 데이터 변환을 적용하는 단계를 더 포함하는 데이터 파일의 압축 방법.

청구항 3.

제 2 항에 있어서, 상기 검색된 변환 데이터 세트들 중 하나 이상에 기초한 데이터 변환이 후속하는 상기 서브 시퀀스는 상기 데이터 변환 이전에 상기 서브 시퀀스와 실질적으로 동일한 것인 데이터 파일의 압축 방법.

청구항 4.

제 2 항에 있어서, 상기 변환 데이터 세트들 중 하나 이상은 랜덤하게 생성된 바이트 레이트들의 시퀀스를 포함하는 것인 데이터 파일의 압축 방법.

청구항 5.

제 2 항에 있어서, 상기 변환 데이터 세트들 중 하나 이상은 바이트 레이트들의 소정의 시퀀스를 포함하는 것인 데이터 파일의 압축 방법.

청구항 6.

제 2 항에 있어서, 상기 변환 데이터 세트들 중 하나 이상은 상기 데이터 파일의 상기 서브 시퀀스 이외의 상기 데이터 파일의 일부로부터 유도된 바이트 값들의 시퀀스를 포함하는 것인 데이터 파일의 압축 방법.

청구항 7.

제 2 항 내지 제 6 항 중 어느 한 항에 있어서, 상기 인덱스에 상기 서브 시퀀스에 적용된 상기 변환 데이터 세트의 상기 변환 데이터 세트 식별자를 더하는 단계를 더 포함하는 것인 데이터 파일의 압축 방법.

청구항 8.

제 2 항 내지 제 6 항 중 어느 한 항에 있어서, 상기 서브 시퀀스 내의 상기 하나 이상의 고유의 값의 위치를 계산하는 단계를 더 포함하는 것인 데이터 파일의 압축 방법.

청구항 9.

제 8 항에 있어서, 상기 서브 시퀀스 내의 상기 하나 이상의 고유의 값의 위치를 계산하는 단계는,

컴퓨터 메모리에 임시 위치 인덱스를 생성하는 단계;

상기 서브 시퀀스로부터 연속하는 바이트 값들을 검색하는 단계;

각각의 바이트 값의 검색시에, 상기 검색된 바이트 값이 고유의 바이트 값 또는 반복된 값인지를 결정하는 단계;

고유의 바이트 값의 검출시에, 2 개의 바이트 값 중 하나를 상기 임시 위치 인덱스에 더하거나 상기 2 개의 바이트 값 중 다른 하나를 상기 임시 위치 인덱스에 더하는 단계;

상기 하나 이상의 고유의 값의 위치를 나타내는 위치 인덱스를 상기 임시 위치 인덱스로부터 생성하는 단계; 및

상기 임시 위치 인덱스로부터 적어도 부분적으로 상기 하나 이상의 고유의 값들의 위치를 나타내는 데이터 값을 계산하는 단계를 더 포함하는 것인 데이터 파일의 압축 방법.

청구항 10.

제 9 항에 있어서, 상기 서브 시퀀스 내에 바이트 수는 상기 임시 위치 인덱스 내의 비트 수와 실질적으로 동일한 것인 데이터 파일의 압축 방법.

청구항 11.

제 9 항 또는 제 10 항에 있어서, 상기 위치 인덱스의 크기는 상기 임시 위치 인덱스의 크기 보다 작은 것인 데이터 파일의 압축 방법.

청구항 12.

제 9 항 또는 제 10 항에 있어서, 상기 서브 시퀀스 내의 고유의 바이트 값들의 순서를 나타내는 순열(permutation) 인덱스를 생성하는 단계; 및

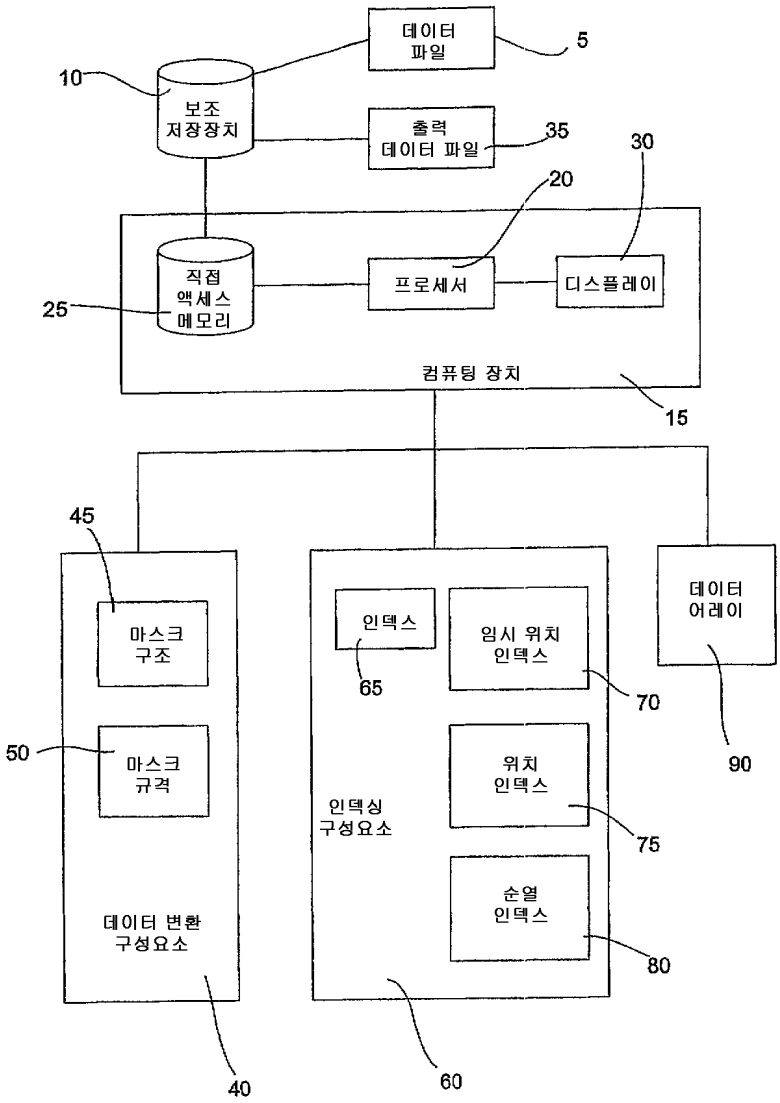
상기 위치 인덱스 및 상기 순열 인덱스로부터 하나 이상의 고유의 값들의 위치를 나타내는 데이터 값을 계산하는 단계를 더 포함하는 것인 데이터 파일의 압축 방법.

청구항 13.

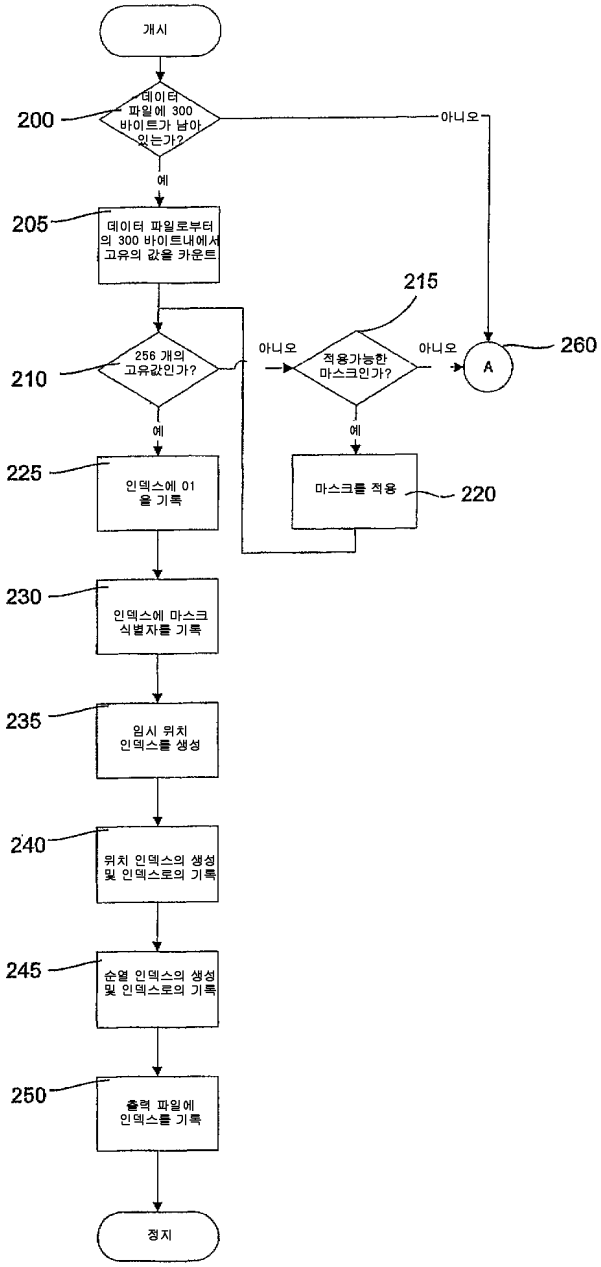
제 12 항에 있어서, 상기 하나 이상의 고유의 값들의 위치를 나타내는 상기 데이터 값을 형성하도록 상기 위치 인덱스 및 상기 순열 인덱스를 연결하는 단계를 더 포함하는 것인 데이터 파일의 압축 방법.

도면

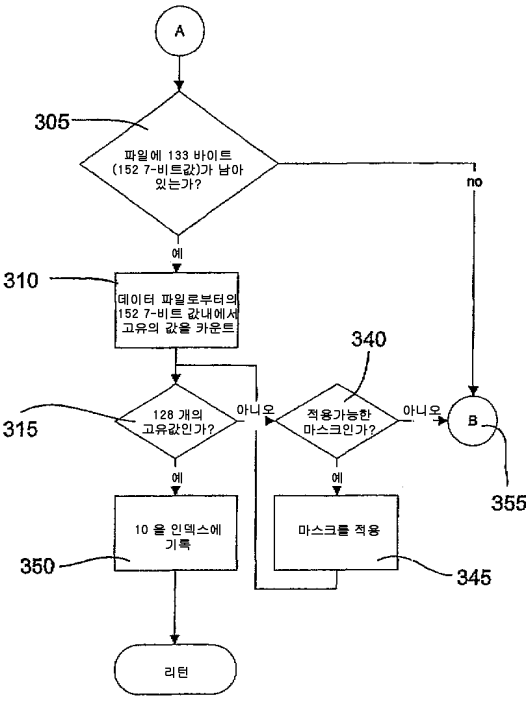
도면1


데이터 파일
5
10
보조 저장장치
출력 데이터 파일
35
30
20
직접 액세스 메모리
25
프로세서
디스플레이
컴퓨팅 장치
15
45
마스크 구조
50
마스크 규격
데이터 변환 구성요소
40
인덱스
65
임시 위치 인덱스
70
위치 인덱스
75
인덱싱 구성요소
순열 인덱스
80
60
데이터 어레이
90

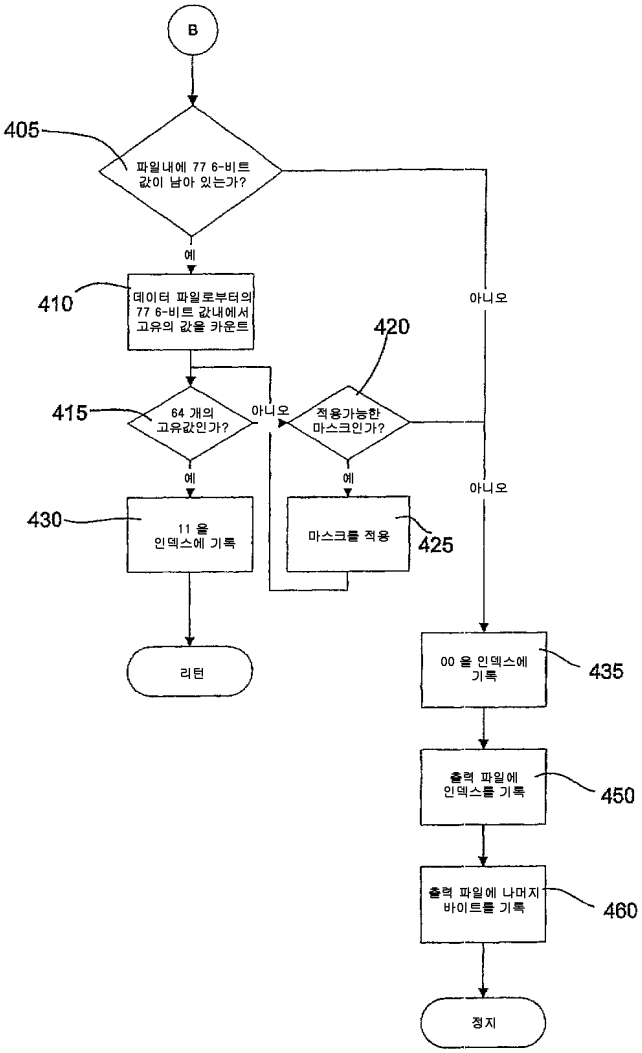
도면2


개시
200
데이터 파일에 300 바이트가 남아 있는가?
아니오
예
205
데이터 파일로부터의 300 바이트내에서 고유의 값을 카운트
215
210
256 개의 고유값인가?
아니오
적용가능한 마스크인가?
아니오
A
260
예
예
225
인덱스에 01을 기록
마스크를 적용
220
230
인덱스에 마스크 식별자를 기록
235
임시 위치 인덱스를 생성
240
위치 인덱스의 생성 및 인덱스로의 기록
245
순열 인덱스의 생성 및 인덱스로의 기록
250
출력 파일에 인덱스를 기록
정지

도면3


A
305
파일에 133 바이트 (152 7-비트값)가 남아 있는가?
no
예
310
데이터 파일로부터의 152 7-비트 값내에서 고유의 값을 카운트
340
315
128 개의 고유값인가?
아니오
적용가능한 마스크인가?
아니오
B
355
예
예
10 을 인덱스에 기록
마스크를 적용
350
345
리턴

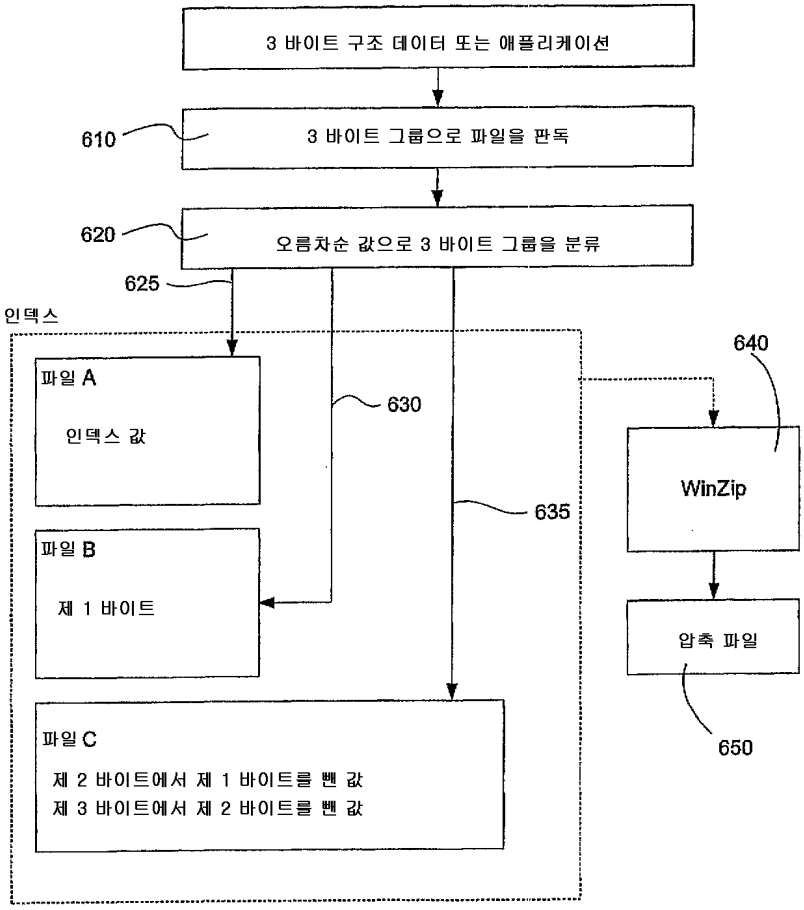
도면4


B
405
파일내에 77 6-비트 값이 남아 있는가?
예
아니오
410
데이터 파일로부터의 77 6-비트 값내에서 고유의 값을 카운트
420
415
64 개의 고유값인가?
아니오
적용가능한 마스크인가?
예
예
아니오
430
11 을 인덱스에 기록
마스크를 적용
425
리턴
00 을 인덱스에 기록
435
출력 파일에 인덱스를 기록
450
출력 파일에 나머지 바이트를 기록
460
정지

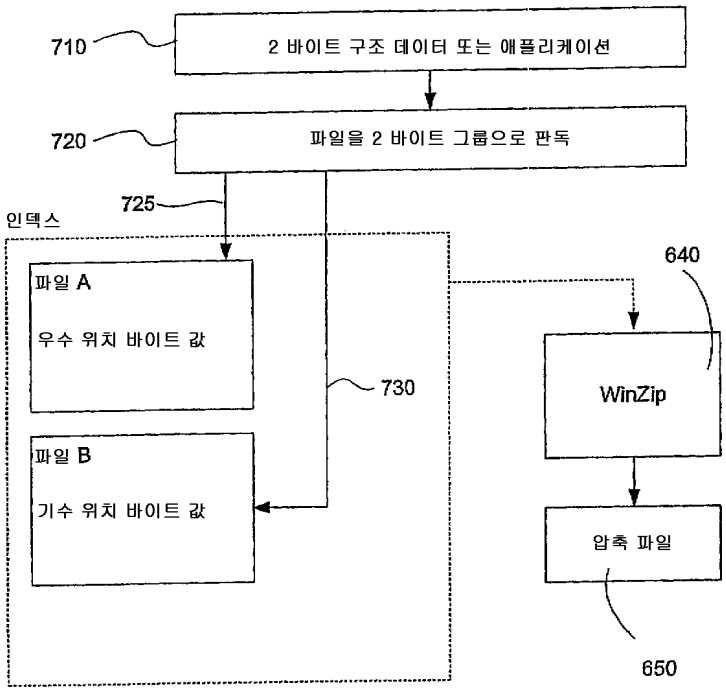
도면5

시퀀스 크기	바이트(비트) 수	시퀀스 크기를 나타내는 인덱스 비트	필요한 경우에 편차를 나타내는 인덱스 비트	적용된 마스크를 나타내는 비트	팩토리얼 값을 나타내는데 요구되는 비트	오더 조합을 나타내는 비트	총계	절약
377	256 (2048)	2	7	16	1684	337	2046	0.10%
377	256 (2048)	2		16	1684	337	2039	0.44%
350	256 (2048)	2	7	16	1684	290	1999	2.39%
350	256 (2048)	2		16	1684	290	1992	2.73%
320	256 (2048)	2	7	16	1684	227	1936	5.47%
320	256 (2048)	2		16	1684	227	1929	5.81%
300	256 (2048)	2	7	16	1684	178	1886	7.91%
300	256 (2048)	2		16	1684	178	1879	8.25%
152	(892)	2		16	717	93	828	7.17%
77	(384)	2		16	296	48	362	5.73%

도면6


3 바이트 구조 데이터 또는 애플리케이션
610
3 바이트 그룹으로 파일을 판독
620
오름차순 값으로 3 바이트 그룹을 분류
625
인덱스
파일 A
인덱스 값
630
635
파일 B
제 1 바이트
파일 C
제 2 바이트에서 제 1 바이트를 뺀 값
제 3 바이트에서 제 2 바이트를 뺀 값
640
WinZip
압축 파일
650

도면7


710
2 바이트 구조 데이터 또는 애플리케이션
720
파일을 2 바이트 그룹으로 판독
725
인덱스
파일 A
우수 위치 바이트 값
730
파일 B
기수 위치 바이트 값
640
WinZip
압축 파일
650