(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2018년06월07일
(11) 등록번호 10-1865264
(24) 등록일자 2018년05월31일

(51) 국제특허분류(Int. Cl.)
G06F 17/00 (2006.01) *G06F 17/30* (2006.01)
(21) 출원번호 10-2013-7032073
(22) 출원일자(국제) 2011년10월09일
심사청구일자 2016년09월02일
(85) 번역문제출일자 2013년12월03일
(65) 공개번호 10-2014-0038441
(43) 공개일자 2014년03월28일
(86) 국제출원번호 PCT/US2011/055532
(87) 국제공개번호 WO 2012/166190
국제공개일자 2012년12월06일
(30) 우선권주장
13/152,733 2011년06월03일 미국(US)
(56) 선행기술조사문헌
KR1020070114923 A

(73) 특허권자
마이크로소프트 테크놀로지 라이센싱, 엘엘씨
미국 워싱턴주 (우편번호 : 98052) 레드몬드 원 마이크로소프트 웨이
(72) 발명자
마이클 벤자민 에이
미국 워싱턴주 98052-6399 레드몬드 원 마이크로소프트 웨이 엘씨에이 - 인터내셔널 페이턴츠 마이크로소프트 코포레이션
(74) 대리인
제일특허법인

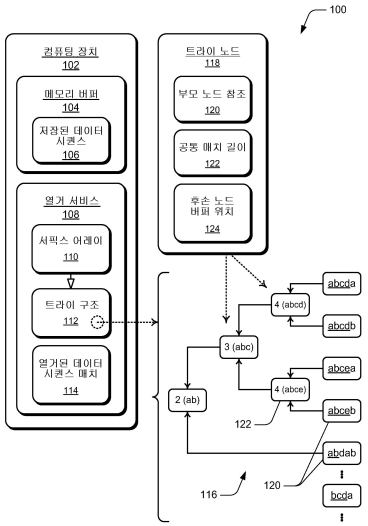
전체 청구항 수 : 총 20 항 심사관 : 이복현

(54) 발명의 명칭 **압축 매치 열거 기법**

(57) 요 약

압축 매치 열거의 실시예에서, 메모리 버퍼에 저장된 데이터 시퀀스를 나타내는 서픽스 어레이가 생성될 수 있다. 그 후 상기 서픽스 어레이는 트라이 구조로 변환될 수 있는데, 상기 트라이 구조는 서픽스 어레이의 위치에서 생성되기 때문에 메모리 버퍼에서 상기 서픽스 어레이를 덮어 쓴다. 상기 트라이 구조는 서픽스 어레이의 하나 이상의 서픽스를 각각 나타내는 노드를 포함하며, 연속하는 서픽스가 트라이 구조 내 기존 노드와 그룹지어지거나 트라이 구조의 새 노드로서 추가된다. 그 후 트라이 구조로부터 결정되는대로 데이터 시퀀스 매치가 열거될 수 있다.

대 표 도 - 도1



명 세 서

청구범위

청구항 1

메모리 버퍼에 저장된 데이터 시퀀스를 나타내는 서픽스 어레이(suffix array)를 생성하는 단계- 상기 서픽스 어레이는 상기 메모리 버퍼 내의 버퍼 위치들로서 생성되고, 상기 버퍼 위치들은 제각기의 버퍼 위치에서 데이터 시퀀스를 시작하는 데이터 스트링에 따라 정렬됨 -와,

상기 서픽스 어레이를 트라이 구조(trie structure)로 변환하는 단계- 상기 트라이 구조는 상기 서픽스 어레이의 위치에서(in-place) 생성됨에 따라 상기 트라이 구조는 상기 메모리 버퍼에서 상기 서픽스 어레이를 덮어씀 -와,

상기 트라이 구조의 노드에 의해 각각 표시되는 적어도 2개의 데이터 시퀀스 중에서 공통되는 서브스트링을 포함하는 데이터 시퀀스들에 대한 데이터 시퀀스 매치 정보를 열거(enumerate)하는 단계를 포함하는

방법.

청구항 2

제1항에 있어서,

상기 서픽스 어레이는 상기 서픽스 어레이의 연속하는 서픽스들로부터 상기 트라이 구조를 점진적으로 업데이트(incrementally update)함으로써 상기 트라이 구조로 변환되는

방법.

청구항 3

제1항에 있어서,

상기 트라이 구조는 각각이 상기 서픽스 어레이의 하나 이상의 서픽스를 나타내는 노드들을 포함하고, 각각의 연속하는 서픽스는 상기 트라이 구조 내 기존 노드와 그룹화되거나 상기 트라이 구조의 새 노드로서 추가되는

방법.

청구항 4

제3항에 있어서,

상기 서픽스 어레이의 서픽스가 상기 기존 노드와 데이터 시퀀스의 공통 매치 길이를 가질 경우 상기 서픽스는 상기 트라이 구조 내 상기 기존 노드와 그룹화되는

방법.

청구항 5

제1항에 있어서,

상기 트라이 구조는 각각이 부모 노드로의 참조(reference), 후손 노드(descendant nodes)의 데이터 시퀀스에 대한 공통 매치 길이, 및 가장-최근-순회된(most-recently-traversed) 후손 노드의 버퍼 위치를 포함하는 노드들을 포함하는

방법.

청구항 6

제5항에 있어서,

상기 후손 노드는 노드의 직계 자식 노드(immediate child node)를 포함하고, 상기 트라이 구조는,

상기 트라이 구조가, 각각이 적어도 2개의 직계 자식 노드를 갖는 하나 이상의 비-리프 노드(non-leaf node)를 포함한다는 것,

상기 후손 노드의 상기 데이터 시퀀스에 대한 공통 매치 길이가 최대라는 것, 및,

상기 트라이 구조 내 노드의 총 개수가 최소화되는 것에 기초하여 생성되는

방법.

청구항 7

제1항에 있어서,

상기 트라이 구조의 생성이 완료된 후에 상기 트라이 구조를 검색(search)하여 상기 데이터 시퀀스 매치 정보를 열거하는 단계를 더 포함하는

방법.

청구항 8

제1항에 있어서,

미래의 데이터 시퀀스 매치 정보를 위해 가장-최근-순회된 노드를 지정하도록 상기 가장-최근-순회된 노드의 노드 필드를 현재 위치로 업데이트하는 단계를 더 포함하는

방법.

청구항 9

컴퓨팅 장치로서,

열거 서비스(enumeration service)를 구현하기 위해 적어도 메모리 및 프로세서를 포함하고, 상기 열거 서비스는

메모리 버퍼에 저장된 데이터 시퀀스를 나타내는 서픽스 어레이를 생성하고,

상기 서픽스 어레이를 트라이 구조로 변환하며- 상기 트라이 구조는 상기 서픽스 어레이의 위치에서 생성됨에 따라 상기 트라이 구조는 상기 메모리 버퍼에서 상기 서픽스 어레이를 덮어쓰고, 상기 트라이 구조는 각각이 상기 서픽스 어레이의 하나 이상의 서픽스를 나타내는 노드들을 포함하고, 각각의 연속하는 서픽스는 상기 트라이 구조 내 기존 노드와 그룹화되거나 상기 트라이 구조의 새 노드로서 추가됨 -,

상기 트라이 구조의 노드에 의해 각각 표시되는 적어도 2개의 데이터 시퀀스 중에서 공통되는 서브스트링을 포함하는 데이터 시퀀스들에 대한 데이터 시퀀스 매치 정보를 열거하도록 구성되는

컴퓨팅 장치.

청구항 10

제9항에 있어서,

상기 열거 서비스는 또한 상기 서픽스 어레이의 서픽스가 상기 기존 노드와 데이터 시퀀스의 공통 매치 길이를 가질 경우 상기 서픽스를 상기 트라이 구조 내 상기 기존 노드와 그룹화하는

컴퓨팅 장치.

청구항 11

제9항에 있어서,

상기 서픽스 어레이는 상기 메모리 버퍼 내 버퍼 위치들의 어레이이며, 상기 버퍼 위치들은 제각기 버퍼 위치에서 데이터 시퀀스를 시작하는 데이터 스트링에 따라 알파벳 순으로 정렬되고,

상기 열거 서비스는 또한 상기 서픽스 어레이의 연속하는 서픽스들로부터 상기 트라이 구조를 점진적으로 업데이트함으로써 상기 서픽스 어레이를 상기 트라이 구조로 변환하도록 구성된

컴퓨팅 장치.

청구항 12

제9항에 있어서,

상기 트라이 구조의 노드들은 각각 부모 노드로의 참조, 후손 노드의 데이터 시퀀스에 대한 공통 매치 길이, 및 가장-최근-순회된 후손 노드의 버퍼 위치를 포함하는

컴퓨팅 장치.

청구항 13

제12항에 있어서,

상기 후손 노드는 노드의 직계 자식 노드를 포함하고,

상기 열거 서비스는 또한,

상기 트라이 구조가, 각각이 적어도 2개의 직계 자식 노드를 갖는 하나 이상의 비-리프 노드를 포함한다는 것,

상기 후손 노드의 상기 데이터 시퀀스에 대한 공통 매치 길이가 최대라는 것, 및,

상기 트라이 구조 내 노드의 총 개수가 최소화되는 것

에 기초하여 상기 트라이 구조를 생성하도록 구성된

컴퓨팅 장치.

청구항 14

제9항에 있어서,

상기 열거 서비스는 또한, 상기 트라이 구조의 생성이 완료된 후에 상기 트라이 구조를 검색하여 상기 데이터 시퀀스 매치 정보를 열거하도록 구성된

컴퓨팅 장치.

청구항 15

실행가능한 명령어를 저장하는 하나 이상의 컴퓨터 판독가능 저장 매체 장치로서,

상기 명령어의 실행에 응답하여, 컴퓨팅 장치가

메모리 버퍼에 저장된 데이터 시퀀스를 나타내는 서픽스 어레이를 생성하고,

상기 서픽스 어레이를 트라이 구조로 변환하며- 상기 트라이 구조는 상기 서픽스 어레이의 위치에서 생성되어 상기 트라이 구조는 상기 메모리 버퍼에서 상기 서픽스 어레이를 덮어쓰고, 상기 트라이 구조는 각각이 상기 서픽스 어레이의 하나 이상의 서픽스를 나타내는 노드들을 포함하고, 각각의 연속하는 서픽스는 상기 트라이 구조 내 기존 노드와 그룹화되거나 상기 트라이 구조의 새 노드로서 추가됨 -,

상기 트라이 구조를 검색하여, 상기 트라이 구조의 노드에 의해 각각 표시되는 적어도 2개의 데이터 시퀀스 중에서 공통되는 서브스트링을 포함하는 데이터 시퀀스들에 대한 데이터 시퀀스 매치 정보를 열거하는

하나 이상의 컴퓨터 판독가능 저장 매체 장치.

청구항 16

제15항에 있어서,

실행가능한 추가의 명령어를 더 저장하되, 상기 추가의 명령어의 실행에 응답하여 상기 컴퓨팅 장치는 상기 서픽스 어레이의 연속하는 서픽스들로부터 상기 트라이 구조를 점진적으로 업데이트함으로써 상기 서픽스 어레이를 상기 트라이 구조로 변환하는

하나 이상의 컴퓨터 판독가능 저장 매체 장치.

청구항 17

제15항에 있어서,

실행가능한 추가의 명령어를 더 저장하되, 상기 추가의 명령어의 실행에 응답하여 상기 컴퓨팅 장치는 상기 트라이 구조 내 노드들의 노드 필드를 업데이트하고, 각 노드의 노드 필드는 부모 노드로의 참조, 후손 노드의 데이터 시퀀스에 대한 공통 매치 길이, 및 가장-최근-순회된 후손 노드의 버퍼 위치를 포함하는

하나 이상의 컴퓨터 판독가능 저장 매체 장치.

청구항 18

제17항에 있어서,

실행가능한 추가의 명령어를 더 저장하되, 상기 추가의 명령어의 실행에 응답하여 상기 컴퓨팅 장치는

상기 후손 노드가 노드의 직계 자식 노드를 포함한다는 것,

상기 트라이 구조가, 각각이 적어도 2개의 직계 자식 노드를 갖는 하나 이상의 비-리프 노드를 포함한다는 것,

상기 후손 노드의 상기 데이터 시퀀스에 대한 공통 매치 길이가 최대라는 것, 및,

상기 트라이 구조 내 노드의 총 개수가 최소화되는 것

에 기초하여 상기 트라이 구조를 생성하는

하나 이상의 컴퓨터 판독가능 저장 매체 장치.

청구항 19

제15항에 있어서,

실행가능한 추가의 명령어를 더 저장하되, 상기 추가의 명령어의 실행에 응답하여 상기 컴퓨팅 장치는 상기 트라이 구조의 생성이 완료된 후에 상기 트라이 구조를 검색하여 상기 데이터 시퀀스 매치 정보를 열거하는

하나 이상의 컴퓨터 판독가능 저장 매체 장치.

청구항 20

제15항에 있어서,

실행가능한 추가의 명령어를 더 저장하되, 상기 추가의 명령어의 실행에 응답하여 상기 컴퓨팅 장치는 상기 메모리 버퍼 내의 버퍼 위치들로서 상기 서픽스 어레이를 생성하고, 상기 버퍼 위치들은 제각기의 버퍼 위치에서 데이터 시퀀스를 시작하는 데이터 스트링에 따라 정렬되는

하나 이상의 컴퓨터 판독가능 저장 매체 장치.

발명의 설명

기 술 분 야

배 경 기 술

[0001] 컴퓨팅 장치는 메모리 및 그 밖의 다른 컴퓨팅 장치를 덜 사용하여 데이터를 저장, 프로세싱, 유지(maintain) 및/또는 통신하기 위해 데이터 바이트를 압축하기 위한 다양한 데이터 압축 기법을 수행한다. 종래의 데이터 압축 기법은 프로세싱 자원 관점에서 비효율적이거나 및/또는 데이터를 압축하기 위해 데이터 매치(data match)(가령, 반복되는 바이트 시퀀스)를 찾는 데 신뢰할 만하지 않을 수 있다. 예를 들어, 임의의 LZ77 압축 구현, 가령, LZX 및 LZMA에 대한 핵심적인 해결과제는 가장 작은 압축된 데이터를 생성하는 데이터 매치를 효율적이고 신뢰할만한 수준으로 찾는 것이다.

[0002] 다양한 LZ77 압축 알고리즘은 반복되는 바이트 시퀀스를 결정하고 (거리(distance), 길이(length)) 쌍을 이용해

매치를 인코딩하는 것이다. 압축 알고리즘이 버퍼를 처음부터 끝까지 처리하므로, 각각의 위치에서 가능한 매치는 버퍼의 현재 위치에서의 바이트와 동일한 버퍼의 이전 위치에서의 바이트 시퀀스이다. 버퍼로 되돌아 가는 거리가 짧을수록 보다 적은 비트로 인코딩될 수 있고, 보다 긴 길이일수록 더 많은 데이터를 커버한다. 거리(distance)는 버퍼 내 데이터 매치들 간 바이트 거리를 나타내고, 길이(length)는 매치하는 데이터 바이트의 수를 나타낸다. 우수한 압축률을 얻기 위해, 알고리즘은 버퍼 내 각각의 위치에 대해 각각의 가능한 길이에 대해 최단 거리를 열거(enumerate)할 수 있어야 한다. 빨라지기 위해, 알고리즘은 그들의 길이에 대해 최단 거리가 아닌 매치를 열거하는 데 시간을 써서는 안된다. 예를 들어, 버퍼 내 일부 위치에서, 가능한 매치의 전체 집합이 (거리=50, 길이=3), (100, 4), (120, 3), (150, 4), (200, 5)일 수 있다. 알고리즘은 (50, 3), (100, 4), 및 (200, 5)만 열거할 것인데, 왜냐하면 나머지 두 (120, 3) 및 (150, 4)는 적어도 같은 길이(가령, 3 및 4의 길이)이지만 더 짧은 거리인 매치로 대체되기 때문이다. 최적화 측면에서, 알고리즘은 2개의 최적화 기준이 더 긴 길이 및 더 짧은 거리인 매치의 파레토 프론티어(Pareto frontier)를 빠르게 열거해야 한다.

[0003] LZX 알고리즘은 스플레이 트리(splay tree)를 이용하여 압축 매치를 결정하고 문제를 푼다. 스플레이 트리는 새로운 요소가 루트(root)에 삽입되는 이진 트리(binary tree)이다. 이는 알고리즘이 매치를 결정하기 위해 트리를 검색할 때 가장 최근(most-recent), 따라서 최단 거리 매치를 먼저 만나게 된다는 특성을 제공한다. 알고리즘은 트리가 불균형이 될 때, 가령, 스트링이 알파벳 순으로 삽입될 때 형편없게 동작하며, 실제로, LZX 알고리즘은 큰 매치 히스토리로 형편없게 확장된다.

[0004] LZMA 알고리즘은 해시 체인(hash chain), 이진 트리, 및 패트리샤 트라이(Patricia tries)의 변형을 이용하여 압축 매치를 결정하고 문제를 풀 수 있다. 또한 트리의 각각의 노드에서 가장 최근 삽입된(most-recently-inserted) 데이터 스트링의 일부 개념을 이용해 수정되는 경우 문제를 풀 수 있는 공간 효율적인 트리 구현 기법이 존재한다. 그러나 이들 기법은 트리의 루트로부터 계층구조(hierarchy)의 하향으로 하위 레벨 노드로 트리 구조를 순회하도록 구현되며, 가장 최근 매치(most recent match)가 또한 긴 매치(long match)일 때 차선이다.

발명의 내용

해결하려는 과제

과제의 해결 수단

[0005] 개요

[0006] 이 개요는 압축 매치 열거의 단순화된 개념을 소개하며, 개념들은 이하의 구체적인 내용 및/또는 도면에서 추가로 상세히 기재된다. 이 개요는 청구되는 대상의 필수 특징을 설명하기 위한 것이 아니며, 청구되는 사항의 범위를 결정하거나 제한하려 사용되는 것도 아니다.

[0007] 리프-투-루트 트라이(leaf-to-root trie) 구조를 이용해 저장된 데이터 시퀀스에서 모든 데이터 매치 가능성을 열거하기 위해 압축 매치 열거가 구현된다. 실시예에서, 메모리 버퍼에 저장된 데이터 시퀀스를 나타내는 서픽스 어레이가 생성될 수 있다. 그 후 상기 서픽스 어레이는 트라이 구조로 변환될 수 있는데, 상기 트라이 구조는 서픽스 어레이의 위치에서(in-place) 생성되기 때문에 상기 메모리 버퍼에서 서픽스 어레이를 덮어 쓴다. 상기 트라이 구조는 서픽스 어레이의 하나 이상의 서픽스를 각각 나타내는 노드를 포함하고, 각각의 연속하는 서픽스는 상기 트라이 구조 내 기존 노드와 그룹지어지거나 상기 트라이 구조의 새 노드로서 추가된다. 서픽스 어레이의 서픽스가 기존 노드와 데이터 시퀀스의 공통 매치 길이를 가질 때 상기 서픽스는 트라이 구조 내 상기 기존 노드와 그룹지어질 수 있다. 그 후 상기 데이터 시퀀스 매치가 트라이 구조로부터 결정되는대로 열거될 수 있다.

[0008] 또 다른 실시예에서, 서픽스 어레이는 메모리 버퍼 내 버퍼 위치의 어레이이며, 상기 버퍼 위치는 각자의 버퍼 위치에서의 데이터 시퀀스를 시작하는 데이터 스트링에 의해 알파벳 순으로 정렬된다. 상기 서픽스 어레이는 서픽스 어레이의 연속하는 서픽스로부터 트라이 구조를 점진적으로 업데이트함으로써 상기 트라이 구조로 변환될 수 있다. 트라이 구조의 노드 각각은, 부모 노드로의 참조, 후손 노드의 데이터 시퀀스에 대한 공통 매치 길이(가령, 한 노드의 직계 자식 노드를 포함하도록), 및 가장-최근-순회된 후손 노드의 버퍼 위치를 포함한다. 상기 트라이 구조는 다음을 기초로 생성될 수 있다: 트라이 구조는 적어도 2개의 직계 자식 노드를 각각 갖는 하

나 이상의 비-리프 노드를 포함함, 후손 노드의 데이터 시퀀스에 대한 공통 매치 길이는 최대화됨, 트라이 구조 내 노드의 총 개수는 최소화됨.

도면의 간단한 설명

[0009] 압축 매치 열거의 실시예가 다음의 도면을 참조하여 기재된다. 동일한 부호가 전체적으로 도면에서 나타난 유사한 특징, 구성요소를 참조하는 데 사용될 수 있다.

도 1은 압축 매치 열거의 실시예가 구현될 수 있는 예시적 시스템을 도시한다.

도 2는 하나 이상의 실시예에 따라 트라이 구조 및 압축 매치 열거의 일례를 도시한다.

도 3은 하나 이상의 실시예에 따라 압축 매치 열거의 예시적 방법(들)을 도시한다.

도 4는 압축 매치 열거의 실시예를 구현할 수 있는 예시적 장치의 다양한 구성요소를 도시한다.

발명을 실시하기 위한 구체적인 내용

[0010] 저장된 데이터 시퀀스 내 모든 데이터 매치 가능성을 열거하도록 구현된 압축 매치 열거(compression match enumeration)의 실시예가 기재된다. 가령, LZ77 압축 알고리즘의 경우 사용될 수 있는 압축 매치 열거가 데이터 매치를 열거하기 위해 리프-투-루트(leaf-to-root) 트라이 구조(trie structure)를 이용한다. 실시예에서, 메모리 버퍼에 저장된 데이터 시퀀스를 나타내는 서픽스 어레이(suffix array)가 생성된다. 그 후 상기 서픽스 어레이는 트라이 구조로 변환되는 데, 상기 트라이 구조는 서픽스 어레이의 위치에서(in-place) 생성되기 때문에, 메모리 버퍼에서 상기 서픽스 어레이를 덮어 쓴다. 상기 서픽스 어레이의 연속하는 서픽스들로부터 트라이 구조를 점진적으로 업데이트(incrementally update)함으로써, 상기 서픽스 어레이는 트라이 구조로 변환될 수 있다. 노드가 가변 개수의 자식 노드를 가질 수 있기 때문에 일반적으로 복잡한, 상기 트라이 구조의 한 노드의 자식 노드로의 참조를 저장하는 대신, 트라이 구조의 노드 각각은 부모 노드로의 참조, 후손 노드의 데이터 시퀀스에 대한 공통 매치 길이, 및 가장-최근-순회된(most-recently-traversed) 후손 노드의 버퍼 위치를 포함한다. 후손 노드는 한 노드의 직계 자식 노드(immediate child node) 또는 그 밖의 다른 임의의 자식 노드, 손자 노드(grandchild node), 또는 후손 노드(descendant node)를 포함할 수 있다.

[0011] 압축 매치 열거의 특징 및 개념이 임의의 개수의 여러 다른 장치, 시스템, 환경, 네트워크, 및/또는 구성으로 구현될 수 있더라도, 압축 매치 열거의 실시예가 다음의 장치, 시스템, 및 방법의 맥락에서 기재된다.

[0012] 도 1은 압축 매치 열거의 다양한 실시예가 구현될 수 있는 압축 매치 열거 예시적 시스템(100)을 도시한다. 예시적 시스템은 열거된 압축 데이터 매칭을 수행하는 임의의 유형의 서버, 컴퓨터, 또는 장치일 수 있는 컴퓨팅 장치(102)를 포함한다. 본원에서 참조되는 컴퓨팅 장치들 중 임의의 것이, 도 4에서 도시된 예시적 장치를 참조하여 더 설명될, 다양한 구성요소, 가령, 하나 이상의 프로세서 및 메모리 장치뿐 아니라 임의의 개수의 서로 다른 구성요소들 및 이들의 조합 없이 구현될 수 있다.

[0013] 예시적 시스템(100)에서, 컴퓨팅 장치(102)는 저장된 데이터 시퀀스(106)를 저장 또는 유지하는 메모리 버퍼(104)를 포함한다. 메모리 버퍼는 데이터 저장을 가능하게 하는 임의의 유형의 컴퓨터 판독형 저장 매체, 가령, 임의의 유형의 메모리, 저장 매체 및/또는 적합한 전자 데이터 저장장치로서 구현될 수 있다. 저장된 데이터 시퀀스(106)는 인식되고 더 작은 데이터 표현으로 압축될 수 있는 반복되는 바이트 시퀀스를 가질 수 있는 메모리에 저장된 바이트이다.

[0014] 압축 매치 열거의 실시예는 저장된 데이터 시퀀스 내 모든 데이터 매치 가능성을 열거하도록 구현된다. 예를 들어, "abacde"의 데이터 스트링은 제 1 버퍼 위치에서 데이터 시퀀스를 시작할 수 있다. "abacfd"의 제 1 가능한 매치가 제 2 버퍼 위치에서 결정될 수 있고, 매치는 "abac"의 4 바이트 매치에 대해 (길이=4, 거리=1000)로 표현될 수 있으며, 제 1 버퍼 위치와 제 2 버퍼 위치 간의 거리는 1000바이트이다. "abacdf"의 제 2 가능한 매치가 제 3 버퍼 위치에서 결정될 수 있고, 상기 매치는 "abacd"의 5 바이트 매치에 대해 (길이=5, 거리=10,000)로 표현될 수 있고, 제 1 버퍼 위치와 제 3 버퍼 위치 간의 거리는 10,000바이트이다.

[0015] 또한 컴퓨팅 장치(102)는 본원에 기재된 압축 매치 열거의 실시예를 구현하는 열거 서비스(108)를 포함한다. 상기 열거 서비스는 컴퓨터 실행형 명령, 가령, 소프트웨어 애플리케이션 및/또는 알고리즘으로서 구현될 수 있고, 컴퓨팅 장치의 하나 이상의 프로세서에 의해 실행될 수 있다. 이 예시에서, 열거 서비스는 컴퓨팅 장치의 구성요소로서 구현되는 것으로서 도시된다. 대안으로서, 열거 서비스는 압축 매치 열거의 실시예를 구현하기 위

한 독립적인 소프트웨어 애플리케이션, 알고리즘, 또는 네트워크-기반 서비스로서 구현될 수 있다.

[0016] 실시예에서, 열거 서비스(108)는 메모리 버퍼(104)에 저장되는 저장된 데이터 시퀀스(106)를 나타내는 서픽스 어레이(suffix array)(110)를 생성하도록 구현된다. 상기 서픽스 어레이는 메모리 버퍼 내 모든 버퍼 위치(가령, 서픽스)의 어레이이며, 상기 버퍼 위치는 각자의 버퍼 위치 각각에서의 데이터 시퀀스를 시작하는 데이터 스트링에 의해 알파벳순으로 저장될 수 있다. 서픽스 어레이를 생성하기 위한 기법이 알려져 있고, 다양한 기법들 중 임의의 기법이 열거 서비스에 의해 서픽스 어레이를 생성하기 위해 구현될 수 있다.

[0017] 그 후 상기 열거 서비스(108)는 서픽스 어레이(110)를 트라이 구조(112)로 변환하는데, 상기 트라이 구조가 서픽스 어레이의 위치에서(in-place) 생성됨으로써 상기 트라이 구조가 상기 서픽스 어레이를 메모리 버퍼(104)에서 덮어 쓴다. 서픽스 어레이로부터 트라이 구조를 생성 또는 구축하기 위해 추가적인 메모리가 필요하지 않기 때문에 이는 메모리를 보존한다. 서픽스 어레이의 연속하는 서픽스들로부터 트라이 구조를 점진 업데이트함으로써 상기 서픽스 어레이는 트라이 구조로 변환될 수 있다. 상기 트라이 구조는 루트 노드에서 시작하여 하위 레벨 계층구조로 점차적으로 하향 확장하는 것 대신 리프-투-루트(leaf-to-root)로부터 생성된다. 리프 노드가 트라이 구조에 추가되어, 트라이 구조의 루트 노드로 점차적으로 축소될 수 있다, 즉, 루트 노드를 결정할 수 있다. 그 후 트라이 구조가 생성되거나 구성된 후, 열거 서비스가 트라이 구조를 검색하는 것을 개시하여, 상기 트라이 구조로부터 결정된 데이터 시퀀스 매치(114)를 열거할 수 있다.

[0018] 서픽스 어레이(110)의 서픽스들로부터 생성된 몇 개의 트라이 노드(118)를 포함하는 트라이 구조(112)의 예시(116)가 도시된다. 트라이 구조는 상기 서픽스 어레이(110)의 서픽스에 각각 대응하는 리프 노드, 가령, 예시적 리프 노드(120)를 포함한다. 또한 상기 트라이 구조는 비-리프(non-leaf) 노드, 예를 들어, 비-리프 노드(122)를 포함한다. 또한 리프 노드(120)는 트리 구조 내 하나의 부모 노드의 하나의 자식 노드일 수 있고, 반면에, 비-리프 노드(122)는 트라이 구조 내 자식 노드와 부모 노드 모두일 수 있다.

[0019] 트라이 구조(112)의 예시(116)가 큰 데이터 스트링으로부터의 서픽스들 중 일부만 알파벳 순으로 도시한다. 메모리 버퍼(104) 내 저장된 데이터 시퀀스(106)는 압축 매치 열거가 적용되는 매우 큰 데이터 세트일 수 있다. 화살표는 부모 노드 참조 포인터를 나타내고, 숫자는 특정 노드의 직계 후손 노드를 포함하여 모든 후손 노드의 데이터 시퀀스에 대한 공통 매치 길이를 나타낸다. 예를 들어, 데이터 시퀀스 "abceb"를 갖는 리프 노드(120)는 비-리프 노드(122)로의 부모 노드 참조를 포함하며, 여기서 4의 공통 매치 길이는 2개의 자식 노드의 데이터 시퀀스를 나타낸다.

[0020] 기하학상으로 말하자면, 트라이 구조가 생성될 때 부모 노드 포인터 중 어느 것도 서로를 교차하지 않을 평면 다이어그램(planar diagram)이기 때문에, 트라이 구조는 서픽스 어레이로부터 효율적으로 구성될 수 있다. 트라이 구조(112)에서 각각의 다음 번 리프 노드(120)가 연결될 때(그리고 비-리프 항목이 나중 항목에 의해 참조될 수 없어질 때), 새 리프 노드가 연결될 수 있는 노드들의 세트가 이 평면 속성 때문에 작고, 이는 압축 매치 열거의 실시예에서 효율적이고 단순한 알고리즘을 도출한다.

[0021] 트라이 구조(112)는 기본적으로, 매칭하는 서브스트링이 하나의 공통 노드 하에서 그룹지어지고, 이들 노드가 더 짧은 매치들을 효율적으로 그룹짓는 또 다른 노드 하에서 그룹지어지고, 이런 식으로 반복되는 트리(tree)이다. 노드-투-자식 노드 정보(node-to-child node information)가 필요하거나 저장되지 않는다. 오히려, 트라이 구조의 트라이 노드(118)가 부모 노드 참조, 후손 노드의 데이터 시퀀스에 대한 공통 매치 길이(가령, 직계 자식 노드를 포함하기 위해), 및 가장-최근-순회된 후손 노드의 노드 버퍼 위치에 대한 노드 필드(node field)를 포함한다. 보모 노드 참조 및 공통 매치 길이에 대한 필드 모두, 예를 들어, 부모 노드 참조 인덱스를 위한 26비트와 공통 데이터 시퀀스 길이를 위한 6비트를 이용해, 하나의 32비트 값에 저장될 수 있다. 더 긴 길이는 32비트 값에 저장 가능한 최대 길이로 한정될 수 있다.

[0022] 상기 열거 서비스(108)가 트라이 구조(112)를 구성할 때, 열거 서비스의 3개의 알고리즘 불변항(invariant)이 유지된다. 첫째, 매 비-리프 노드(122)는 적어도 2개의 직계 자식 노드를 가진다. 둘째, 노드의 자식 노드의 공통 매치 길이가 최대이다(가령, 10개의 스트링 위치가 공통적으로 길이 L을 갖고, 상기 스트링 위치 중 2개는 공통적으로 길이 L+1을 갖는 경우, 이들 2개의 스트링 위치는 그들 고유의 노드 하에서 그룹지어진다). 셋째, 이들 제약 내에서, 트라이 구조의 노드들의 총 개수는 최소화된다(가령, 2개의 스트링 위치 또는 하위노드가 하나의 공통 노드 하에서 그룹지어질 수 있을 경우, 이들은 상기 공통 노드 하에서 그룹지어져야 한다).

[0023] 열거 서비스(108)는 서픽스 어레이(110)의 서픽스(가령, 버퍼 위치)를, 서픽스 어레이에서 발생한 순서로, 처리하고, 상기 서픽스 어레이와 함께 계산된 가장 긴 공통 프리픽스 필드(longest common prefix field)를 기초로

결정을 한다. 가장 긴 공통 프리픽스 필드는 한 서픽스가 자신 밑으로 사전 순으로(lexicographically) 가장 가까운 서픽스와 얼마나 많이 매치하는지에 대한 길이를 나타낸다. 증가하는 매치 길이의 작은 스택이 유지되고, 프리픽스 길이가 감소할 때 스택을 언롤링(unrolling)함으로써, 트라이 노드(118)가 구성된다. 하나의 구현예에서, 매치 길이가 6비트 숫자들로서 한정(capped)되기 때문에 스택이 작다. 스택의 항목(entry)이 미래의 노드가 연결될 수 있는 모든 기존 노드들의 세트를 나타내거나, 트라이 구조(112)가 생성될 때 새로운 리프 노드(120)가 생성될 수 있다. 서픽스 어레이의 각각의 서픽스에 대해, 하나의 서픽스는 기존의 적정 길이노드와 함께 그룹지어지거나 트라이 구조의 새로운 리프 노드를 생성하도록 사용될 수 있다. 적정 길이 노드(appropriate-length node)에 대한 검색 중에 스택이 언롤링될 때 비-리프 노드는 부모 노드를 할당받는다. 이들 그룹화 결정은 앞서 언급된 3개의 알고리즘 불변항을 기초로 이뤄진다.

[0024] 트라이 구조(112) 내 비-리프 노드(122)에 대한 공간이 서픽스 어레이 자체로부터 할당되고, 할당된 비-리프 노드의 개수가 처리된 서픽스의 개수보다 작음이 쉽게 증명될 수 있기 때문에, 이는 서픽스 어레이(110)를 새로운 트라이 구조로 점진적으로 덮어 쓰는 "제위치(in-place)" 방법이다. 앞서 기재한 바와 같이, 2개의 큰 데이터 구조(가령, 서픽스 어레이(110) 및 트라이 구조(112)가 한 번에 둘 모두 저장되지 않기 때문에 이는 메모리를 보존한다. 각각의 리프 노드(120)는 하나의 서픽스(가령, 버퍼 위치)에 대응하며, 각각의 리프 노드의 부모 모드가, 역 서픽스 어레이(inverse suffix array)를 덮어 씀으로써 저장된다.

[0025] 열거 서비스(108)는 또한 매치 열거 검색 방법을 구현하고, 순서대로 버퍼 위치 각각에 대해, 트라이 구조 내 리프 노드로부터 루트 노드로의 부모 노드 포인터를 따른다. 각각의 비-리프 노드(122)(가령, 트라이 노드(118)에 의해 나타내어지는 것)가 부모 노드 참조, 공통 매치 길이, 및 비-리프 노드가 가장-최근-순회된 직계 자식 노드를 포함하기 위해 후손 노드의 후손 노드 버퍼 위치(가령, 이 프로세스 동안 상기 비-리프 노드를 가장 최근 "방문"한 후손 노드의 버퍼 위치)를 저장하거나 유지한다. 그 밖의 다른 노드로부터의 다음 번 순회를 위해, 이들 노드 필드가 하나의 리프 노드에서 루트 노드로의 리프-투-루트 경로를 따라 각각의 길이에 대해 최단 거리 데이터 시퀀스 역방향 매치(backwards-match)를 제공한다. 결과적으로, 검색 단계 중 트라이 구조에 대해 변하는 유일한 노드 필드는 각각의 비-리프 노드와 연관된 가장-최근-순회된 후손 노드에 대한 후손 노드 버퍼 위치 필드이다.

[0026] 데이터 시퀀스의 가장 최근 매치가 긴 매치(long match)이고, 트라이 구조(112)에서 이보다 짧은 매치가 다수 나타나는 경우, 트라이 구조가 점차적으로 순회(progressively traverse)될 때마다 동일한 데이터 매치가 반복적으로 발생할 수 있다. 예를 들어, 길이 10의 최단 거리 매치가 길이 9, 8, 7 등등의 최단 거리 매치일수도 있다. 단순한 해결책은 트라이 구조 내 각각의 비-리프 노드에 대해 메모리의 8바이트, 가령, 부모 노드 참조를 위해 약 4바이트, 후손 노드 버퍼 위치(가령, 가장-최근-본(most-recently-seen) 위치)를 위해 대략 거의 4 바이트, 공통 매치 길이를 위한 약간의 비트를 이용하는 것이다. 그러나 트라이 구조 내 비-리프 노드당 8바이트를 이용하고 리프 노드당 4바이트를 이용하는 것은 12 곱하기 노드의 개수(12n)의 총 메모리 사용을 야기할 것이다. 이 총 메모리 사용은 트라이 구조 내 비-리프 노드당 4바이트만 이용함으로써, 8 곱하기 노드의 개수(8n)로 감소될 수 있다.

[0027] 메모리를 절약하기 위해, 열거 서비스는 리프-투-부모 참조가 순서대로만 사용되며, 위치에 대한 데이터 매치가 찾아지면, 상기 위치에 대한 리프 데이터는 더 이상 필요하지 않다는 사실을 이용한다. 메모리 버퍼(104)가 순방향으로(forward) 순회됨에 따라, 트라이 구조(112)를 구성하는 동안 덮어 써졌던 앞서 역 서픽스 어레이였던 것으로부터 메모리 공간이 이용 가능해지고, 이 이용 가능한 메모리 공간이 모든 정보를 저장하기 위해 사용될 수 있다. 하나의 구현예에서, 이 이용 가능한 메모리 공간은 노드 부모 저장공간에 인접할 가능성이 높지 않기 때문에 부모 자체를 찾기 위해 필드가 "부모 노드 공간"에 저장된다. 리프-투-부모 어레이가 추가 메모리로서 사용될 때, 가장-최근-본 노드 필드가 거의 이러한 필드이다. 따라서 트라이 구조의 비-리프 부분(가령, 서픽스 어레이이도록 사용된 것)에서, 부모 노드 참조가 가장-최근-본 위치로 대체되고, 가장-최근-본 위치의 리프 부모 노드에 대해 앞서 사용됐던 본래 부모 노드 참조가 메모리에 저장되며, 노드로부터 가장-최근-본 위치를 앎으로써, 도달될 수 있다. 따라서 검색 방법이 순회하거나 방문한 노드 각각에 대해, 3개의 노드 필드가 부모 노드 참조, 가장-최근-순회된 후손 노드, 및 후손 노드의 공통 매치 길이에 대해 결정될 수 있다.

[0028] 이는, 메모리 활용을 절약할 뿐 아니라, 길면서 가까이 있는(nearby) 데이터 매치가 결정될 때 트라이 구조(112)에서 순회될 필요가 없는 노드를 스킵(skip)한다. 이는 가장 최근인 전체 위치가 실제로 자신의 리프-투-루트 경로 내 각각의 노드에 대한 가장-최근-순회된(가령, 가장-최근-본) 위치이며, 메모리 공간은 단 하나의 부모 노드 참조를 위한 것이기 때문이다. 초기에, 부모 노드가 트라이 구조의 루트 노드임을 나타내기 위해 널(NULL) 참조가 저장될 수 있다. 그 후, 다음 번 노드 위치가 처리되면, 다음 번 노드 위치가 루트 노드에서 본

가장-최근-순회된 것이 되고, 이전 가장-최근-순회된 위치의 부모 참조가 업데이트되어, 두 위치 모두에 공통되는 노드(즉, 편리하게도, 순회 중 "현재" 노드)가 참조된다. 이는 루트 노드에서만이 아니라 트라이 구조의 매 노드에서 이뤄지며, 이는 오래된 위치(older position)일수록 트라이 구조에서 더 아래에 위치하게 한다. 트라이 구조를 계속하여 변경하고 업데이트하는 이러한 기법은, 트라이 구조를 순회할 때 불필요한 단계, 가령, 동일한 매치 길이를 복수 번 열거하거나 자명하게 차선(supoptimal)인 매치를 열거하는 단계를 스킵하는 데 효율적이다.

[0029] 도 2는 트라이 구조(200)의 일례 및 도 1을 참조하여 기재된 열거 서비스(108)가 본원에 기재된 압축 매치 열거의 실시예에 따라 트라이 구조를 검색 및 업데이트하기 위해 구현되는 방식을 도시한다. 서픽스 어레이로부터 변환되고 임의의 매치 열거 전에 구성된 바와 같은 트라이 구조가 (202)로 나타난다. 상기 트라이 구조의 노드는 노드(204 내지 220)로서 식별되고, 리프 노드의 바닥 층이 또한 P1 내지 P4로서 식별되어, 열거 서비스에 의해 리프 노드가 처리되는 순서를 나타낼 수 있다. 매치 열거를 수행하기 위한 2개의 기본 원리는 다음과 같다: 첫째, 이전에 순회된 비-리프 노드가 비-리프 노드가 가장-최근-순회됐던 리프 노드를 가리키고(point to), 둘째, 리프 노드 및 미-순회된(not-yet-traversed) 노드가 일부 비-리프 노드(대략적으로, 일부 경우 루트 노드일 수 있는 부모 노드라고 지칭됨)를 가리킨다.

[0030] (202)로 도시된 트라이 구조에서 (224)로 도시된 트라이 구조로의 제 1 전이(transition)(222)에서, 열거 서비스(108)는 리프 노드(204)(가령, 제 1 처리된 리프 노드 P1)와 연관된 서픽스(가령, 버퍼 위치)에 대한 데이터 매치를 열거하도록 시도한다. 이 예에서, 제 1 처리되는 리프 노드에 대해 어떠한 매치도 발견되지 않지만, 트라이 구조가 업데이트되어, 리프 노드(204)에서 루트 노드(220)로의 경로 내 각각의 노드가 이제 각자 고유의 리프 노드(가령, 이들 노드의 가장 최근 "방문자")를 참조하게 된다. 예를 들어, 리프 노드(204)에서 루트 노드(220)로의 경로는 노드(212, 216, 및 218)를 통과한다. 이들 노드 각각이 순회될 때, 후손 노드 버퍼 위치에 대한 각자의 노드 필드가 리프 노드(204)를 참조하도록 업데이트된다. 덧붙여, 리프 노드(204)의 부모 노드 참조 필드가 루트 노드(220)를 참조하도록 업데이트된다.

[0031] (224)로 도시된 트라이 구조에서 (228)로 도시된 트라이 구조로의 제 2 전이(226)에서, 열거 서비스(108)는 리프 노드(208)로부터의 검색 경로(search path)를 따라서, 리프 노드(가령, 제 2 처리된 리프 노드 P2)와 연관된 서픽스에 대한 매치를 열거할 수 있다. 이 예에서, (다이어그램에서 특정되지 않은) 일부 길이의 매치가 (가령, 비-리프 노드(214)가 순회된 후) 순회되는 제 2 비-리프 노드(216)에서 열려된다. 노드(216)로부터, 매칭되는 리프 노드(204)의 위치가 결정된다(가령, 노드(216)가 (224)로 도시된 리프 노드(204)를 참조한다). 다음 번 노드로 순회할, 이 예시에서 루트 노드(220)인 부모 노드가 또한 리프 노드(204)의 부모 노드 참조 필드로부터 결정된다(가령, 노드(204)가 (224)로 도시된 루트 노드(220)를 참조한다). 이 예시에서, 리프 노드(208)에 대한 본래의 리프-투-루트 경로 내 비-리프 노드(218)가 스킵되며, 이는 이하에서 더 설명될 바와 같이 다뤄질 것이다. 덧붙여, 리프 노드(208)의 부모 노드 참조 필드가 루트 노드(220)를 참조하도록 업데이트된다.

[0032] (228)로 도시된 트라이 구조에서 (232)로 도시되는 트라이 구조로의 제 3 전이(230)에서, 열거 서비스(108)는 리프 노드(206)로부터의 검색 경로를 따라, 리프 노드(가령, 제 3 처리된 리프 노드 P3)와 연관된 서픽스에 대한 매치를 열거할 수 있다. 이 예시에서, 2개의 매치가 리프 노드(204) 및 리프 노드(208)에서 열거된다. 리프 노드(208)에서의 부모 노드 참조 필드가 노드(216)를 참조하도록 업데이트되고, 검색 경로 내 비-리프 노드(212 및 216)가 가장 최근 노드인 리프 노드(206)를 참조하도록 업데이트되어 비-리프 노드(212 및 216)를 순회하도록 한다. 덧붙여, 리프 노드(206)의 부모 노드 참조 필드가 루트 노드(220)를 참조하도록 업데이트된다.

[0033] (232)로 도시된 트라이 구조에서 (236)로 도시된 트라이 구조로의 마지막 전이(234)에서, 열거 서비스(108)는 리프 노드(210)(가령, 제 4 처리된 리프 노드 P4)로부터의 검색 경로를 따른다. 앞서 리프 노드(208)에 대한 본래 리프-투-루트 경로에서 스킵되었던 노드인 비-리프 노드(218)에 직면한다. 상기 비-리프 노드(218)는, 트라이 구조 내에서 상기 비-리프 노드(218) 아래에 있는 그의 부모 노드(216)를 참조하는 제 1 리프 노드(204)를 여전히 참조한다. 검색 경로가 트라이 구조 내 더 상위를 순회해야 하기 때문에 이는 문제가 있는 것으로 보일 수 있다. 그러나 이는 리프 노드(204)의 부모 노드(216)가 현재 비-리프 노드(218)보다 긴 길이를 가짐을 검출함으로써 간단하게 핸들링될 수 있다. 그 후 열거 서비스는 노드(216)로부터 노드(206)를 통과하는 검색 경로를, 현재 비-리프 노드(218)보다 더 짧은 길이를 갖는 노드, 이 경우 루트 노드(220)를 만날 때까지, 계속 따르고, 더 짧은 길이는 후손 노드보다는 조상 노드를 나타낸다. 따라서 리프 노드(206)의 부모 노드 참조 필드가 비-리프 노드(218)를 참조하도록 업데이트되고, 리프 노드(208)의 부모 노드 참조 필드는 루트 노드(220)를 참조하도록 업데이트된다. 각각의 리프 노드에 대해 검색 경로를 따르고 트라이 구조를 순회하기 위한 알고리즘

의 동적 속성이, 처리 비용 거의 없이도, 열거된 매치를 정확하게 제공한다.

[0034] 압축 매치 열거의 하나 이상의 실시예에 따라, 도 3을 참조하여 예시적 방법(300)이 기재된다. 일반적으로, 본원에 기재된 서비스, 기능, 방법, 절차, 컴포넌트, 및 모듈 중 임의의 것이 소프트웨어, 펌웨어, 하드웨어(가령, 고정 로직 회로), 수동 처리, 또는 이들의 임의의 조합으로 구현될 수 있다. 소프트웨어 구현은 컴퓨터 프로세서에 의해 실행될 때 특정된 작업을 수행하는 프로그램 코드를 나타낸다. 예시적 방법이, 소프트웨어, 애플리케이션, 루틴, 프로그램, 객체, 컴포넌트, 데이터 구조, 절차, 모듈, 기능, 및 등등을 포함할 수 있는 컴퓨터 실행 가능한 명령의 일반적인 맥락에서 기재될 수 있다. 상기 프로그램 코드는 컴퓨터 프로세서에 로컬 및/또는 원격으로 위치하는 하나 이상의 컴퓨터 판독형 저장 매체 장치에 저장될 수 있다. 또한 상기 방법은 복수의 컴퓨터 장치에 의해 분산 컴퓨팅 환경에서 실시될 수 있다. 덧붙여, 본원에 기재된 특징은 플랫폼-독립적이며, 다양한 프로세서를 갖는 다양한 컴퓨팅 플랫폼에서 구현될 수 있다.

[0035] 도 3은 압축 매치 열거의 예시적 방법(300)을 도시한다. 방법의 블록들이 기재된 순서는 한정사항으로 의도된 것이 아니며, 임의의 개수의 기재된 방법의 블록들이 임의의 순서로 조합되어 방법 또는 대안 방법을 구현할 수 있다.

[0036] 블록(302)에서, 메모리 버퍼에 저장된 데이터 시퀀스를 나타내는 서픽스 어레이가 생성된다. 예를 들어, 컴퓨팅 장치(10)(도 1)에서 열거 서비스(108)는 메모리 버퍼(104)에 저장된 저장 데이터 시퀀스(106)를 나타내는 서픽스 어레이(110)를 생성한다. 상기 서픽스 어레이는 메모리 버퍼 내 버퍼 위치(buffer position)의 어레이이고, 버퍼 위치는 각자의 데이터 위치에서의 데이터 시퀀스를 시작하는 데이터 스트링에 의해 알파벳 순으로 정렬된다.

[0037] 블록(304)에서, 상기 서픽스 어레이는 트라이 구조로 변환되는데, 상기 트라이 구조는 상기 서픽스 어레이의 위치에서(in-place) 생성되기 때문에 상기 트라이 구조는 메모리 버퍼에서 상기 서픽스 어레이를 덮어 쓴다. 예를 들어, 열거 서비스(108)는 서픽스 어레이(110)를 트라이 구조(112)로 변환하고, 상기 트라이 구조는 서픽스 어레이의 위치에서(in-place) 생성되기 때문에 상기 메모리 버퍼에서 서픽스 어레이를 덮어 쓴다. 상기 서픽스 어레이는 서피스 어레이의 연속하는 서픽스들로부터 트라이 구조를 점진적으로 업데이트함으로써 상기 트라이 구조로 변환된다. 상기 트라이 구조는 각각 상기 서픽스 어레이의 하나 이상의 서픽스를 나타내는 노드를 포함하고, 각각의 연속하는 서픽스는 상기 트라이 구조 내 기존 노드와 그룹지어지거나 트라이 구조의 새 노드로서 추가된다. 서픽스가 기존 노드와 데이터 시퀀스의 공통 매치 길이를 가질 때 서픽스 어레이의 서픽스는 트라이 구조 내 상기 기존 노드와 그룹지어질 수 있다. 덧붙여, 트라이 구조는 다음을 바탕으로 생성된다: 트라이 구조는 각각 적어도 두 개의 직계 자식 노드를 갖는 하나 이상의 비-리프 노드를 포함함; 후손 노드의 데이터 시퀀스에 대한 공통 매치 길이는 최대임; 그리고 트라이 구조 내 노드의 총 개수는 최소화됨.

[0038] 블록(306)에서 트라이 구조는 각각의 리프 노드에서 루트 노드까지 검색되고, 블록(308)에서 데이터 시퀀스 매치는 트라이 구조로부터 결정되는대로 열거된다. 예를 들어, 열거 서비스(108)는 각각의 리프 노드(204)(도 2)로부터 시작하여 트라이 구조를 검색하고, 루트 노드(220)로의 부모 노드 참조의 경로를 따라, 트라이 구조(112)로부터 결정되는 데이터 시퀀스 매치(114)를 열거할 수 있다.

[0039] 블록(310)에서, 가장-최근-순회된 노드의 노드 필드가 현재 위치로 업데이트되어, 미래 데이터 시퀀스 매치를 위해 가장-최근-순회된 노드를 지정할 수 있다. 예를 들어, 열거 서비스(108)는 가장-최근-순회된 노드의 노드 필드를 현재 위치로 업데이트하여 미래 데이터 시퀀스 매치를 위해 가장-최근-순회된 노드를 지정할 수 있다. 가장-최근-순회된 노드의 노드 필드는 부모 노드로의 참조, 후손 노드의 데이터 시퀀스에 대한 공통 매치 길이(가령, 노드의 직계 자식 노드를 포함하기 위해), 및 가장-최근-순회된 후손 노드의 버퍼 위치를 포함한다.

[0040] 도 4는 앞서 도 1-3을 참조하여 기재된 장치들 중 임의의 것 또는 상기 장치에 의해 구현되는 서비스로서 구현될 수 있는 예시적 장치(400)의 다양한 구성요소를 도시한다. 실시예에서, 장치는 소비자, 컴퓨터, 휴대용, 사용자, 통신, 전화기, 내비게이션, 텔레비전, 가전기구, 게임, 미디어 재생 및/또는 전자 장치의 임의의 형태로 된 고정형 또는 이동형 장치 중 임의의 하나 또는 이들의 조합으로서 구현될 수 있다. 또한 장치가 사용자, 소프트웨어, 펌웨어, 하드웨어, 및/또는 장치의 조합을 포함하는 논리 장치를 기술하도록, 상기 장치는 사용자(즉, 사람) 및/또는 장치를 운영하는 개체와 연관될 수도 있다.

[0041] 장치(400)는 장치 데이터(404), 가령, 수신된 데이터, 수신 중인 데이터, 브로드캐스팅되도록 스케줄링된 데이터, 데이터의 데이터 패킷 등의 유선 및/또는 무선 통신을 가능하게 하는 통신 장치(402)를 포함한다. 상기 장치 데이터 또는 그 밖의 다른 장치 콘텐츠는 장치의 구성 설정, 장치에 저장되는 미디어 콘텐츠, 및/또는 장치

의 사용자와 연관되는 정보를 포함할 수 있다. 장치에 저장되는 미디어 콘텐츠는 임의의 유형의 오디오, 비디오, 및/또는 이미지 데이터를 포함할 수 있다. 상기 장치는 임의의 유형의 데이터, 미디어 콘텐츠, 및/또는 입력이 수신될 때 이용하는 하나 이상의 데이터 입력(406), 가령, 사용자 선택형 입력, 및 그 밖의 다른 임의의 유형의 오디오, 비디오 및/또는 임의의 콘텐츠 및/또는 데이터 소스로부터 수신된 이미지 데이터를 포함한다.

[0042] 상기 장치(400)는 또한 통신 인터페이스(408), 가령, 직렬, 병렬, 네트워크, 또는 무선 인터페이스 중 임의의 하나 이상을 포함한다. 상기 통신 인터페이스는 장치와 통신 네트워크 간 연결 및/또는 통신 링크를 제공하며, 이를 통해, 전자, 컴퓨팅, 및 통신 장치가 장치와 데이터를 통신한다.

[0043] 장치(400)는 장치의 동작을 제어하기 위해 다양한 컴퓨터 실행형 명령을 처리하는 하나 이상의 프로세서(410)(가령, 마이크로프로세서, 제어기, 및 등등 중 임의의 것)를 포함한다. 이를 대체하여 또는 이에 추가로, 장치는 전체적으로 (412)로 식별되는 프로세싱 및 제어 회로와 연결되어 구현되는 소프트웨어, 하드웨어, 펌웨어, 또는 고정 로직 회로 중 임의의 하나 또는 이들의 조합에 의해 구현될 수 있다. 도시되지 않더라도, 장치는 장치 내 다양한 구성요소들을 연결하는 시스템 버스 또는 데이터 전송 시스템을 포함할 수 있다. 상기 시스템 버스는 서로 다른 버스 구조, 가령, 메모리 버스 또는 메모리 제어기, 주변장치 버스, 전역 직렬 버스, 및/또는 다양한 버스 아키텍처 중 임의의 것을 이용하는 프로세서 또는 로컬 버스 중 임의의 하나 또는 이들의 조합을 포함할 수 있다.

[0044] 또한 상기 장치(400)는 데이터 저장을 가능하게 하는 하나 이상의 메모리 장치(416)(가령, 컴퓨터 판독형 저장 매체), 가령, 랜덤 액세스 메모리(RAM), 비-휘발성 메모리(가령, 리드 온리 메모리(ROM), 플래시 메모리, 등), 및 디스크 저장 장치를 포함한다. 디스크 저장 장치는 임의의 유형의 자기 또는 광학 저장 장치, 가령, 하드 디스크 드라이브, 레코딩 가능한 및/또는 재기록 가능한 디스크 및 등등으로서 구현될 수 있다. 상기 장치는 또한 대용량 저장 매체 장치를 포함할 수 있다.

[0045] 컴퓨터 판독형 매체는 컴퓨팅 장치에 의해 액세스되는 임의의 이용 가능한 매체일 수 있다. 비제한적 예를 들면, 컴퓨터 판독형 매체는 저장 매체 및 통신 매체를 포함할 수 있다. 저장 매체는 정보, 가령, 컴퓨터 판독형 명령, 데이터 구조, 프로그램 모듈, 또는 그 밖의 다른 데이터를 저장하는 임의의 방법 또는 기법으로 구현된 휘발성 및 비-휘발성, 이동식 및 비-이동식 매체를 포함한다. 저장 매체의 비제한적 예를 들면, RAM, ROM, EEPROM, 플래시 메모리 또는 그 밖의 다른 메모리 기법, CD-ROM, DVD(digital versatile disk), 또는 그 밖의 다른 광학 저장장치, 자기 카세트, 자기 테이프, 자기 디스크 저장 또는 그 밖의 다른 자기 저장 장치, 또는 정보를 저장하기 위해 사용될 수 있고 컴퓨터에 의해 액세스될 수 있는 그 밖의 다른 임의의 매체가 있다.

[0046] 통신 매체는 컴퓨터 판독형 명령, 데이터 구조, 프로그램 모듈, 또는 그 밖의 다른 데이터를 변조된 데이터 신호, 가령, 반송파 또는 그 밖의 다른 수송 메커니즘으로 구현하는 것이 일반적이다. 또한 통신 매체는 임의의 정보 전달 매체를 포함한다. 변조된 데이터 신호는 신호에 정보를 인코딩하기 위한 방식으로 설정 또는 변경된 하나 이상의 특성을 가진다. 비-제한적 예를 들면, 통신 매체는 유선 매체, 가령, 유선 네트워크 또는 직접 배선 연결 및 무선 매체, 가령, 음향, RF, 적외선, 및 그 밖의 다른 무선 매체를 포함한다.

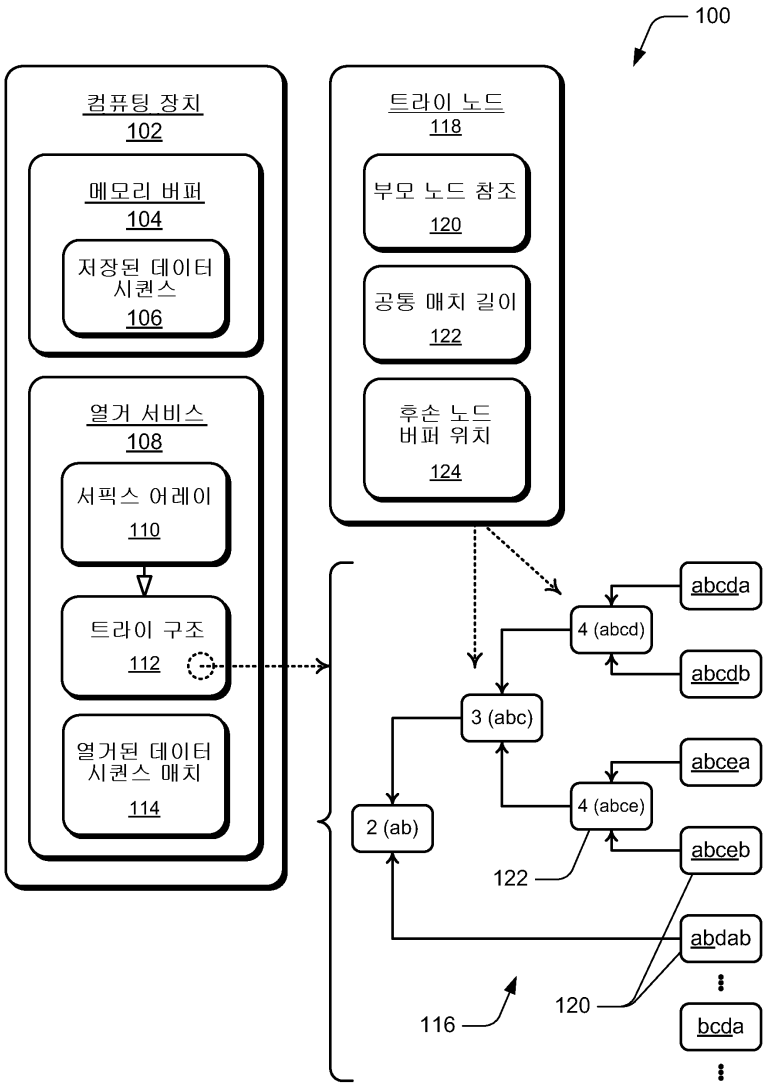
[0047] 메모리 장치(414)는 장치 데이터(404), 그 밖의 다른 유형의 정보 및/또는 데이터, 및 다양한 장치 애플리케이션(416)을 저장하기 위한 데이터 저장 메커니즘을 제공한다. 예를 들어, 운영 체제(418)는 메모리 장치에 의해 소프트웨어 애플리케이션으로서 유지될 수 있고, 프로세서 상에서 실행된다. 상기 장치 애플리케이션은 또한 장치 관리자, 가령, 임의의 형태의 제어 애플리케이션, 소프트웨어 애플리케이션, 신호 프로세싱 및 제어 모듈, 특정 장치 고유의 코드, 특정 장치의 하드웨어 추상화 층, 및 등등을 포함할 수 있다. 이 예에서, 장치 애플리케이션(416)은 본원에서 기재된 압축 매치 열거의 실시예를 구현하는 열거 서비스(420)를 포함한다.

[0048] 상기 장치(400)는 또한, 오디오 시스템(424)에 대해 오디오 데이터를 생성하고, 및/또는 디스플레이 시스템(426)을 위해 디스플레이 데이터를 생성하는 오디오 및/또는 비디오 프로세싱 시스템(422)을 포함한다. 상기 오디오 시스템 및/또는 디스플레이 시스템은 오디오, 비디오, 디스플레이 및/또는 이미지 데이터를 처리, 디스플레이, 및/또는 그 밖의 다른 방식으로 렌더링하는 임의의 장치를 포함할 수 있다. 디스플레이 데이터 및 오디오 신호가 RF(라디오 주파수) 링크, S-비디오 링크, 복합 비디오 링크, 컴포넌트 비디오 링크, DVI(digital video interface), 아날로그 오디오 연결, 또는 그 밖의 다른 유사한 통신 링크를 통해 오디오 장치 및/또는 디스플레이 장치로 전달될 수 있다. 구현예에서, 오디오 시스템 및/또는 디스플레이 시스템은 장치의 외부 구성요소이다. 대안적으로, 오디오 시스템 및/또는 디스플레이 시스템은 예시적 장치의 일체화된 구성요소, 가령, 일체화된 터치-스크린 디스플레이이다.

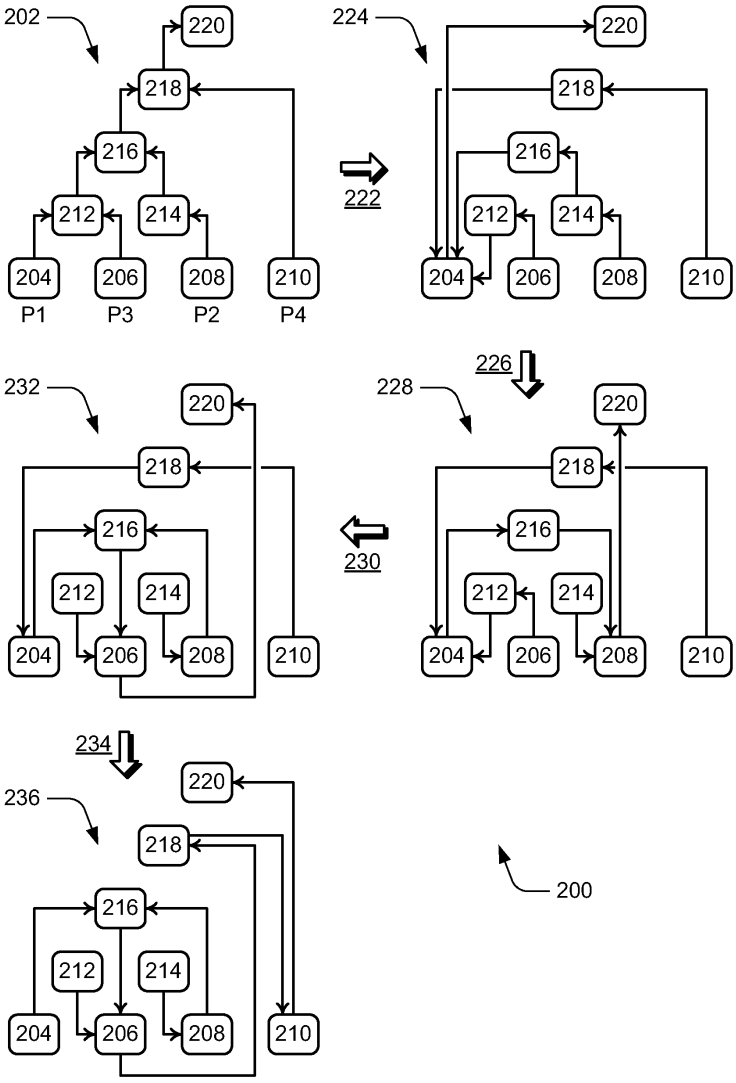
[0049] 압축 매치 열거의 실시예가 특징 및/또는 방법에 특정된 기재로 개시되었지만, 특허청구범위의 대상은 기재된 특정 특징 또는 방법에 반드시 한정되는 것은 아니다. 오히려 특정 특징 및 방법이 압축 매치 열거의 예시적 구현예로서 개시된다.

도면

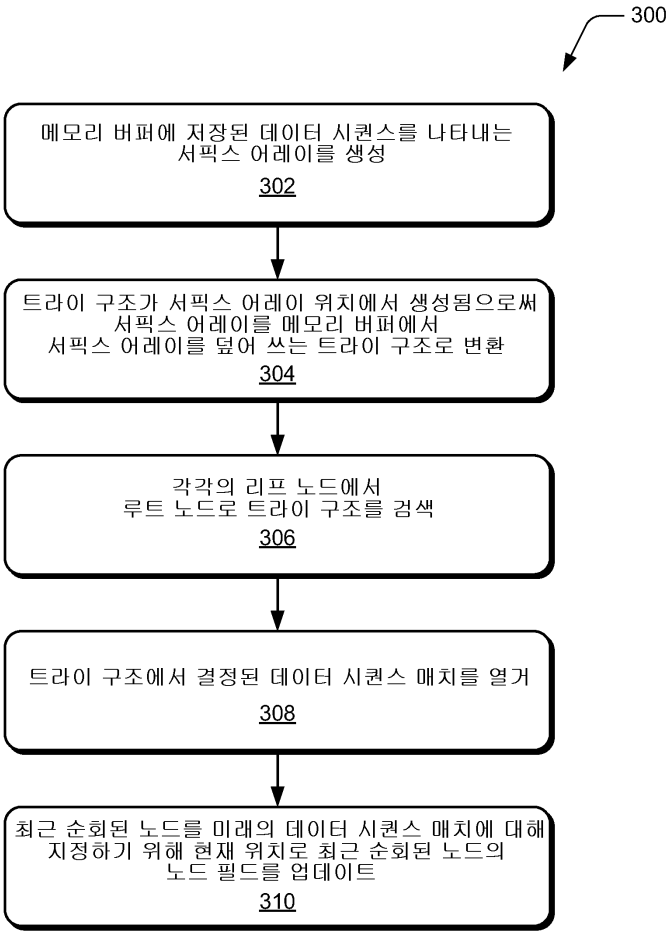
도면1



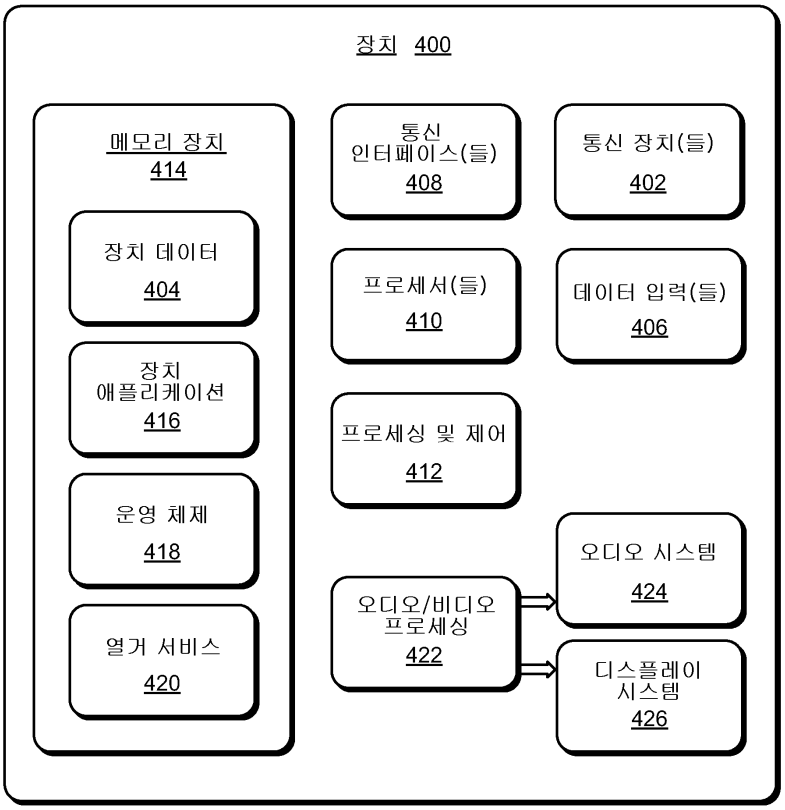
도면2


202
220
218
216
212
214
204
206
208
210
P1
P3
P2
P4
222
224
226
228
230
232
234
236
200

도면3


300
메모리 버퍼에 저장된 데이터 시퀀스를 나타내는 서픽스 어레이를 생성
302
트라이 구조가 서픽스 어레이 위치에서 생성됨으로써 서픽스 어레이를 메모리 버퍼에서 서픽스 어레이를 덮어 쓰는 트라이 구조로 변환
304
각각의 리프 노드에서 루트 노드로 트라이 구조를 검색
306
트라이 구조에서 결정된 데이터 시퀀스 매치를 열거
308
최근 순회된 노드를 미래의 데이터 시퀀스 매치에 대해 지정하기 위해 현재 위치로 최근 순회된 노드의 노드 필드를 업데이트
310

도면4


장치 400
메모리 장치
414
장치 데이터
404
장치
애플리케이션
416
운영 체제
418
열거 서비스
420
통신
인터페이스(들)
408
프로세서(들)
410
프로세싱 및 제어
412
오디오/비디오
프로세싱
422
통신 장치(들)
402
데이터 입력(들)
406
오디오 시스템
424
디스플레이
시스템
426