



(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2014년04월09일
(11) 등록번호 10-1382945
(24) 등록일자 2014년04월01일

(51) 국제특허분류(Int. Cl.)

G06F 9/38 (2006.01)

(21) 출원번호 10-2012-0115119

(22) 출원일자 2012년10월17일

심사청구일자 2012년10월17일

(56) 선행기술조사문헌

KR101081726 B1*

JP04598031 B2

JP2009259241 A

*는 심사관에 의하여 인용된 문헌

(73) 특허권자

인하대학교 산학협력단

인천광역시 남구 인하로 100, 인하대학교 (용현동)

(72) 발명자

최원익

인천 연수구 해돋이로84번길 10, 607동 504호 (송도동, 송도풍림아이원6단지아파트)

김민철

인천 남구 토금남로 79, 101동 811호 (용현동, 우성아파트)

(74) 대리인

양성보

전체 청구항 수 : 총 2 항

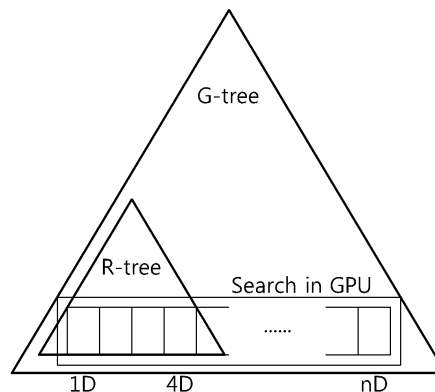
심사관 : 지정훈

(54) 발명의 명칭 GPU 병렬 검색을 통한 R-트리 계열의 계층형 색인 구조의 범위 질의 가속화 방법

(57) 요약

GPU 병렬 검색을 통한 R-트리 계열의 계층형 색인 구조의 범위 질의 가속화 방법이 개시된다. R-트리 계열의 계층형 색인 구조의 범위질의 병렬 가속화 방법은, GPU(graphic processing unit)에서의 병렬 검색을 위한 G-트리 구조를 이용하여 범위 질의를 처리하는 단계를 포함할 수 있으며, 이때 상기 범위 질의를 처리하는 단계는, 주어진 데이터의 전체 차원(dimensions) 중에서 저차원에 해당되는 일부 차원에 상기 R-트리의 인덱싱을 적용하고 나머지 차원에 대한 검색을 상기 GPU를 이용하여 수행할 수 있다.

대표도 - 도5



이 발명을 지원한 국가연구개발사업

과제고유번호 2011- 0026987

부처명 교육과학기술부

연구사업명 기초연구사업 일반연구자지원사업

연구과제명 GPU를 이용한 시공간데이터베이스 가속화 기법

기 여 율 1/1

주관기관 인하대학교 산학협력단

연구기간 2010.09.01 ~ 2015.08.31

특허청구의 범위

청구항 1

삭제

청구항 2

GPU(graphic processing unit)에서의 병렬 검색을 위한 G-트리 구조를 이용하여 범위 질의를 처리하는 단계
를 포함하고,

상기 범위 질의를 처리하는 단계는,

주어진 데이터의 전체 차원(dimensions) 중에서 저차원에 해당되는 일부 차원에 R-트리의 인덱싱을 적용하고 나머지 차원에 대한 검색을 상기 GPU를 이용하여 수행하되,

상기 R-트리가 상기 일부 차원만을 색인하여 단말 노드의 후보군을 검색하는 단계; 및

상기 후보군에 해당되는 단말 노드의 나머지 차원을 상기 R-트리를 이용하지 않고 상기 GPU를 이용하여 병렬 처리하는 단계

를 포함하는 R-트리 계열의 계층형 색인 구조의 범위질의 병렬 가속화 방법.

청구항 3

GPU(graphic processing unit)에서의 병렬 검색을 위한 G-트리 구조를 이용하여 범위 질의를 처리하는 단계
를 포함하고,

상기 범위 질의를 처리하는 단계는,

주어진 데이터의 전체 차원(dimensions) 중에서 저차원에 해당되는 일부 차원에 R-트리의 인덱싱을 적용하고 나머지 차원에 대한 검색을 상기 GPU를 이용하여 수행하되,

상기 G-트리 구조에서 너비 우선 검색(breadth first search) 방식을 적용하여 상기 범위 질의를 처리하는 것을 특징으로 하는 R-트리 계열의 계층형 색인 구조의 범위질의 병렬 가속화 방법.

명세서

기술분야

[0001] 본 발명의 실시예들은 R-트리 계열의 계층형 색인 구조에서 범위 질의를 처리하는 방법에 관한 것이다.

배경기술

[0002] R-tree는 공간자료 데이터베이스 시스템에서 가장 널리 사용되는 자료구조로서, 다차원의 자료를 관리하는데 매우 적합하며, 공간 데이터에 대한 삽입, 삭제, 검색 등의 작업을 효율적으로 수행할 수 있는 구조를 가지고 있는 색인 구조이다.

[0003] 한국공개특허 제10-2011-0068578호(공개일 2011년 06월 22일)에는 GPU(Graphics Processing Units)를 트리 형태의 색인 구조에 적용하여 R-tree에서 범위 질의(Range Query)를 처리하는 기술이 개시되어 있다.

[0004] 그러나, R-tree 계열의 계층형 색인구조에서 고차원 색인의 경우 차원이 증가할수록 MBR(minimum bounding rectangle)의 중복이 매우 심해져 색인의 효과를 볼 수 없다. R-tree 계열 중에서 고차원 인덱싱에 많이 사용되는 X-tree의 경우 중복이 심해지면 일부 노드를 슈퍼 노드로 바꿔 선형성을 증가시키지만 이 또한 매우 높은 고차원이나 선택도가 높아질 경우 성능이 매우 악화되게 된다.

발명의 내용

해결하려는 과제

[0005] R-트리 계열의 계층형 색인 구조에서 차원이 증가할수록 MBR의 중복이 심해지는 문제를 해결할 수 있는 범위 질의 병렬 처리 방법을 제공한다.

과제의 해결 수단

[0006] 본 발명의 실시예에 따르면, R-트리 계열의 계층형 색인 구조의 범위질의 병렬 가속화 방법은 GPU(graphic processing unit)에서의 병렬 검색을 위한 G-트리 구조를 이용하여 범위 질의를 처리하는 단계를 포함할 수 있으며, 이때 상기 범위 질의를 처리하는 단계는, 주어진 데이터의 전체 차원(dimensions) 중에서 저차원에 해당되는 일부 차원에 상기 R-트리의 인덱싱을 적용하고 나머지 차원에 대한 검색을 상기 GPU를 이용하여 수행할 수 있다.

[0007] 일 측면에 따르면, 상기 범위 질의를 처리하는 단계는, 상기 R-트리가 상기 일부 차원만을 색인하여 단말 노드의 후보군을 검색하는 단계; 및 상기 후보군에 해당되는 단말 노드의 나머지 차원을 상기 R-트리를 이용하지 않고 상기 GPU를 이용하여 병렬 처리하는 단계를 포함할 수 있다.

[0008] 다른 측면에 따르면, 상기 범위 질의를 처리하는 단계는, 상기 G-트리 구조에서 너비 우선 검색(breadth first search) 방식을 적용하여 상기 범위 질의를 처리할 수 있다.

발명의 효과

[0009] 본 발명의 실시예에 따르면, 고차원의 데이터가 주어질 경우 R-tree가 최적의 성능을 보여주는 저차원만을 색인하여 단말 노드의 후보군을 검색한 후 나머지 차원은 GPU를 이용하여 병렬 처리함으로써 고차원 색인에서도 안정적인 검색 성능을 제공할 수 있다.

[0010] 본 발명의 실시예에 따르면, G-tree에서 깊이 우선 검색 방식이 아닌, 너비 우선 검색 방식을 적용함으로써 GPU에서 병렬로 데이터를 처리하는데 방해요소가 되는 레벨 간의 데이터 의존성을 제거할 수 있다.

도면의 간단한 설명

[0011] 도 1은 CPU의 메모리 내에 저장되어 있는 R-tree의 노드 분포를 나타낸 것이다.

도 2는 배열을 활용한 노드 저장 방식을 설명하기 위한 도면이다.

도 3은 도 5의 노드 저장 방식에 대한 GPU의 접근 패턴을 나타낸 도면이다.

도 4는 깊이 우선 검색 방식과 너비 우선 검색 방식을 설명하기 위한 도면이다.

도 5는 본 발명의 일 실시예에 있어서, G-tree의 개략적인 개념도를 도시한 것이다.

도 6 및 도 7은 본 발명에서 제안한 G-tree와 GPU에서의 선형검색, R-tree 계열 중에서 고차원 인덱싱에 많이 사용되는 X-tree의 성능을 비교한 그래프이다.

도 8 내지 도 10은 G-tree와 GPU의 선형검색 성능을 비교한 그래프이다.

도 11은 차원 별 검색 수행 시간을 나타낸 그래프이다.

발명을 실시하기 위한 구체적인 내용

[0012] 이하, 본 발명의 실시예를 첨부된 도면을 참조하여 상세하게 설명한다.

[0013] 본 실시예들은 다차원의 공간 데이터를 저장하는 색인 자료구조인 R-tree 계열의 계층형 색인 구조에서 범위 질의를 처리하는 검색 시스템에 적용될 수 있다. 본 명세서에서, 'R-tree'는 R*-tree, X-tree, SS-tree 등 모든 R-tree 변종 계열을 통칭한 것이다.

[0014] 본 실시예에서는 R-tree 계열의 계층형 색인 구조에서 차원이 증가할수록 MBR의 중복이 심해지는 문제를 해결하여 고차원 색인에서 안정적인 검색 성능을 얻기 위한 기법을 제공할 수 있다.

[0015] 먼저, G-tree의 기본적인 구조와 범위 질의 병렬화 방법에 대해서 설명한다.

- [0016] G-tree의 노드 구조
- [0017] R-tree는 노드를 삽입할 때 CPU의 메모리에 새로운 공간을 할당하고 이곳에 노드를 저장한다. 이러한 노드 저장 방식은 R-tree의 노드들이 메모리 곳곳에 퍼져있게 만든다. 도 1은 CPU의 메모리 내에 저장되어 있는 R-tree의 노드 분포를 도시한 것이다.
- [0018] 하지만 이와 같은 CPU의 메모리 저장 방식은 GPU 메모리 접근 특성에 적합하지 않다. 따라서 GPU를 위해서 R-tree를 구성할 때는 노드들이 메모리에 선형적으로 정렬되어 있어야 한다. 이와 같은 저장 방식을 이루기 위해서 R-tree의 노드들을 배열에 선형적으로 저장한다. 그러나 단순히 배열에 노드들을 저장하는 것만으로는 최적의 메모리 읽기 성능을 이룰 수 없다.
- [0019] 도 2는 배열을 활용한 두 가지 방식의 차이를 보여준다.
- [0020] 도 2의 (a)와 같은 구조체를 배열에 저장할 때는 도 2의 (b)와 (c) 같은 방식으로 저장할 수 있다. 도 2의 (b)는 일반적으로 구조체를 배열 형태로 저장한 방식이다.
- [0021] 이와 같은 방식으로 저장할 경우 메모리에서 선형적인 접근은 가능하지만 메모리 트랜잭션 내에 하나의 워프가 정렬되어있지 못한다. 따라서, 추가적인 메모리 접근이 필요하게 된다. 하지만 도 2의 (c)와 같은 배열의 구조체 방식은 다수의 쓰레드들이 동시에 접근하는 구조체의 요소들(min_x, min_y 등)이 인접한 메모리에 존재한다. 그러므로, 이와 같은 저장 구조를 이용할 경우 선형적으로 정렬된 메모리 접근이 가능하게 되어 최적의 메모리 읽기가 가능하게 된다.
- [0022] 도 3은 도 2에 대한 GPU의 접근 패턴을 나타낸 그림이다.
- [0023] 도 3의 (a)는 도 2의 (b)에 대한 접근 패턴이다. 이와 같은 접근 패턴에서는 구조체의 각 요소들이 메모리에 퍼져있기 때문에 한번의 메모리 트랜잭션으로는 하나의 워프 내의 쓰레드들이 모두 필요한 데이터를 읽어 올 수 없다. 따라서 도 2의 (c)를 접근하는 방식인 도 3의 (b)와 같은 접근 패턴을 사용해야 한다. 그러나 도 3의 (b)에는 첫 번째 요소(min_x)에 대해서는 최적의 읽기를 이룰 수 있지만, 두 번째 요소(min_y)에 대해서는 2번의 읽기를 수행해야만 워프 내의 모든 쓰레드들이 필요한 데이터를 읽어 올 수 있게 된다. 그러므로 도 3의 (b)는 도 3의 (c)와 같이 최적화를 수행할 있다. 도 3의 (c)는 디스크 기반의 R-tree에서 주로 사용되는 디스크 블록크기(4KB)로 노드의 크기를 결정하는 것과는 다르게 각 노드 별로 요소의 개수를 32의 배수로 결정함으로써 최적의 메모리 읽기 성능을 얻을 수 있다.
- [0024] 범위 질의 병렬화 방법
- [0025] 일반적으로 R-tree에서는 깊이 우선 검색(Depth-First-Search, DFS)을 주로 사용한다. 깊이 우선 검색 방식은 질의 영역과 겹치는 노드를 발견하면 바로 자식 노드로 이동하여 검색을 진행한다. 이와 같은 검색 방식은 하위 레벨의 노드와 상위 레벨의 노드 사이의 이동이 빈번하게 일어나 상위 레벨 노드와 하위 레벨 노드 간의 데이터 의존성이 생긴다. 이와 같은 의존성은 GPU에서 병렬로 데이터를 처리하는데 방해요소로 작용한다.
- [0026] 따라서 데이터 의존성을 제거하기 위해서 G-tree에서는 깊이 우선 검색 대신 너비 우선 검색(Breadth-First-Search, BFS)을 사용한다. 너비 우선 검색은 각 레벨의 검색이 끝나기 전까지는 하위 레벨로 이동하지 않기 때문에 데이터 의존성을 크게 줄일 수 있다.
- [0027] 도 4는 깊이 우선 검색 방식과 너비 우선 검색 방식의 차이점을 도시하고 있다.
- [0028] 도 4의 (a)는 깊이 우선 검색 방식을 나타내는 그림으로 괄호의 숫자는 검색 순서를 나타낸다. 도 4의 (b)는 너비 우선 검색을 나타내는 그림으로 위에서와 마찬가지로 괄호의 숫자는 검색 순서를 나타낸다. 도 4의 (b)에서는 각 레벨을 동시에 처리함으로써 GPU에서의 병렬화가 가능함을 보여준다.
- [0029] 이하에서는, 고차원 색인 문제를 해결하기 위한 본 발명에 따른 범위 질의 처리 방법을 상세히 설명하기로 한다.
- [0030] G-tree 구조 및 검색 기법
- [0031] R-tree는 고차원으로 갈수록 MBR의 중복이 심각해지고, 이런 MBR의 중복은 R-tree의 검색 연산의 성능에 직접적인 영향을 미친다. 이와 같은 이유로 고차원의 데이터에 대해서 R-tree의 효율은 극도로 나빠지게 된다. 하지만 R-tree는 저차원(2개 내지 5개의 차원)일 때는 아주 훌륭한 인덱싱 성능을 발휘한다는 점을 활용하여 G-tree에서는 G-tree 구성 시에 R-tree의 인덱싱을 저차원에만 적용하여 R-tree의 성능을 최적으로 끌어내며 나머지

차원에 대한 검색은 GPU의 멀티 코어를 활용하여 병렬화 함으로써 성능 향상을 이루어낼 수 있다.

[0032] 이와 같은 G-tree의 개략적인 개념도가 도 5에 도시되어 있다.

[0033] 도 5를 참조하면, 본 실시예에서는 주어진 데이터의 전체 차원들(dimensions)(예컨대, 1D~nD) 중에서 저차원에 해당되는 일부(예컨대, 1D~4D까지 4개의 차원)만을 R-tree를 구성하는데 사용하고 나머지 차원들(예컨대, 5D~nD)은 R-tree 구성에 이용하지 않고 단말 노드에 저장을 해놓는다.

[0034] 이렇게 생성된 G-tree는 R-tree를 이용해 색인된 차원을 상기 '범위 질의 병렬화 방법' 항목에서 설명한 검색 방식을 이용해 병렬 검색하고 검색을 통해 걸러진 단말 노드의 나머지 차원을 GPU를 이용해 검사 후 최종 결과를 얻게 된다.

[0035] 아래 표 1의 환경에서 범위 질의 처리 과정을 실험하였다.

표 1

[0036]

CPU	i7-960 3.2GHz	
RAM	12GB	
GPU	GTX 580	
	코어	512
	RAM	1.5GB
	Compute Capability	2.0

[0037] 표 1과 같이, CPU는 인텔사의 i7-960으로 3.2GHz의 클럭을 가지고 있다. 사용된 PC의 메모리는 총 12GB이며, GPU는 Nvidia사의 GTX 580이다. GTX 580은 512개의 코어와 1.5GB의 메모리를 가지고 있으며, compute capability는 2.0이다. 실험에 이용된 데이터는 이미지에서 추출한 32~96 차원의 SURF 특징점으로 약 120만개의 데이터이다.

[0038] 도 6과 7은 본 발명에서 제안한 G-tree와 GPU에서의 선형검색, R-tree 계열 중에 고차원 인덱싱에 많이 사용되는 X-tree의 성능을 비교한 그래프이다.

[0039] 도 6은 32차원, 도 7은 64 차원에서의 실험을 나타낸 결과이다. 도 6과 도 7에 따르면, 각 그래프에서 가장 상단의 선이 X-tree로 G-tree나 GPU의 선형검색에 비해서 매우 느린 속도를 보여준다.

[0040] 도 8 내지 도 10은 G-tree와 GPU의 선형검색의 성능을 비교한 그래프이다.

[0041] 도 8은 32차원, 도 9는 64차원, 도 10은 96차원의 실험 결과를 나타낸 것이다. 각 도면에서 선택도 0~60%까지는 G-tree가 GPU의 선형검색보다 더 나은 성능을 보여준다. 하지만 60~100%구간에서는 G-tree의 성능이 GPU의 선형검색보다 나쁜데, 이것은 60%이상이 되면 상당수의 단말 노드가 선택되기 때문에 오히려 내부 노드의 인덱싱 효과가 부하로 나타나기 시작한다. 하지만 실제 질의에서 약 20%내외의 선택도가 주로 사용되기 때문에 실 사용에서는 문제가 없을 것으로 판단된다.

[0042] 도 11은 차원 별 수행시간을 나타낸 그래프이다.

[0043] G-tree, GPU의 선형검색 모두 차원에 대해서 비슷한 기울기로 선형적 시간 증가를 보여주는 것을 확인할 수 있다. 다만, G-tree의 경우 선택도에 따라서 절편의 위치가 크게 바뀌게 되고 선택도가 낮을수록 GPU의 선형검색에 비해서 좋은 성능을 보여준다.

[0044] 이와 같이, 본 발명의 실시예에 따르면, 고차원의 데이터가 주어질 경우 R-tree가 최적의 성능을 보여주는 저차원만을 색인하여 단말 노드의 후보군을 검색한 후 나머지 차원은 GPU를 이용하여 병렬 처리함으로써 고차원 색인에서도 안정적인 검색 성능을 제공할 수 있다.

[0045] 본 발명의 실시예에 따른 방법들은 다양한 컴퓨터 시스템을 통하여 수행될 수 있는 프로그램 명령(instruction) 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 본 발명을 위하여 특별히 설계되고 구성된 것들이거나 컴퓨터 소프트웨어 당업자에게 공지되어 사용 가능한 것일 수도 있다. 또한, 상술한 파일 시스템은 컴퓨터 판독이 가능한 기록 매체에 기록될 수 있다.

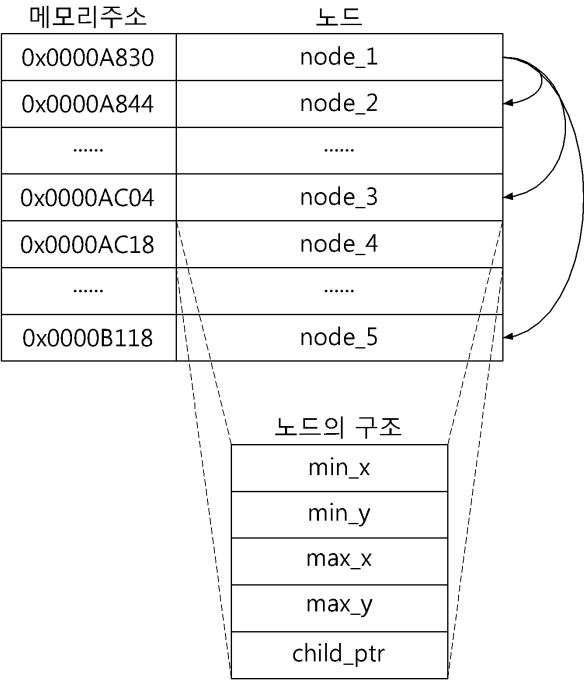
[0046] 이상과 같이 본 발명은 비록 한정된 실시예와 도면에 의해 설명되었으나, 본 발명은 상기의 실시예에 한정되는

것은 아니며, 본 발명이 속하는 분야에서 통상의 지식을 가진 자라면 이러한 기재로부터 다양한 수정 및 변형이 가능하다.

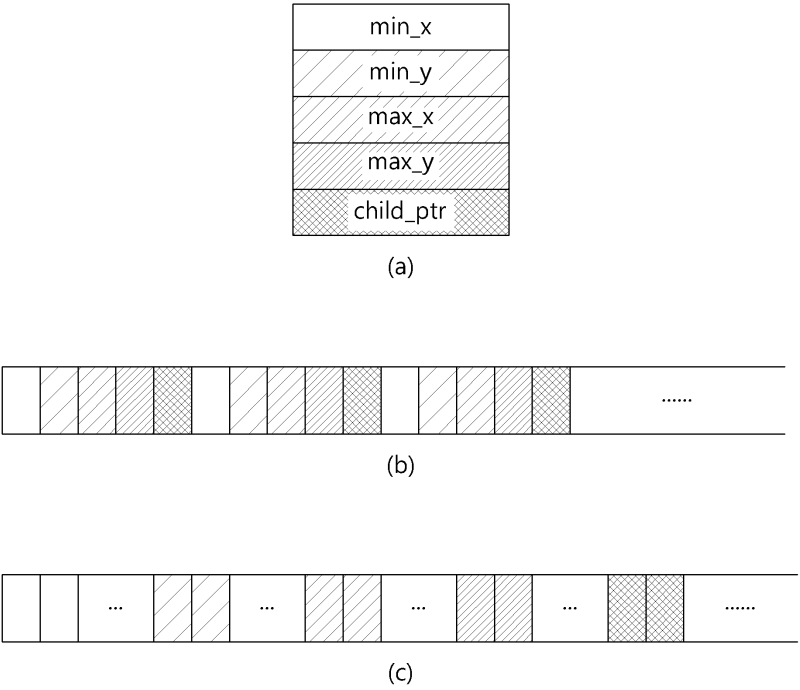
[0047] 그러므로, 본 발명의 범위는 설명된 실시예에 국한되어 정해져서는 아니 되며, 후술하는 특허청구범위뿐 아니라 이 특허청구범위와 균등한 것들에 의해 정해져야 한다.

도면

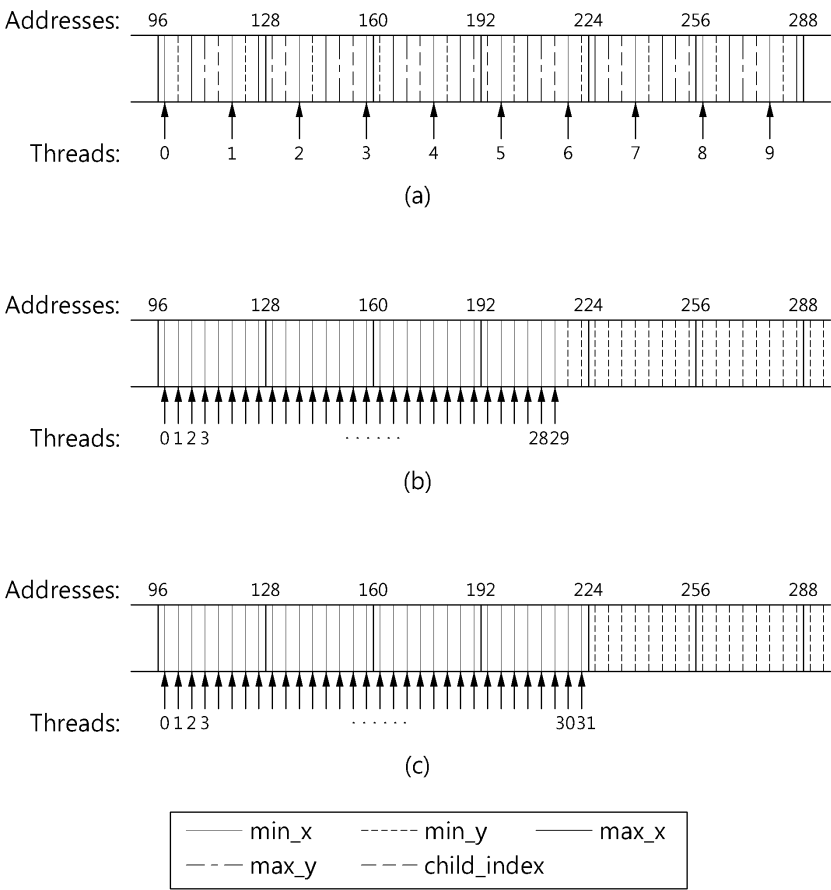
도면1



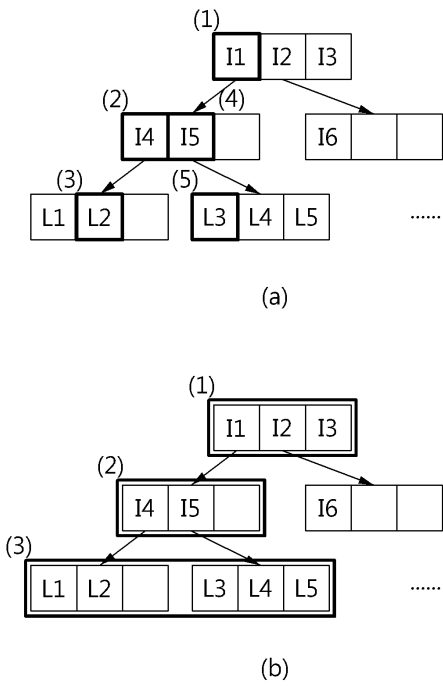
도면2



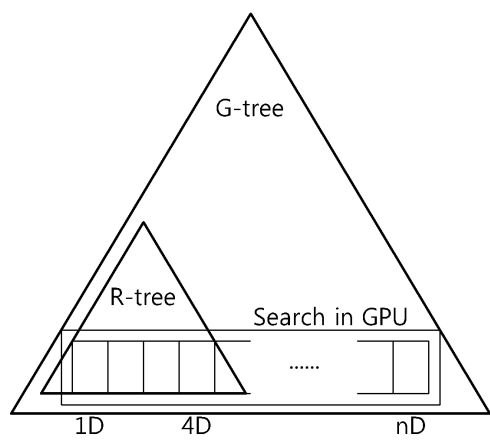
도면3



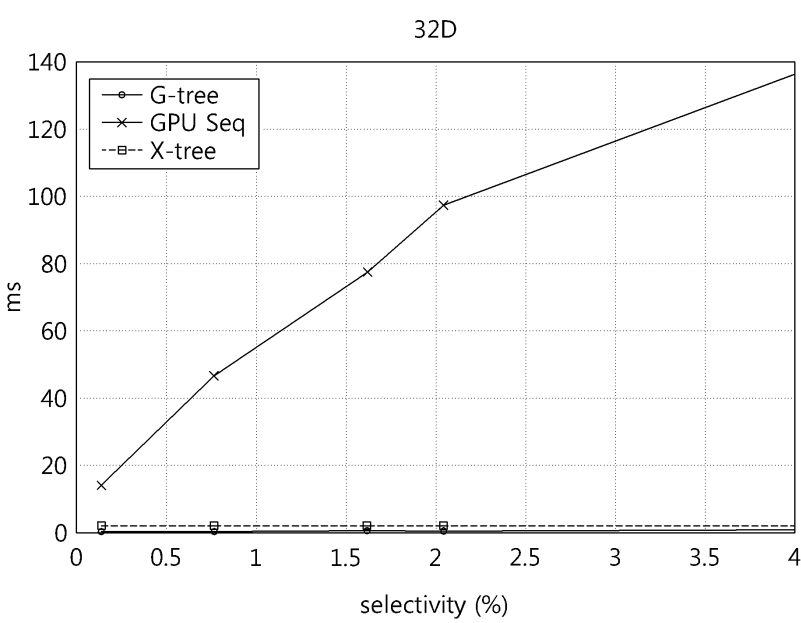
도면4



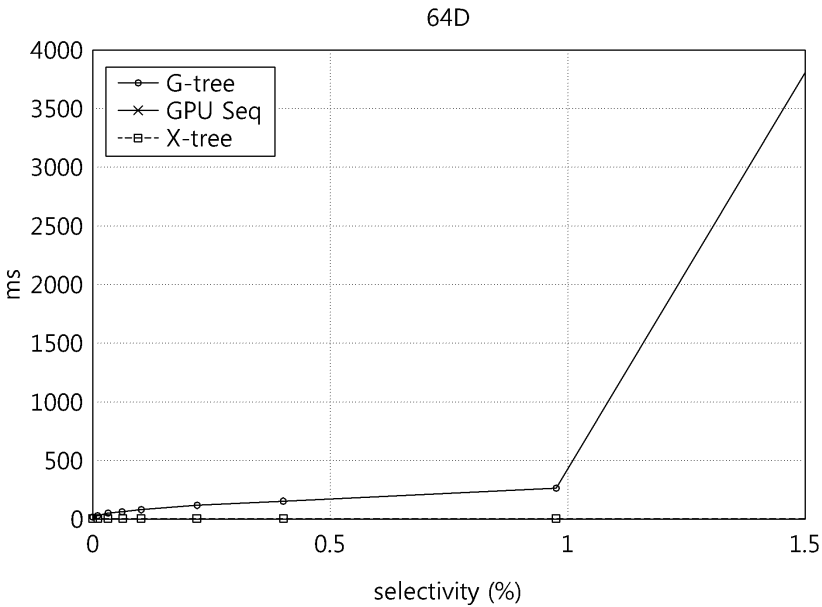
도면5



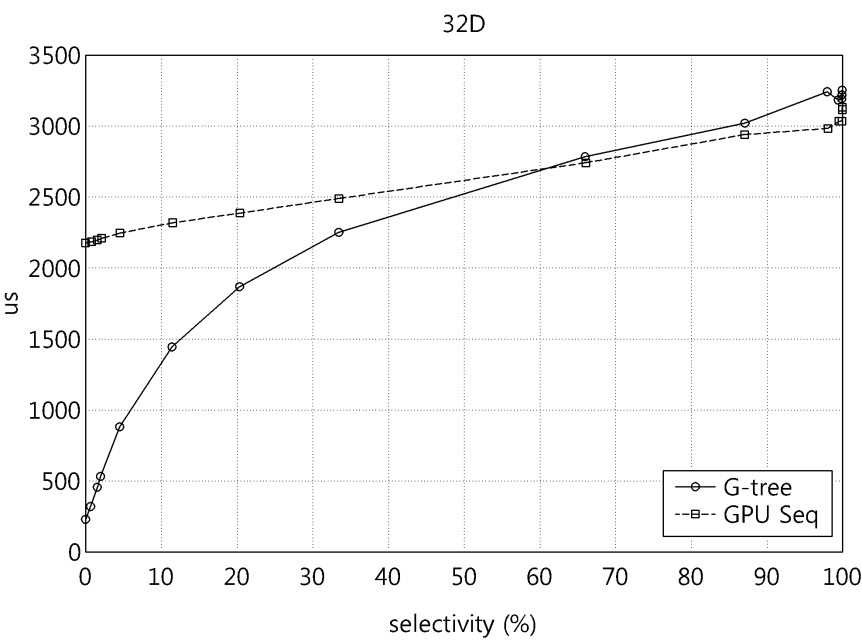
도면6



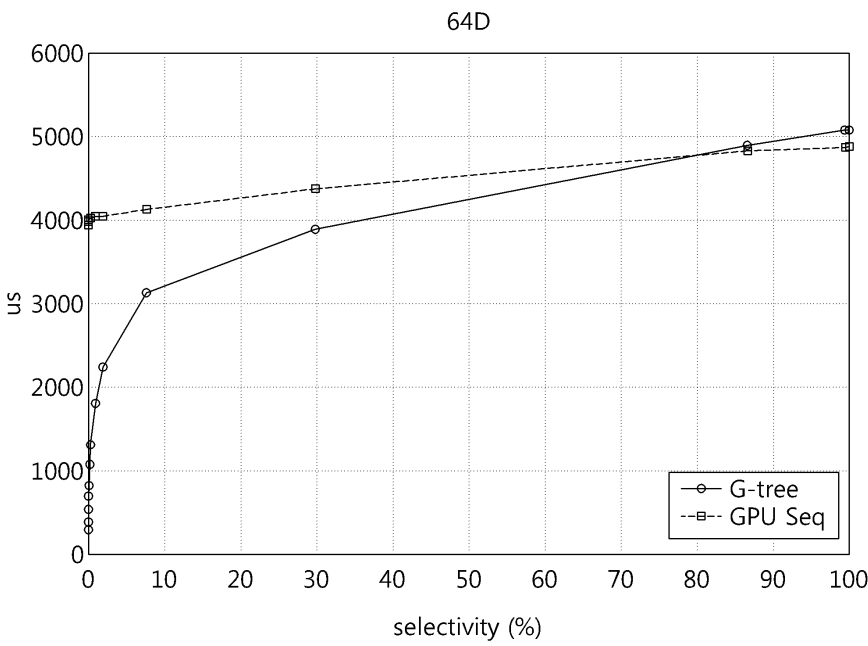
도면7



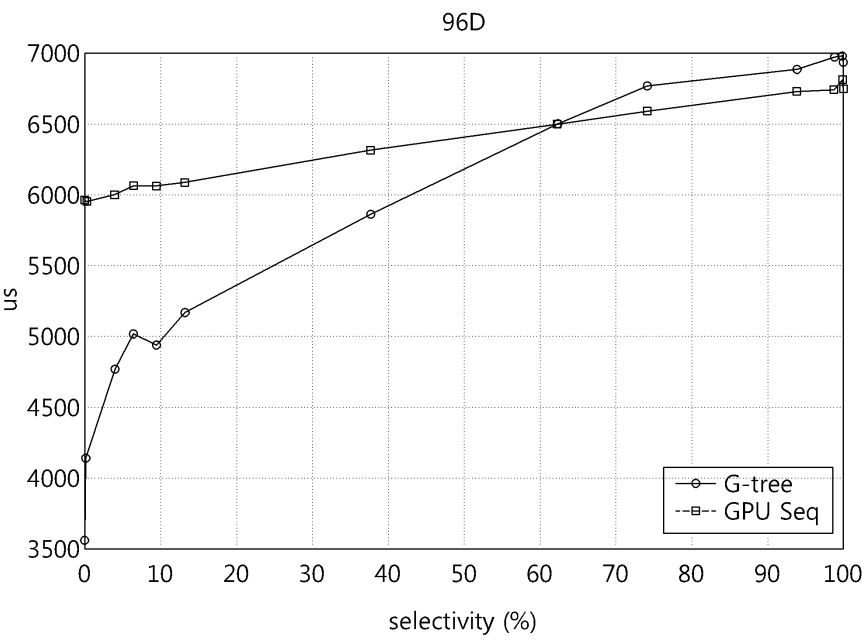
도면8



도면9



도면10



도면11

