ICDE papers 2

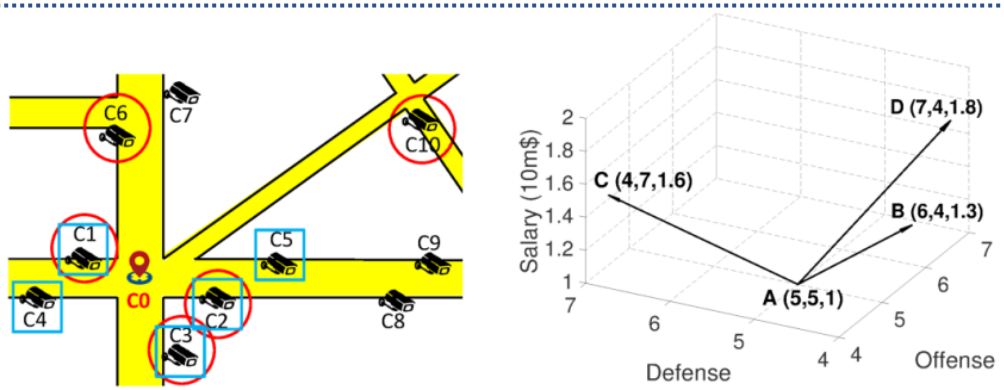
논문 요약

Contents

Importance of the diversity in the similarity search

- 유사성 검색(Similarity search)는 정보 검색(retrieval)과 데이터 마이닝 어플리케이션에서 중요한 task
  - 주로 object들은 다차원 공간에서 점(point)들로 표현됨
  - 유사성 검색에서는 Query가 점 형태로 주어졌을 때, query와 가장 유사한 K개의 근접한 이웃(K-nearest neighbor)들을 찾음
    - 유사성 측정을 위해 Cosine similarity 또는 Euclidean distance를 사용
- 유사성 뿐만 아니라 다양성(diversity) 역시 query 결과에 중요한 의미를 부여한다는 것이 언급되고 있음
  - Query가 목적하는 것이 모호(ambiguous)할 경우, 유사성 검색에서 불필요한 유사 결과가 나올 수 있기에 다각적인 결과를 요구할 필요가 있음

    예시) 검색 엔진 또는 추천 시스템에서, Keyword: “Apple” ; Company? Fruit?



(a) Camera Access (b) NBA Player Trade

Figure 1: Introduction Example

a. c0 구역에서 범죄 발생, 5-근접 이웃 검색 시, 북쪽과 북동쪽에서의 정보는 잃게 됨

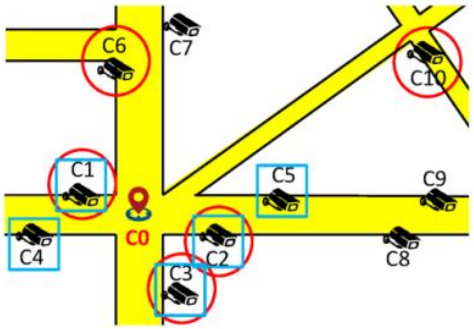
b. NBA 선수를 교환하는 경우, 선수들의 능력치 또는 연봉에 따라서 다양한 가능성을 고려해볼 필요가 있음

Problem of the existing diversity measurement

- 거리 기반 다양성 측정 방식은 방향까지 고려할 것이라는 보장이 없음

  예시) Camera Access 예시에서의 C2와 C9 그리고 r-DisC (*M. Drosou, et al. 2014*)를 이용한 다양성 측정

- 높은 차원 공간에서는 concentration 현상(phenomenon)에 의해서, point들을 구분하기 힘들어 짐
  - 모든 점들 간의 거리가 굉장히 유사해 질 것임



(a) Camera Access

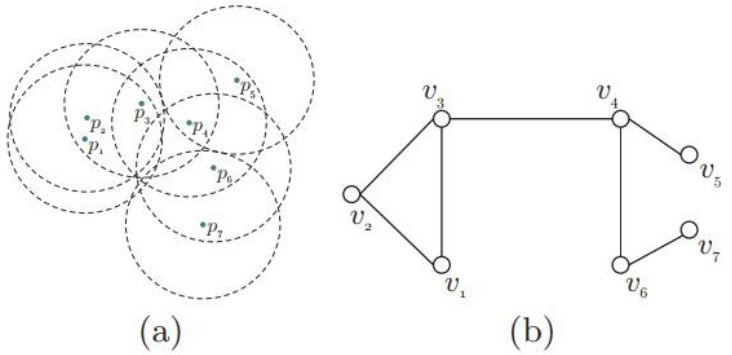
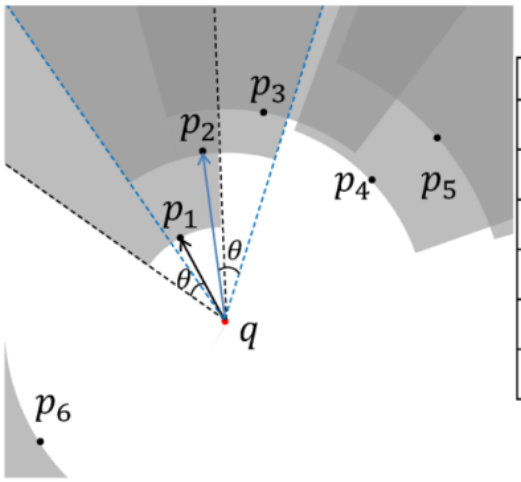


Figure 3: (a) Minimum r-DisC diverse subsets for the depicted objects: $\{p_1, p_4, p_7\}$, $\{p_2, p_4, p_7\}$, $\{p_3, p_5, p_6\}$, $\{p_3, p_5, p_7\}$ and (b) their graph representation.

Proposed method to find diverse neighbor

- 논문에서는 공간 각도(spatial angle)를 기반으로 한 다양성의 새로운 관점을 제시함
  - Query 점 q가 주어졌을 때,
    q를 각기 다른 방향에서 포함하는 가까운 점들의 집합을 찾는 것
    - 해당 집합내에 있는 점들을 angular diverse neighbors라 부름
- Angular diverse neighbor를 찾기 위해서는 다른 점들에게 dominate 되지 않는 점을 찾으면 됨
  - Query 점 q가 주어졌을 때, 점 p가 점 p'를 dominate 하려면 두가지 조건이 만족해야 함
    1. 점 p가 점 p' 보다 점 q 에 가까이 있음
    2. 각도 $\angle pqp'$ 가 주어진 angular threshold보다 작음



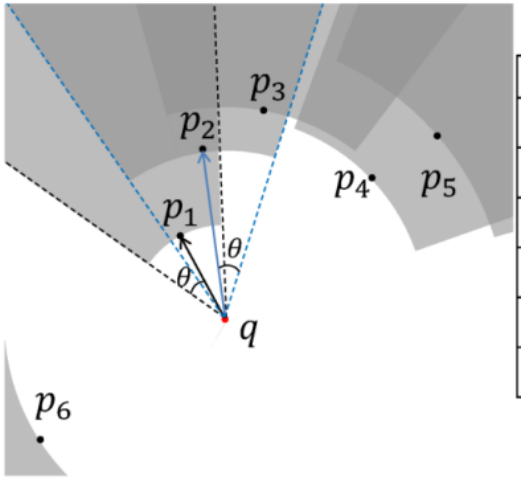
Point	coordinates	$dist(\cdot, q)$	$min\theta(\cdot, q, D)$
p_1	(-2, 4)	4.47	180°
p_2	(-1, 8)	8.06	19.44°
p_3	(2, 10)	10.20	18.43°
p_4	(6, 6)	8.49	52.13°
p_5	(10, 9)	13.46	3.01°
p_6	(-11, -6)	12.53	92.05°

Figure 2: An illustrative example, where $\theta = 20°$

- Angular threshold가 $20°$ 일때, Angular diverse neighbor 는 p_1, p_4, p_6
- 이를 위한 알고리즘의 복잡도는 $O(n^2)$
- 적절한 Angular threshold를 찾는 것 역시 또 다른 문제

Improved proposed method

- Sorted-Scan Algorithm : $O(n^2)$
  - Query q에 가까운 순서대로 점 p들을 정렬 후, angular diverse neighbor를 찾기 위해 각 점들을 하나씩 검사 → 비 효율적인 복잡도
- Two-Scan Algorithm : $O(|R|n + n^2)$
  - 일부 점들을 선택하여 Reference 집합인 R을 생성 후, R에 속한 점들에게 dominate되지 않는 점들을 후보군으로 선택
  - 다시 한번 후보군들을 R을 제외한 다른 점들과 비교하여 dominate 되는지 검사 후 최종 angular diverse neighbor 선정
  - Reference 점 집합 R을 어떻게 선택하냐 에 따라서 효율이 달라짐



Point	coordinates	$dist(\cdot, q)$	$min\theta(\cdot, q, D)$
p_1	(-2, 4)	4.47	180°
p_2	(-1, 8)	8.06	19.44°
p_3	(2, 10)	10.20	18.43°
p_4	(6, 6)	8.49	52.13°
p_5	(10, 9)	13.46	3.01°
p_6	(-11, -6)	12.53	92.05°

Figure 2: An illustrative example, where $\theta = 20°$

- Two-Stage Algorithm
  - k개의 angular diverse neighbor를 선정하는 알고리즘
  - Minimum dominated angle 개념을 이용
  - Angular threshold가 증가할 수 록 angular diverse neighbor 개수가 감소한다는 점을 응용함

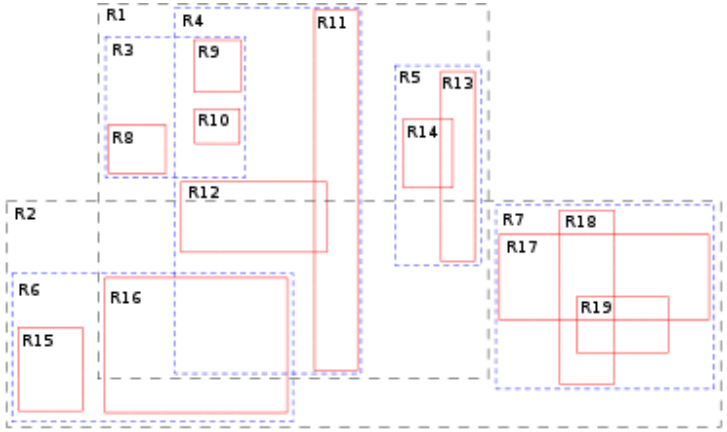
Contents

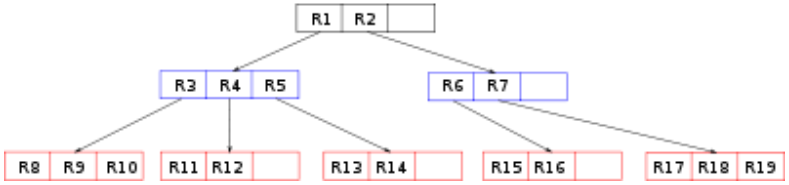
1. Finding Diverse Neighbors in High Dimensional Space

2. Improving Spatial Data Processing by Clipping Minimum Bounding Boxes

Minimum Bounding Boxes

- 공간 데이터를 분석 및 처리할 때, Oracle Spatial, IBM Informix 등 여러 데이터베이스들은 주로 R-Tree를 이용한 공간 처리 기술이 진행되고 있음
  - R-Tree는 공간 접근(spatial access) 방식으로 쓰이는 트리 구조
    - 주로 지리 좌표계, 사각형 또는 다각형(polygon) 과 같은 다차원 정보를 색인(index)하기 위해 사용됨
  - R-Tree의 key idea는, 핵심 요소인 MBB(minimum bounding box)를 이용해, 가까이 위치해 있는 객체끼리 묶은 뒤, 상위 레벨의 트리가 하위 레벨 트리의 MBB를 포함하게 구성





- MBB는 d차원 데이터를 포함하는, 축과 평행한 가장 작은 사각형
  - MBB는 (Ⅰ) 간단하게 계산할 수 있고, (Ⅱ) 저장을 위해서 오직 두개의 점을 필요로 하며, (Ⅲ) MBB간 overlap/intersection을 확인하기 위한 비용이 매우 저렴함
  - 위와 같은 장점 덕분에, MBB intersection는 공간 indexing에서 가장 자주 사용되는 연산으로 자리잡음

Problem of the existing MBBs in R-tree variant

- 공간 데이터 partitioning의 질은 일반적으로 MBB의 coverage와 overlap으로 측정됨
- Overlap: MBB끼리 덜 겹칠수록 좋음

  예) Query 사각형이 overlap된 MBB들과 intersect 한다면, tree의 여러 갈래를 따라 내려가야 함
- Coverage: MBB가 불필요한 공간을 차지하는 것을 피해야함

  예) 필요하지 않는 공간이 많을 수록 query 사각형과 겹칠 수 있는 공산(likelihood)이 증가
  - Dead space: 어떠한 object도 포함하고 있지 않은 node 용량의 percentage
- 실제 데이터를 이용한 실험에서, R-tree variant들은 MBB의 overlap 현상을 완화하는데 성공했지만 dead space 문제와 그에 따른 I/O 최적화를 다루는 데에는 여전히 어려움을 겪음

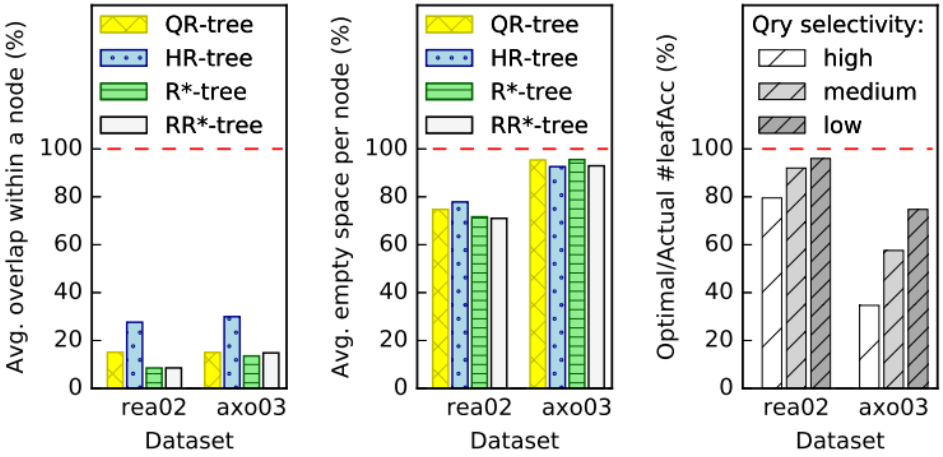


Fig. 1: Performance of four R-tree variants.

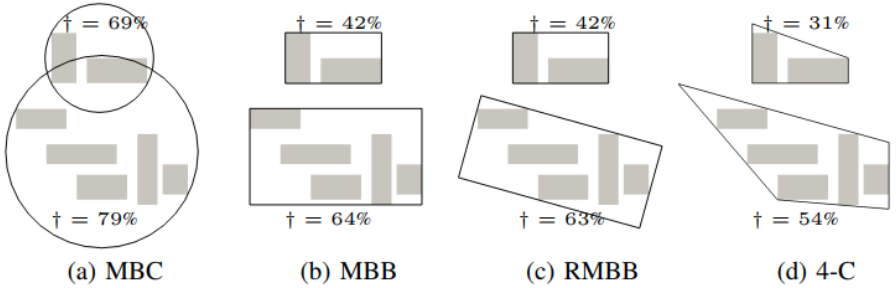


Fig. 8: Visualization of different bounding methods over the two leaf nodes and their dead space (†).

Clipping proposal: clipped bounding boxes

- 논문에서 제안하는 Clipped bounding box는 MBB의 모서리들을 사각형 형태로 잘라낸 간단하며, non-convex 인 다각형
- 각각의 clip(잘라낸 곳)은 한개의 d차원 점과 d-bit flag로 표현함
- 모서리를 잘라내는 것은 기존의 MBB의 관점에서는 추가적인 개념으로, 다른 어떠한 R-tree variant에도 플러그인 형태로 적용될 수 있는 장점을 지님
- Clip point는 Pareto 최적화 방식(i.e. skylines in database)에 기반하여 생성되며, 이렇게 만들어진 point들은 작은 auxiliary 데이터 구조로 관리되므로 적은 overhead를 발생시킴

Figure 2: Skyline of Manhattan

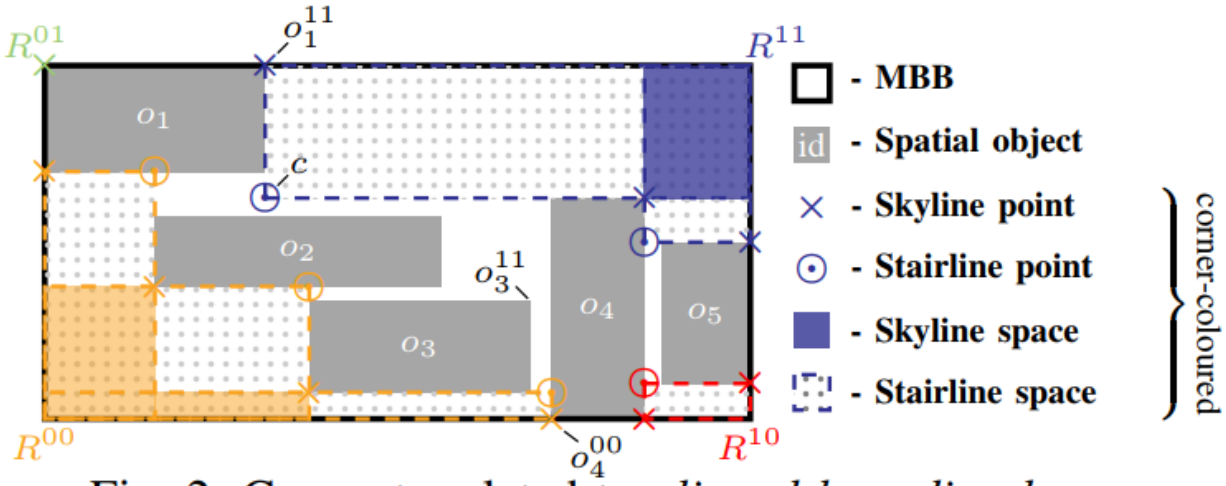
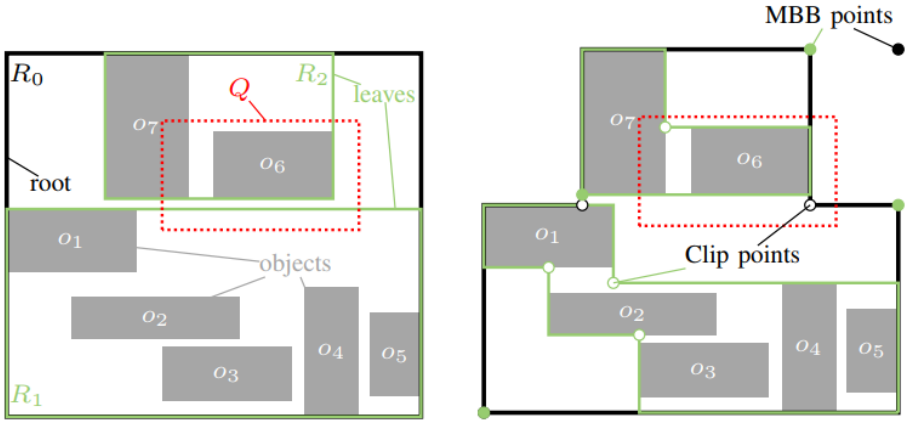


Fig. 2: Concepts related to *clipped bounding boxes*.

Layout and structure of clipped R-trees

- (노드 당 가질 수 있는 entry의 수) $M = 5, \mathrm{m} = 2$ 인 traditional R-tree에 clip 개념을 적용한 경우를 나타낸 Figure 3
  - Query Q는 (a)에서 R_1의 dead space에 부분적으로 intersect 하기 때문에, 총 3개의 node(R_0, R_1, R_2)를 스캔해야 하지만, MBB가 clip된 (b)에서는 R_1의 스캔을 피할 수 있음
- R-tree를 clip하는 경우, 기존의 데이터 구조는 *정확히* 유지하고, 해당 구조에 추가적인 디렉토리 테이블(auxiliary structure)을 추가함
  - 이 테이블은 R-tree 노드의 id로 index 되며 (entry 1 → R_1), array pointer를 가짐
  - Array에는 clip point의 bitmask와 point의 실제 좌표가 들어있음
  - 빠르게 Intersection을 감지하기 위해 clip된 정도의 크기를 기준으로 정렬되어 있음



(a) A two-level R-tree (b) A clipped R-tree

Fig. 3: An example of an R-tree before (a) and after (b) clipping, given 7 objects, o_1–o_7 and a range query, Q.

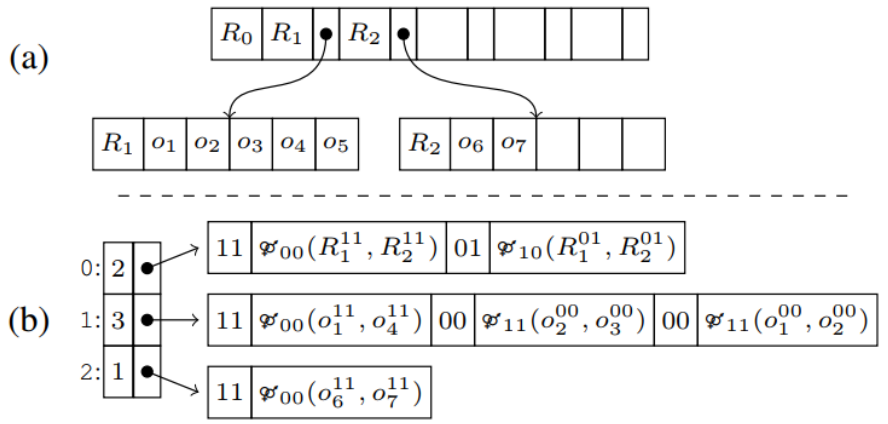


Fig. 4: The physical layout of the R-tree from Figure 3a (a) and the auxiliary structure (b) of clip points introduced in Figure 3b.