스카이라인 빠르게 계산하기

Efficient Skyline and Skycube Computation Using Point-Based Space Partitioning

이 종 욱
포항공과 대학교
정보 및 데이터베이스 시스템 연구실

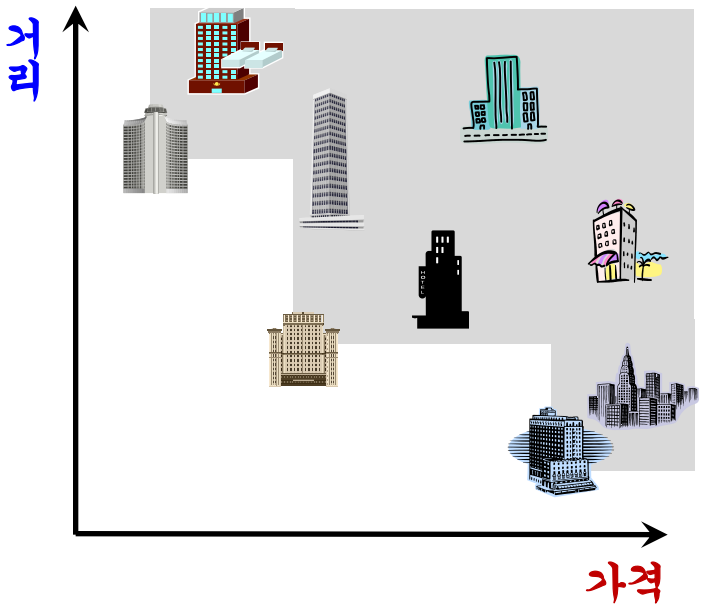
2011년 1월 7일

줄거리

- 스카이라인이 뭐지??
- 점 기반 공간 분할
- 스카이라인 계산 방법
- 성능 실험
- 좀 더 생각하기
- 결론

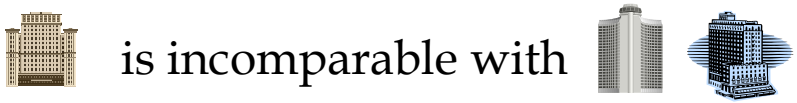
스카이라인??

- 마리나 리조트에서 가깝고! 가격이 싼! 횟집 찾기
  - 스카이라인 – 다른 점들에 종속되지 않는 점들의 집합
    (***Skyline***: a set of points that are ***not dominated by*** any other points.)



a **dominates** b.
→ a is no worse than b on all dimensions.

a is **incomparable** with b.
→ a is not dominated by b and vice versa.



기존의 계산 방법 (1 / 3)

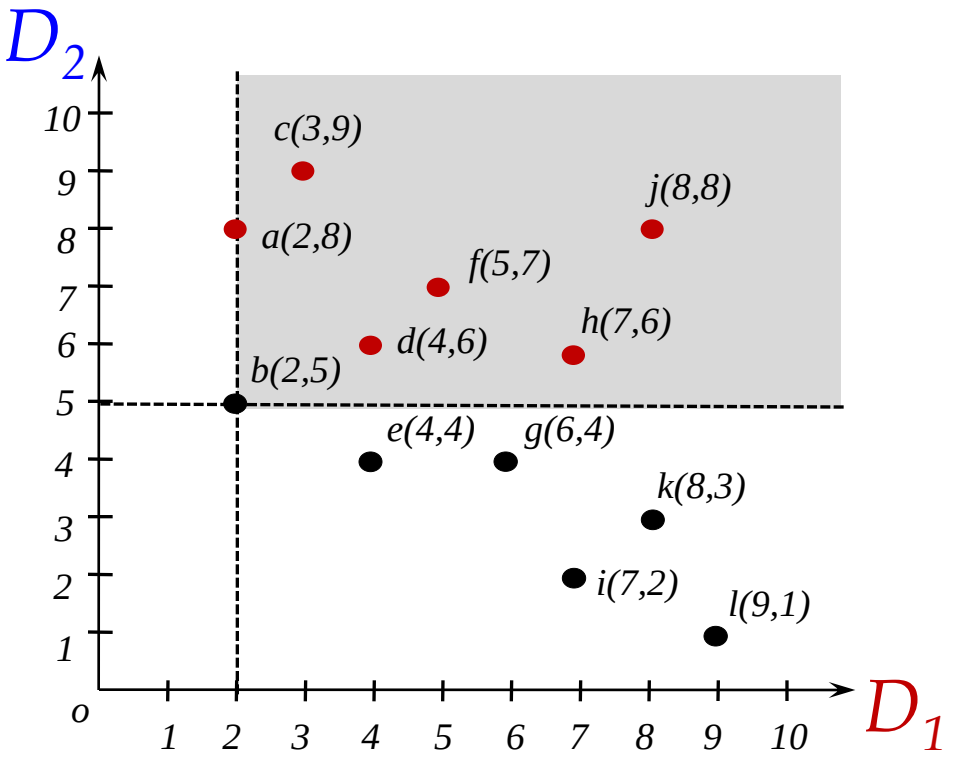
- The categorization of existing algorithms
  - **Sorting-based algorithms**: *optimizing point order* to prune out non-skyline points early on.
  - **Partitioning-based algorithms**: *grouping points into subregions* to perform region-based dominance comparisons.

They focused solely on optimizing dominance!!

- Our work
  - Exploit **point-based space partitioning** to optimize both **dominance** and **incomparability**.

기존의 계산 방법 (2 / 3)

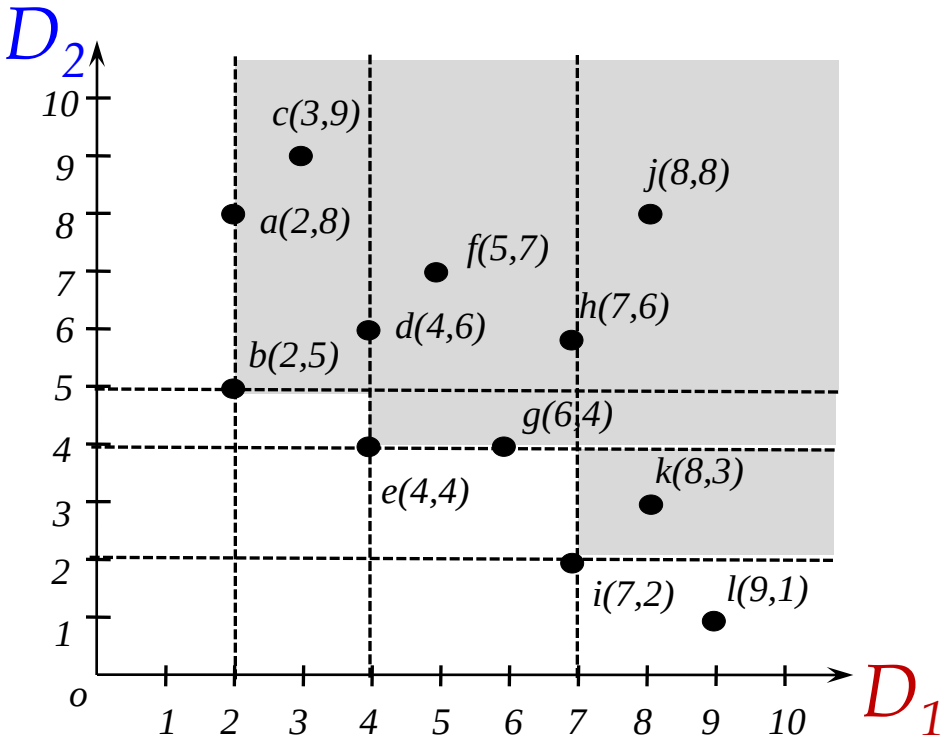
- Sorting-based algorithm – SFS
  - Access points in decreasing order of **dominance regions**.
  - Prune out non-skyline points early on.



b(40), *e(36)*, *f(35)*, *g(24)*, ...

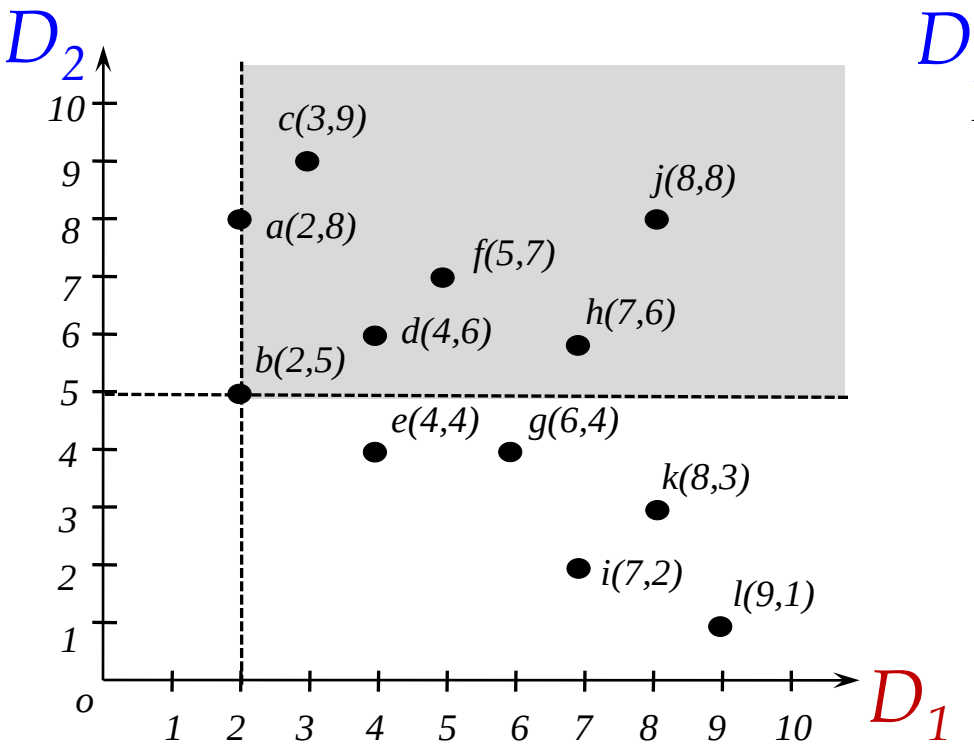
기존의 계산 방법 (3 / 3)

- Partitioning-based algorithm – BBS
  - Access points in descending order of **Manhattan distances**.
  - Remove points in the dominance region at once.

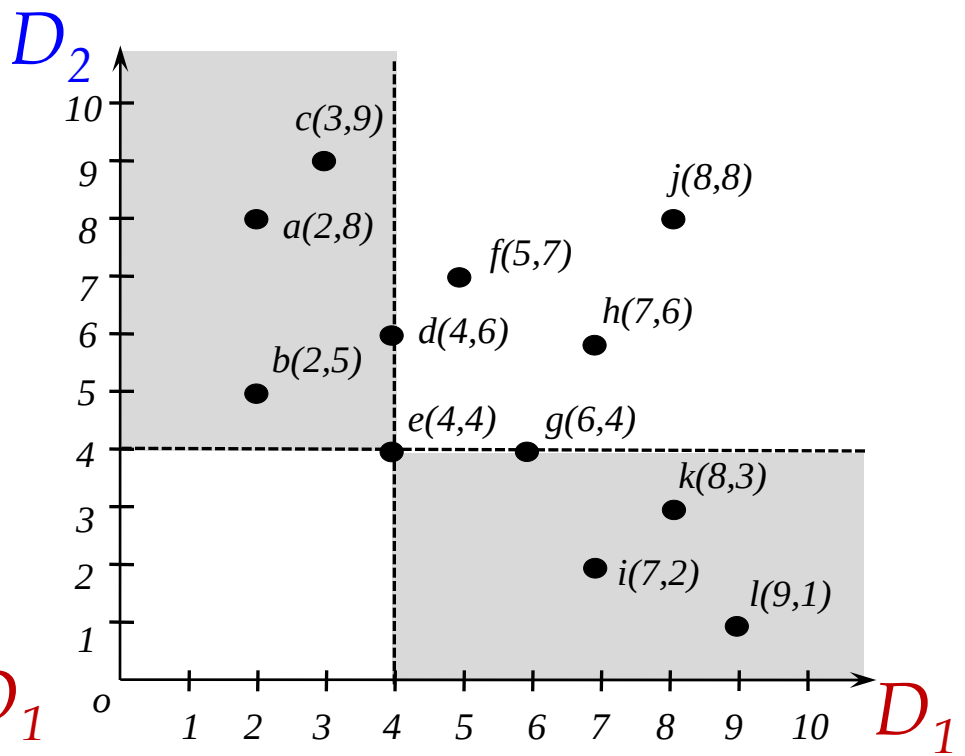


b(7), e(8), i(9), l(10), ...

비교 불가능성이 매우 중요!! (1 / 2)



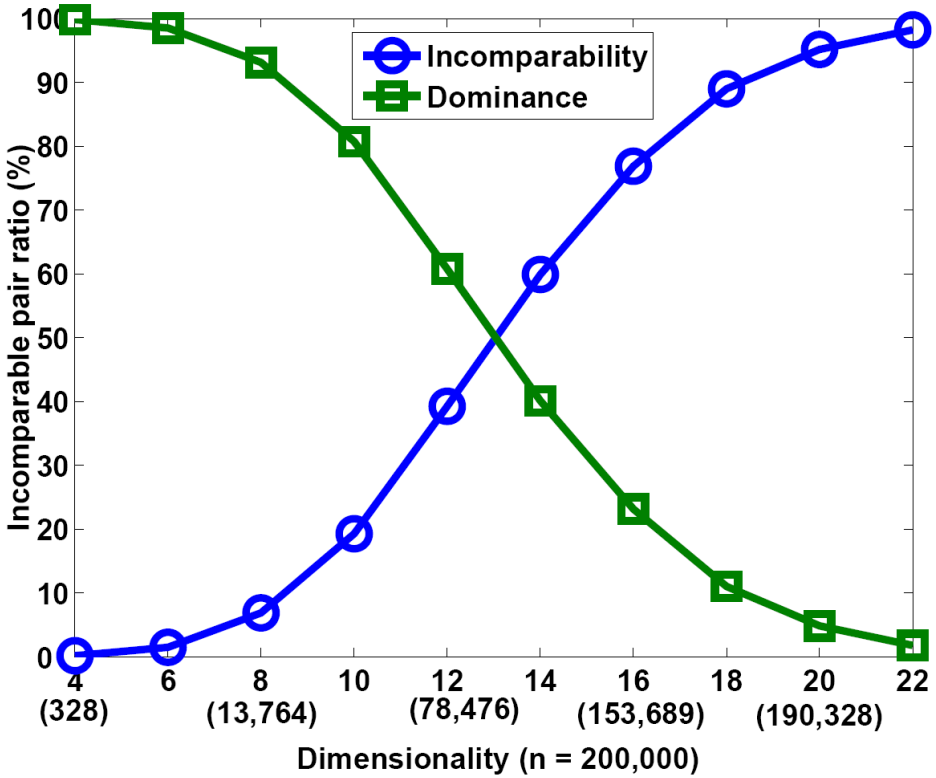
Dominance region



Incomparable regions

비교 불가능성이 매우 중요!! (2 / 2)

- Must treat **incomparability** as the *first-class citizen* for cost optimization, especially, for *high-dimensional* data.



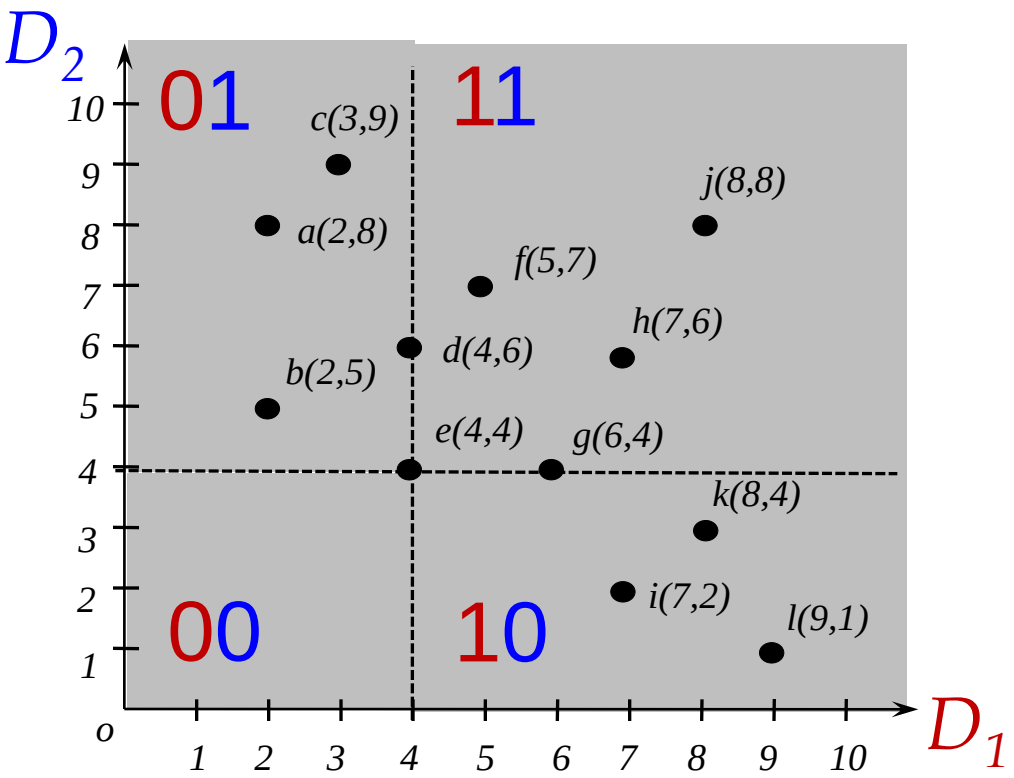
200K points with uniform and independent distribution

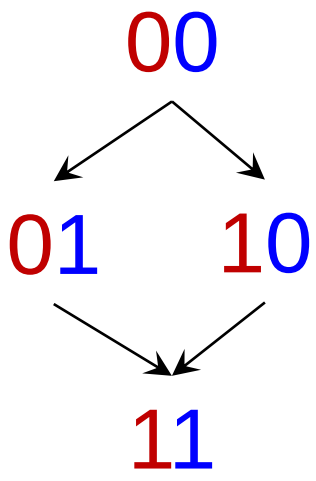
줄거리

- 스카이라인이 뭐지??
- **점 기반 공간 분할**
- 스카이라인 계산 방법
- 성능 실험
- 좀 더 생각하기
- 결론

점 기반 공간 분할?!

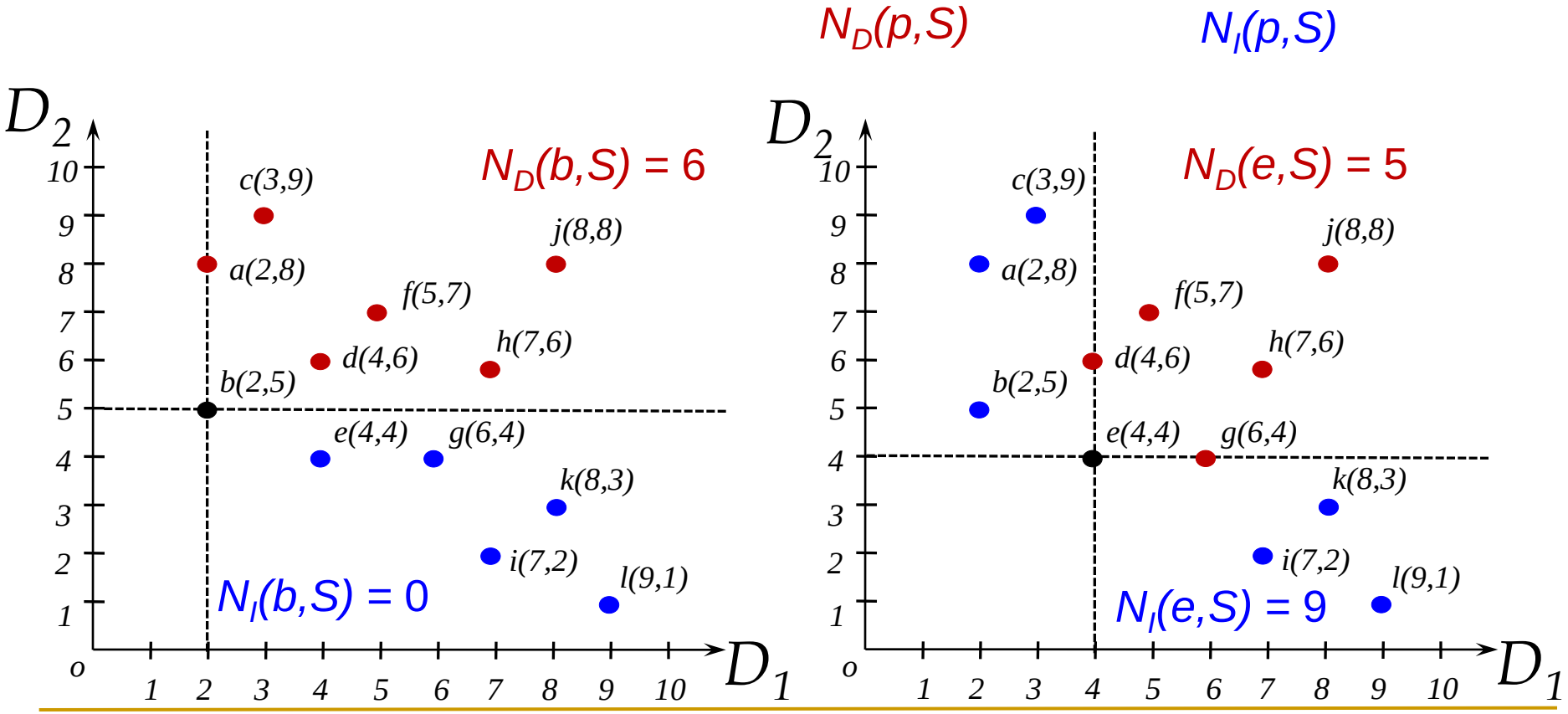
- Mapping points into subregions
  - For a point ***e***, each point is mapped into a ***binary vector***.





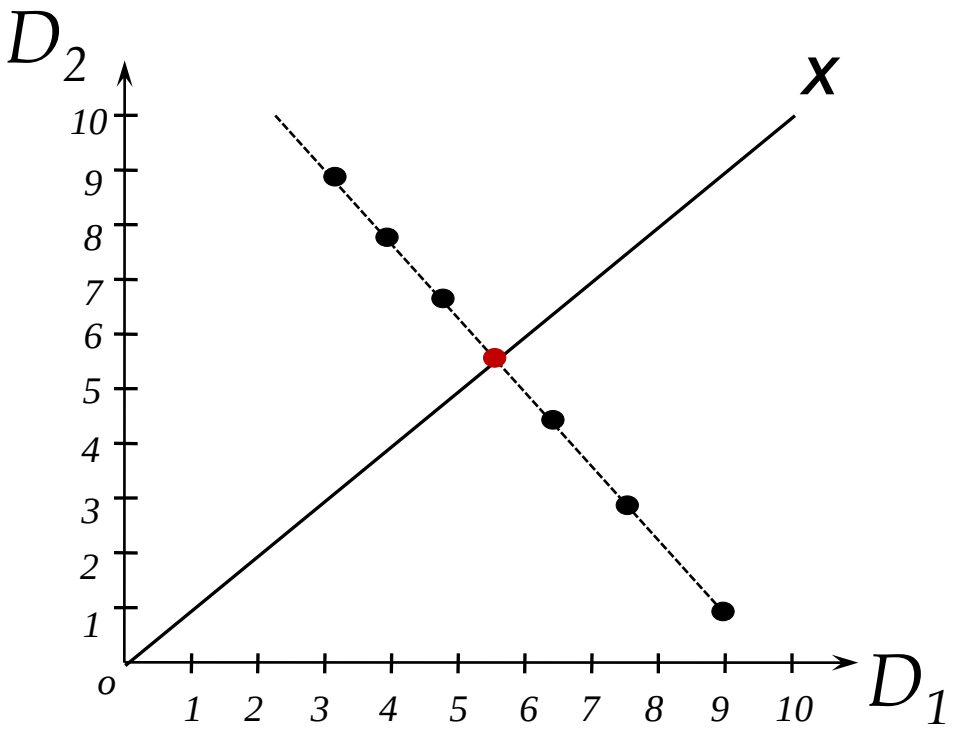
어떻게 공간 분할 점을 선택하지?? (1 / 2)

- Which point is better as a pivot point?
  - Select a point maximizing both **dominance** and **incomparability**



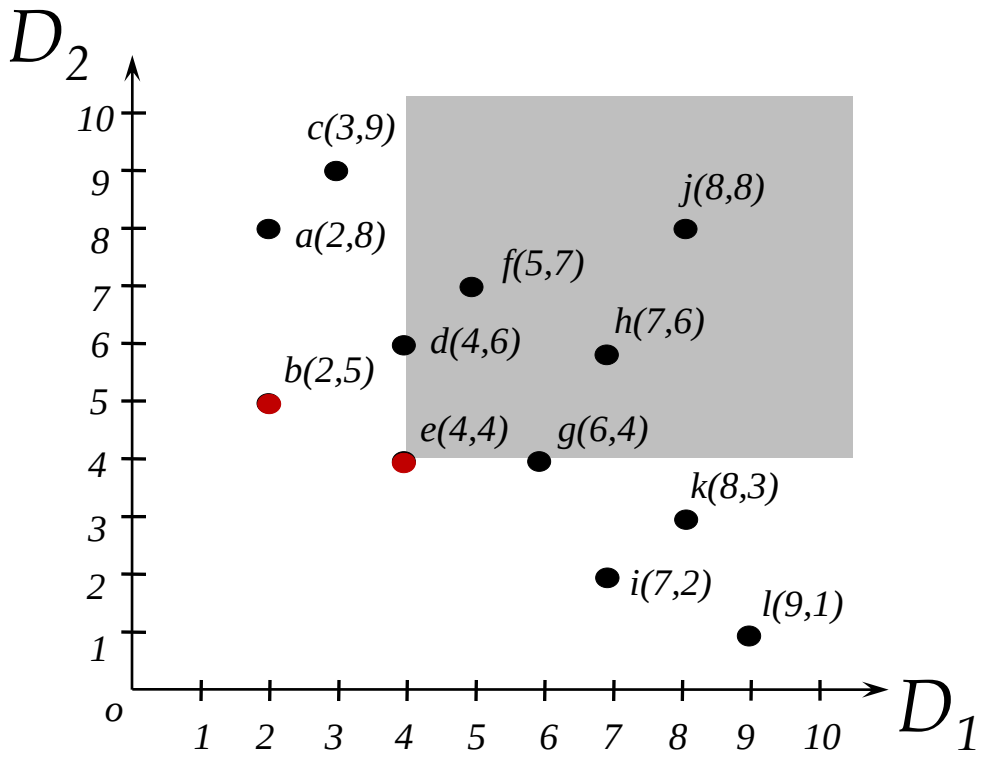
어떻게 공간 분할 점을 선택하지?? (2 / 2)

- Using the projection on a diagonal line ***X***, find a point with **higher incomparability**.
  - Assume that data distribution is **uniform** and **independent**.



가장 좋은 공간 분할 점 찾기 예제

- Suppose that points are accessed in alphabetical order.
  - Access points in a ***linear scan*** manner.
    - Find a point that is **not dominated** and **well-balanced**.

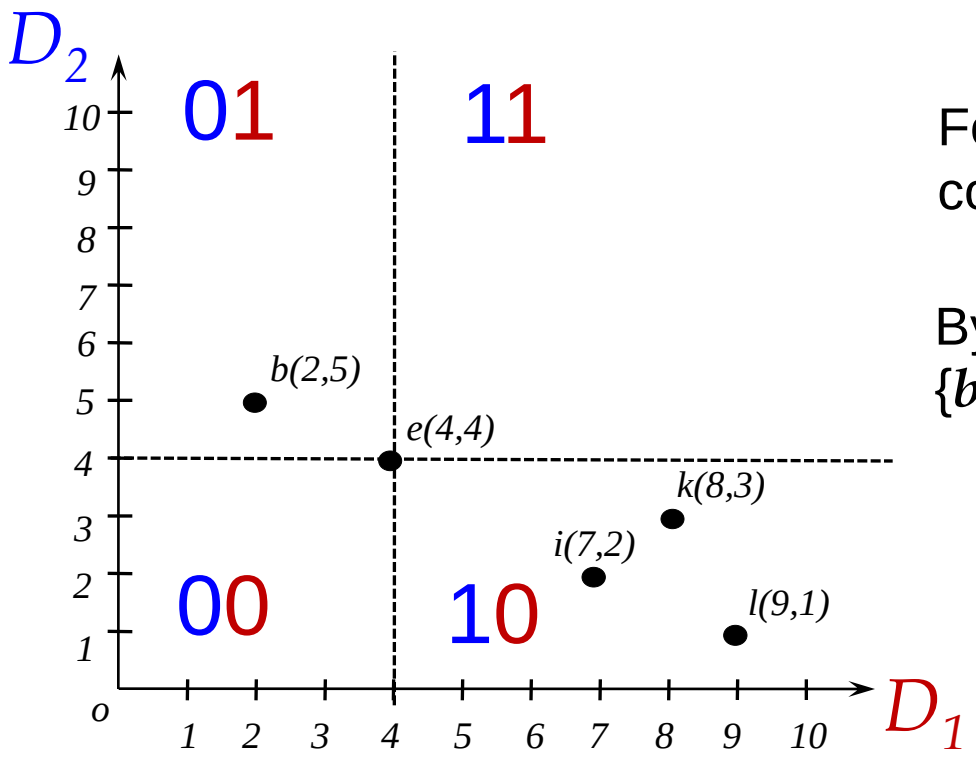


줄거리

- 스카이라인이 뭐지??
- 점 기반 공간 분할
- **스카이라인 계산 방법**
- 성능 실험
- 좀 더 생각하기
- 결론

BSkyTree-S

- Sorting-based algorithm using our proposed pivot point selection

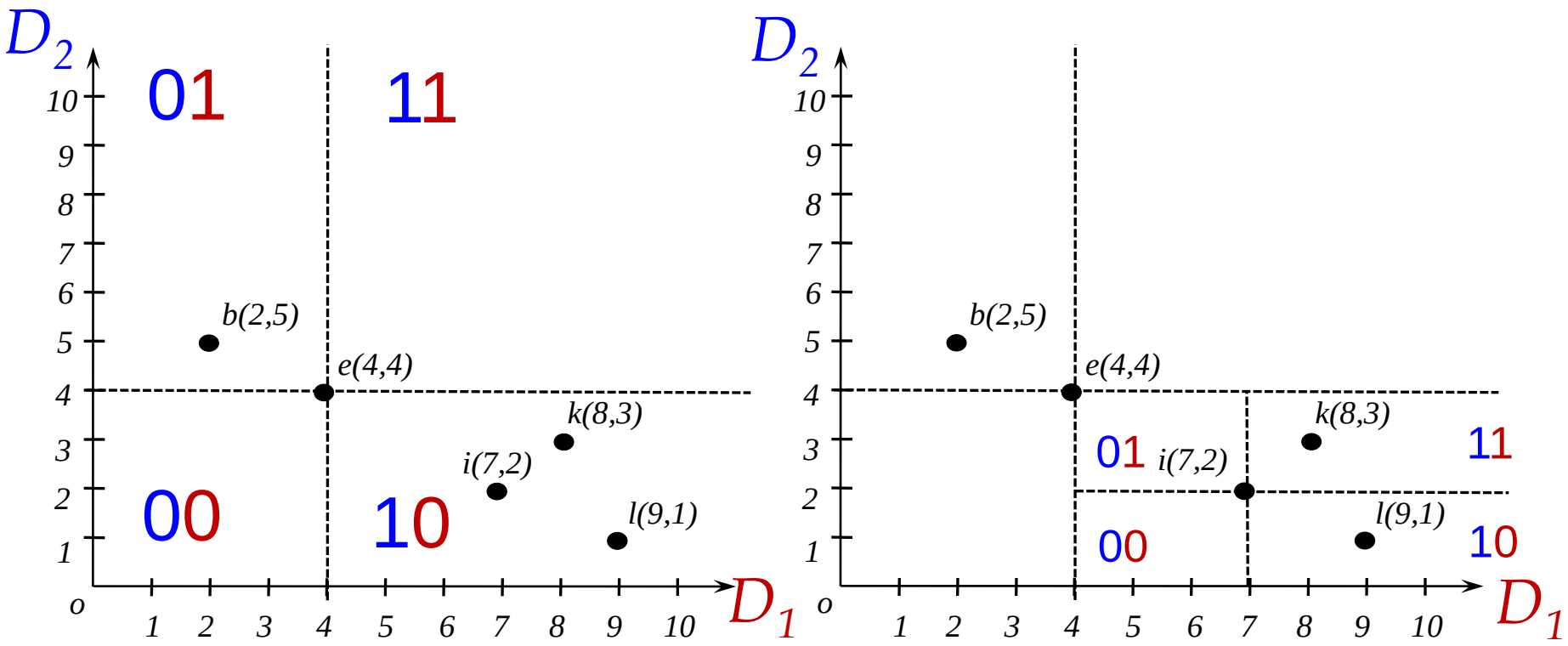


For remaining points, point-wise comparisons are performed.

Bypass the comparisons between $\{b\}$ and $\{i, k, l\}$.

BSkyTree-P

- Partitioning-based algorithm using our proposed pivot point selection: **recursive** partitioning



즐거리

- 스카이라인이 뭐지??
- 점 기반 공간 분할
- 스카이라인 계산 방법
- **성능 실험**
- 좀 더 생각하기
- 결론

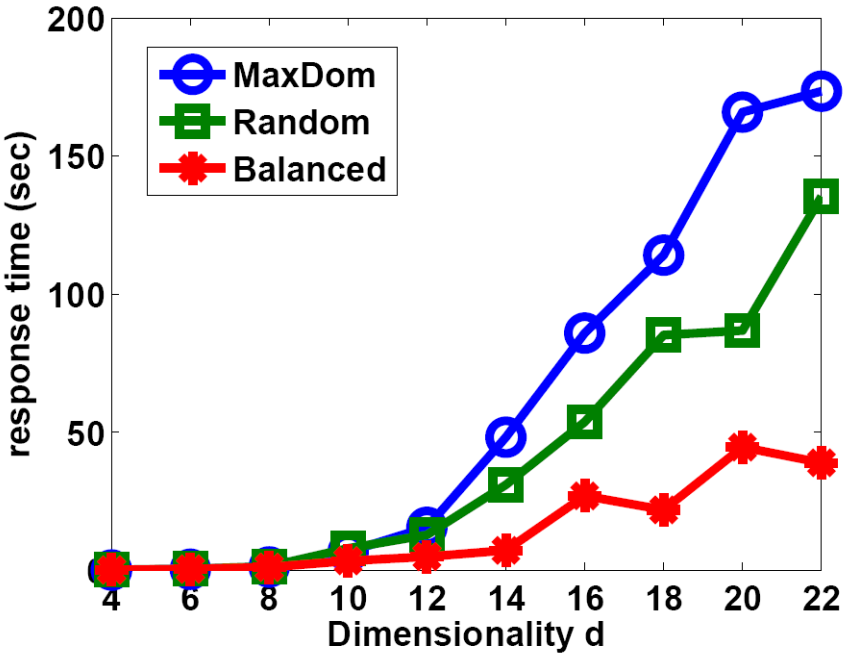
성능 실험 (1 / 4)

- Experimental settings
  - Distribution: Independent, Anti-correlated
  - Dimensionality *d*: 4 ~ 22 (default *d* = 12)
  - Cardinality *n*: 200K ~ 1,000K (default *n* = 200K)

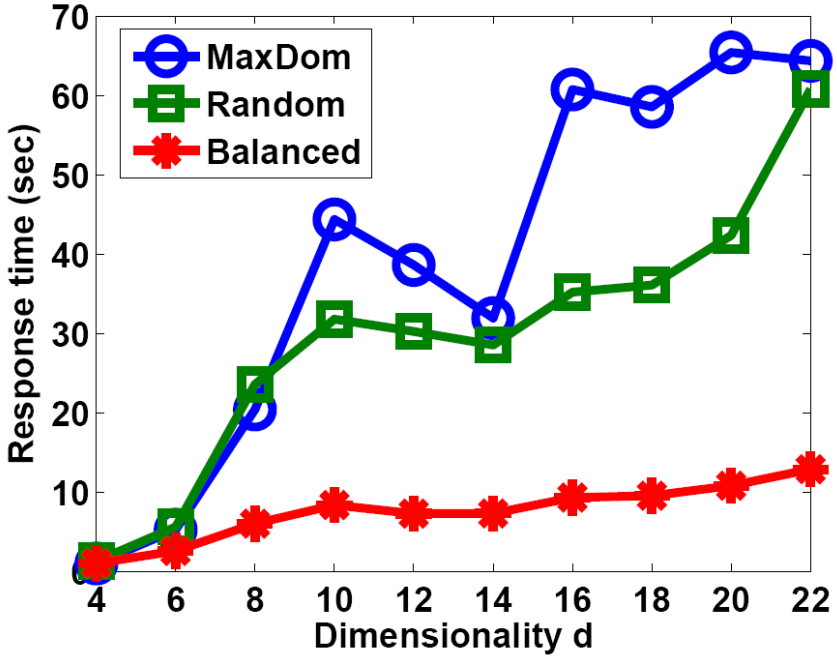
- Compared algorithms
  - Pivot point selection: MaxDom, Random vs. ***Balanced***
  - Scalability: SFS, SaLSa, SSkyline vs. ***BSkyTree-S***, ***BSkyTree-P***

성능 실험 (2 / 4)

- Effect of pivot point selection over *dimensionality*
  - BSkytree-P + arbitrary pivot point selections



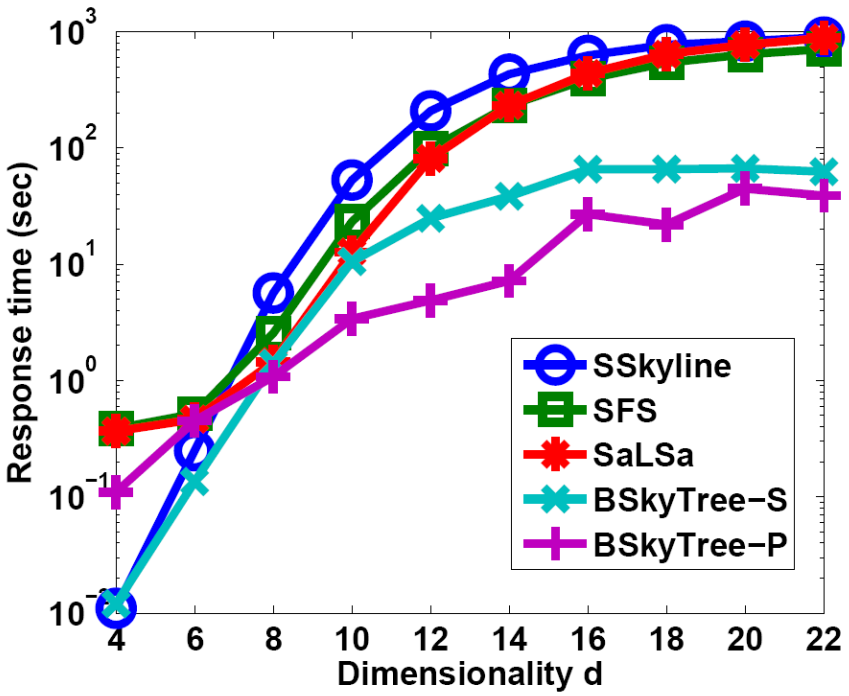
Independent



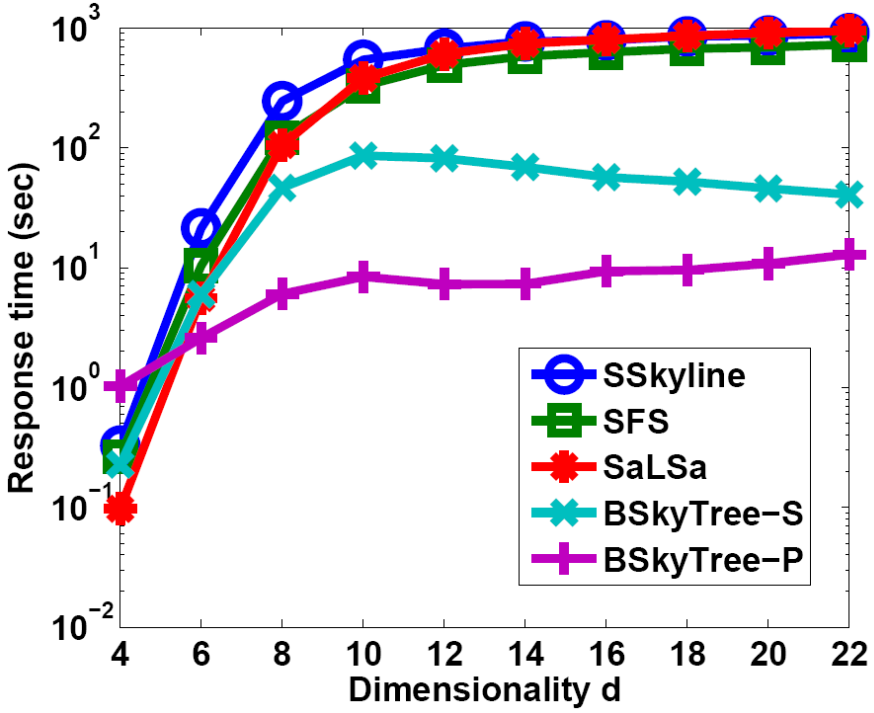
Anti-correlated

성능 실험 (3 / 4)

- Scalability of our proposed algorithm over *dimensionality*
  - Recursive partitioning reduces # of comparisons from incomparability.



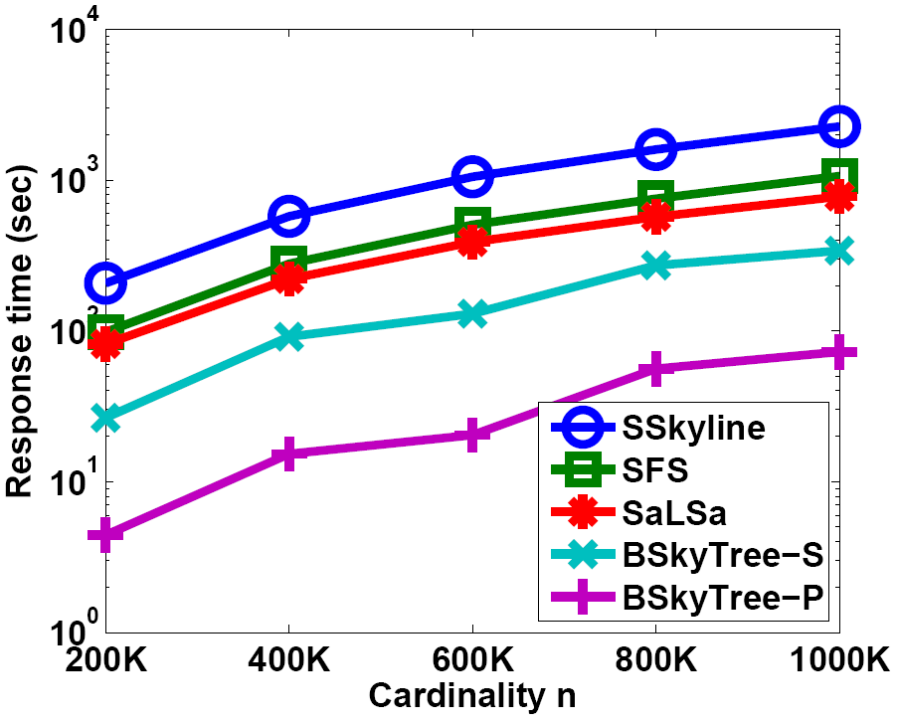
Independent



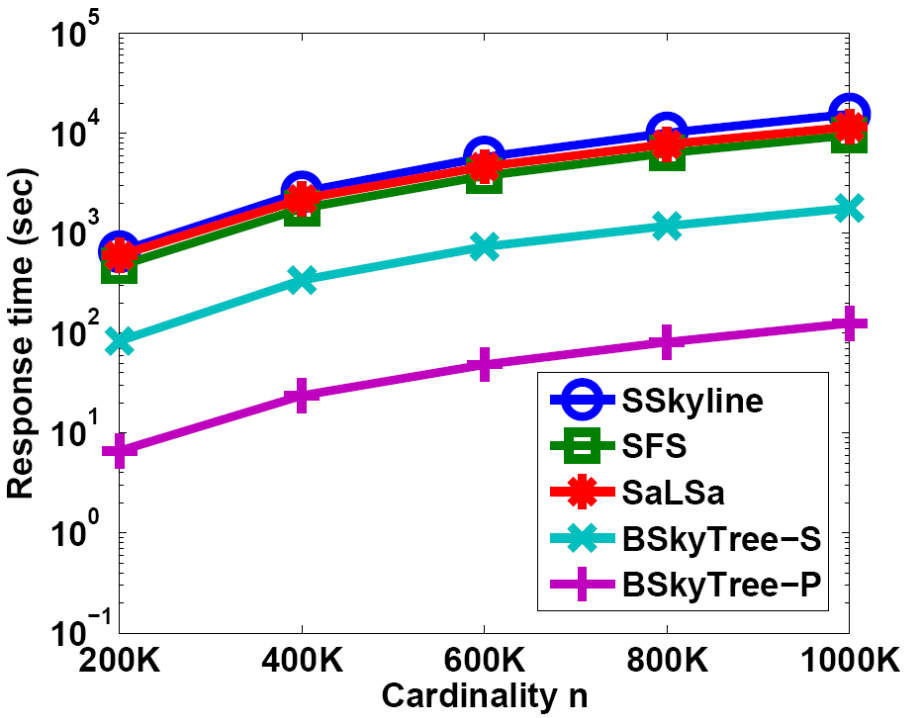
Anti-correlated

성능 실험 (4 / 4)

- Scalability of our proposed algorithm over *n*



Independent



Anti-correlated

줄거리

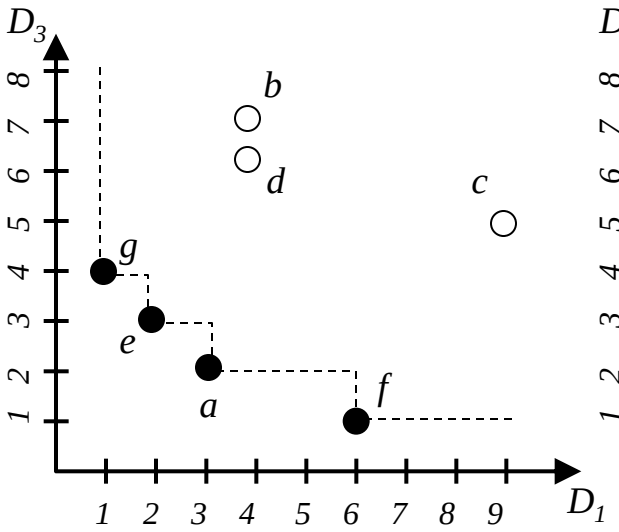
- 스카이라인이 뭐지??
- 점 기반 공간 분할
- 스카이라인 계산 방법
- 성능 실험
- **좀 더 생각하기**
- 결론

부분 공간 스카이라인??

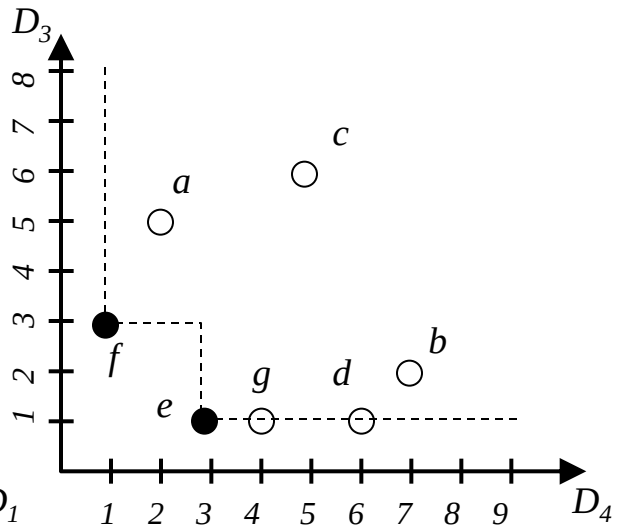
- What if users may issue skyline queries based on arbitrary *subsets* of dimensions?
  - ***subspace skylines*** can be very different!

	D_1	D_2	D_3	D_4
a	3	4	2	5
b	4	6	7	2
c	9	7	5	6
d	4	3	6	1
e	2	2	3	1
f	6	1	1	3
g	1	3	4	1

4-dimensional dataset



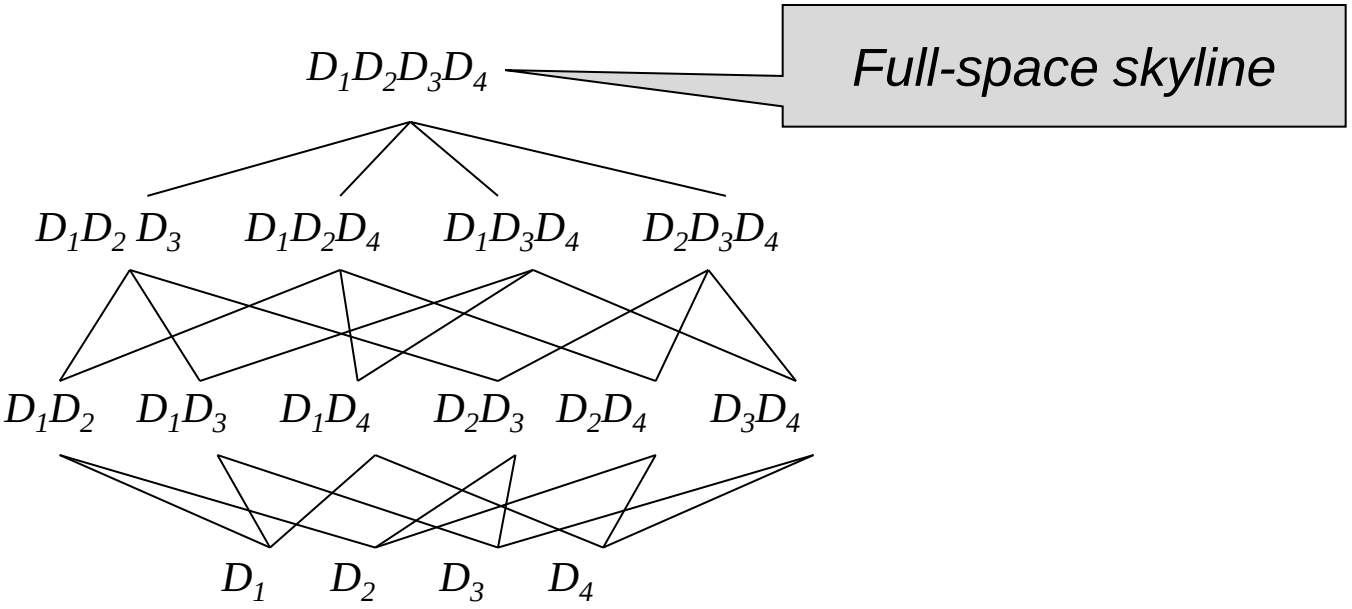
Skyline on $\langle D_1, D_3 \rangle$



Skyline on $\langle D_3, D_4 \rangle$

스카이큐브??

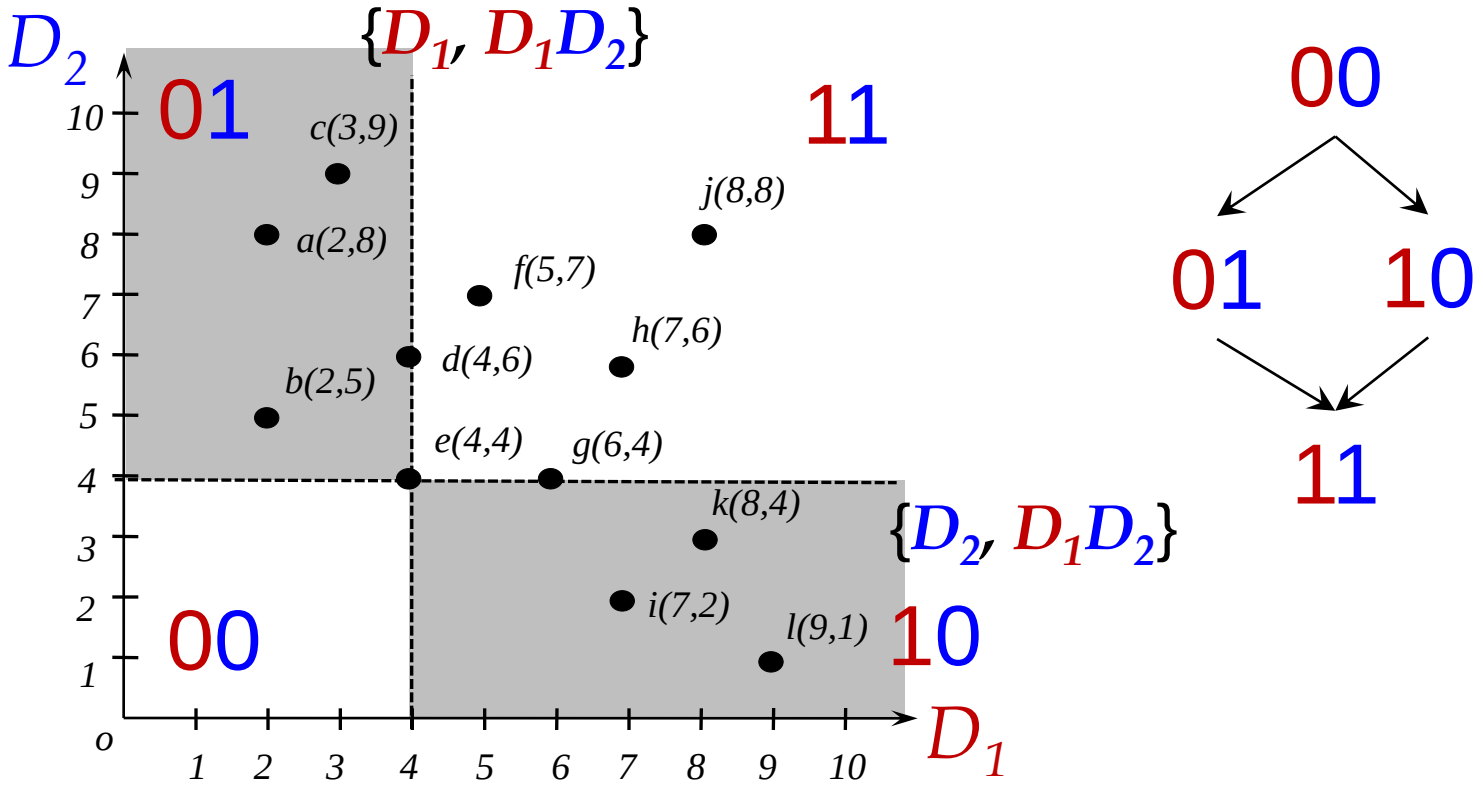
- A d-dimensional space contains $2^d - 1$ **subspaces**.
  - A ***skycube*** is the collection of *all* possible subspace skylines.



  - Multiple subspace skylines may share ***redundant*** computation!!

어떻게 스카이큐브를 계산하지??

- Mapping points into subregions
  - Restricting possible subspaces to be a skyline point



줄거리

- 스카이라인이 뭐지??
- 점 기반 공간 분할
- 스카이라인 계산 방법
- 성능 실험
- 좀 더 생각하기
- **결론**

결론

- **점 기반 공간 분할**을 이용하여 **스카이라인**을 빠르게 계산하는 방법을 찾았습니다.
  - 특히, 종속성과 비교불가능성을 둘 다 고려한 **최적화된 공간 분할 점**을 찾는 방법을 이용하였습니다.

- 점 기반 공간 분할을 활용하여 **스카이큐브**를 빠르게 계산할 수 있는 방법을 찾았습니다.

감사합니다!!

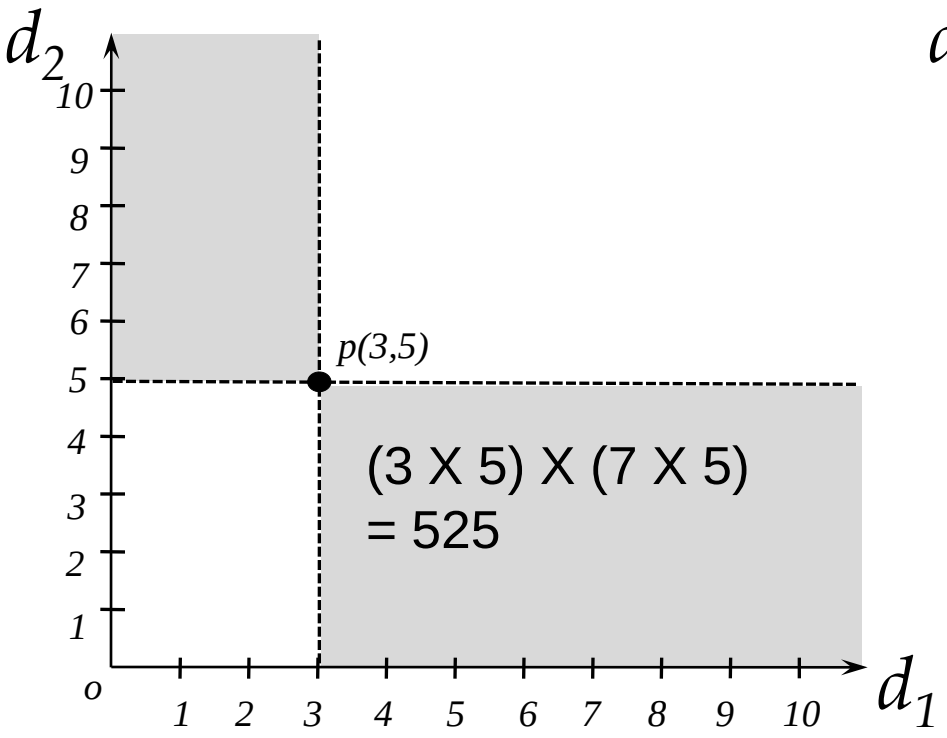
어떻게 공간 분할 점을 선택하지?? (1 / 5)

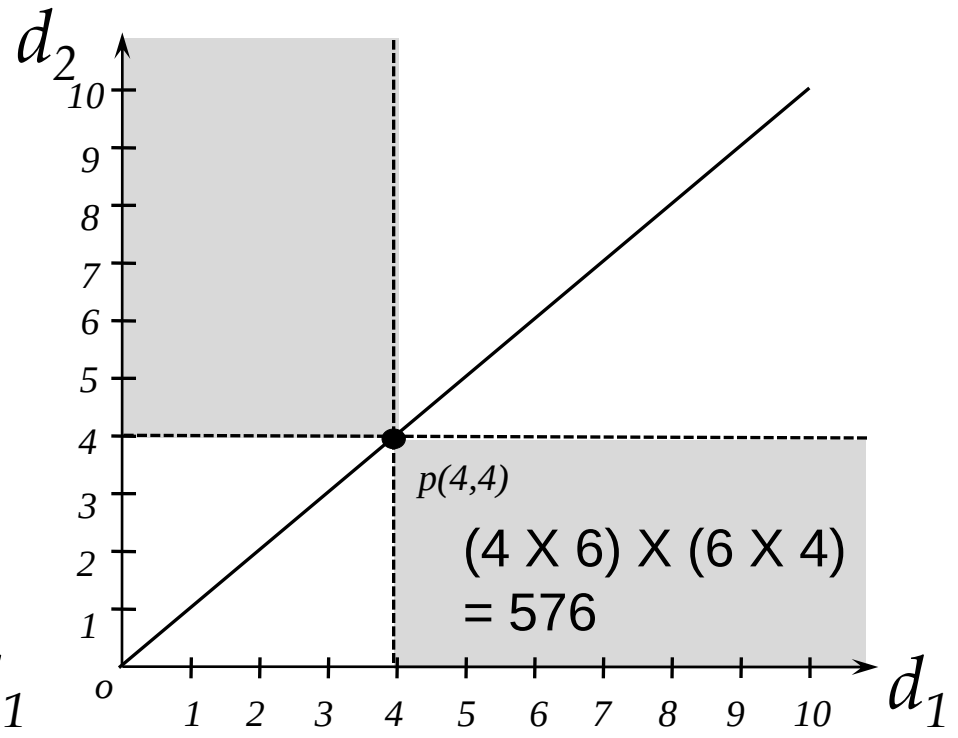
- Exact optimization Incur a prohibitive cost of $O(4^d)$.

- A systematic two-phase approximation approach
  - First phase: consider **skyline points** as the pivot candidates.
  - Second phase: select a skyline point **maximizing incomparability**.
    - Assume that data distribution is uniform and independent.

어떻게 공간 분할 점을 선택하지?? (2 / 5)

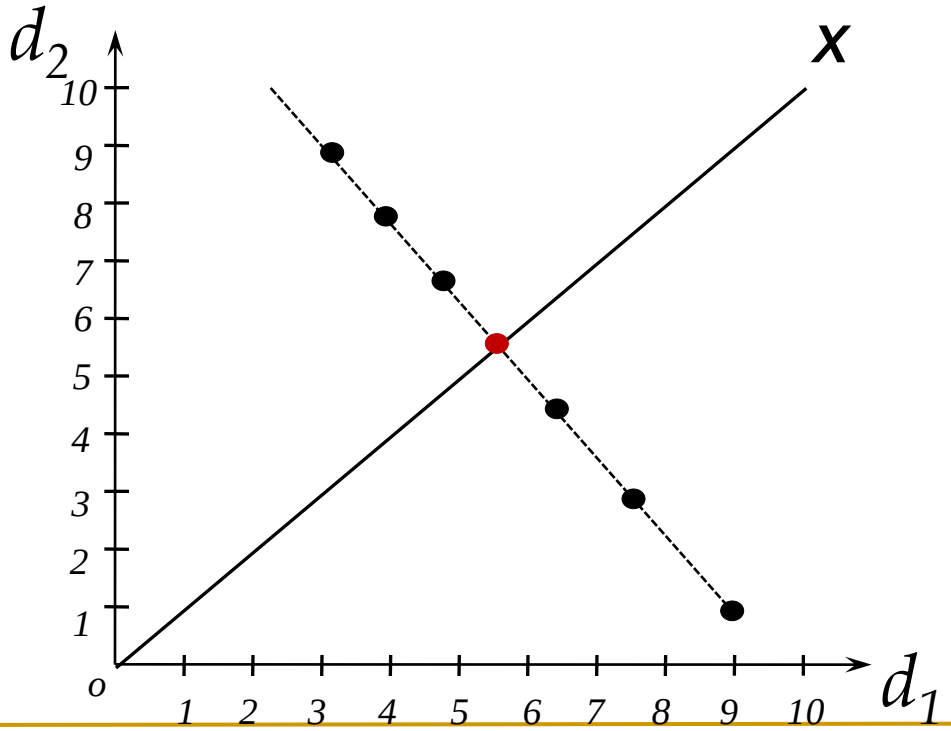
- Use the volume of regions to approximate the cardinality.





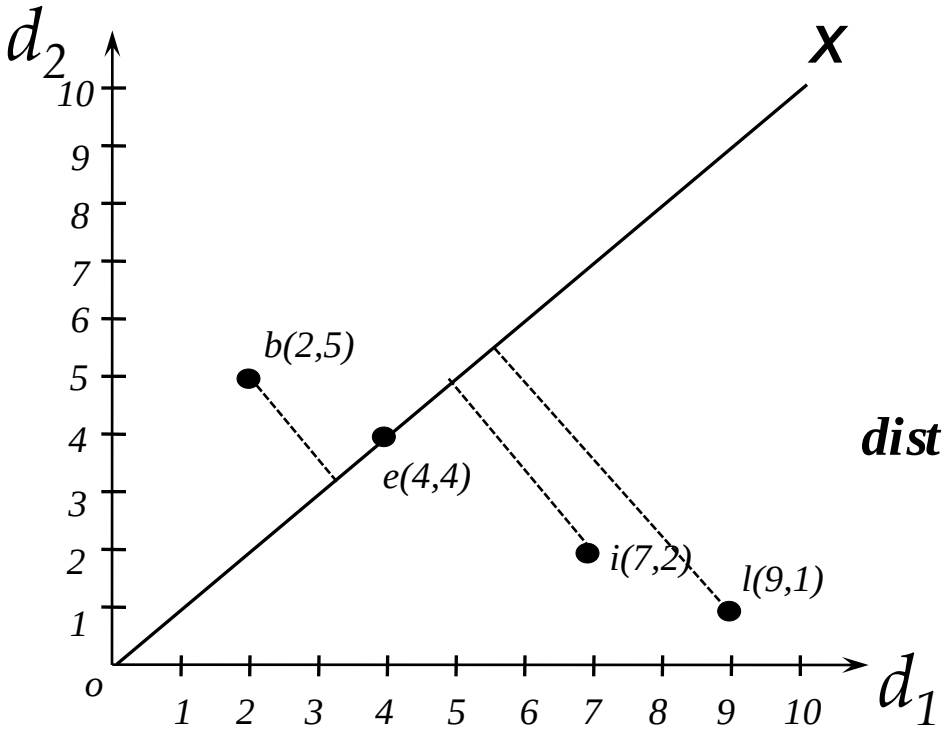
어떻게 공간 분할 점을 선택하지?? (3 / 5)

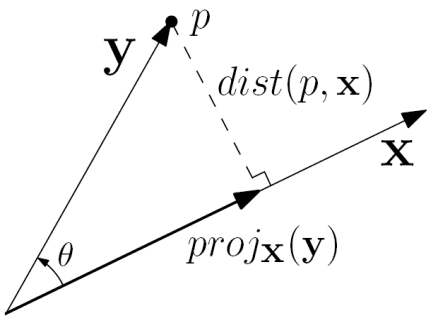
- We prove, for any point *p*, its projection on the diagonal ***X*** has higher incomparability.
  - Find local optimum among points on the dotted line.
- ***dist(p, X)*** quantifies the difference from local optimum.



어떻게 공간 분할 점을 선택하지?? (4 / 5)

- Assuming skyline projections are inherently skewed, ***dist(p, X)*** can approximate incomparability regions.
  - Incomparability over local optima does not vary much.





$$dist(p, X) = \sqrt{\| p \|^2 - \| proj_x(y) \|^2}$$

$$= \sqrt{\frac{(p_1 - p_2)^2 + \ldots + (p_{d-1} - p_d)^2}{d}}$$

→ O(d²).

어떻게 공간 분할 점을 선택하지?? (5 / 5)

- Use a upper bound of ***dist(p, X)*** for p=(p_1, …, p_d)
  - Find a minimizing $p_{max} - p_{min}$ instead of computing *dist(p, X)*.

$$dist(p, X) = \sqrt{\frac{(p_1 - p_2)^2 + \ldots + (p_{d-1} - p_d)^2}{d}} \leq \sqrt{\frac{(p_{max} - p_{min})^2 \times \binom{d}{2}}{d}}$$

→ O(d).

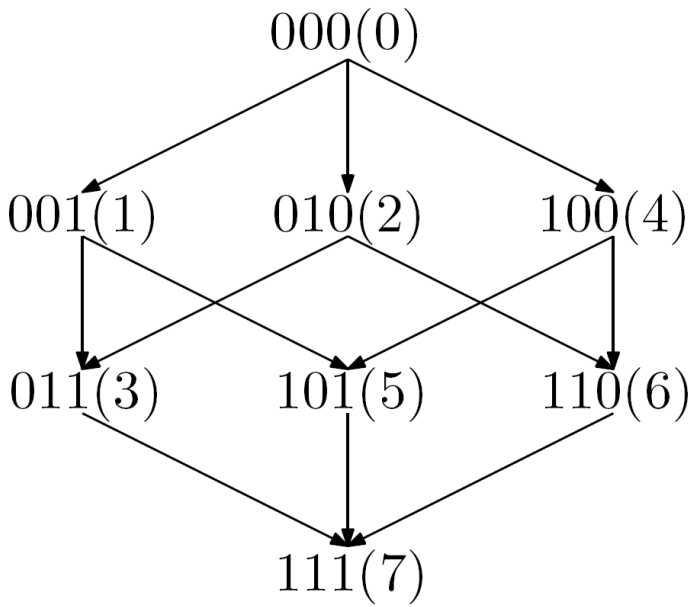
$$dist(p, X) \leq (p_{max} - p_{min}) \times \sqrt{\frac{\binom{d}{2}}{d}}$$

$$p_{\max} = \max_{i \in [1,d]} p_i$$

$$p_{\min} = \min_{i \in [1,d]} p_i$$

고차원으로 확장하기 (1 / 1)

- Example of a lattice when dimensionality is three.
  - Dominance: a top node and a bottom node
  - Partial dominance: self-pair, reachable node pairs
  - Incomparability: non-reachable node pairs



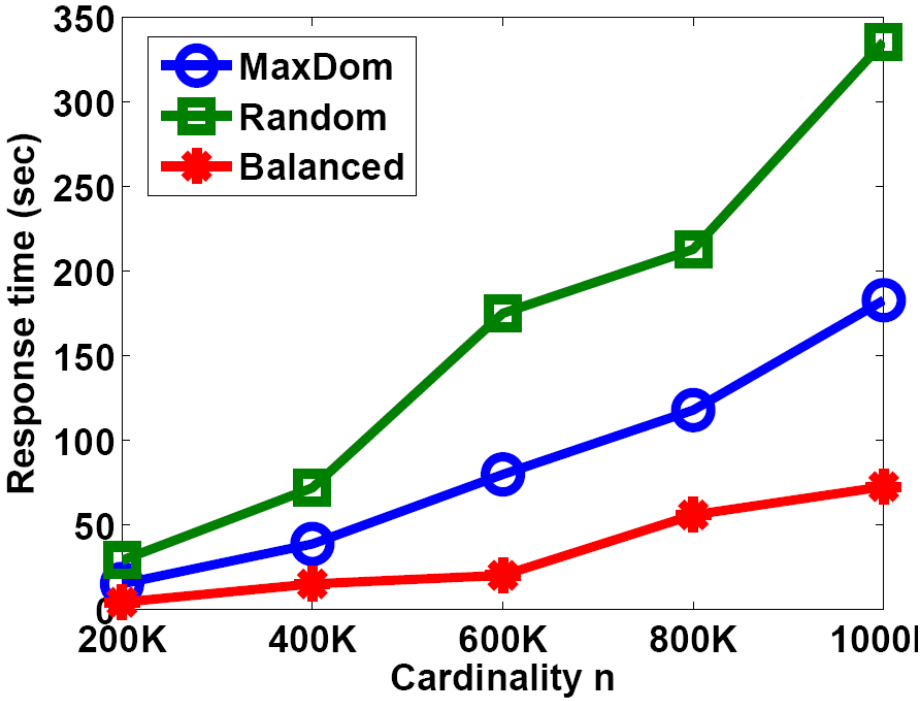
Dominant pairs: (000, 111)

Incomparable pairs:

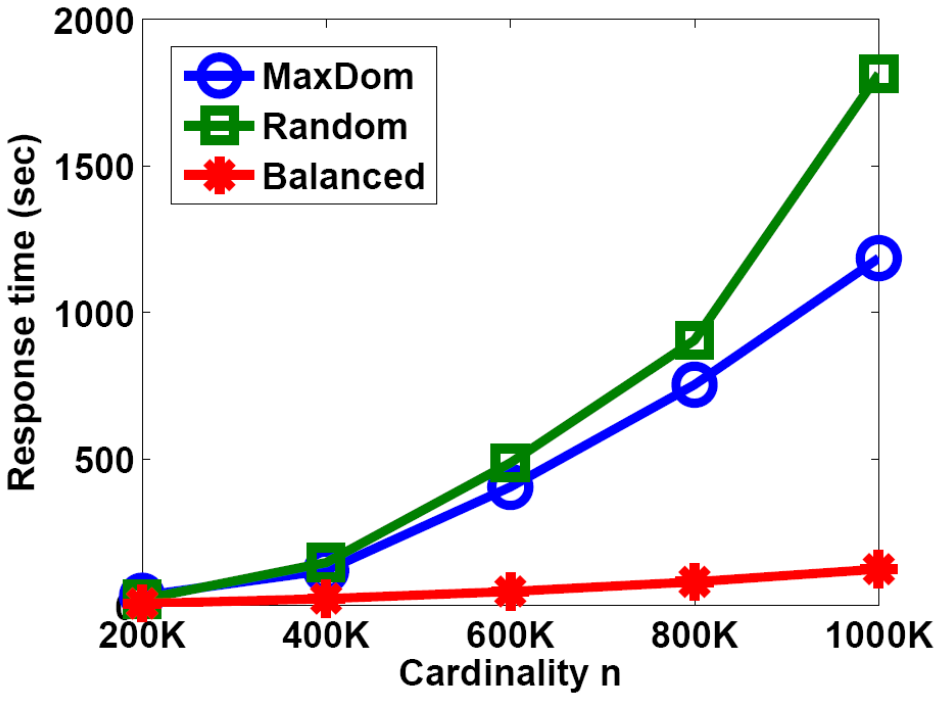
(001, 010), (001, 100), (001, 110)
(010, 100), (010, 101), (100, 011)

추가 성능 실험 (1 / 1)

- Effect of pivot point selection over *cardinality*
  - BSkytree-P + arbitrary pivot point selection



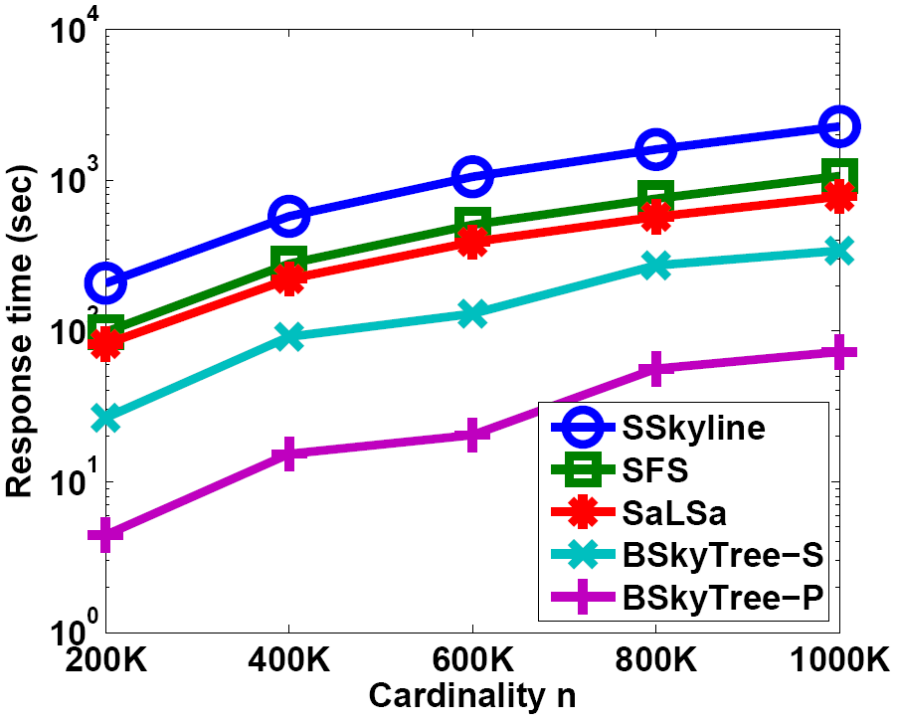
Independent



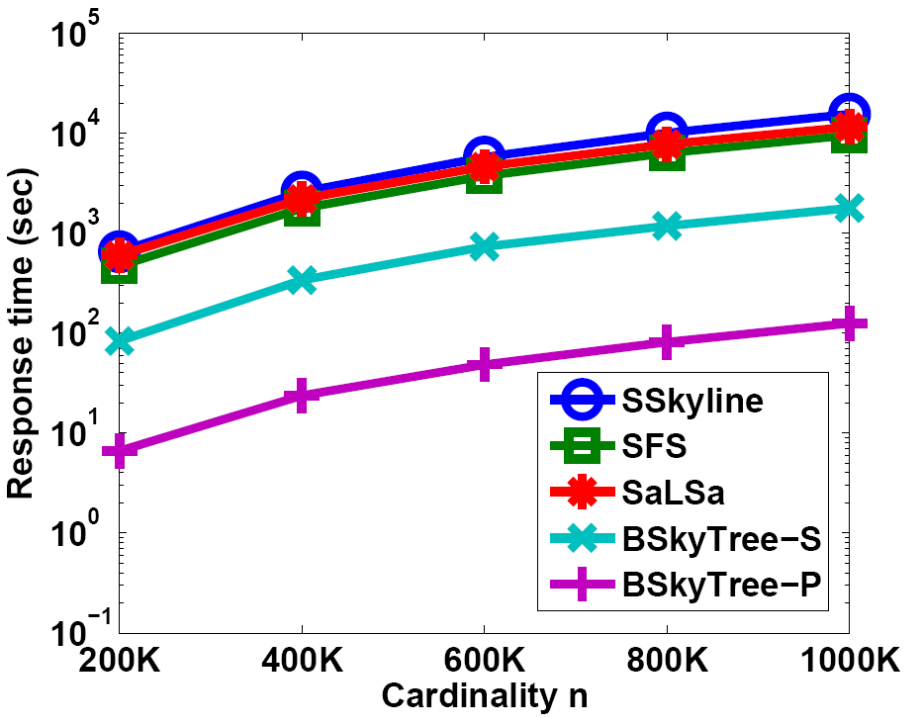
Anti-correlated

추가 성능 실험 (2 / 2)

- Scalability of our proposed algorithm over *n*



Independent



Anti-correlated