Data Mining: Foundation, Techniques and Applications

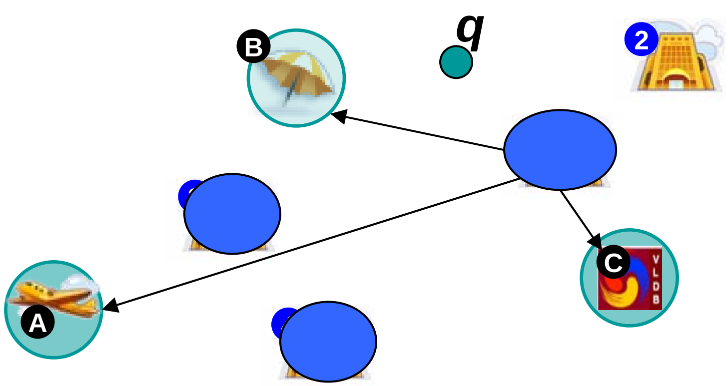
Lesson 9: Skyline/Dominance Relationship Analysis

Anthony Tung(鄧锦浩)
School of Computing
National University of Singapore

Li Cuiping(李翠平)
School of Information
Renmin University of China

Introduction

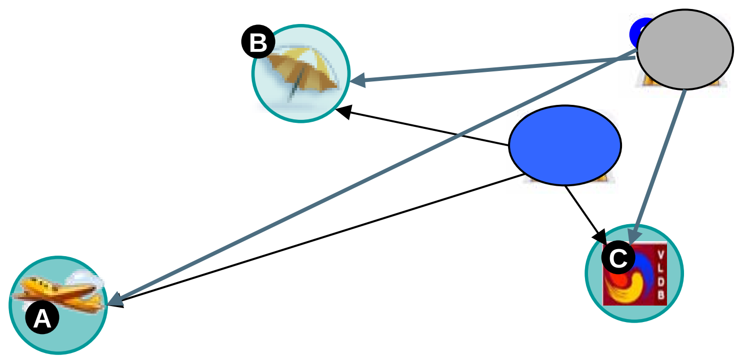
Motivation – Example 1



1. *Find Hotels <u>“close”</u> to Airport (A), Beach (B), and Conference (C) ?*

2. *Which is the nearest hotel (to q)?*

Motivation – Example 1



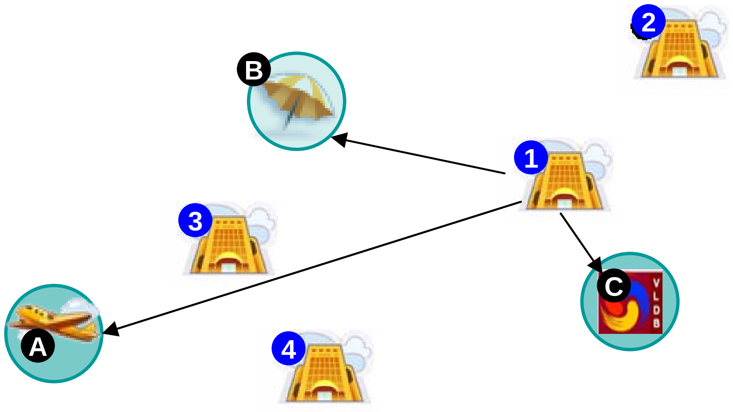
is better choice than in all attributes (A,B,C)

Or

Dominates

Motivation – Example 1

- Skyline – Hotels {1,3,4}



- User A 

  - Weight conference : 0.2
  - Weight airport : 0.4
  - Weight beach : 0.4
- User B 

  - Weight conference : 0.45
  - Weight airport : 0.45
  - Weight beach : 0.1

Motivation – Example 2

Which buildings can we see?

[Higher and closer to river]

Skyline : Definition

All points $p \in D$, which are not dominated by any other point in the dataset

Today’s Talk

Introduction of

Skyline

related research

Table of Contents

Table of Contents

- Basic
  - Dominance Property
    - D-dominance
    - Subspace dominance
    - K-dominance
  - Algorithms
    - Basic Algorithms
    - Advance Algorithms
  - Dominant Relation Analysis
    - Data cube for dominant relation analysis
    - Neighborhood dominant analysis

Dominance

Given two points $p = \{p_1, p_2, \ldots, p_N\}$ and $q = \{q_1, q_2, \ldots, q_N\}$ in N-dimension space,

p is said to dominate another point q on N,

if and only if

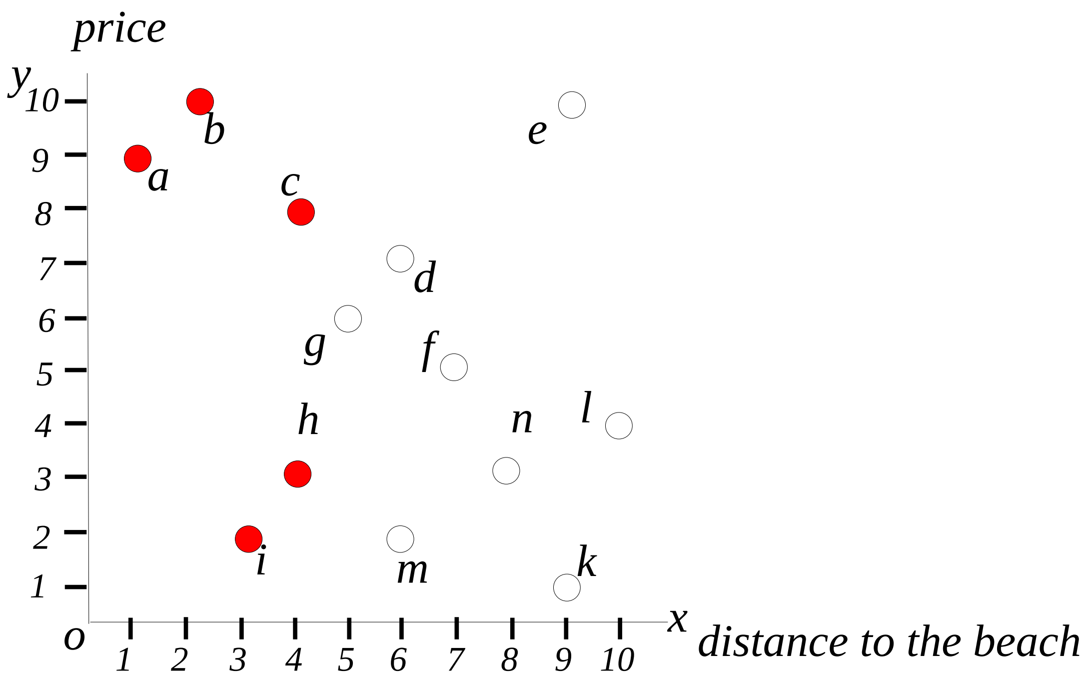
$$\forall\ k \in [1,N],\ p_k \geqslant q_k$$

$$\&$$

$$\exists\ t \in [1,N],\ p_t > q_t$$

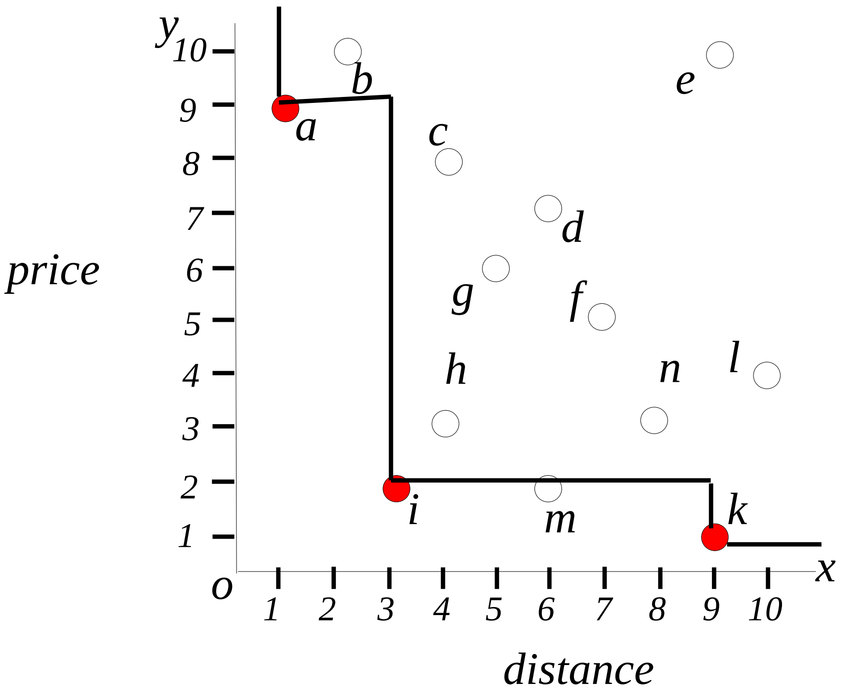
Dominance in 1D

Query : “Give me hotels near the beach”
– Search for min distance



Dominance in 2D

Query : “Give me cheap hotels near the beach”
- Search for min distance and min price



Dominance in N Dimension (N > 2)

- 2004 NBA dataset
- **Who are the best players?**
  - **i.e. not "dominated" by any other player**

Name	Points	Rebounds	Assists	Steals
Tracy McGrady	2003	484	448	135
Kobe Bryant	1819	392	398	86
Shaquille O'Neal	1669	760	200	36
Yao Ming	1465	669	61	34
Dwyane Wade	1854	397	520	121
Steve Nash	1165	249	861	74

Dominance in N Dimension (N > 2)

- 2004 NBA dataset
- **Who are the best players?**
  - **i.e. not “dominated” by any other player**

Name	Points	Rebounds	Assists	Steals
Tracy McGrady	2003	484	448	135
Kobe Bryant	1819	392	398	86
Shaquille O'Neal	1669	760	200	36
Yao Ming	1465	669	61	34
Dwyane Wade	1854	397	520	121
Steve Nash	1165	249	861	74

Dominance in N Dimension (N > 2)

- 2004 NBA dataset
- **Who are the best players?**
  - **i.e. not “dominated” by any other player**

Name	Points	Rebounds	Assists	Steals
Tracy McGrady	2003	484	448	135
Kobe Bryant	1819	392	398	86
Shaquille O'Neal	1669	760	200	36
Yao Ming	1465	669	61	34
Dwyane Wade	1854	397	520	121
Steve Nash	1165	249	861	74

Table of Contents

- Basic
  - Dominance Property
    - D-dominance
    - Subspace dominance
    - K-dominance
  - Algorithms
    - Basic Algorithms
    - Advance Algorithms
  - Dominant Relation Analysis
    - Data cube for dominant relation analysis
    - Neighborhood dominant analysis

Subspace dominance

- Who are the best players? (Only first two attributes)

Name	Points	Rebounds		Assists	Steals
Tracy McGrady	2003	484		448	135
Kobe Bryant	1819	392		398	86
Shaquille O'Neal	1669	760		200	36
Yao Ming	1465	669		61	34
Dwyane Wade	1854	397		520	121
Steve Nash	1165	249		861	74

Subspace dominance

- Who are the best players? (Only first two attributes)

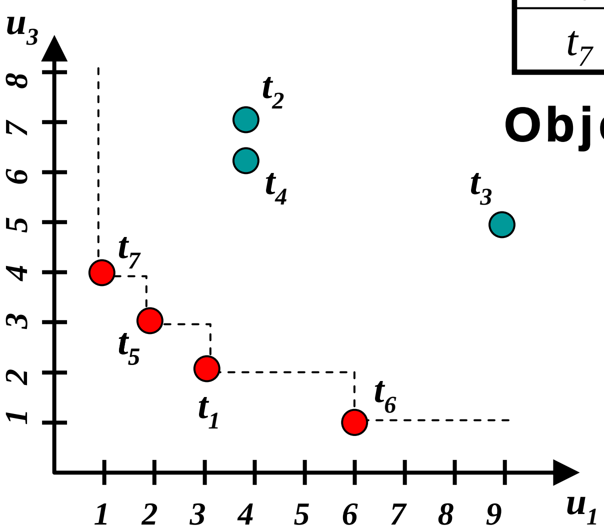
	Name	Points	Rebounds	Assists	Steals
☆	Tracy McGrady	2003	484	448	135
	Kobe Bryant	1819	392	398	86
☆	Shaquille O'Neal	1669	760	200	36
	Yao Ming	1465	669	61	34
	Dwyane Wade	1854	397	520	121
	Steve Nash	1165	249	861	74

Lemma 1: Subspace skyline $\subseteq$ skyline

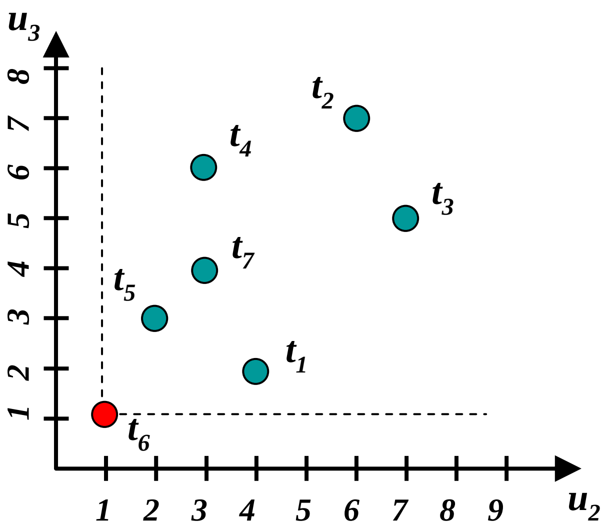
Subspace dominance

	u_1	u_2	u_3	u_4
t_1	3	4	2	5
t_2	4	6	7	2
t_3	9	7	5	6
t_4	4	3	6	1
t_5	2	2	3	1
t_6	6	1	1	3
t_7	1	3	4	1

Objects of *4*-dimensions



Skyline in $\langle u_1, u_3 \rangle$



Skyline in $\langle u_2, u_3 \rangle$

Table of Contents

- Basic
  - Dominance Property
    - D-dominance
    - Subspace dominance
    - K-dominance
  - Algorithms
    - Basic Algorithms
    - Advance Algorithms
  - Dominant Relation Analysis
    - Data cube for dominant relation analysis
    - Neighborhood dominant analysis

k-Dominance

Given two points p = $\{p_1, p_2, \ldots, p_N\}$ and q = $\{q_1, q_2, \ldots, q_N\}$ in N-dimension space.

p is said to k-dominate another point q on N', where N'
$\subseteq$ N, |N'| = k,

if and only if

$$\forall\ s \in N',\ p_s \geqslant q_s$$

&

$$\exists\ t \in N',\ p_t > q_t$$

K-Dominance Example

6-dominant skyline = {p1,p2,p3,p5}

5-dominant skyline = {p1,p2,p3}

	d1	d2	d3	d4	d5	d6
p1	2	2	2	4	4	4
p2	4	4	4	2	2	2
p3	3	3	3	5	3	3
p4	4	4	4	3	3	3
p5	5	5	5	1	5	5

Smaller k, smaller k-dominant skyline

Important Observation 1

- k-dominance can be cyclic
- A 3-dominates B

	d1	d2	d3	d4
A	5	5	5	5
B	1	6	6	6
C	2	1	7	7
D	3	2	1	8

Important Observation 1

- B 3-dominates C

	d1	d2	d3	d4
A	5	5	5	5
B	1	6	6	6
C	2	1	7	7
D	3	2	1	8

Important Observation 1

- C 3-dominates D

	d1	d2	d3	d4
A	5	5	5	5
B	1	6	6	6
C	2	1	7	7
D	3	2	1	8

Important Observation 1

- D 3-dominates A

	d1	d2	d3	d4
A	5	5	5	5
B	1	6	6	6
C	2	1	7	7
D	3	2	1	8

A → B → C → D → A

Table of Contents

- Basic
  - Dominance Property
    - D-dominance
    - Subspace dominance
    - K-dominance
  - Algorithms
    - Basic Algorithms
    - Advance Algorithms
  - Dominant Relation Analysis
    - Data cube for dominant relation analysis
    - Neighborhood dominant analysis

Basic Algorithm 1: Naïve algorithm

- For each tuple *c* in *D* do
  - $\forall$ *d(!= c)* in ***R,***
    - check if d dominates c
    - If yes, c is not skyline

Basic Algorithm 1: Issues

- Time complexity – $O(n^2)$

Basic Algorithm 2 : Simple block nested algorithm

- Let
  - D : Data tuples
  - R : a set of tuple stored in main memory
  - p : an input tuple just read from data D

- Algorithm(p,R):

  1. Remove all x, where x ϵ R and <u>p dominates x</u>
  2. for any x, x ϵ R

     if x dominates p => eliminate p

     else Insert p in R

Basic Algorithm 2 : Issues

- Work well if skyline is less
- Time complexity - $O(n^2)$
- Better IO performance than the naïve algorithm
- Few variant,
  - Move the most dominating tuple in the front part

Basic Algorithm 3 : Divide and Conquer Algorithm

- Let
  - P = Total number of attributes
  - D = input data

- Algorithm(D)

1. Find median(m_k) of any attribute(k)
2. Divide D into two partitions, D1 and D2

   For D1, $\forall$d, d $\in$ D1, d[k] <= m_k

   For D2, $\forall$d, d $\in$ D2, d[k] > m_k
3. Compute skyline s1 for D1 and s2 for D2
4. Merge S1 and S2

Basic Algorithm 3 : Issues

- Best worst case complexity – O(n logn)
- Extension to the basic algorithm
  - M-way partitioning
  - Early Skyline

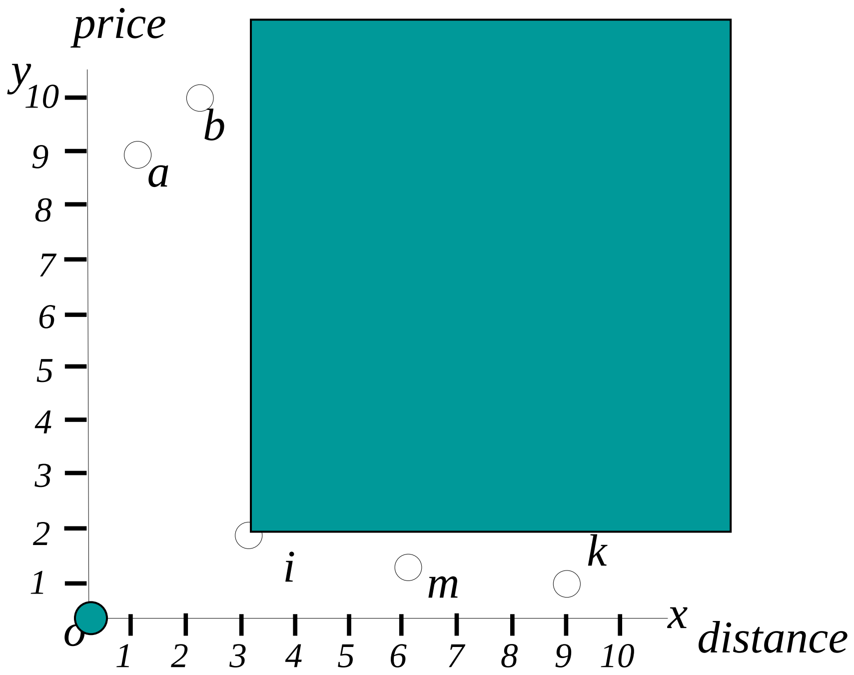
Table of Contents

- Basic
  - Dominance Property
    - D-dominance
    - Subspace dominance
    - K-dominance
  - Algorithms
    - Basic Algorithms
    - Advance Algorithms
  - Dominant Relation Analysis
    - Data cube for dominant relation analysis
    - Neighborhood dominant analysis

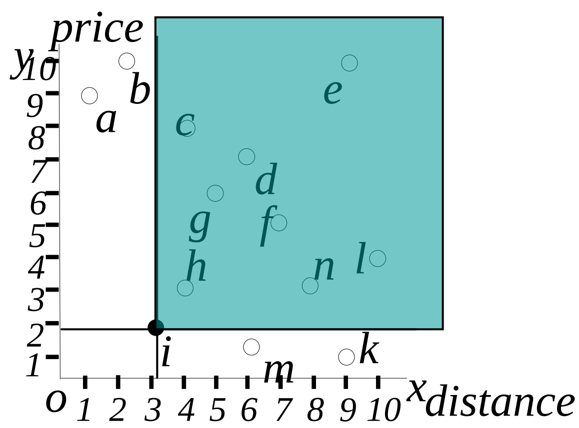
Advanced Algorithm 1: Branch & Bound Skyline

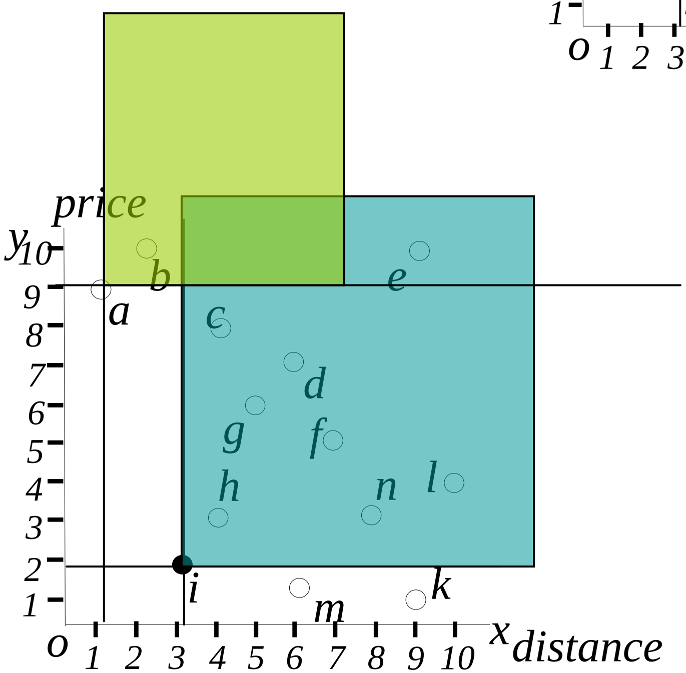
Basics

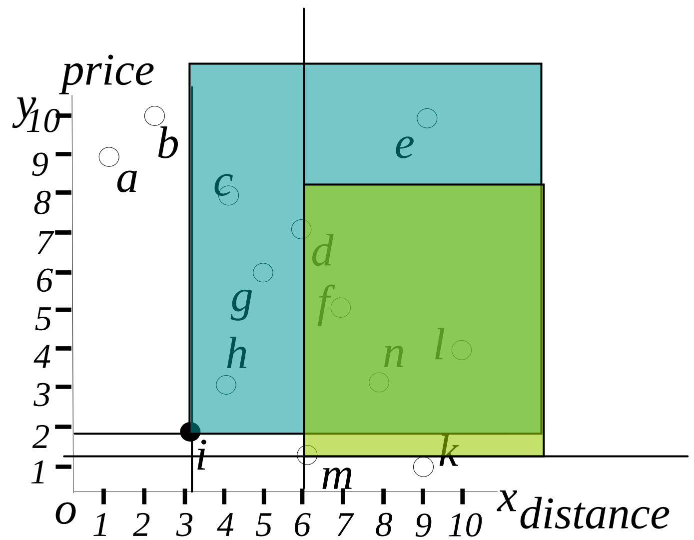
- **Nearest Neighbour**



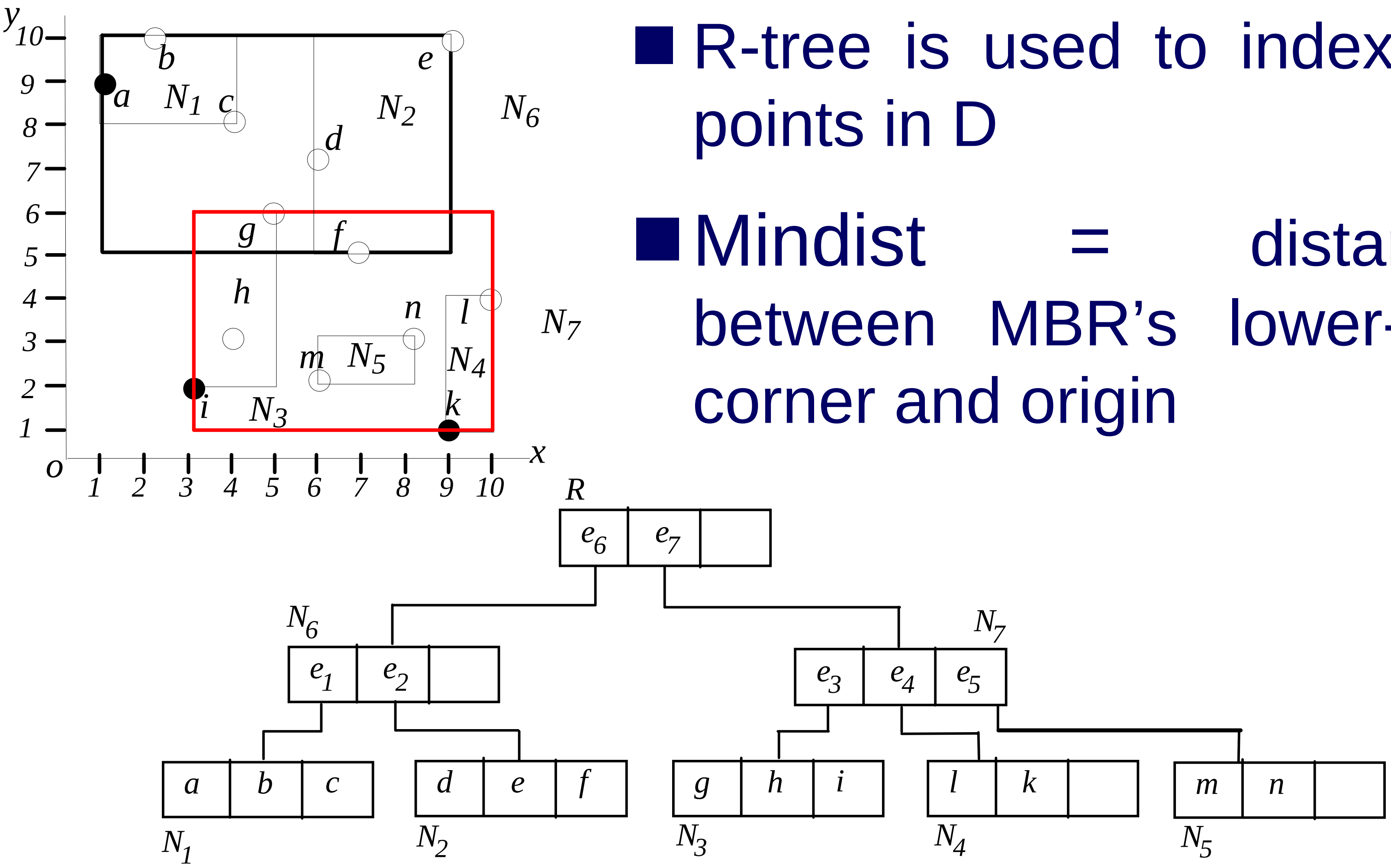
Advanced Algorithm 1: Branch & Bound Skyline





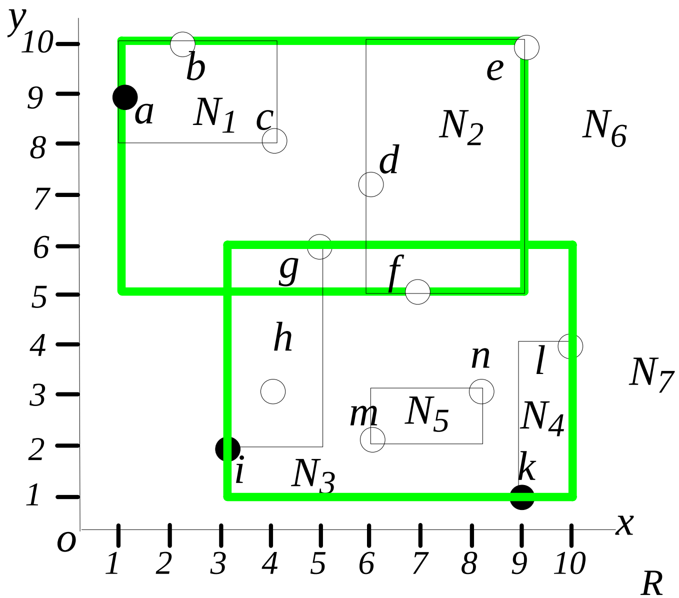


Advanced Algorithm 2: Branch & Bound Skyline



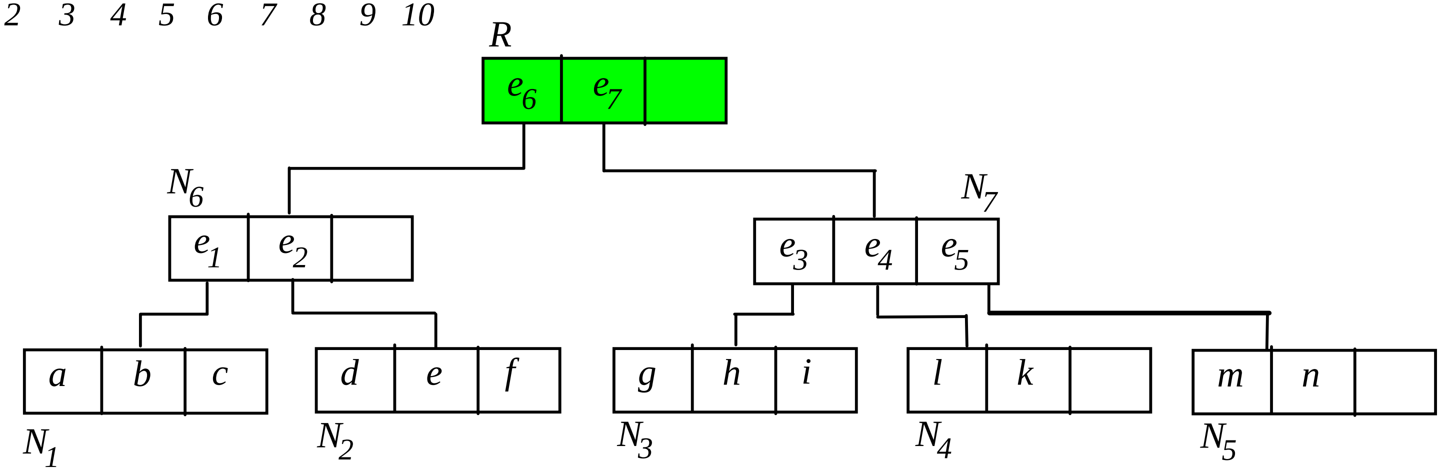
- R-tree is used to index all points in D
- Mindist = distance between MBR's lower-left corner and origin

Advanced Algorithm 2: Branch & Bound Skyline

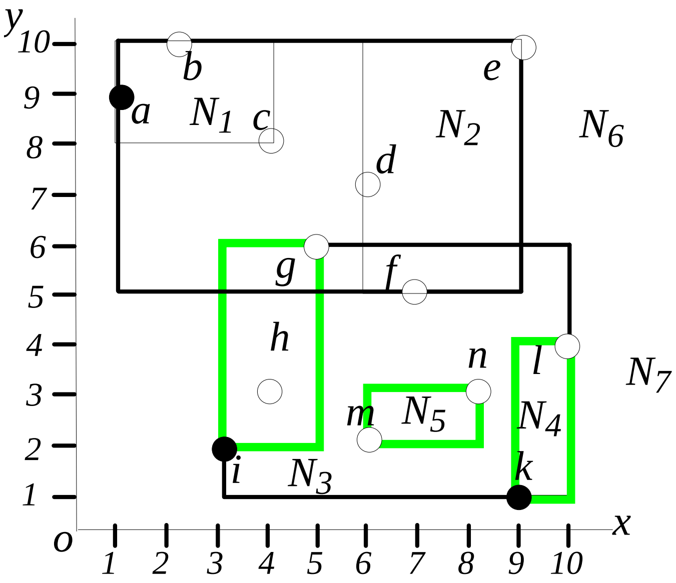


- Each heap entry keeps the mindist of the MBR.

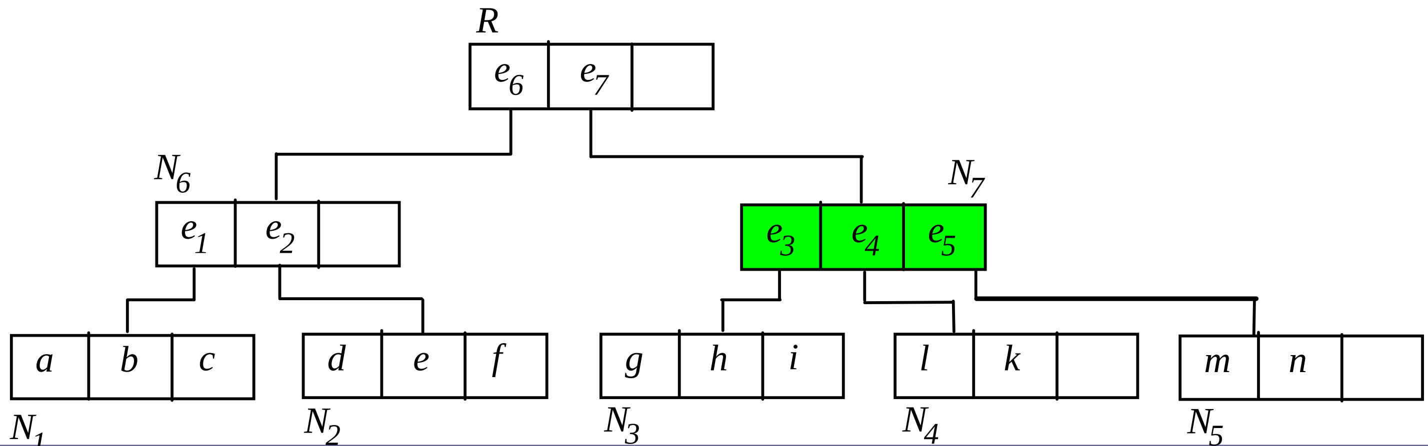
action	heap contents	S
access root	<e_7,4><e_6,6>	$\varnothing$



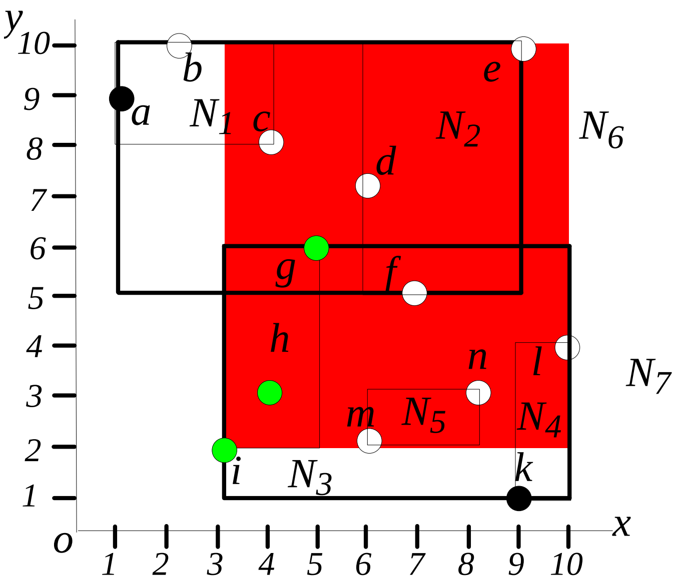
Advanced Algorithm 2: Branch & Bound Skyline



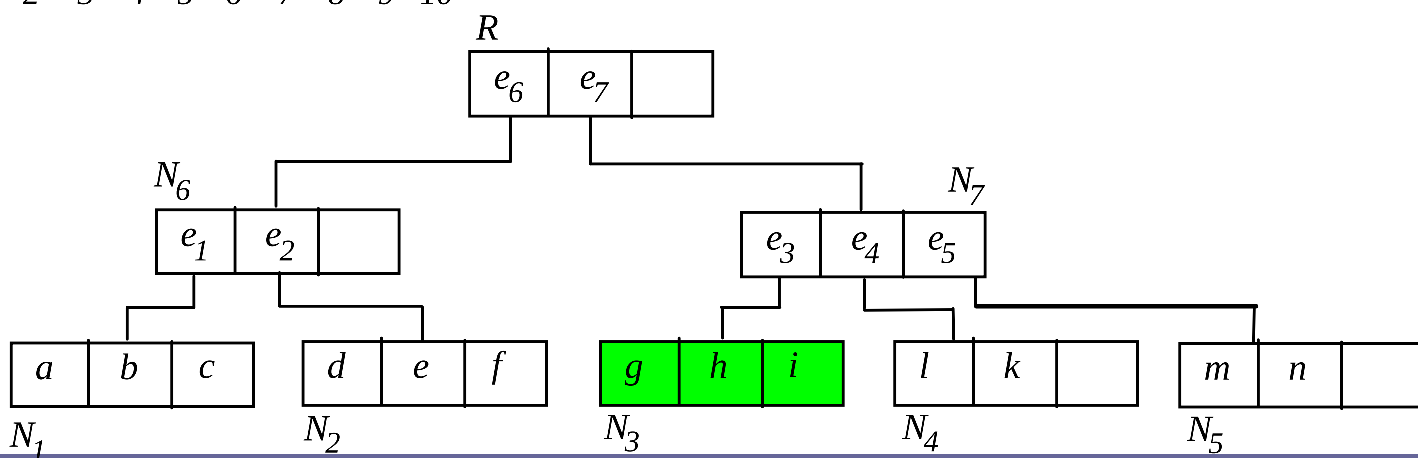
action	heap contents	*S*
access root	$<e_7,4><e_6,6>$	$\varnothing$
expand e_7	$<e_3,5><e_6,6><e_5,8><e_4,10>$	$\varnothing$



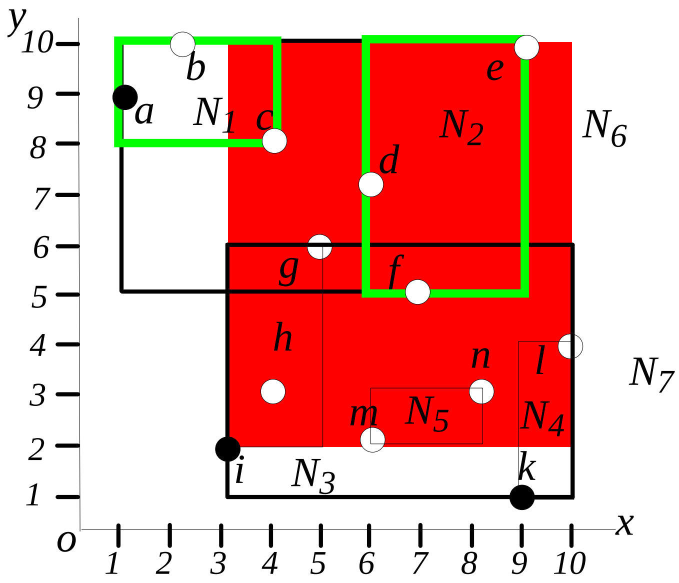
Advanced Algorithm 2 : Branch & Bound Skyline



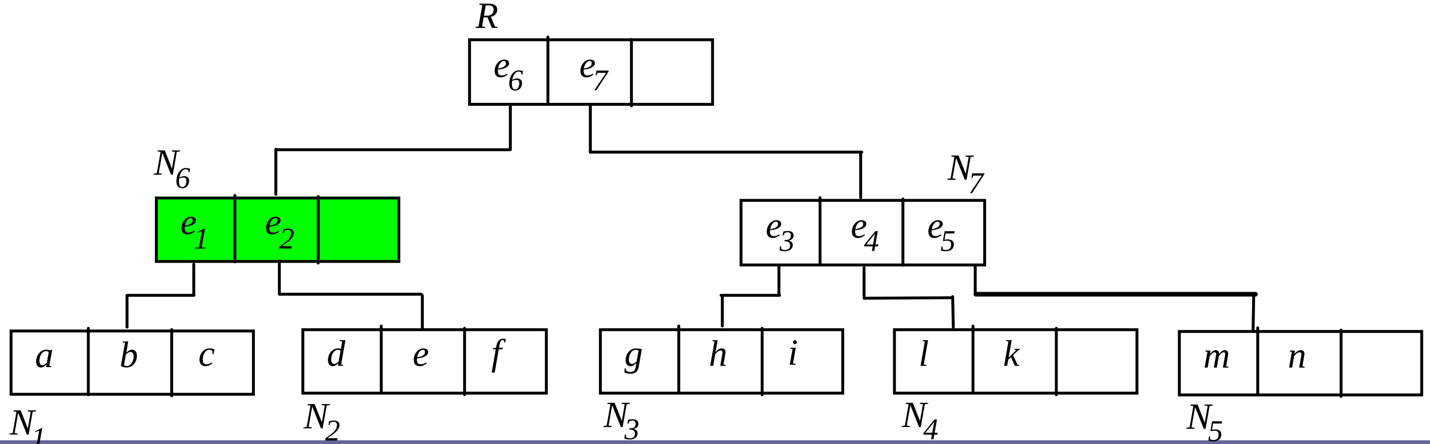
action	heap contents	S
access root	<e_7,4><e_6,6>	∅
expand e_7	<e_3,5><e_6,6><e_5,8><e_4,10>	∅
expand e_3	**<*i*,5>**<e_6,6><e_5,8><e_4,10>	{*i*}



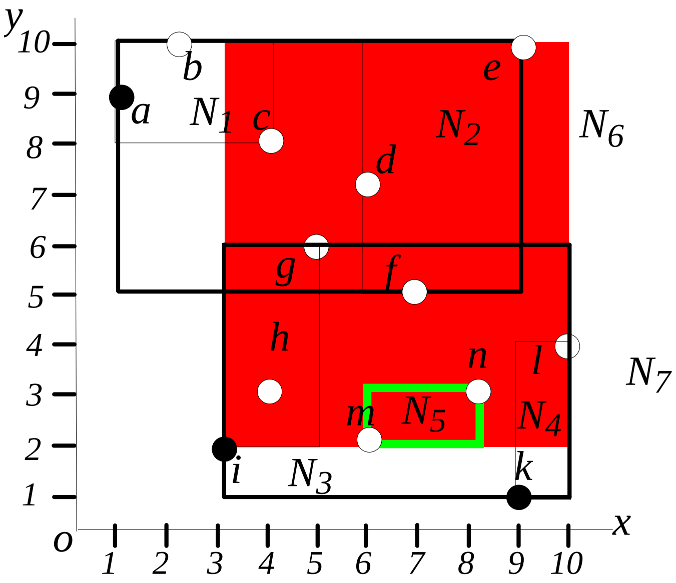
Advanced Algorithm 2 : Branch & Bound Skyline



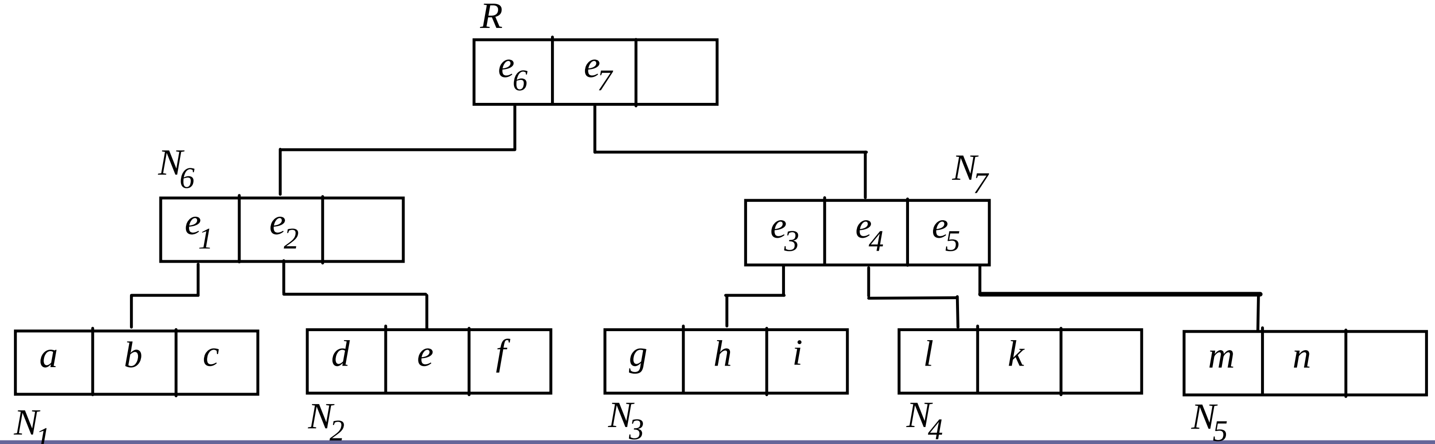
action	heap contents	S
access root	<e_7,4><e_6,6>	∅
expand e_7	<e_3,5><e_6,6><e_5,8><e_4,10>	∅
expand e_3	**<i,5>**<e_6,6><e_5,8> <e_4,10>	{i}
expand e_6	<e_5,8><e_1,9><e_4,10>	{i}



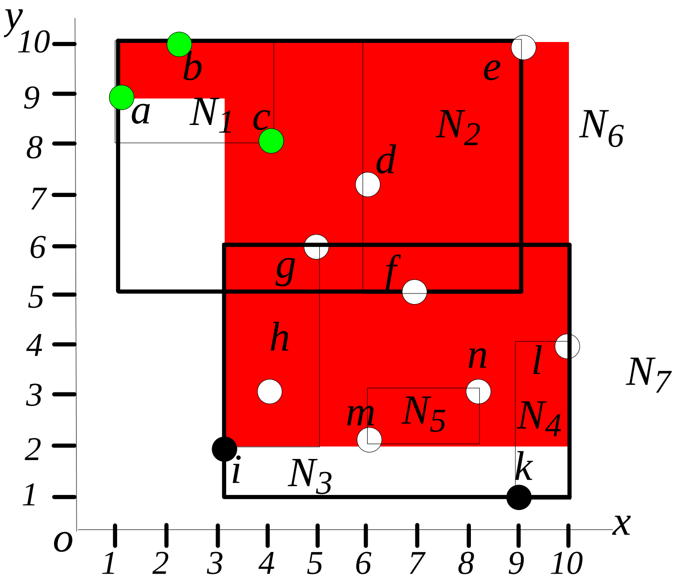
Advanced Algorithm 2 : Branch & Bound Skyline



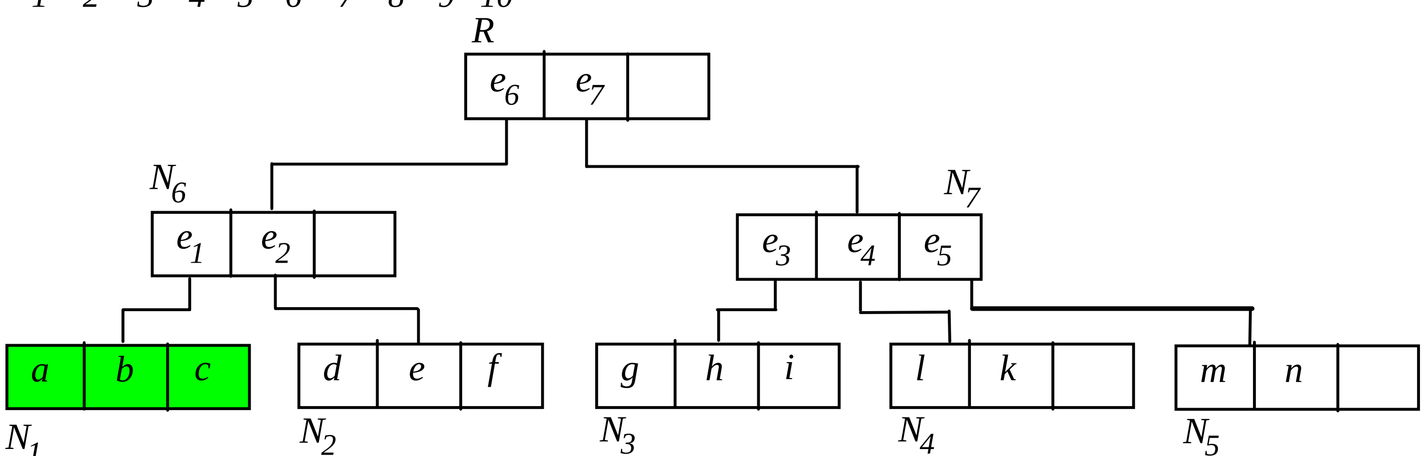
action	heap contents	S
access root	$<e_7,4><e_6,6>$	$\varnothing$
expand e_7	$<e_3,5><e_6,6><e_5,8><e_4,10>$	$\varnothing$
expand e_3	$\boldsymbol{<i,5>}<e_6,6><e_5,8><e_4,10>$	$\{i\}$
expand e_6	$<e_5,8><e_1,9><e_4,10>$	$\{i\}$
remove e_5	$<e_1,9><e_4,10>$	$\{i\}$



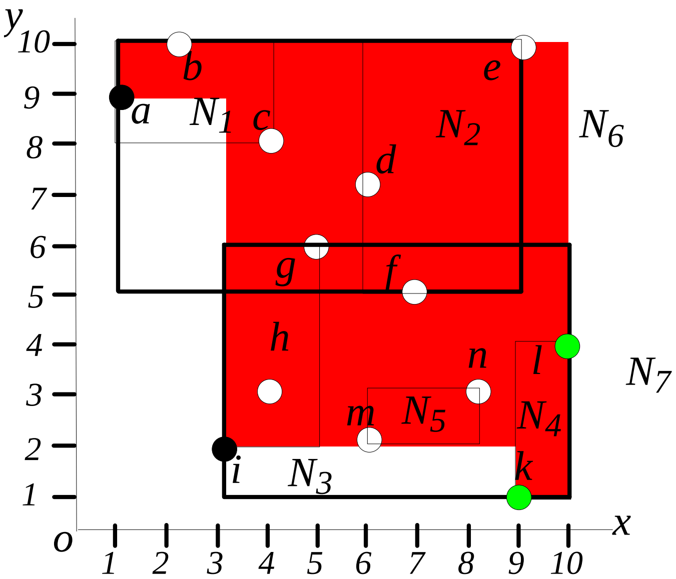
Advanced Algorithm 2 : Branch & Bound Skyline



action	heap contents	*S*
access root	$<e_7,4><e_6,6>$	$\varnothing$
expand e_7	$<e_3,5><e_6,6><e_5,8><e_4,10>$	$\varnothing$
expand e_3	**$<i,5>$**$<e_6,6><e_5,8><e_4,10>$	$\{i\}$
expand e_6	$<e_5,8><e_1,9><e_4,10>$	$\{i\}$
remove e_5	$<e_1,9><e_4,10>$	$\{i\}$
expand e_1	**$<a,10>$**$<e_4,10>$	$\{i,a\}$



Advanced Algorithm 2 : Branch & Bound Skyline



action	heap contents	S
access root	$\langle e_7,4\rangle\langle e_6,6\rangle$	$\varnothing$
expand e_7	$\langle e_3,5\rangle\langle e_6,6\rangle\langle e_5,8\rangle\langle e_4,10\rangle$	$\varnothing$
expand e_3	$\boldsymbol{\langle i,5\rangle}\langle e_6,6\rangle\langle e_5,8\rangle\langle e_4,10\rangle$	$\{i\}$
expand e_6	$\langle e_5,8\rangle\langle e_1,9\rangle\langle e_4,10\rangle$	$\{i\}$
remove e_5	$\langle e_1,9\rangle\langle e_4,10\rangle$	$\{i\}$
expand e_1	$\boldsymbol{\langle a,10\rangle}\langle e_4,10\rangle$	$\{i,a\}$
expand e_4	$\boldsymbol{\langle k,10\rangle}$	$\{i,a,k\}$

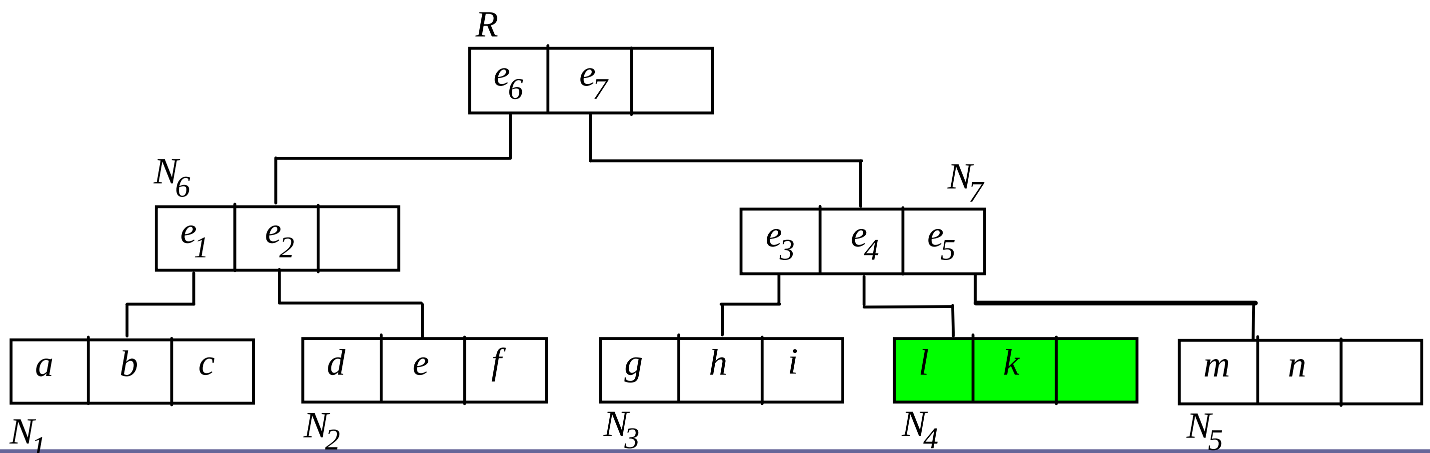
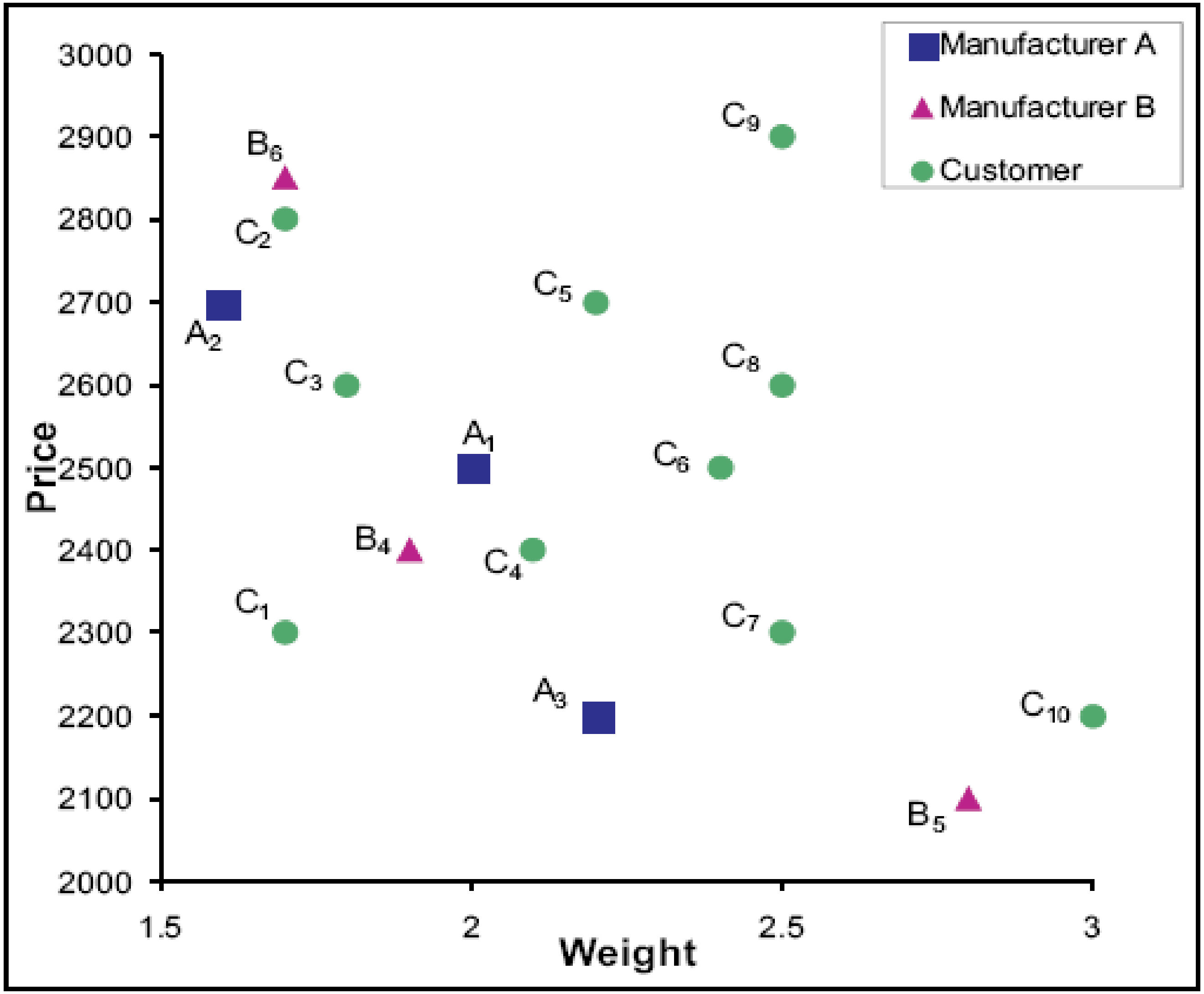


Table of Contents

- Basic
  - Dominance Property
    - D-dominance
    - Subspace dominance
    - K-dominance
  - Algorithms
    - Basic Algorithms
    - Advance Algorithms
  - Dominant Relation Analysis
    - Data cube for dominant relation analysis
    - Neighborhood dominant analysis

Dominant Relationship Analysis : Motivation

Where should I position my laptops ?



Dominant Relationship Analysis : Motivation

Manufacturers： Want to know whether their products are popular with customers compared to their competitors' products

Dominant Relationship Analysis : Motivation

Three Queries manufacturer needs to answer :-

- **Finite Resources & Trade off between price and weight**

  LOQ – Linear Optimization Query

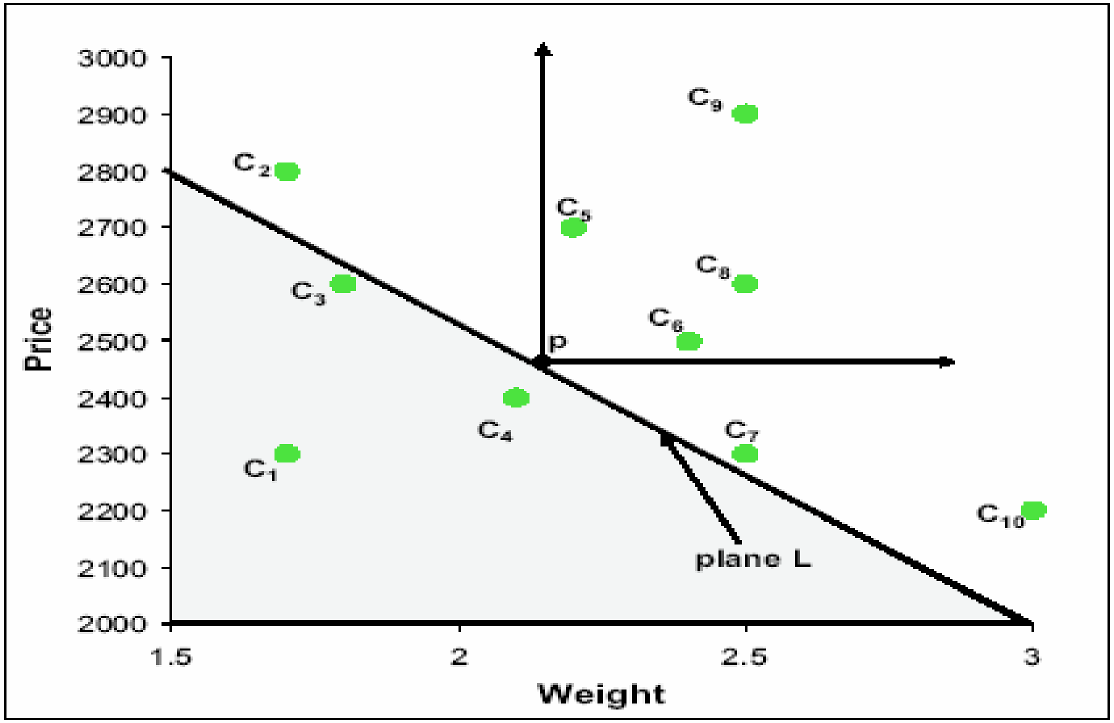
- **Certain set of attributes are important to many customer**

  SAQ – Subspace Analysis Query

- **Identify customers who are dominated only by our products & Identify for a customers who are dominated only by our products as well as competitor**

  CDQ - Comparative Dominant Query

LOQ

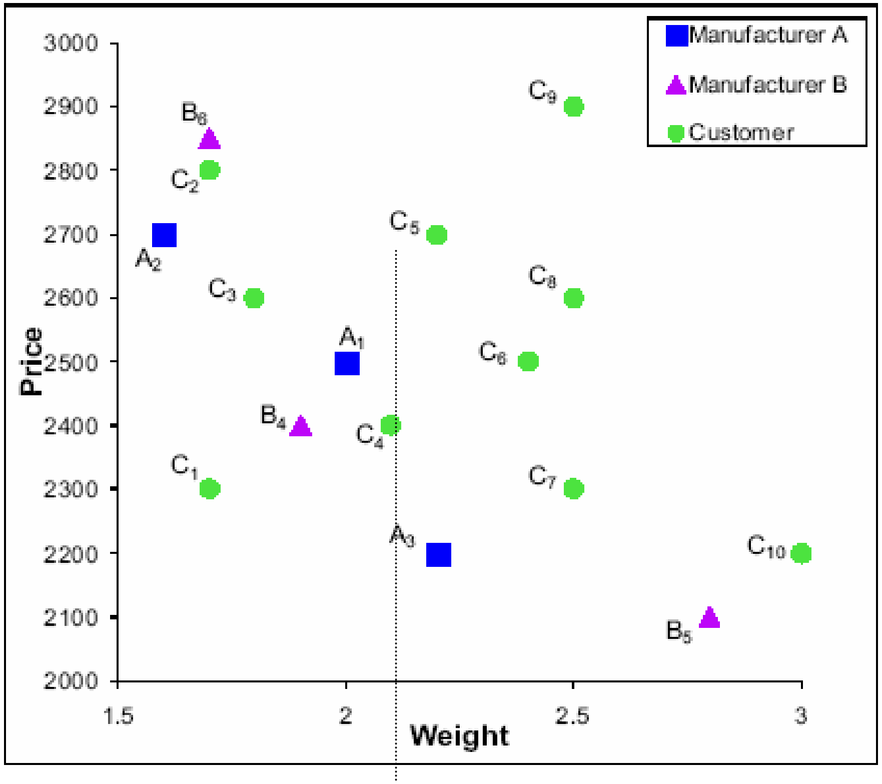


- Definition: find a point on L who dominates the most points

SAQ

Analyze the dominant relationship in the subspace of D

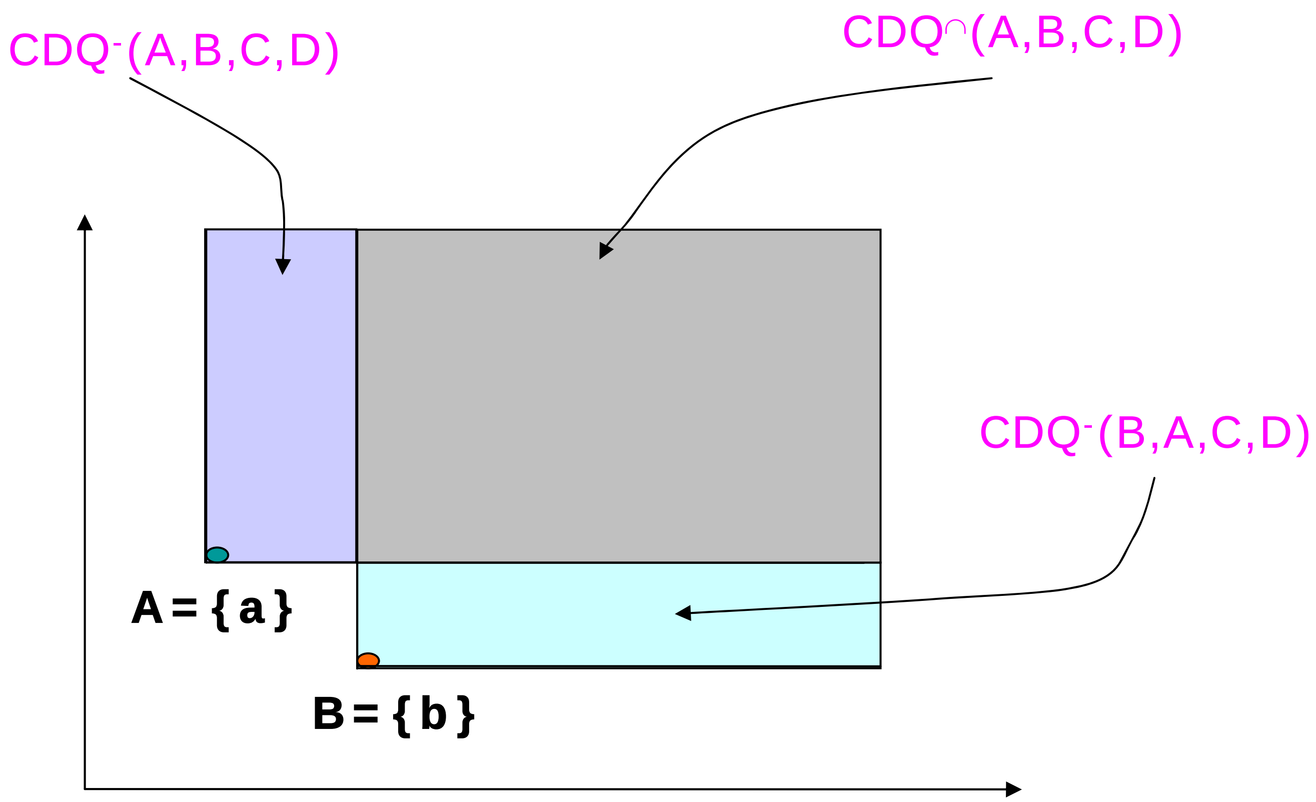
- For example, let D'= {weight}:
  - dominating(A_1, C, D') = { C_5, C_6, C_8, C_9, , C_4, C_7 , C_{10}}
  - dominated(C, A_1, D) = { C_1, C_2, C_3}



CDQ

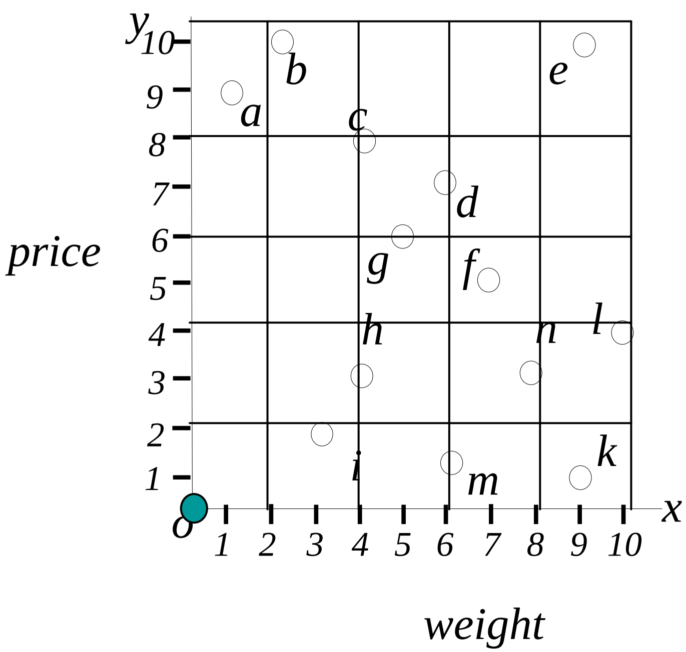
- Comparative Dominant Query(CDQ)
  - compare the set of dominated objects between competitive products
    - Group Dominant: given two sets A, C in an N-dimensional space D, gdominating(A, C, D) is the set of objects in C which are dominated by some object from A
    - $CDQ^{-}(A,B,C,D)=|gdominating(A,C,D)-gdominating(B,C,D)|$
    - $CDQ^{\cap}(A,B,C,D)=|gdominating(A,C,D)\cap gdominating(B,C,D)|$

CDQ

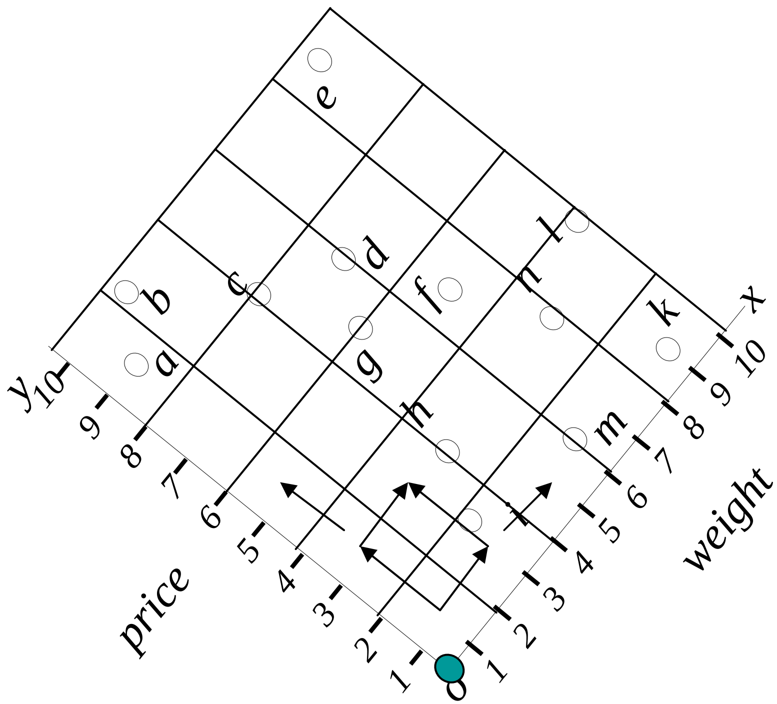


Data Cube for Dominant Relationship Analysis : DADA

Step 1: How to prepare Lattice ?

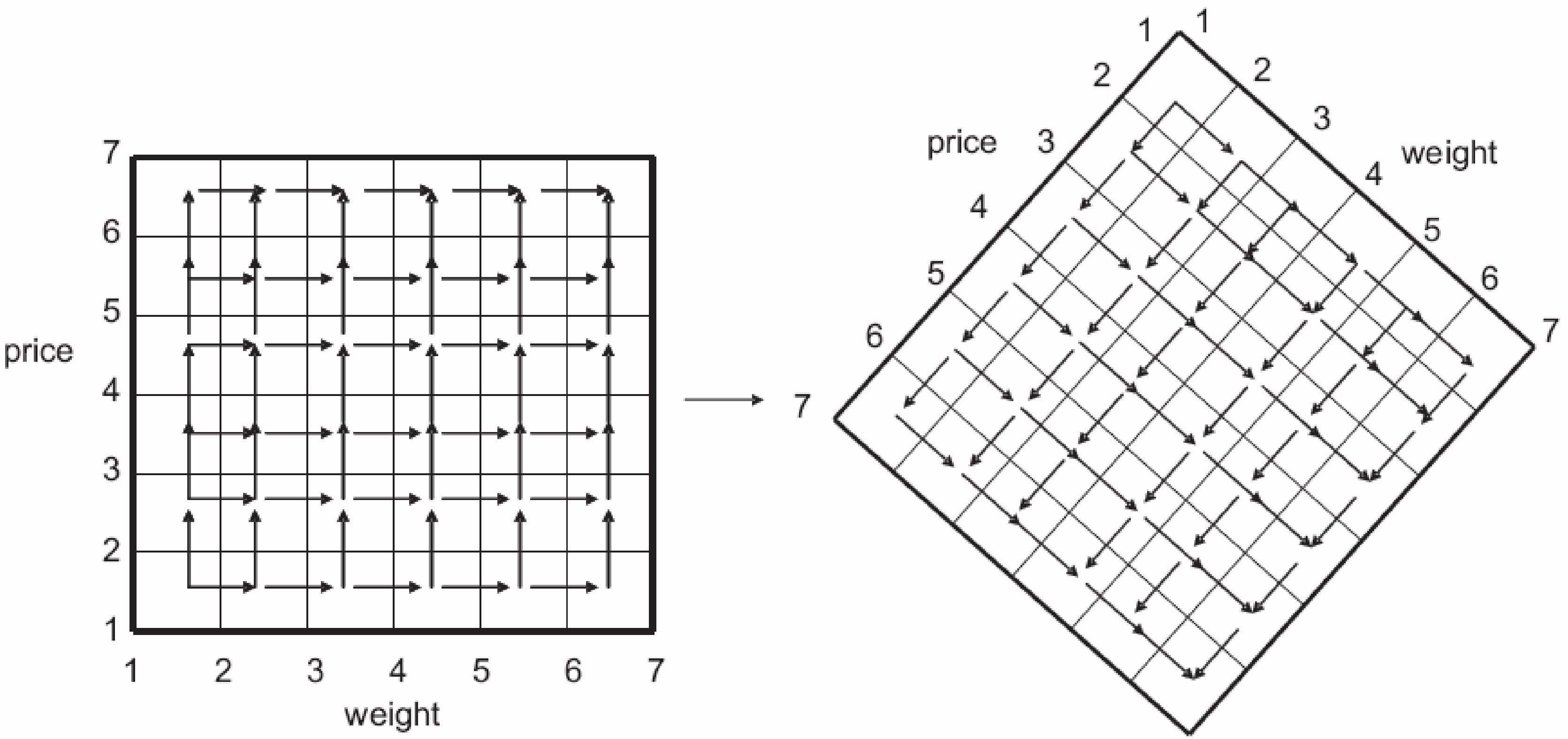


Data



Lattice

Data Cube for Dominant Relationship Analysis : DADA

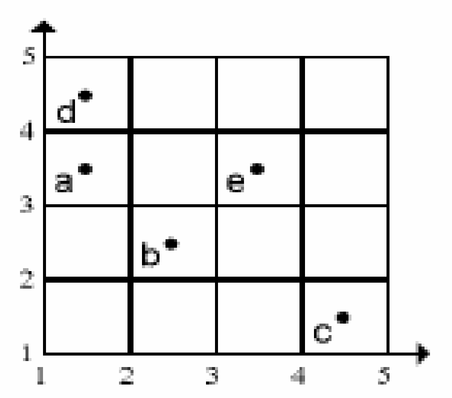


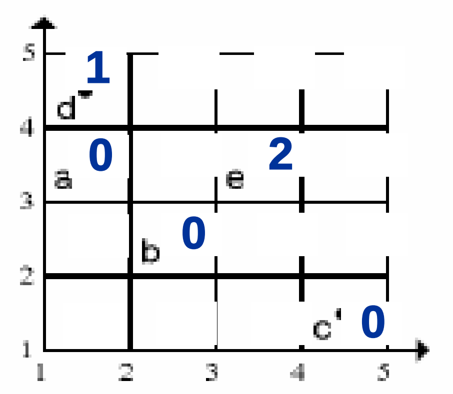
- Dominating lattice: with measure at each cell/node p being dominating(p, C, D).
- Dominated lattice: with measure at each cell/node p being dominated(C, p, D).

Definition of DADA

- DADA is a data cube formed from EITHER of:
  - A dominating lattice
  - A dominated lattice

- Having DADA, skyline points can be easily obtained by return those points whose dominated numbers equal to 0.

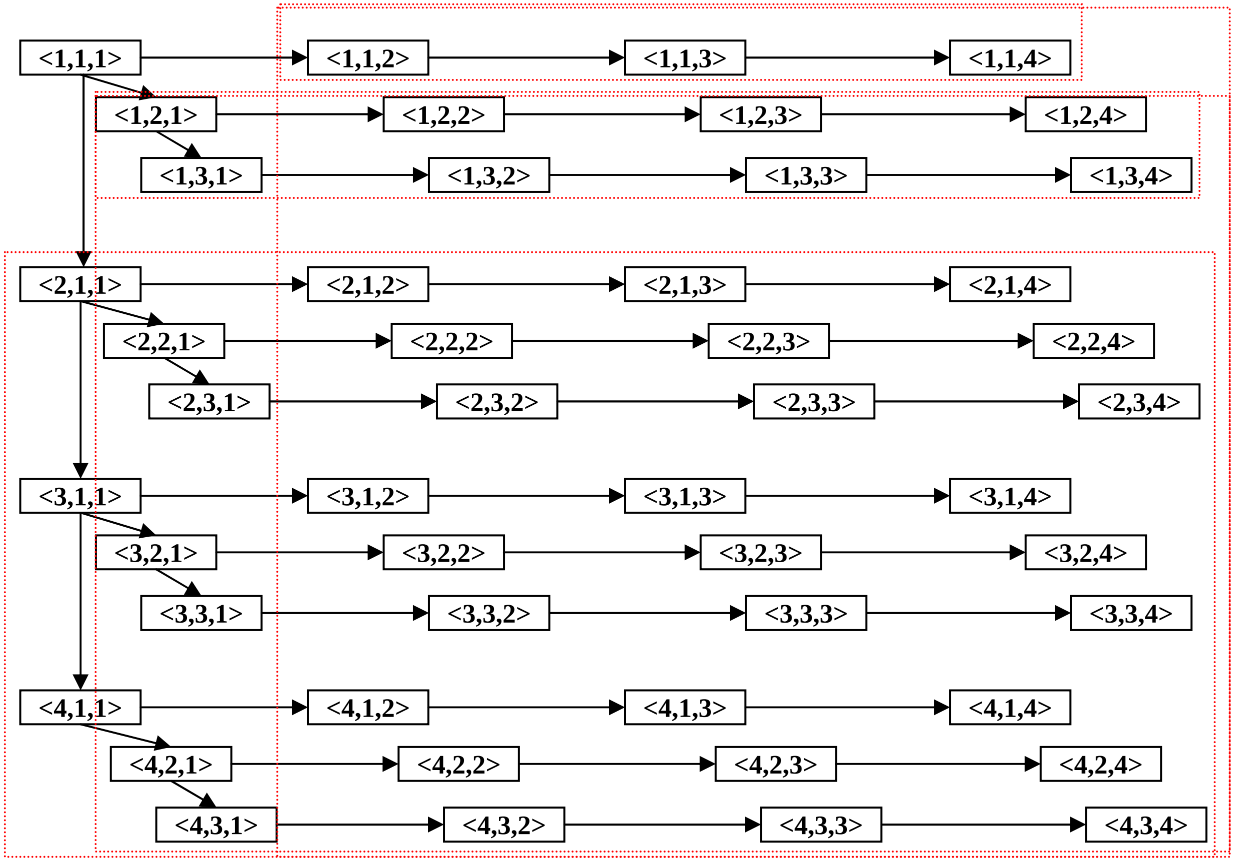




- Besides skyline queries, DADA can be used to answer many other interesting queries, such as LOQ, SAQ, and CDQ.

Computation of DADA

- Main Idea:
  - First, convert the cube lattice into a cell enumeration tree
  - Then, for any cell p, the measure can be obtained by summing up all its children's aggregations.



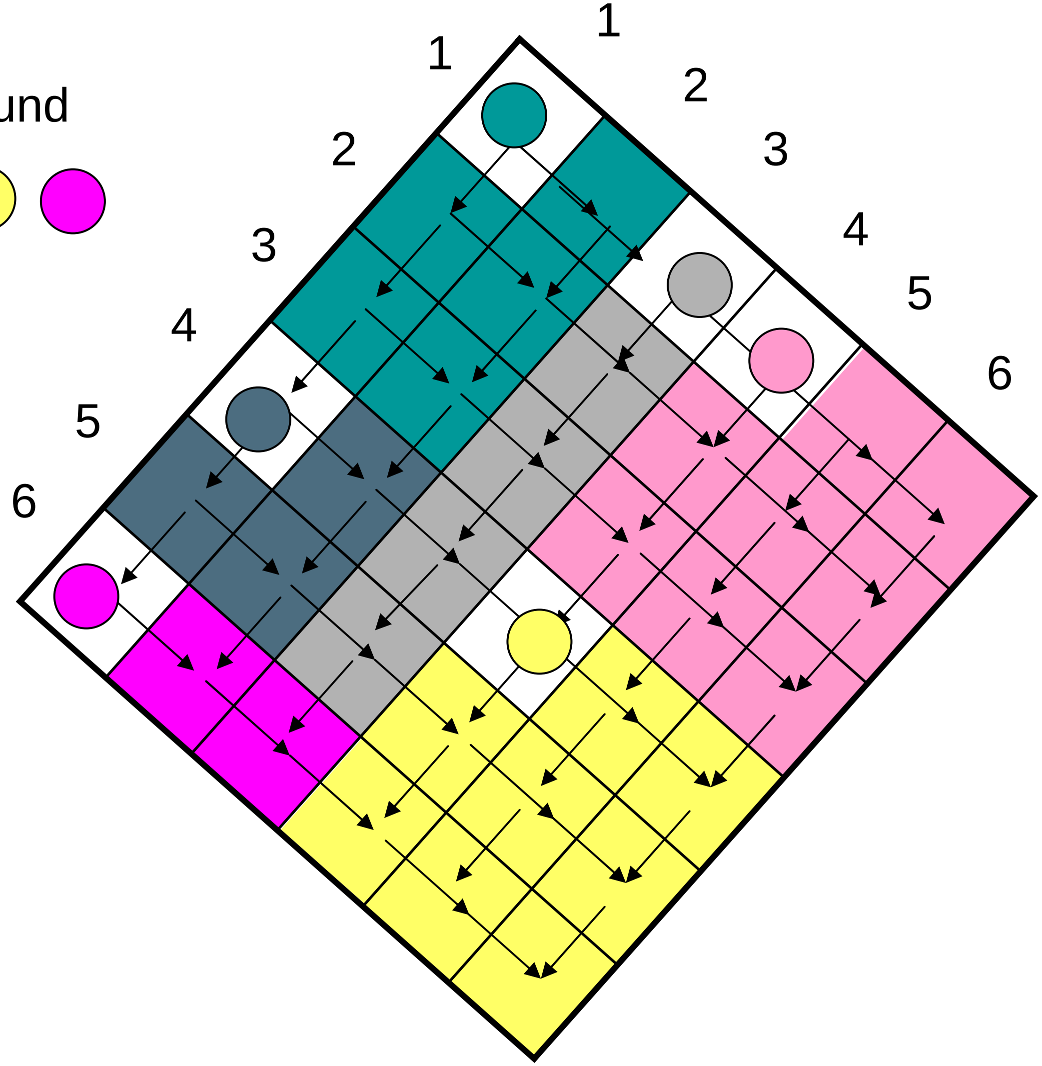
Question: Why we can not always use the full space {X, Y, Z}?

- DADA can be compressed by partitioning all cells into equivalence classes
- The partition must satisfy:
  - Convex
  - *Any equivalence class is the* ***maximal*** *set of cells that:*
    - *dominate the same set of points*
    - *are bounded in a MBR (minimum bounding box).*

Compression

Unique upper bound

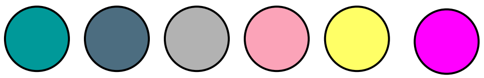
Each partition is represented by a unique upper bound



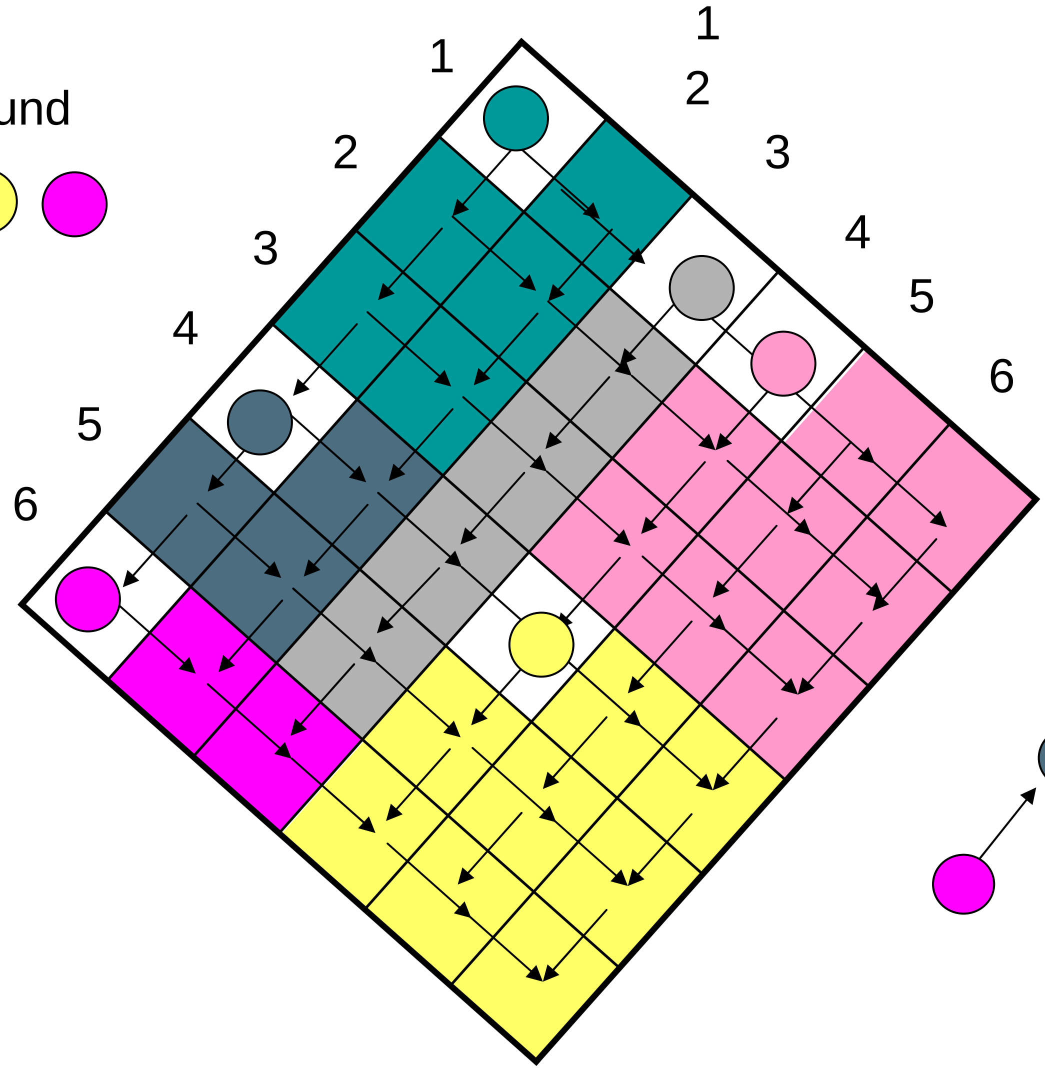
- The advantages of Compression
  - Reduce the storage
  - Enable efficient query processing
    - A D*-tree can be constructed by using the upper bounds as representatives

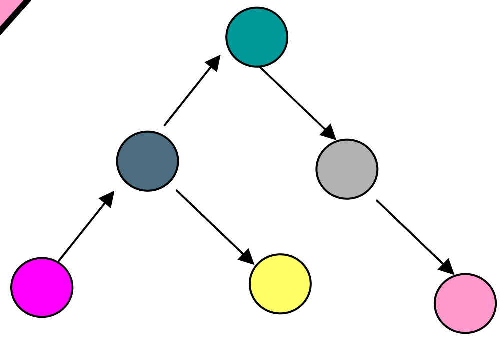
Compression

Unique upper bound



Each partition is represented by a unique upper bound

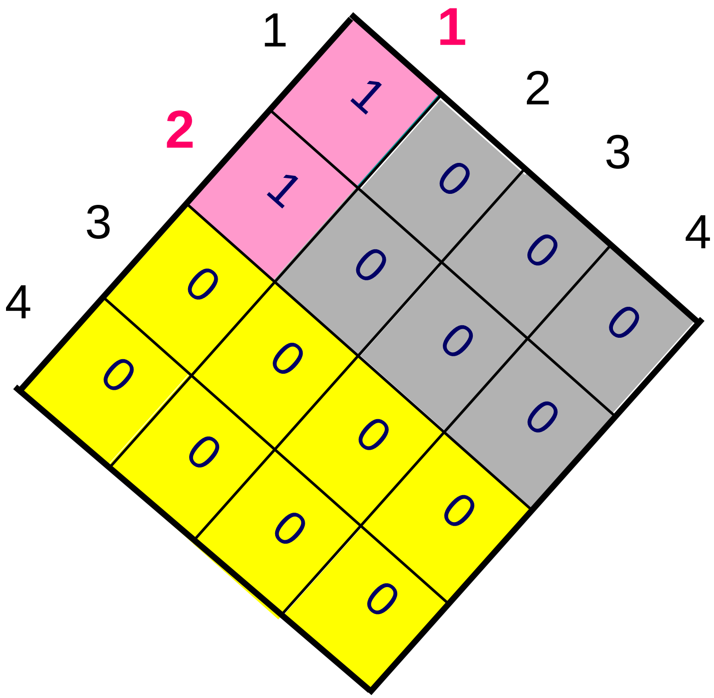


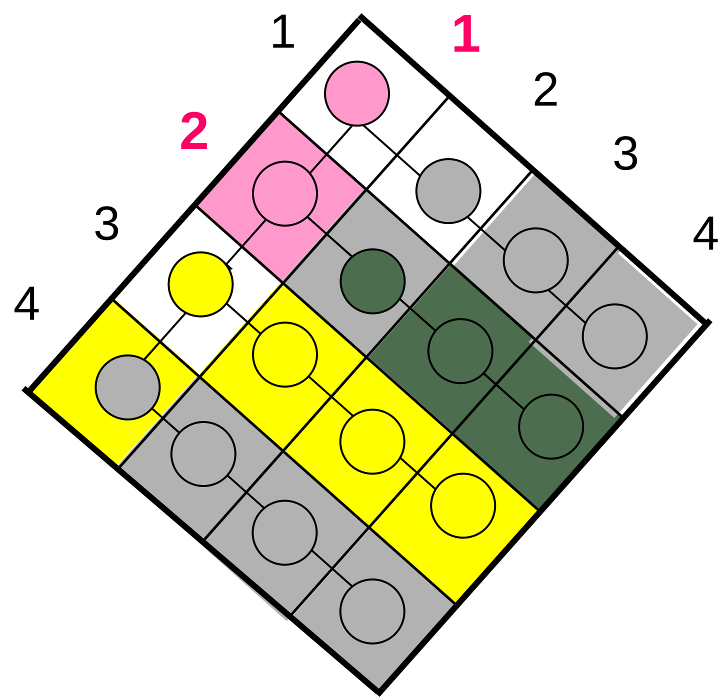


A D* tree

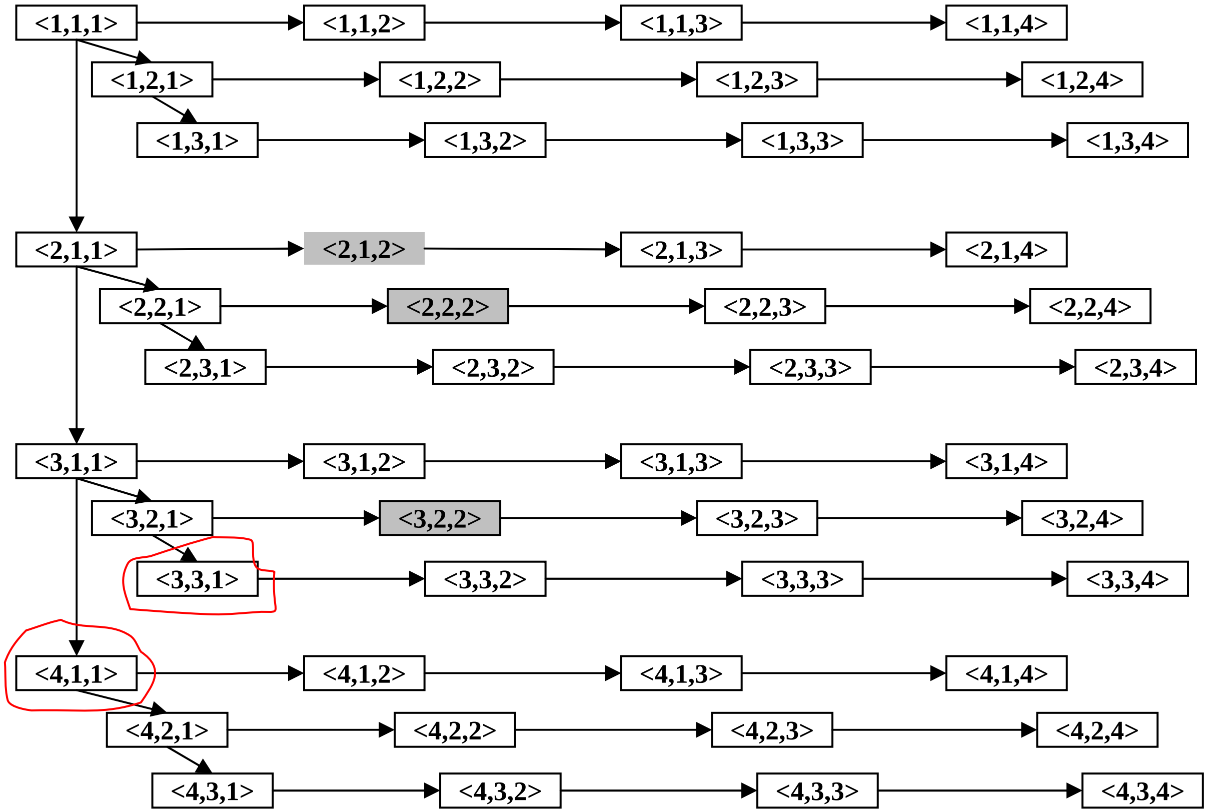
Integrate Compression with Computation

- For example, assume there is only one point in cell <2,1>





- Explores the BUC-like property:
  - whenever a partition on some dimension *d* contains an empty set of points, the number of points dominated by all cells expanded from this partition in subspace $\{D_{d+1},\ldots,D_N\}$ will be 0.



For example: assume there are three true points

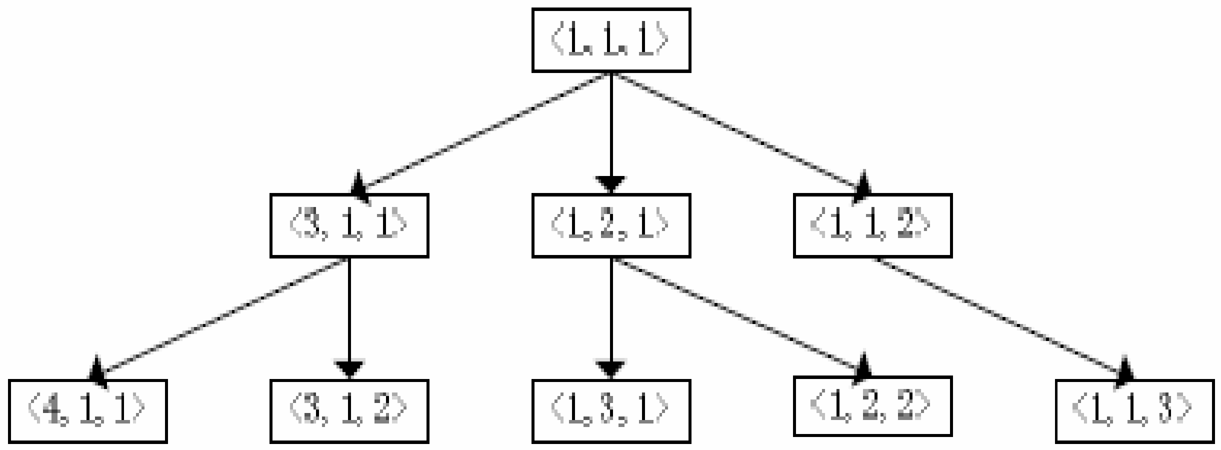
dominating(<4, 1, 1>,C,{X, Y, Z}) = 0

dominating(<3,3, 1>,C,{Y, Z}) = 0

Index version of DADA

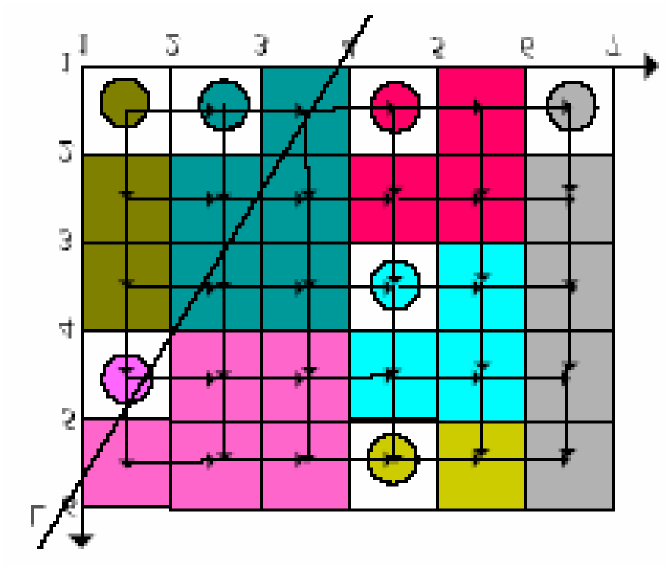
- DADA needs too much space

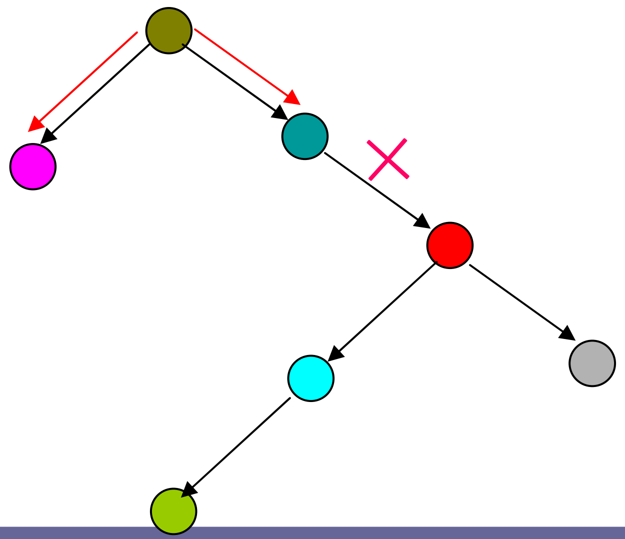
- D*-Tree, a compress and indexed version of DADA



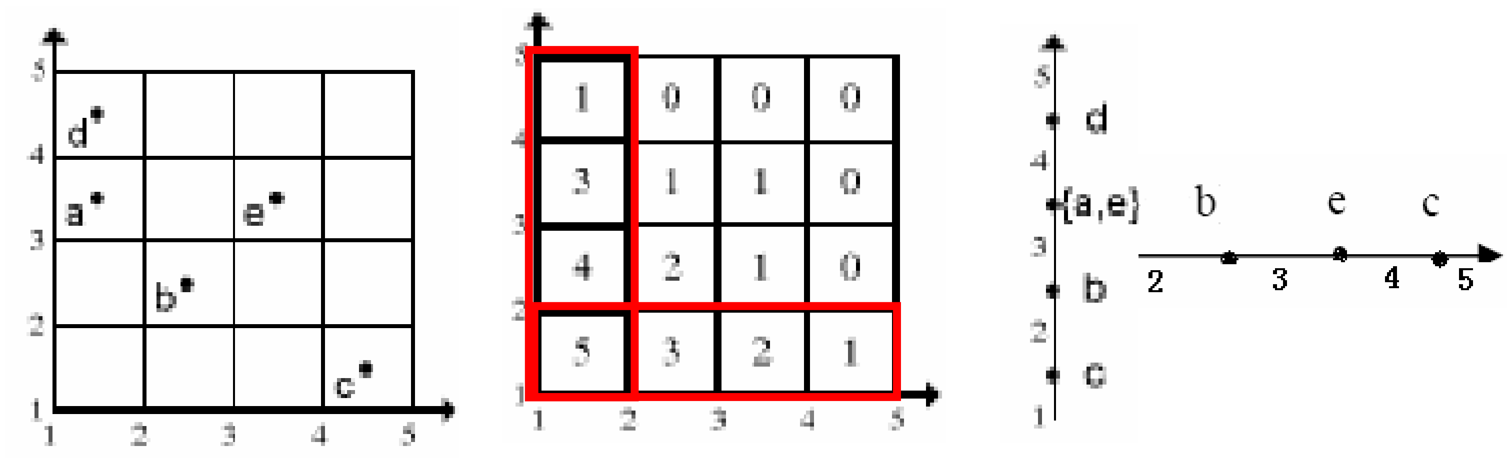
A sample D*-Tree for 3D

- Depth-first search on the D*-tree from the root
- Whenever a class is cut through by the plane L, then all the nodes below it in the D*-tree can be ignored.





- Compare p to all points in C in the dimensions of D’ and ignore the effect of other dimensions.



CDQ Query Processing

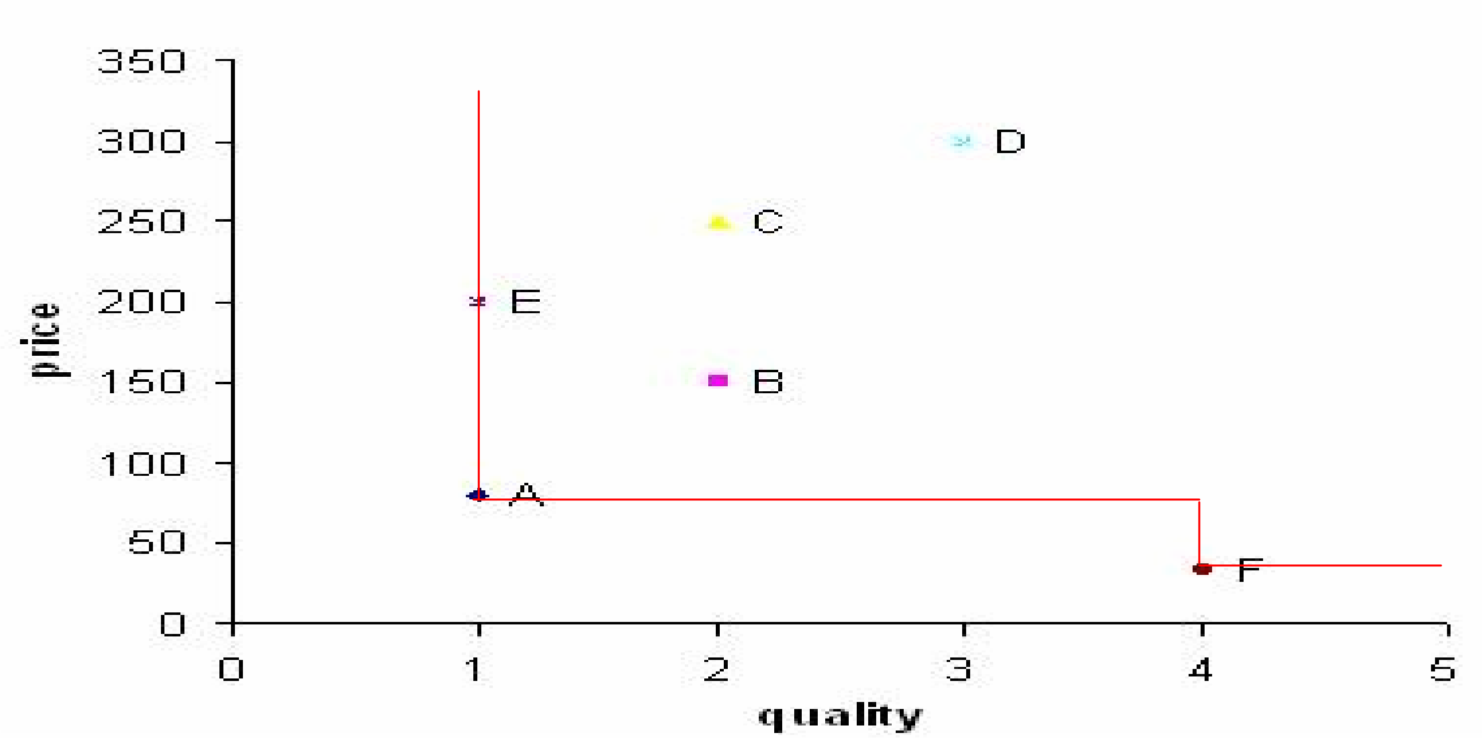
- Maintain two Lists
  - List A, for points dominated by A
  - List B, for points dominated by B
- Algorithm steps:
  - Search the D* tree, once find a class CL whose lower bound is dominated by A, put CL and all its children to List A
  - Search the subtrees of CL, once find a class CL' whose lower bound is dominated by B, put CL' and all its children to List B, pruning off the search on the subtrees of CL'
  - After finished the search, accumulate the nums of all classes on List A and B

Table of Contents

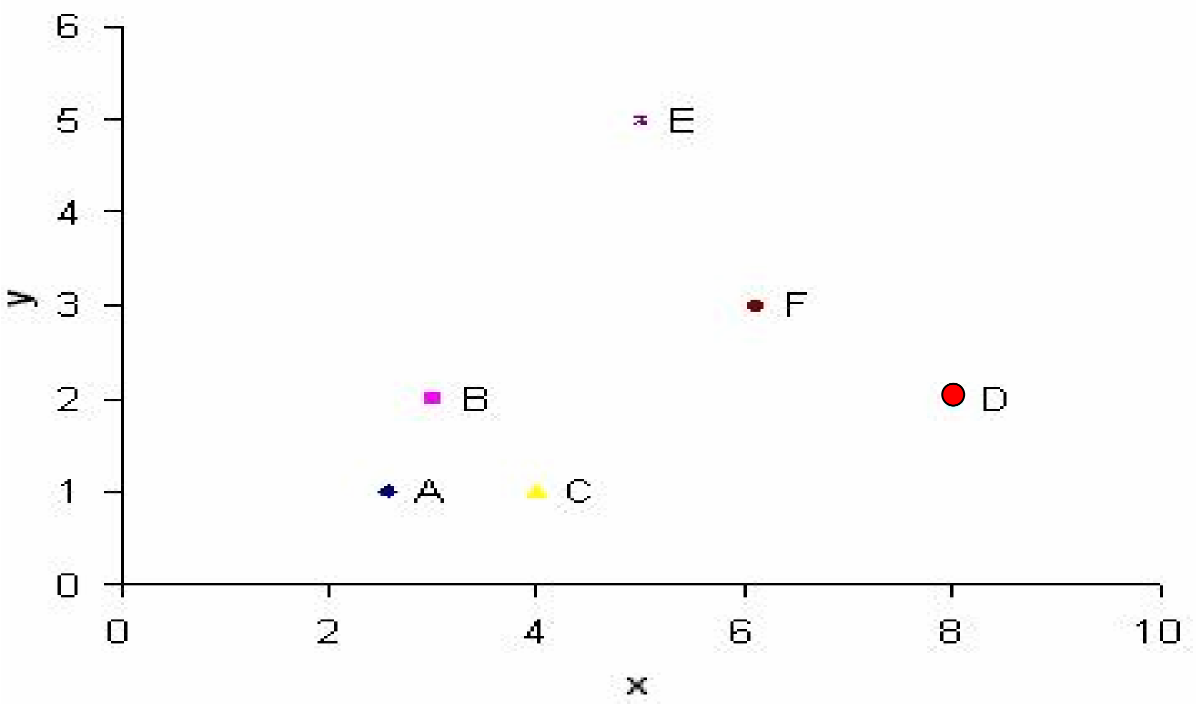
- Basic
  - Dominance Property
    - D-dominance
    - Subspace dominance
    - K-dominance
  - Algorithms
    - Basic Algorithms
    - Advance Algorithms
  - Dominant Relation Analysis
    - Data cube for dominant relation analysis
    - Neighborhood dominant analysis

Skyline

- Example: Hotel (price, Quality)



spatial location



Two Kinds of attributes

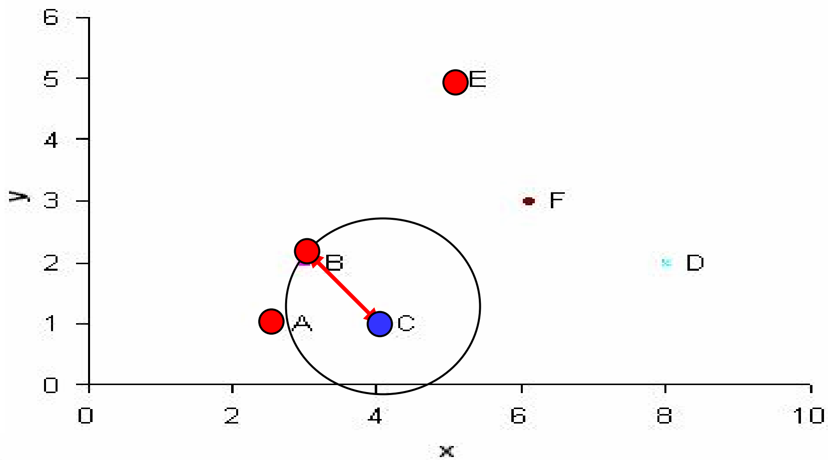
- Unlike the quality and price, the attribute x or y can not be said to be good or better if its value is small or large.

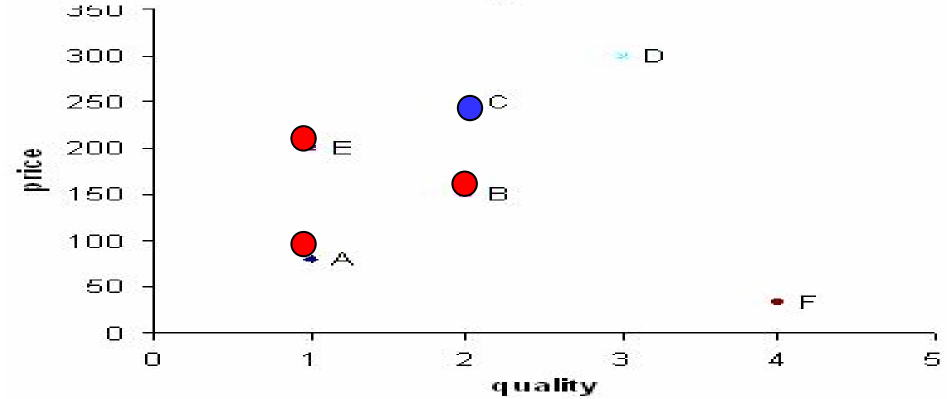
- To distinguish these two types of attributes
  - min/max attributes: such as quality and price
  - Spatial attributes: such as x and y

Perspective of Management

- The objective of a hotel manager:
  - to maximize the price (and consequently, the profit) for a given quality within certain constraints
    - Price and quality of competing hotels
    - The distance to the competing hotels

NDQ



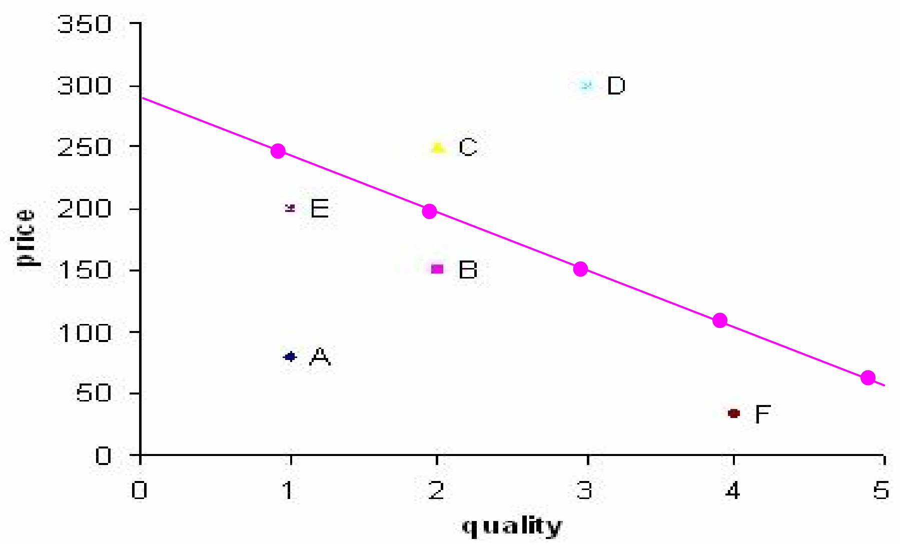


- ND(C) = B
- ndd(C)

Given any arbitrary object q in H, find its nearest dominator ND(q)

LDPQ

- Least Dominated, Profitable Points Query
  - Motivation
    - Hotel manager may want to ask: which hotel *q* is profitable while having the largest distance to its nearest dominator?



- *Since ndd(D) > ndd(C), hotel D is the answer*

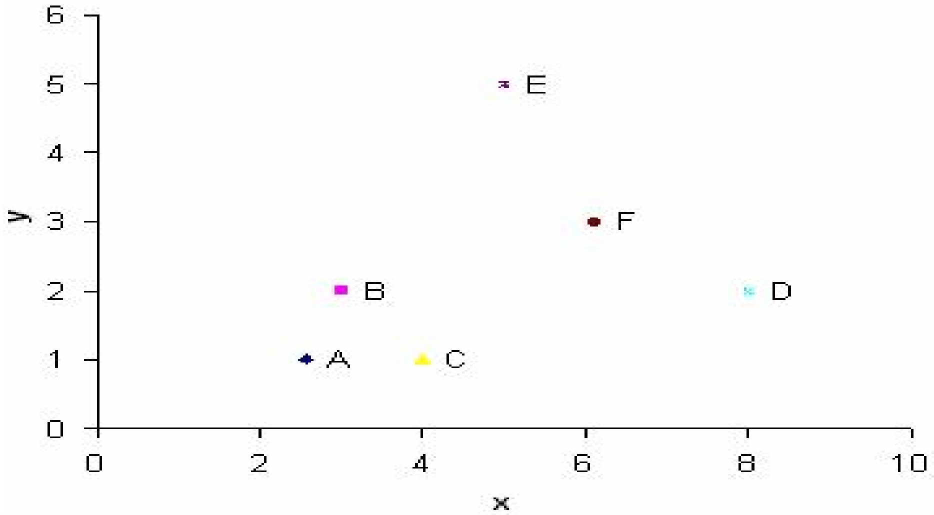
LDPQ

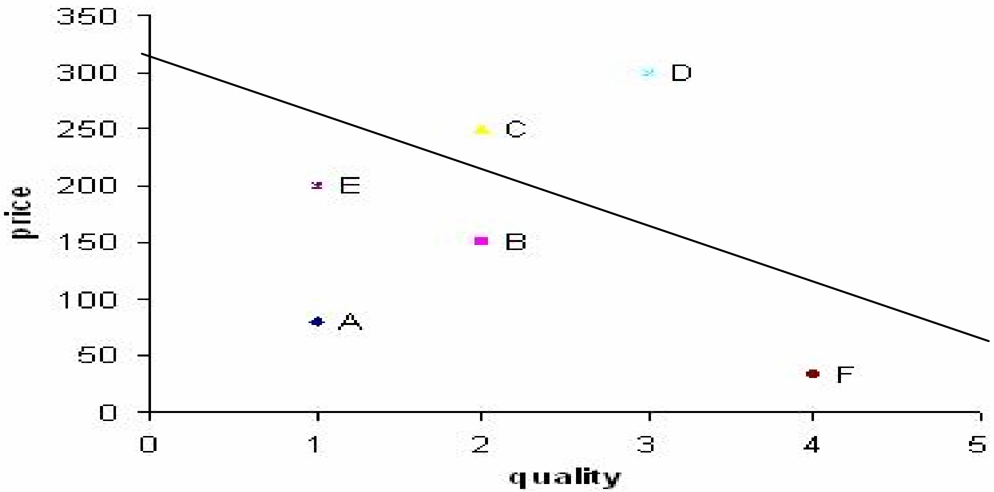
- Definition:
  - Given a dataset H and a hyper plane P, find the point t, which satisfies:
    - t is profitable
    - ndd(t) is the largest among all profitable points

ML2DQ

- Minimal Loss and Least Dominated Points Query
- Definition:
  - Given a profitability constraint and a distance threshold δ, find a hotel q such that:
    - $ndd(q) \geq \delta$
    - the difference between the price charged and the minimal profitable price is the smallest

Example for ML2DQ





- ndd(A) = ∞
- ndd(B) = 1.1
- ndd(E) = 4.6

- Assume δ=4.5

E will be returned

Neighborhood Dominant Queries

- NDQ \ LDPQ \ ML2DQ
  - A Family of query types considering the relationship between min/max and spatial attributes.
- two alternative query processing methods
  - Symmetrical
  - Asymmetrical

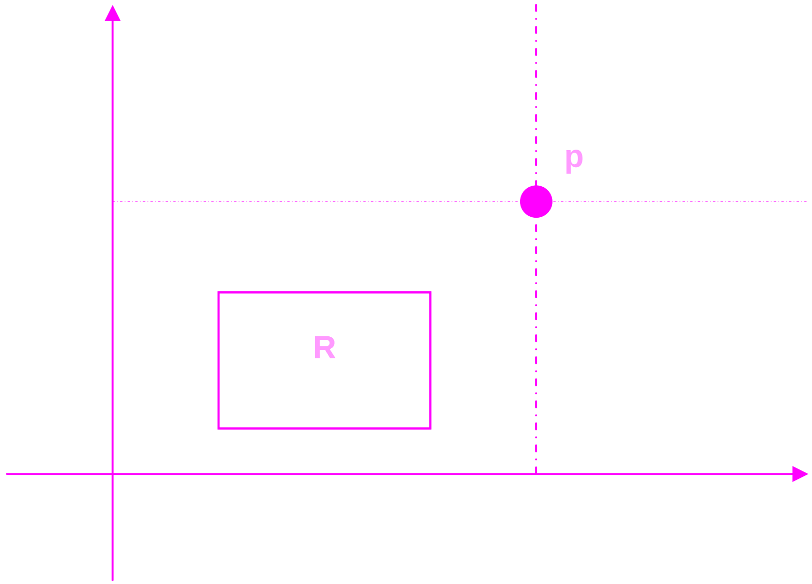
Symmetrical Methods

- treat min/max, spatial attributes as equal
- index them together in one R-Tree

Dominant Relationship (for NDQ)

- The dominant relationships between an MBR R and a given point p can be classified into three cases:
  - if $R_{ui} \leq p_i$ for all min/max attribute I, then

    all points from R definitely dominate p

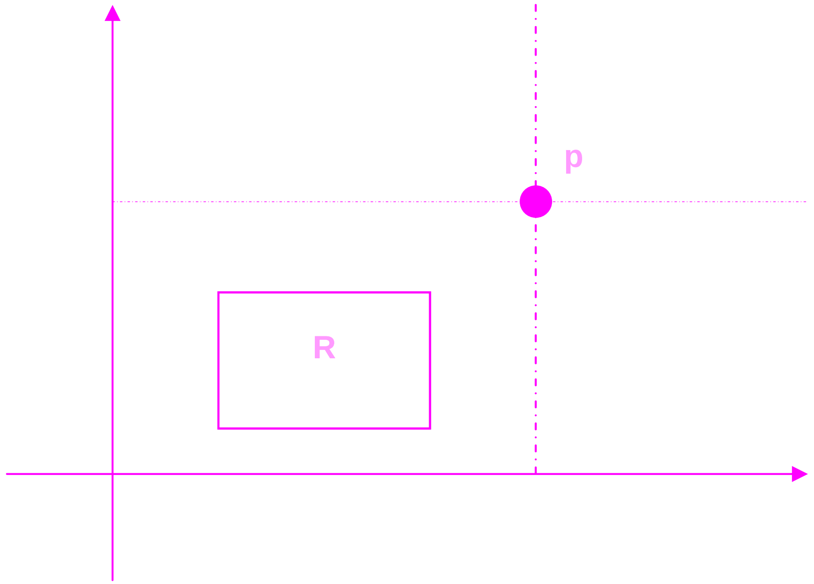


Dominant Relationship (for NDQ)

- The dominant relationships between an MBR R and a given point p can be classified into three cases:
  - if $R_{li} \leq p_i$ for all min/max attribute i,

    $R_{uj} < p_j$ for |D|-1 min/max attributes j

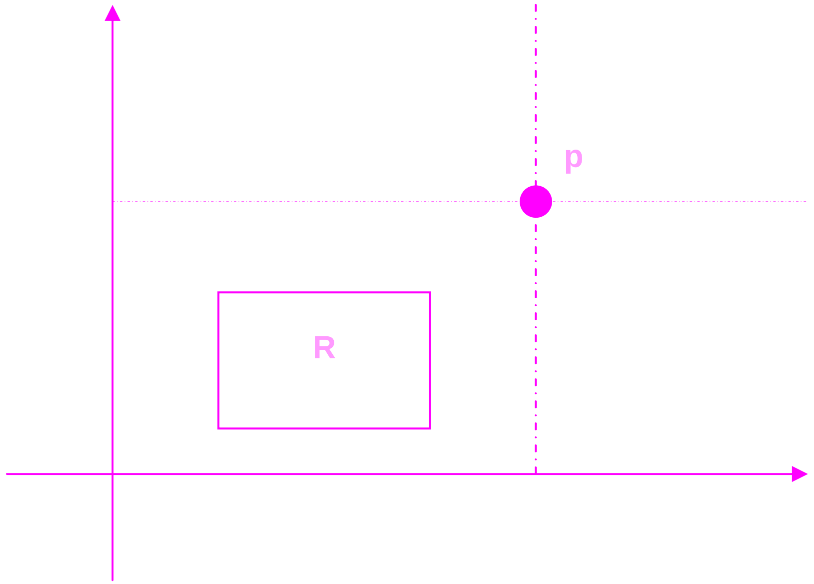
    then some points from R definitely dominate p



Dominant Relationship (for NDQ)

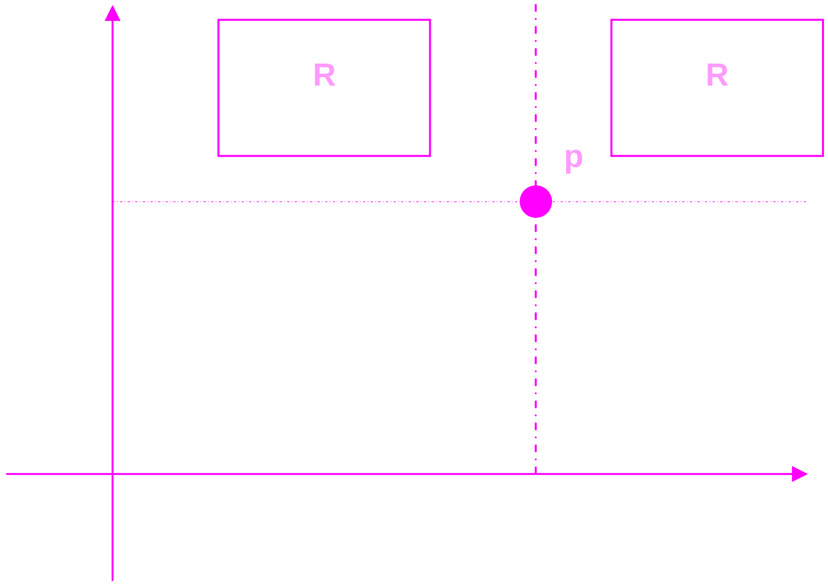
- The dominant relationships between an MBR R and a given point p can be classified into three cases:
  - if $R_{li} \le p_i \le R_{ui}$ for all min/max attribute I,

    then some points from R could dominate p



Dominant Relationship (for NDQ)

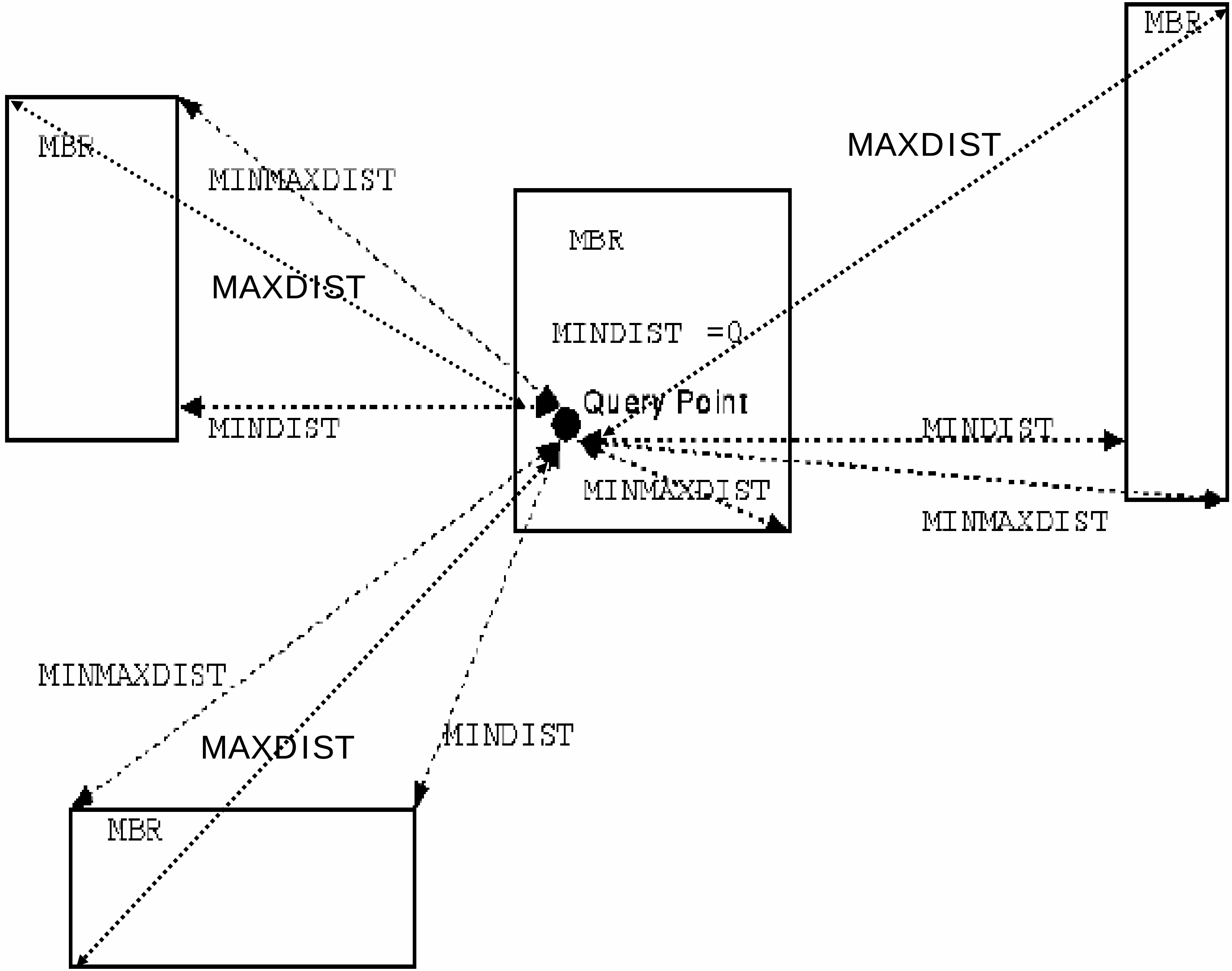
- The dominant relationships between an MBR R and a given point p can be classified into three cases:
  - Other cases: there does not exist dominant relationship between R and p



Spatial Relationship (for NDQ)

- Use three metrics to describe the distance between a MBR R and a point p
  - MINDIST(p,R): the nearest distance between p and any point in R
  - MAXDIST(p,R): the furthest distance between p and any point in R
  - MINMAXDIST(p,R): minimized distance upper bound that guarantee R contains at least one point which can dominate p.

Note: These metrics are computed using only spatial attributes



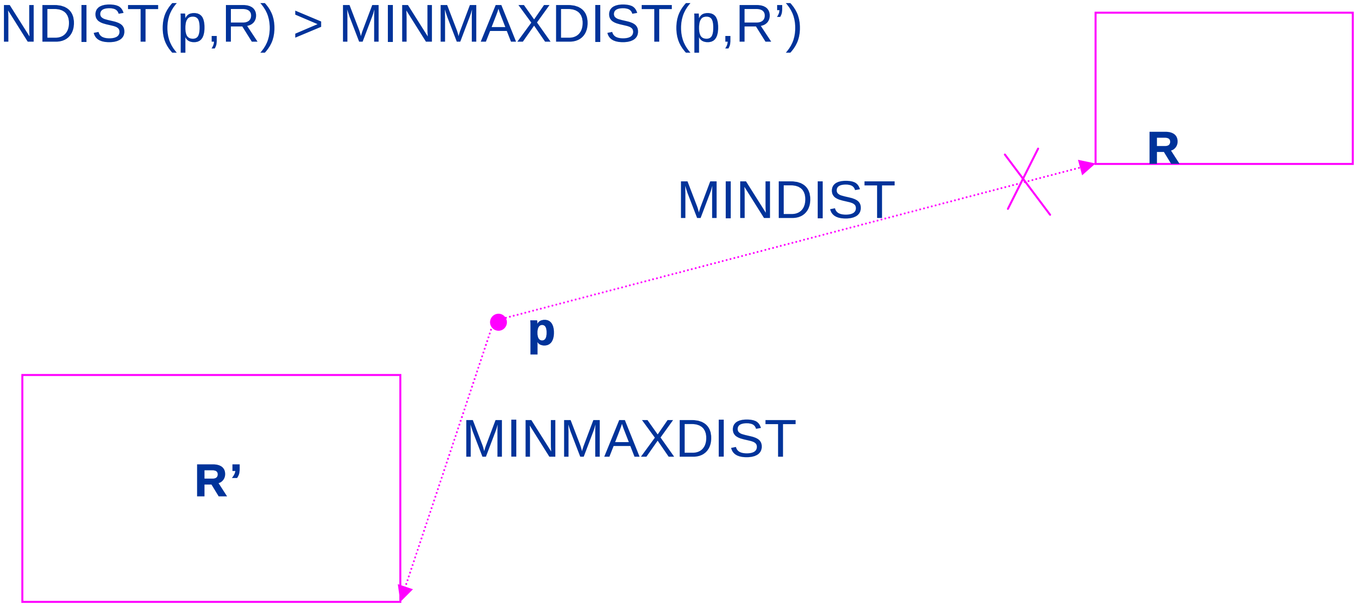
MBR
MINMAXDIST
MAXDIST
MINDIST
MBR
MINDIST =0
Query Point
MINMAXDIST
MBR
MAXDIST
MINDIST
MINMAXDIST
MINMAXDIST
MAXDIST
MINDIST
MBR

Best First Traversal Algorithm

- Start from the root MBR of R-tree, place its children MBRs into the heap
- Within the heap, order MBRs by:
  - Case 3, case 2, case 1
  - MINDIST, ascending
- Beginning from the top MBR of the heap, recursively extracting children of MBRs, and inserting those potential dominators of p into the heap.
- Algorithm terminated when the heap empty

Pruning Strategy 1 (for NDQ)

- An MBR R is discarded if there exists an R’ s.t.
  - p and R’ correspond to case 3
  - MINDIST(p,R) > MINMAXDIST(p,R’)



Pruning Strategy 2 (for NDQ)

- An MBR R is discarded if there exists an R' s.t.
  - p and R' correspond to case 2
  - MINDIST(p, R) > MAXDIST(p, R')

Why not use MINMAXDIST in case 2?

Can not ensure there exists a dominator in this distance

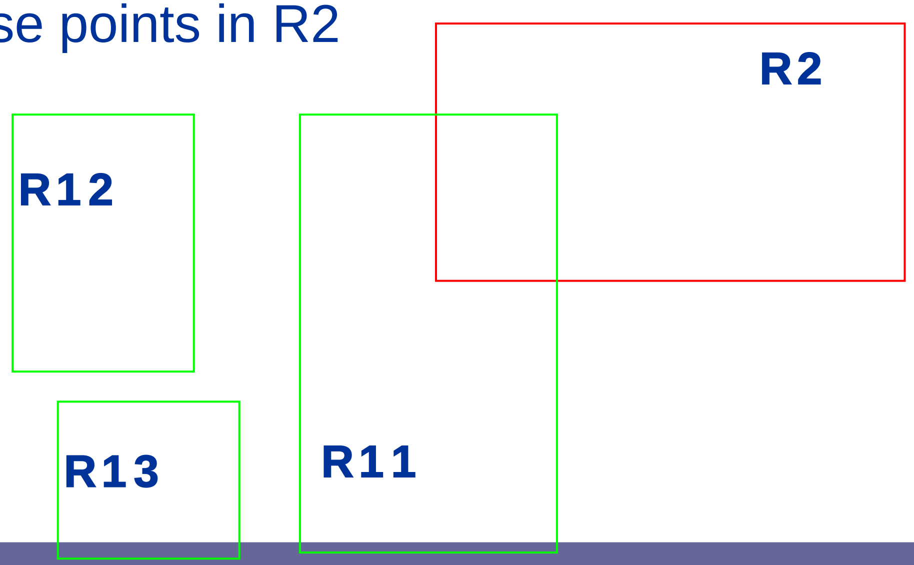
LDPQ with Symmetrical R-tree

- Naïve method:
  - First, perform a NDQ search for all points in the profitable region
  - Second, select the point with the largest nearest dominator distance

- More efficient method:
  - merge above two steps into one

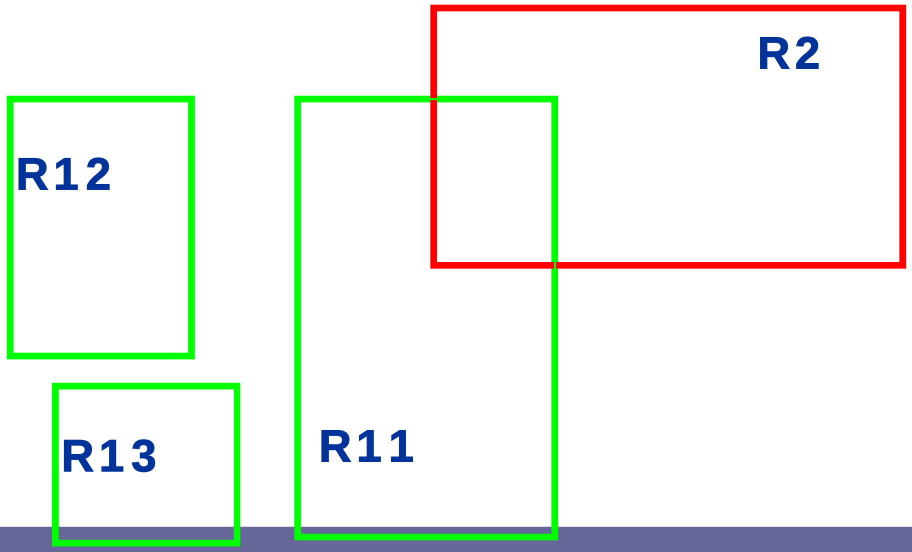
LDPQ with Symmetrical R-tree

- Monitor two types of MBRs
  - PdMBR: MBRs that are potentially dominated by some points and are candidates for the output answers
    - Any MBR in the R-tree can be PdMBR unless it is pruned
  - For each PdMBR R2,
    - PnrMBR: MBRs that potentially contain the nearest dominators for those points in R2



LDPQ with Symmetrical R-tree

- The dominant relationship between MBRs from PdMBR and PnrMBR can be following:
  - Case1 : some points from R1 could dominate some points from R2
  - Case 2: some points from R1 definitely dominate all points from R2
  - Case 3: all points from R1 definitely dominate all points from R2

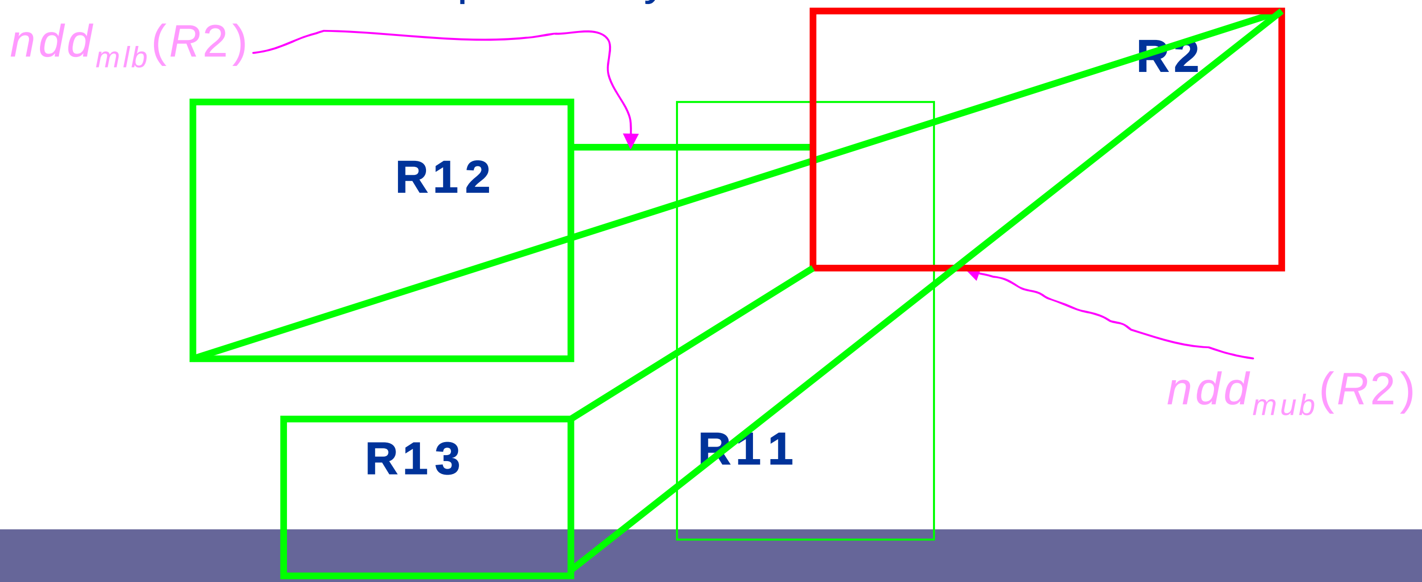


Another three useful Metrics

- MINMINDIST(R1,R2)
- MAXMAXDIST(R1,R2)
- MAXMINMAXDIST(R1,R2)
  - … details can be referenced in the paper

Two Thresholds for Pruning

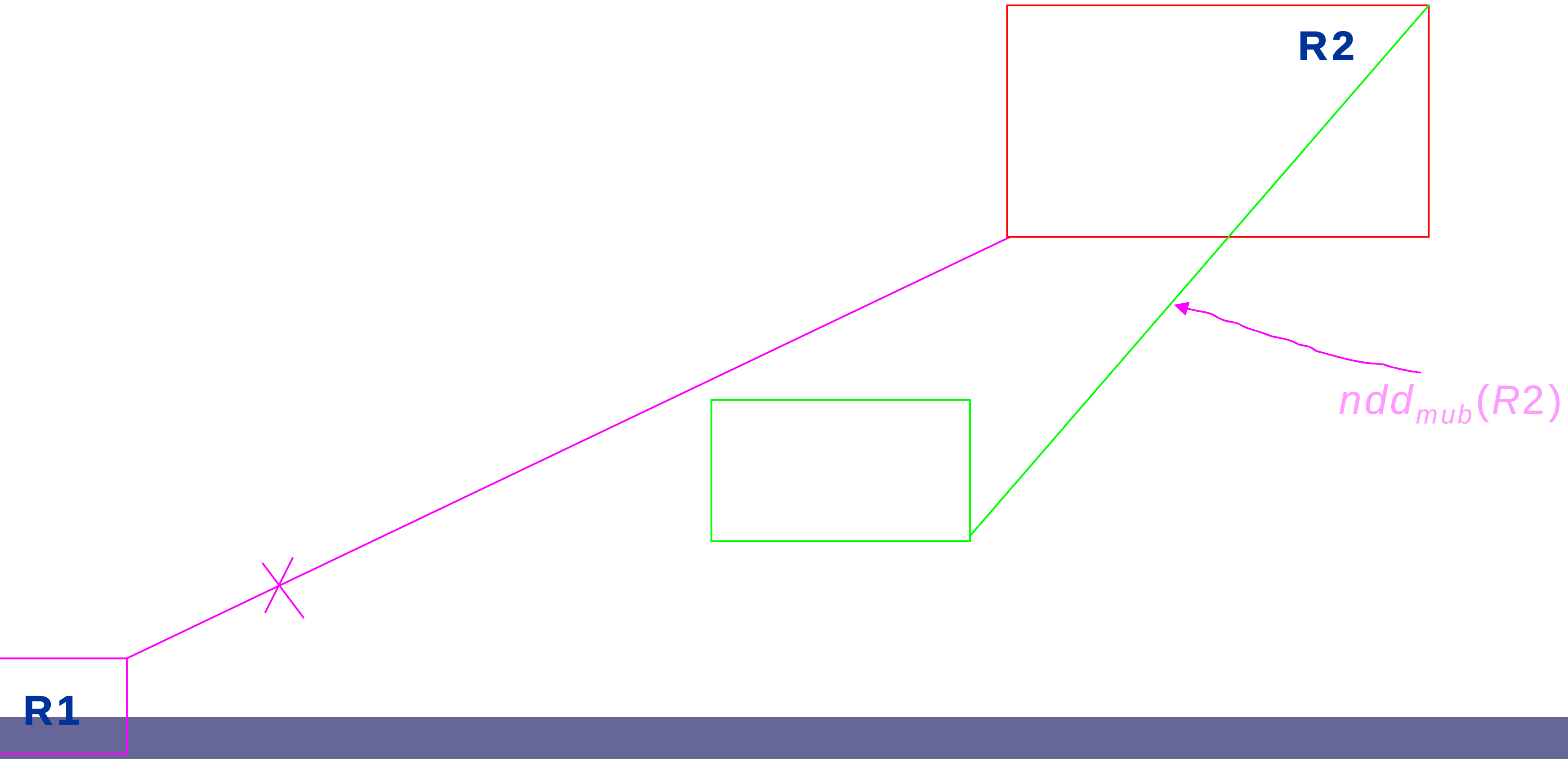
- For each PdMBR R2, maintain two variables:
  - *ndd_{mlb}(R2)*: minimum lower bound distance between R2 and its PnrMBRs
    - case 3 or case 2: updated by MINMINDIST
  - *ndd_{mub}(R2)*: minimum upper bound distance between R2 and its PnrMBR
    - guarantee there is at lease one point can dominate all points in R2
    - case3: updated by MAXMINMAXDIST
    - case2: updated by MAXMAXDIST



Local Pruning (for LDPQ)

- Given R2, R1 can be removed from PnrMBR(R2) if:

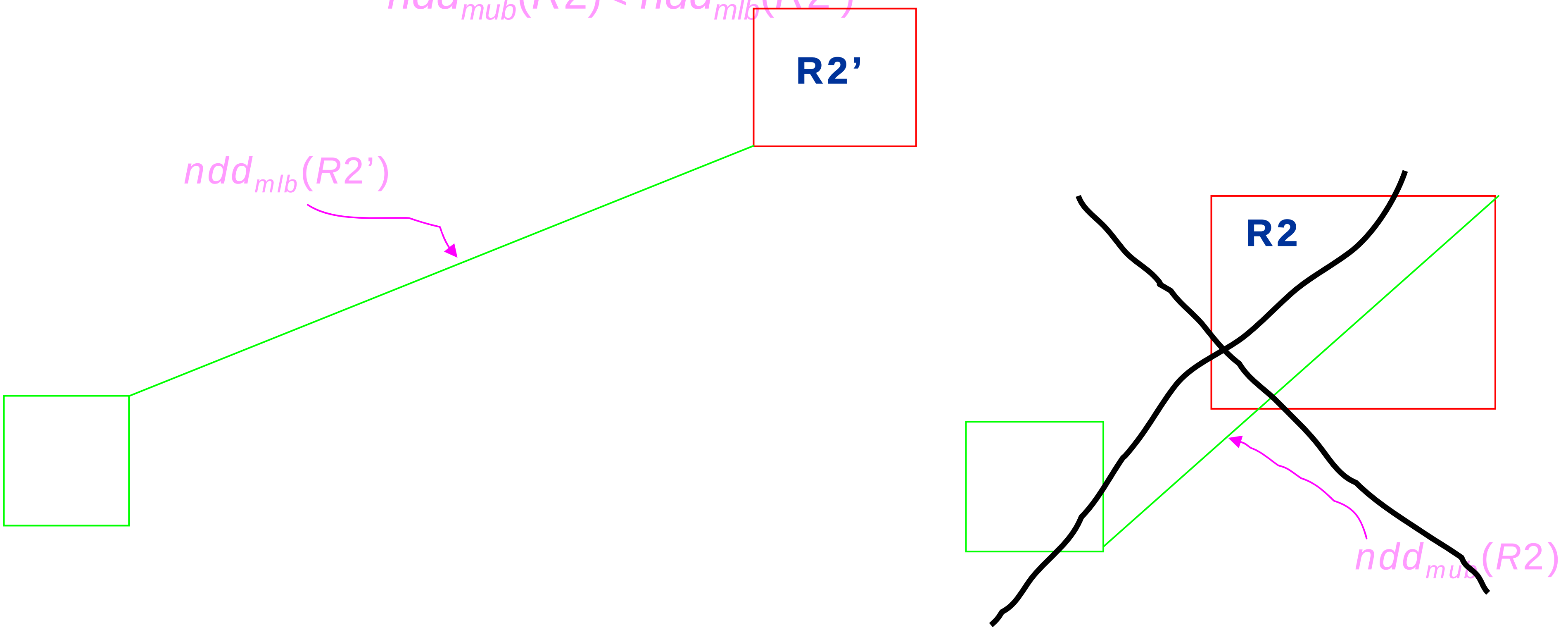
$$\text{MINMINDIST(R1,R2)} > ndd_{mub}(R2):$$



Global Pruning (for LDPQ)

- Any R2 can be removed from PdMBR if there exists a R2’ s.t.

$$ndd_{mub}(R2) < ndd_{mlb}(R2')$$



ML2DQ with Symmetrical R-tree

- The aim of this type query is:
  - to find a point q in the unprofitable region s.t.:
    - the distance to P is minimized
    - $ndd(q) \geq \delta$
- To process this type query:
  - Adopt the same best first search approach as LDPQ
  - Pruning strategies:
    - Only considering the MBRs intersecting the non-profitable region
    - R1 is removed if $ndd(R1) < \delta$
    - R1 is removed if R1 is far away from P

Asymmetrical Methods

- Spatial attributes and min/max attributes play different roles when query is processed.

- The whole process includes two steps:
  - Clustering into micro-cluster (spatial attributes)
  - Constructing a Asymmetrical R-Tree (min/max attributes), and associate the spatial info with the min/max info

The First Step

- Clustering into micro-cluster
  - Points are clustered into k micro-clusters by spatial attributes
  - Finished by a typical pre-processing algorithm BIRCH
  - Each micro-cluster MCi, has:
    - Cluster id: i
    - Mean value: MCi.m
    - Radius: MCi.r

The Second Step

- Constructing an Asymmetrical R-Tree
  - MBRs are formed by min/max attributes

- In order to capture the spatial info
  - Each MBR is associated with a bitmap of size k. each bit represents one micro-cluster
    - If some point of MCi appears also in the MBR, set bit i to 1, otherwise 0

NDQ with Asymmetrical R-Tree

- Given a query point p, and a micro-cluster MCi:
  - *MinDist(p,MCi) = max{dist(p,MCi,m) –MCi.r, 0}*
  - *MaxDist(p,MCi) = dist(p,MCi,m) +MCi.r*

- Based on this, redefine:
  - *MINDIST(R, p)*
  - *MAXDIST(R,p)*
  - *MINMAXDIST(R,p)*
    - *…details can be referenced in the paper*

LDPQ(ML2DQ) with Asymmetrical R-Tree

- Given any two micro-clusters MCi and MCj:
  - *MinDist(MCi,MCj) =max{dist(MCi.m, MCj.m)-MCi.r-MCj.r , 0}*
  - *MaxDist(MCi,MCj) =dist(MCi.m, MCj.m)+MCi.r+MCj.r*

- Based on this, redefine:
  - *MINMINDIST(R1,R2)*
  - *MAXMAXDIST(R1,R2)*
  - *MAXMINMAXDIST(R1,R2)*
    - *…details can be referenced in the paper*