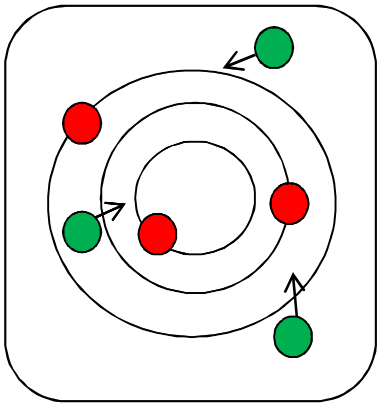
Towards Indexing Functions: Answering Scalar Product Queries

Arijit Khan, Pouya Yanki, Bojana Dimcheva, Donald Kossmann

Systems Group
ETH Zurich

Moving Objects Intersection Finding

ETHzürich



Moving Object Database

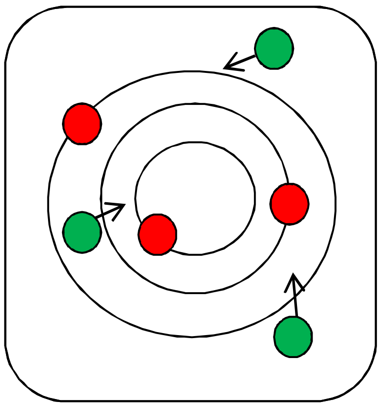
● → r, ω

● → p, q, u, v

- Position at a future time instance t

  - ● [x = r cos(ωt)   y = r sin(wt)]
  - ● [x = p + ut   y = q + vt]

Moving Objects Intersection Finding



Moving Object Database

● → r, ω

● → p, q, u, v

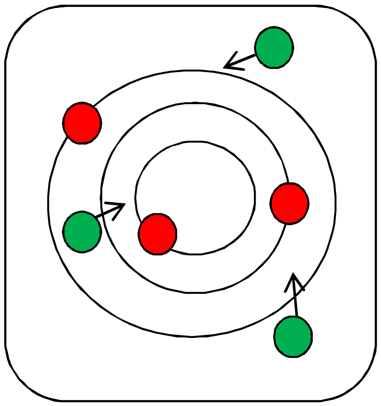
- **Find all object pairs that will be within distance S at time instance t**

$$AX_1 + BX_2 + CX_3 + DX_4 + EX_5 + FX_6 + GX_7 \le S^2$$

$X_1 = r^2 + p^2 + q^2 + 2rp + 2rq$	$A = 1$
$X_2 = 2[u(r-p) + v(r-q)]$	$B = t$
$X_3 = -2rp$	$C = 1+ \sin(\omega t)$
$X_4 = -2rq$	$D = 1+ \cos(\omega t)$
$X_5 = -2ru$	$E = t[1+ \sin(\omega t)]$
$X_6 = -2rv$	$F = t[1+ \cos(\omega t)]$
$X_7 = u^2+v^2$	$G = t^2$

Moving Objects Intersection Finding

ETH zürich



Moving Object Database

(red dot) → r, ω

(green dot) → p, q, u, v

- Find all object pairs that will be within distance S at time instance t

$$AX_1 + BX_2 + CX_3 + DX_4 + EX_5 + FX_6 + GX_7 \le S^2$$

$X_1 = r^2 + p^2 + q^2 + 2rp + 2rq$
$X_2 = 2[u(r-p) + v(r-q)]$
$X_3 = -2rp$
$X_4 = -2rq$
$X_5 = -2ru$
$X_6 = -2rv$
$X_7 = u^2+v^2$

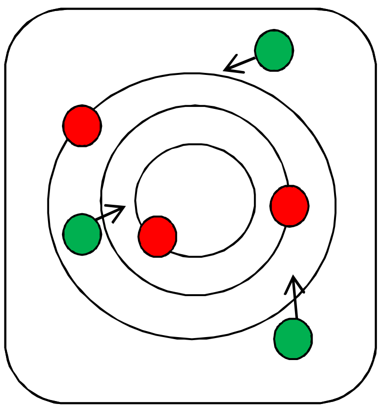
Function (known)

$A = 1$
$B = t$
$C = 1+ \sin(\omega t)$
$D = 1+ \cos(\omega t)$
$E = t[1+ \sin(\omega t)]$
$F = t[1+ \cos(\omega t)]$
$G = t^2$

Query Parameters (unknown)

Moving Objects Intersection Finding

ETH zürich



Moving Object Database

● → r, ω

● → p, q, u, v

- Find all object pairs that will be within distance S at time instance t

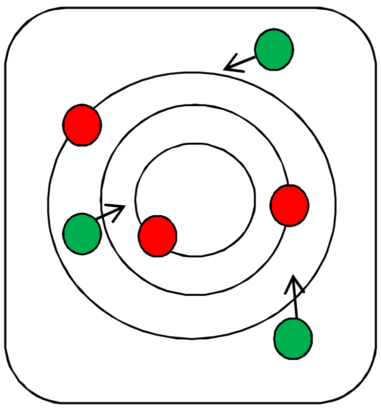
$$\mathbf{A}X_1 + \mathbf{B}X_2 + \mathbf{C}X_3 + \mathbf{D}X_4 + \mathbf{E}X_5 + \mathbf{F}X_6 + \mathbf{G}X_7 \le \mathbf{S^2}$$

Scalar Product Query: $(A\ B\ C\ D\ E\ F\ G) \cdot (X_1\ X_2\ X_3\ X_4\ X_5\ X_6\ X_7) \le S^2$

Query Parameters (unknown) | **Function (known)** | **Inequality Parameter (unknown)**

Moving Objects Intersection Finding

ETH zürich



Moving Object Database

(red dot) → r, ω

(green dot) → p, q, u, v

B-Tree Index will not work !!!

Scalar Product Query: $(A\ B\ C\ D\ E\ F\ G) \cdot (X_1\ X_2\ X_3\ X_4\ X_5\ X_6\ X_7) \le S^2$

- **Query Parameters (unknown)** → $(A\ B\ C\ D\ E\ F\ G)$?
- **Function (known)** → $(X_1\ X_2\ X_3\ X_4\ X_5\ X_6\ X_7)$ ✓
- **Inequality Parameter (unknown)** → S^2 ?

More Applications: Complex SQL Function

ETH zürich

Patient ID	S	B
P1	5	80
P2	6	40
P3	4	70
P4	6	50

Patient Dataset for Heart-Rate Prediction

- **ARIMA Time Series Prediction Model:**
- Heart-Rate at time $t = S \times t + B$

More Applications: Complex SQL Function

ETHzürich

Patient ID	S	B
P1	5	80
P2	6	40
P3	4	70
P4	6	50

Patient Dataset for Heart-Rate Prediction

- **ARIMA Time Series Prediction Model:**
- Heart-Rate at time t = S × t + B
- **Find all patients for whom the predicted heart rate at time t is more than an input threshold H.**

```
CREATE FUNCTION Critical_Patient (
    INPUT double Threshold, double Time
    RETURN PatientID
    FROM Patient
    WHERE  S × Time + B ≥ H )
```

? ? **Unknown**

More Applications: Complex SQL Function

ETH zürich

Patient ID	S	B
P1	5	80
P2	6	40
P3	4	70
P4	6	50

Patient Dataset for Heart-Rate Prediction

- $S \times \mathbf{Time} + B \geq \mathbf{H}$

Unknown

Scalar Product Query

$(\text{Time}, 1) \cdot (S, B) \geq H$

Query Parameters (unknown)

Function (known)

Inequality Parameter (unknown)

More Applications: Complex SQL Function

ETH zürich

Patient ID	S	B
P1	5	80
P2	6	40
P3	4	70
P4	6	50

Patient Dataset for Heart-Rate Prediction

- $S \times \textbf{Time} + B \geq \textbf{H}$

? ?

Unknown

Can we be more efficient than a sequential scan ?

Problem Statement

ETH zürich

- **Inequality Query**

Find all data points x that satisfy:
$(a, F(x)) \geq b$

Applications:

- moving-object-intersection finding
- half-space range search
- complex SQL functions

- **Top-k Nearest Neighbor Query**

Find the top-k data points x satisfying
$(a, F(x)) \geq b$, that also minimize:
$|(a, F(x)) - b| / |a|$

Applications:

- top-k nearest points to hyper plane
- active learning

Related Work

ETH zürich

THEORY

- **Half-space Range Searching**

  - Agarwal et. al. [PODS '98], Matousek et. al. [Computational Geometry '92]

- **Linear Constraint Queries**

  - Goldstein et. al. [PODS '97]

MACHINE LEARNING

- **Nearest Neighbor Queries**

  - Liu et. al. [ICML '12], Jain et. al. [NIPS '10]

DATABASES

- **Top-k Queries with Ranking Function**

  - Chang et. al. [SIGMOD '00], Xin et. al. [SIGMOD '07], Li et. al. [SIGMOD '05], Ilyas et. al. [ACM Comp. Survey '08], Hristidis et. al. [SIGMOD '01], Ram et. al. [KDD '12]

- **Index for Moving Objects**

  - Nascimento et. al. [R-tree, SAC '98], Sistla et. al. [ICDE '97], Kollios et. al. [PODS '99], Saltenis et. al. [TPR-Tree, SIGMOD '00], Jensen et. al. [B^x-Tree, VLDB '04], Tao et. al. [SIGMOD '02], Zhang et. al. [MBR Tree, VLDB J '12]

Related Work

ETH zürich

THEORY

- **Half-space Range Searching**

  - Agarwal et. al. [PODS '98], Matousek et. al. [Computation

- **Linear Constraint Queries**

  - Goldstein et. al. [PODS '97]

MACHINE LEARNING

- **Nearest Nei**

  - Liu et

DATABASES

- **T**

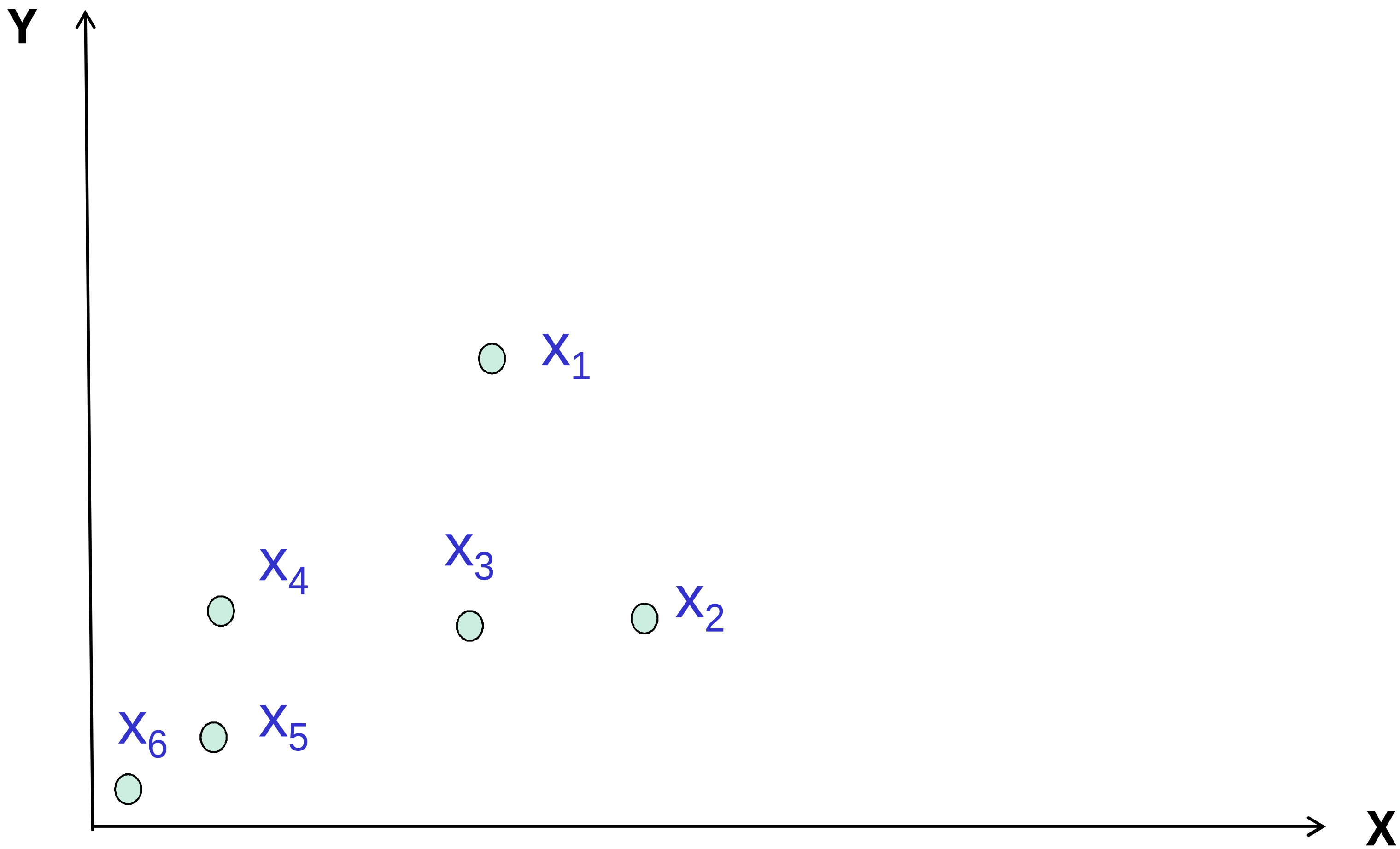
  - Cha], Ilyas
  et. al. [A al. [KDD '12]

- **Index**

  - Nascimen et. al. [ICDE '97], Kollios et. al.
  [PODS '99], Sa , SIGMOD '00], Jensen et. al. [B^x-Tree, VLDB
  '04], Tao et. al.], Zhang et. al. [MBR Tree, VLDB J '12]

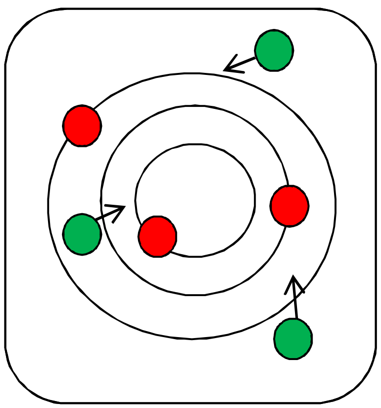
One Index for many problems ☺

Planar Index: Geometrical Indexing



Moving Objects Intersection Finding

ETH zürich



Moving Object Database

→ r, ω

→ p, q, u, v

- Find all object pairs that will be within distance S at time instance t

$$\mathbf{A}X_1 + \mathbf{B}X_2 + \mathbf{C}X_3 + \mathbf{D}X_4 + \mathbf{E}X_5 + \mathbf{F}X_6 + \mathbf{G}X_7 \leq \mathbf{S^2}$$

Scalar Product Query: $(A\ B\ C\ D\ E\ F\ G) \cdot (X_1\ X_2\ X_3\ X_4\ X_5\ X_6\ X_7) \leq S^2$

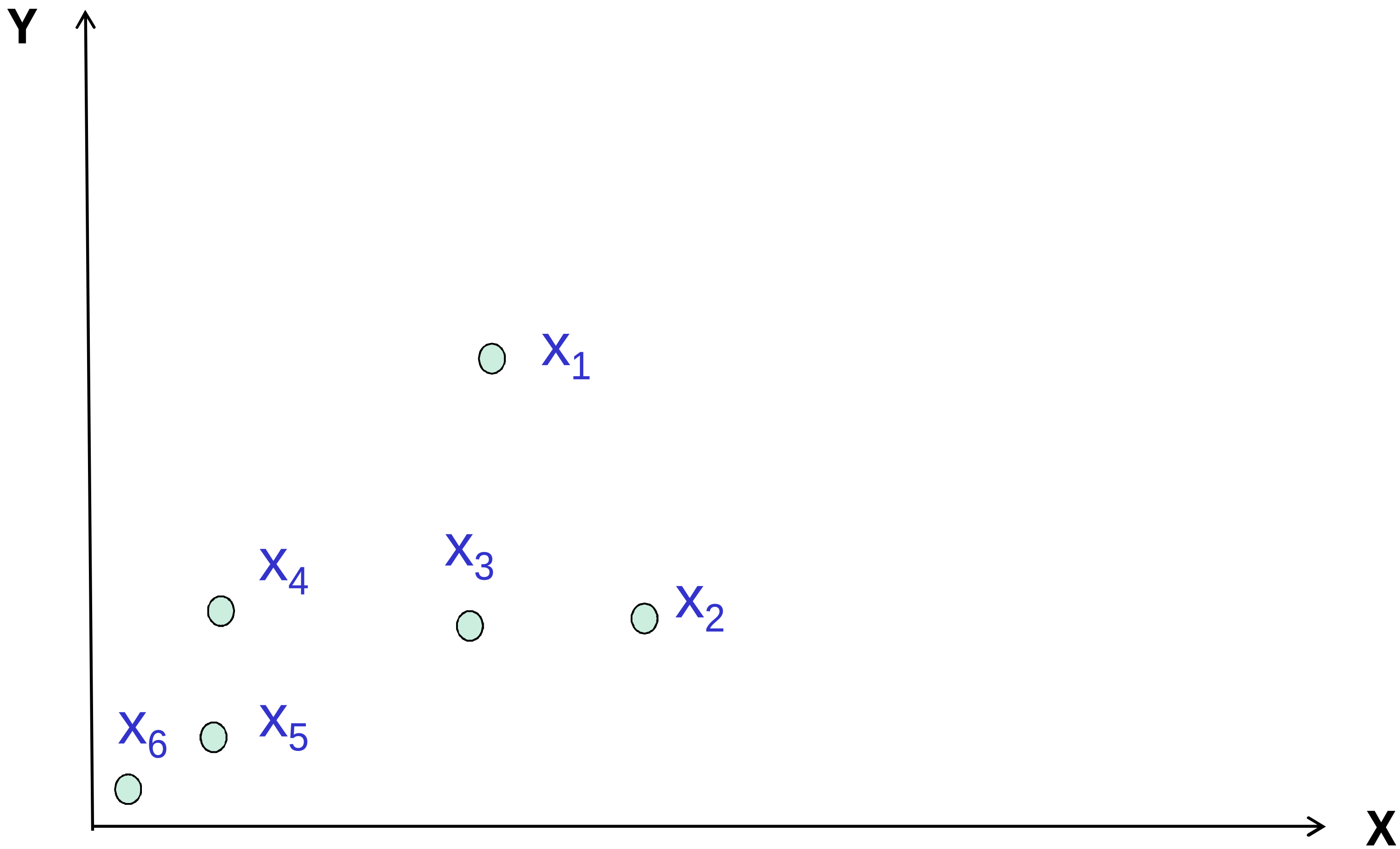
Query Parameters (unknown)

Function (known)

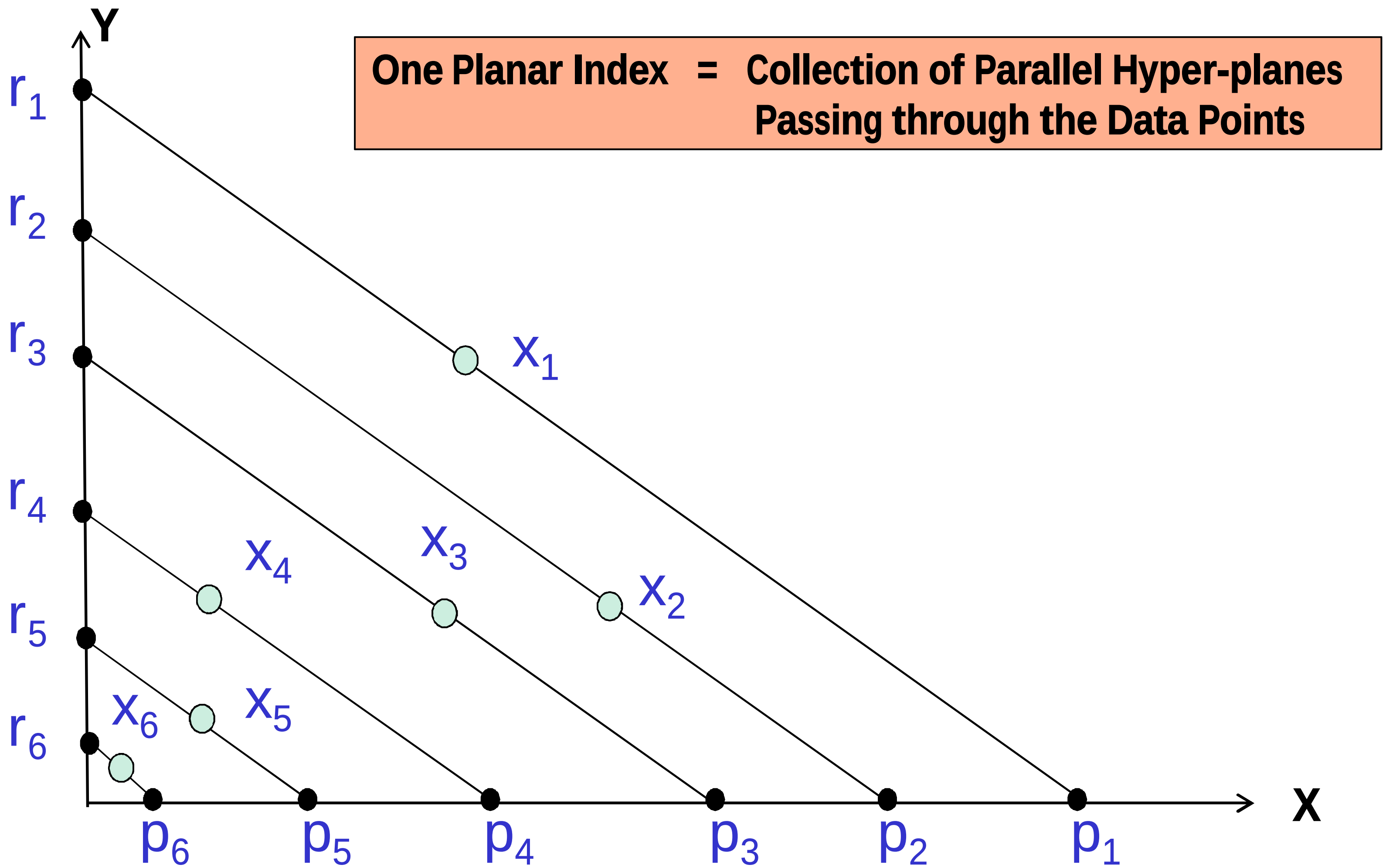
Inequality Parameter (unknown)

Planar Index: Geometrical Indexing

ETH zürich



Planar Index: Geometrical Indexing

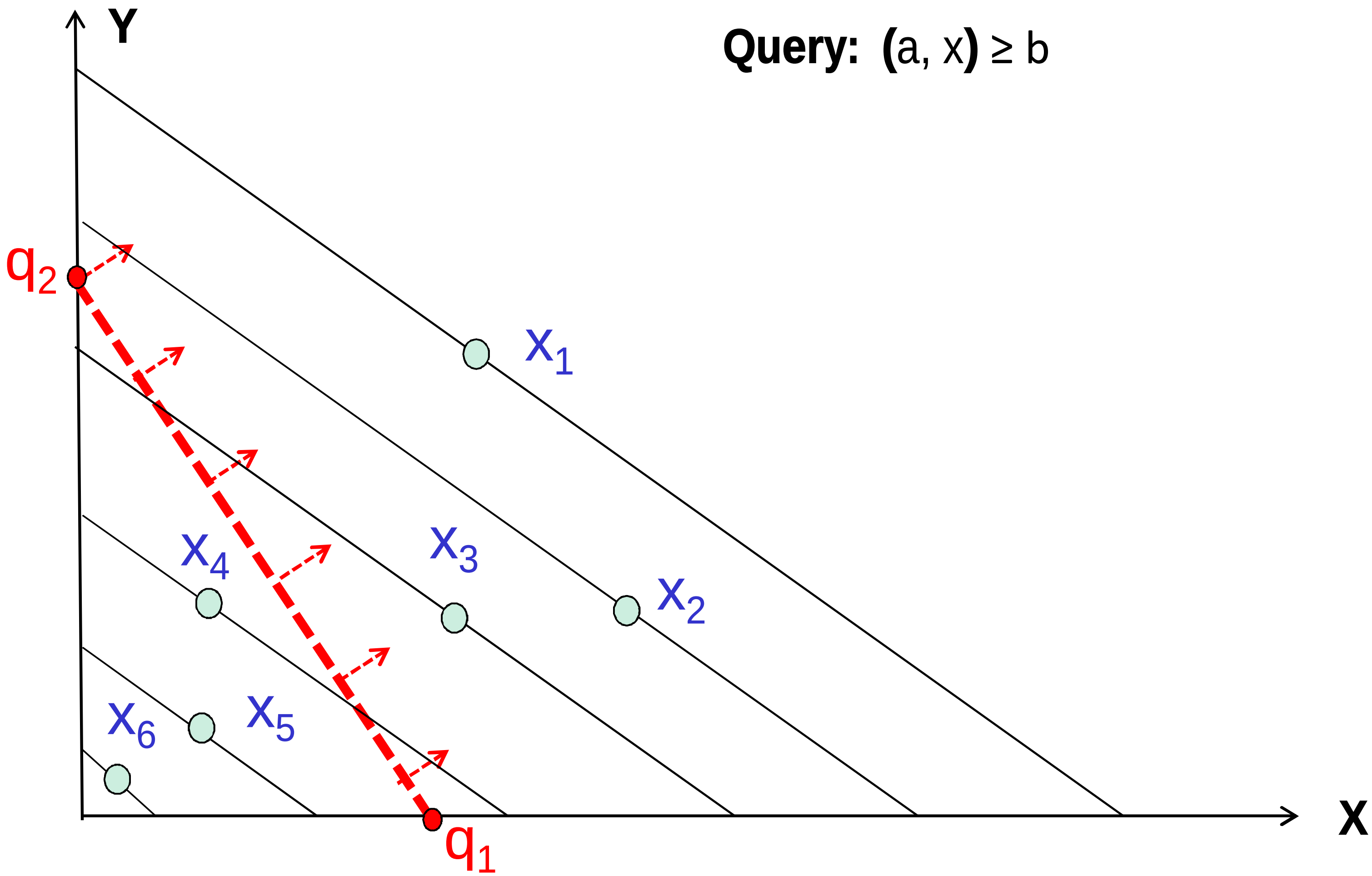


Query Processing using Planar Index

Planar Index: Geometrical Indexing

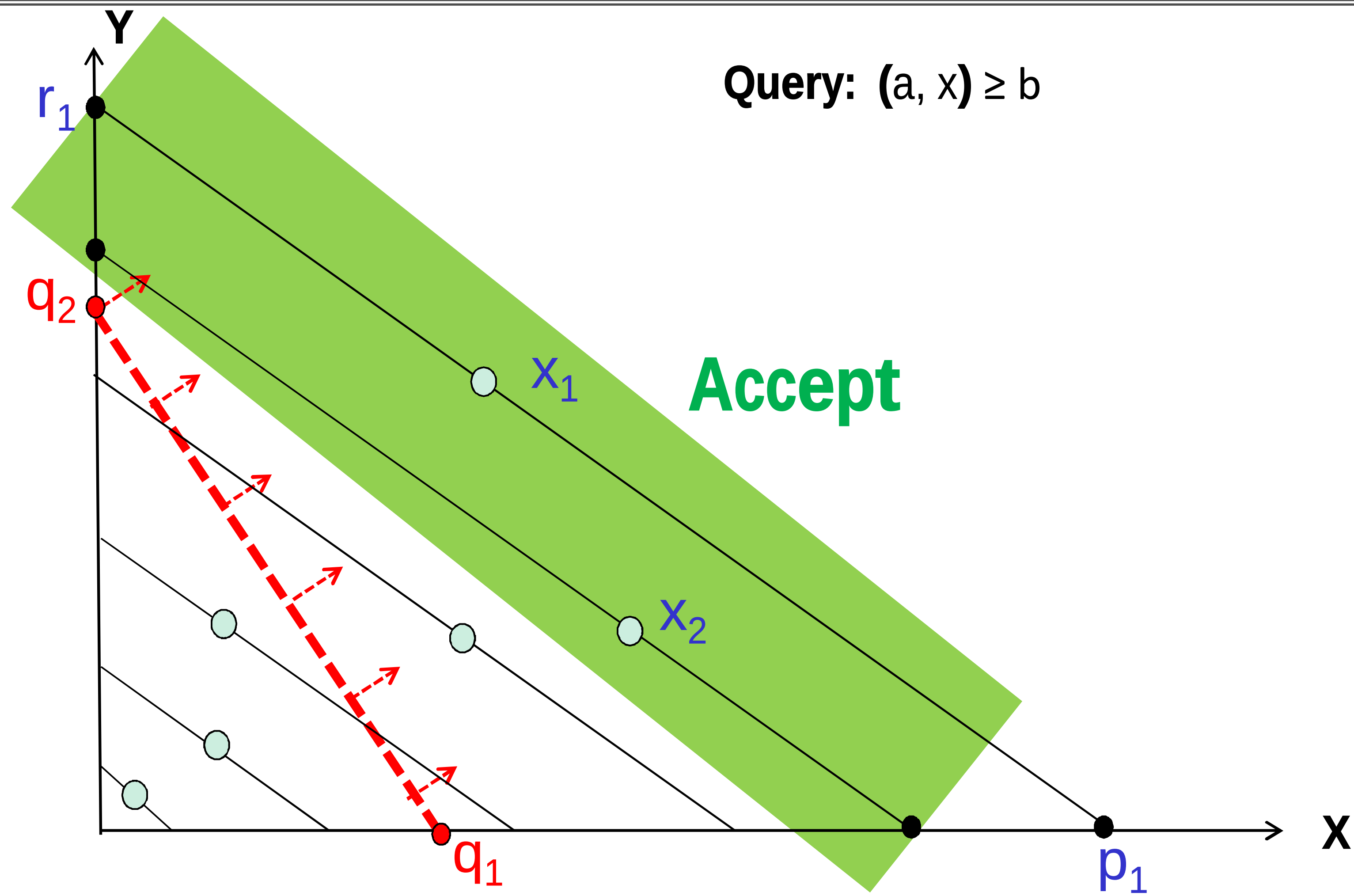
ETHzürich

Query: **(**a, x**)** ≥ b



Query Processing using Planar Index

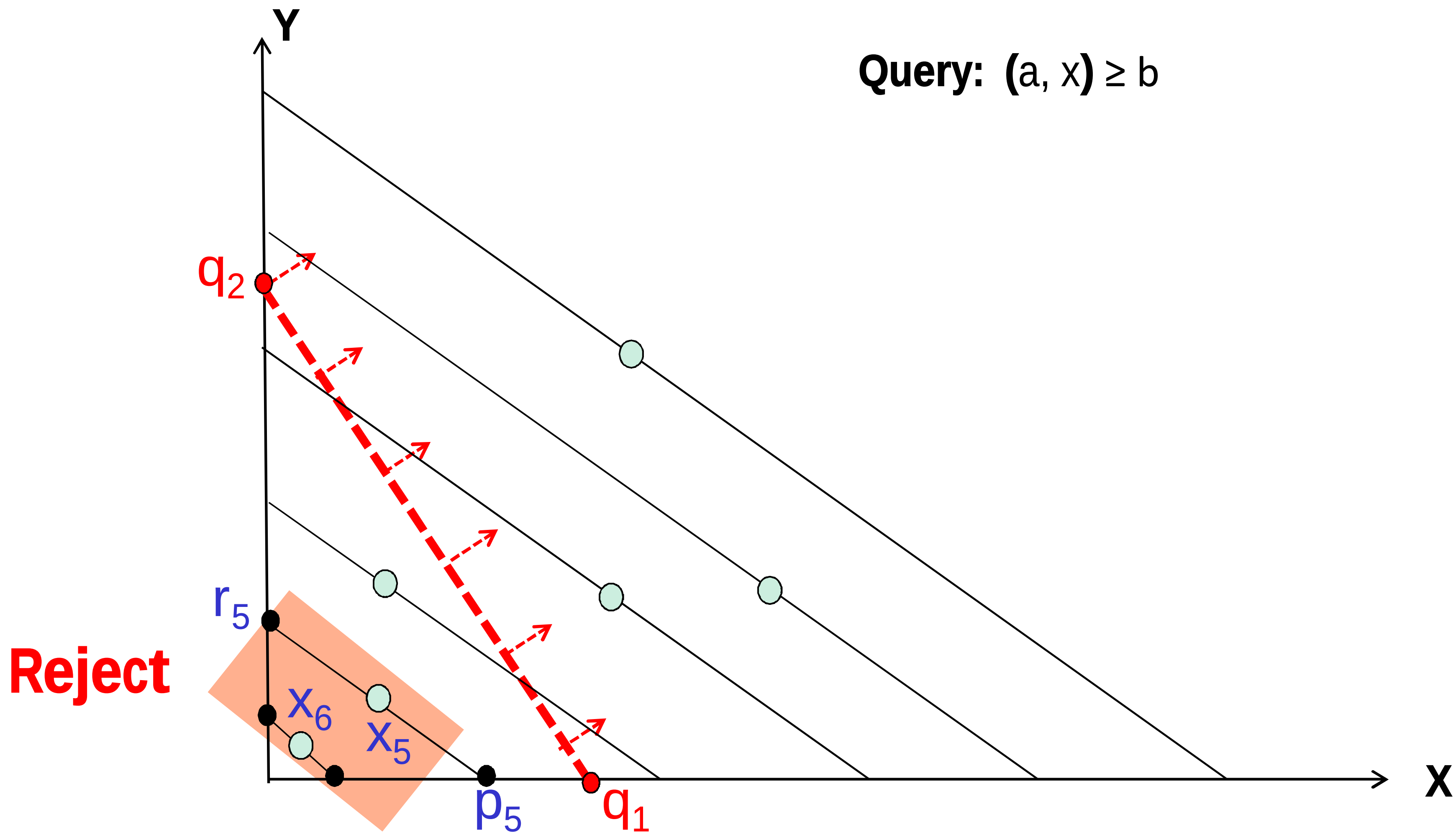
Planar Index: Geometrical Indexing



Query Processing using Planar Index

Planar Index: Geometrical Indexing

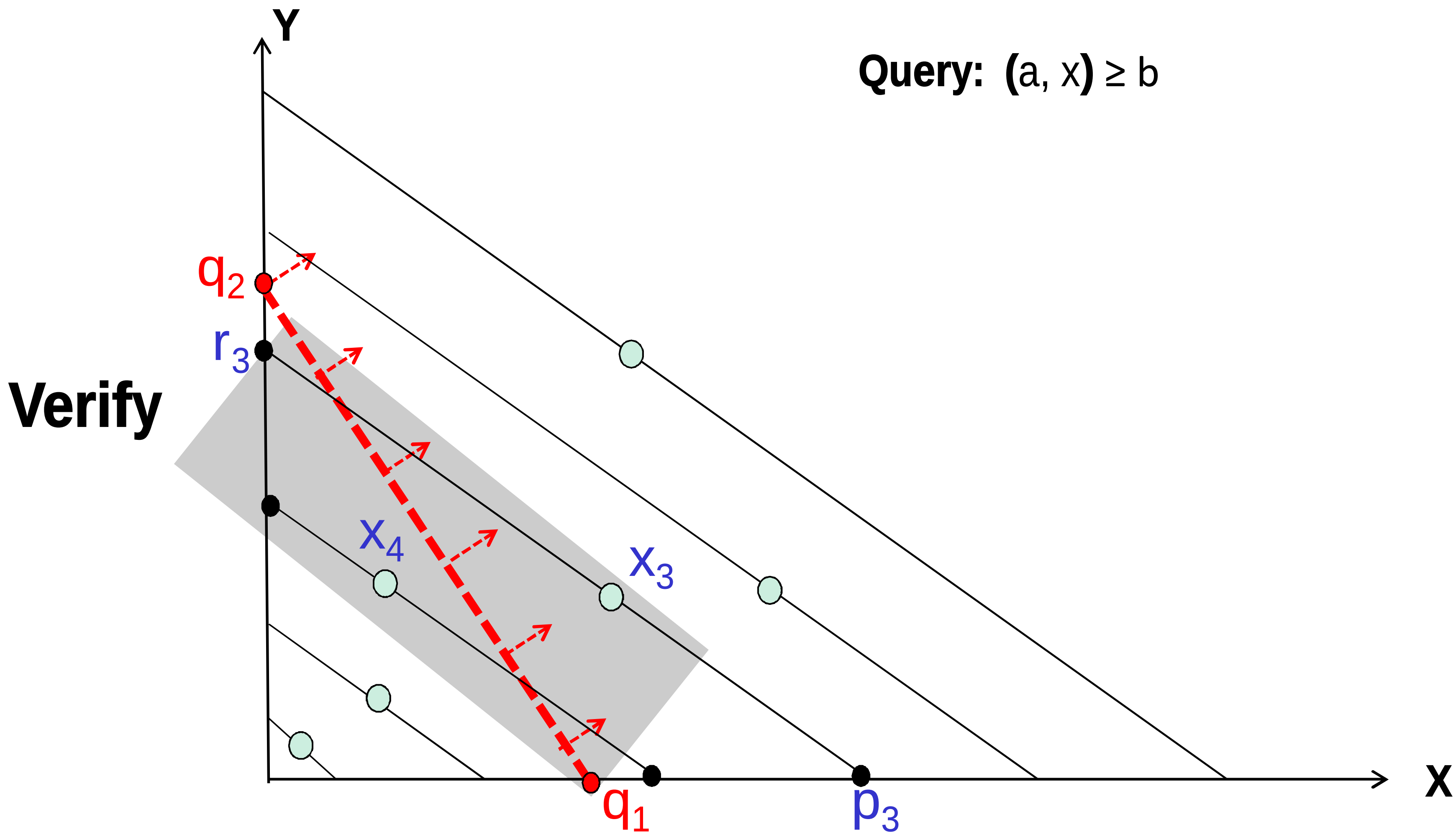
ETHzürich



Query Processing using Planar Index

Planar Index: Geometrical Indexing

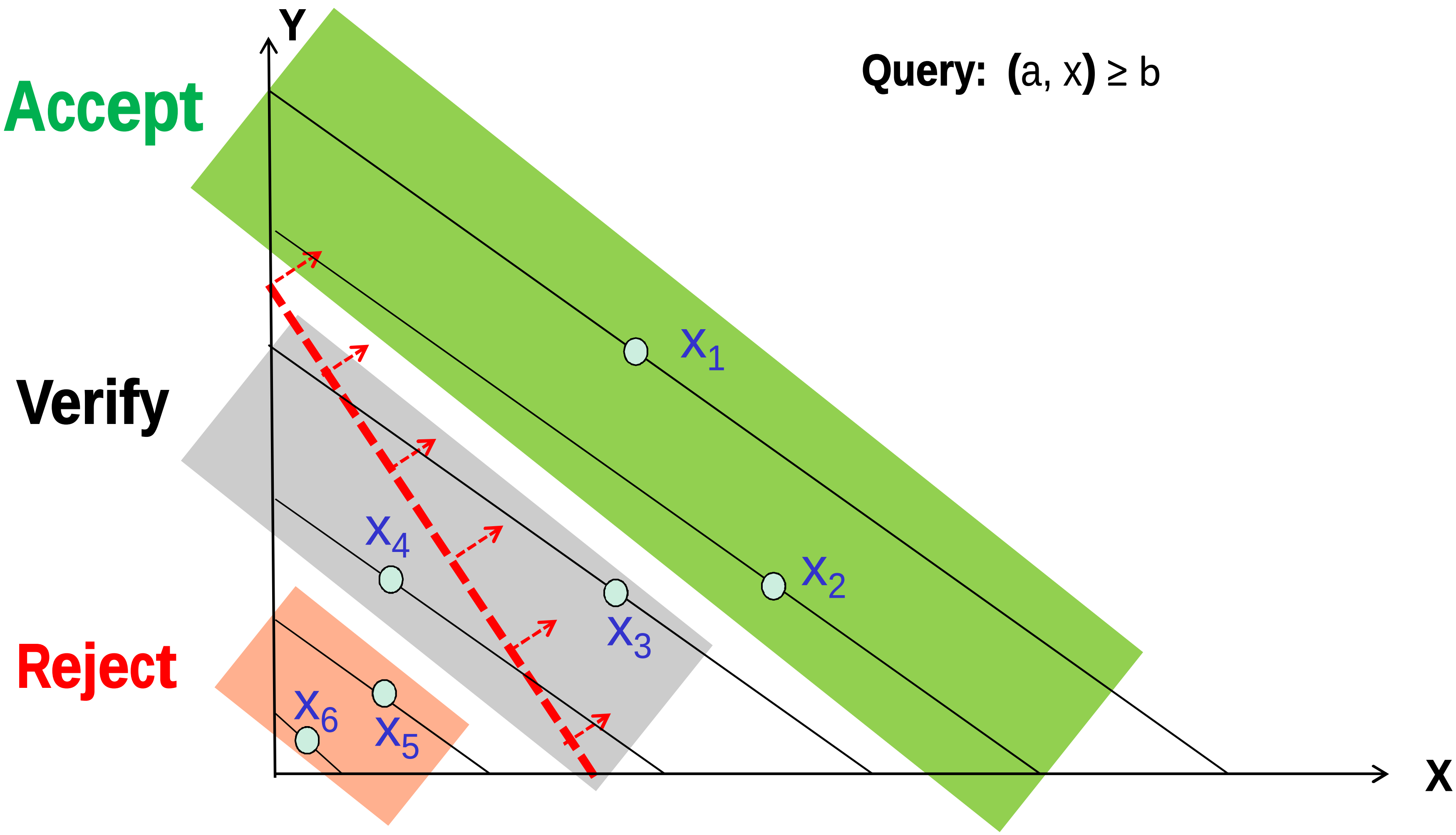
Query: **(**a, x**)** ≥ b



Query Processing using Planar Index

Planar Index: Geometrical Indexing

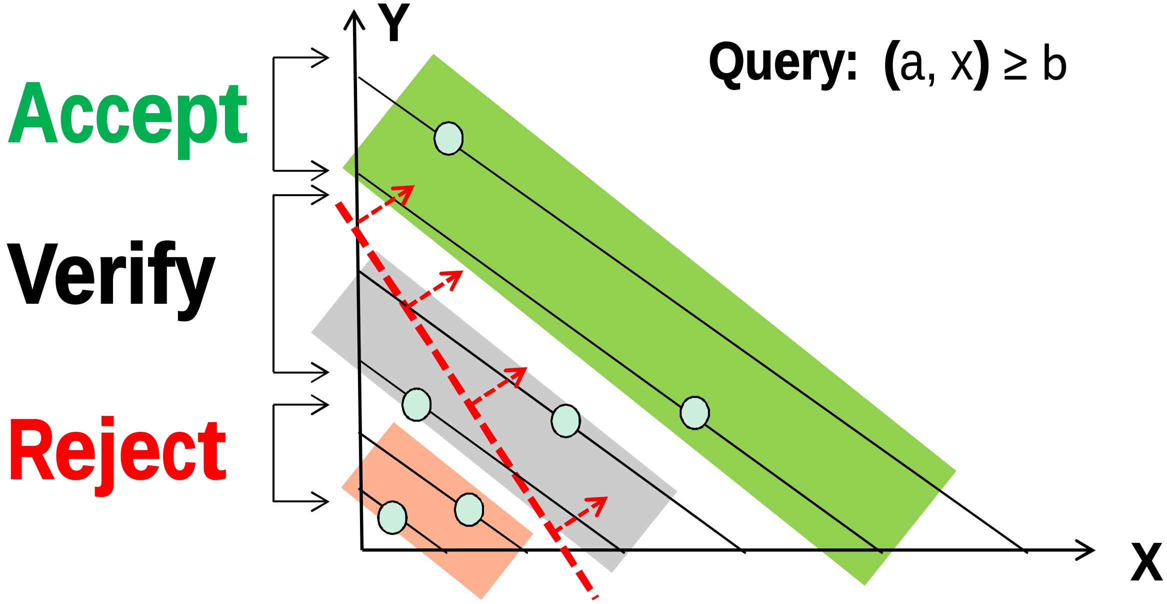
Query: (a, x**)** ≥ b



Query Processing using Planar Index

Planar Index: Time and Space Complexity

ETHzürich



Query Processing using Planar Index

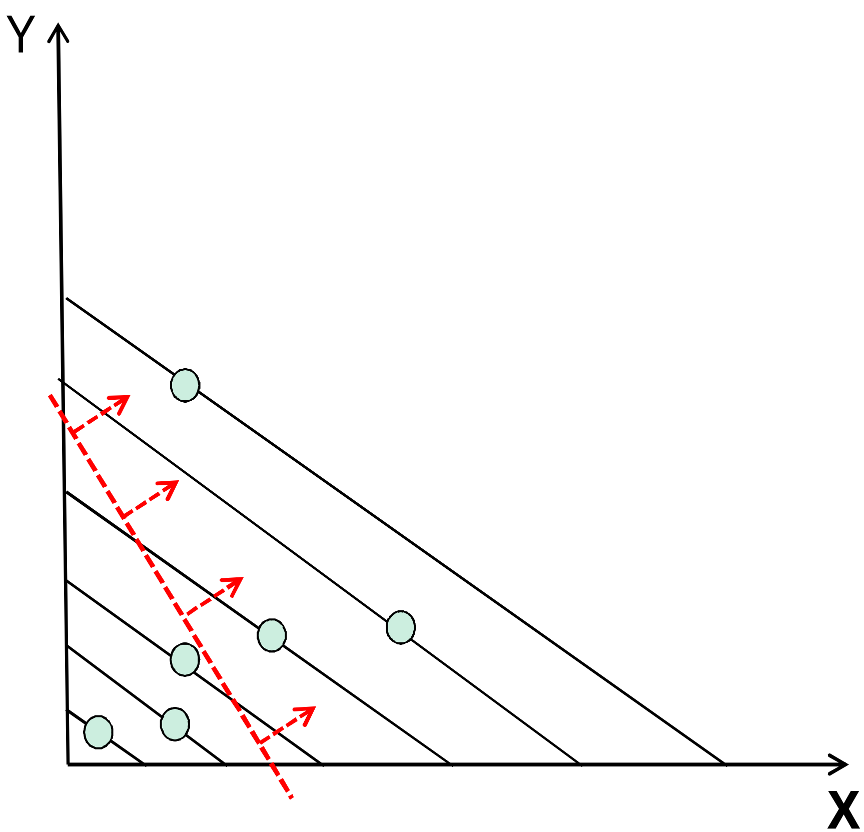
Index Time: O(n log n)

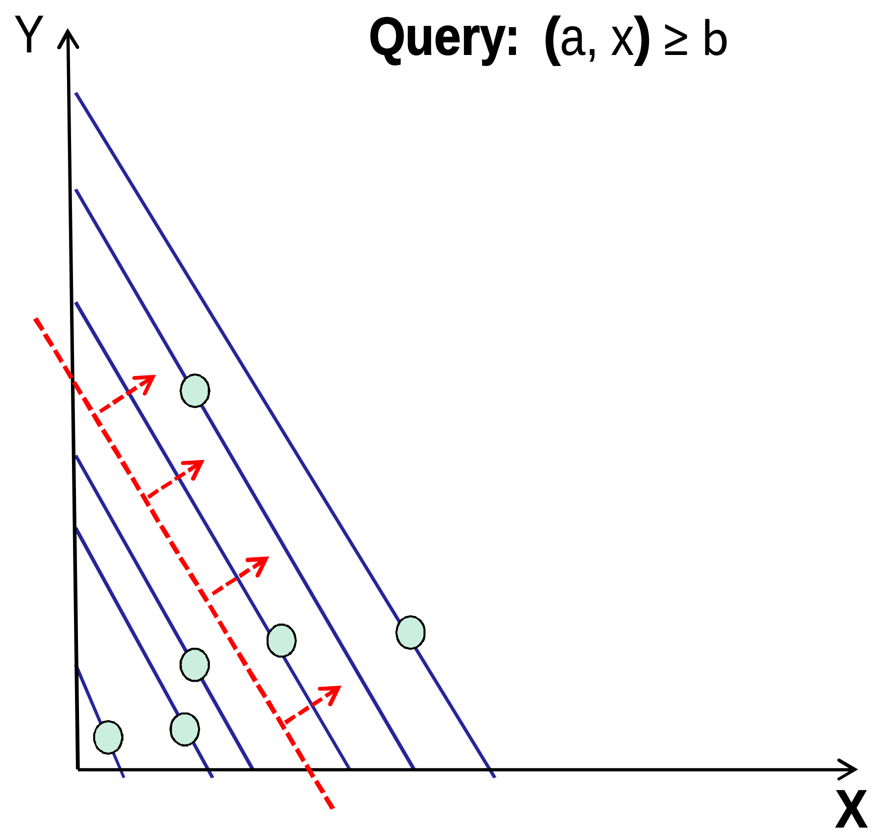
Index Space: O(n)

Query Processing Time: O(d log n + t) ~ O(d n)

Multiple Planar Indices

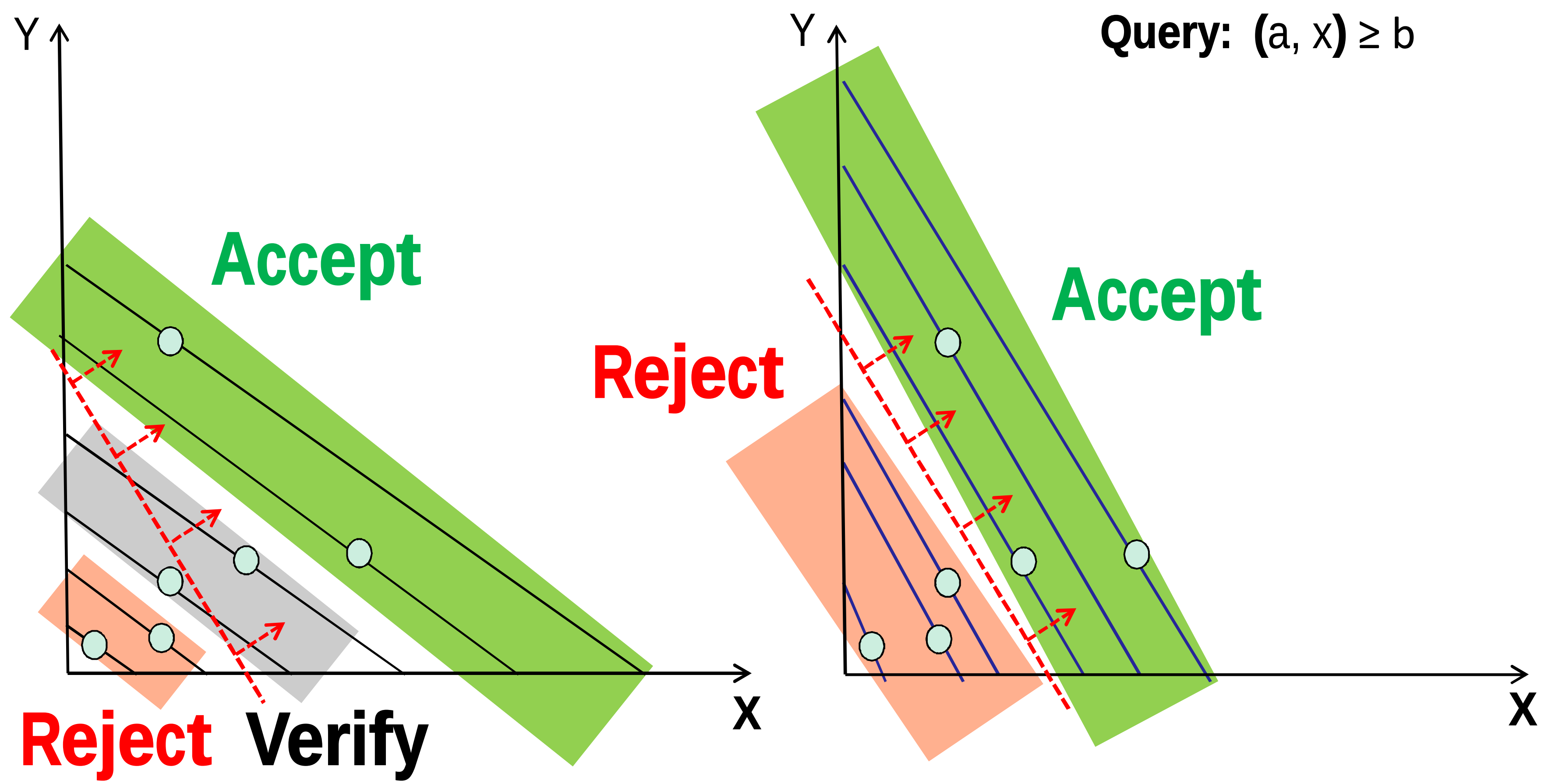
ETH zürich





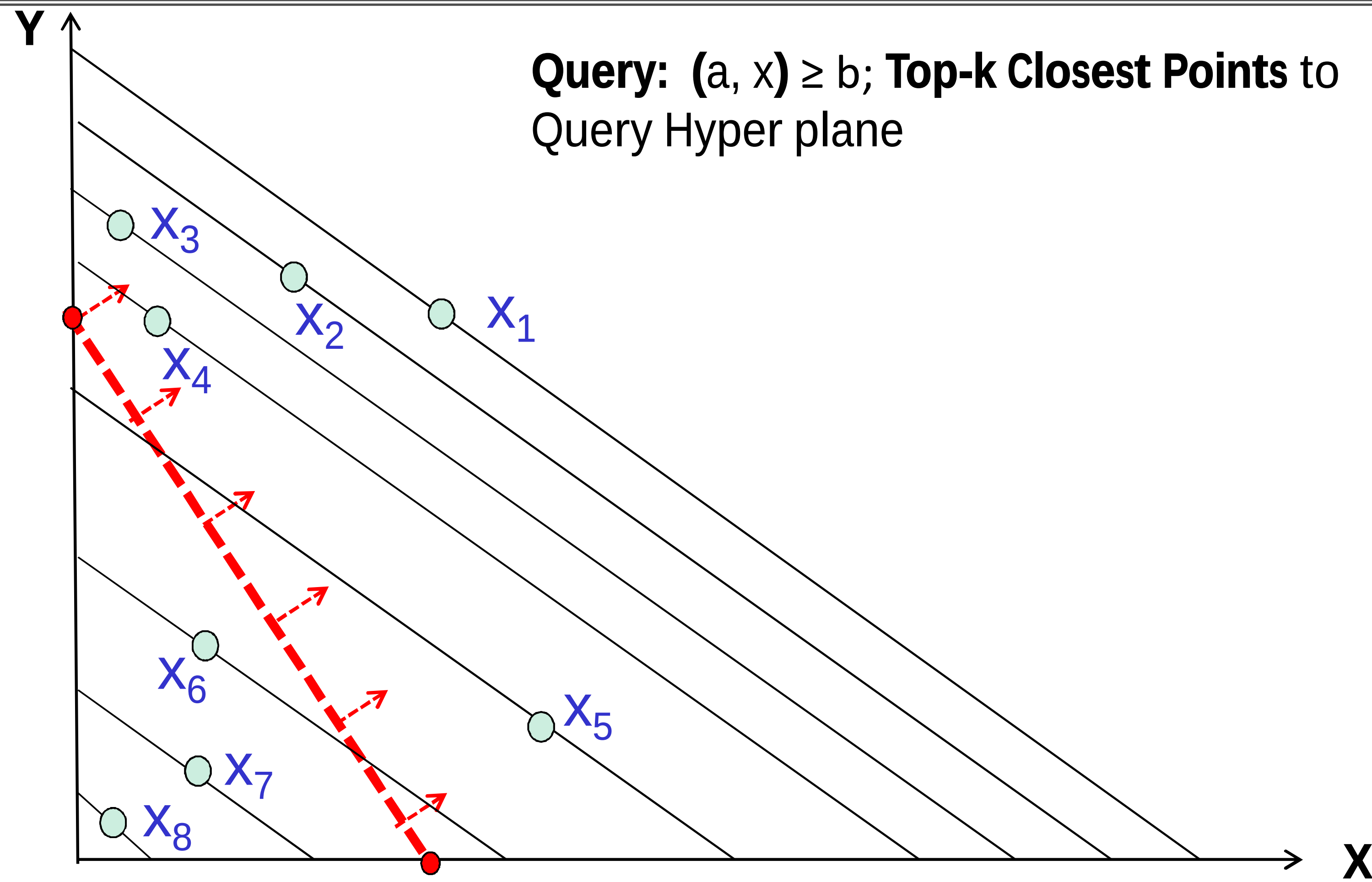
Multiple Planar Indices

Best Index Selection at Query Time



Planar Index at Right is Better for the Given Query

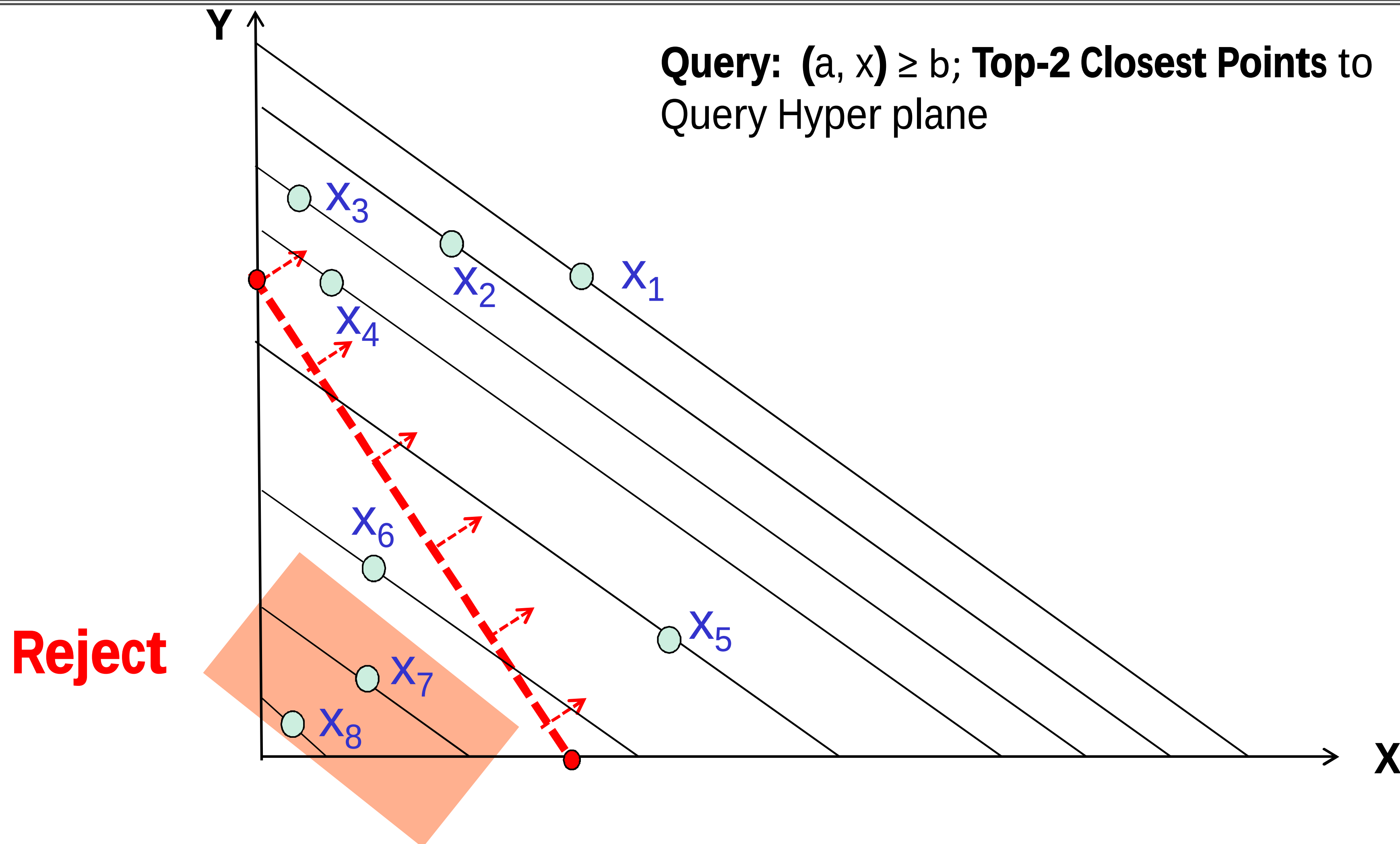
Top-k Nearest Neighbor Query



Top-k Nearest Neighbor Query

Top-k Nearest Neighbor Query

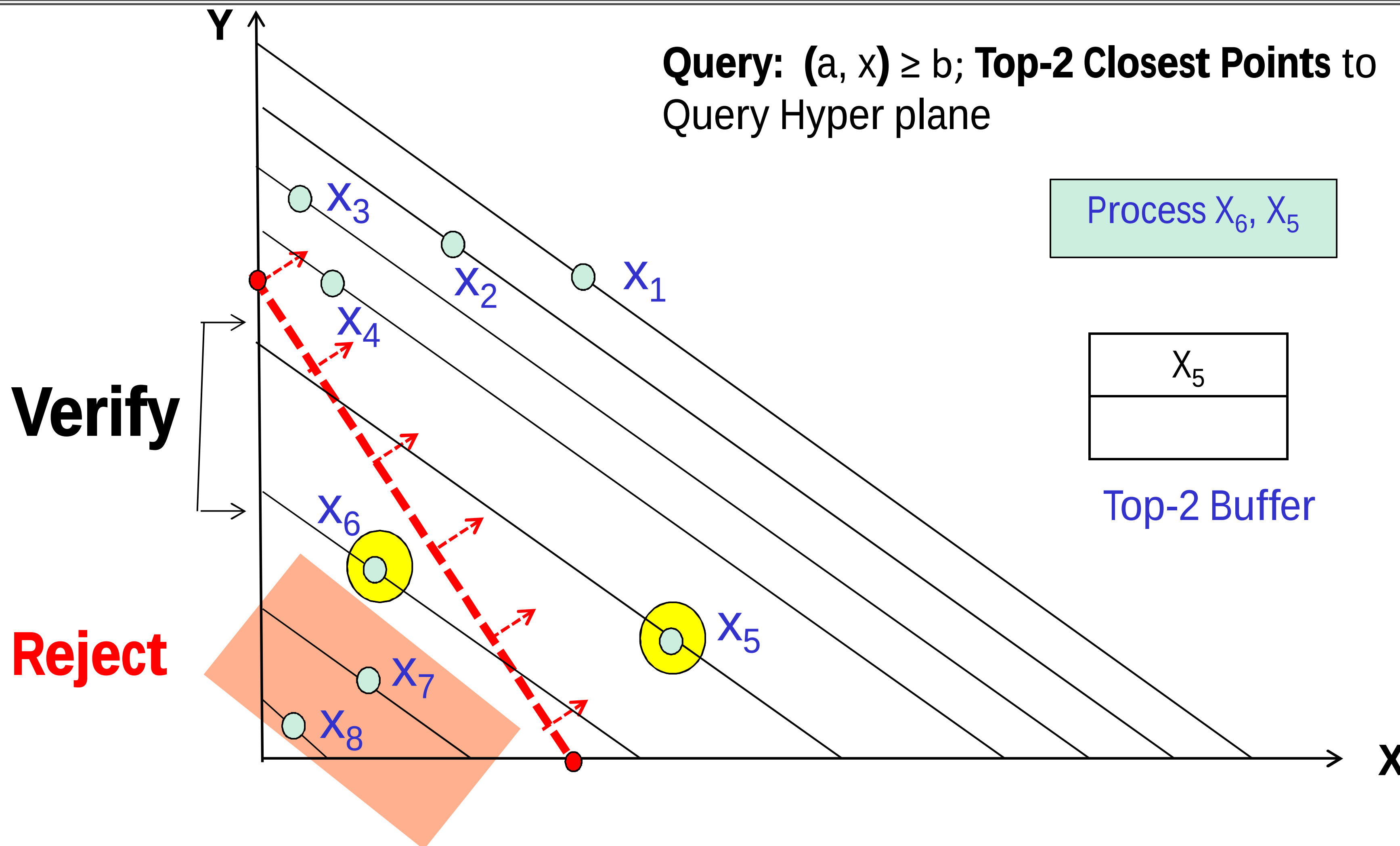
ETH zürich



Top-k Nearest Neighbor Query

Top-k Nearest Neighbor Query

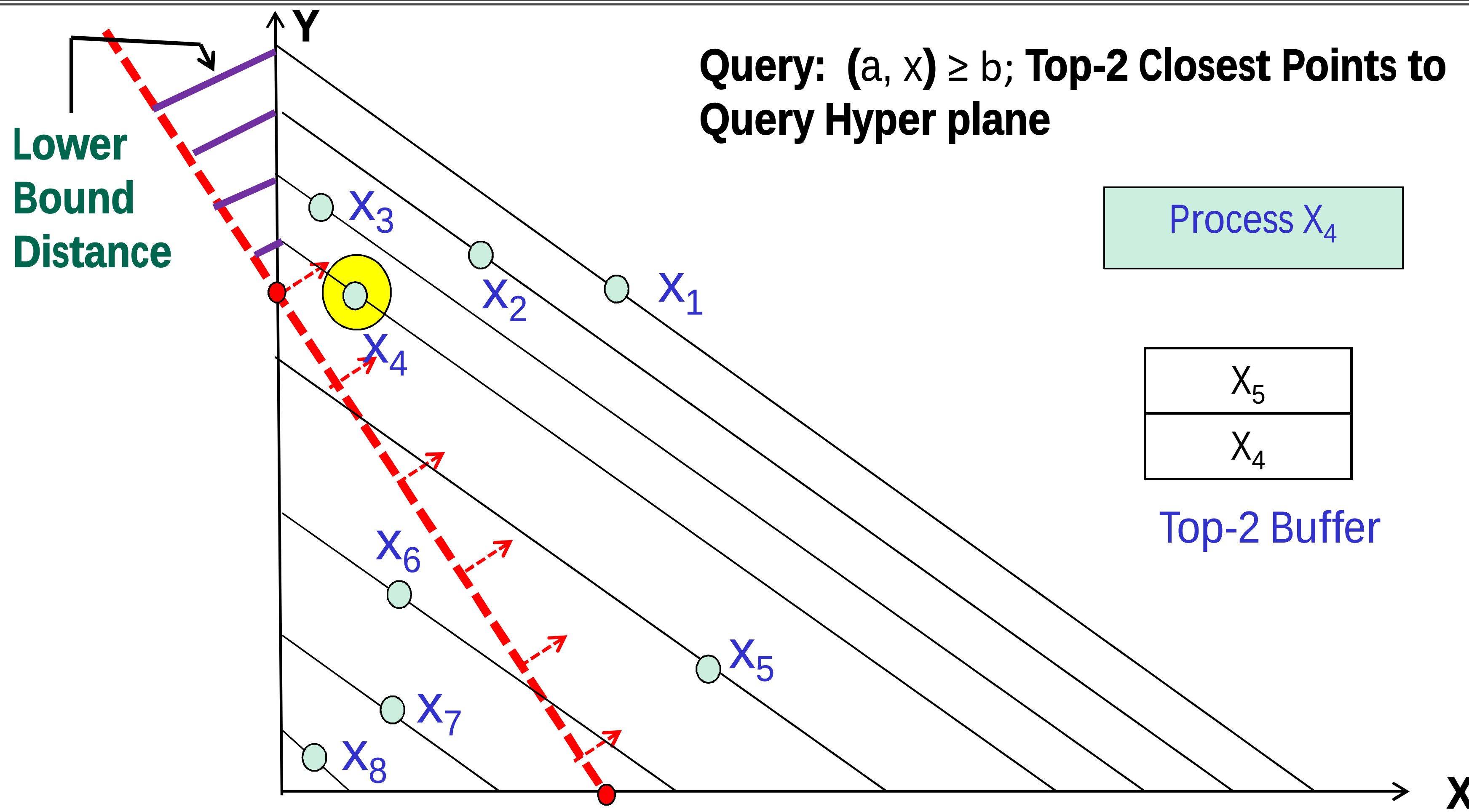
Query: (a, x**)** ≥ b; **Top-2 Closest Points** to Query Hyper plane



Top-k Nearest Neighbor Query

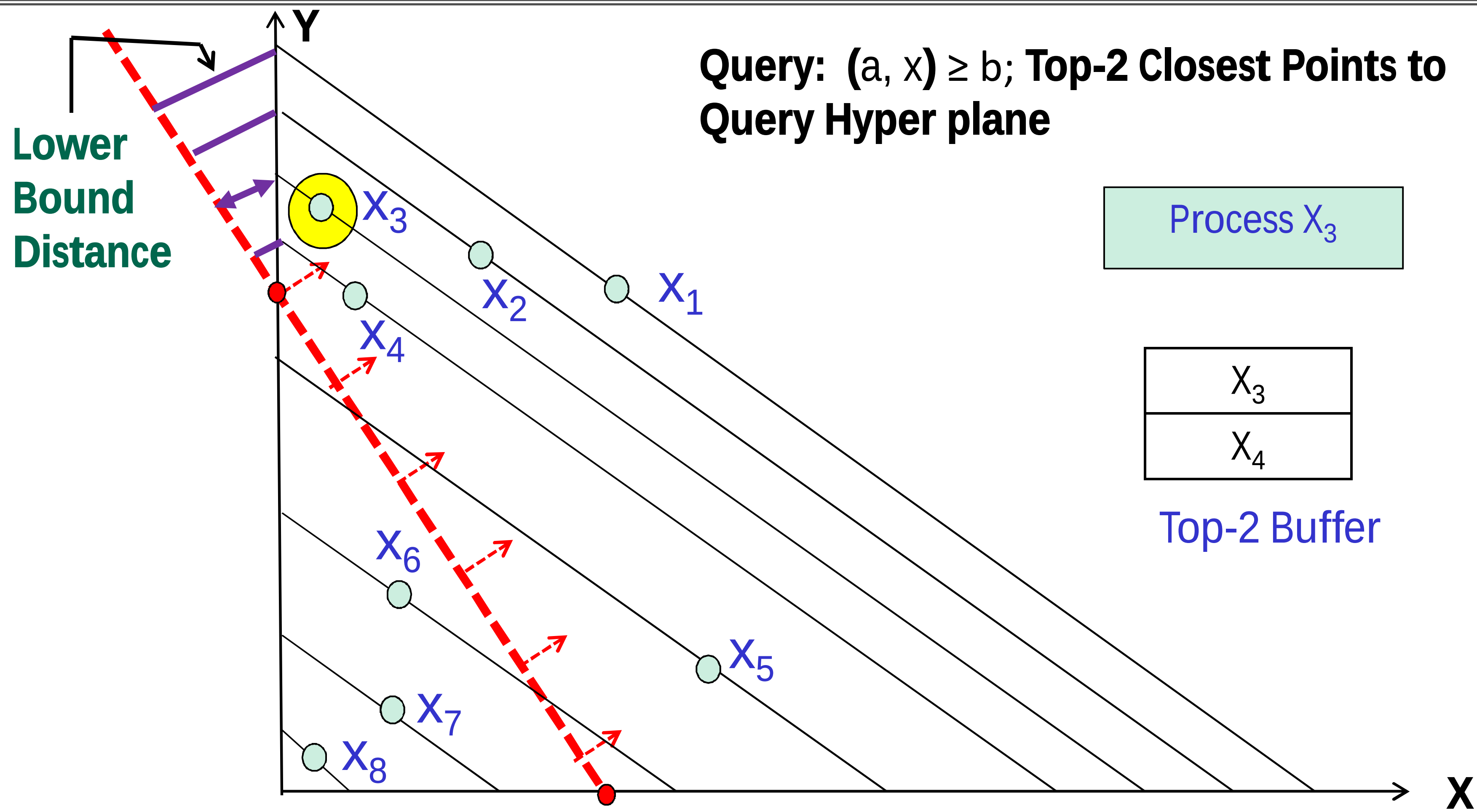
Top-k Nearest Neighbor Query

Query: (a, x**)** ≥ b; **Top-2 Closest Points to Query Hyper plane**



Top-k Nearest Neighbor Query

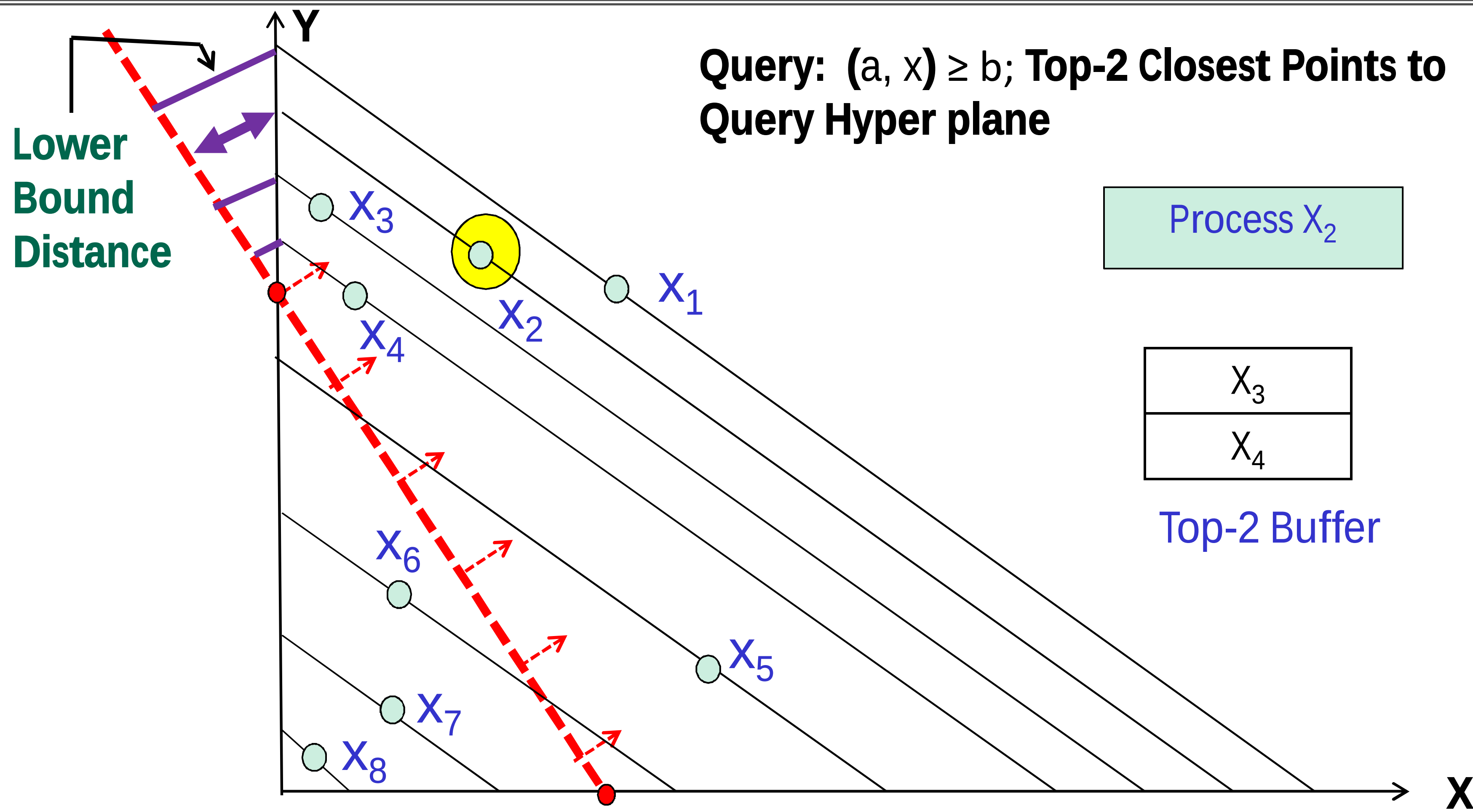
Top-k Nearest Neighbor Query



Top-k Nearest Neighbor Query

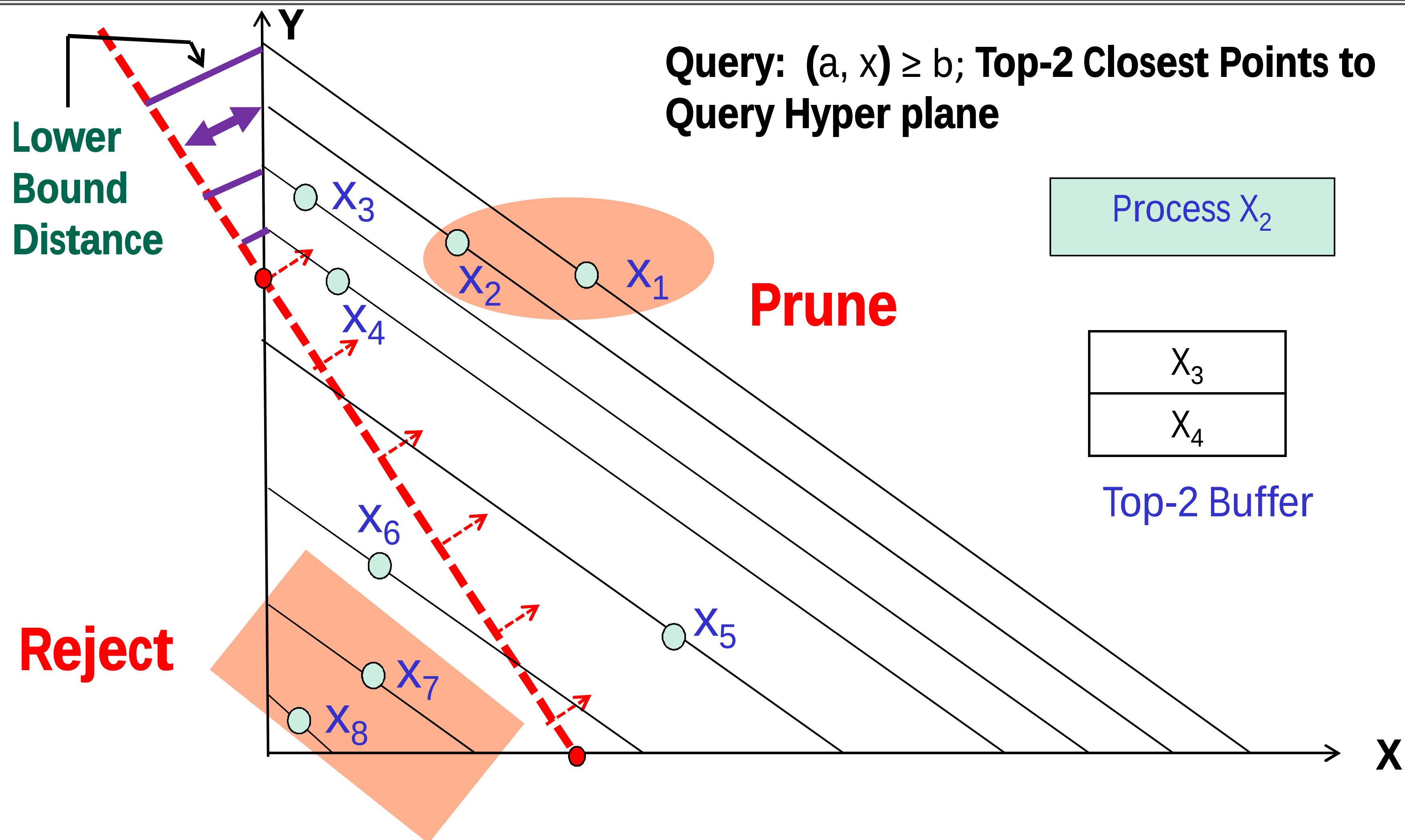
Top-k Nearest Neighbor Query

Query: (a, x**)** ≥ b; **Top-2 Closest Points to Query Hyper plane**



Top-k Nearest Neighbor Query

Top-k Nearest Neighbor Query



Top-k Nearest Neighbor Query

List of Experiments

ETH zürich

- **Datasets:**
  - **Real-World:** CMoment, Ctexture, Electricity Consumption
  - **Synthetic:** Independent, Correlated, Anti-Correlated

- **List of Experiments:**
  - Efficiency vs. No of Index
  - Efficiency vs. No of Dimension
  - Efficiency vs. Randomness of Query
  - Efficiency vs. Query Selectivity
  - Pruning Capacity vs. No of Index
  - Pruning Capacity vs. No of Dimension
  - Pruning Capacity vs. Randomness of Query
  - Pruning Capacity vs. Query Selectivity
  - Scalability of Index Building, Query Processing
  - Dynamic Index Updating
  - Memory Usage of Planar Index

Experimentally Evaluated Planar Index in:

- Moving-Object Intersection
- Top-k Nearest Neighbor Query

Dataset and Query

ETH zürich

- **Datasets:**

	# Data Points	# Dimension	# Attribute Range
CMoment (Real-World)	68,040	9	$(-4.15, 4.59)$
Independent (Synthetic)	1,000,000	2 - 14	(1, 100)

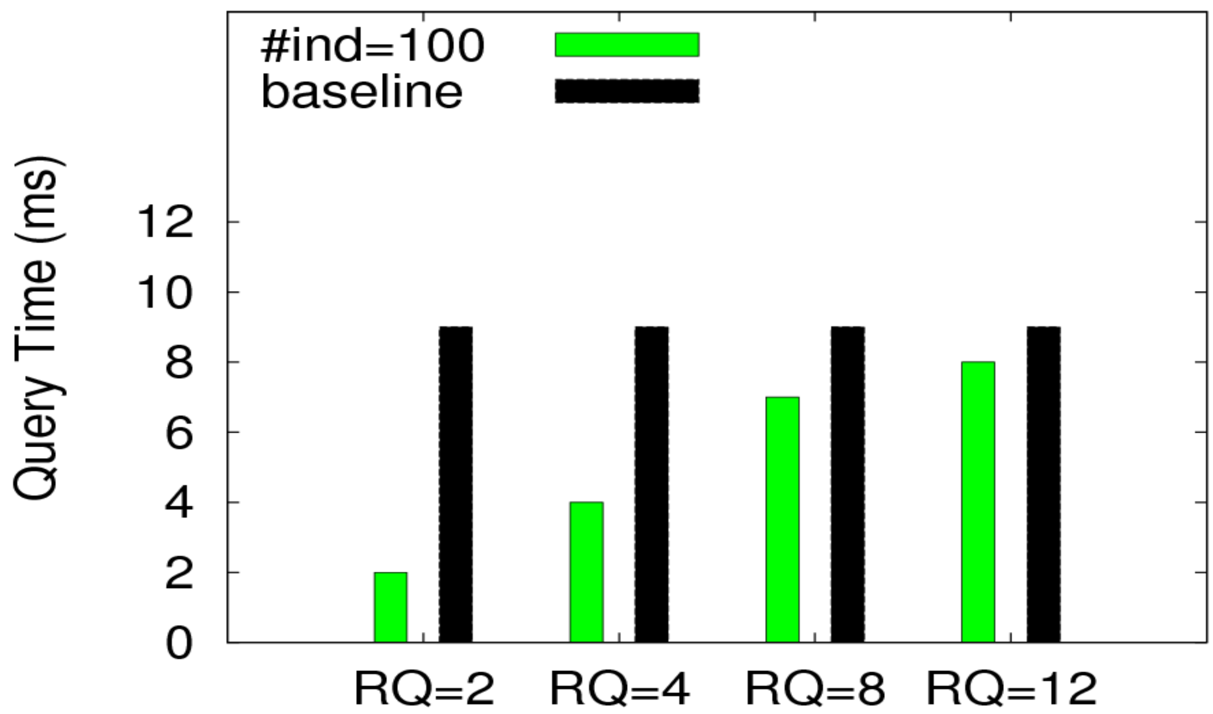
- **Query:**

$$Q_1 X_1 + Q_2 X_2 + \ldots + Q_d X_d \geq \underline{75}\,(Q_1 + Q_2 + \ldots + Q_d)$$

Query Selectivity

- **Randomness of Query (QR):** $Q_i \in (1,n)$

Efficiency (Real-World Dataset)



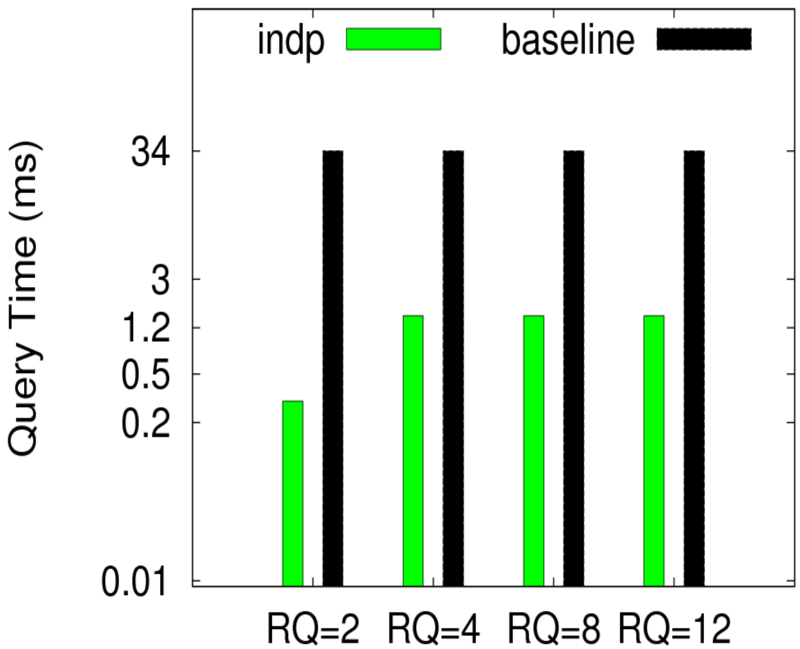
Dimension = 9

Index = 100

1.13 ~ 4.5 times better than Baseline

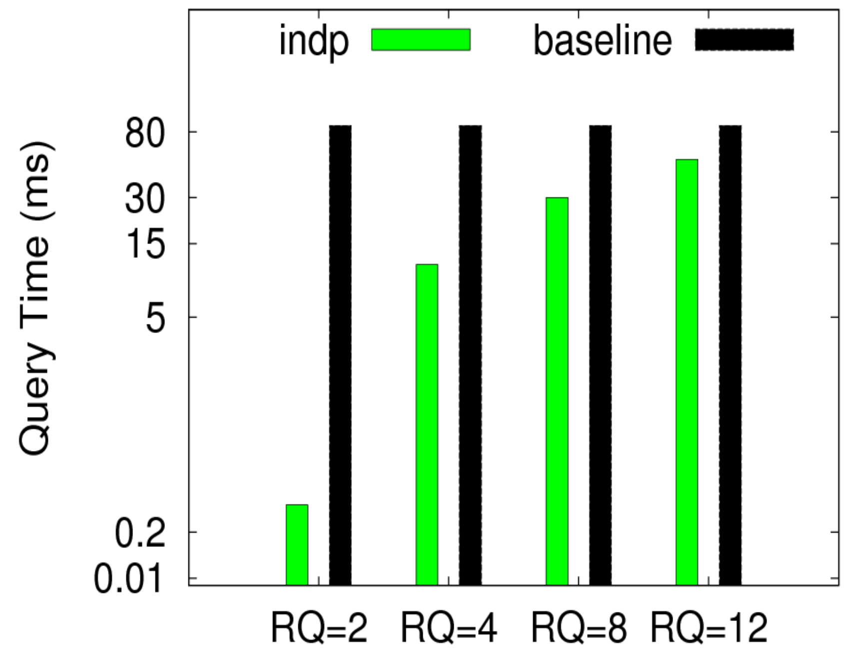
Efficiency (Synthetic Dataset)

Index = 100



Dimension = 2

12 ~ 170 times better than Baseline



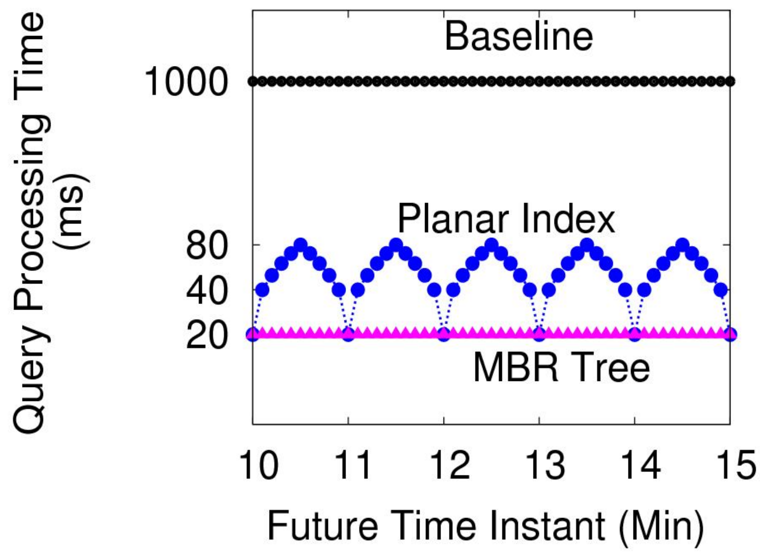
Dimension = 6

1.6 ~ 400 times better than Baseline

Application: Moving Object Intersection

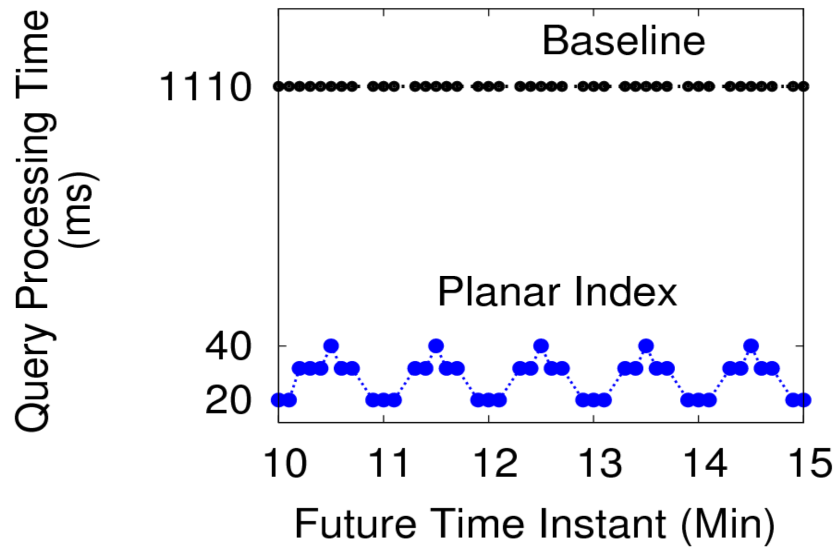
ETH zürich

Intersection Finding among 5K × 5K Moving Objects



Objects moving with uniform velocity

12 ~ 50 times better than Baseline



Objects moving with acceleration

27 ~ 55 times better than Baseline

Conclusion

ETH zürich

- Scalar product query widely applicable
- Planar index – one generalized index for many problems
- Application in moving object intersection finding
- **Future Work:** Dynamic updates in planar indices based on past query workload

Software and Dataset: (Publicly Available) http://people.inf.ethz.ch/khana/software/scalar.tar.gz