On Authenticated Query Processing via Untrusted Service Providers

Jie Wu

Department of Computer and Information Sciences
Temple University

Trustcom 2018

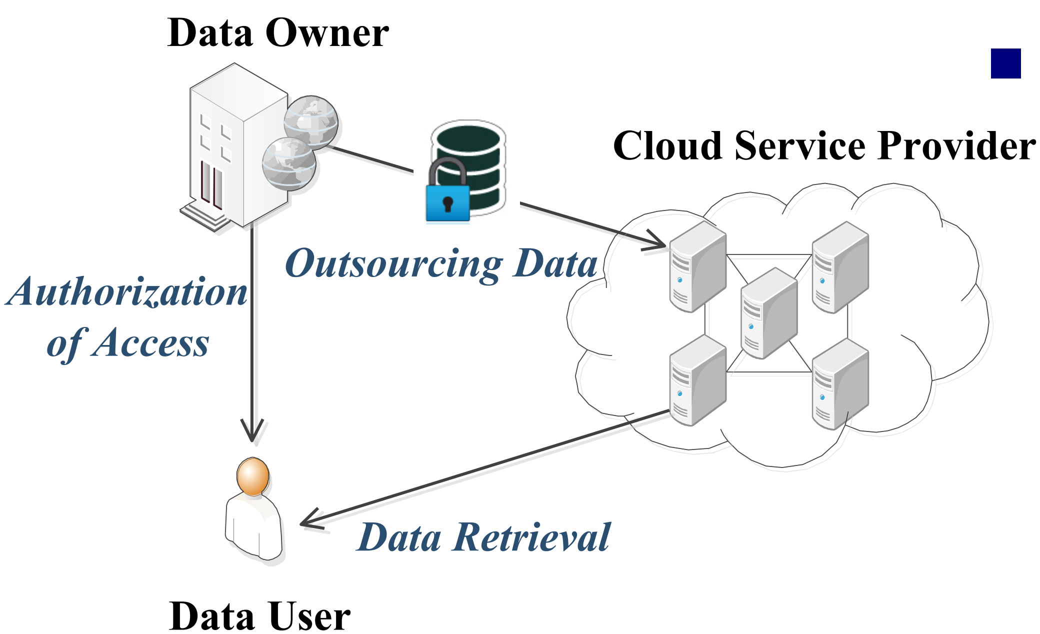
Road Map

1. Era of Big Data
   - Trusting vs. suspicious
2. Search Privacy
   - Individual search using homomorphic encryption
   - Collaborative search using trusted third party
3. Function Query
   - Function query based on ranks
   - Verification object (VO)
4. Conclusions
5. Sample On-going Projects

1. Era of Big Data

90% of world's data was generated over last two years!

More and more personal data is stored in the cloud



- Do you trust CSP?
  - Yes: selected a trusted CSP
  - No: Apply searchable encryption at CSP

Trusting vs. Suspicious

Facebook process 500 terabytes (10^{12}) of data daily, yet…

Revealed: 50 million Facebook profiles harvested for Cambridge Analytica in major data breach

Whistleblower describes how firm linked to former Trump adviser Steve Bannon compiled user data to target American voters

- 'I made Steve Bannon's psychological warfare tool': meet the data war whistleblower
- Mark Zuckerberg breaks silence on Cambridge Analytica

To be trusting is to be fooled from time to time.

To be suspicious is to live in torment.

Trusting: Trust (vs. Reputation)

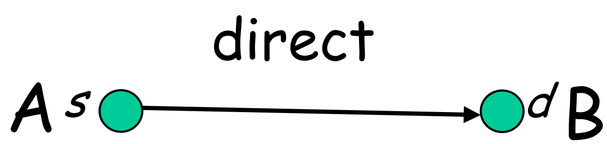
- Reputation (objective)
  - What is generally said or believed about somebody (say B)

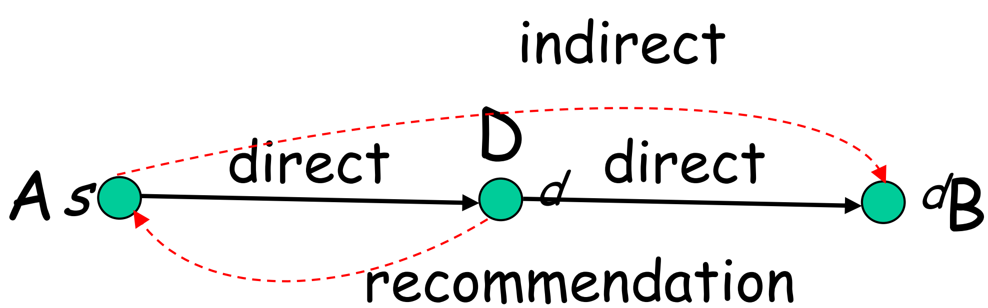
- Trust (subjective: judgment + opinion)
  - Trust is the subjective probability by which A expects that another B performs a given action

- Trust in multiple disciplines [Computing Survey'16]
  - Economics, sociology, psychology, biology, political science, computer science, social networks …
  - Computational (e.g. reliability model) vs. non-computational

How to Build Trust?

- First-hand (direct) and second-hand (recommendation)





- E.g. New department chair's trust management

Suspicious: Search Neutrality

Search engines: no editorial policies other than that their results be comprehensive, impartial and based solely on relevance.

How Google's search algorithm spreads false information with a rightwing bias

Search and autocomplete algorithms prioritize sites with rightwing bias, and far-right groups trick it to boost propaganda and misinformation in search rankings

A fake ranking of a hospital unit under Baidu costed life and money of a patient in China

媒体评魏泽西事件：天堂里没有莆田系和百度

摘要：4月13日，陕西咸阳，告别仪式过后，魏则西的遗体被推去火化，父母坐在殡仪馆外等候。魏则西的死，捅破了百度医疗竞价排名、莆田系承包科室现象、医疗监管漏洞等诸多医疗乱象的窗户纸。这家位于北京二环边上的三甲医院，曾像“救命稻草”一样被魏则西和父母紧紧握在手中。

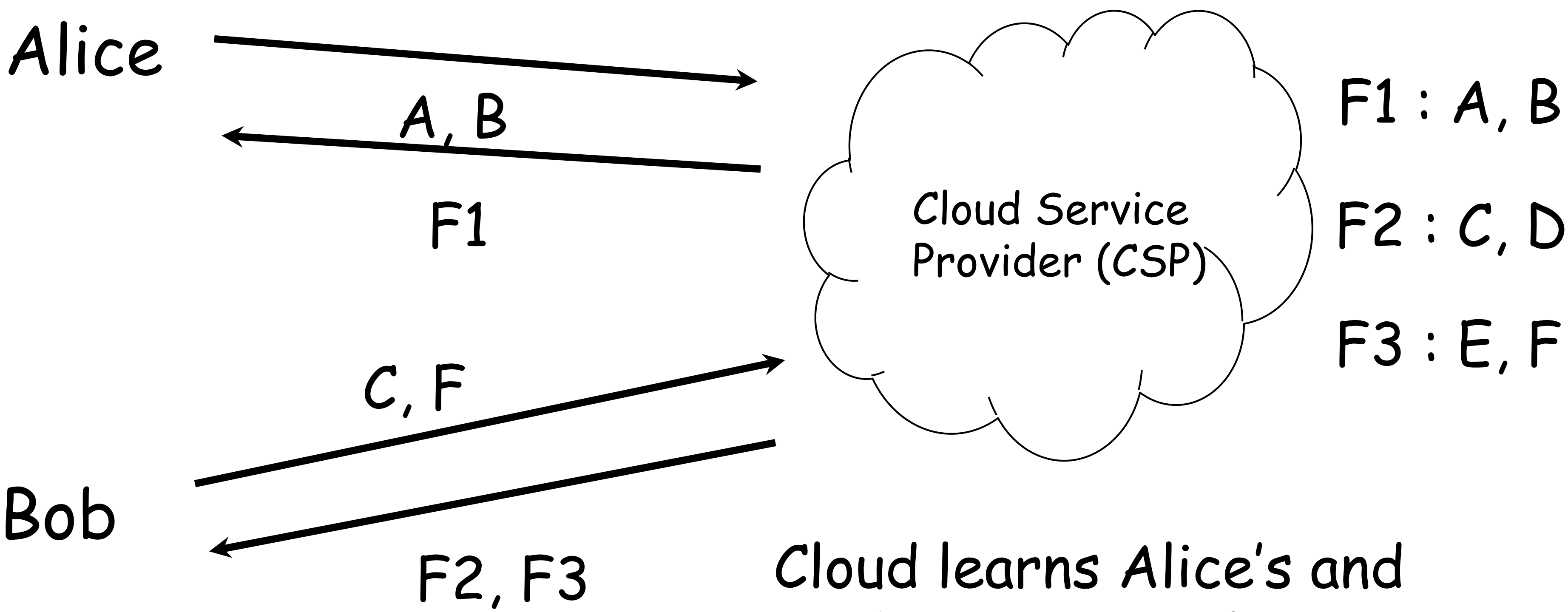
4月13日，陕西咸阳，告别仪式过后，魏则西的遗体被推去火化，父母坐在殡仪馆外等候。

Security and Privacy

- Data security
  - Tampered data, data loss or replacement, data leakage, …
- Query authentication
  - Sound and complete
- How to protect user privacy ….
  - defending against data privacy and search privacy.
- … while ensuring good system performance.
  - conserving bandwidth, reducing energy through minimizing computation and communication.

2. Search Privacy

Users querying the cloud (forward index file: key list)



Cloud learns Alice's and Bob's queries and responses.

Search Privacy

Cloud neither learns what the user is searching for, nor which files are returned to a user.

Cloud: semi-trusted (i.e., honest but curious).

It will obey general rules, but still wants to know some additional information.

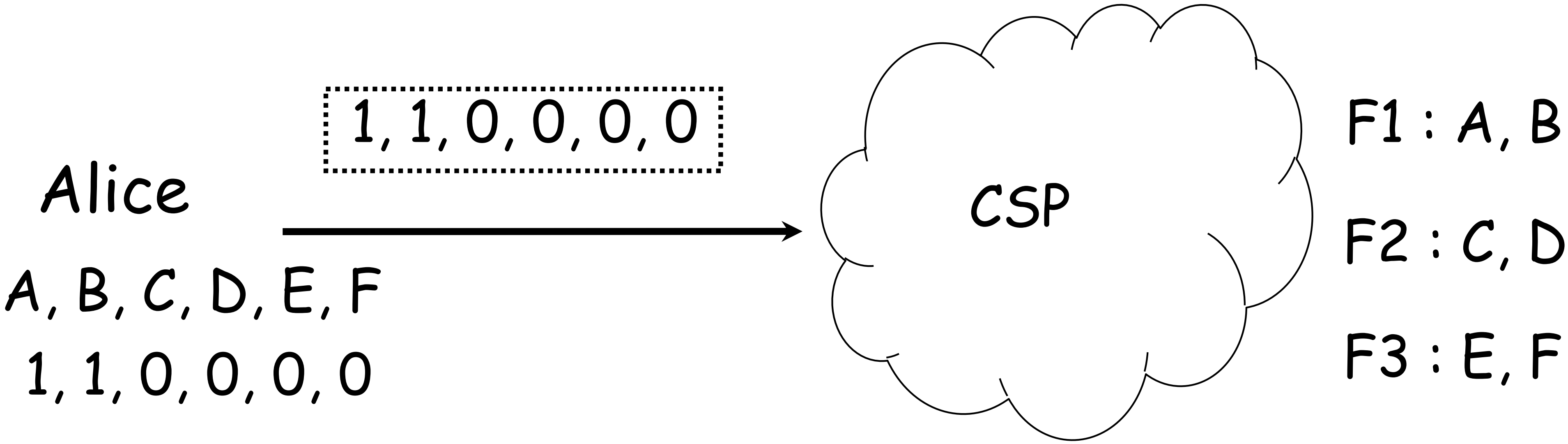
Searchable Encryption

- A user builds index (forward/inverted) for a collection of files.

- A user utilizes symmetric/asymmetric key encryption and searchable encryption (SE) to encrypt file contents and indexes, respectively.

- Later, a user generates a trapdoor (an encrypted query with SE) to retrieve all the files containing keyword w and performs decryption locally.

Prior Solution [CRYPTO'05]



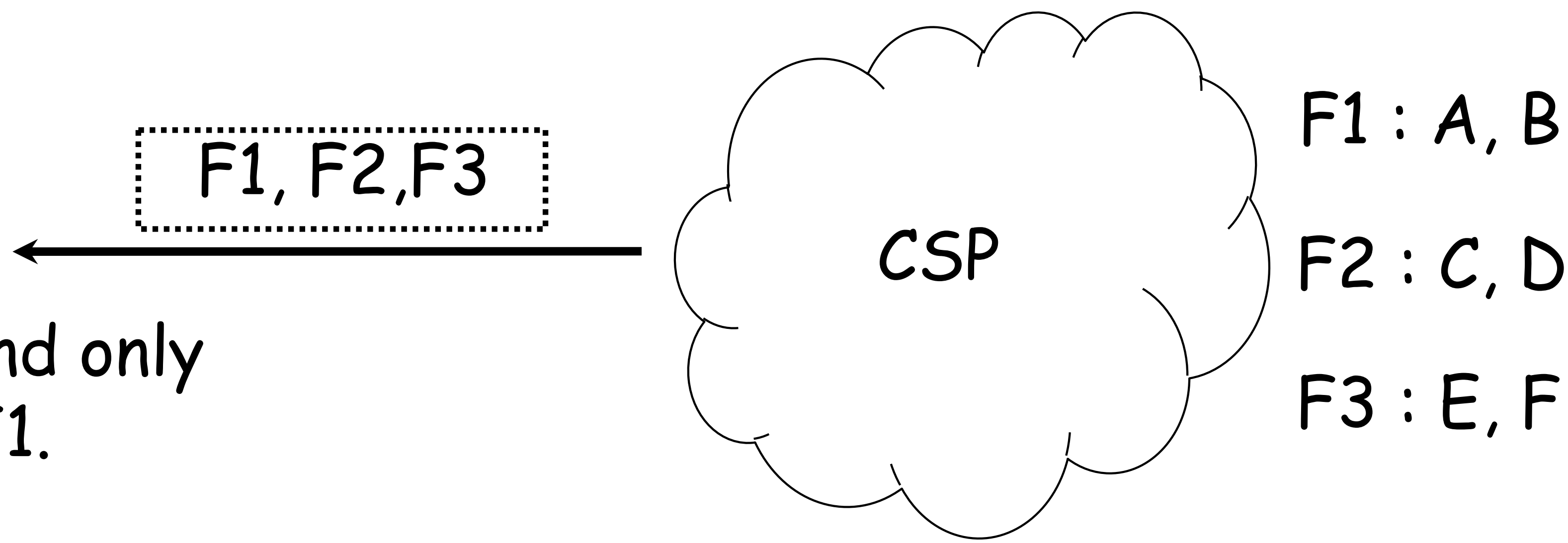
Alice issues a special query 1, 1, 0, 0, 0, 0

with a dictionary encrypted using homomorphic encryption

Cloud compares all files and returns all.

Prior Solution (cont'd)

Alice



Decrypts and only gets back F1.

Cloud computes and returns to Alice $\boxed{\text{F1, F2,F3}}$

$\boxed{\text{F1, F2,F3}}$ is a compressed version of F1,F2,F3.

Cloud does not know what files are returned.

Brief Background

Homomorphic encryption allows us to perform some operations on encrypted data without decryption.

Let E() be encryption.

- $E(x)*E(y) = E(x+y)$ where $E(x) = f(g^x)$ (Paillier system)
- $E(X)^y = E(X * y)$

key trick: map unwanted file F to 0

- $E(0)^{|F|}=E(0*|F|) = E(0)$
- Users encrypt interests in E(0) or E(1)

Returned files can be easily compressed without conflict, as all unwanted files are now E(0)

Looking Forward…

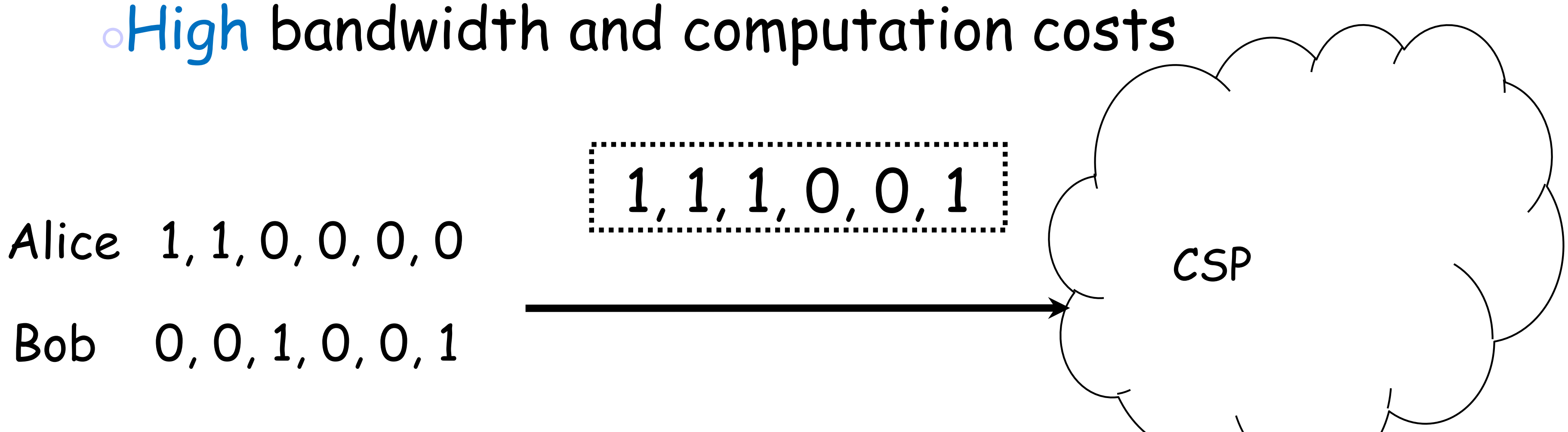
Full homomorphic encryption has perfect security, but too expensive

More efficient solutions leak some information (e.g., search pattern, access pattern, …)

- Security and efficiency trade-offs (e.g., collaborative search)
- Query expressiveness (e.g., function query)
- Untrusted CSP (i.e., malicious)

Collaborative Search [JPDC'12]

- Prior research solves the privacy problem.
- But the overhead is high
  - High bandwidth and computation costs

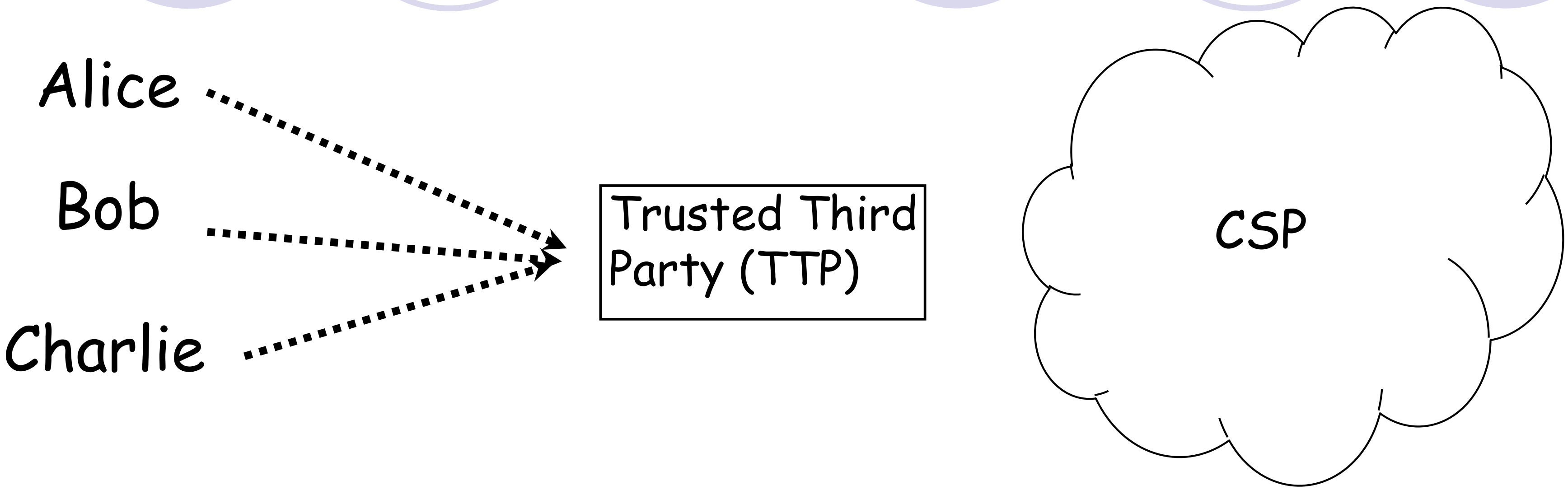


A lightweight SE used that reveals access pattern

We can combine queries

- Efficiency: 1 computation for n users
- Privacy: Hide access pattern from CSP

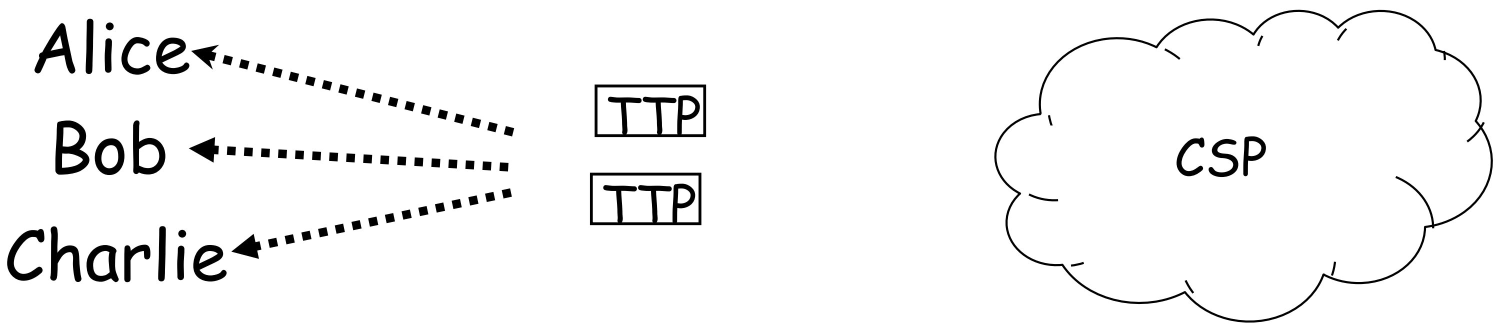
Trusted Third Party (TTP)



- Deploy a Trusted Third Party (TTP)
- Users forward their queries to TTP and TTP to CSP as one single query.

Extensions [JCST'17]

- Multiple TTPs: resolve bottleneck at TTP



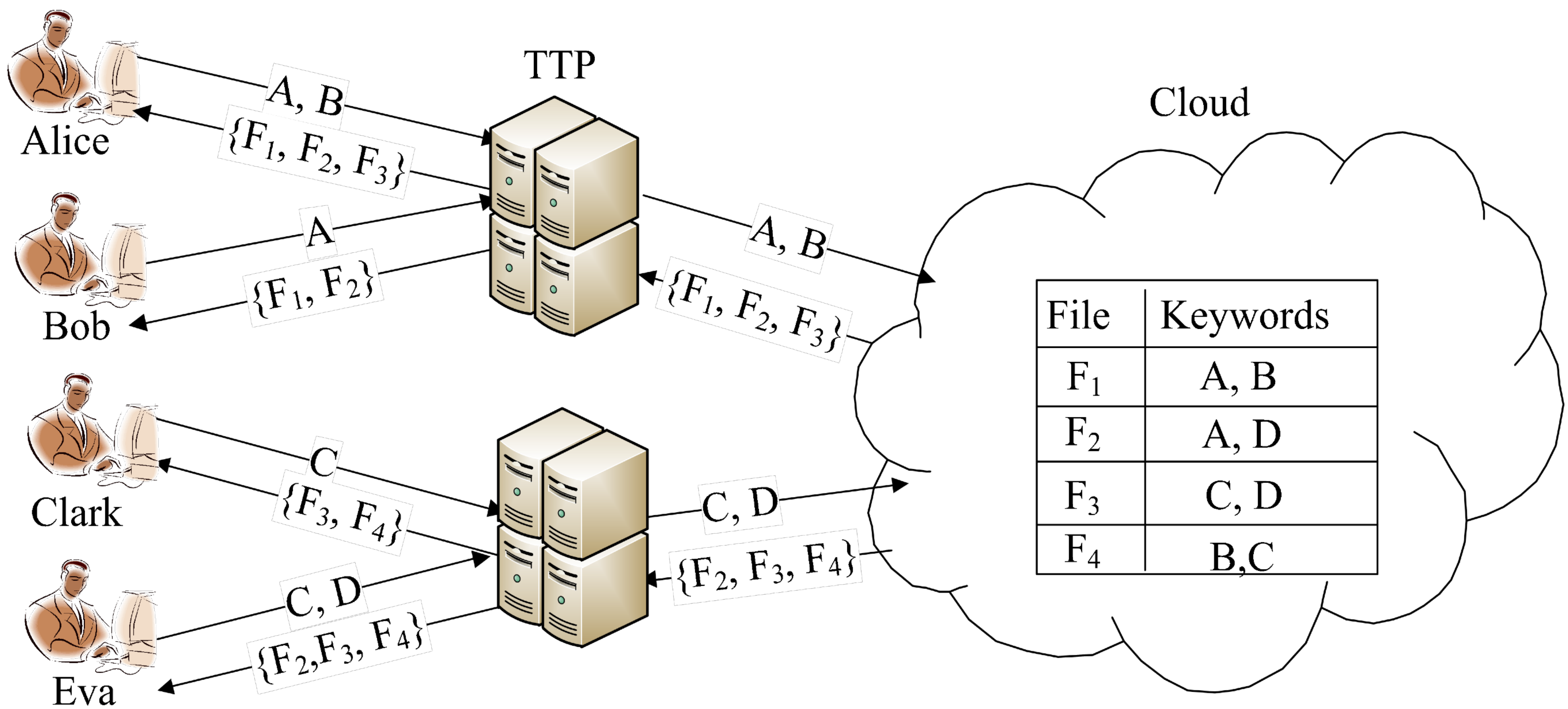
- **Cost Efficiency:** For a given group #, group users with overlapping keywords to minimize # of 1s in each group.

- **Load Balancing:** For a given U, create a minimum # of groups such that 1s in each group are bounded by U.

- **Robustness:** For a given K, use one of the grouping criteria in such a way that each query appears in at least K different groups.

Grouping Example

- 10 files before grouping and 6 files after grouping



Problem Formulation

- Classifying n users into k groups of equal size, so that the number of keywords in k combined queries, i.e., the total number of 1s, is minimized.

- Basic idea: K-Mean-based Dynamic Grouping

- Choose k queries as the seeds, and classify the queries that are closest to the seeds.

K-Mean-based Dynamic Grouping

$Q_1 = \langle 11100000 \rangle$	$Q_5 = \langle 00000111 \rangle$
$Q_2 = \langle 11000000 \rangle$	$Q_6 = \langle 00000011 \rangle$
$Q_3 = \langle 11000000 \rangle$	$Q_7 = \langle 00000011 \rangle$
$Q_4 = \langle 00010000 \rangle$	$Q_8 = \langle 00001000 \rangle$

- P1: Random
- P4: Random Robust
- KMDG: P2

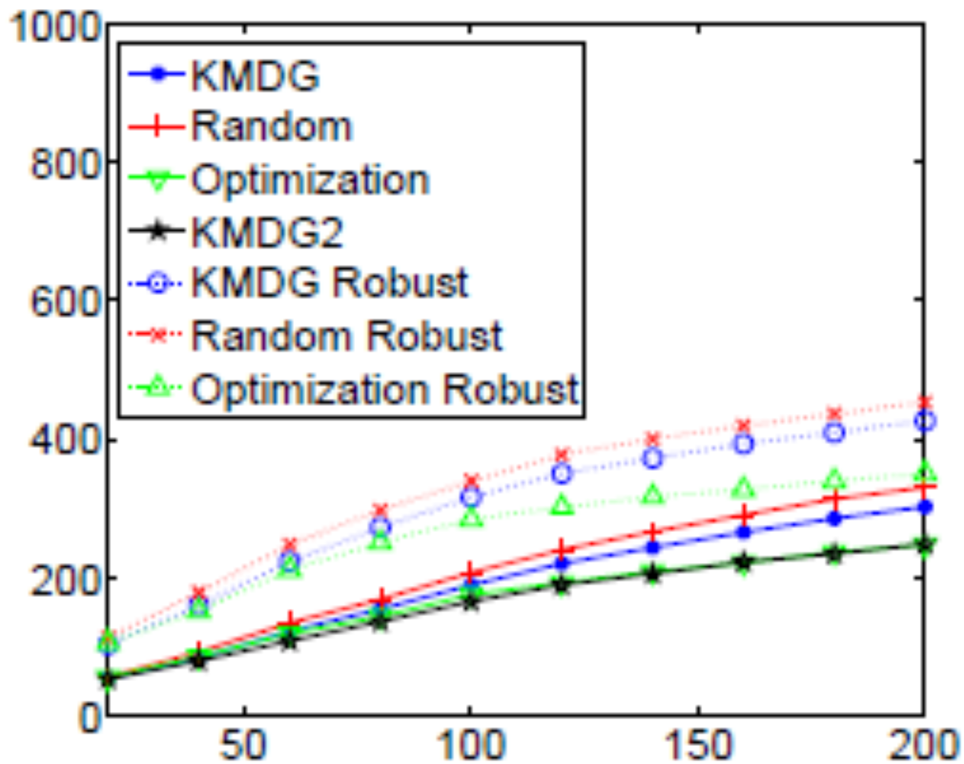
  Balance group size
- KMDG2: P3

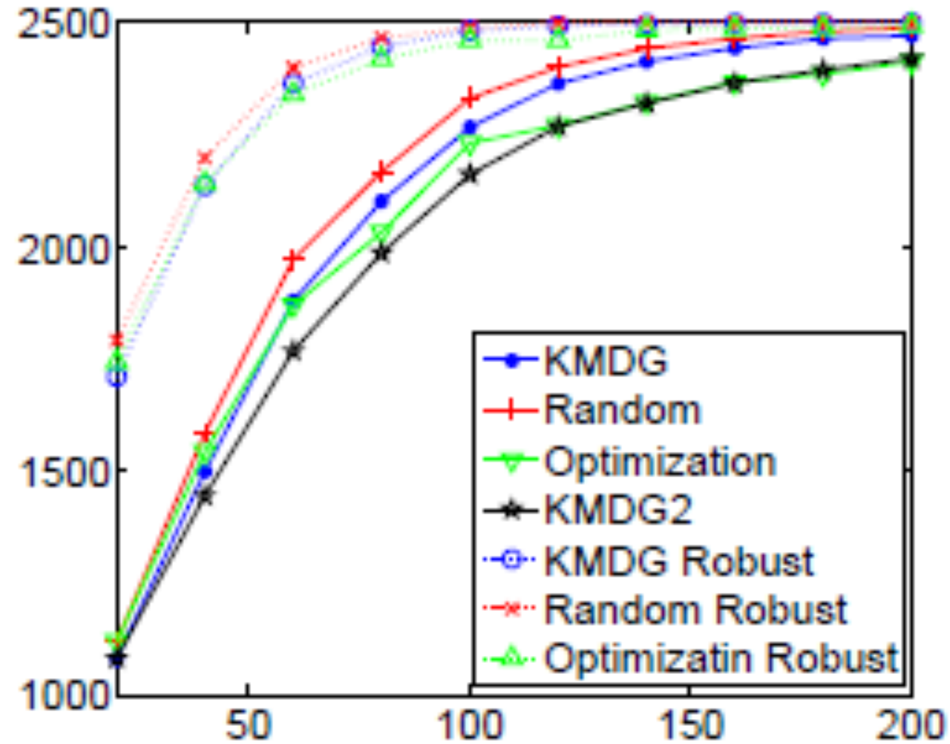
  Balance # of 1s
- KMDG Robust: P5

  Duplication

	g_1	g_2	g_3	g_4
P1	Q_1 11100000 Q_5 00000111	Q_2 11000000 Q_6 00000011	Q_3 11000000 Q_7 00000011	Q_4 00010000 Q_8 00001000
P2	Q_1 11100000 Q_4 00010000	Q_2 11000000 Q_3 11000000	Q_6 00000011 Q_7 00000011	Q_5 00000111 Q_8 00001000
P3	Q_1 11100000	Q_2 11000000 Q_3 11000000 Q_4 00010000	Q_5 00000111	Q_6 00000011 Q_7 00000011 Q_8 00001000
P4	Q_1 11100000 Q_5 00000111 Q_2 11000000 Q_6 00000011	Q_3 11000000 Q_7 00000011 Q_4 00010000 Q_8 00001000	Q_1 11100000 Q_5 00000111 Q_2 11000000 Q_6 00000011	Q_3 11000000 Q_7 00000011 Q_4 00010000 Q_8 00001000
P5	Q_1 11100000 Q_4 00010000 Q_2 11000000 Q_3 11000000	Q_6 00000011 Q_7 00000011 Q_5 00000111 Q_8 00001000	Q_1 11100000 Q_4 00010000 Q_2 11000000 Q_3 11000000	Q_6 00000011 Q_7 00000011 Q_5 00000111 Q_8 00001000

Experiment Results





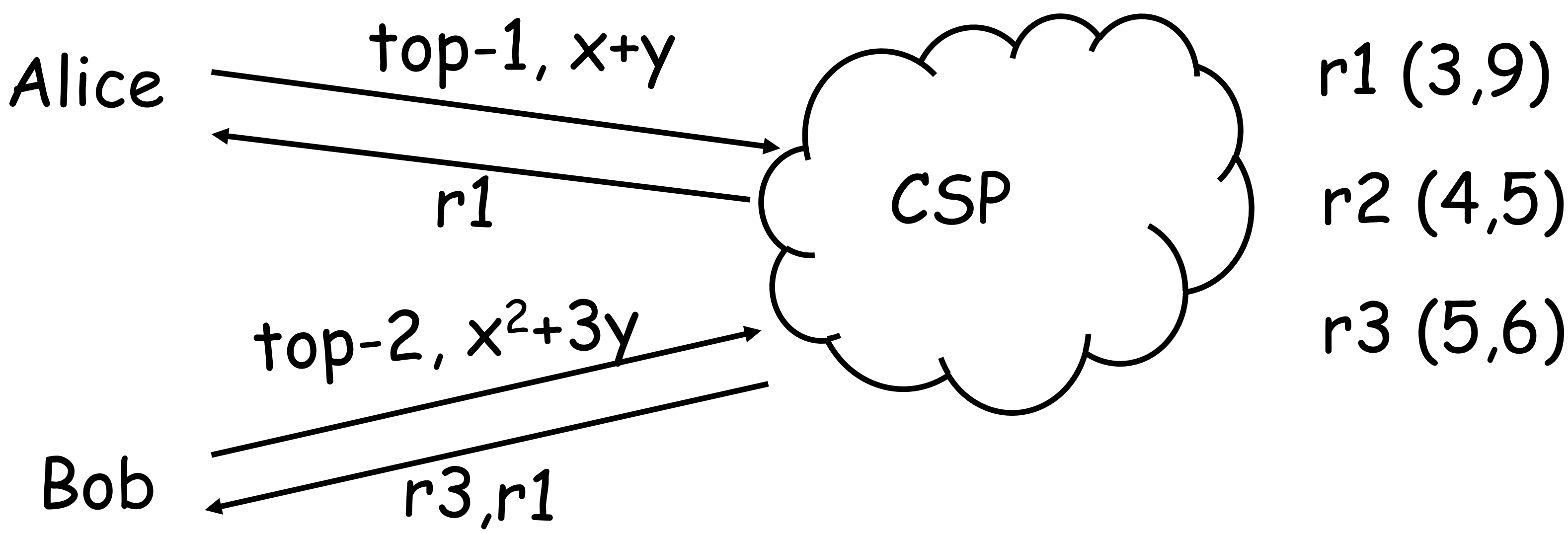
The total number of 1 Bandwidth (MB)

X axis: number of users

K=5 and dictionary size 100

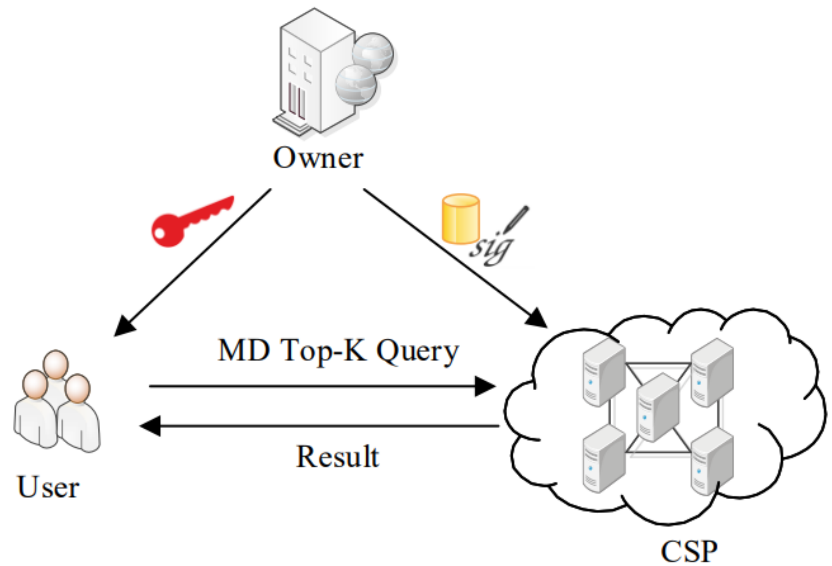
3. Function Query

Users querying the CSP through a function query



But the CSP is untrusted (i.e., malicious)

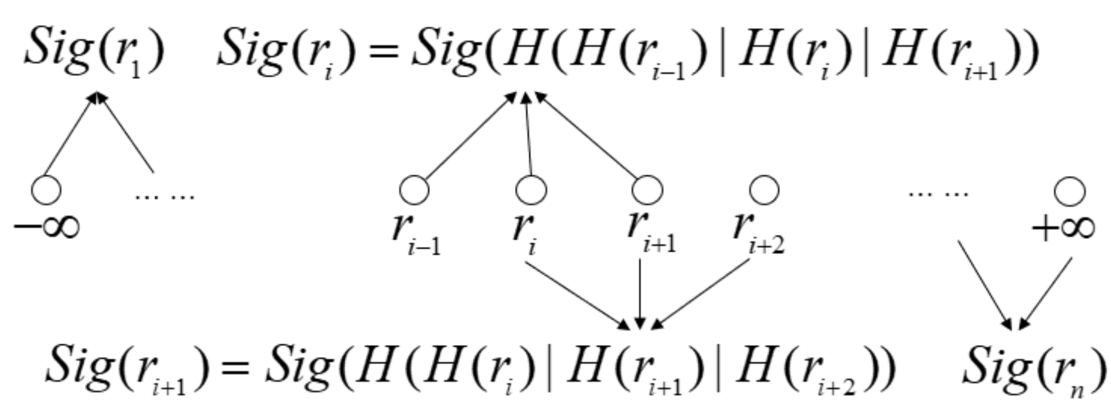
Basic Techniques



- Query authentication
  - Soundness (signature (*r, Sig(r)*)): no tamper
  - Completeness (signature chain): no omit or replace
- Verification object (VO) for auditability
  - Generated by the server, VO provides an independent means of verifying correctness

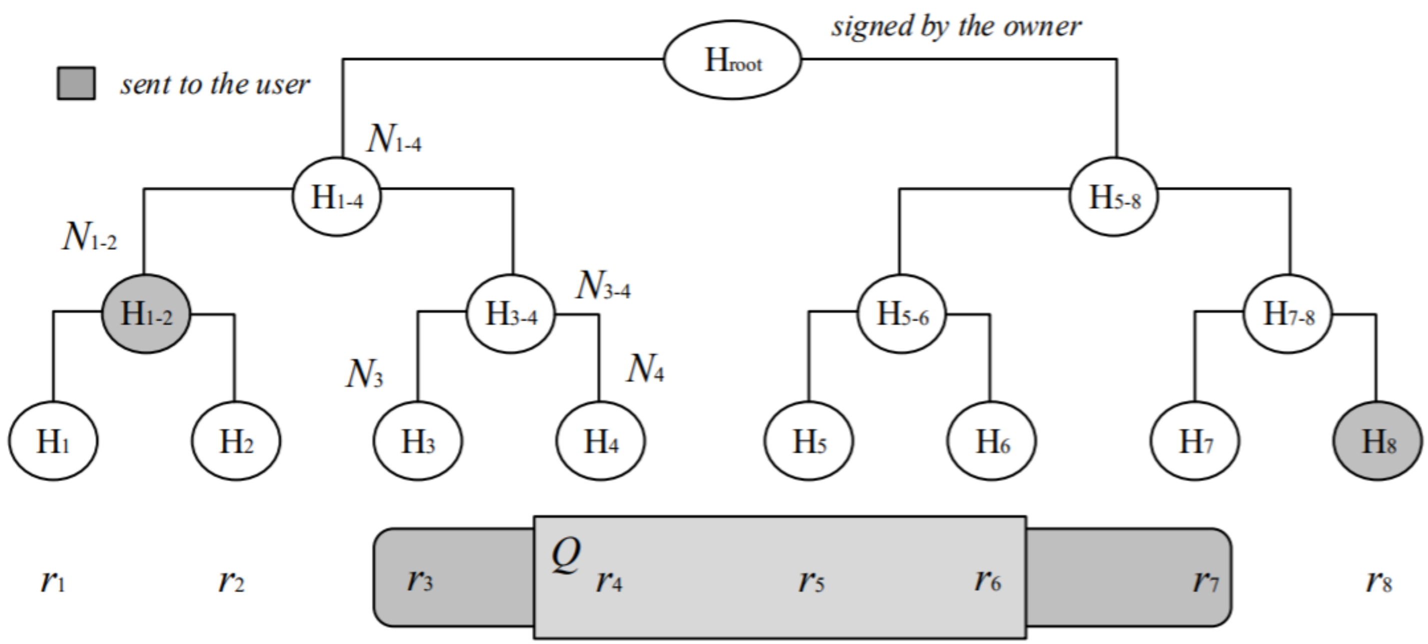
VO basic: Signature Chain

- Basic idea
  - Bind "neighbors" in a given "ranking" function
- Signature chain
  - Each record is chained with its left and right neighbors
  - H(·): one-way hash function



Merkle Hash Tree

- Merkle hash tree
  - Each internal node is a hash result of two children nodes
  - If the range query result is $\{r_4, r_5, r_6\}$, the VO is $\{H_{root}, r_3, r_7, H_{1\text{-}2}, H_8\}$



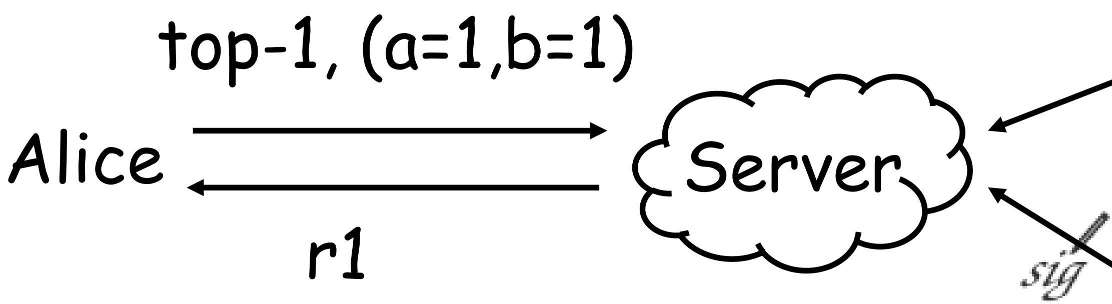
Function Query [ICDE'16]

- Function Query (FQ): map records into functions
  - Univariate Linear Function r1(3,9) → $f1=3a+9$
  - Multivariate Linear Function r1(3,9) → $f1=3a+9b$
  - Multivariate High Degree Function r1(3,9) → $f1=3a+9b^2$
- FQ type
  - Range FQ: retrieve $10 < f(a,b) < 20$
  - Top-K FQ: retrieve top-3 $f(a,b)$
  - KNN FQ : retrieve 3 nearest neighbors of $f(a,b)$

Interval Priority Order [ICDE'16]

Owner

f1=3a+9b ← r1(3,9)
f2=4a+5b ← r2(4,5)
f3=5a+3b ← r3(5,3)

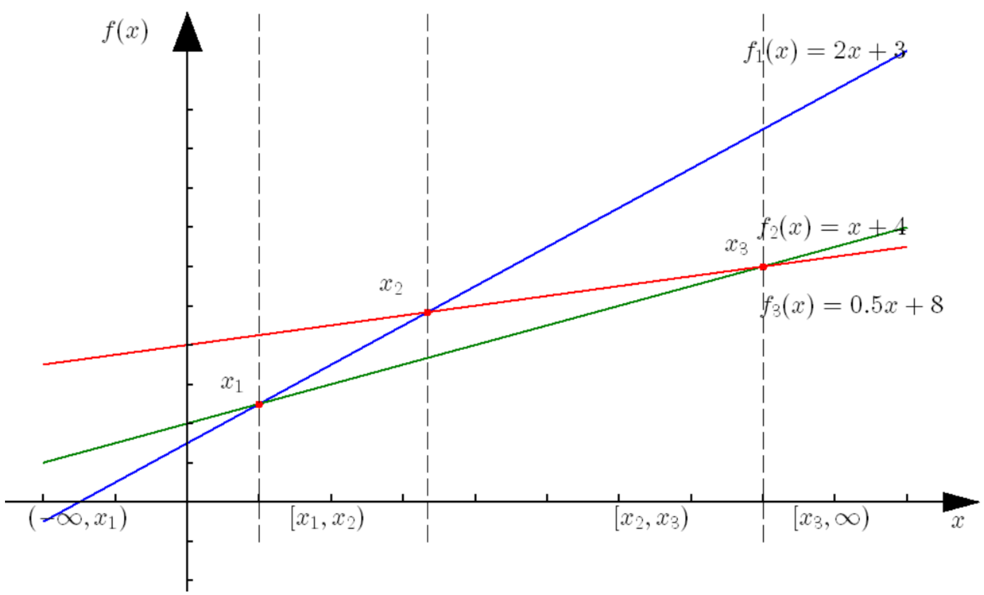


Interval	List
a<2b	f1>f2>f3
2b<a<3b	f1>f3>f2
3b<a<4b	f3>f1>f2
a>4b	f3>f2>f1

a are b are weights in the ranking

The size of the signatures is bounded by $O(n^2)$

Univariate Linear Function



Sorted list

f_3	f_3	f_1	f_1
f_2	f_1	f_3	f_2
f_1	f_2	f_2	f_3
f_0	f_0	f_0	f_0
$(-\infty, x_1)$	$[x_1, x_2)$	$[x_2, x_3)$	$[x_3, \infty)$

Intervals

Sorted list of functions in each interval

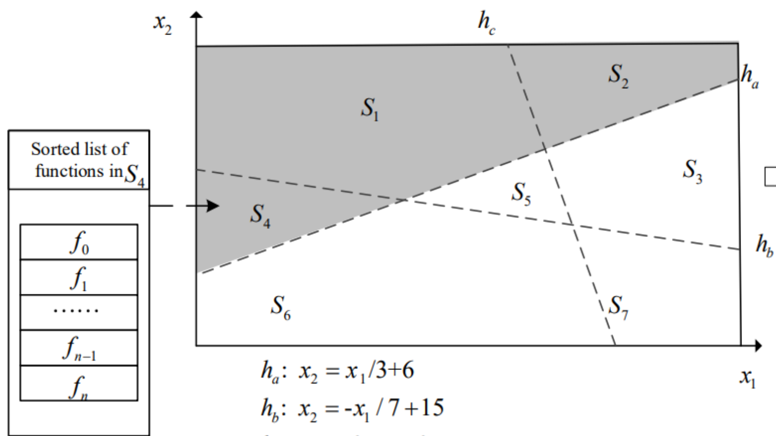
Signatures $Sig_{[x_2,x_3)}(r_3 \mid r_1) = Sig(H(H(r_3) \mid H(r_1) \mid x_2 \mid x_3))$

$Sig(r_2 \mid r_3)$	$Sig(r_1 \mid r_3)$	$Sig(r_3 \mid r_1)$	$Sig(r_2 \mid r_1)$
$Sig(r_1 \mid r_2)$	$Sig(r_2 \mid r_1)$	$Sig(r_2 \mid r_3)$	$Sig(r_3 \mid r_2)$
$Sig(r_0 \mid r_1)$	$Sig(r_0 \mid r_2)$		$Sig(r_0 \mid r_3)$
$(-\infty, x_1)$	$[x_1, x_2)$	$[x_2, x_3)$	$[x_3, \infty)$

Intervals

Corresponding signature chain in each interval

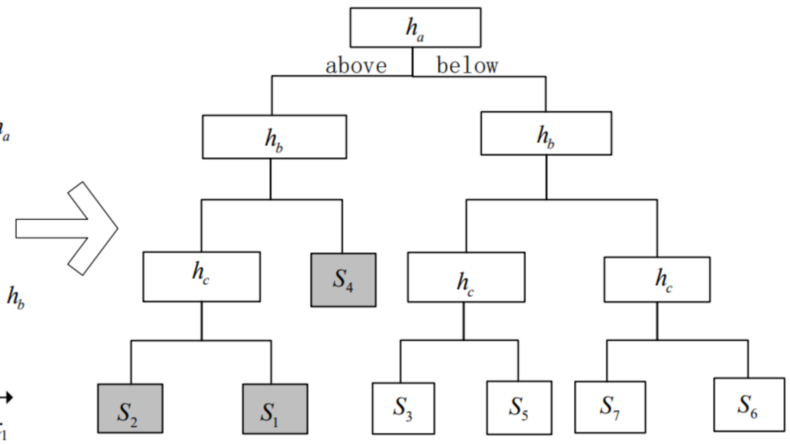
Multivariate Linear Function



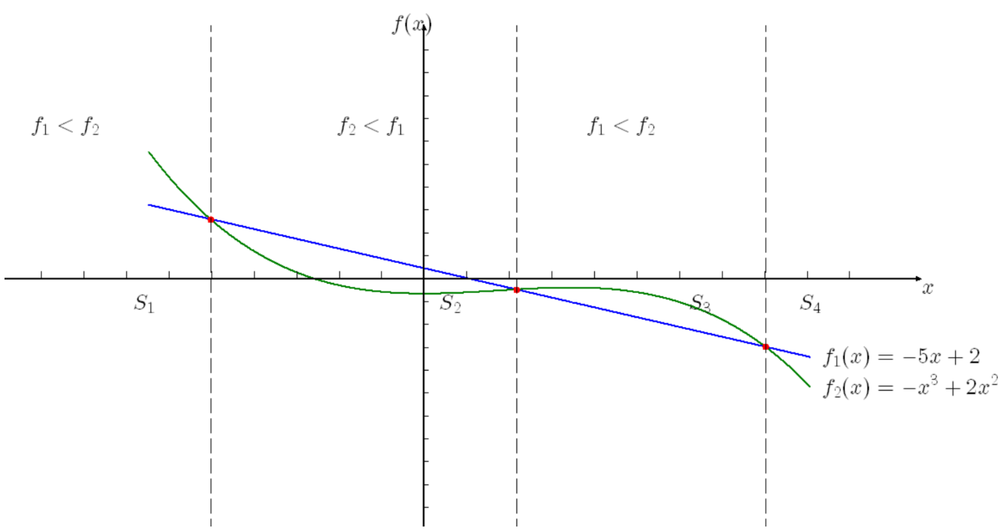
h_a: $x_2 = x_1/3+6$

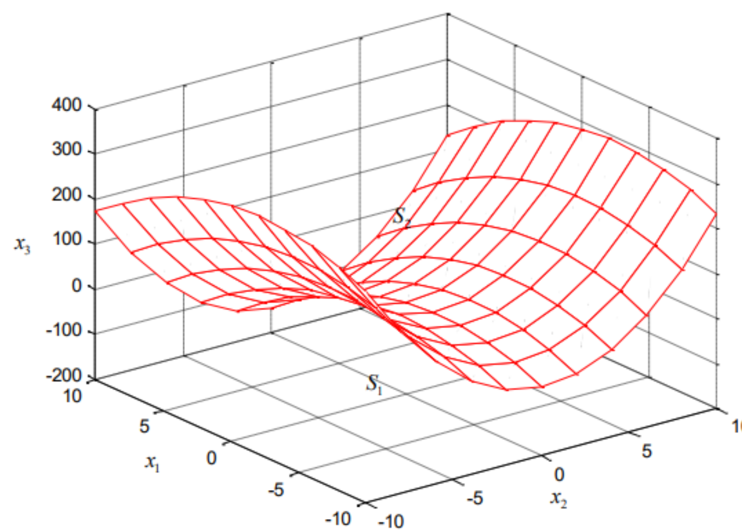
h_b: $x_2 = -x_1/7+15$

h_c: $x_2 = -2x_1+72$



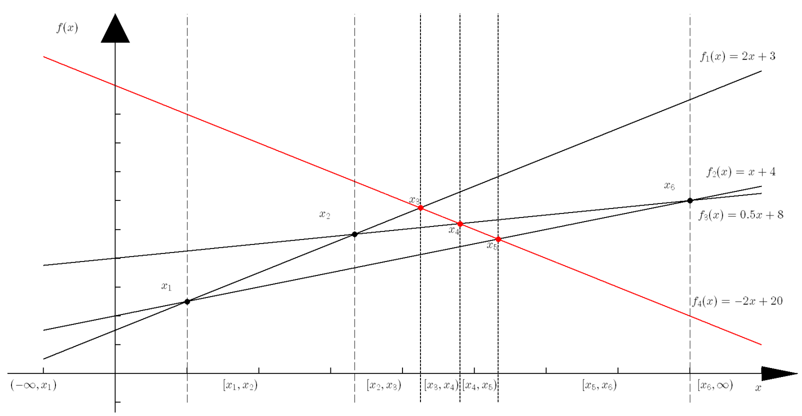
Multivariate High Degree Function





$f_1 = 3x_1^2 - 2x_1 + 4x_2 + 3,\ f_2 = x_2^2 + 5x_2 + x_3$

Data Update

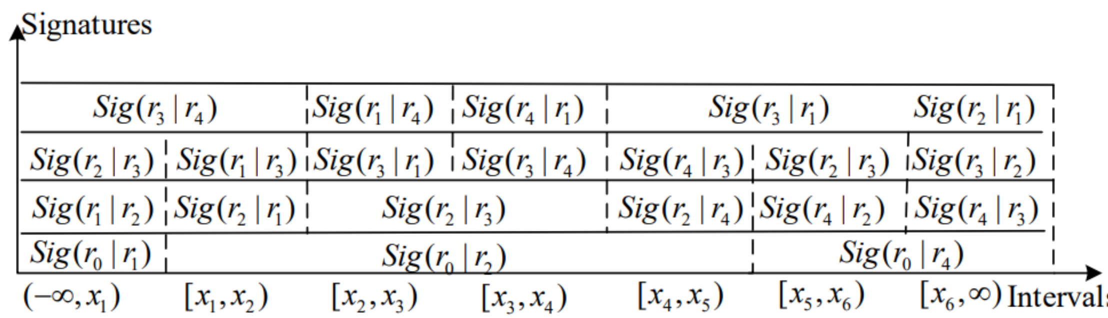


Sorted list

$(-\infty, x_1)$	$[x_1, x_2)$	$[x_2, x_3)$	$[x_3, x_4)$	$[x_4, x_5)$	$[x_5, x_6)$	$[x_6, \infty)$
f_4	f_4	f_4	f_1	f_1	f_1	f_1
f_3	f_3	f_1	f_4	f_3	f_3	f_2
f_2	f_1	f_3	f_3	f_4	f_2	f_3
f_1	f_2	f_2	f_2	f_2	f_4	f_4
f_0	f_0	f_0	f_0	f_0	f_0	f_0

Intervals

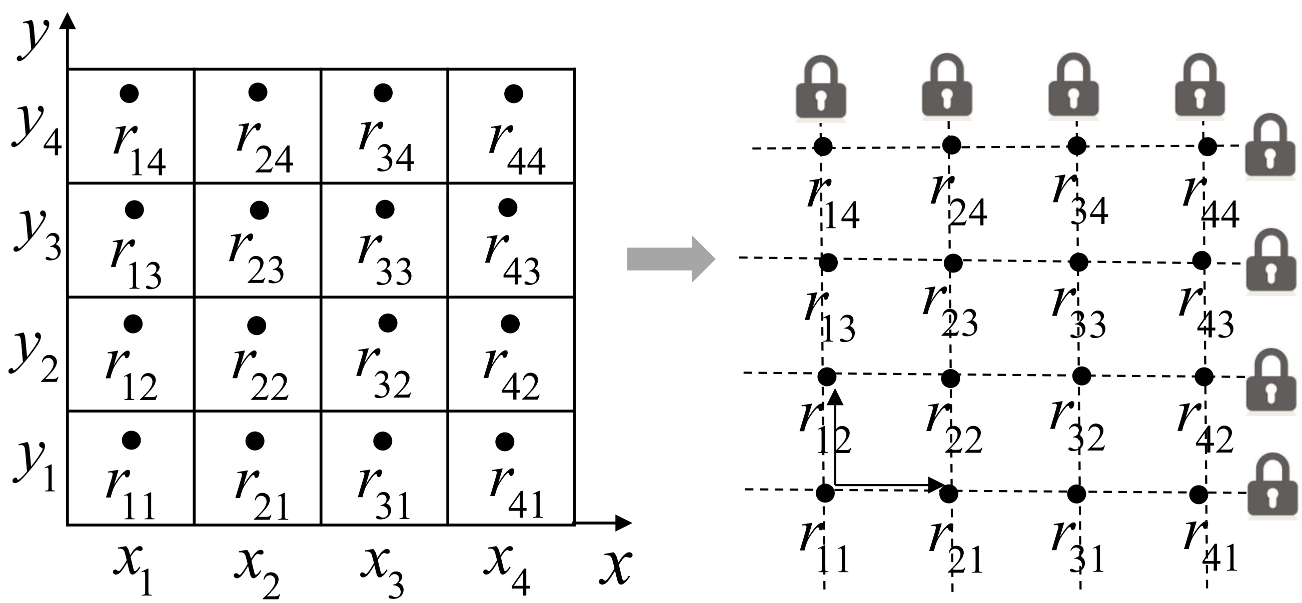
Sorted list of functions in each interval



Corresponding signature chain in each interval

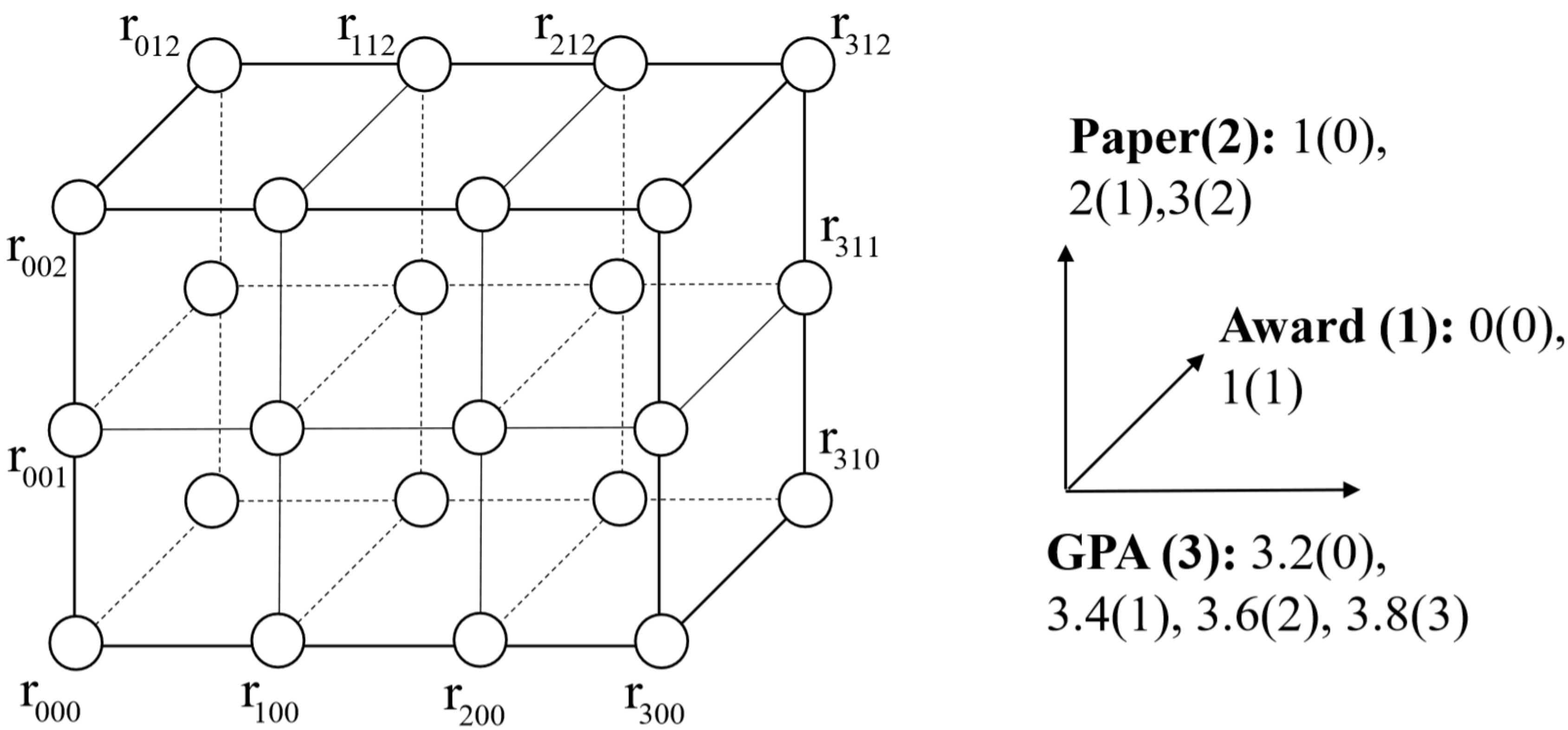
Add a new function f_4, compute new intersections x_3, x_4, x_5.

Dimension Decomposition [IWQoS'18]



- Deploy multiple signature chains on multiple dimensions
  - Assuming each term in a polynomial function is positive
- Each data chained with its neighbor in each dimension
  - r11 is chained with its neighbor r21 and r12 in x and y dimensions
- O(n) signatures for n records, rather than O(n^2)

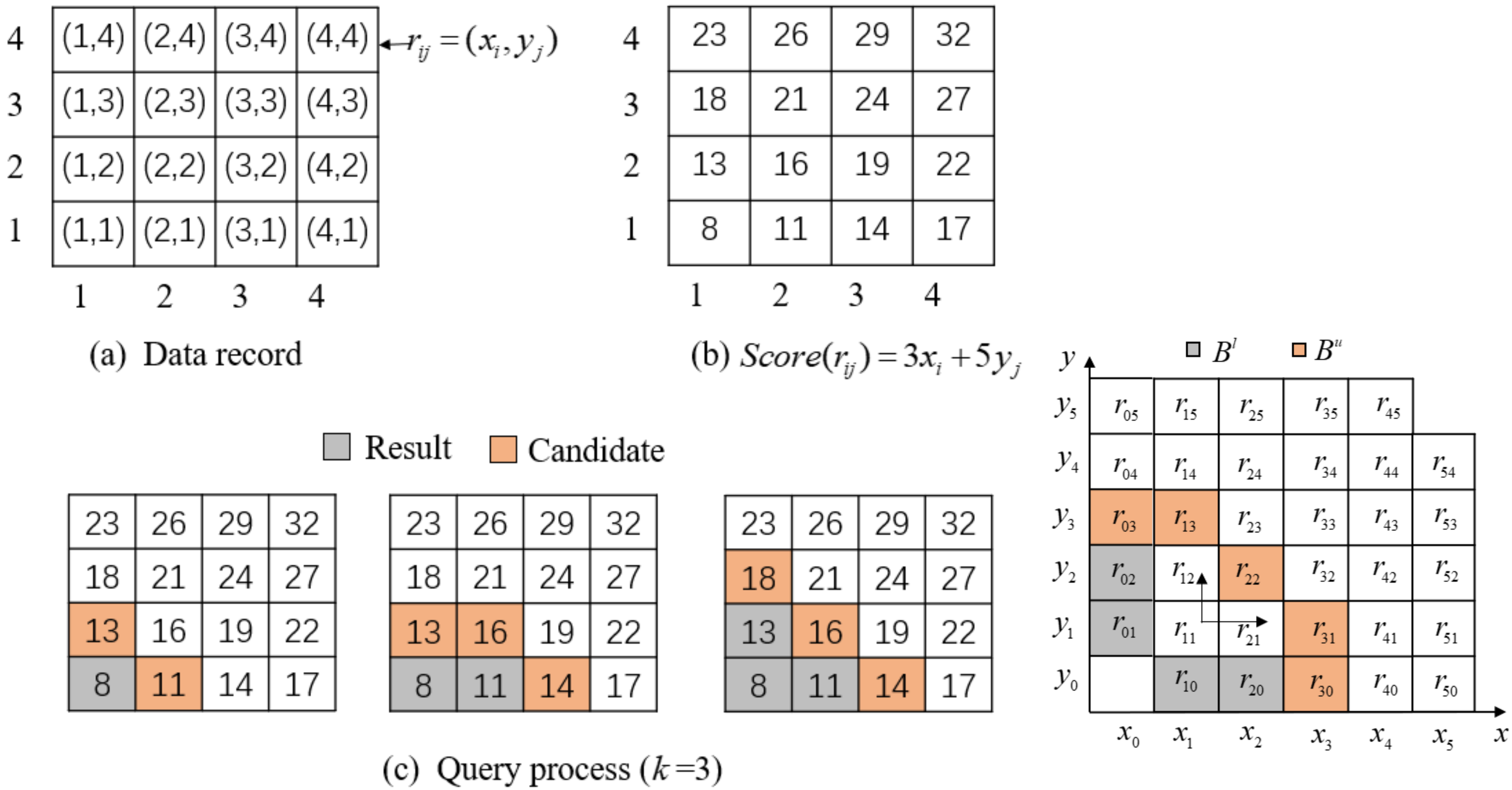
Example of MD Top-K Query



Example: "r_{312}" : a student (GPA, Award, Paper) = (3.8, 1, 3)

Find top-3 student ranking by Score= 5GPA+3Award+2Paper^2

Query and VO

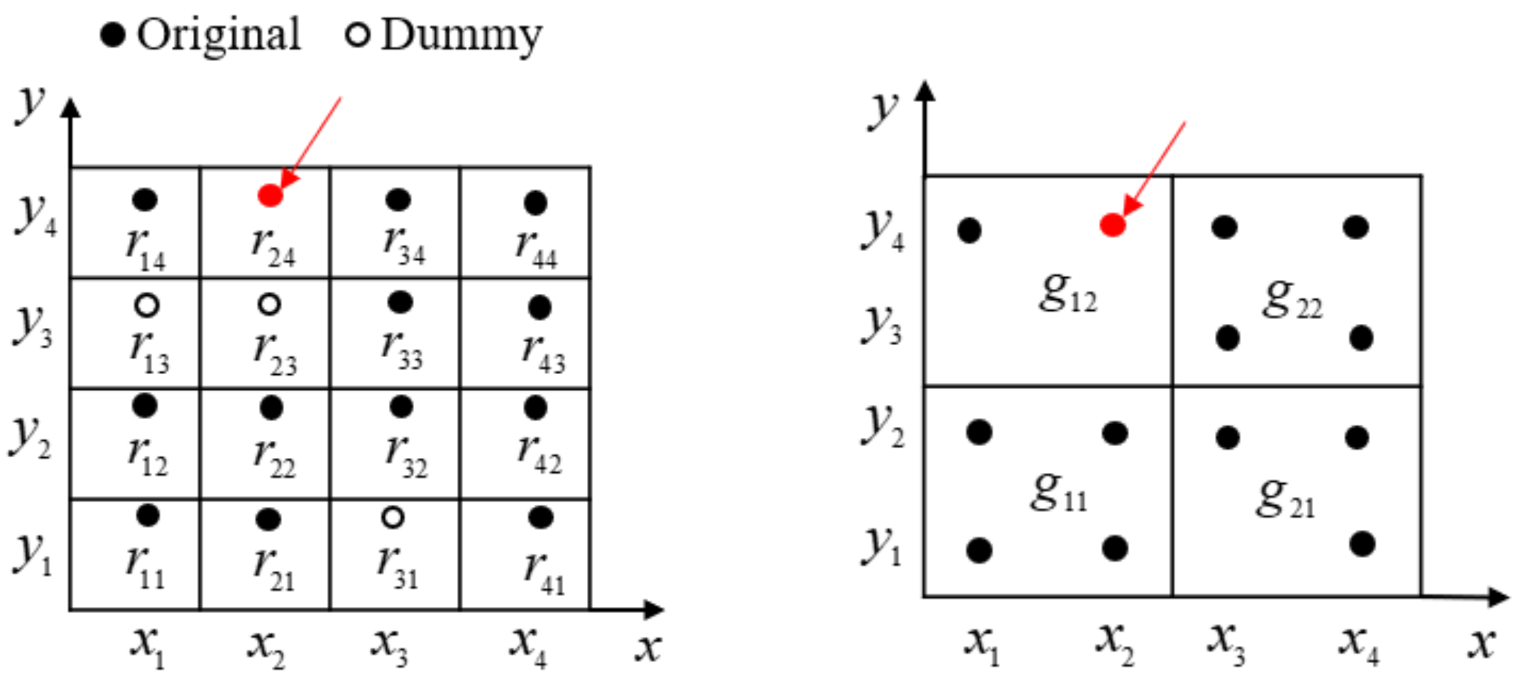


(a) Data record

(b) $Score(r_{ij}) = 3x_i + 5y_j$

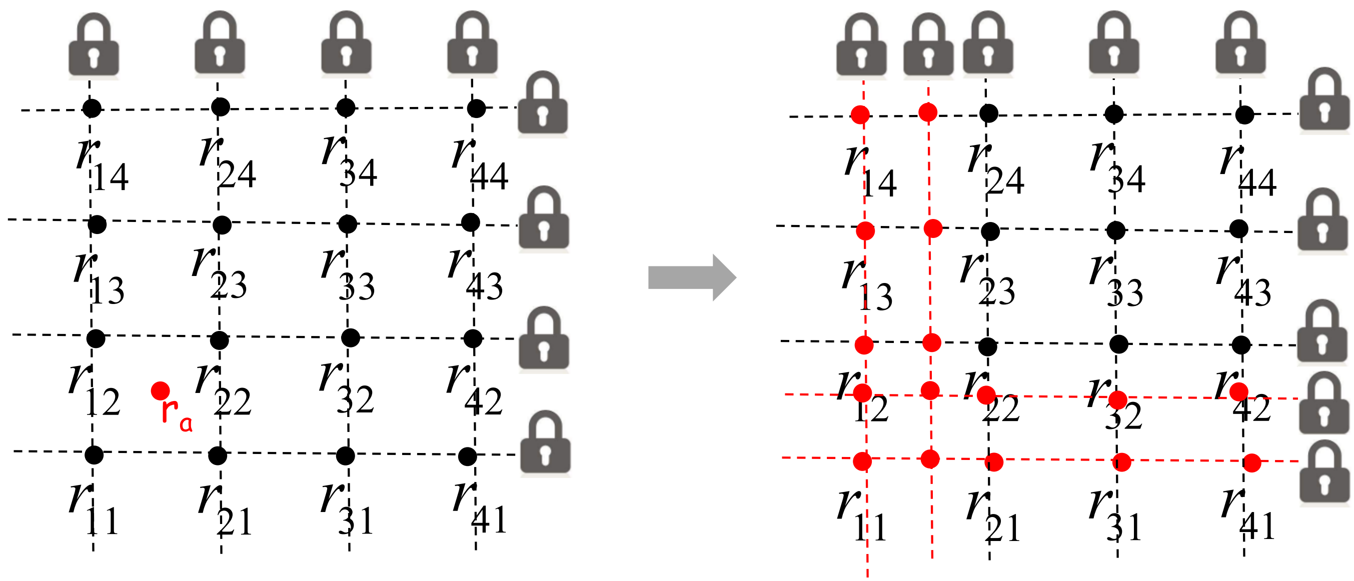
(c) Query process (k=3)

Data Insert



- Sparse: dummy data in a grid
- Dense: multiple data in a grid

Data Update



- If the owner wants to add a new record r_a, it should construct $O(\sqrt[d]{n})$ new signatures.

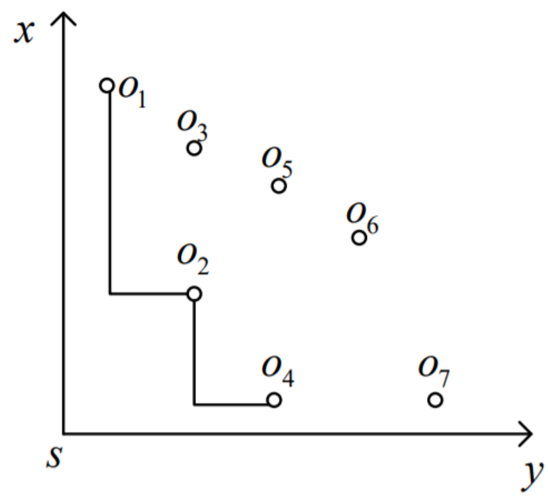
Analytical Study

n: number of records, k: number of query results

d: number of dimensions

Comparison	ICDE'16	IWQoS'18
Query definition	Defined by owner	Defined by user
Signature Construction	$O(n^2)$	$O(n)$ to $O(n^d)$
VO Construction	$O(k)$	$O(k)$
Verify Cost	$O(k)$	$O(k)$
Data Update	$O(n^2)$	$O(\sqrt[d]{n})$

Skyline Query

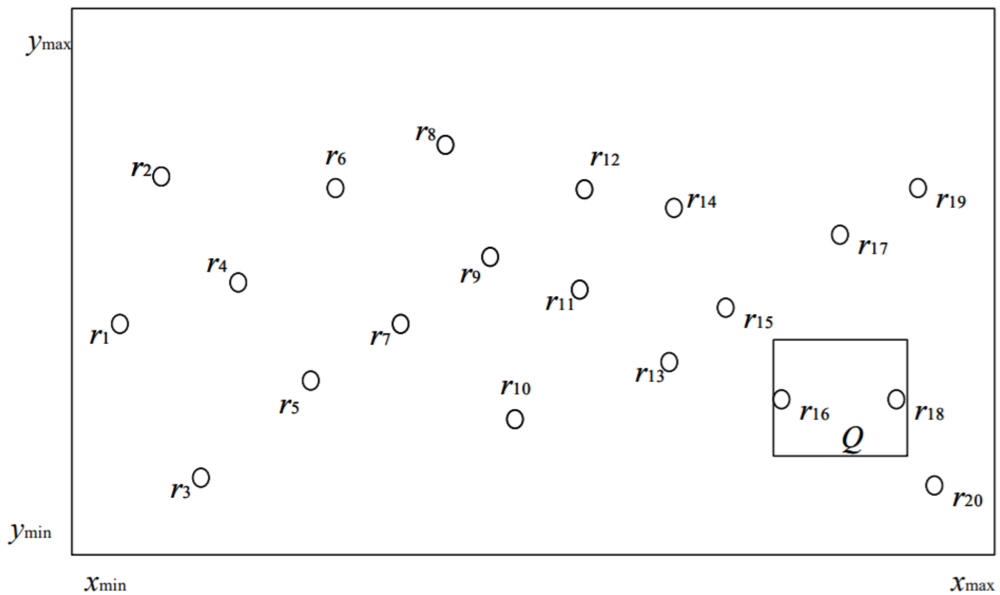


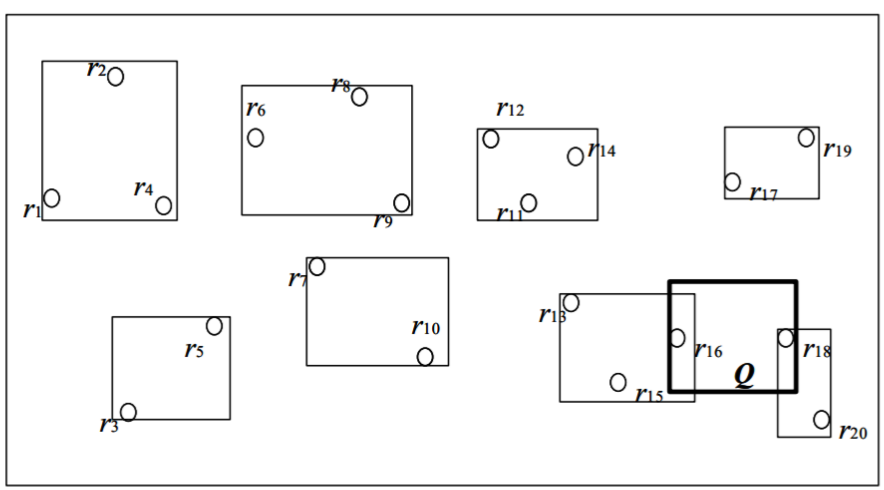
(x, y) = (distance, price)

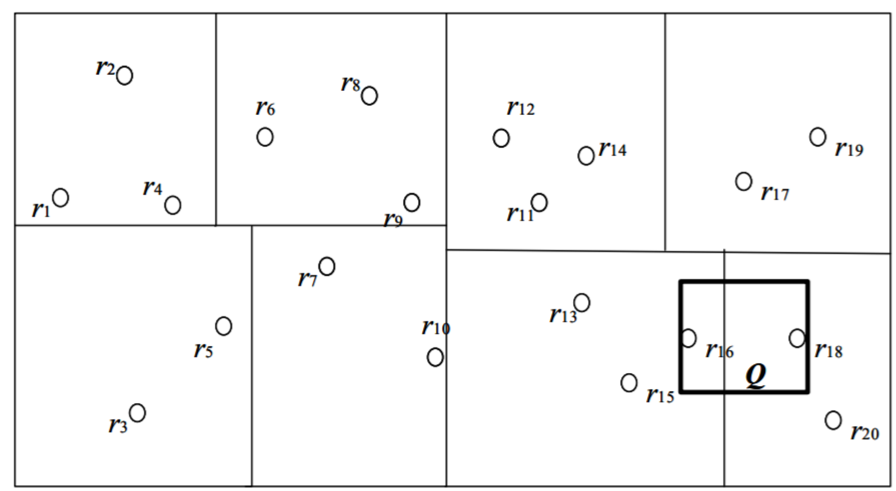
- A skyline query retrieves all dominating nodes.
- A point o_i dominates o_j if coordinate of o_i on each dimension is no larger than that of o_j.
- Bind point with its skyline neighbors for different locations of s.

Spatial Query and Partition

- KNN in 2-D
- Partition
  - Cluster
  - Grid

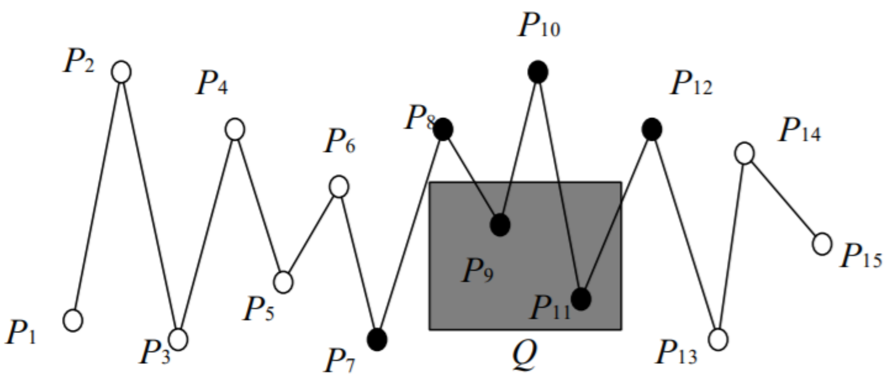


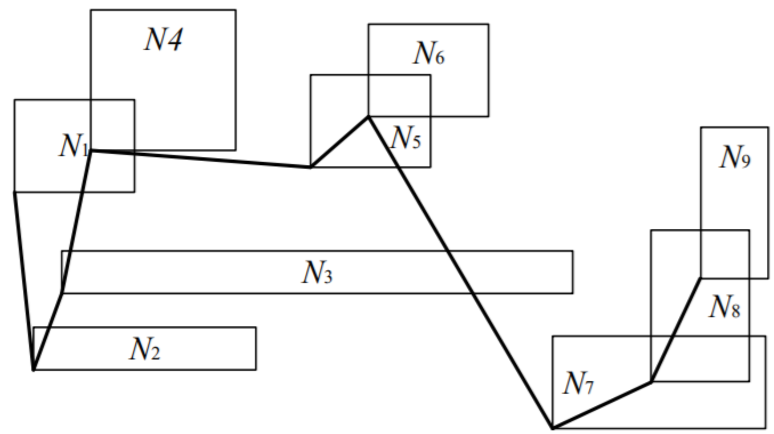




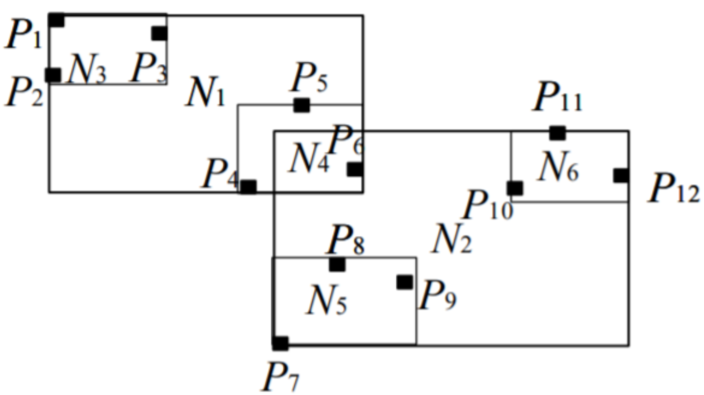
Different Bindings

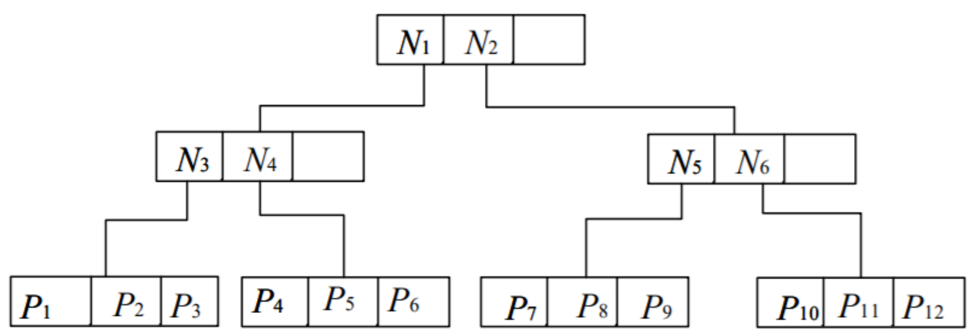
Line of points and line of blocks (rectangles)



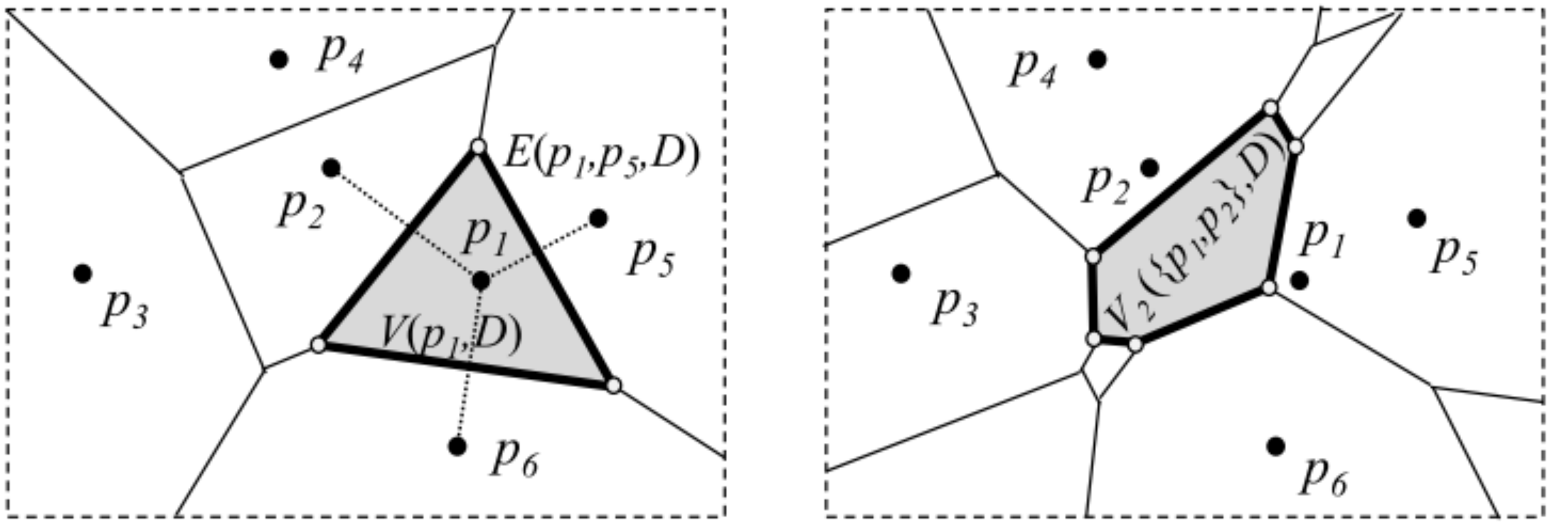


Hierarchical: blocks of blocks





Moving KNN and Safe Region

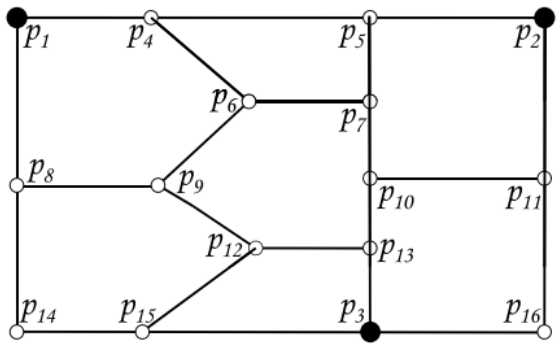


(a) Voronoi cells (b) order-k Voronoi cells, at $k = 2$

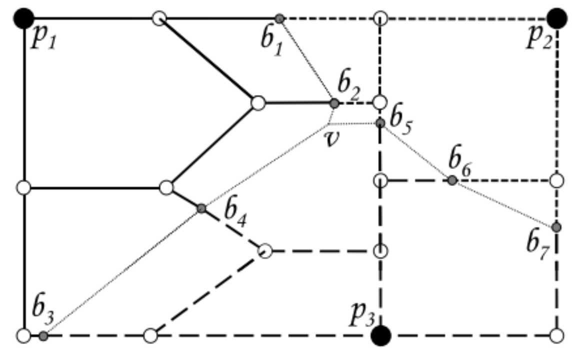
- The generator of $V(p_1,D)$ is $\{p_2, p_5, p_6\}$.
- The generator of $V(\{p_1, p_2\}, D)$ is $\{p_3, p_4, p_5, p_6\}$.

Looking Forward…

Voronoi diagrams for road networks

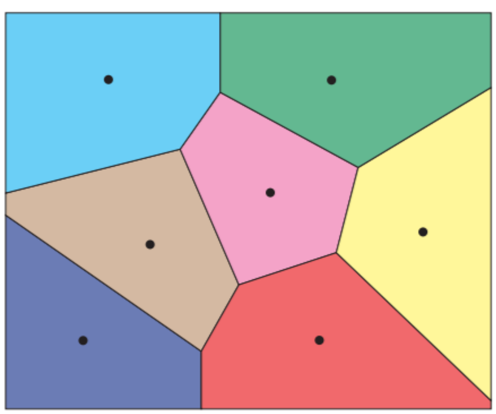


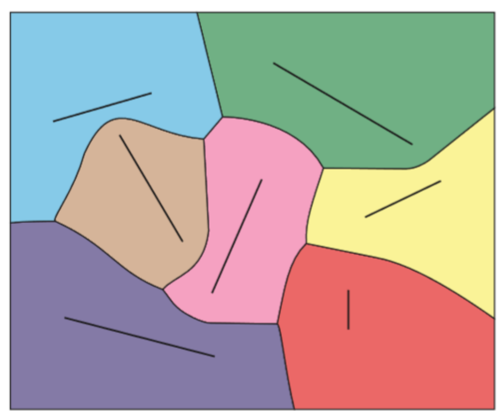
(a) Road network



(b) Network Voronoi diagram

A driver may issue a moving kNN query to obtain k nearest expressways or shopping mall





4. Conclusions

- Cloud Service Provider (CSP)
  - Trust: direct and indirect trust
  - Suspicious: cryptography
    - semi-trusted: searchable encryption
    - untrusted: function queries in n-D space
- On-going Projects
  - Ambient key generation
  - Moving target defense and intractability
  - Security-performance trade-off

Future…

- Fully secure, but probably too expensive
  - Differential privacy
  - Full homomorphic encryption

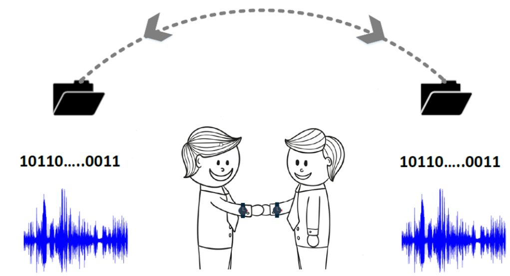
- Decentralized trust
  - Blockchain: each node is equally untrusted
  - Idea: Index table stored in distributed ledger
  - Extension: node's trust score based on behavior
- Science of Security (S & P 2017)

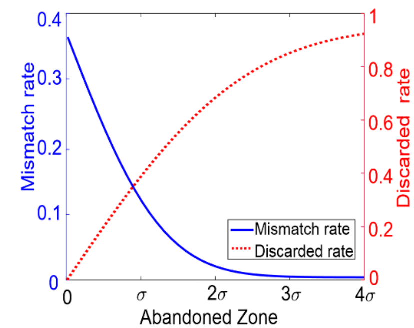
References

- [Computing Survey'17] W. Jiang, G. Wang, M. Z. Alam Bhuiyan, and **J. Wu**, " Understanding Graph-based Trust Evaluation in Online Social Networks: Methodologies and Challenges," *ACM Computing Surveys*, Vol. 49, No. 1, 2016.
- [CRYPTO'05] R. Ostrovsky and W. Skeith III, "Private Searching on Streaming Data," *Proc. of CRYPTO*, 2005.
- [JPDC'12] Q. Liu, C. C. Tan, **J. Wu**, and G. Wang, "Cooperative Private Searching in Clouds," *Journal of Parallel and Distributed Computing,* Vol. 72, No. 8, 2012, 1019-1031.
- [JCST'17] Q. Liu, Y. Guo, **J. Wu**, and G. Wang," Effective Query Grouping Strategy in Clouds," *Journal of Computer Science and Technology* , Vol. 32, No. 3, 2017, 1231 - 1249.
- [ICDE'16] G. Yang, Y. Cai, and Z. Hum, "Authentication of Function Queries," *Proc. of ICDE*, 2916.
- [IWQoS'18] X. Zhu, **J. Wu**, W. Chang, G. Wang, and Q. Liu," Authentication of Multi-dimensional Top-K Query on Untrusted Server," *Proc. of* IEEE/ACM IWQoS, 2018.

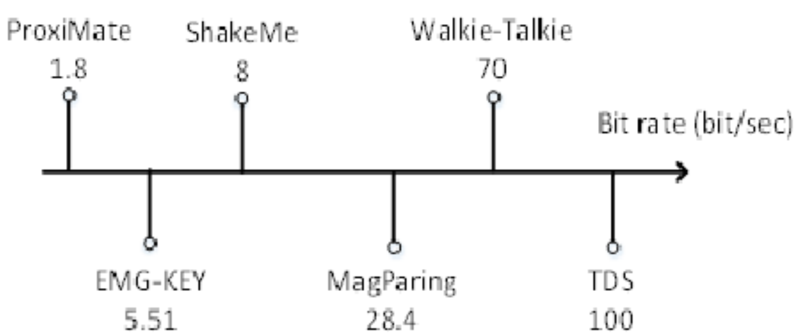
5. Samples: Ambient Key Generation

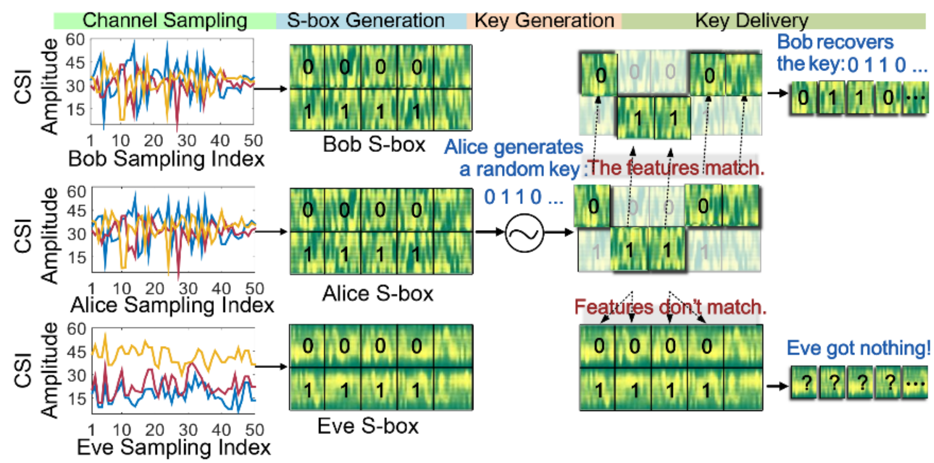
- Random signals (which signal?)
  - Shaking trajectory (ShakeMe, IUCC 2015)
  - Gait (Walkie-Talkie, IPSN 2016)
  - Magnetic signals (MagParing, TIFS 2016)
  - Electromyography (EMG-KEY, Sensys 2016)
  - Ambient wireless signals (ProxiMate, Mobisys 2011)
  - Channel state information (TDS, CCS 2016)





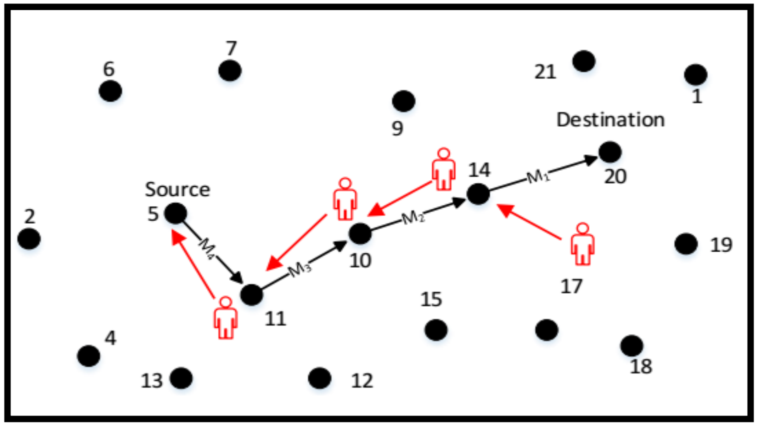
- Quantization
  - Performance and security trade-offs
  - Usability





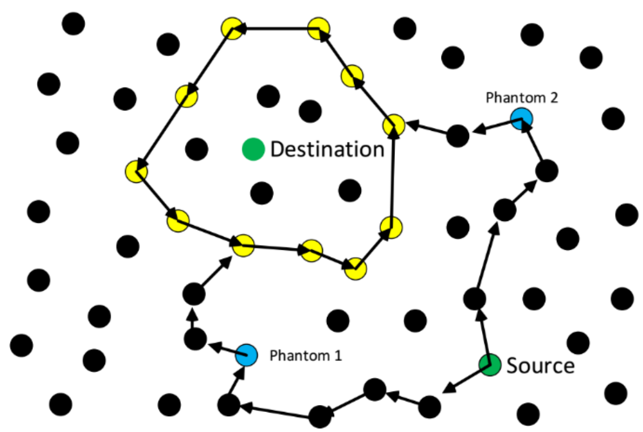
Moving Target Defense

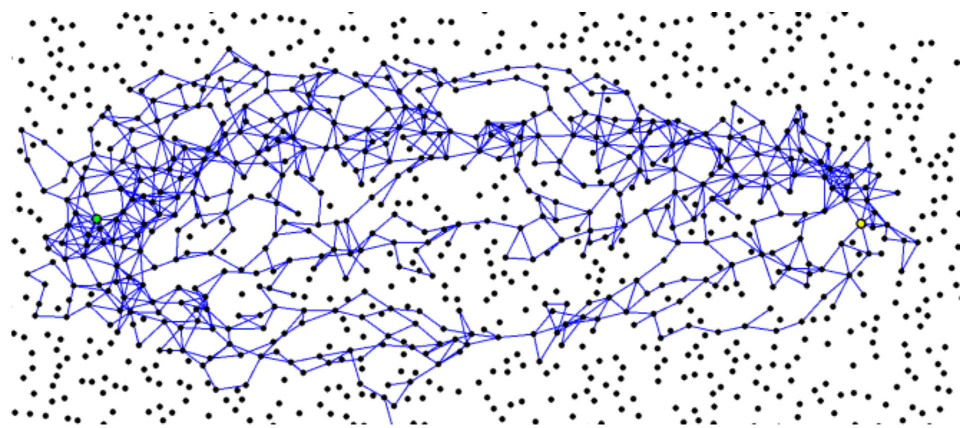
- Source and destination location privacy



(Panda-hunter game)

- Phantom and Probabilistic Routing



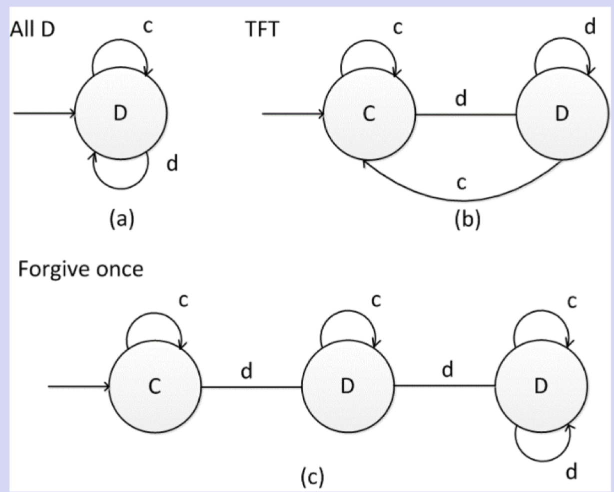


Intractability: Adversarial Model

- It is unclear how smart an adversary can be
  - Deep learning vs. maintaining security and privacy
  - An adversary can use a sophisticated ML method

- Repeated prisoner's dilemma
  - Cooperate (C) or Defect (D)
  - Payoff metrics

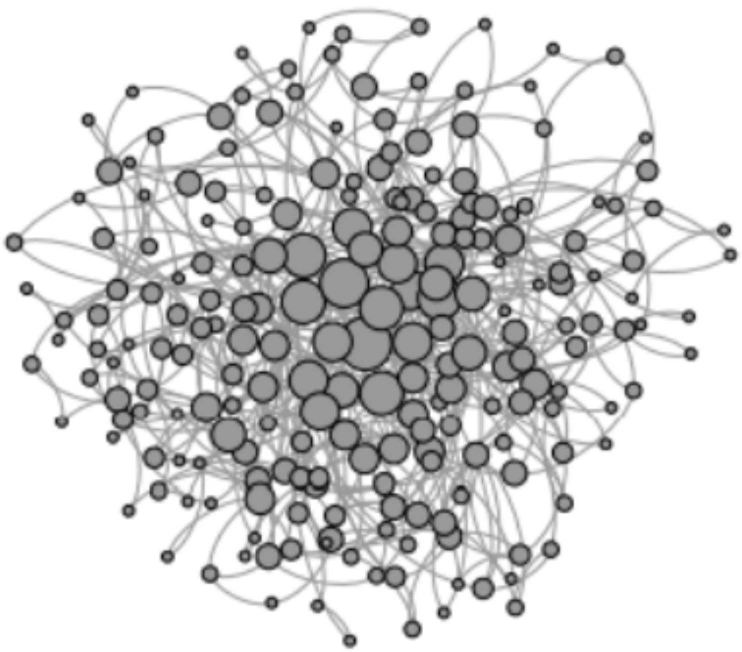
$$\begin{array}{cc} & \begin{array}{cc} C_2 & D_2 \end{array} \\ \begin{array}{c} C_1 \\ D_1 \end{array} & \begin{pmatrix} 3,3 & 0,5 \\ 5,0 & 1,1 \end{pmatrix} \end{array}$$



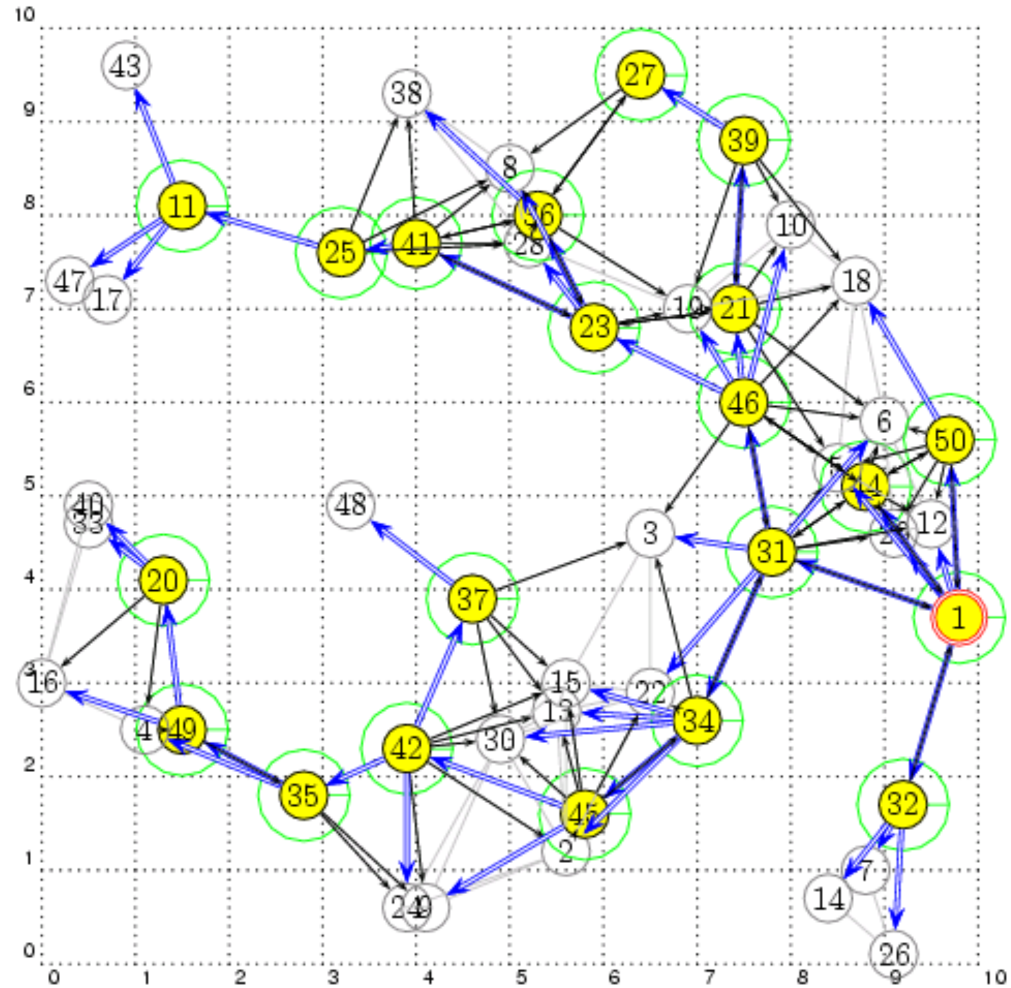
- Genetic algorithm: mutation and crossover
  - 148 bits with 16 recent states: chromosomes

Self-Organizing Solutions

- Hierarchical military command chains



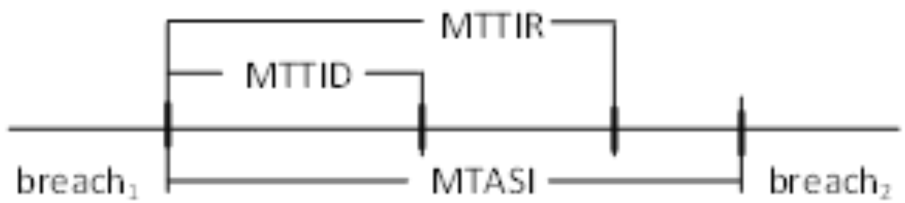
- Dynamic connected dominating set (CDS) rotation



Performance-Security Tradeoff

Dependability includes security

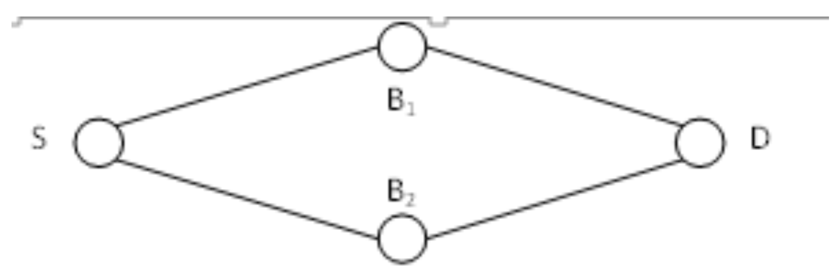
- Mean time between security incidents (MTBSI)
- Mean time to incident discovery (MTTID)
- Mean time to incident recovery (MTTIR)



Performability: work completed before the next security breach

Degradation

- B_1: Level 1 breach, 1,000 hrs
- B_2: Level 4 breach, 5 hrs



Questions

www.cis.temple.edu/~wu