Efficient Proximity Search in Time-accumulating High-dimensional Data using Multi-level Block Indexing

Changhun Han Suji Kim Ha-myung Park

Kookmin University, Korea

Intro

Efficient Proximity Search in **Time-accumulating High-dimensional Data**

Intro

Efficient Proximity Search in **Time-accumulating** **High-dimensional Data**

500 hours of video / min

95M post / day

60000 tracks / day

Intro

Efficient Proximity Search in **Time-accumulating High-dimensional Data**

Most similar video

Most related post

Most similar track

Intro

Efficient Proximity Search in **Time-accumulating High-dimensional Data**

Most similar video

Most related post

Most similar track

Intro

Efficient Proximity Search in **Time-accumulating High-dimensional Data**

within **Specific Time Windows**

Most similar video

in last 2 year

Most related post

in last 1 month

Most similar track

in 2005~2010

Intro

Efficient Proximity Search in **Time-accumulating High-dimensional Data**

within **Specific Time Windows**

↓

But how?

Most similar video
in last 2 year

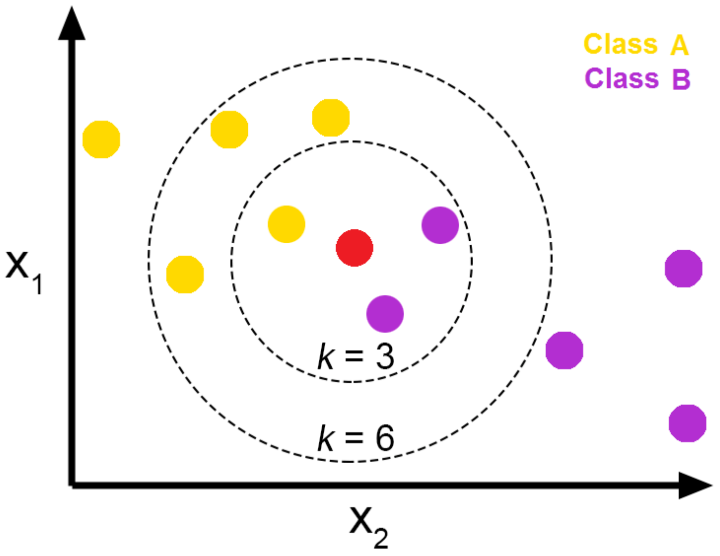
Most related post
in last 1 month

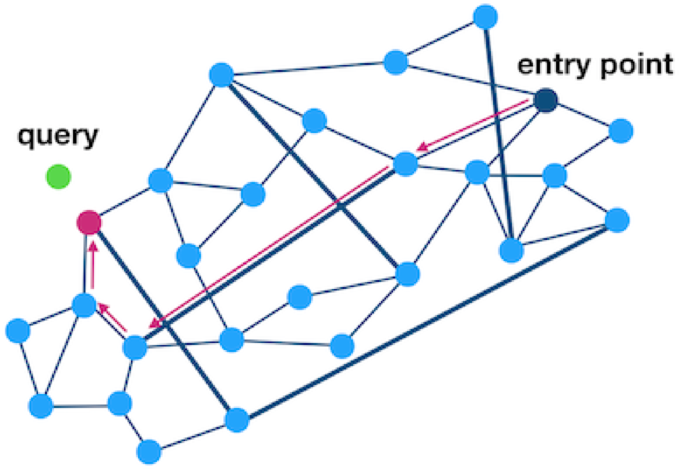
Most similar track
in 2005~2010

Index

Index

Preliminaries : Related Work

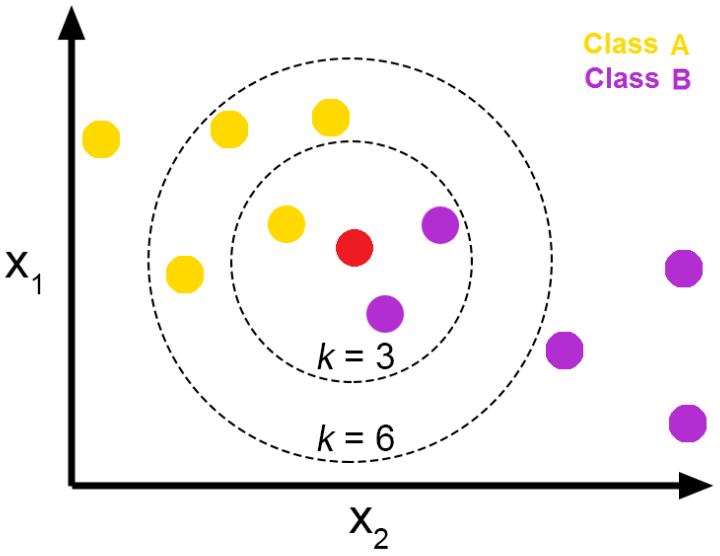


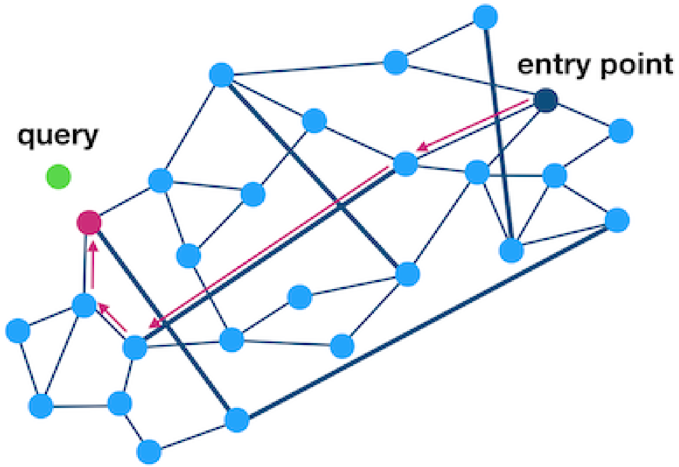


K-Nearest Neighbor Search (KNN Search)

Efficiently find nearest neighbor

Preliminaries : Related Work





K-Nearest Neighbor Search (KNN Search)

Efficiently find nearest neighbor

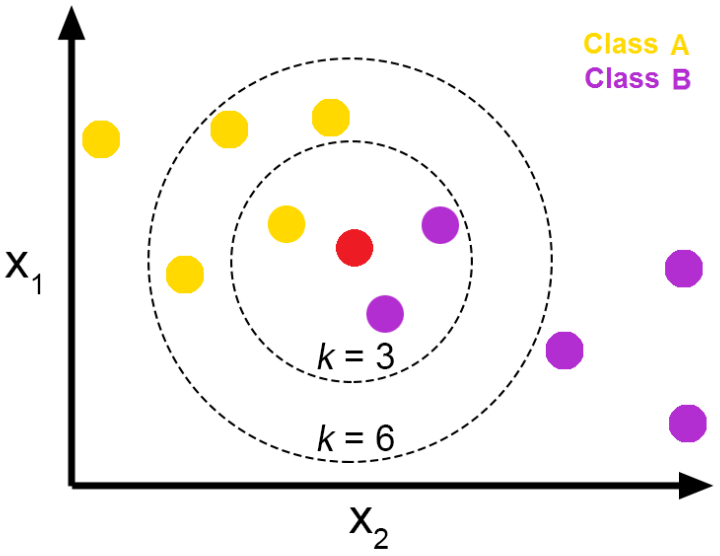
- **Approximate KNN Search**

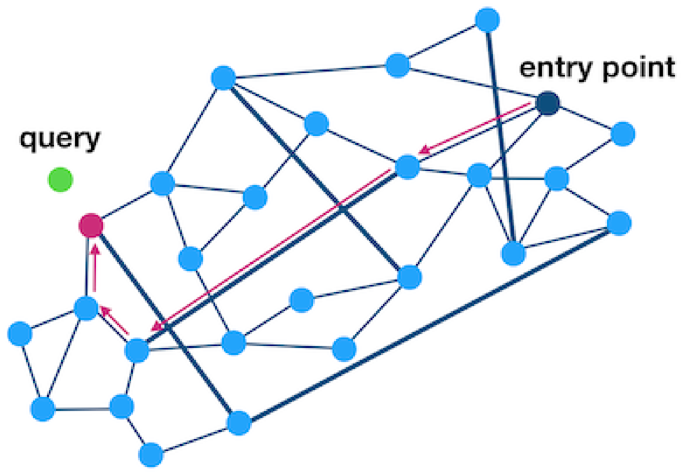
Search speed ⬆ Accuracy ⬇

Suitable for high-dimensional data:

Graph-based, PQ-based, ⋯

Preliminaries : Related Work





K-Nearest Neighbor Search (KNN Search)

Efficiently find nearest neighbor

- **Approximate KNN Search**

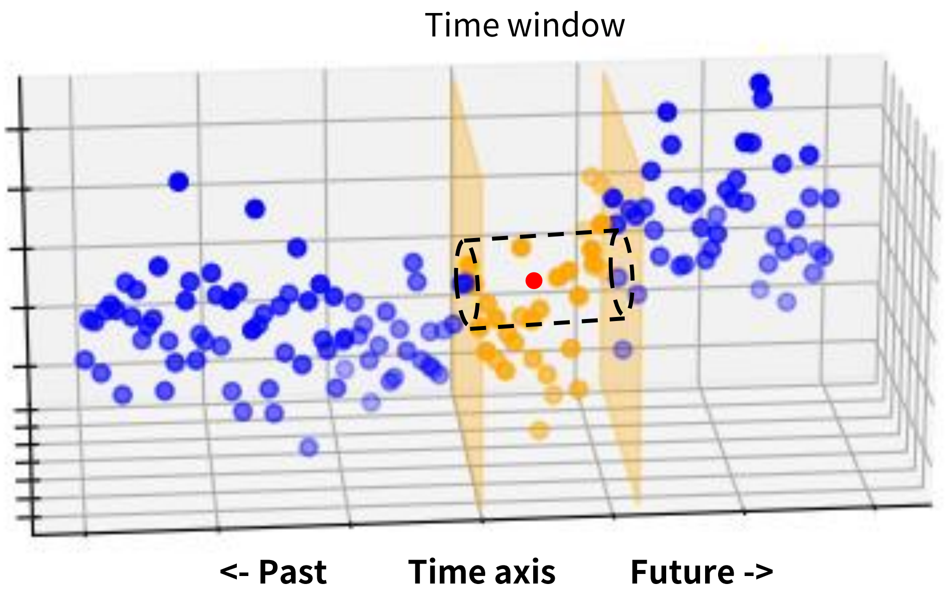
Search speed ⬆ Accuracy ⬇

Suitable for high-dimensional data:

Graph-based, PQ-based, ⋯

within **Specific Time Windows?**

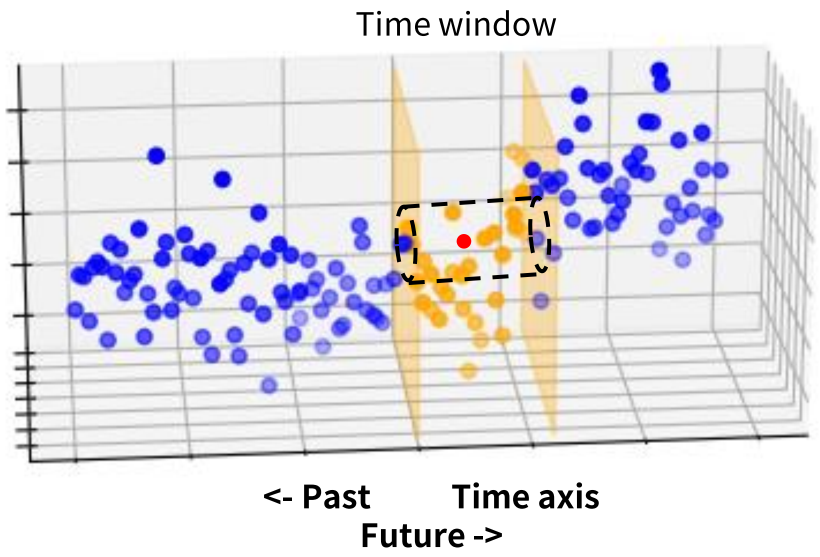
Preliminaries : Related Work

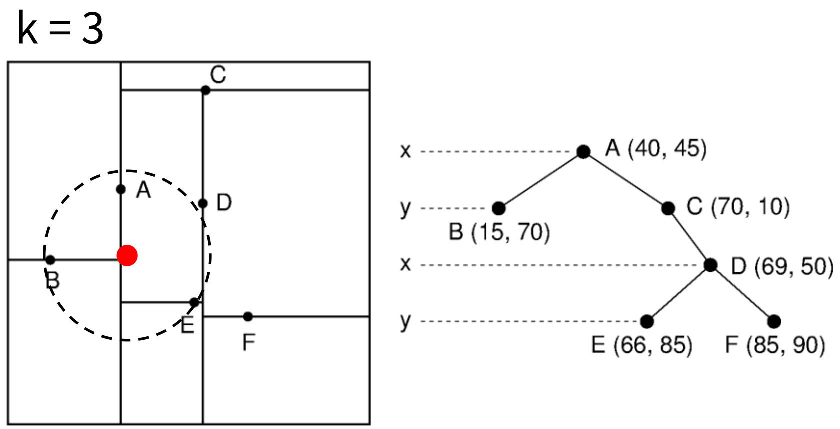


Time-restricted K-Nearest Neighbor Search (TKNN Search)

Efficiently find nearest neighbor **within specific time window**

Preliminaries : Related Work





Time-restricted K-Nearest Neighbor Search (TKNN Search)

Efficiently find nearest neighbor **within specific time window**

- **Spatio-Temporal K-Nearest Neighbor Search (STKNN Search)**

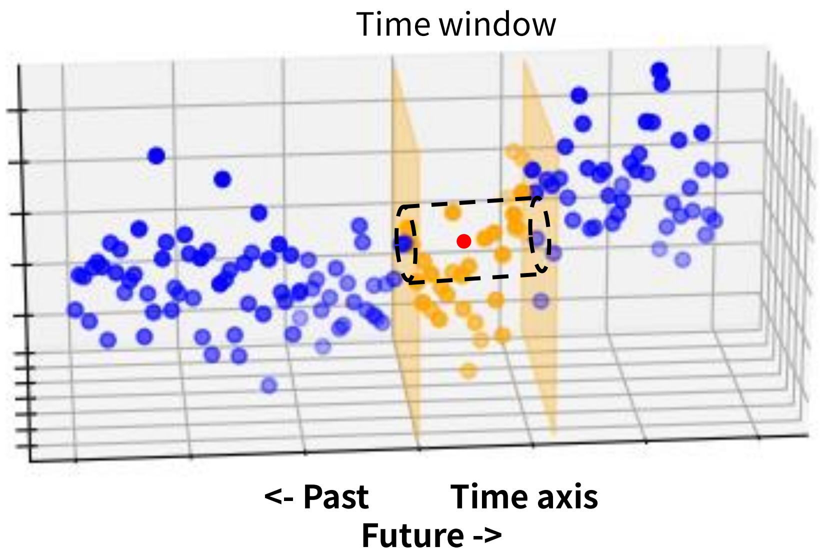
Indexing **temporal** and **spatial** information

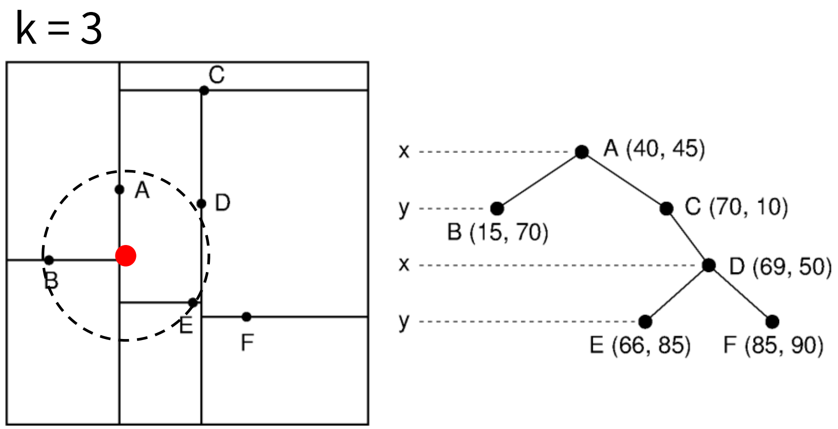
Can process **temporal** or **spatial** queries

Quad-tree based, R* tree based, …

Focused on processing queries in 2~3D (+time axis)

Preliminaries : Related Work





Time-restricted K-Nearest Neighbor Search (TKNN Search)

Efficiently find nearest neighbor **within specific time window**

- **Spatio-Temporal K-Nearest Neighbor Search (STKNN Search)**

  Indexing **temporal** and **spatial** information

  Can process **temporal** or **spatial** queries

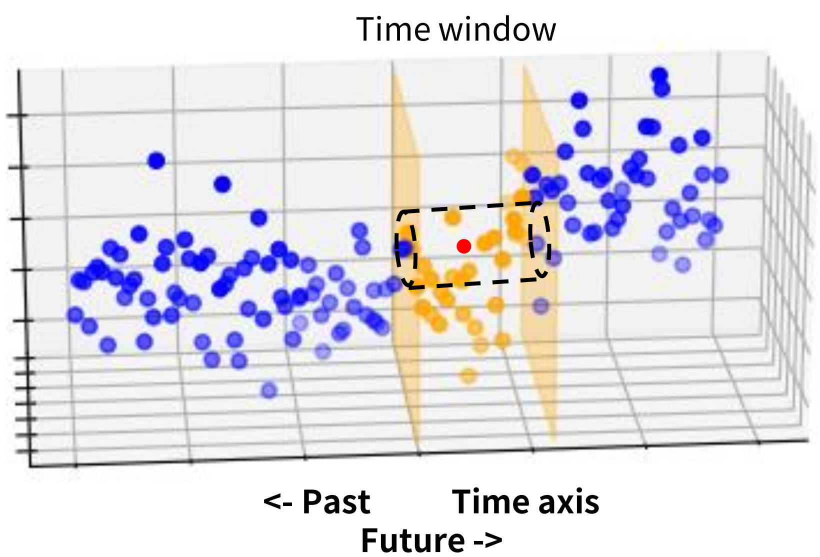
  Quad-tree based, R* tree based, …

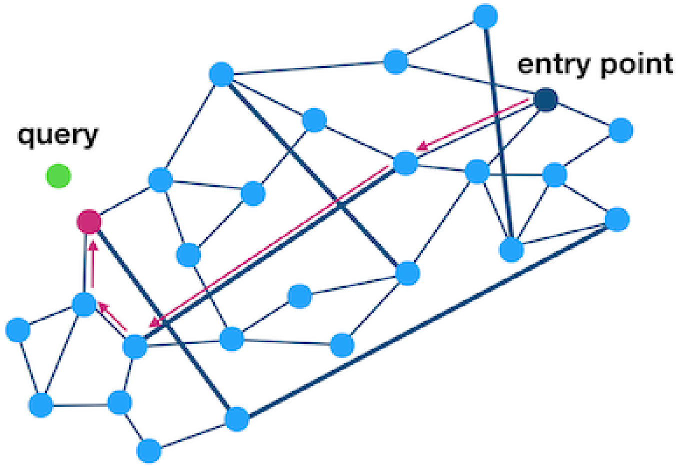
  Focused on processing queries in 2~3D (+time axis)

  ➔ **Tree based: Curse of dimensionality**

  Not suitable for high-dimensional data

Preliminaries : Related Work





Time-restricted K-Nearest Neighbor Search (TKNN Search)

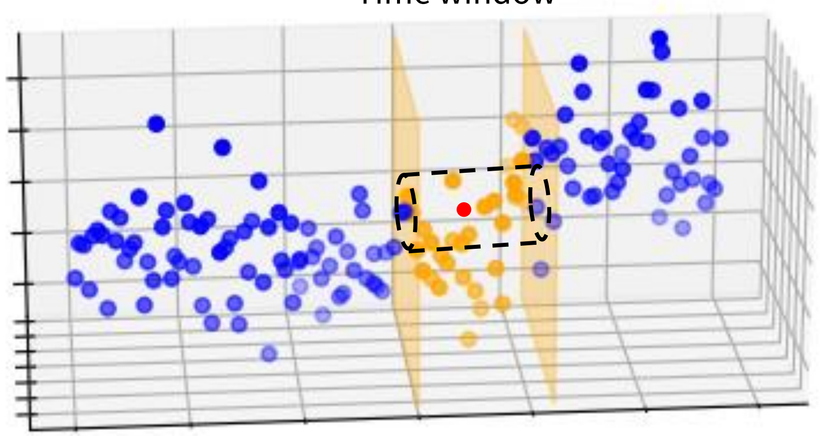
Efficiently find nearest neighbor **within specific time window**

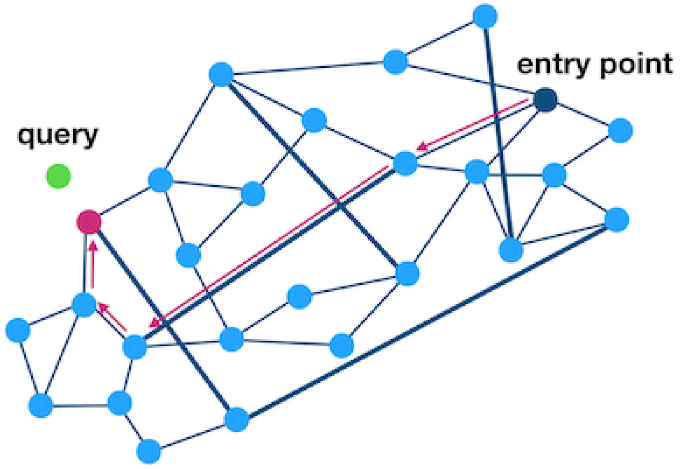
- **Approaches suitable for high-dimensional data?**

  Graph-based, PQ based, ⋯

  Scarcely researched

Preliminaries : Related Work





Time-restricted K-Nearest Neighbor Search (TKNN Search)

Efficiently find nearest neighbor **within specific time window**

- **Approaches suitable for high-dimensional data?**

  Graph-based, PQ based, …

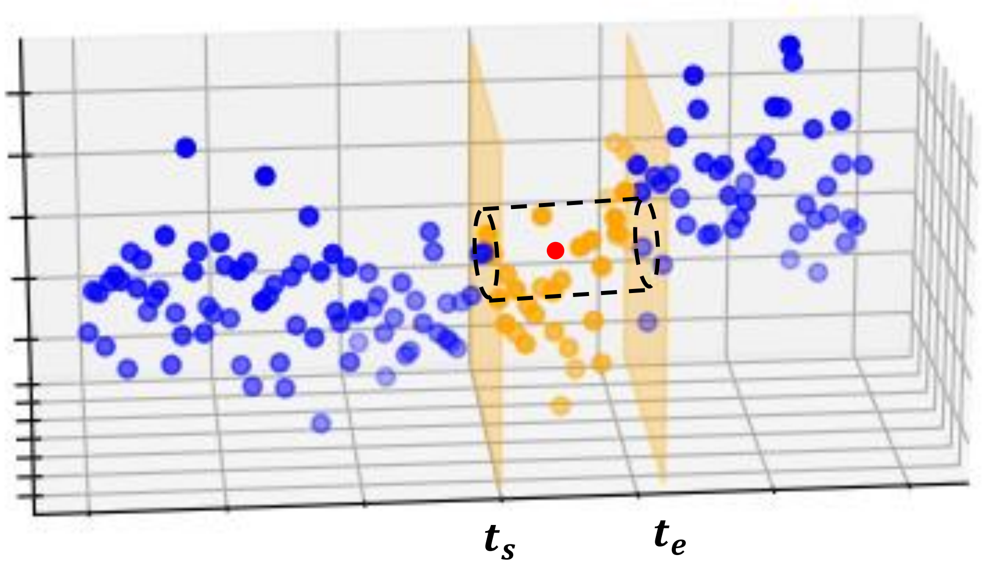
  Scarcely researched

  → Multi-level Block Indexing (MBI)

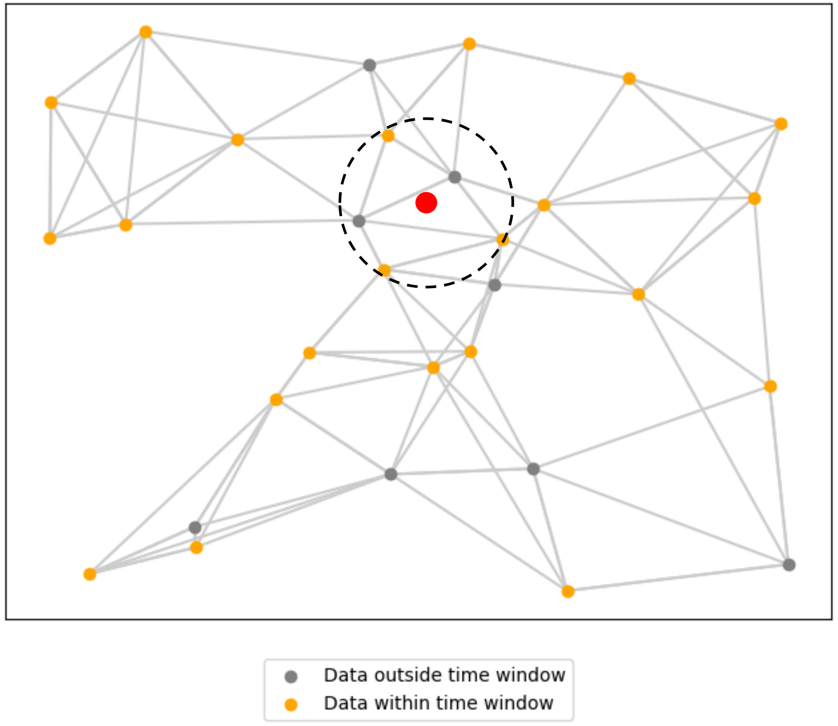
Index

Preliminaries : Simple Approaches for TKNN Search

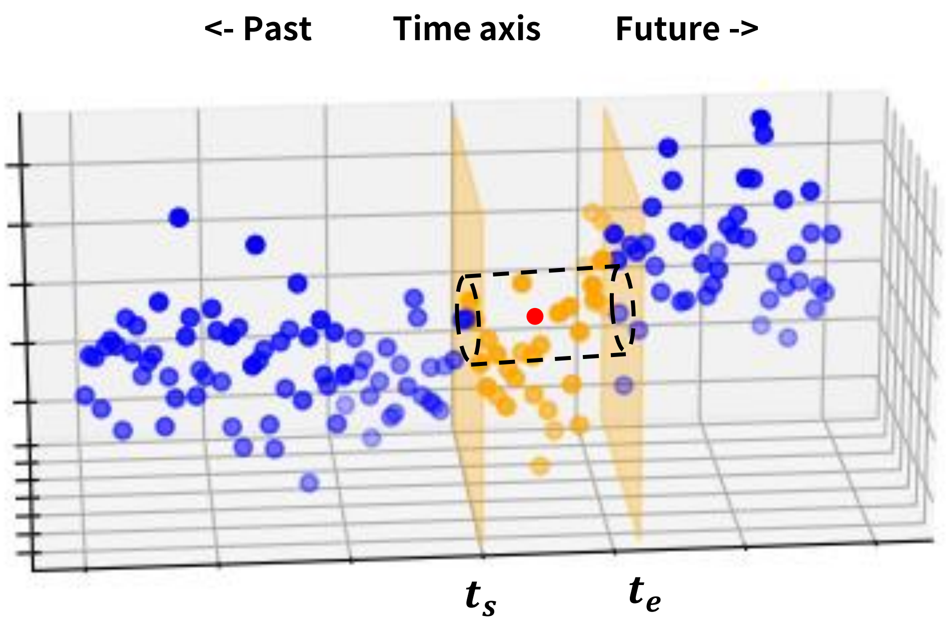
Binary Search and Brute-Force (BSBF)



Search and Filtering (SF)



Preliminaries : Simple Approaches for TKNN Search



Binary Search and Brute-Force (BSBF)

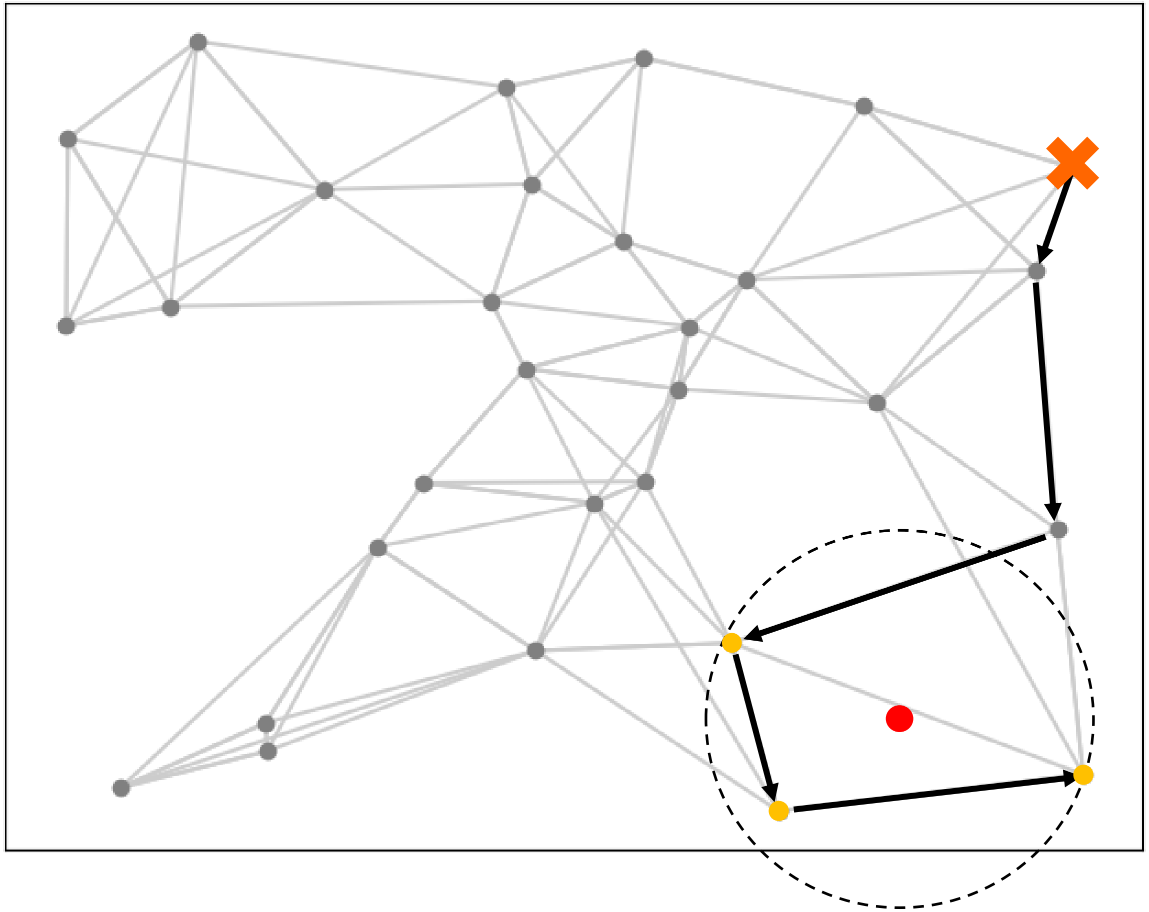
1. Find first item in time window using Binary search

2. Check all items within the time window

Performance depends size of time window

Narrow time window ✔ Wide time window ✖

Preliminaries : Simple Approaches for TKNN Search (SF)



Graph-based KNN Search

a. Start at random data point of NN Graph

b. Iteratively,

   1) Move towards the neighbor that becomes closest to the query point

   2) Update KNN result.

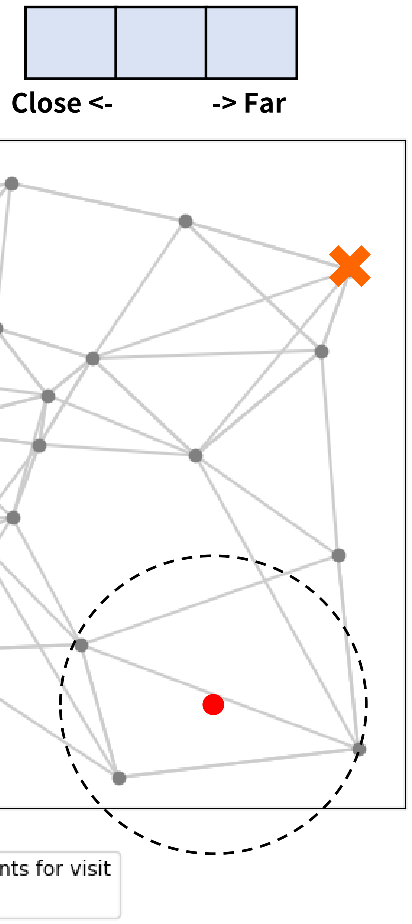
c. Until no longer possible to update the results.

Preliminaries : Simple Approaches for TKNN Search (SF)

- Start at random data point.

K = 3

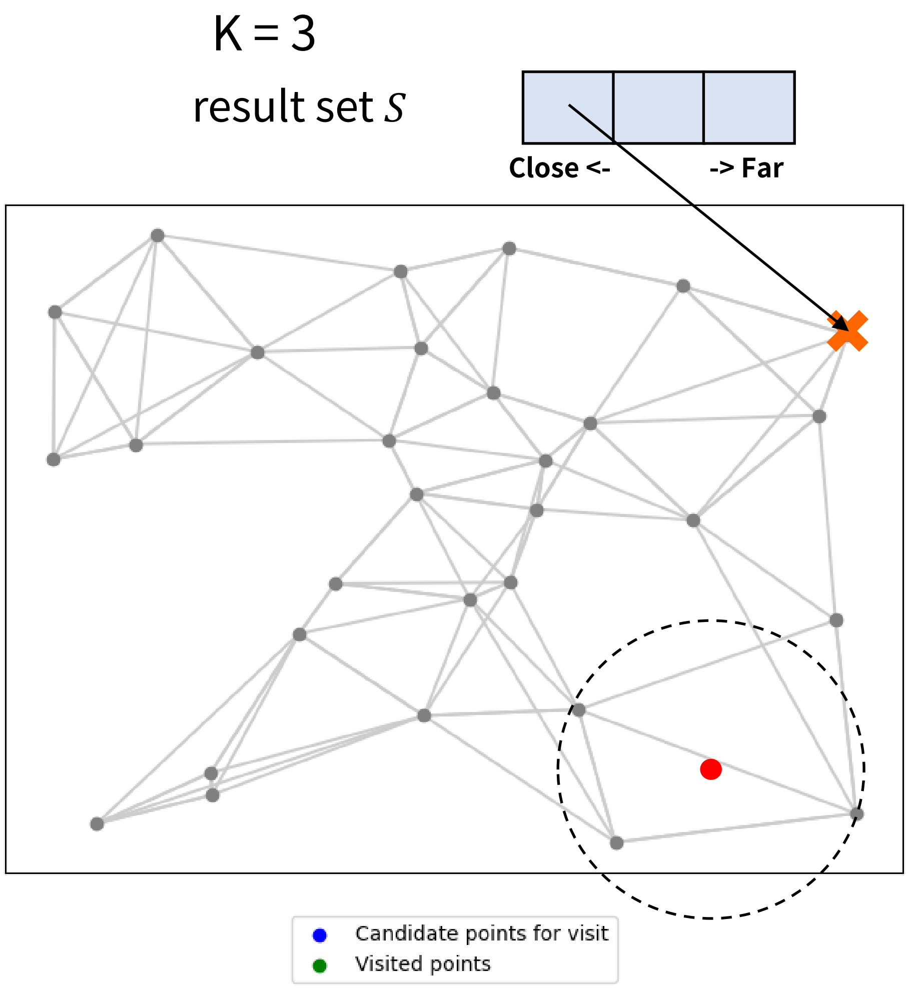
result set S



Preliminaries : Simple Approaches for TKNN Search (SF)

Graph-based KNN Search

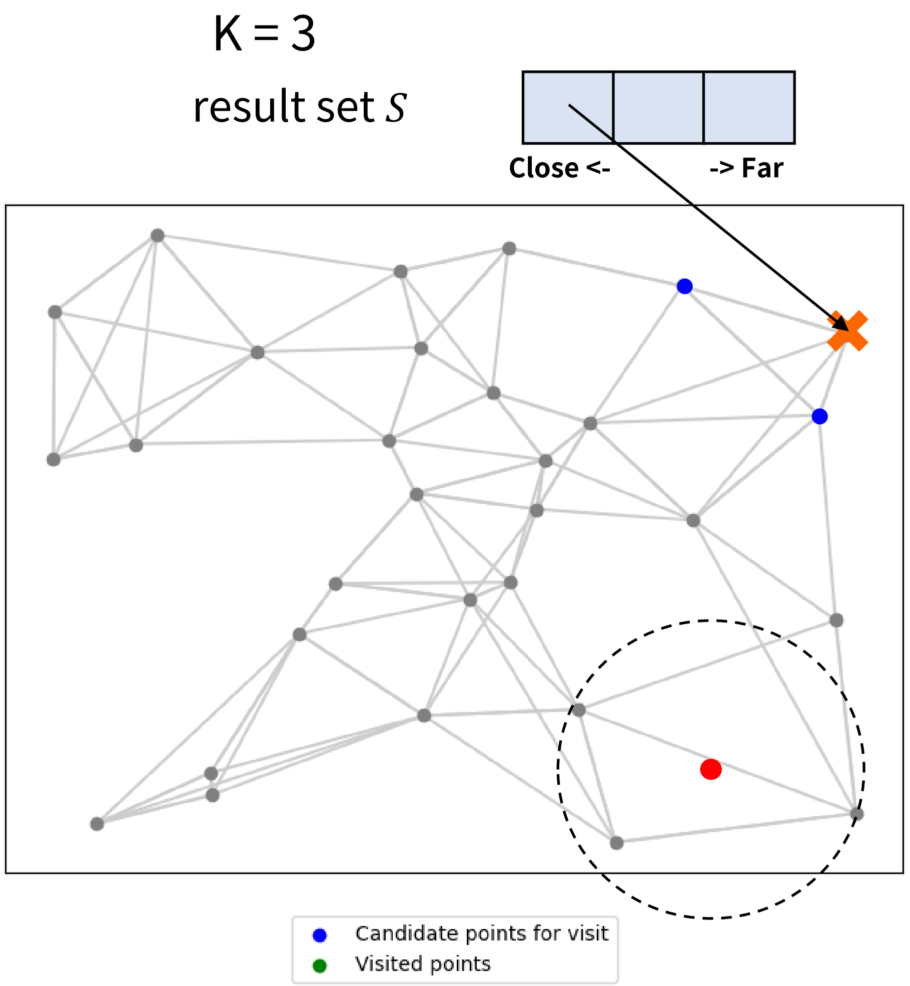
- Update result set by current point.



Preliminaries : Simple Approaches for TKNN Search (SF)

Graph-based KNN Search

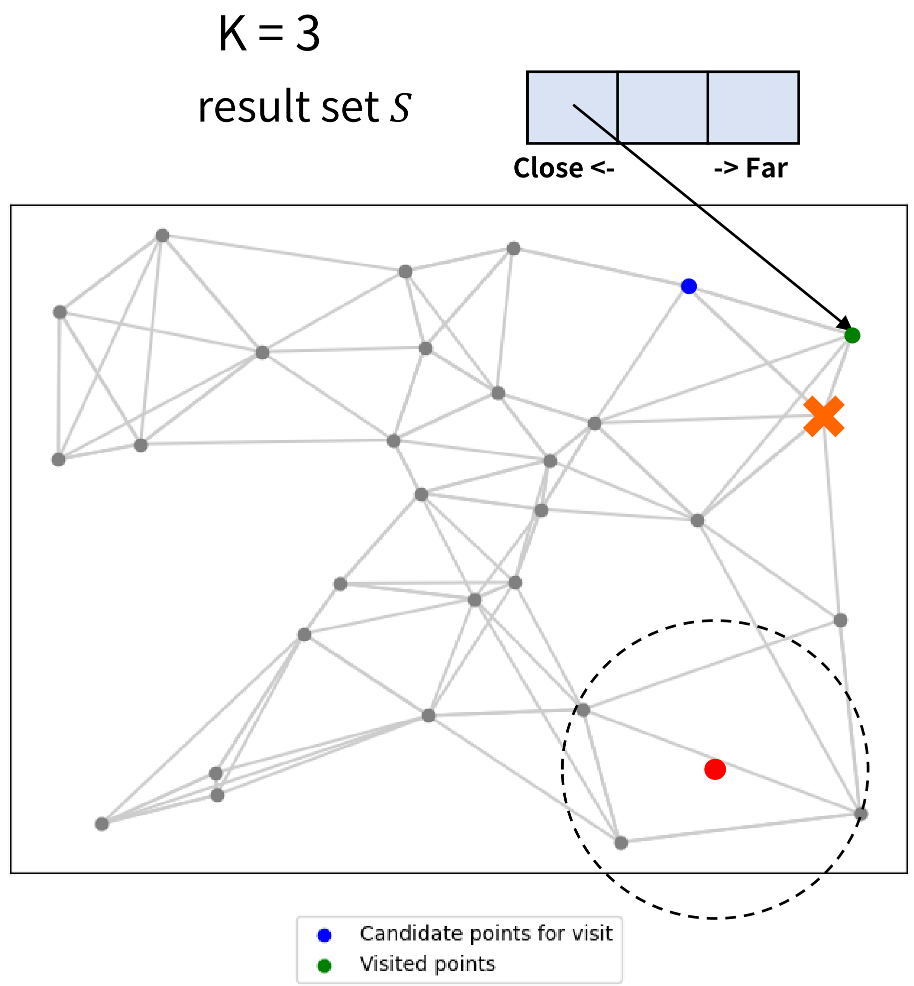
- Add neighbors of the current point to the list of candidates for visiting.



Preliminaries : Simple Approaches for TKNN Search (SF)

Graph-based KNN Search

- Select the closest data point to the query point from the candidates for visit and move towards it.



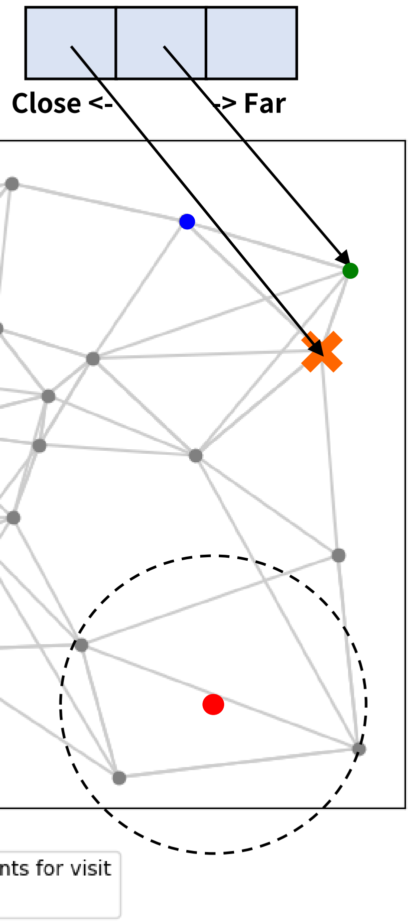
Preliminaries : Simple Approaches for TKNN Search (SF)

Graph-based KNN Search

- Update result set by current point.

K = 3

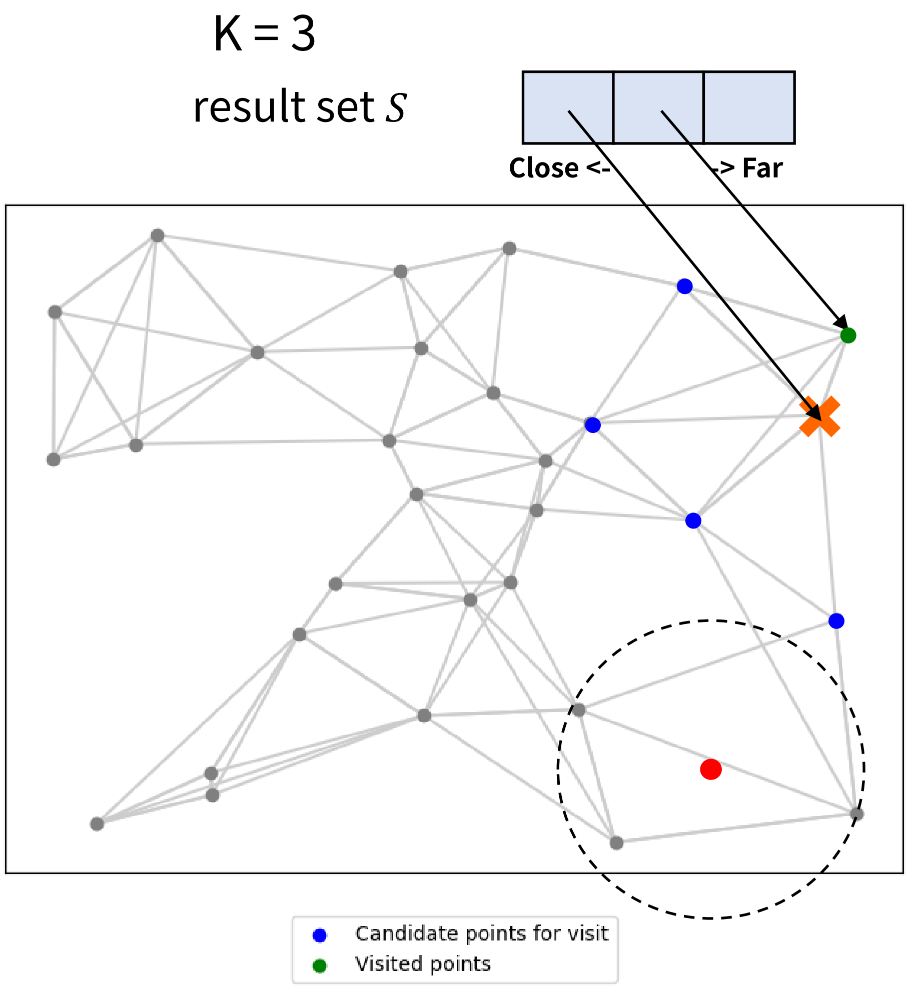
result set S



Preliminaries : Simple Approaches for TKNN Search (SF)

Graph-based KNN Search

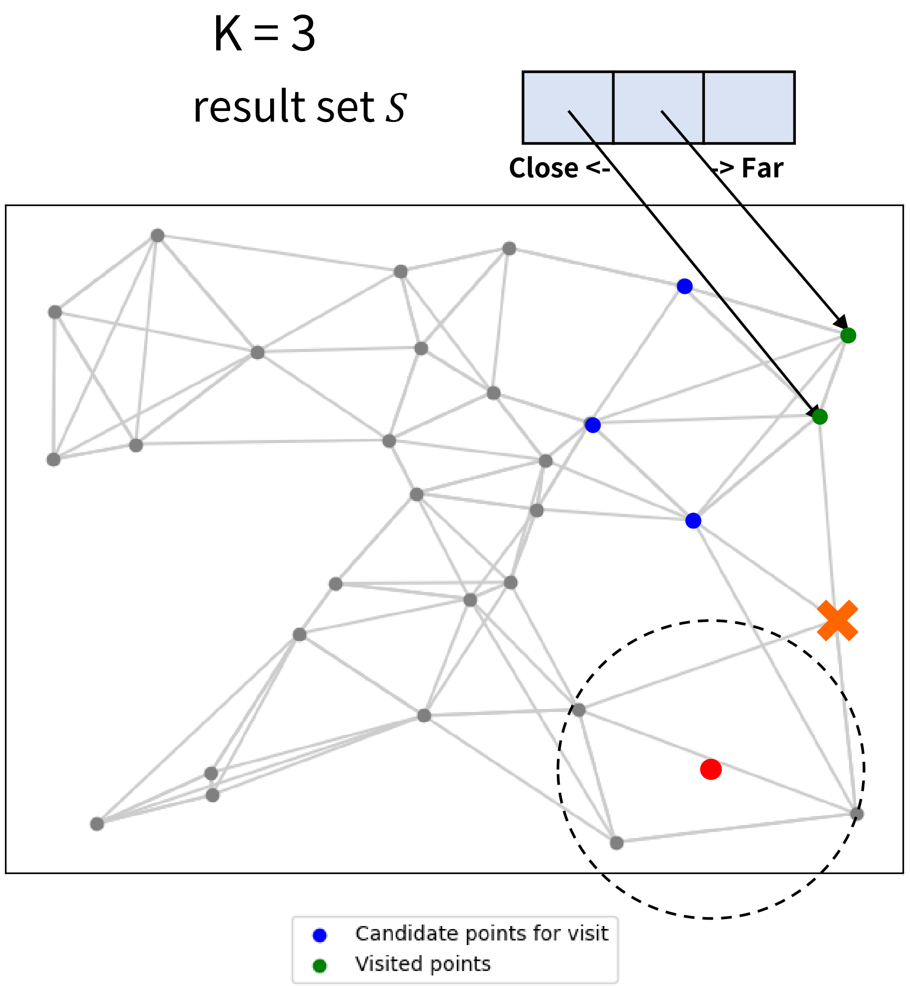
- Add neighbors of the current point to the list of candidates for visiting.



Preliminaries : Simple Approaches for TKNN Search (SF)

Graph-based KNN Search

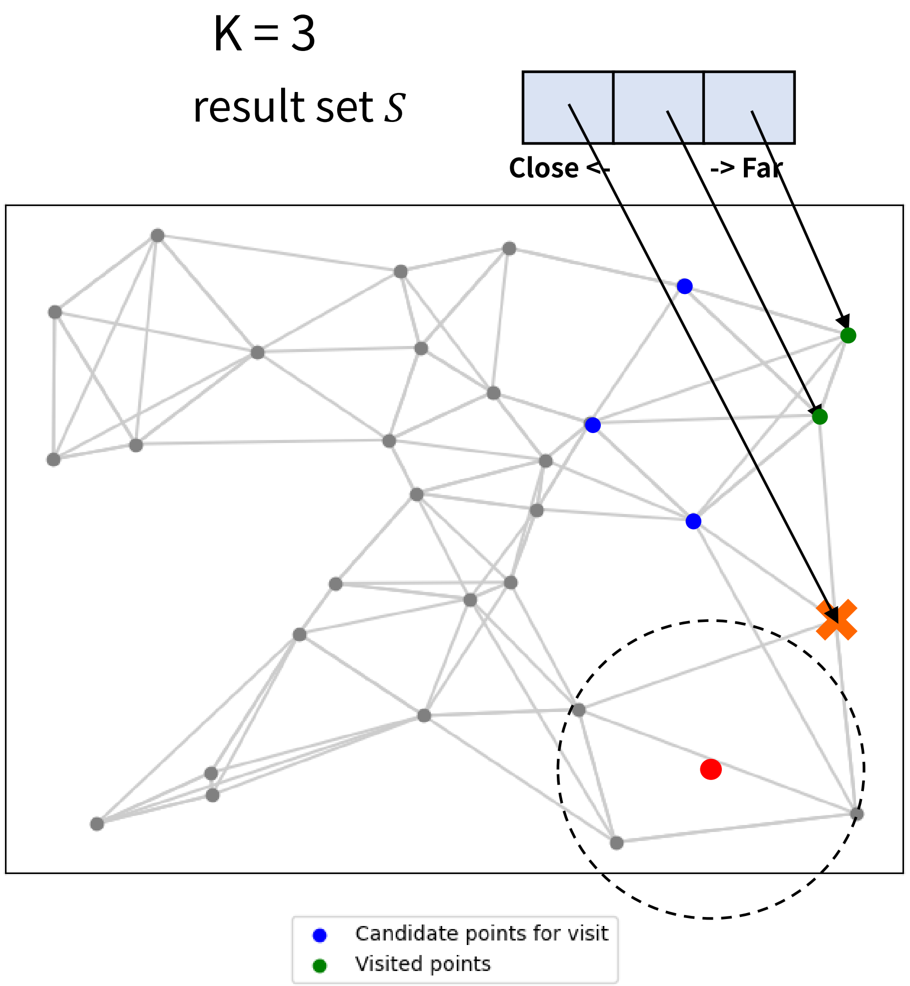
- Select the closest data point to the query point from the candidates for visit and move towards it.



Preliminaries : Simple Approaches for TKNN Search (SF)

Graph-based KNN Search

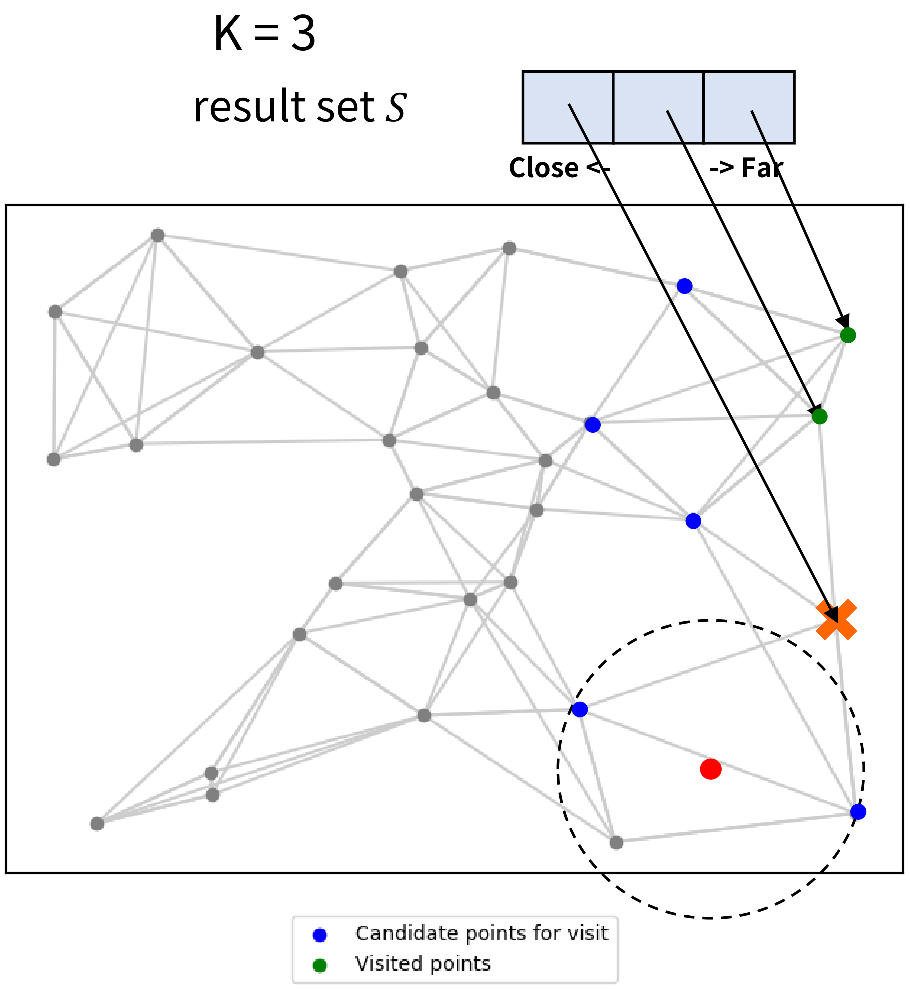
- Update result set by current point.



Preliminaries : Simple Approaches for TKNN Search (SF)

Graph-based KNN Search

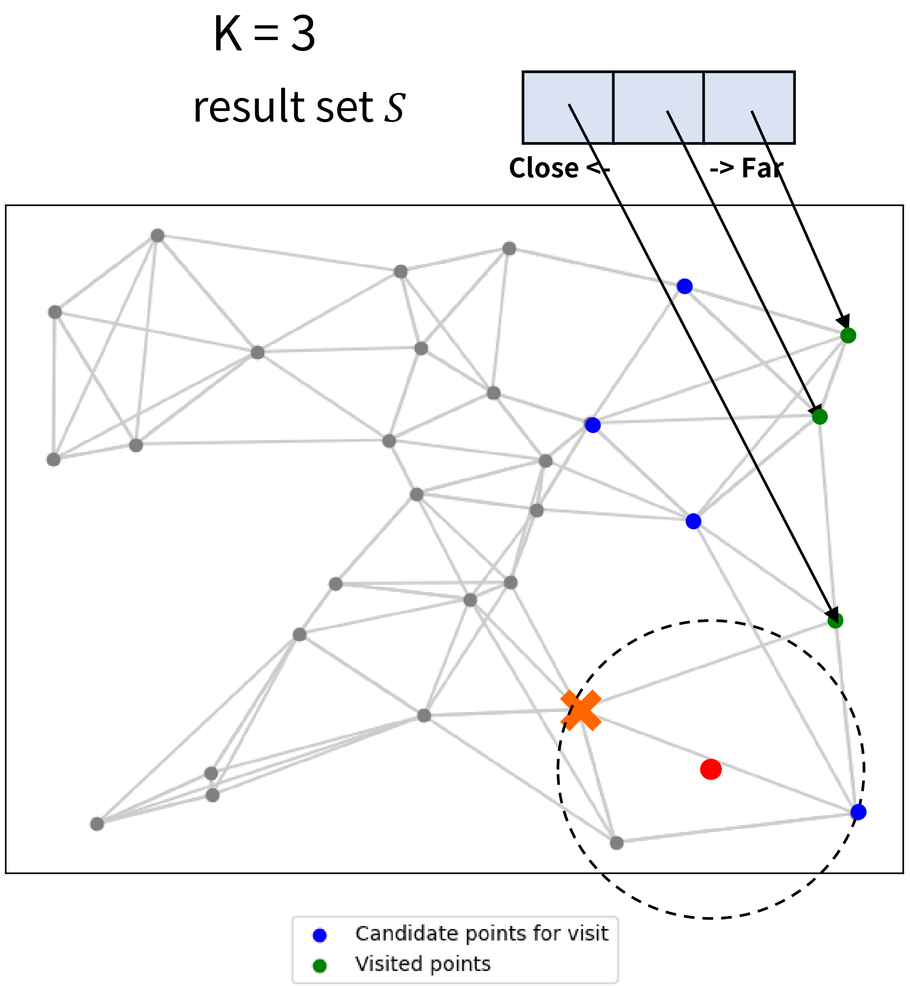
- Add neighbors of the current point to the list of candidates for visiting.



Preliminaries : Simple Approaches for TKNN Search (SF)

Graph-based KNN Search

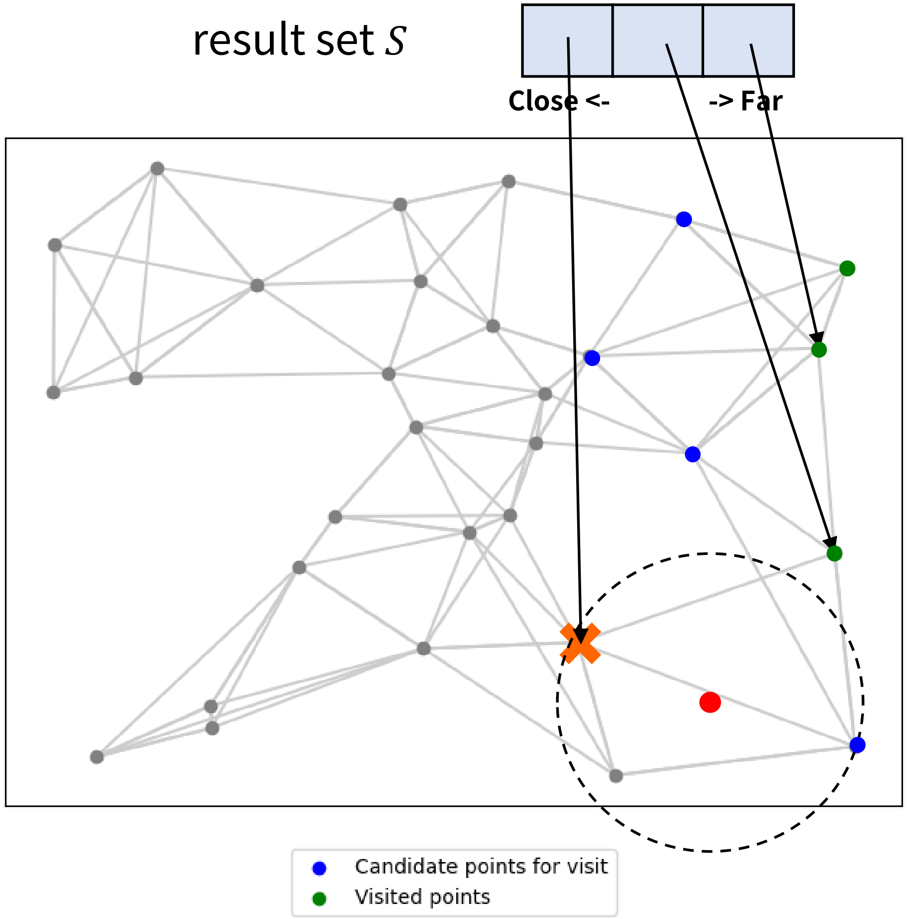
- Select the closest data point to the query point from the candidates for visit and move towards it.



Preliminaries : Simple Approaches for TKNN Search (SF)

K = 3

result set S



Graph-based KNN Search

- Update result set by current point.

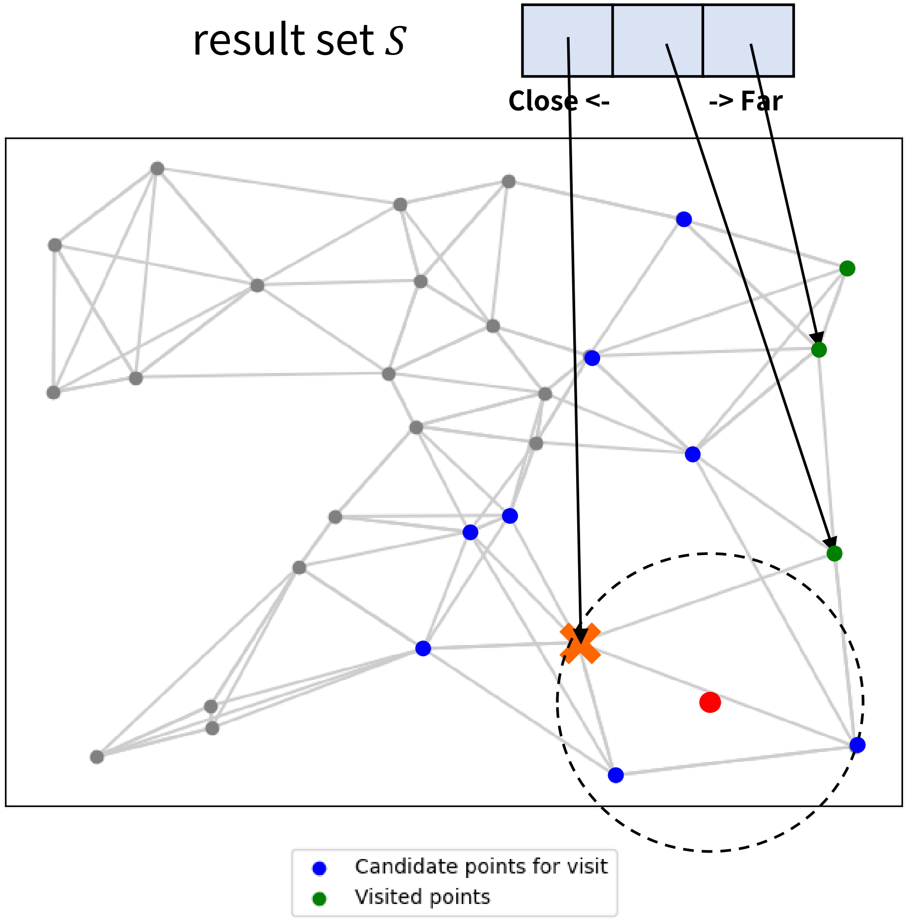
Preliminaries : Simple Approaches for TKNN Search (SF)

Graph-based KNN Search

- Add neighbors of the current point to the list of candidates for visiting.

K = 3

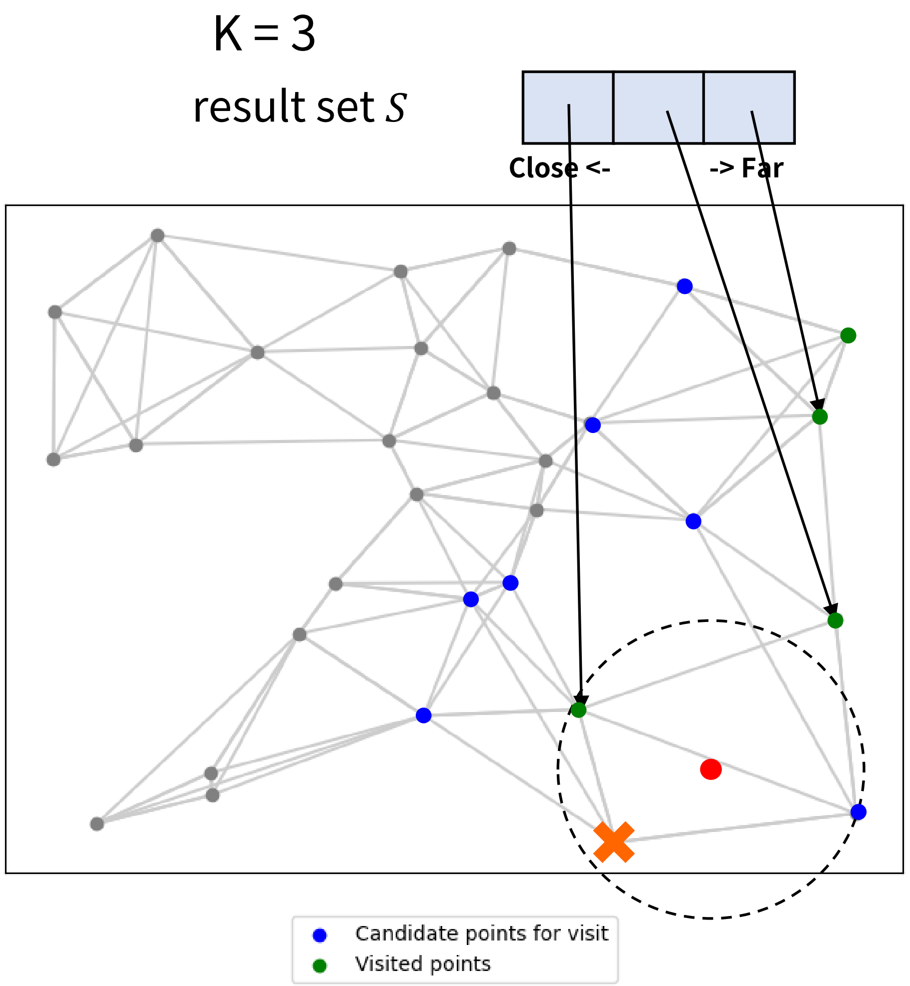
result set S



Preliminaries : Simple Approaches for TKNN Search (SF)

Graph-based KNN Search

- Select the closest data point to the query point from the candidates for visit and move towards it.



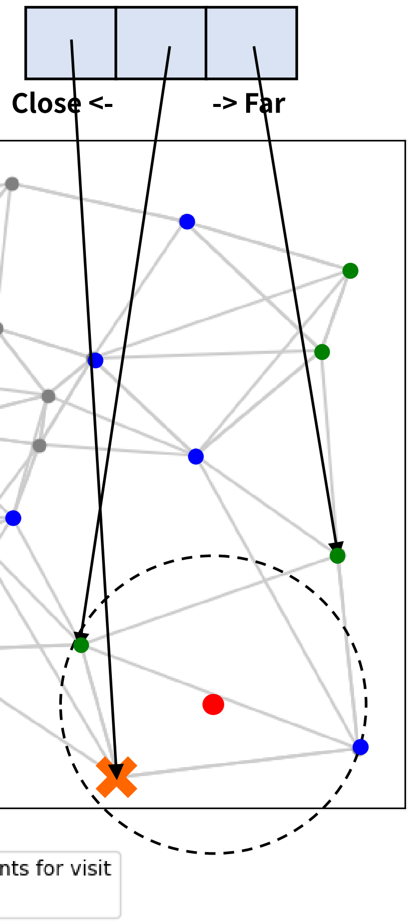
Preliminaries : Simple Approaches for TKNN Search (SF)

Graph-based KNN Search

- Update result set by current point.

K = 3

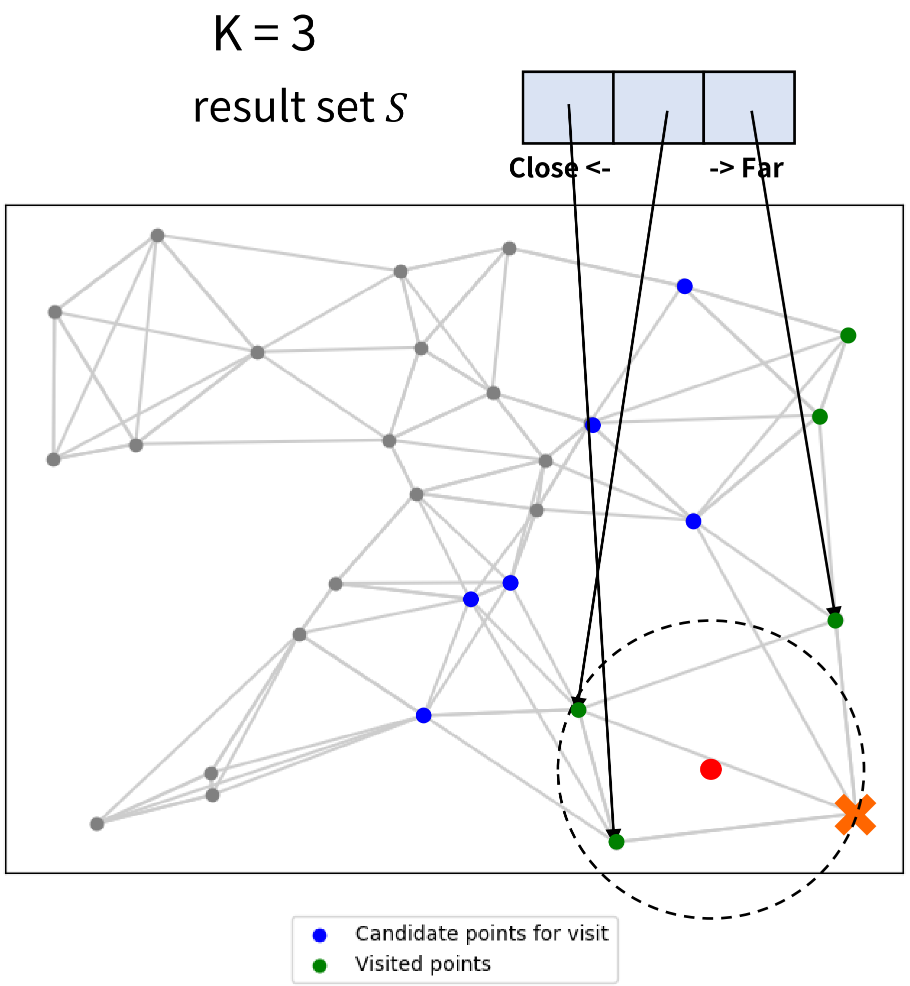
result set S



Preliminaries : Simple Approaches for TKNN Search (SF)

Graph-based KNN Search

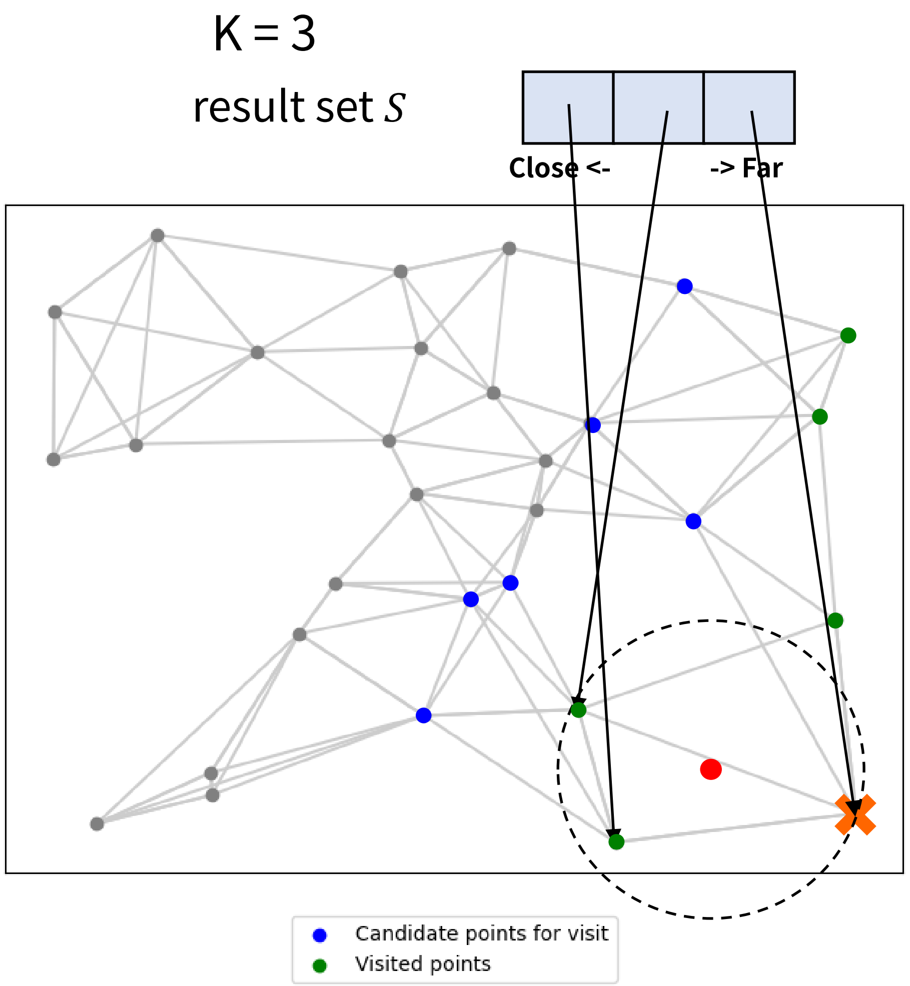
- Select the closest data point to the query point from the candidates for visit and move towards it.



Preliminaries : Simple Approaches for TKNN Search (SF)

Graph-based KNN Search

- Update result set by current point.



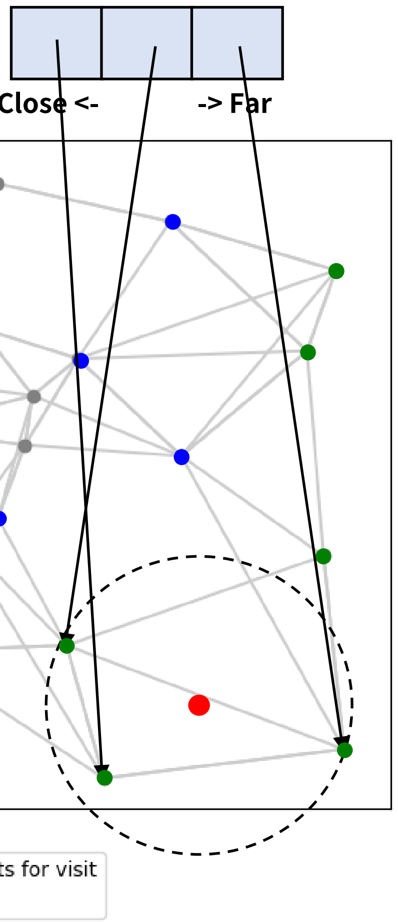
Preliminaries : Simple Approaches for TKNN Search (SF)

Graph-based KNN Search

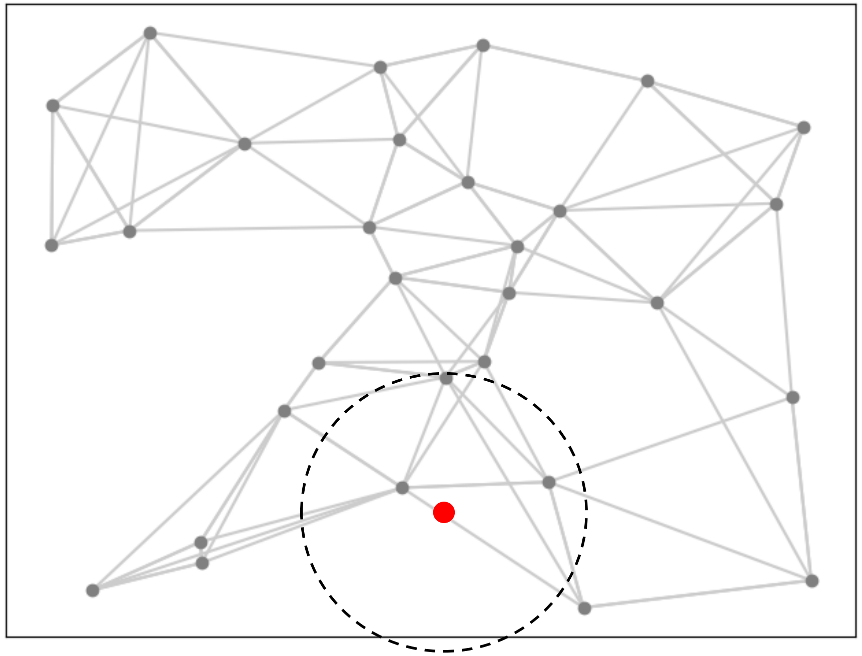
- Select the closest data point to the query point from the candidates for visit and move towards it.
- **Stopped** because it moved much farther away than current results.

K = 3

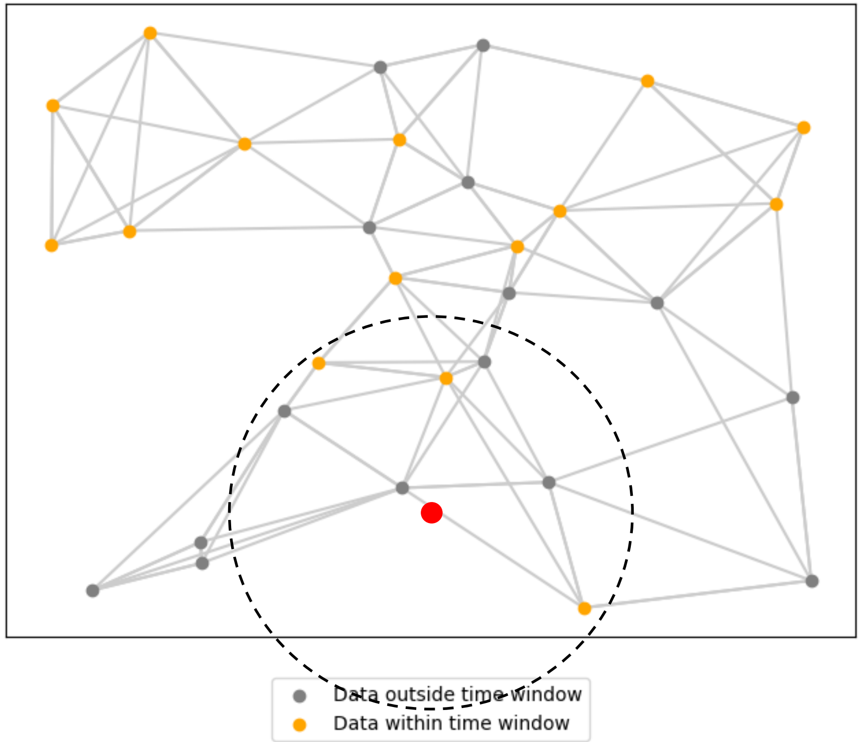
result set S



Preliminaries : Simple Approaches for TKNN Search (SF)



Within time window



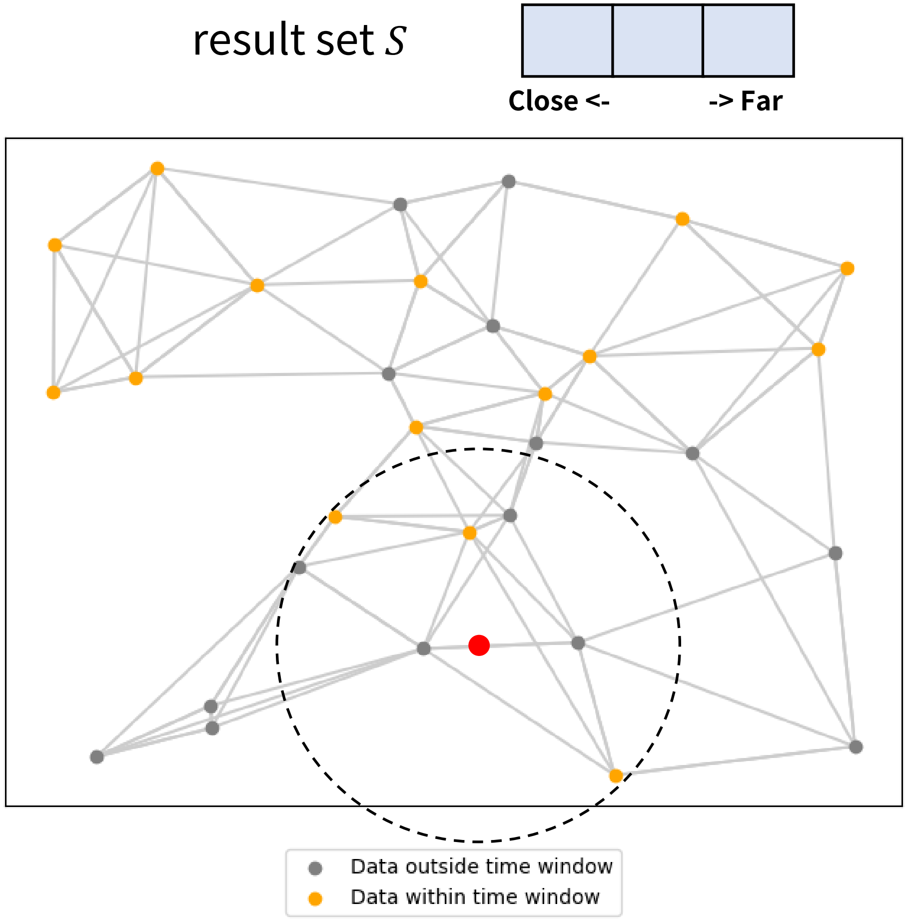
Graph-based KNN Search

Search and Filtering: TKNN Search

Preliminaries : Simple Approaches for TKNN Search (SF)

K = 3

result set S



Search and Filtering (SF)

- Update result set by current point.

⬇

- Update result set by current point, **when current point is in time window.**

Preliminaries

2. Simple approaches for TKNN Search

2.2 Search and Filtering (SF)

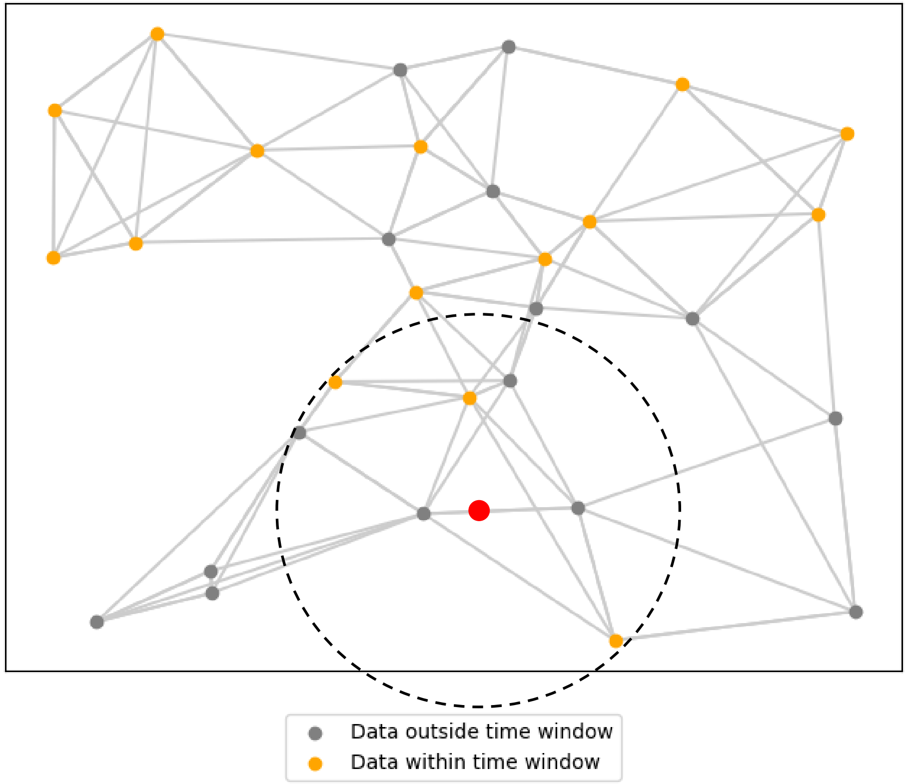
Graph-based KNN Search + Filtering

- Graph-based
  - ➔ Suitable for high-dimensional data
  - ➔ Search Speed ✓

K = 3

result set S

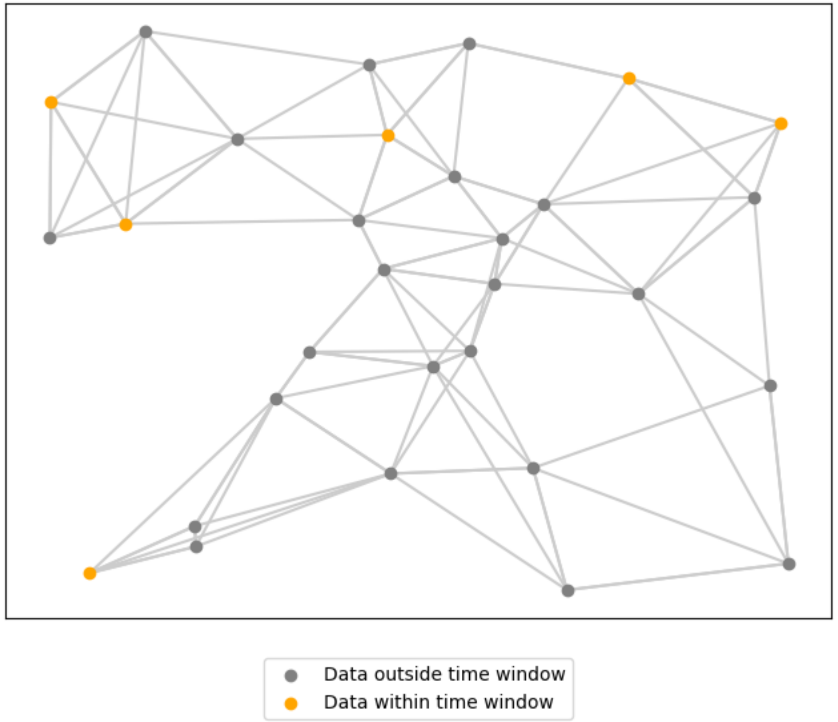
Close <- -> Far



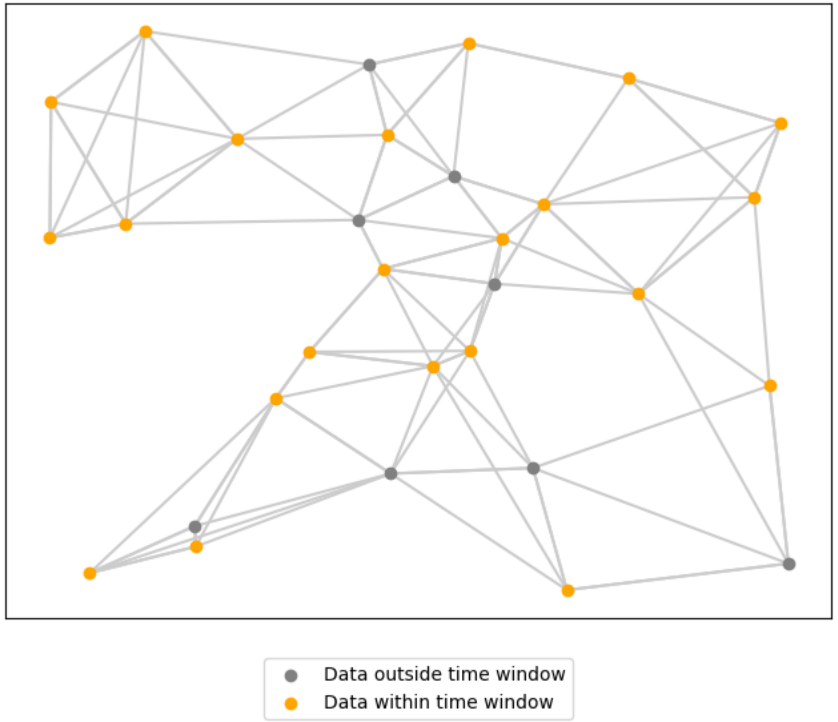
Preliminaries

2. Simple approaches for TKNN Search

2.2 Search and Filtering (SF)



Narrow time window



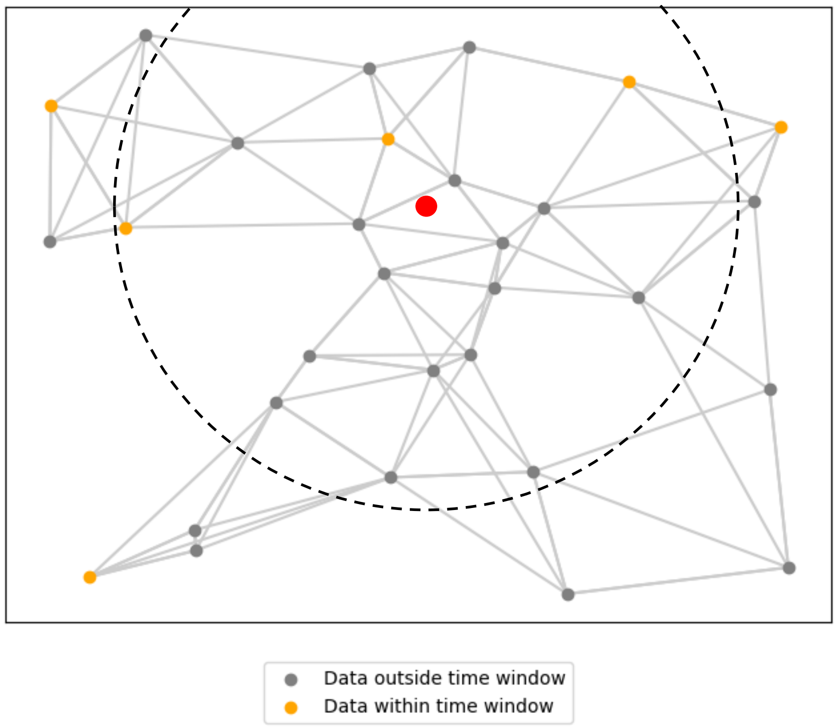
Wide time window

Preliminaries

2. Simple approaches for TKNN Search

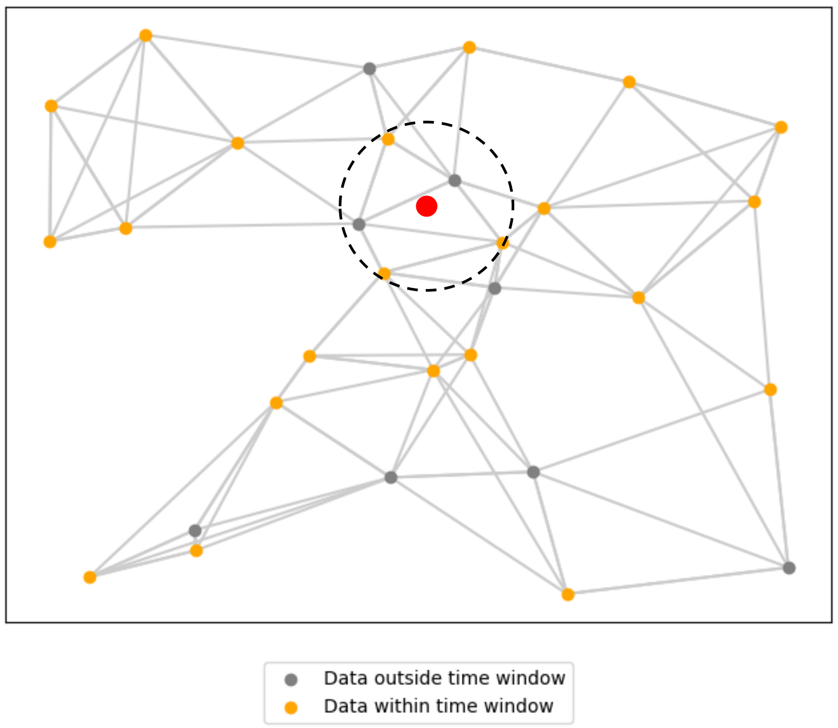
2.2 Search and Filtering (SF)

K = 3



Narrow time window

Visit at least 19 Nodes



Wide time window

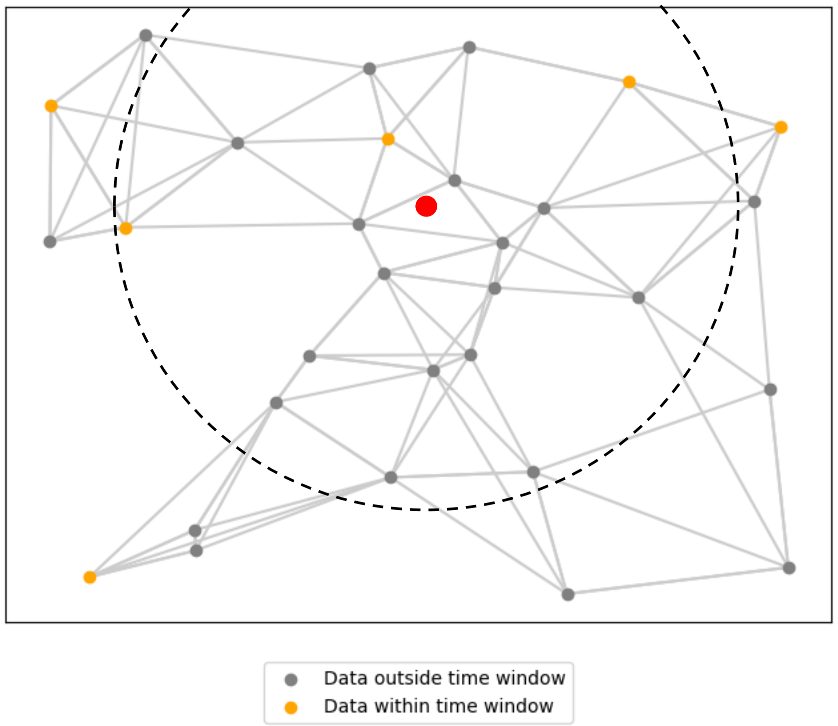
Visit at least 5 Nodes

Preliminaries

2. Simple approaches for TKNN Search

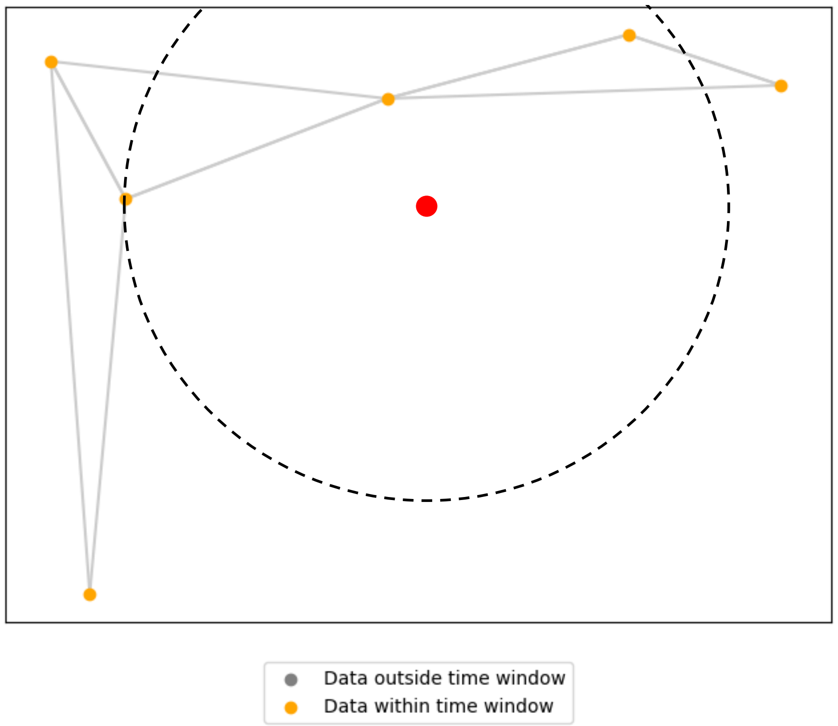
2.2 Search and Filtering (SF)

K = 3



Narrow time window

Visit at least 19 Nodes



Narrow time window (ideal)

Visit at least 3 Nodes

Preliminaries

2. Simple approaches for TKNN Search

Binary Search and Brute-Force (BSBF)

Narrow time window ✓

Wide time window ✗

Search and Filtering (SF)

Narrow time window ✗

Wide time window ✓

Preliminaries

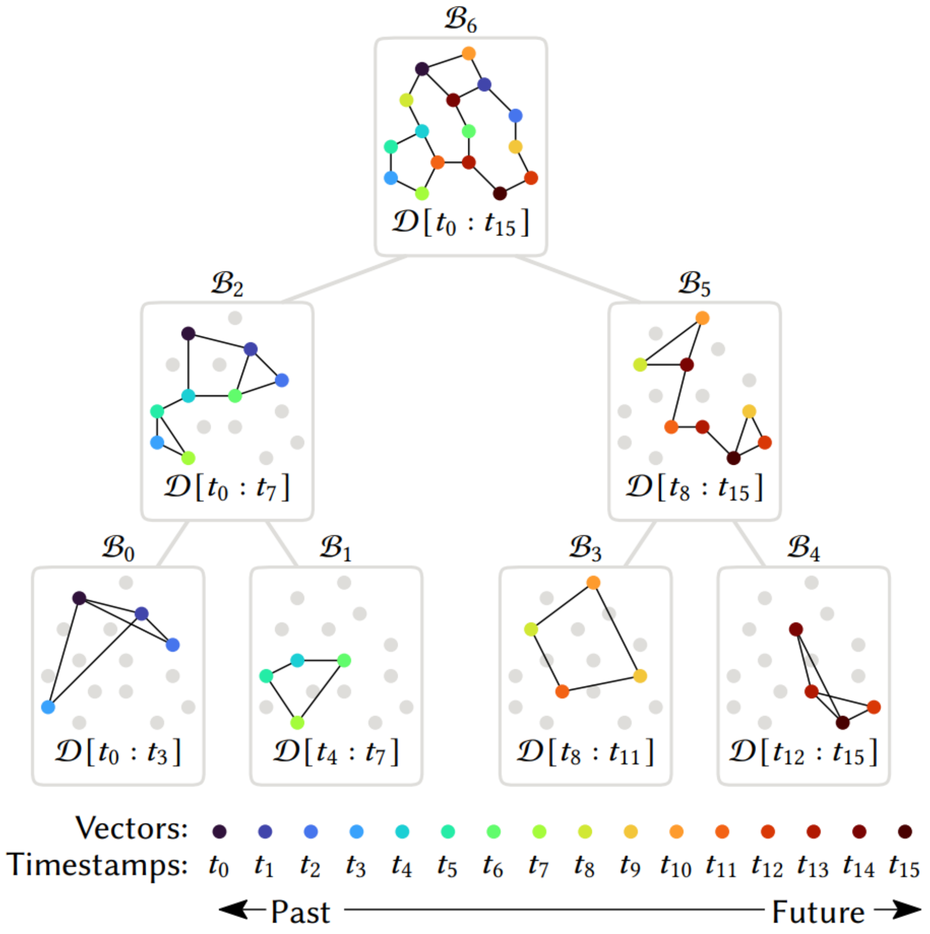
2. Simple approaches for TKNN Search

time window	Binary Search and Brute-Force (BSBF)	Search and Filtering (SF)	Multi-level Block Indexing (MBI)
Narrow	✓	✗	✓
Wide	✗	✓	✓

Index

Proposed Method

1. Overview of MBI

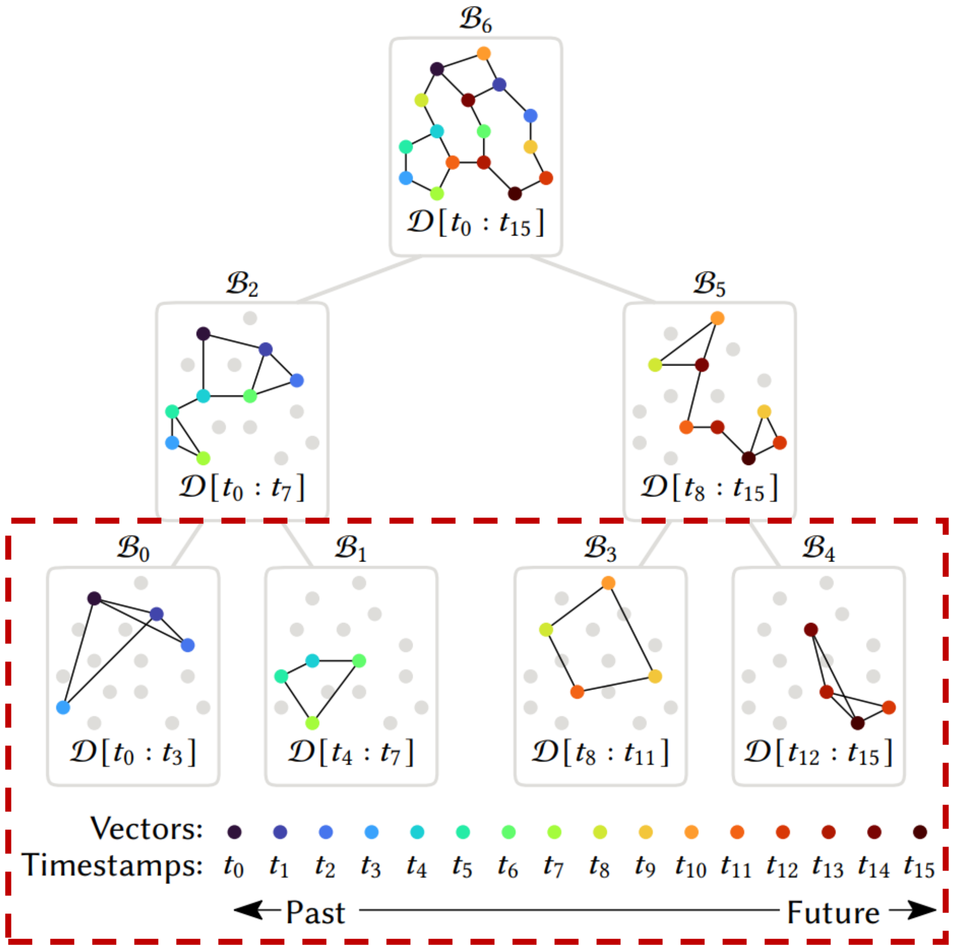


Key idea

Binary tree of **SF** blocks

Proposed Method

1. Overview of MBI

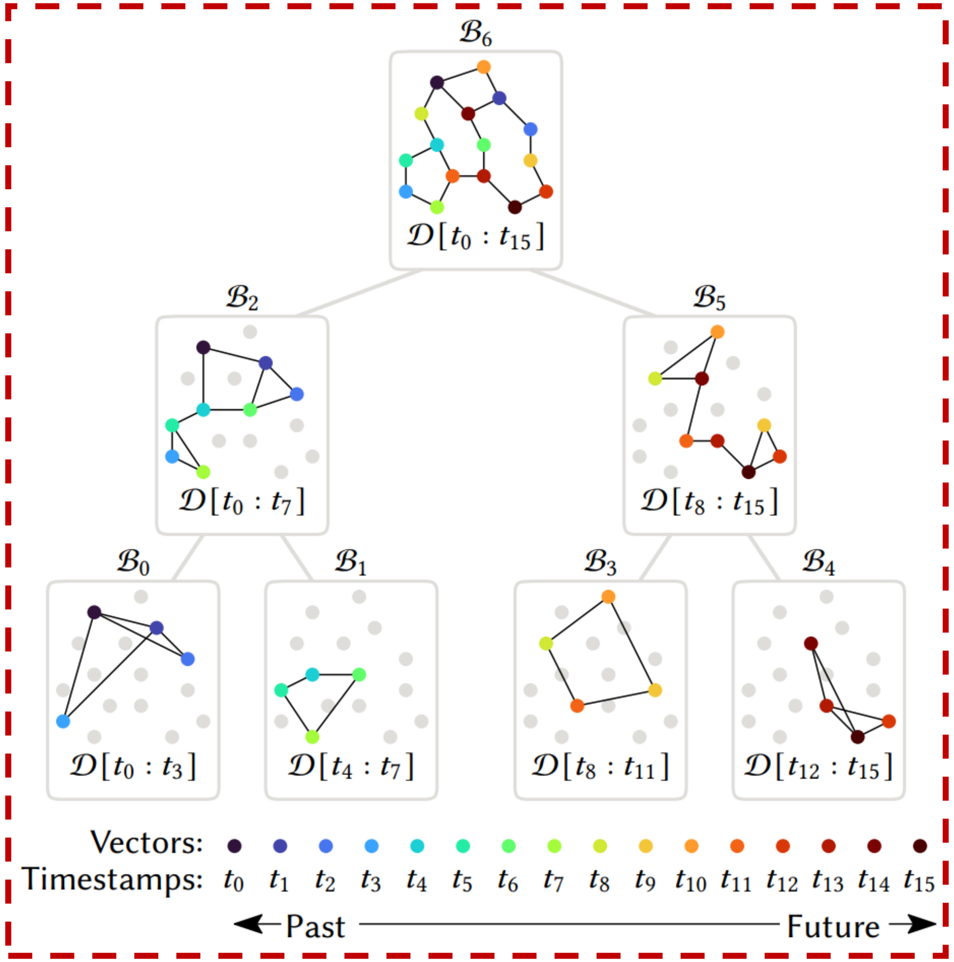


Build binary tree

Divide blocks based on time

Proposed Method

1. Overview of MBI



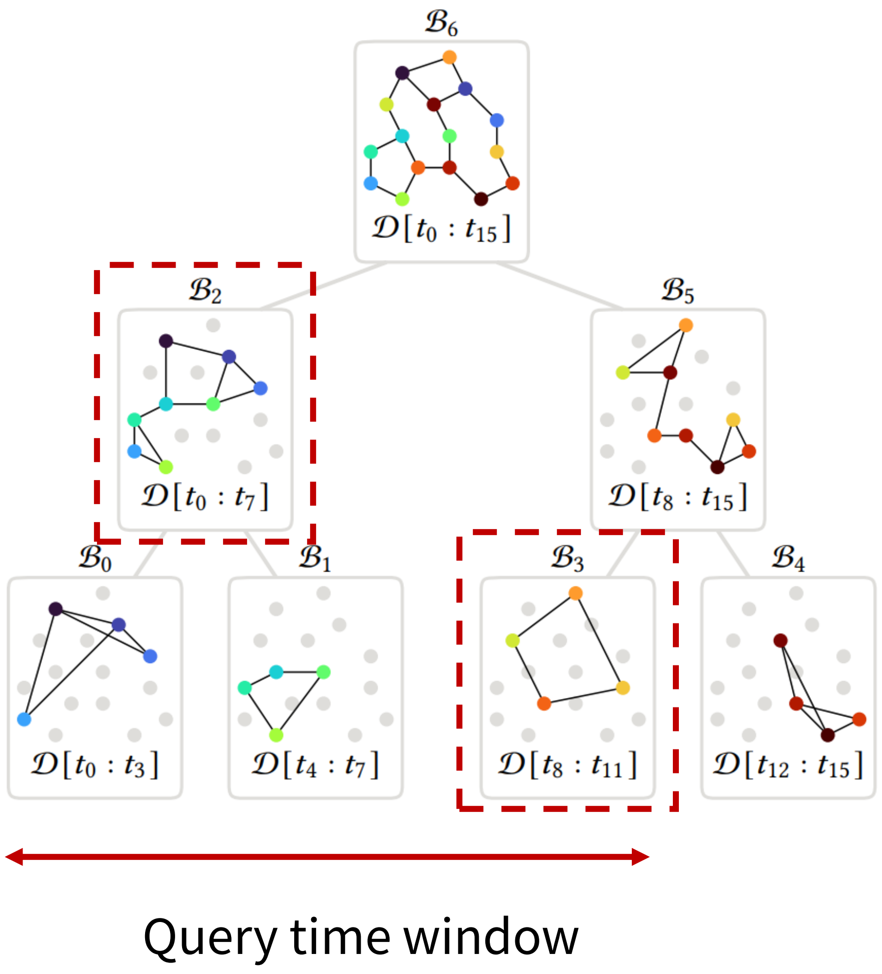
Build binary tree

Divide blocks based on time

Higher level -> Wide time range

Proposed Method

1. Overview of MBI

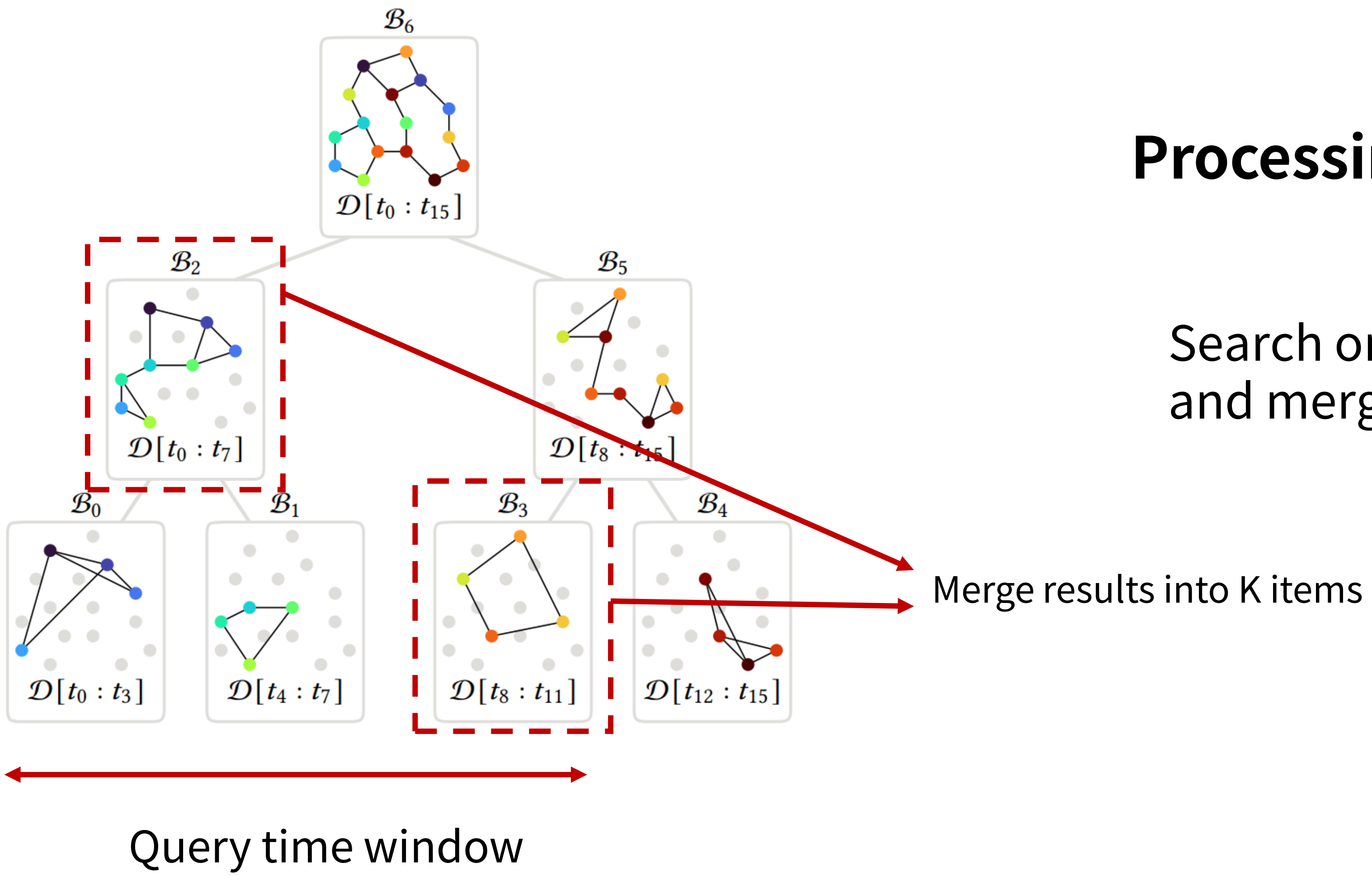


Processing query

Select blocks corresponding to the query time window

Proposed Method

1. Overview of MBI



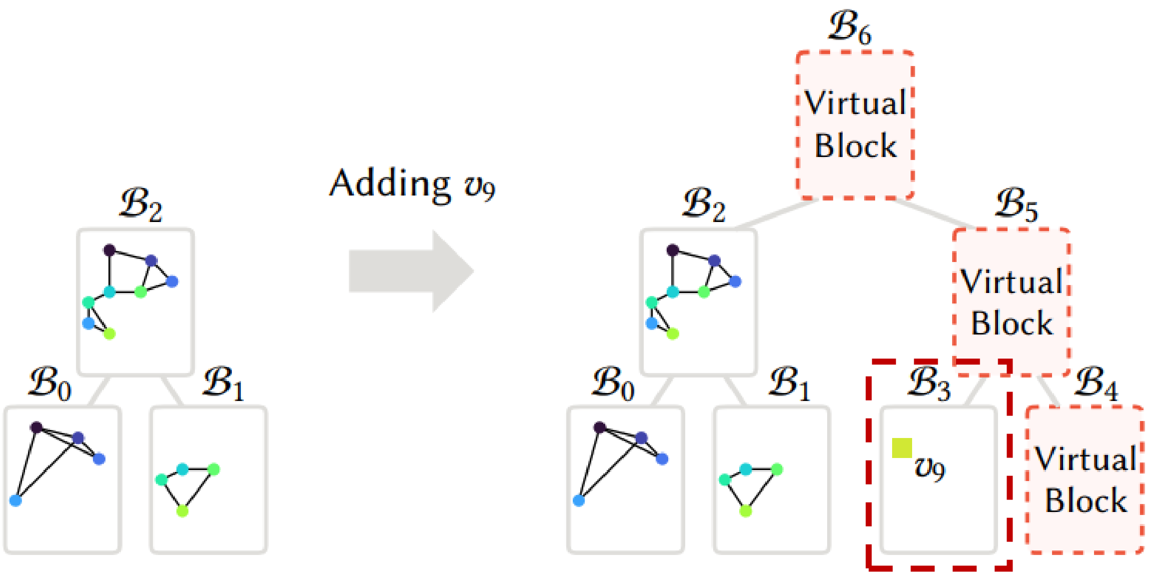
Processing query

Search on each blocks,
and merge results.

Index

Proposed Method

2. Insertion and Indexing process of MBI



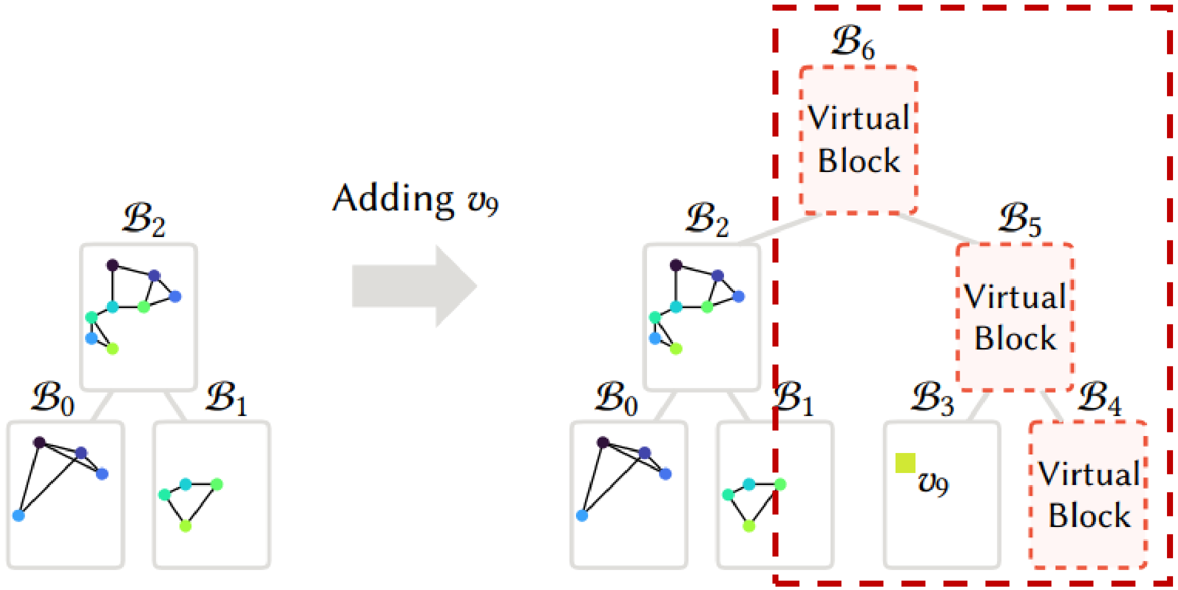
Insertion of data

Insert to new node

(Do not build KNN Graph yet)

Proposed Method

2. Insertion and Indexing process of MBI



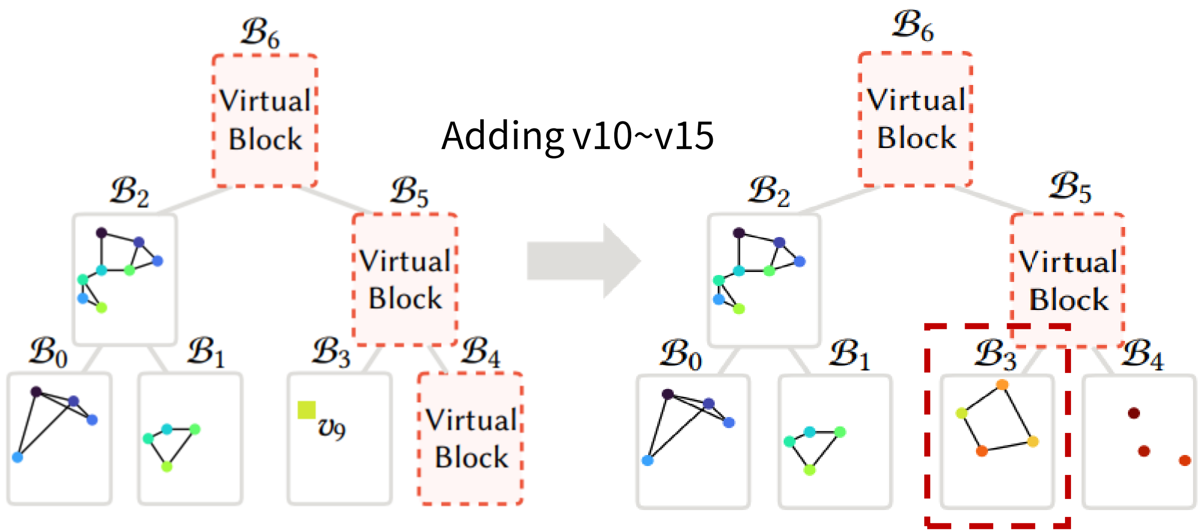
Insertion of data

Create virtual nodes

to maintain perfect binary tree structure

Proposed Method

2. Insertion and Indexing process of MBI



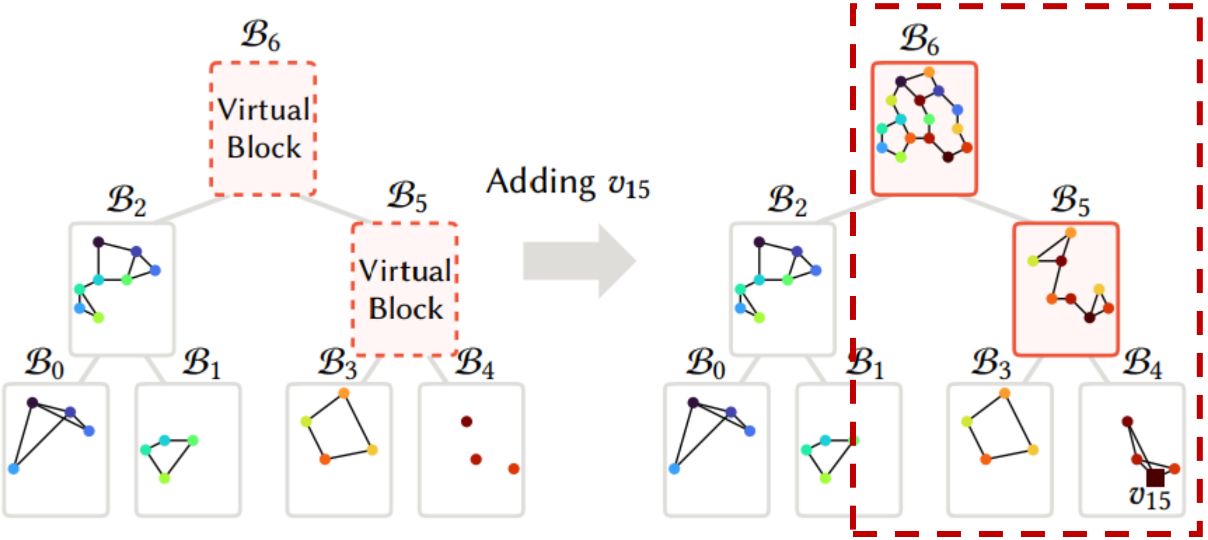
Build KNN graph of block

Build KNN graph

when leaf block become full

Proposed Method

2. Insertion and Indexing process of MBI



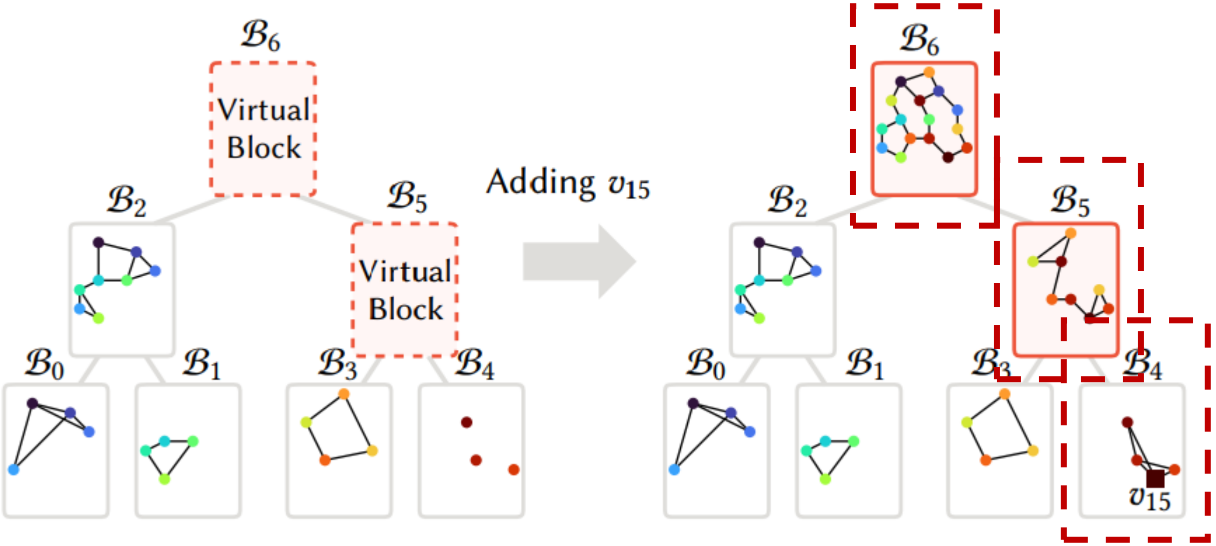
Build KNN graph of block

Build KNN graph

when own time range become full

Proposed Method

2. Insertion and Indexing process of MBI



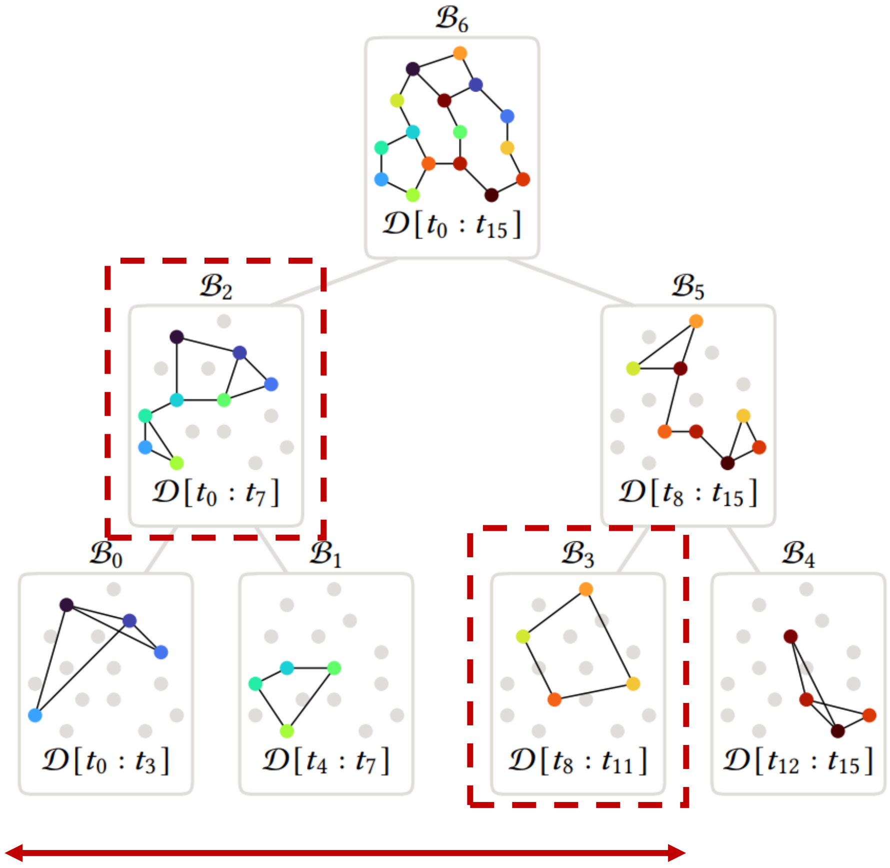
Build KNN graph of block

Can be easily parallelized

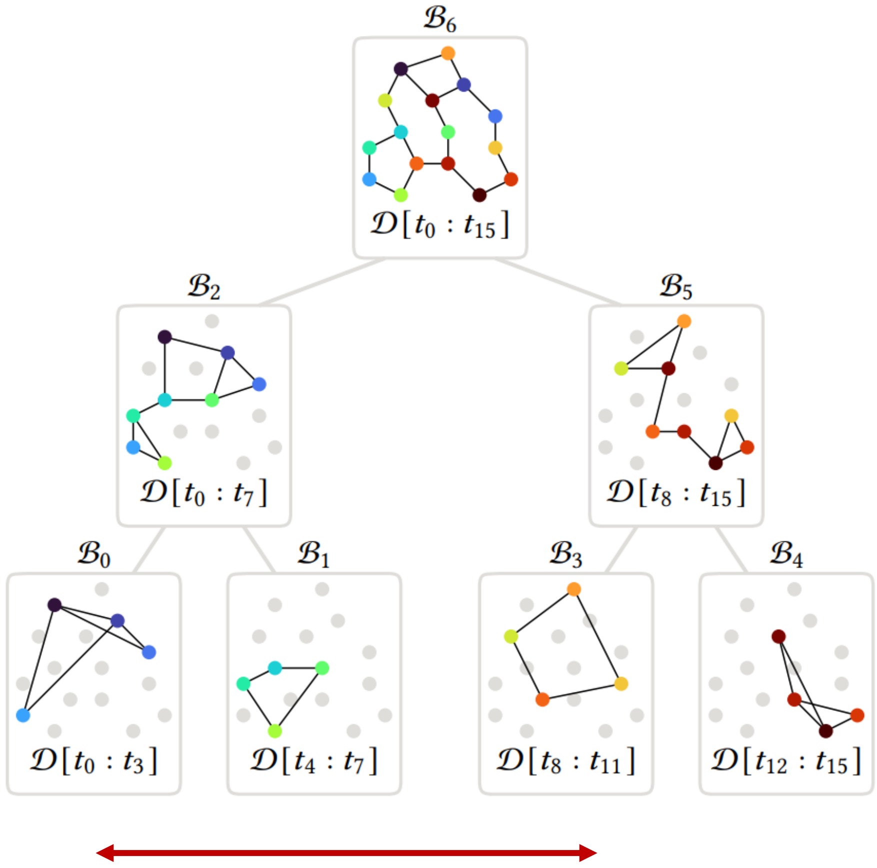
Index

Proposed Method

3. Query process of MBI



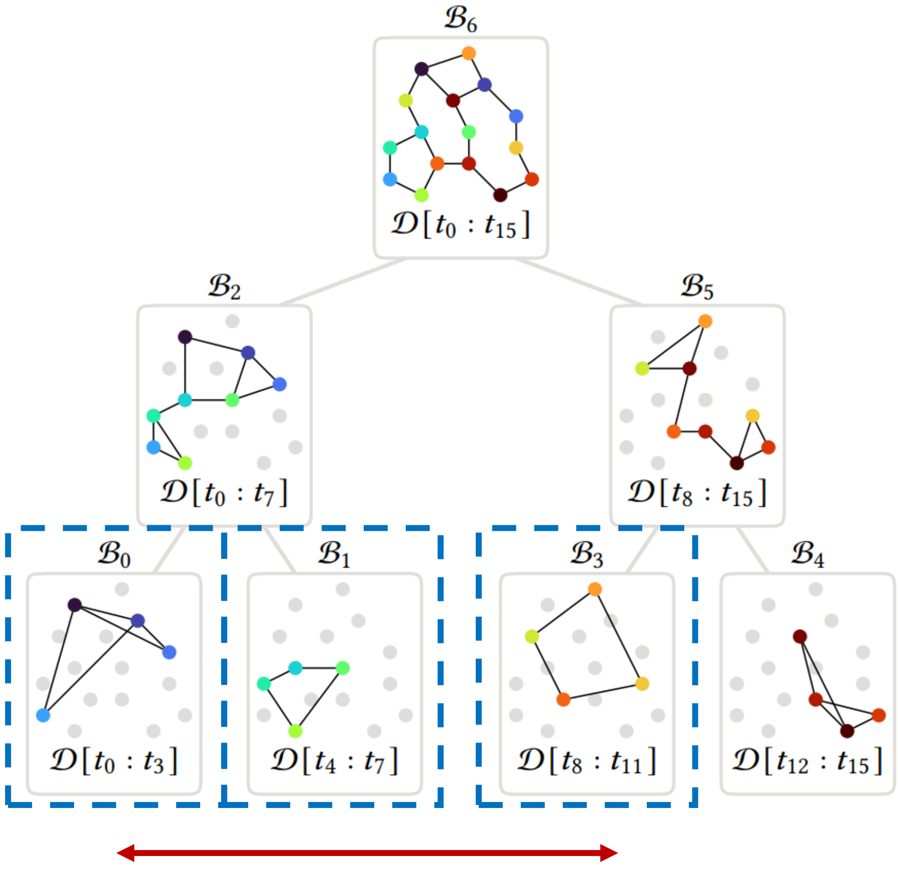
Fine

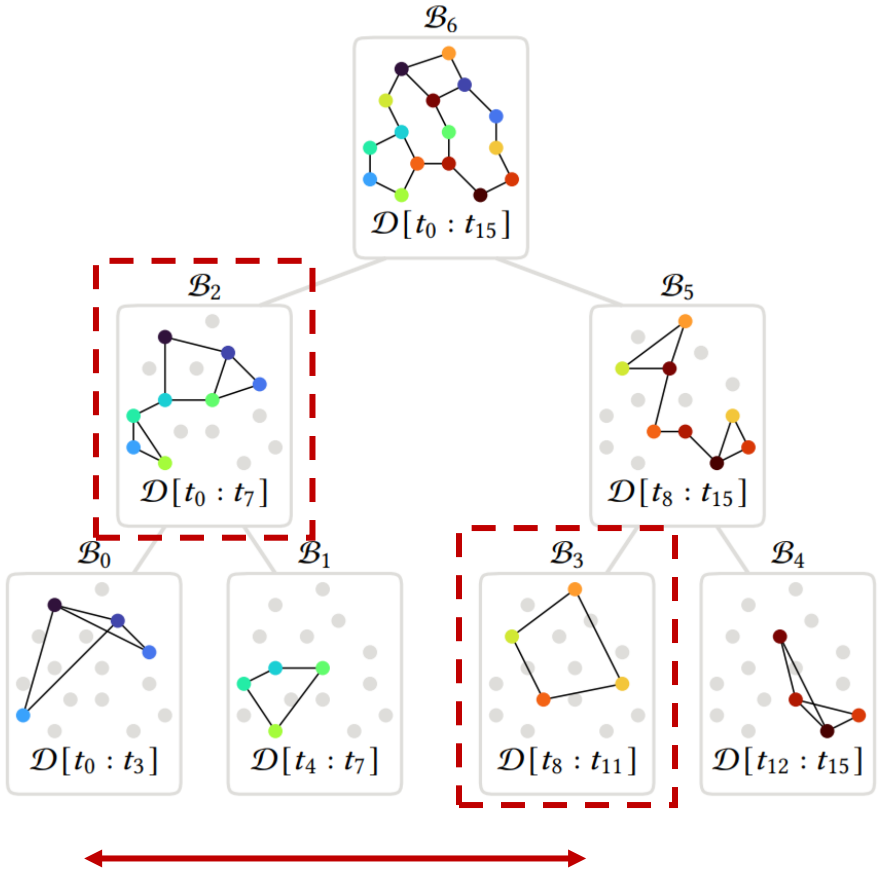


How should blocks be selected?

Proposed Method

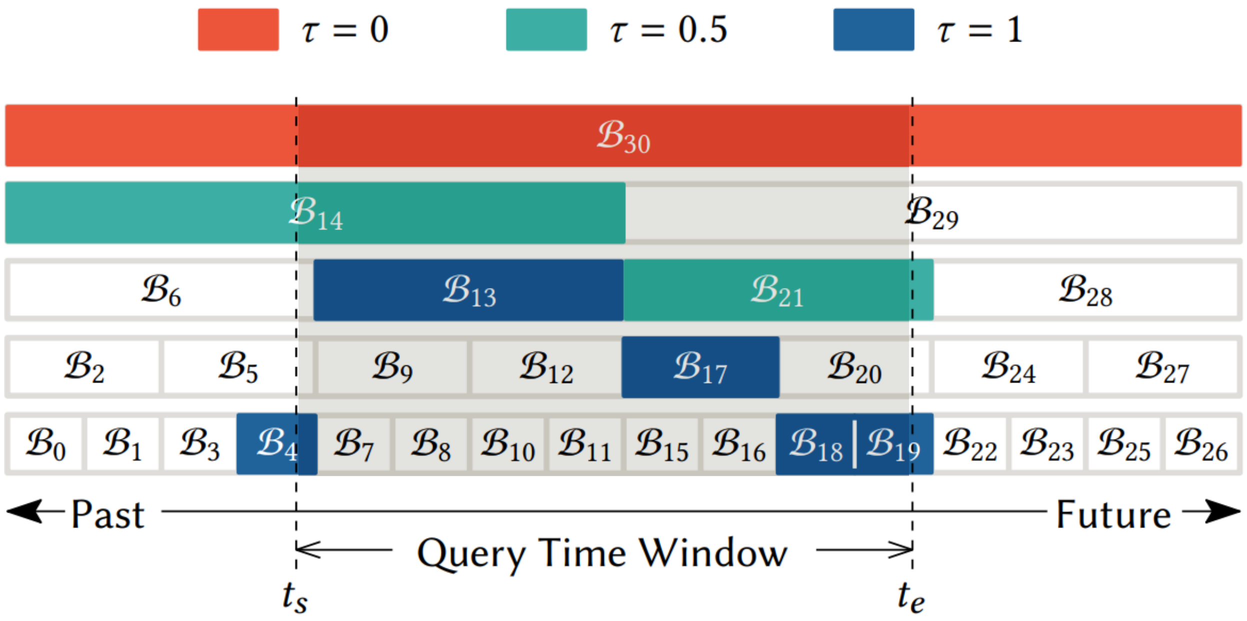
3. Query process of MBI





Proposed Method

3. Query process of MBI

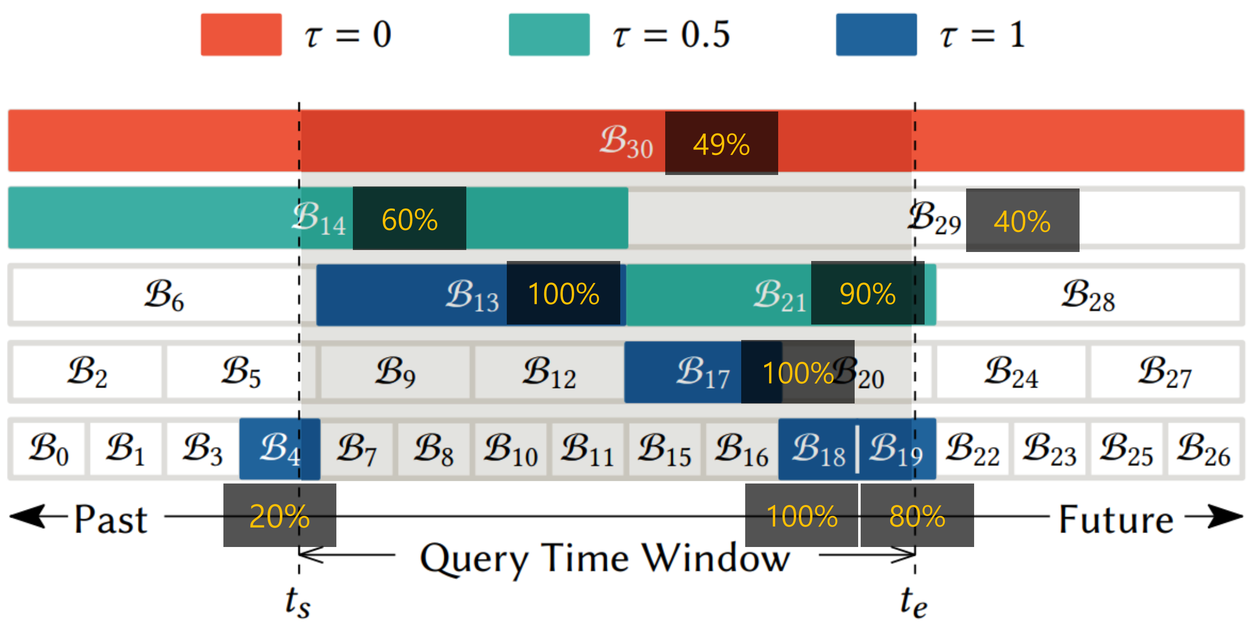


Select blocks to query

Parameter τ

Proposed Method

3. Query process of MBI

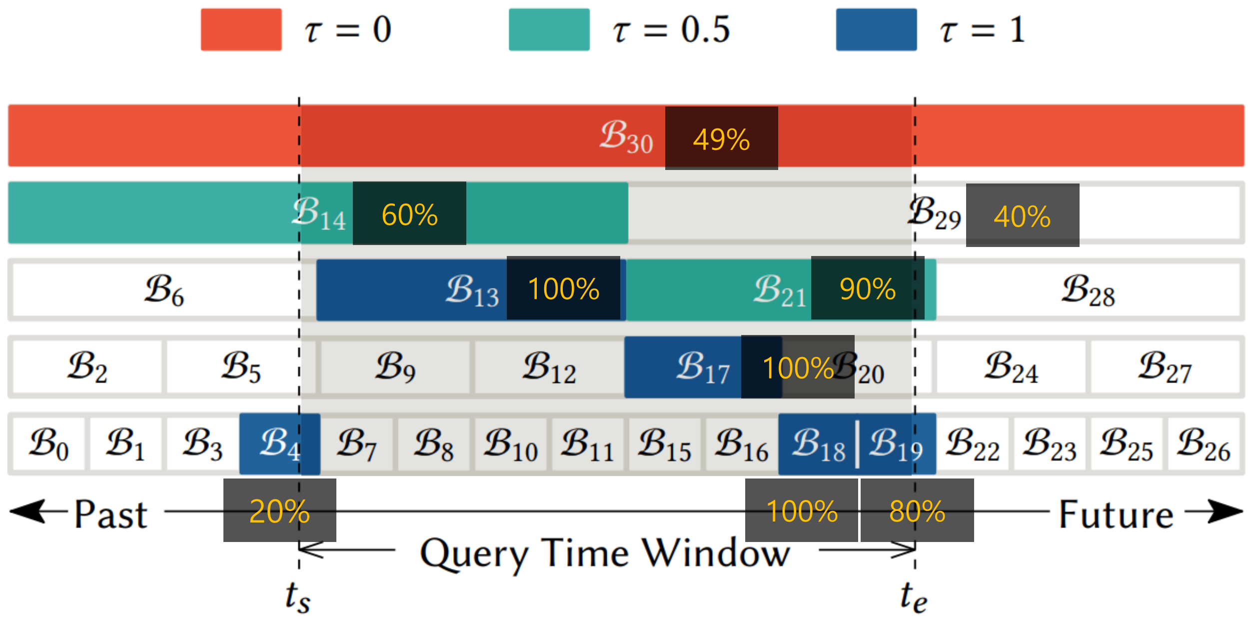


The ratio of the block included in the query time window

Select blocks to query

Proposed Method

3. Query process of MBI



The ratio of the block included in the query time window

Select blocks to query

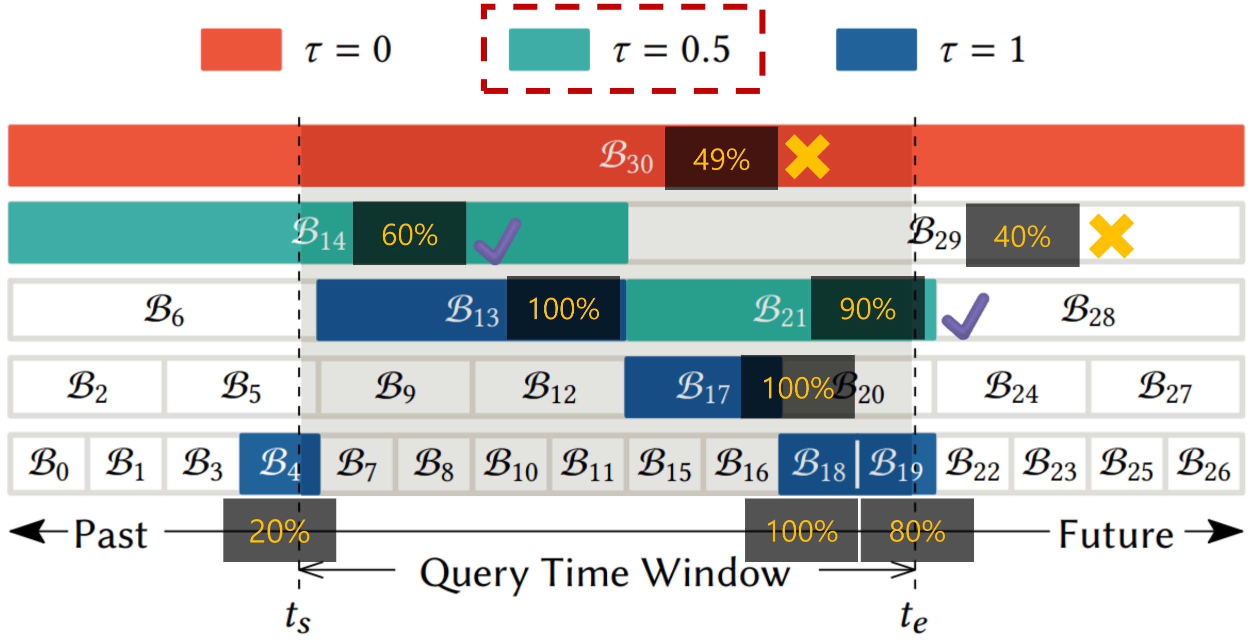
Start at root,

If the ratio > τ, select and stop

Else, move to child nodes

Proposed Method

3. Query process of MBI



The ratio of the block included in the query time window

Select blocks to query

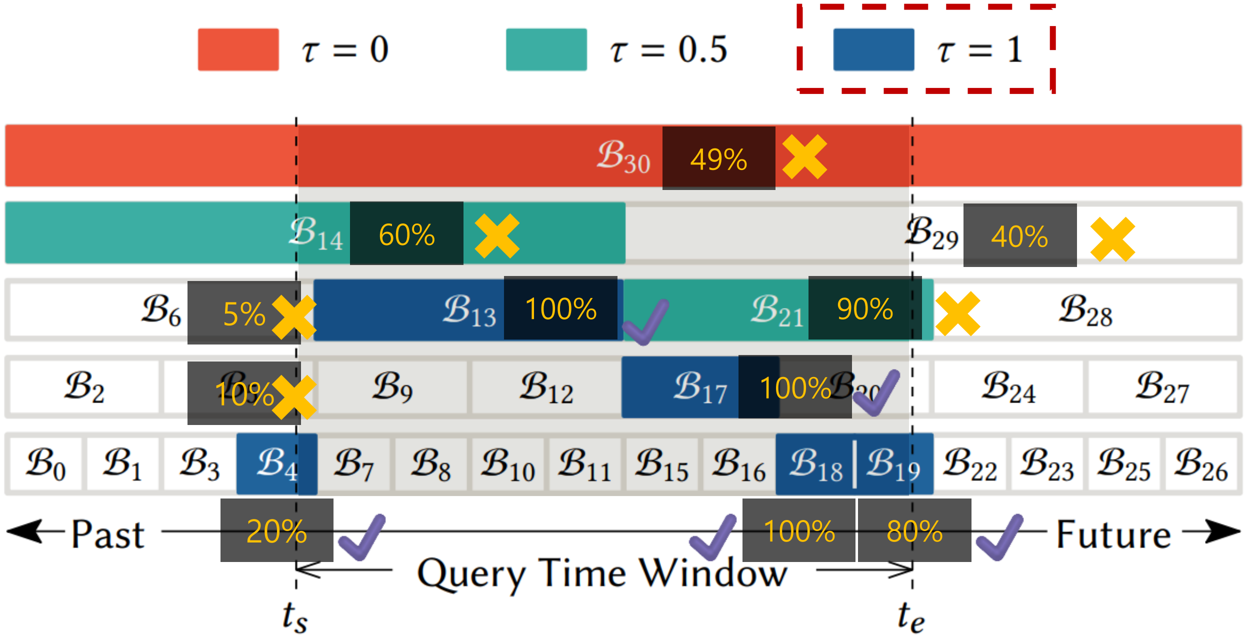
Start at root,

If the ratio > τ, select and stop

Else, move to child nodes

Proposed Method

3. Query process of MBI



The ratio of the block included in the query time window

Select blocks to query

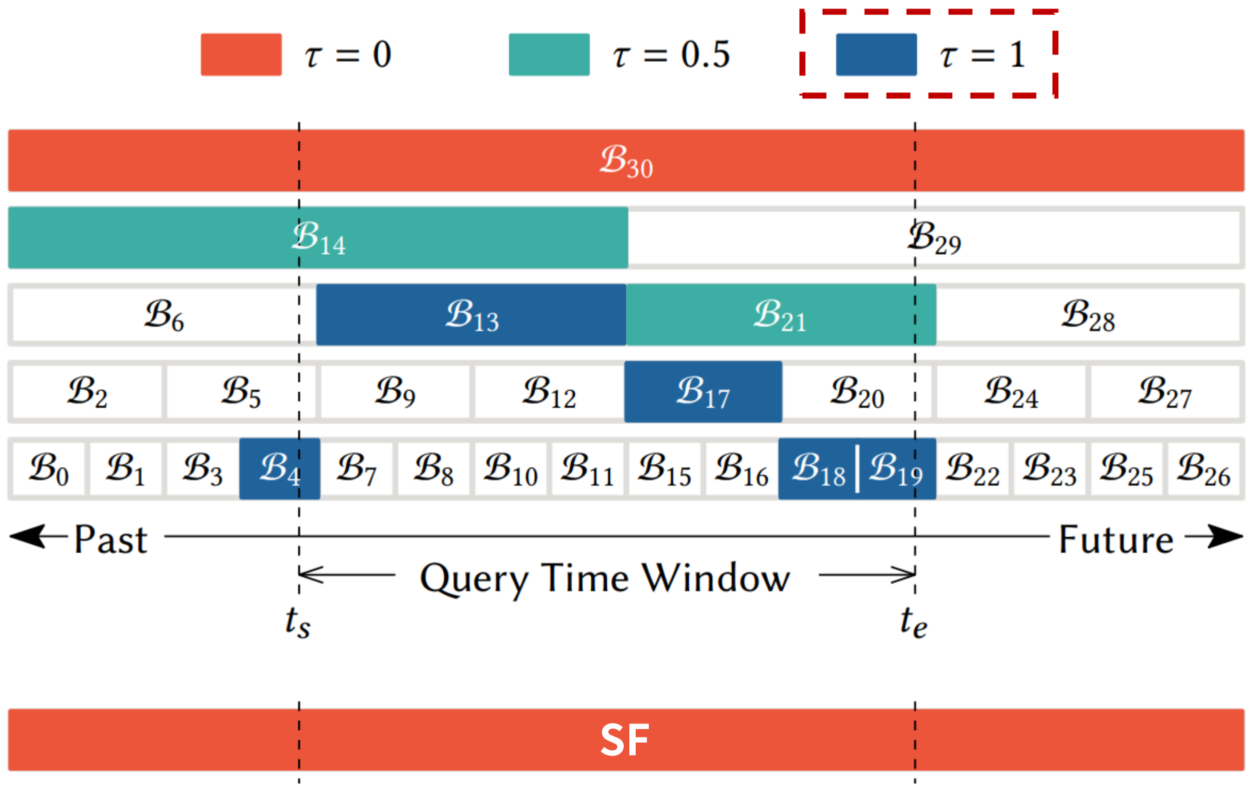
Start at root,

If the ratio > τ, select and stop

Else, move to child nodes

Proposed Method

3. Query process of MBI



Effect of Parameter τ

High τ, fit well

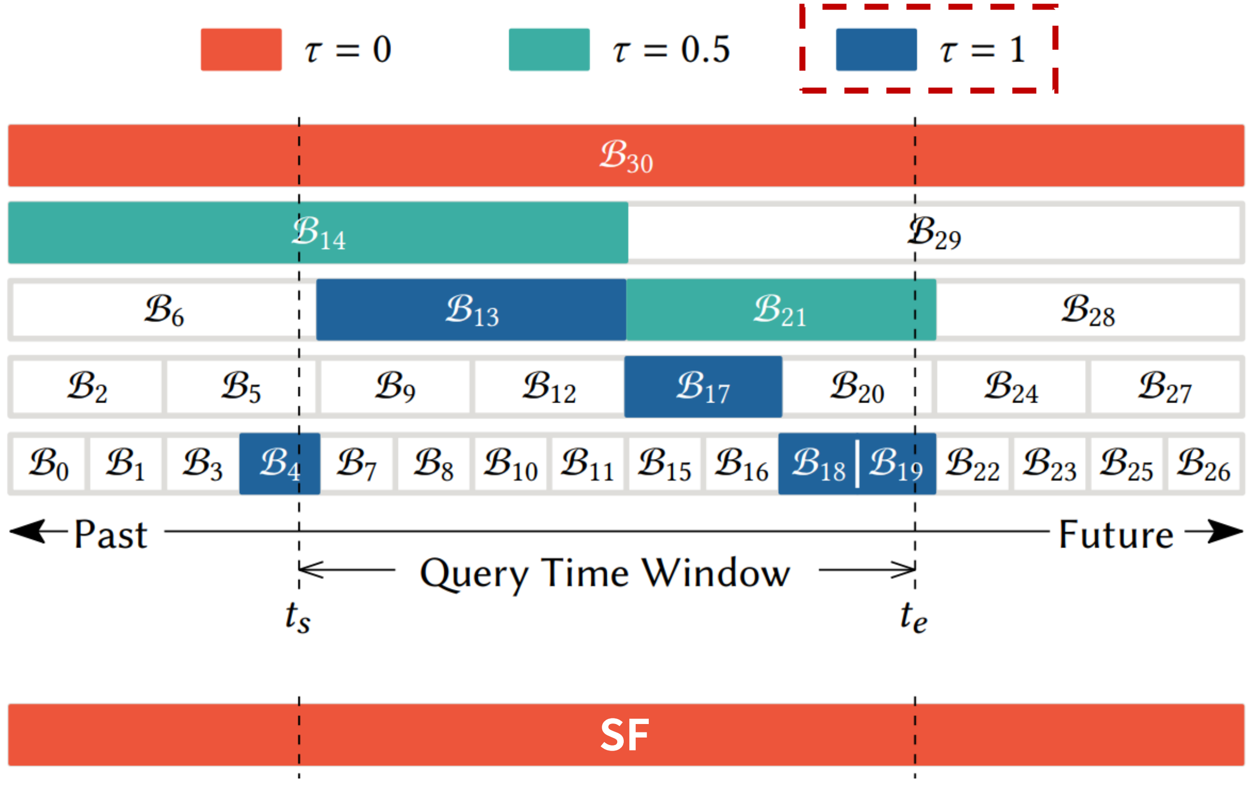
But have to query on many blocks

Low τ, fit bad (like weakness of SF)

But can query on lesser blocks

Proposed Method

3. Query process of MBI



Effect of Parameter τ

High τ, fit well

But have to query on many blocks

Low τ, fit bad (like weakness of SF)

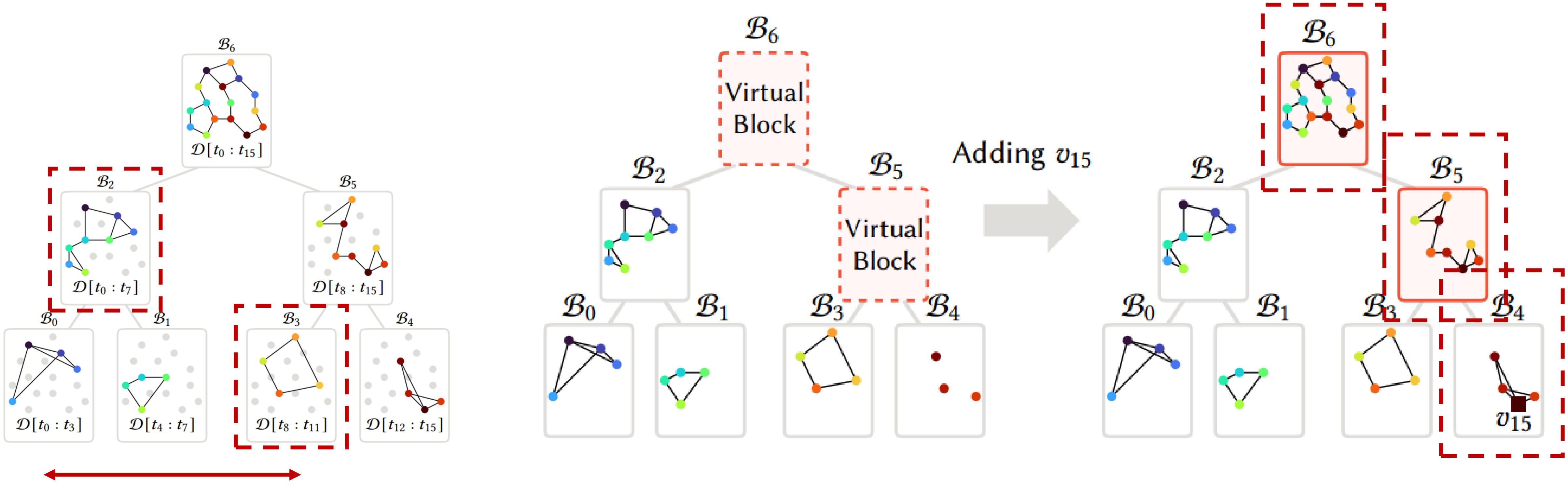
But can query on lesser blocks

If τ<=0.5,

only up to two blocks will be selected.

Proposed Method

Summary of MBI



Index

1. Intro
2. Preliminaries
3. Proposed Method
4. Experiments
5. Conclusion

Experiments Challenges

Q1. Search Performance

Does MBI provide the best search?

Q2. Scalability

How well does MBI scale up in terms of the data size?

Q3. Data Insertion time

How efficiently can new data be inserted into?

Experiments

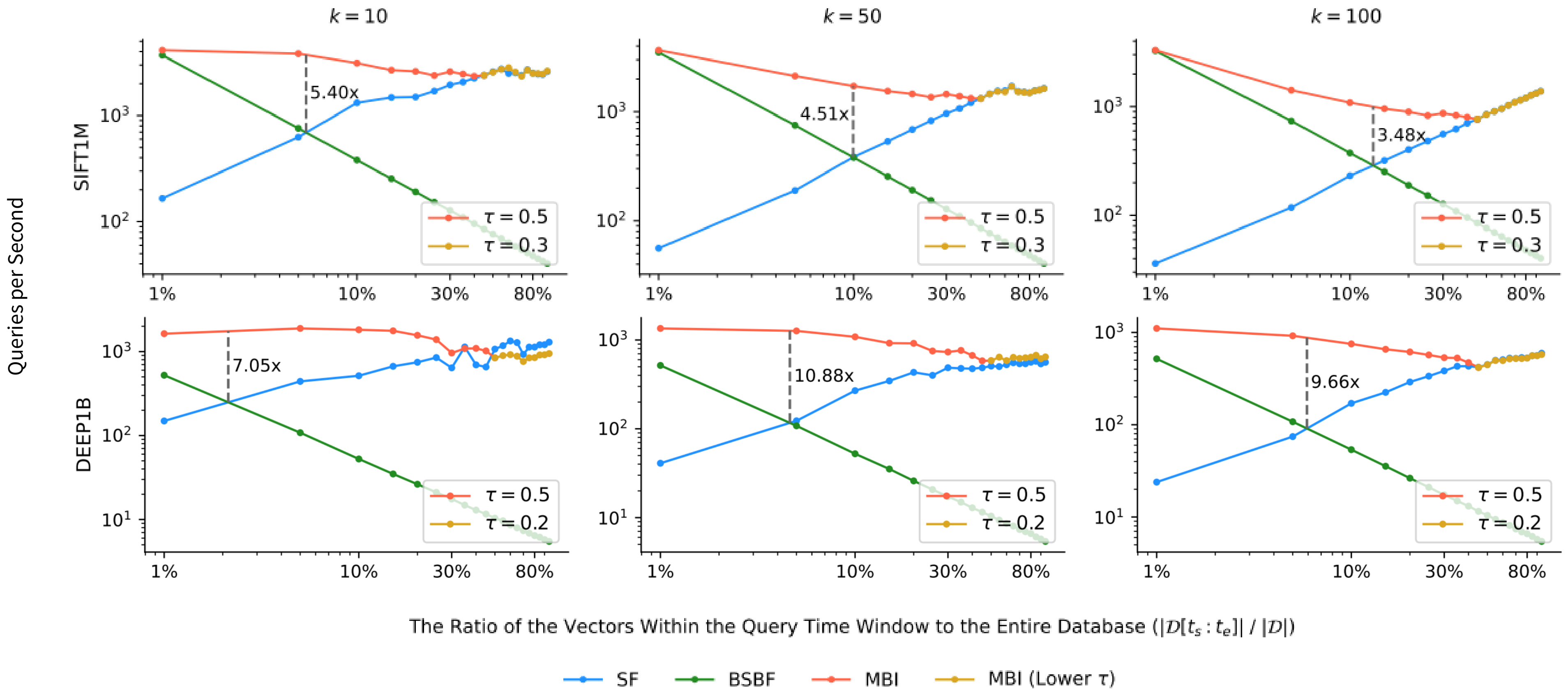
1. Datasets

	Datasets	# items		Dim.	Distance	Source
		Train	Test			
Include Time Label	MovieLens	57,571	200	32	Angular	GroupLens[6]
	COMS	291,180	200	128	Angular	KMA[7]
Synthetic Time Label	GloVe-100	1,183,514	10,000	100	Angular	Pennington et al.[8] [33]
	SIFT1M	1,000,000	10,000	128	Euclidean	Jégou et al.[9] [21]
	GIST1M	1,000,000	1,000	960	Euclidean	
	DEEP1B	9,990,000	10,000	96	Angular	Babenko et al.[10] [5]

Experiments

2. Performance

Queries per second when recall@k is set to 0.995



Preliminaries

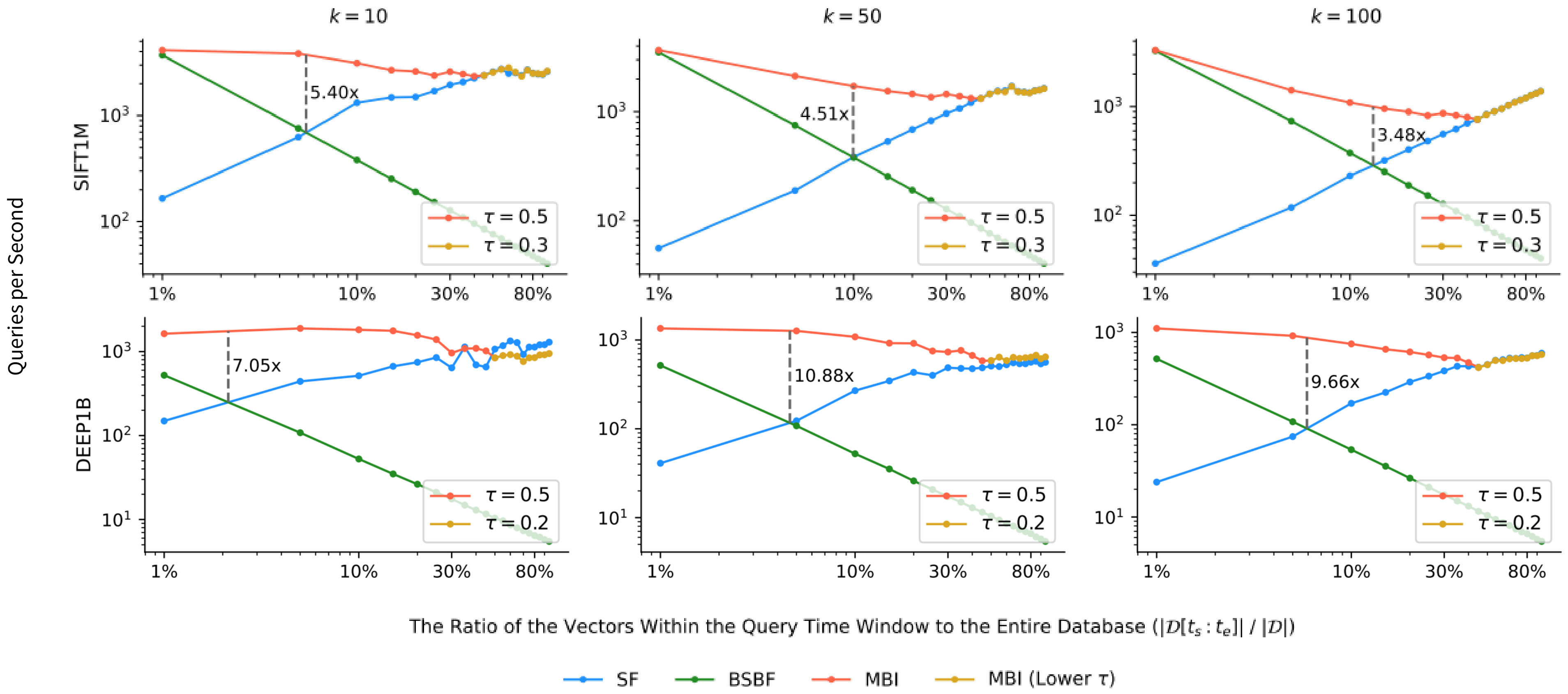
2. Simple approaches for TKNN Search

time window	Binary Search and Brute-Force (BSBF)	Search and Filtering (SF)	Multi-level Block Indexing (MBI)
Narrow	✓	✗	✓
Wide	✗	✓	✓

Experiments

2. Performance

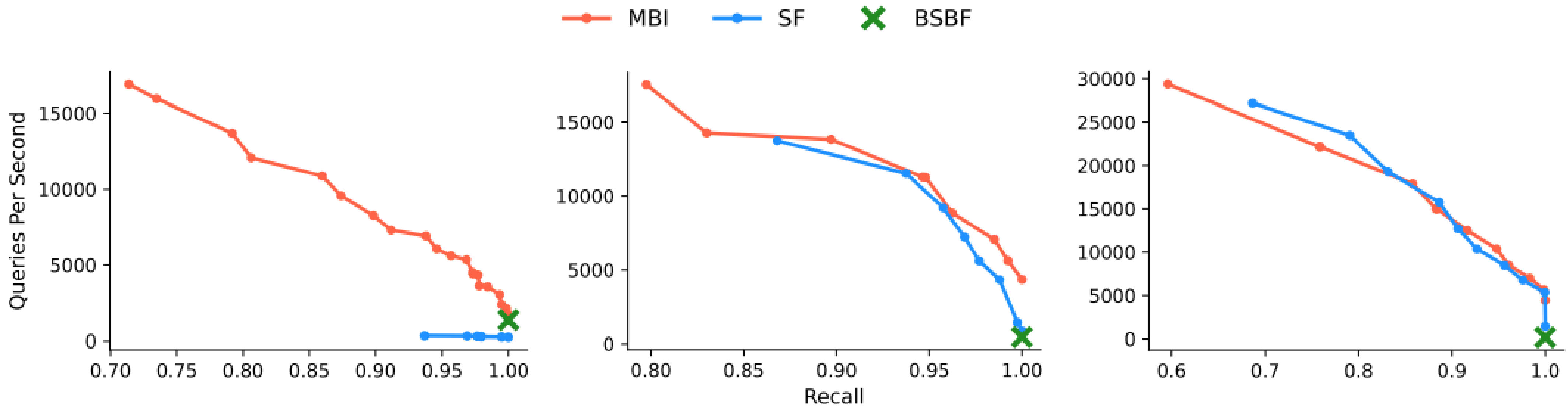
Queries per second when recall@k is set to 0.995



Uniform performance regardless of the time window

Experiments

2. Performance

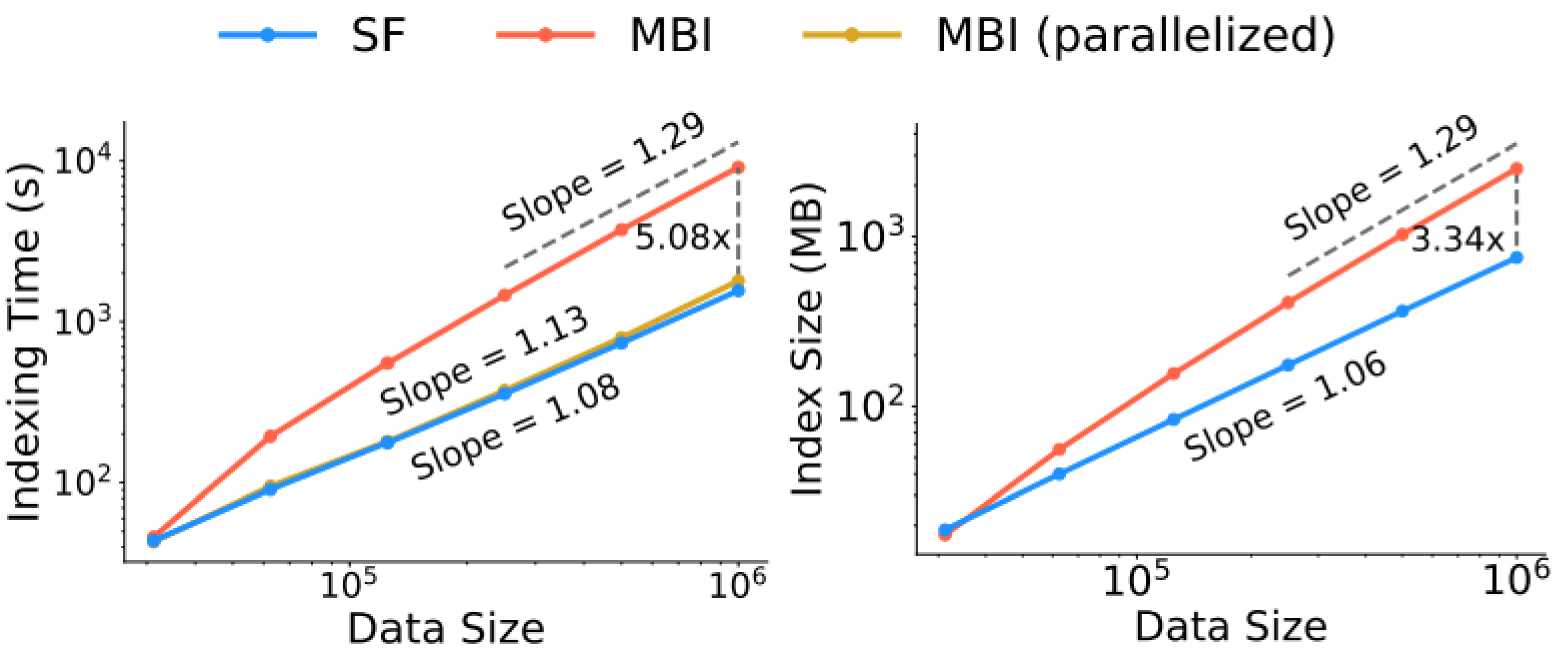


(a) $|\mathcal{D}[t_s : t_e]|/|\mathcal{D}| = 10\%$ **(b)** $|\mathcal{D}[t_s : t_e]|/|\mathcal{D}| = 30\%$ **(c)** $|\mathcal{D}[t_s : t_e]|/|\mathcal{D}| = 80\%$

Uniform performance regardless of the time window

Experiments

3. Scalability

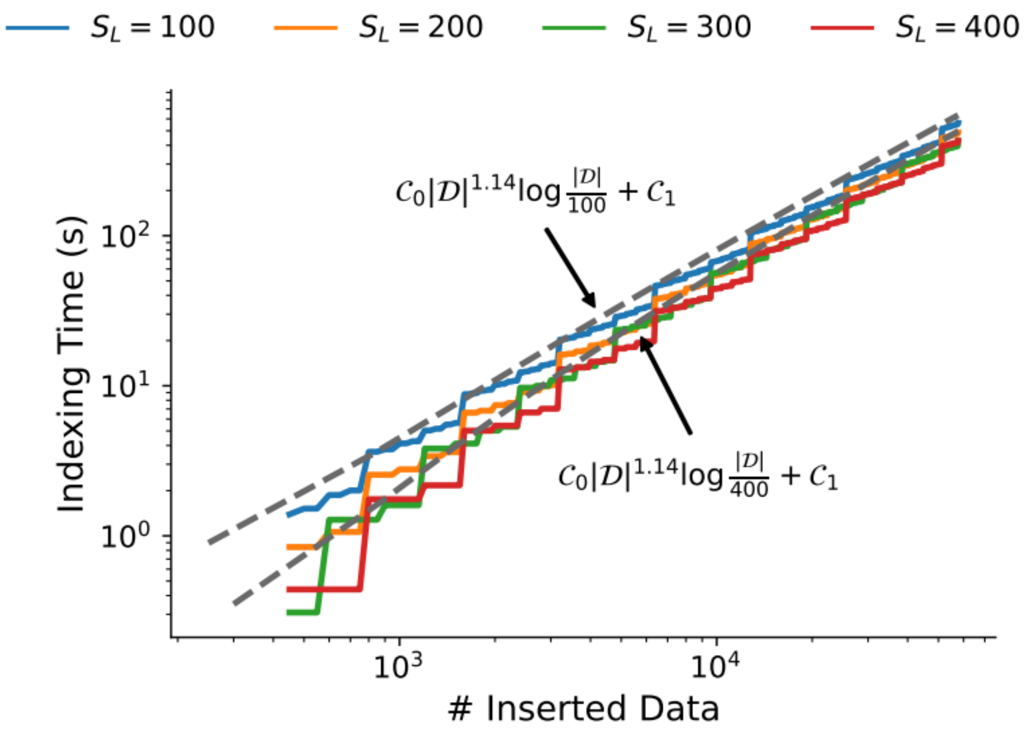


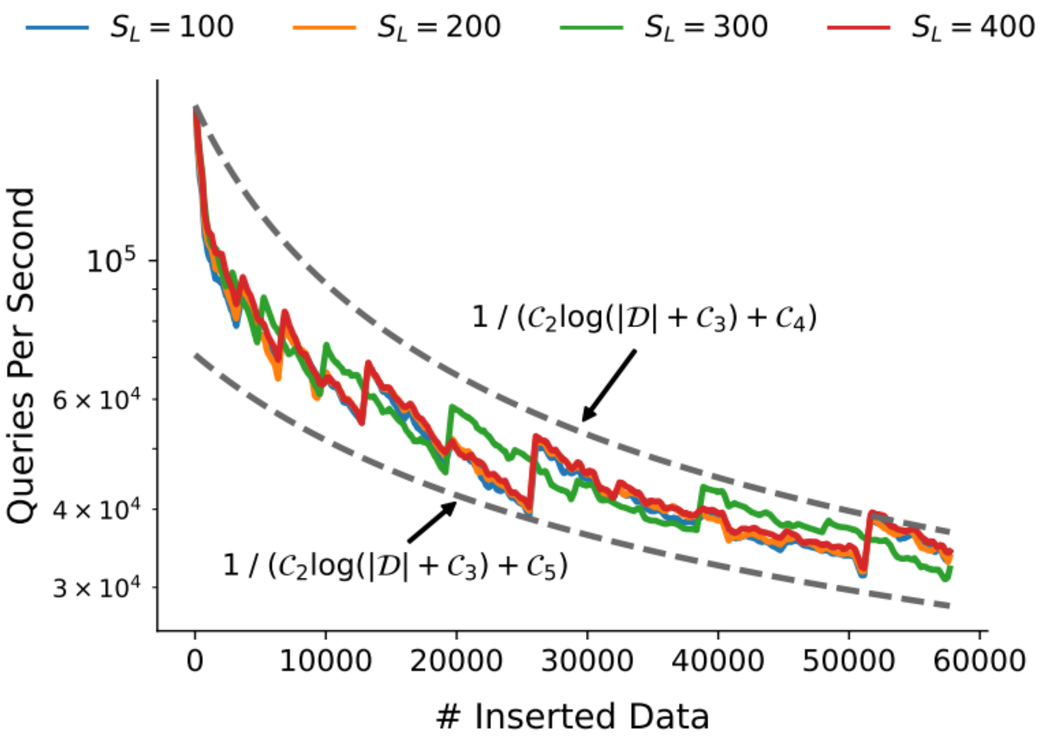
Fits within the theoretical complexity

Parallelization works well

Experiments

3. Scalability

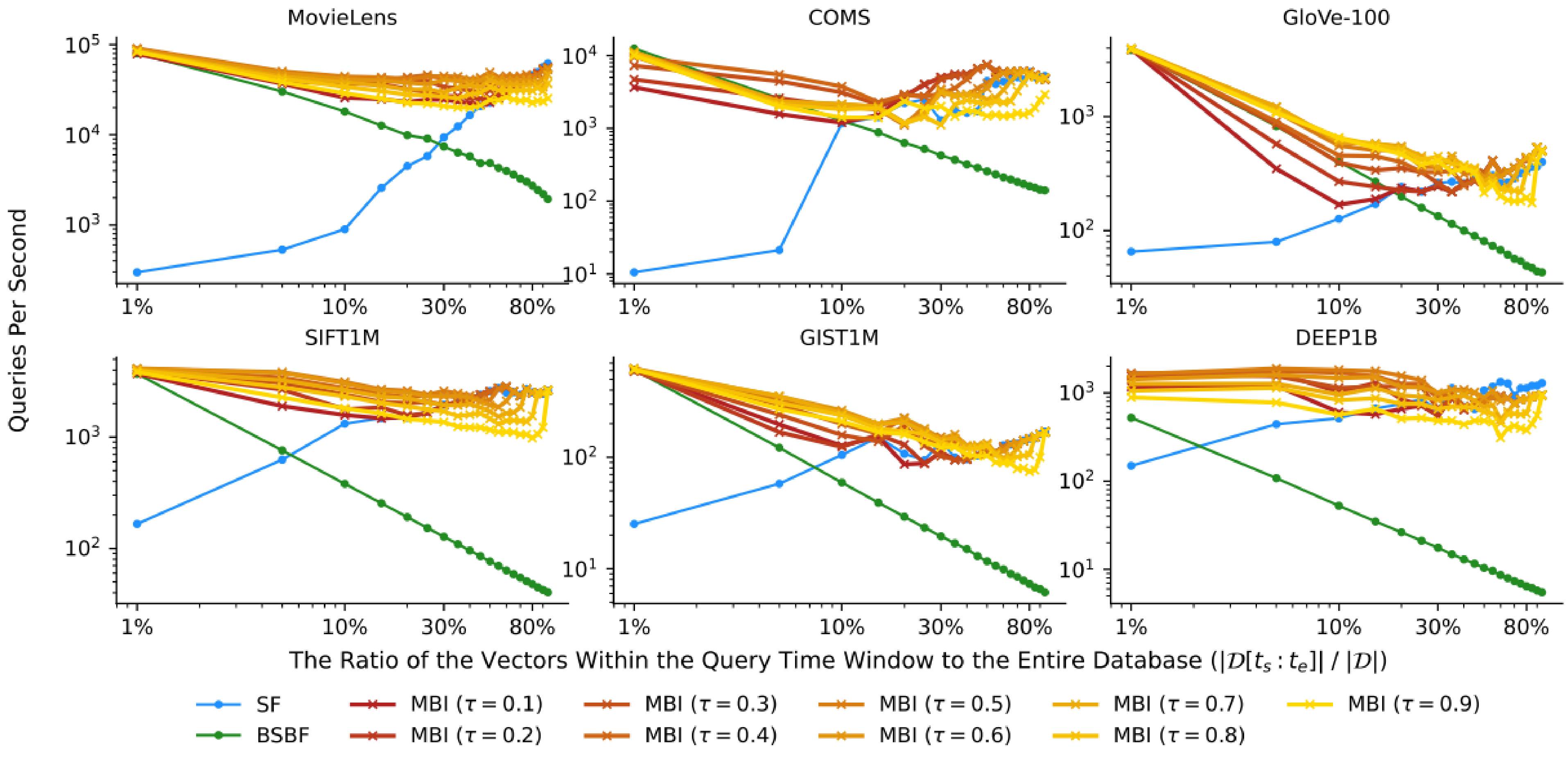




Performances fit within the theoretical complexity

Experiments

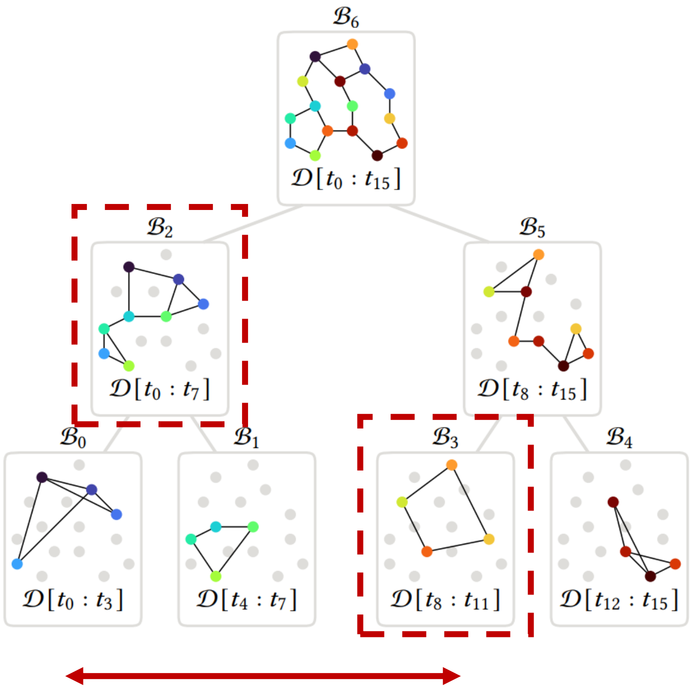
4. Effect of Parameter τ



Fits within the analysis (τ <= 0.5)

Index

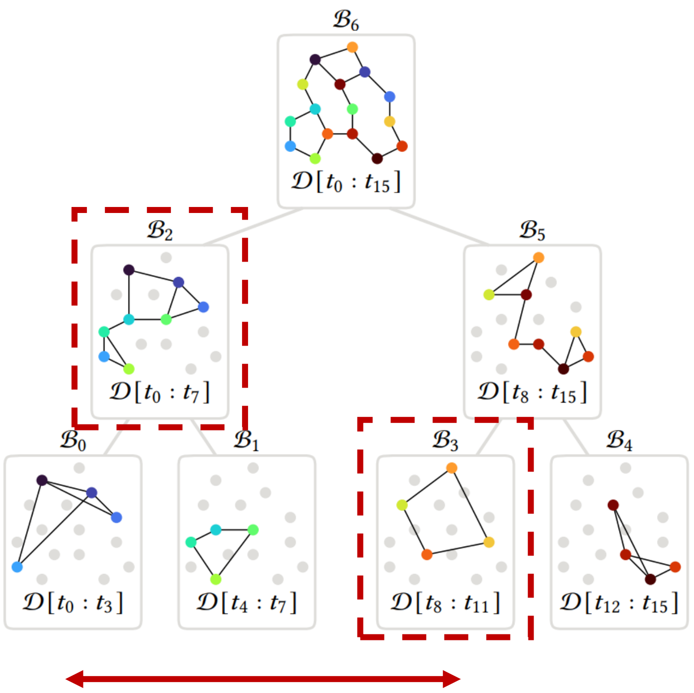
Conclusion



Multi-level Block Indexing

- Uniform and superior TKNN search performance.
- Efficiently handles data insertion.
- Highly scalable

Conclusion



Multi-level Block Indexing

- Uniform and superior TKNN search performance.
- Efficiently handles data insertion.
- Highly scalable

Efficient Proximity Search in **Time-accumulating High-dimensional Data using Multi-level Block Indexing**

Thank you

https://bkshin.tistory.com/entry/%EB%A8%B8%EC%8B%A0%EB%9F%AC%EB%8B%9D-6-K-%EC%B5%9C%EA%B7%BC%EC%A0%91%EC%9D%B4%EC%9B%83KNN

https://opendsa-server.cs.vt.edu/ODSA/Books/CS3/html/KDtree.html

https://pynndescent.readthedocs.io/en/latest/how_pynndescent_works.html