Efficient Spatial Dataset Search over Multiple Data Sources

arXiv:2311.13383v1 [cs.DB] 22 Nov 2023

Wenzhe Yang, Sheng Wang, Yuan Sun, Zhiyu Chen, Zhiyong Peng

目录

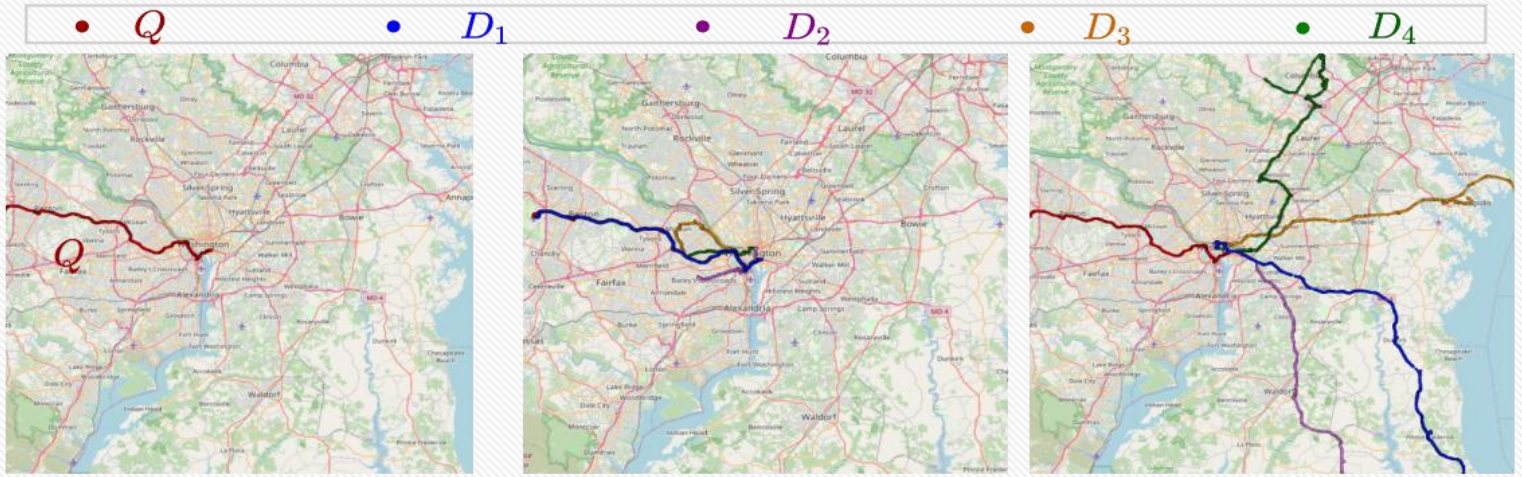
1 查询模型

- Maximum Intersection Query(MIQ)

  → Join search

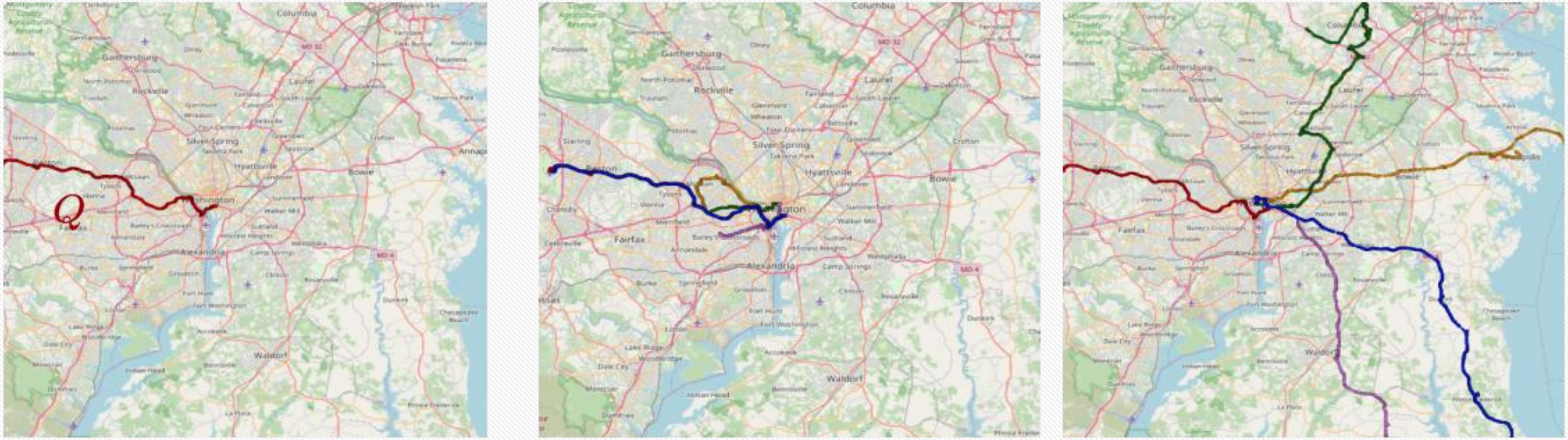
- Maximum Coverage Query with a Connection constraint (MCQC)

  → Union search



(a) Query dataset (b) Join search (c) Union search

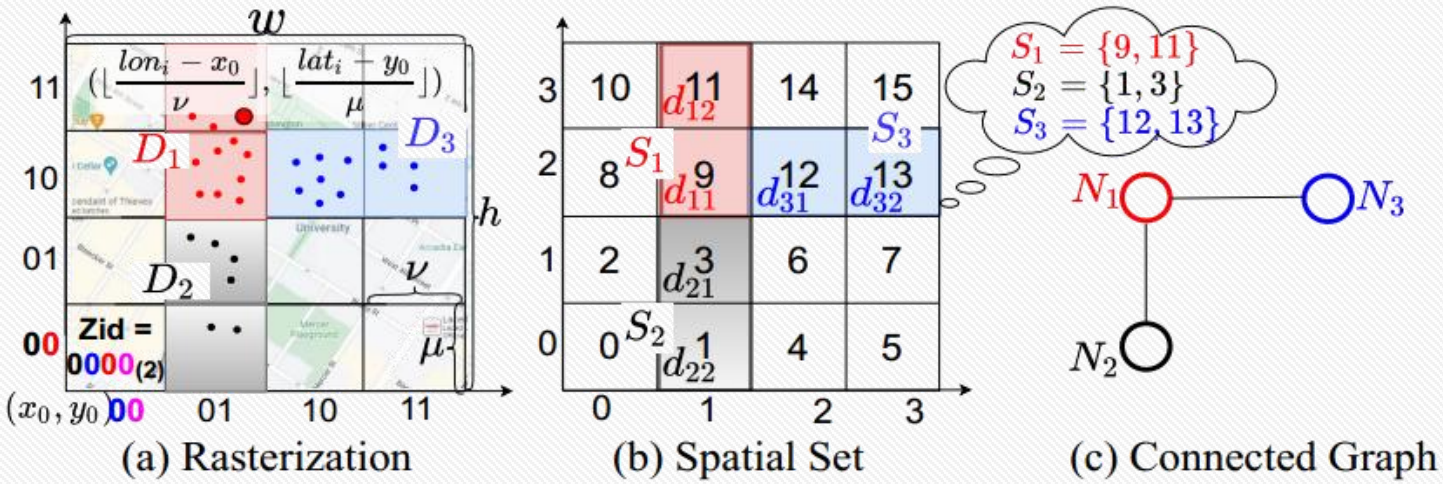
• Q • D_1 • D_2 • D_3 • D_4



(a) Query dataset (b) Join search (c) Union search

2 数据模型

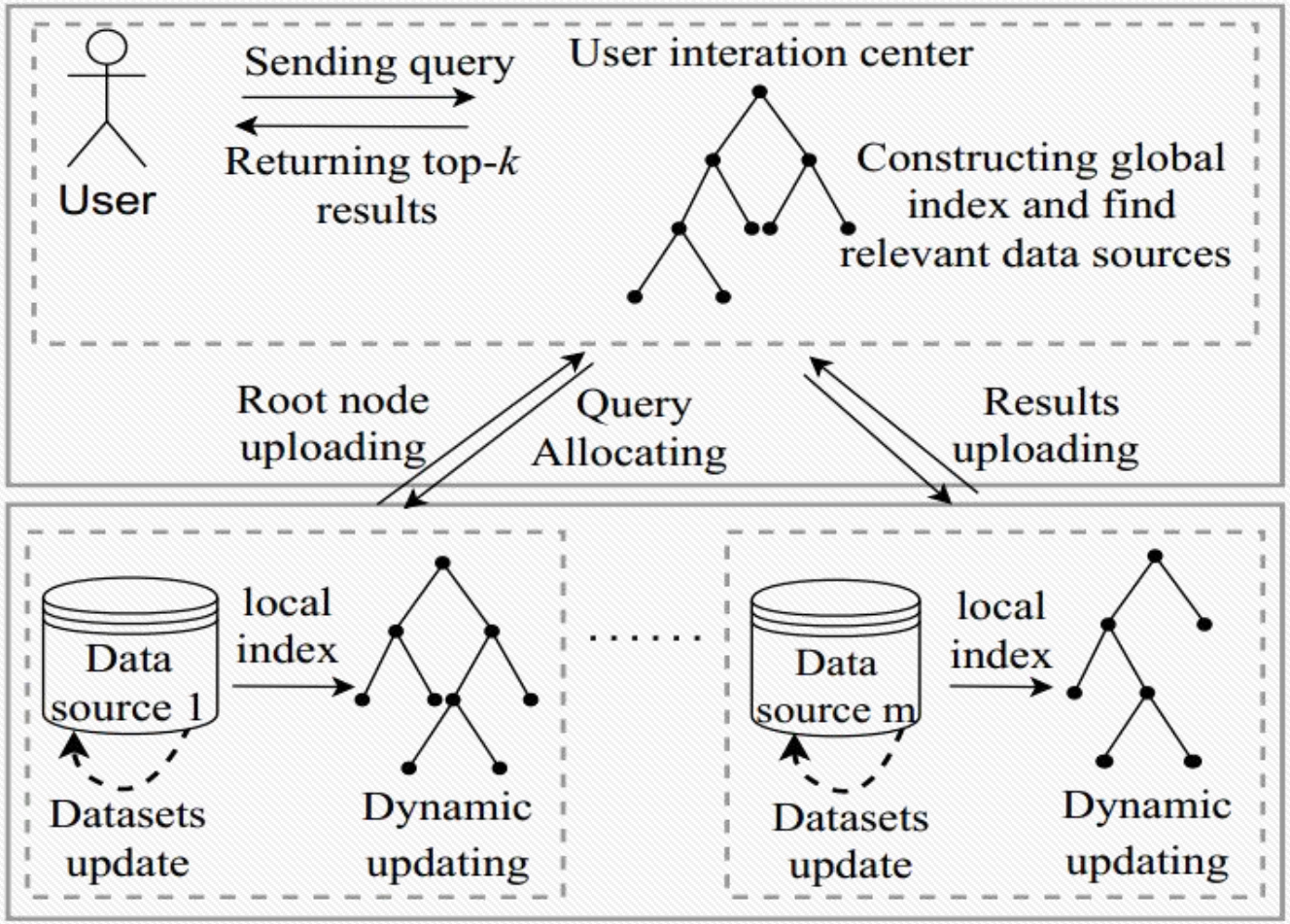
- Spatial Data Source $D = \{D_1, D_2, ..., D_{|D|}\}$
- Spatial Set $S_D = \{d_1, d_2, ..., d_{|S_D|}\}$
- Spatial Set Distance $dist(S_Q, S_D) = \min\{||q_i, d_j||^2 : q_i \in S_Q, d_j \in S_D\}$
- δ−Connectivity $dist(S_Q, S_D) \leq \delta$
- Connected Graph $G(V, E)$



(a) Rasterization (b) Spatial Set (c) Connected Graph

3 搜索框架

Multi-source Spatial Dataset Search (MSDS)



4 索引构建

- Local Index Construction
- Global Index Construction
- Dataset Graph Construction

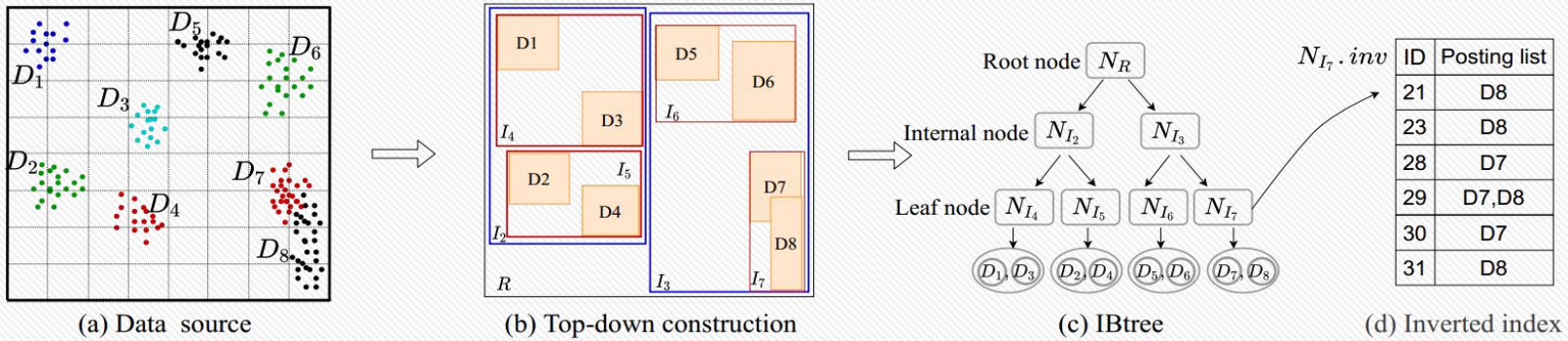
4 索引构建

· Local Index Construction

—— IBtree

· Global Index Construction

· Dataset Graph Construction



(a) Data source (b) Top-down construction (c) IBtree (d) Inverted index

ID	Posting list
21	D8
23	D8
28	D7
29	D7,D8
30	D7
31	D8

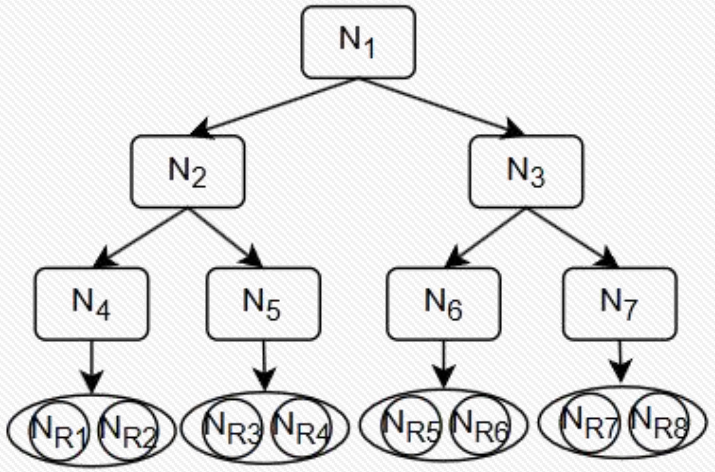
(Dataset Node)： N_D = (id, rect, p, r, pa, s)

(Internal Node): N_I = (rect, p, r, ch, pa)

(Leaf Node): N_L =(rect, p, r, ch, pa, inv)

4 索引构建

- Local Index Construction
- Global Index Construction
- Dataset Graph Construction

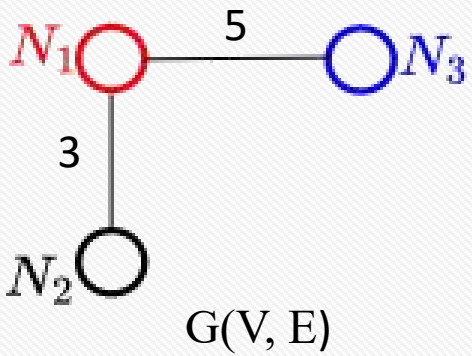


4 索引构建

· Local Index Construction

· Global Index Construction

· Dataset Graph Construction

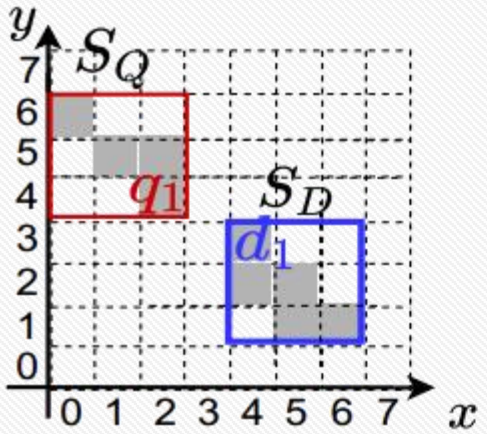


G(V, E)

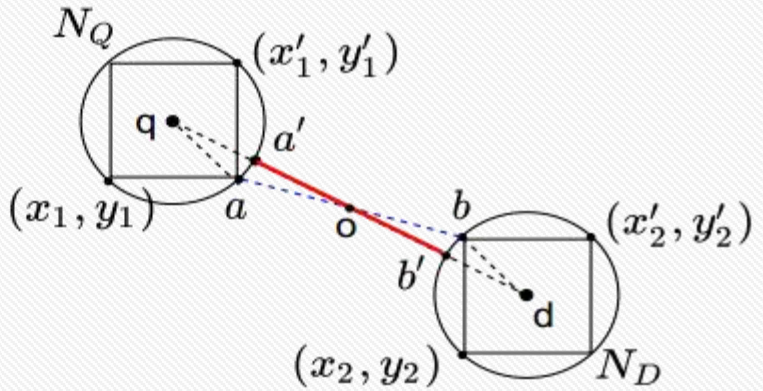
距离下界：$lb(N_Q, N_D) = \max\{||N_Q.p, N_D.p||^2 - N_Q.r - N_D.r, 0\}$

4 索引构建

- Local Index Construction
- Global Index Construction
- Dataset Graph Construction



(a) Set distance



(b) Lower bound

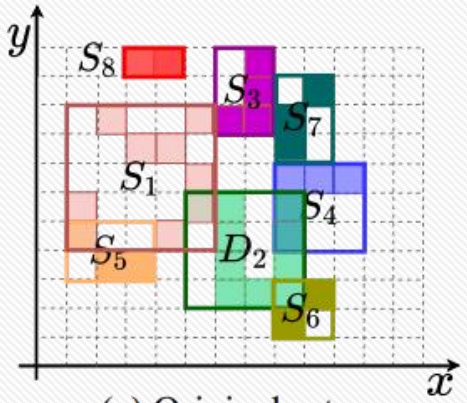
距离下界：$lb(N_Q, N_D) = \max\{||N_Q.p, N_D.p||^2 - N_Q.r - N_D.r, 0\}$

例子：$dist(S_Q, S_D) = ||q_1, d_1||^2 = \sqrt{5} \approx 2.236$

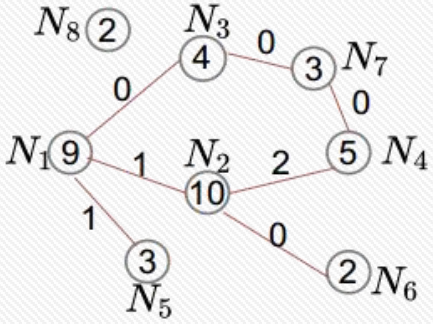
$lb(N_Q, N_D) = \max\{\sqrt{5} - \sqrt{2} - \sqrt{2}, 0\} \approx 2.172 \leq 2.236$

4 索引构建

- Local Index Construction
- Global Index Construction
- Dataset Graph Construction



(a) Original sets



(b) Dataset graph

距离下界：$lb(N_Q, N_D) = \max\{||N_Q.p, N_D.p||^2 - N_Q.r - N_D.r, 0\}$

距离下界与IBtree结合加快数据集图构建

两种查询分发策略

- 仅向剪枝后的候选数据源发送查询请求
- 仅向本地数据源传输MBR区域信息

基于IBtree的加速MIQ搜索算法

Lemma 3. ***(MBRBound)*** *Let N_Q denote a query node with the set representation $S_Q = \{q_1, q_2, \ldots, q_n\}$, N_L denote a leaf node containing multiple dataset nodes, and f denote the capacity of the leaf node, the intersection between N_Q and N_L is upper bounded by $\sum_{i=1}^{n} \phi(q_i)$,*

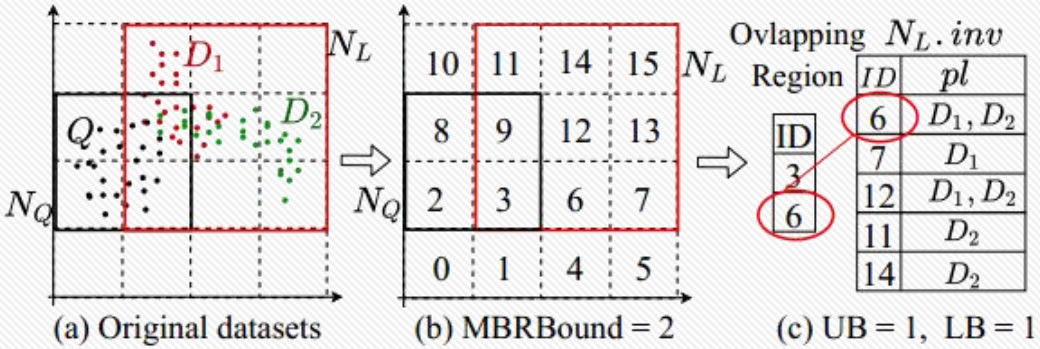
$$\phi(q_i) = \begin{cases} 1, & if\ q_i \in (N_Q.rect \cap N_L.rect), \\ 0, & otherwise \end{cases}$$

Lemma 4. ***(UpperBound)*** *The upper bound of intersection between leaf node N_L and query node N_Q is $\sum_{i=1}^{n} \phi(q_i)\varphi(q_i)$.*

$$\varphi(q_i) = \begin{cases} 1, & if\ q_i \in N_L.inv, \\ 0, & otherwise \end{cases}$$

Lemma 5. ***(LowerBound)*** *The lower bound of intersection between N_L and N_Q is $\sum_{i=1}^{n} \phi(q_i)\varphi(q_i)$.*

$$\varphi(q_i) = \begin{cases} 1, & if\ q_i \in N_L.inv \& |q_i.pl| = f, \\ 0, & otherwise \end{cases}$$



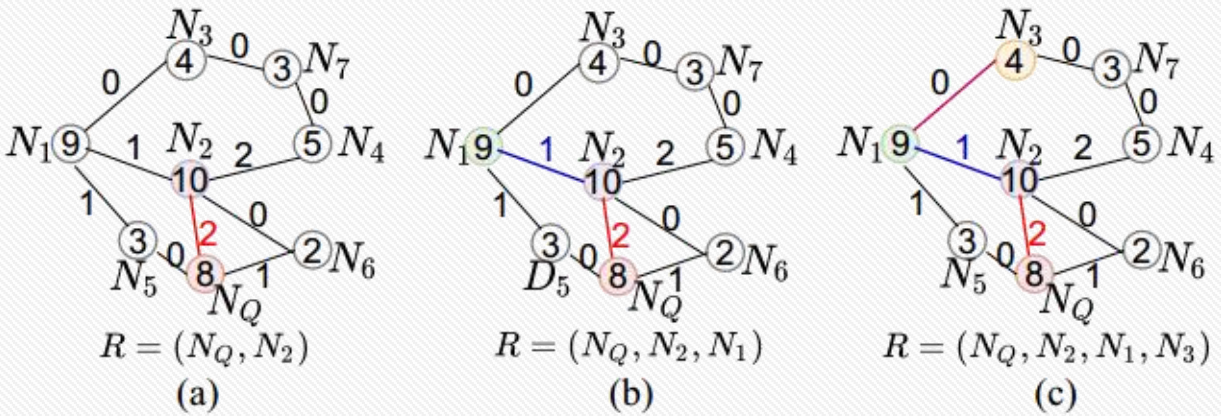
(a) Original datasets (b) MBRBound = 2 (c) UB = 1, LB = 1

基于局部索引的贪心方法加速MCQC搜索

- 贪婪空间合并算法(GASM)

$$g(S_D, \mathcal{R}) = |S_D \cup (\cup_{S_i \in \mathcal{R}} S_i)| - |\cup_{S_i \in \mathcal{R}} S_i|.$$

- 基于数据集图的贪心算法 (GADG)



动态索引更新

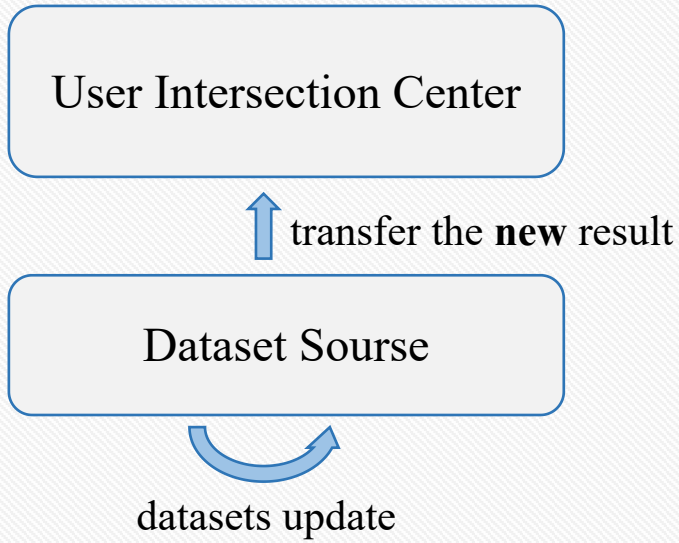
(Dataset Node)： N_D = (id, rect, p, r, pa, s)

(Internal Node): N_I = (rect, p, r, ch, pa)

(Leaf Node): N_L =(rect, p, r, ch, pa, inv)

Top-k动态搜索

- MIQ
- MCQC



User Intersection Center - 1

Dataset Sourse - 5

Parameter	Settings
k: number of results	{10, 20, 30, 40, 50}
n: number of queries	{ 10, 20, 30, 40, 50 }
θ: resolution	{10, 11, 12, 13, 14 }
δ: connectivity	{0, 5, 10, 15, 20 }
f: leaf node capacity	{10, 100, 200, 300, 400 }
β: number of datasets updates	{100, 150, 200, 250, 300}

7 实验

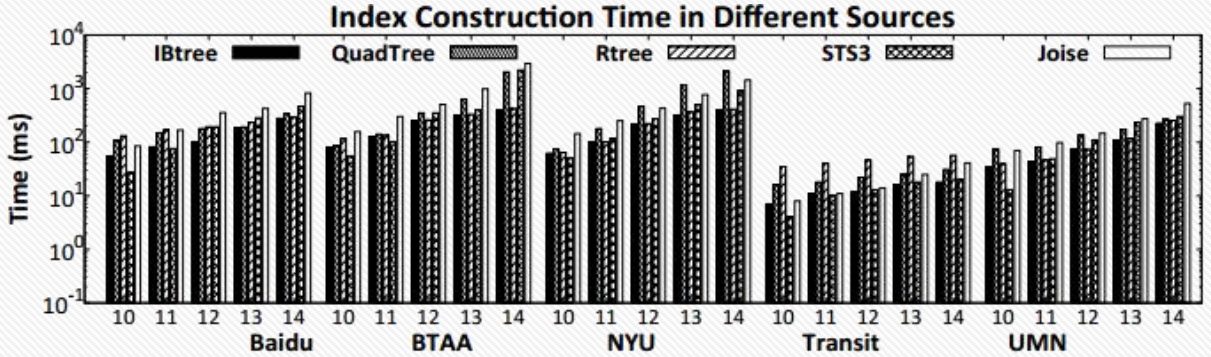


Fig. 10. IBtree Index construction time in MIQ.

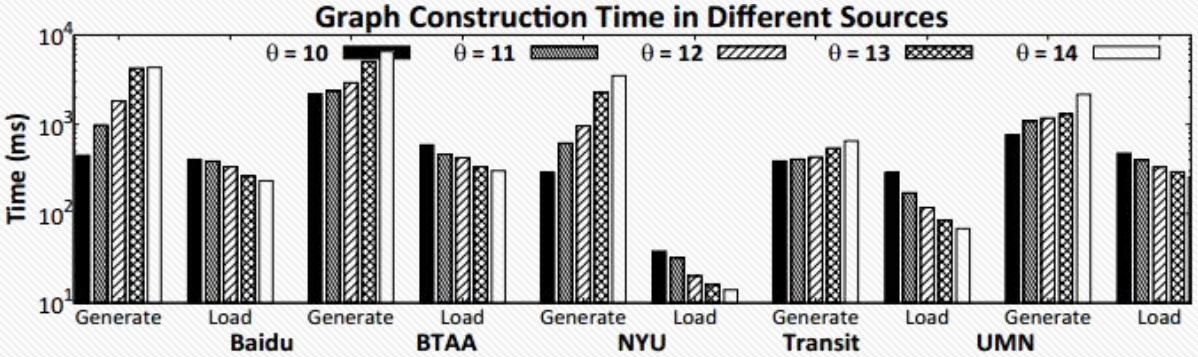


Fig. 11. Dataset graph construction time in MCQC.

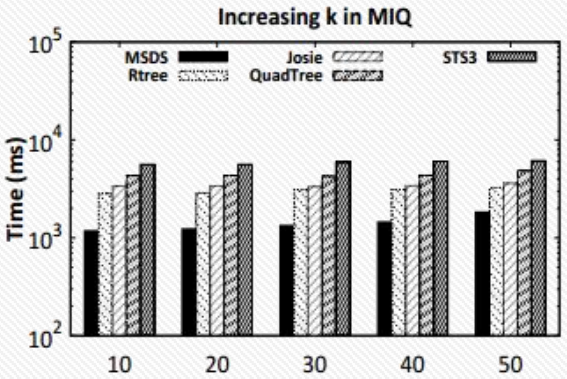


Fig. 12. Top-k search time with the increase of k.

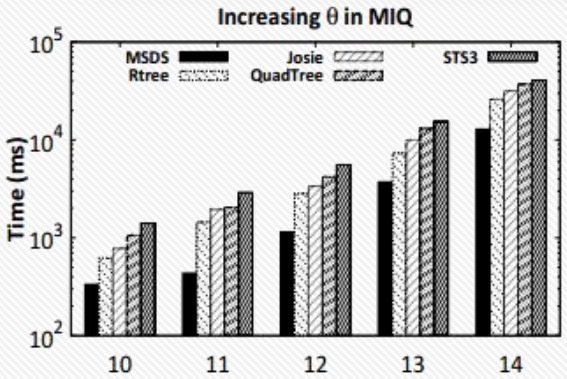


Fig. 13. Top-k search time with the increase of θ.

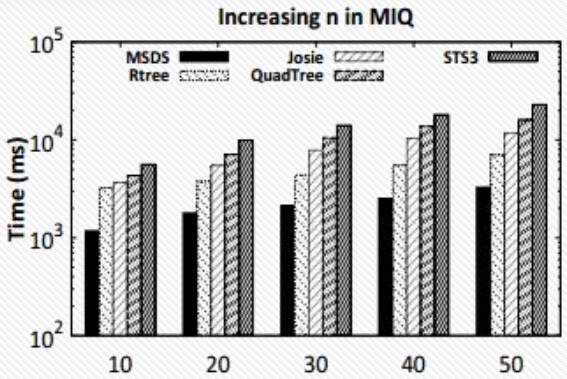


Fig. 14. Top-k search time with the increase of n.

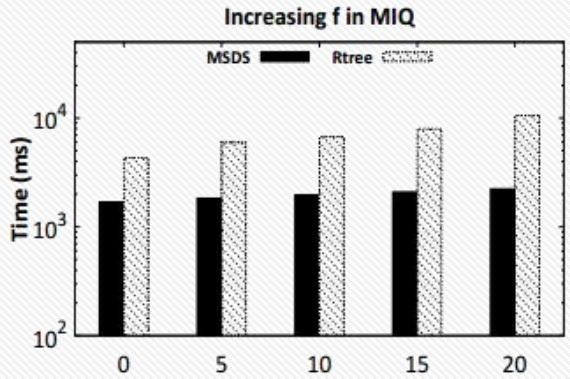


Fig. 15. Top-k search time with the increase of f.

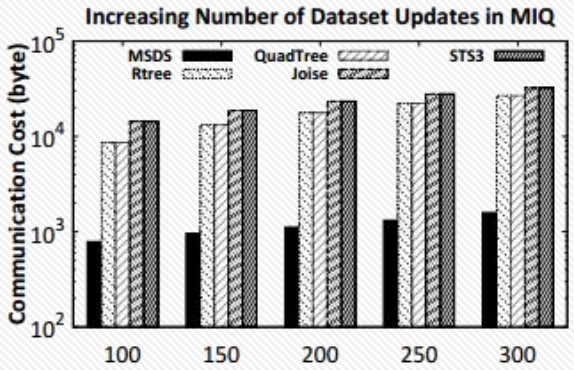


Fig. 16. Communication cost with the increase of updates.

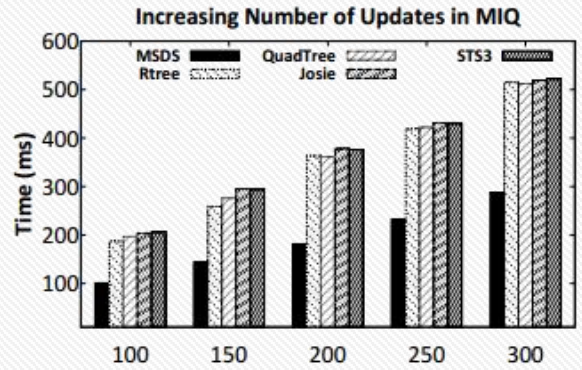


Fig. 17. Transmission time with the increase of updates.

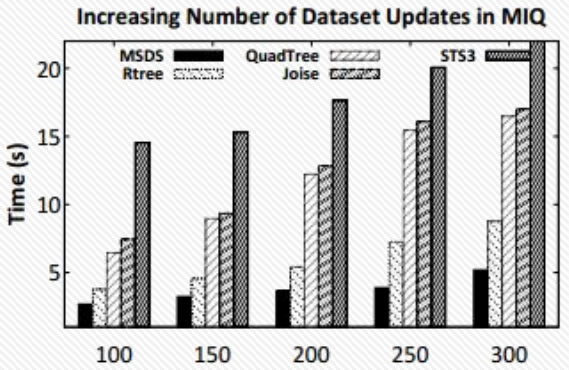


Fig. 18. Search time with the increase of updates.

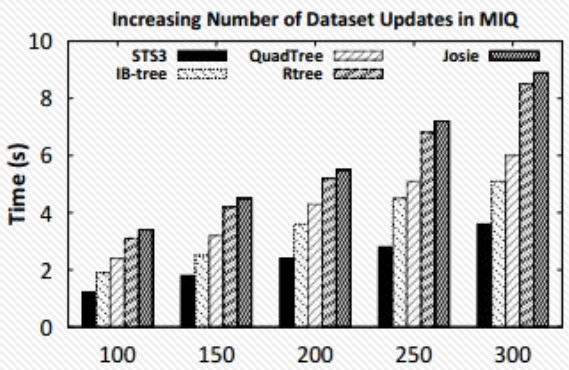


Fig. 19. Index updating time with the increase of updates.

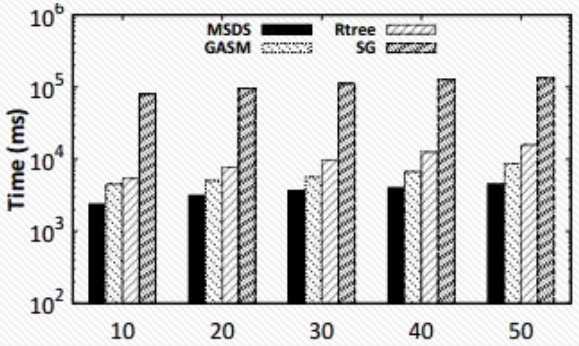


Fig. 20. Top-k search time with the increase of k.

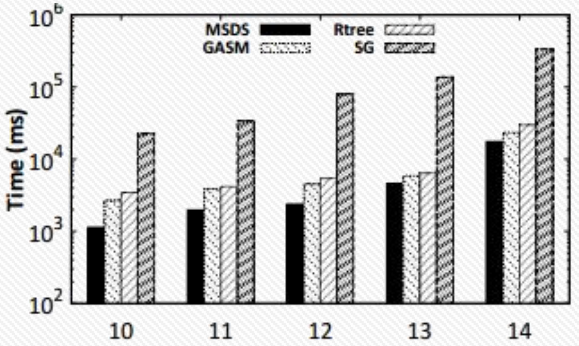


Fig. 21. Top-k search time with the increase of θ.

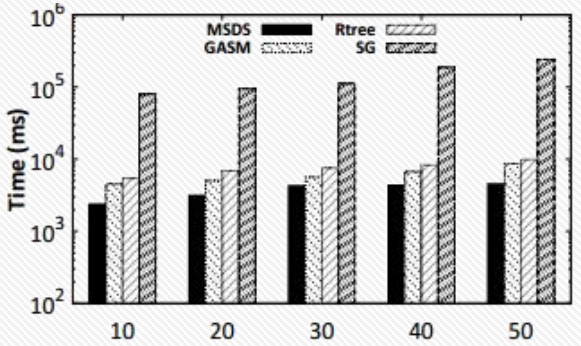


Fig. 22. Top-k search time with the increase of n.

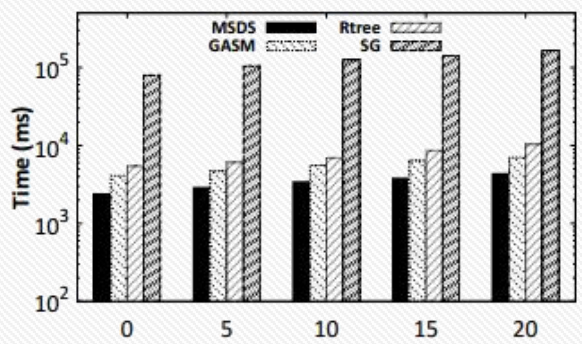


Fig. 23. Top-k search time with the increase of δ.

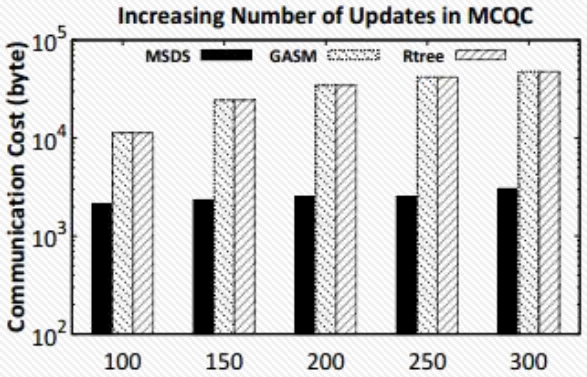


Fig. 24. Communication cost with the increase of updates.

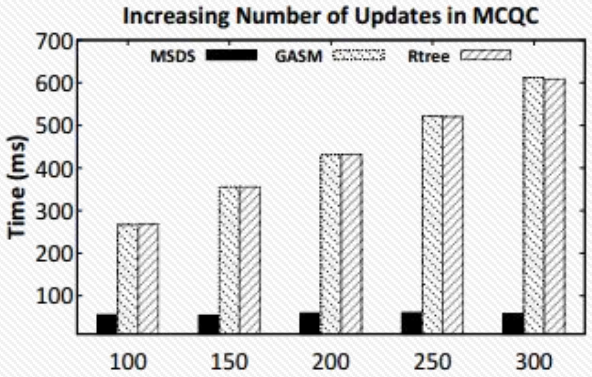


Fig. 25. Transmission time with the increase of updates.

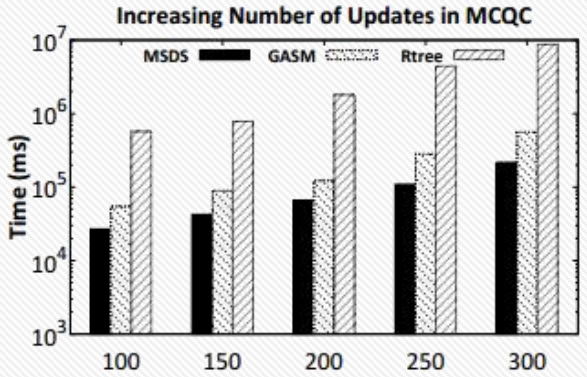


Fig. 26. Search time with the increase of updates.

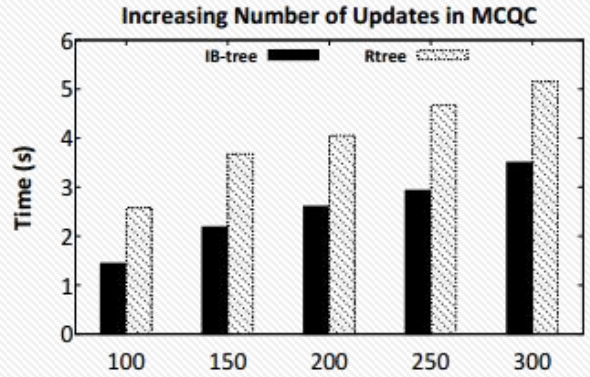


Fig. 27. Index updating time with the increase of updates.

总结

- 定义了两个搜索问题，MIQ和MCQC
- 提出了MSDS框架
- 构建了IBree和Dataset Graph
- 静态搜索中设计了一种基于IBtree的搜索算法和两种启发式贪婪算法
- 动态搜索中设计了索引动态更新策略和top-k动态搜索方法

谢谢