Flexible Aggregate Similarity Search

Yang Li[1], Feifei Li[2], Ke Yi[3], Bin Yao[2], Min Wang[4]

[1]Department of Computer Science and Engineering
Shanghai Jiao Tong University, China

[2]Department of Computer Science
Florida State University, USA

[3]Department of Computer Science
Hong Kong University of Science and Technology

[4]HP Labs China

Outline

Introduction and motivation

- Similarity search (aka nearest neighbor search, NN search) is a fundamental tool in retrieving the most relevant data w.r.t. user input in working with massive data: extensively studied.

- However, often time, users may be interested at retrieving objects that are *similar* to *a group Q of query objects*, *instead of just one*.

Introduction and motivation

- However, often time, users may be interested at retrieving objects that are *similar* to *a group Q of query objects*, *instead of just one*.

- Given an aggregation σ, a similarity/distance function d, a dataset P, and any query group Q:

$$r_p = \sigma\{d(p, Q)\} = \sigma\{d(p, q_1), \ldots, d(p, q_{|Q|})\}, \text{ for any } p$$

aggregate similarity distance of p

Introduction and motivation

- However, often time, users may be interested at retrieving objects that are *similar* to *a group Q of query objects*, *instead of just one*.

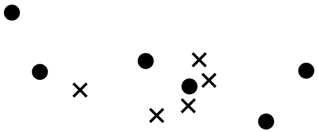
- Given an aggregation σ, a similarity/distance function d, a dataset P, and any query group Q:

$$r_p = \sigma\{d(p, Q)\} = \sigma\{d(p, q_1), \ldots, d(p, q_{|Q|})\}, \text{ for any } p$$

aggregate similarity distance of p

Find $p^* \in P$ having the smallest r_p value ($r_{p^*} = r^*$).

- However, often time, users may be interested at retrieving objects that are *similar* to *a group Q of query objects*, *instead of just one*.



× : group Q of query points

● : dataset P

Figure: Aggregate similarity search in Euclidean space: max and sum.

Introduction and motivation

- However, often time, users may be interested at retrieving objects that are *similar* to *a group Q of query objects*, *instead of just one*.

agg$=$ max, $p^* = p_3$, $r^* = d(p_3, q_1)$

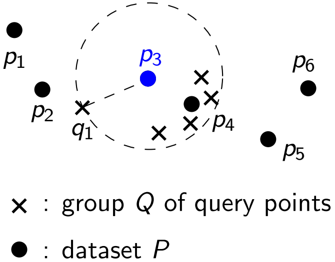


Figure: Aggregate similarity search in Euclidean space: max and sum.

- However, often time, users may be interested at retrieving objects that are *similar* to *a group Q of query objects*, *instead of just one*.

agg=sum, $p^* = p_4$, $r^* = \sum_{q \in Q} d(p_4, q)$

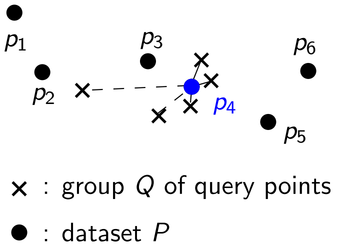


Figure: Aggregate similarity search in Euclidean space: max and sum.

- However, often time, users may be interested at retrieving objects that are *similar* to *a group Q of query objects*, *instead of just one*.
- Aggregate similarity search (ANN) may need to deal with data in high dimensions.

Outline

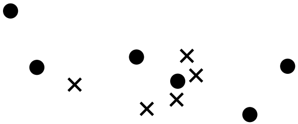
Existing methods for ANN

- R-tree method: branch and bound principle [PSTM04, PTMH05].
- Some other heuristics to further improve the pruning.
- Can be extended to other metric space using M-tree [RBTFT08].
- Limitations:
  - No bound on the query cost.
  - Query cost increases quickly as dataset becomes larger and/or dimension goes higher.

- [PSTM04]: Group Nearest Neighbor Queries. In *ICDE*, 2004.
- [PTMH05]: Aggregate nearest neighbor queries in spatial databases. In *TODS*, 2005.
- [RBTFT08]: A Novel Optimization Approach to Efficiently Process Aggregate Similarity Queries in Metric Access Methods. In *CIKM*, 2008.

Our approach for $\sigma = \max$: AMAX1

- We proposed AMAX1 (TKDE'10):

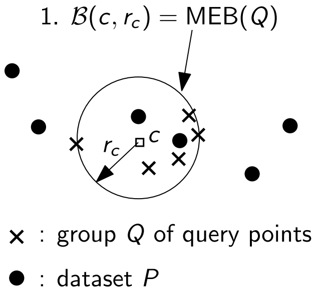


× : group Q of query points

● : dataset P

Our approach for $\sigma = \max$: AMAX1

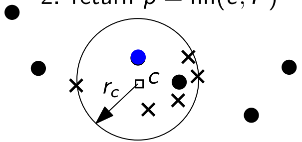
- We proposed AMAX1 (TKDE'10):
  - $\mathcal{B}(c, r_c)$ is a ball centered at c with radius r_c;
  - MEB(Q) is the minimum enclosing ball of a set of points Q;



Our approach for $\sigma = \max$: AMAX1

- We proposed AMAX1 (TKDE'10):
  - $\mathcal{B}(c, r_c)$ is a ball centered at c with radius r_c;
  - MEB(Q) is the minimum enclosing ball of a set of points Q;
  - nn(c, P) is the nearest neighbor of a point c from the dataset P.

1. $\mathcal{B}(c, r_c) = \text{MEB}(Q)$
2. return $p = \text{nn}(c, P)$



× : group Q of query points

● : dataset P

Our approach for $\sigma = \max$: AMAX1

- An algorithm returns (p, r_p) for $\text{ANN}(Q, P)$ is an c-approximation iff $r^* \leq r_p \leq c \cdot r_p$.

Our approach for $\sigma = \max$: AMAX1

- An algorithm returns (p, r_p) for $\textsc{Ann}(Q, P)$ is an c-approximation iff $r^* \leq r_p \leq c \cdot r_p$.

Theorem

AMAX1 *is a* $\sqrt{2}$*-approximation in any dimension* d *given (exact)* $\mathrm{nn}(c, P)$ *and* $\mathrm{MEB}(Q)$*.*

Our approach for $\sigma = \max$: AMAX1

- An algorithm returns (p, r_p) for $\textsc{Ann}(Q, P)$ is an c-approximation iff $r^* \leq r_p \leq c \cdot r_p$.

Theorem

AMAX1 is a $\sqrt{2}$-approximation in any dimension d given (exact) $\mathrm{nn}(c, P)$ and $\mathrm{MEB}(Q)$.

Theorem

In any dimension d, given an α-approximate MEB algorithm and an β-approximate NN algorithm, AMAX1 is an $\sqrt{\alpha^2 + \beta^2}$-approximation.

Our approach for $\sigma = \max$: AMAX1

- An algorithm returns (p, r_p) for $\text{ANN}(Q, P)$ is an c-approximation iff $r^* \leq r_p \leq c \cdot r_p$.
- In low dimensions, BBD-tree [AMNSW98] gives $(1+\epsilon)$-approximate NN search; in high dimensions, LSB-tree [TYSK10] gives $(2+\epsilon)$-approximate NN search with high probability; and $(1+\epsilon)-$ MEB algorithm exists even in high dimensions [KMY03].

Theorem

AMAX1 *is a* $\sqrt{2}$*-approximation in any dimension* d *given (exact)* $\text{nn}(c, P)$ *and* $\text{MEB}(Q)$*.*

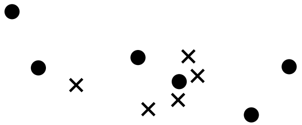
Theorem

In any dimension d*, given an* α*-approximate* MEB *algorithm and an* β*-approximate NN algorithm,* AMAX1 *is an* $\sqrt{\alpha^2+\beta^2}$*-approximation.*

- [AMNSW98]: An Optimal Algorithm for Approximate Nearest Neighbor Searching in Fixed Dimensions. In *JACM*, 1998.
- [TYSK10]: Efficient and Accurate Nearest Neighbor and Closest Pair Search in High Dimensional Space. In *TODS*, 2010.
- [KMY03]: Approximate Minimum Enclosing Balls in High Dimensions Using Core-Sets. In *JEA*, 2003

Our approach for σ = sum: ASUM1

- We proposed ASUM1 (TKDE'10):



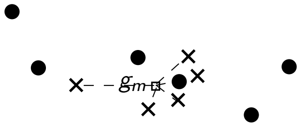
× : group Q of query points

● : dataset P

Our approach for σ = sum: ASUM1

- We proposed ASUM1 (TKDE'10):
  - let g_m be the geometric median of Q;

1. g_m is the geometric median of Q



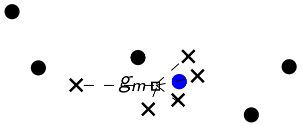
× : group Q of query points

● : dataset P

Our approach for σ = sum: ASUM1

- We proposed ASUM1 (TKDE'10):
  - let g_m be the geometric median of Q;
  - return nn(g_m, P).

1. g_m is the geometric median of Q
2. return nn(g_m, P)



× : group Q of query points

● : dataset P

Our approach for σ = sum: Asum1

- Using the Weiszfeld algorithm (iteratively re-weighted least squares), g_m can be computed to an arbitrary precision efficiently.

Theorem

Asum1 *is a* 3*-approximation in any dimension* d *given (exact) geometric median and* $\text{nn}(c, P)$*.*

Our approach for $\sigma = \text{sum}$: $\textsc{Asum1}$

- Using the Weiszfeld algorithm (iteratively re-weighted least squares), g_m can be computed to an arbitrary precision efficiently.

Theorem

$\textsc{Asum1}$ *is a* 3*-approximation in any dimension* d *given (exact) geometric median and* $\text{nn}(c, P)$.

Theorem

In any dimension d*, given an* β*-approximate NN algorithm,* $\textsc{Asum1}$ *is an* 3β*-approximation.*

Our approach for $\sigma = \text{sum}$: ASUM1

- Using the Weiszfeld algorithm (iteratively re-weighted least squares), g_m can be computed to an arbitrary precision efficiently.
- Both AMAX1 and ASUM1 can be easily extended to work for kANN search while the bounds are maintained.

Theorem

ASUM1 *is a* 3*-approximation in any dimension* d *given (exact) geometric median and* $\text{nn}(c, P)$.

Theorem

In any dimension d*, given an* β*-approximate NN algorithm,* ASUM1 *is an* 3β*-approximation.*

Outline

1. Motivation and Problem Formulation
2. Basic Aggregate Similarity Search
3. Flexible Aggregate Similarity Search
4. Experiments

Definition of flexible aggregate similarity search

- *Flexible aggregate similarity search (*FANN*)*: given support $\phi \in (0, 1]$ and find an object in P that has the best aggregate similarity to (*any*) $\phi|Q|$ query objects (our work in SIGMOD'11).

- *Flexible aggregate similarity search* (FANN): given support $\phi \in (0, 1]$ and find an object in P that has the best aggregate similarity to (*any*) $\phi|Q|$ query objects (our work in SIGMOD'11).

$$\sigma = \max,\ \phi = 40\%,\ p^* = p_4,\ r^* = d(p_4, q_3)$$

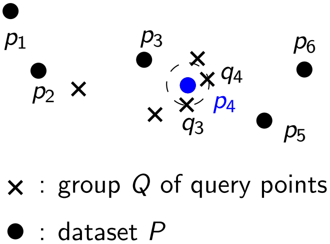


Figure: FANN in Euclidean space: max, $\phi = 0.4$.

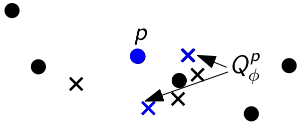
Exact methods for FANN

- For $\forall p \in P$, $r_p = \sigma(p, Q^p_\phi)$, where Q^p_ϕ is p's $\phi|Q|$ NNs in Q.

Exact methods for FANN

- For $\forall p \in P$, $r_p = \sigma(p, Q_\phi^p)$, where Q_ϕ^p is p's $\phi|Q|$ NNs in Q.

Q_ϕ^p : top $\phi|Q|$ NNs of p in Q



× : group Q of query points

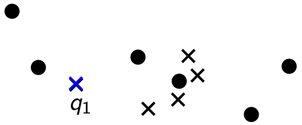
● : dataset P

$\phi = 0.4,\ |Q| = 5,\ \phi|Q| = 2$

Exact methods for FANN

- For $\forall p \in P$, $r_p = \sigma(p, Q^p_\phi)$, where Q^p_ϕ is p's $\phi|Q|$ NNs in Q.
- R-tree method, with the branch and bound principle, can still be applied based on this observation.
- In high dimensions, take the brute-force-search (BFS) approach:
  - For each $p \in P$, find out Q^p_ϕ and calculate r_p.

Approximate methods for σ = sum: ASUM

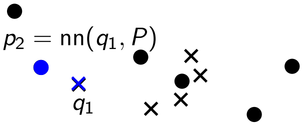


× : group Q of query points

● : dataset P

$\phi = 0.4$, $|Q| = 5$, $\phi|Q| = 2$, σ = sum

Approximate methods for $\sigma = \text{sum}$: ASUM

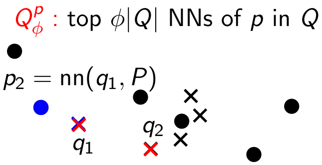


× : group Q of query points

● : dataset P

$\phi = 0.4$, $|Q| = 5$, $\phi|Q| = 2$, $\sigma = \text{sum}$

Approximate methods for $\sigma = \text{sum}$: ASUM



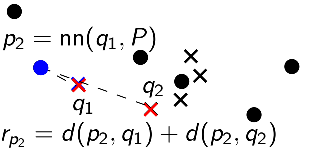
× : group Q of query points

● : dataset P

$\phi = 0.4$, $|Q| = 5$, $\phi|Q| = 2$, $\sigma = \text{sum}$

Approximate methods for $\sigma = \text{sum}$: ASUM

Q^p_ϕ : top $\phi|Q|$ NNs of p in Q



× : group Q of query points

● : dataset P

$\phi = 0.4$, $|Q| = 5$, $\phi|Q| = 2$, $\sigma = \text{sum}$

- Repeat this for every $q_i \in Q$, return the p with the smallest r_p.

Approximation quality of ASUM

Theorem

In any dimension d, given an exact NN algorithm, ASUM *is an 3-approximation.*

Theorem

In any dimension d, given an β-approximate NN algorithm, ASUM *is an $(\beta + 2)$-approximation.*

Approximation quality of ASUM

Theorem

In any dimension d, given an exact NN algorithm, ASUM *is an 3-approximation.*

Theorem

In any dimension d, given an β-approximate NN algorithm, ASUM *is an $(\beta + 2)$-approximation.*

- ASUM only needs $|Q|$ times of NN search in P...

Approximation quality of ASUM

Theorem

In any dimension d, given an exact NN algorithm, ASUM *is an 3-approximation.*

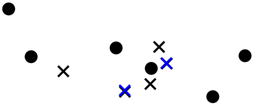
Theorem

In any dimension d, given an β-approximate NN algorithm, ASUM *is an* $(\beta + 2)$*-approximation.*

- ASUM only needs $|Q|$ times of NN search in P...
- ASUM still needs $|Q|$ times of NN search in P!

An improvement to ASUM

randomly select a subset of Q!

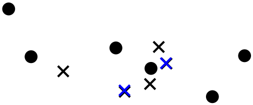


× : group Q of query points

● : dataset P

An improvement to Asum

randomly select a subset of Q!



× : group Q of query points

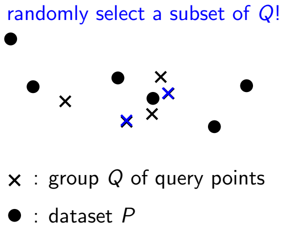
● : dataset P

Theorem

For any $0 < \varepsilon, \lambda < 1$, *executing* Asum *algorithm only on a random subset of* $f(\phi, \varepsilon, \lambda)$ *points of* Q *returns a* $(3+\varepsilon)$*-approximate answer to* Fann *search in any dimensions with probability at least* $1 - \lambda$, *where*

$$f(\phi, \varepsilon, \lambda) = \frac{\log \lambda}{\log(1 - \phi\varepsilon/3)} = O(\log(1/\lambda)/\phi\varepsilon).$$

An improvement to ASUM



Theorem

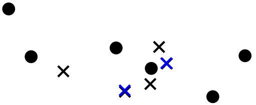
For any $0 < \varepsilon, \lambda < 1$, executing ASUM *algorithm only on a random subset of $f(\phi, \varepsilon, \lambda)$ points of Q returns a $(3+\varepsilon)$-approximate answer to* FANN *search in any dimensions with probability at least $1-\lambda$, where*

$$f(\phi, \varepsilon, \lambda) = \frac{\log \lambda}{\log(1 - \phi\varepsilon/3)} = O(\log(1/\lambda)/\phi\varepsilon).$$

- For $|Q| = 1000$, $\phi = 0.4$, $\lambda = 10\%$, $\varepsilon = 0.5$, only needs 33 NN search in any dimension.

An improvement to ASUM

randomly select a subset of Q!



× : group Q of query points

● : dataset P

Theorem

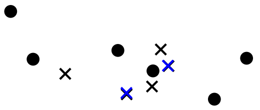
For any $0 < \varepsilon, \lambda < 1$, executing ASUM *algorithm only on a random subset of $f(\phi, \varepsilon, \lambda)$ points of Q returns a $(3+\varepsilon)$-approximate answer to* FANN *search in any dimensions with probability at least $1-\lambda$, where*

$$f(\phi, \varepsilon, \lambda) = \frac{\log \lambda}{\log(1 - \phi\varepsilon/3)} = O(\log(1/\lambda)/\phi\varepsilon).$$

- For $|Q| = 1000$, $\phi = 0.4$, $\lambda = 10\%$, $\varepsilon = 0.5$, only needs 33 NN search in any dimension. (much less in practice, $\frac{1}{\phi}$ is enough!)

An improvement to ASUM

randomly select a subset of Q!



× : group Q of query points

● : dataset P

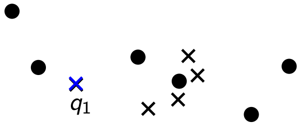
Theorem

For any $0 < \varepsilon, \lambda < 1$, executing ASUM *algorithm only on a random subset of $f(\phi, \varepsilon, \lambda)$ points of Q returns a $(3+\varepsilon)$-approximate answer to* FANN *search in any dimensions with probability at least $1 - \lambda$, where*

$$f(\phi, \varepsilon, \lambda) = \frac{\log \lambda}{\log(1 - \phi\varepsilon/3)} = O(\log(1/\lambda)/\phi\varepsilon).$$

- For $|Q| = 1000$, $\phi = 0.4$, $\lambda = 10\%$, $\varepsilon = 0.5$, only needs 33 NN search in any dimension. (much less in practice, $\frac{1}{\phi}$ is enough!)
- Independent of dimensionality, $|P|$, and $|Q|$!

Approximate methods for $\sigma = \max$: AMAX



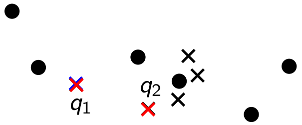
× : group Q of query points

● : dataset P

$\phi = 0.4$, $|Q| = 5$, $\phi|Q| = 2$, $\sigma = \max$

Approximate methods for $\sigma = \max$: AMAX

Q_ϕ^q : top $\phi|Q|$ NNs of q in Q, including q

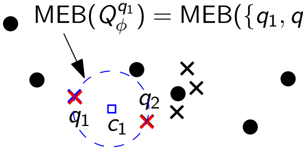


× : group Q of query points

● : dataset P

$\phi = 0.4$, $|Q| = 5$, $\phi|Q| = 2$, $\sigma = \max$

Q_ϕ^q : top $\phi|Q|$ NNs of q in Q, including q

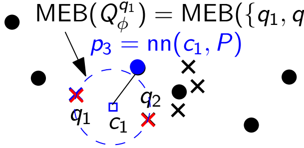


× : group Q of query points

● : dataset P

$\phi = 0.4$, $|Q| = 5$, $\phi|Q| = 2$, $\sigma = \max$

Q_ϕ^q : top $\phi|Q|$ NNs of q in Q, including q



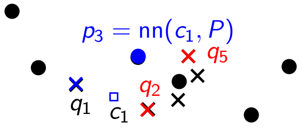
× : group Q of query points

● : dataset P

$\phi = 0.4$, $|Q| = 5$, $\phi|Q| = 2$, $\sigma = \max$

Approximate methods for $\sigma = \max$: Amax

$Q_{\phi}^{p_3}$: top $\phi|Q|$ NNs of p_3 in Q



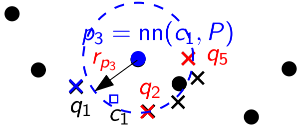
× : group Q of query points

● : dataset P

$\phi = 0.4$, $|Q| = 5$, $\phi|Q| = 2$, $\sigma = \max$

Approximate methods for $\sigma = \max$: AMAX

$Q_\phi^{p_3}$: top $\phi|Q|$ NNs of p_3 in Q



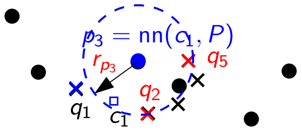
× : group Q of query points

● : dataset P

$\phi = 0.4$, $|Q| = 5$, $\phi|Q| = 2$, $\sigma = \max$

Approximate methods for $\sigma = \max$: AMAX

$Q_\phi^{p_3}$: top $\phi|Q|$ NNs of p_3 in Q



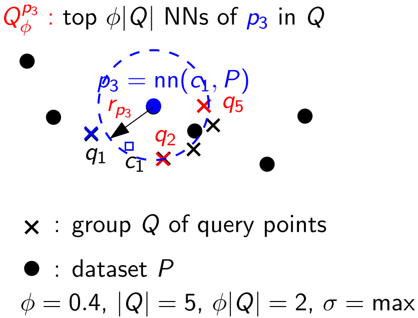
× : group Q of query points

● : dataset P

$\phi = 0.4,\ |Q| = 5,\ \phi|Q| = 2,\ \sigma = \max$

- Repeat this for every $q_i \in Q$, return the p with the smallest r_p.

Approximate methods for $\sigma = \max$: AMAX



- Repeat this for every $q_i \in Q$, return the p with the smallest r_p.
- Identical to ASUM, except using $p = \text{nn}(c_i, P)$ instead of $p = \text{nn}(q_i, P)$, where c_i is the center of $\text{MEB}(q_i, Q_\phi^{q_i})$.

Approximation quality of AMAX

Theorem

In any dimension d, given an exact NN algorithm, AMAX *is an* $(1+2\sqrt{2})$*-approximation.*

Theorem

In any dimension d, given an β*-approximate NN algorithm,* AMAX *is an* $((1+2\sqrt{2})\beta)$*-approximation.*

Approximation quality of AMAX

Theorem

In any dimension d, given an exact NN algorithm, AMAX *is an* $(1+2\sqrt{2})$*-approximation.*

Theorem

In any dimension d, given an β*-approximate NN algorithm,* AMAX *is an* $((1+2\sqrt{2})\beta)$*-approximation.*

- AMAX only needs $|Q|$ times of MEB and $|Q|$ NN search in P...

Approximation quality of AMAX

Theorem

In any dimension d, given an exact NN algorithm, AMAX *is an* $(1 + 2\sqrt{2})$*-approximation.*

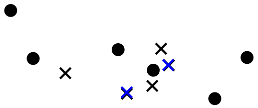
Theorem

In any dimension d, given an β-approximate NN algorithm, AMAX *is an* $((1 + 2\sqrt{2})\beta)$*-approximation.*

- AMAX only needs $|Q|$ times of MEB and $|Q|$ NN search in P...
- AMAX still needs $|Q|$ times of MEB and $|Q|$ NN search in P!

An improvement to AMAX

randomly select a subset of Q!

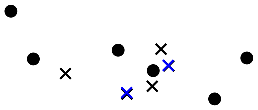


× : group Q of query points

● : dataset P

An improvement to AMAX

randomly select a subset of Q!



$\times$: group Q of query points

$\bullet$: dataset P

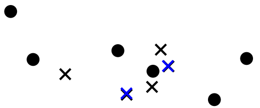
Theorem

For any $0 < \lambda < 1$, *executing* AMAX *algorithm only on a random subset of* $f(\phi, \lambda)$ *points of* Q *returns a* $(1 + 2\sqrt{2})$*-approximate answer to the* FANN *query with probability at least* $1 - \lambda$ *in any dimensions, where*

$$f(\phi, \lambda) = \frac{\log \lambda}{\log(1 - \phi)} = O(\log(1/\lambda)/\phi).$$

An improvement to AMAX

randomly select a subset of Q!



× : group Q of query points

● : dataset P

Theorem

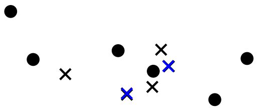
For any $0 < \lambda < 1$, executing AMAX *algorithm only on a random subset of $f(\phi, \lambda)$ points of Q returns a $(1 + 2\sqrt{2})$-approximate answer to the* FANN *query with probability at least $1 - \lambda$ in any dimensions, where*

$$f(\phi, \lambda) = \frac{\log \lambda}{\log(1 - \phi)} = O(\log(1/\lambda)/\phi).$$

- For $|Q| = 1000$, $\phi = 0.4$, $\lambda = 10\%$, only needs 5 MEB and NN search in any dimension.

An improvement to AMAX

randomly select a subset of Q!



× : group Q of query points

● : dataset P

Theorem

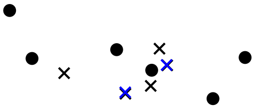
For any $0 < \lambda < 1$, executing AMAX *algorithm only on a random subset of $f(\phi, \lambda)$ points of Q returns a $(1 + 2\sqrt{2})$-approximate answer to the* FANN *query with probability at least $1 - \lambda$ in any dimensions, where*

$$f(\phi, \lambda) = \frac{\log \lambda}{\log(1-\phi)} = O(\log(1/\lambda)/\phi).$$

- For $|Q| = 1000$, $\phi = 0.4$, $\lambda = 10\%$, only needs 5 MEB and NN search in any dimension. (even less in practice, $\frac{1}{\phi}$ is enough!)

An improvement to AMAX

randomly select a subset of Q!



× : group Q of query points

● : dataset P

Theorem

For any $0 < \lambda < 1$, *executing* AMAX *algorithm only on a random subset of* $f(\phi, \lambda)$ *points of* Q *returns a* $(1 + 2\sqrt{2})$*-approximate answer to the* FANN *query with probability at least* $1 - \lambda$ *in any dimensions, where*

$$f(\phi, \lambda) = \frac{\log \lambda}{\log(1 - \phi)} = O(\log(1/\lambda)/\phi).$$

- For $|Q| = 1000$, $\phi = 0.4$, $\lambda = 10\%$, only needs 5 MEB and NN search in any dimension. (even less in practice, $\frac{1}{\phi}$ is enough!)
- Independent of dimensionality, $|P|$, and $|Q|$!

Other issues

- All algorithms for FANN can be extended to work for top-k FANN.

Other issues

- All algorithms for FANN can be extended to work for top-k FANN.
- Most algorithms work for any metric space, except AMAX which works for metric space when MEB is properly defined.

Outline

1. Motivation and Problem Formulation
2. Basic Aggregate Similarity Search
3. Flexible Aggregate Similarity Search
4. Experiments

Experiments: setup and datasets

- Experiments are performed in a Linux machine with 4GB of RAM and an Intel Xeon 2GHz CPU.
- Datasets:
  - 2-dimension: Texas (*TX*) points of interest and road-network dataset from the Open Street Map project: 14 million points (we have other 49 states as well).

Experiments: setup and datasets

- Experiments are performed in a Linux machine with 4GB of RAM and an Intel Xeon 2GHz CPU.
- Datasets:
  - 2-dimension: Texas (*TX*) points of interest and road-network dataset from the Open Street Map project: 14 million points (we have other 49 states as well).
  - 2-6 dimensions: synthetic datasets of random clusters (*RC*).

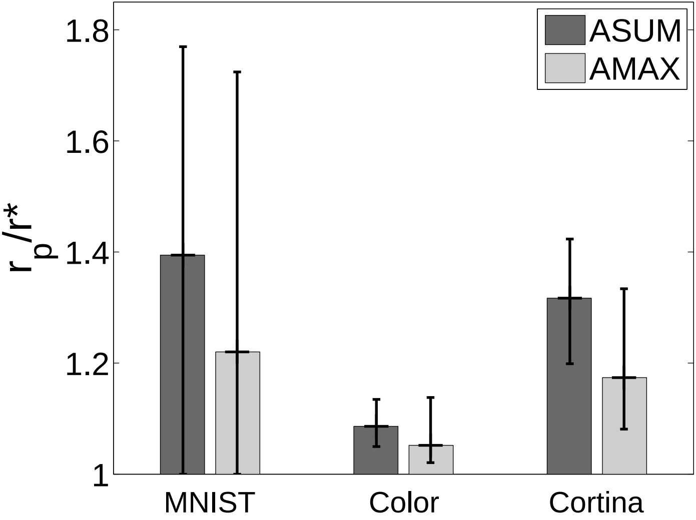
Experiments: setup and datasets

- Experiments are performed in a Linux machine with 4GB of RAM and an Intel Xeon 2GHz CPU.
- Datasets:
  - 2-dimension: Texas (*TX*) points of interest and road-network dataset from the Open Street Map project: 14 million points (we have other 49 states as well).
  - 2-6 dimensions: synthetic datasets of random clusters (*RC*).
  - High dimensions: datasets from http://kdd.ics.uci.edu/databases/CorelFeatures/CorelFeatures.data.html, http://yann.lecun.com/exdb/mnist/, and http://www.scl.ece.ucsb.edu/datasets/index.htm

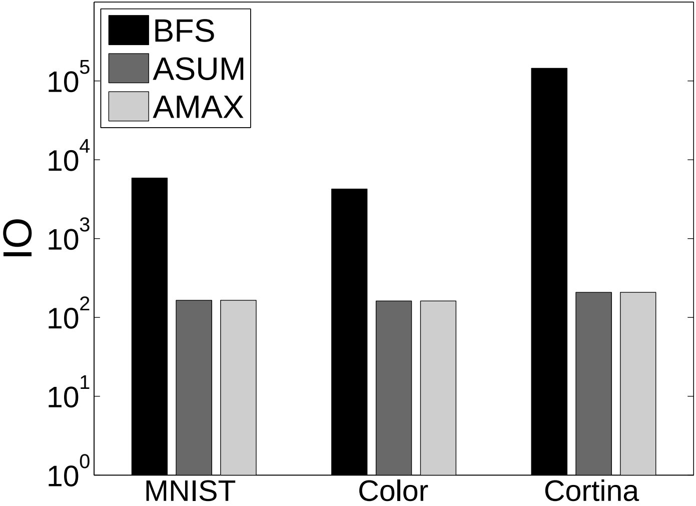
dataset	number of points	dimensionality
TX	$14,000,000$	2
RC	synthetic	$2-6$
Color	$68,040$	32
MNIST	$60,000$	50
Cortina	$1,088,864$	74

  - report the average of 40 independent queries, as well as the 5%-95% interval.
  - sampling rate of $\frac{1}{\phi}$ is enough for both Asum and Amax!

High dimensions: query cost, all datasets



High dimensions: query cost, all datasets

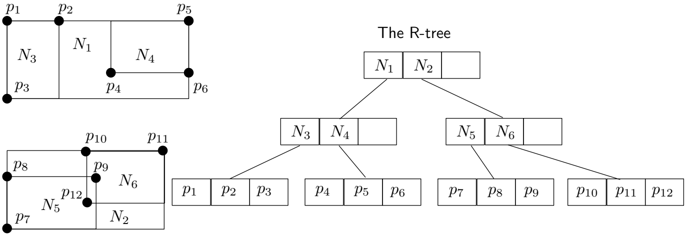


Thank You

Q and A

Existing methods for ANN

- R-tree method: brunch and bound principle [PSTM04, PTMH05].

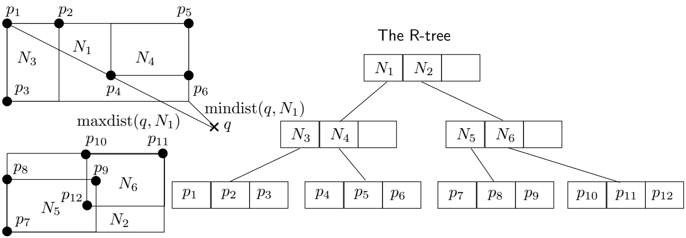


- [PSTM04]: Group Nearest Neighbor Queries. In *ICDE*, 2004.
- [PTMH05]: Aggregate nearest neighbor queries in spatial databases. In *TODS*, 2005.

Existing methods for ANN

- R-tree method: brunch and bound principle [PSTM04, PTMH05].
  - For a query point q and a MBR node N_i:

$$\forall p \in N_i, \text{mindist}(q, N_i) \le d(p, q) \le \text{maxdist}(q, N_i).$$

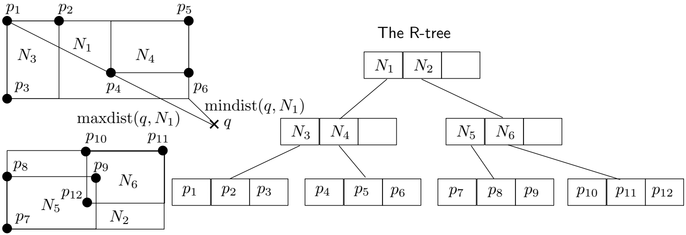


- [PSTM04]: Group Nearest Neighbor Queries. In *ICDE*, 2004.
- [PTMH05]: Aggregate nearest neighbor queries in spatial databases. In *TODS*, 2005.

Existing methods for ANN

- R-tree method: brunch and bound principle [PSTM04, PTMH05].
  - For a query group Q and $\sigma = \max$,

$$\forall p \in N_i, \max_{q \in Q}(\text{mindist}(q, N_i)) \le r_p \le \max_{q \in Q}(\text{maxdist}(q, N_i)).$$

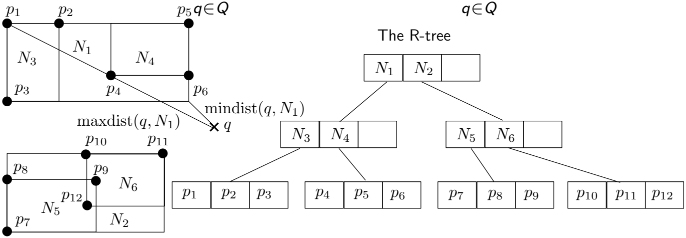


- [PSTM04]: Group Nearest Neighbor Queries. In *ICDE*, 2004.
- [PTMH05]: Aggregate nearest neighbor queries in spatial databases. In *TODS*, 2005.

Existing methods for ANN

- R-tree method: brunch and bound principle [PSTM04, PTMH05].
  - For a query group Q and $\sigma = \text{sum}$,

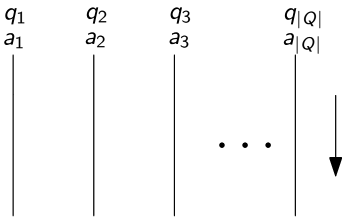
$$\forall p \in N_i, \sum_{q \in Q} \text{mindist}(q, N_i) \leq r_p \leq \sum_{q \in Q} \text{maxdist}(q, N_i).$$



- [PSTM04]: Group Nearest Neighbor Queries. In *ICDE*, 2004.
- [PTMH05]: Aggregate nearest neighbor queries in spatial databases. In *TODS*, 2005.

The List algorithm for FANN

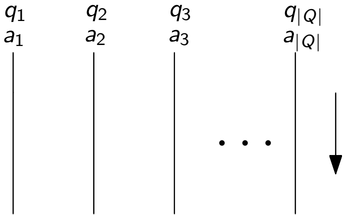
- The List algorithm for any dimensions:



For $\forall p \in P,\ a_i = d(p, q_i)$

The List algorithm for FANN

- The List algorithm for any dimensions:

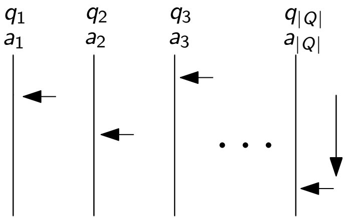


For $\forall p \in P,\ a_i = d(p, q_i)$

$r_p = \sigma(p, Q_\phi^p)$ is monotone w.r.t. a_i's

The List algorithm for FANN

- The List algorithm for any dimensions:



For $\forall p \in P$, $a_i = d(p, q_i)$

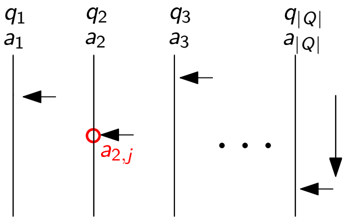
$r_p = \sigma(p, Q_\phi^p)$ is monotone w.r.t. a_i's

apply the *TA algorithm* [FLN01]

- [FLN01]: Optimal Aggregation Algorithms for Middleware. In *PODS*, 2001.

The List algorithm for FANN

- The List algorithm for any dimensions:



For $\forall p \in P$, $a_i = d(p, q_i)$

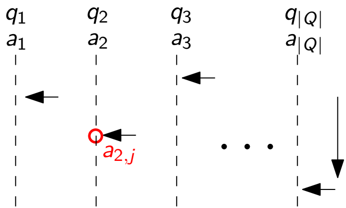
$r_p = \sigma(p, Q_\phi^p)$ is monotone w.r.t. a_i's

apply the *TA algorithm* [FLN01]

$a_{2,j}$ is the jth NN of q_2 in P!

- [FLN01]: Optimal Aggregation Algorithms for Middleware. In *PODS*, 2001.

- The List algorithm for any dimensions:



For $\forall p \in P$, $a_i = d(p, q_i)$

$r_p = \sigma(p, Q_\phi^p)$ is monotone w.r.t. a_i's

apply the *TA algorithm* [FLN01]

$a_{2,j}$ is the jth NN of q_2 in P!

- [FLN01]: Optimal Aggregation Algorithms for Middleware. In *PODS*, 2001.

Experiments: defaults

- Default values:

Symbol	Definition	Default
M	$\|Q\|$	200
ϕ	support	0.5
$\mathcal{A}$	query group volume	5% (of the entire data space)
	points in a query group	random cluster distribution

Experiments: defaults

- Default values:

Symbol	Definition	Default
M	$\lvert Q\rvert$	200
ϕ	support	0.5
$\mathcal{A}$	query group volume	5% (of the entire data space)
	points in a query group	random cluster distribution

- Default values for low dimensions:

Symbol	Definition	Default
N	$\lvert P\rvert$	2,000,000
d	dimensionality	2
	dataset	TX, when $d = 2$
	dataset	RC, when vary d from 2 to 6

Experiments: defaults

- Default values:

Symbol	Definition	Default
M	$\|Q\|$	200
ϕ	support	0.5
$\mathcal{A}$	query group volume	5% (of the entire data space)
	points in a query group	random cluster distribution

- Default values for low dimensions:

Symbol	Definition	Default
N	$\|P\|$	2,000,000
d	dimensionality	2
	dataset	*TX*, when $d = 2$
	dataset	*RC*, when vary d from 2 to 6

- Values for high dimensions:

Symbol	Definition	Default
N	$\|P\|$	200,000
d	dimensionality	30
	default dataset	*Cortina*

Experiments: defaults

- Default values:

Symbol	Definition	Default
M	$\|Q\|$	200
ϕ	support	0.5
$\mathcal{A}$	query group volume	5% (of the entire data space)
	points in a query group	random cluster distribution

- Default values for low dimensions:

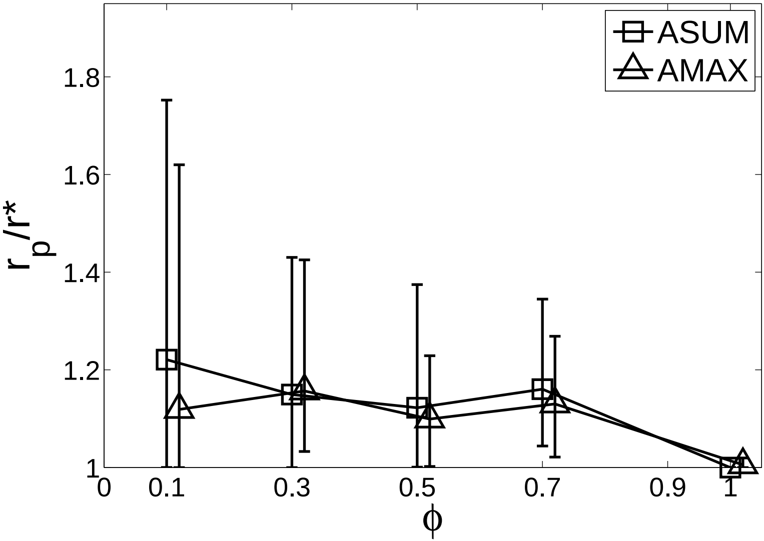
Symbol	Definition	Default
N	$\|P\|$	2,000,000
d	dimensionality	2
	dataset	*TX*, when $d = 2$
	dataset	*RC*, when vary d from 2 to 6

- Values for high dimensions:

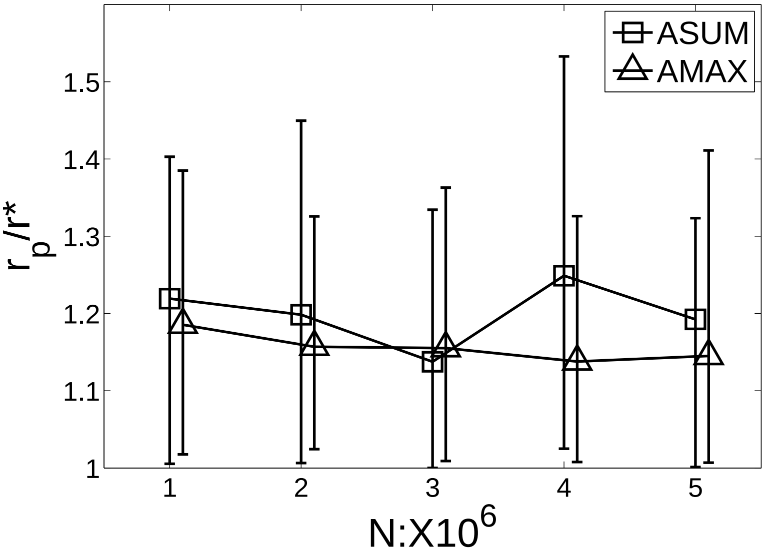
Symbol	Definition	Default
N	$\|P\|$	200,000
d	dimensionality	30
	default dataset	*Cortina*

- report the average of 40 independent, randomly generated queries, as well as the 5%-95% interval.
- sampling rate of $\frac{1}{\phi}$ is enough for both ASUM and AMAX!

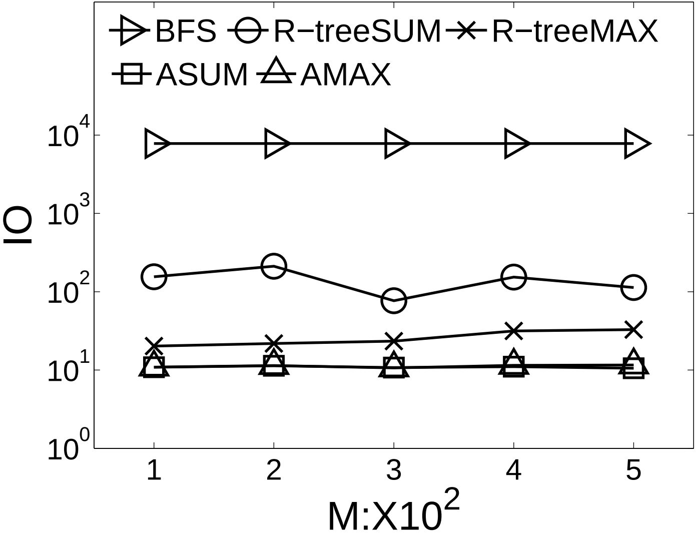
Low dimensions: approximation quality



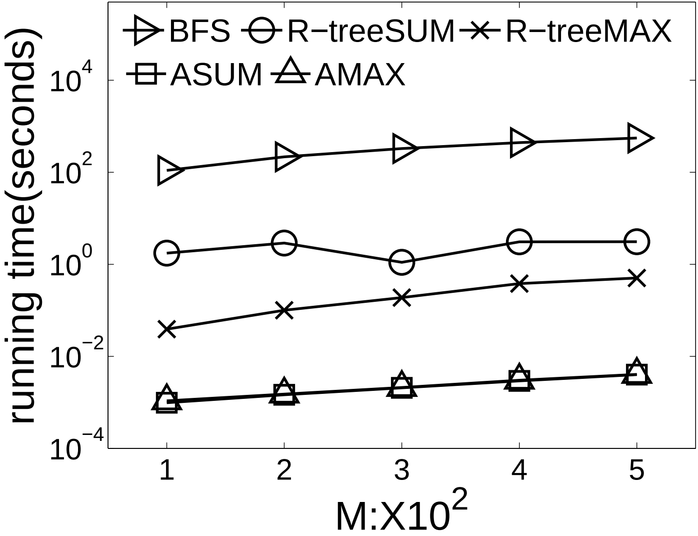
Low dimensions: approximation quality



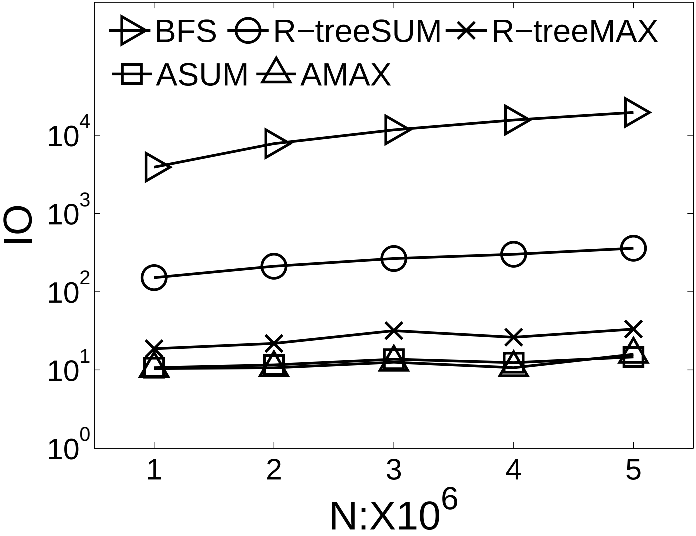
Low dimensions: query cost, vary M



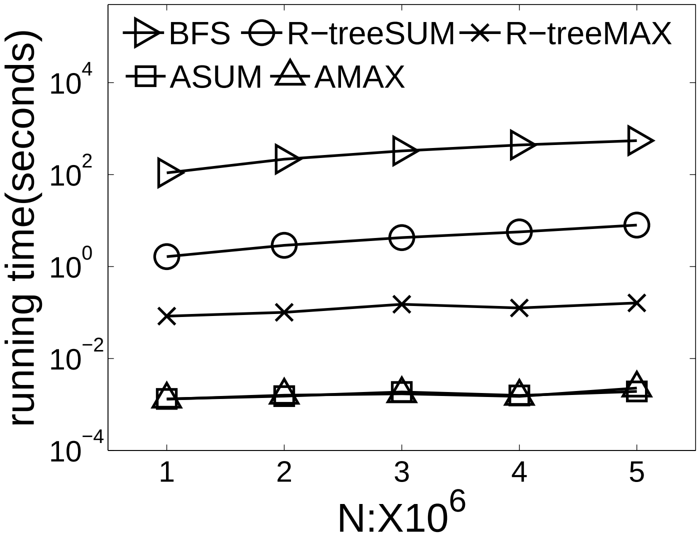
Low dimensions: query cost, vary M



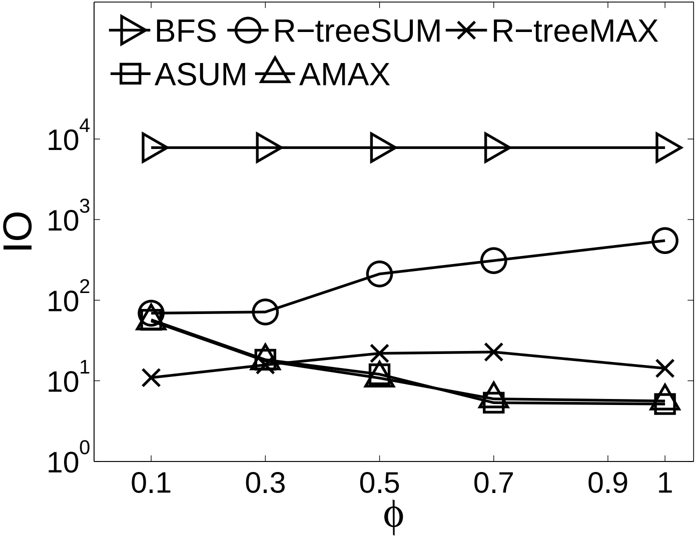
Low dimensions: query cost, vary N



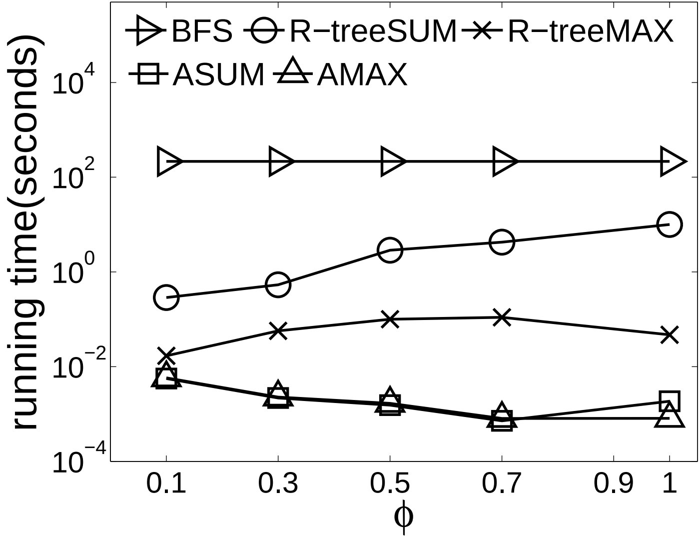
Low dimensions: query cost, vary N



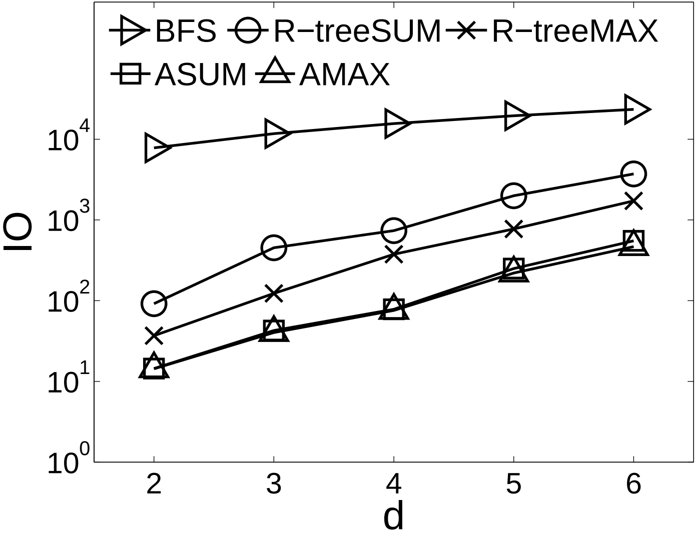
Low dimensions: query cost, vary ϕ



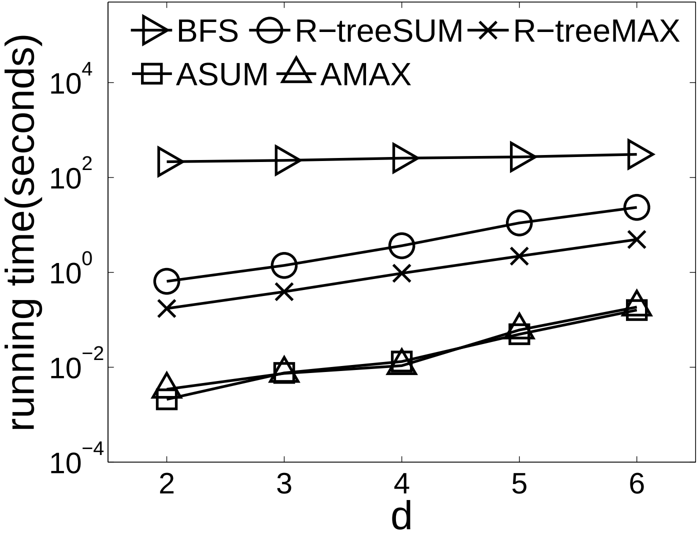
Low dimensions: query cost, vary ϕ



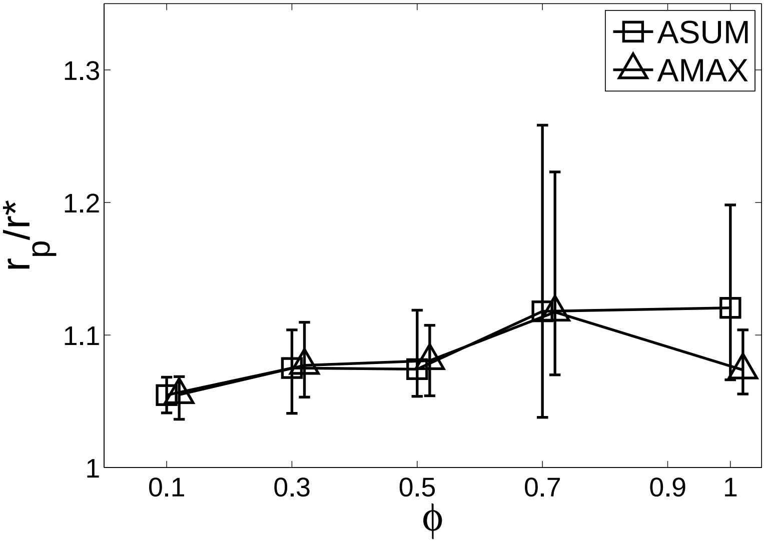
Low dimensions: query cost, vary *d*



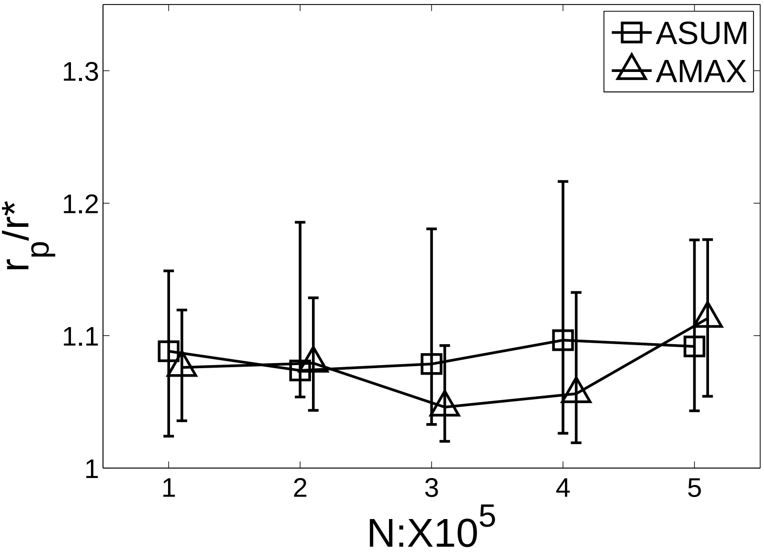
Low dimensions: query cost, vary d



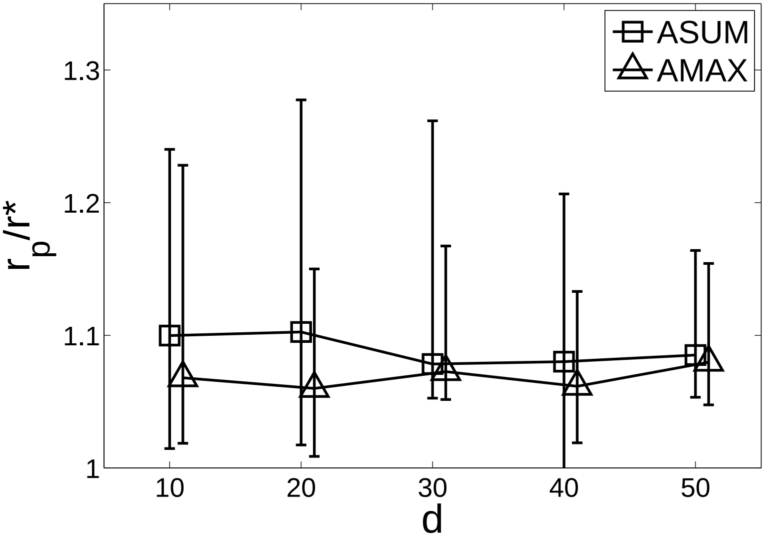
High dimensions: approximation quality



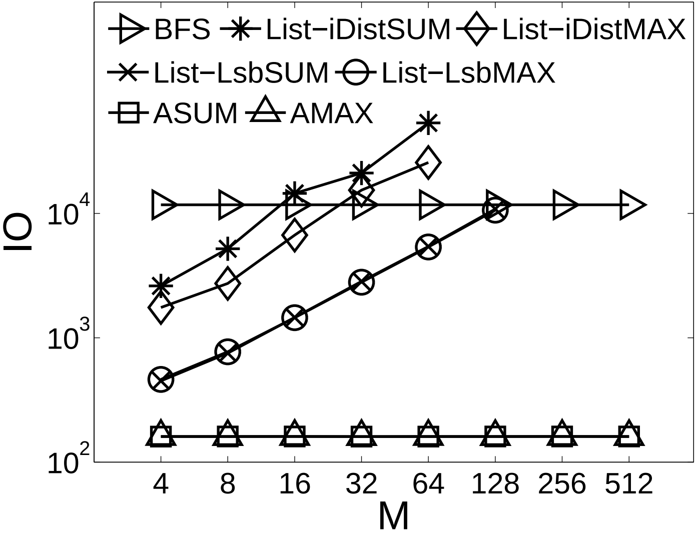
High dimensions: approximation quality



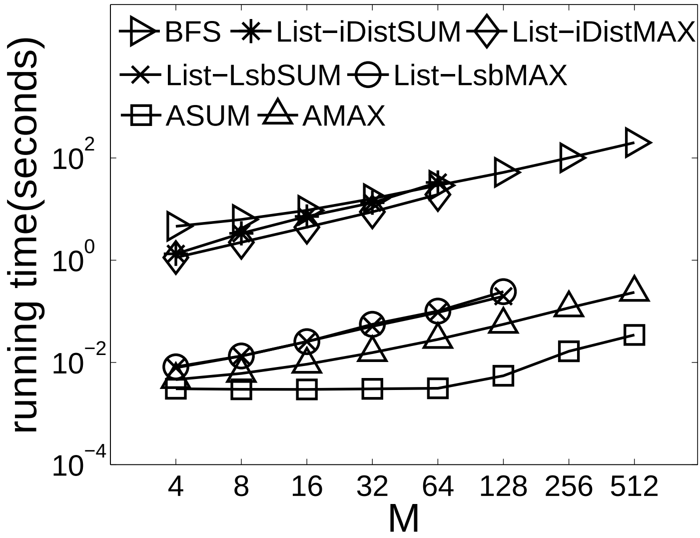
High dimensions: approximation quality



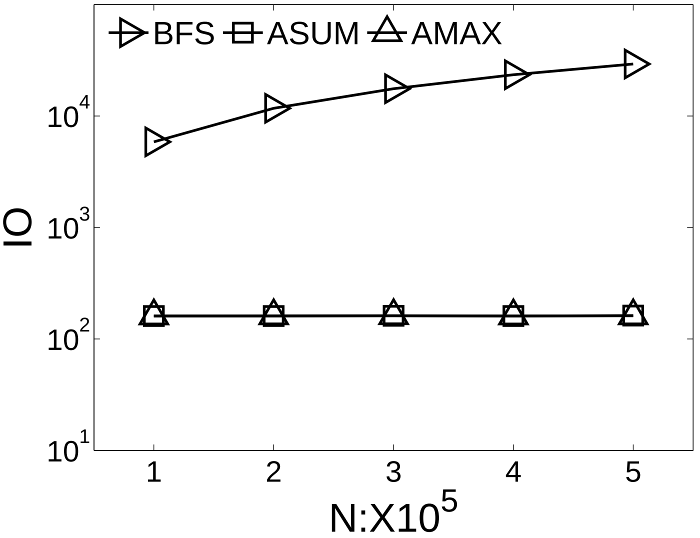
High dimensions: query cost, vary M



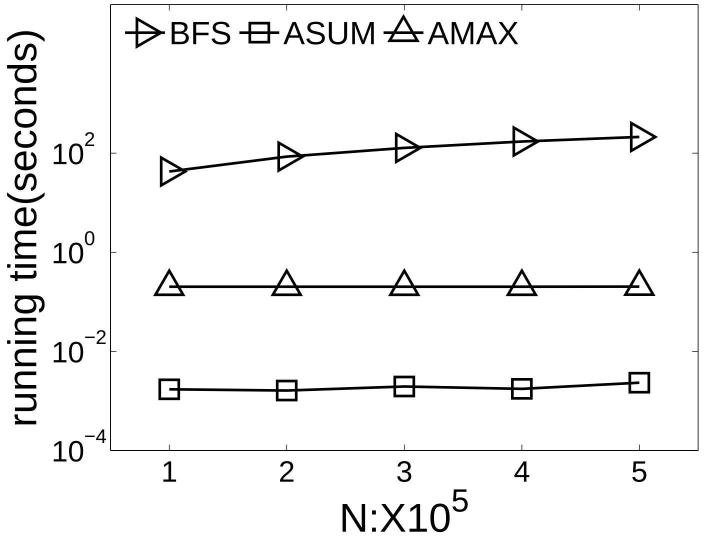
High dimensions: query cost, vary M



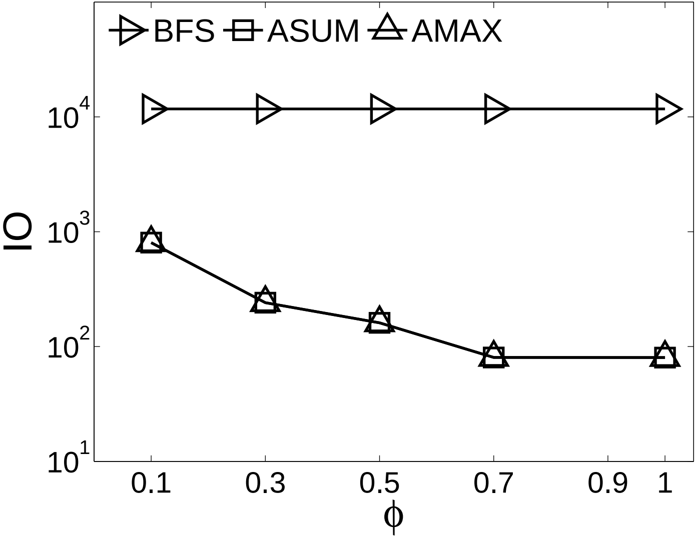
High dimensions: query cost, vary N



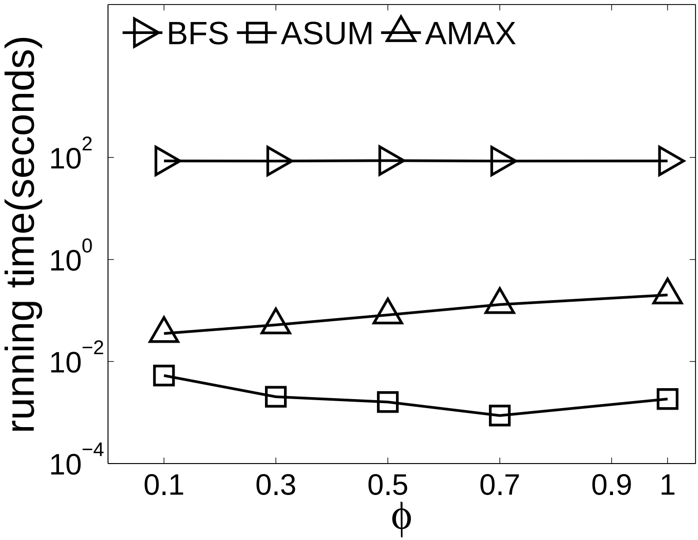
High dimensions: query cost, vary N



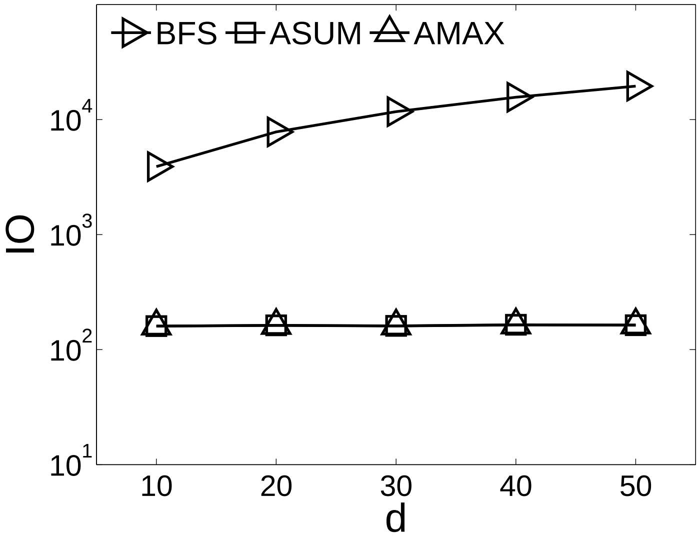
High dimensions: query cost, vary ϕ



High dimensions: query cost, vary ϕ



High dimensions: query cost, vary d



High dimensions: query cost, vary d