Dual-tree Algorithms in Statistics

Ryan Riegel

rriegel@cc.gatech.edu

Computational Science and Engineering

College of Computing

Georgia Institute of Technology

Outline

(Relevant citations at top of slide)

1. Recap of yesterday: single-tree algorithms
2. Motivation and intuition for dual-tree algorithms
3. Several examples, including demo of All-NN
4. Case study #1: quasar identification
5. Formal algebraic foundations
6. The general algorithm and its parameters
7. Case study #2: affinity propagation

Recap

Yesterday, we considered a problem best solved by a single-tree algorithm:

- Given one query and a set of references, determine the sum of forces acting on the query

Recap

Barnes-Hut solution approach:

- Form a spatial tree (e.g. oct-tree) on the references
- For each query, process nodes:
  - If $\frac{R}{W} > thresh$, approximate with center of mass
  - Else, recurse on the node and sum up child results

Recap

Barnes-Hut solution approach:

- Form a spatial tree (e.g. oct-tree) on the references
- For each query, process nodes:
  - If $\frac{R}{W} > thresh$, approximate with center of mass
  - Else, recurse on the node and sum up child results

Reasoning about the potential function ($\frac{1}{r^2}$) permits bounded error via choice of threshold.

Recap

Fast Multi-pole Method is similar:

- Annotate spatial tree with order expansion statistics (fast bottom-up computation)
- For each query, process nodes:
  - If $\frac{R}{W} > thresh$, approximate with order expansion
  - Else, recurse on the node and sum up child results

Recap

Fast Multi-pole Method is similar:

- Annotate spatial tree with order expansion statistics (fast bottom-up computation)
- For each query, process nodes:
  - If $\frac{R}{W} > thresh$, approximate with order expansion
  - Else, recurse on the node and sum up child results

Added accuracy of order expansion permits more aggressive pruning while still with bounded error.

Motivation

Complexity analysis:

- Tree-building is $O(N \log N)$: $O(N)$ work at each level, $O(\log N)$ levels (in a balanced tree)
- Work is $O(\log N)$ per query; $O(M \log N)$ overall

Motivation

Consider $M \in O(N)$:

Motivation

Consider $M \in O(N)$:

- Theorist's response: "What's the problem?"; overall computation is already $O(N \log N)$ from tree-building

Motivation

Consider $M \in O(N)$:

- Theorist's response: "What's the problem?"; overall computation is already $O(N \log N)$ from tree-building
- Maybe the tree already exists

Motivation

Consider $M \in O(N)$:

- Theorist's response: "What's the problem?"; overall computation is already $O(N \log N)$ from tree-building
- Maybe the tree already exists
- Tree-building tends to be very fast

Motivation

Gray and Moore, NIPS 2000

Dual-tree algorithms (a.k.a. generalized N-body methods):

Motivation

Gray and Moore, NIPS 2000

Dual-tree algorithms (a.k.a. generalized N-body methods):

- The most logical extension of single-tree algorithms: form trees for references and queries

Motivation

Gray and Moore, NIPS 2000

Dual-tree algorithms (a.k.a. generalized N-body methods):

- The most logical extension of single-tree algorithms: form trees for references and queries
- After tree-building, time improved $O(N \log N) \rightsquigarrow O(N)$; much better than traditional $O(N^2)$ for nested loops

Motivation

Gray and Moore, NIPS 2000

Dual-tree algorithms (a.k.a. generalized N-body methods):

- The most logical extension of single-tree algorithms: form trees for references and queries
- After tree-building, time improved $O(N \log N) \rightsquigarrow O(N)$; much better than traditional $O(N^2)$ for nested loops
- Yield exact results or have bounded approximation error (absolute or relative)

Motivation

Gray and Moore, NIPS 2000

Dual-tree algorithms (a.k.a. generalized N-body methods):

- The most logical extension of single-tree algorithms: form trees for references and queries
- After tree-building, time improved $O(N \log N) \rightsquigarrow O(N)$; much better than traditional $O(N^2)$ for nested loops
- Yield exact results or have bounded approximation error (absolute or relative)
- Track record: fastest, most accurate methods to date

Hype

Gray and Moore, NIPS 2000
Many other papers

Applications include:

- Nonparametric methods in machine learning:

Hype

Gray and Moore, NIPS 2000
Many other papers

Applications include:

- Nonparametric methods in machine learning:
  - The n-point corrleation and range-count

Hype

Gray and Moore, NIPS 2000
Many other papers

Applications include:

- Nonparametric methods in machine learning:
  - The n-point corrleation and range-count
  - All-k-nearest-neighbors (All-NN)

Hype

Gray and Moore, NIPS 2000
Many other papers

Applications include:

- Nonparametric methods in machine learning:
  - The n-point corrleation and range-count
  - All-k-nearest-neighbors (All-NN)
  - Kernel density estimation (KDE)

Hype

Gray and Moore, NIPS 2000
Many other papers

Applications include:

- Nonparametric methods in machine learning:
  - The n-point corrleation and range-count
  - All-k-nearest-neighbors (All-NN)
  - Kernel density estimation (KDE)
  - Kernel discriminant analysis (KDA)

Hype

Gray and Moore, NIPS 2000
Many other papers

Applications include:

- Nonparametric methods in machine learning:
  - The n-point corrleation and range-count
  - All-k-nearest-neighbors (All-NN)
  - Kernel density estimation (KDE)
  - Kernel discriminant analysis (KDA)
  - Local linear regression and others

Hype

Gray and Moore, NIPS 2000
Many other papers

Applications include:

- Nonparametric methods in machine learning:
  - The n-point corrleation and range-count
  - All-k-nearest-neighbors (All-NN)
  - Kernel density estimation (KDE)
  - Kernel discriminant analysis (KDA)
  - Local linear regression and others
  - More...

Hype

Gray and Moore, NIPS 2000
Many other papers

Applications include:

- Nonparametric methods in machine learning
- Manifold methods via All-NN and others

Hype

Gray and Moore, NIPS 2000
Many other papers

Applications include:

- Nonparametric methods in machine learning
- Manifold methods via All-NN and others
- Astronomy: quasar identification via KDA

Hype

Gray and Moore, NIPS 2000
Many other papers

Applications include:

- Nonparametric methods in machine learning
- Manifold methods via All-NN and others
- Astronomy: quasar identification via KDA
- Physics: multi-body potentials, fitting wave functions

Hype

Gray and Moore, NIPS 2000
Many other papers

Applications include:

- Nonparametric methods in machine learning
- Manifold methods via All-NN and others
- Astronomy: quasar identification via KDA
- Physics: multi-body potentials, fitting wave functions
- Biology: protein folding, solvent-accessible surfaces

Hype

Gray and Moore, NIPS 2000
Many other papers

Applications include:

- Nonparametric methods in machine learning
- Manifold methods via All-NN and others
- Astronomy: quasar identification via KDA
- Physics: multi-body potentials, fitting wave functions
- Biology: protein folding, solvent-accessible surfaces
- (I conjecture) products of sparse matrices and other LA

Intuition

Gray and Moore, NIPS 2000

General algorithmic sketch:

- Form spatial trees for both queries and references
- For pairs of tree nodes:
  - If “bounds” suggest a result for the pair, use it
  - Else, recurse on all pairs of child nodes

Intuition

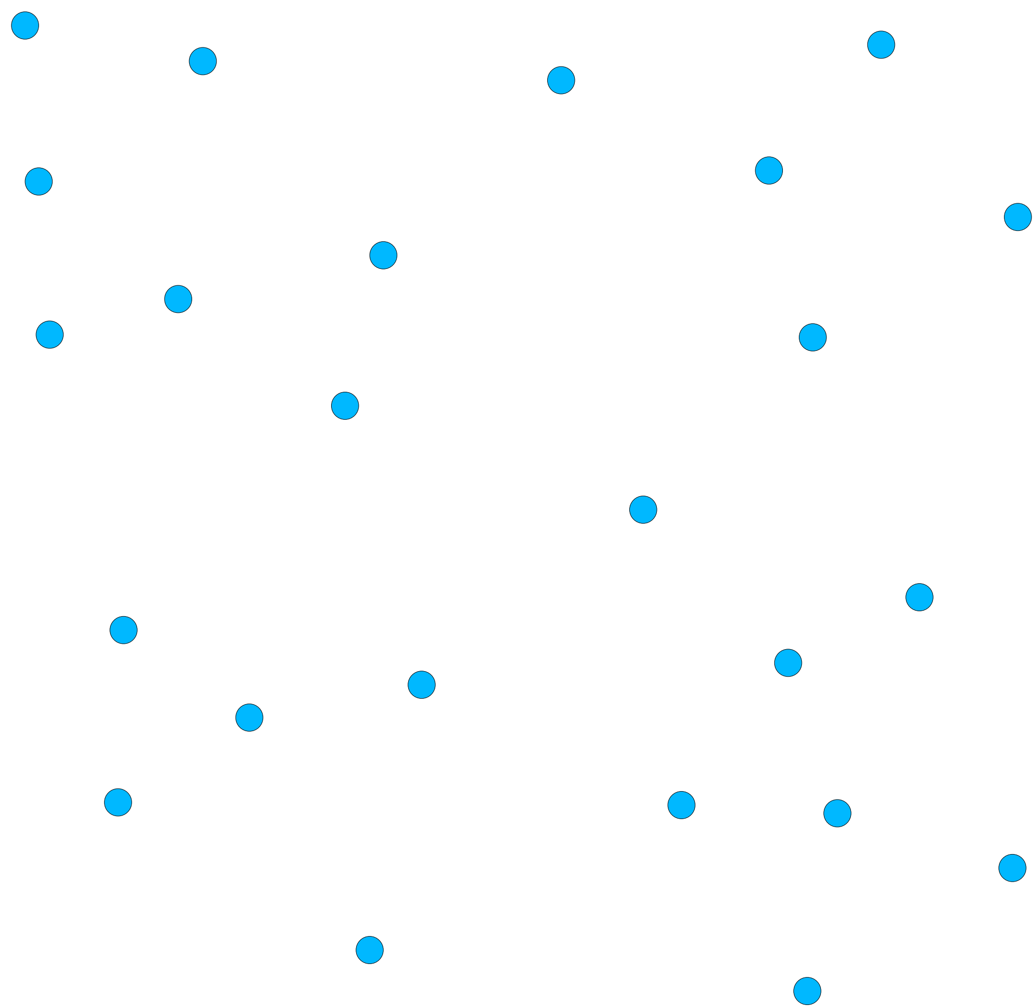
Gray and Moore, NIPS 2000

General algorithmic sketch:

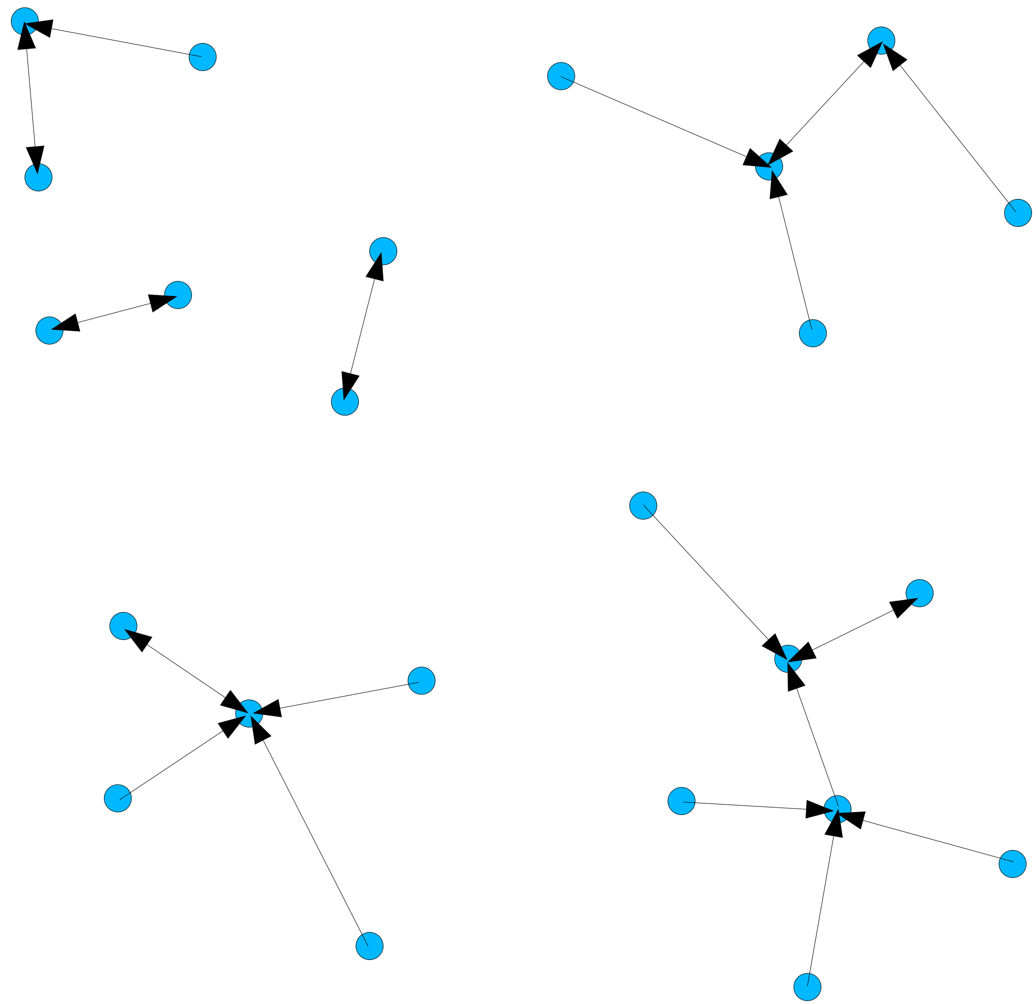
- Form spatial trees for both queries and references
- For pairs of tree nodes:
  - If “bounds” suggest a result for the pair, use it
  - Else, recurse on all pairs of child nodes

“Bounds” are often based on min/max distances between nodes; e.g. the range of a kernel applied to the distances.

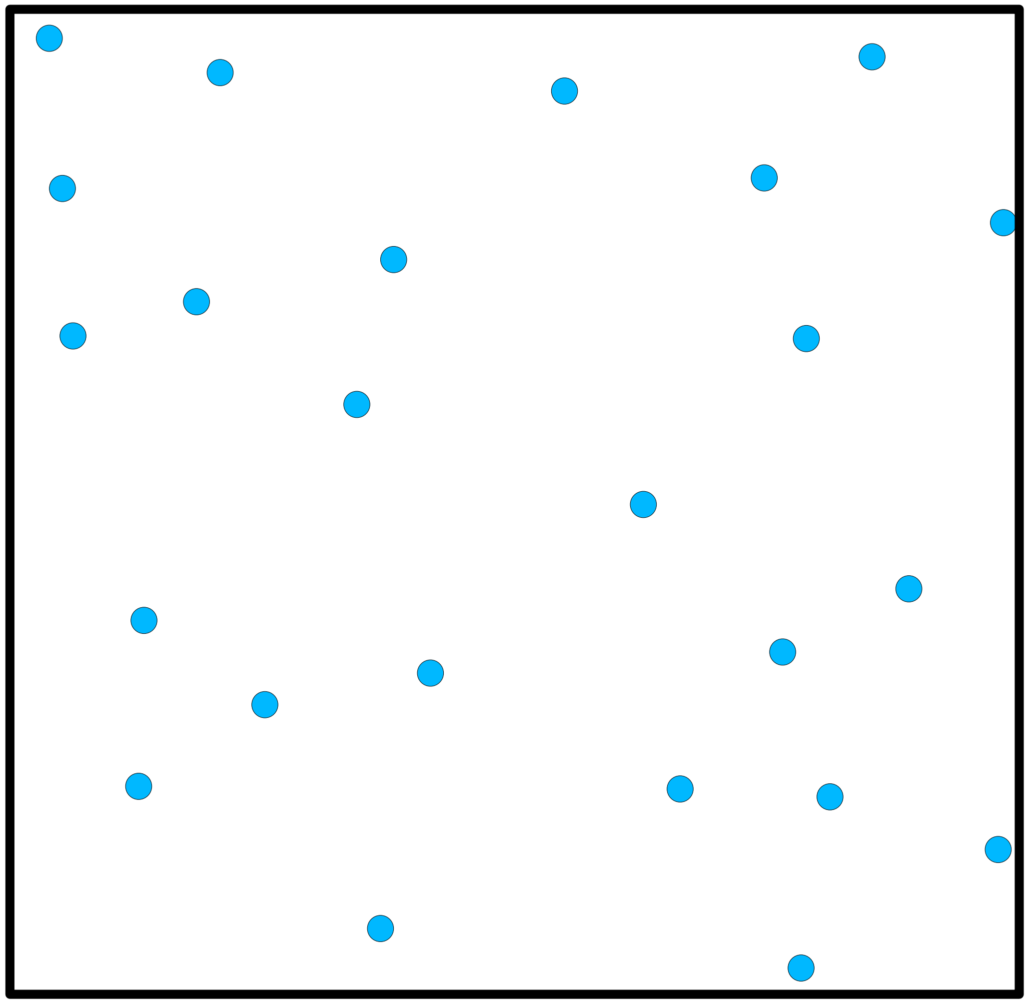
Monochromatic all-nearest-neighbors: $\operatorname*{map}_{q \in X} \operatorname*{argmin}_{r \in X - q} d(q, r)$



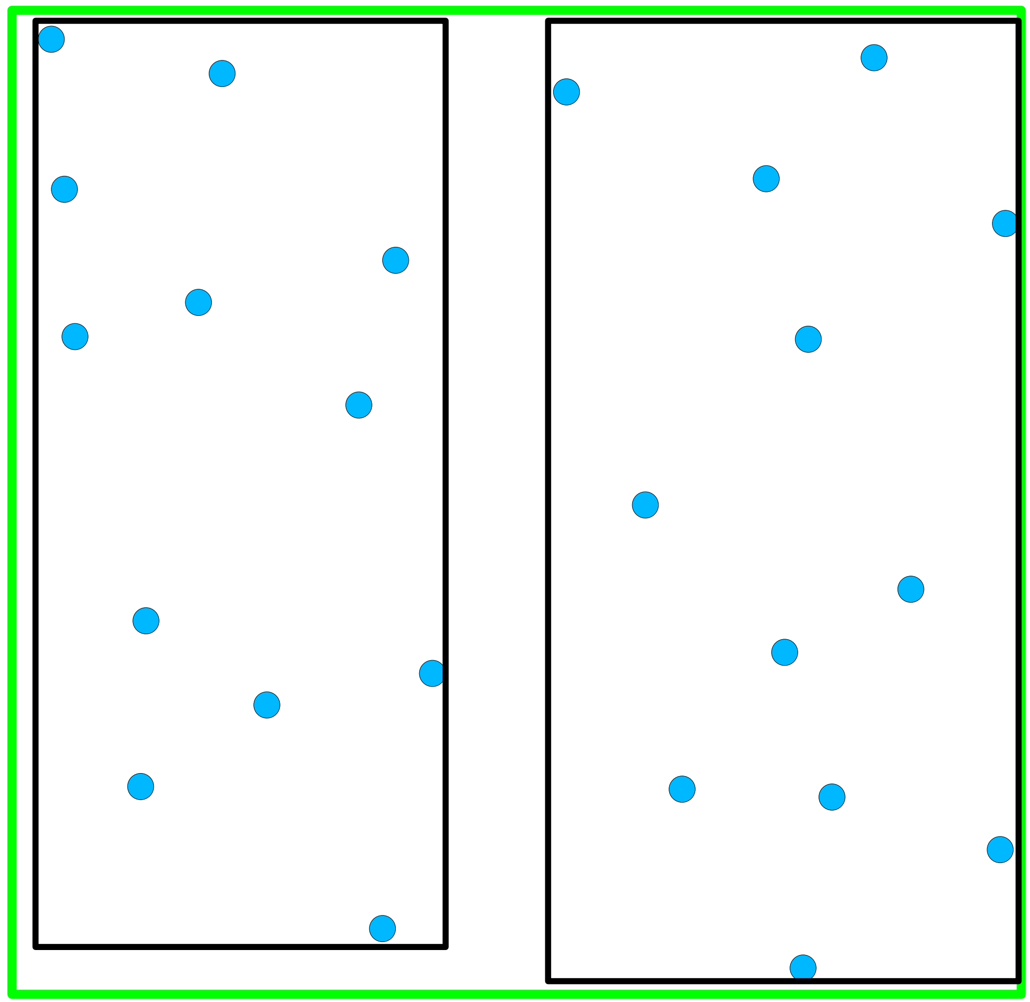
Monochromatic all-nearest-neighbors: $\operatorname*{map}_{q \in X} \operatorname*{argmin}_{r \in X - q} d(q, r)$



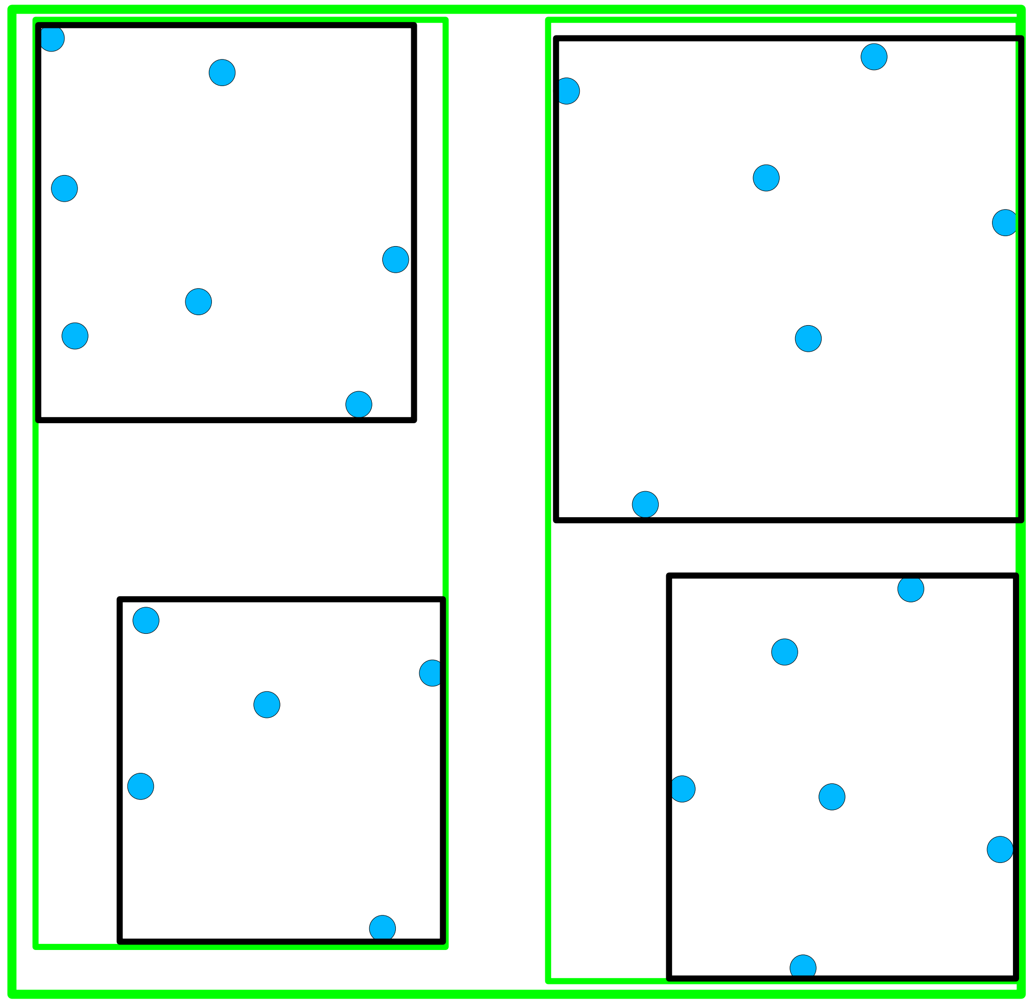
Monochromatic all-nearest-neighbors: $\operatorname*{map}_{q \in X} \operatorname*{argmin}_{r \in X - q} d(q, r)$



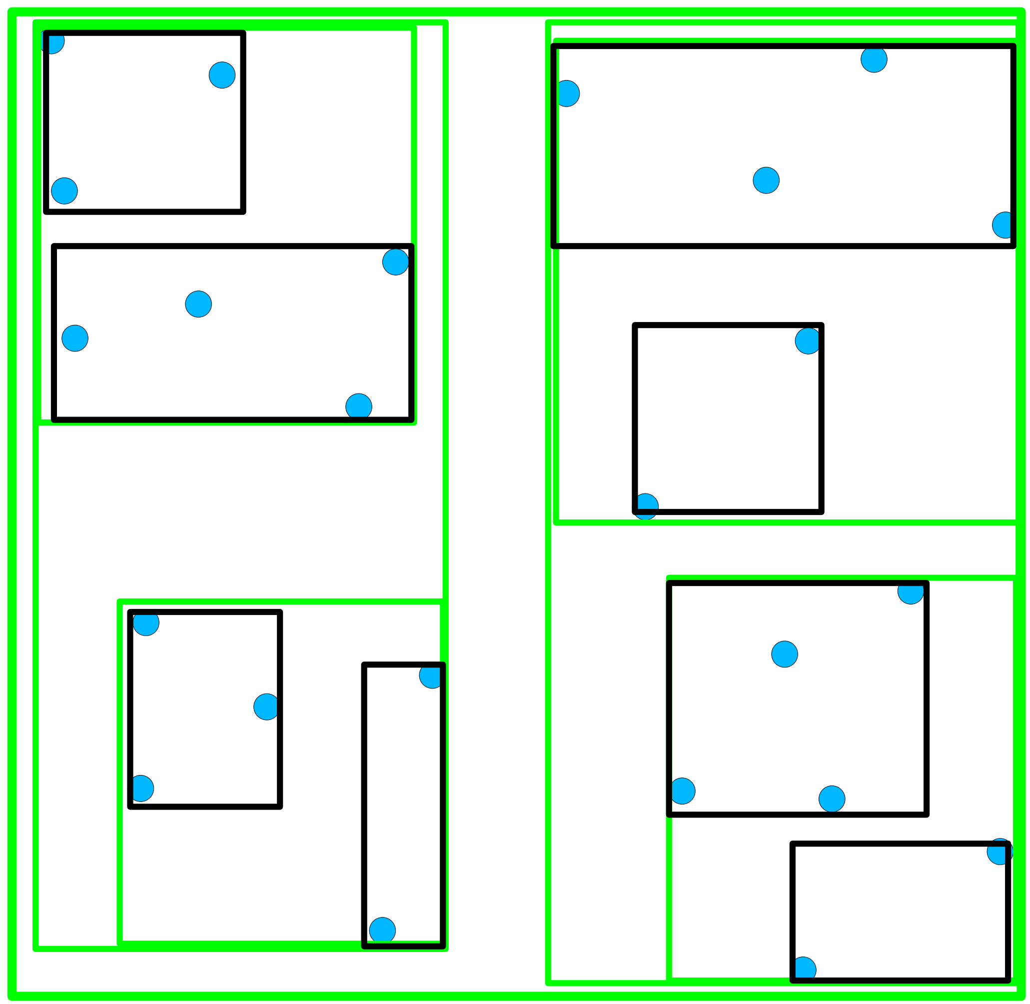
Monochromatic all-nearest-neighbors: $\operatorname{map}_{q \in X} \operatorname{argmin}_{r \in X - q} d(q, r)$



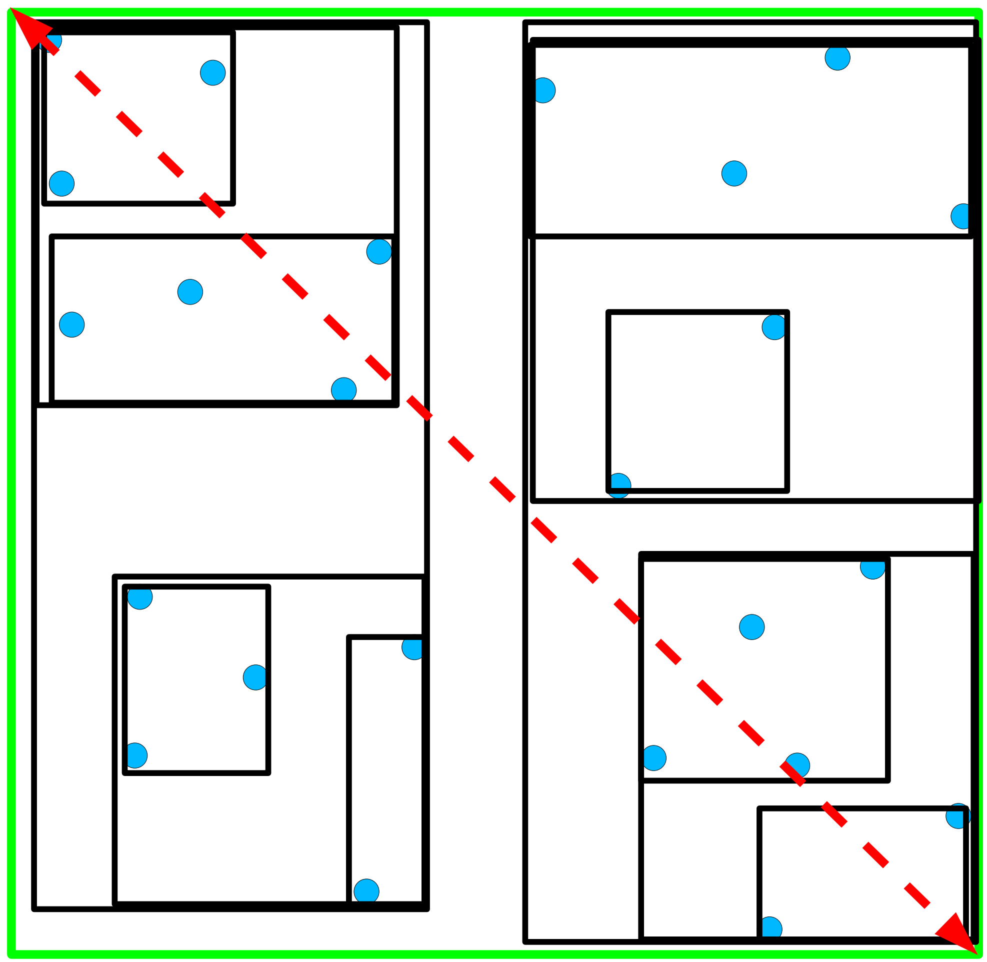
Monochromatic all-nearest-neighbors: $\operatorname*{map}_{q \in X} \operatorname*{argmin}_{r \in X - q} d(q, r)$



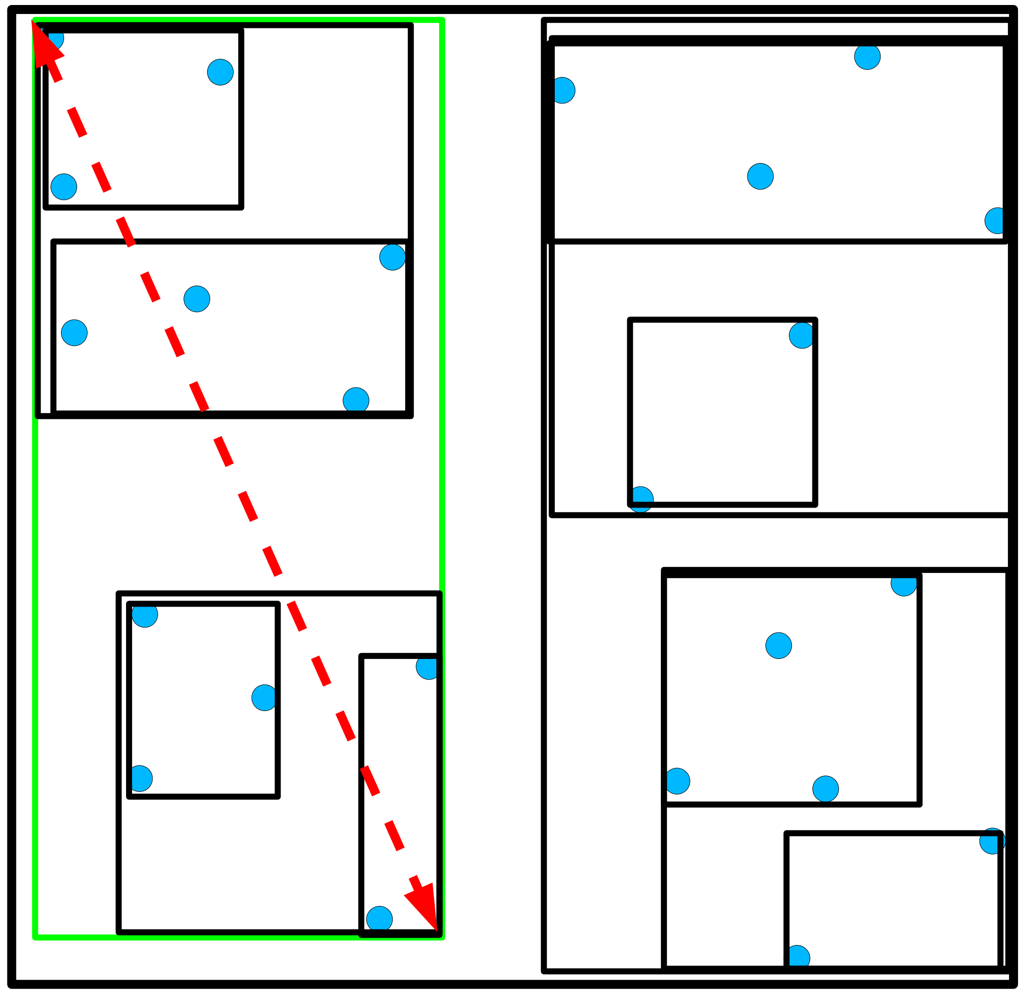
Monochromatic all-nearest-neighbors: $\underset{q \in X}{\text{map}} \underset{r \in X-q}{\text{argmin}}\, d(q,r)$



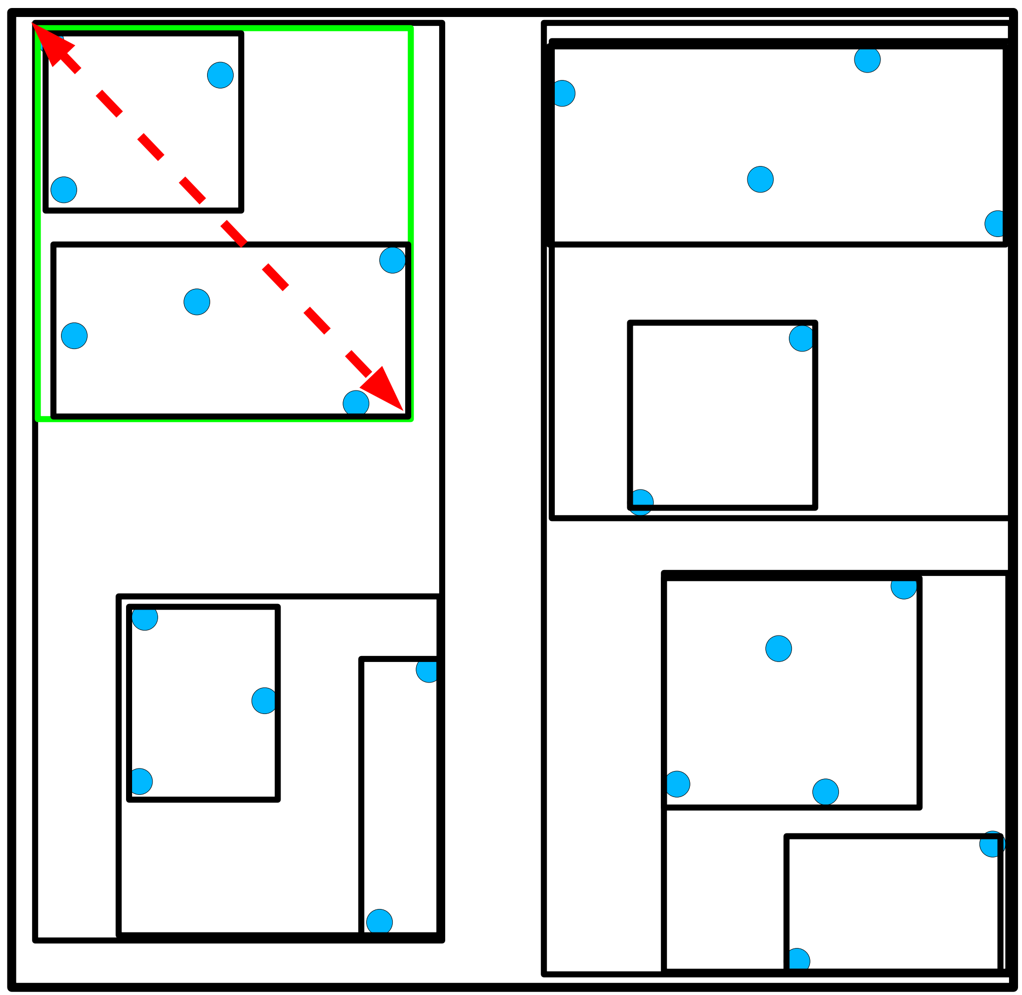
Monochromatic all-nearest-neighbors: $\operatorname*{map}_{q \in X} \operatorname*{argmin}_{r \in X - q} d(q, r)$



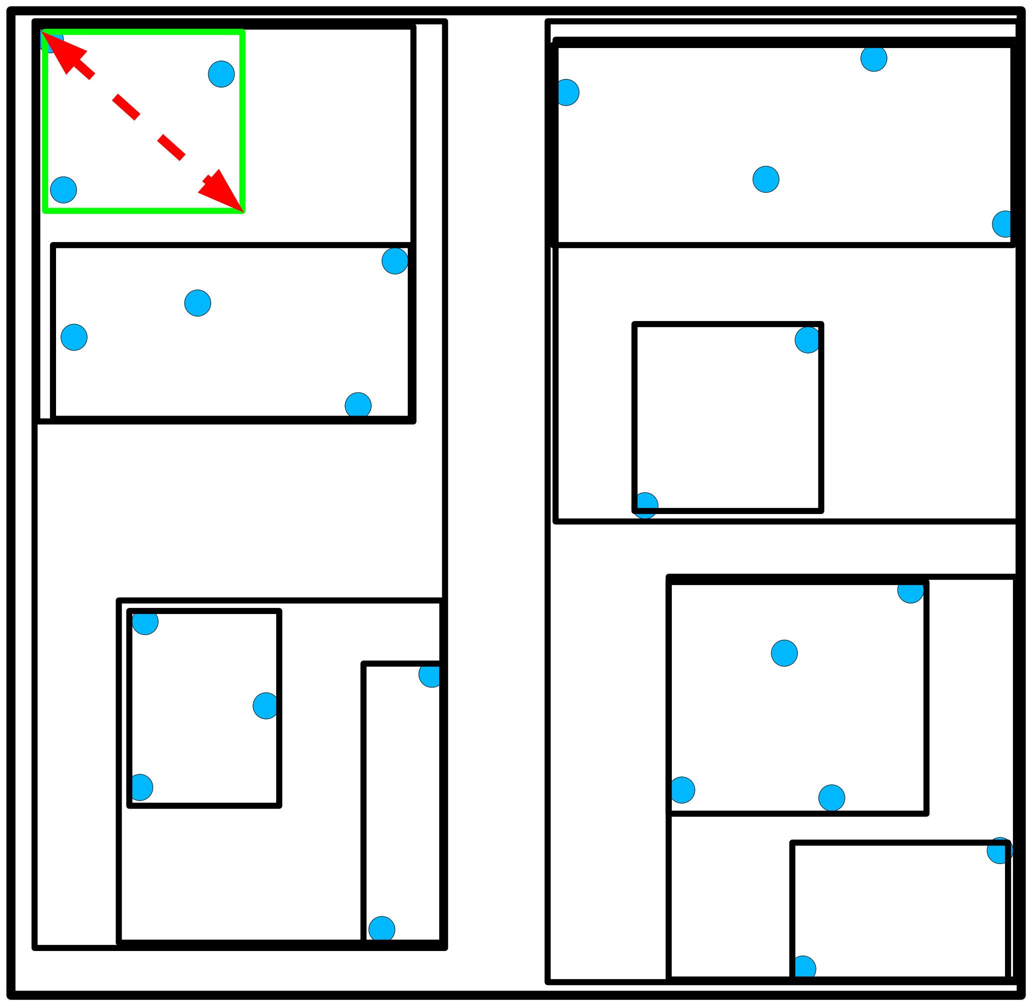
Monochromatic all-nearest-neighbors: $\underset{q \in X}{\mathrm{map}} \underset{r \in X-q}{\mathrm{argmin}}\, d(q,r)$



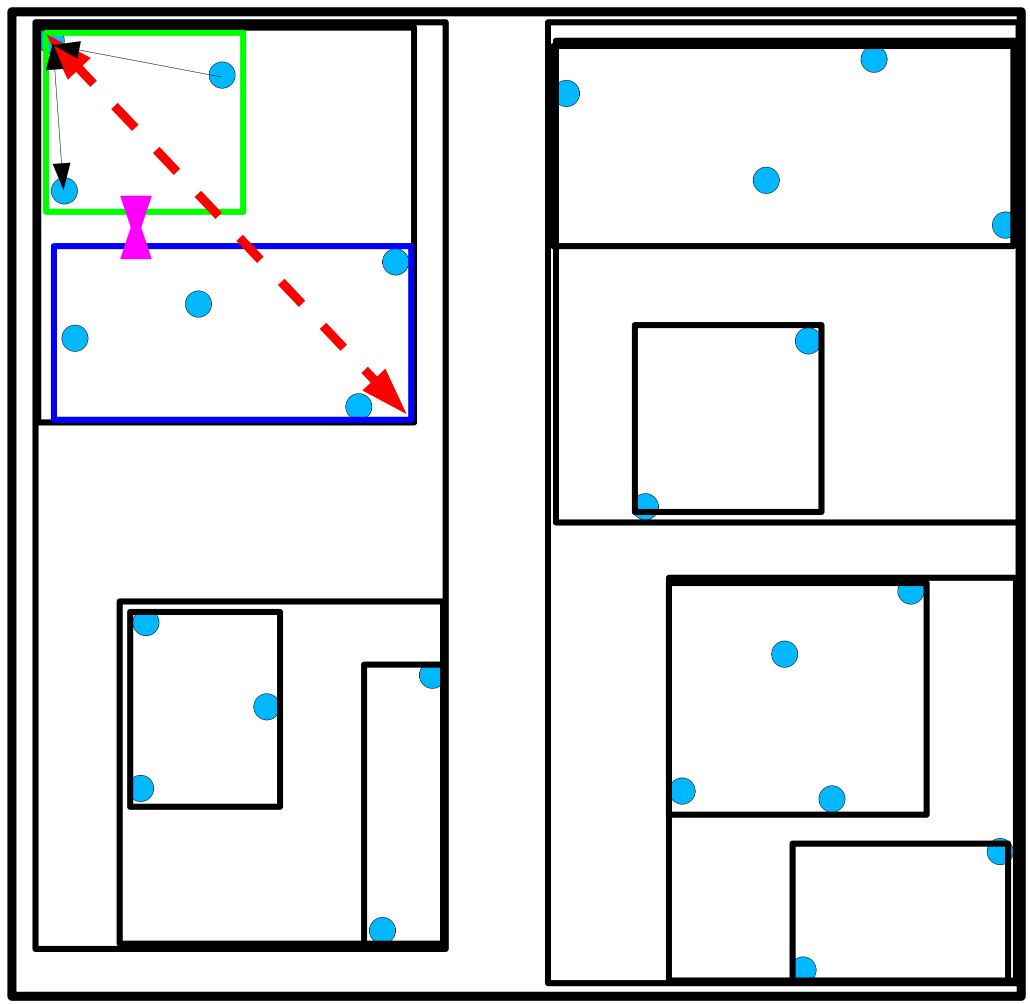
Monochromatic all-nearest-neighbors: $\operatorname*{map}_{q \in X} \operatorname*{argmin}_{r \in X - q} d(q, r)$



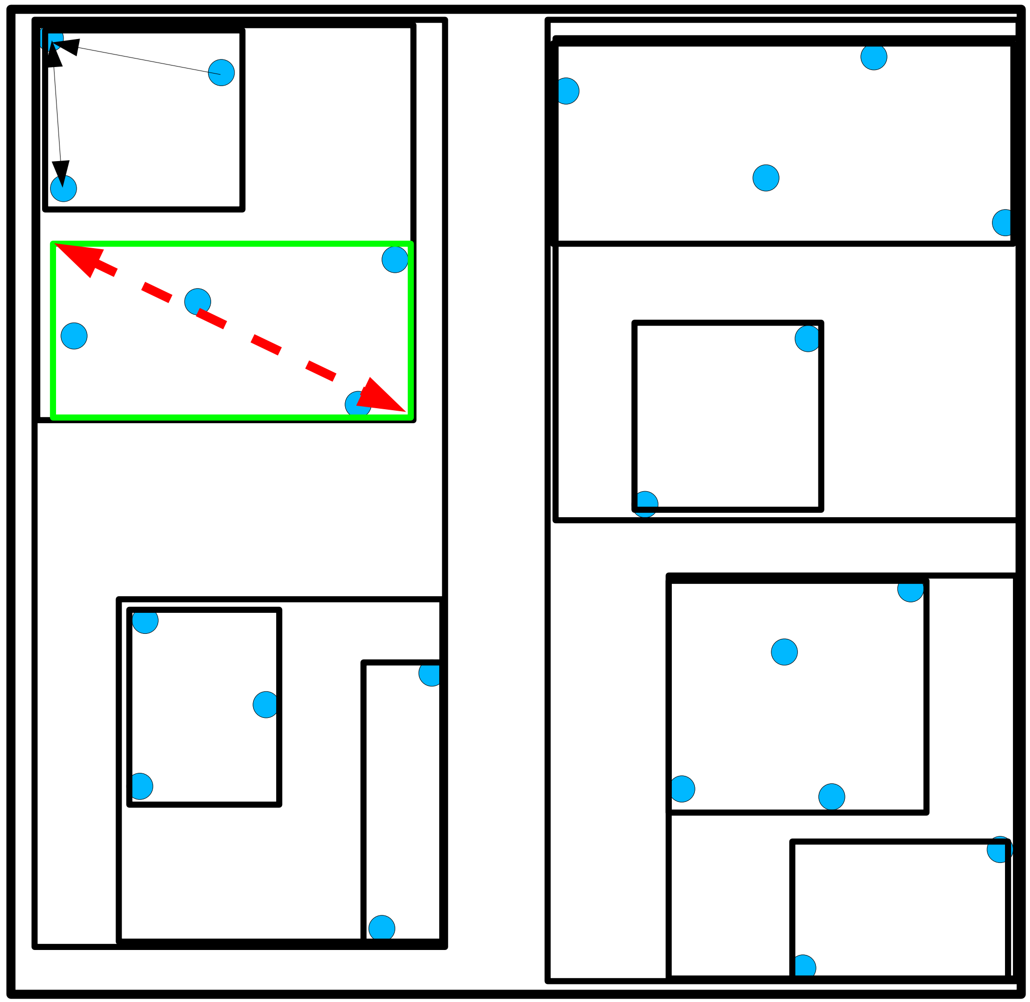
Monochromatic all-nearest-neighbors: $\operatorname*{map}_{q \in X} \operatorname*{argmin}_{r \in X - q} d(q, r)$



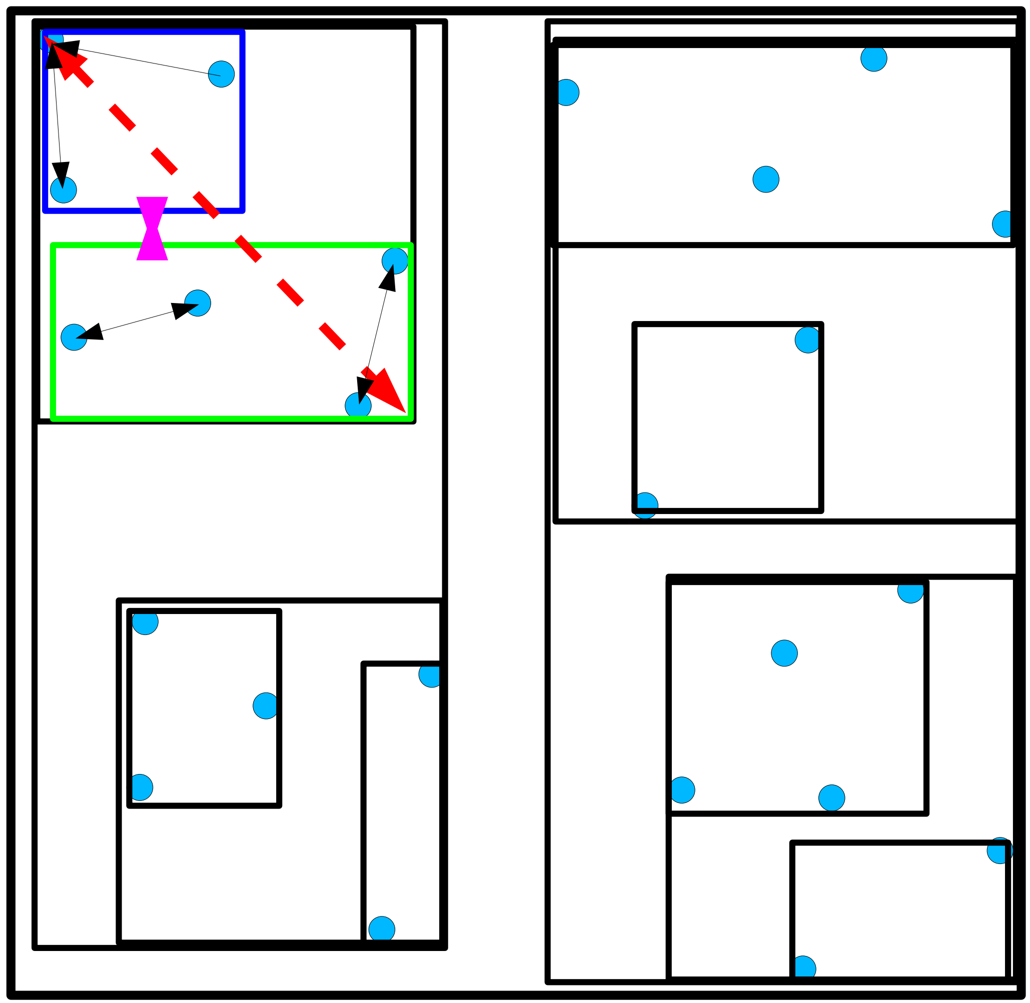
Monochromatic all-nearest-neighbors: $\operatorname*{map}_{q \in X} \operatorname*{argmin}_{r \in X - q} d(q, r)$



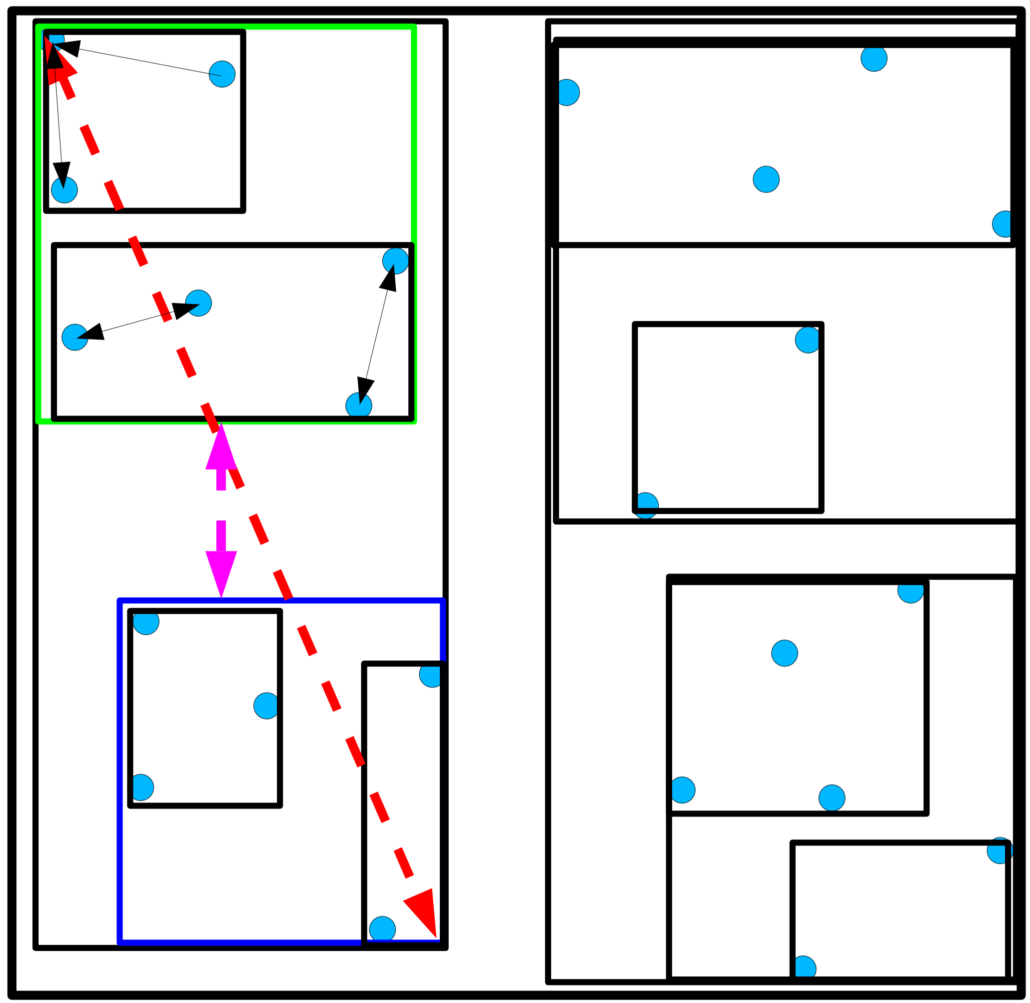
Monochromatic all-nearest-neighbors: $\operatorname*{map}_{q \in X} \operatorname*{argmin}_{r \in X - q} d(q, r)$



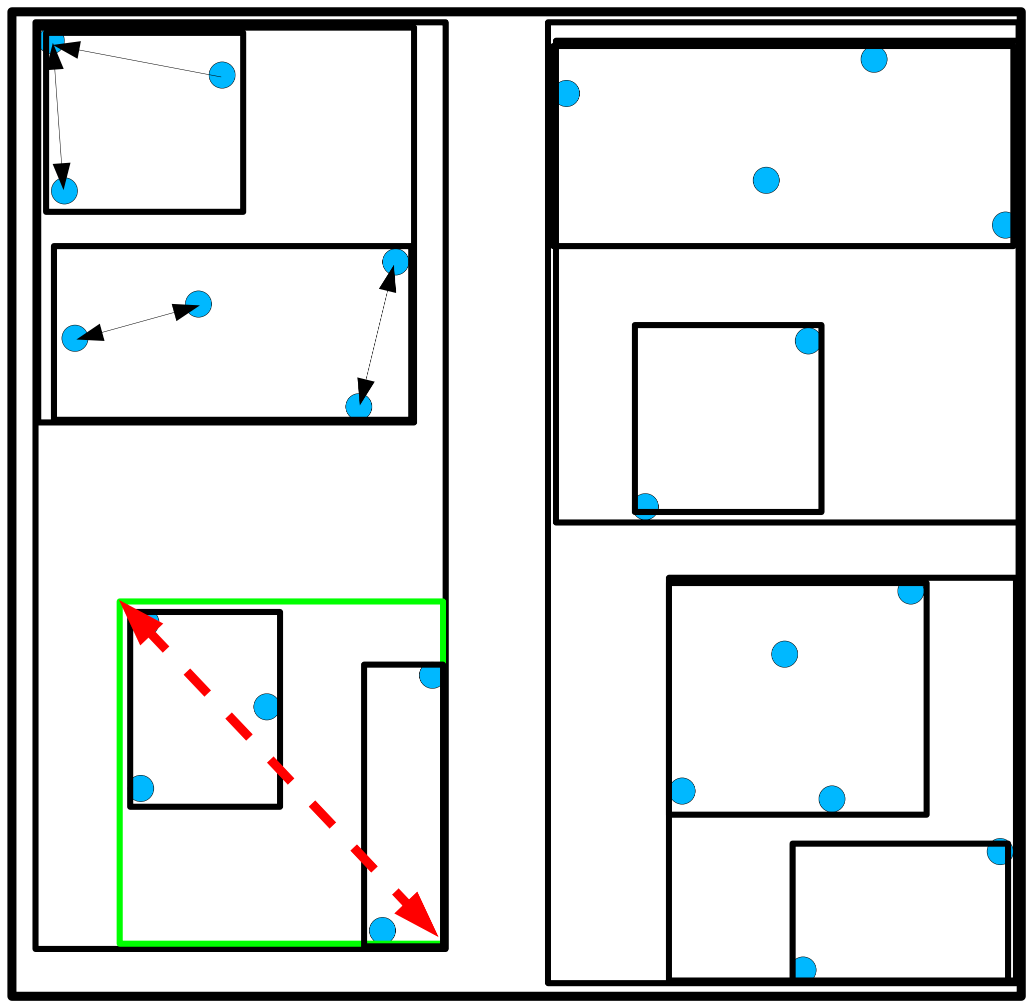
Monochromatic all-nearest-neighbors: $\operatorname{map}_{q \in X} \operatorname{argmin}_{r \in X - q} d(q, r)$



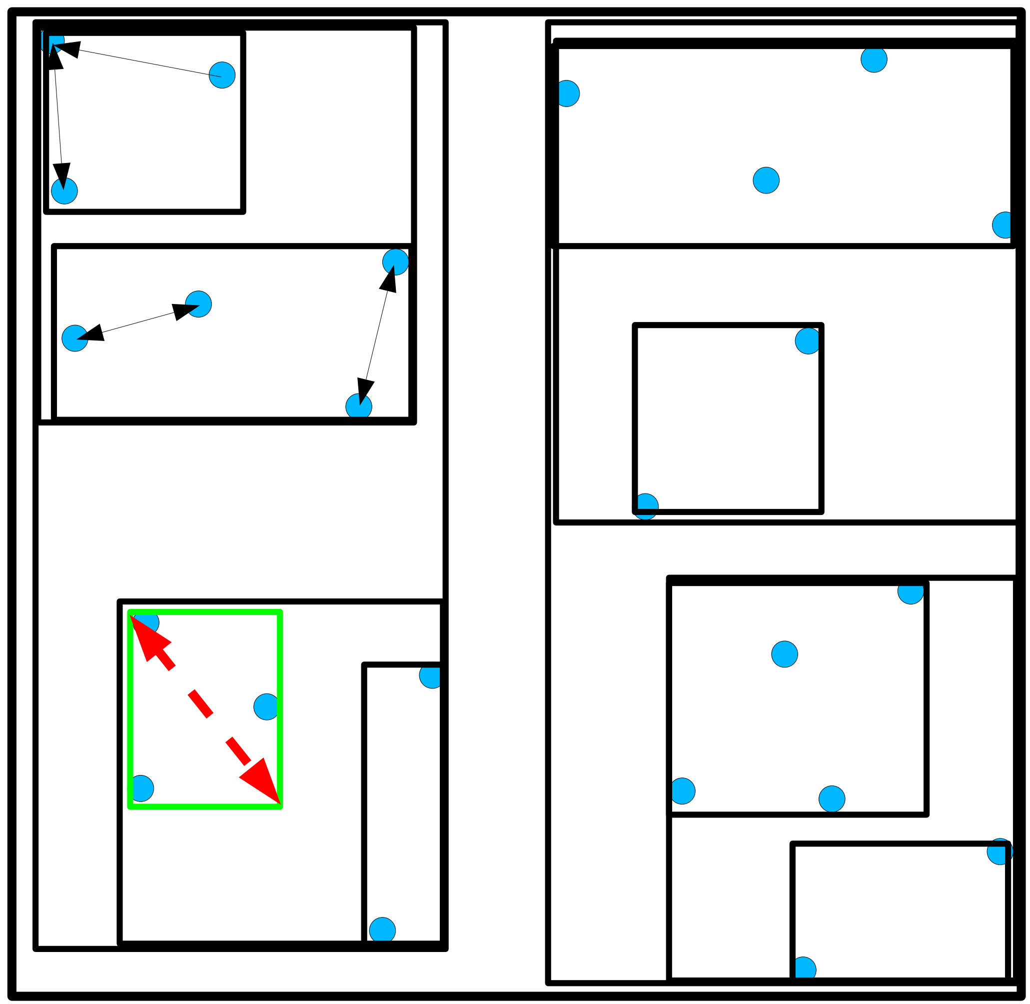
Monochromatic all-nearest-neighbors: $\operatorname*{map}_{q \in X} \operatorname*{argmin}_{r \in X - q} d(q, r)$



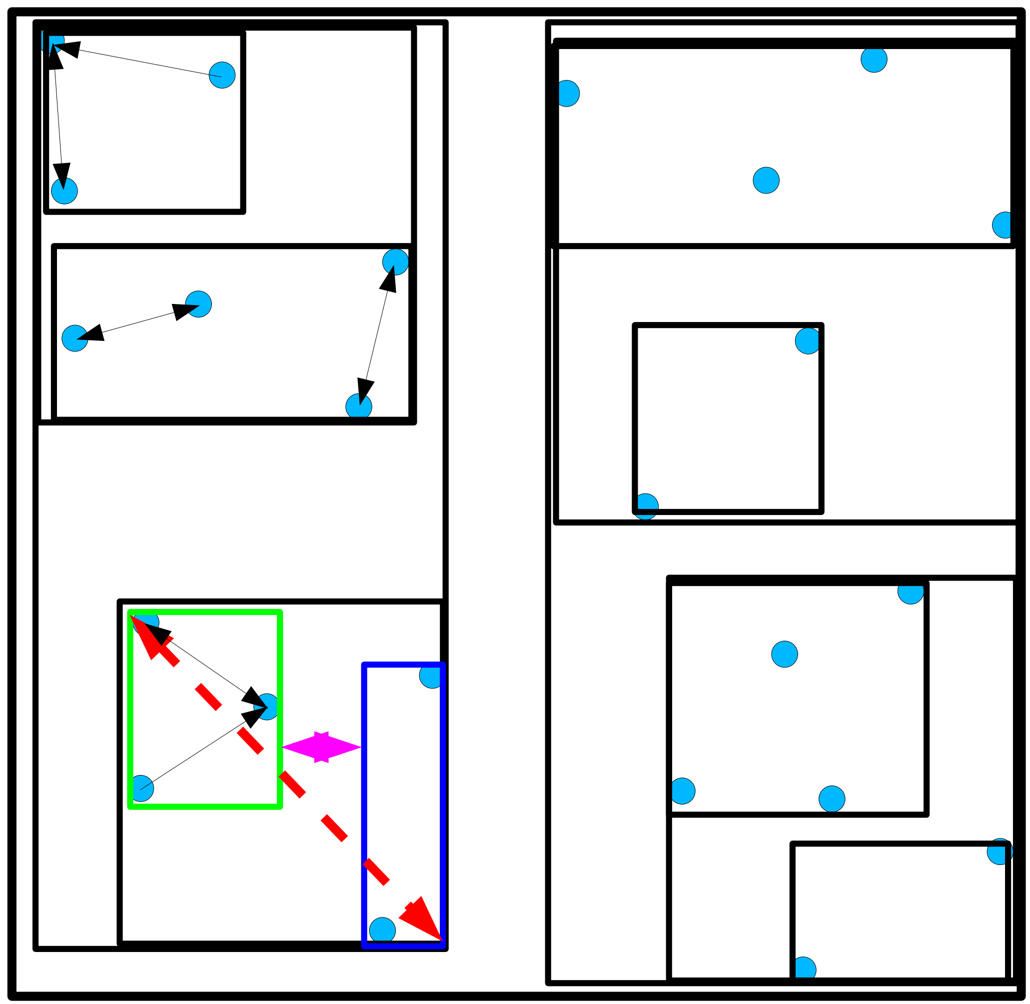
Monochromatic all-nearest-neighbors: $\operatorname*{map}_{q \in X} \operatorname*{argmin}_{r \in X - q} d(q, r)$



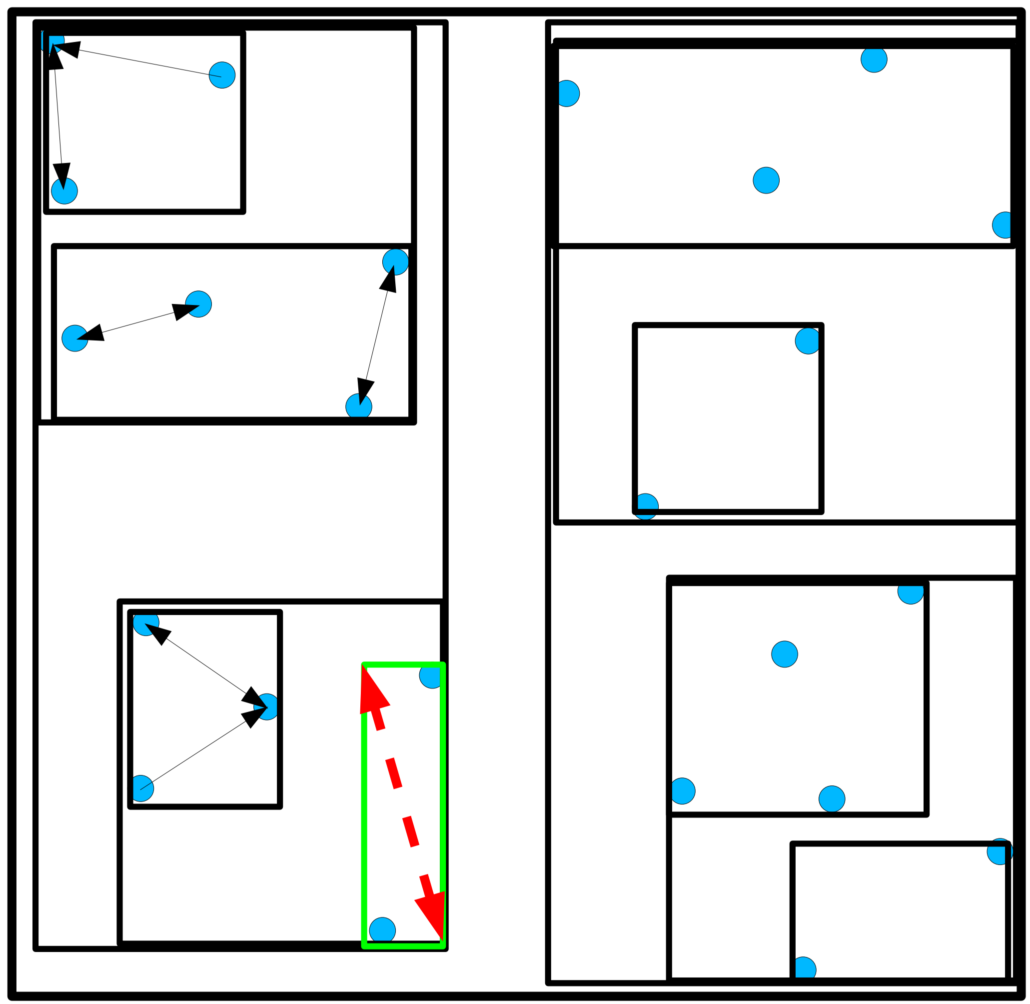
Monochromatic all-nearest-neighbors: $\operatorname{map}_{q \in X} \operatorname{argmin}_{r \in X - q} d(q, r)$



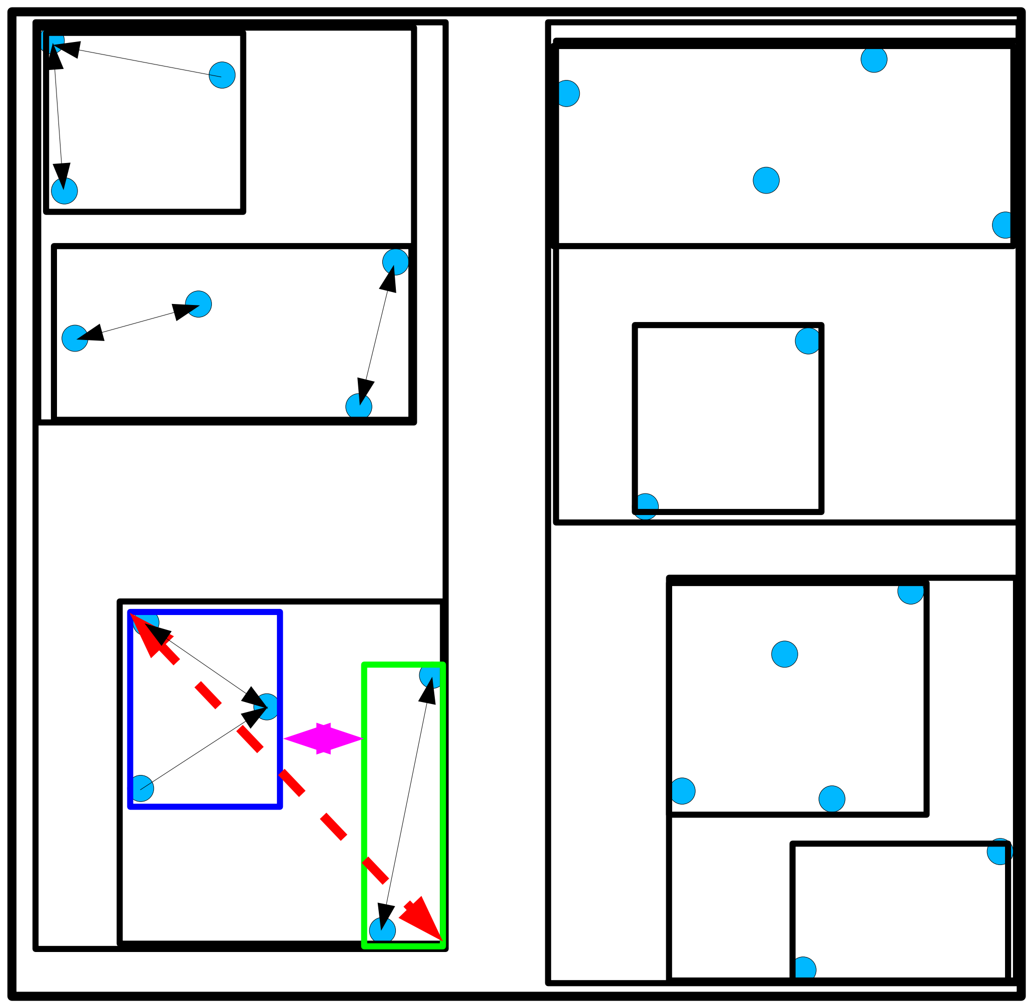
Monochromatic all-nearest-neighbors: $\operatorname{map}_{q \in X} \operatorname{argmin}_{r \in X - q} d(q, r)$



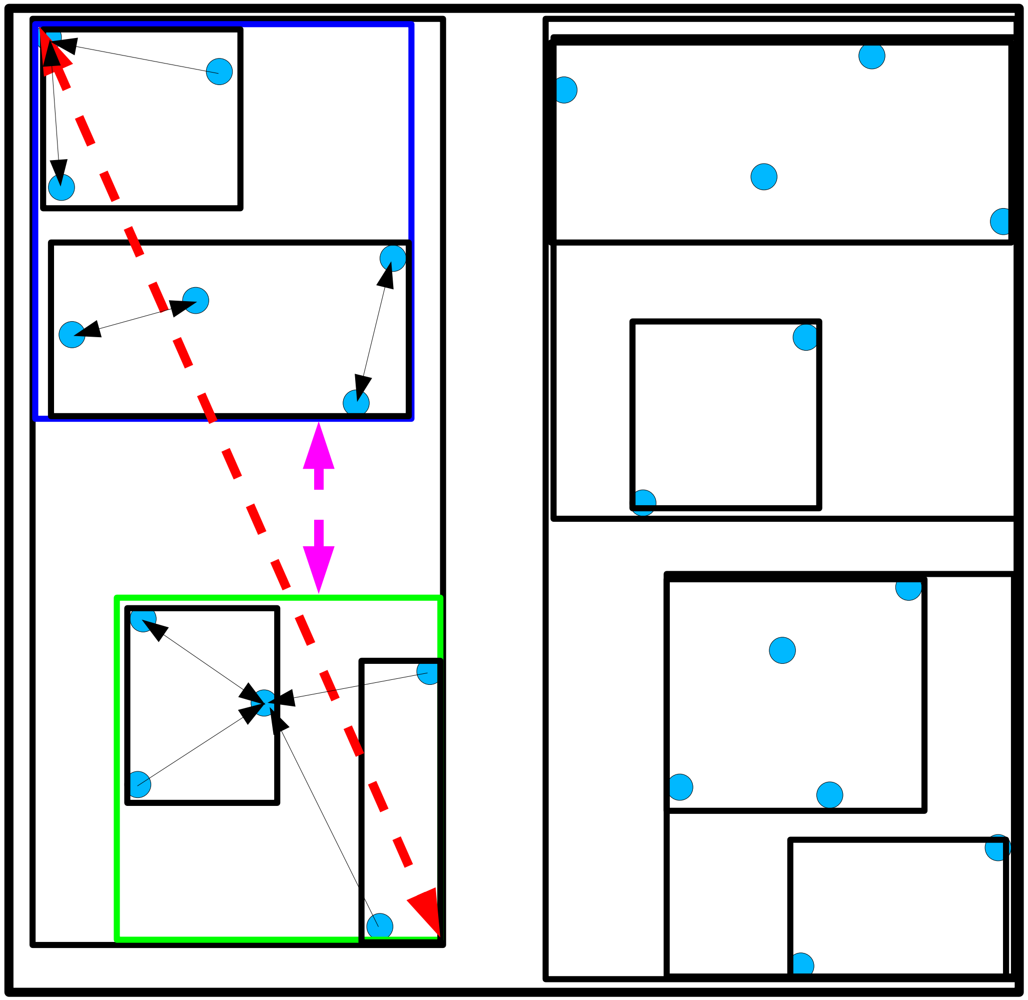
Monochromatic all-nearest-neighbors: $\operatorname*{map}_{q \in X} \operatorname*{argmin}_{r \in X - q} d(q, r)$



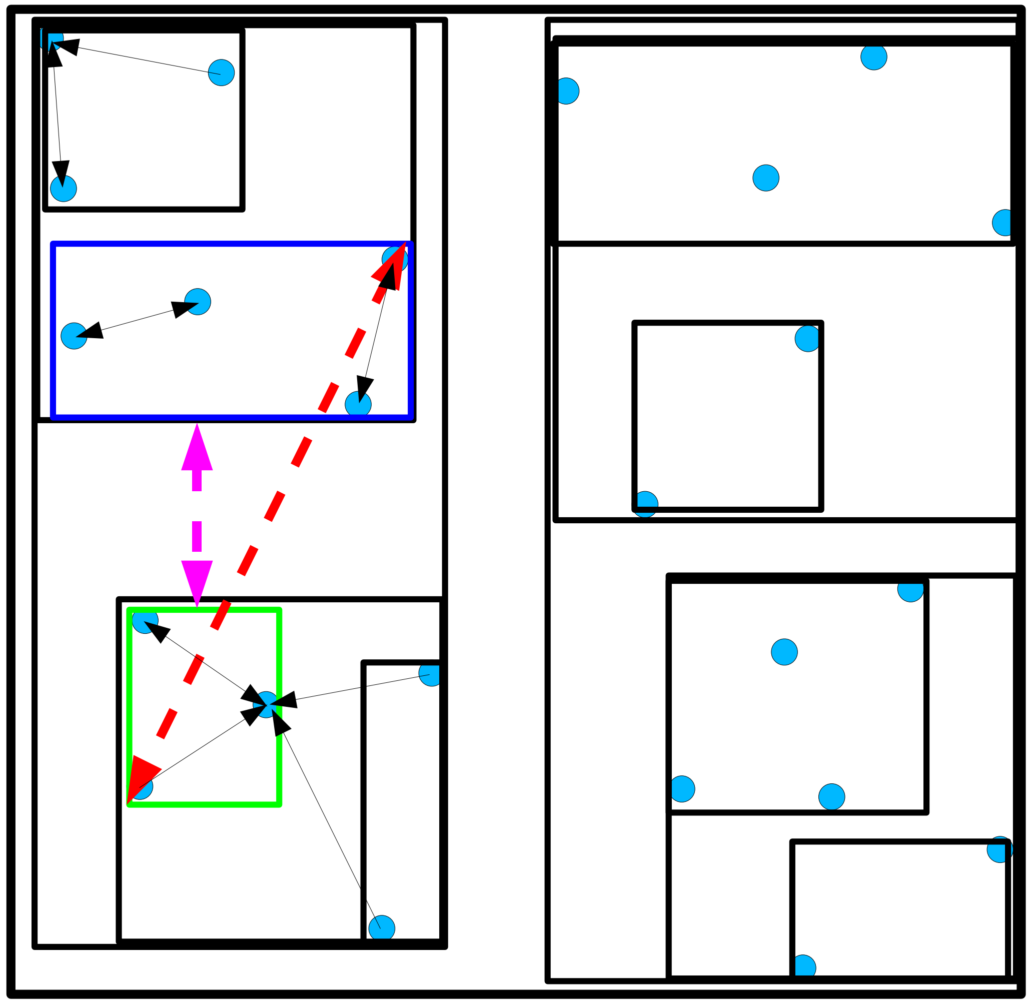
Monochromatic all-nearest-neighbors: $\operatorname*{map}_{q \in X} \operatorname*{argmin}_{r \in X - q} d(q, r)$



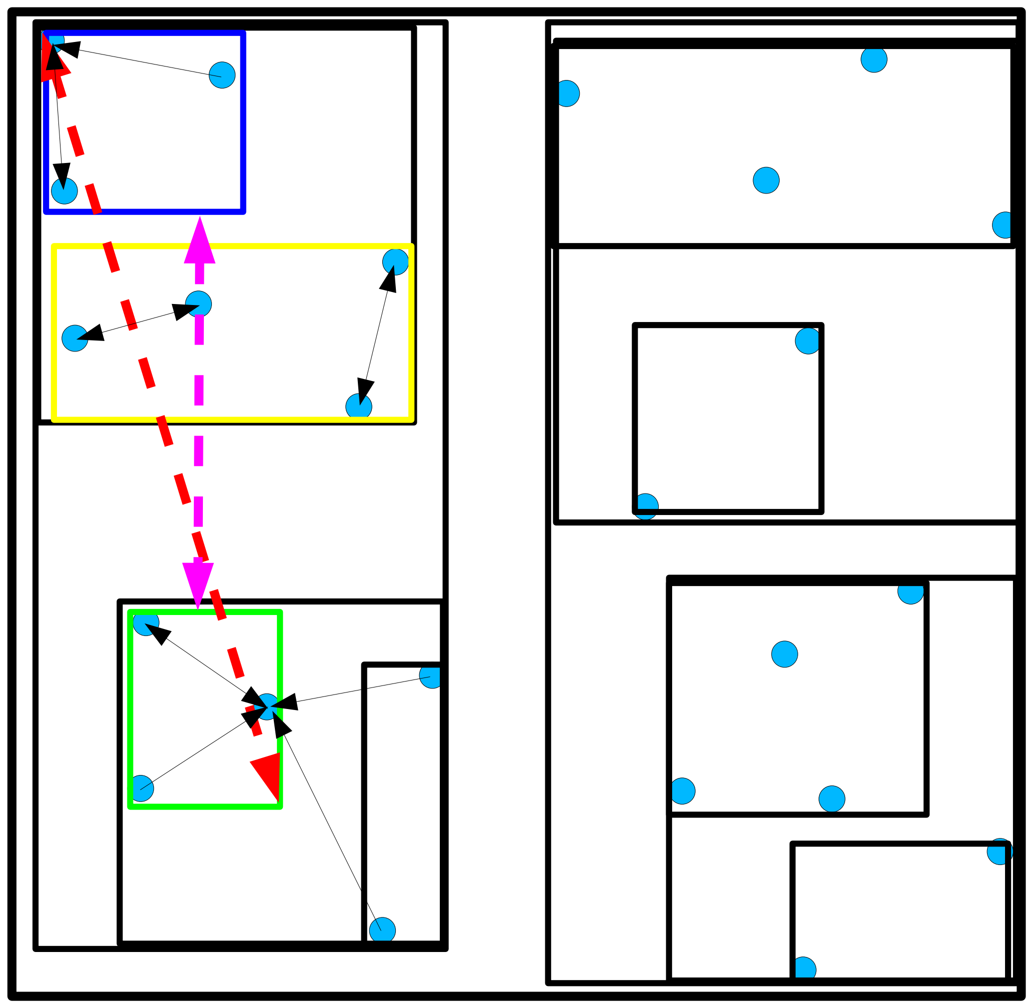
Monochromatic all-nearest-neighbors: $\operatorname{map}_{q \in X} \operatorname{argmin}_{r \in X - q} d(q, r)$



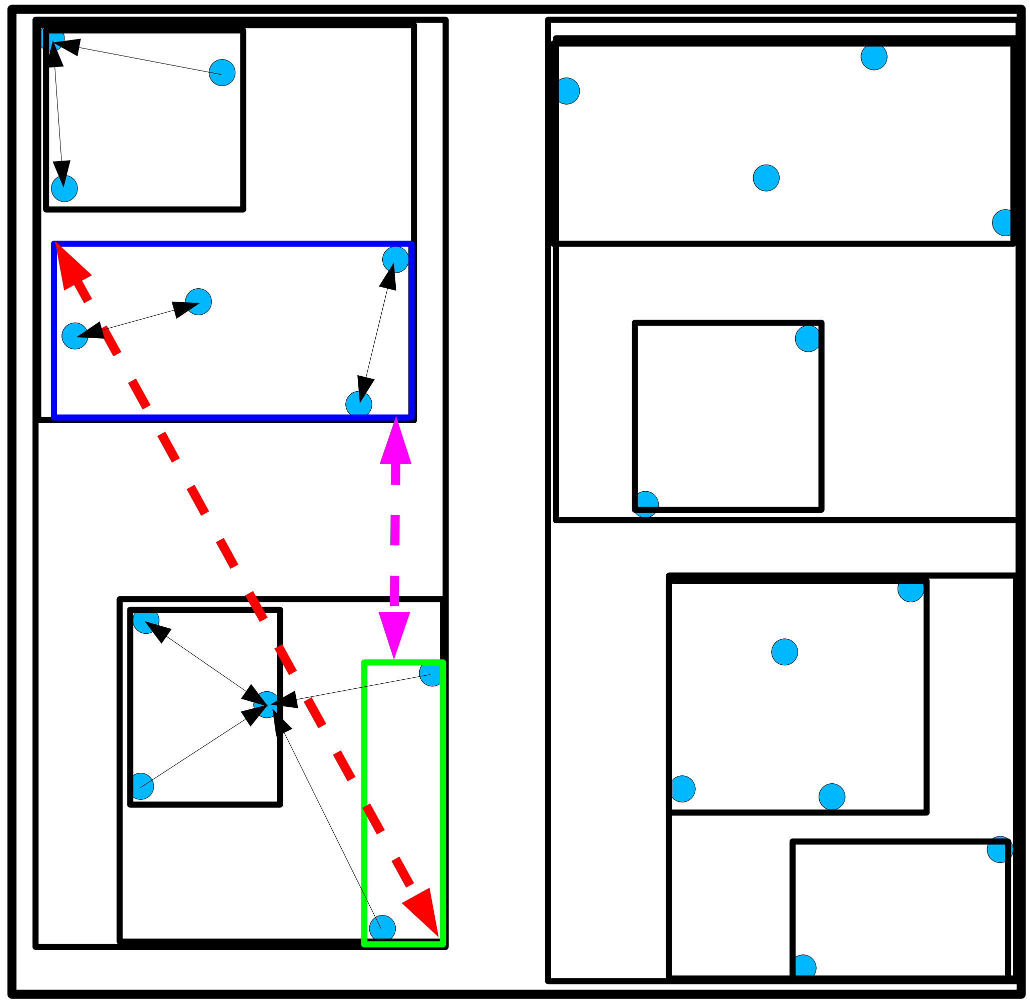
Monochromatic all-nearest-neighbors: $\operatorname*{map}_{q \in X} \operatorname*{argmin}_{r \in X - q} d(q, r)$



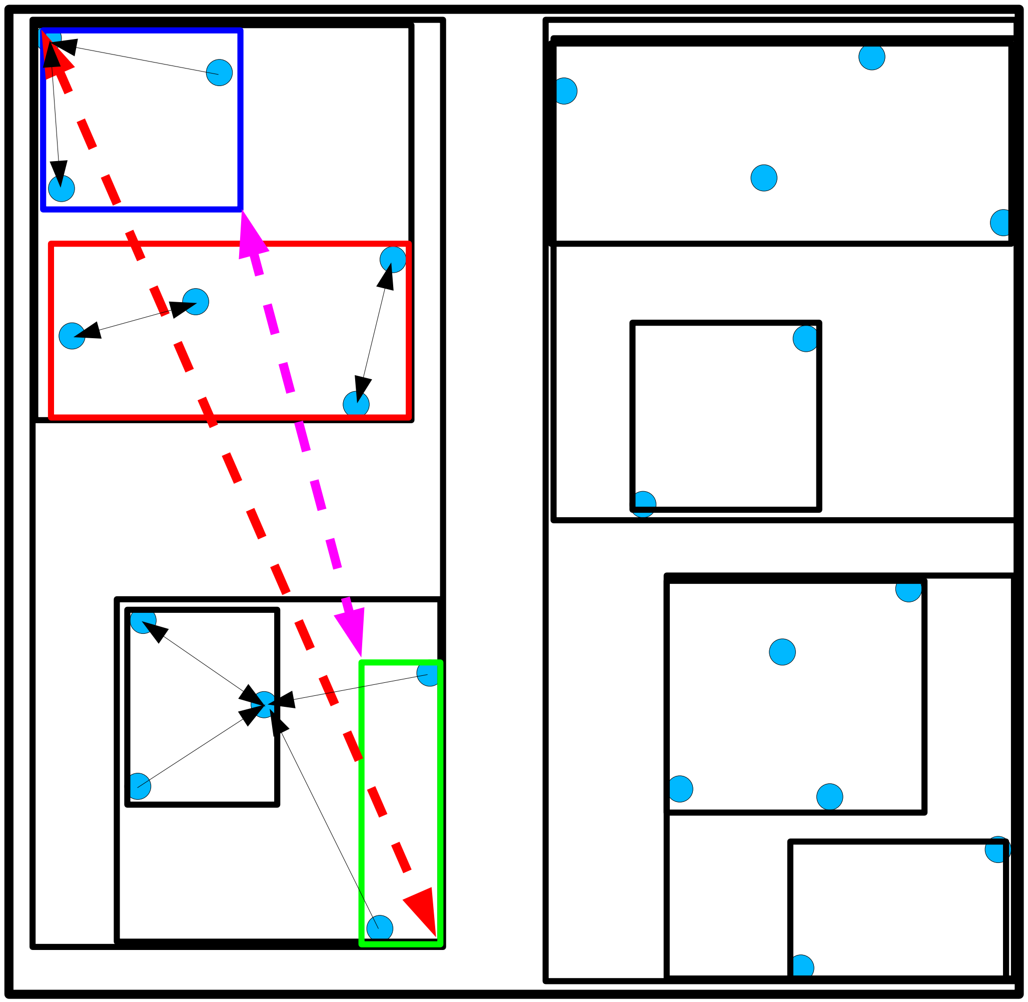
Monochromatic all-nearest-neighbors: $\underset{q \in X}{\operatorname{map}} \underset{r \in X - q}{\operatorname{argmin}} d(q, r)$



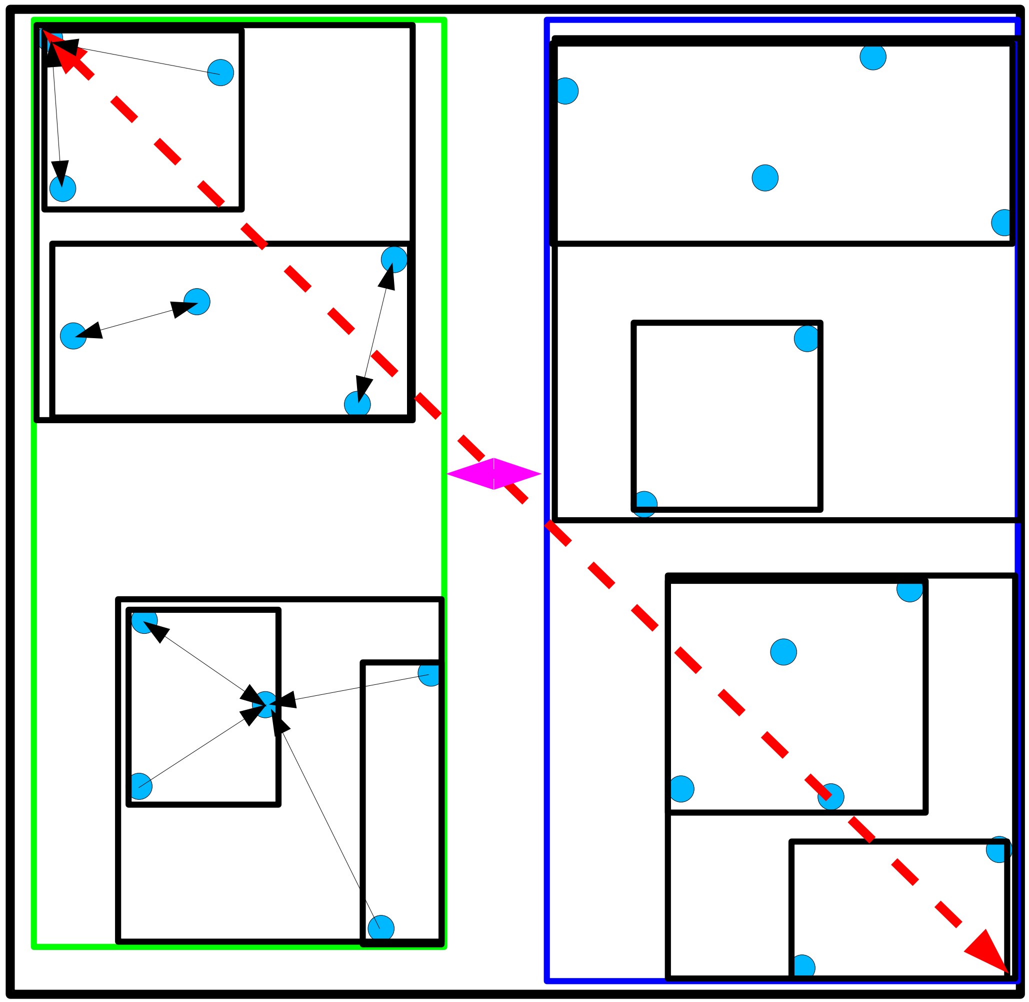
Monochromatic all-nearest-neighbors: $\operatorname*{map}_{q \in X} \operatorname*{argmin}_{r \in X - q} d(q, r)$



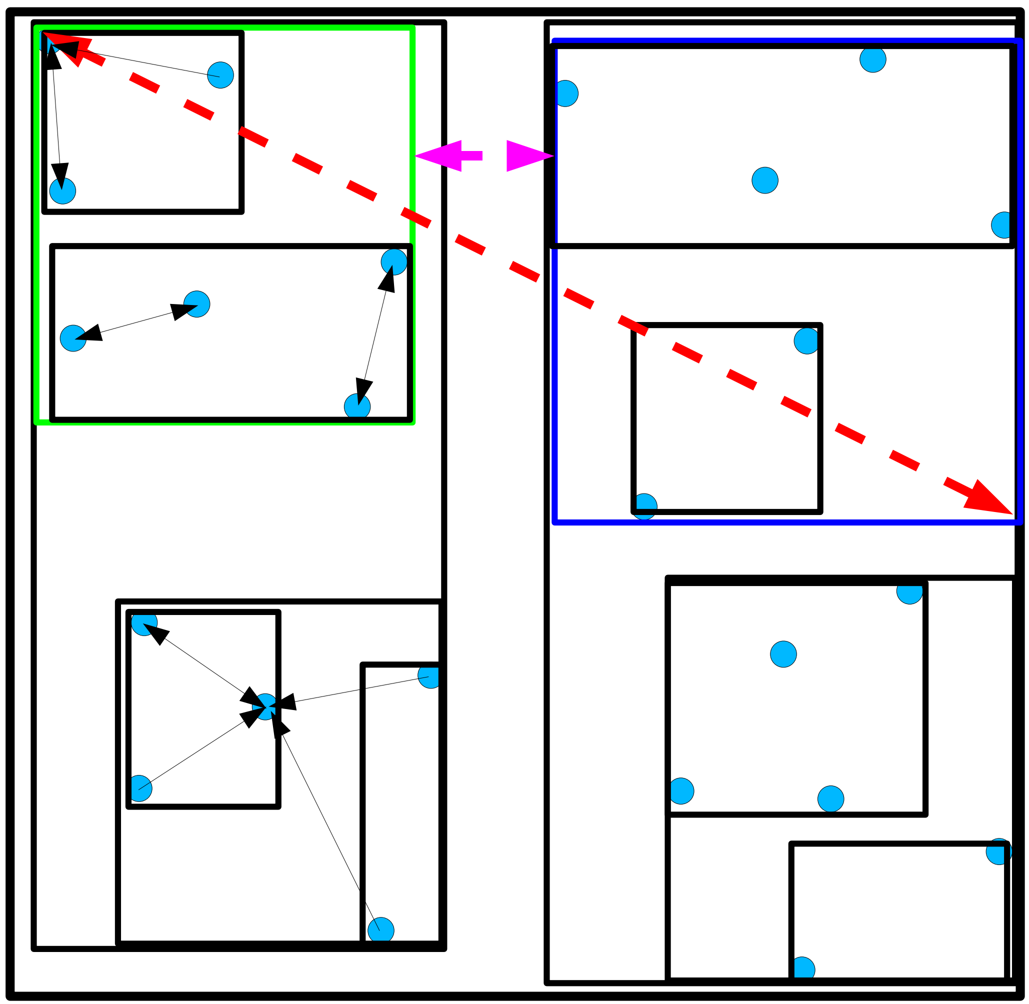
Monochromatic all-nearest-neighbors: $\operatorname*{map}_{q \in X} \operatorname*{argmin}_{r \in X - q} d(q, r)$



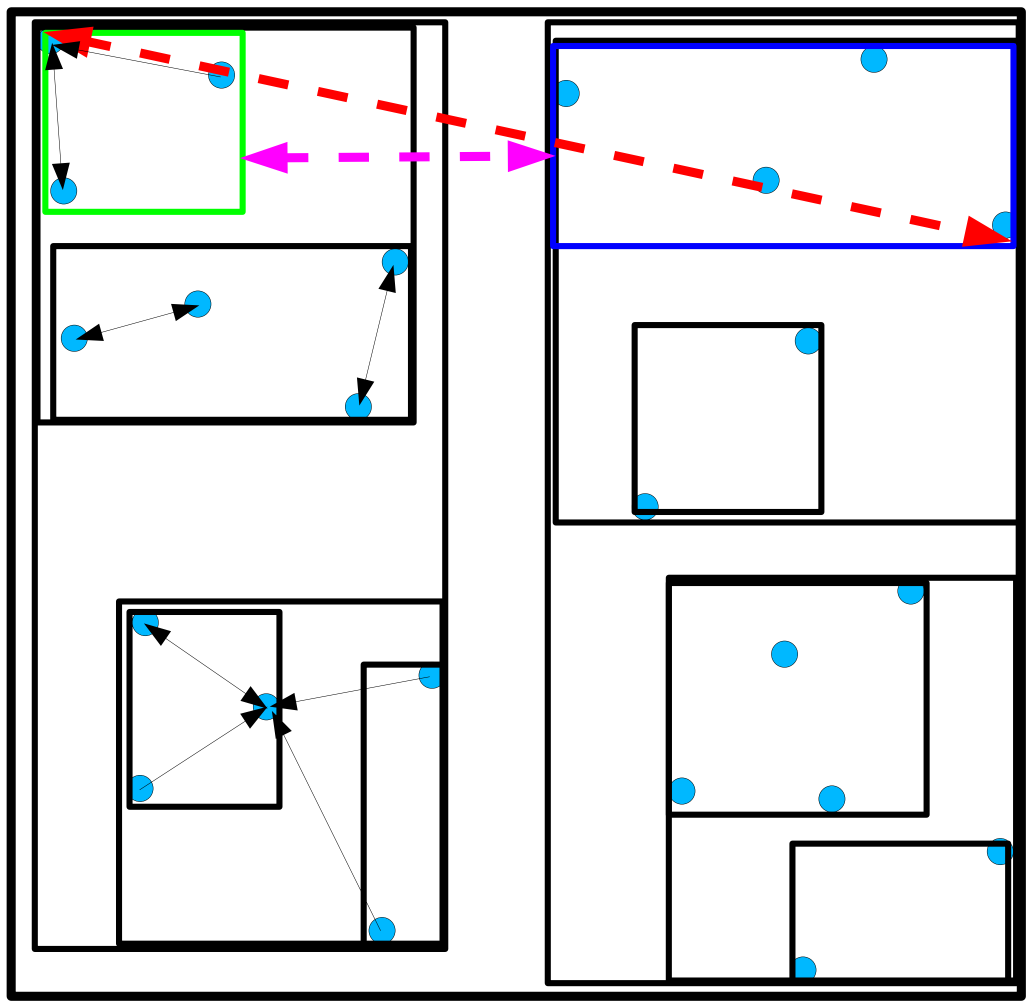
Monochromatic all-nearest-neighbors: $\operatorname*{map}_{q\in X} \operatorname*{argmin}_{r\in X-q} d(q,r)$



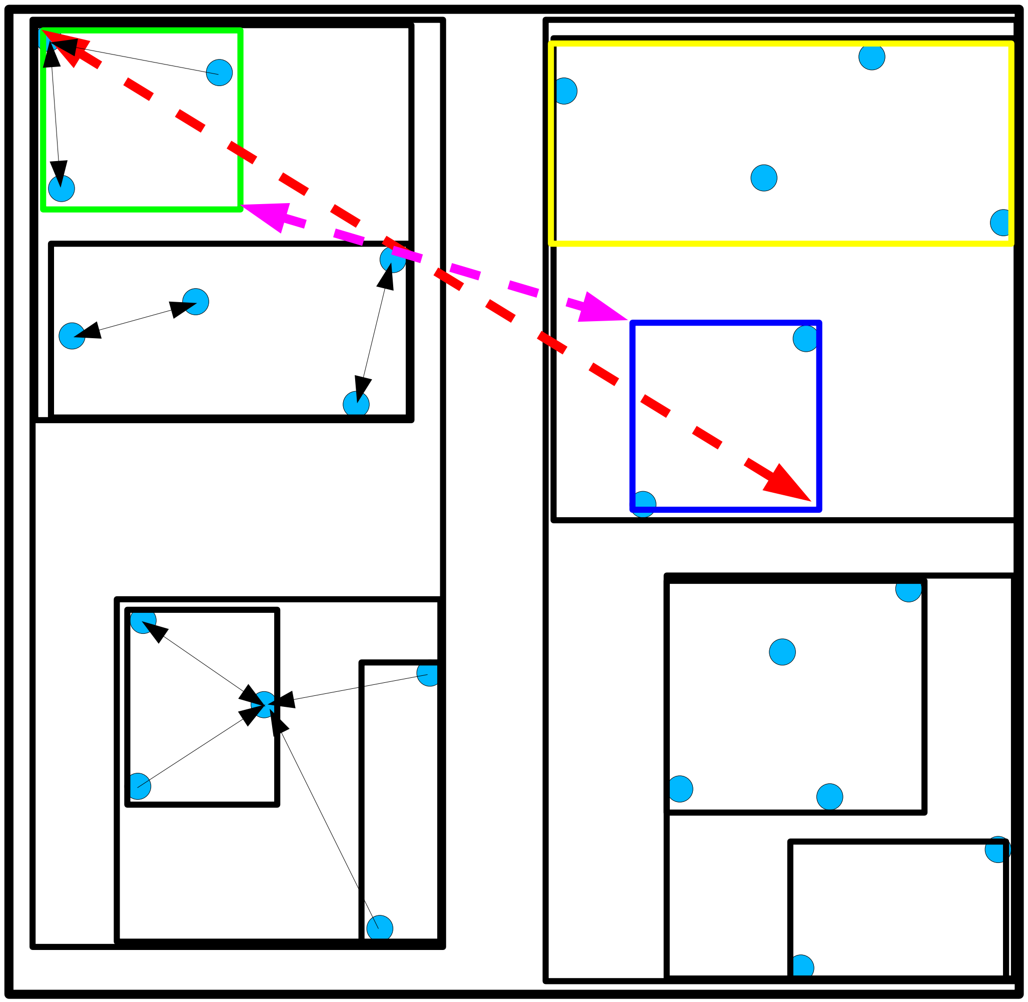
Monochromatic all-nearest-neighbors: $\underset{q \in X}{\mathrm{map}} \underset{r \in X - q}{\mathrm{argmin}} \, d(q, r)$



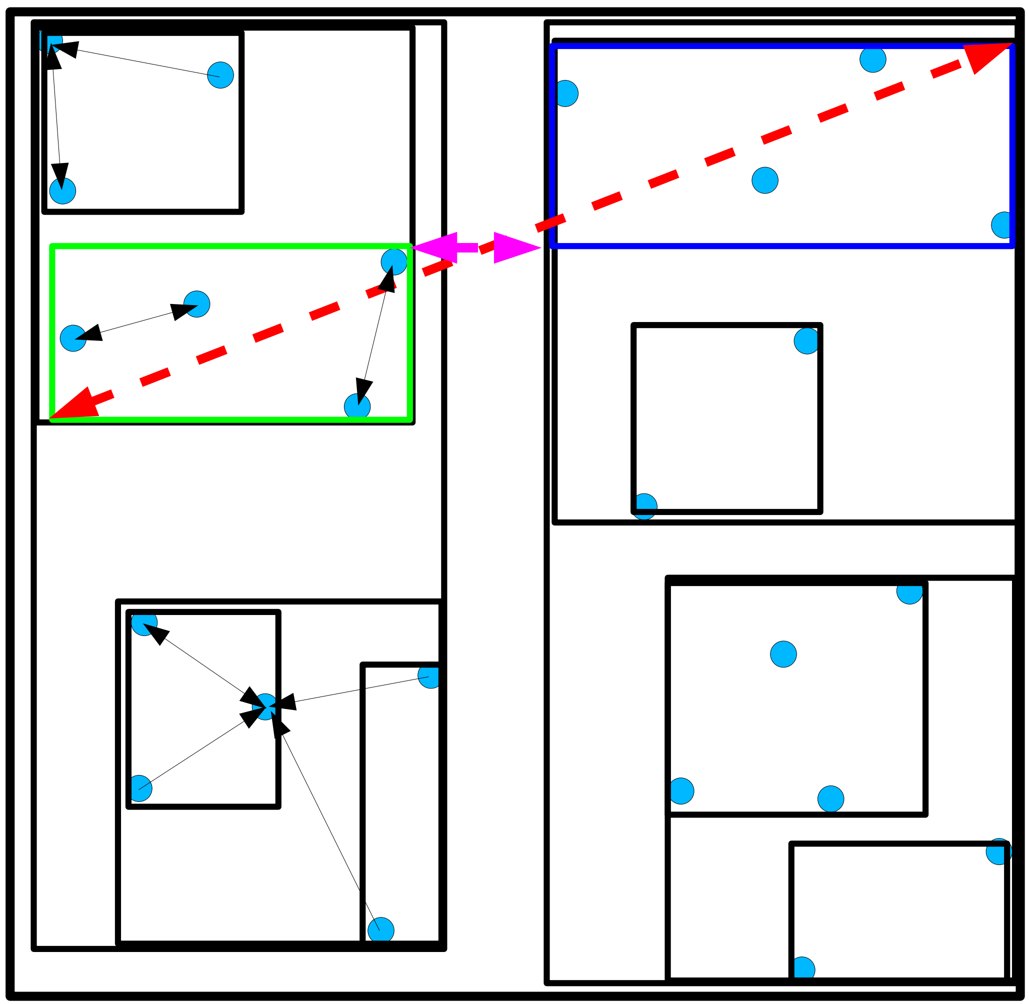
Monochromatic all-nearest-neighbors: $\operatorname*{map}_{q \in X} \operatorname*{argmin}_{r \in X - q} d(q, r)$



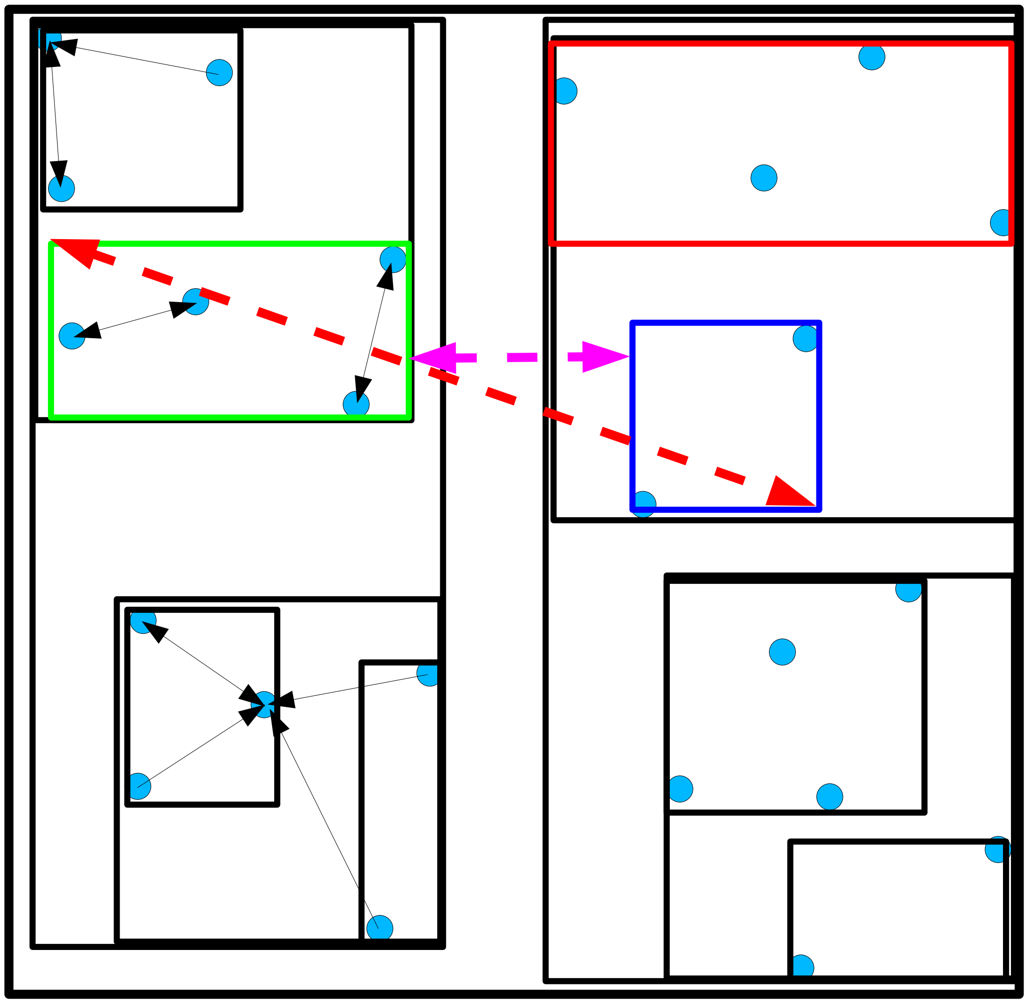
Monochromatic all-nearest-neighbors: $\underset{q \in X}{\mathrm{map}} \underset{r \in X - q}{\mathrm{argmin}} \, d(q, r)$



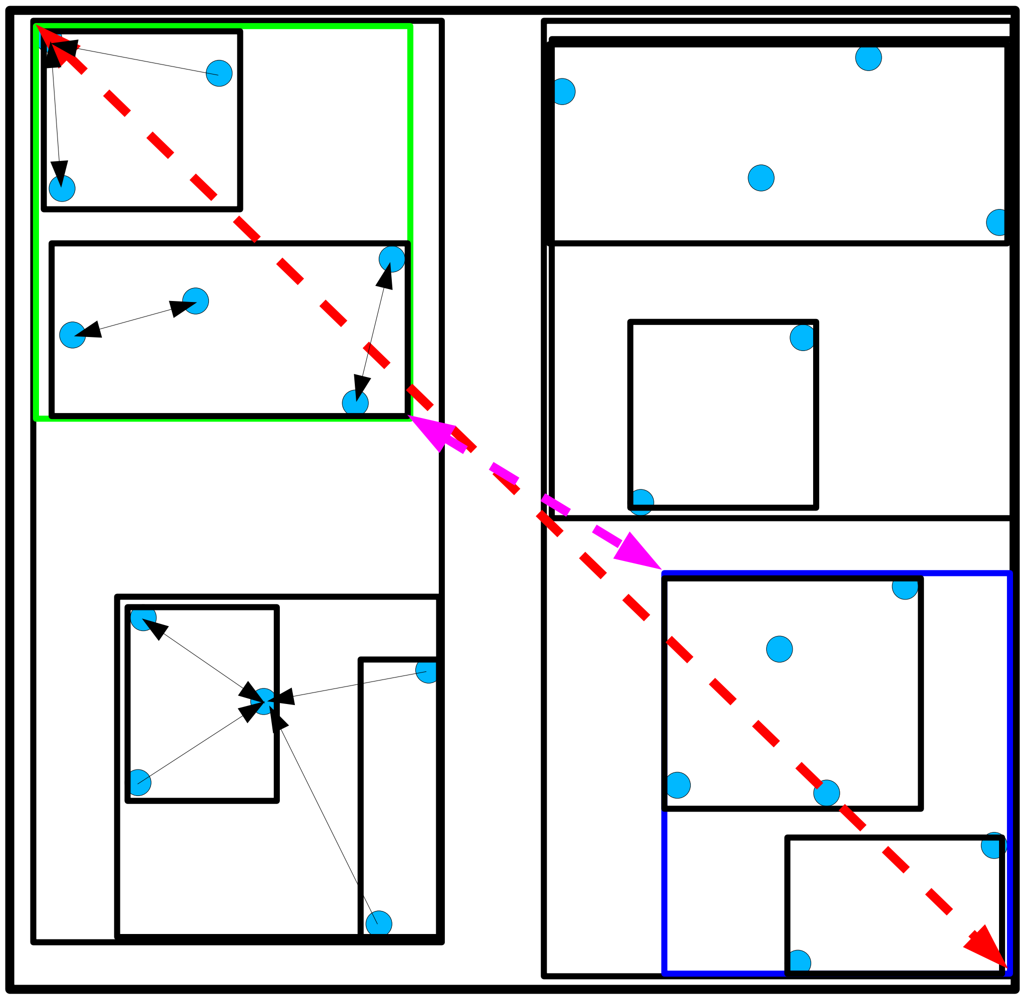
Monochromatic all-nearest-neighbors: $\underset{q \in X}{\mathrm{map}} \underset{r \in X - q}{\mathrm{argmin}} \, d(q, r)$



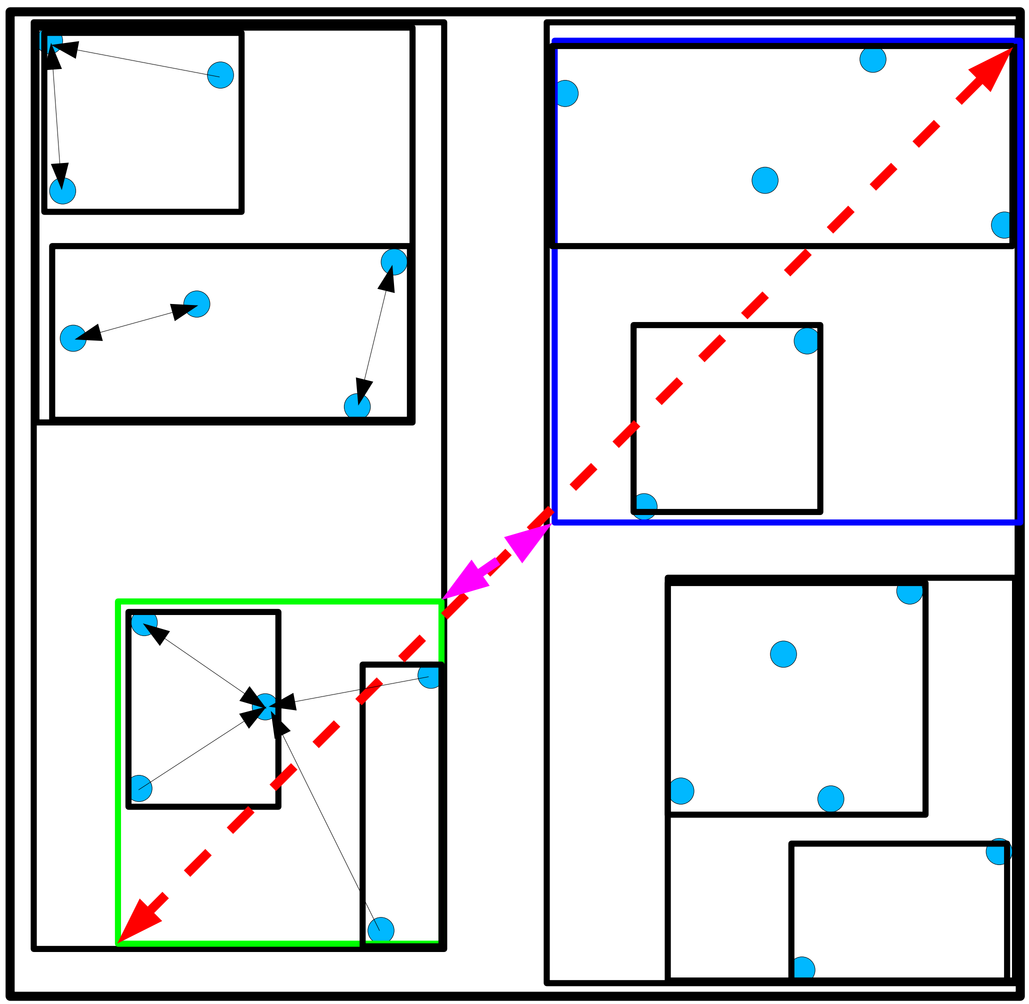
Monochromatic all-nearest-neighbors: $\operatorname*{map}_{q \in X} \operatorname*{argmin}_{r \in X - q} d(q, r)$



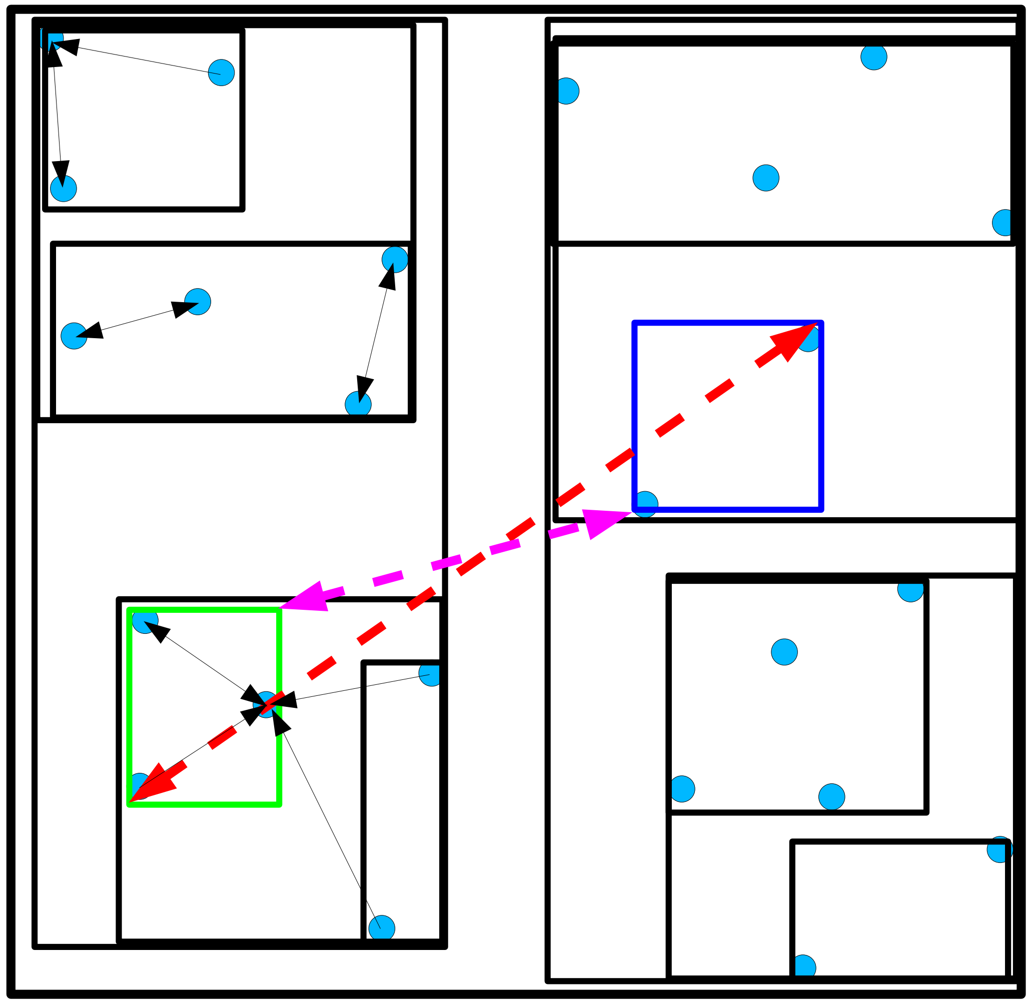
Monochromatic all-nearest-neighbors: $\operatorname*{map}_{q \in X} \operatorname*{argmin}_{r \in X - q} d(q, r)$



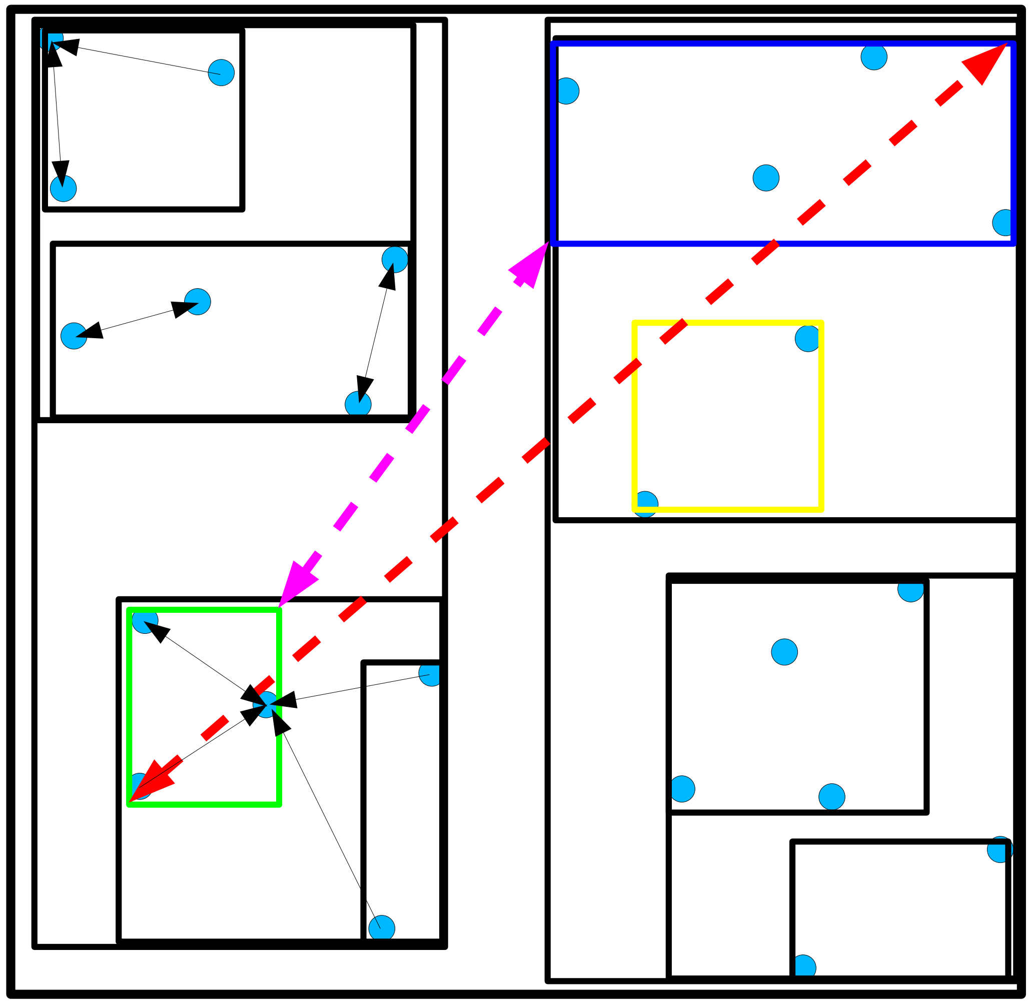
Monochromatic all-nearest-neighbors: $\operatorname*{map}_{q \in X} \operatorname*{argmin}_{r \in X - q} d(q, r)$



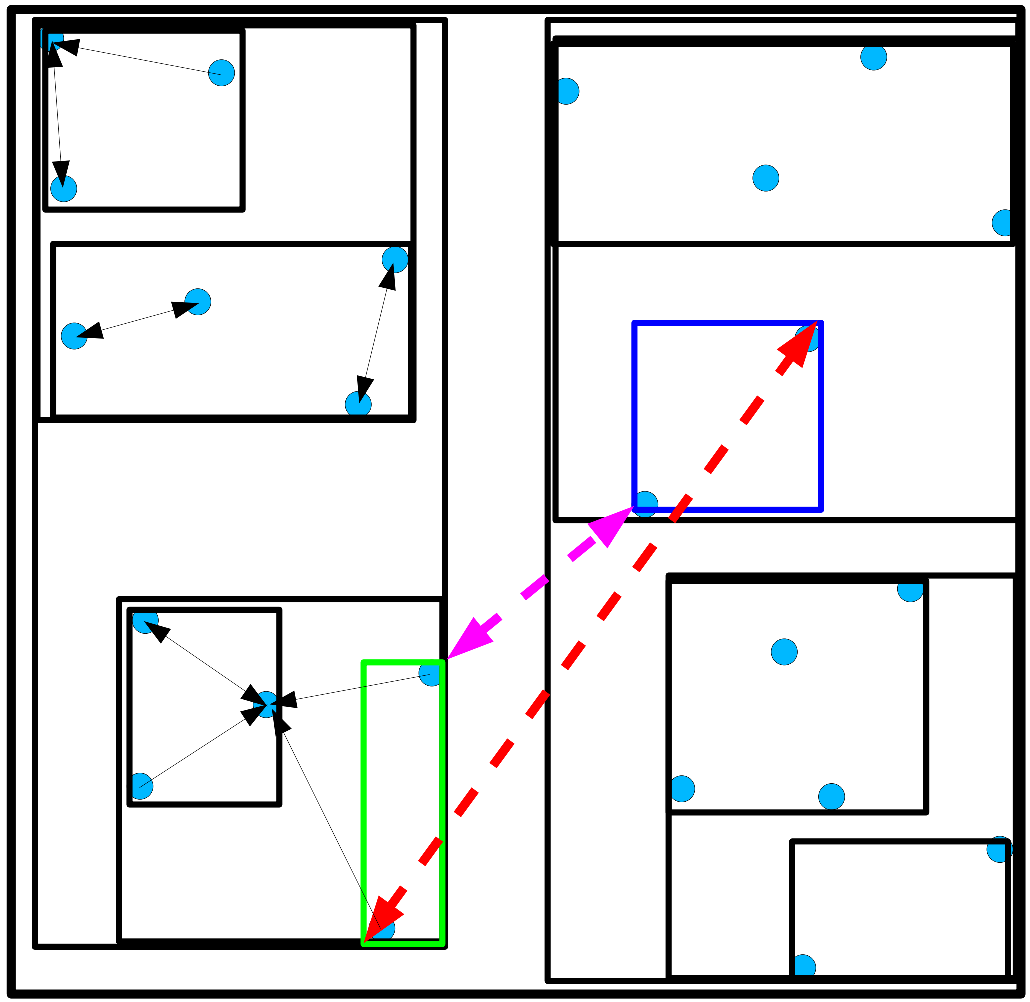
Monochromatic all-nearest-neighbors: $\underset{q \in X}{\operatorname{map}} \underset{r \in X - q}{\operatorname{argmin}} d(q, r)$



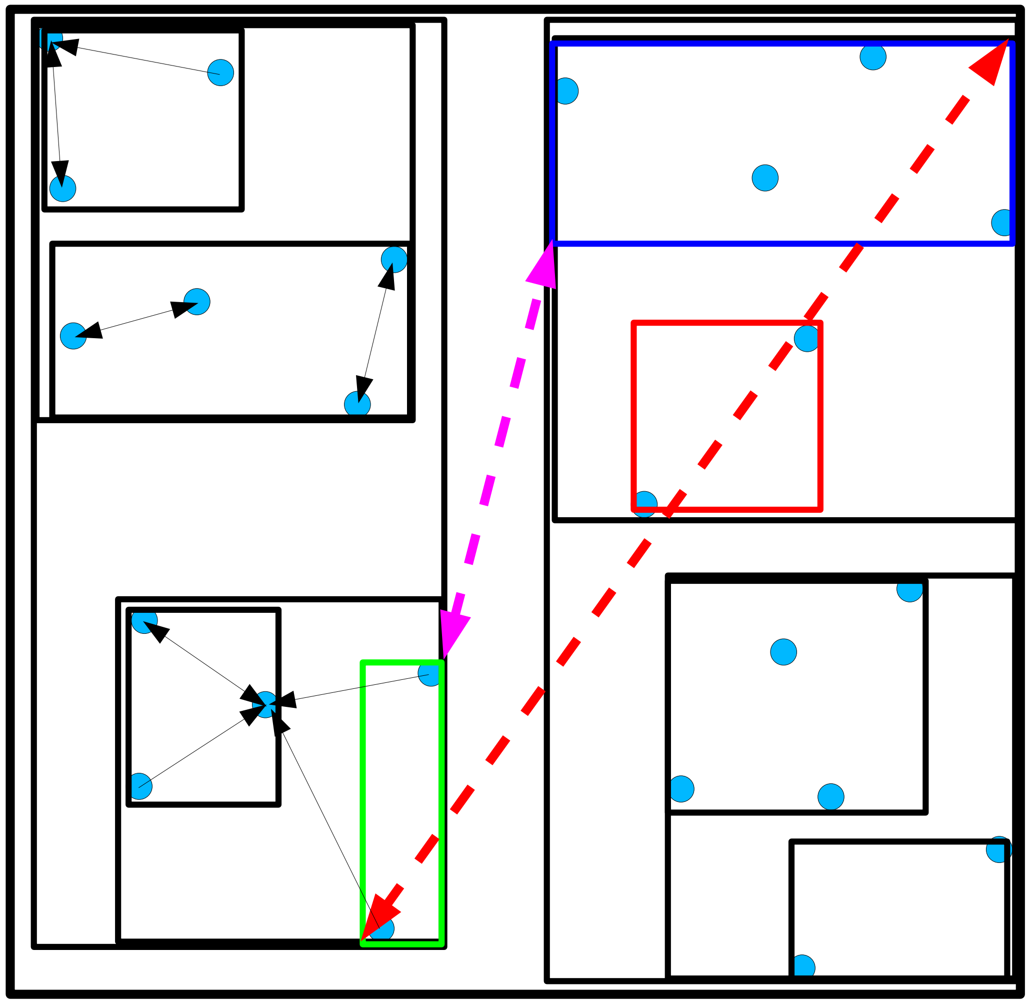
Monochromatic all-nearest-neighbors: $\operatorname*{map}_{q \in X} \operatorname*{argmin}_{r \in X - q} d(q, r)$



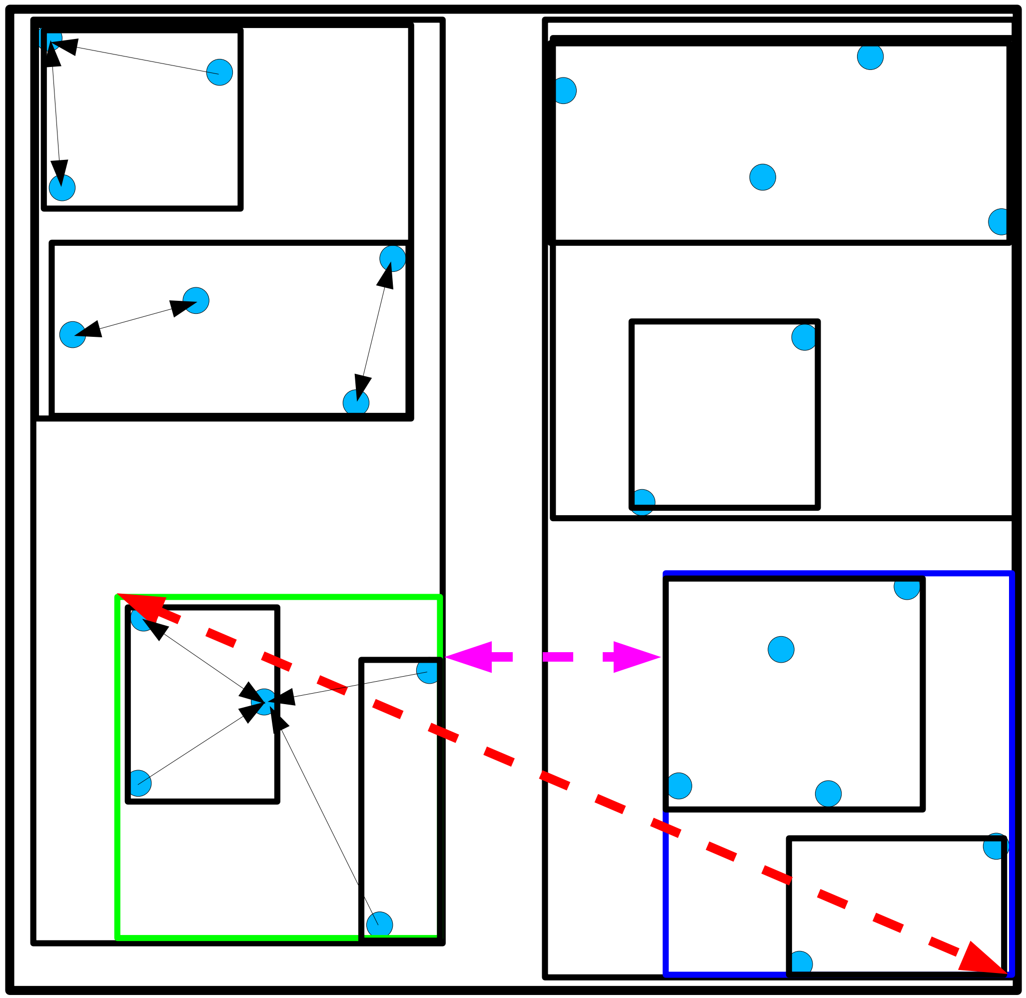
Monochromatic all-nearest-neighbors: $\operatorname*{map}_{q \in X} \operatorname*{argmin}_{r \in X - q} d(q, r)$



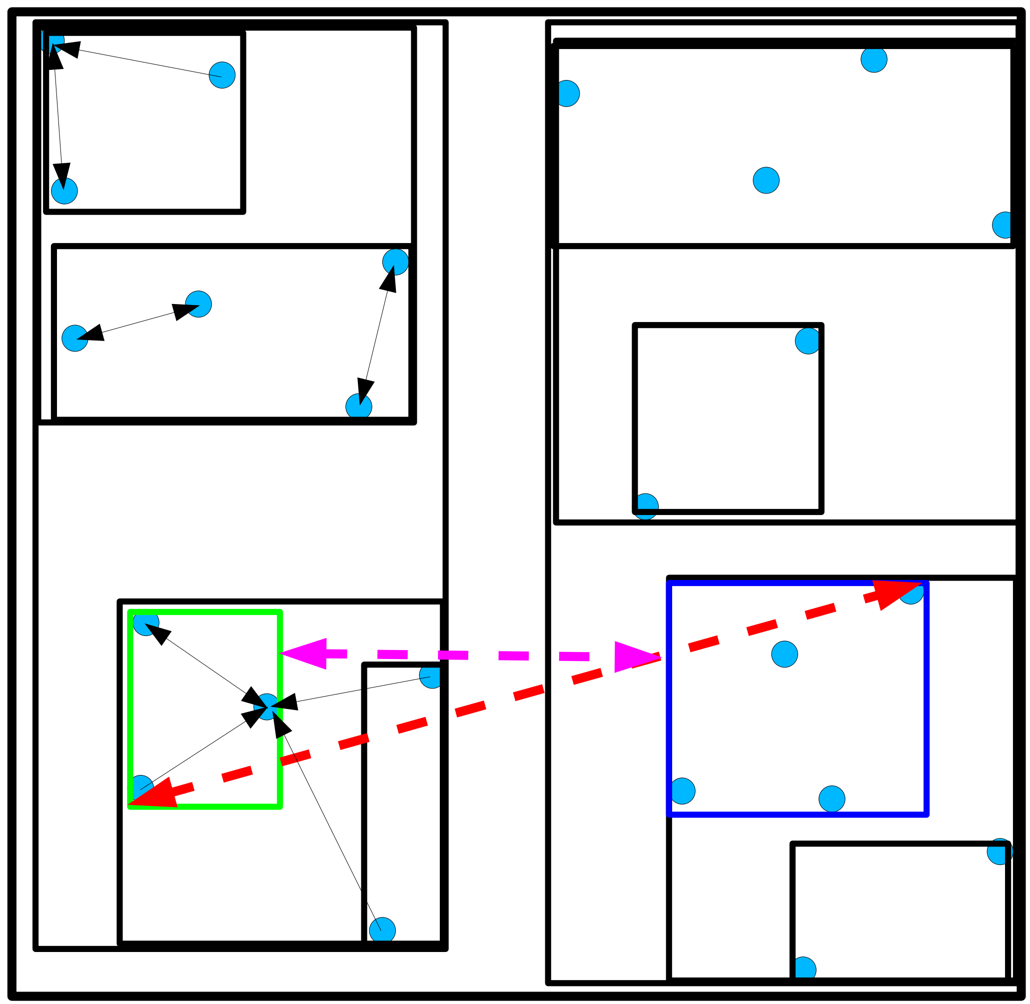
Monochromatic all-nearest-neighbors: $\operatorname*{map}_{q \in X} \operatorname*{argmin}_{r \in X - q} d(q, r)$



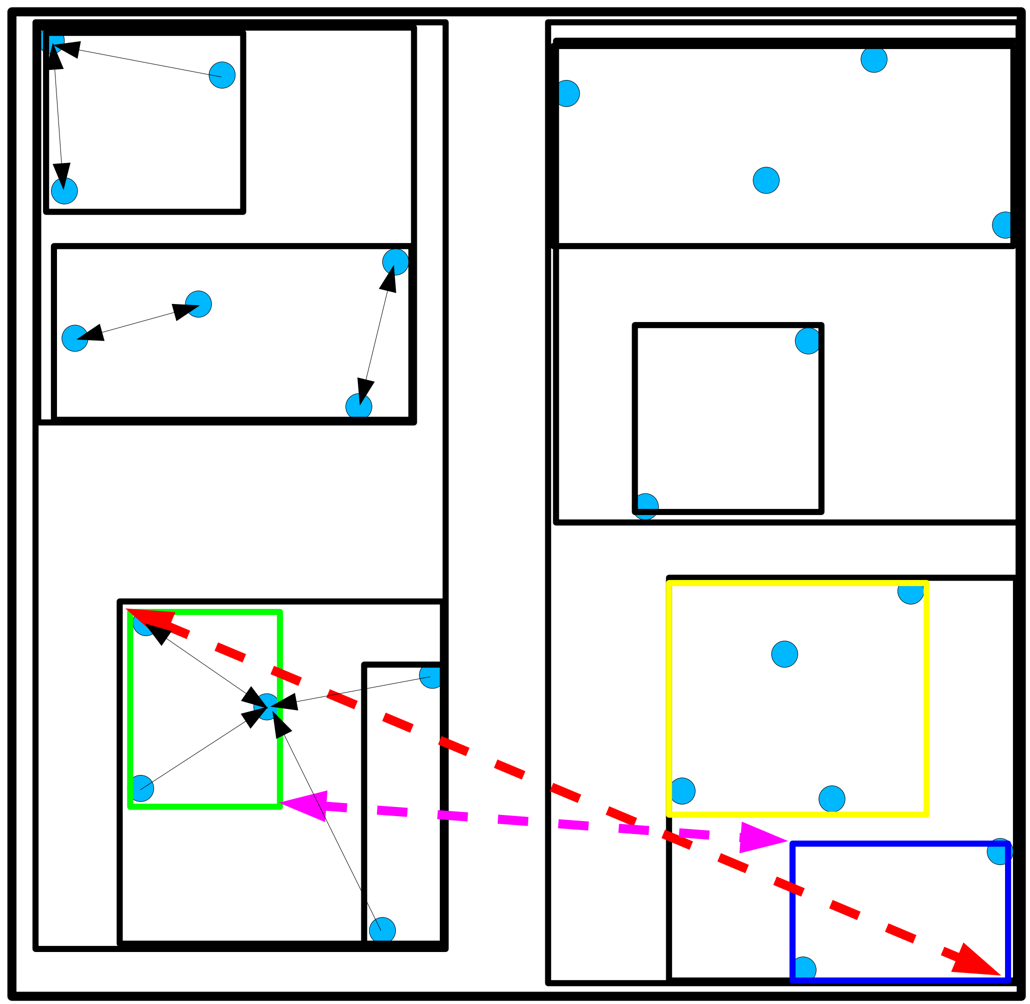
Monochromatic all-nearest-neighbors: $\underset{q \in X}{\text{map}} \underset{r \in X - q}{\text{argmin}}\, d(q, r)$



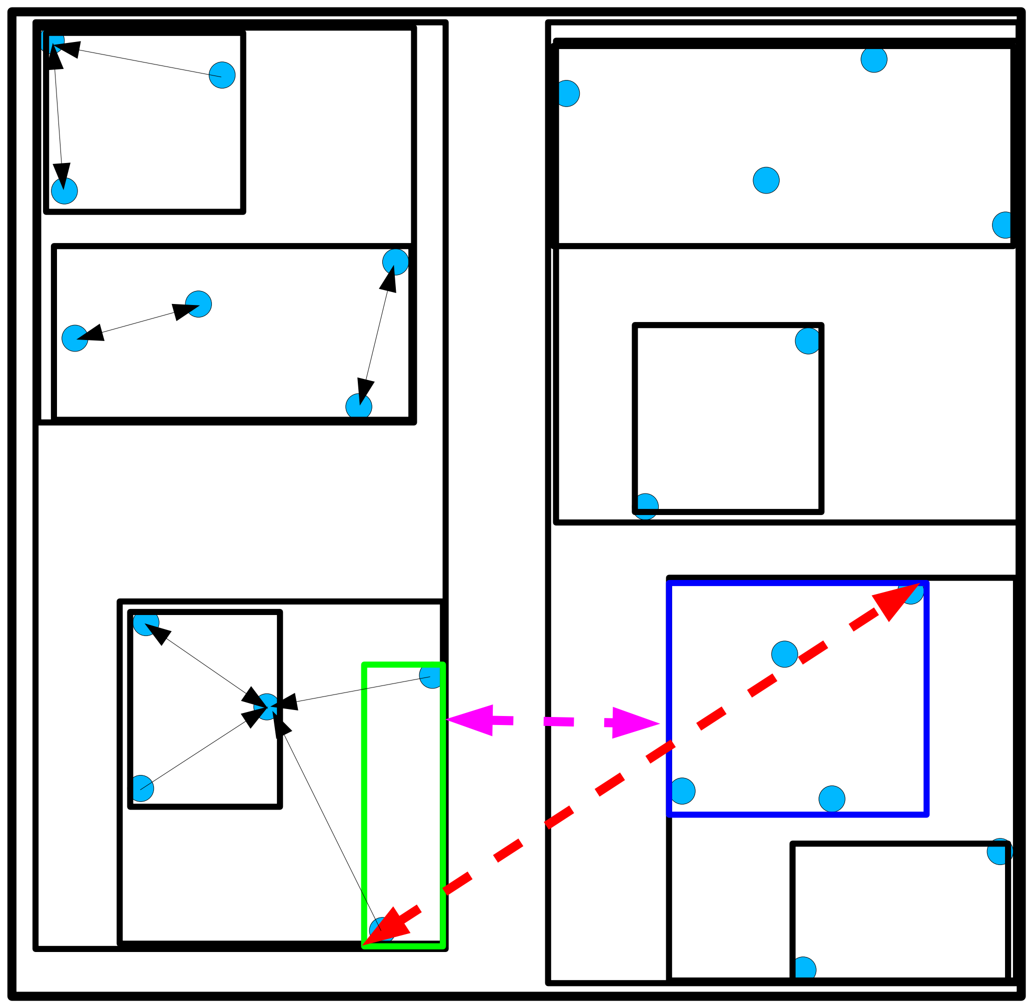
Monochromatic all-nearest-neighbors: $\operatorname*{map}_{q \in X} \operatorname*{argmin}_{r \in X - q} d(q, r)$



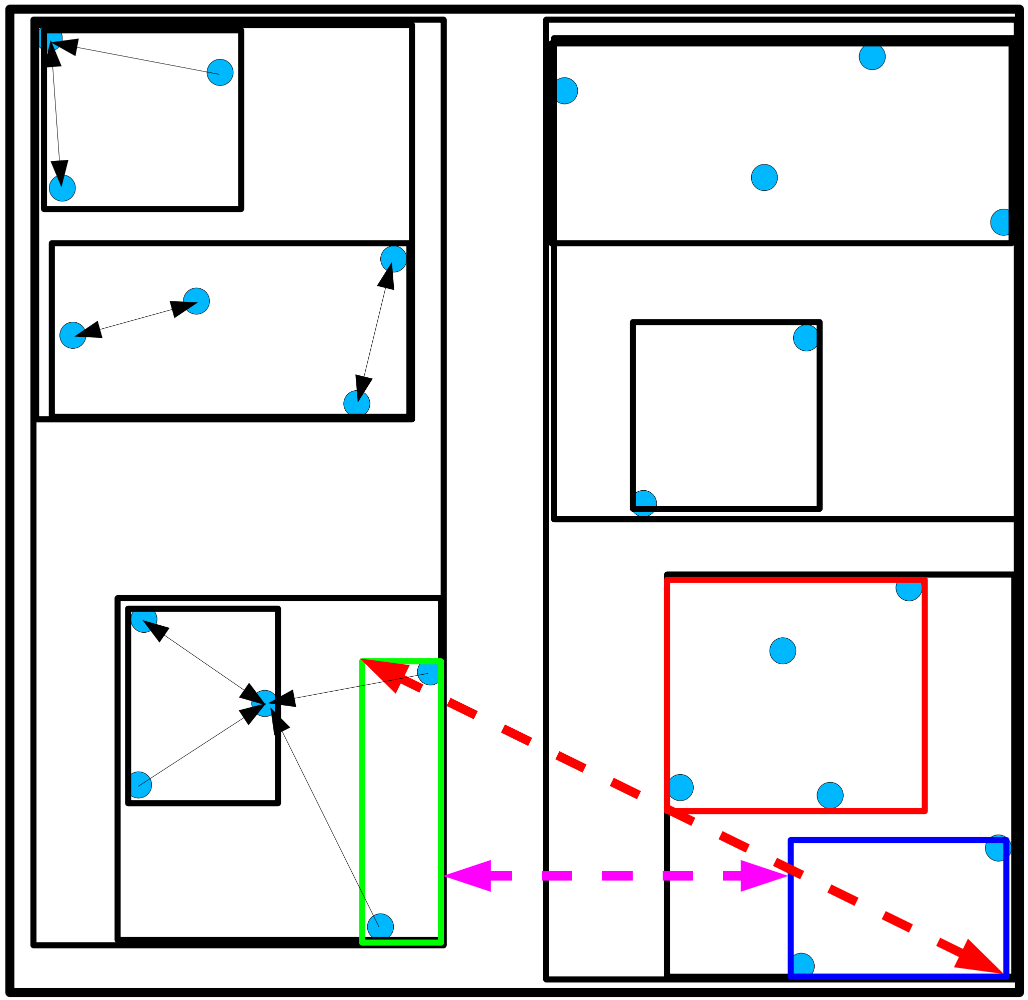
Monochromatic all-nearest-neighbors: $\operatorname*{map}_{q \in X} \operatorname*{argmin}_{r \in X - q} d(q, r)$



Monochromatic all-nearest-neighbors: $\underset{q \in X}{\text{map}} \; \underset{r \in X - q}{\text{argmin}} \; d(q, r)$



Monochromatic all-nearest-neighbors: $\operatorname*{map}_{q \in X} \operatorname*{argmin}_{r \in X - q} d(q, r)$



Ex: Two-point Correlation

Gray and Moore, NIPS 2000

$$\sum_{x_1 \in X} \sum_{x_2 \in X} I(d(x_1, x_2) \leq h)$$

$$\boxed{\begin{array}{l} \textbf{function}\ \mathrm{tpc}(X_1, X_2) \\ \quad \textbf{if}\ d^l(X_1, X_2) > h,\ \textbf{return}\ 0 \\ \quad \textbf{if}\ d^u(X_1, X_2) \leq h,\ \textbf{return}\ |X_1| \cdot |X_2| \\ \quad \textbf{return}\ \mathrm{tpc}(X_1^L, X_2^L) + \mathrm{tpc}(X_1^L, X_2^R) \\ \qquad + \mathrm{tpc}(X_1^R, X_2^L) + \mathrm{tpc}(X_1^R, X_2^R) \end{array}}$$

Ex: Range Count

Gray and Moore, NIPS 2000

$$\operatorname*{map}_{q \in Q} \sum_{r \in R} I(d(q,r) \le h)$$

$$\begin{array}{l}
\textbf{init } \forall q \in Q^{\text{root}}, a(q) = 0 \\
\textbf{function } \text{rng}(Q, R) \\
\quad \textbf{if } d^l(Q,R) > h, \ \textbf{return} \\
\quad \textbf{if } d^u(Q,R) \le h, \\
\quad\quad \forall q \in Q, a(q) \mathrel{+}= |R|; \ \textbf{return} \\
\quad \text{rng}(Q^L, R^L); \ \text{rng}(Q^L, R^R) \\
\quad \text{rng}(Q^R, R^L); \ \text{rng}(Q^R, R^R)
\end{array}$$

Ex: All-nearest-neighbors

Gray and Moore, NIPS 2000

$$\operatorname*{map}_{q \in Q} \operatorname*{argmin}_{r \in R} d(q, r)$$

$$
\begin{array}{l}
\textbf{init } \forall q \in Q^{\text{root}}, a(q) = \infty \\
\textbf{function } \text{allnn}(Q, R) \\
\quad \textbf{if } a^u(Q) \leq d^l(Q, R), \ \textbf{return} \\
\quad \textbf{if } (Q, R) = (\{q\}, \{r\}), \\
\qquad a(q) = \min\{a(q), d(q, r)\}; \ \textbf{return} \\
\quad \textbf{prioritize } \{R^1, R^2\} = \{R^L, R^R\} \textbf{ by } d^l(Q^L, \cdot) \\
\qquad \text{allnn}(Q^L, R^1); \ \text{allnn}(Q^L, R^2) \\
\quad \textbf{prioritize } \{R^1, R^2\} = \{R^L, R^R\} \textbf{ by } d^l(Q^R, \cdot) \\
\qquad \text{allnn}(Q^R, R^1); \ \text{allnn}(Q^R, R^2)
\end{array}
$$

Ex: Kernel Density Estimation

Lee *et al.*, NIPS 2005
Lee and Gray, UAI 2006

$$\operatorname*{map}_{q \in Q} \sum_{r \in R} K_h(q, r)$$

$$\begin{array}{l}
\textbf{init } \forall q \in Q^{\text{root}}, a(q) = 0;\ b = 0 \\
\textbf{function } \mathrm{kde}(Q, R, b) \\
\quad \textbf{if } K_h^u(Q,R) - K_h^l(Q,R) < (a^l(Q) + b)\frac{|R| \cdot \epsilon}{|R^{\text{root}}|}, \\
\qquad \forall q \in Q, a(q) \mathrel{+}= K_h^l(Q,R);\ \textbf{return} \\
\quad \textbf{prioritize } \{R^1, R^2\} = \{R^L, R^R\} \textbf{ by } d^l(Q^L, \cdot) \\
\qquad \mathrm{kde}(Q^L, R^1, b + K_h^l(Q^L, R^2));\ \mathrm{kde}(Q^L, R^2, b) \\
\quad \textbf{prioritize } \{R^1, R^2\} = \{R^L, R^R\} \textbf{ by } d^l(Q^R, \cdot) \\
\qquad \mathrm{kde}(Q^R, R^1, b + K_h^l(Q^R, R^2));\ \mathrm{kde}(Q^R, R^2, b)
\end{array}$$

Ex: Kernel Discriminant Analysis

Gray and Riegel, COMPSTAT 2006
Riegel *et al.*, SIAM Data Mining 2008

$$\operatorname*{map}_{q \in Q} \operatorname*{argmax}_{C \in \{C_1, C_2\}} \frac{P(C)}{|R_C|} \sum_{r \in R_C} K_{h_C}(q, r)$$

$$
\begin{array}{l}
\textbf{init } \forall q \in Q^{\mathrm{root}}, a(q) = \delta(Q^{\mathrm{root}}, R^{\mathrm{root}}) \\
\mathrm{enqueue}(Q^{\mathrm{root}}, R^{\mathrm{root}}) \\
\textbf{while } \mathrm{dequeue}(Q, R) \quad \text{// Main loop of } \mathrm{kda} \\
\quad \textbf{if } a^l(Q) > 0 \textbf{ or } a^u(Q) < 0, \textbf{return} \\
\quad \forall q \in Q, a(q) \mathrel{-}= \delta(Q, R) \\
\quad \forall q \in Q^L, a(q) \mathrel{+}= \delta(Q^L, R^L) + \delta(Q^L, R^R) \\
\quad \forall q \in Q^R, a(q) \mathrel{+}= \delta(Q^R, R^L) + \delta(Q^R, R^R) \\
\quad \mathrm{enqueue}(Q^L, R^L);\ \mathrm{enqueue}(Q^L, R^R) \\
\quad \mathrm{enqueue}(Q^R, R^L);\ \mathrm{enqueue}(Q^R, R^R)
\end{array}
$$

Case Study: Quasar Identification

Riegel *et al.*, SIAM Data Mining 2008
(Sumbitted) Richards *et al.*, AAS 2008

Mining for quasars in the Sloan Digital Sky Survey:

Case Study: Quasar Identification

Riegel *et al.*, SIAM Data Mining 2008
(Sumbitted) Richards *et al.*, AAS 2008

Mining for quasars in the Sloan Digital Sky Survey:

- Brightest objects in the universe

Case Study: Quasar Identification

Riegel *et al.*, SIAM Data Mining 2008
(Sumbitted) Richards *et al.*, AAS 2008

Mining for quasars in the Sloan Digital Sky Survey:

- Brightest objects in the universe
- Thus, the farthest/oldest we can see

Case Study: Quasar Identification

Riegel *et al.*, SIAM Data Mining 2008
(Sumbitted) Richards *et al.*, AAS 2008

Mining for quasars in the Sloan Digital Sky Survey:

- Brightest objects in the universe
- Thus, the farthest/oldest we can see
- Believed to be active galactic nuclei: giant black holes

Case Study: Quasar Identification

Riegel *et al.*, SIAM Data Mining 2008
(Sumbitted) Richards *et al.*, AAS 2008

Mining for quasars in the Sloan Digital Sky Survey:

- Brightest objects in the universe
- Thus, the farthest/oldest we can see
- Believed to be active galactic nuclei: giant black holes
- Implications for dark matter, dark energy, etc.

Case Study: Quasar Identification

Riegel *et al.*, SIAM Data Mining 2008
(Sumbitted) Richards *et al.*, AAS 2008

Mining for quasars in the Sloan Digital Sky Survey:

- Brightest objects in the universe
- Thus, the farthest/oldest we can see
- Believed to be active galactic nuclei: giant black holes
- Implications for dark matter, dark energy, etc.
- Peplow, Nature 2005 uses one of our catalogs to verify the cosmic magnification effect predicted by relativity

Case Study: Quasar Identification

Riegel *et al.*, SIAM Data Mining 2008
(Sumbitted) Richards *et al.*, AAS 2008

Trained a KDA classifier on 4D spectra data from about 80k known quasars and 400k non-quasars.

Identified about 1m quasars from 40m unknown objects.

Case Study: Quasar Identification

Riegel *et al.*, SIAM Data Mining 2008
(Sumbitted) Richards *et al.*, AAS 2008

Trained a KDA classifier on 4D spectra data from about 80k known quasars and 400k non-quasars.

Identified about 1m quasars from 40m unknown objects.

Took 640 seconds in serial; half of that was tree-building.

Case Study: Quasar Identification

Riegel *et al.*, SIAM Data Mining 2008
(Sumbitted) Richards *et al.*, AAS 2008

Trained a KDA classifier on 4D spectra data from about 80k known quasars and 400k non-quasars.

Identified about 1m quasars from 40m unknown objects.

Took 640 seconds in serial; half of that was tree-building. Naïve's takes 380 hours, excluding bandwidth learning.

Case Study: Quasar Identification

Riegel *et al.*, SIAM Data Mining 2008
(Sumbitted) Richards *et al.*, AAS 2008

Trained a KDA classifier on 4D spectra data from about 80k known quasars and 400k non-quasars.

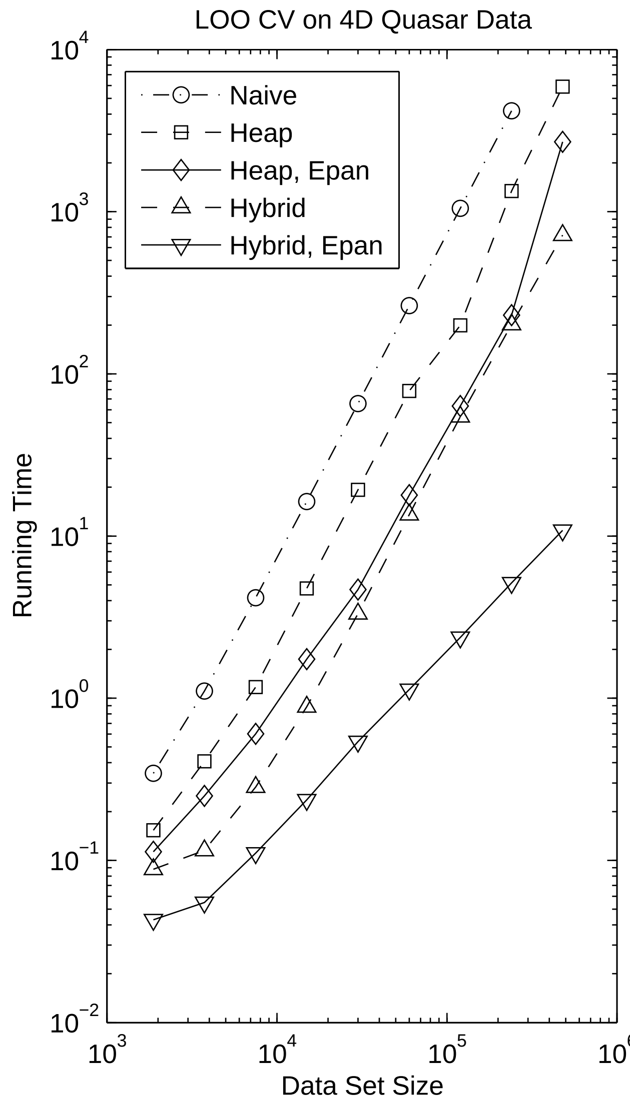
Identified about 1m quasars from 40m unknown objects.

Took 640 seconds in serial; half of that was tree-building. Naïve's takes 380 hours, excluding bandwidth learning.

Algorithmic parameters are key to performance:

- Hybrid breadth-depth first expansion
- Epanechnikov kernel (choice of f) to maximize pruning
- Multi-bandwidth algorithm for faster bandwidth fitting

Case Study: Quasar Identification



GNPs, Formally Speaking

Boyer, Riegel, and Gray's *THOR Project*
(Planned) Riegel *et al.*, NIPS 2008 or JMLR 2008

Higher-order reduce problem $\Psi = g \circ \psi$, with

$$\psi(X_1, \ldots, X_n) = \bigotimes_{x_1 \in X_1}{}_1 \cdots \bigotimes_{x_n \in X_n}{}_n f(x_1, \ldots, x_n)$$

GNPs, Formally Speaking

Boyer, Riegel, and Gray's *THOR Project*
(Planned) Riegel *et al.*, NIPS 2008 or JMLR 2008

Higher-order reduce problem $\Psi = g \circ \psi$, with

$$\psi(X_1, \ldots, X_n) = \bigotimes_{x_1 \in X_1}{}_1 \cdots \bigotimes_{x_n \in X_n}{}_n f(x_1, \ldots, x_n)$$

subject to decomposability requirement

$$\psi(\ldots, X_i, \ldots) = \psi(\ldots, X_i^L, \ldots) \otimes_i \psi(\ldots, X_i^R, \ldots)$$

for all $1 \leq i \leq n$ and partitions $X_i^L \cup X_i^R = X_i$.

GNPs, Formally Speaking

Boyer, Riegel, and Gray's *THOR Project*
(Planned) Riegel *et al.*, NIPS 2008 or JMLR 2008

Higher-order reduce problem $\Psi = g \circ \psi$, with

$$\psi(X_1, \ldots, X_n) = \bigotimes_{x_1 \in X_1}{}_1 \cdots \bigotimes_{x_n \in X_n}{}_n f(x_1, \ldots, x_n)$$

subject to decomposability requirement

$$\psi(\ldots, X_i, \ldots) = \psi(\ldots, X_i^L, \ldots) \otimes_i \psi(\ldots, X_i^R, \ldots)$$

for all $1 \leq i \leq n$ and partitions $X_i^L \cup X_i^R = X_i$.

We'll also need some means of bounding the results of ψ.

Decomposability

(Planned) Riegel *et al.*, NIPS 2008 or JMLR 2008

Decomposability is restrictive; always possible for problems formed by combinations of map and some one other $\otimes$.

Decomposability

(Planned) Riegel *et al.*, NIPS 2008 or JMLR 2008

Decomposability is restrictive; always possible for problems formed by combinations of map and some one other $\otimes$.

It is equivalent to

$$\bigotimes_{x_1 \in X_1}{}_1 \cdots \bigotimes_{x_n \in X_n}{}_n f(x_1, \cdots, x_n) = \bigotimes_{x_{p_1} \in X_{p_1}}{}_{p_1} \cdots \bigotimes_{x_{p_n} \in X_{p_n}}{}_{p_n} f(x_1, \cdots, x_n)$$

for all permutations p of the set $\{1, \ldots, n\}$,

Decomposability

(Planned) Riegel *et al.*, NIPS 2008 or JMLR 2008

Decomposability is restrictive; always possible for problems formed by combinations of map and some one other $\otimes$.

It is equivalent to

$$\bigotimes_{x_1 \in X_1}{}_1 \cdots \bigotimes_{x_n \in X_n}{}_n f(x_1, \cdots, x_n) = \bigotimes_{x_{p_1} \in X_{p_1}}{}_{p_1} \cdots \bigotimes_{x_{p_n} \in X_{p_n}}{}_{p_n} f(x_1, \cdots, x_n)$$

for all permutations p of the set $\{1, \ldots, n\}$, and to

$$\begin{aligned}&(\psi(X_i^L, X_j^L) \otimes_i \psi(X_i^R, X_j^L)) \otimes_j (\psi(X_i^L, X_j^R) \otimes_i \psi(X_i^R, X_j^R)) \\ &\quad = (\psi(X_i^L, X_j^L) \otimes_j \psi(X_i^L, X_j^R)) \otimes_i (\psi(X_i^R, X_j^L) \otimes_j \psi(X_i^R, X_j^R))\end{aligned}$$

Decomposability

$$\psi(X,Y) = \bigodot_{x \in X} \bigotimes_{y \in Y} f(x,y)$$

$$\begin{array}{c}
(\ f(x_1,y_1) \otimes f(x_1,y_2) \otimes \cdots \otimes f(x_1,y_M)\ ) \\
\odot \\
(\ f(x_2,y_1) \otimes f(x_2,y_2) \otimes \cdots \otimes f(x_2,y_M)\ ) \\
\odot \\
\vdots \\
\odot \\
(\ f(x_N,y_1) \otimes f(x_N,y_2) \otimes \cdots \otimes f(x_N,y_M)\ )
\end{array}$$

Decomposability

$$\psi(X,Y) = \psi(X,Y^L) \otimes \psi(X,Y^R)$$

$$\begin{pmatrix} f(x_1,y_1) \\ \odot \\ f(x_2,y_1) \\ \odot \\ \vdots \\ \odot \\ f(x_N,y_1) \end{pmatrix} \otimes \begin{pmatrix} (\ f(x_1,y_2) \otimes \cdots \otimes f(x_1,y_M)\ ) \\ \odot \\ (\ f(x_2,y_2) \otimes \cdots \otimes f(x_2,y_M)\ ) \\ \odot \\ \vdots \\ \odot \\ (\ f(x_N,y_2) \otimes \cdots \otimes f(x_N,y_M)\ ) \end{pmatrix}$$

Transforming Problems into GNPs

(Planned) Riegel *et al.*, NIPS 2008 or JMLR 2008

("Serial" GNPs.) Decomposable or not,

$$g_1\Big(\bigotimes_{x_1\in X_1}{}_1\, g_2\Big(\bigotimes_{x_2\in X_2}{}_2 \cdots g_n\Big(\bigotimes_{x_n\in X_n}{}_n f(x_1,\ldots,x_n)\Big)\cdots\Big)\Big)$$

may be transformed into nested GNPs by replacing every other operator with map and factoring intermediate g_i out.

Transforming Problems into GNPs

(Planned) Riegel *et al.*, NIPS 2008 or JMLR 2008

("Serial" GNPs.) Decomposable or not,

$$g_1\Big(\bigotimes_{x_1 \in X_1}{}_1\, g_2\Big(\bigotimes_{x_2 \in X_2}{}_2 \cdots g_n\Big(\bigotimes_{x_n \in X_n}{}_n\, f(x_1, \ldots, x_n)\Big)\cdots\Big)\Big)$$

may be transformed into nested GNPs by replacing every other operator with map and factoring intermediate g_i out.

("Parallel" GNPs.) Also GNP-able are problems such as:

$$\operatorname*{map}_i \frac{\sum_j w_{ij} K(x_i, x_j)}{\sum_j K(x_i, x_j)}$$

Transforming Problems into GNPs

(Planned) Riegel *et al.*, NIPS 2008 or JMLR 2008

("Serial" GNPs.) Decomposable or not,

$$g_1\Big(\bigotimes_{x_1 \in X_1}{}_1\, g_2\Big(\bigotimes_{x_2 \in X_2}{}_2 \cdots g_n\Big(\bigotimes_{x_n \in X_n}{}_n\, f(x_1,\ldots,x_n)\Big)\cdots\Big)\Big)$$

may be transformed into nested GNPs by replacing every other operator with map and factoring intermediate g_i out.

("Parallel" GNPs.) Also GNP-able are problems such as:

$$\operatorname*{map}_i \frac{\sum_j w_{ij} K(x_i, x_j)}{\sum_j K(x_i, x_j)}$$

("Multi" GNPs.) Wrap problem with map to vary parameter.

The Algorithm

Boyer, Riegel, and Gray's *THOR Project*
(Planned) Riegel *et al.*, ICML 2008 or JMLR 2008

"One algorithm to solve them all":

$$\psi(X_1, \dots, X_n) \leftarrow \begin{cases} a & \text{if bounds prove it is safe to prune to } a, \\ f(x_1, \dots, x_n) & \text{if each } X_i = \{x_i\}\text{, i.e. is leaf,} \\ \psi(\dots, X_i^L, \dots) \otimes_i \psi(\dots, X_i^R, \dots) & \text{otherwise} \end{cases}$$

The Algorithm

Boyer, Riegel, and Gray's *THOR Project*
(Planned) Riegel *et al.*, ICML 2008 or JMLR 2008

"One algorithm to solve them all":

$$\psi(X_1, \ldots, X_n) \leftarrow \begin{cases} a & \text{if bounds prove it is safe to prune to } a, \\ f(x_1, \ldots, x_n) & \text{if each } X_i = \{x_i\}\text{, i.e. is leaf,} \\ \psi(\ldots, X_i^L, \ldots) \otimes_i \psi(\ldots, X_i^R, \ldots) & \text{otherwise} \end{cases}$$

Regarding speed, pruning is everything.

Pruning

Boyer, Riegel, and Gray's *THOR Project*
(Planned) Riegel *et al.*, ICML 2008 or JMLR 2008

Roughly, three kinds:

Pruning

Boyer, Riegel, and Gray's *THOR Project*
(Planned) Riegel *et al.*, ICML 2008 or JMLR 2008

Roughly, three kinds:

- Intrinsic pruning depends only on bounds of $\psi(X_1, \ldots, X_n)$ (ex: n-point correlation)

Pruning

Boyer, Riegel, and Gray's *THOR Project*
(Planned) Riegel *et al.*, ICML 2008 or JMLR 2008

Roughly, three kinds:

- Intrinsic pruning depends only on bounds of $\psi(X_1, \ldots, X_n)$ (ex: n-point correlation)
- Extrinsic pruning depends on bounds of $\psi(X_1, \ldots, X_n)$ and past work (ex: all-nearest-neighbors)

Pruning

Boyer, Riegel, and Gray's *THOR Project*
(Planned) Riegel *et al.*, ICML 2008 or JMLR 2008

Roughly, three kinds:

- Intrinsic pruning depends only on bounds of $\psi(X_1, \ldots, X_n)$ (ex: n-point correlation)
- Extrinsic pruning depends on bounds of $\psi(X_1, \ldots, X_n)$ and past work (ex: all-nearest-neighbors)
- Termination pruning depends only on past work (ex: kernel discriminant analysis)

Pruning

Boyer, Riegel, and Gray's *THOR Project*
(Planned) Riegel *et al.*, ICML 2008 or JMLR 2008

Roughly, three kinds:

- Intrinsic pruning depends only on bounds of $\psi(X_1, \ldots, X_n)$ (ex: n-point correlation)
- Extrinsic pruning depends on bounds of $\psi(X_1, \ldots, X_n)$ and past work (ex: all-nearest-neighbors)
- Termination pruning depends only on past work (ex: kernel discriminant analysis)

Approximation is a form of extrinsic pruning.

Pruning

Boyer, Riegel, and Gray's *THOR Project*
(Planned) Riegel *et al.*, ICML 2008 or JMLR 2008

Roughly, three kinds:

- Intrinsic pruning depends only on bounds of $\psi(X_1, \ldots, X_n)$ (ex: n-point correlation)
- Extrinsic pruning depends on bounds of $\psi(X_1, \ldots, X_n)$ and past work (ex: all-nearest-neighbors)
- Termination pruning depends only on past work (ex: kernel discriminant analysis)

Approximation is a form of extrinsic pruning.

Kind of pruning determined by problem specification.
Ease of pruning influenced by algorithmic parameters.

Algorithmic Parameters

An implementation must answer these questions:

Algorithmic Parameters

An implementation must answer these questions:

- How to partition data? E.g. what kind of trees to use? Non-binary? No tree (ex: Baeza-Yeats)?

Algorithmic Parameters

An implementation must answer these questions:

- How to partition data? E.g. what kind of trees to use? Non-binary? No tree (ex: Baeza-Yeats)?
- What expansion pattern? Depth-first? Breath-first? Something else? Which branches first, heuristically?

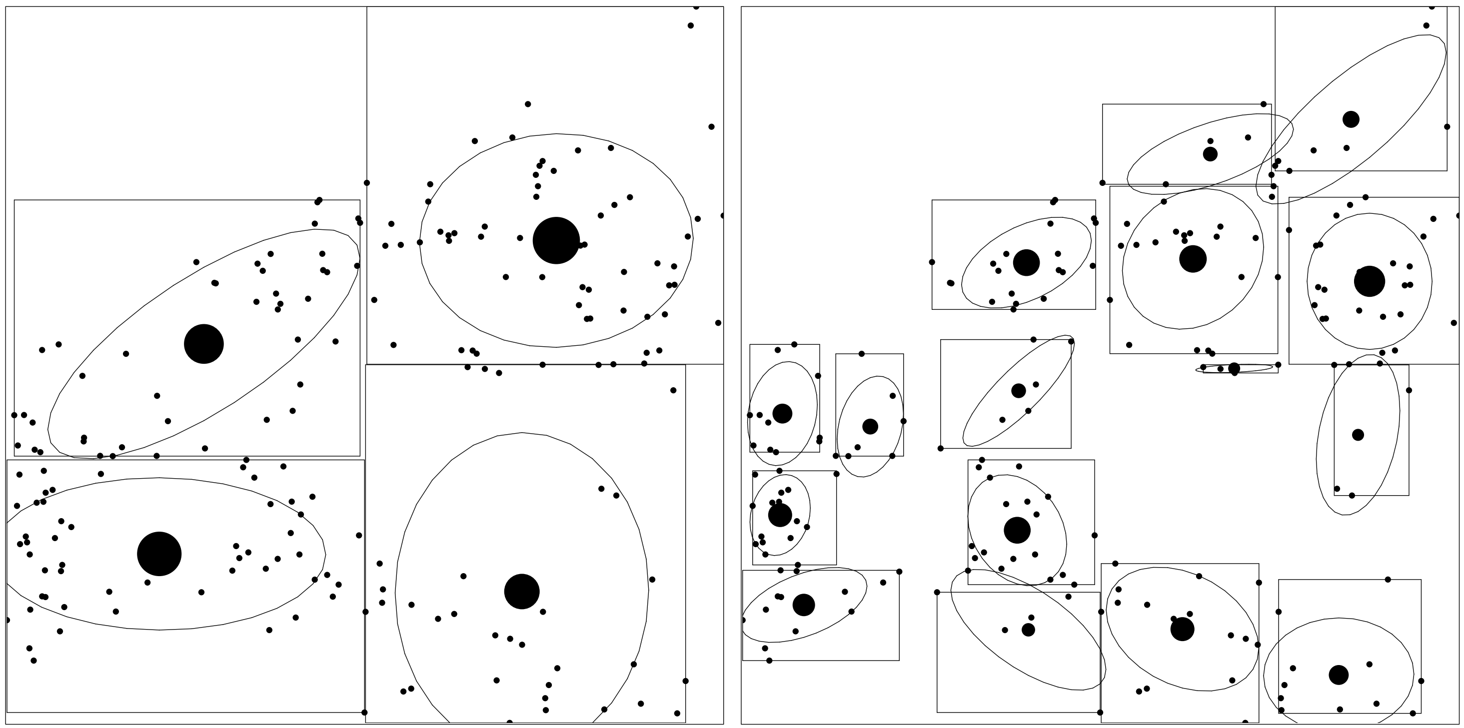
Algorithmic Parameters

An implementation must answer these questions:

- How to partition data? E.g. what kind of trees to use? Non-binary? No tree (ex: Baeza-Yeats)?
- What expansion pattern? Depth-first? Breath-first? Something else? Which branches first, heuristically?
- (Higher-level.) What scale of data structures to use? Does the problem fit in RAM? Need to be parallel?

Trees

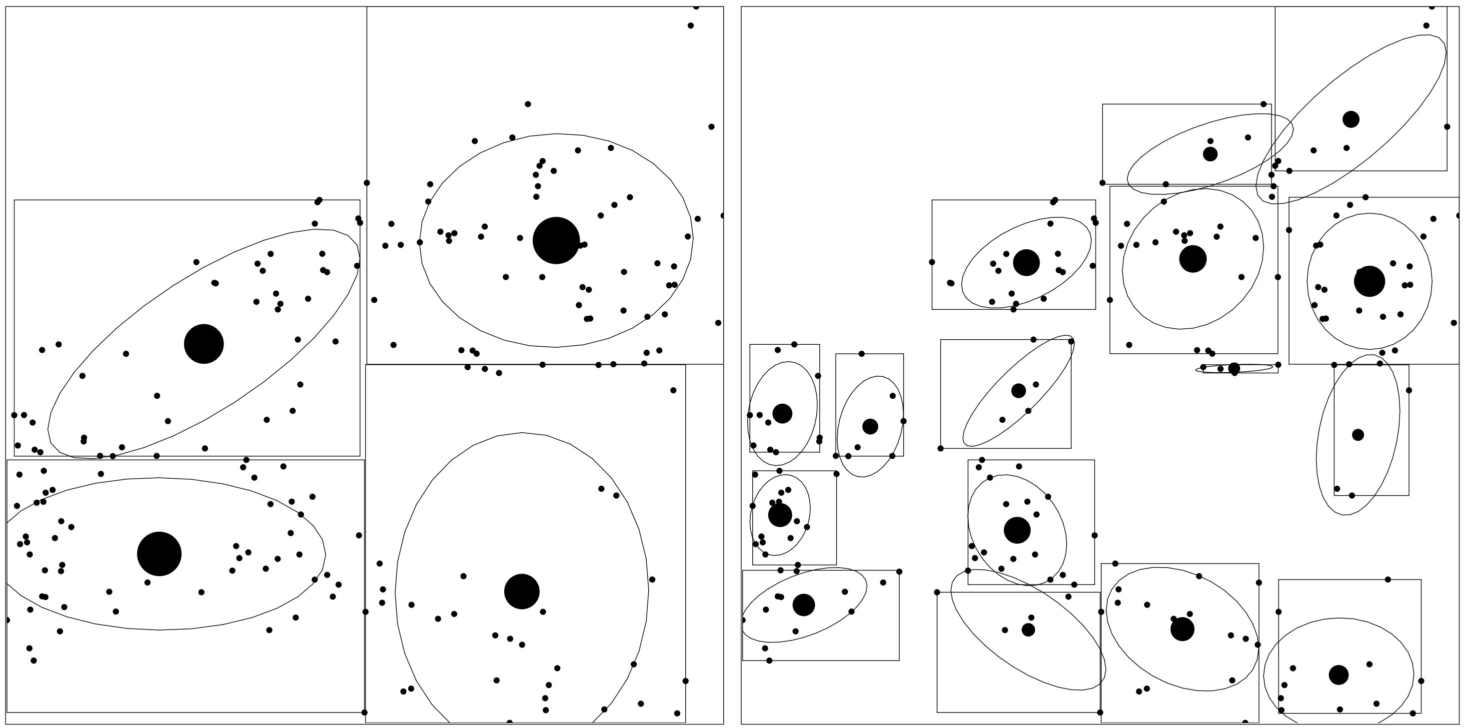
Gray and Lee's *Proximity Project*, 2005



Many options: kd-trees, ball trees, cover trees, sorted lists.

Trees

Gray and Lee's *Proximity Project*, 2005



Many options: kd-trees, ball trees, cover trees, sorted lists.

Aside: tree building constitutes graph partitioning and may (attempt to) minimize some loss function.

Expansion Pattern

Boyer, Riegel, and Gray's *THOR Project*
(Planned) Riegel *et al.*, ICML 2008 or JMLR 2008

Describes the order we replace

$$\psi(\ldots, X_i, \ldots) \leftarrow \psi(\ldots, X_i^L, \ldots) \otimes_i \psi(\ldots, X_i^R, \ldots)$$

Expansion Pattern

Boyer, Riegel, and Gray's *THOR Project*
(Planned) Riegel *et al.*, ICML 2008 or JMLR 2008

Describes the order we replace

$$\psi(\ldots, X_i, \ldots) \leftarrow \psi(\ldots, X_i^L, \ldots) \otimes_i \psi(\ldots, X_i^R, \ldots)$$

- DFS has least overhead, sensitive to heuristic

Expansion Pattern

Boyer, Riegel, and Gray's *THOR Project*
(Planned) Riegel *et al.*, ICML 2008 or JMLR 2008

Describes the order we replace

$$\psi(\ldots, X_i, \ldots) \leftarrow \psi(\ldots, X_i^L, \ldots) \otimes_i \psi(\ldots, X_i^R, \ldots)$$

- DFS has least overhead, sensitive to heuristic
- BFS has more overhead, less senitive to heuristic

Expansion Pattern

Boyer, Riegel, and Gray's *THOR Project*
(Planned) Riegel *et al.*, ICML 2008 or JMLR 2008

Describes the order we replace

$$\psi(\ldots, X_i, \ldots) \leftarrow \psi(\ldots, X_i^L, \ldots) \otimes_i \psi(\ldots, X_i^R, \ldots)$$

- DFS has least overhead, sensitive to heuristic
- BFS has more overhead, less senitive to heuristic
- Priority queue has highest overhead but makes the most of its heuristic; adds need for operators to have inverses (i.e. to form groups)

Expansion Pattern

Boyer, Riegel, and Gray's *THOR Project*
(Planned) Riegel *et al.*, ICML 2008 or JMLR 2008

Describes the order we replace

$$\psi(\ldots, X_i, \ldots) \leftarrow \psi(\ldots, X_i^L, \ldots) \otimes_i \psi(\ldots, X_i^R, \ldots)$$

- DFS has least overhead, sensitive to heuristic
- BFS has more overhead, less senitive to heuristic
- Priority queue has highest overhead but makes the most of its heuristic; adds need for operators to have inverses (i.e. to form groups)

Hybrid breadth-depth first pattern: achieves breadth-first behavior in $O(N)$ space for query-reference problems.

Problem Scale

Boyer, Riegel, and Gray's *THOR Project*

Simple in-memory data structures, memory-mapped files, or parallelized/distributed data management.

Some observations:

Problem Scale

Boyer, Riegel, and Gray's *THOR Project*

Simple in-memory data structures, memory-mapped files, or parallelized/distributed data management.

Some observations:

- All GNPs are parallelizable, some more so than others

Problem Scale

Boyer, Riegel, and Gray's *THOR Project*

Simple in-memory data structures, memory-mapped files, or parallelized/distributed data management.

Some observations:

- All GNPs are parallelizable, some more so than others
- All GNPs can benefit greatly from multicore processors

Problem Scale

Boyer, Riegel, and Gray's *THOR Project*

Simple in-memory data structures, memory-mapped files, or parallelized/distributed data management.

Some observations:

- All GNPs are parallelizable, some more so than others
- All GNPs can benefit greatly from multicore processors
- Opportunity to use cache-oblivious trees (vEB, etc.)

THOR Coding Framework

Boyer, Riegel, and Gray's *THOR Project*

Speed-oriented C++ framework for problems of forms

$$\operatorname*{map}_{q \in Q} g\left(\bigotimes_{r \in R} f(q, r) \right) \quad \text{and} \quad g\left(\bigotimes_{x_1 \in X_1} \bigotimes_{x_2 \in X_2} f(x_1, x_2) \right)$$

THOR Coding Framework

Boyer, Riegel, and Gray's *THOR Project*

Speed-oriented C++ framework for problems of forms

$$\operatorname*{map}_{q \in Q} g\left(\bigotimes_{r \in R} f(q, r) \right) \quad \text{and} \quad g\left(\bigotimes_{x_1 \in X_1} \bigotimes_{x_2 \in X_2} f(x_1, x_2) \right)$$

- Coding entails filling a few dozen function stubs

THOR Coding Framework

Boyer, Riegel, and Gray's *THOR Project*

Speed-oriented C++ framework for problems of forms

$$\operatorname*{map}_{q \in Q} g\Bigg(\bigotimes_{r \in R} f(q, r)\Bigg) \quad \text{and} \quad g\Bigg(\bigotimes_{x_1 \in X_1} \bigotimes_{x_2 \in X_2} f(x_1, x_2)\Bigg)$$

- Coding entails filling a few dozen function stubs
- Easy variation of tree type, expansion pattern, etc.

THOR Coding Framework

Boyer, Riegel, and Gray's *THOR Project*

Speed-oriented C++ framework for problems of forms

$$\operatorname*{map}_{q \in Q} g\left(\bigotimes_{r \in R} f(q, r) \right) \quad \text{and} \quad g\left(\bigotimes_{x_1 \in X_1} \bigotimes_{x_2 \in X_2} f(x_1, x_2) \right)$$

- Coding entails filling a few dozen function stubs
- Easy variation of tree type, expansion pattern, etc.
- Automatic parallelization (multicore and distributed)

Case Study: Affinity Propagation

(Planned) Riegel *et al.*, NIPS 2008 or JMLR 2008

Recent clustering method:

Case Study: Affinity Propagation

(Planned) Riegel *et al.*, NIPS 2008 or JMLR 2008

Recent clustering method:

- Frey and Dueck, *Science* 2007

Case Study: Affinity Propagation

(Planned) Riegel *et al.*, NIPS 2008 or JMLR 2008

Recent clustering method:

- Frey and Dueck, *Science* 2007
- Finds exemplars in a data set in attempt to minimize square reconstruction error

Case Study: Affinity Propagation

(Planned) Riegel *et al.*, NIPS 2008 or JMLR 2008

Recent clustering method:

- Frey and Dueck, *Science* 2007
- Finds exemplars in a data set in attempt to minimize square reconstruction error
- Number of clusters to find is unspecified, but influenced by a “preference” parameter

Case Study: Affinity Propagation

(Planned) Riegel *et al.*, NIPS 2008 or JMLR 2008

Recent clustering method:

- Frey and Dueck, *Science* 2007
- Finds exemplars in a data set in attempt to minimize square reconstruction error
- Number of clusters to find is unspecified, but influenced by a “preference” parameter
- Presented as fast alternative to zillions of random restarts of k-centers algorithm

Case Study: Affinity Propagation

(Planned) Riegel *et al.*, NIPS 2008 or JMLR 2008

For similarity matrix S (pref along diag), update R and A

$$r_{ij} \leftarrow s_{ij} - \max_{j' \neq j}(a_{ij'} + s_{ij'})$$

$$a_{ij} \leftarrow \kappa^{-}_{ij}\Big(\sum_{i' \neq i} \kappa^{+}_{i'j}(r_{i'j})\Big)$$

Case Study: Affinity Propagation

(Planned) Riegel *et al.*, NIPS 2008 or JMLR 2008

For similarity matrix S (pref along diag), update R and A

$$r_{ij} \leftarrow s_{ij} - \max_{j' \neq j}(a_{ij'} + s_{ij'})$$

$$a_{ij} \leftarrow \kappa^{-}_{ij}\Bigg(\sum_{i' \neq i} \kappa^{+}_{i'j}(r_{i'j})\Bigg)$$

Damping of R and A helps convergence.

Case Study: Affinity Propagation

(Planned) Riegel *et al.*, NIPS 2008 or JMLR 2008

Naïvely $O(N^2)$; can be improved if S is made to be sparse, but this introduces uncontrolled error.

Case Study: Affinity Propagation

(Planned) Riegel *et al.*, NIPS 2008 or JMLR 2008

Naïvely $O(N^2)$; can be improved if S is made to be sparse, but this introduces uncontrolled error.

Alterately, if no damping, we can rearrange into GNPs

$$\alpha_i \leftarrow \underset{j}{\operatorname{argmax2}}(\kappa^+_{ij}(\kappa^+_{ij}(s_{ij} + \alpha_{i[j]}) - \rho_j) - s_{ij})$$

$$\rho_j \leftarrow \sum_i \kappa^+_{ij}(s_{ij} + \alpha_{i[j]})$$

Case Study: Affinity Propagation

(Planned) Riegel *et al.*, NIPS 2008 or JMLR 2008

Naïvely $O(N^2)$; can be improved if S is made to be sparse, but this introduces uncontrolled error.

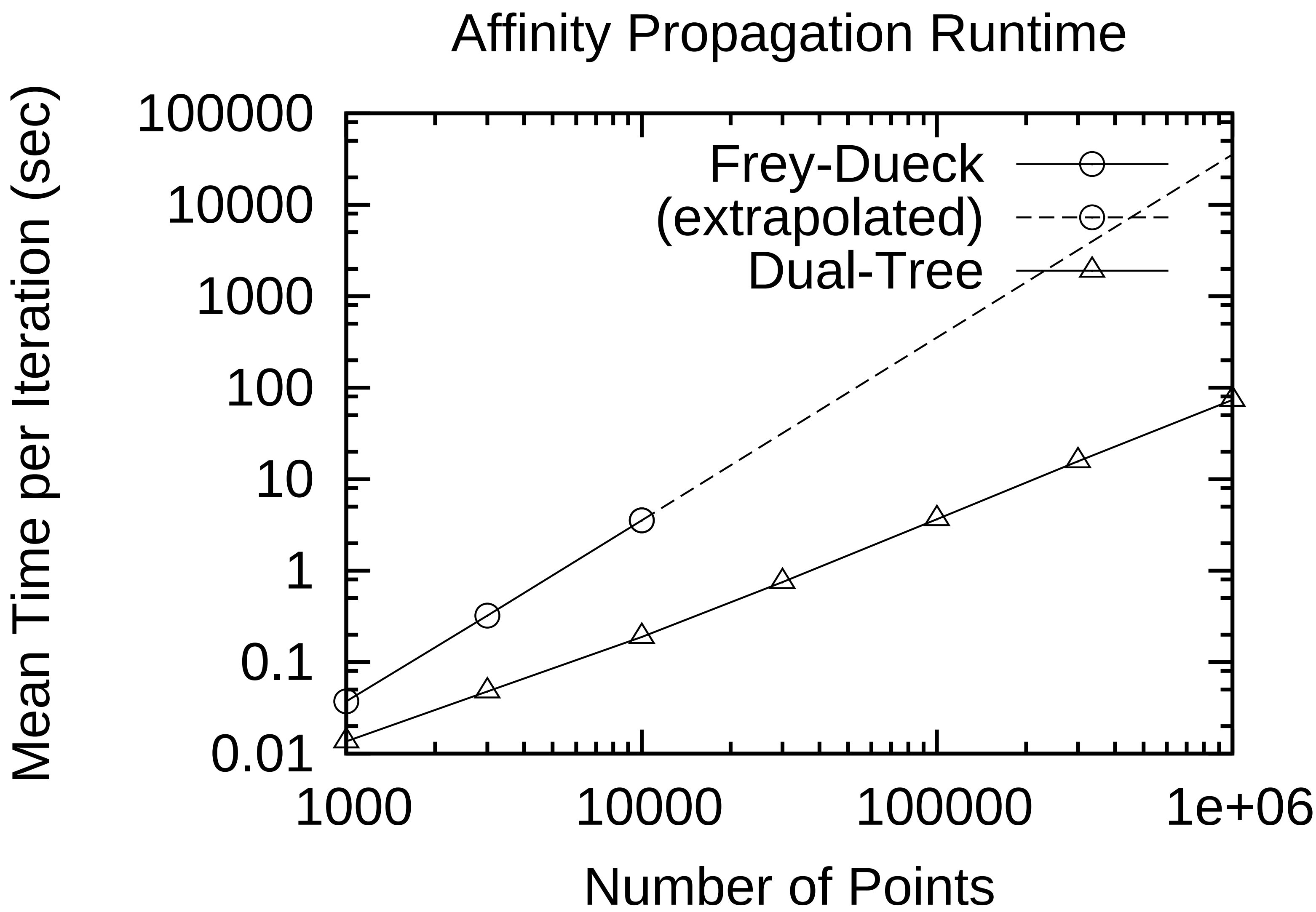
Alterately, if no damping, we can rearrange into GNPs

$$\alpha_i \leftarrow \underset{j}{\mathrm{argmax2}}(\kappa^+_{ij}(\kappa^+_{ij}(s_{ij} + \alpha_{i[j]}) - \rho_j) - s_{ij})$$

$$\rho_j \leftarrow \sum_i \kappa^+_{ij}(s_{ij} + \alpha_{i[j]})$$

Can pull other tricks to get things to converge.

Case Study: Affinity Propagation



fin.