SCOUT: Prefetching for Latent Structure Following Queries

Siting Ma

Introduction

- One problem many scientists share is the **analysis of the massive spatial models** they build.
- What is a **massive spatial model**? e.g., the arterial tree, neuron fibers, etc.
- Each query takes long to execute, and the total time for executing a sequence of queries significantly delays data analysis.
- **Prefetching** the spatial data **reduces the response time** considerably, but known approaches do not prefetch with high accuracy.

Example of Massive Spatial Model

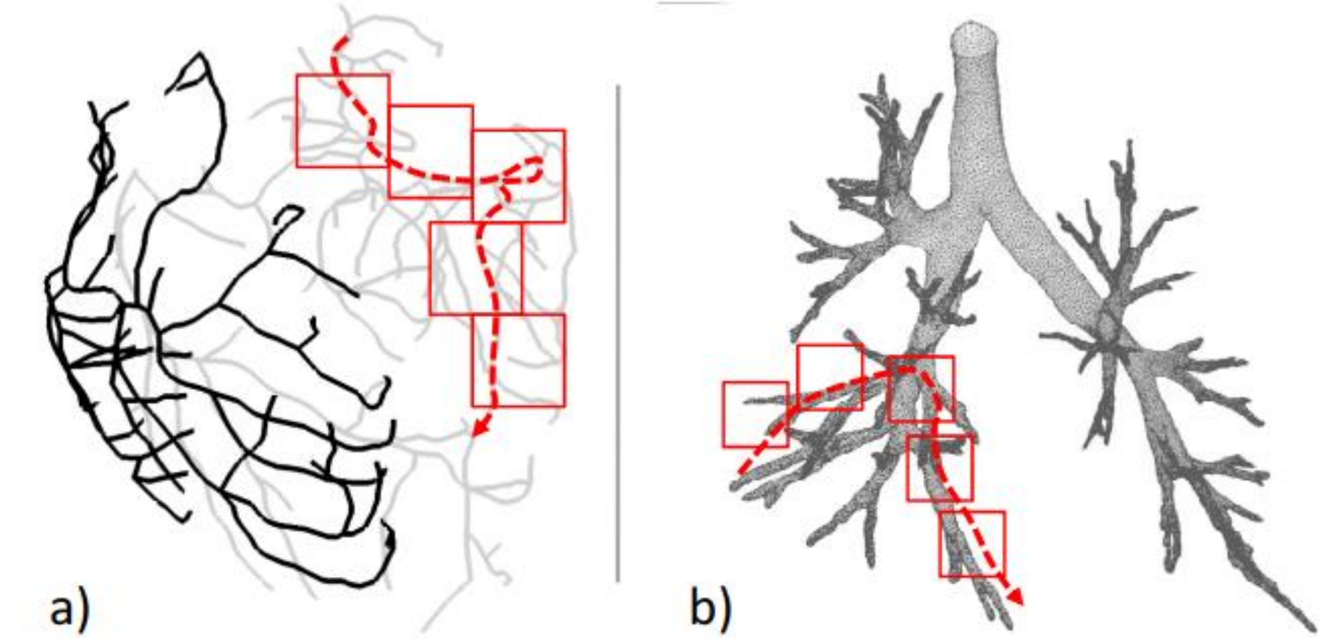


Figure 1: Guided spatial query sequences on a 3D model of (a) a pig's heart arterial tree and (b) a human lung airway track with gaps between the queries in the sequence. The dotted line indicates the guiding structure.

Goal

- To build and analyze spatial models, the scientists use spatial indexes that help them to efficiently retrieve precisely defined parts of the model by executing range and other spatial queries.
- Many such indexes provide fast **random** access to spatial data.
- However, **navigational access** is needed, i.e., moving from one location to another one nearby, based on a certain logic.
- **Goal:** use prefetch to make prediction guided by navigational access, to increase prefetch accuracy and decrease time consumed.

Neuroscience Use Cases

- **Ad-hoc Queries**: These queries are used to correct errors in the models introduced in the imaging process.
- **Model Building**: To place synapses, i.e., the elements connecting the neurons, neuroscientists need to follow some of the branches and detect where their proximity to another branch falls below a given threshold.
- **Walkthrough Visualization**: The 3D walkthrough visualization is primarily used for discovering structural anomalies,

Related Work

- Prefetching along graph structures is also used in different contexts.
- **Web pages prefetching** uses a **page-link graph** to predict the user's next request.
- The following approaches used to prefetch spatial data are categorized as **static methods**, **trajectory extrapolation methods** and **learning approaches**.

Related Work - Static Methods

- Static methods use heuristics for predicting the future query location and do not consider any past information.
- The **Layered** approach segments the spatial data into a grid and prefetches all surrounding grid cells.
- The **Hilbert-Prefetch** works similarly but assigns each cell a Hilbert value and prefetches cells with similar Hilbert values comparing to current location.
- The **multimap approach** works at the disk level and ensures that data close in space is physically stored close together on disk. When reading entire disk pages from the disk, the I/O subsystem automatically prefetches data around current location.

Related Work - Trajectory Extrapolation Methods

- Trajectory extrapolation methods assume that navigational access to spatial data follows a path.
- They use the past query locations, interpolate them with a polynomial and extrapolate the new location.
- The **Straight Line Extrapolation** uses the last two query positions and a simple linear extrapolation.
- The **Polynomial** approach uses several previous query positions as well as a polynomial of degree two to extrapolate the next query location.

Related Work - Trajectory Extrapolation Methods

- The **Velocity** approach additionally uses the user's velocity on the motion path.
- The **EWMA** approach uses exponential weighted moving averages to assign each past movement vector of the query a weight, adds up all vectors and extrapolates the future movement.
- The parameter λ controls the weight assigned to previous queries: the last query is weighted with λ, the second to last with $(1 - \lambda) * \lambda$, etc.

Related Work – Learning Approaches

- User behavior can also be learned and be used to prefetch data because users always chose to explore the spatial data along similar paths or guiding structures.
- The **sequence pattern mining approach** collects past user behavior and mines it to anticipate future user behavior.
- There are lots of other learning approaches based on analysis of past user behavior, however, learning from past user behavior does not significantly improve prediction accuracy.
- Because the models are huge, which contains virtually infinitely many possible paths, reducing the probability that sufficient users take the same paths.

Idea of SCOUT

- Known prefetch approaches for spatial data do not prefetch with high accuracy, because they only rely on previous query ***positions***.
- Idea of SCOUT: instead of using the position of past queries, considers the previous query **content** and identifies the **guiding structure** among the many spatial structures in the range query results.
- Idea of SCOUT: using an **approximate graph representation** of the spatial objects and **traverses** it to predict the location of the next query.
- SCOUT is independent of the complexity or geometry of the dataset.

SCOUT By Step

- Step 1: load the result of query q in the sequence.
- Sept 2: reconstruct the structures in q and build a graph to approximate the structures.
- Step 3: traverse the graph to find the locations where it exits q.
- Step 4: predict next query by linear extrapolation at the exit location.

SCOUT Detail - Guiding Structure

- SCOUT can directly use **explicit** representations of guiding structure information to build a graph.
- SCOUT cannot directly use implicit guiding structures directly and a preprocessing step is required to make the guiding structures explicit.

SCOUT Detail - Graph Representation

- With an explicit guiding structure in the dataset, SCOUT can predict the next query location by summarizing the spatial structures in the query result as a graph representation.

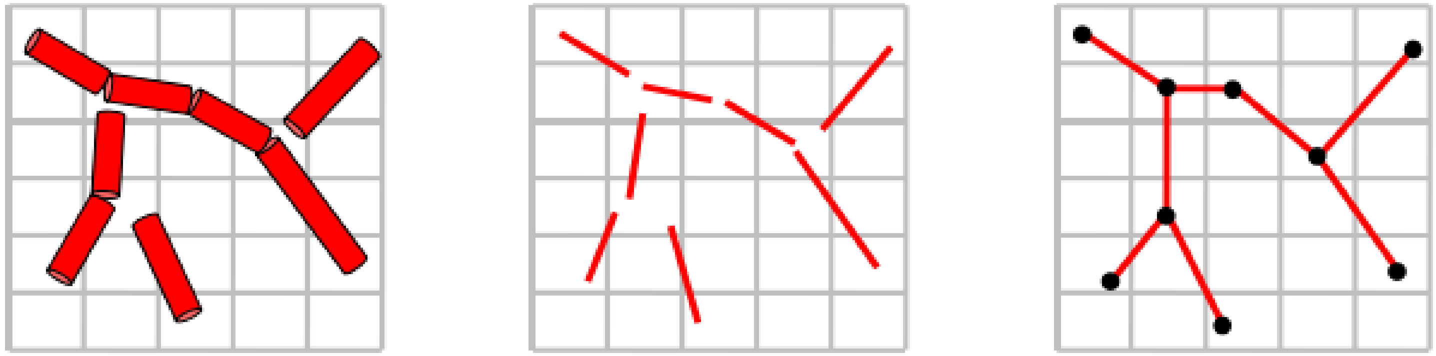


Figure 4: Building the approximate graph by mapping objects to grid cells and connecting them if they occupy the same cell.

SCOUT Detail - Iterative Candidate Pruning

- To identify the structure the user follows, SCOUT exploits the fact that all queries in the guided spatial range query sequence must contain the structure followed.

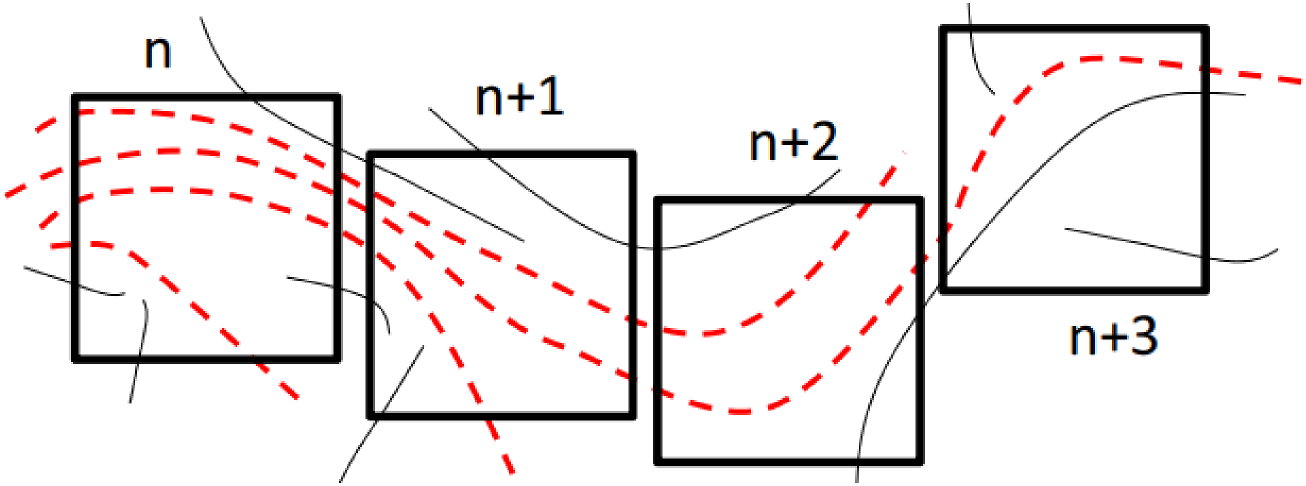


Figure 5: Pruning the irrelevant structures (solid lines) from the candidate set (dashed lines) in subsequent queries (solid squares) of the sequence.

SCOUT Detail - Prefetch Window Duration

- Instead of estimating DW, SCOUT uses an incremental prefetch technique which stops once the user issues the next range query in the sequence.

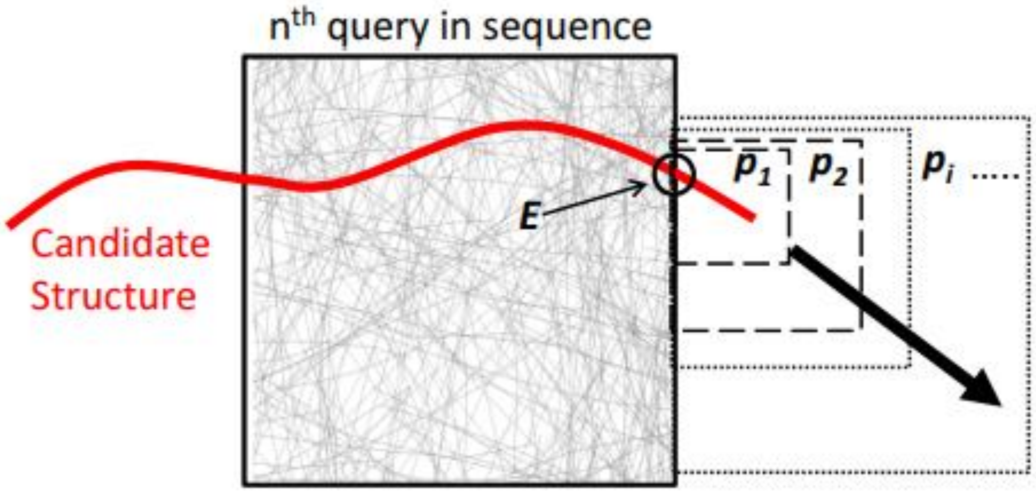


Figure 6: Incremental prefetching queries p_i (dashed lines) are executed along the extrapolated axis starting at E.

SCOUT Detail - Prefetching Strategies

- Deep Prefetching vs Broad Prefetching

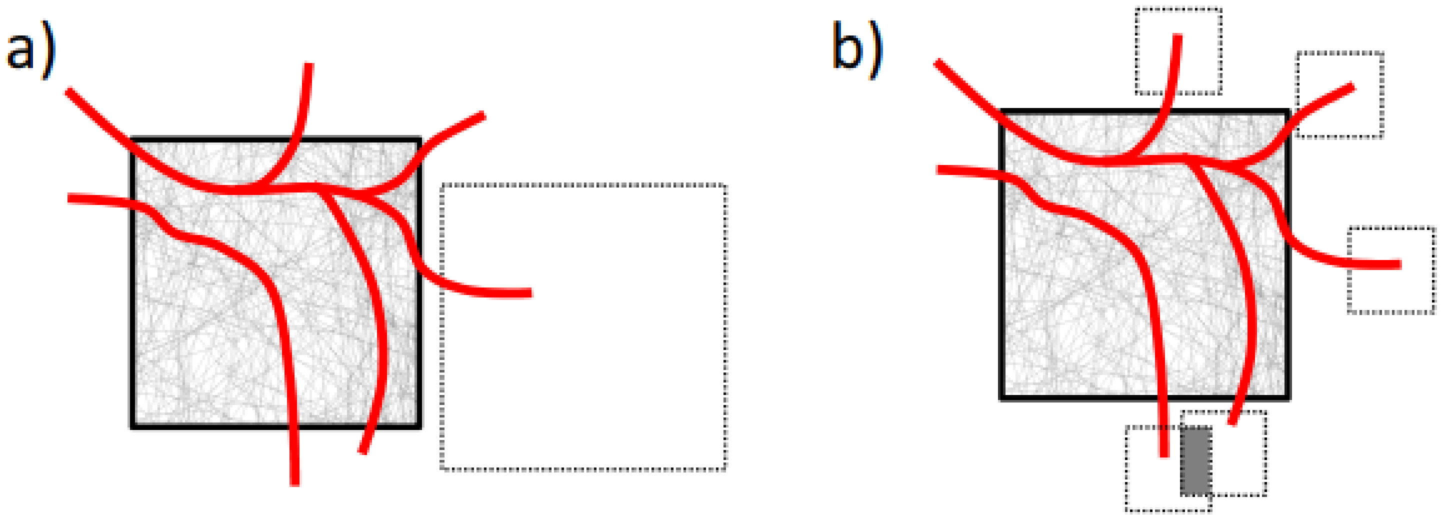


Figure 7: Deep (a) and Broad Prefetching (b).

SCOUT Detail - Sparse Graph Construction

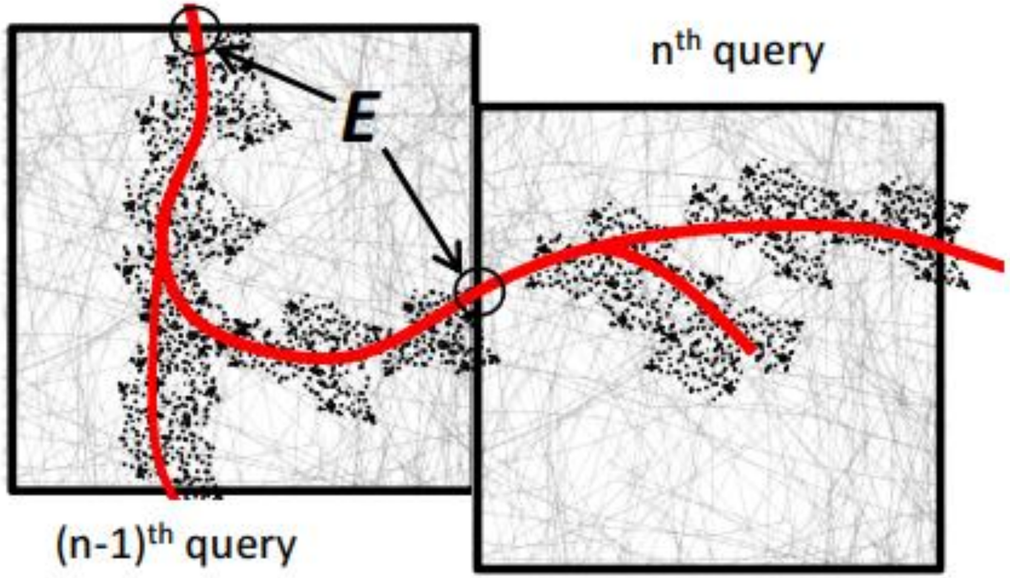


Figure 8: Construction of sparse graph (dotted) using only the relevant pages around the candidate structure (solid curve).

SCOUT Detail - Gap Traversal

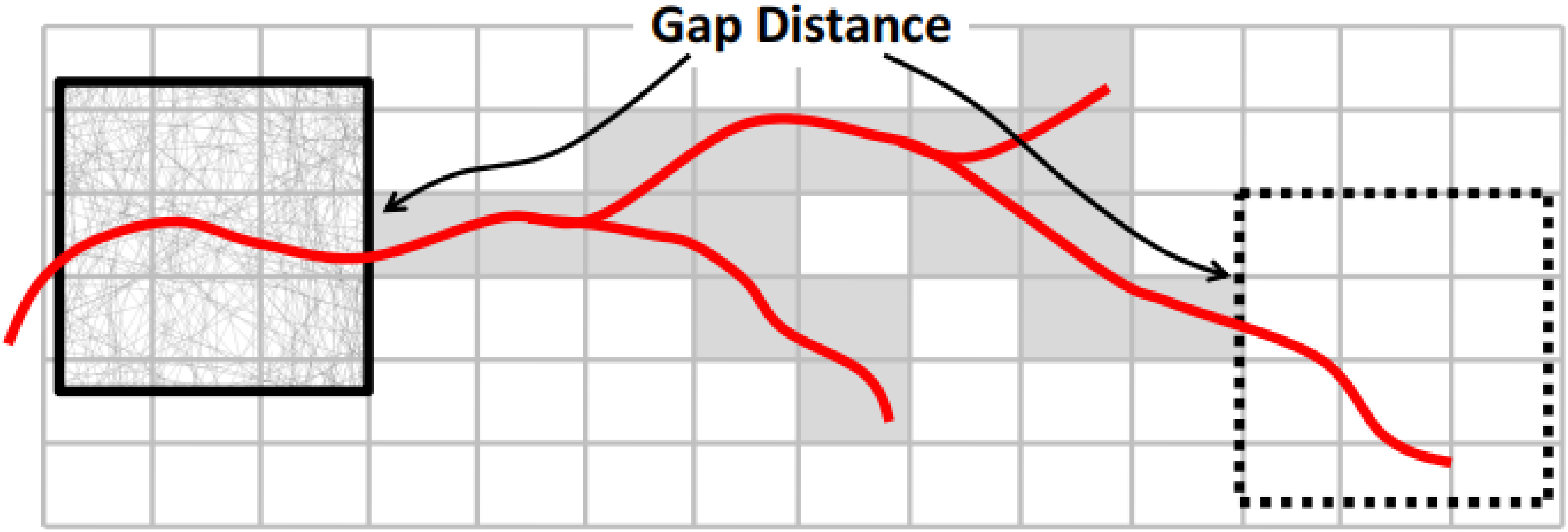


Figure 9: Traversing pages in the gap region using neuron fiber as a guiding structure.

Results

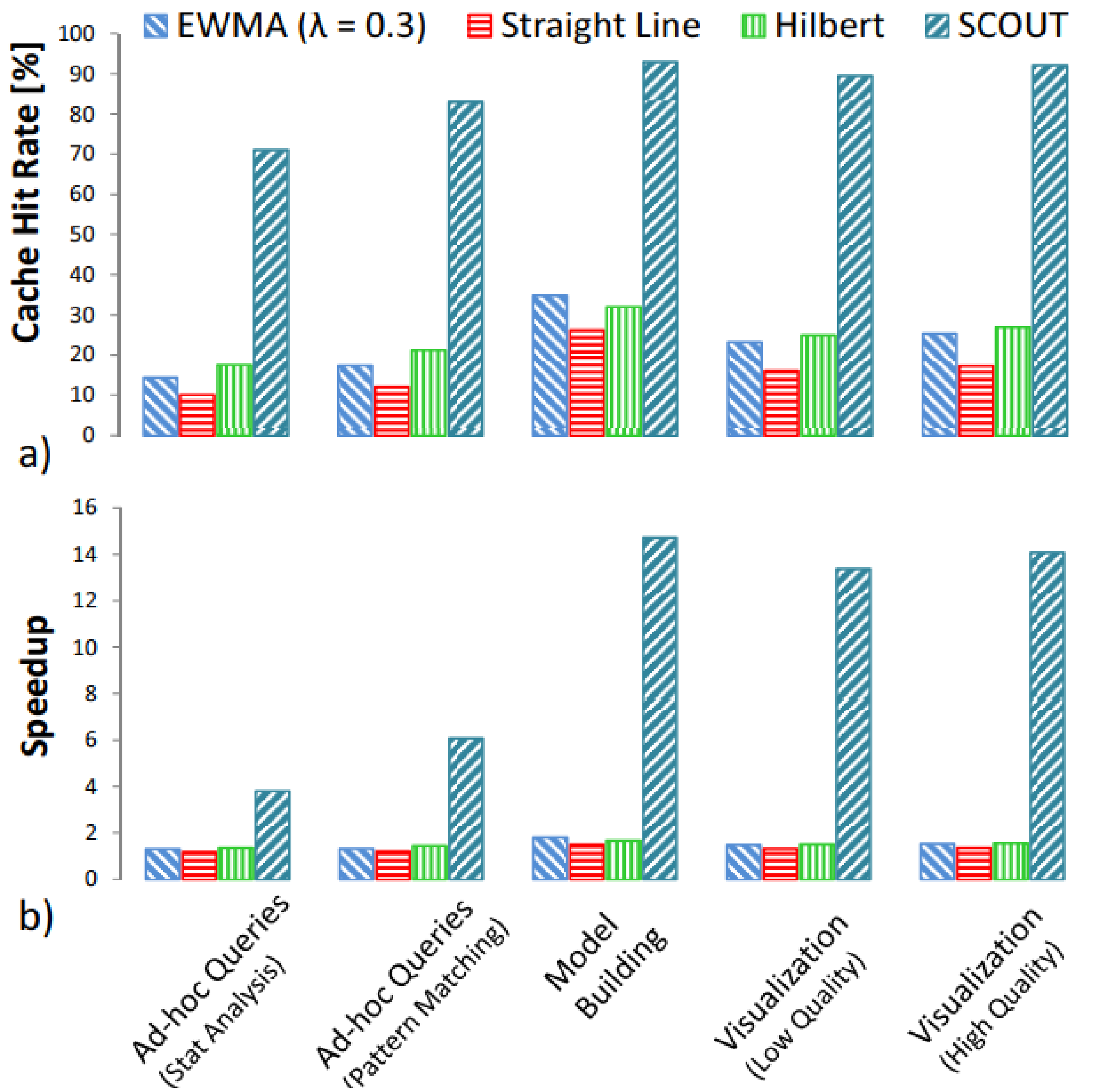


Figure 11: (a) Accuracy of the approaches for all microbenchmarks. (b) Speedup of the approaches for all microbenchmarks (compared to no prefetching).

Results

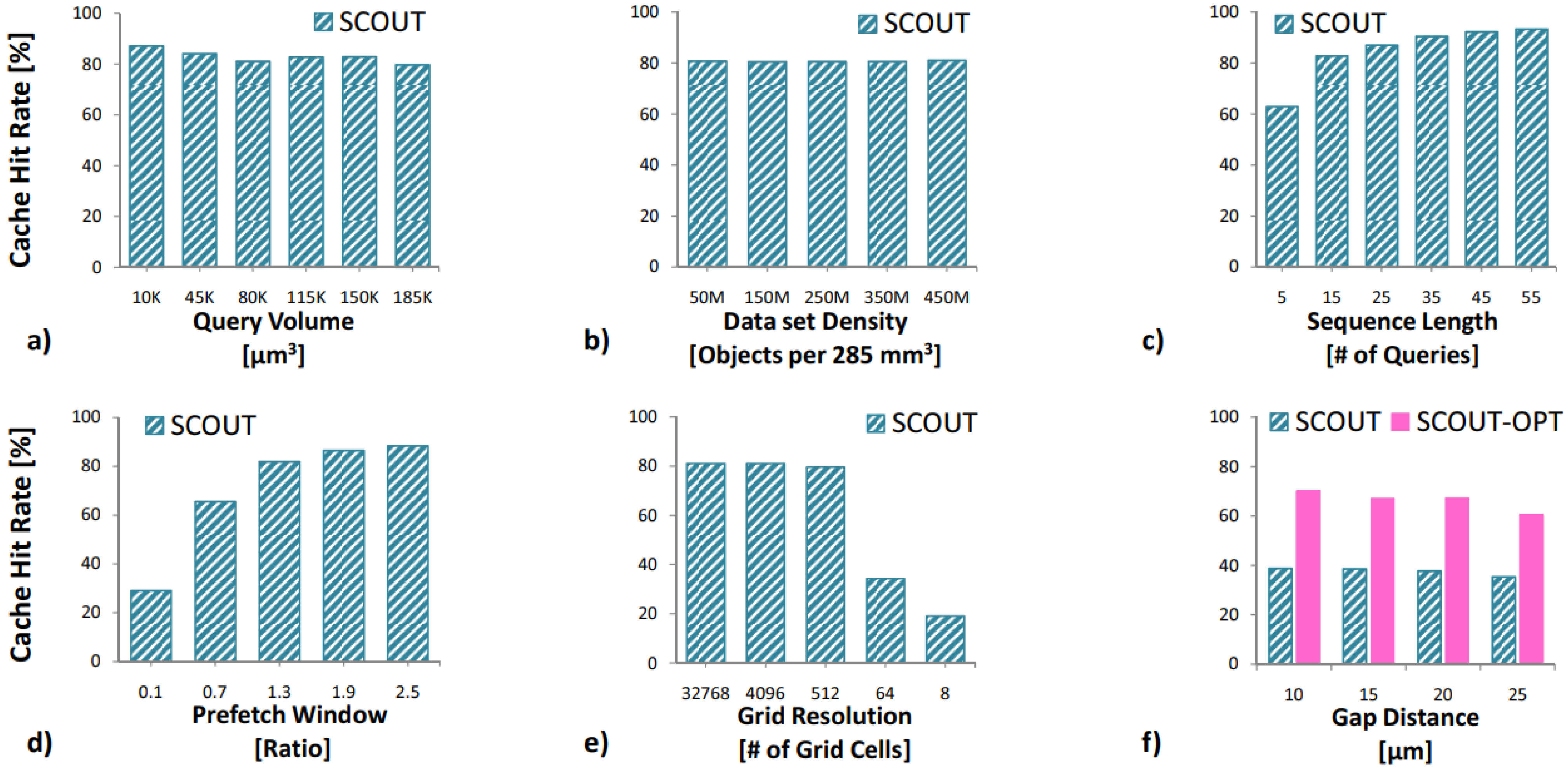
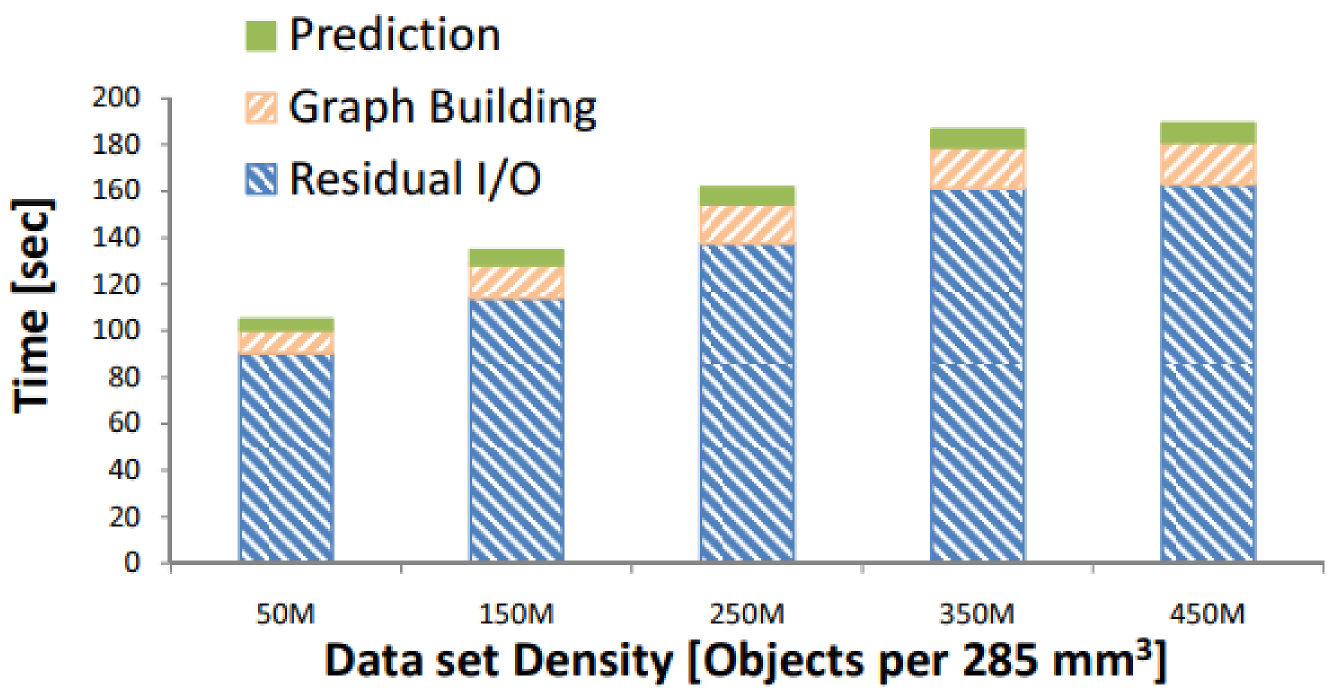


Figure 13: Sensitivity analysis of prediction accuracy.

Results


Prediction
Graph Building
Residual I/O
Time [sec]
200
180
160
140
120
100
80
60
40
20
0
50M
150M
250M
350M
450M
Data set Density [Objects per 285 mm³]

Results

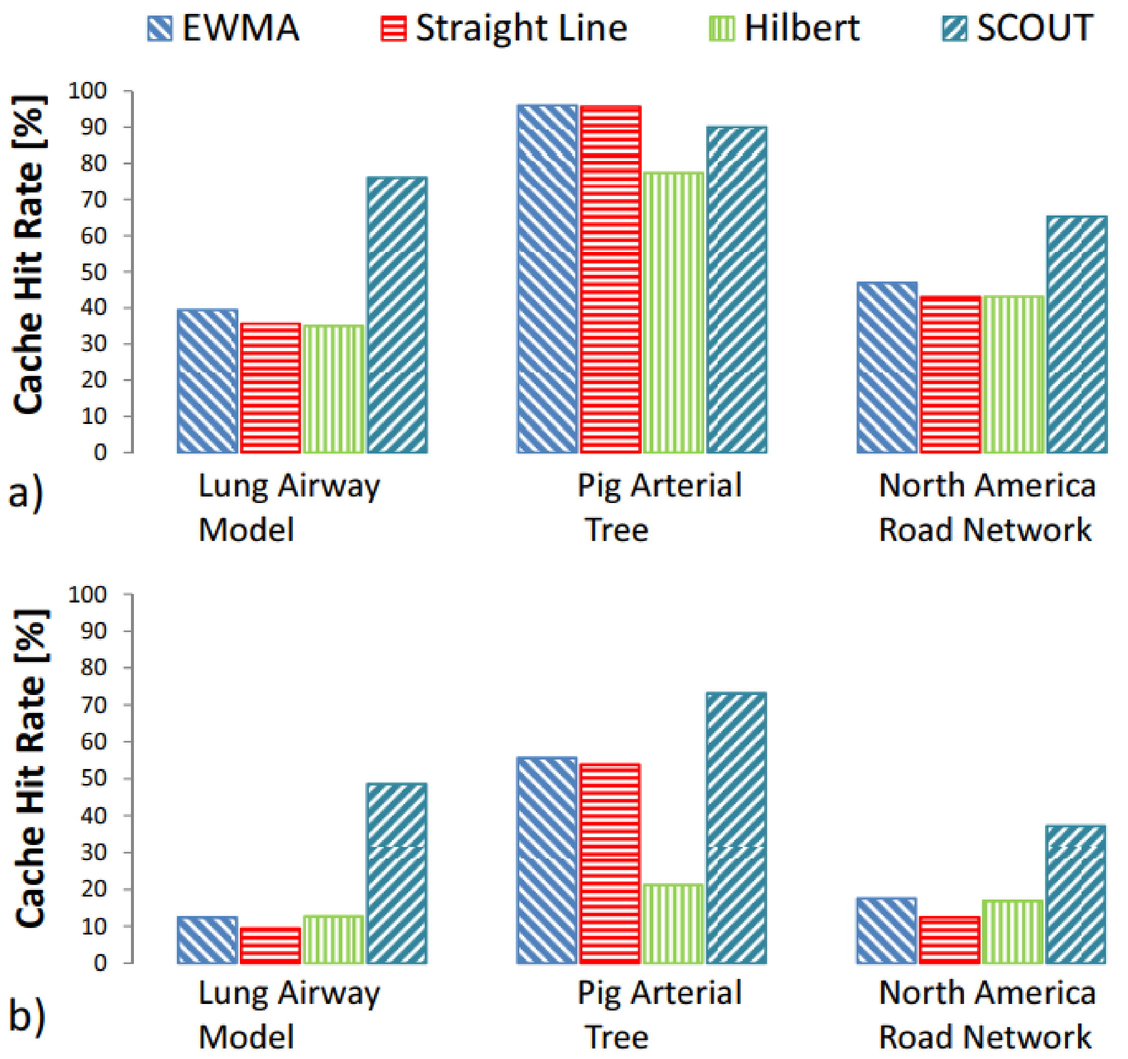


Figure 17: Prediction accuracy comparison for various spatial datasets using (a) small (b) large volume queries.