Projects: Implementation and Experimental Evaluation of Multidimensional Data Structures

Professors: S. Sioutas, G. Vonitsanos (Postdoc Researcher@CEID)

Goal: The major task is the implementation and experimental evaluation of a variety of multi-dimensional data structures in a programming language of your preference (we suggest Python, C,C#,C++ or Java, Go, Scala..e.t.c.). You could use artificial synthetic-data sets or real-data sets to evaluate the performance of the following fundamental operations: Build, Insert, Delete, Update, Searching (Similarity, kNN) Queries.

You can download real datasets from the following URLs:

Find Open Datasets and Machine Learning Projects | Kaggle

20 Free Datasets for Data Science Projects | Built In

[1]https://freegisdata.rtwilson.com/

[1]https://www.census.gov/geographies/mapping-files/time-series/geo/tiger-line-file.html

[1]https://www.kaggle.com/datasets/gauravpendharkar/tiger-detection-dataset

https://paperswithcode.com/datasets?task=trajectory-forecasting

https://scikit-learn.org/stable/tutorial/text_analytics/working_with_text_data.html

***Choose one of the following two (2) projects:**

Project-1: **Multi-dimensional Data Indexing and Similarity Query Processing**: Develop Multidimensional Access Methods based on **k-d trees, quad trees, Range Trees and R-trees** respectively to support ***k-dimensional queries***. Consider the case where k<=5 (indexing on 5 at most attributes). Then, after the 1st phase of indexing, in a 2nd phase, perform ***similarity queries*** according to a specific textual attribute (f.e. review comments, etc) based on **LSH** technique. The final task is an exhausted evaluation performance comparison among the 4 proposed schemes: **k-d + LSH, Quad + LSH, Range + LSH, R-trees + LSH**.

Use-Case: Movies Metadata Cleaned Dataset (1900–2025)

https://www.kaggle.com/datasets/mustafasayed1181/movies-metadata-cleaned-dataset-19002025

A massive dataset with 946K+ movies metadata ready for analysis and ML. This dataset contains a cleaned and structured collection of movie metadata sourced from The Movie Database (TMDB), covering films released between 1900 and 2025. It includes over 946,000 movies with detailed information such as genres, production companies, budgets, revenues, popularity, ratings, and more. This dataset is ideal for data science, analytics, and machine learning projects related to the film industry — including trend analysis, box office prediction, and recommendation systems.

Dataset Columns Description

Column	Description

id	Unique movie identifier
title	Official movie title
adult	Boolean flag indicating adult content
original_language	Original spoken language (ISO 639-1 code)
origin_country	List of production countries
release_date	Movie release date
genre_names	List of genres associated with the movie
production_company_names	Names of involved production companies
budget	Reported production budget (USD)
revenue	Worldwide gross revenue (USD)
runtime	Duration in minutes
popularity	Popularity score (as provided by TMDB)

vote_average	Average user rating
vote_count	Number of votes received

The data in this dataset was collected and preprocessed using the TMDB API. All movie information is © TMDB — provided under their Terms of Use. This dataset is not endorsed or certified by TMDB. Users must comply with TMDB's attribution and API usage policies when using this data.

movies_dataset_cleaned(3 files)**:** It includes the main dataset file, documentation, and an exploratory analysis notebook.

- movies_dataset.csv → The main cleaned dataset (946K+ movie records).
- README.md → Full dataset description, column definitions, and usage notes.
- analysis_notebook.ipynb → Exploratory Data Analysis (EDA) and sample visualizations.

Each file is part of a unified effort to provide a well-structured, ready-to-use dataset for movie analytics, machine learning, and data visualization projects.

Example: Consider the Data Movies Dataset from Kaggle with 14-dimensions (attributes). W*e would like to detect the* ***N-top most similar*** **Production-Company-Names** *or* **Genre-Names** *of Movies* **released** *during* ***2000 up to 2020****, took* **popularity** *from* ***3 up to 6****,* **vote-average** *from* ***3 up to 5*** *and played* **(runtime)** *from* ***30 up to 60 times****, the* **origin-country** *is 'US' or 'GB' and the* **original-language** *is 'En'. Parameter N is a user defined parameter (f.e. N=3).*

Project-2: Develop the following geometric data structures. We suggest the following real data set:

- https://paperswithcode.com/datasets?task=trajectory-forecasting
- https://freegisdata.rtwilson.com/

1. [1]**3D R-trees for Spatio-Temporal Query Processing in a dataset of planar trajectories**: Implement 3-dimensional R-trees to index trajectories of moving object on the plane. Each trajectory is a set of 3-dimensional points of the format (x,y,t), representing the spatial position (x,y) of mobile object at time instant t. Evaluate experimentally the time performance of 3-dimensional range queries. For example, queries that select moving objects passed from a specific spatial terrain during a specific time interval [t1, t2]. f.e. "Find the number of vehicles passed from Olgas' Square with spatial coordinates [x1, x2] x [y1, y2] from 12:00 am up to 14:00 am".

2. [1]**Interval trees** and **Segment trees**. Evaluate the time performance of the basic operations, ***interval and stabbing Queries*** respectively and prove experimentally that time responses follow the theoretical complexity.

3. Consider again the Data Movies Dataset from Kaggle and especially the set of 2-dimensional points ***P1= {(budget, popularity)}***. Compute the **Convex Hull** of this 2-dimensional set **CH(P1)**. For the same 2D dataset and to find out the movies with MIN budget and MAX popularity, develop the SKYLINE OPERATOR that is implemented as a PREDICATE in SQL as follows:

```
SELECT ... FROM ... WHERE ...
GROUP BY ... HAVING ...
SKYLINE OF [DISTINCT] d1 [MIN | MAX | DIFF],…..., dm [MIN | MAX | DIFF]
ORDER BY ...
```

Evaluate the time and space performance of your proposed methods and prove experimentally that the response time follows the theoretical complexities.

4. [1]**Line Segment Intersection:** Develop Line Segment Intersection Algorithms using the basic sweep line technique. Evaluate the time and space performance of your proposed method and prove experimentally that the response time follows the theoretical complexity.

Background Knowledge: Data Structures, Algorithms and Complexity, Databases, Object Oriented Programming (C++, JAVA), Functional Programming (Python, Scala).

References:

1. Book (“advanced data structures”, A.K. Tsakalidis)
2. https://en.wikipedia.org/wiki/Range_tree
3. https://en.wikipedia.org/wiki/K-d_tree
4. https://en.wikipedia.org/wiki/Quadtree
5. https://en.wikipedia.org/wiki/Interval_tree
6. https://en.wikipedia.org/wiki/Segment_tree
7. https://en.wikipedia.org/wiki/Priority_search_tree
8. https://en.wikipedia.org/wiki/Bloom_filter
9. https://en.wikipedia.org/wiki/MinHash
10. https://en.wikipedia.org/wiki/R-tree
11. https://en.wikipedia.org/wiki/Convex_hull
12. https://en.wikipedia.org/wiki/Voronoi_diagram
13. https://en.wikipedia.org/wiki/Sweep_line_algorithm
14. https://en.wikipedia.org/wiki/Line_segment_intersection

(***) Deliverables: Zip or Rar file with executable files. Deadline: ~ 1st WEEK of February, 2025.