The B^{dual} – tree:
Indexing Moving Objects by Space Filling Curves in the Dual Space

By

Man Lung Yiu, Yufei Tao & Nikos Mamoulis
Department of Computer Science
University of Hong Kong
E-mail: mlyiu2@cs.hku.hk, taoyf@cs.cityu.edu.hk, nikos@cs.hku.hk
, nikos@cs.hku.hk

Presented by

Jan Maffert Johansen
Department of Computer Science
Aalborg University
E-mail: jmjo01@cs.aau.dk

The B^{dual} – tree:
Indexing Moving Objects by Space Filling Curves in the Dual Space

- Introduction
- Related work
- Problem definition
- The B^{dual}-tree
  - Structure and updating
  - Query algorithms
- Experiments
- Conclusions

The B^{dual} – tree:

Indexing Moving Objects by Space Filling Curves in the Dual Space

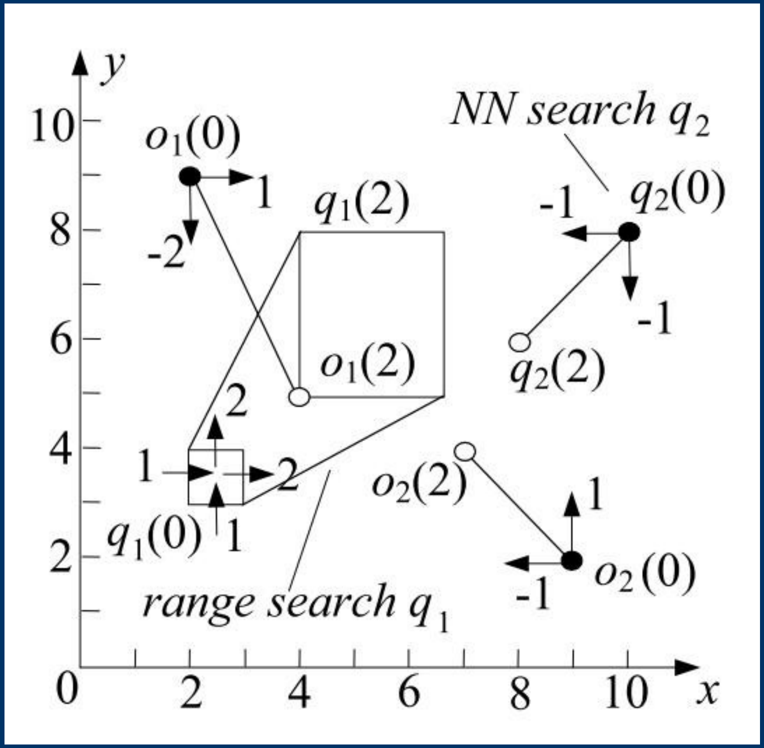
- **Introduction**
- Related work
- Problem definition
- The B^{dual}-tree
  - Structure and updating
  - Query algorithms
- Experiments
- Conclusions

The B^{dual} – tree:
Indexing Moving Objects by Space Filling Curves in the Dual Space

- Introduction:
  - Spatio-temporal objects are spatial objects that change over time.
  - A spatio-temporal database is an index that supports efficient storage and retrieval of information about these objects.
  - Spatio-temporal databases can be divided in two categories:
    - Historical retrieval.
    - Predictive search.

The B^{dual} – tree:
Indexing Moving Objects by Space Filling Curves in the Dual Space

- Introduction:
  - Predictive Search
  - Which objects qualify a predicate in the future.

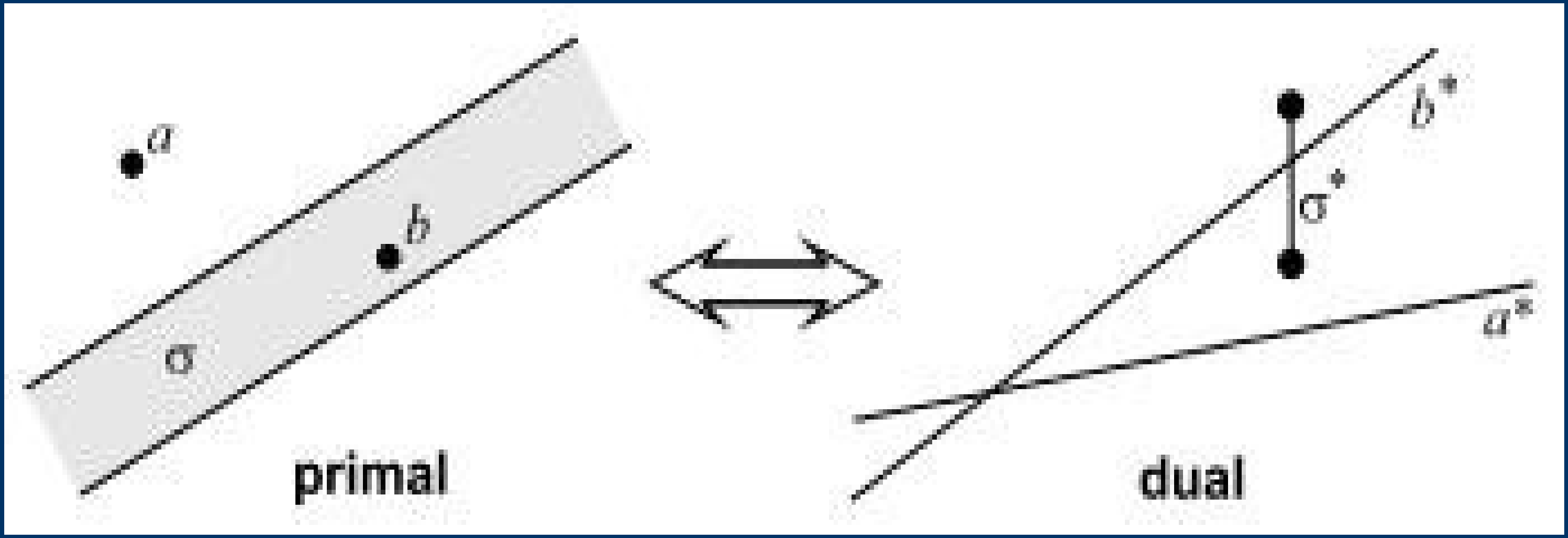


The B^{dual} – tree:
Indexing Moving Objects by Space Filling Curves in the Dual Space

- Introduction
- **Related work**
- Problem definition
- The B^{dual}-tree
  - Structure and updating
  - Query algorithms
- Experiments
- Conclusions

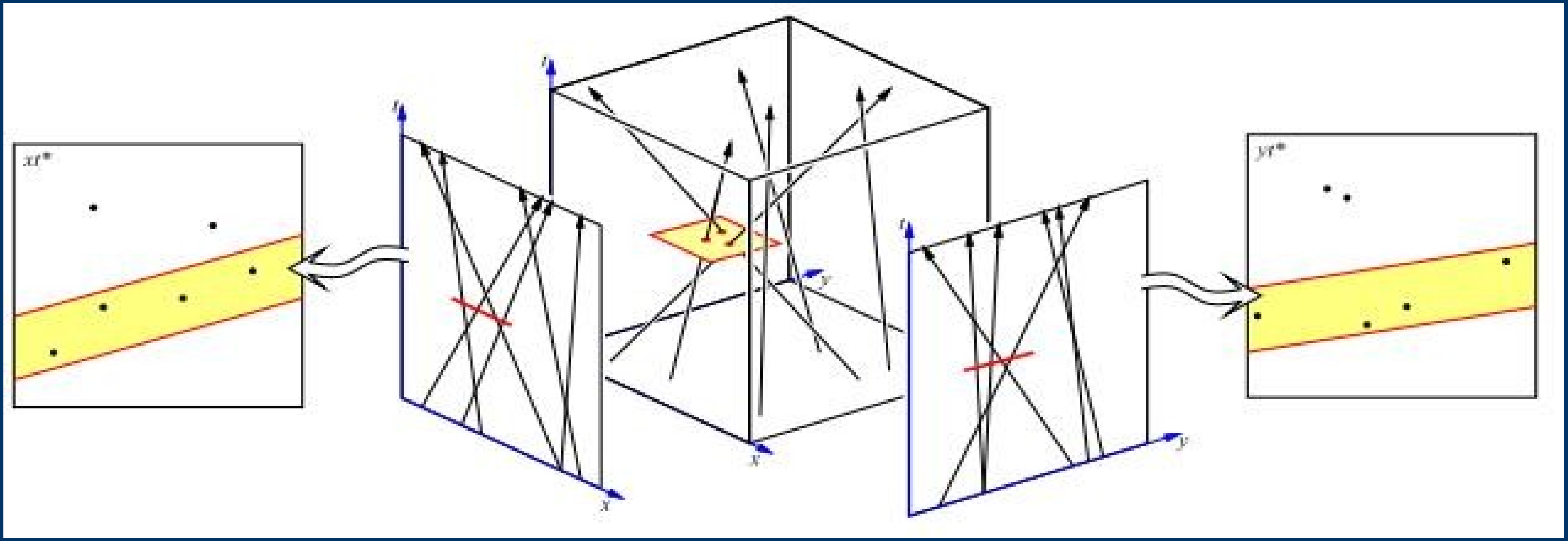
The B^{dual} – tree:
Indexing Moving Objects by Space Filling Curves in the Dual Space

- Related Work: Dual Space Indexing
  - Duality Transformation



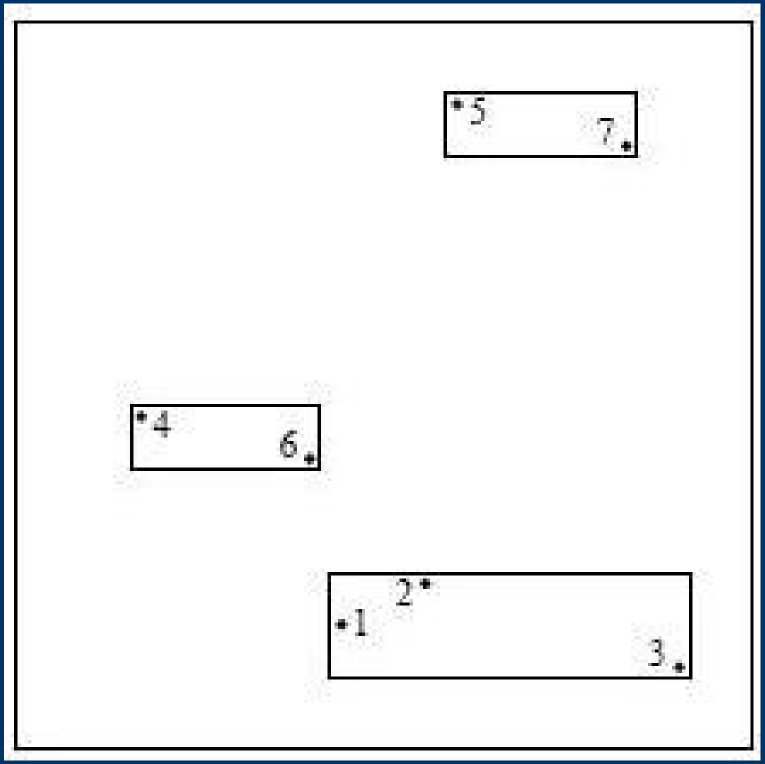
The B^{dual} – tree:
Indexing Moving Objects by Space Filling Curves in the Dual Space

- Related Work: Dual Space Indexing
  - Splits (x,y,t) into (x,x(t)*) and (y,y(t)*)



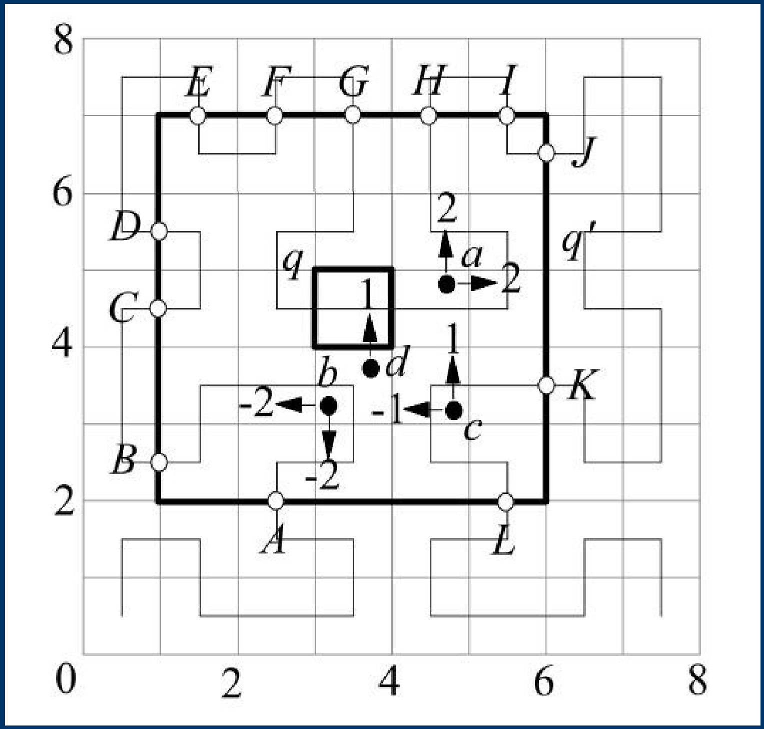
The B^{dual} – tree:
Indexing Moving Objects by Space Filling Curves in the Dual Space

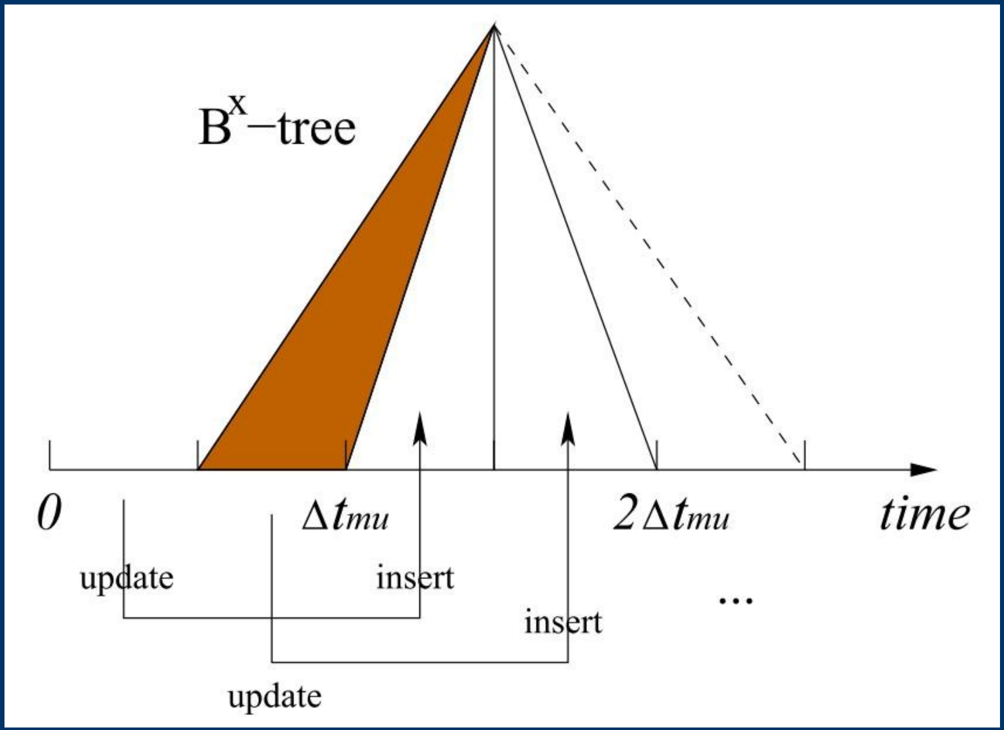
- Related Work: Time-Parameterized R-tree
  - TPR-tree is an R-tree augmented with velocity information to index moving objects.



The B^{dual} – tree:
Indexing Moving Objects by Space Filling Curves in the Dual Space

- Related Work: B^x-tree
  - Uses a space filling curve to index moving objects in B^+-trees





The B^{dual} – tree:

Indexing Moving Objects by Space Filling Curves in the Dual Space

- Introduction
- Related work
- **Problem definition**
- The B^{dual}-tree
  - Structure and updating
  - Query algorithms
- Experiments
- Conclusions

The B^{dual} – tree:
Indexing Moving Objects by Space Filling Curves in the Dual Space

- Problem definition: moving point
  - A *d*-dimensional moving point ***o*** is represented with
    - A reference timestamp $\boldsymbol{o.t_{ref}}$
    - Its coordinates ***o[1], o[2], ..., o[d]*** at time $\boldsymbol{o.t_{ref}}$
    - Its current velocities ***o.v[1], o.v[2], ..., o.v[d]***

The B^{dual} – tree:
Indexing Moving Objects by Space Filling Curves in the Dual Space

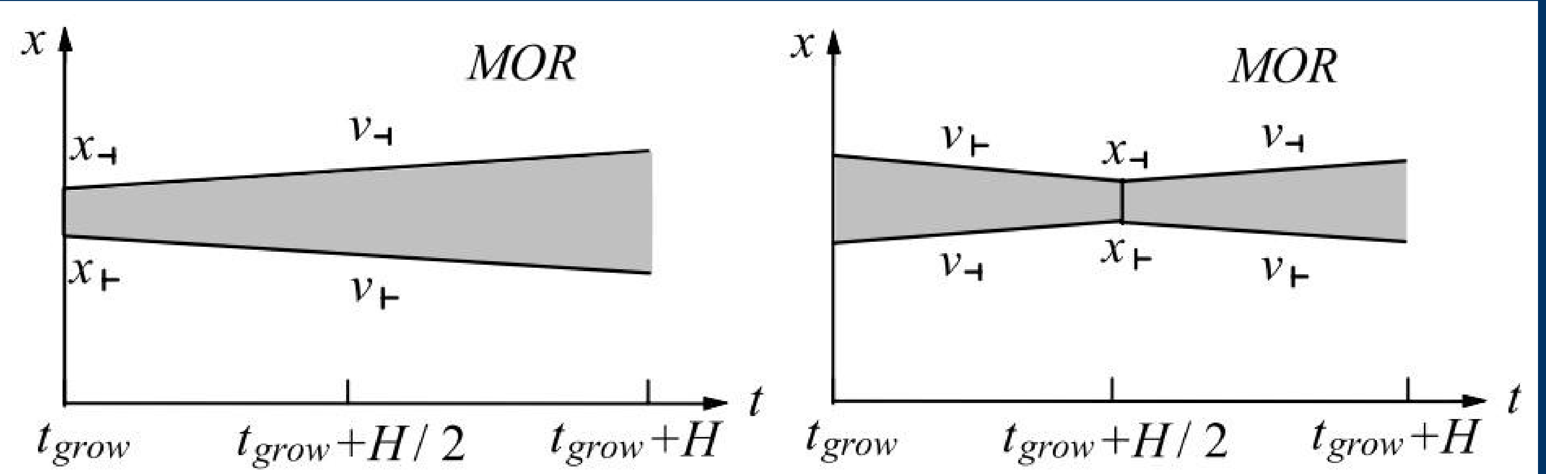
- Problem definition: MOR
  - A *d*-dimensional moving rectangle (MOR) ***r*** is captured by
    - A reference timestamp $\boldsymbol{r.t_{ref}}$
    - A spatial box (SBox), a 2*d*-dimensional vector
    - A velocity box (VBox), a 2*d*-dimensional vector

The B^{dual} – tree:
Indexing Moving Objects by Space Filling Curves in the Dual Space

- Motivation
- Related work
- Problem definition
- **The B^{dual}-tree**
  - Structure and updating
  - Query algorithms
- Experiments
- Conclusions

The B^{dual} – tree:
Indexing Moving Objects by Space Filling Curves in the Dual Space

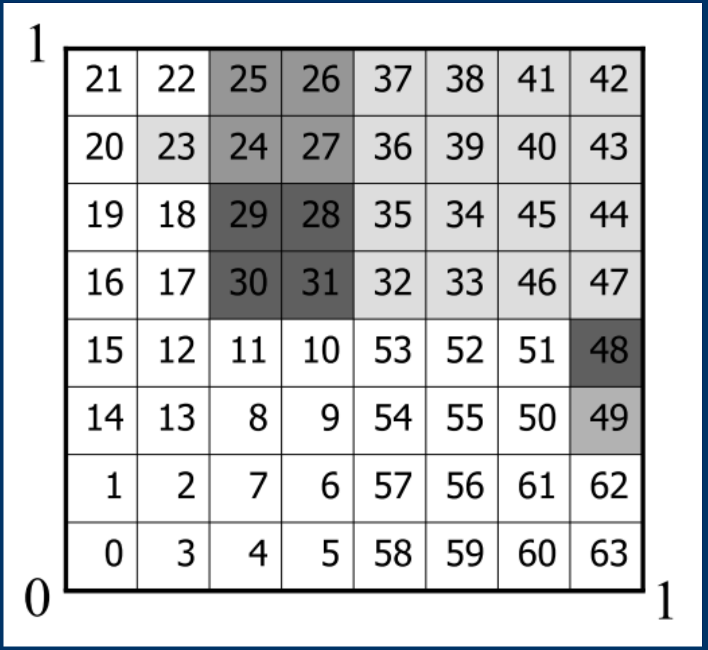
- The B^{dual}-tree: Structure and updating
  - Two parameters:
    - Horizon ***H***, the farthest future time that can be efficiently queried.
    - Reference time $\boldsymbol{T_{ref}}$, the reference time for duality transformation.



(a) $T_{ref} = t_{grow}$ (b) $T_{ref} = t_{grow} + H/2$

The B^{dual} – tree:
Indexing Moving Objects by Space Filling Curves in the Dual Space

- The B^{dual}-tree: Structure and updating
  - Duality Transformation
    - *2d*-dimensions: for every dimension ***i*** in the original data there is a pair ***(o[i](T$_{ref}$) , o.v[i])***
  - Mapping to 1D
    - Hilbert Space filling curve



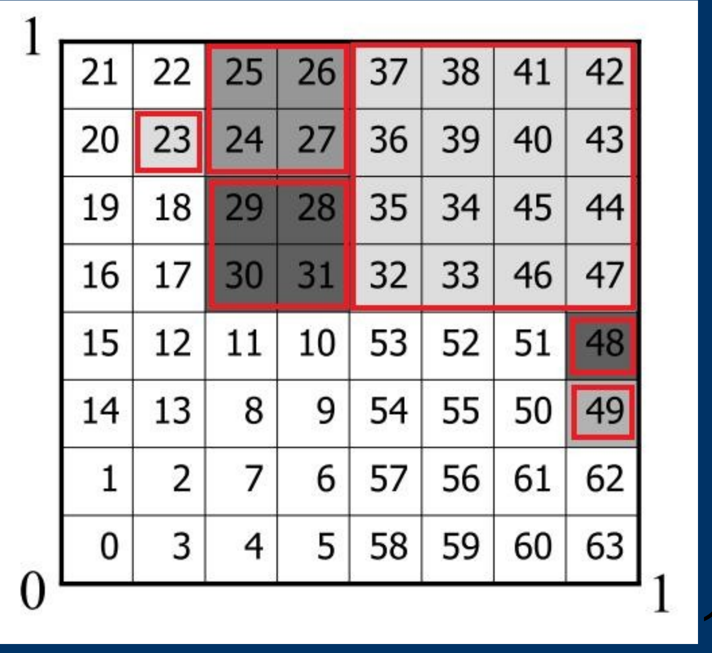
The B^{dual} – tree:

Indexing Moving Objects by Space Filling Curves in the Dual Space

- The B^{dual}-tree: Structure and updating
  - The B^{dual}-tree consists of two identical B^{+}-trees ***BT_1*** and ***BT_2***
  - When one of the trees is in a *growing state*, the other is in a *shrinking state*
  - In the *growing state,* objects can both be inserted and deleted from the tree
  - In the *shrinking state* objects can only be deleted from the tree

The B^{dual} – tree:
Indexing Moving Objects by Space Filling Curves in the Dual Space

- The B^{dual}-tree: Query Algorithms
  - Perfect MORs
    - Created by *2d*-dimensional square of cells
    - The cells have continuous values
    - Query of range [23, 49]
    - Breaks down into 6 perfect MORs
      [23, 23]
      [24, 27]
      [28, 31]
      [32, 47]
      [48, 48]
      [49, 49]



The B^{dual} – tree:
Indexing Moving Objects by Space Filling Curves in the Dual Space

- The B^{dual}-tree: Query Algorithms
  - The perfect MORs translate into subtrees in the B^{dual}-tree
  - Each childnode of a subtree is checked to see if it intersects the query range
    - If not it is pruned from the result
    - Else its children are checked for intersection with the query area
  - In this manner the tree is recursively checked until the leafs are encountered

The B^{dual} – tree:
Indexing Moving Objects by Space Filling Curves in the Dual Space

- Motivation
- Related work
- Problem definition
- The B^{dual}-tree
  - Structure and updating
  - Query algorithms
- **Experiments**
- Conclusions

The B^{dual} – tree:
Indexing Moving Objects by Space Filling Curves in the Dual Space

- Experiments:
  - Experimental data space is two-dimensional
  - Each dimension has a domain of [0, 1000]
  - Objects simulate aircraft travelling between airports.
  - Time horizon of 50 time units.
  - Query cost is measured by averaging over a workload of 100 queries, issued at different current times as the index runs.

The B^{dual} – tree:
Indexing Moving Objects by Space Filling Curves in the Dual Space

- Experiments:
  - 31 experiments are performed in the categories of Space requirements, Update performance and Query performance
  - In all experiments B^{dual}-tree performed significantly better than the worst index and in most cases only slightly worse than the best index
  - Conclusion of the article is that the B^{dual}-tree is superior to teh other indexes

The B^{dual} – tree:

Indexing Moving Objects by Space Filling Curves in the Dual Space

- Motivation
- Related work
- Problem definition
- The B^{dual}-tree
  - Structure and updating
  - Query algorithms
- Experiments
- **Conclusions**

The B^{dual} – tree:
Indexing Moving Objects by Space Filling Curves in the Dual Space

- Conclusions: Our Project
  - Deals with indexing of moving objects
  - Duality transformation.
  - Was implemented on disk, whereas our project focuses on main memory.
  - Might be implemented for performance studies
  - Overall conclusion: Highly relevant to our project

The B^{dual} – tree:
Indexing Moving Objects by Space Filling Curves in the Dual Space

- Conclusions: Pros
  - Good illustrations in the beginning of the article showing the problem definition and related work
  - Good introduction, motivation and related work sections.
  - Experimental evidence of performance.

The B^{dual} – tree:

Indexing Moving Objects by Space Filling Curves in the Dual Space

- Conclusions: Cons
  - Confusing shorthand writing style in article:

    $$x(y) = 2(3)$$
  - Alot of inline math
  - Information is very compressed.
  - Lack of illustration in the part of the article describing the B^{dual}-tree itself.