R-trees:
An Average Case Analysis

R-trees - performance analysis

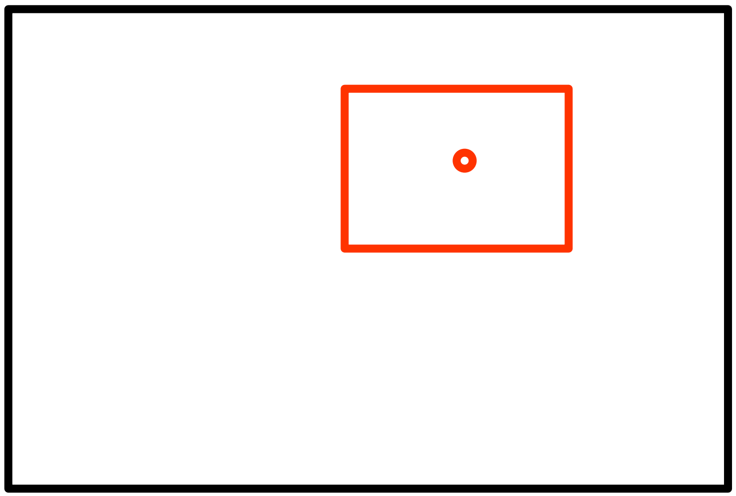
- How many disk (=node) accesses we’ll need for
  - range
  - nn
  - spatial joins
- why does it matter?

R-trees - performance analysis

- A: because we can design insert, split, delete algorithms accordingly
- B: we can do query-optimization
- motivating question: on, e.g., split, should we try to minimize the area (volume)? the perimeter? the overlap? or a weighted combination? why?

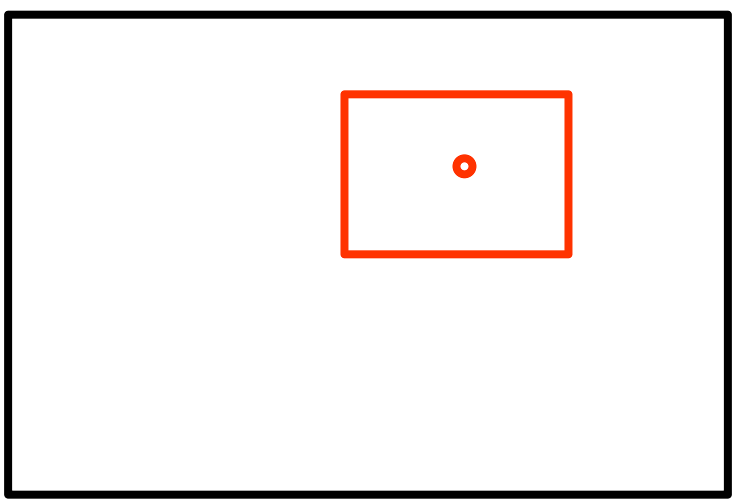
R-trees - performance analysis

- How many disk accesses (expected value) for range queries?
  - query distribution wrt location?
  - “ “ wrt size?



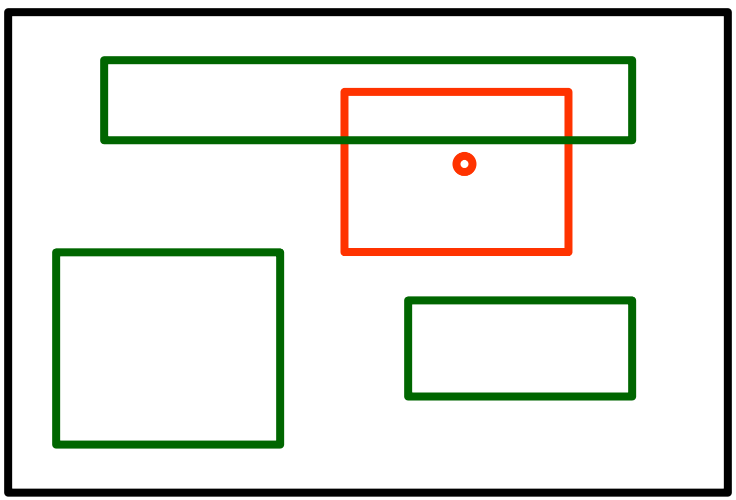
R-trees - performance analysis

- How many disk accesses for range queries?
  - query distribution wrt location? uniform; (biased)
  - “ “ wrt size? uniform



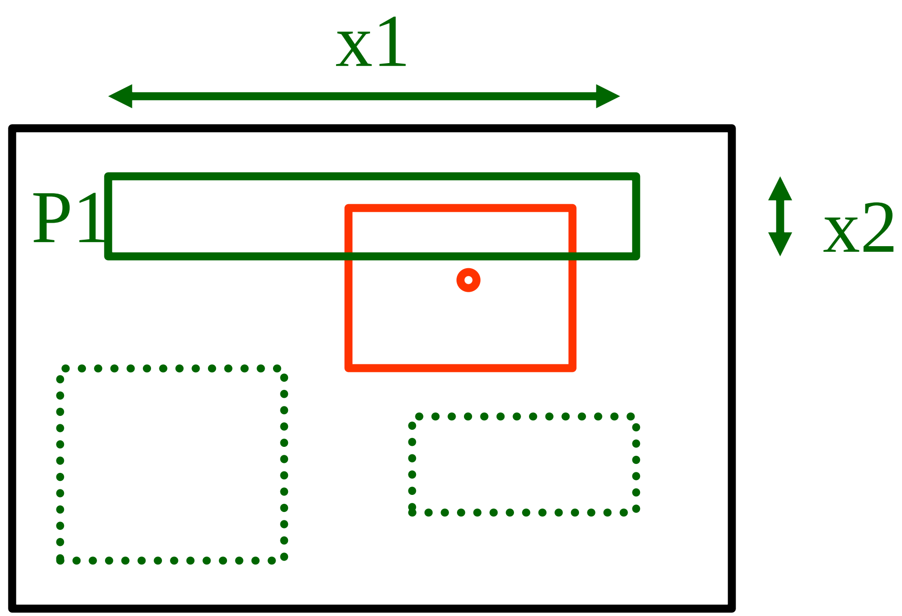
R-trees - performance analysis

- easier case: we know the positions of data nodes and their MBRs, eg:



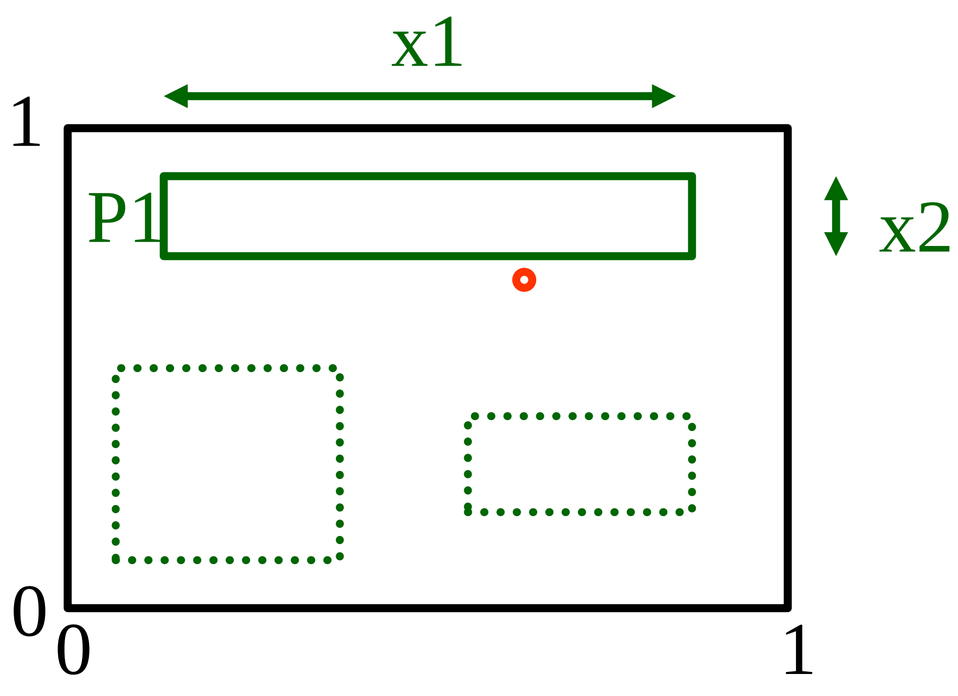
R-trees - performance analysis

- How many times will P1 be retrieved (unif. queries)?



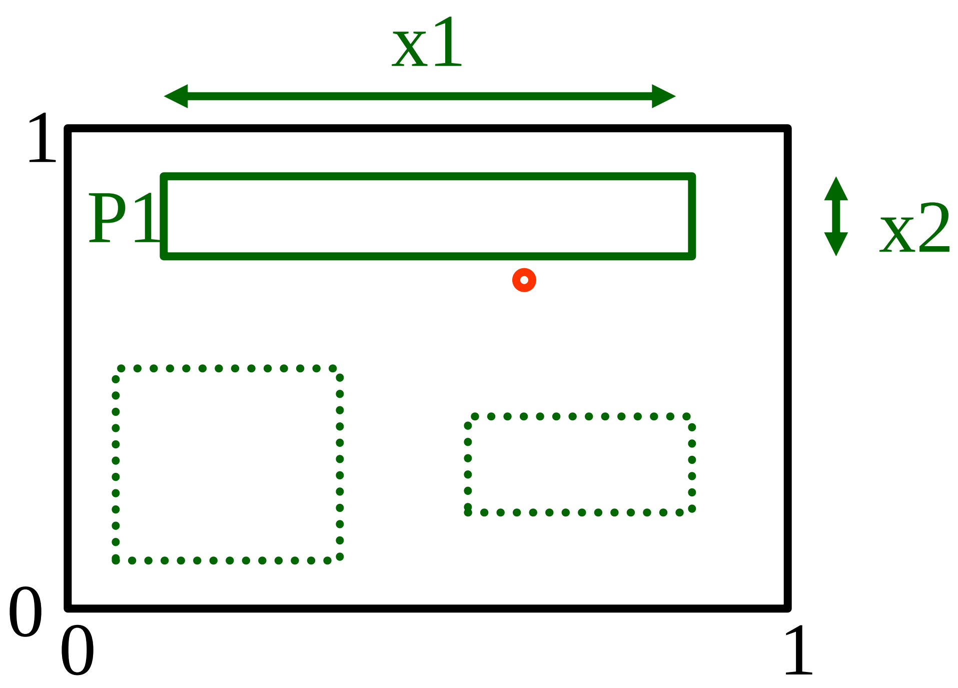
R-trees - performance analysis

- How many times will P1 be retrieved (unif. POINT queries)?



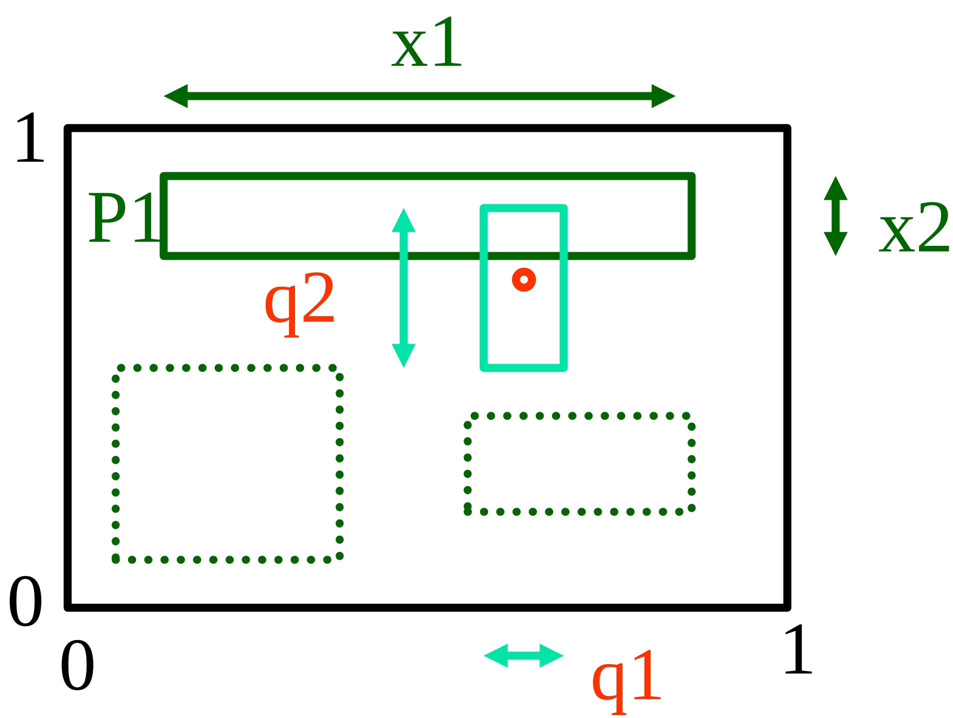
R-trees - performance analysis

- How many times will P1 be retrieved (unif. POINT queries)? A: x1*x2



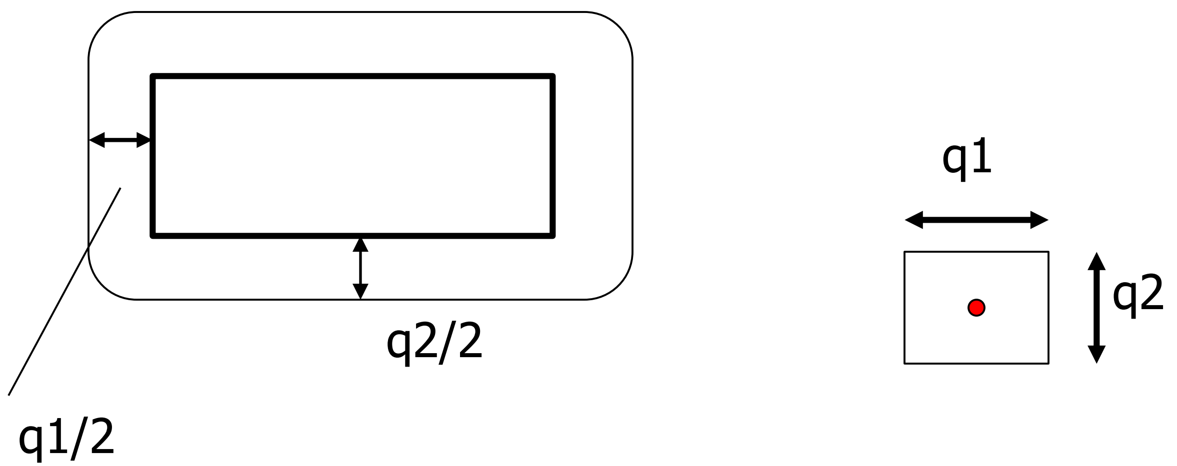
R-trees - performance analysis

- How many times will P1 be retrieved (unif. queries of size q1xq2)?



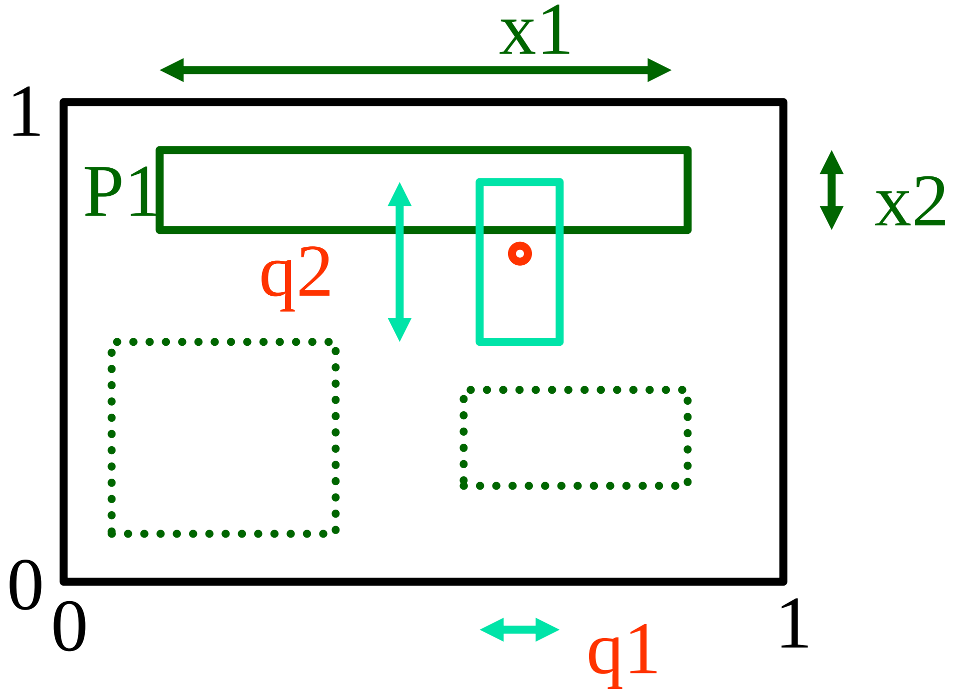
R-trees - performance analysis

- Minkowski sum



R-trees - performance analysis

- How many times will P1 be retrieved (unif. queries of size q1xq2)? A: (x1+q1)*(x2+q2)



R-trees - performance analysis

- Thus, given a tree with n nodes (i=1, ... n) we expect

$$DA(q_1, q_2) = \sum_i^n (x_{i,1} + q_1)(x_{i,2} + q_2)$$

$$= \sum_i^n x_{i,1} * x_{i,2} +$$

$$q_1 \sum_i^n \; x_{i,2} + q_2 \sum_i^n \; x_{i,1}$$

$$+ q_1 * q_2 * n$$

R-trees - performance analysis

- Thus, given a tree with n nodes (i=1, ... n) we expect

$$DA(q_1, q_2) = \sum_i^n (x_{i,1} + q_1)(x_{i,2} + q_2)$$

$$= \sum_i^n x_{i,1} * x_{i,2} +$$ ⟶ 'volume'

$$q_1 \sum_i^n x_{i,2} + q_2 \sum_i^n x_{i,1}$$ ⟶ 'surface area'

$$+ q_1 * q_2 * n$$ ⟶ count

R-trees - performance analysis

Observations:

- for point queries: only volume matters
- for horizontal-line queries: (q2=0): vertical length matters
- for large queries (q1, q2 >> 0): the count N matters
- formula: easily extendible to n dimensions

R-trees - performance analysis

Conclusions:

- splits should try to minimize area and perimeter
- ie., we want few, small, square-like parent MBRs

--- similar to R*-tree optimizations

More general Model

- What if we have only the dataset $\mathcal{D}$ and the set of query distribution $\mathcal{S}$?
- We should “predict” the structures of a “good” R-tree for this dataset. Then, use the previous model to estimate the average query performance for S

Uniform dataset

- Assume that the dataset (that contains only rectangles) is uniformly distributed in space.
- **Density** of a set of N MBRs is the average number of MBRs that contain a given point in space. OR the total area covered by the MBRs over the area of the work space.
- N boxes with average size **s**= (s_1,s_2), D(N,**s**) = N s_1 s_2
- If s_1=s_2=s, then:

(one more assumption here: squares)

$$D = N\ s^2 \Rightarrow s = \sqrt{\frac{D}{N}}$$

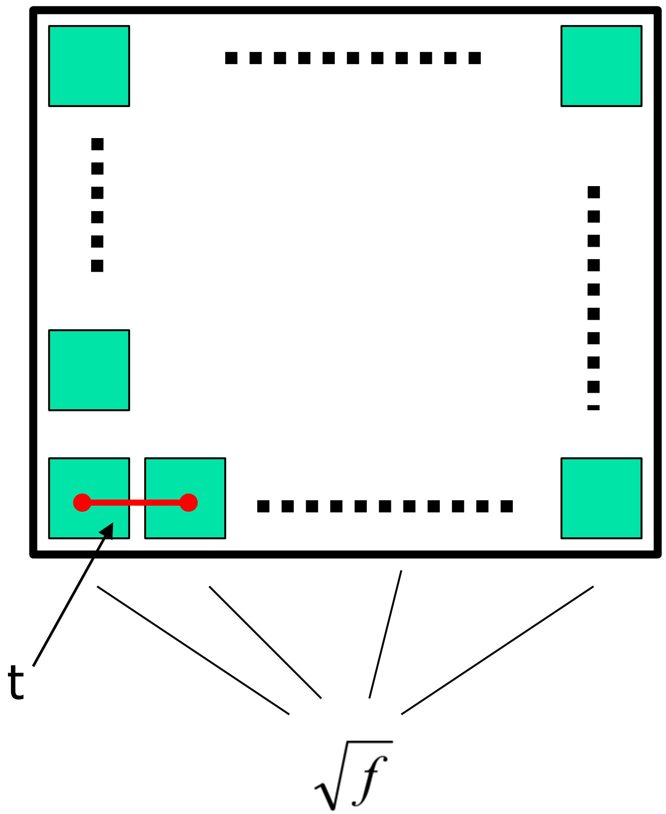
Density of Leaf nodes

- Assume a dataset of N rectangles. If the average page capacity is f, then we have $N_{ln} = N/f$ leaf nodes.
- If D_1 is the density of the leaf MBRs, and the average area of each leaf MBR is s_1^2, then:

$$D_1 = \frac{N}{f} s_1^{\,2} \Rightarrow s_1 = \sqrt{D_1 \frac{f}{N}}$$

- So, we can estimate s_1, from N, f, D_1
- We need to estimate D_1 from the dataset's density…

Estimating D_1



Consider a leaf node that contains f MBRs.

(if MBR square)

Then for each side of the leaf node MBR we have:

$\sqrt{f}$ MBRs

Also, N_{ln} leaf nodes contain N MBRs, uniformly distributed.

The average distance between the centers of two consecutive MBRs is $t = \frac{1}{\sqrt{N}}$ (assuming $[0,1]^2$ space)

Estimating D_1

- Combining the previous observations we can estimate the density at the leaf level, from the density of the dataset:

$$D_1 = \{1 + \frac{\sqrt{D} - 1}{\sqrt{f}}\}^2$$

- We can apply the same ideas recursively to the other levels of the tree.

R-trees–performance analysis

- Assuming Uniform distribution:

$$DA(q) = 1 + \sum_{j=1}^{1+h} \{(\sqrt{D_j} + q\sqrt{\frac{N}{f^j}})^2\}$$

where $D_j = \{1 + \frac{\sqrt{D_{j-1}} - 1}{\sqrt{f}}\}^2 \quad and \quad D_0 = D$

And D is the density of the dataset, f the fanout [TS96], N the number of objects

References

- Christos Faloutsos and Ibrahim Kamel. “Beyond Uniformity and Independence: Analysis of R-trees Using the Concept of Fractal Dimension”. Proc. ACM PODS, 1994.
- Yannis Theodoridis and Timos Sellis. “A Model for the Prediction of R-tree Performance”. Proc. ACM PODS, 1996.