アルゴリズム2　発表

R-Tree

Team B

臼井 優,　岡本 悠吾,　矢野 健斗,　Faadhil as

R-Treeと線形探索の実行速度比較

臼井 優,　岡本 悠吾,　矢野 健斗

2023年11月24日

使用したデータ　（臼井）

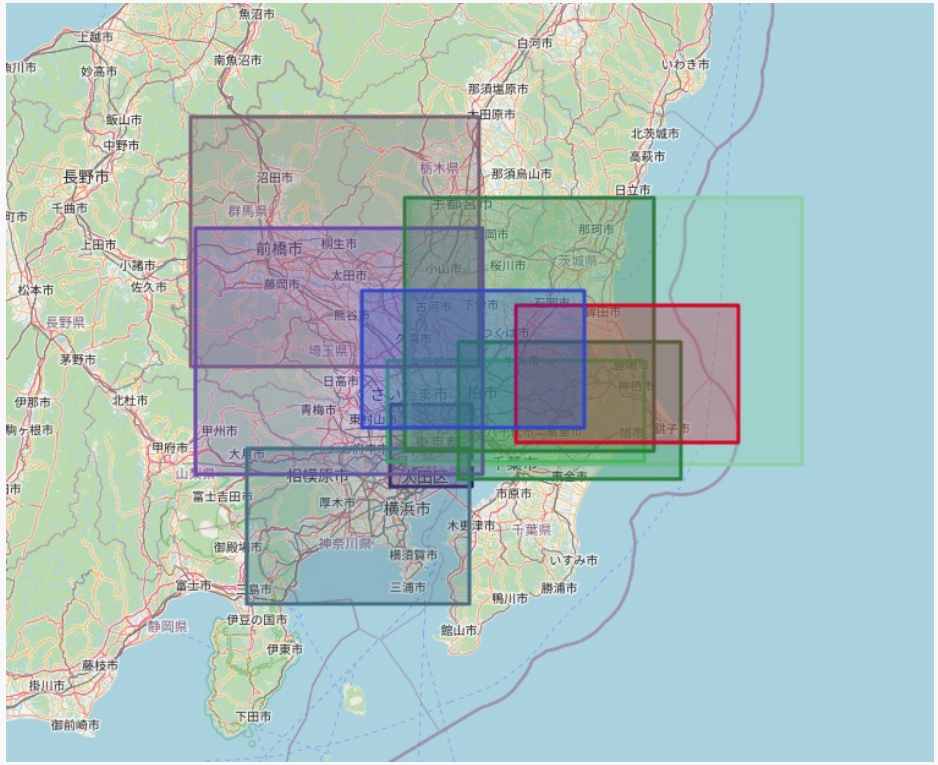
- G空間情報センターの国土数値情報（福祉施設）
- GeoJSONファイル形式
- 関東（東京都，神奈川県，群馬県，埼玉県，千葉県，茨城県，栃木県）のデータを集約して使用
- データ数は42771
- データの詳細

	都道府県名	市区町村名	所在地	公共施設大分類	公共施設小分類	福祉施設細分類	名称	管理者	定員	原典資料名	主題属性取得資料名	geometry
0	東京都	あきる野市	伊奈477-1	19	19012	1779	介護老人保健施設オキドキ	0	51	地理院地図	東京都社会福祉施設一覧	POINT (139.25512 35.73315)
1	東京都	あきる野市	引田388	16	16011	804	秋川文化幼稚園	4	260	地理院地図	都道府県等が公開する幼稚園に関する情報	POINT (139.26828 35.72659)
2	東京都	あきる野市	引田928	19	19008	442	一の谷児童館	3	-1	地理院地図	東京都社会福祉施設一覧	POINT (139.27099 35.72009)
3	東京都	あきる野市	雨間1067-6	19	19013	801	よつぎ第一保育園	9	157	地理院地図	東京都社会福祉施設一覧	POINT (139.29814 35.72025)
4	東京都	あきる野市	雨間1815	19	19007	556	あきる野福祉工房	0	-1	地理院地図	東京都社会福祉施設一覧	POINT (139.29661 35.71605)

比較の方法　（臼井）

・10個のバウンディボックスを設定

・そのバウンディボックス内に存在する施設の数をカウントするという探索実施し実行時間を比較




```python
import folium
import random
from typing import List

def generate_random_color() -> str:
    # ランダムなRGB色を生成
    return "#{:06x}".format(random.randint(0, 0xFFFFFF))

def plot_kanto_subareas(search_areas: List[tuple]) -> None:
    """
    関東地方のバウンディボックスを地図上に描画する。

    Parameters:
        search_areas (List[Tuple[float, float, float, float]]): 探索範囲の情報がリストになっている。

    Returns:
        None
    """
    # マップの初期設定
    m = folium.Map(location=[35.9, 139.6], zoom_start=10)

    # 各サブエリアのバウンディングボックスを描画
    # lon:経度, lat:緯度
    for i, (min_lon, min_lat, max_lon, max_lat) in enumerate(search_areas):
        color = generate_random_color()
        bounding_box = [[min_lat, min_lon], [max_lat, max_lon]]
        folium.Rectangle(bounding_box, color=color,
                         fill=True,
                         fill_color=color,
                         fill_opacity=0.3,
                         # popup=f"Subarea {i+1}"
                        ).add_to(m)

    # 地図を表示
    m.save('kanto_map.html')
```

左の地図の描画に使ったコード

線形探索とR-Treeによる探索の実装　（臼井）

・R-Treeの構築＆線形探索の関数定義

```
# 探索する範囲(バウンディボックス)をリストにする
search_areas = [
                bounding_box_1,
                bounding_box_2,
                bounding_box_3,
                bounding_box_4,
                bounding_box_5,
                bounding_box_6,
                bounding_box_7,
                bounding_box_8,
                bounding_box_9,
                bounding_box_10
              ]
```

```
# 線形探索 関数
def linear_search(search_area: tuple) -> int:
    linear_result = []
    for i in range(len(gdf)):
        # 各行の"geometry"列から座標を取得
        x, y = gdf["geometry"][i].x, gdf["geometry"][i].y

        # 範囲内にあるかどうかを判定し、結果をリストに追加
        if search_area[0] <= x <= search_area[2] and search_area[1] <= y <= search_area[3]:
            linear_result.append(i)
    return len(linear_result)
```

```
# R-Treeの構築
spatial_index = index.Index()
for i, geometry in enumerate(gdf.geometry):
    spatial_index.insert(i, geometry.bounds)
```

・探索を実行

```
# 探索結果を格納するリスト
linear_result = []
Rtree_result = []
# 実行時間を格納するリスト
linear_time = []
Rtree_time = []

# 探索
for search_area in search_areas:
    start_time_linear = time.time()
    linear_result.append(linear_search(search_area))
    linear_elapsed_time = time.time() - start_time_linear
    linear_time.append(linear_elapsed_time)

    start_time_rtree = time.time()
    rtree_result = list(spatial_index.intersection(search_area))
    rtree_elapsed_time = time.time() - start_time_rtree
    Rtree_result.append(len(rtree_result))
    Rtree_time.append(rtree_elapsed_time)
```

実行結果　（臼井）

	探索結果	線形探索	R-Tree	線形/R-Tree
範囲1	7877	0.71257	0.00222	320.4倍
範囲2	13058	0.63540	0.00242	262.5倍
範囲3	16667	0.66078	0.00300	219.9倍
範囲4	9420	0.63478	0.00178	357.4倍
範囲5	22308	0.63788	0.00424	150.4倍
範囲6	12551	0.63997	0.00223	286.7倍
範囲7	15977	0.63795	0.00279	228.8倍
範囲8	6925	0.65192	0.00138	471.0倍
範囲9	2416	0.64407	0.00053	1210.3倍
範囲10	11082	0.64882	0.00196	330.5倍

・全ての範囲の探索においてR-Treeの方が実行時間が短かった.

・探索結果の数が少ないほど線形探索と比較してR-Treeによる探索がかなり
高速に実行されている. ⇨ 探索範囲に存在する最小外接矩形の数が少ないから？

使用したデータと比較方法(矢野)

・国土数値情報（学校教育法に規定する全国の幼稚園、小学校、中学校、義務教育学校、高等学校、中等教育学校、特別支援学校、大学、高等専門学校、専修学校、各種学校、又は就学前の子どもに関する教育、保育等の総合的な提供の推進に関する法律に規定する幼保連携型認定こども園）

・関東地方（東京都，神奈川県，群馬県，埼玉県，千葉県，茨城県，栃木県）のデータを使用

・データ数は14867

・東京都内に存在する学校の数をカウントし実行時間を比較

実行結果 （矢野）

	線形探索	R-Tree
1	0.3119835	0.0023758
2	0.6854743	0.0035457
3	0.6460096	0.0040371
4	0.6460096	0.0042374
5	0.6551113	0.0036706
6	0.6810674	0.0025405
7	0.6766860	0.0050210
8	0.6676940	0.0032513
9	0.6579005	0.0055339
10	0.6244785	0.0055339

・10回中10回R-Treeの方が速かった

使用したデータ：全国の医療機関の場所
探索区間：四国と九州のそれぞれの都道府県

	P04_001	P04_002	P04_003	P04_004	P04_005	P04_006	P04_007	P04_008	P04_009	P04_010	geometry
0	1	医療法人樹恵会石田病院	標津郡中標津町りんどう町5番地6	内　リハ	None	None	4	60	9	9	POINT (144.92707 43.54428)
1	1	町立別海病院	野付郡別海町別海西本町103番地9	内　精　心内　小　外　整　産婦　耳　皮　リハ　脳神経内科	None	None	2	84	1	9	POINT (145.12249 43.39132)
2	1	標津町国民健康保険標津病院	標津郡標津町北1条西5丁目6番1	内　外	None	None	2	35	1	9	POINT (145.12101 43.65922)
3	1	町立中標津病院	標津郡中標津町西10条南9丁目1番地1	内　精　循　小　外　整　産婦　眼　耳　皮　泌　リハ　放　麻	None	None	2	173	1	2	POINT (144.97211 43.53651)
4	1	江村精神科内科病院	根室市有磯町2丁目25番地	内　精	None	None	4	101	9	9	POINT (145.58978 43.33285)

```
tokushima =  (134.19, 33.51, 134.36, 33.52)
kagawa = (133.41, 34.06, 134.20, 34.14)
ehime = (132.27, 33.20, 133.16, 33.56)
kochi = (132.32 , 32.47, 133.38, 33.32)
fukuoka = (130.06, 33.33, 130.23, 33.41)
saga=  (129.51, 33.22, 130.13, 33.34)
nagasaki = (128.21, 32.14, 129.26, 32.46)
kumamoto = (130.49, 32.10, 131.03, 32.49)
ooita = (131.40, 33.02, 131.55, 33.15)
miyazaki =  (131.19, 32.03, 131.46, 32.44)

area_name = ["徳島","香川","愛媛","高知","福岡","佐賀","長崎","熊本","大分","宮崎"]
area = [tokushima, kagawa, ehime, kochi, fukuoka, saga, nagasaki, kumamoto, ooita, miyazaki]
```

探索範囲でそれぞれ探索すると，10回ともR-treeの勝ち

```
宮崎
R-tree探索にかかった時間: 0.000076秒
R-tree探索結果のインデックス: 65
線形探索にかかった時間: 1.691976秒
線形探索結果のインデックス: 65
R-treeの勝ち
```

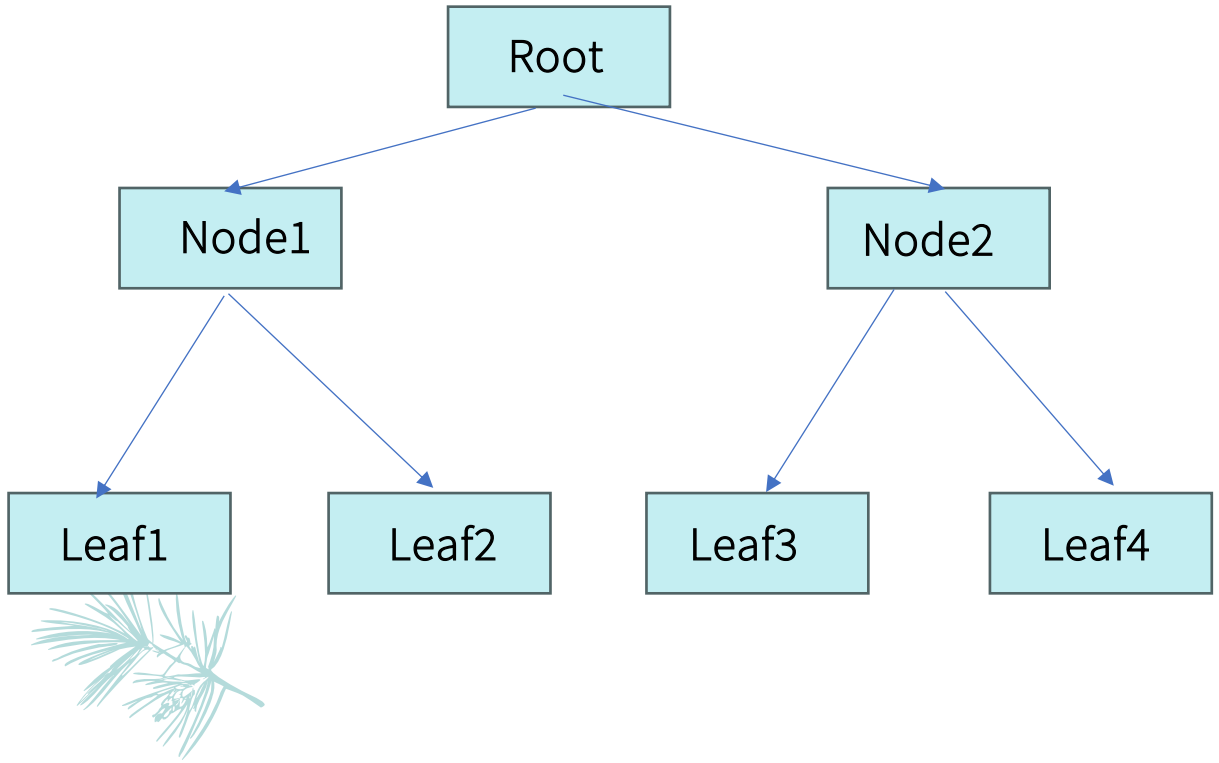
```
: print("R-treeの勝利数：" + str(win) + "回")
```

```
R-treeの勝利数：10回
```

R-Tree Algorithm

Faadhil As

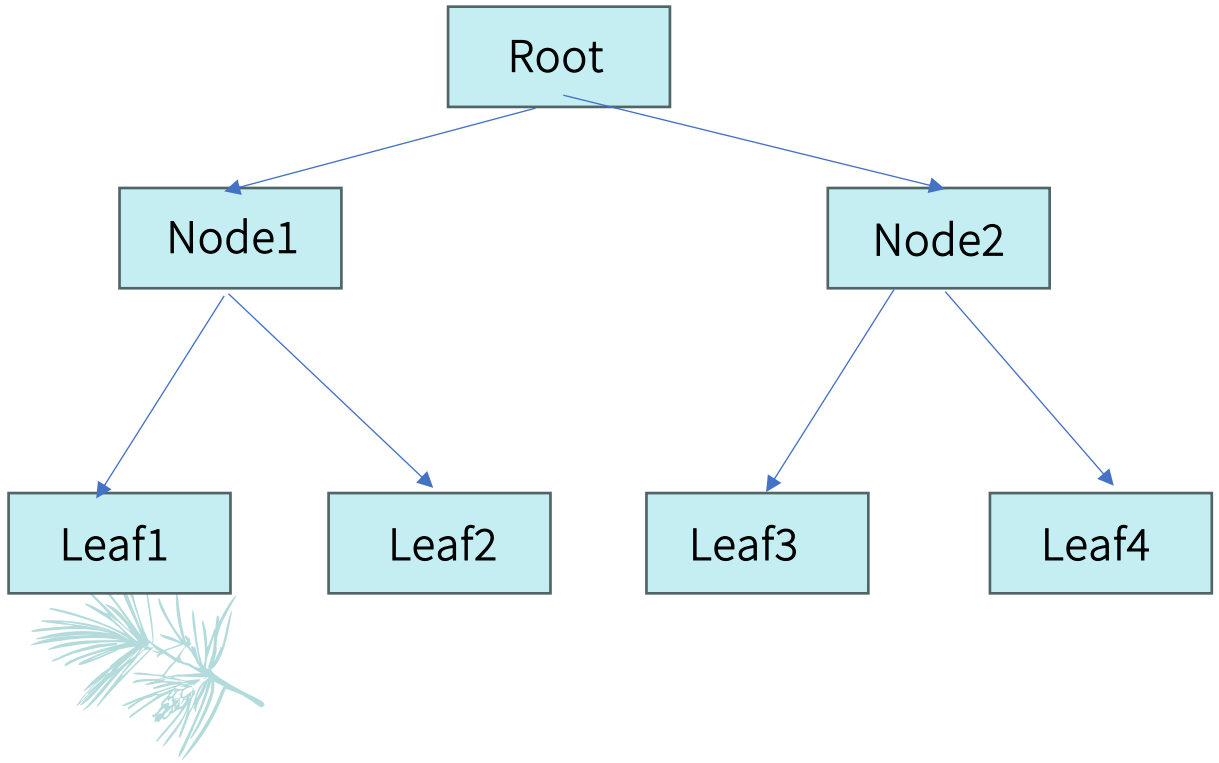
R-Tree



R-Tree Algorithm

- Efficiently organizes/searches spatial information
- Suited for geographical databases, map services.
- Organizes data hierarchically using rectangles.
- Simplifies finding objects in specific regions.

R-Tree



R-Tree contents

- Node

[I, tuple identifier]

I = n-dimensional rectangle which is the bounding box of the spatial object indexed
Tuple = refers to a tuple in the database

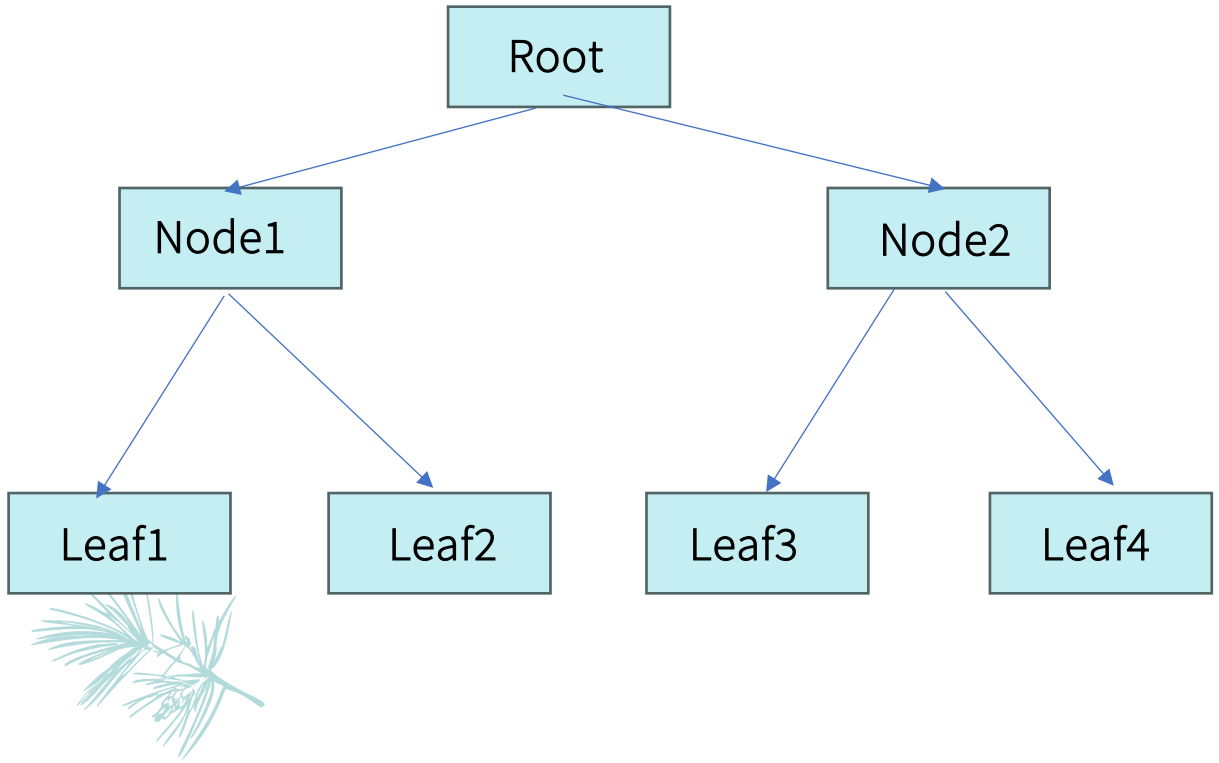
- Non-leaf node

[I, child-pointer]

I = covers all rectangles in the lower node's entries.
child-pointer = the address of a lower node

- Root = *topmost node that contains entries representing the bounding rectangles encompassing all spatial data in the lower levels of the tree.*

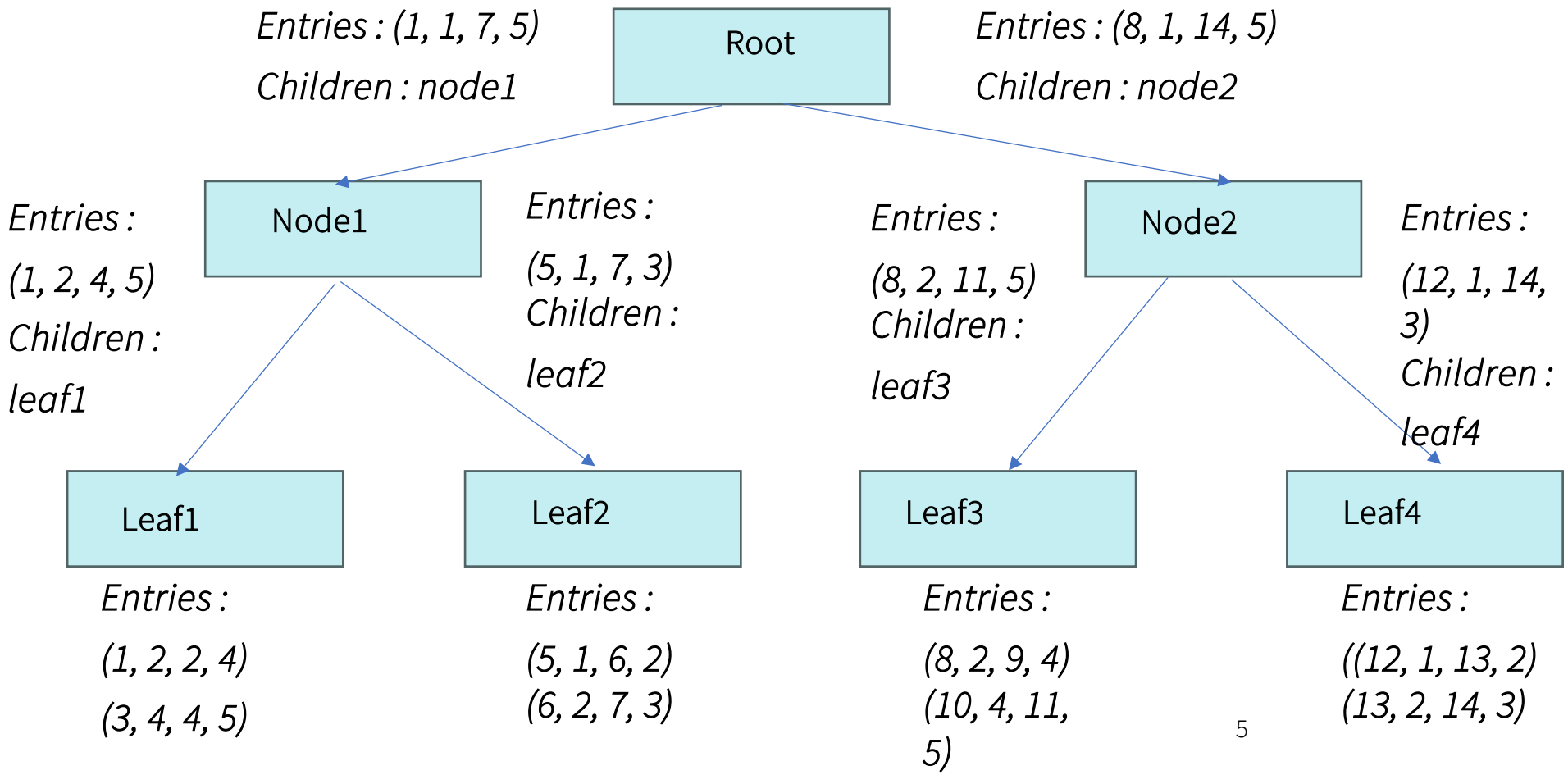
R-Tree



R-Tree Properties

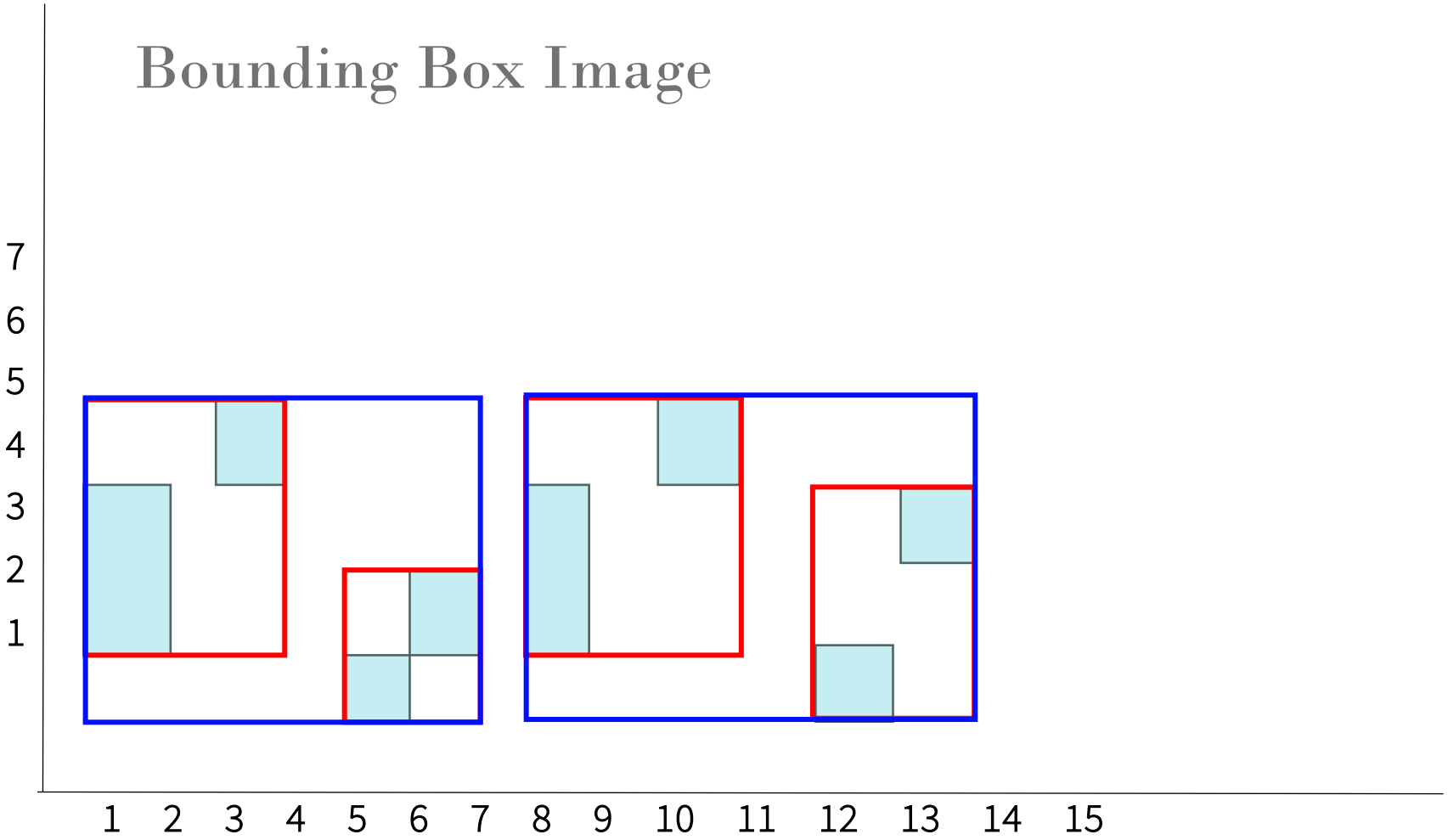
- Every leaf node contains between m and M index records unless it is the root
- For each index record (I, tuple-identifier) in a leaf node, I is the smallest rectangle that spatially contains the n-dimensional data object represented by the indicated tuple.
- For each entry (/, child —pointer) in a non-leaf node, I is the smallest rectangle that spatially contains the rectangles in the child node.
- The root node has at least two children unless it is a leaf.
- All leaves appear on the same level.

R-Tree application

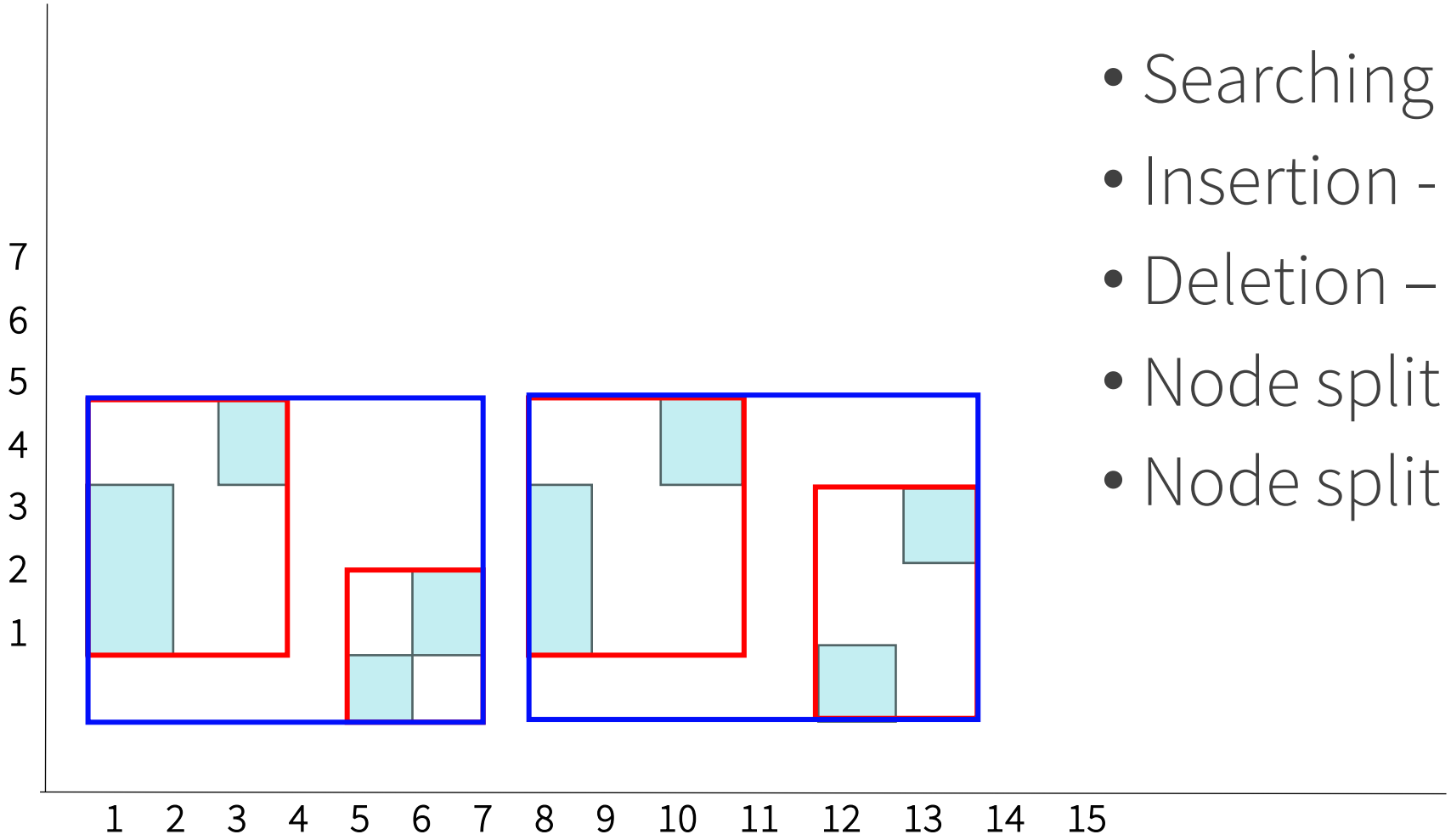


5

R-Tree application



R-Tree application



R-Tree Operation

- Searching
- Insertion - Update tree
- Deletion – Condense tree
- Node split – Quadratic cost
- Node split – Linear cost