Spatial Data

Immanuel Trummer
itrummer@cornell.edu
www.itrummer.org

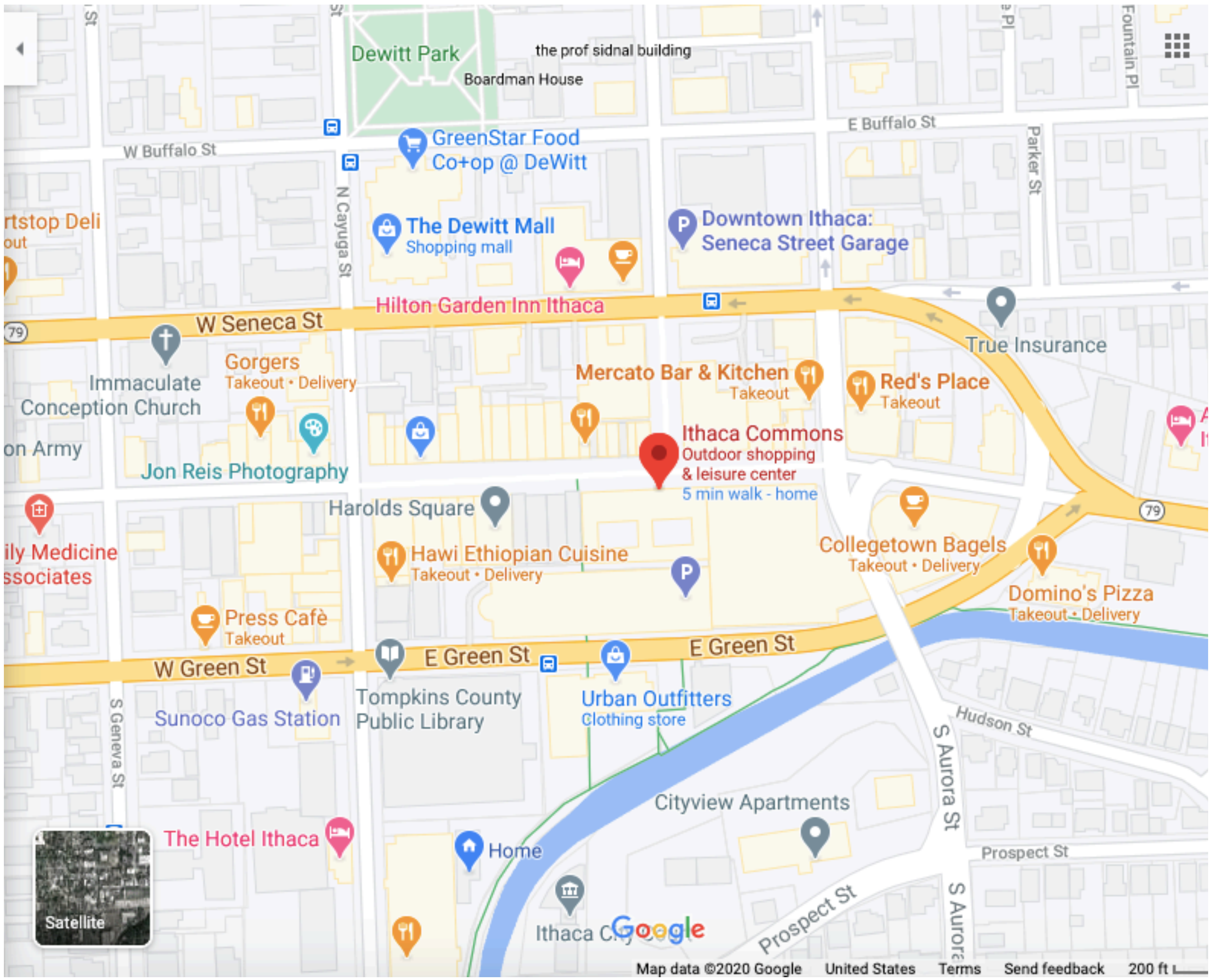
[RG, Sec. 28]

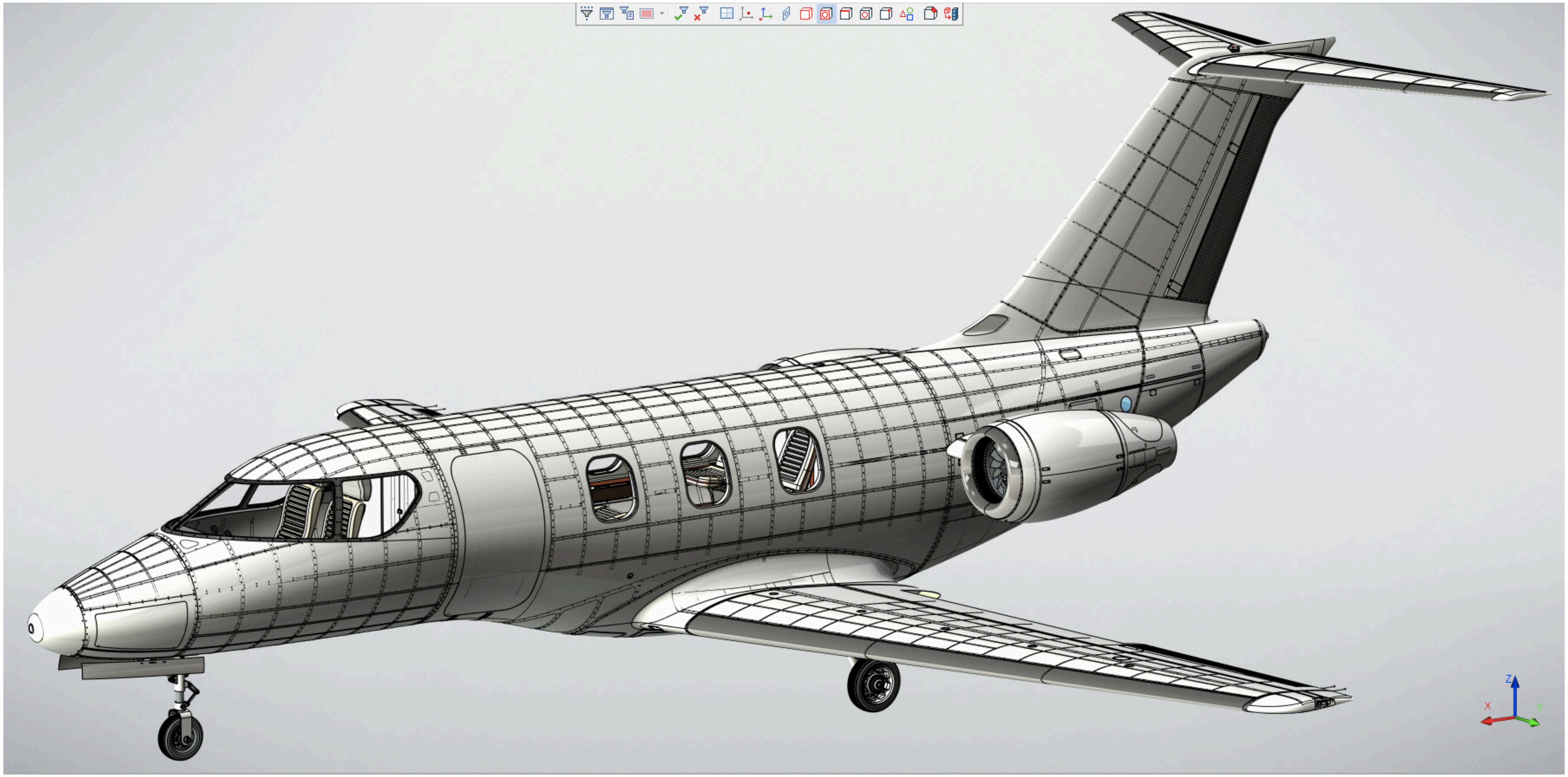
Outlook: Beyond Relational Data

- Graph data
- Data streams
- Spatial data

Outlook:
Beyond Relational Data

- Graph data
- Data streams
- **Spatial data**


Dewitt Park
the prof sidnal building
Boardman House
E Buffalo St
W Buffalo St
GreenStar Food Co+op @ DeWitt
N Cayuga St
Parker St
The Dewitt Mall
Shopping mall
Downtown Ithaca: Seneca Street Garage
Hilton Garden Inn Ithaca
W Seneca St
True Insurance
Gorgers
Takeout • Delivery
Immaculate Conception Church
Mercato Bar & Kitchen
Takeout
Red's Place
Takeout
Ithaca Commons
Outdoor shopping & leisure center
5 min walk - home
Jon Reis Photography
Harolds Square
Collegetown Bagels
Takeout • Delivery
Hawi Ethiopian Cuisine
Takeout • Delivery
Domino's Pizza
Takeout • Delivery
Press Cafè
Takeout
W Green St
E Green St
Tompkins County Public Library
Urban Outfitters
Clothing store
Sunoco Gas Station
S Geneva St
Hudson St
S Aurora St
Cityview Apartments
The Hotel Ithaca
Home
Prospect St
Satellite
Map data ©2020 Google
United States
Terms
Send feedback
200 ft



Source: Wikipedia

Types of Spatial Data

- **Point** data
  - Characterized completely by the location
- **Region** data
  - Defined by a boundary (e.g., line or surface)
  - May have anchor location (e.g., centroid)

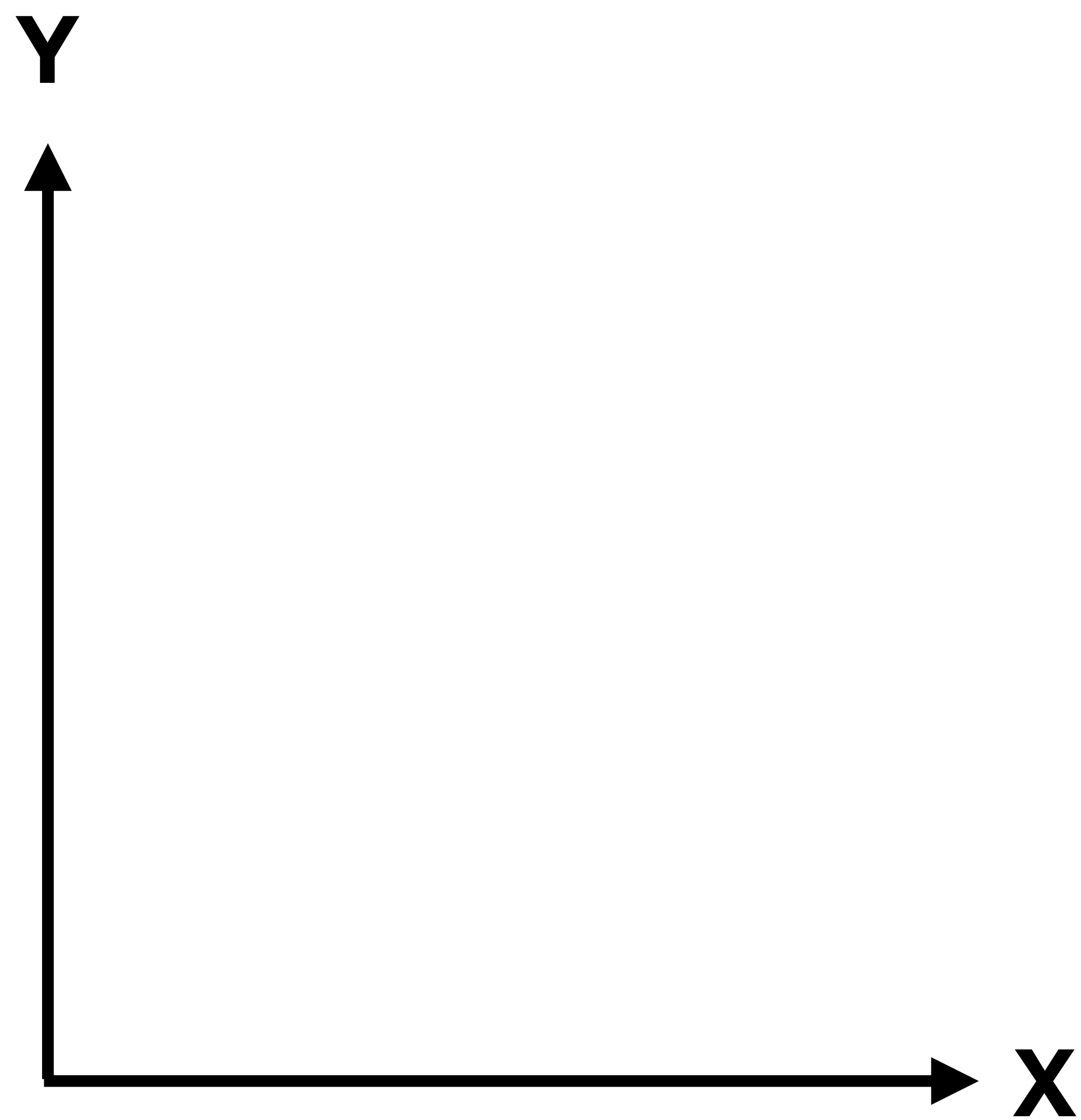
Types of Spatial Queries

- **Spatial range** queries
  - E.g., show me restaurants in Ithaca
- **Nearest neighbor** queries
  - E.g., show me the nearest gas station
- **Spatial joins**
  - E.g., show hiking trails with parking within 100 m

Outlook: Indexing

- **B+ trees** for spatial data
- Space-filling **curves**
- **Region** quad tree
- **Grid** files
- The **R tree**

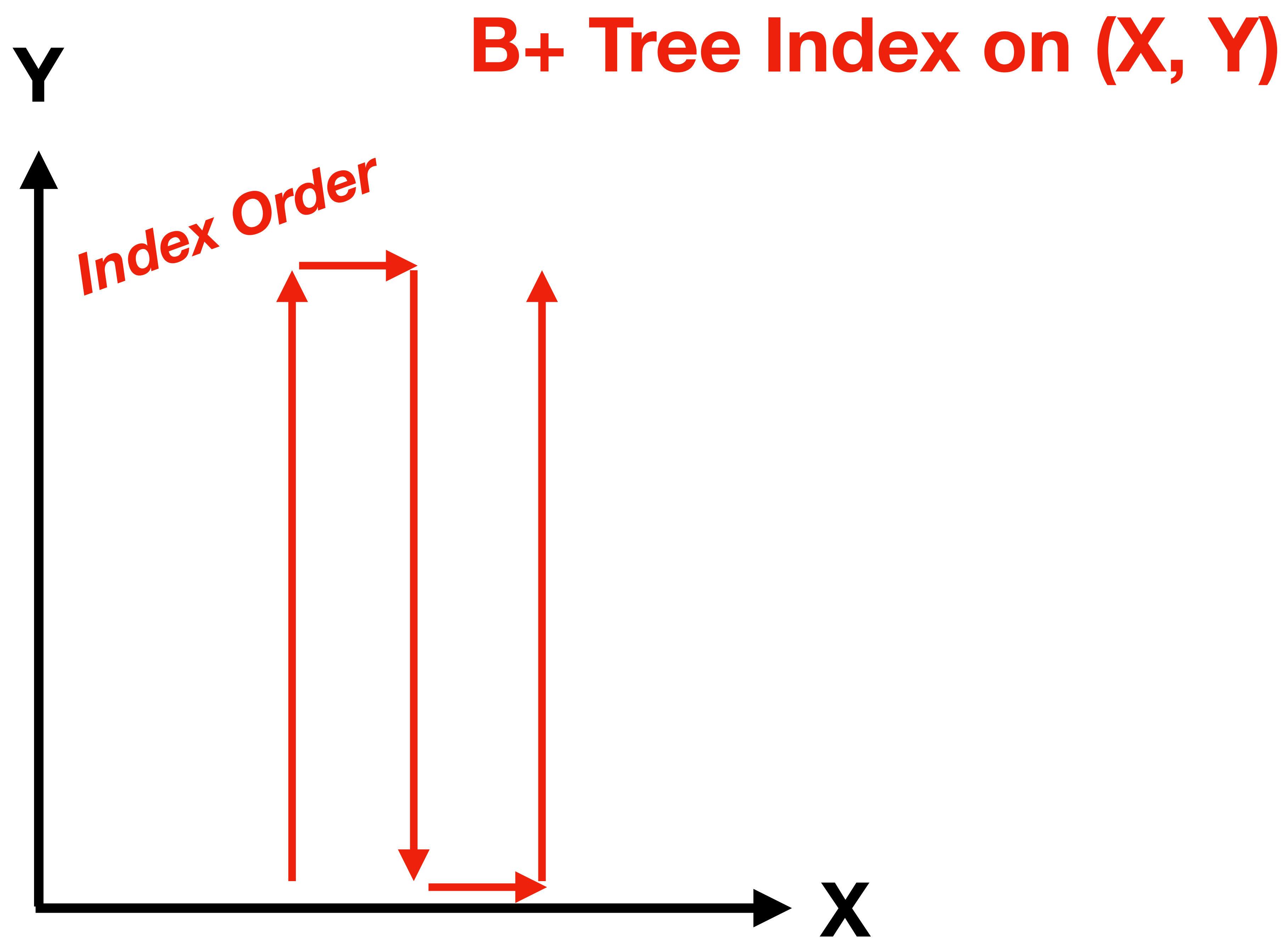
B+ Trees for Spatial Data



B+ Trees for Spatial Data

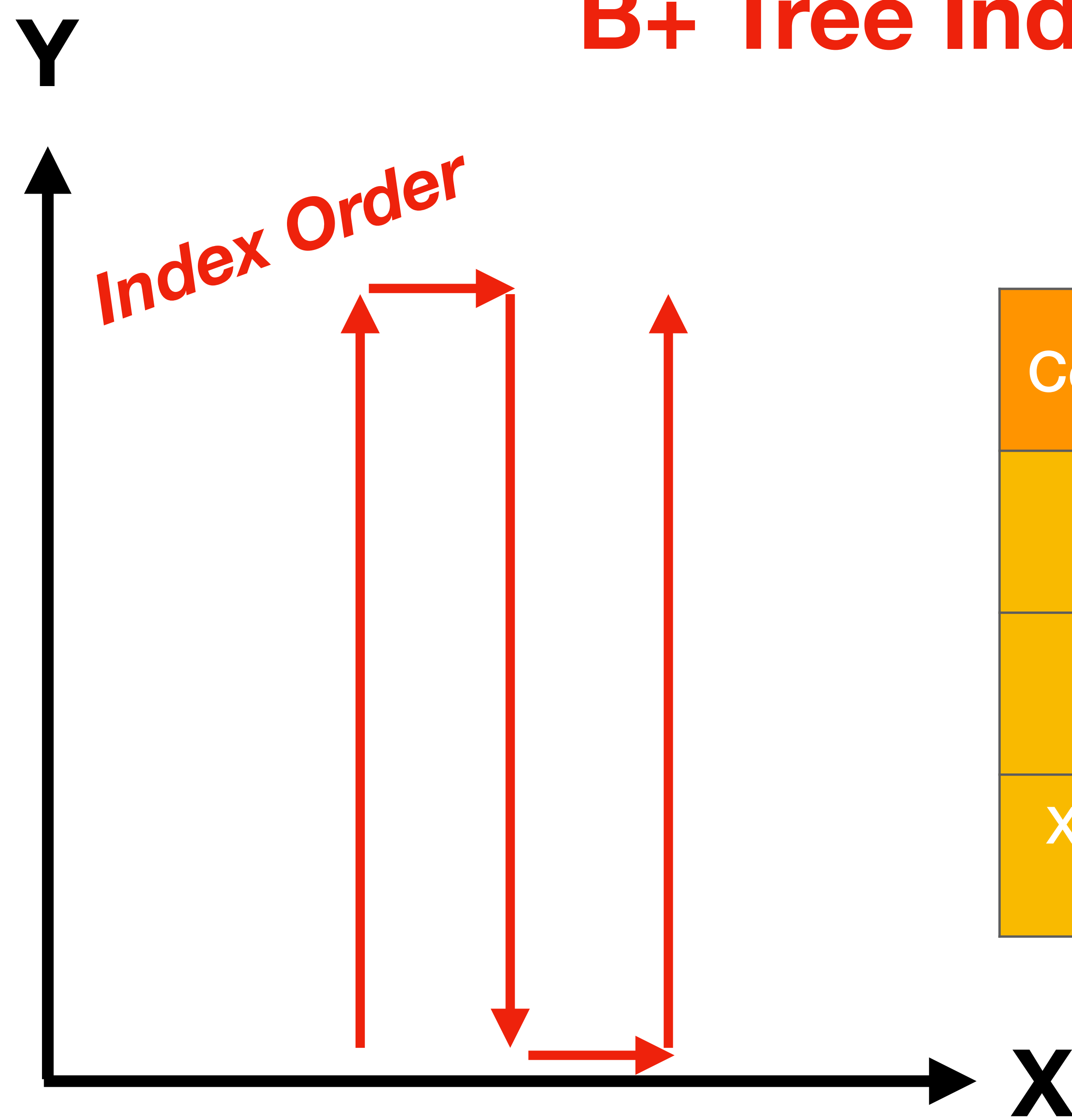
B+ Tree Index on (X, Y)

B+ Trees for Spatial Data



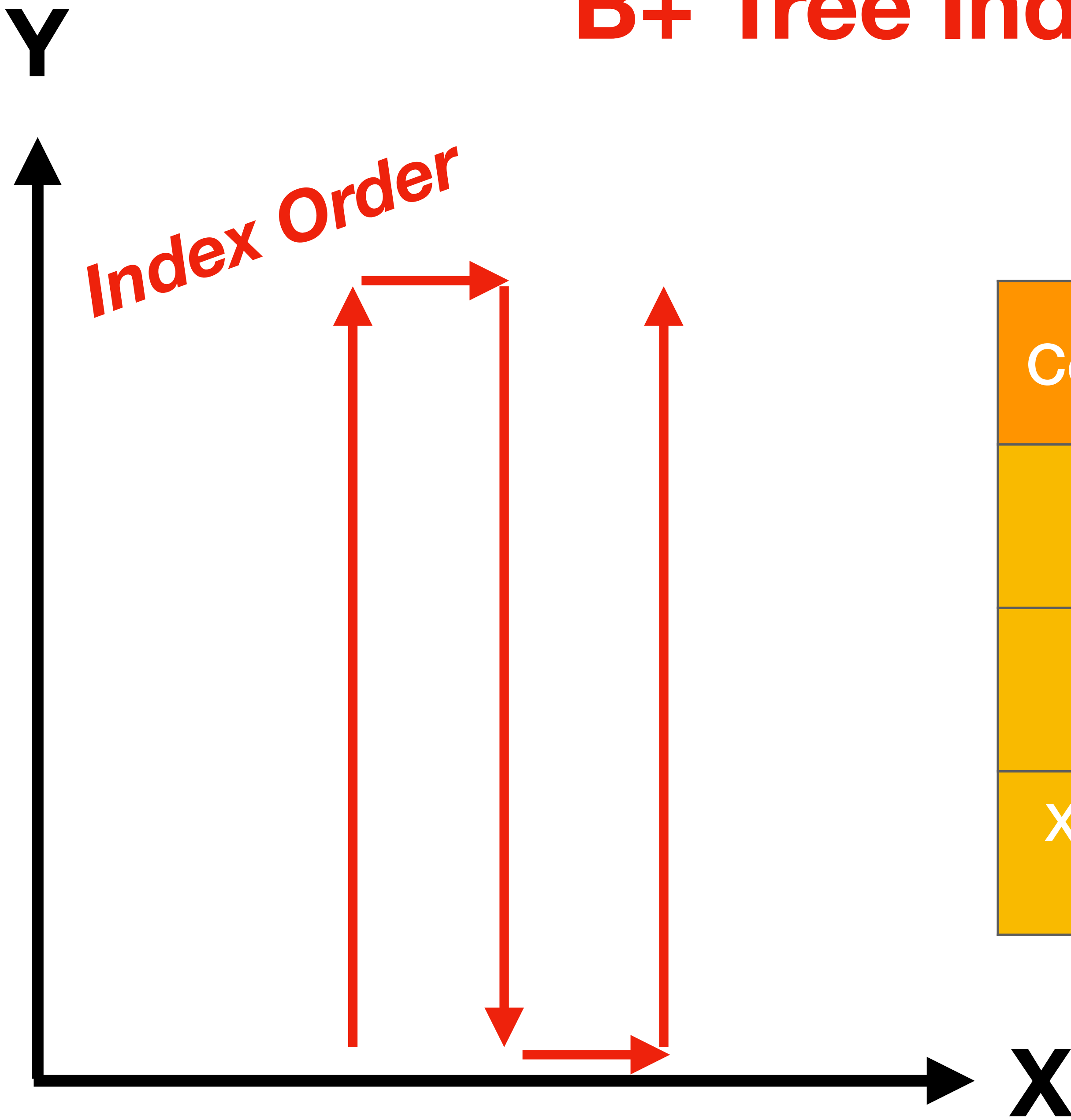
B+ Trees for Spatial Data

B+ Tree Index on (X, Y)



Condition	Applicable
X<3	
Y>2	
X<2 AND Y<3	

B+ Trees for Spatial Data



Condition	Applicable
X<3	
Y>2	
X<2 AND Y<3	

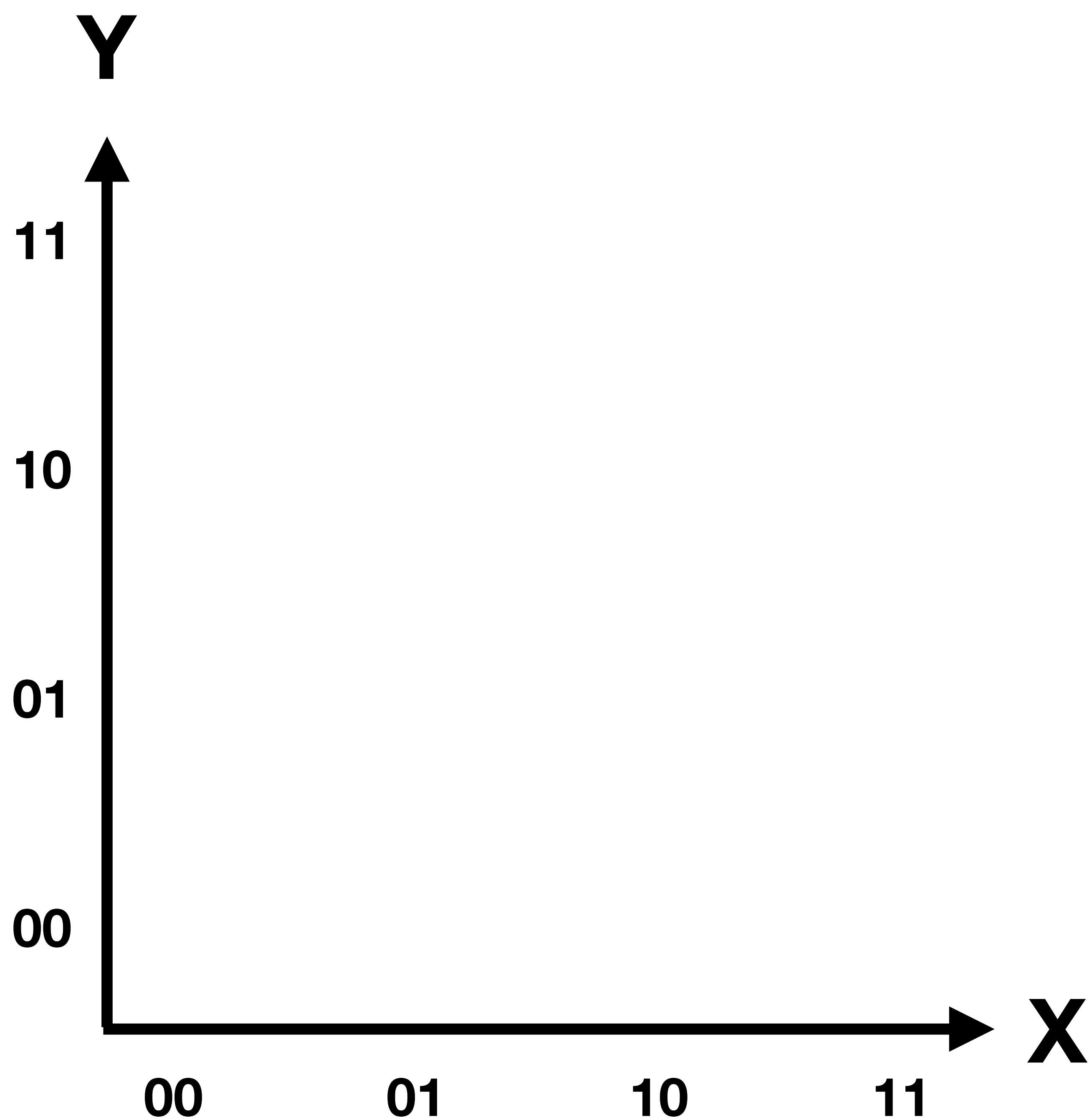
Problem with B+ Trees

- **Close points** (in 2D) not close in index
- Answering range queries etc. **inefficient**
- Could use one **tree per dimension** and merge RIDs
  - But leads to various overheads!

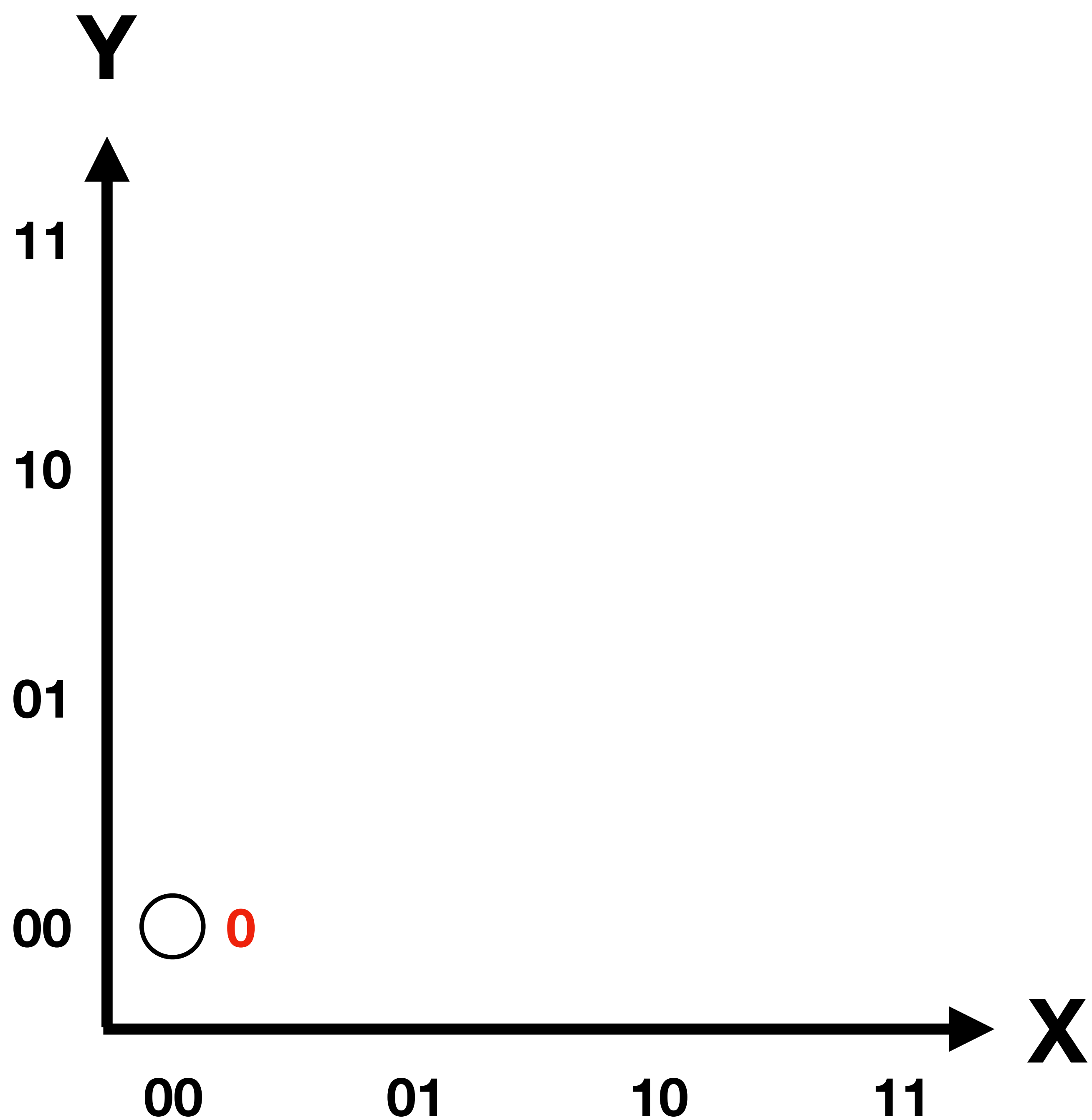
Z-Ordering

- **Numbers** each space coordinate
- **Close points** have close numbers
  - Not always, can avoid (Hilbert curves)
- **Binary** representation for each coordinate
  - E.g., $(a_1a_2...a_n, b_1b_2...b_n)$ for 2D
- Z-Ordering assigns number $a_1b_1a_2b_2...a_nb_n$

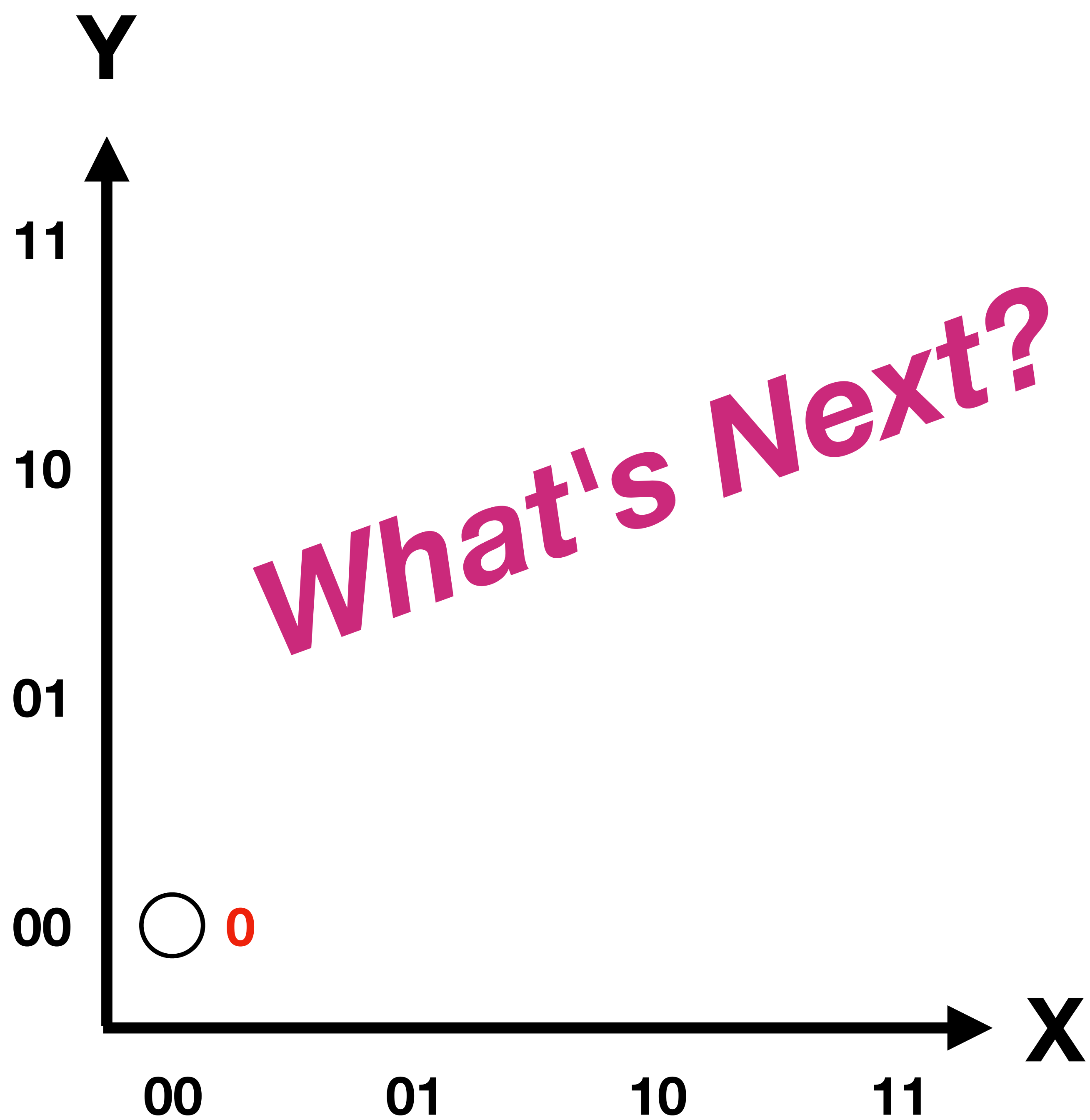
Z-Ordering



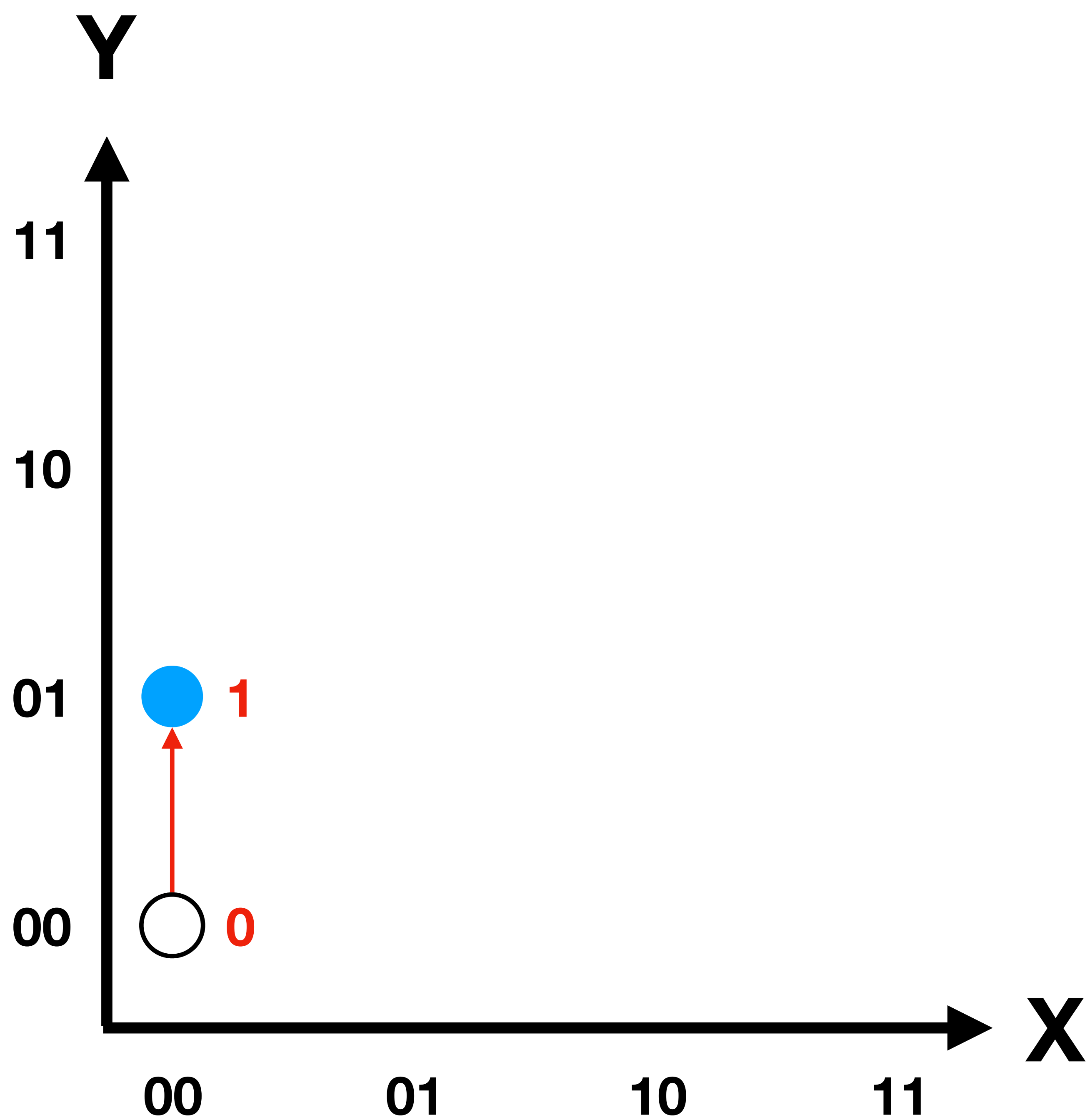
Z-Ordering



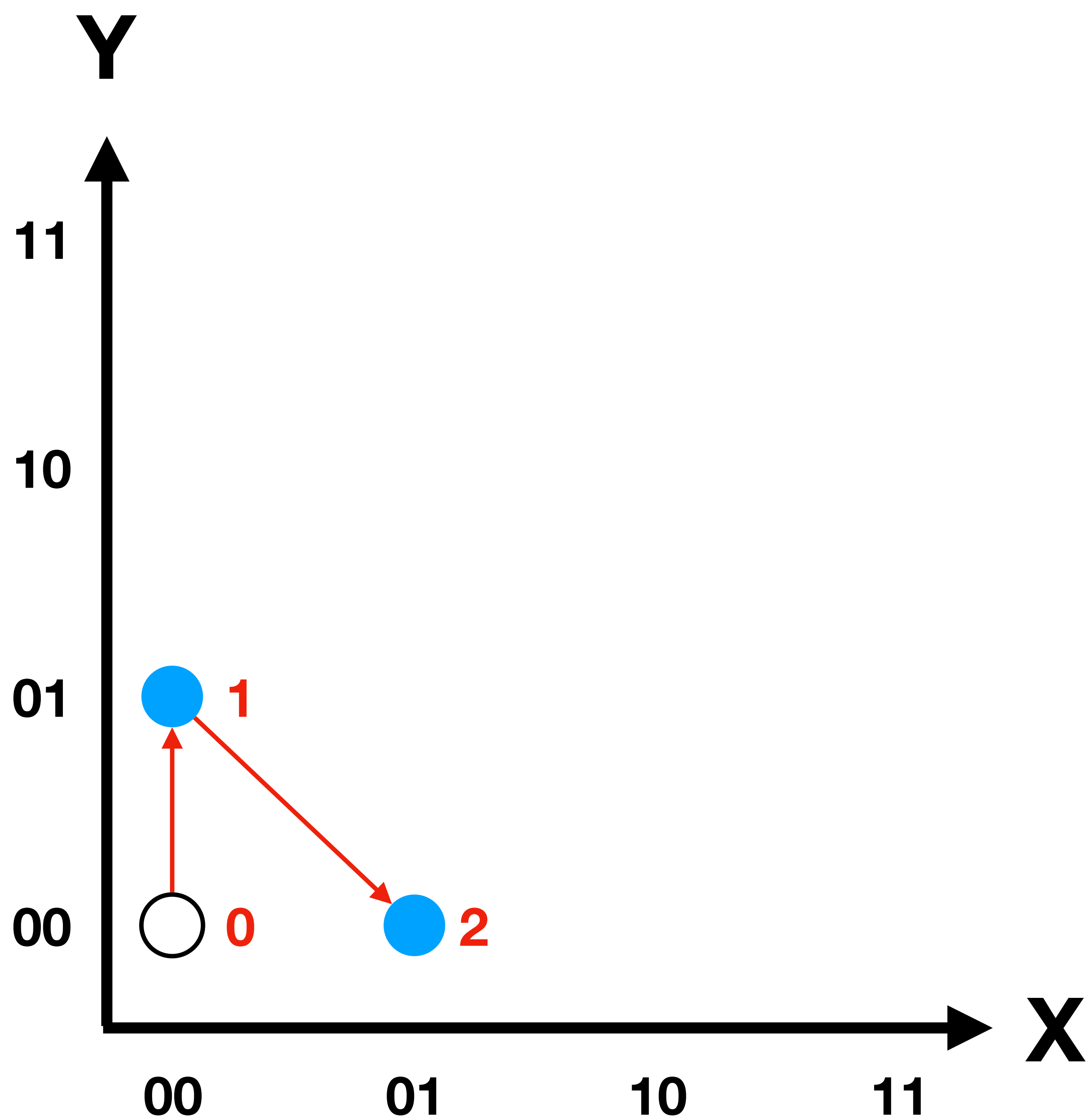
Z-Ordering



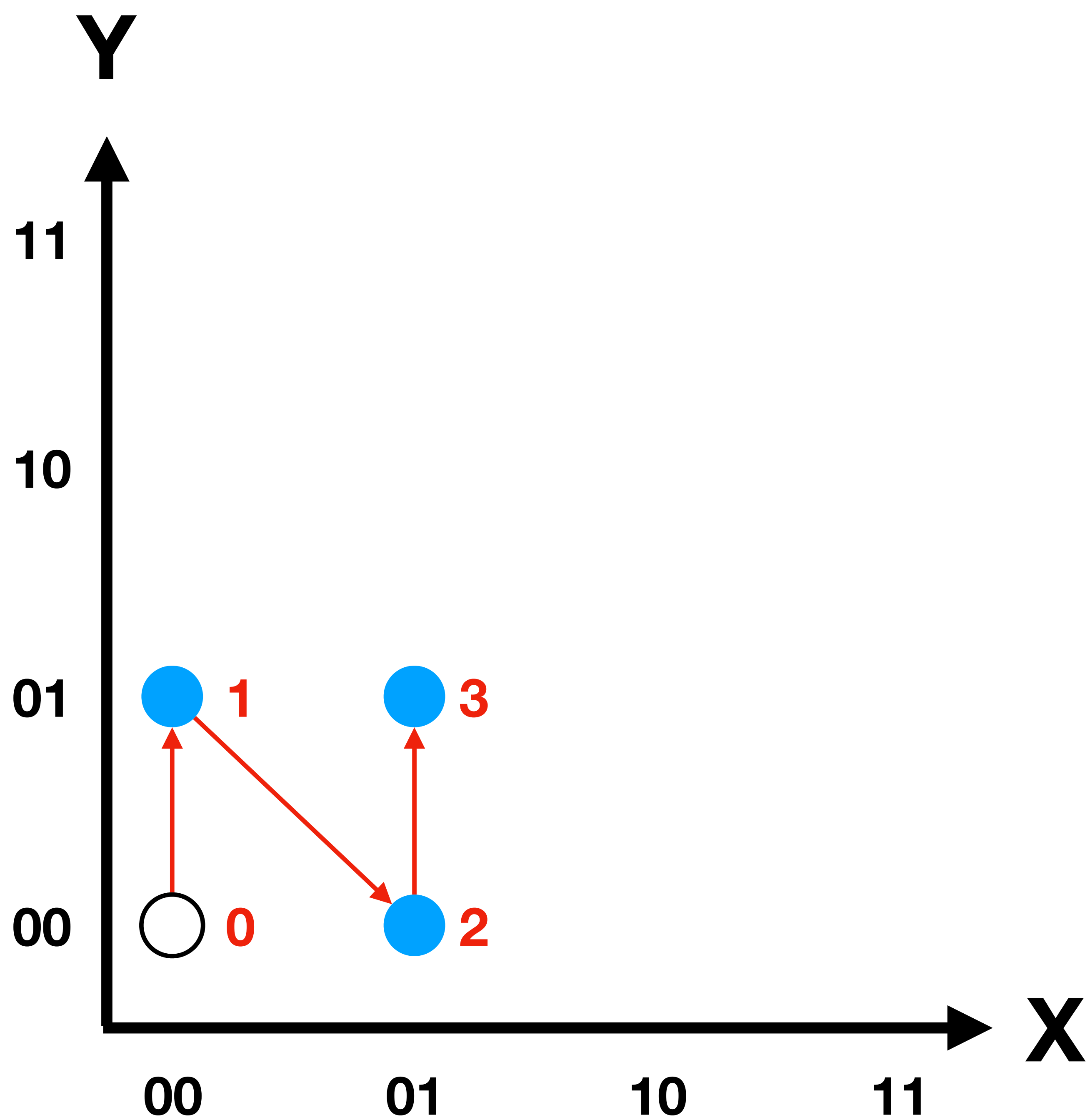
Z-Ordering



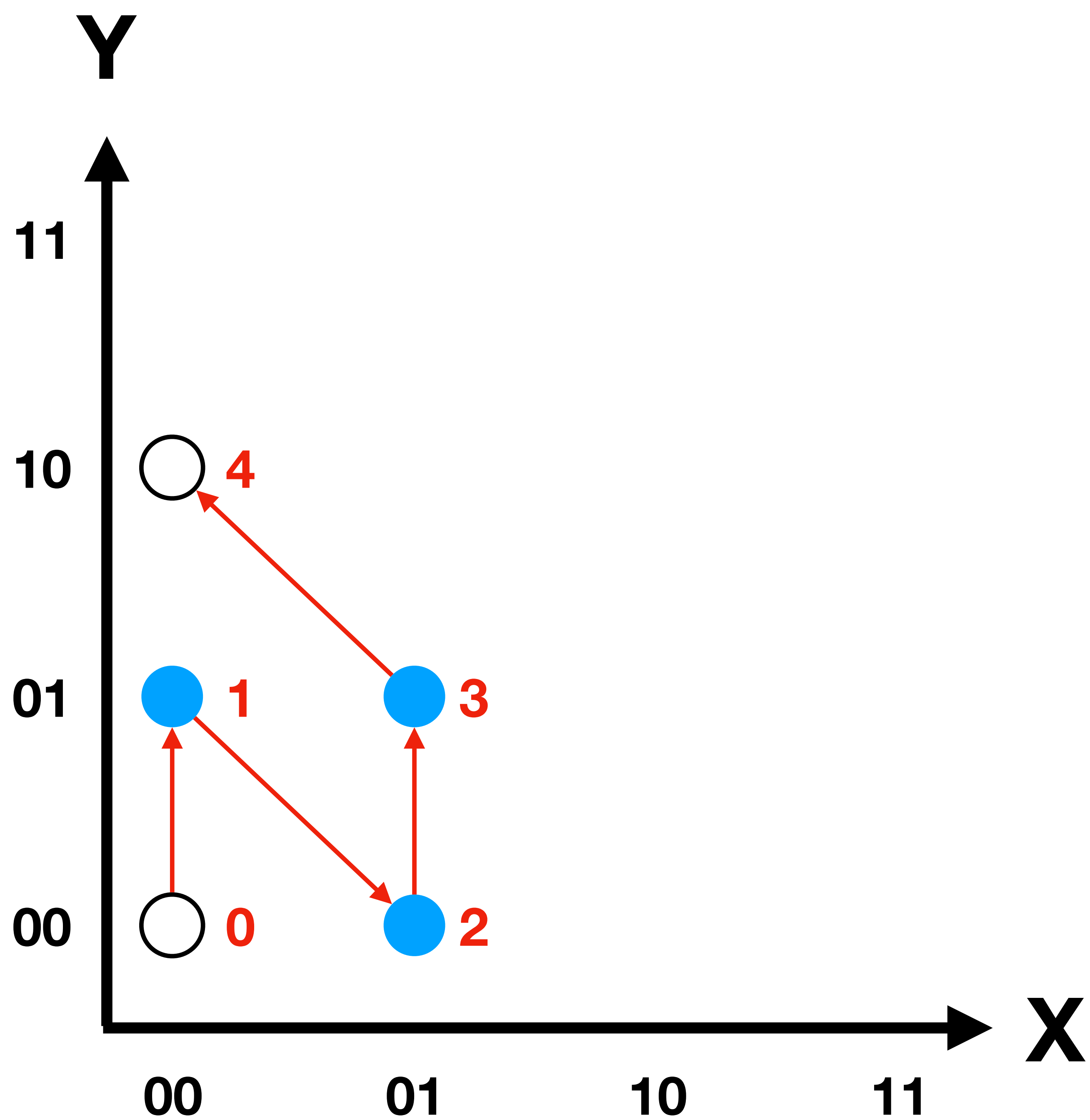
Z-Ordering



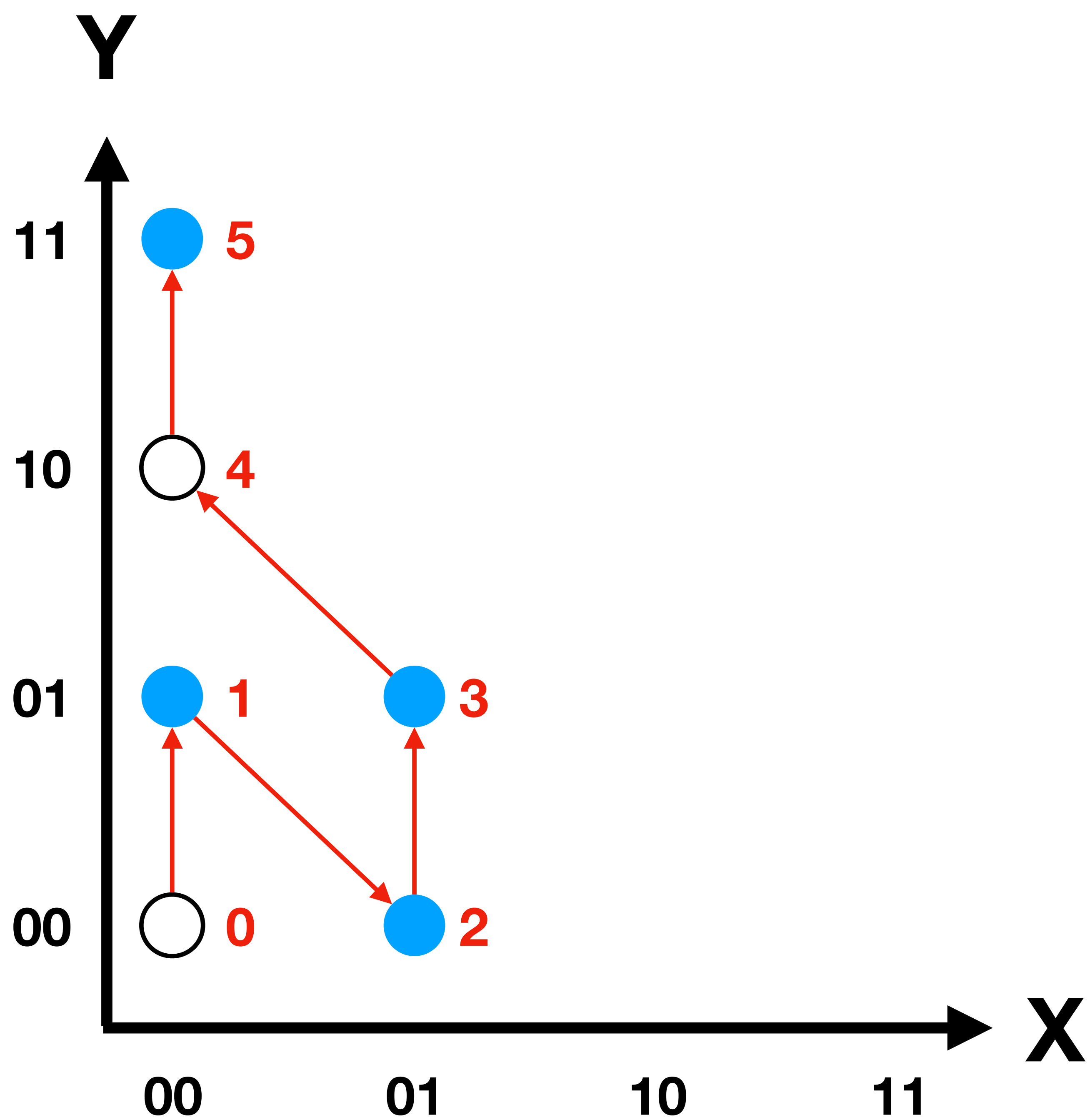
Z-Ordering



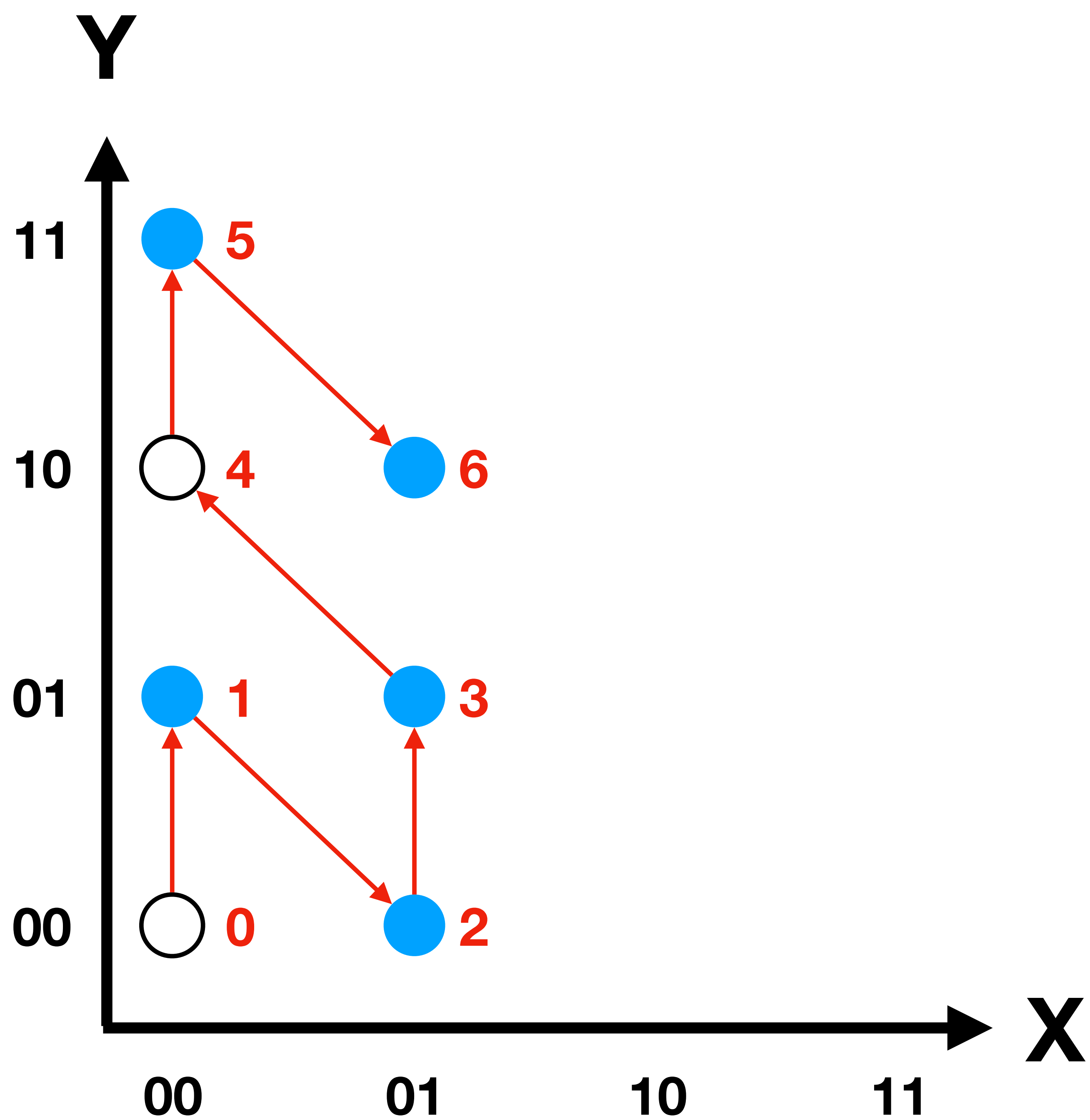
Z-Ordering



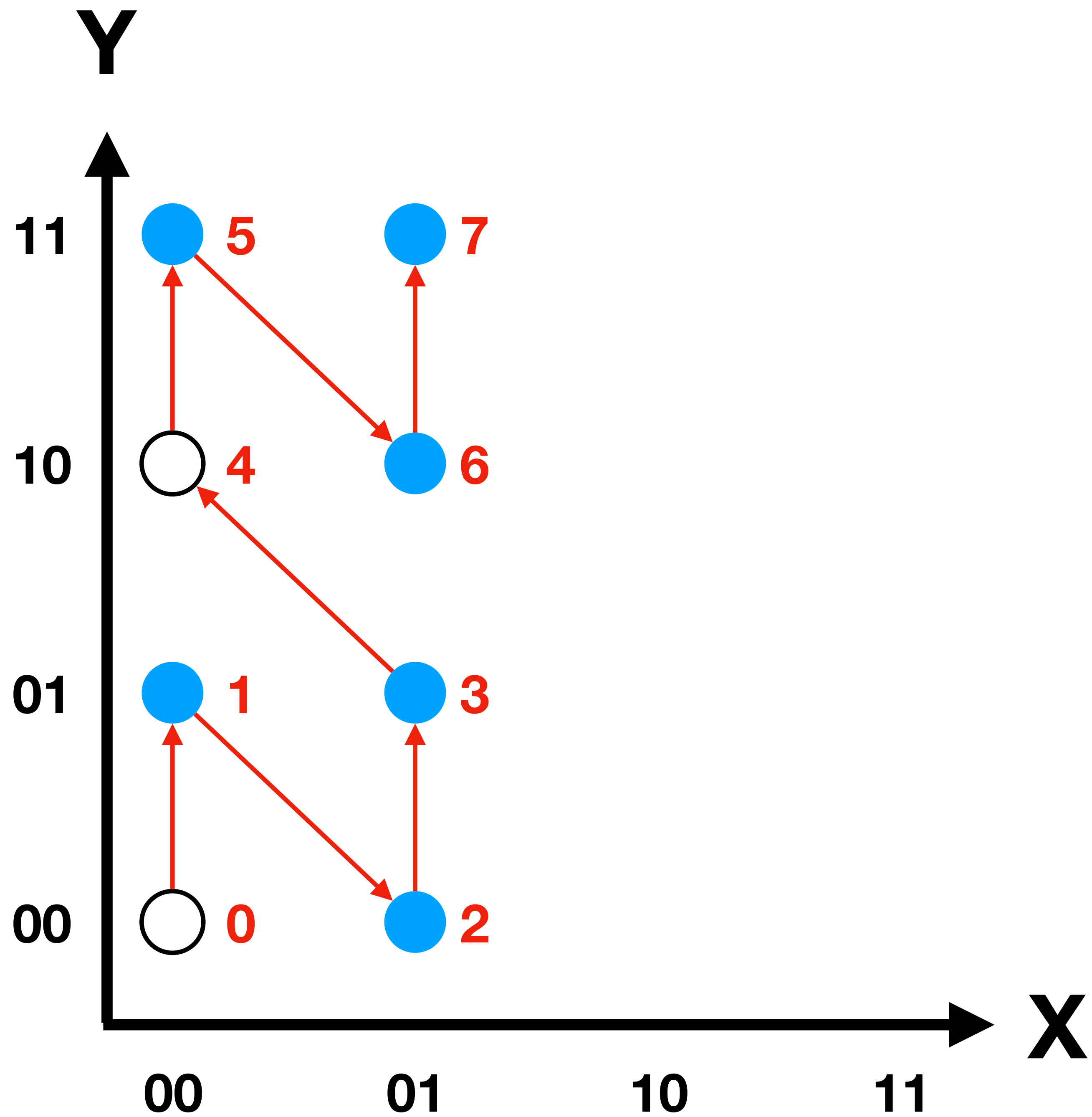
Z-Ordering



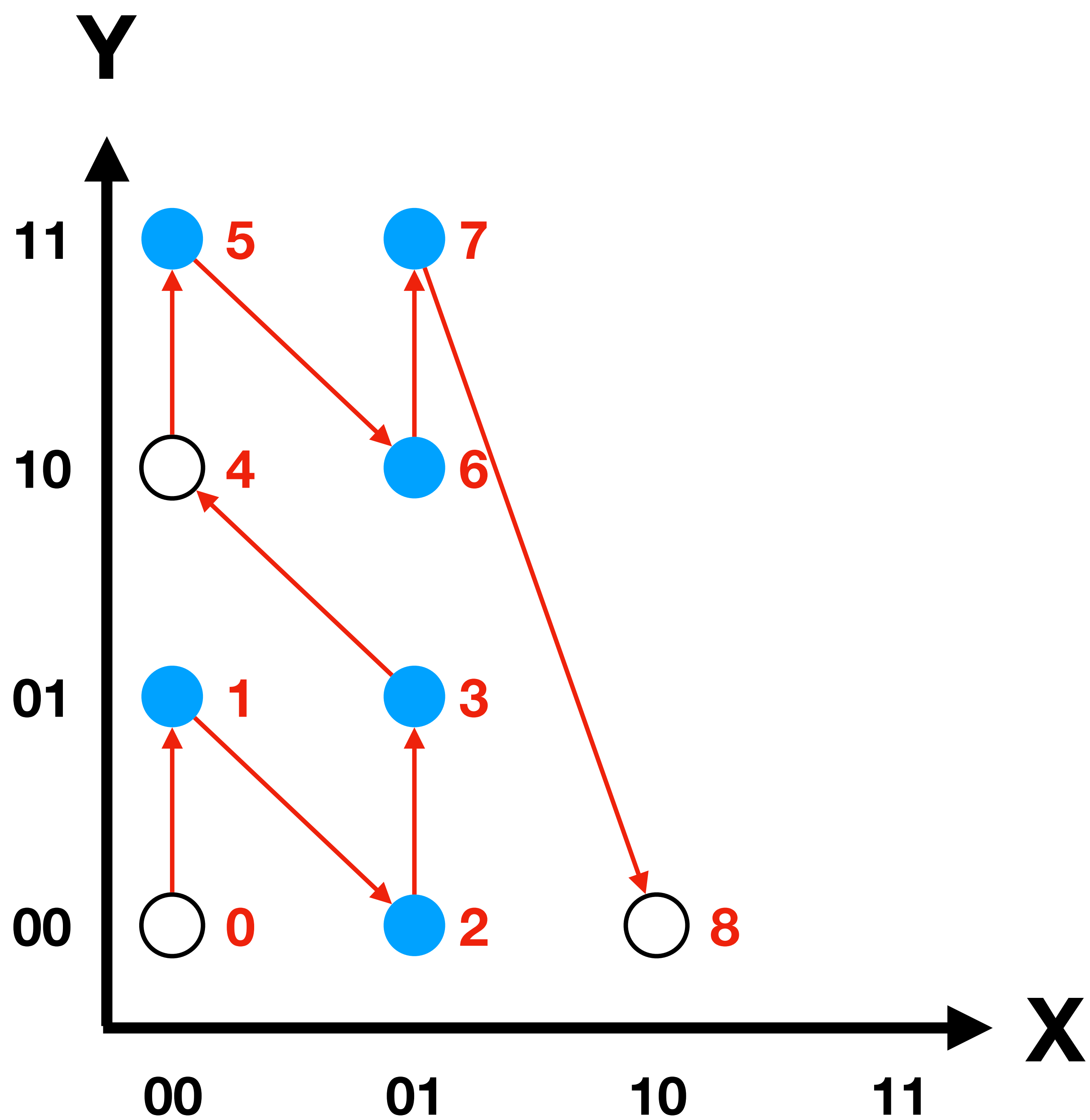
Z-Ordering



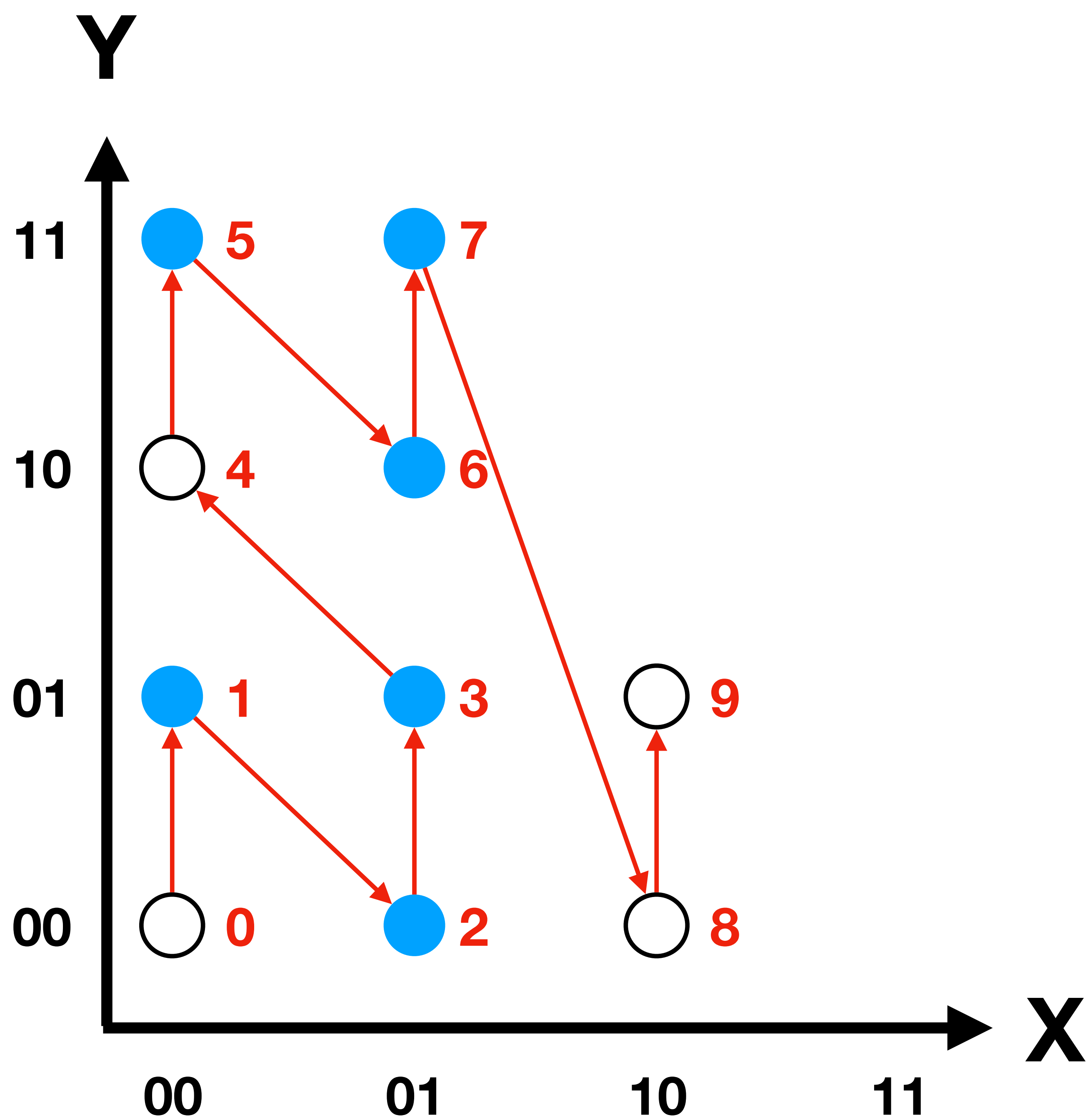
Z-Ordering



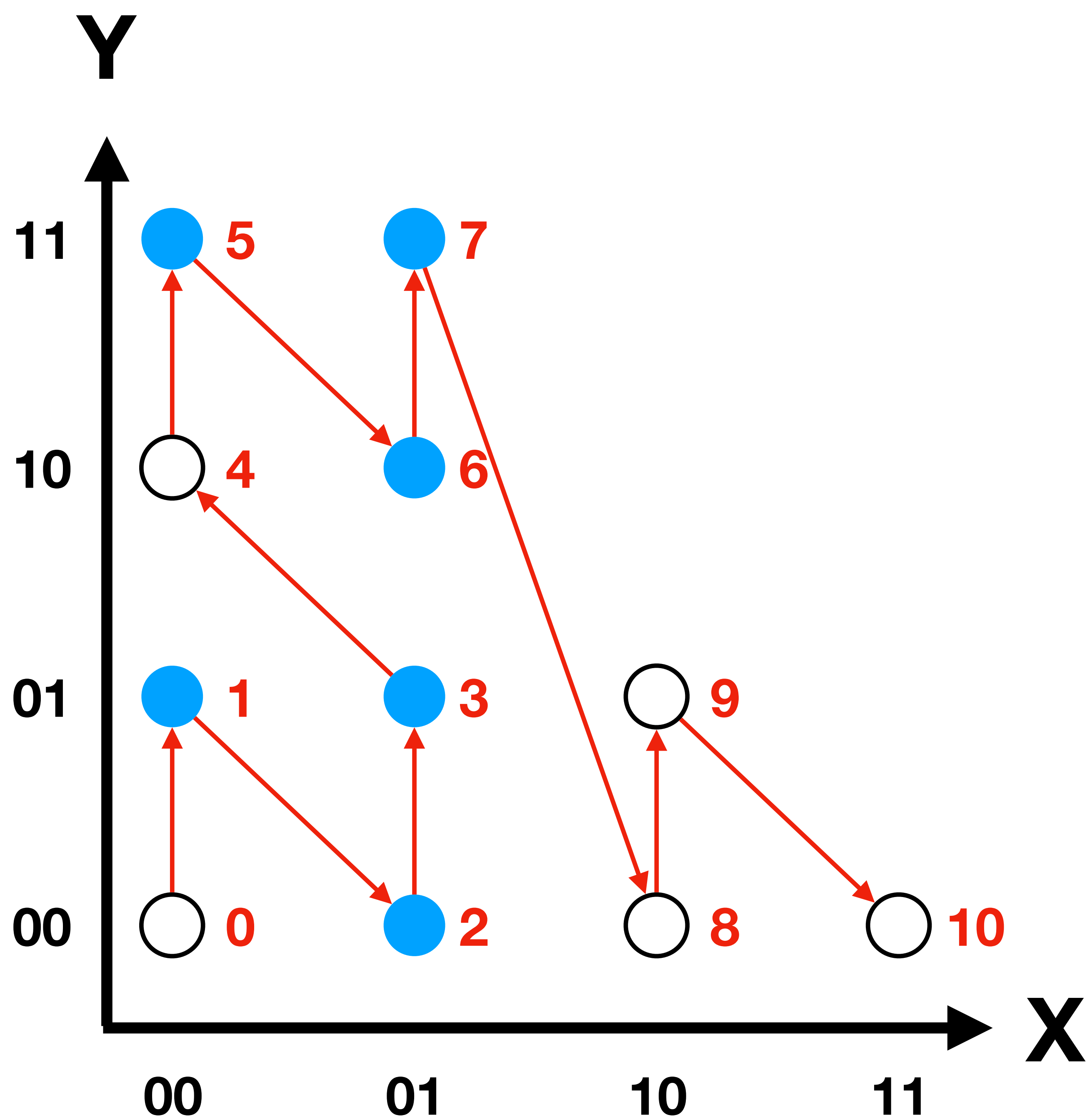
Z-Ordering



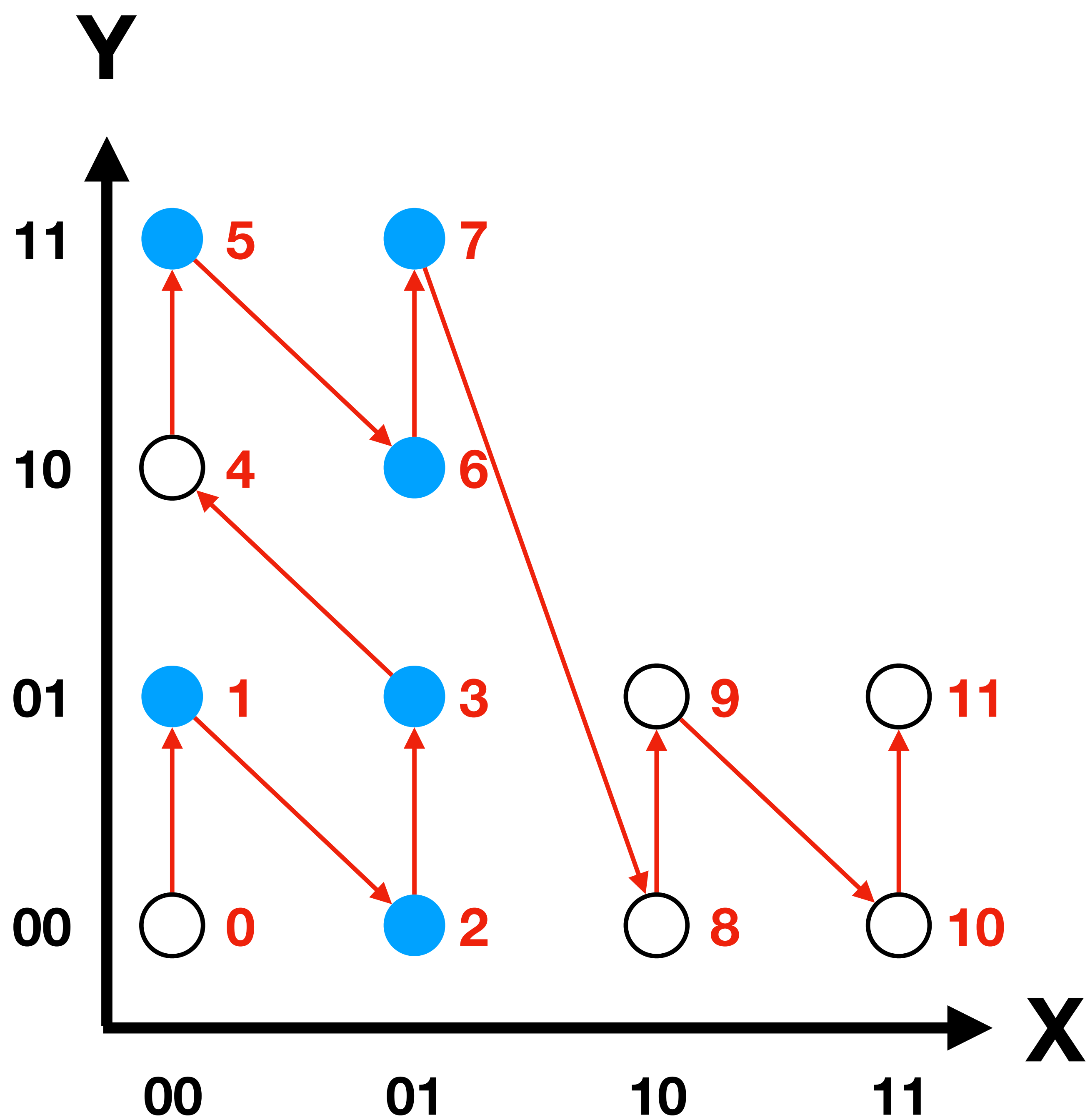
Z-Ordering



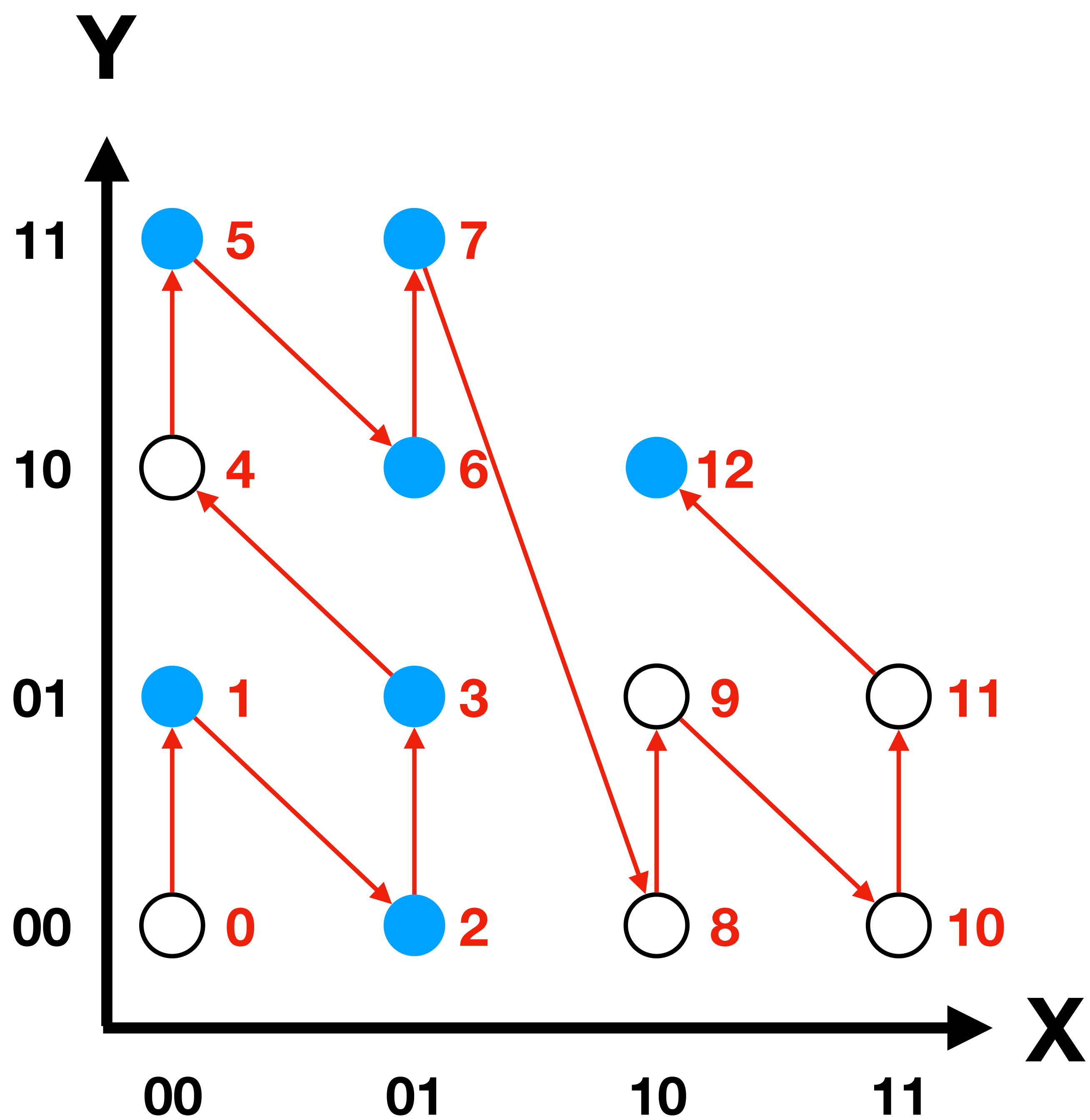
Z-Ordering



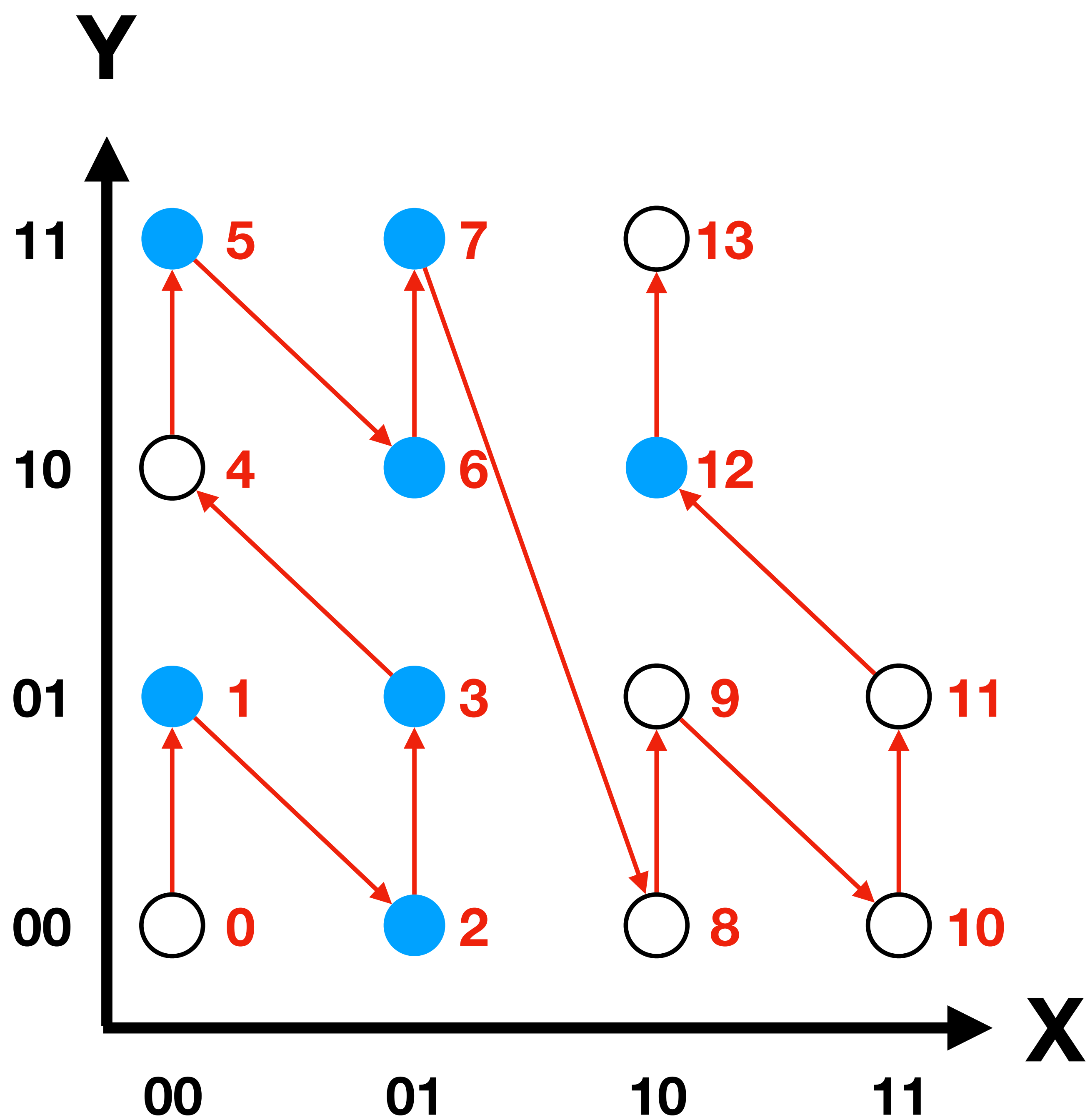
Z-Ordering



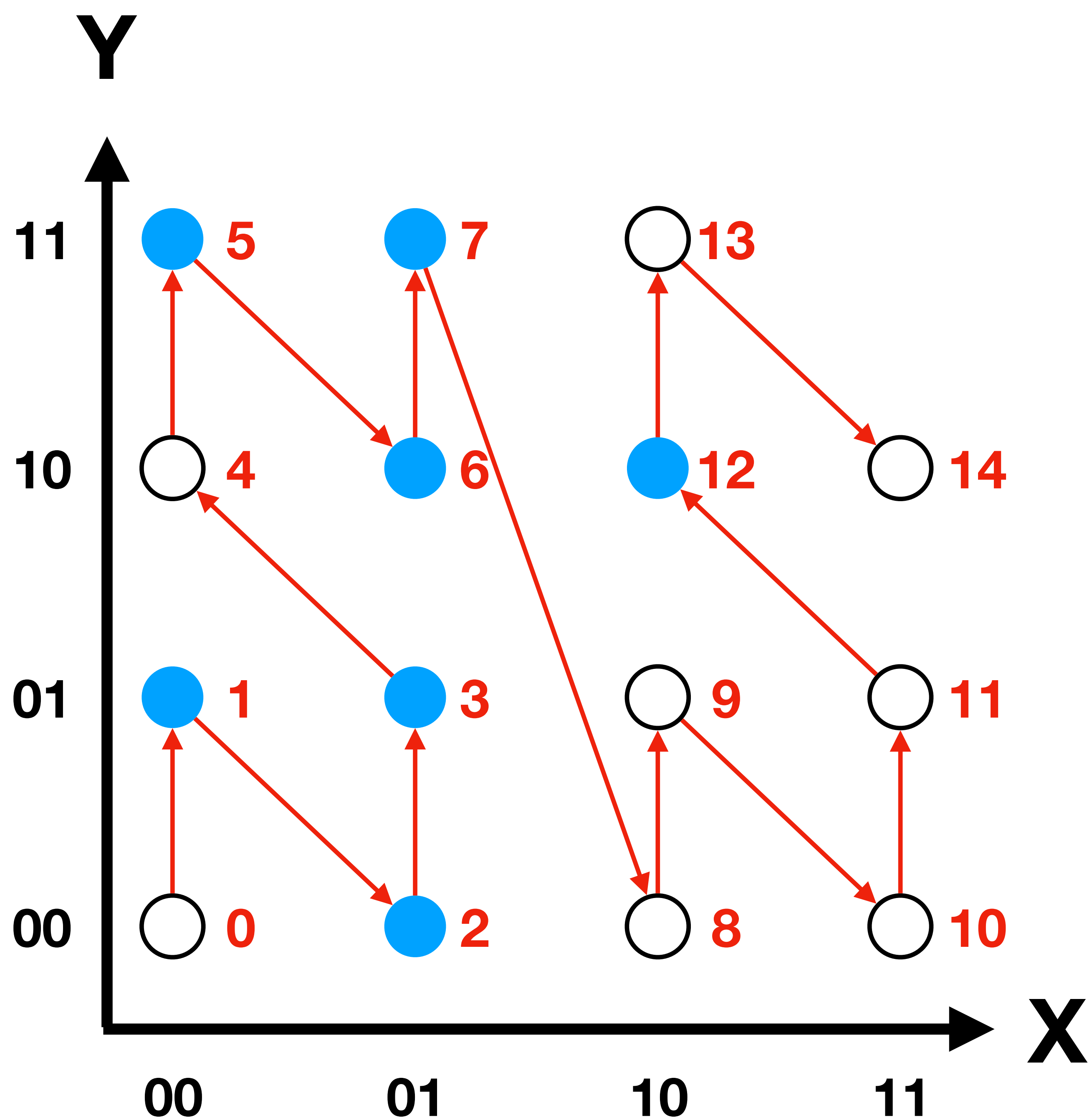
Z-Ordering



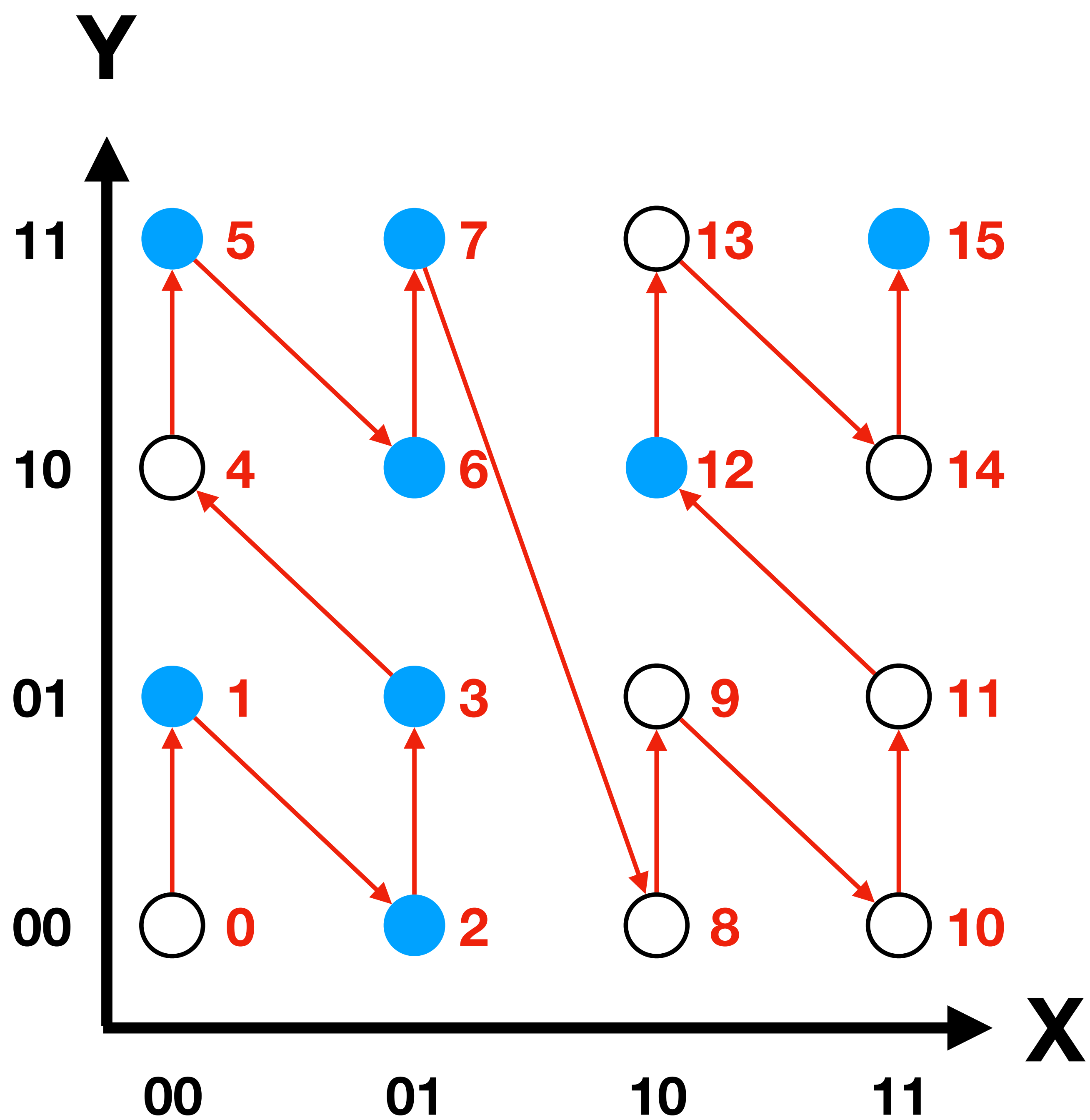
Z-Ordering



Z-Ordering



Z-Ordering



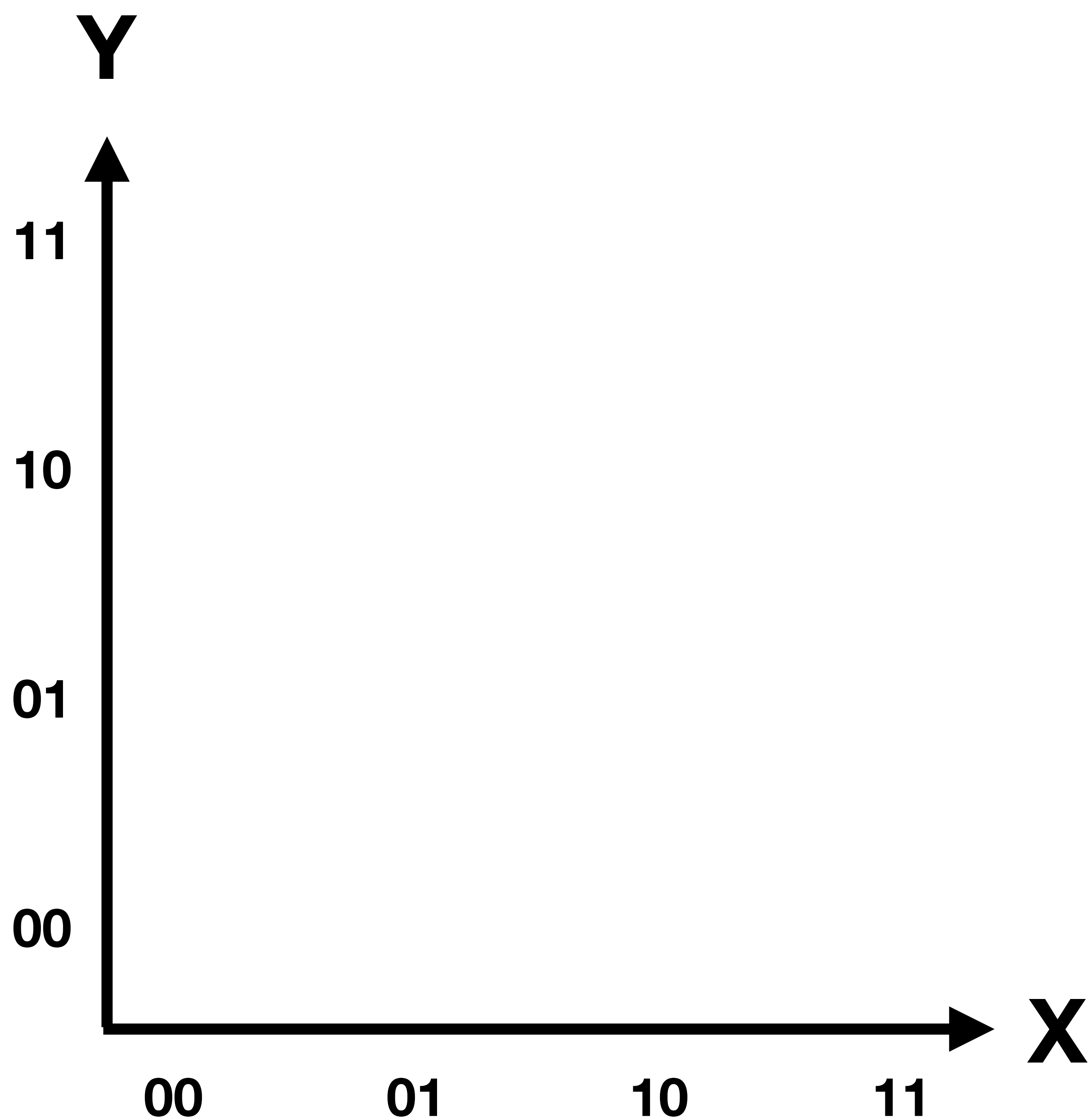
Indexing with Z-Ordering

- Z-Ordering **reduces** multi-dimensional space to 1D
- Can use **standard index** (e.g., B+ tree) to index Z value
- E.g., translate XD **range queries** to 1D range queries
  - May still require some additional filtering

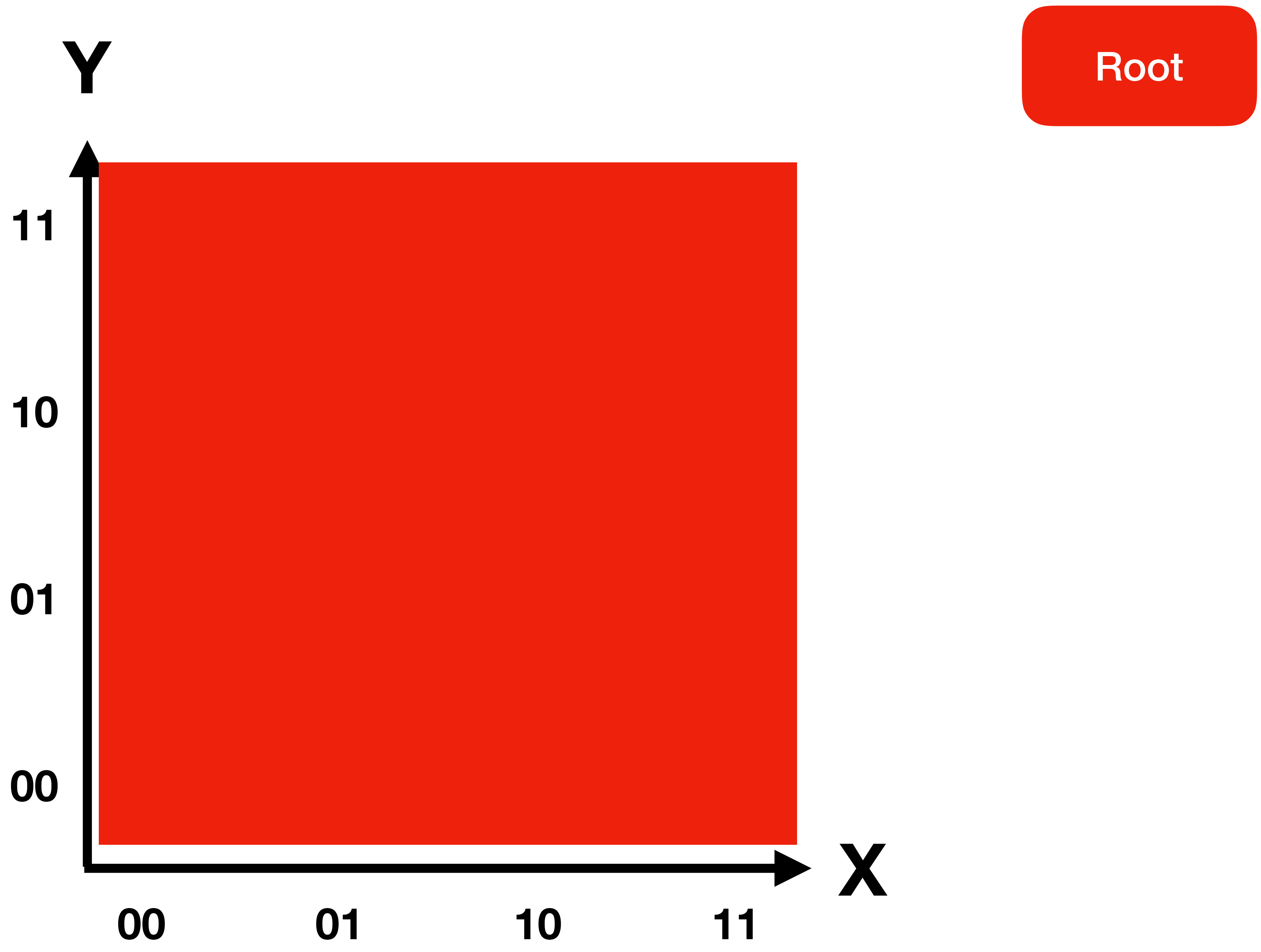
Region Quad Tree

- Z-ordering enables us to store **points** efficiently
- Storing entire **regions** as set of points is inefficient
- **Region quad trees** divide space recursively
  - In 2D: each region is divided into four quadrants
  - Quadrants are associated with child nodes in tree

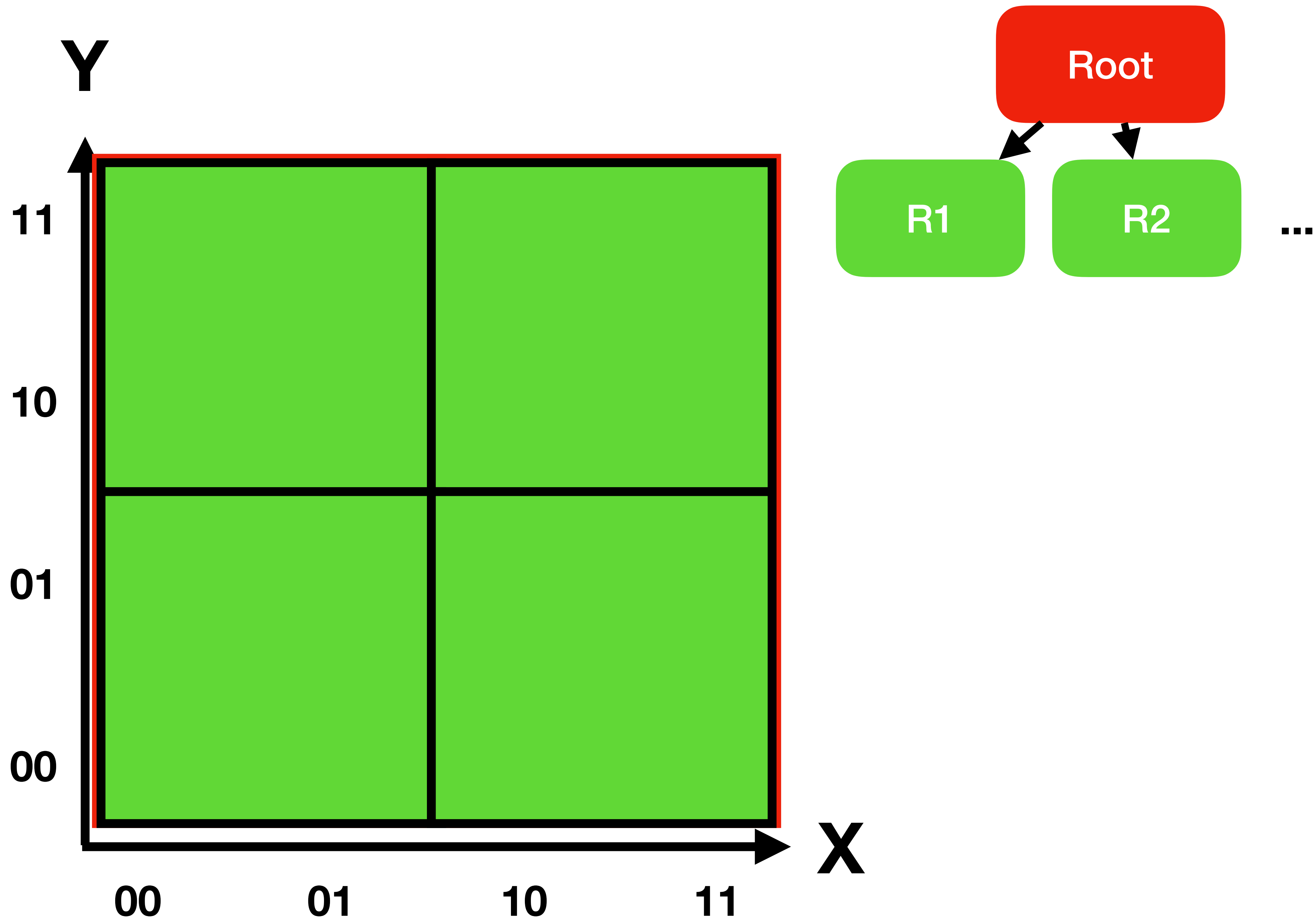
Region Quad Trees



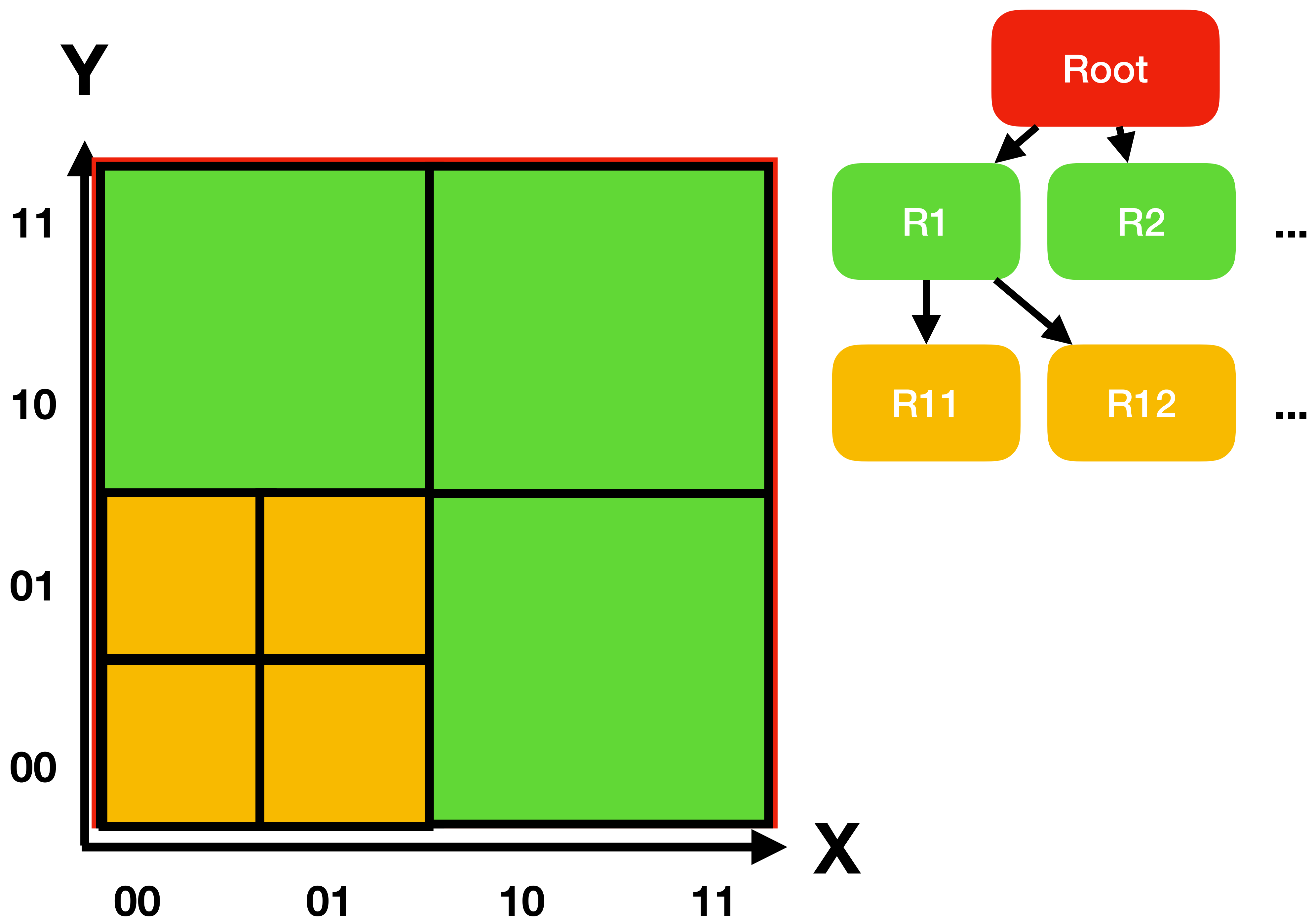
Region Quad Trees



Region Quad Trees



Region Quad Trees



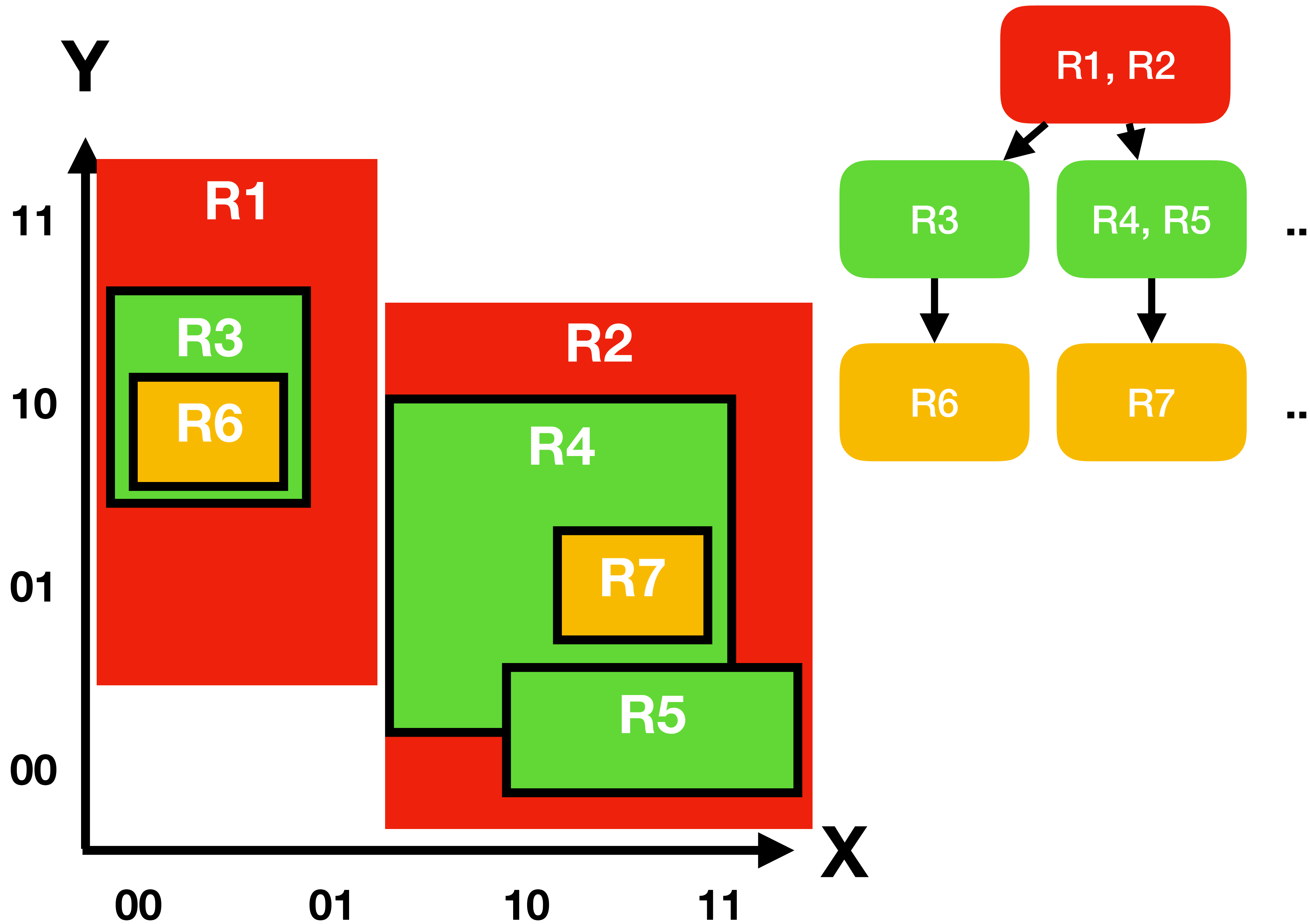
Grid Files

- Region quad trees partition **independently** of data
- This is not optimal if data is highly **skewed**
- Grid files **adapt** space partitioning to data
- More **fine-grained** representation for denser areas
- See **book** for more details

R Trees

- Adaption of **B+ tree** to handle spatial data
- **Search key**: multi-dimensional bounding box
- **Data entries**: (bounding box, rid)
    - Box is smallest box to contain object
- **Index entries**: (bounding box, pointer to child)

R Tree Illustration



R Trees: Lookups

- Compute **bounding box** for query object
  - Can be single point or region
- Start at **root node** of R tree
- Check children **containing** query object
  - May need to check multiple children

R Trees: Insertions

- Compute **bounding box** for inserted object
- Start at **root** node and proceed to leafs
- Select child needing **minimal extension** for object
- **Insert** object at leaf node
  - May have to **enlarge** bounding boxes on path to leaf
  - May have to **rebalance** the tree

R Tree Illustration

How to insert This Object?