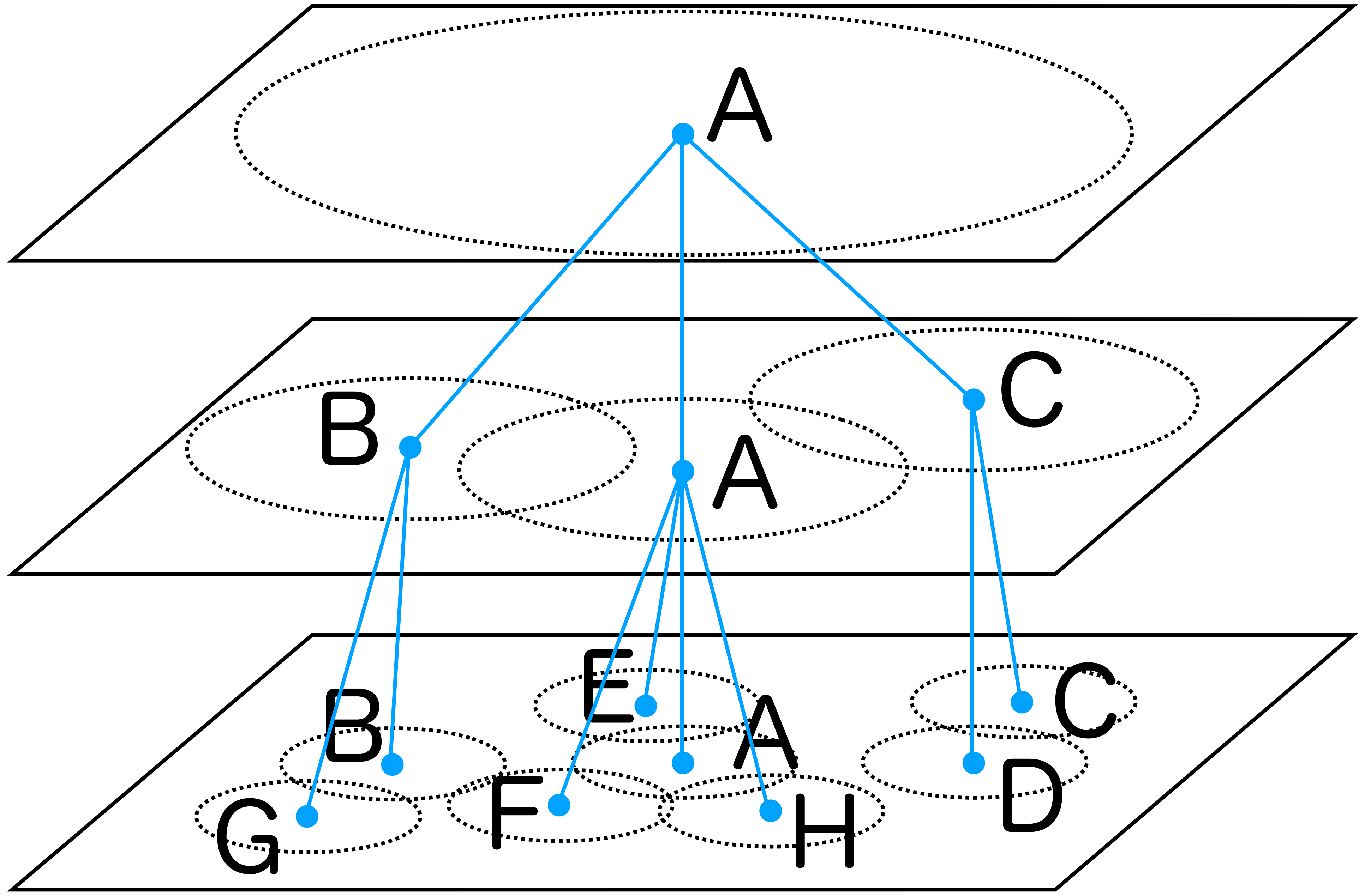
Parallel Cover Trees and their Applications

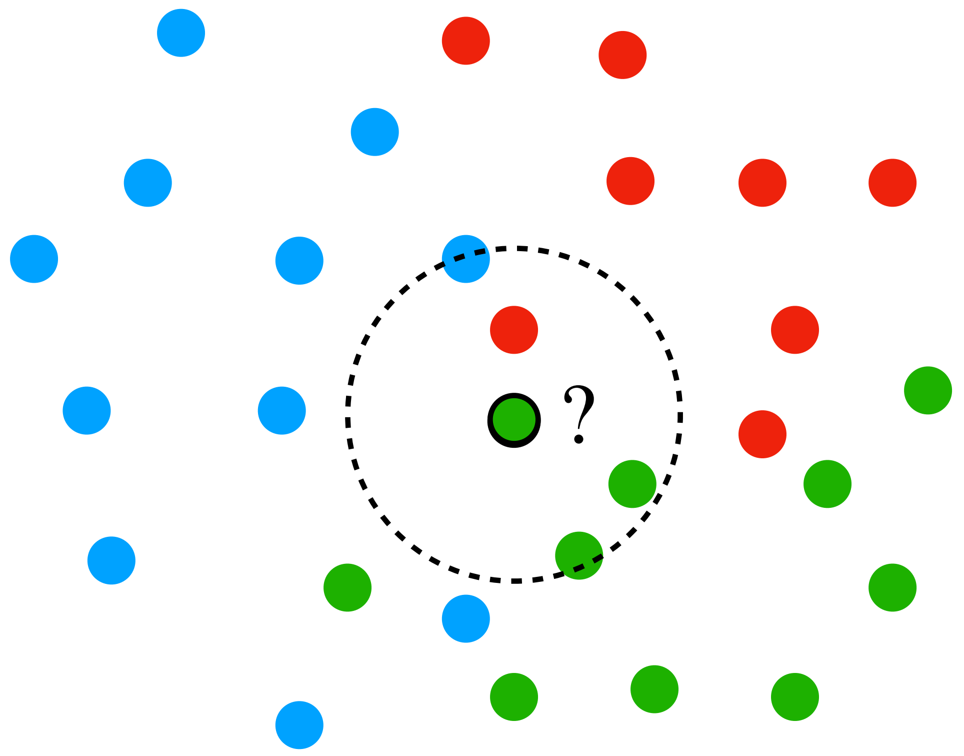
Yan Gu, Zachary Napier, Yihan Sun, **Letong Wang**
UC Riverside

SPAA 2022, Philadelphia

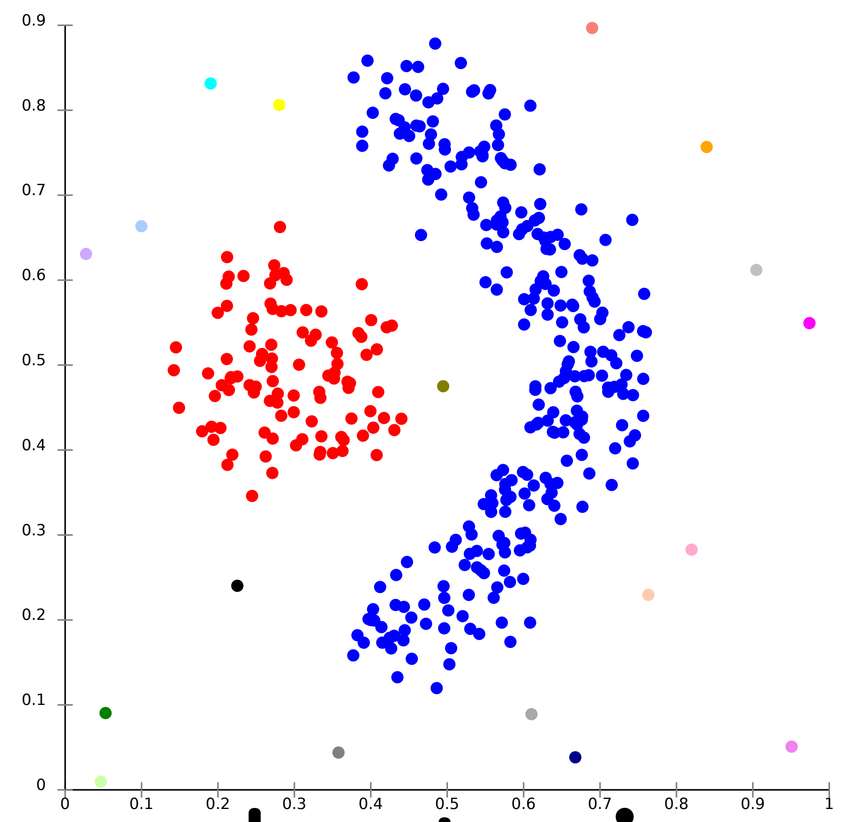


Efficient nearest and k-nearest neighbor search

Nearest and K-nearest neighbor search are common primitives



classification



clustering

KNN classification

K-means Clustering

Density-Based Clustering

Single-Linkage Clustering

Recommendation systems

Euclidean Minimum Spanning Tree

Information retrieval

Bichromatic Closest Pair

Preliminaries: metric space

A metric space (X, d_X) is defined on a set X and with a distance function

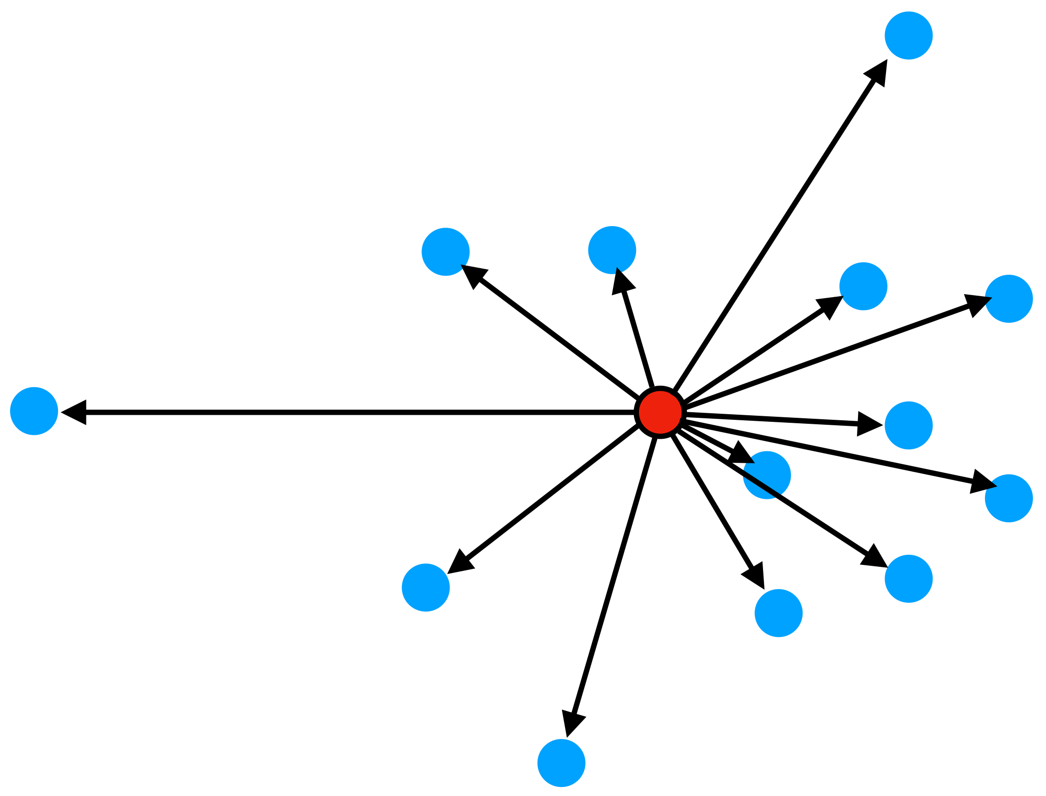
$d_X : X \times X \to \mathbb{R}^*$ that satisfies properties: for $x, y, z \in X$,

$d_X(x, y) = 0 \iff x = y$ identity of indiscernibles

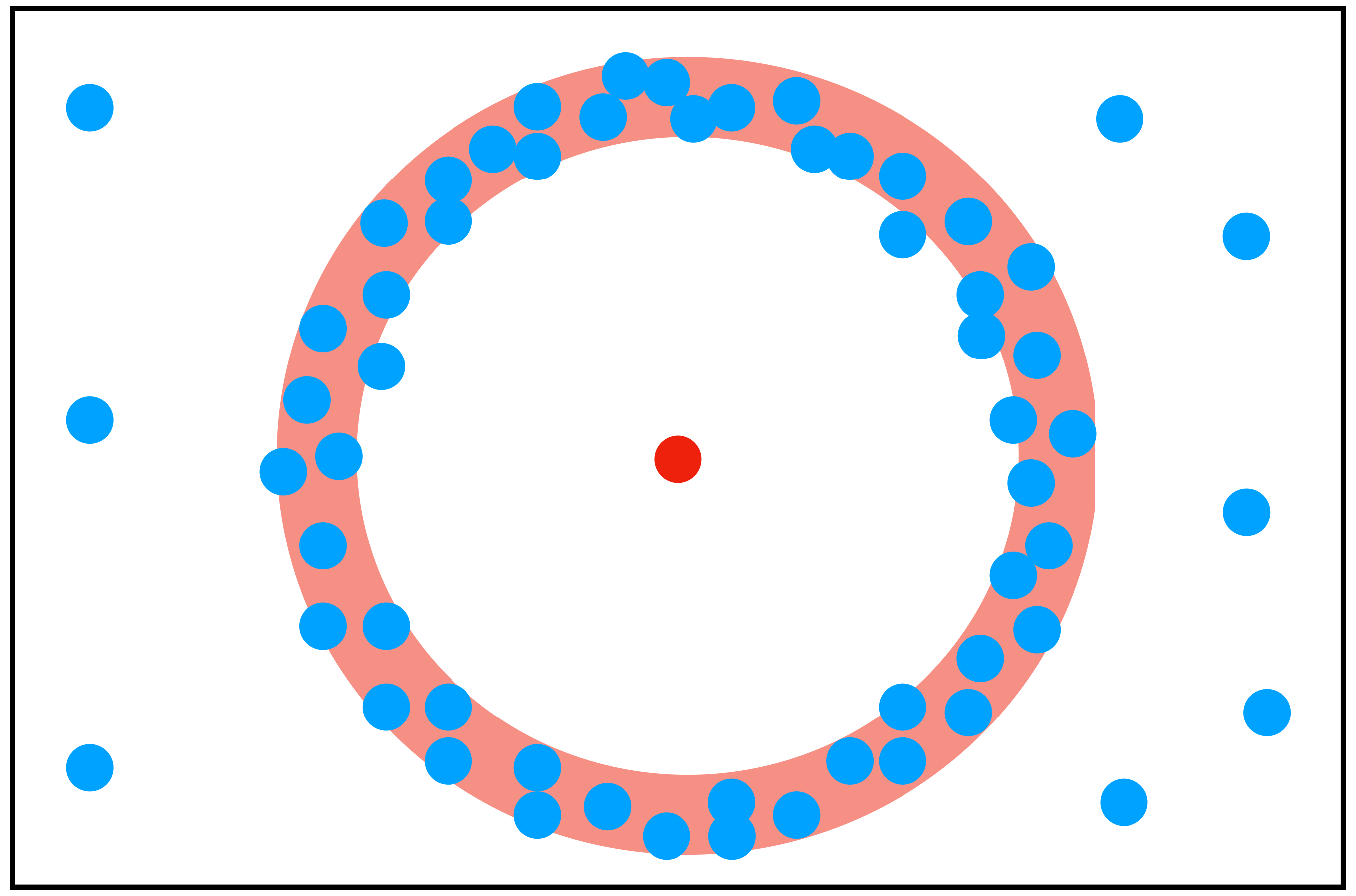
$d_X(x, y) = d_X(y, x)$ symmetric

$d_X(x, y) \leq d_X(x, z) + d_X(z, y)$ triangle inequality

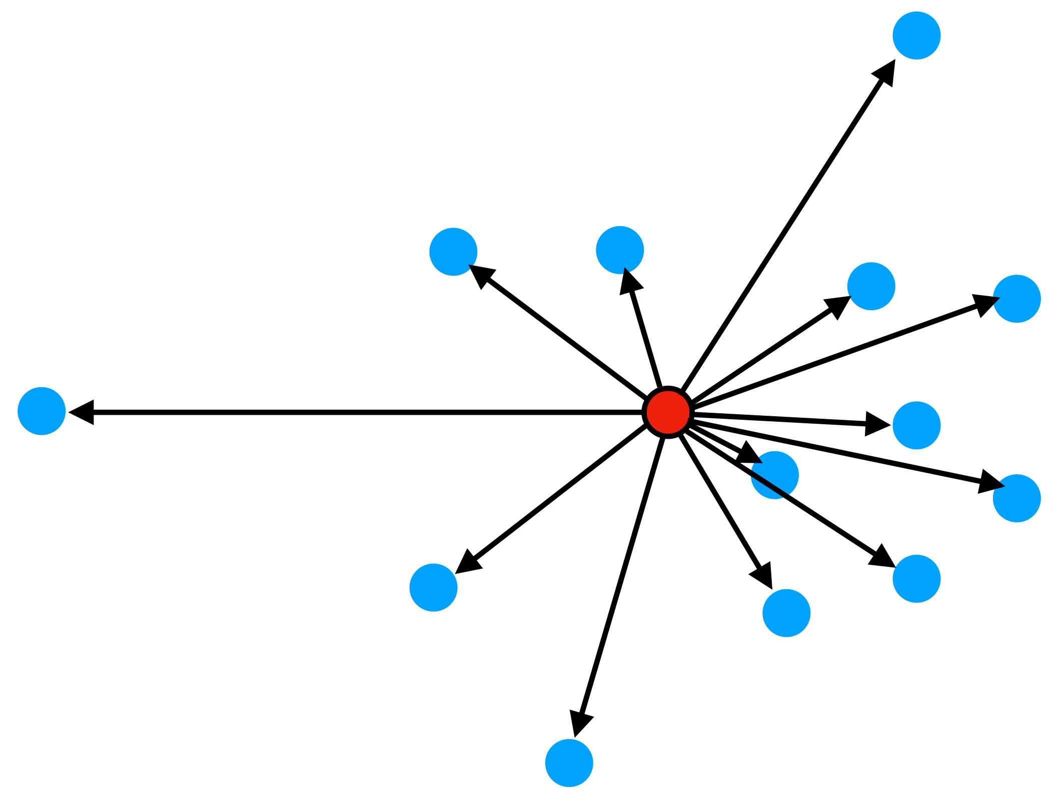
Exact NNS on general metrics is challenging



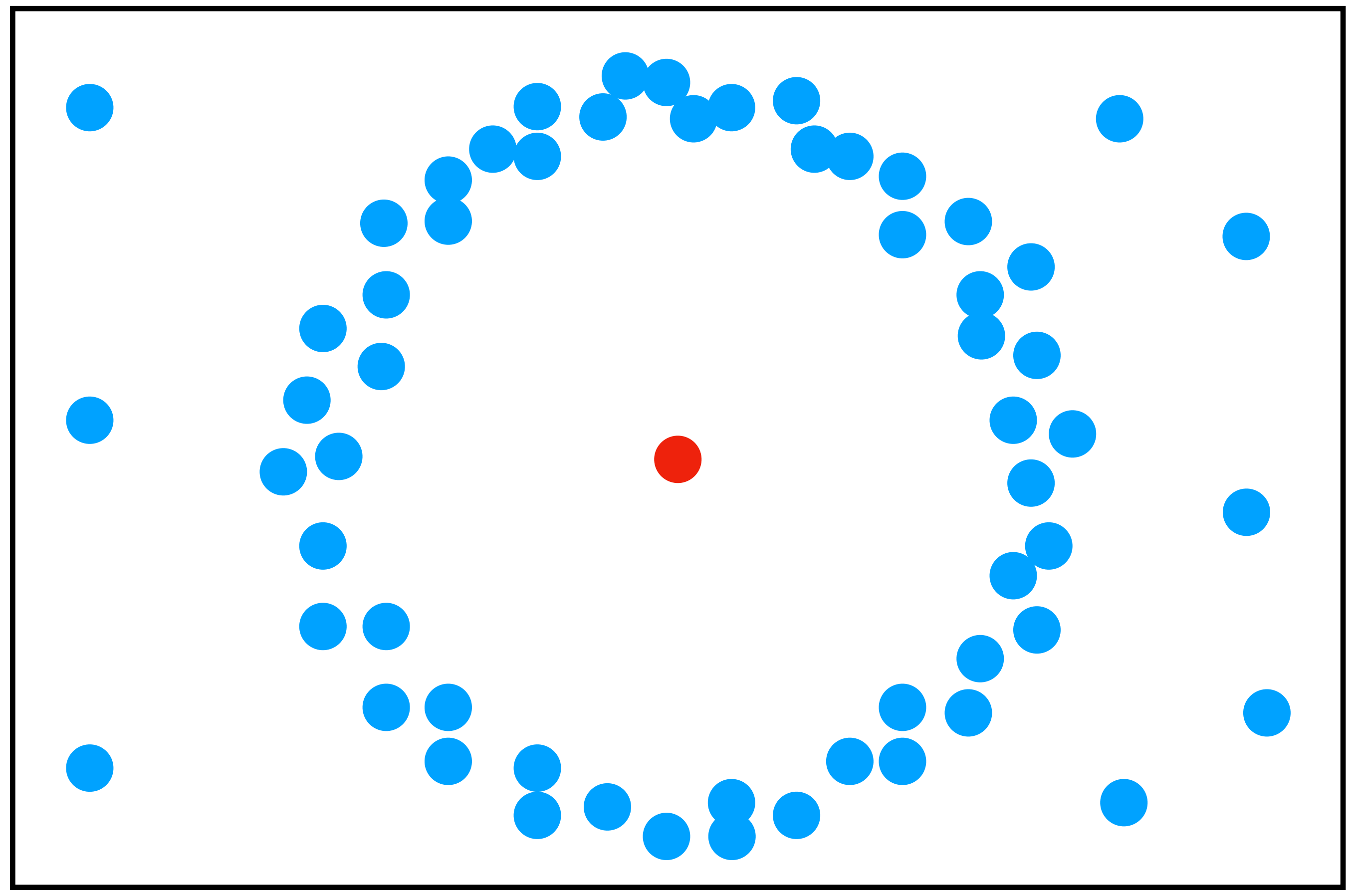
Can we do it in sub-linear work on arbitrary input?

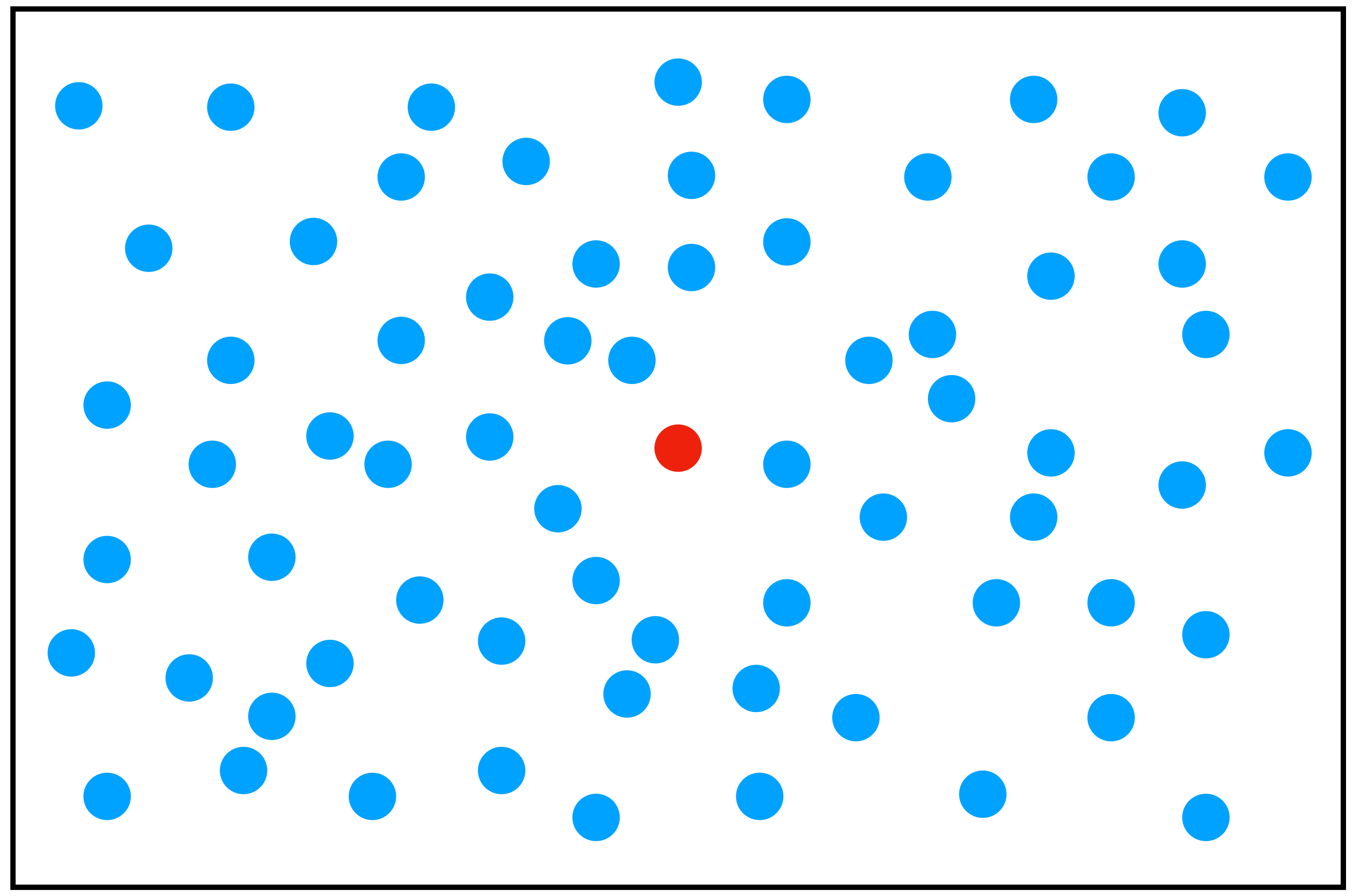


Exact NNS on general metrics is challenging



Can we do it in sub-linear work on arbitrary input?





How to measure how “good” a distribution is?

Metrics in real-world usually have nice properties to explore

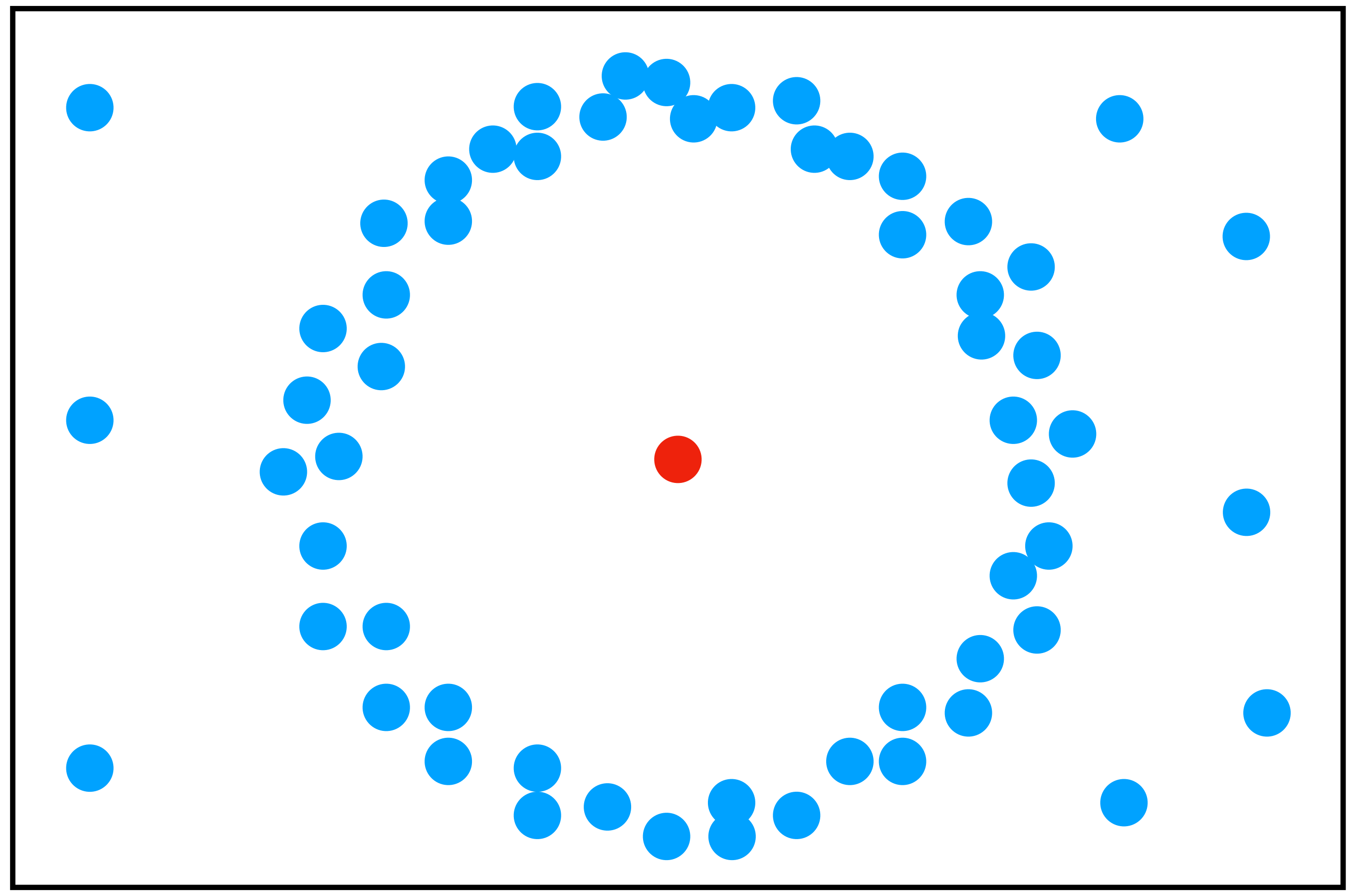
Low “Expansion Rate”

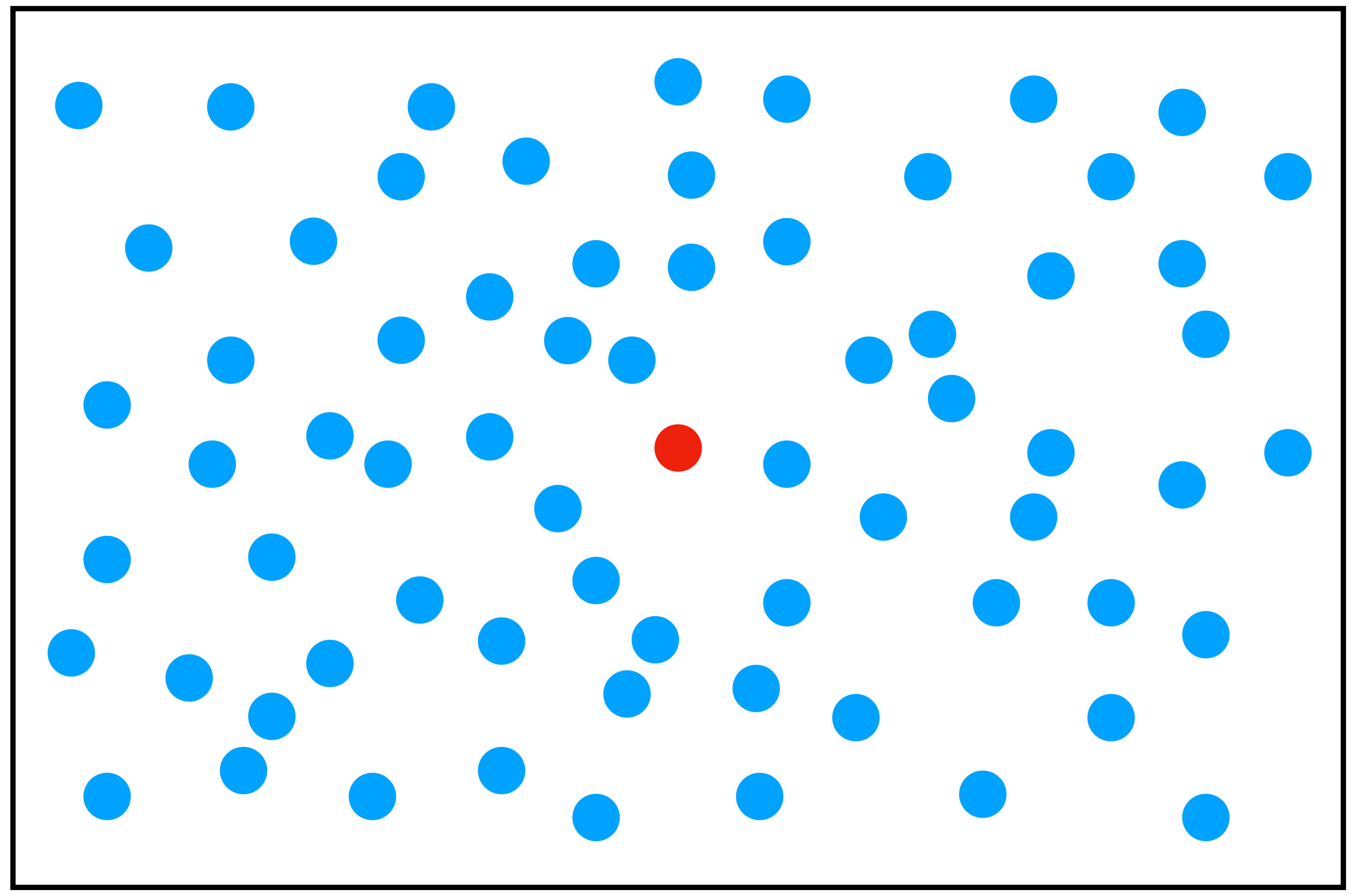
Bounded “Aspect Ratio”

Metrics in real-world usually have nice properties to explore

Low “Expansion Rate”

Bounded “Aspect Ratio”





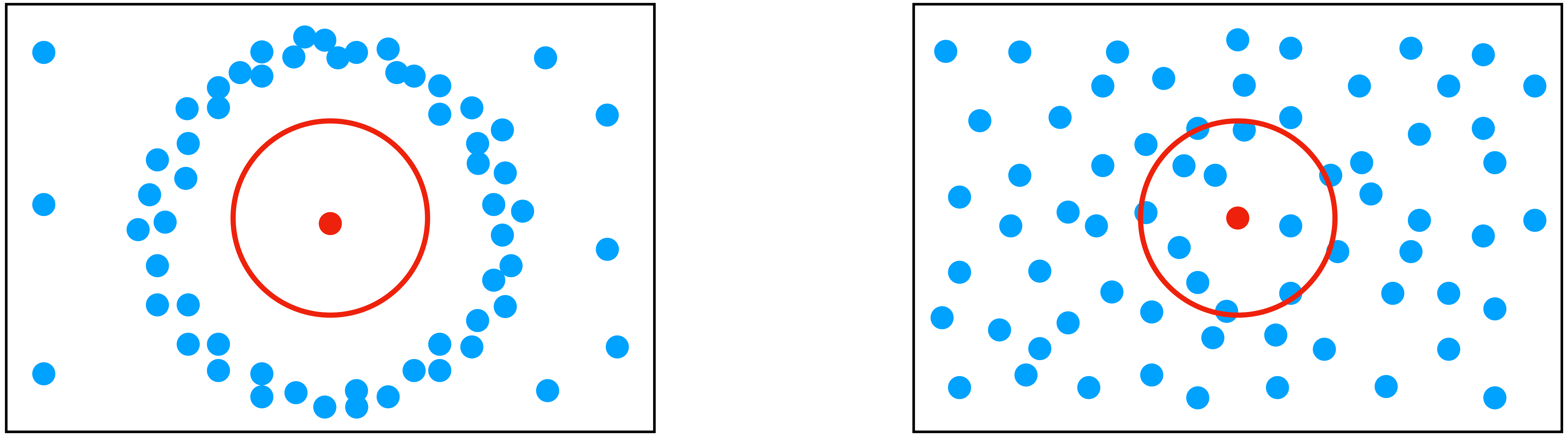
Low “Expansion Rate”[Karger, Ruhl 2002]

For each point, when doubling a radius, the number of points within the new radius is at most c times that within the original radius

low expansion rate $\rightarrow c = O(1)$

lower $c \rightarrow$ more smoothly the density changes

$c = \Theta(n)$ $c = O(1)$



[Karger, Ruhl 2002] David R. Karger, and Matthias Ruhl. "Finding nearest neighbors in growth-restricted metrics." ACM symposium on Theory of computing(STOC). 2002.

Metrics in real-world usually have nice properties to explore

Low “Expansion Rate”

Bounded “Aspect Ratio”

Bounded "Aspect Ratio"

Aspect ratio is defined as the ratio of the maximum distance over smallest distance

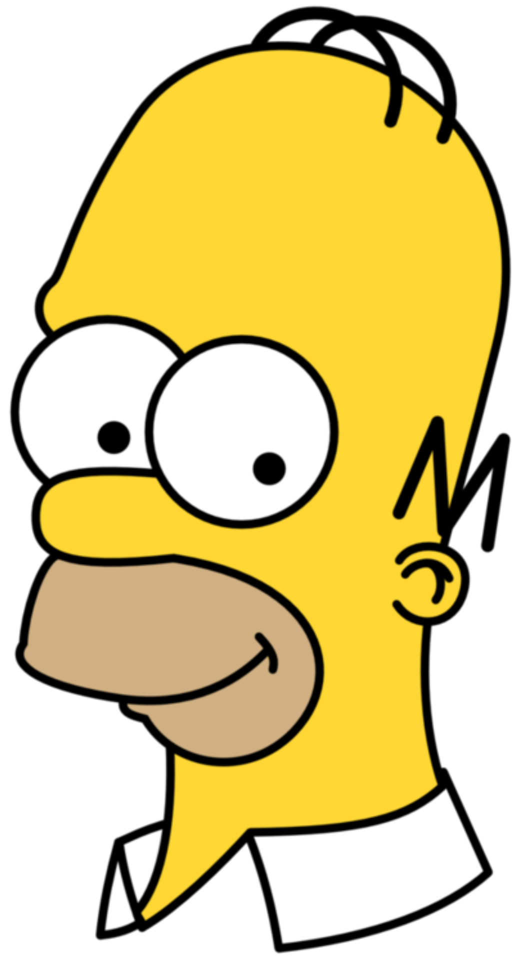
$$\Delta = \frac{\max\{d(x,y)\,|\,x,y \in X\}}{\min\{d(x,y)\,|\,x,y \in X\}}$$

Earth

1.27×10^7m

source: NASA

Hair



7×10^{-5}m

$$\Delta = \frac{1.27 \times 10^7}{7 \times 10^{-5}} < 2 \times 10^{11}$$

$$n \geq 1 \times 10^6$$

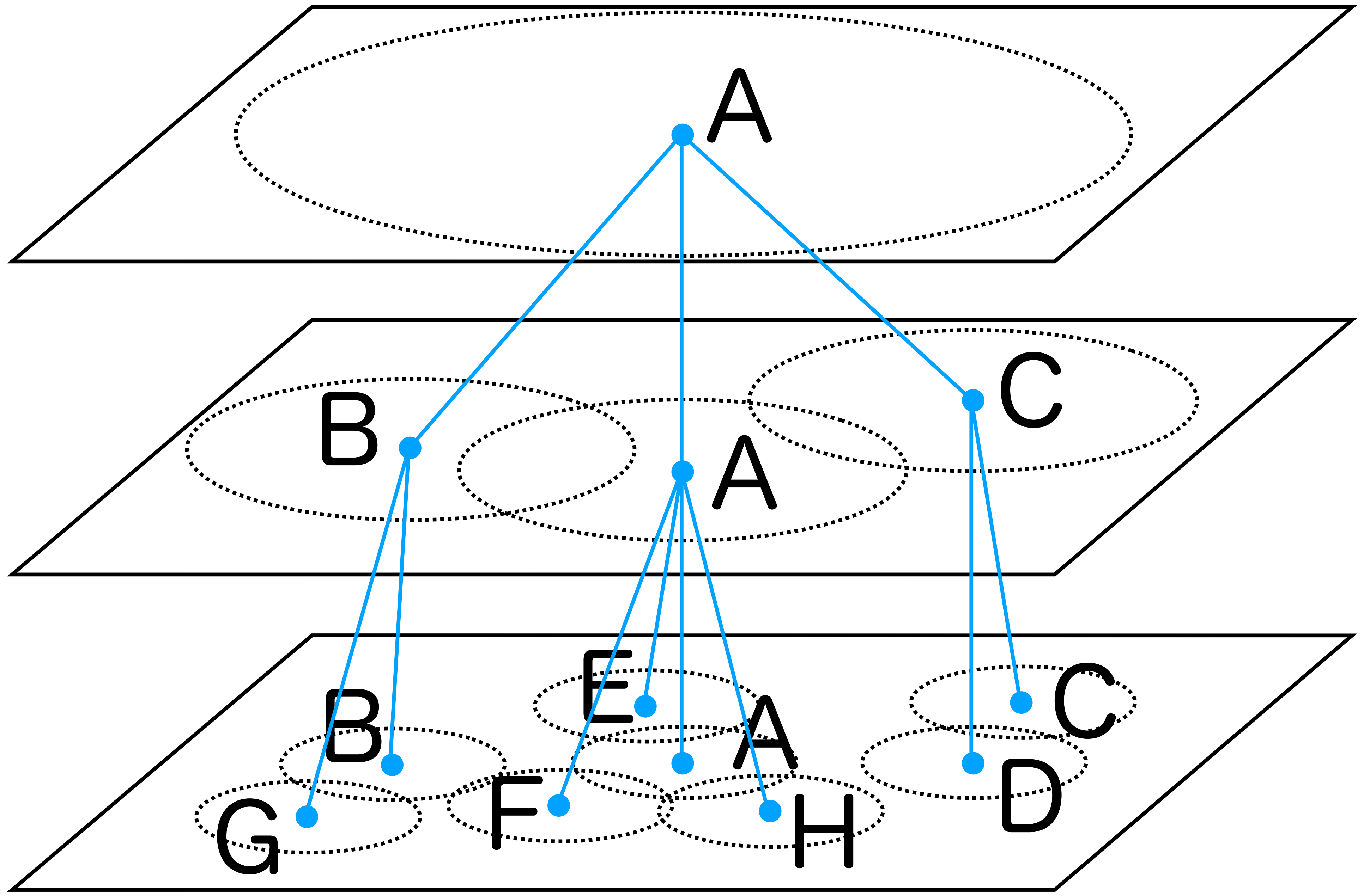
$$\Delta < n^2$$

$\Delta < n^K$ for some constant $K > 0$

Cover tree [BKL 2006] is a canonical solution supporting NNS

Low "Expansion Rate"

Bounded "Aspect Ratio"



Nesting

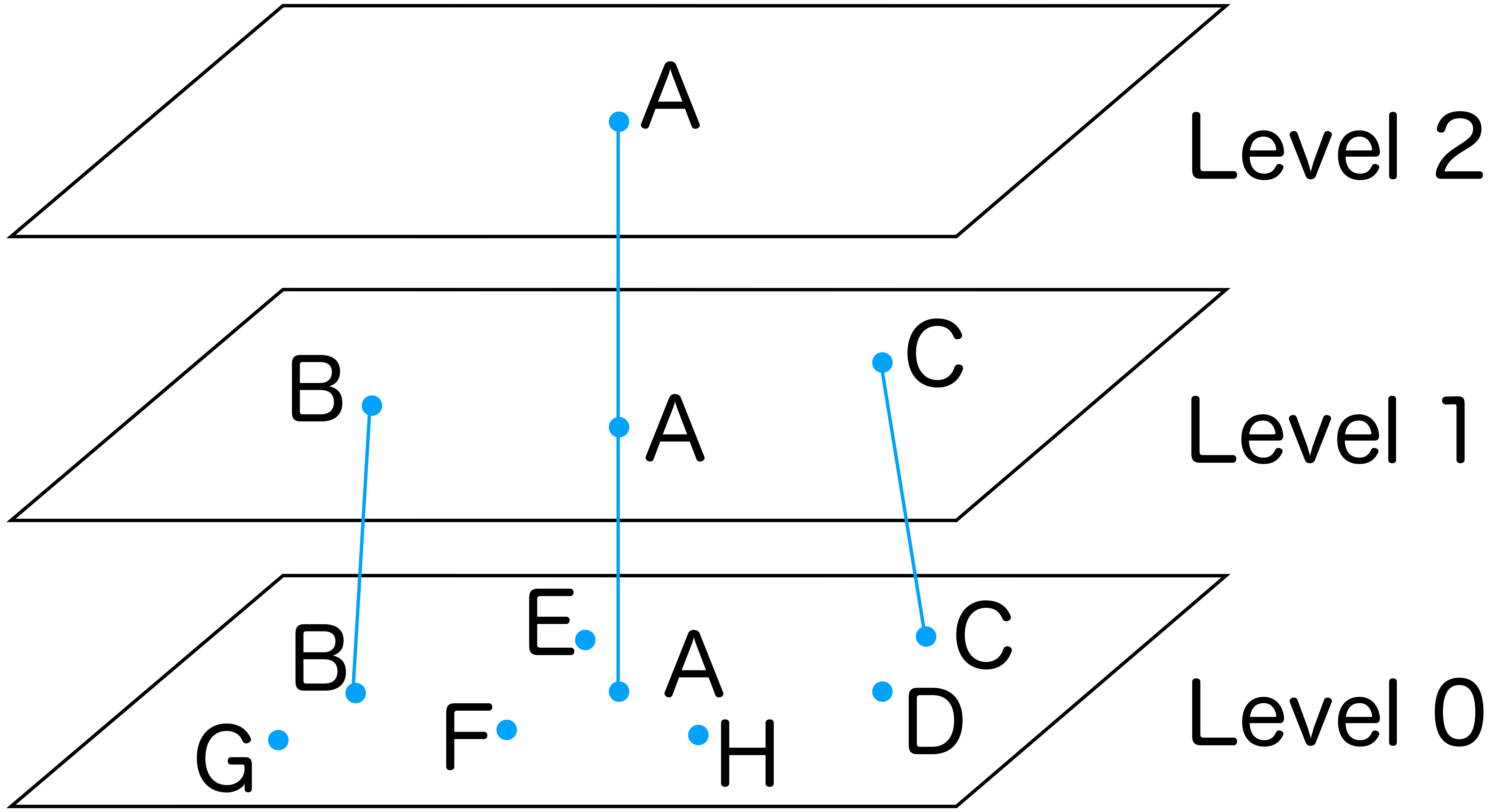
Covering

Separation

[BKL 2006] Beygelzimer, Alina, Sham Kakade, and John Langford. "Cover trees for nearest neighbor." Proceedings of the 23rd International Conference on Machine Learning (ICML). 2006.

Cover tree

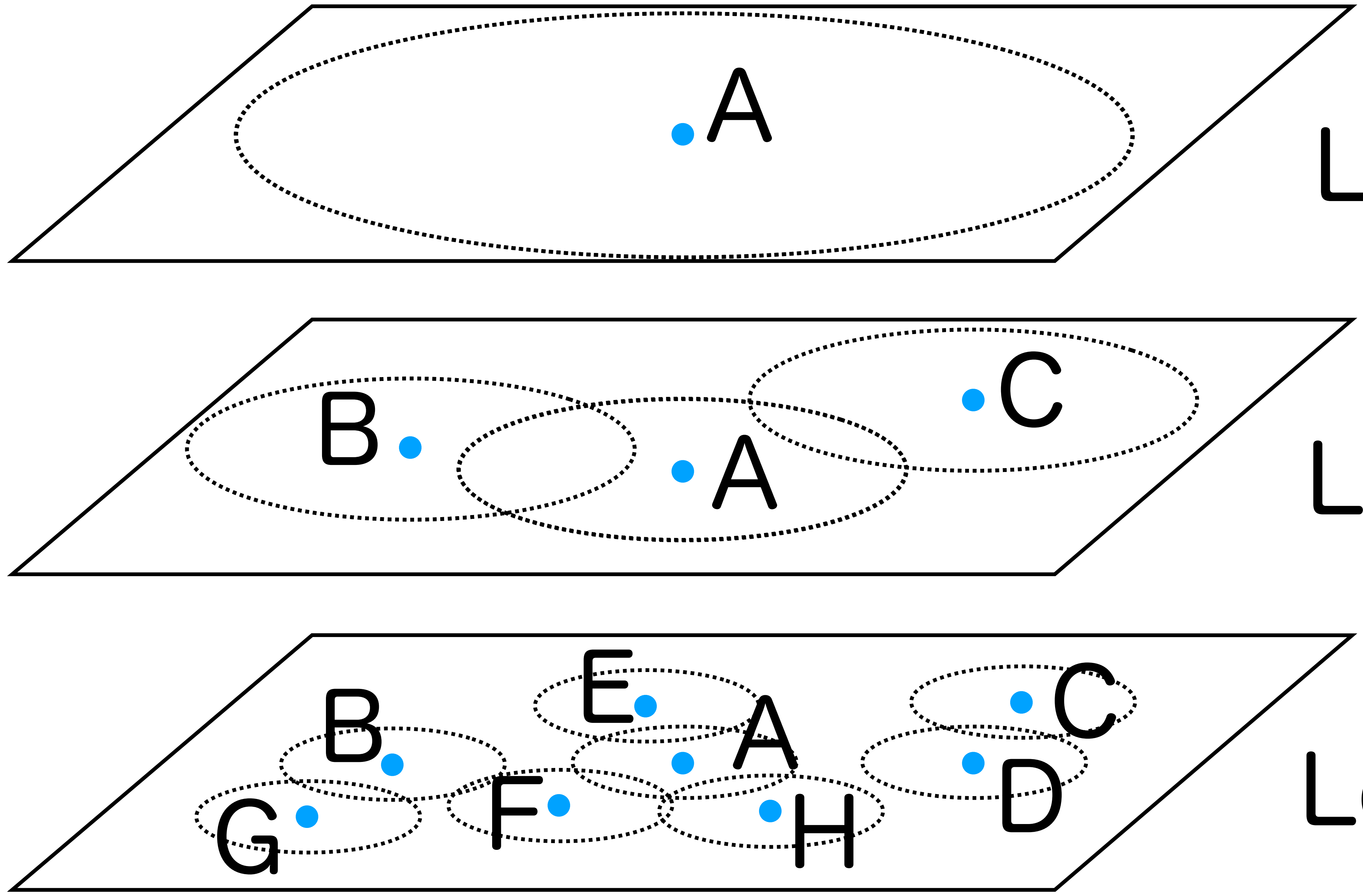
Nesting The bottom level contains all the points
The tree nodes at one level is a subset of nodes at the lower level



What is the parent of tree node?

Cover tree

Covering A node can cover the nodes in the lower level within the covering radius.



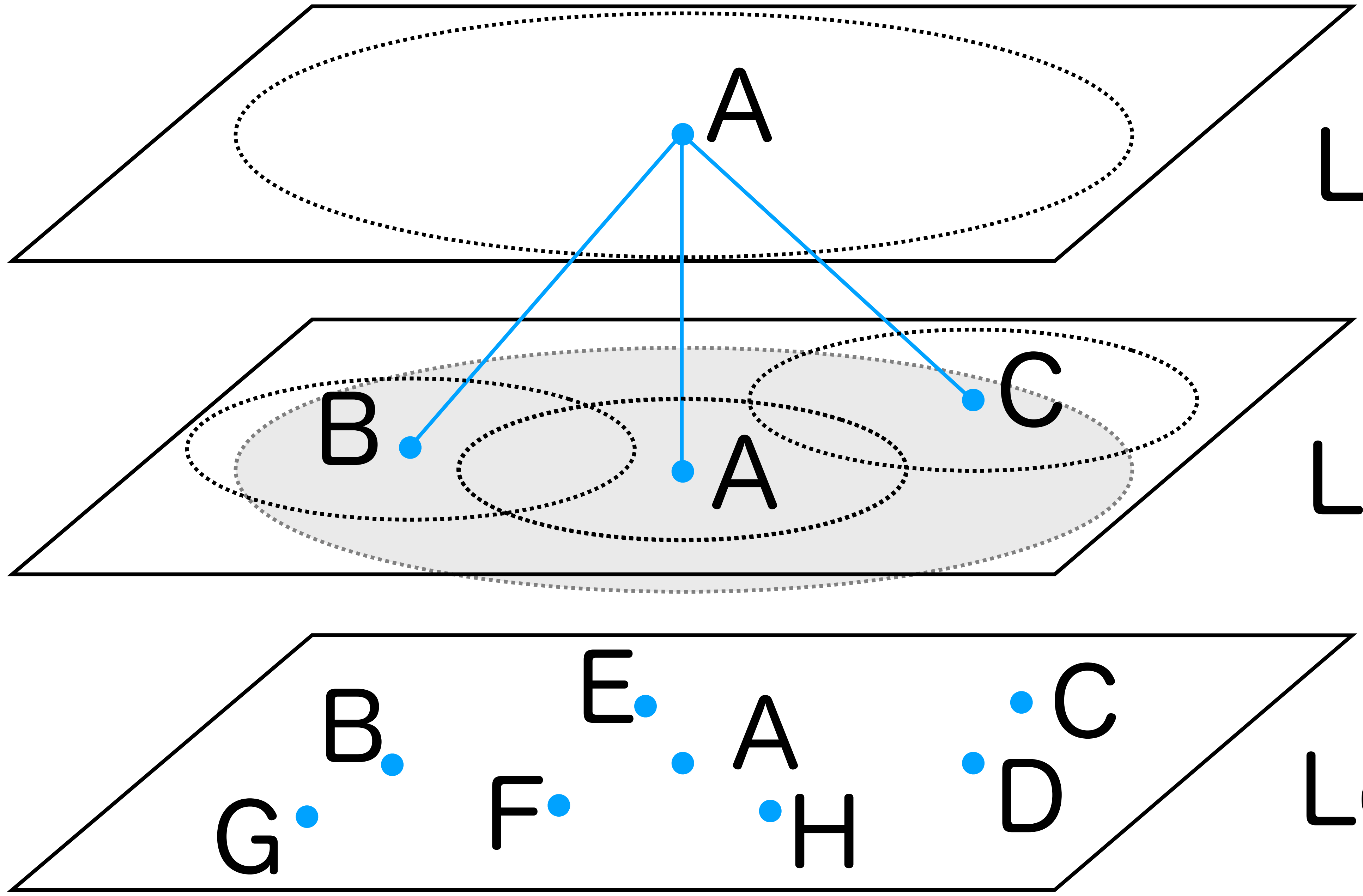
Level 2, r = 2^2

Level 1, r = 2^1

Level 0, r = 2^0

Cover tree

Covering A node can cover the nodes in the lower level within the covering radius. A node is covered by some node at a higher level.



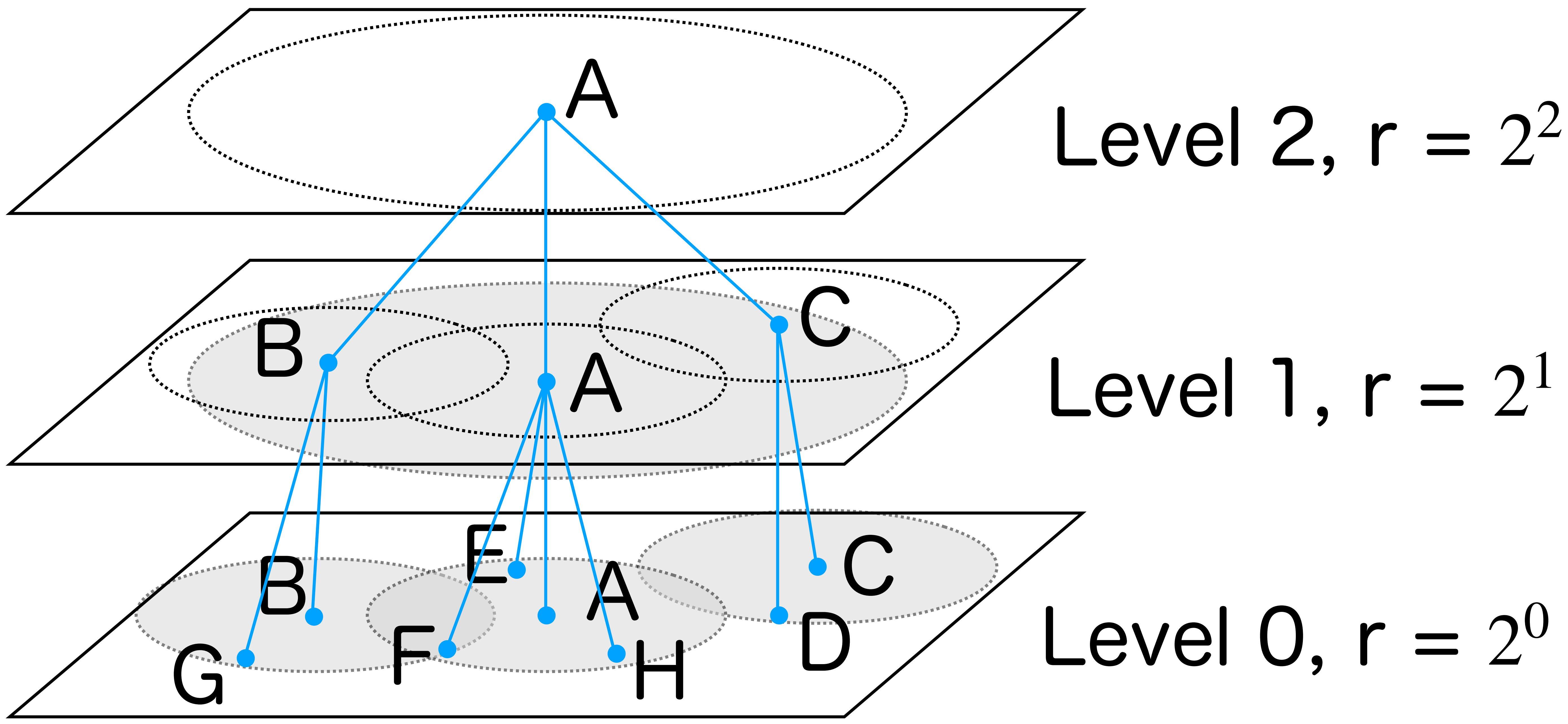
Level 2, $r = 2^2$

Level 1, $r = 2^1$

Level 0, $r = 2^0$

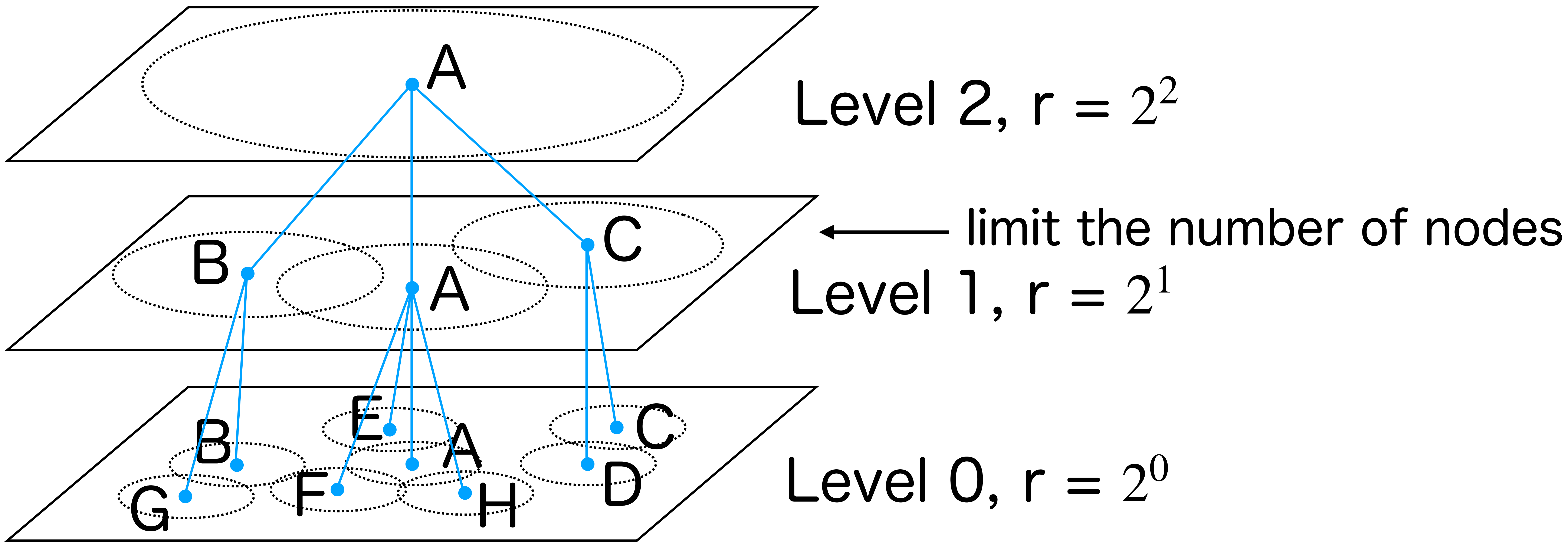
Cover tree

Covering A node can cover the nodes in the lower level within the covering radius.
A node is covered by some node at a higher level.



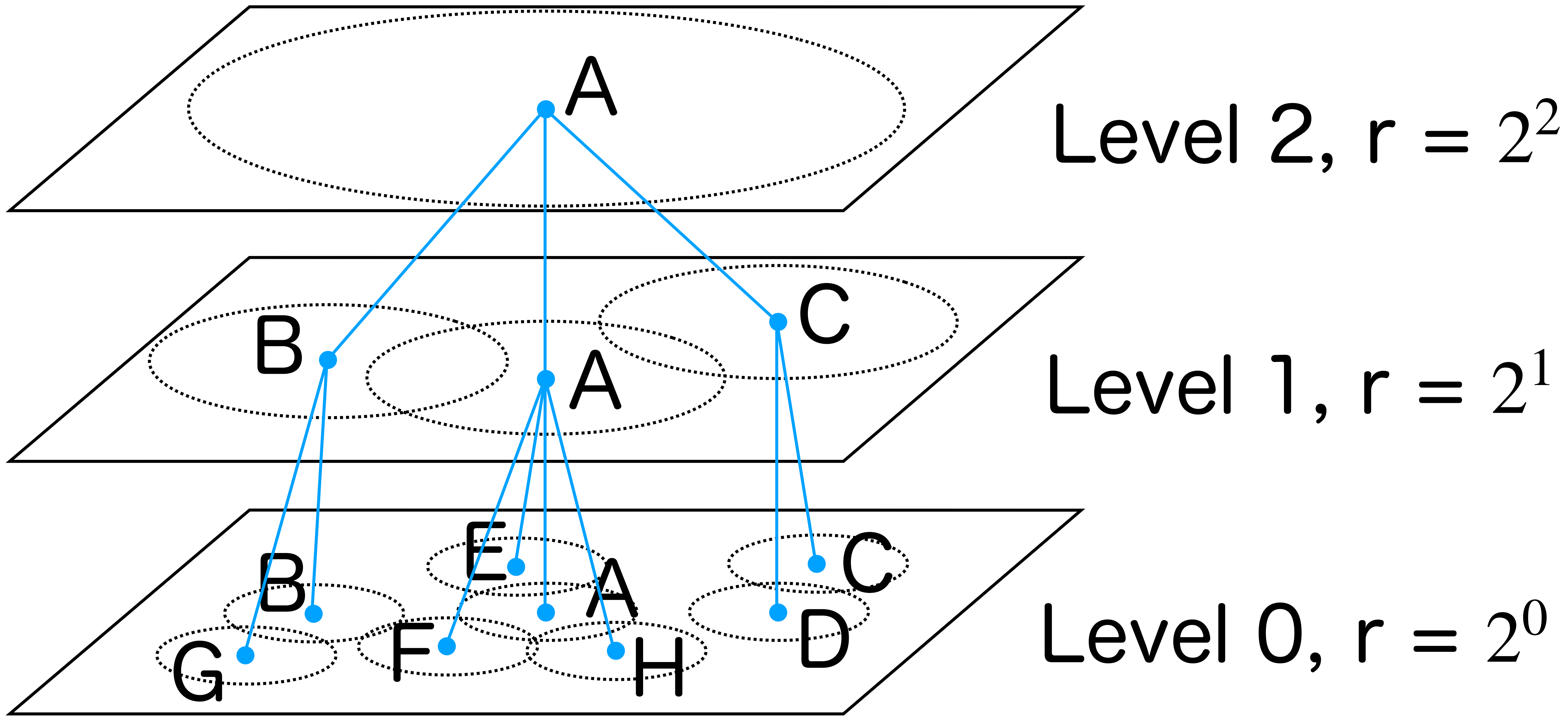
Cover tree

Separation All tree nodes at the same level are separated by the covering radius



Cover tree

Nesting **Covering** **Separation**



Cover tree

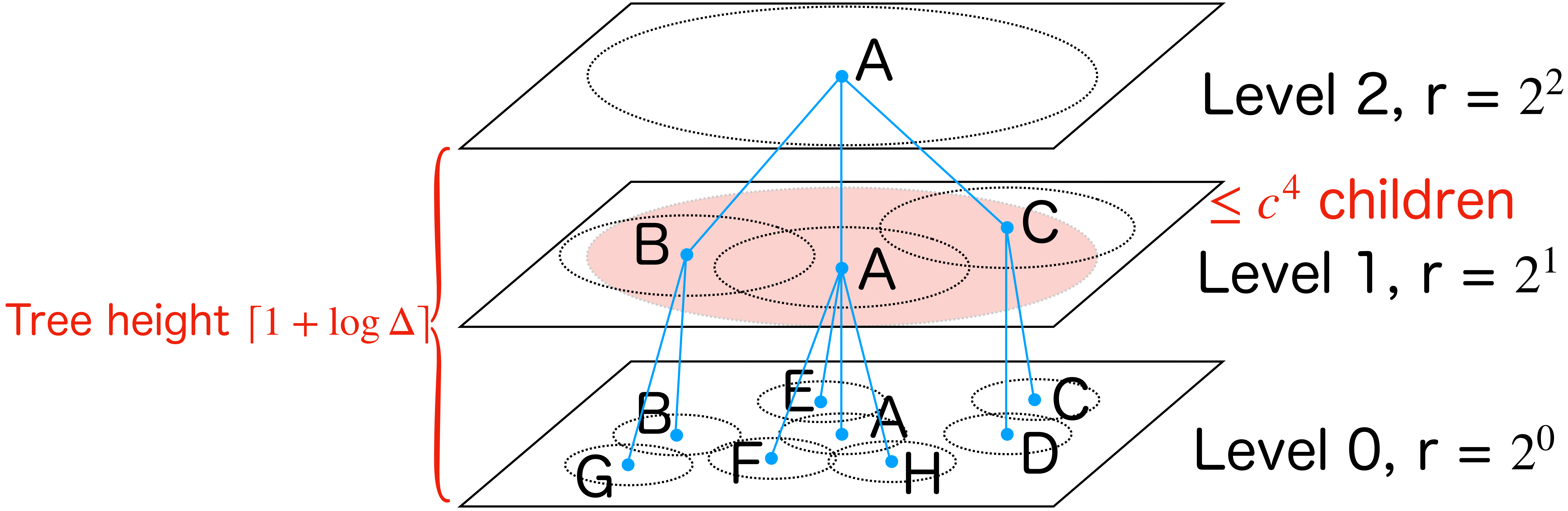
Nesting

Covering

Separation

The number of children of any node is no more than c^4

The height of the tree $\leq \lceil 1 + \log(\Delta) \rceil$



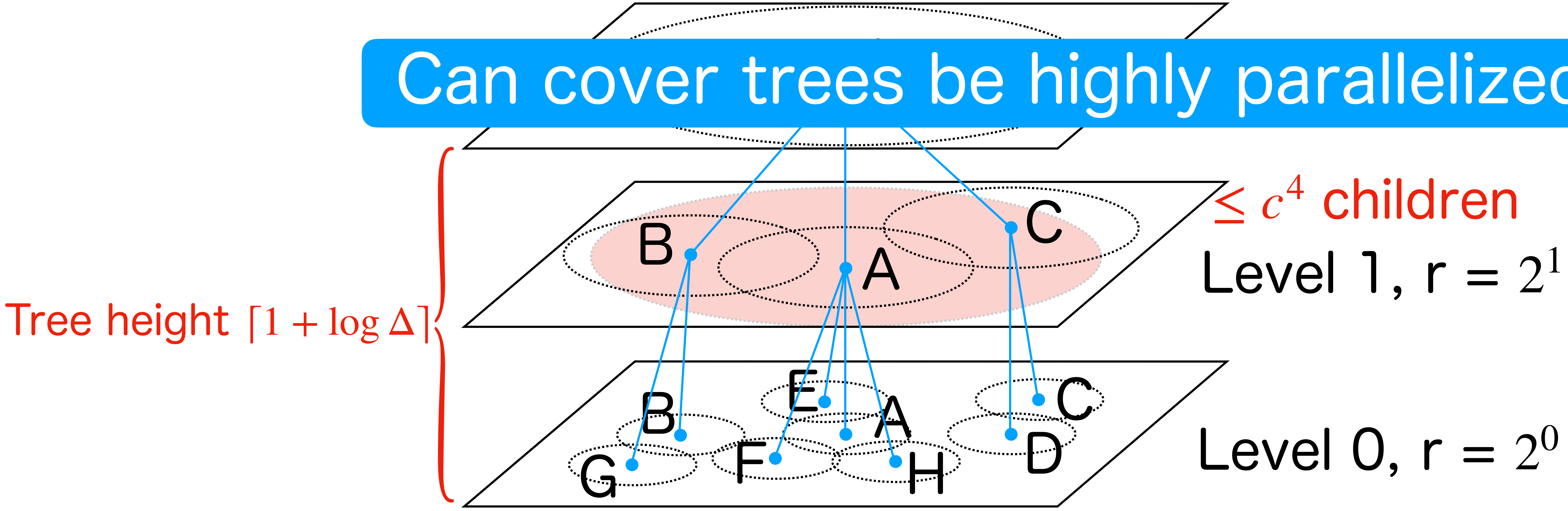
Insert/delete/NNS-query has logarithmic cost

The number of children of any node is no more than c^4

Low "Expansion Rate" → c is a constant → Bounded degree

The hight of the tree is no more than $\lceil 1 + \log(\Delta) \rceil$

Bounded "Aspect Ratio" → $\Delta < n^K$ → Logarithmic height



Parallel updates on a cover tree are hard

Two papers “claimed” they parallelized the cover tree, but neither preserves the theoretical bound

Sharma and Joshi’s algorithm [2010] has no bound.

Izbicki and Shelton’s version [2015] relaxes the separation property (query is linear).

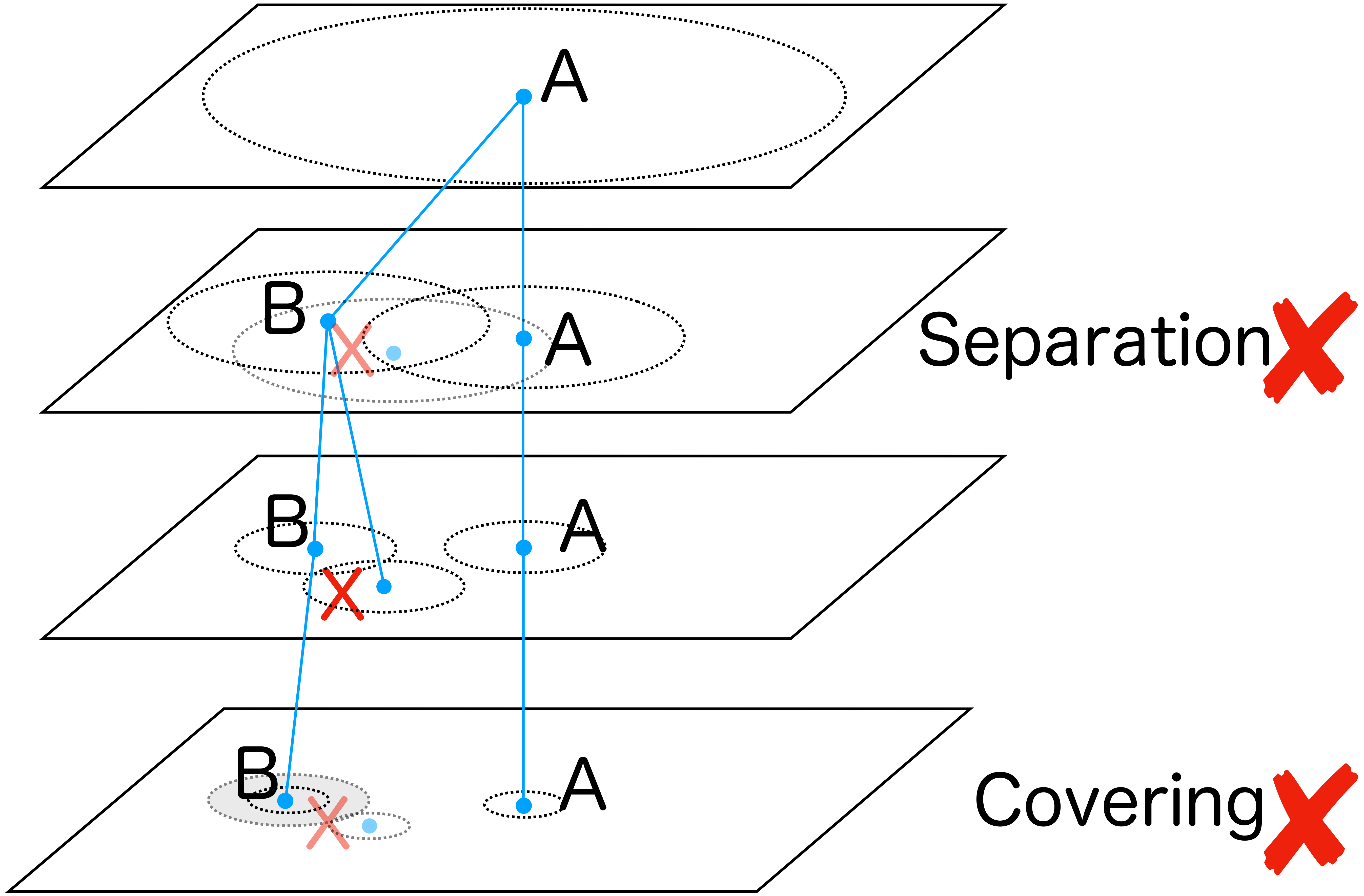
Parallelizing cover trees has been open for 15 years.

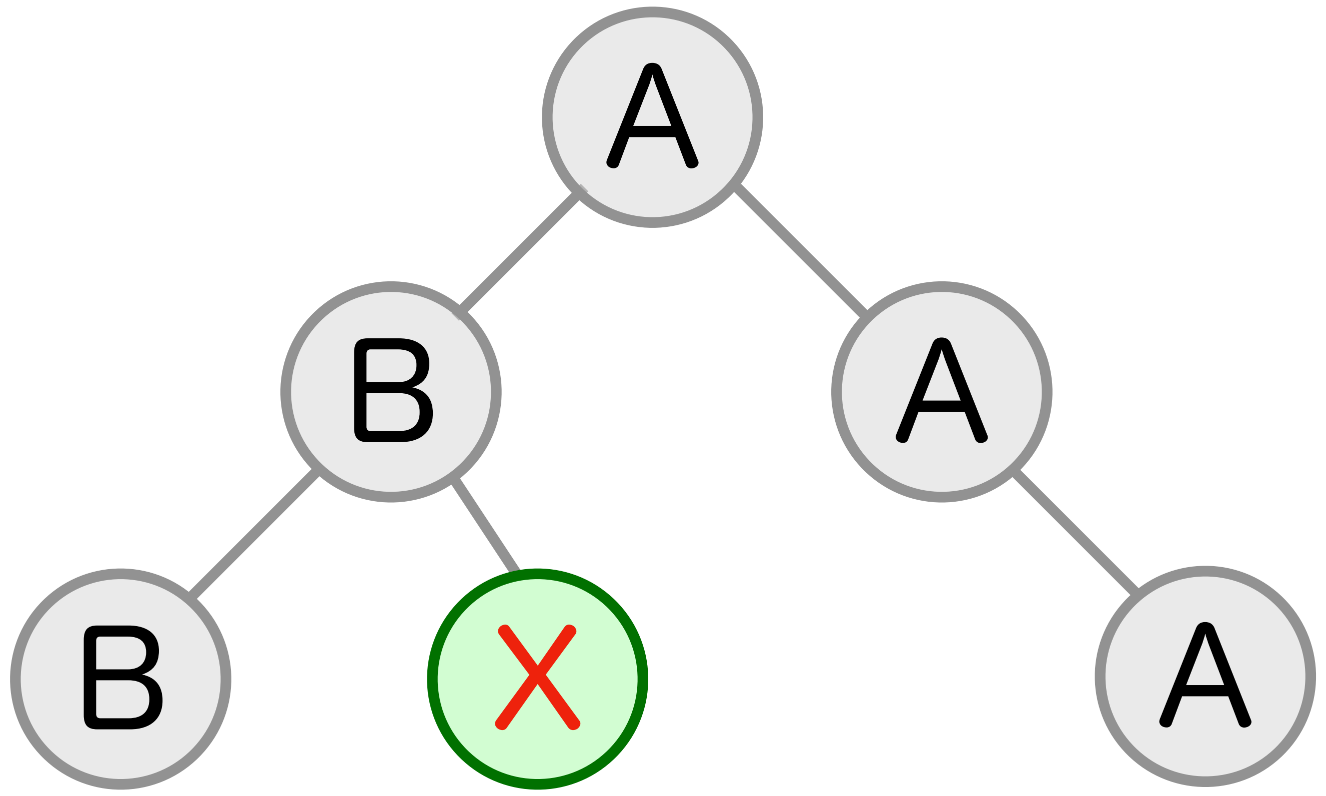
No known other **parallel** data structure have the same theoretical guarantee.

Why parallel is so hard?

Parallel insertion on a cover tree is hard

Insert X and Y to a cover tree with two points
Insert X independently

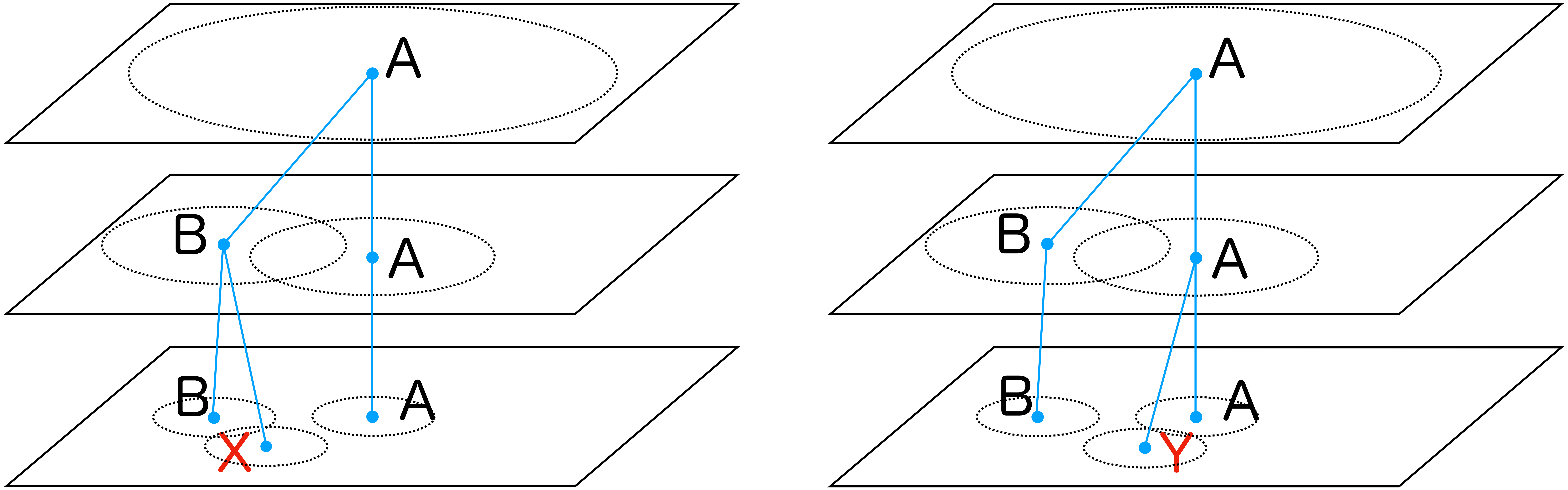




Parallel insertion on a cover tree is hard

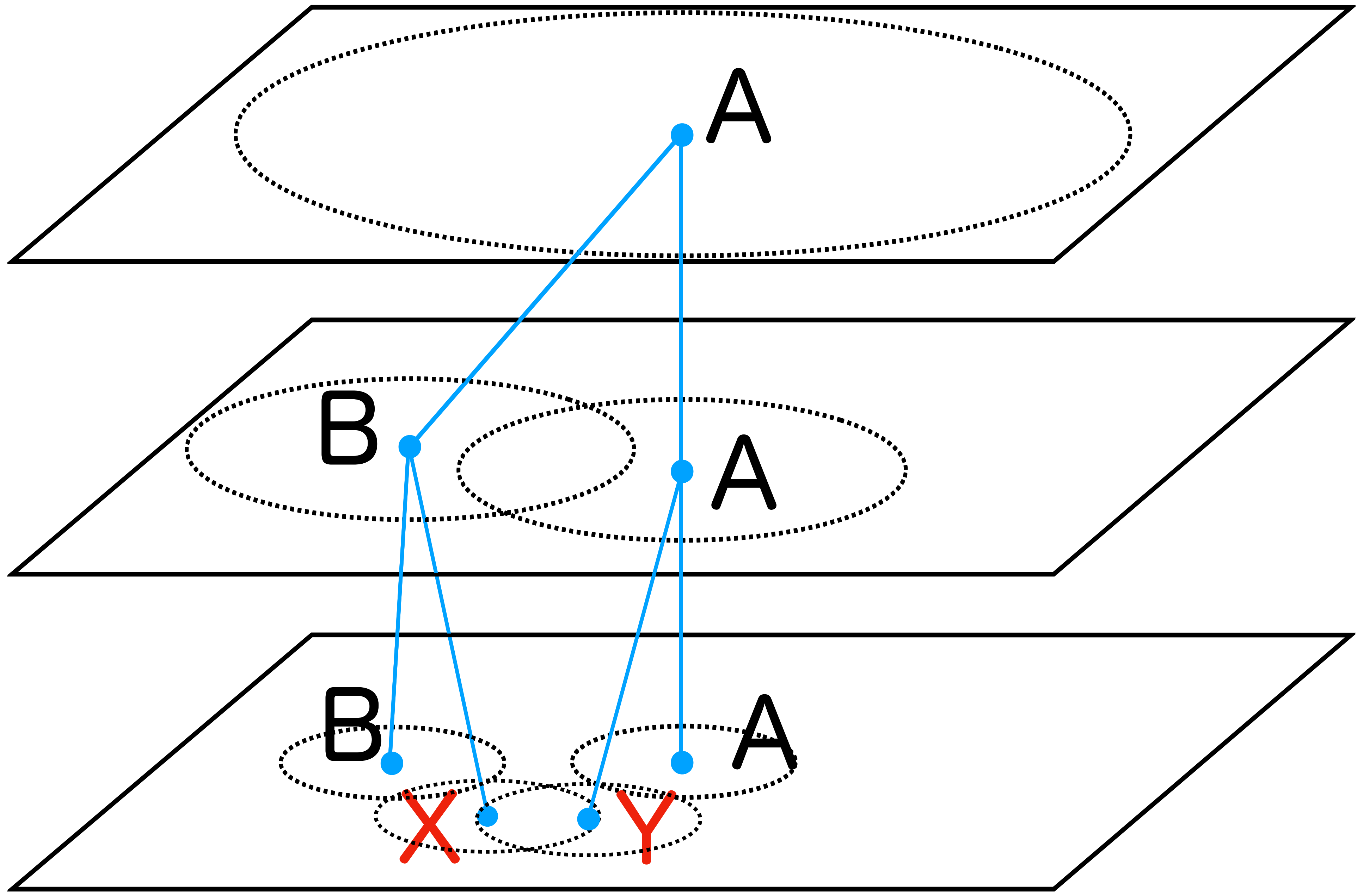
Insert X and Y to a cover tree with two points

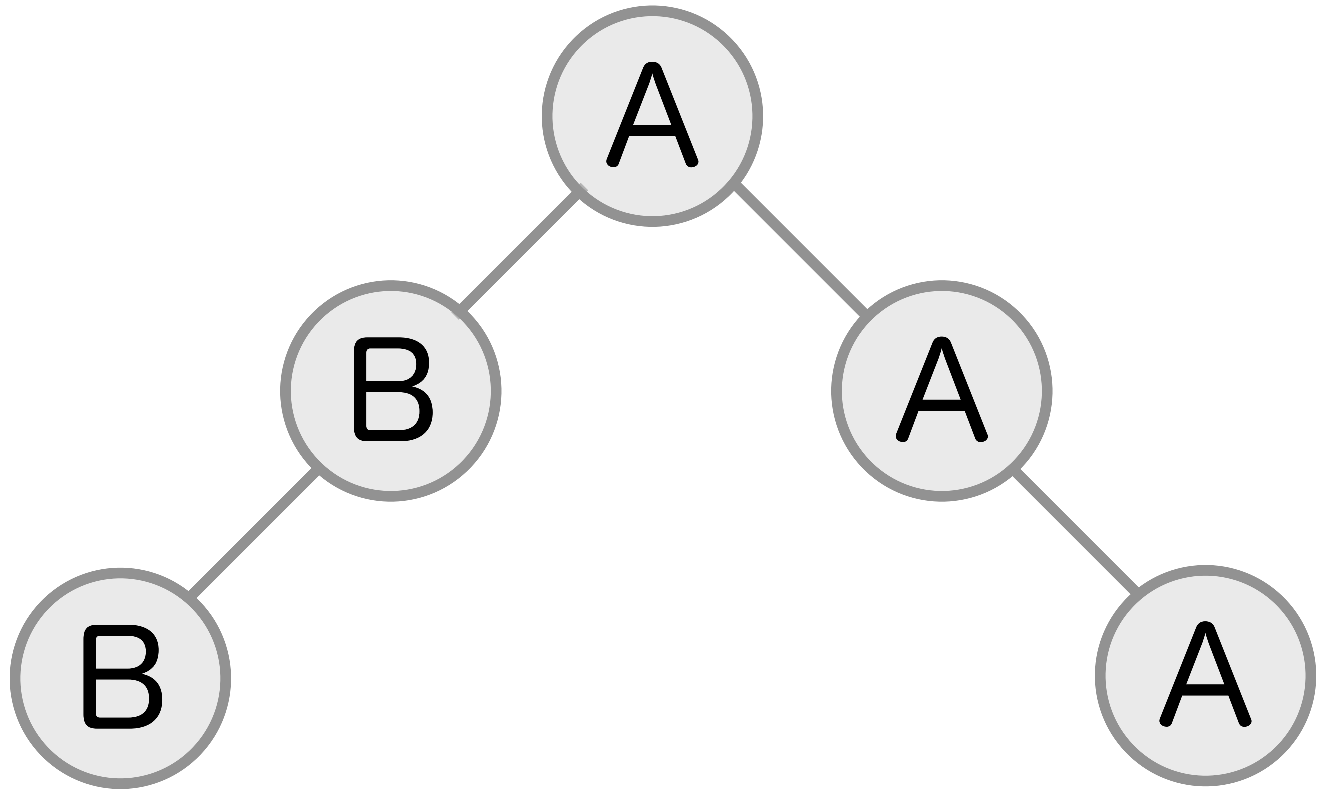
Insert Y independently



Parallel insertion on a cover tree is hard

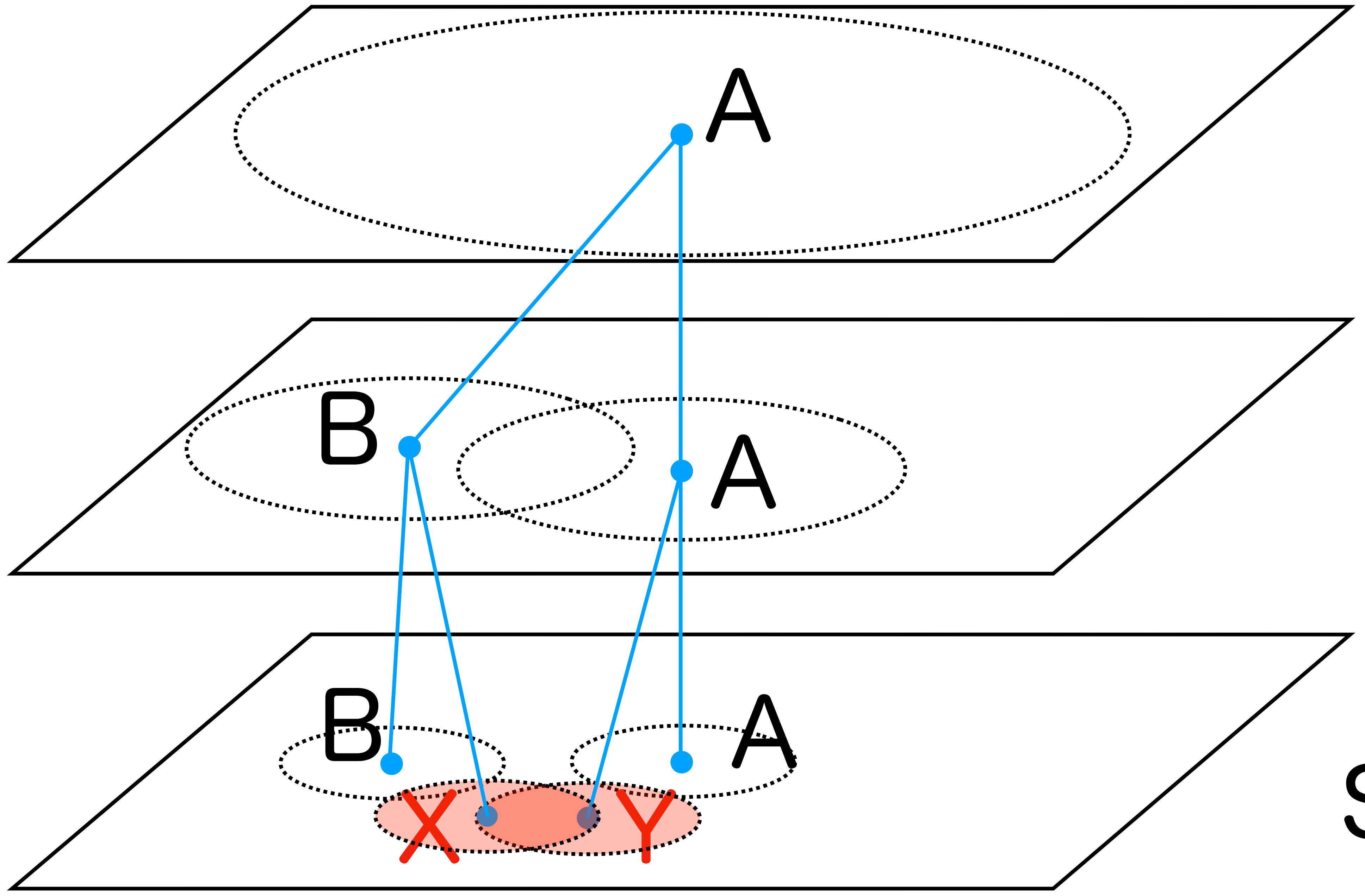
Parallel insert X and Y independently

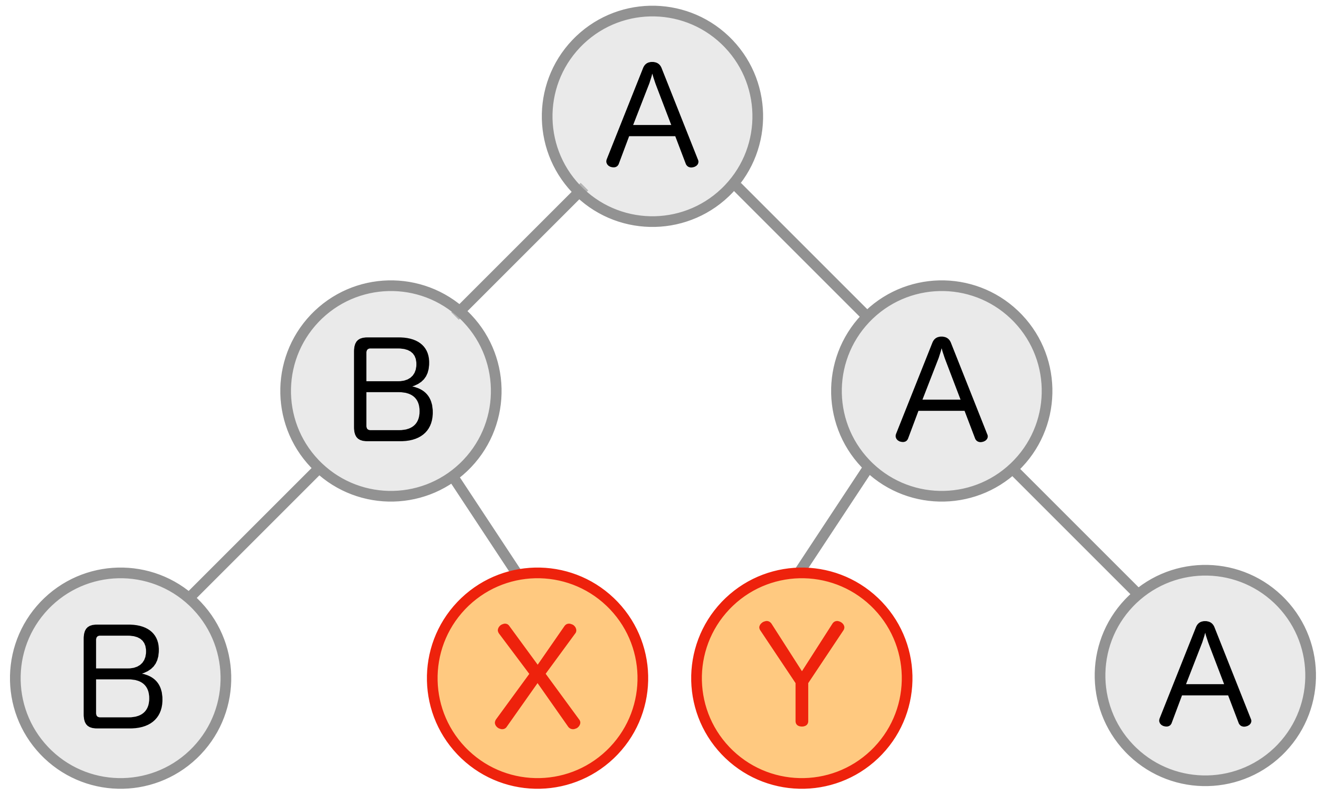




Parallel insertion on a cover tree is hard

Parallel insert X and Y independently

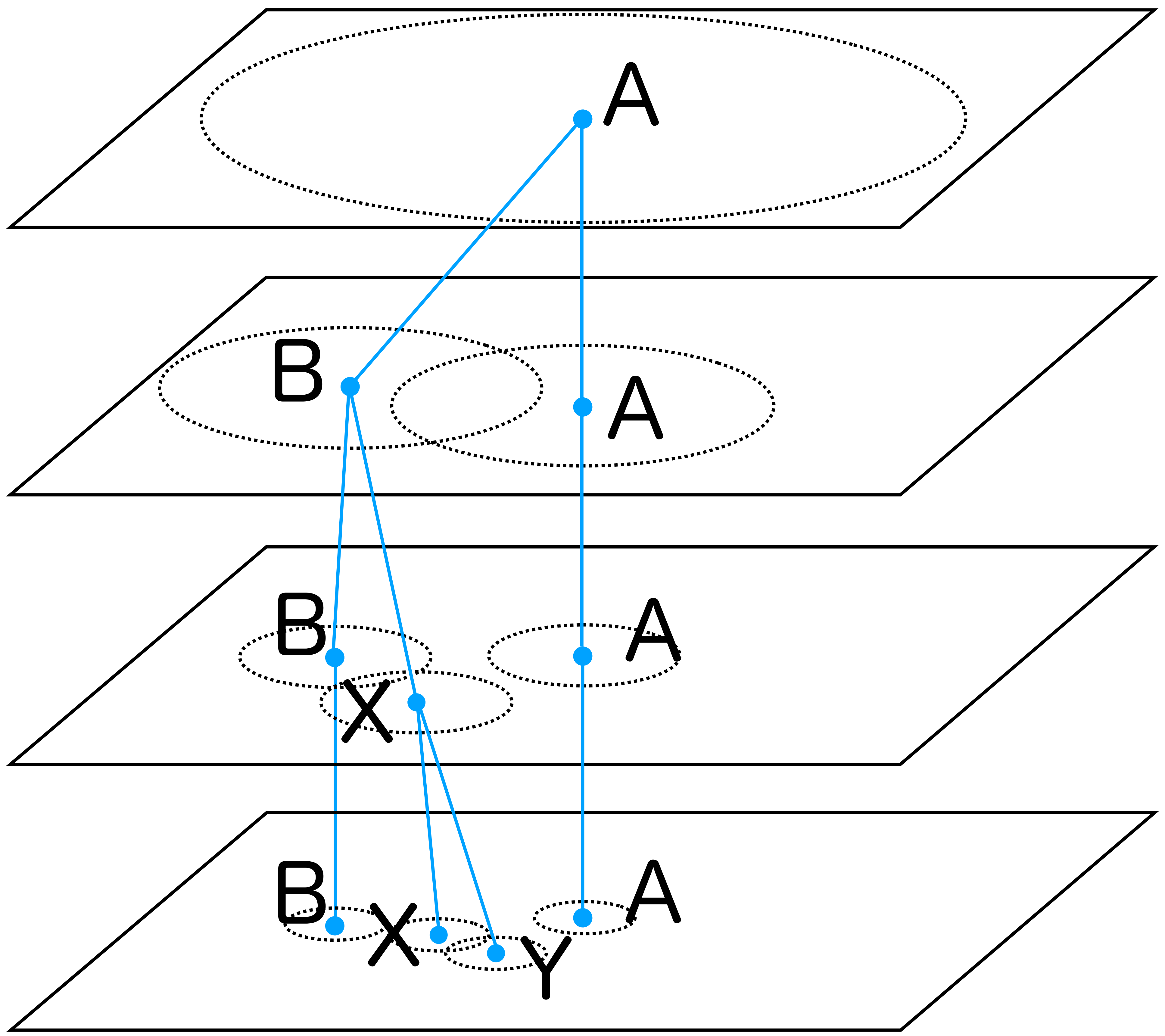




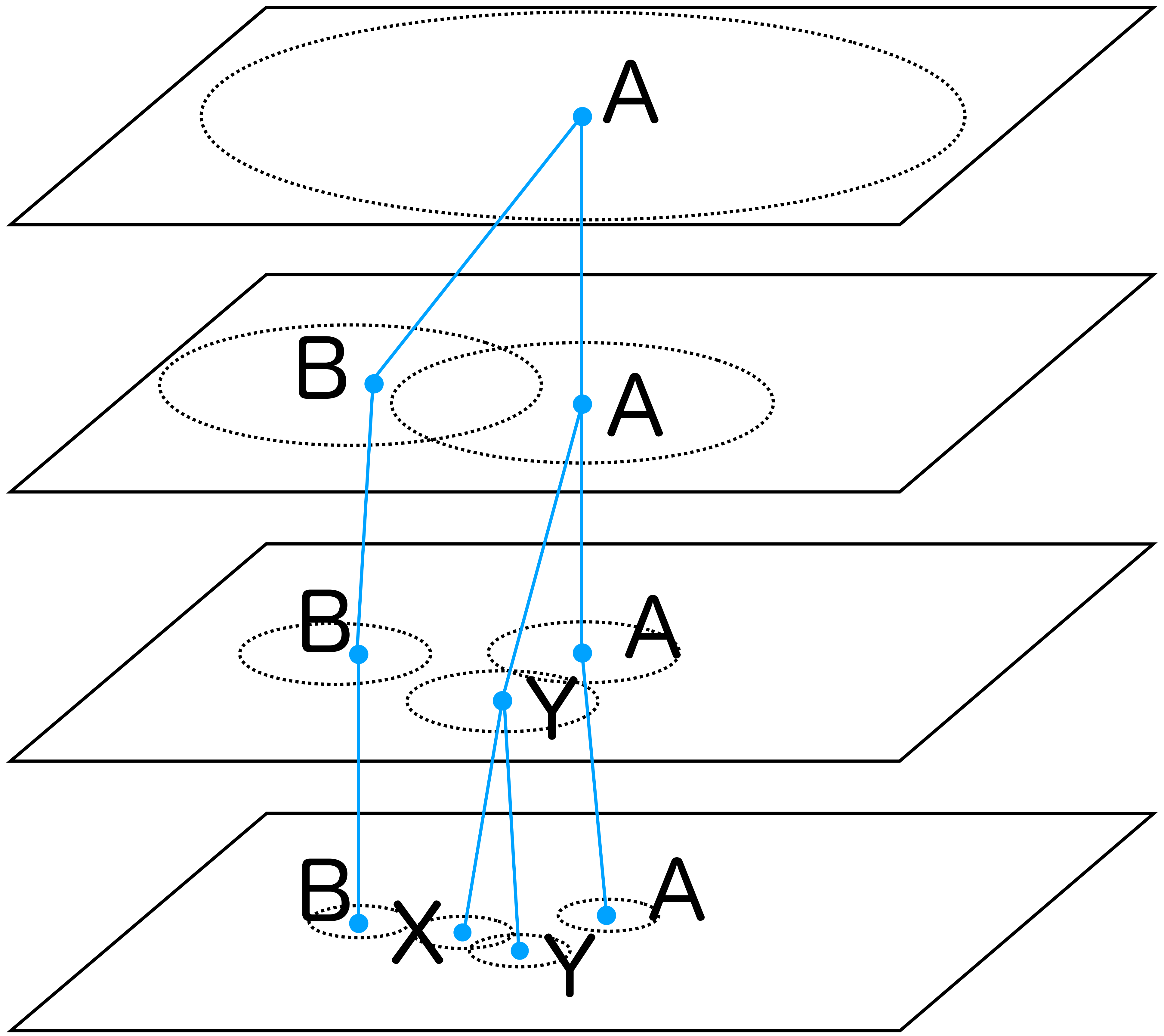
Separation

Parallel insertion on a cover tree is hard

first insert X then Y

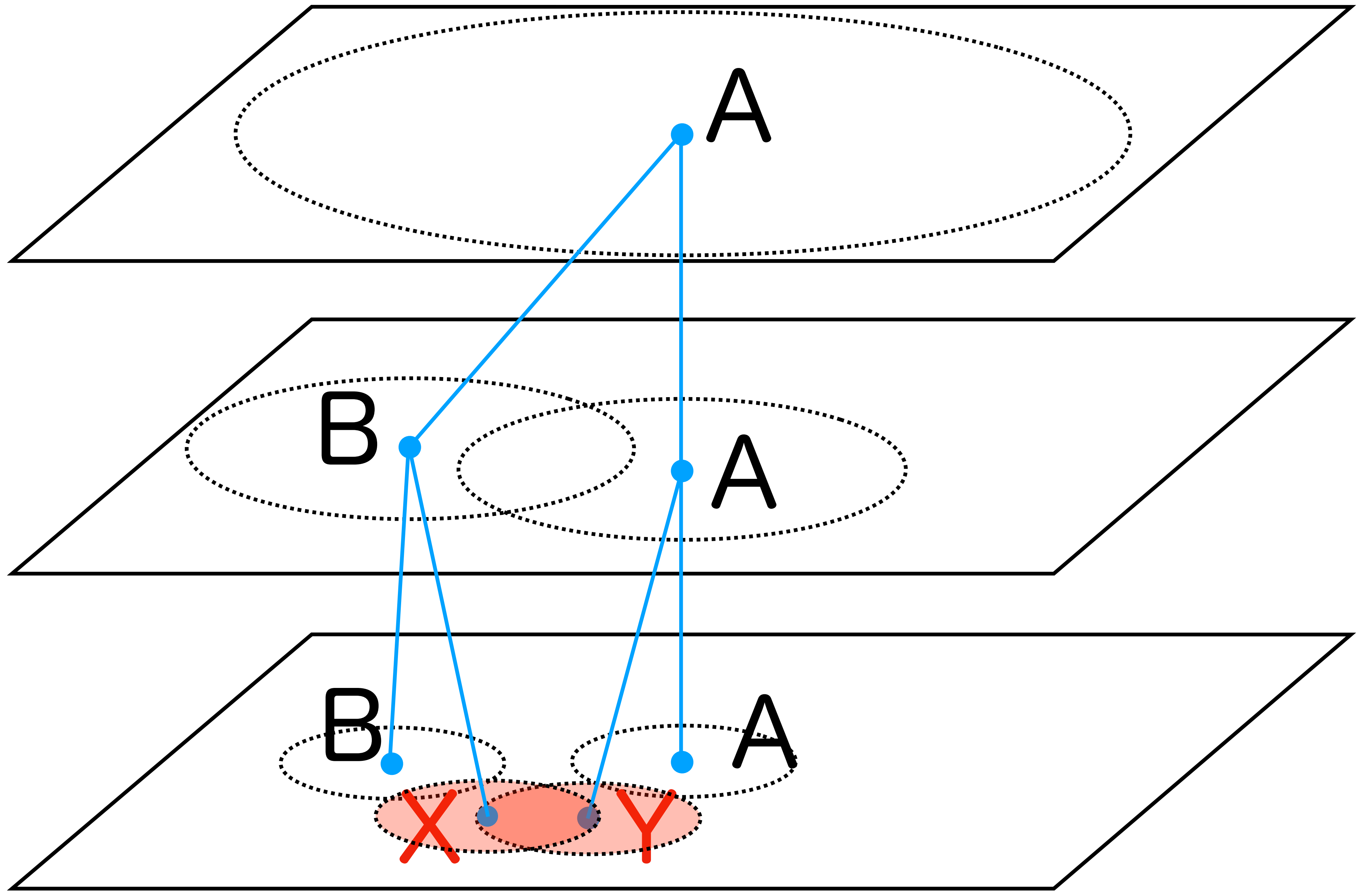


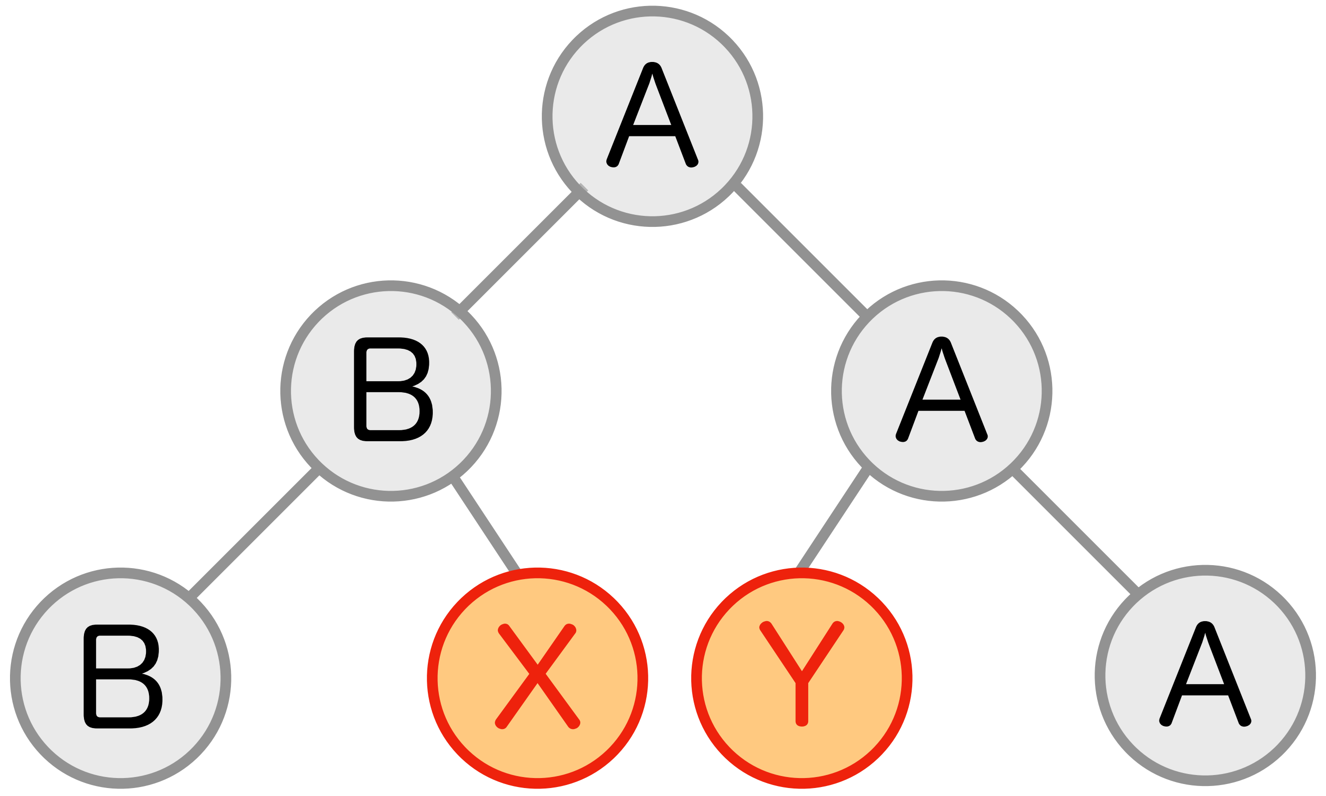
first insert Y then X



Parallel insertion on a cover tree is hard

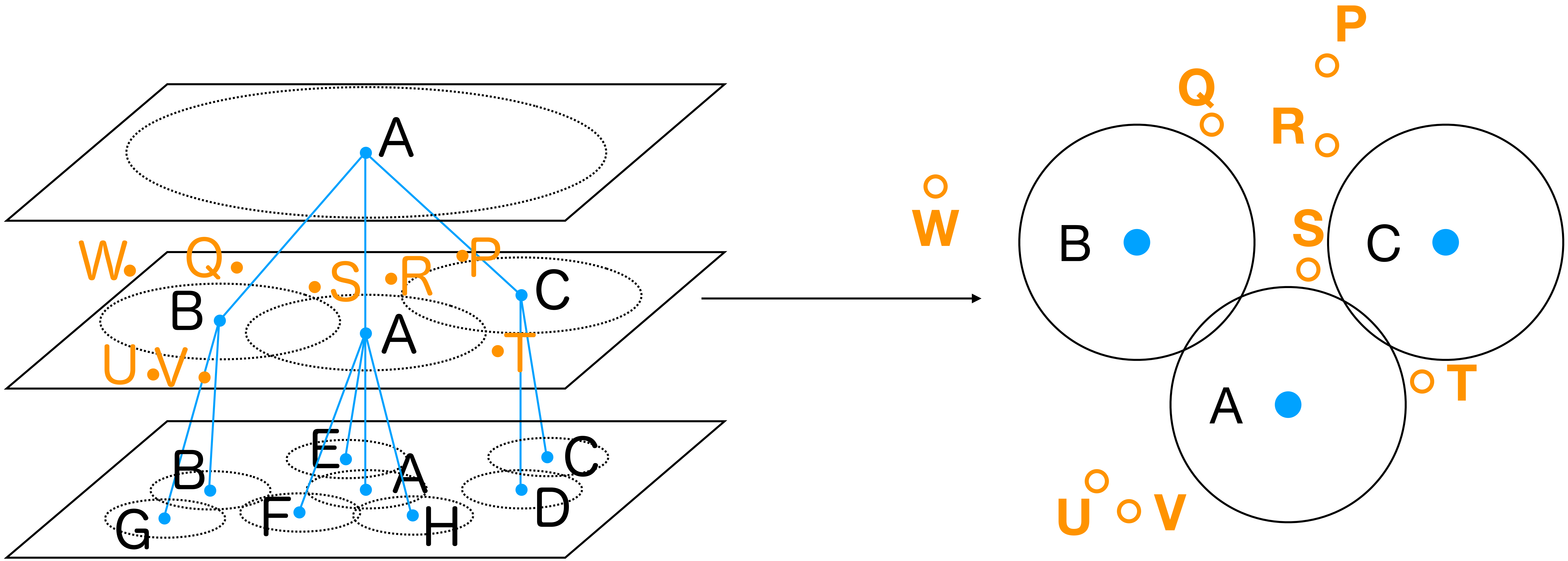
Parallel insert X and Y independently





Our parallel cover tree algorithms

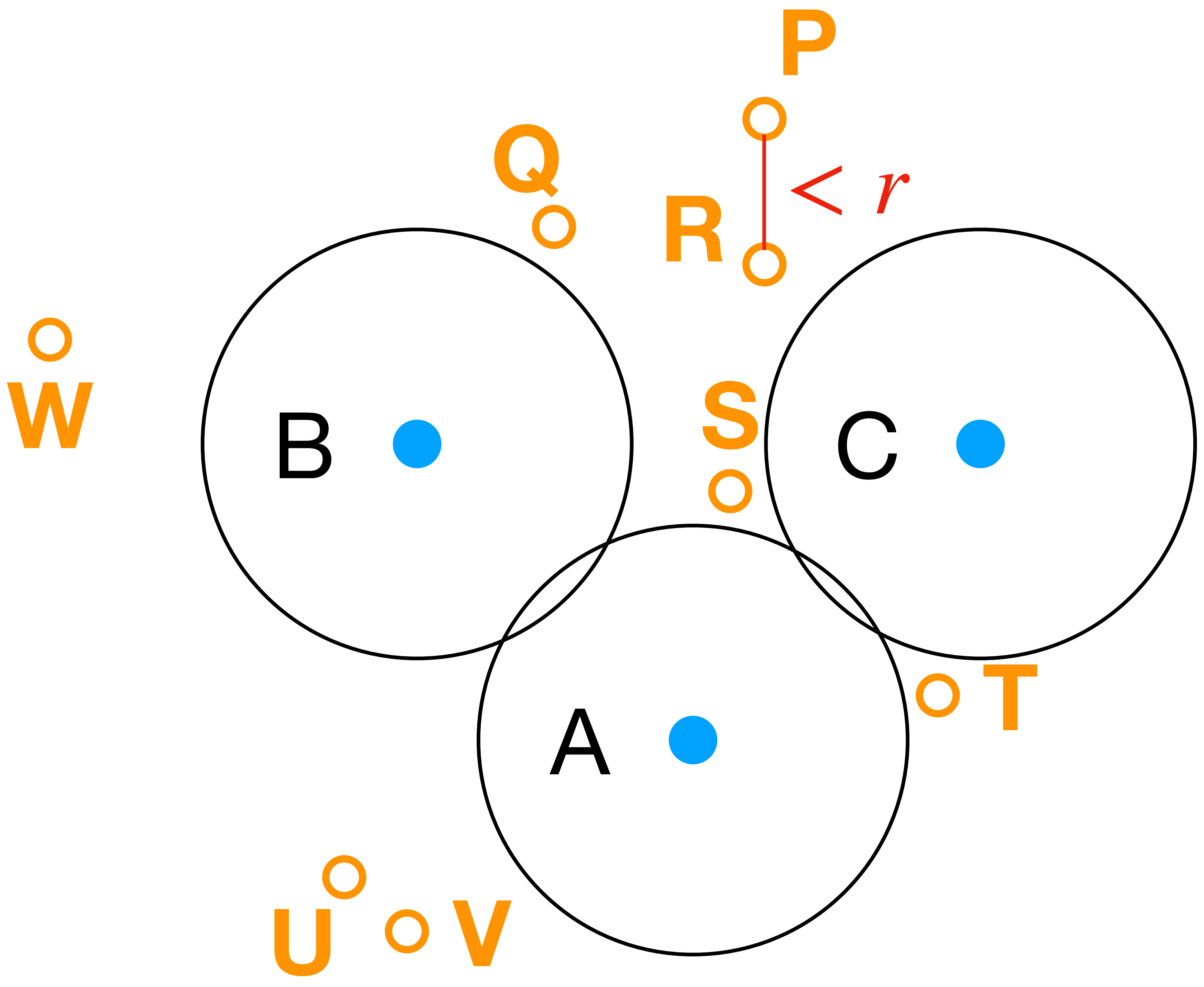
Parallel Insertion



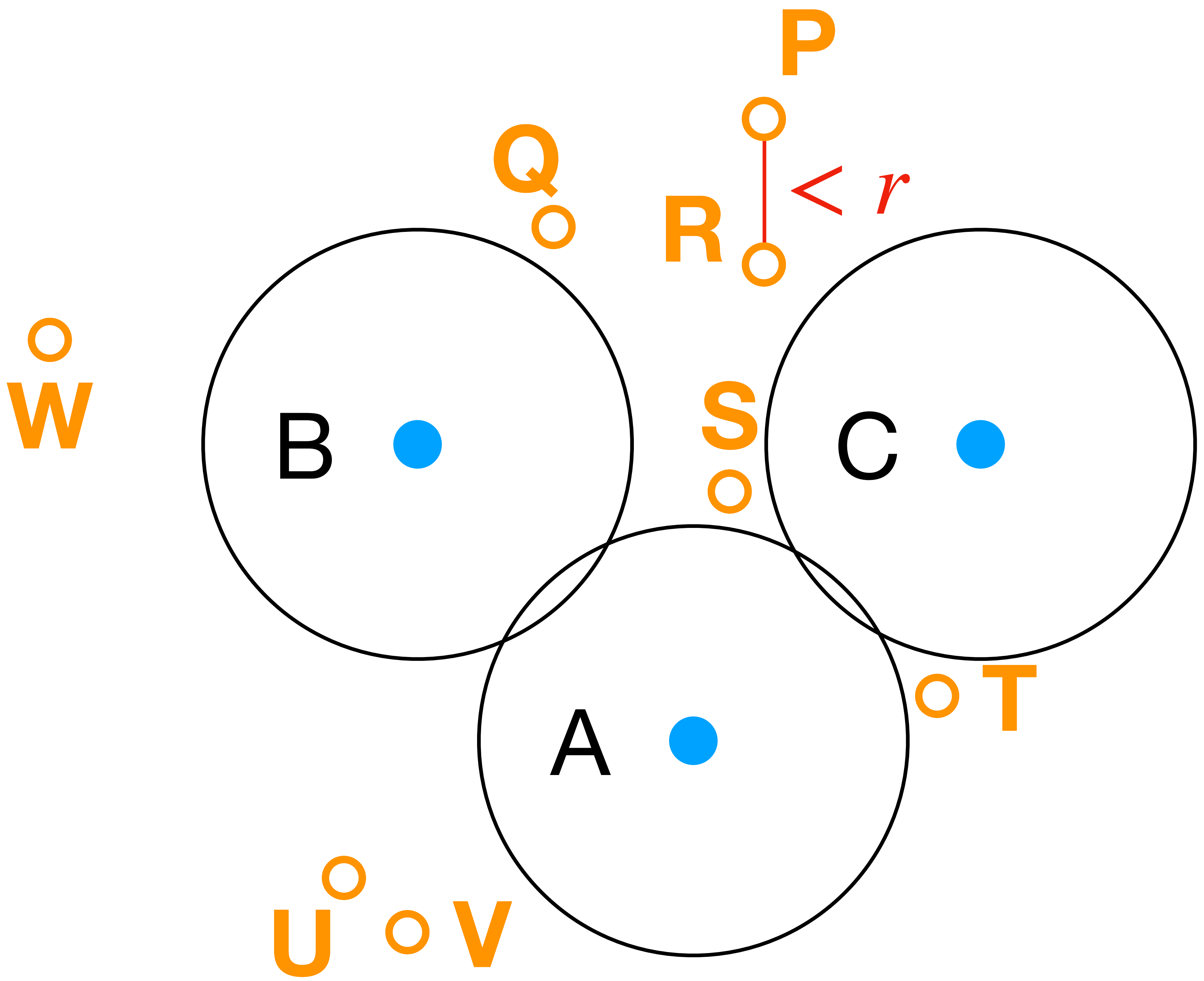
Parallel Insertion

Not all of them can be inserted at this level.

Separation



Model the conflict relations as a graph

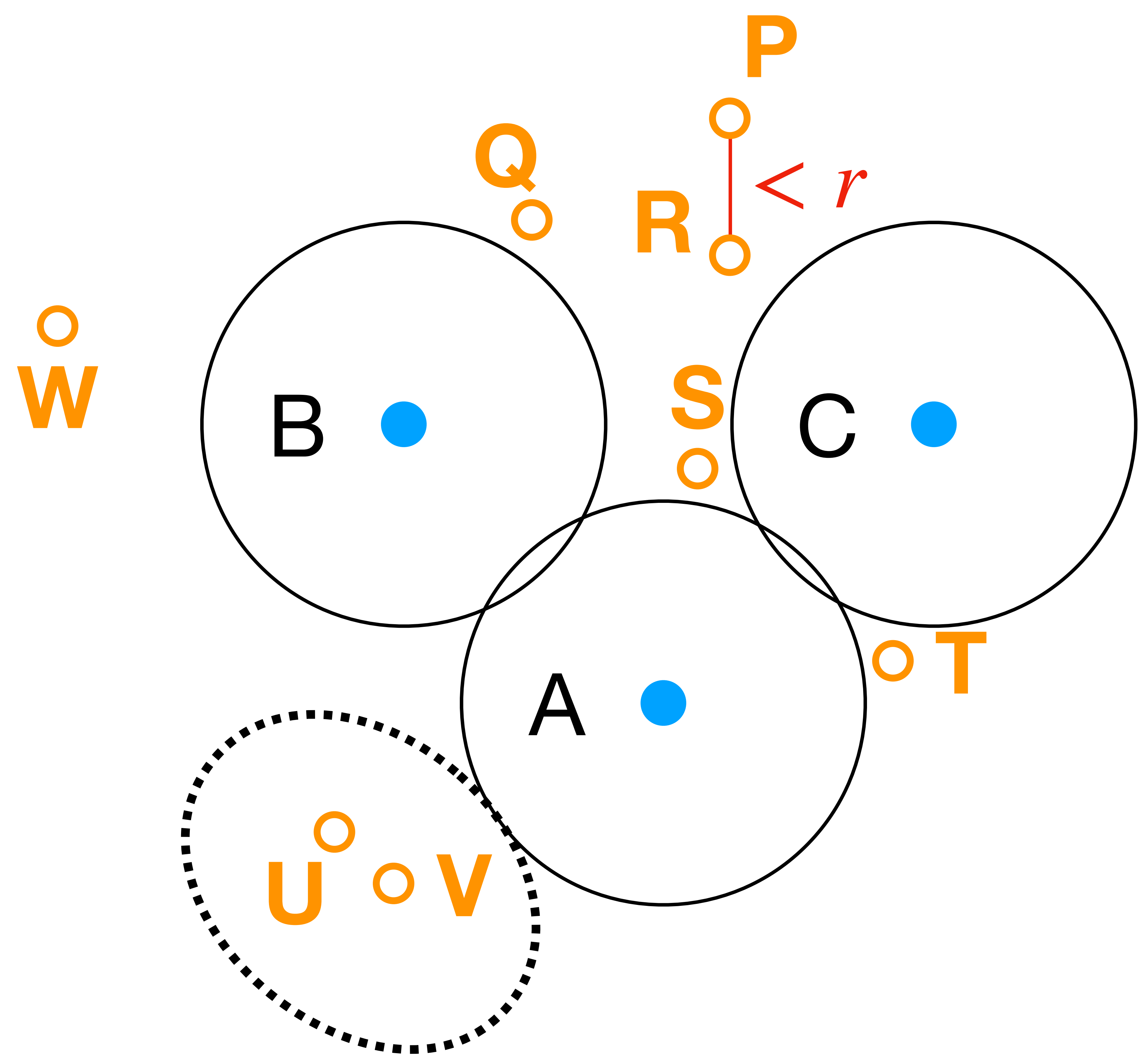


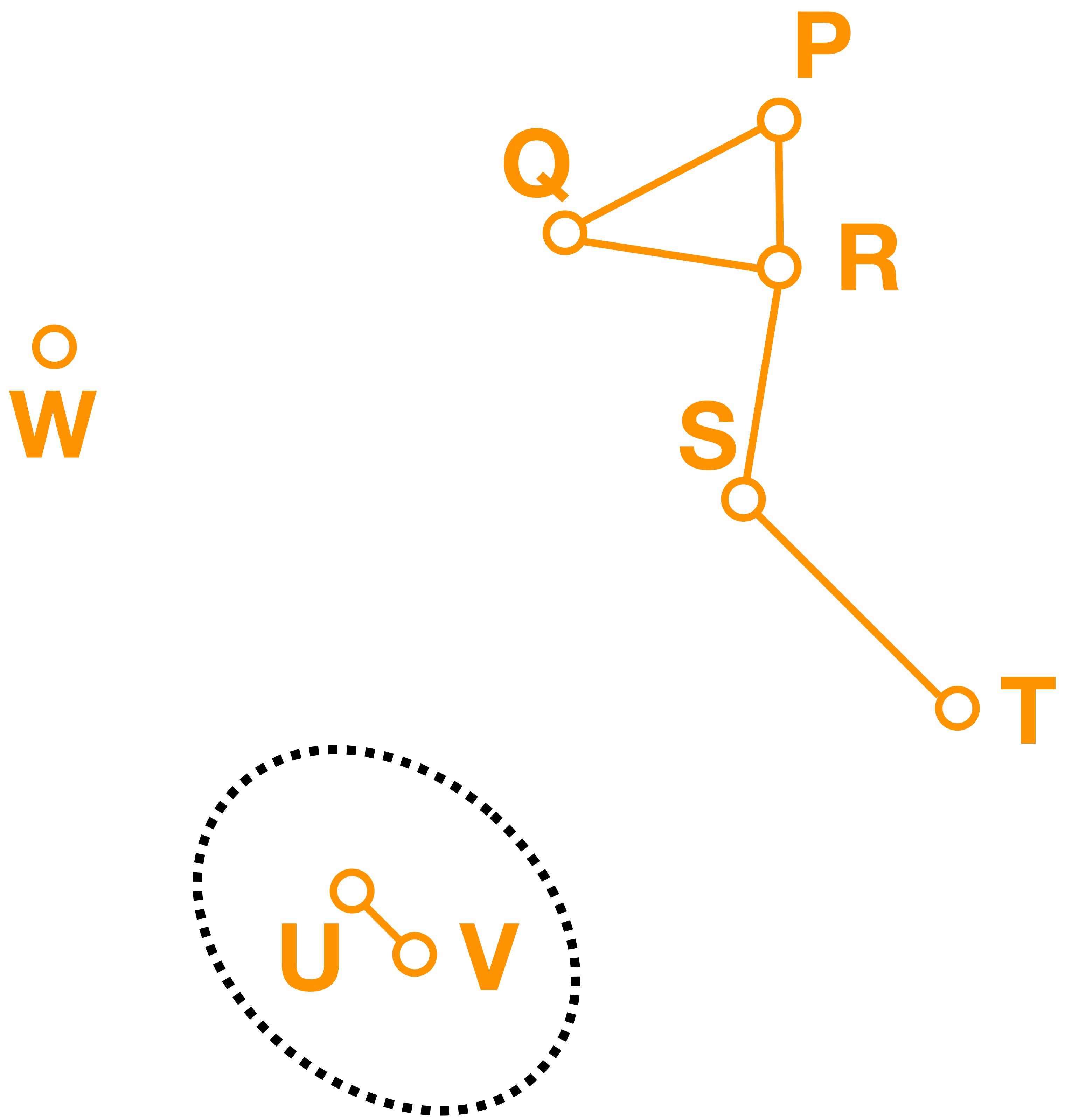
Model the conflict relations as a graph

If two points distance is smaller than r, we add and edge between them.

Separation For each edge, at most one endpoint can be selected

Some of them have to be inserted, why?



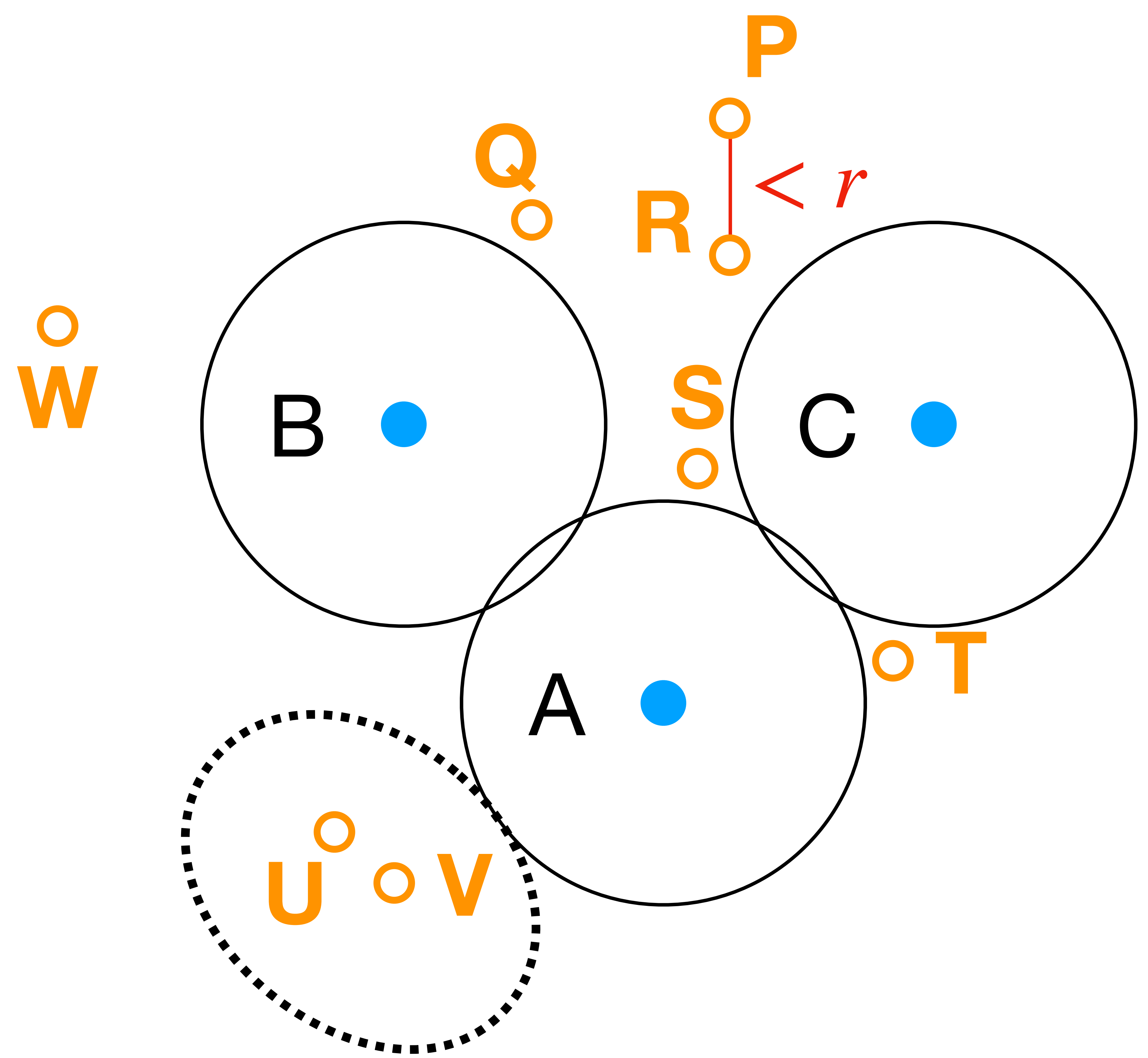


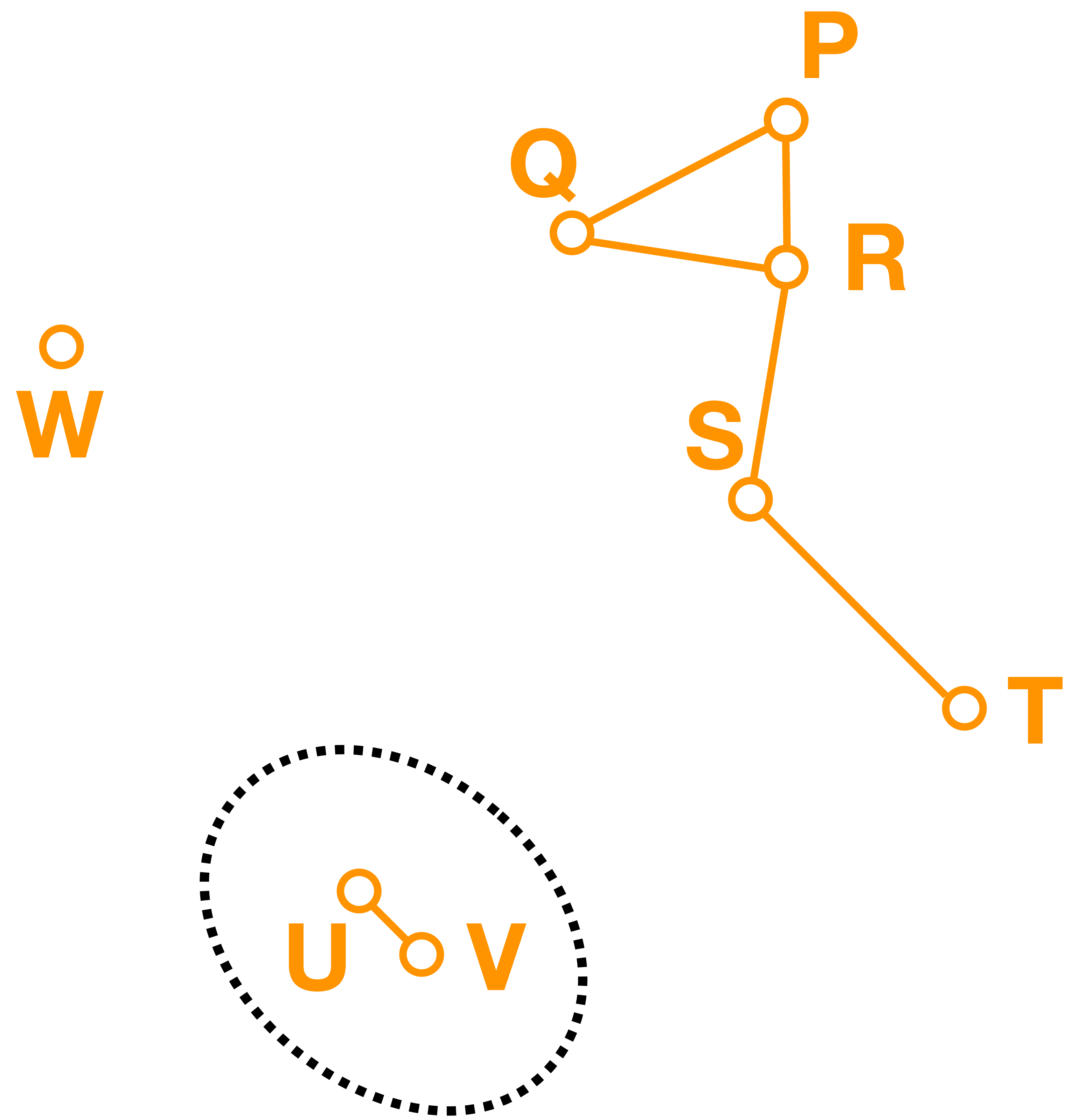
Model the conflict relations as a graph

If two points distance is smaller than r, we add and edge between them.

Separation For each edge, at most one endpoint can be selected

Some of them have to be inserted, why? **Covering**





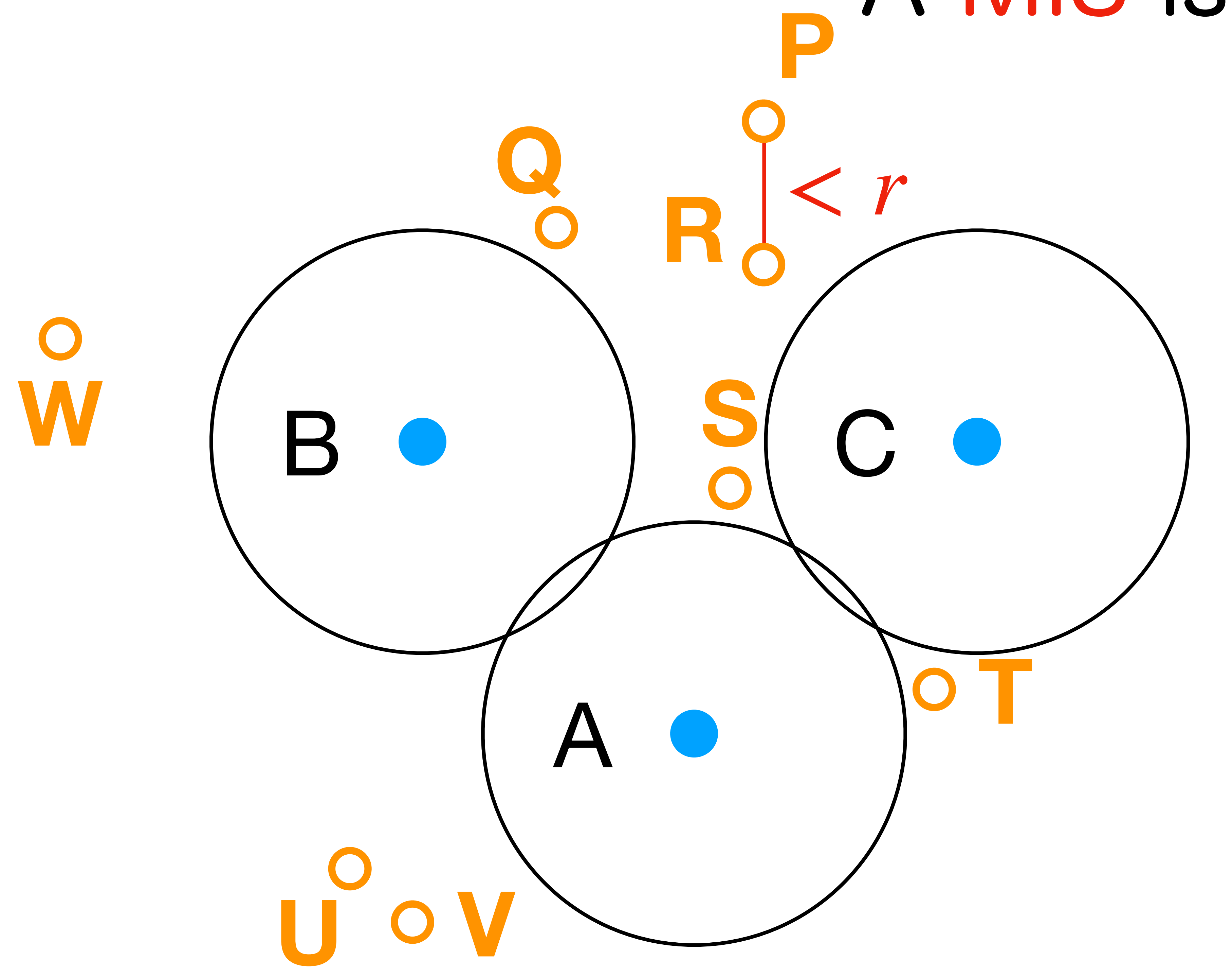
Model the conflict relations as a graph

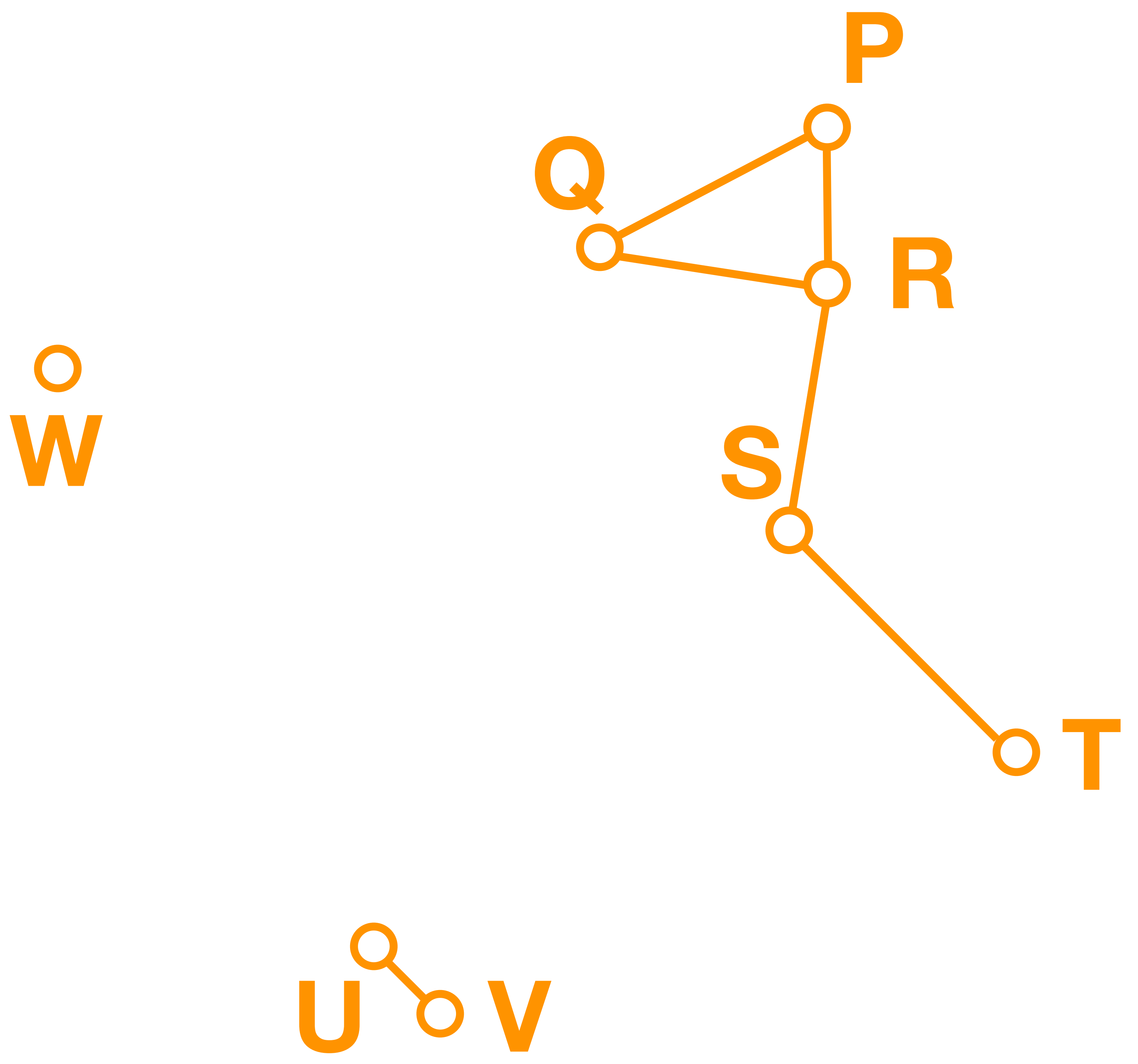
If two points distance is smaller than r, we add and edge between them.

For each edge, at most one endpoint can be selected

For each point and its neighbor(s), at least one of them must be selected

A MIS is a feasible solution!





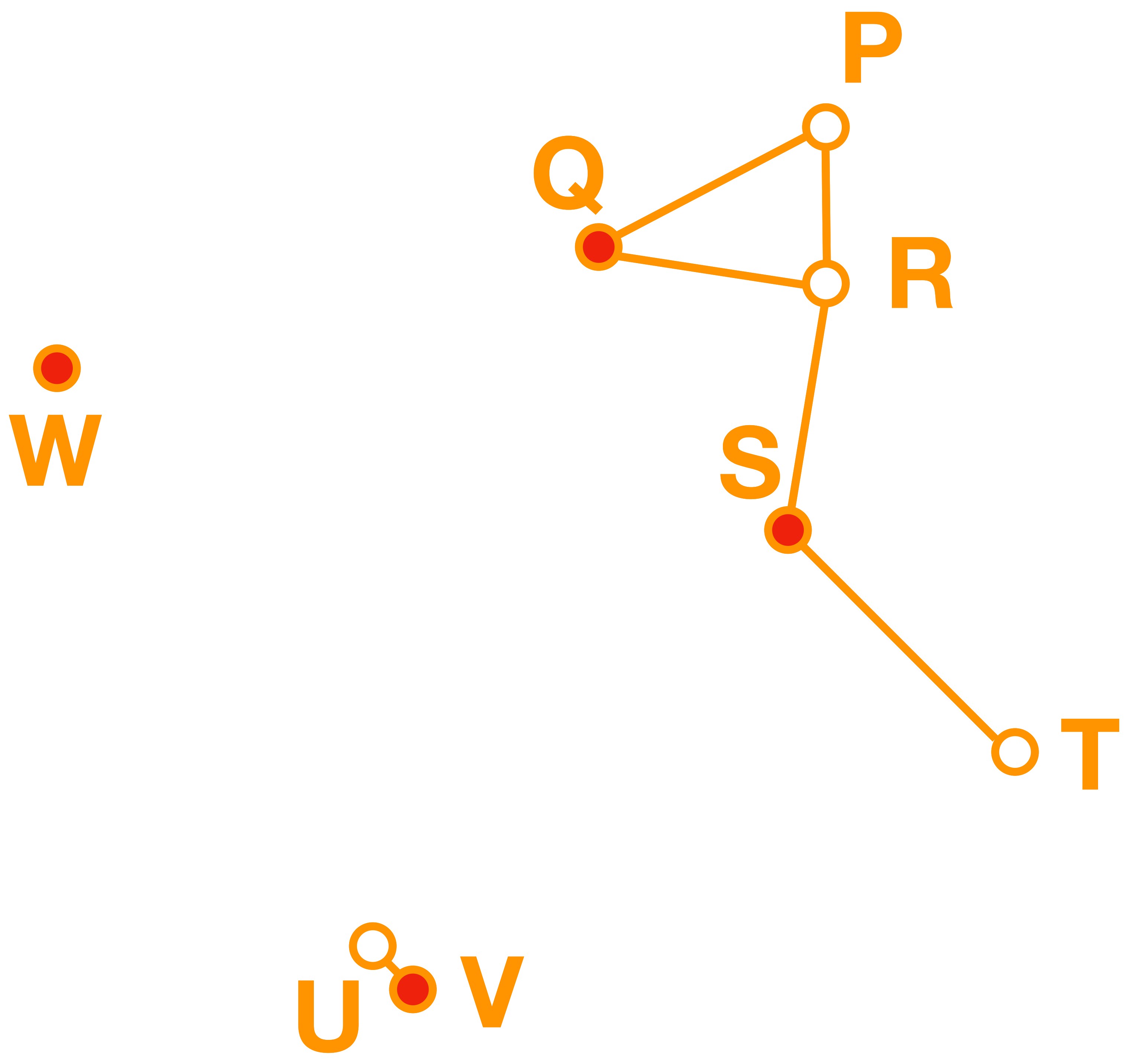
Maximal Independent Set (MIS)

Independent: each edge has at most one endpoint selected.

For each edge, at most one endpoint can be selected

Maximal: we can not add more vertices while maintaining independent.

For each point and its neighbor(s), at least one of them must be selected

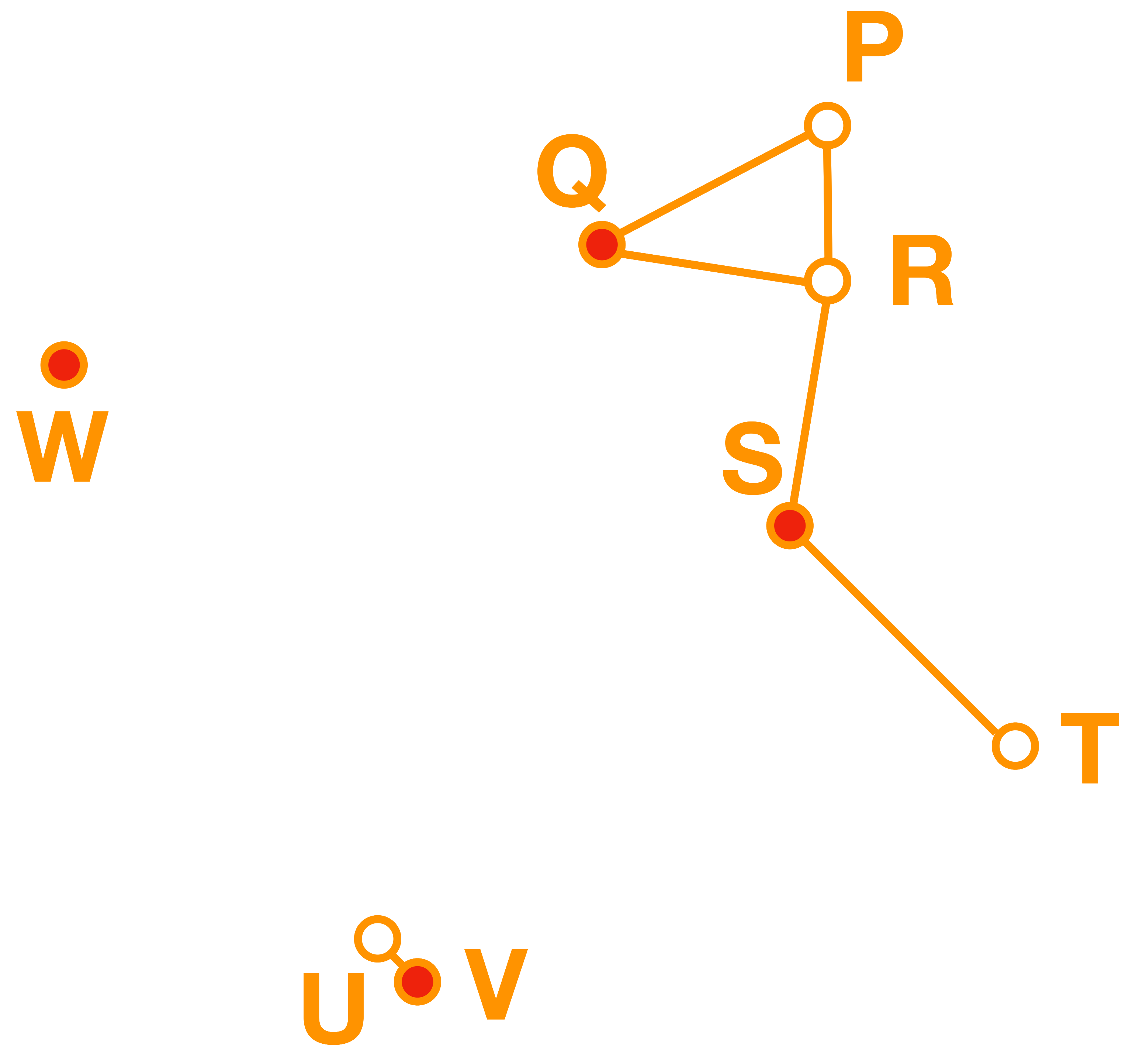


An MIS exactly gives us what we need.

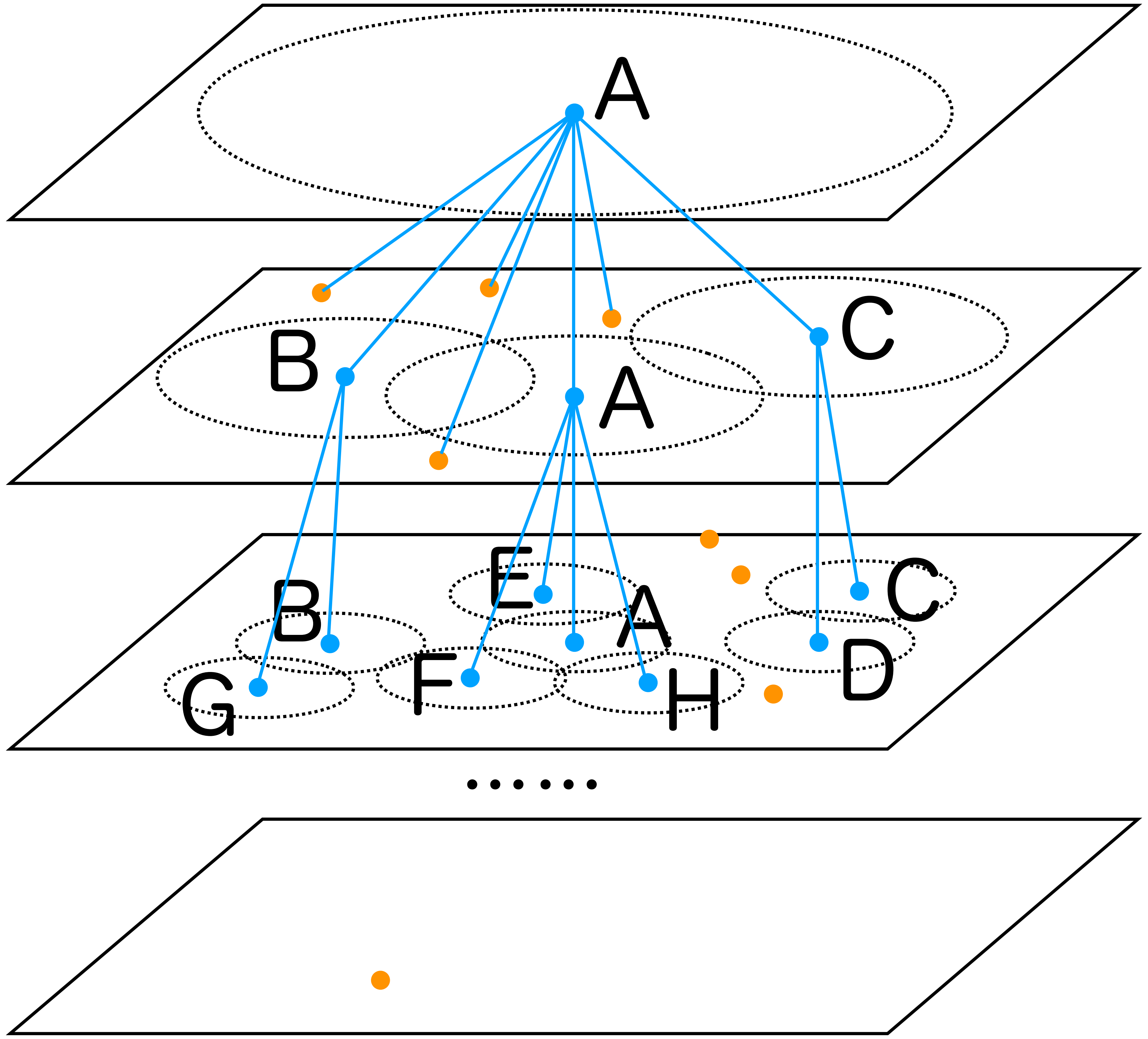
$$I = \{Q, W, S, V\}$$

Maximal Independent Set (MIS)

Insert points in MIS to the current level
The rest points will be dealt with recursively in lower levels



$$I = \{Q, W, S, V\}$$



Key Techniques

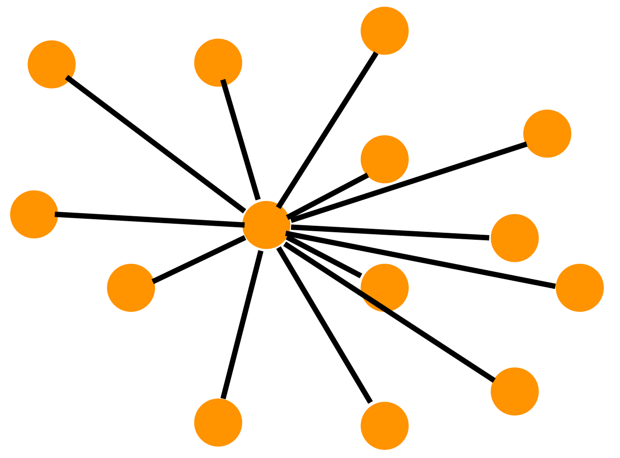
Maximal Independent Set (MIS) → a valid cover tree

Work-efficiency

Key Techniques

Maximal Independent Set (MIS) → a valid cover tree

Work-efficiency



Insert m points to an empty tree in parallel?

Each point will conflict with all the other points.

$O(m^2)$ edges

$O(m \log m)$ sequential work

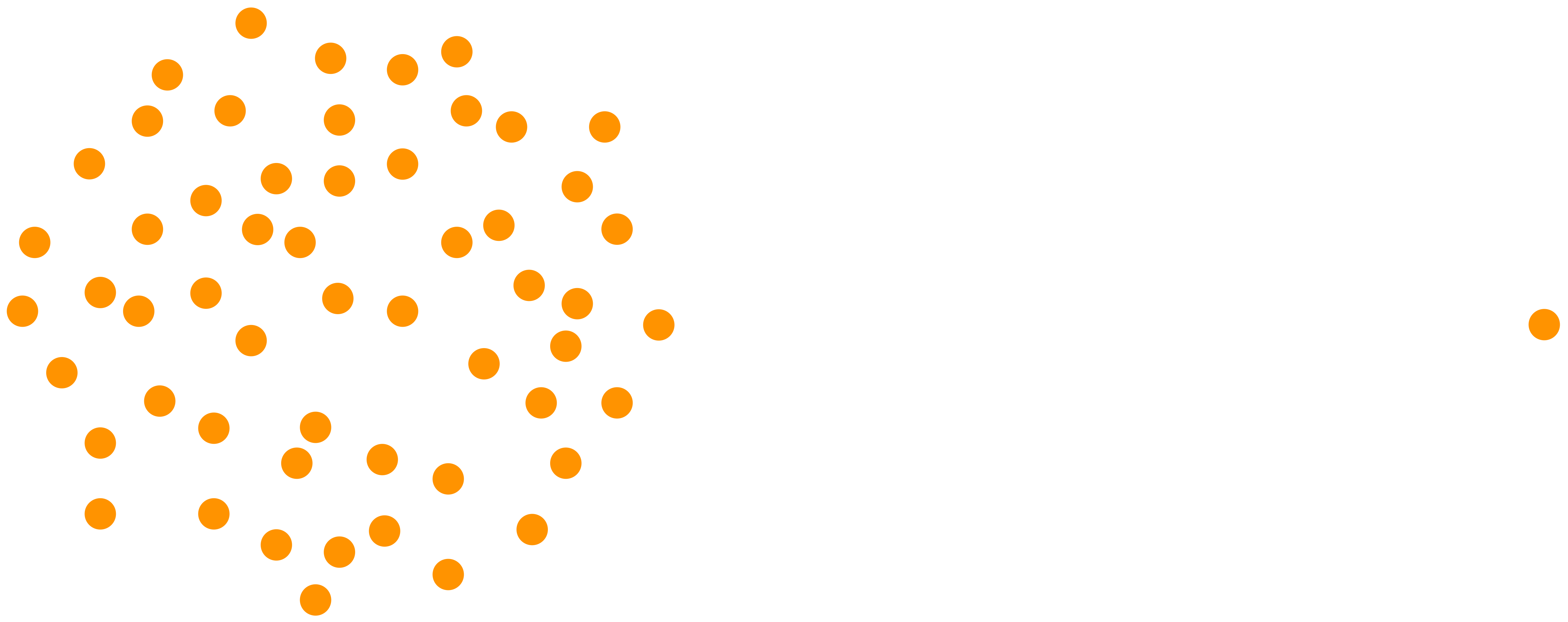
Key Techniques

Maximal Independent Set (MIS) → a valid cover tree

Prefix-doubling

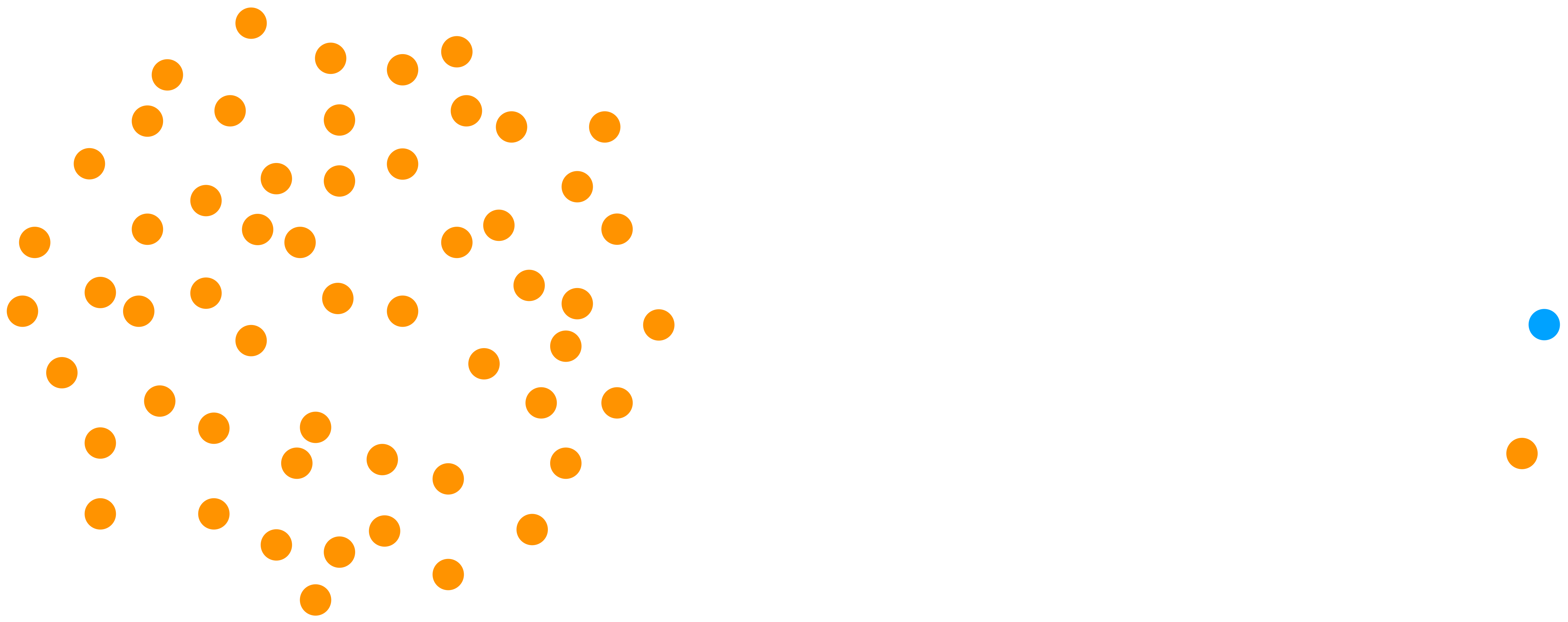
Prefix doubling

Partition the insertion batch S into $\log_2|S|$ sub-batches with size $1,1,2,4,8,\cdots$



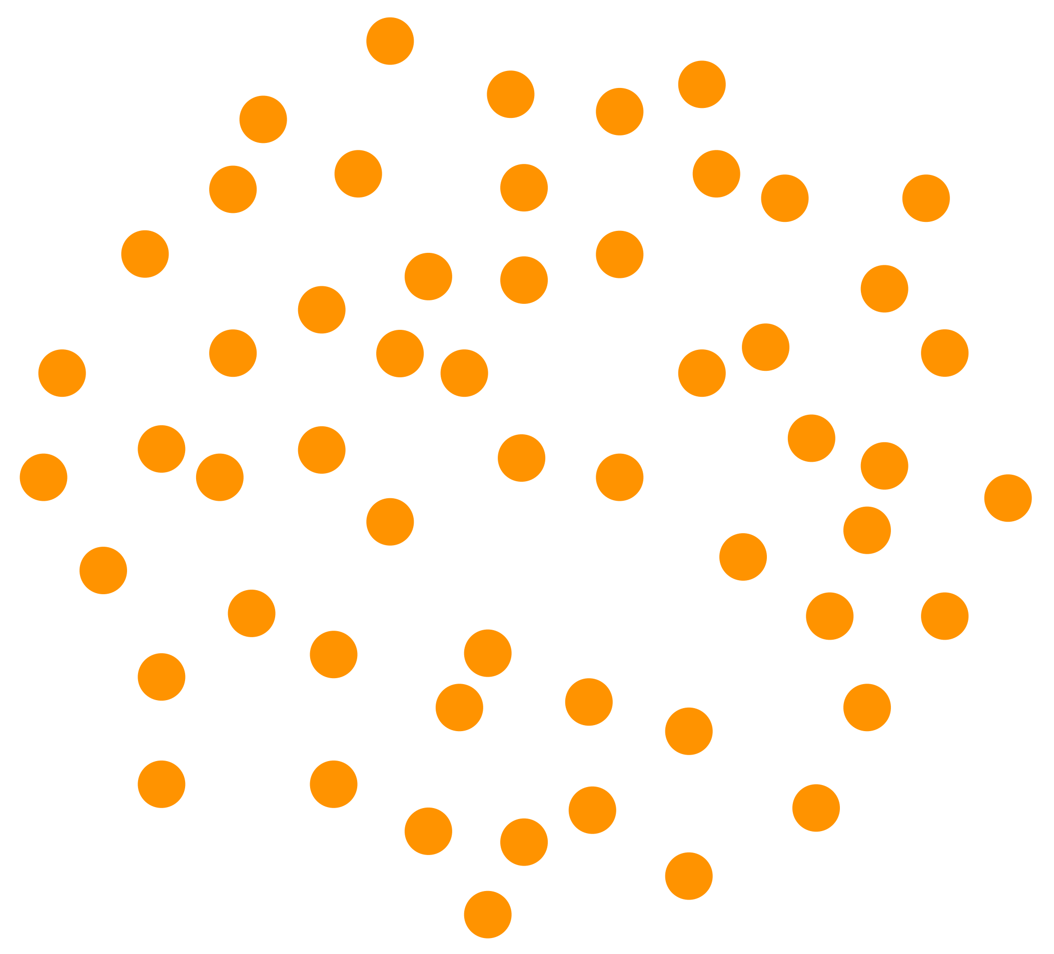
Prefix doubling

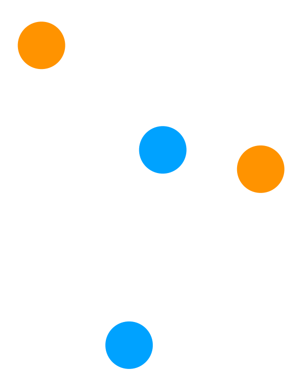
Partition the insertion batch S into $\log_2|S|$ sub-batches with size 1,1,2,4,8,⋯



Prefix doubling

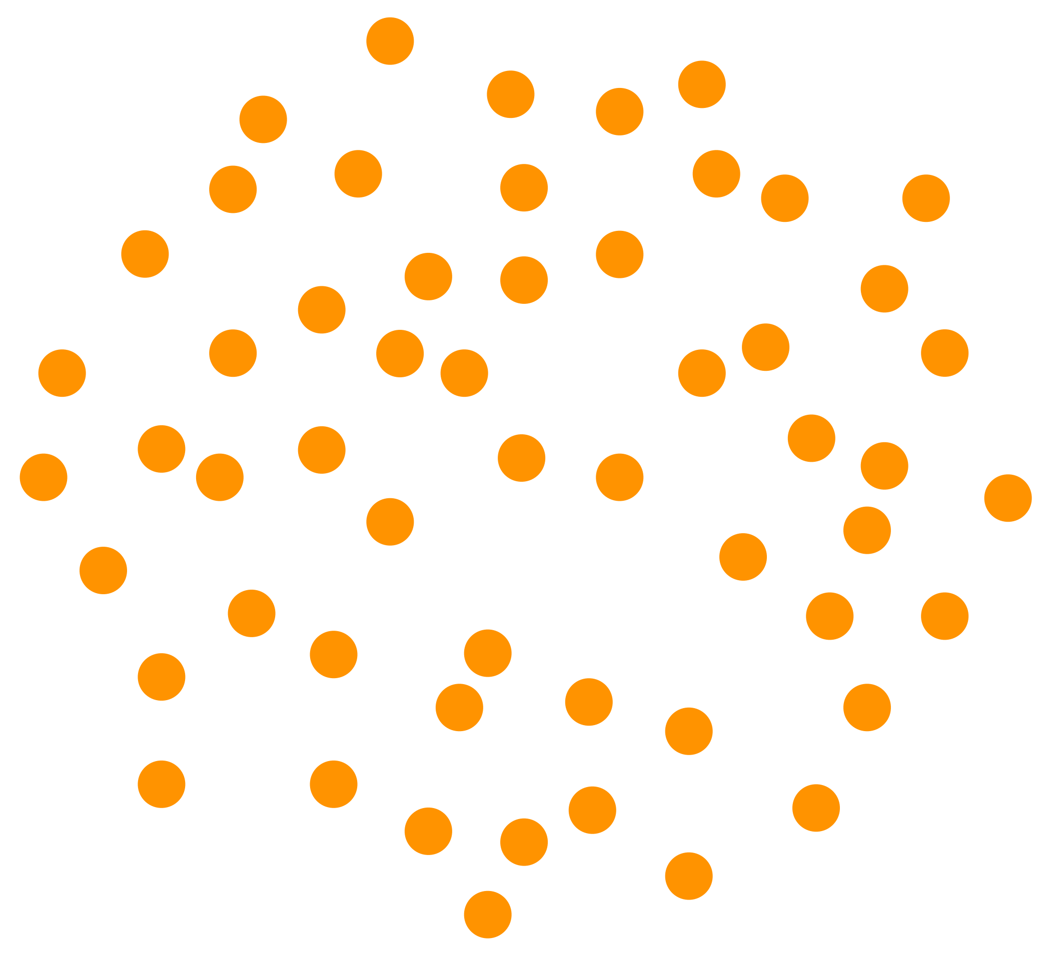
Partition the insertion batch S into $\log_2|S|$ sub-batches with size 1,1,2,4,8,$\cdots$

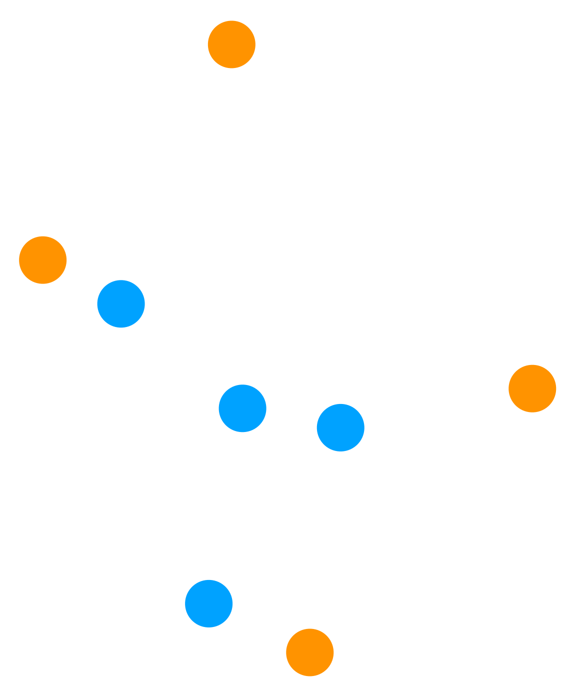




Prefix doubling

Partition the insertion batch S into $\log_2|S|$ sub-batches with size 1,1,2,4,8,⋯



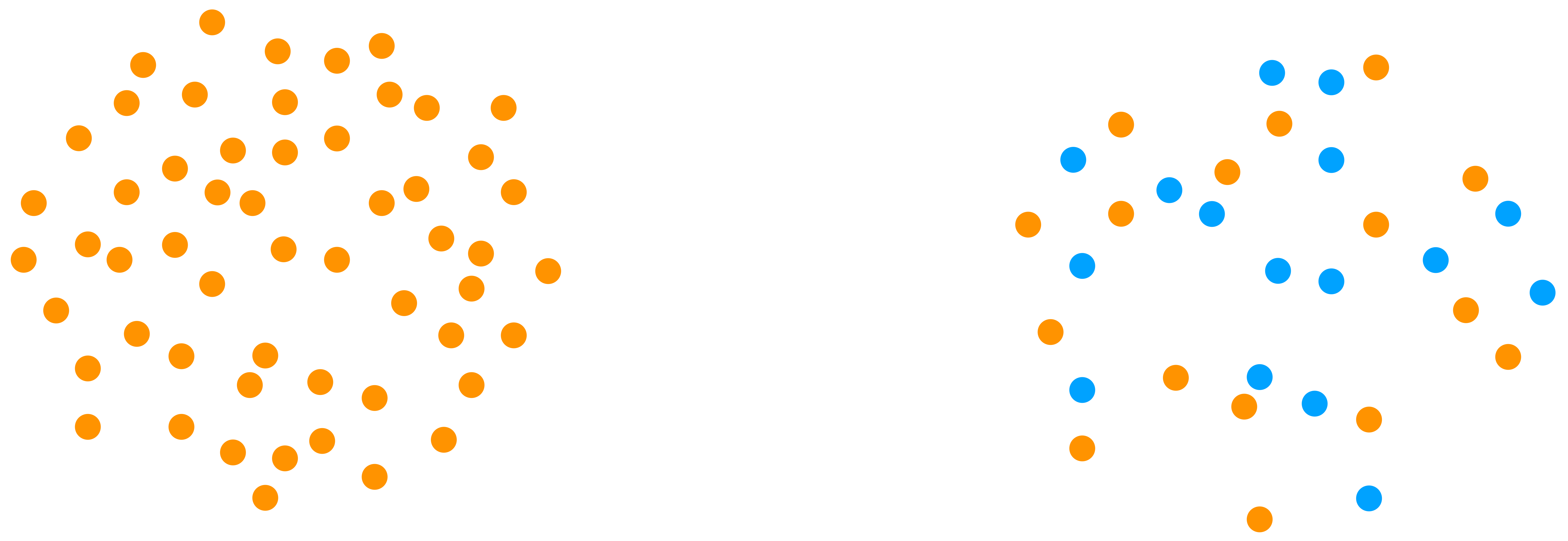


Prefix doubling

Partition the insertion batch S into $\log_2|S|$ sub-batches with size 1,1,2,4,8,⋯

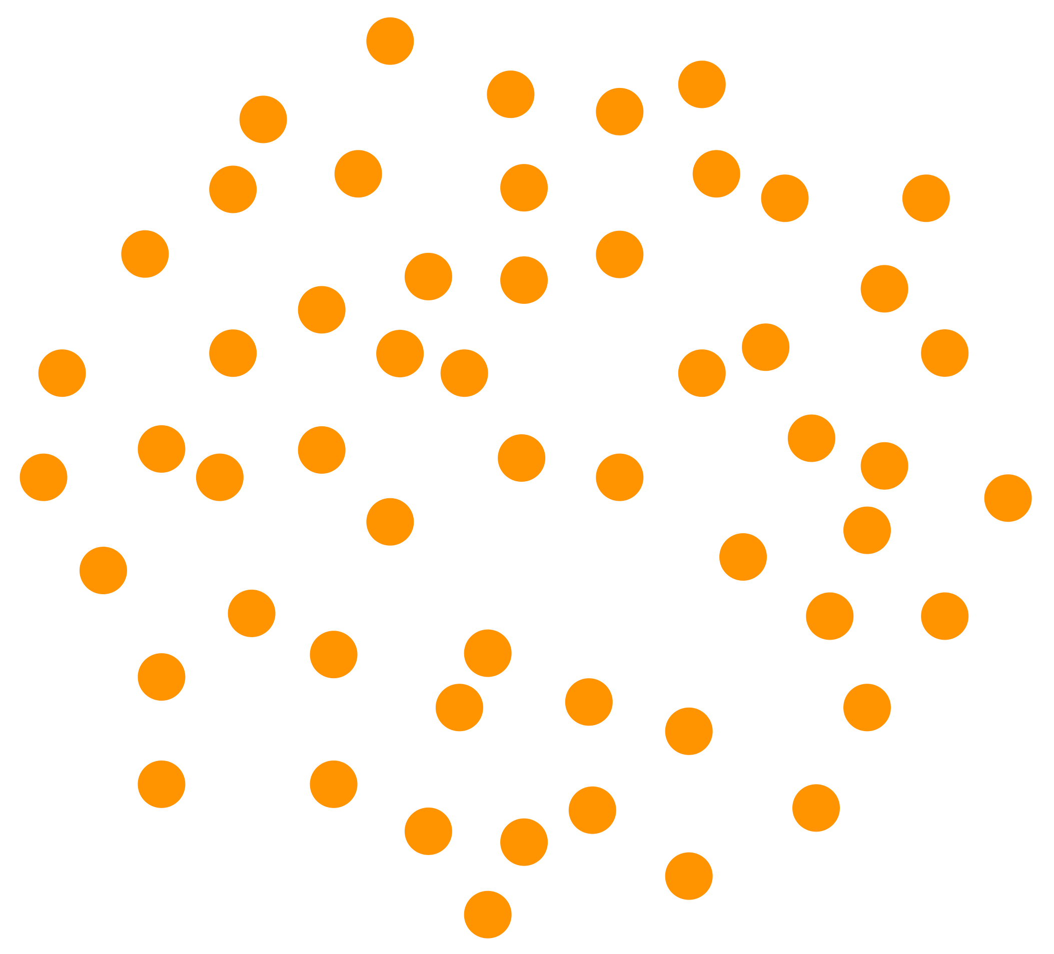
Prefix doubling

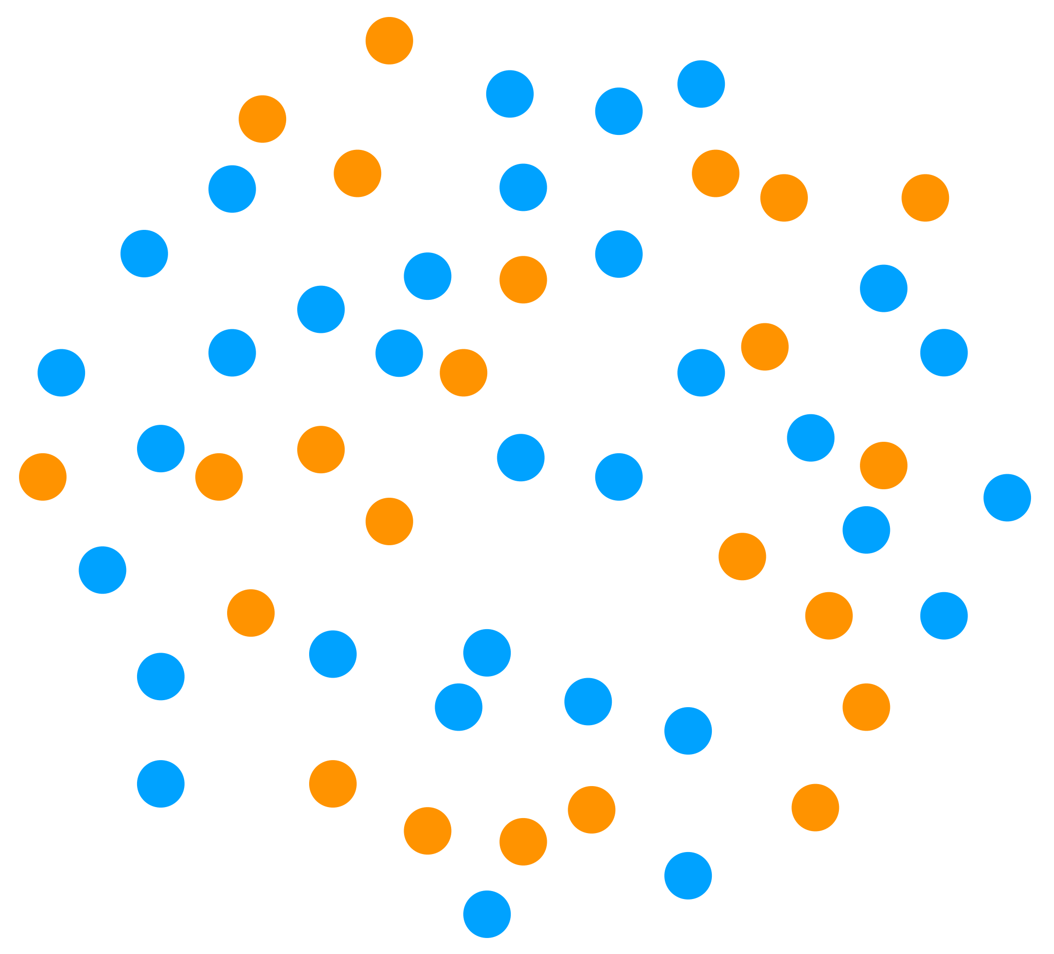
Partition the insertion batch S into $\log_2|S|$ sub-batches with size $1,1,2,4,8,\cdots$



Prefix doubling

Partition the insertion batch S into $\log_2|S|$ sub-batches with size 1,1,2,4,8,⋯



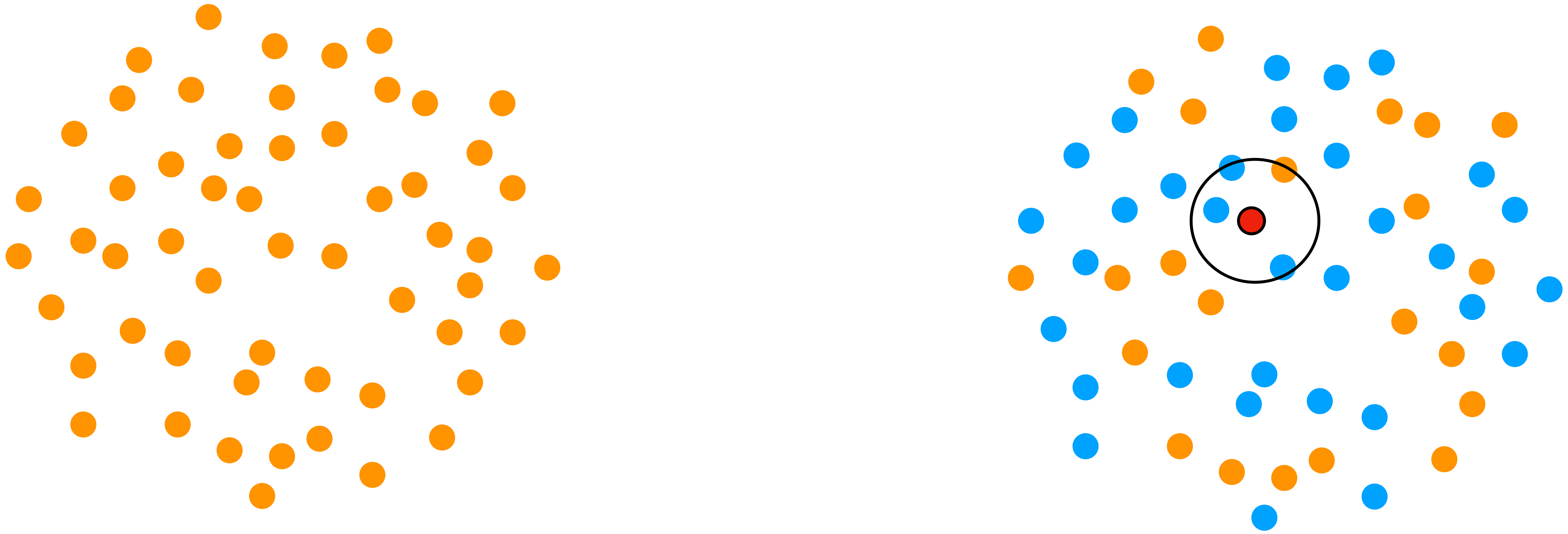


The current cover tree contains

at least as many points as the group being inserted

Prefix doubling

Partition the insertion batch S into $\log_2|S|$ groups with size 1,1,2,4,8,⋯⋯



The number of neighbors of a point is $O(1)$ in expectation and $O(\log n)$ *whp*

Bound the cost running MIS [Lemma 4.7]

Computational Model and Notations

- Binary-forking model (with test-and-set)
- Standard Work-Span evaluation:
  - Work: total number of operations
  - Span (depth): number of operations on the longest dependence chain
- Work-efficiency:
  - The work is asymptotically the same as the best sequential solution

Cost Analysis for MIS

- Binary-forking model (with test-and-set)
- Parallel Maximal Independent Set on a graph $G = (V, E)$ [Shen et al. 2022]
  - Work: $O(|V| + |E|)$
  - Span: $O(\log |V| \log d_{max})$ whp, where d_{max} is the maximum degree
- When inserting m points to a cover tree with n points

The number of a point's neighbors is $O(1)$ in expectation and $O(\log n)$ whp

$|V| = O(m)$

$|E| = O(m)$ in expectation

$d_{max} = O(\log n)$ whp

Work: $O(m)$ in expectation

Span: $O(\log m \log \log n)$ whp

Parallel insertion is work-efficient

Inserting m points to a cover tree with n points

$O(c^5 mH(T))$ expected work $\quad\quad$ $H(T)$ is tree height

$O(H(T)\log m(\log c + \log m \log\log n))$ span *whp*

Parallel insertion is work-efficient

Inserting m points to a cover tree with n points

$O(c^5 mH(T))$ expected work $\quad$ $H(T)$ is tree height

$c^5 H(T) \longrightarrow$ **single-insertion sequentially**

$O(H(T)\log m(\log c + \log m \log\log n))$ span *whp*

top-down on each level

prefix-doubling

Constructing every graph

every MIS

When assuming $c = O(1)$ and $H(T) = \Theta(\log n)$, inserting m points to a cover tree contains n points costs:

Work: $O(m \log n)$ in expectation

Span: $O(\log n \log^2 m \log\log n)$ with high probability

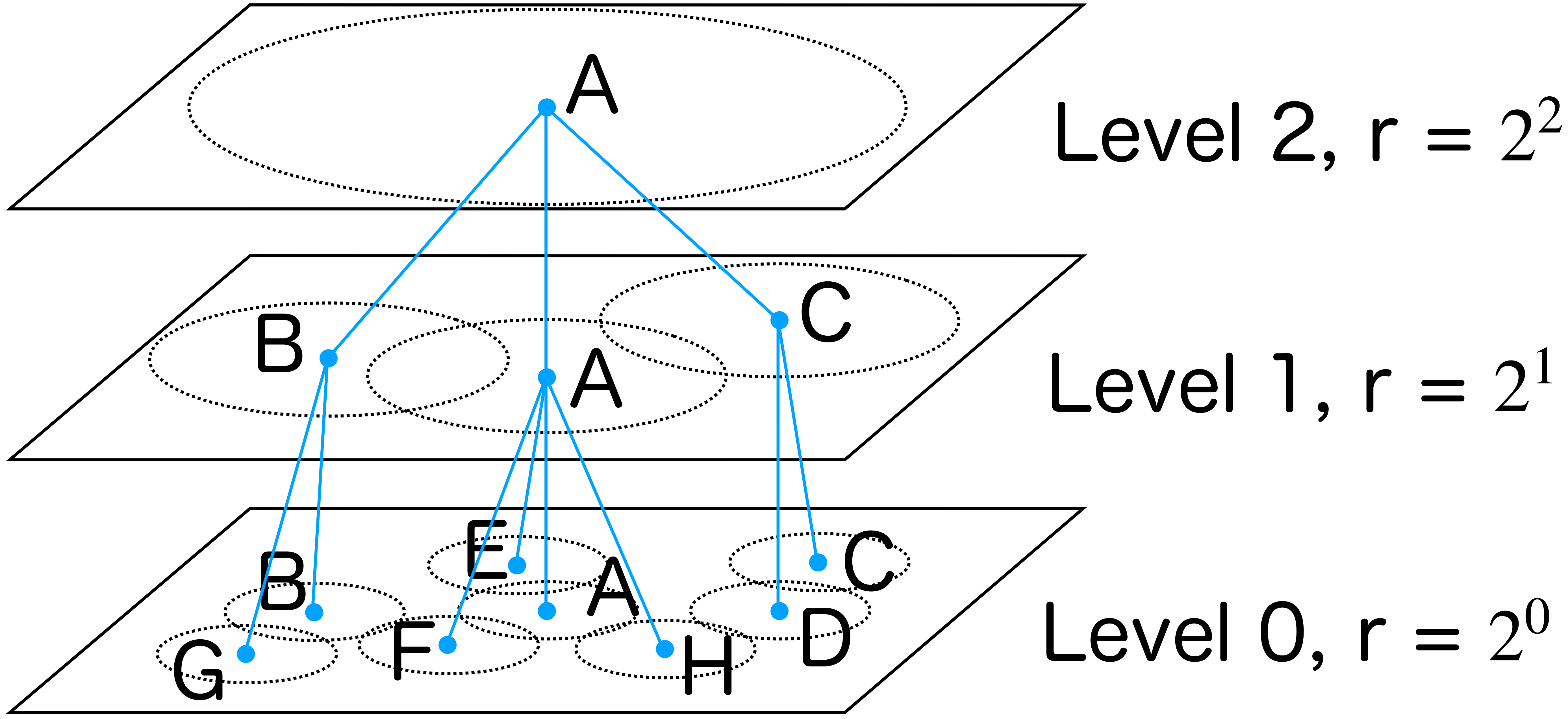
Key Techniques

Maximal Independent Set (MIS) → Correctness & Parallelism

Prefix-doubling → Work-efficiency

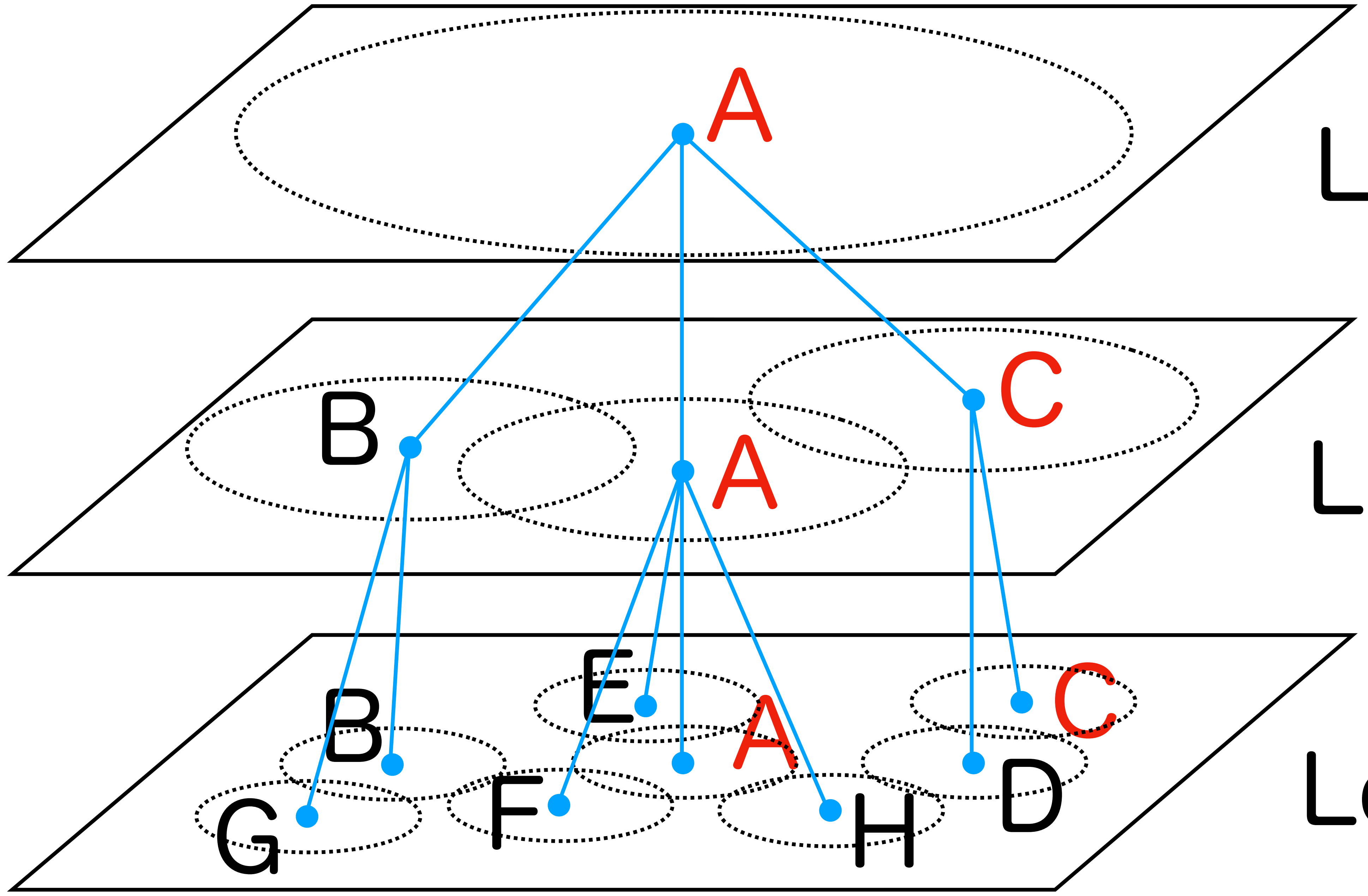
Parallel deletion is similar to insertion

Parallel deletion is in the bottom-up order



Parallel deletion is similar to insertion

Parallel deletion is in the bottom-up order



Level 2, r = 2^2

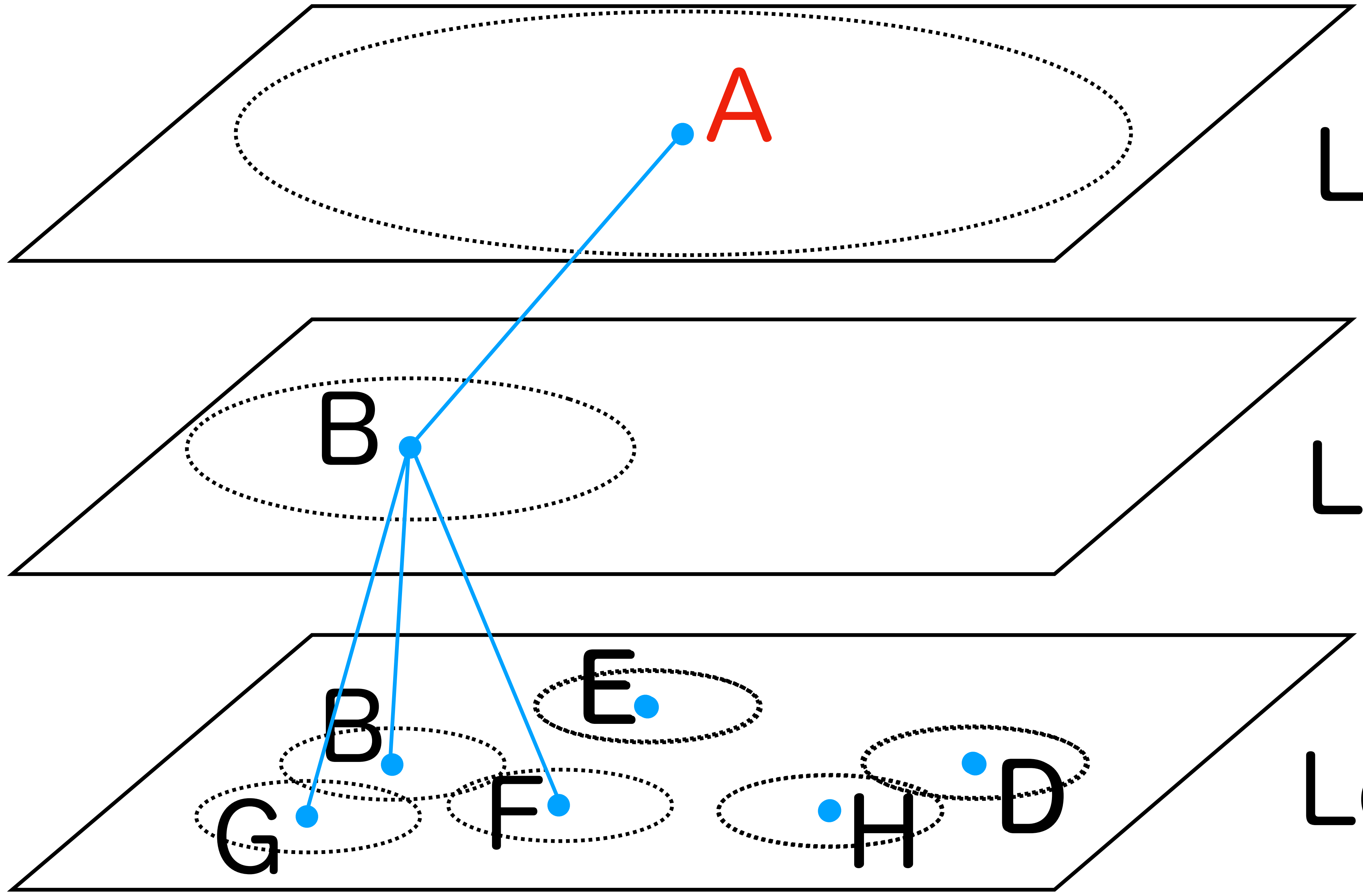
Level 1, r = 2^1

Level 0, r = 2^0

Parallel deletion is similar to insertion

Parallel deletion is in the bottom-up order

Orphans are either reassigned



Level 2, r = 2^2

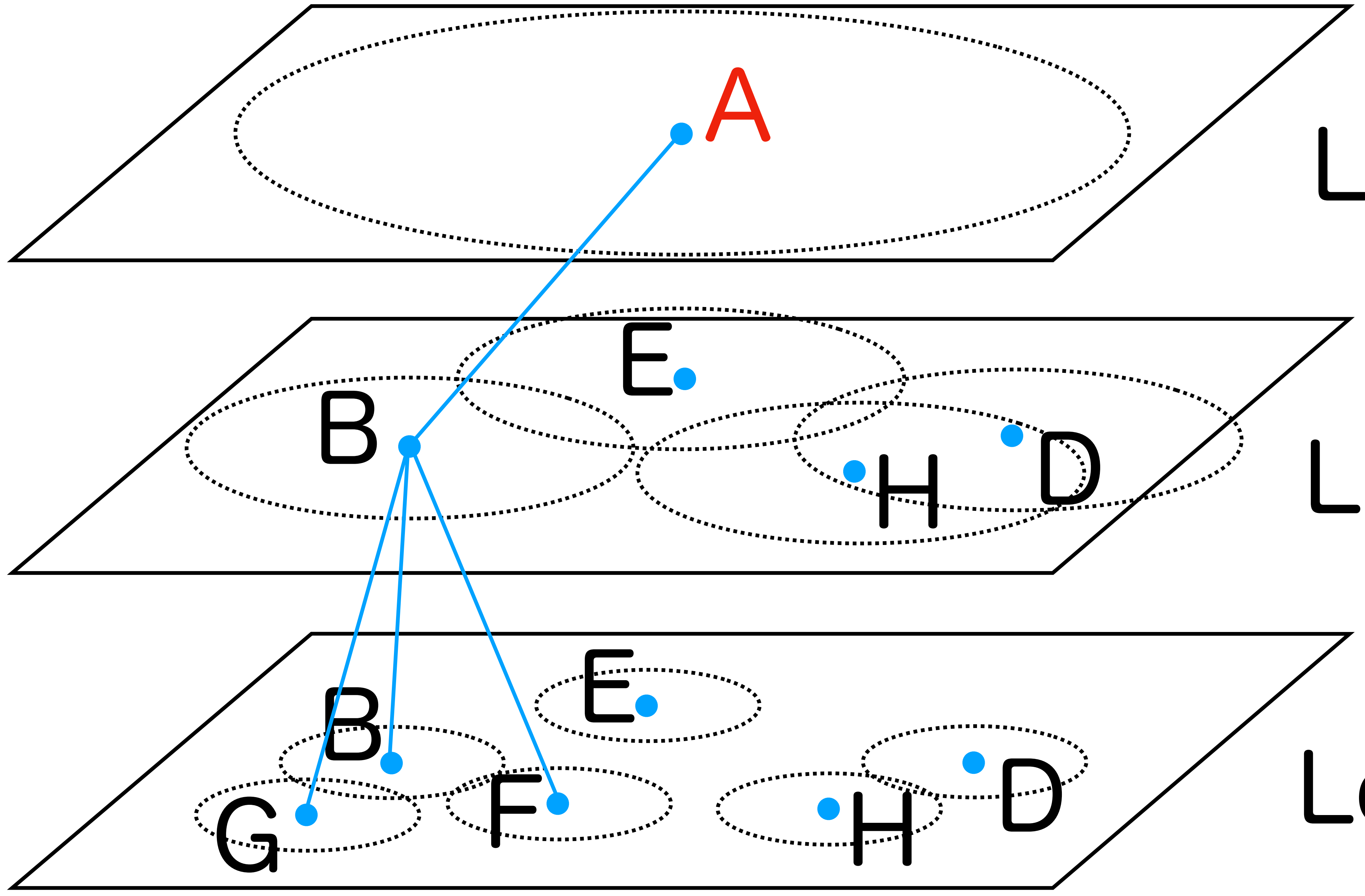
Level 1, r = 2^1

Level 0, r = 2^0

Parallel deletion is similar to insertion

Parallel deletion is in the bottom-up order

Orphans are either redistributed or promoted up



Level 2, r = 2^2

Level 1, r = 2^1

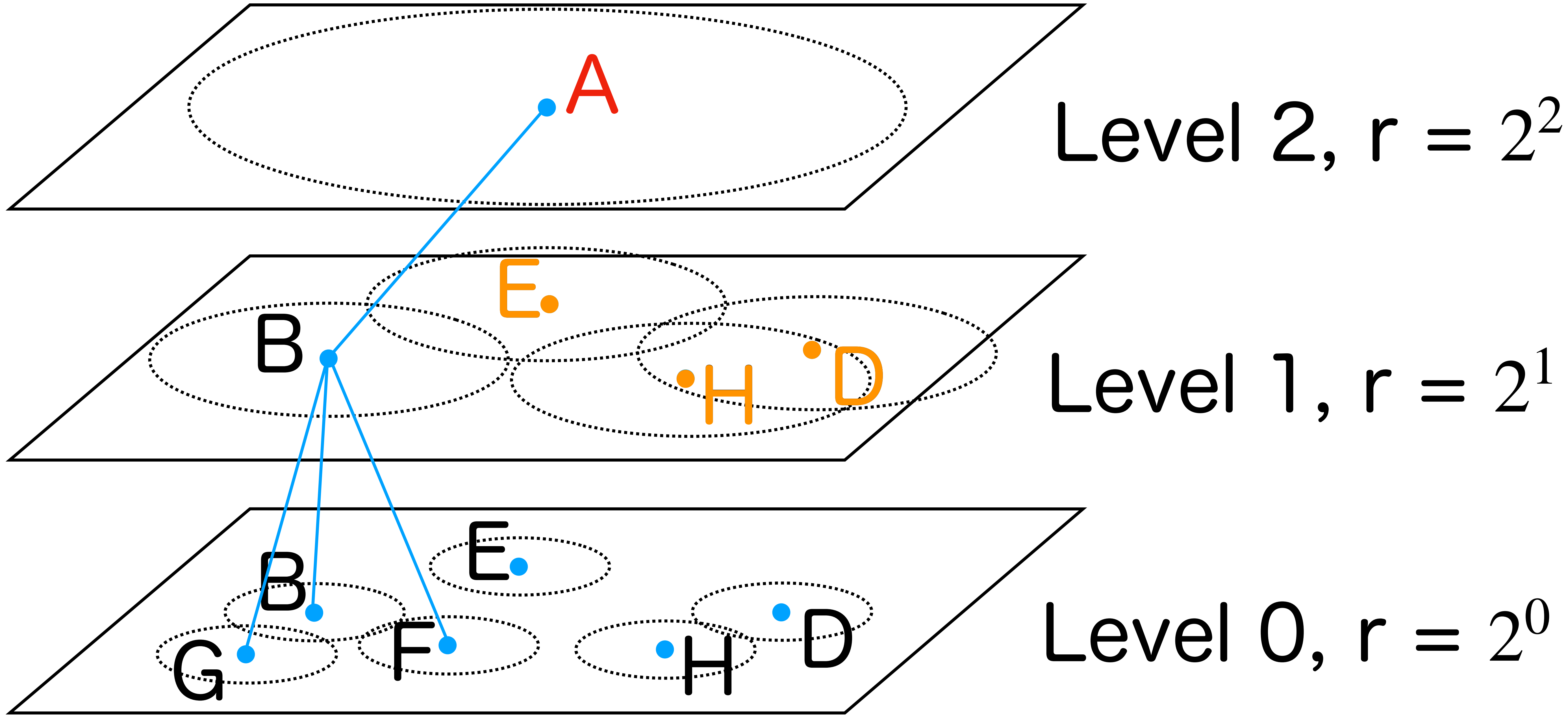
Level 0, r = 2^0

Parallel deletion is similar to insertion

Parallel deletion is in the bottom-up order

Orphans are either redistributed or promoted up

Run MIS on promoted nodes

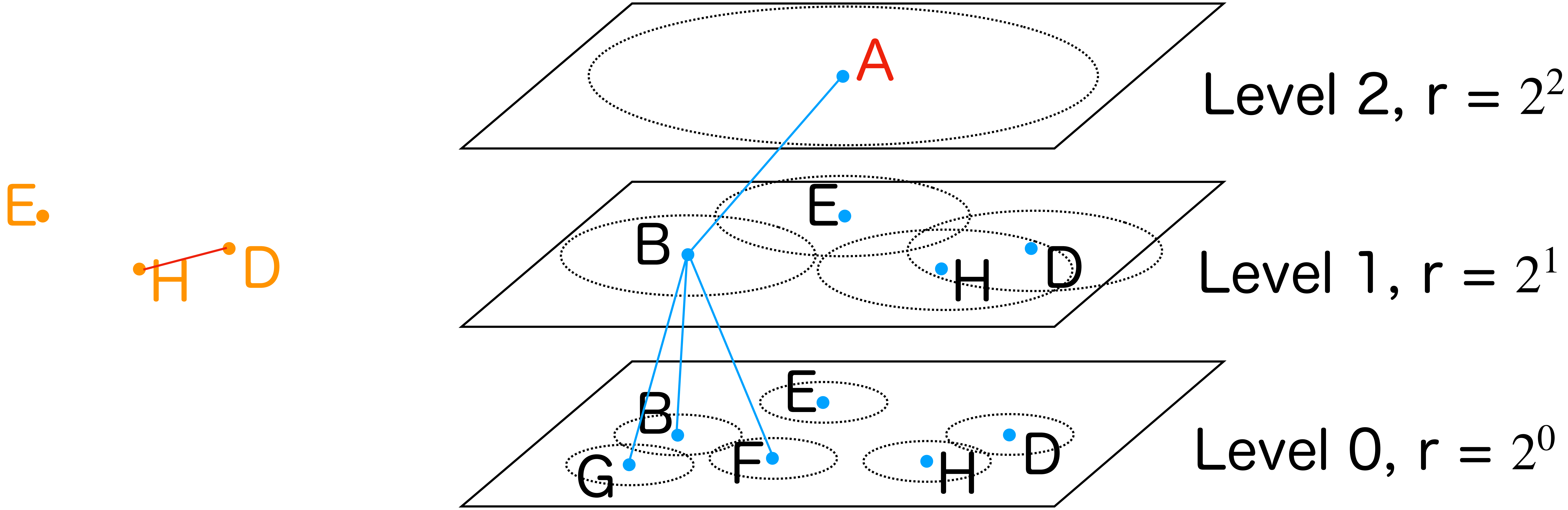


Parallel deletion is similar to insertion

Parallel deletion is in the bottom-up order

Orphans are either redistributed or promoted up

- Run MIS on promoted nodes

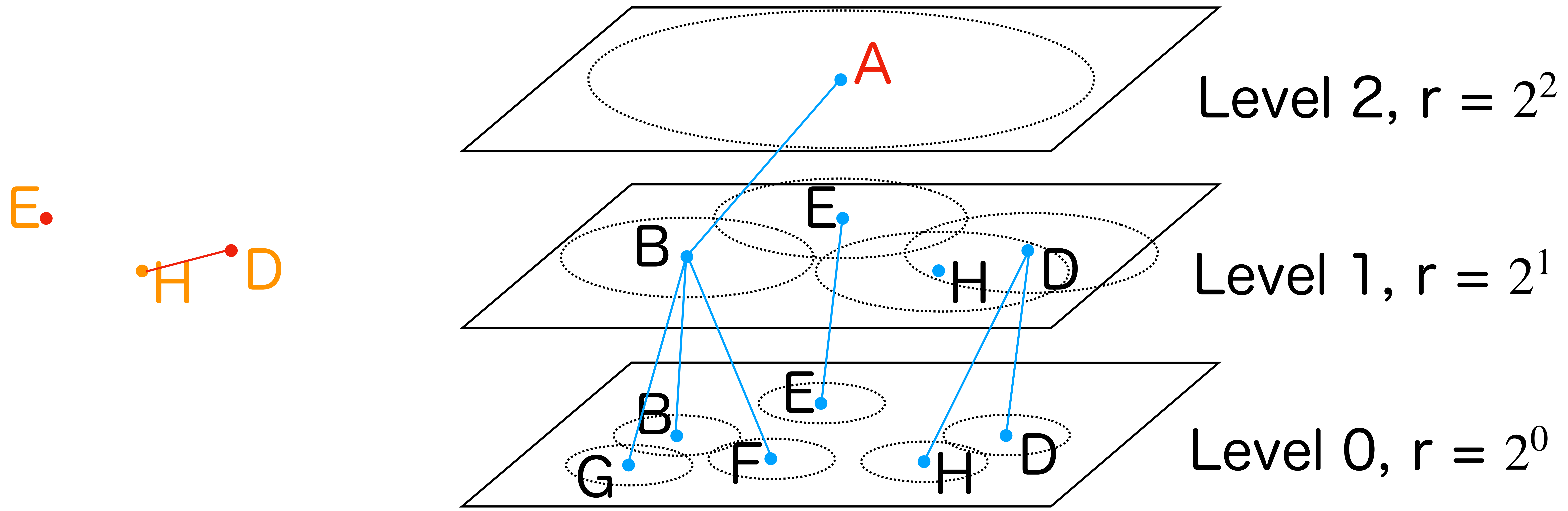


Parallel deletion is similar to insertion

Parallel deletion is in the bottom-up order

Orphans are either redistributed or promoted up

- Run MIS on promoted nodes

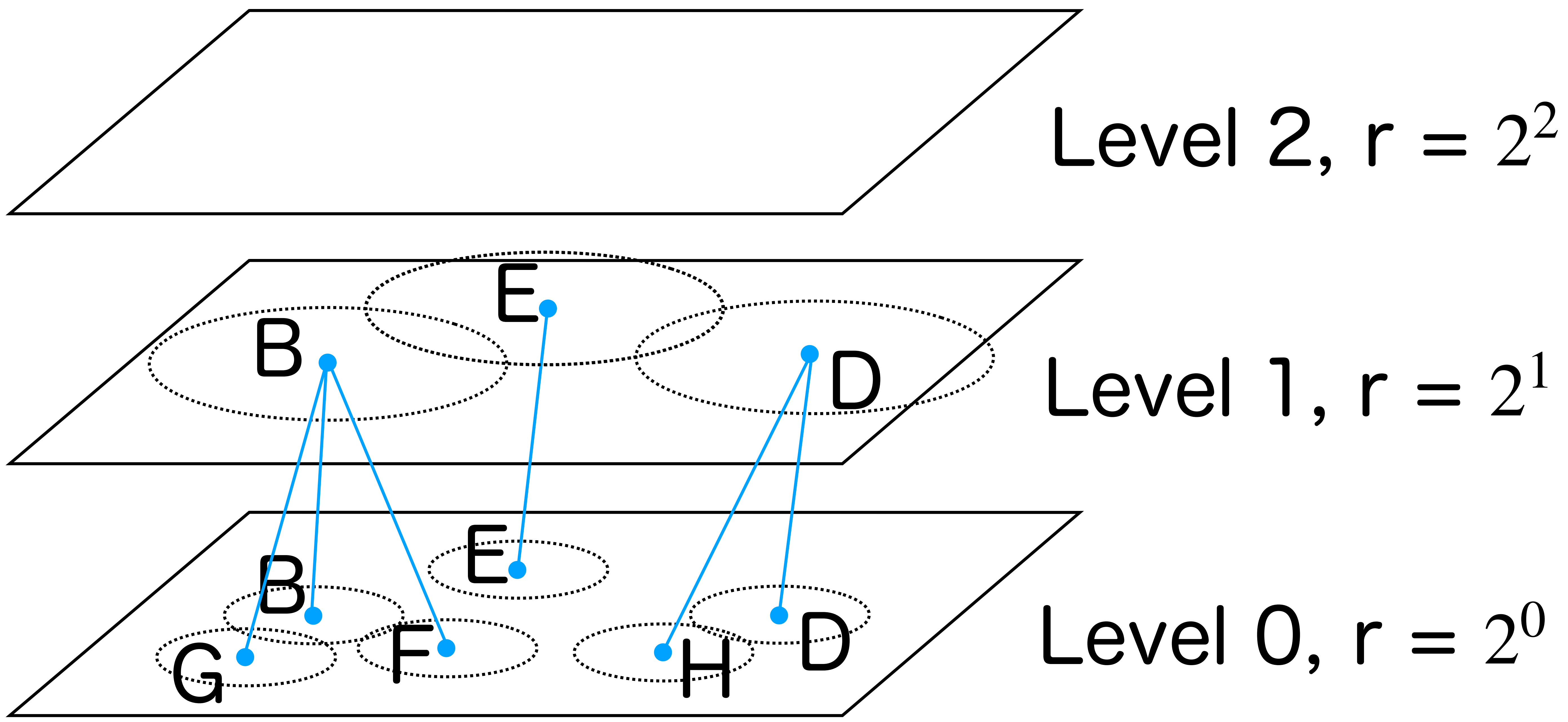


Parallel deletion is similar to insertion

Parallel deletion is in the bottom-up order

Orphans are either redistributed or promoted up

- Run MIS on promoted nodes

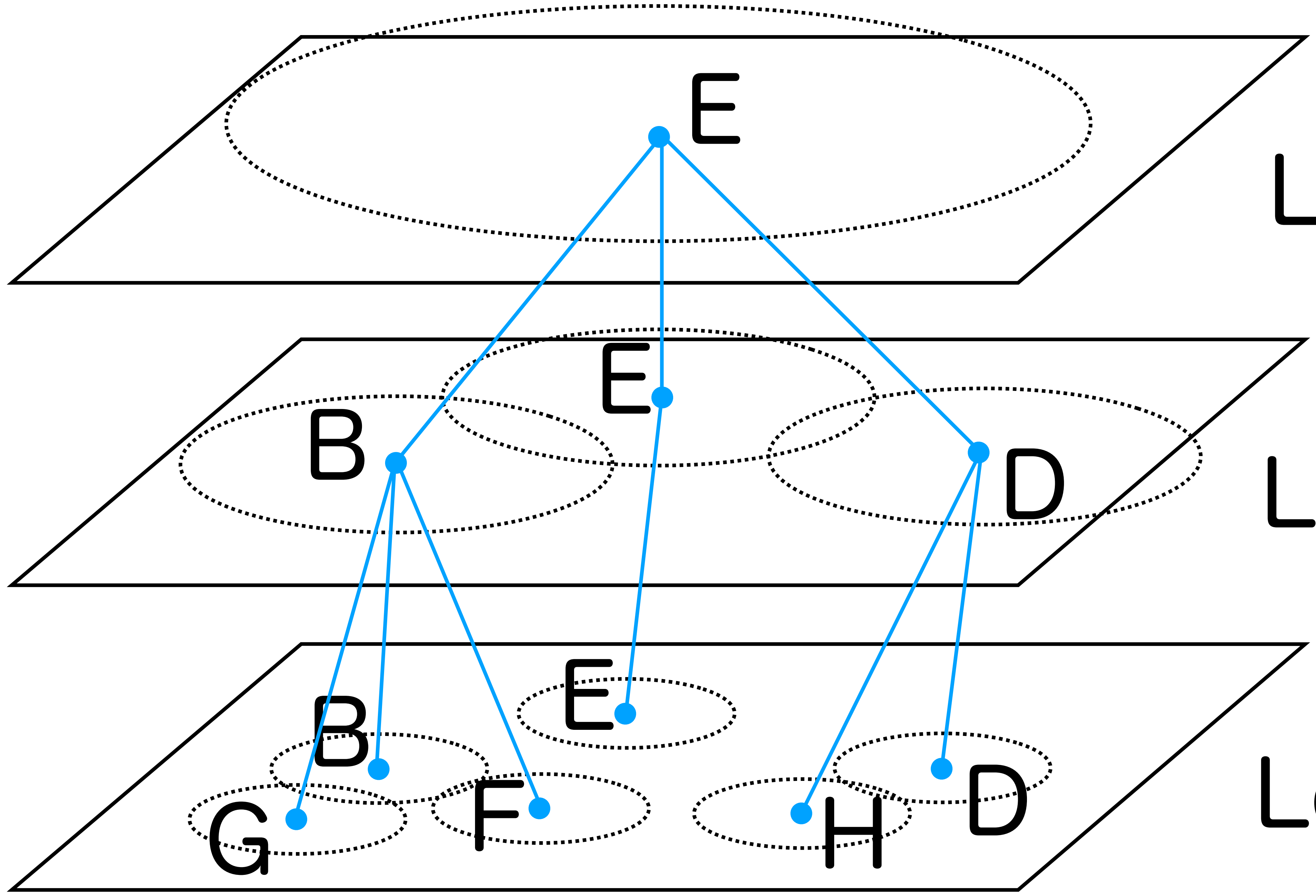


Parallel deletion is similar to insertion

Parallel deletion is in the bottom-up order

Orphans are either redistributed or promoted up

- Run MIS on promoted nodes



Level 2, r = 2^2

Level 1, r = 2^1

Level 0, r = 2^0

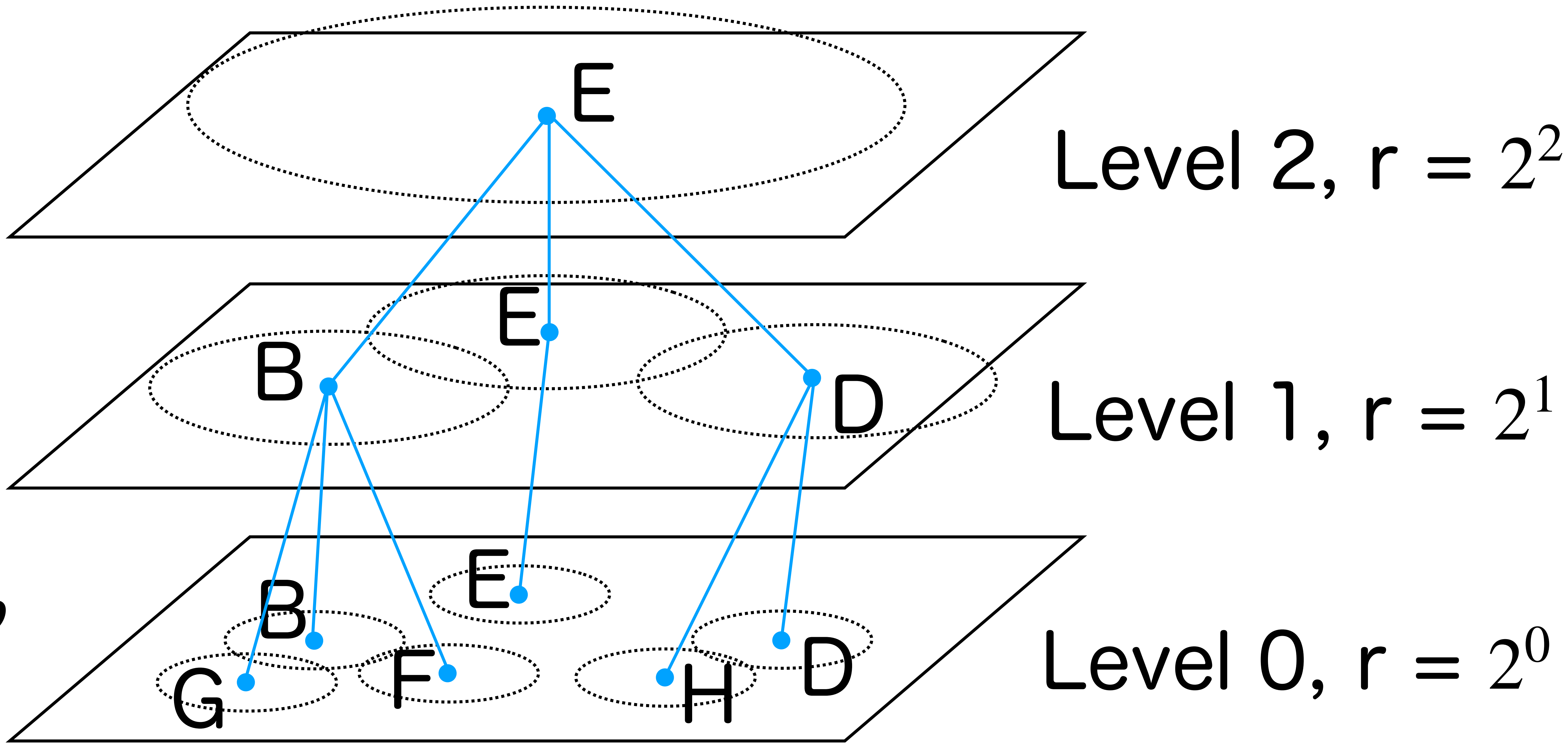
Parallel deletion is similar to insertion

Parallel deletion is in the bottom-up order

Orphans are either redistributed or promoted up

- Run MIS on promoted nodes

Don't need prefix doubling



$O(c^9 mH(T))$ expected work

$O(H(T)\log c(\log c + \log m))$ span *whp*

Applications

- Euclidean Minimum Spanning Tree
- Single-Linkage Clustering
- Bichromatic Closest Pair (BCP)
- Density-Based Clustering

When assuming $c = O(1)$

and $H(T) = \Theta(\log n)$

$\tilde{O}(n)$ expected work

$O(\log^3 n \log\log n)$ span *whp*

- k-NN Graph Construction

$O(kn \log k \log n)$ work

$O(\log n \cdot (k \log k + \log^2 n \log\log n))$ span *whp*

Applications

Euclidean Minimum Spanning Tree

Single-Linkage Clustering

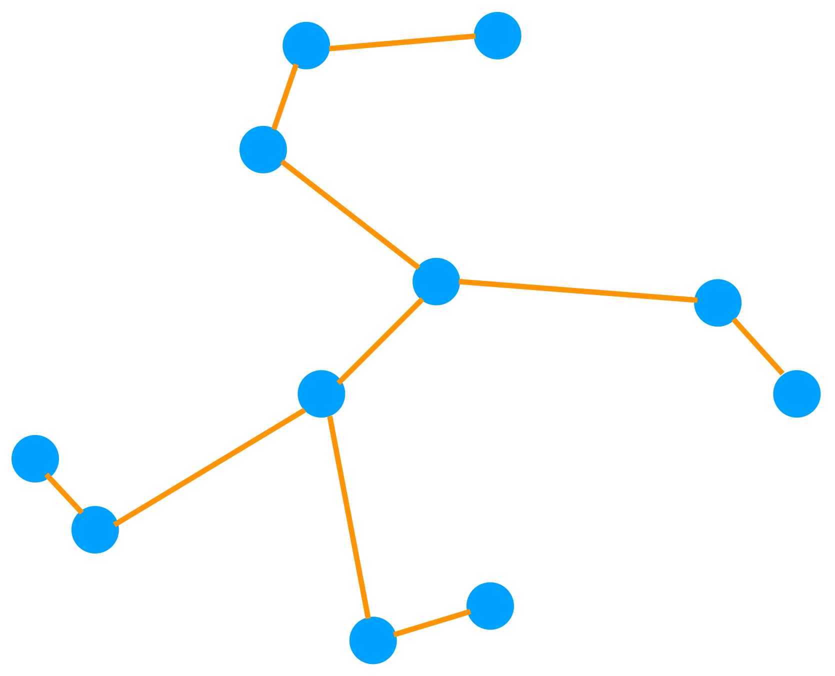
Bichromatic Closest Pair (BCP)

Density-Based Clustering

k-NN Graph Construction

Euclidean Minimum Spanning Tree (EMST)

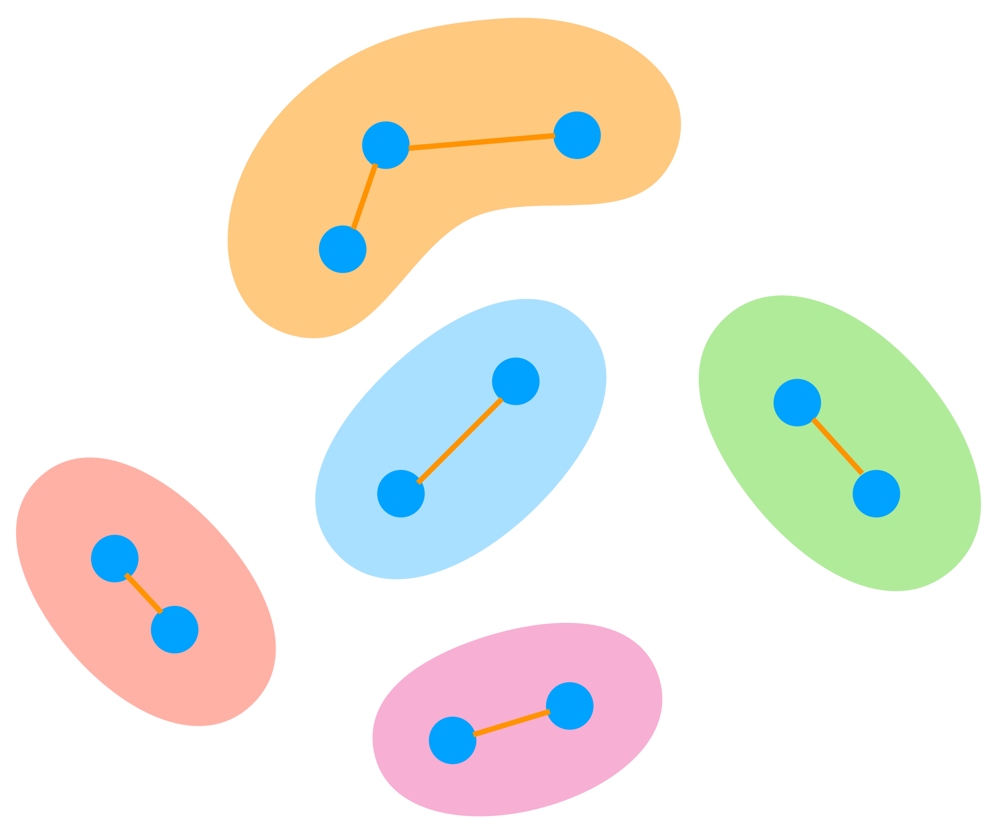
Given a set of n points $S \in \mathbb{R}^d$, EMST finds the MST on the complete graph constructed from S, where edge weights are pairwise Euclidean distances.



We apply parallel cover tree to Borůvka's MST algorithm

Euclidean Minimum Spanning Tree (EMST)

Given a set of n points $S \in \mathbb{R}^d$, EMST finds the MST on the complete graph constructed from S, where edge weights are pairwise Euclidean distances.



We apply parallel cover tree to Borůvka's MST algorithm

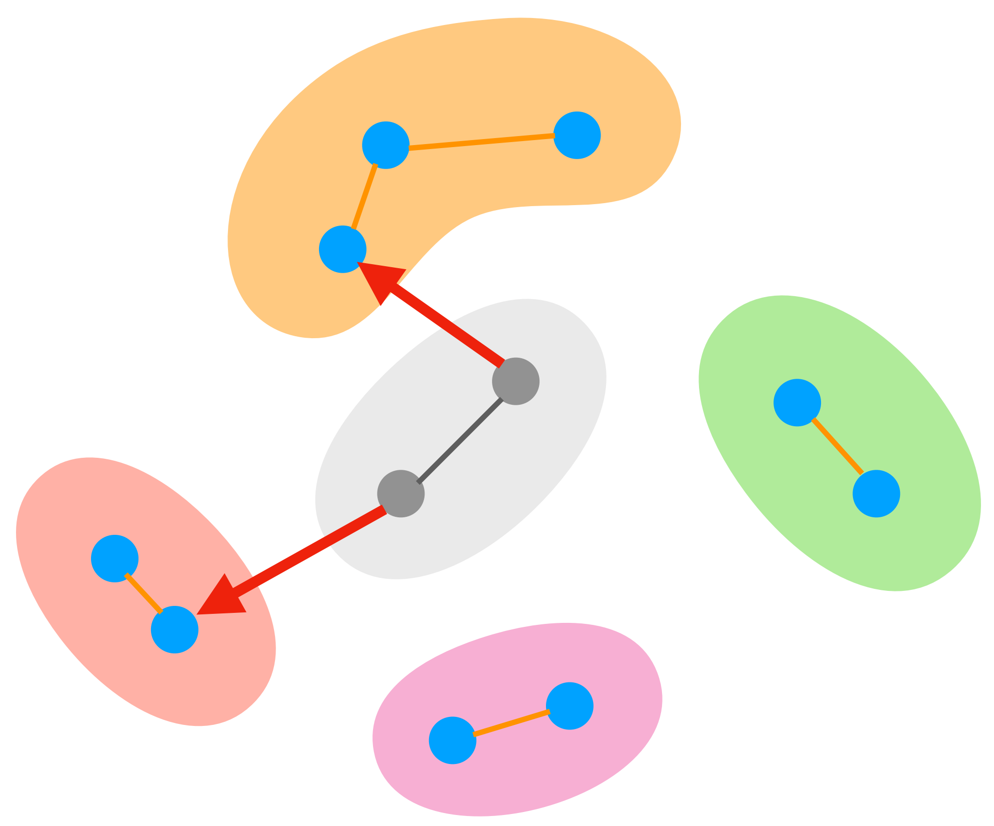
Cluster-NN:

Construct a cover tree on n points

Delete the cluster

Euclidean Minimum Spanning Tree (EMST)

Given a set of n points $S \in \mathbb{R}^d$, EMST finds the MST on the complete graph constructed from S, where edge weights are pairwise Euclidean distances.



We apply parallel cover tree to Borůvka's MST algorithm

Cluster-NN:

Construct a cover tree on n points

Delete the cluster → Query NNS → Take minimum

Query cluster-NN in parallel?

Persistent Trees

Conclusion

Cover tree | Low expansion rate | Bounded aspect ratio → Efficient NNS

↓

Parallel Insertion/Deletion

- MIS Correctness & Parallelism
- Prefix-doubling Work-efficiency

m is number of inserted/deleted points

n is cover tree size

Insertion: $O(m \log n)$ expected work, $O(\log n \log^2 m \log \log n)$ span *whp*

Deletion: $O(m \log n)$ expected work, $O(\log n \log m)$ span *whp*

↓

Applications

Persistent-tree