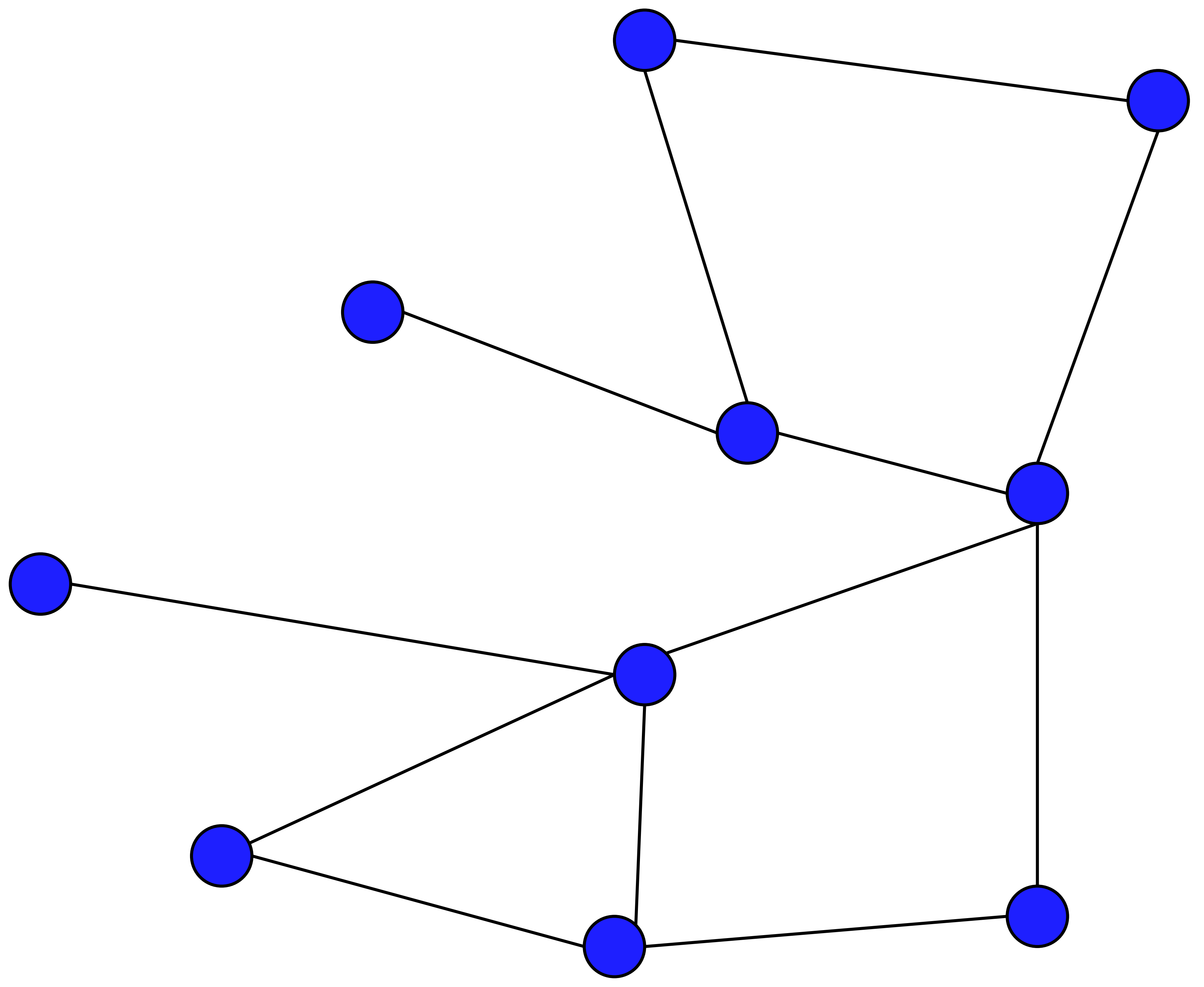
Project Catapult

…one month later

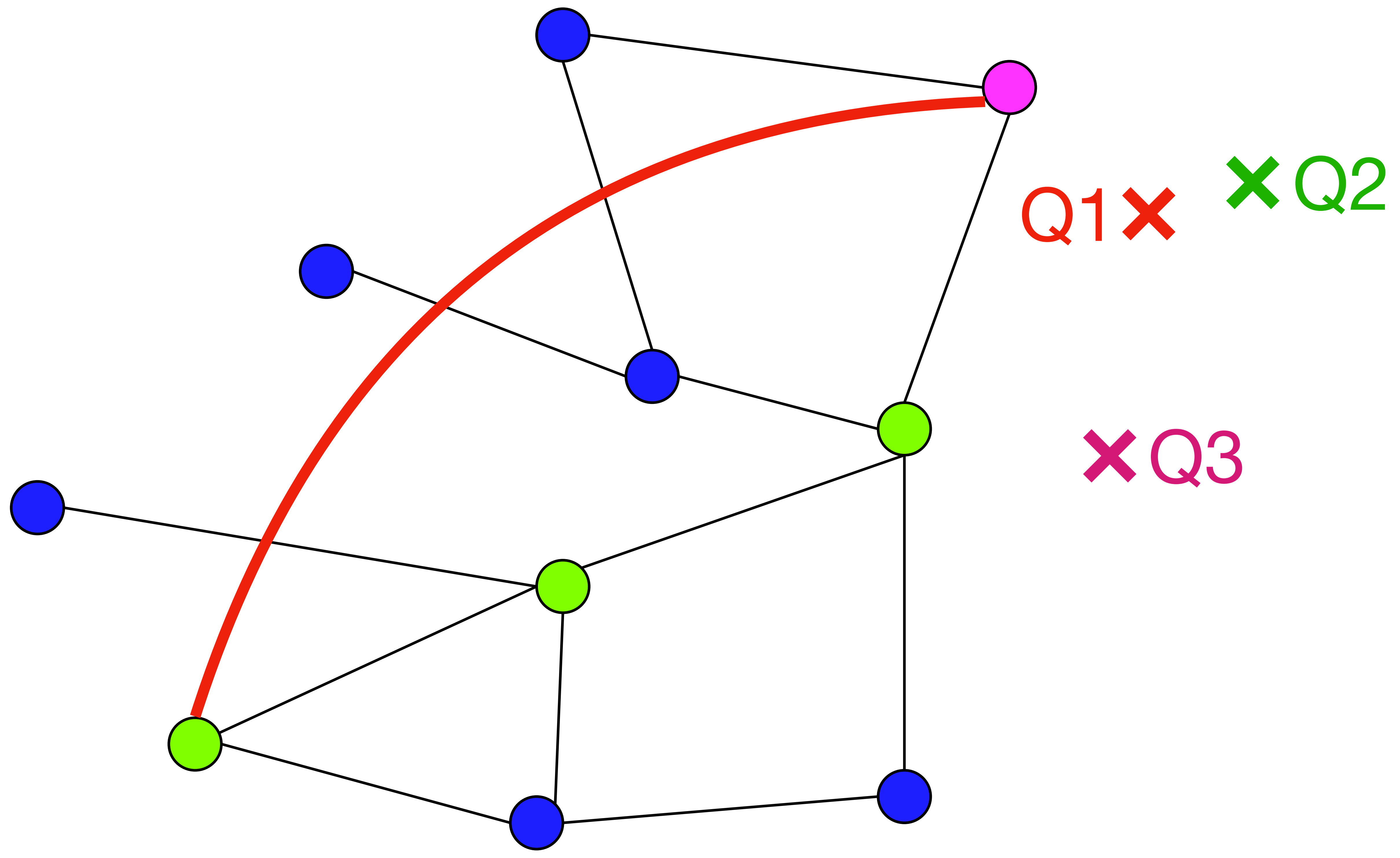
Mathis Randl for SaCS, 15/10/25

Graph databases



✕ Q1

Catapults to the rescue



What do we have now?

What do we have?

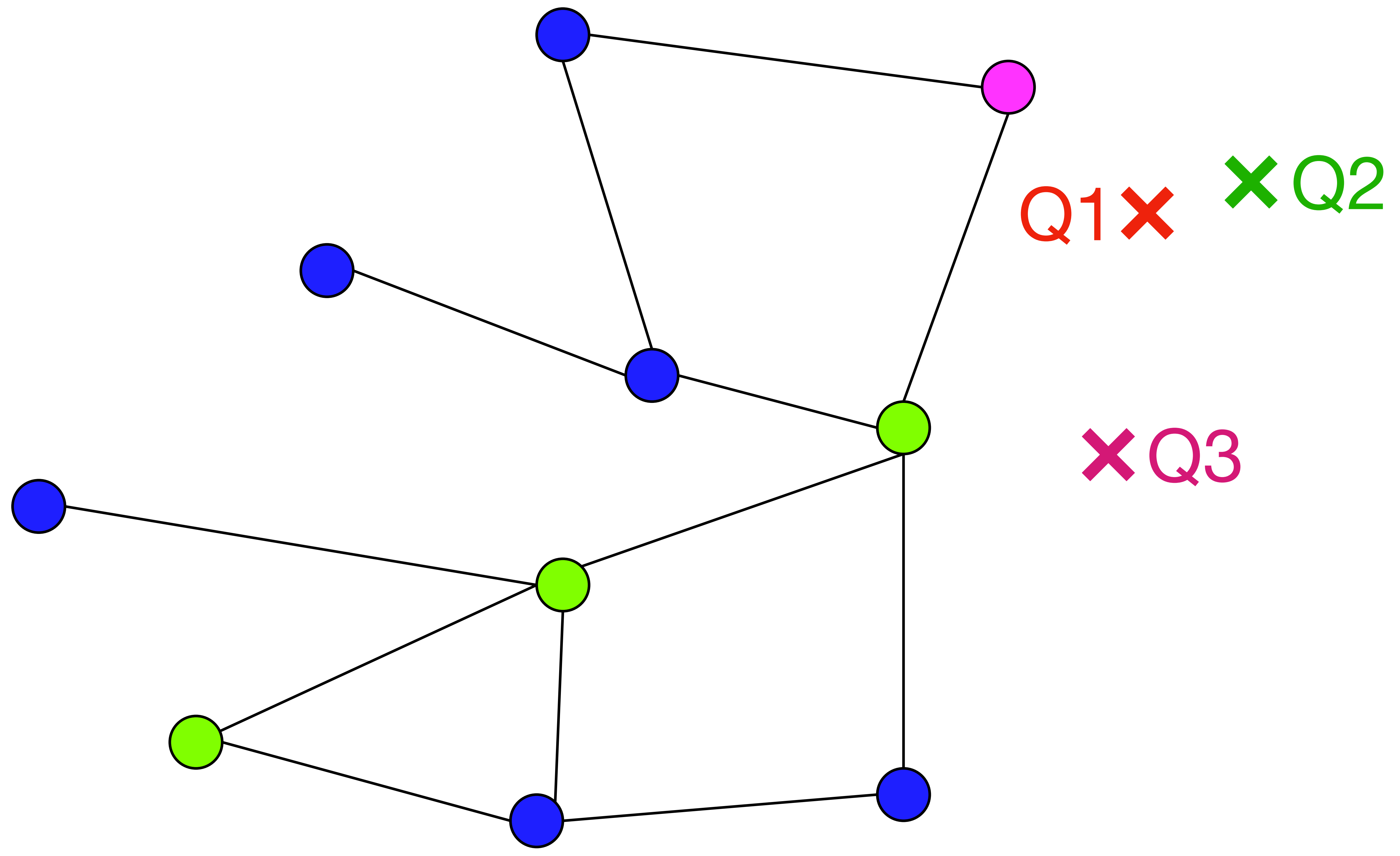
- Search engine operational, including catapults

- Some things still WIP

- Works on toy data / large-scale experiments coming

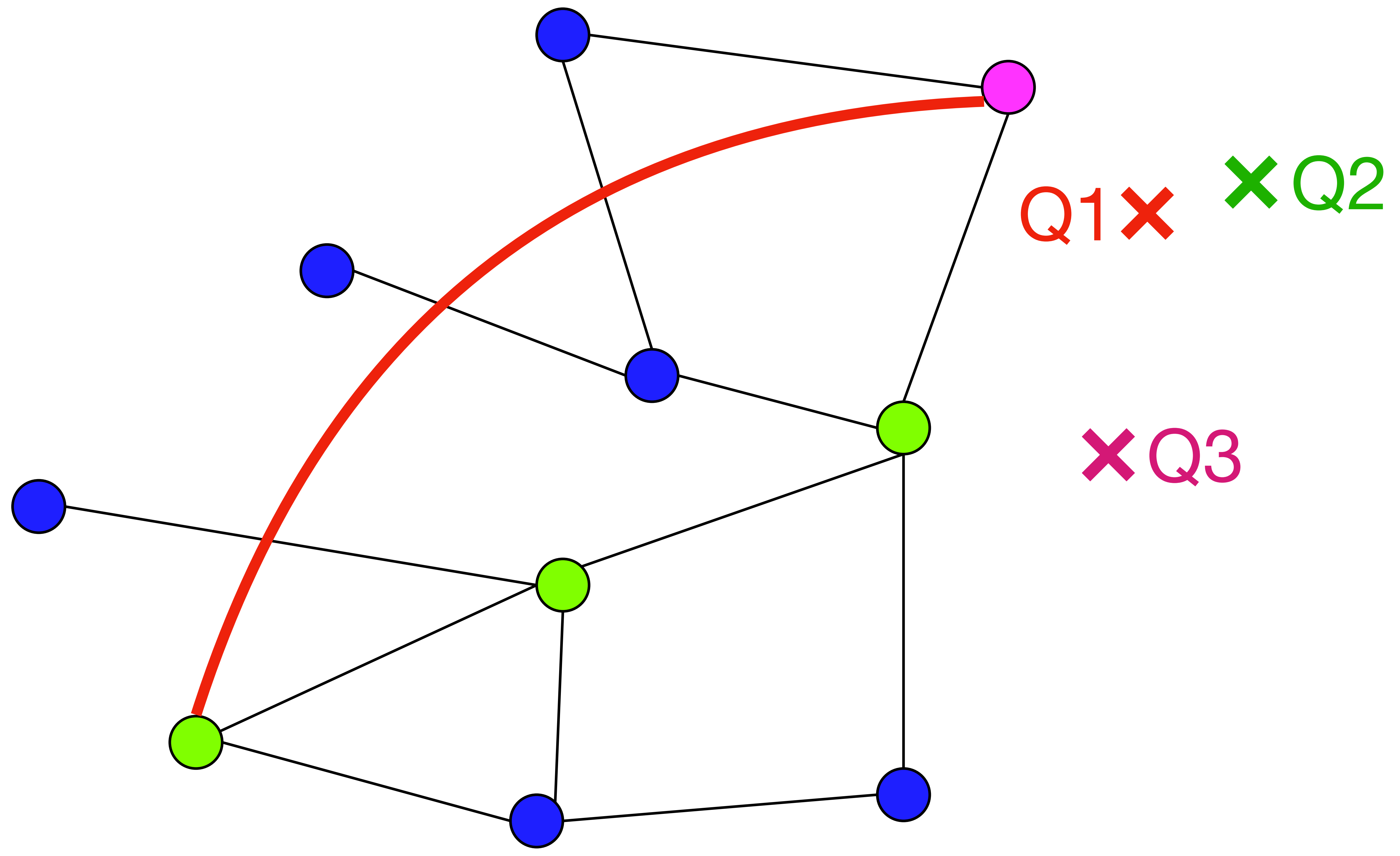
Generic graph search (baseline)

✅ implemented



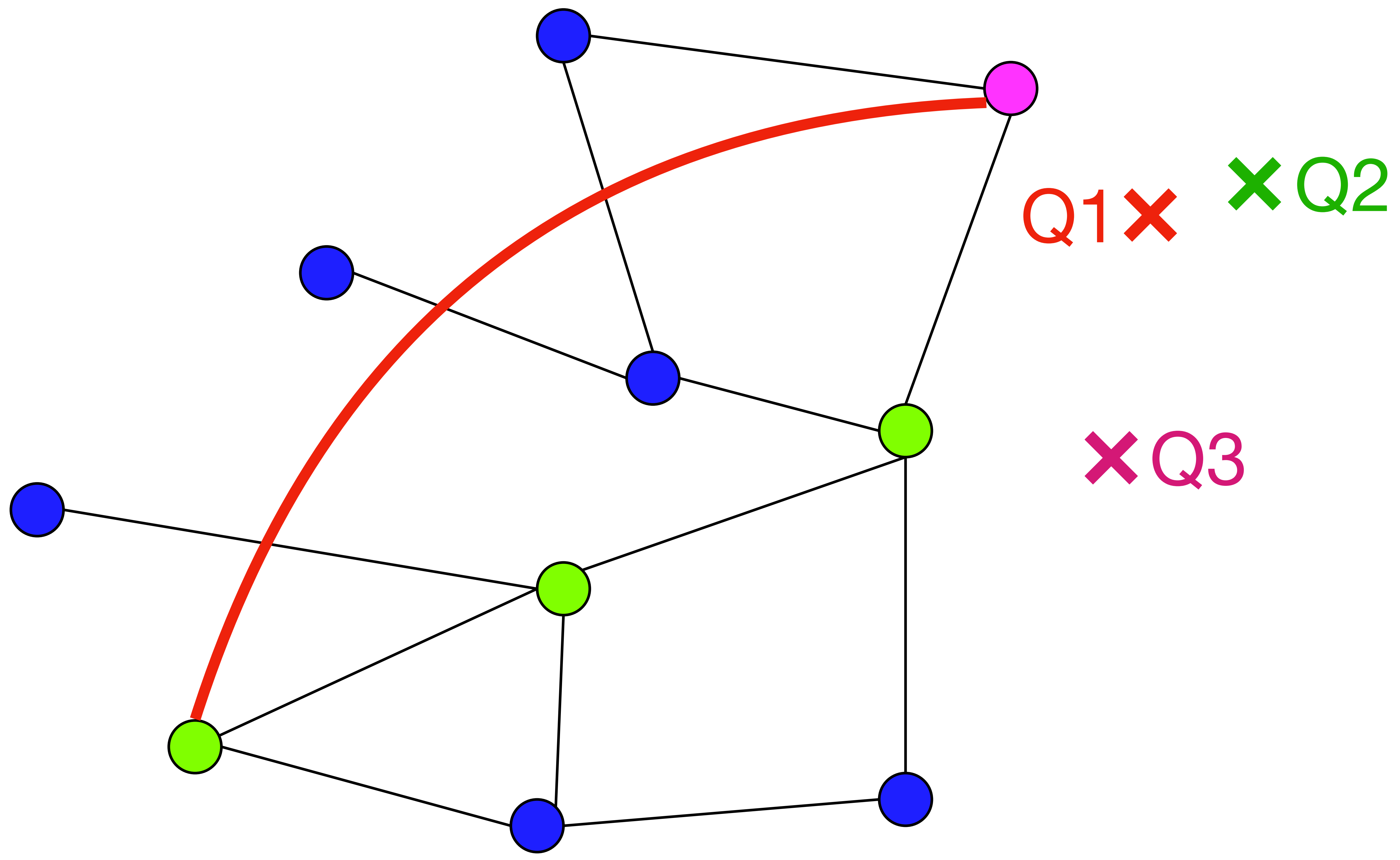
Catapult management

🟧 **Implemented with FIFO only**



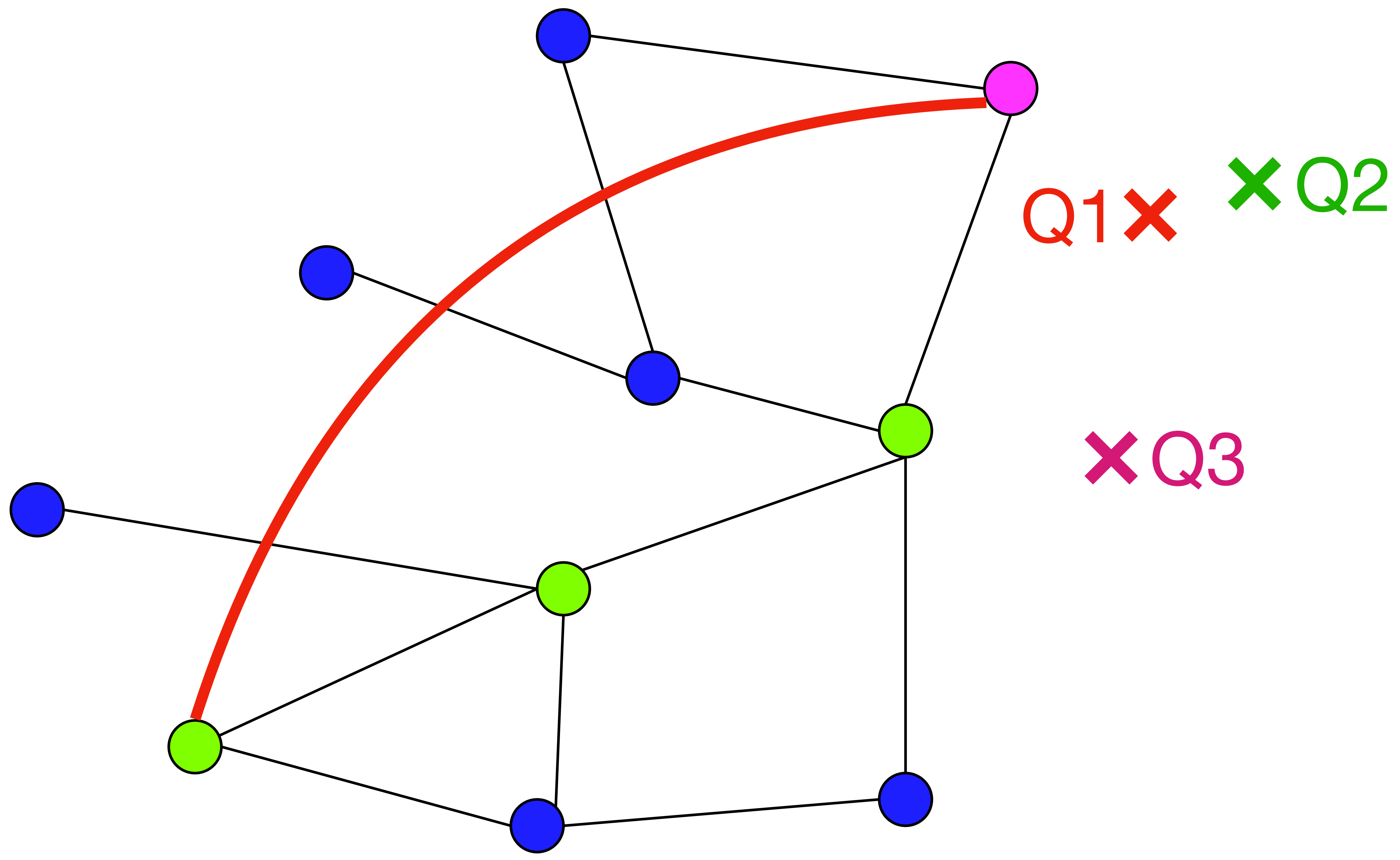
LSH for entry-point selection

✅ **implemented**



LSH for entry-point selection

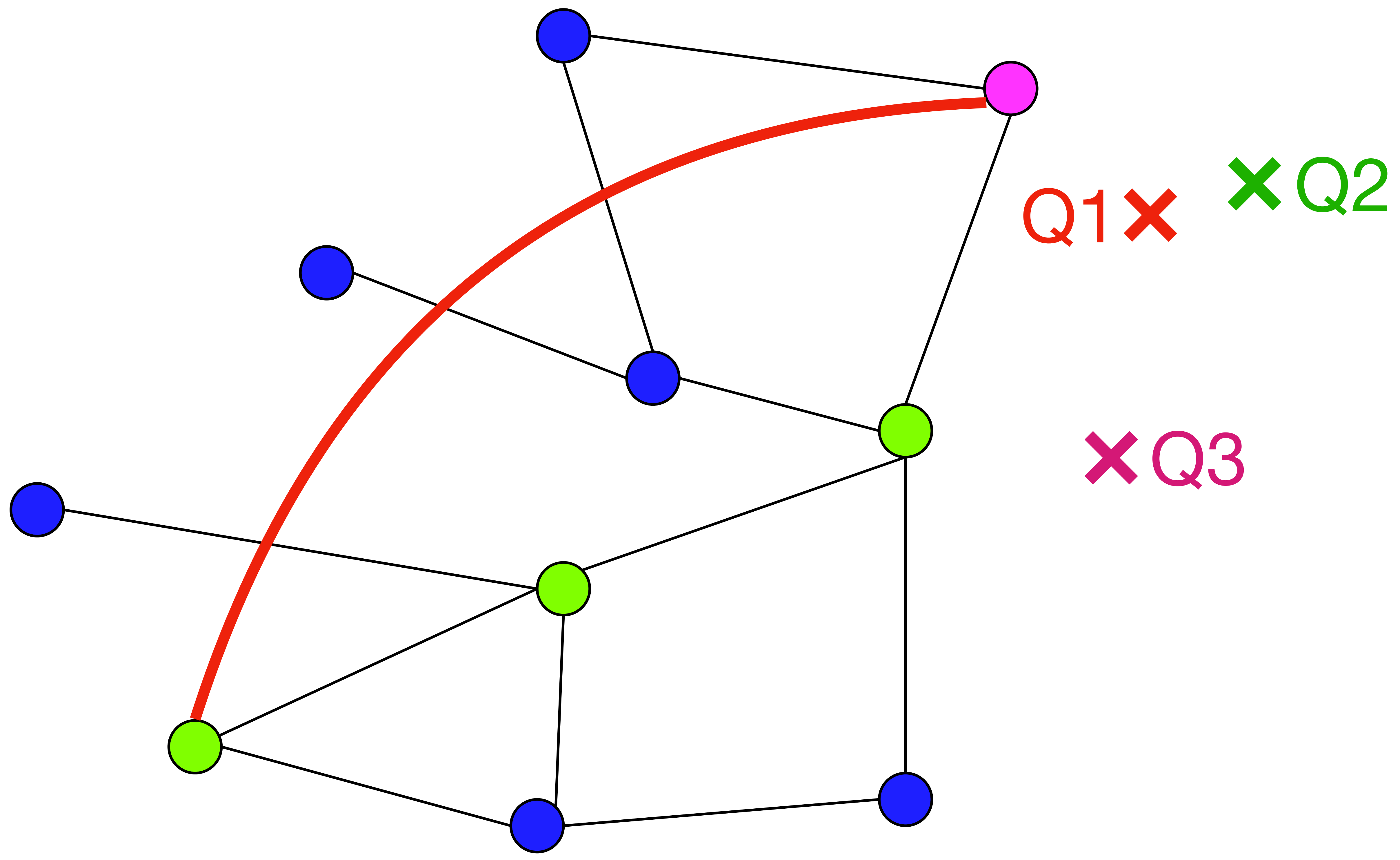
✅ implemented



Hash bits	Starting nodes
11001	1, 4, 5
10010	9, 11, 18
…	…

LSH for entry-point selection

✅ implemented



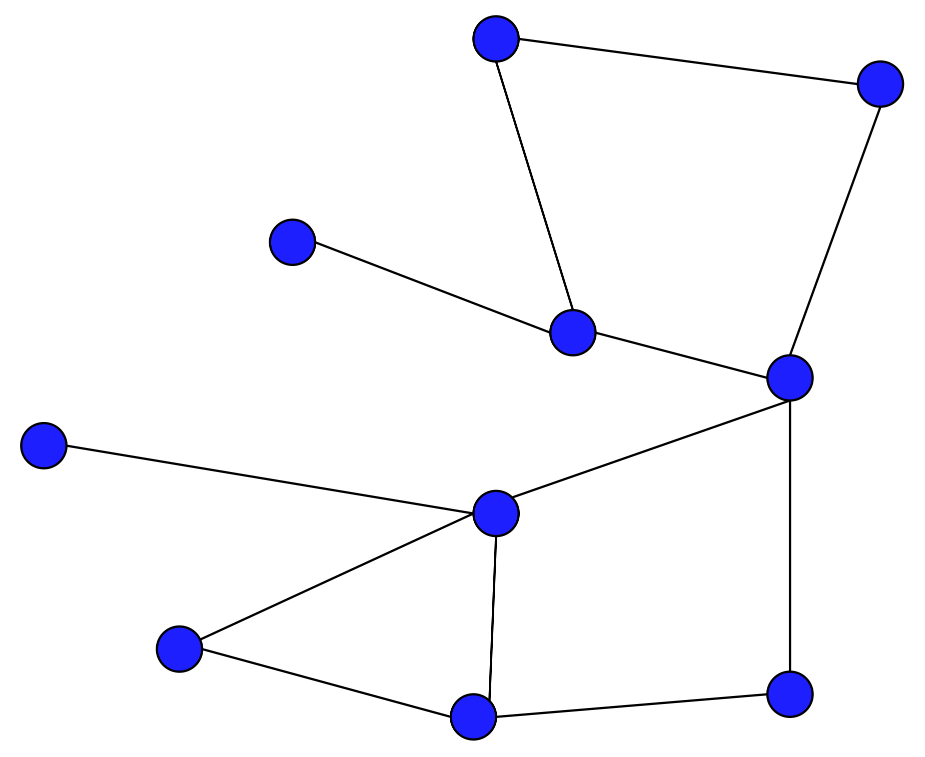
Hash bits	Starting nodes
11001	1, 4, 5
10010	9, 11, 18
…	…

New entries are random

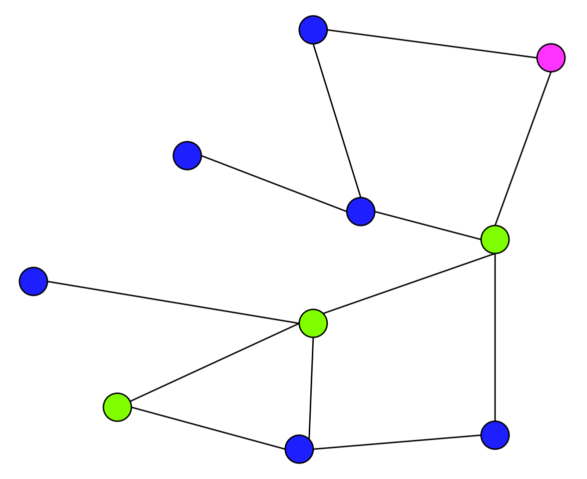
What now?

Hijacking the initial graph building

🟧 **WIP**



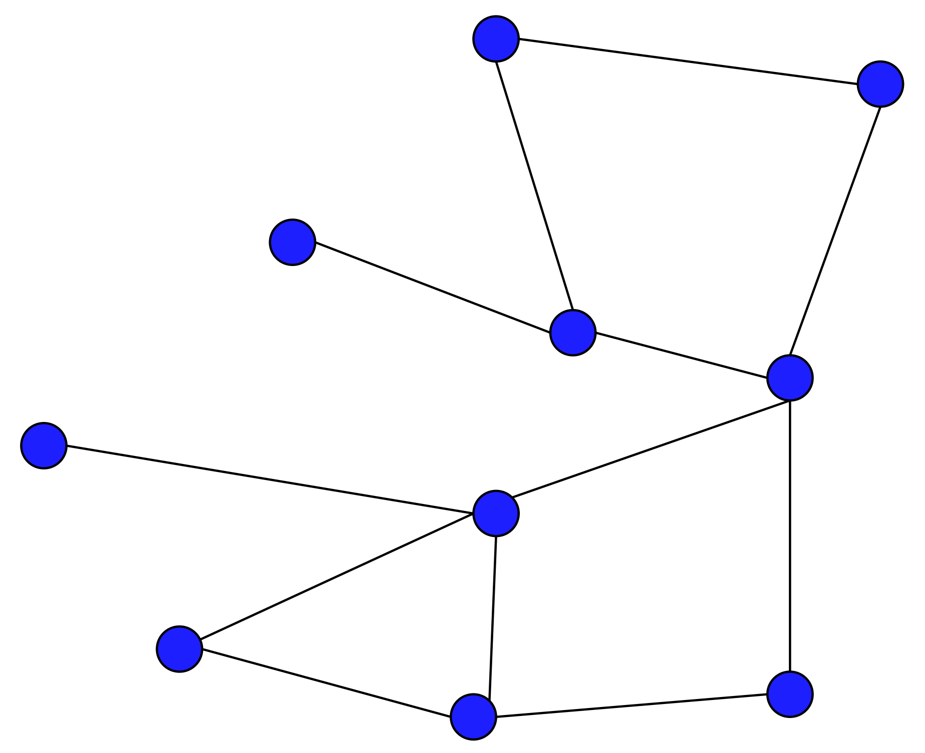
Vamana builds
a graph



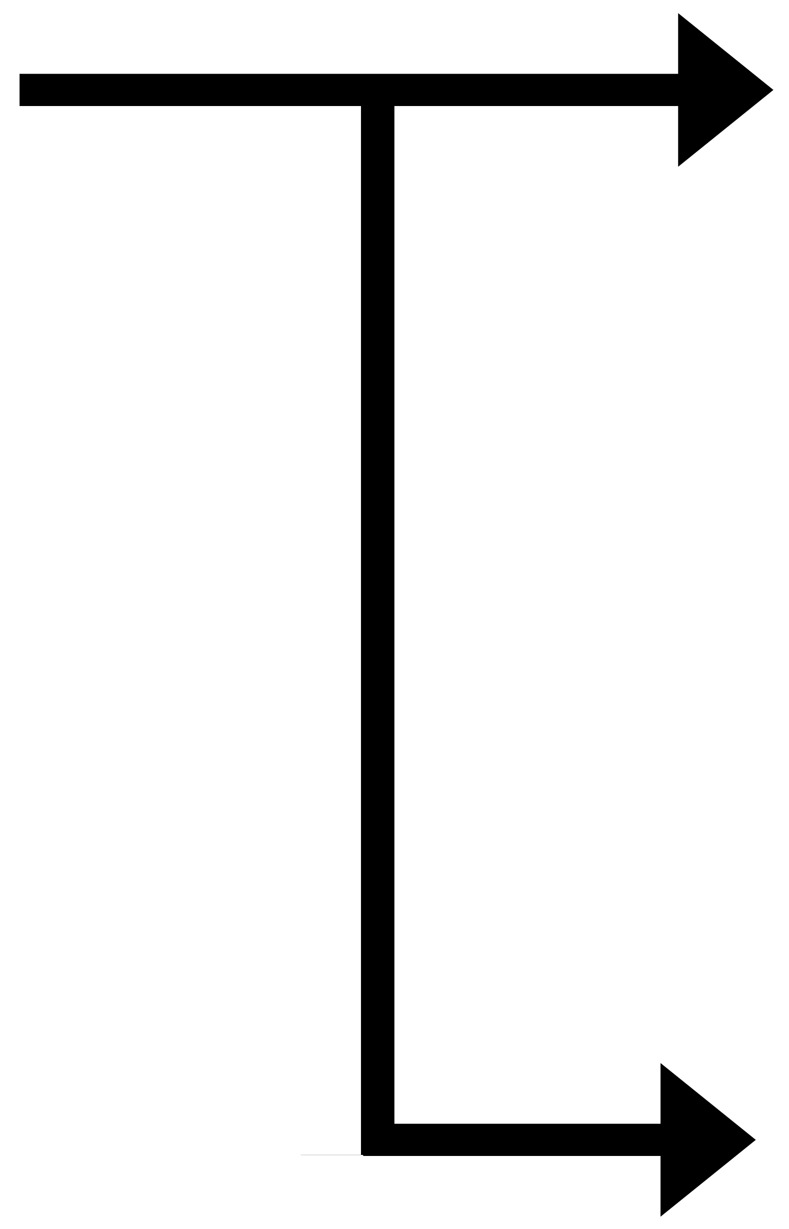
DiskANN
exploits it

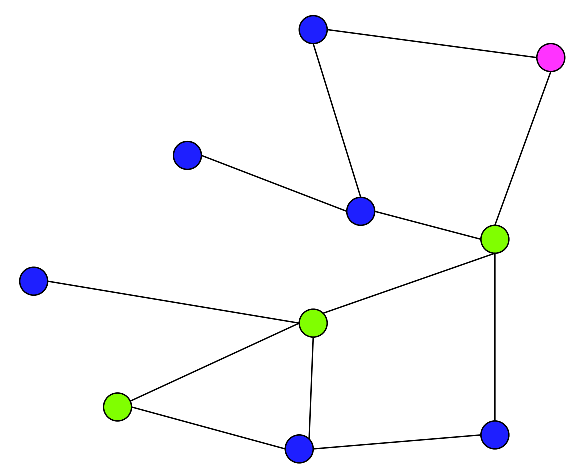
Hijacking the initial graph building

🟧 **WIP**

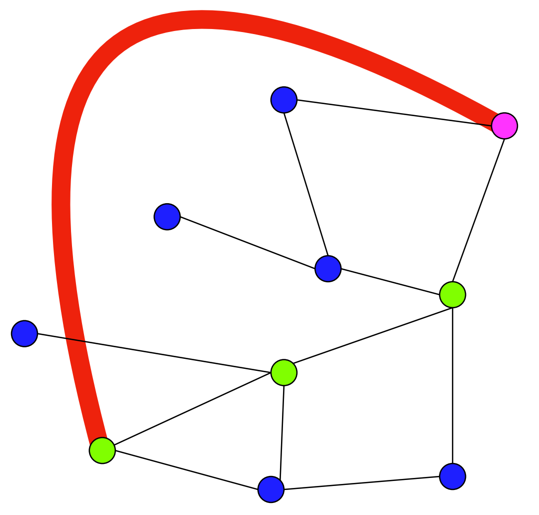


Vamana builds a graph





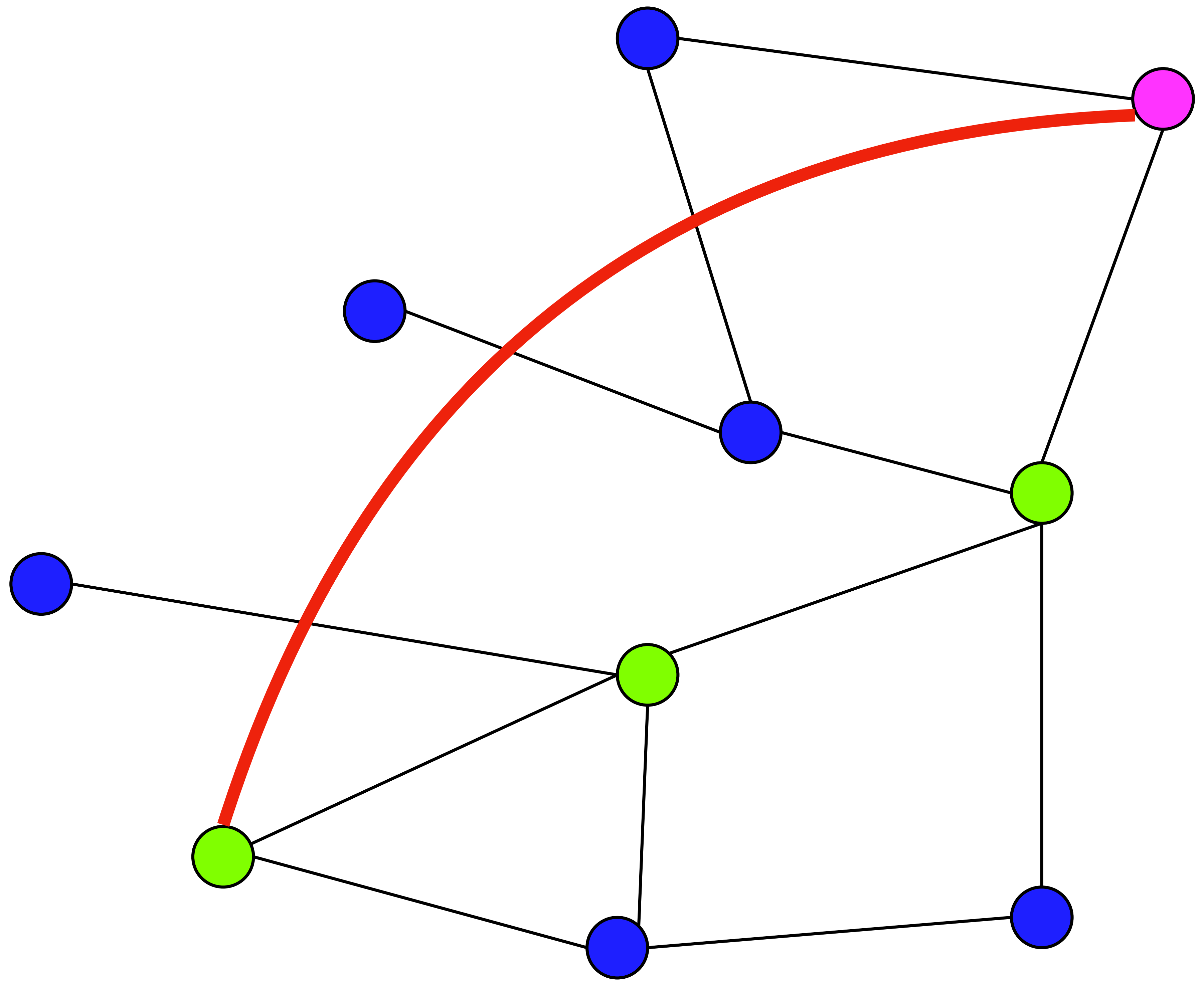
DiskANN exploits it



Catapult exploits it

Multithreading

❌ **not implemented**



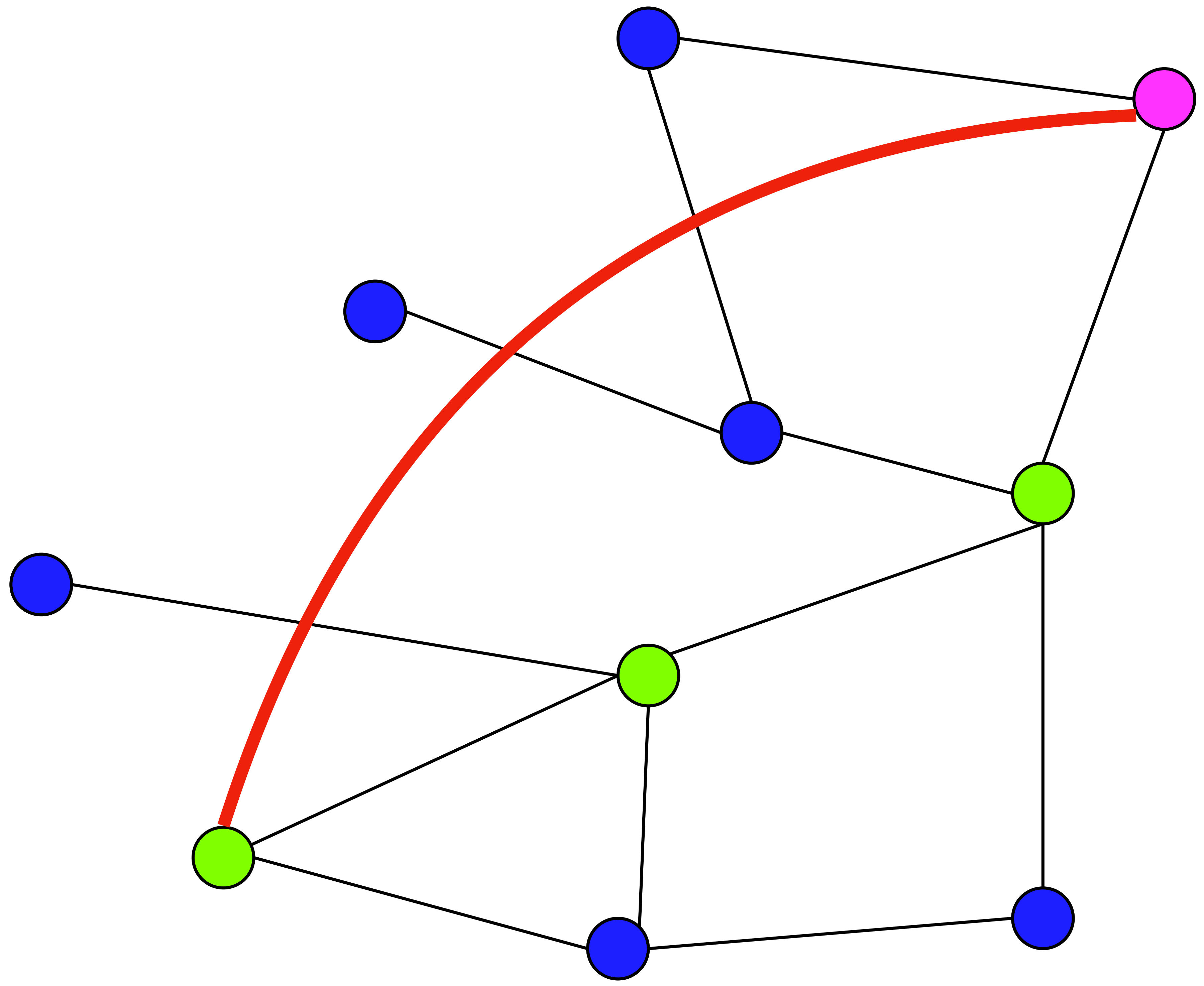
```rust
pub struct Node {
    pub neighbors: NeighborSet,
    pub catapults: NeighborSet,
    pub payload: Box<[f32]>,
}
```

```rust
pub type NeighborSet = Vec<usize>;
```

-> Not thread-safe

Profiling / CI / regression analysis

❌ **not implemented**



<- What is slow in there?
What should be optimized?

Core experiment of paper

- Show we are better than baseline and/or DiskANN C++ implementation
- #hops, #edges explored, ~~wall time,~~ latency, throughput, quality of vectors
- Involve some other databases? FAISS-HNSW?
- CPU only (Catapult/graph-based things not GPU friendly)

Conclusion

- « Tout va bien dans le meilleur des mondes » [1]

[1] *Candide*, Voltaire