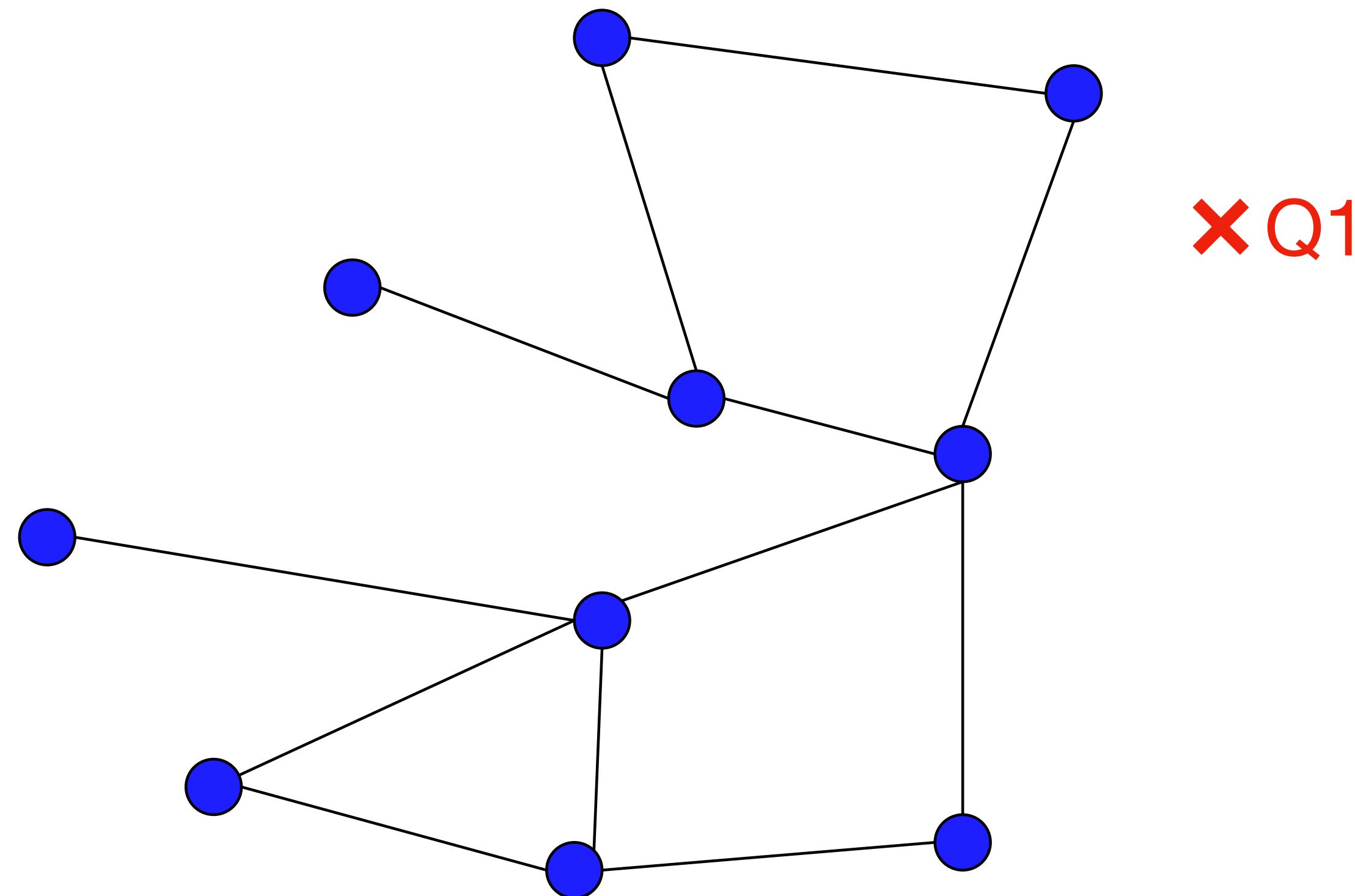


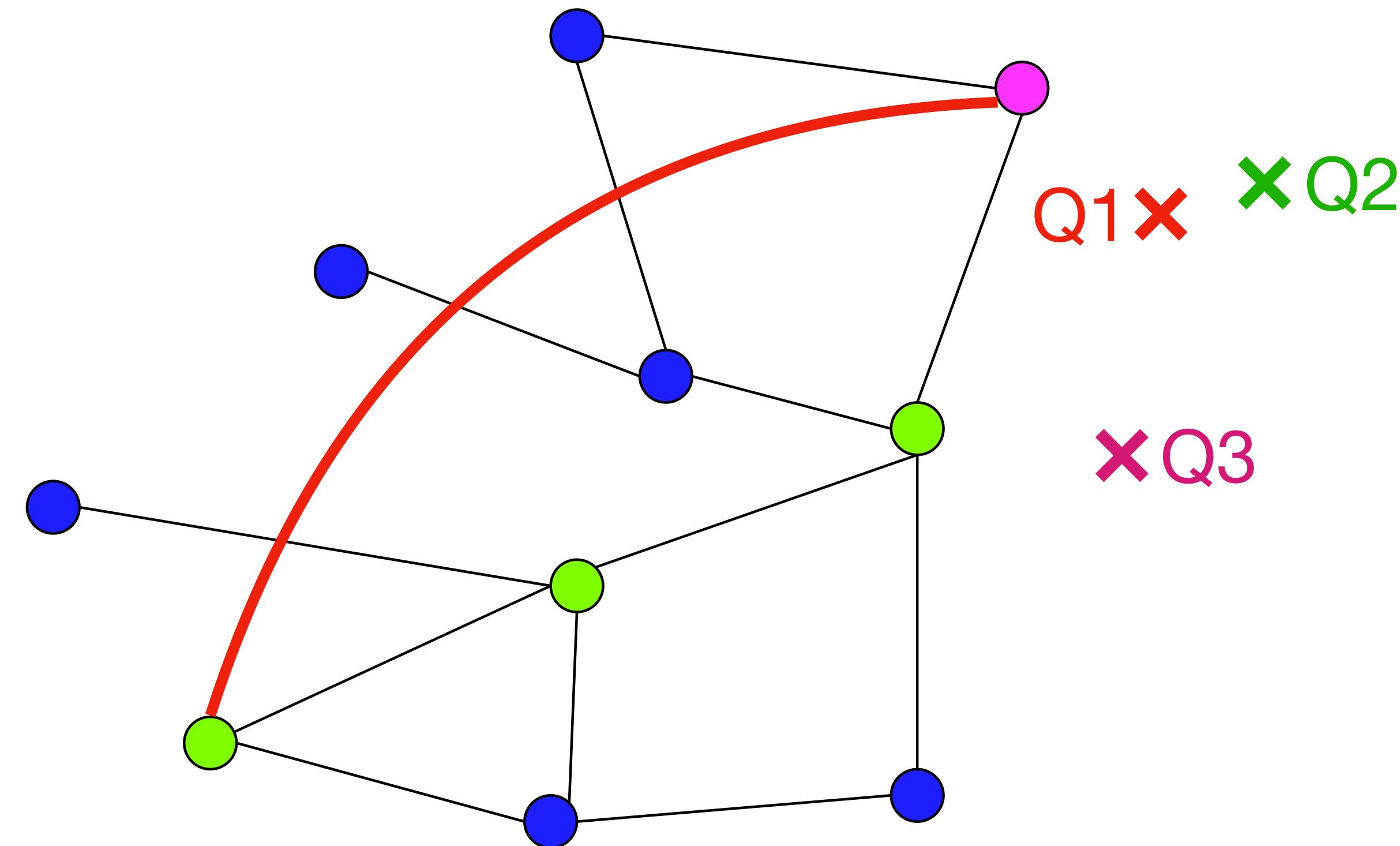
# **Project Catapult**

**...one month later**

# Graph databases



# Catapults to the rescue



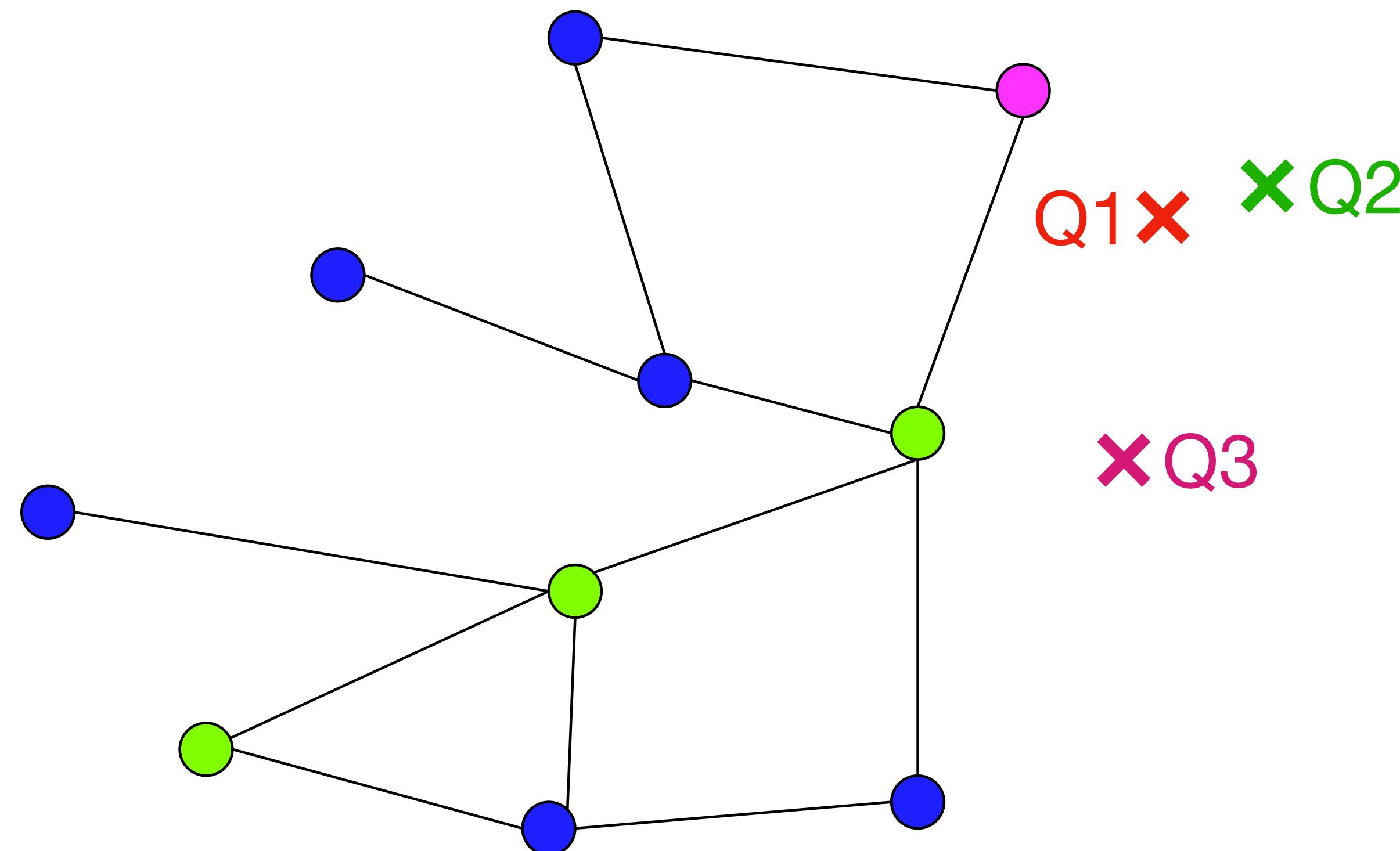
**What do we have now?**

# Where are we standing?

- Search engine operational, including catapults
- Some things still WIP
- Works on toy data / large-scale experiments coming

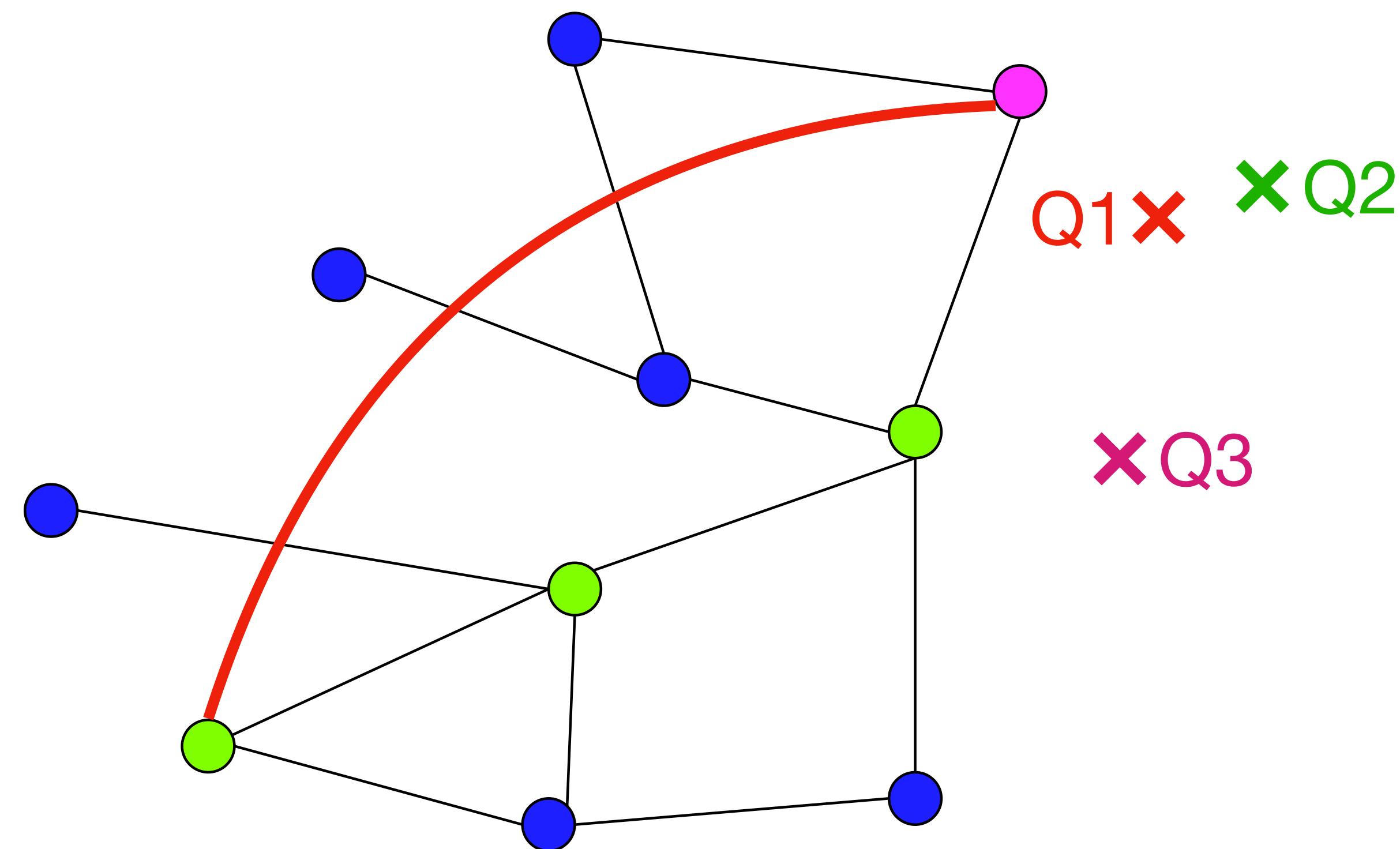
# Generic graph search (baseline)

✓ implemented



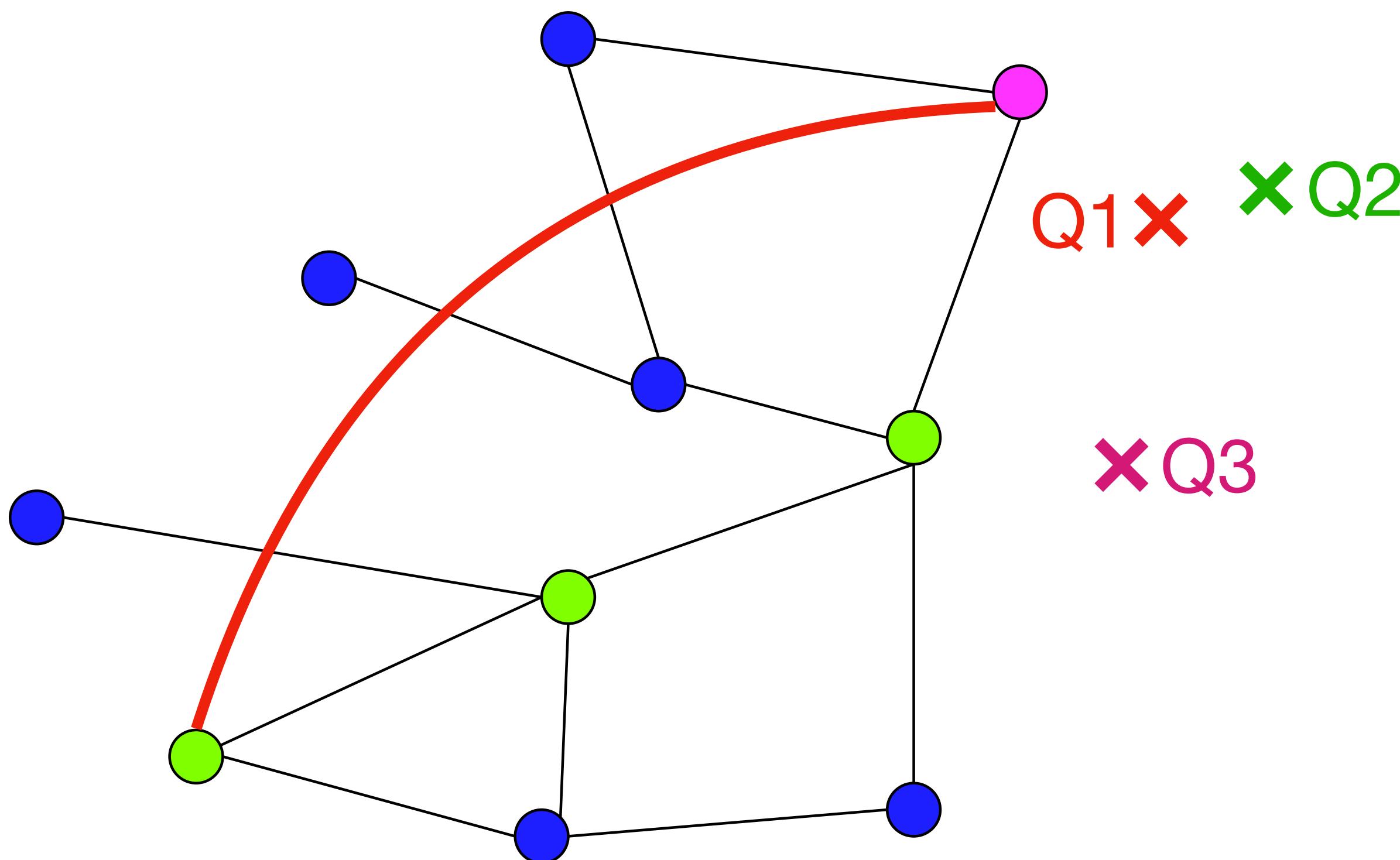
# Catapult management

■ Implemented with FIFO only



# LSH for entry-point selection

✓ implemented



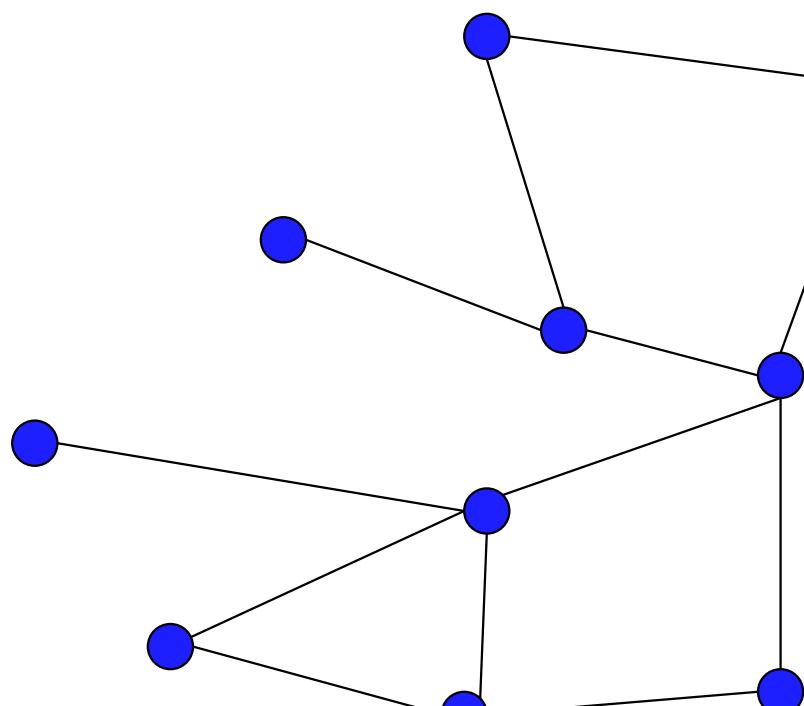
Hash bits	Starting nodes
11001	1, 4, 5
10010	9, 11, 18
...	...

New entries  
are random

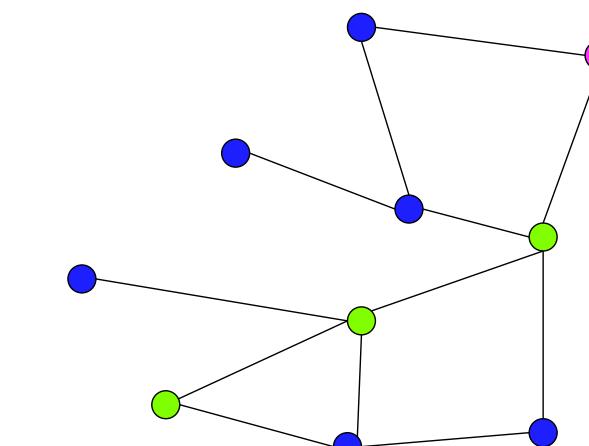
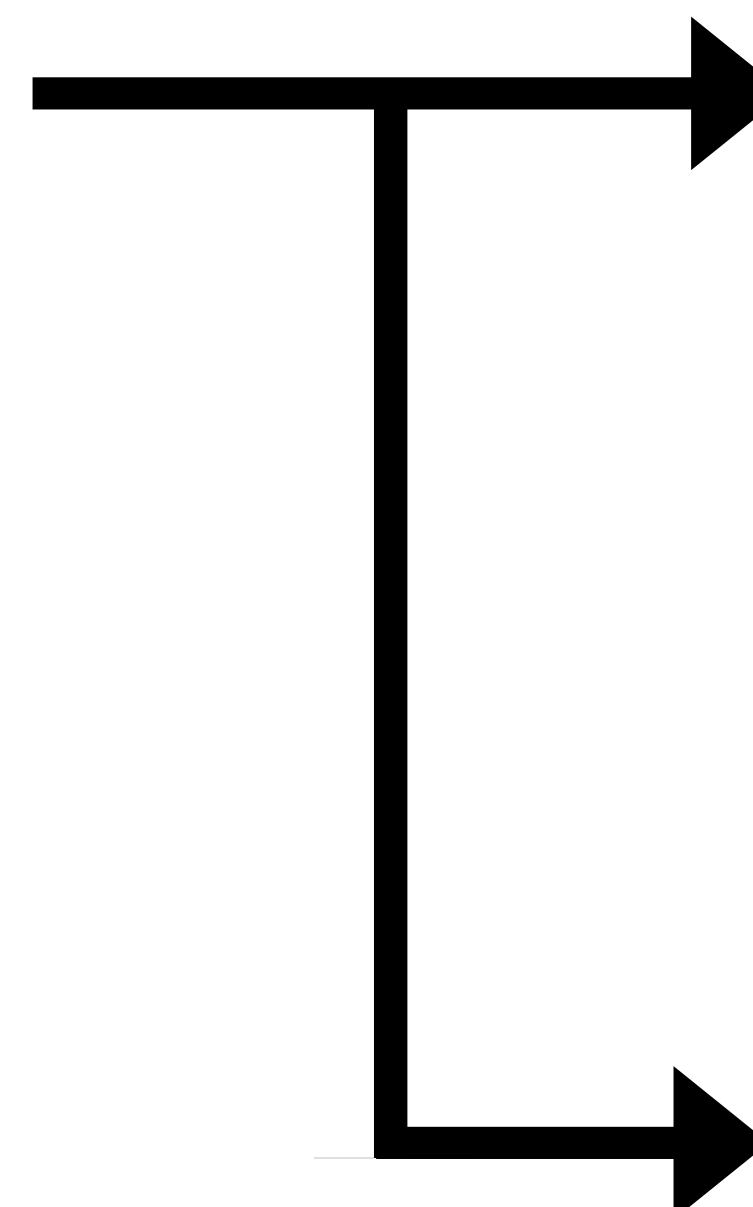
# What now?

# Hijacking the initial graph building

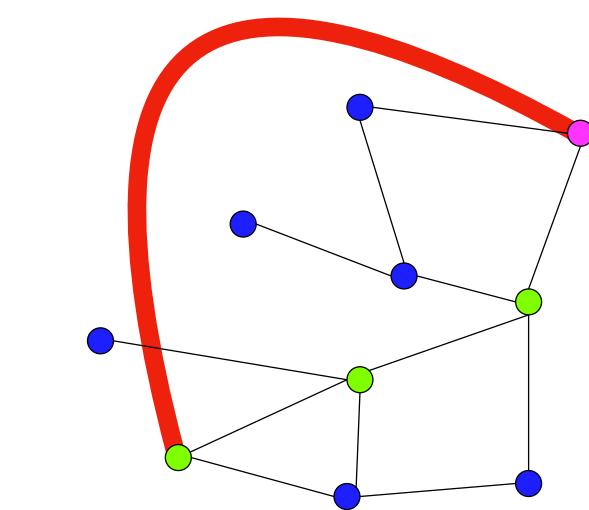
WIP



Vamana builds  
a graph



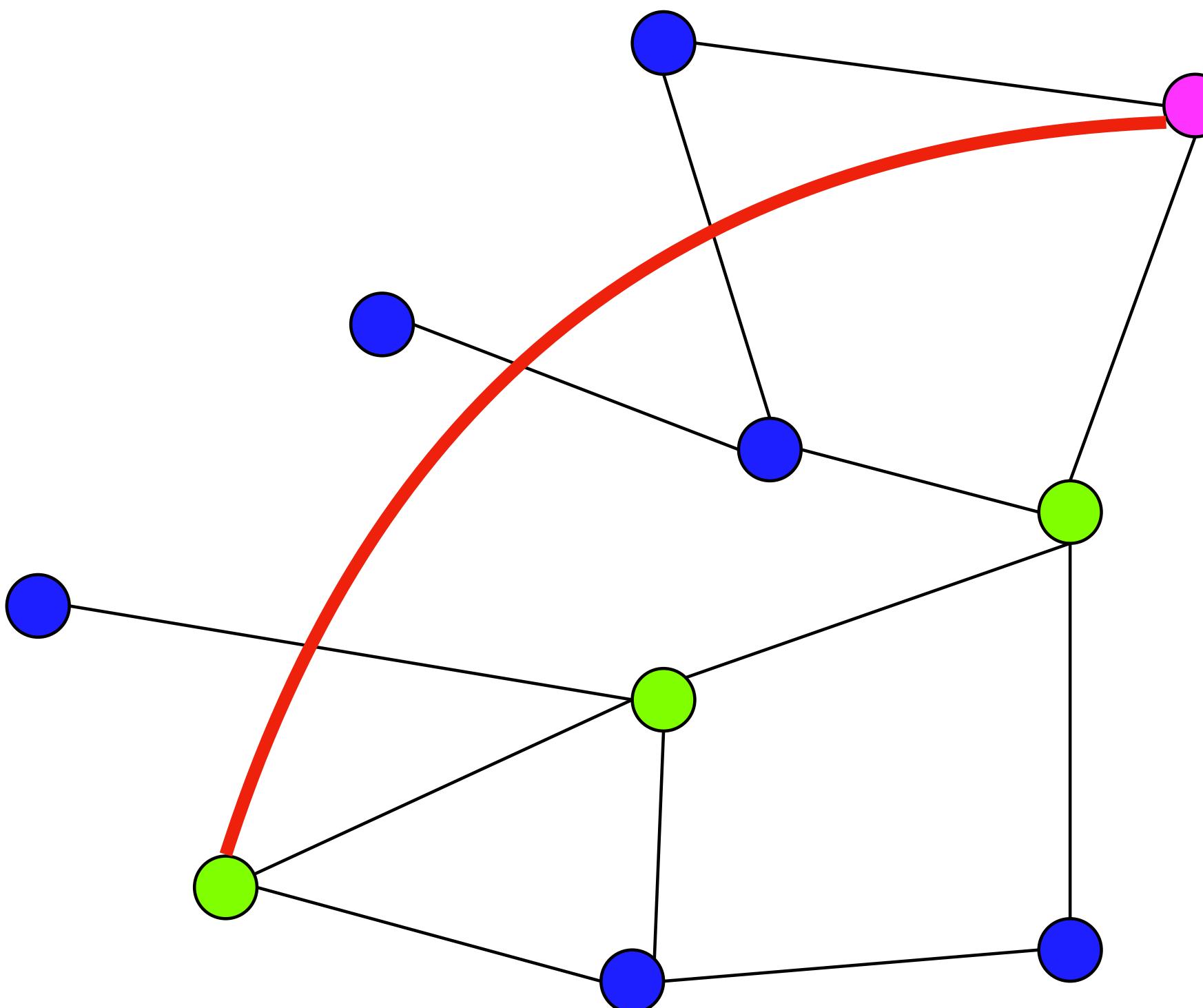
DiskANN  
exploits it



Catapult  
exploits it

# Multithreading

✗ not implemented



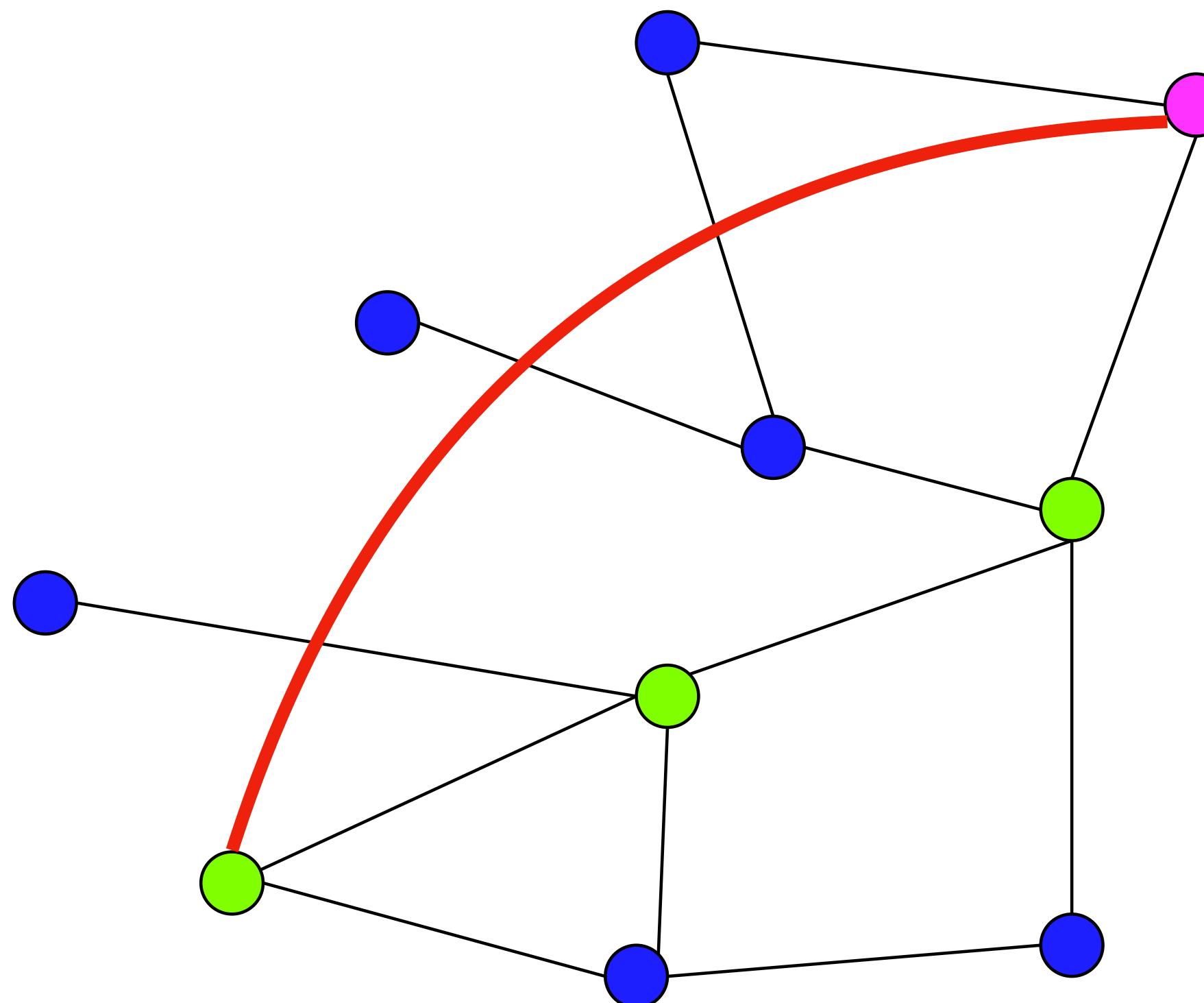
```
3  pub struct Node {  
4      pub neighbors: NeighborSet,  
5      pub catapults: NeighborSet,  
6      pub payload: Box<[f32]>,  
7  }
```

```
1  pub type NeighborSet = Vec<u32>;
```

-> 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, 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