

On Exploiting Logical Dependencies for Minimizing Additive Cost Metrics in Resource-Limited Crowdsensing

*Shaohan Hu, Shen Li, Shuochao Yao, Lu Su,
Ramesh Govindan, Reginald Hobbs, Tarek Abdelzaher*





**Shelter has
vacancy**

**Comm.
center up**

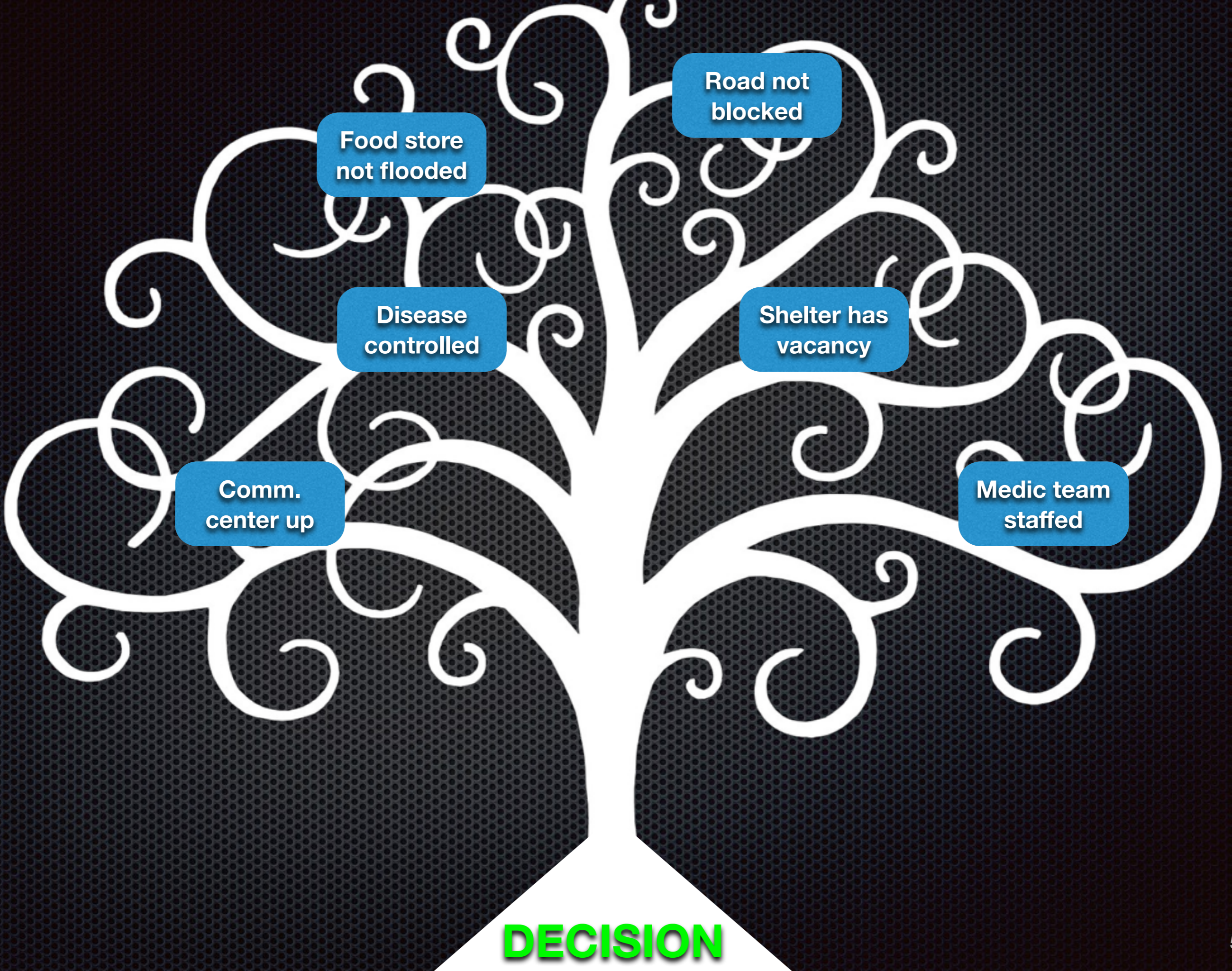
**Road not
blocked**

DECISION

**Disease
controlled**

**Medic team
staffed**

**Food store
not flooded**



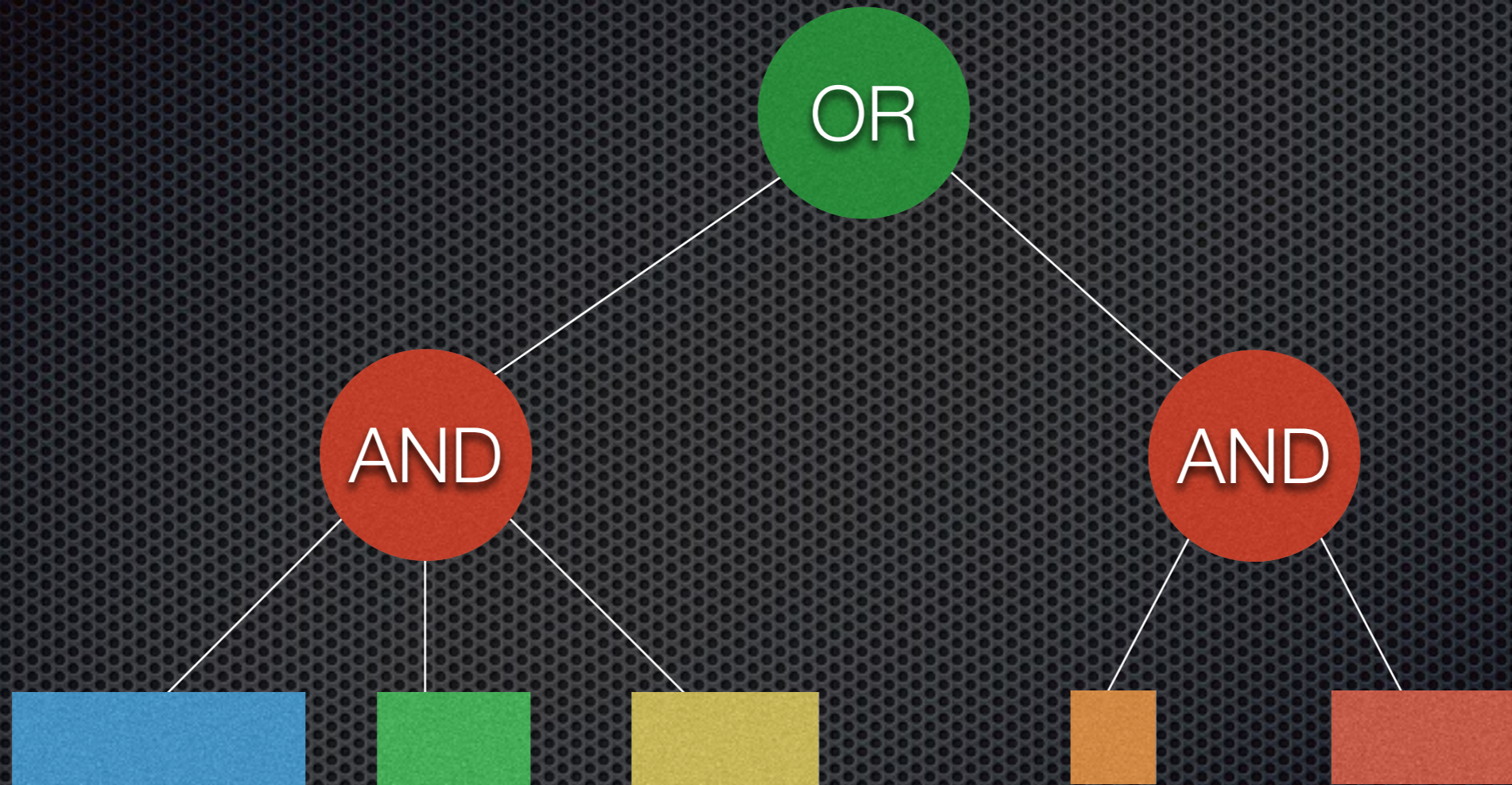
Decision Maker





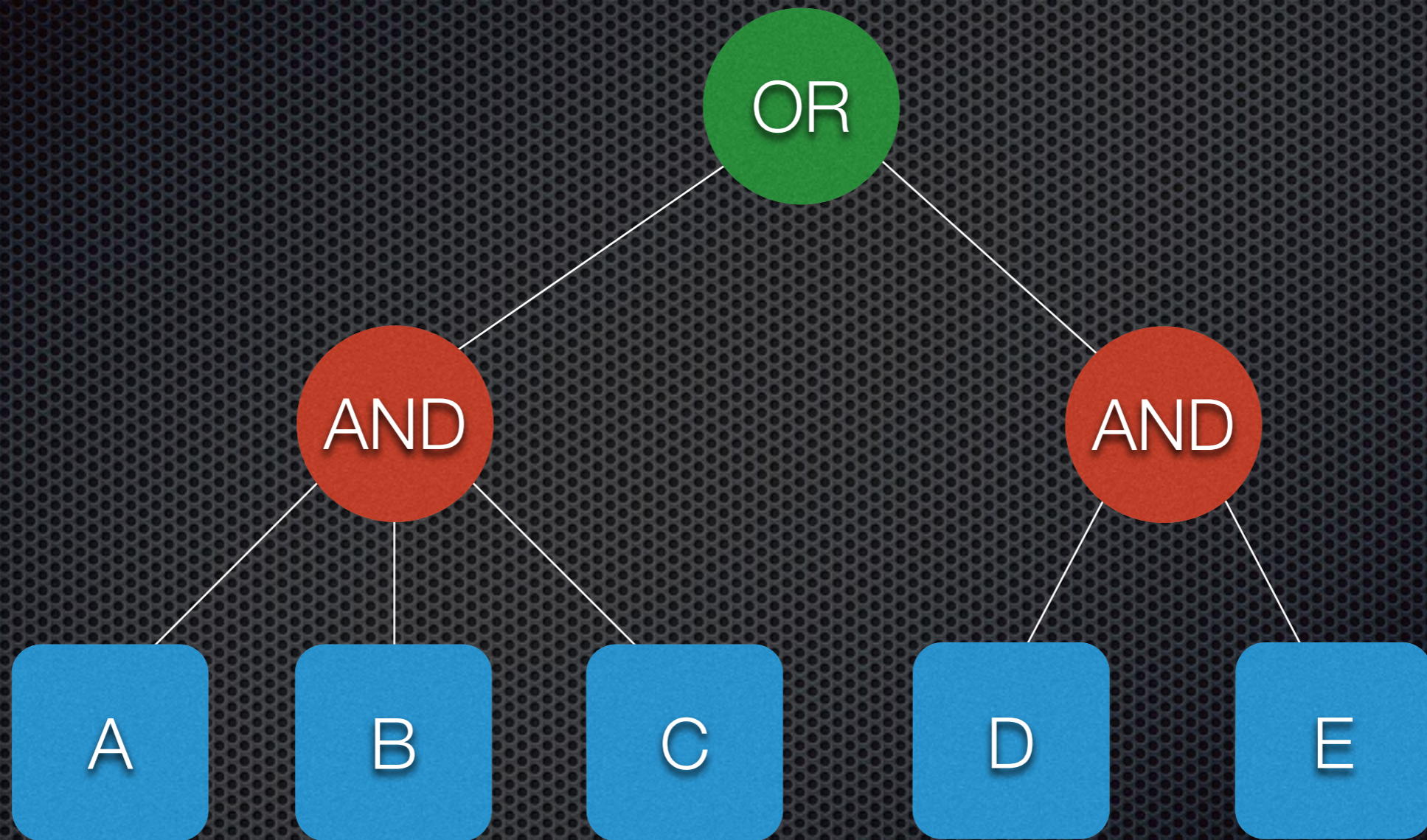
1

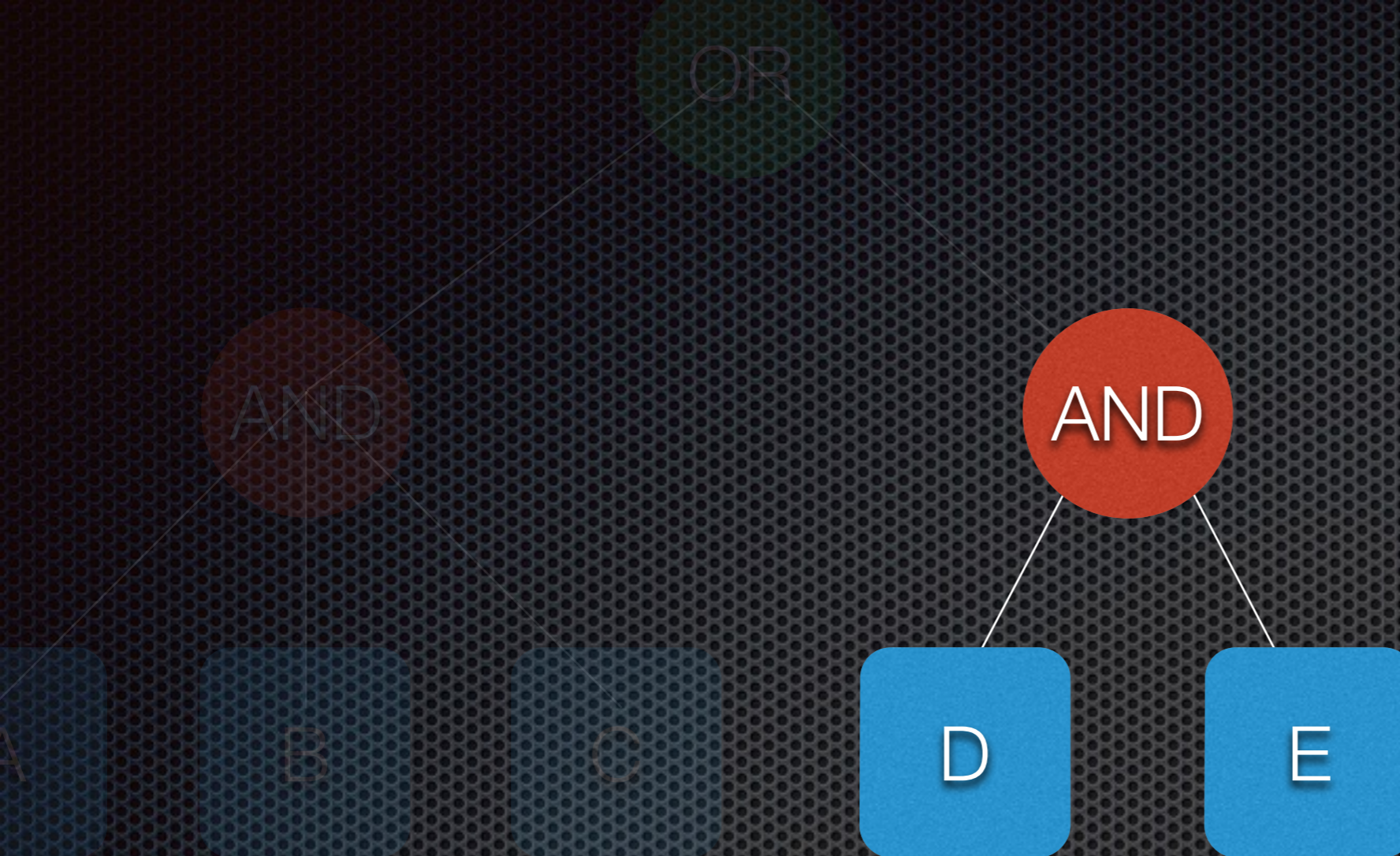






Order?





C

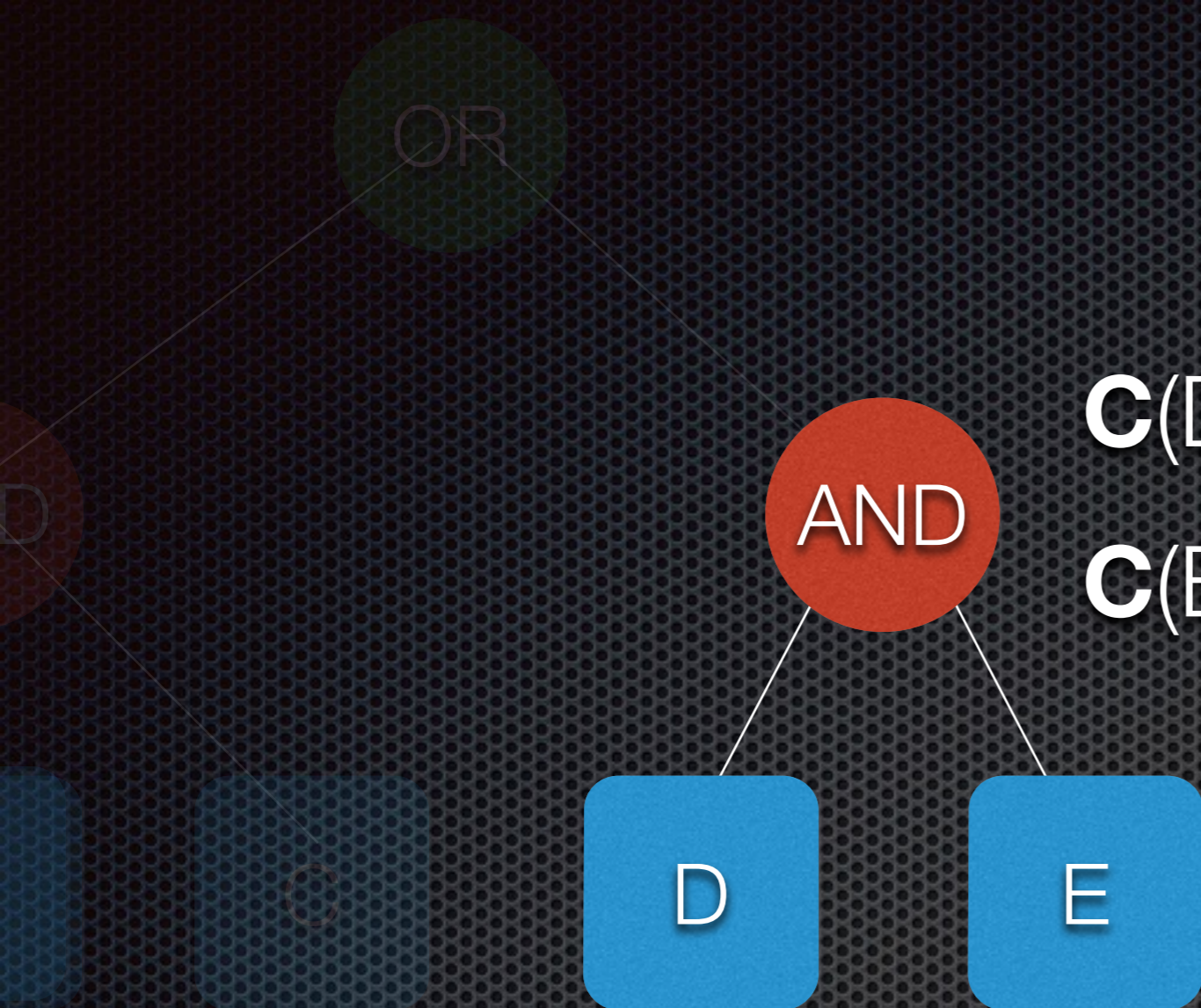
4

3

P(True)

0.3

0.6



$$C(D \rightarrow E) = 4 + 0.3(3) = 4.9$$

$$C(E \rightarrow D) = 3 + 0.6(4) = 5.4$$

C

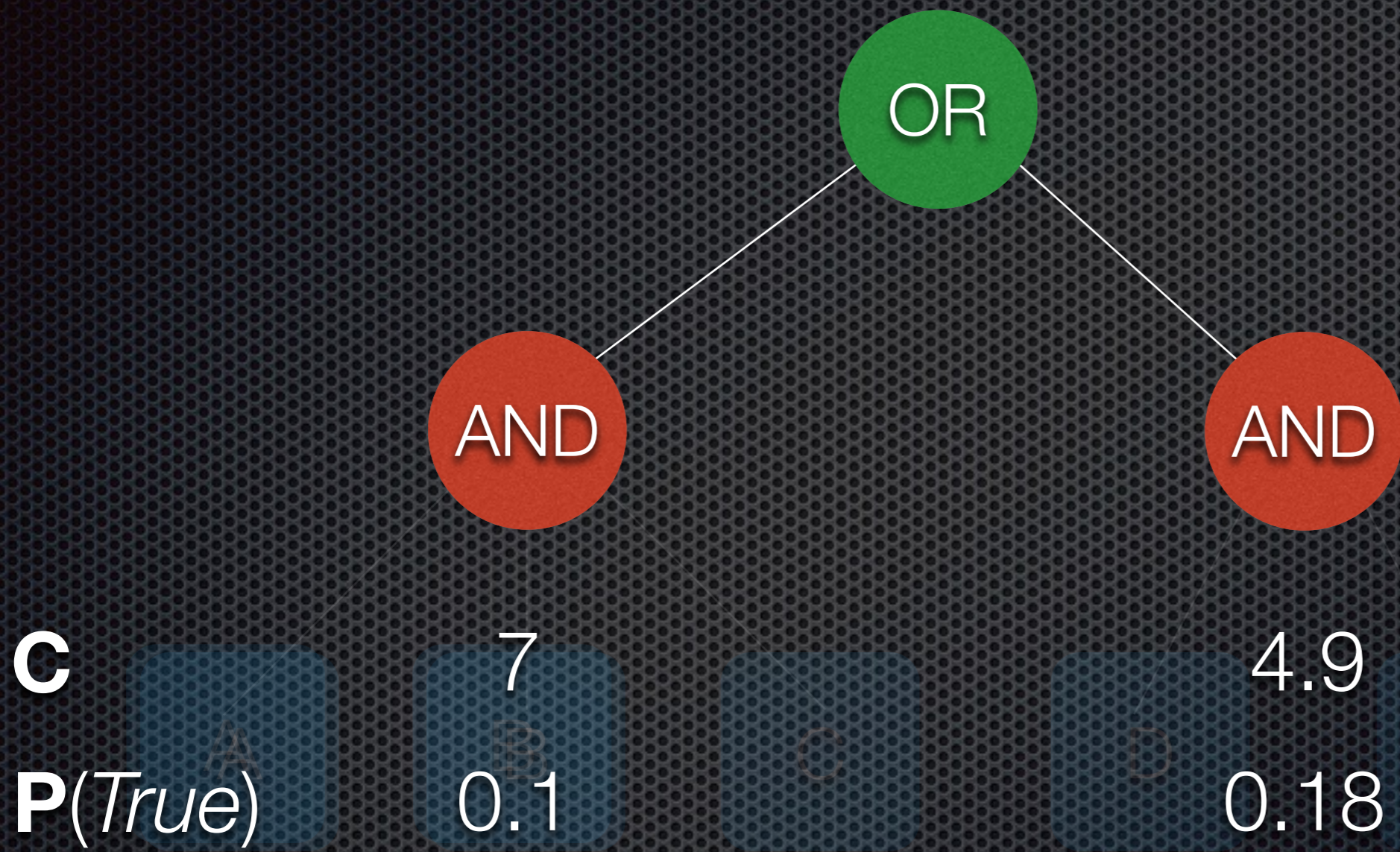
4

3

P(True)

0.3

0.6



$$\mathbf{C(L \rightarrow R)} = 7 + (1 - 0.1) \times 4.9 = \mathbf{11.41}$$

$$\mathbf{C(R \rightarrow L)} = 4.9 + (1 - 0.18) \times 7 = \mathbf{10.64}$$

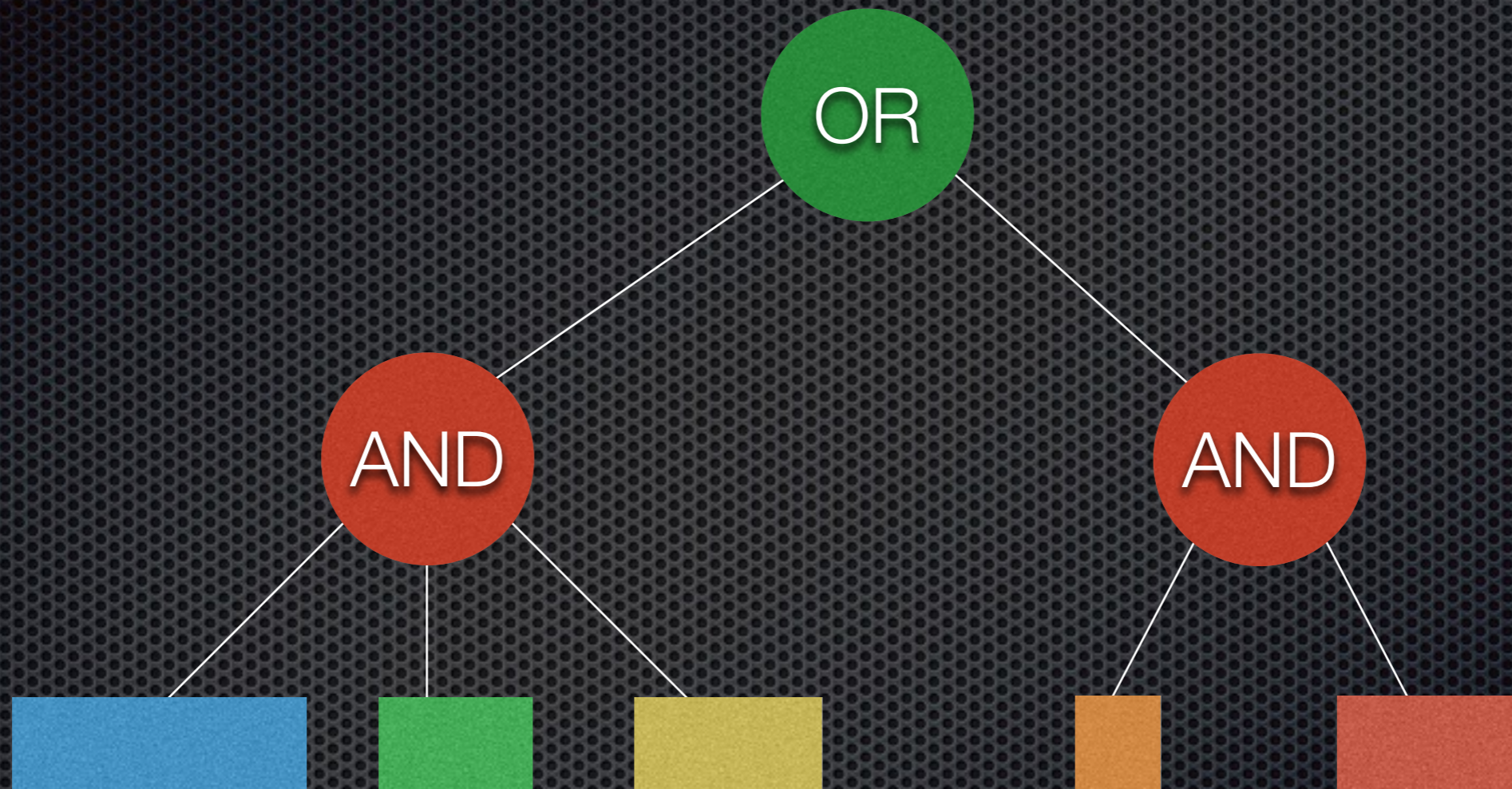


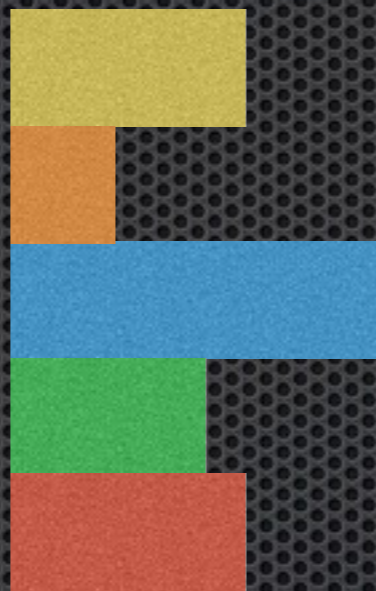
key: $\frac{\text{Shortcut probability}}{\text{cost}}$

2

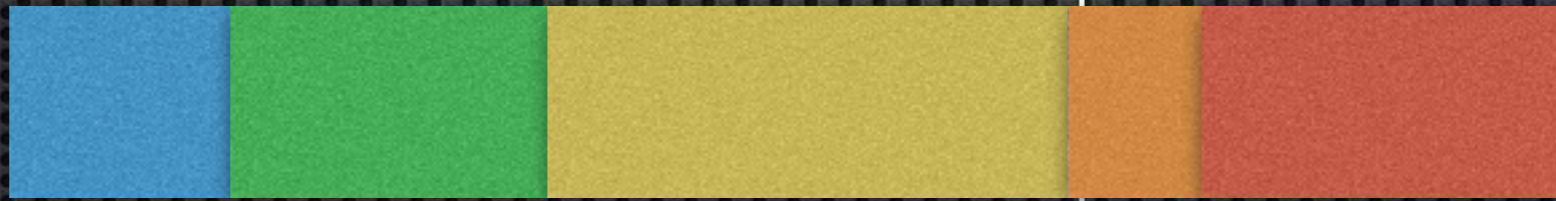


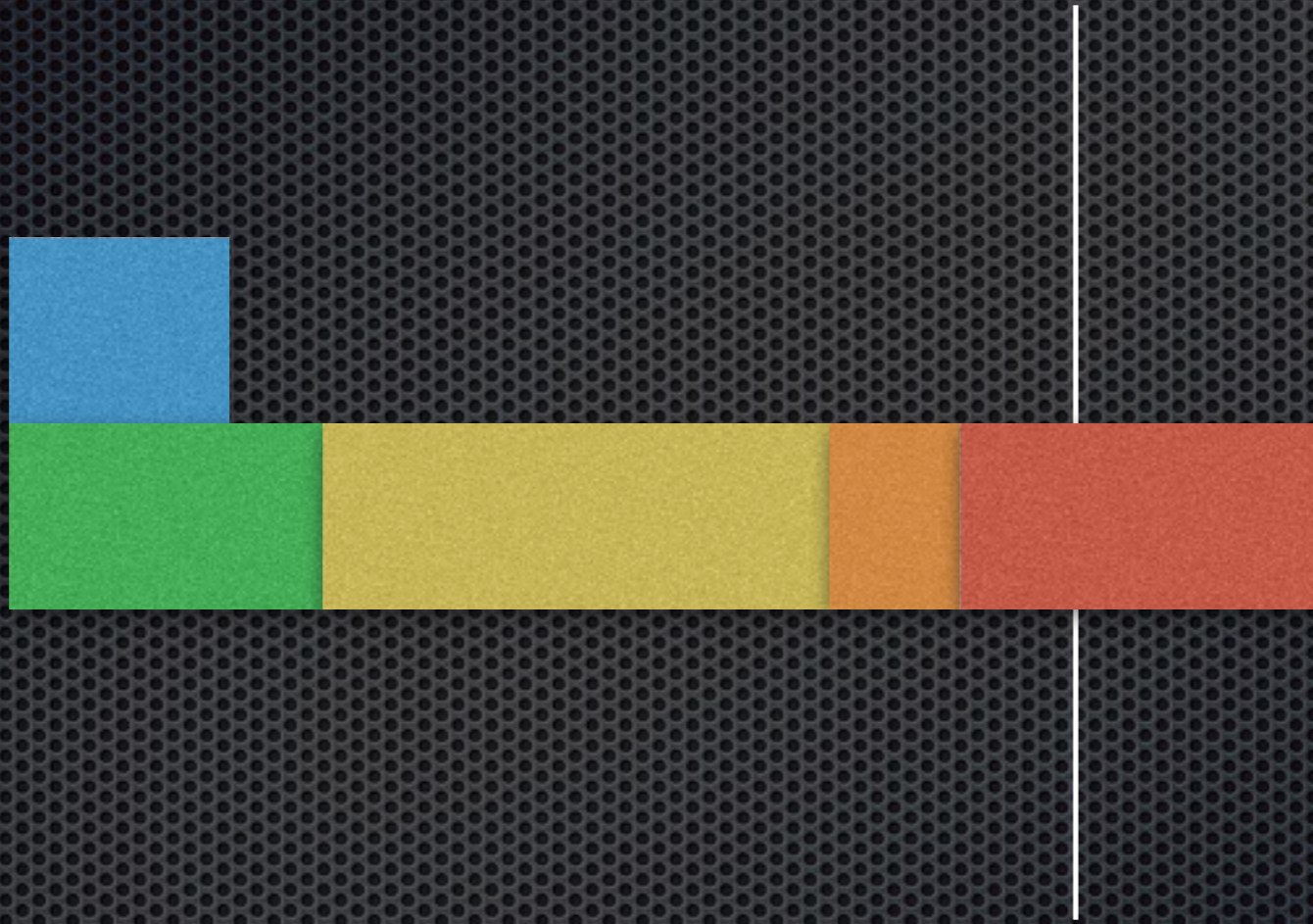
DEADLINE

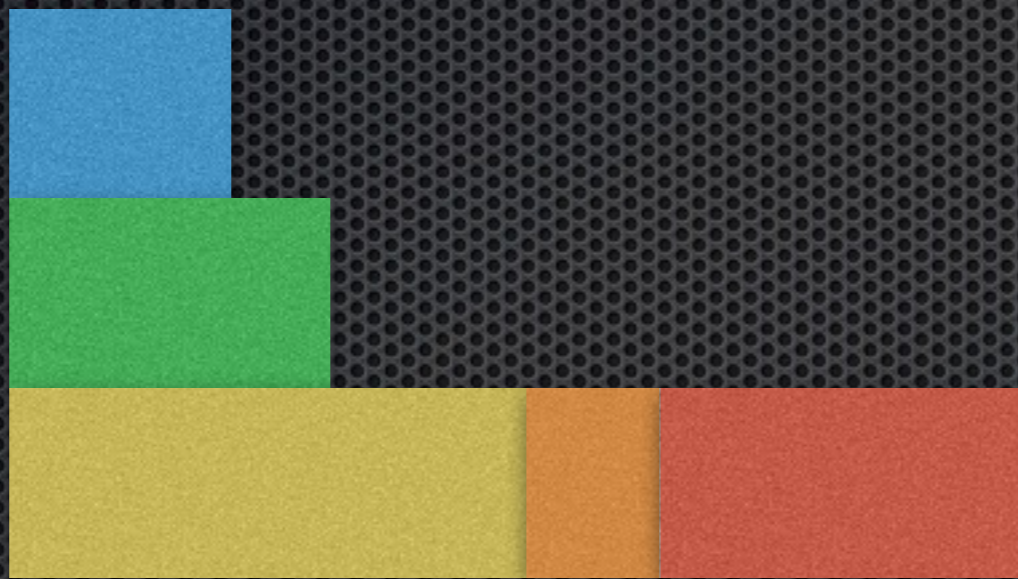




How much?







Complete Algorithm Sketch

- ✦ Concurrent queries with deadlines
- ✦ Greedy approach:
 - ✦ Compute most cost-effective object to retrieve next
 - ✦ Check expected latencies against deadlines
 - ✦ Increase parallel retrieval level if misses expected

Evaluation

- Simulation experiments
- An application scenario

Simulation Experiments

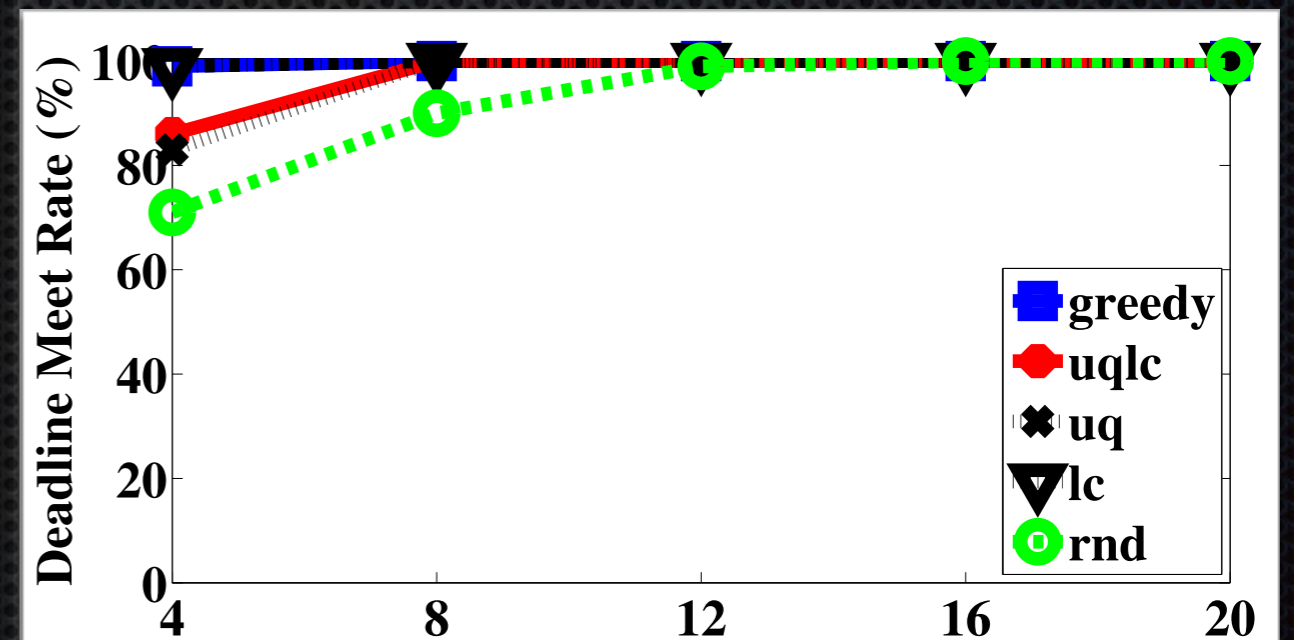
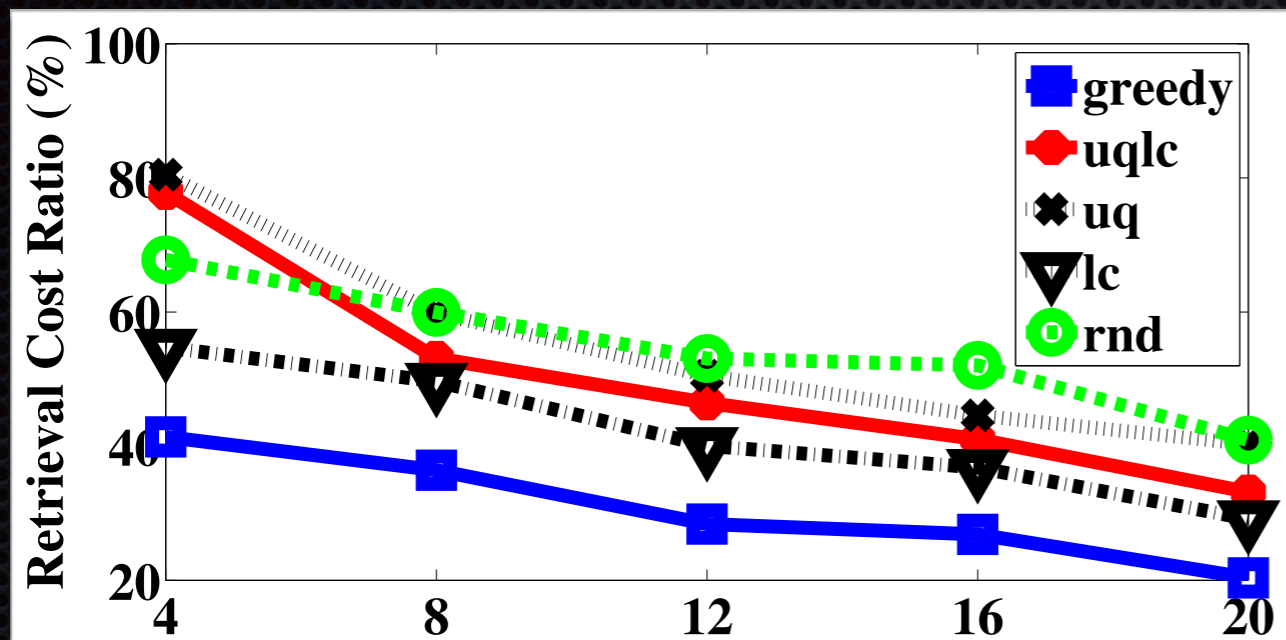
✦ Baselines

- ✦ Random order
- ✦ Lowest cost object first
- ✦ Most urgent query first
- ✦ Most urgent query's lowest cost object first

✦ Settings

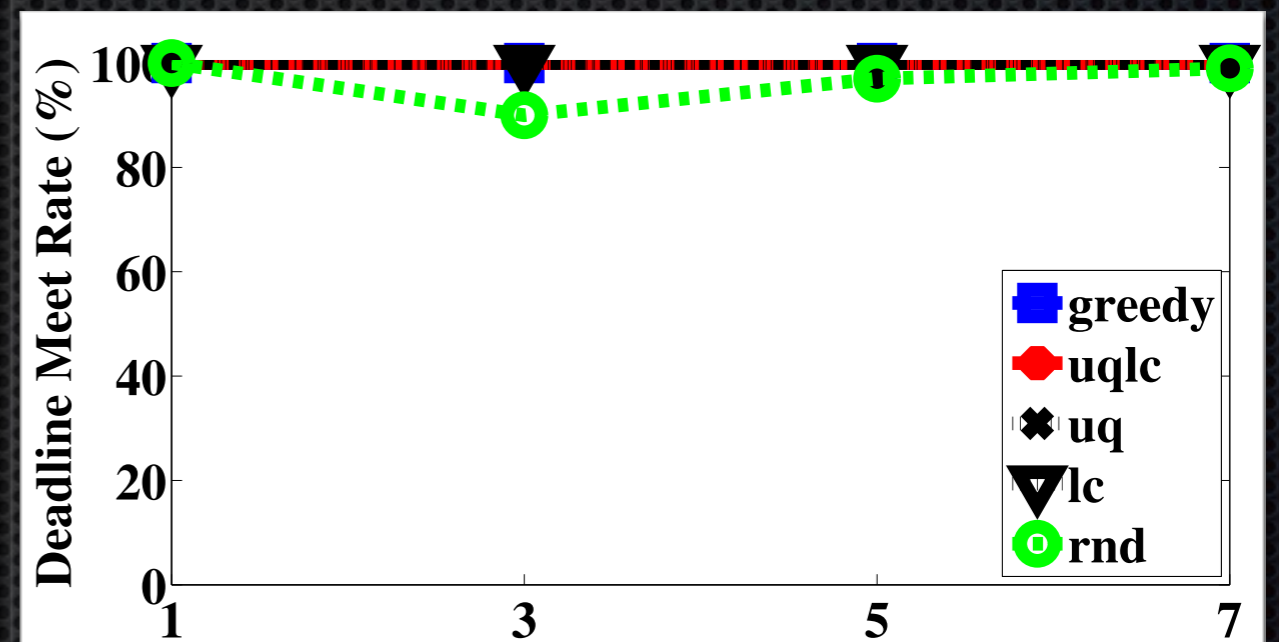
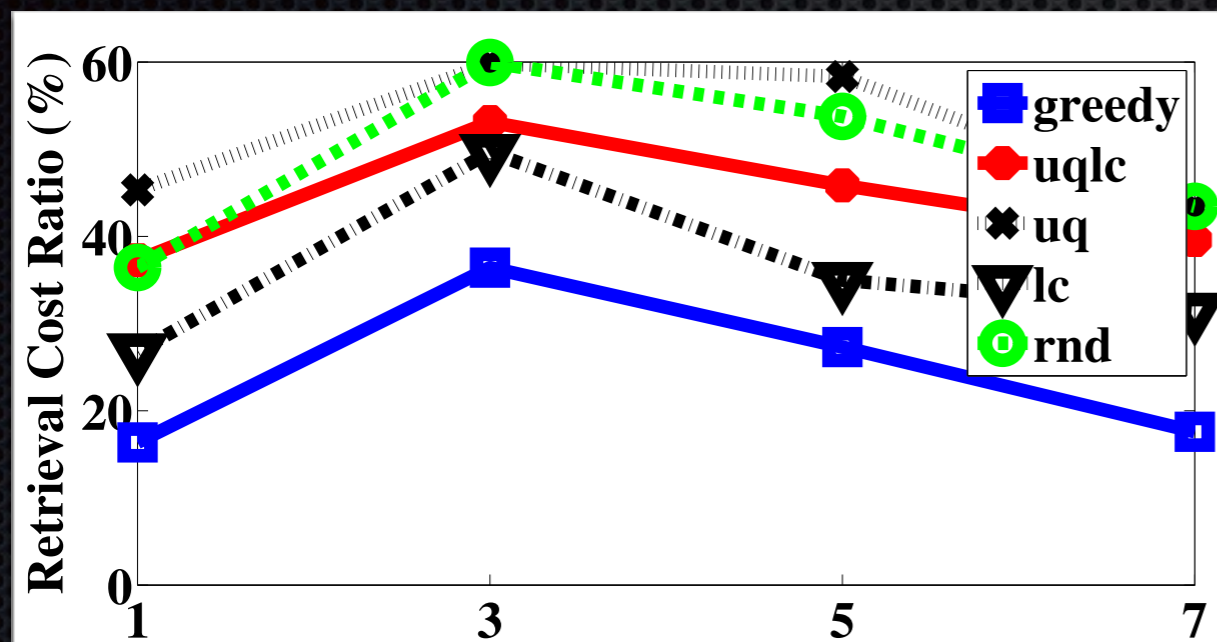
- ✦ # Objects per query: 4, **8**, 12, 16, 20
- ✦ # ANDs per query: 1, **3**, 5, 7
- ✦ # concurrent queries: 5, 10, 15, **20**, 25
- ✦ Deadline levels ($\frac{\text{deadline}}{\sum \text{latencies}}$): 0.5, 1, **2**, 4, 8

Simulation Results



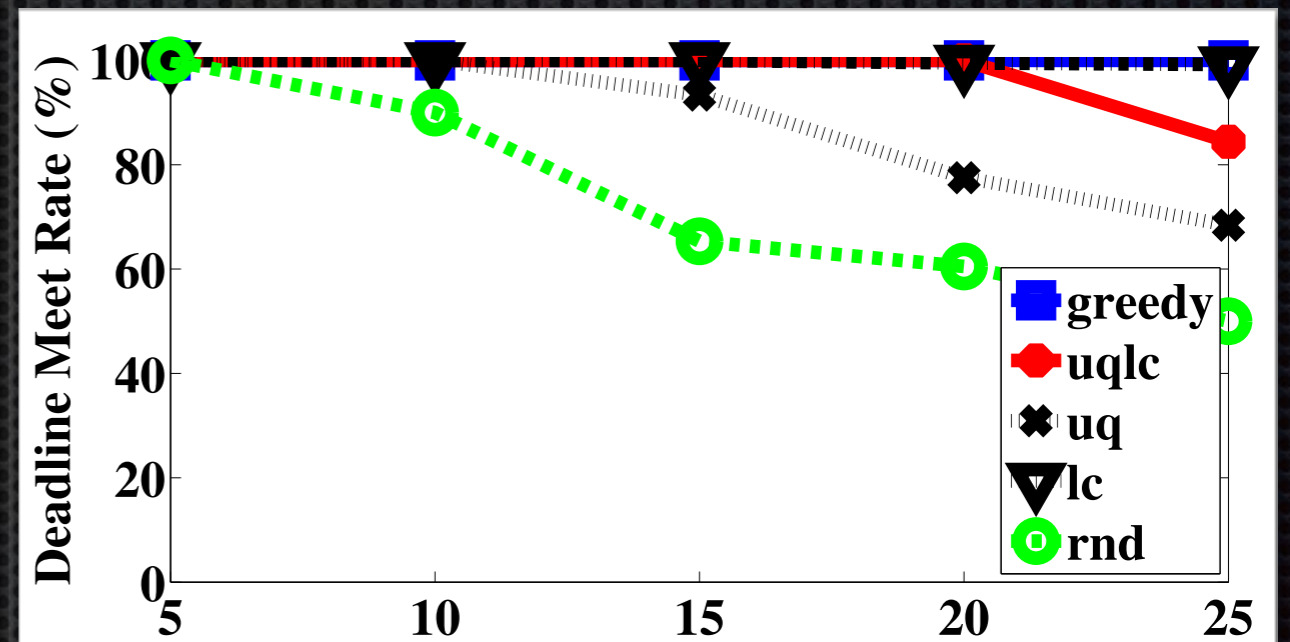
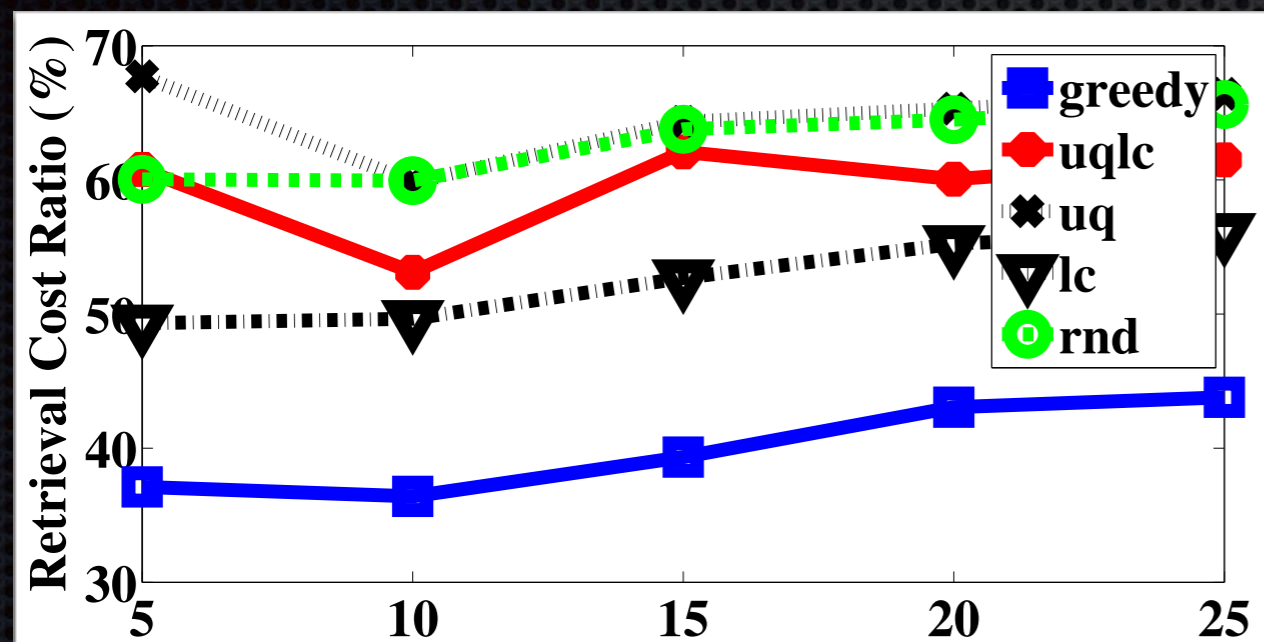
Varying # objects per query

Simulation Results



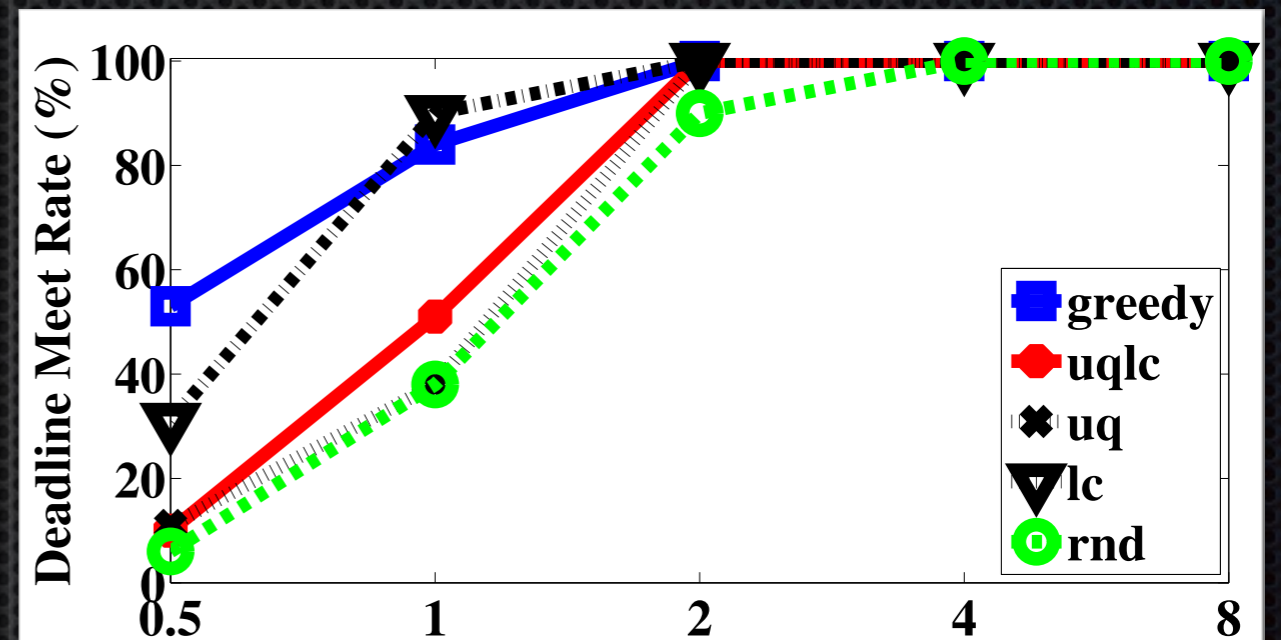
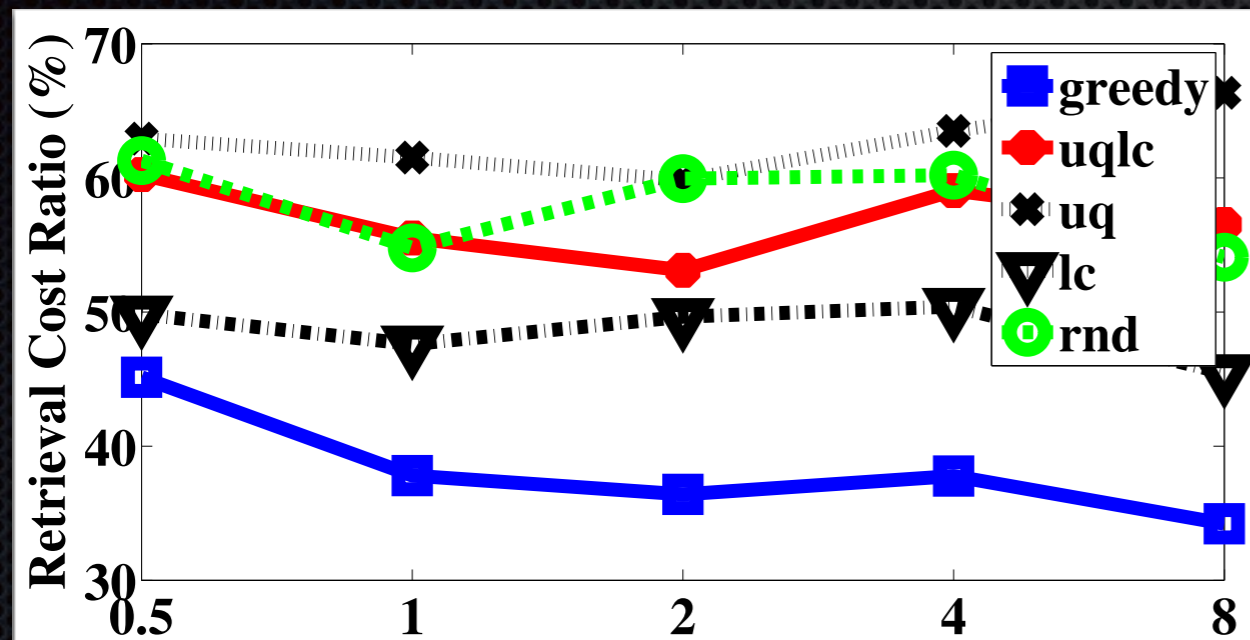
Varying # ANDs per query

Simulation Results



Varying # concurrent queries

Simulation Results



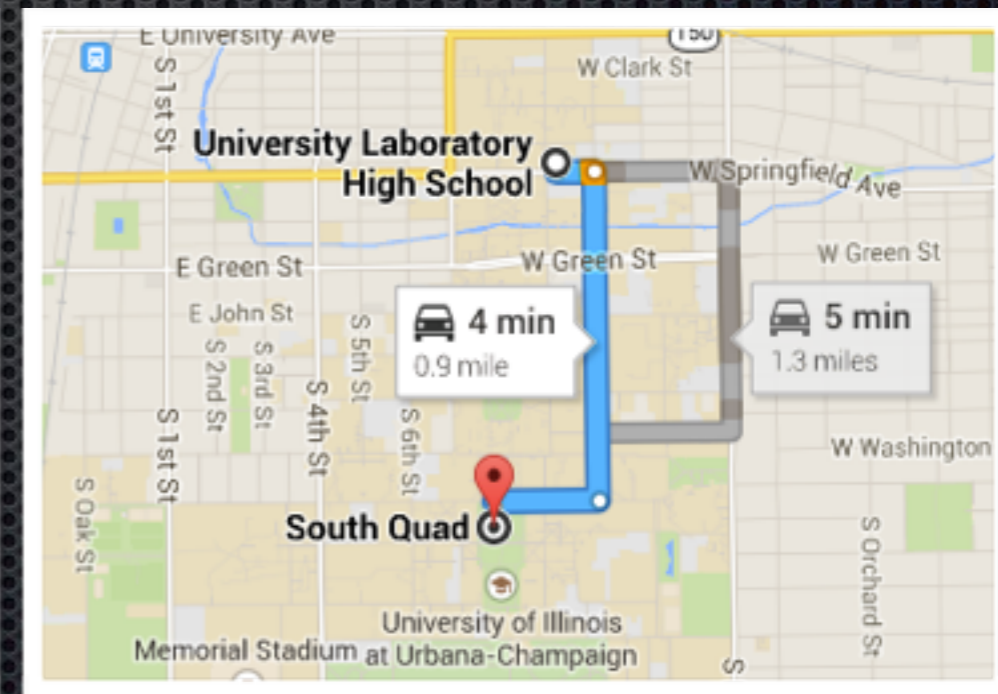
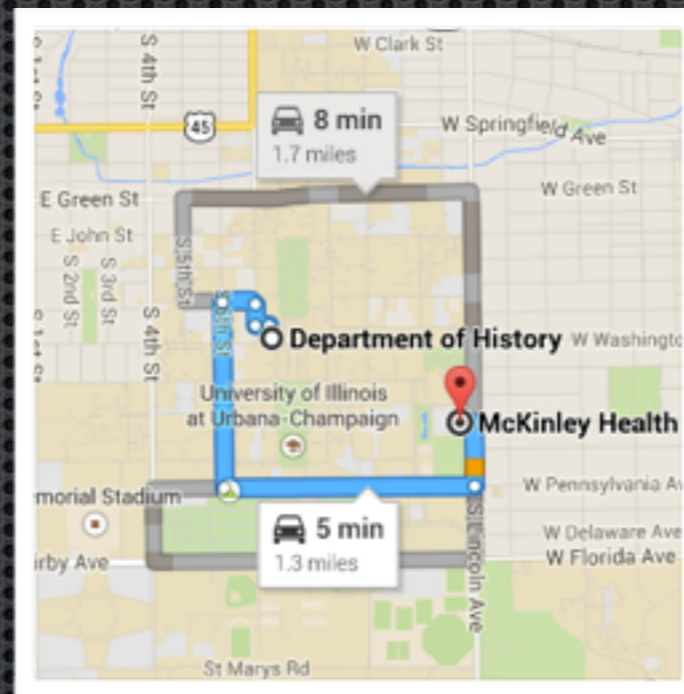
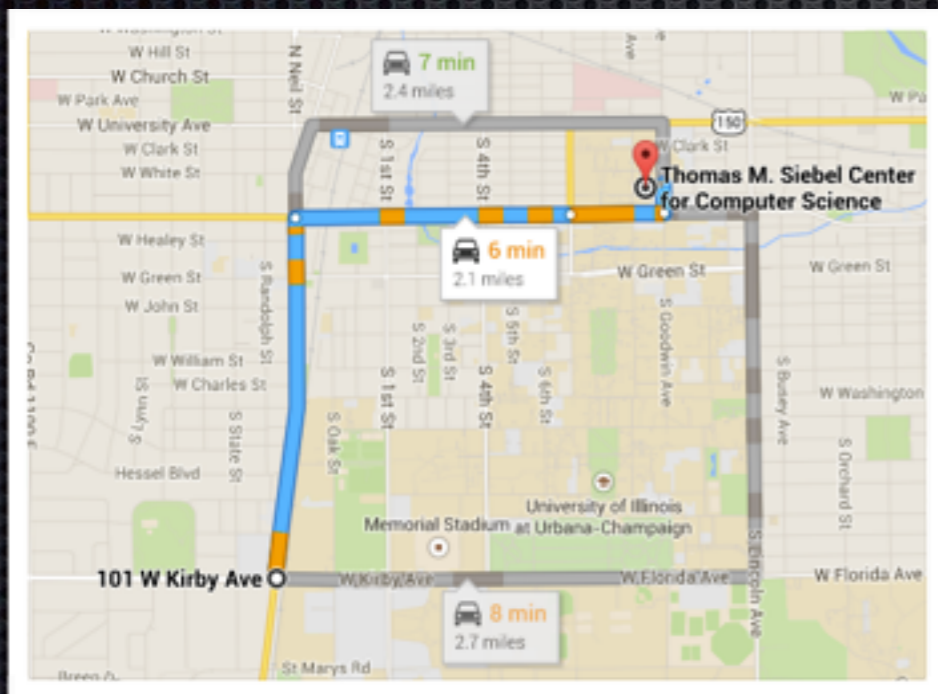
Varying queries' deadline levels

Application: Route Finding

- ✦ Find routes for $\langle \text{src}, \text{dst} \rangle$ pairs
 - ✦ Each candidate route: **AND** of its segments
 - ✦ Routing result: **OR** of all candidate routes
- ✦ Visual verification for route segment conditions

Route Finding Example Run

- 3 disaster response teams' computed candidate routes



	Greedy	Least-cost
# Road conditions retrieved	4	9
Retrieval cost ratio (%)	28.95	58.56
Deadline meet rate (%)	100	100

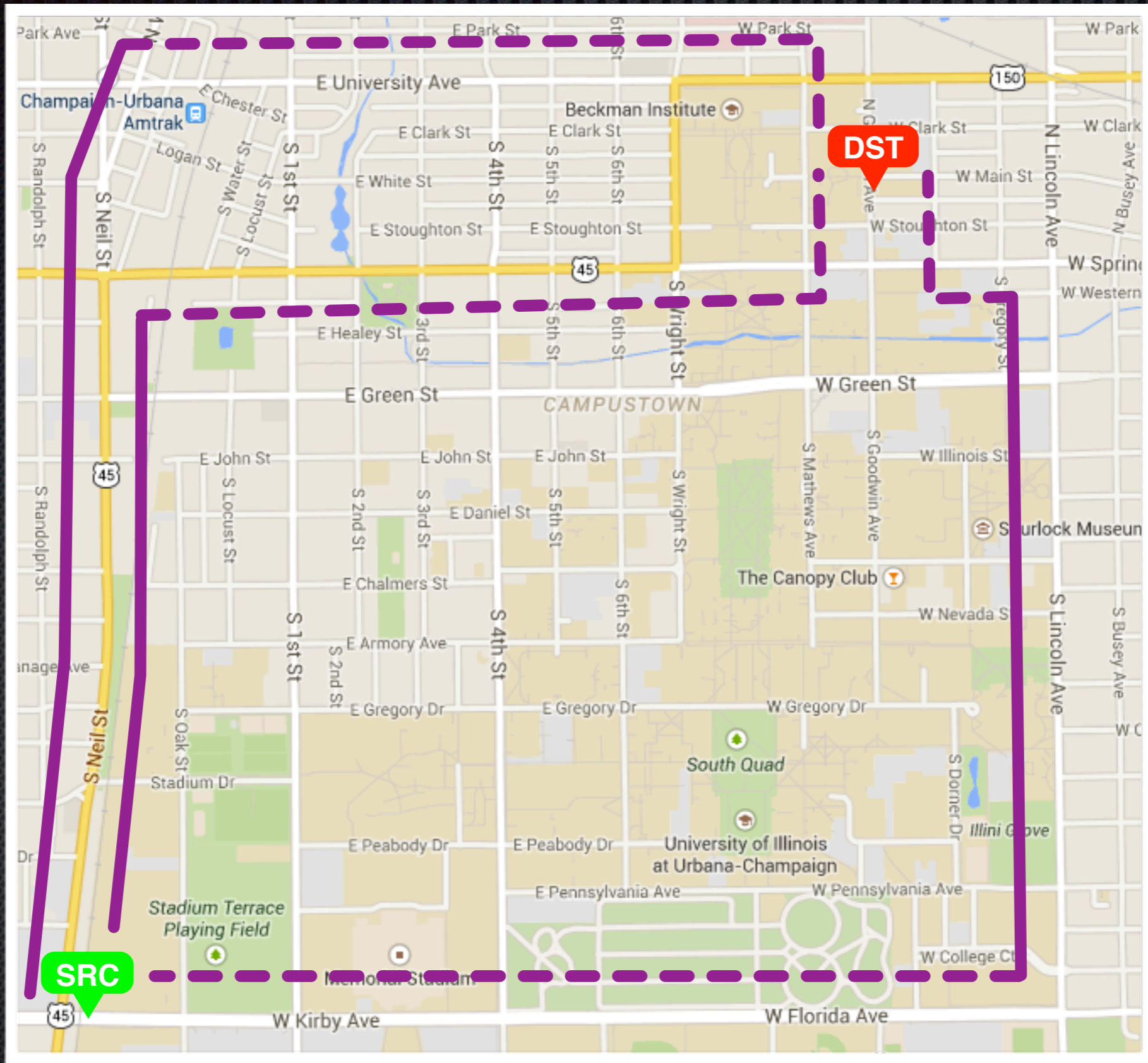
Conclusion

- ✦ Data retrieval algorithms for crowdsensing under resource constraints
- ✦ Minimizing cost under timeliness requirements
- ✦ Promising results through simulations and concrete route finding application scenario

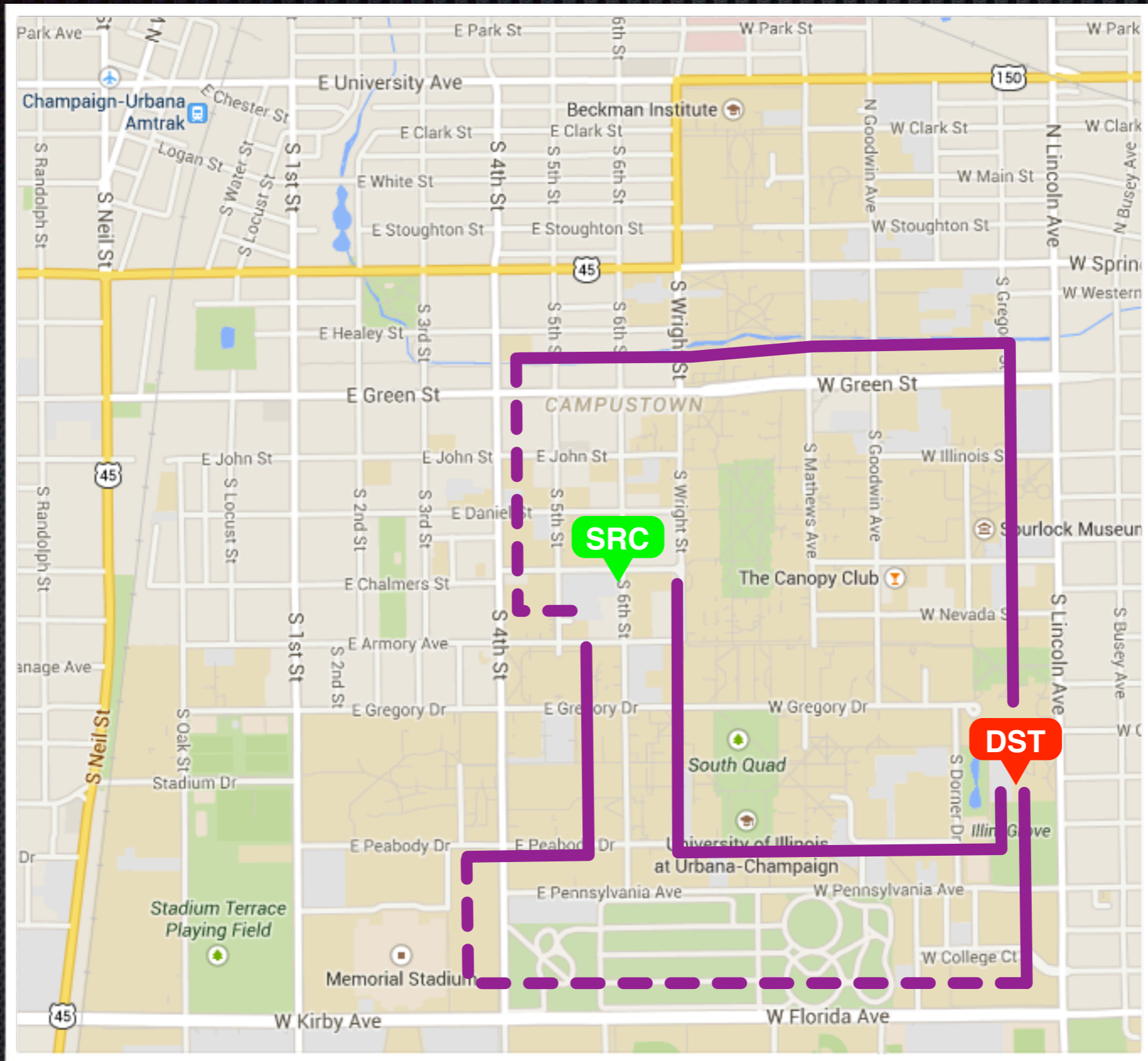
Thanks

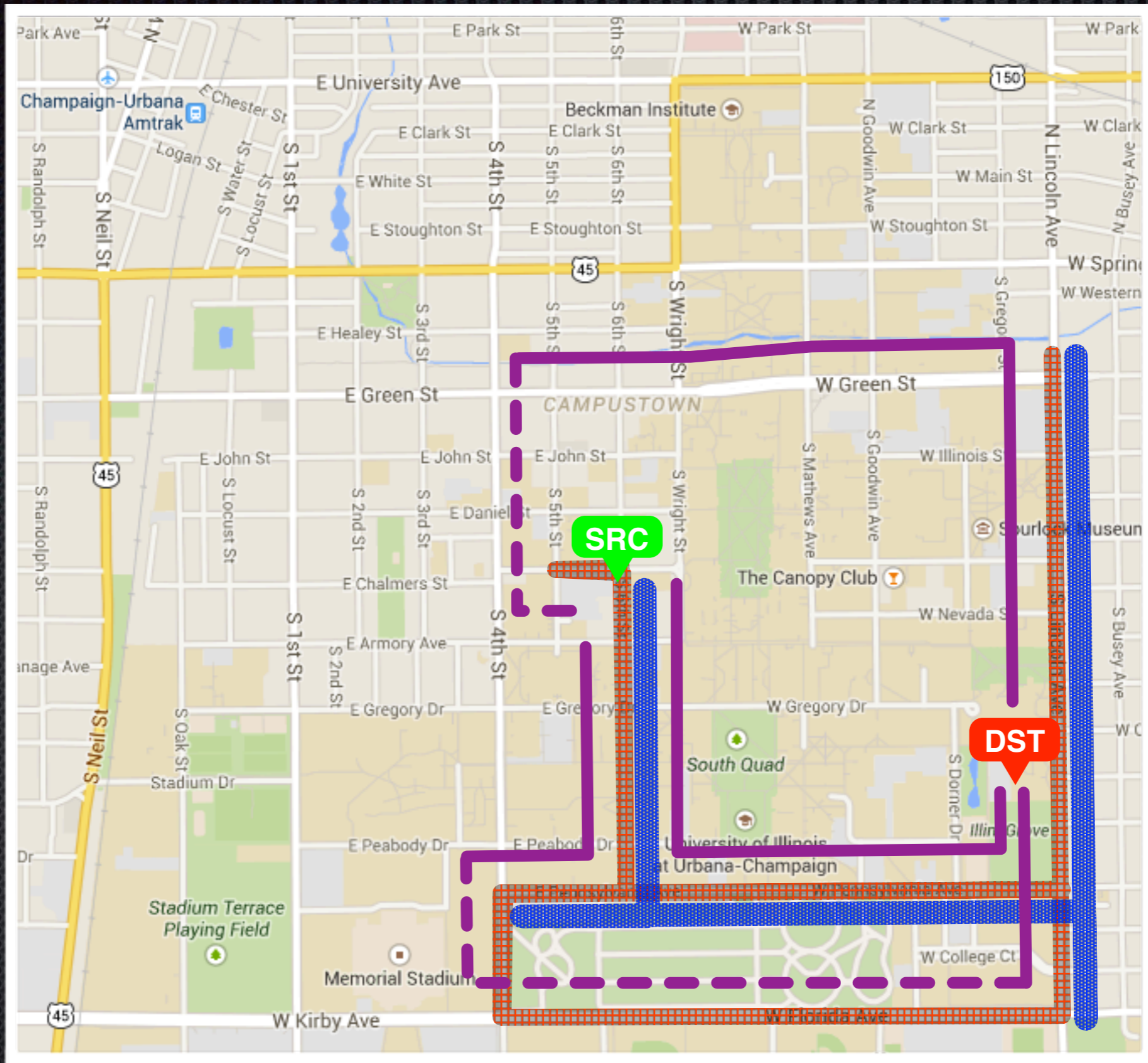
backup

Team 1



Team 2





Team 3

