COMP20007 Design of Algorithms: Week 5

Goals of the lectures

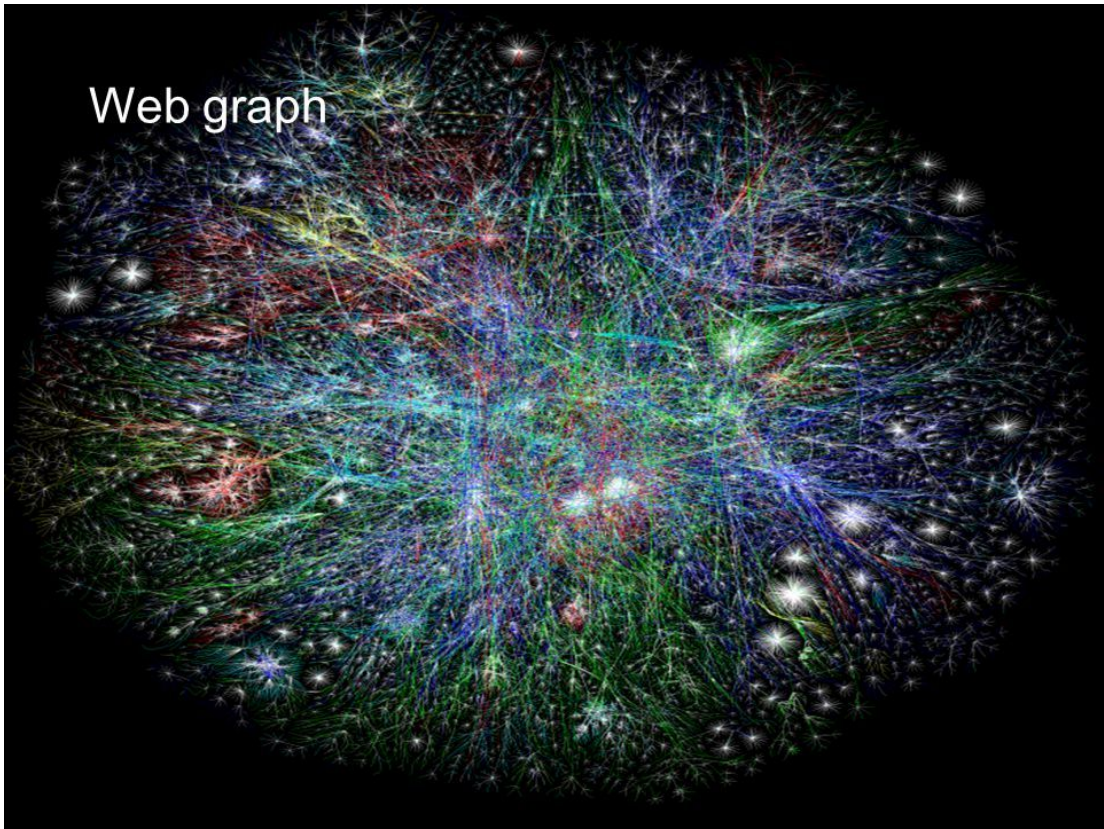
Student reps

MST next Friday

Discussion forum

Labs for help with project work

Commenting



What's covered in the mid-semester test?

Things you already knew:

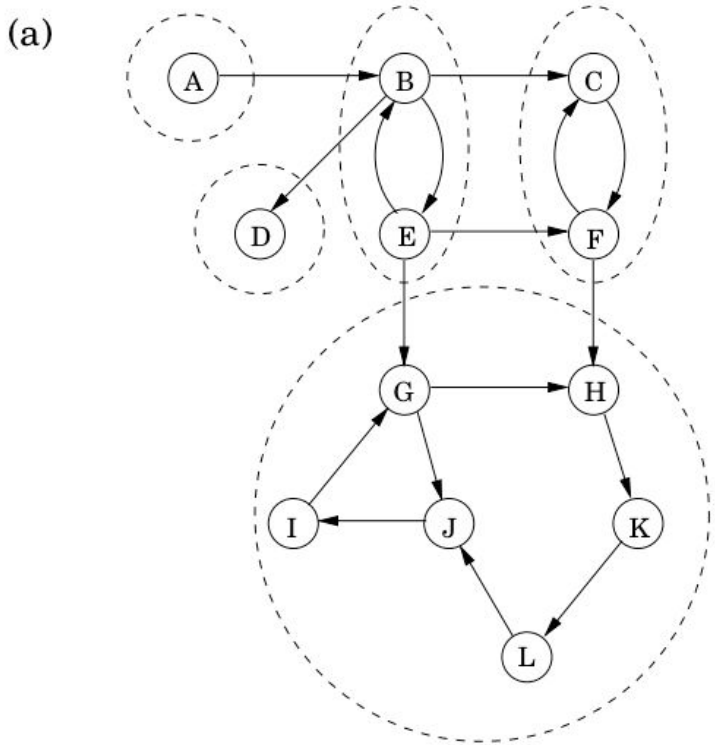
- Array, Linked List, Binary Search Tree, Hash Table
- Stack, Queue, Priority Queue
- Sorting algorithms

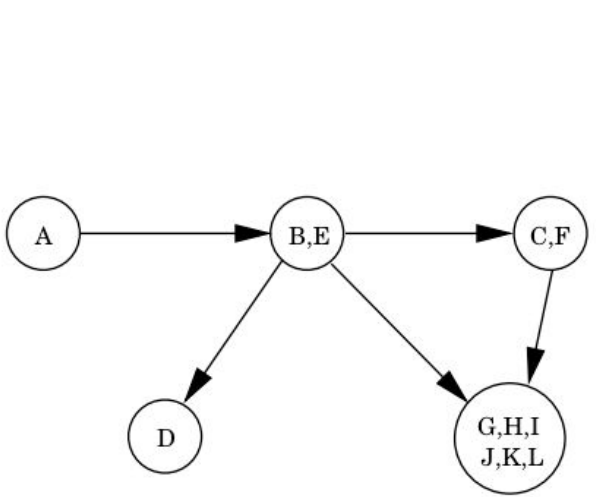
The first five weeks of semester:

- Divide and conquer
- Formal analysis: recurrence relations, Master Theorem, big O
- Graphs: definition, properties, representations, topological sort
- Traversal: DFS, BFS, pre and post numbering, strongly-connected components
- Dijkstra's algorithm, Bellman-Ford

Reading: DPV 0.3, 2.2–2.5, 3, 4

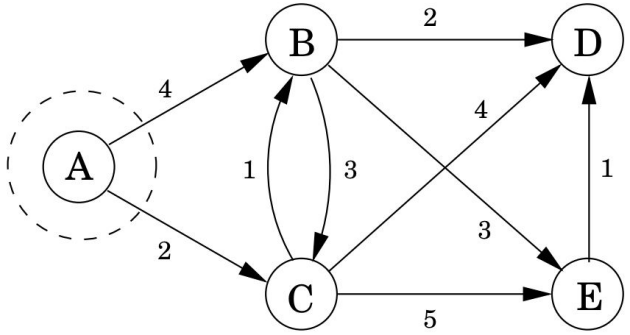
Strongly Connected Components (SCCs) Revisited





- motivation
- pre and post numbers
- source and sink SCCs
- Kosaraju's Algorithm

Dijkstra's Algorithm Revisited




```
initialize Q
foreach node v:
    add v to Q with priority ∞
update priority of source node s in Q to 0
while Q not empty:
     v = element of Q with lowest priority
     remove v from Q
     foreach edge (v, u):
          dist(u) = min(dist(v) + w_vu, dist(v))
          update priority of u in Q
```

What priority queue operations are performed and how often? n=#nodes m=#edges

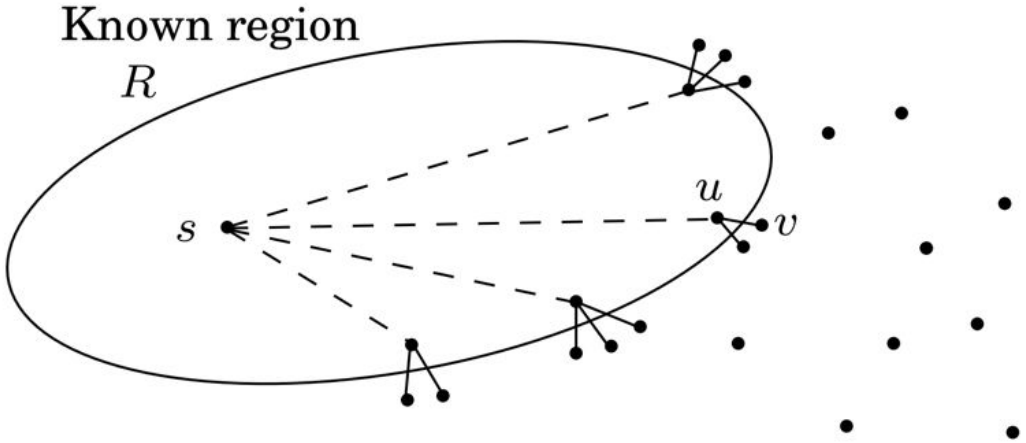
Cost of different priority queue implementations

What are the implementation possibilities?

What are the operations

Tabulate

Dijkstra's Algorithm: Correctness



v is the node outside *R* for which the smallest value of distance(*s*, *u*) + l(*u*, *v*)

At the end of each iteration of the while loop, the following conditions hold: (1) there is a value d such that all nodes in R are at distance $\leq d$ from s and all nodes outside R are at distance $\geq d$ from s, and (2) for every node u, the value $\texttt{dist}(u)$ is the length of the shortest path from s to u whose intermediate nodes are constrained to be in R (if no such path exists, the value is ∞).