Dec. 4 – Prim’s Algorithm
Wade Fagen-Ulmschneider
Kruskal’s Algorithm

<table>
<thead>
<tr>
<th>Priority Queue:</th>
<th>Total Running Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Heap</td>
<td>$O(n + m) + O(m \lg(n))$</td>
</tr>
<tr>
<td>Sorted Array</td>
<td>$O(n + m \lg(n)) + O(m)$</td>
</tr>
</tbody>
</table>

KruskalMST(G):

1. DisjointSets forest
2. foreach (Vertex v : G):
   - forest.makeSet(v)
3. PriorityQueue Q // min edge weight
4. foreach (Edge e : G):
   - Q.insert(e)
5. Graph T = (V, {})
6. while |T.edges()| < n-1:
7.     Vertex (u, v) = Q.removeMin()
8.     if forest.find(u) != forest.find(v):
9.         T.addEdge(u, v)
10.        forest.union( forest.find(u), forest.find(v) )
11.    return T
Kruskal’s Algorithm

Which Priority Queue Implementation is better for running Kruskal’s Algorithm?

• Heap:

• Sorted Array:
Mattox Monday

Exam 12

• Programming Exam
Mattox Monday

• Exam 13
  • Second Chance!
  • You may pick one of the previous 12 exams to retake.
    • (Let the exam you choose be denoted as examN)
  • exam13score = max(avg(exam1..exam12), retake)
  • examNscore = max(examN, retake)

If you are happy with your exam average, you can stay home. 😊
CS Education Week: Hour of Code

This week is the 111th birthday of Grace Hopper, a pioneer of the field of Computer Science.

Tonight, CS@Illinois + Women in Computer Science is hosting an Hour of Code

**Volunteer:** Help others program their first line of code!

Tonight, Dec. 4, 6:00pm – 8:00pm
Basement of Siebel
Partition Property

Consider an arbitrary partition of the vertices on $G$ into two subsets $U$ and $V$. 
Partition Property

Consider an arbitrary partition of the vertices on $G$ into two subsets $U$ and $V$.

Let $e$ be an edge of minimum weight across the partition.

Then $e$ is part of some minimum spanning tree.
Partition Property

The partition property suggests an algorithm:
Prim’s Algorithm

PrimMST(G, s):
Input: G, Graph;
        s, vertex in G, starting vertex
Output: T, a minimum spanning tree (MST) of G

foreach (Vertex v : G):
    d[v] = +inf
    p[v] = NULL
    d[s] = 0

PriorityQueue Q   // min distance, defined by d[v]
Q.buildHeap(G.vertices())
Graph T           // "labeled set"

repeat n times:
    Vertex m = Q.removeMin()
    T.add(m)
    foreach (Vertex v : neighbors of m not in T):
        if cost(v, m) < d[v]:
            d[v] = cost(v, m)
            p[v] = m

return T
Prim’s Algorithm

```
6 PrimMST(G, s):
7   foreach (Vertex v : G):
8       d[v] = +inf
9       p[v] = NULL
10      d[s] = 0
11
12     PriorityQueue Q // min distance, defined by d[v]
13     Q.buildHeap(G.vertices())
14     Graph T          // "labeled set"
15
16     repeat n times:
17        Vertex m = Q.removeMin()
18        T.add(m)
19        foreach (Vertex v : neighbors of m not in T):
20           if cost(v, m) < d[v]:
21              d[v] = cost(v, m)
22              p[v] = m
```
Prim’s Algorithm

Sparse Graph:

Dense Graph:

```
PrimMST(G, s):
  foreach (Vertex v : G):
    d[v] = +inf
    p[v] = NULL
  d[s] = 0
  PriorityQueue Q // min distance, defined by d[v]
  Q.buildHeap(G.vertices())
  Graph T // "labeled set"

  repeat n times:
    Vertex m = Q.removeMin()
    T.add(m)
    foreach (Vertex v : neighbors of m not in T):
      if cost(v, m) < d[v]:
        d[v] = cost(v, m)
        p[v] = m
```

<table>
<thead>
<tr>
<th></th>
<th>Adj. Matrix</th>
<th>Adj. List</th>
</tr>
</thead>
<tbody>
<tr>
<td>Heap</td>
<td>$O(n^2 + m \lg(n))$</td>
<td>$O(n \lg(n) + m \lg(n))$</td>
</tr>
<tr>
<td>Unsorted Array</td>
<td>$O(n^2)$</td>
<td>$O(n^2)$</td>
</tr>
</tbody>
</table>
MST Algorithm Runtime:

• Kruskal’s Algorithm: \(O(n + m \lg(n))\)

• Prim’s Algorithm: \(O(n \lg(n) + m \lg(n))\)

• What must be true about the connectivity of a graph when running an MST algorithm?

• How does \(n\) and \(m\) relate?
MST Algorithm Runtime:

• Kruskal’s Algorithm:
  \[ O(n + m \ lg(n)) \]

• Prim’s Algorithm:
  \[ O(n \ lg(n) + m \ lg(n)) \]
CS 225 – Things To Be Doing

Exam 12 (programming) starts Monday, last programming exam before the final!

More Info: https://courses.engr.illinois.edu/cs225/fa2017/exams/

MP7: The final MP!
Extra Credit (+14): Monday, Dec. 4 at 11:59pm
Due: Monday, Dec. 11 at 11:59pm

Lab: lab_graphs due Sunday
lab_graphs: Due Sunday @ 11:59pm

New POTDs every M/W/F
Worth +1 Extra Credit / problem (up to +40 total)