Day 7 Notes

Zach Neveu

May 15, 2019

1 Agenda

- Quiz
- Greedy Algorithms
- Intro to matching
- Announce next homework
- Announce next project
- Reading

2 Greedy Algorithms

- Classic Case: Minimum Spanning Tree
- Also useful for many other problems
- Single "greedy algorithm" really at the root of all of them

Head Partition (review from Day 6)

- Node-based solution - finds optimal solution
- Edge-based solultion - finds optimal solution, also very similar to MST
  - Go edge-by-edge, add edge if it does not conflict

Generic Greedy Algorithm

```
def generalGreedy(g):
  sort(g.edges)
  soln = {}
  for edge in g.edges:
    if not conflicts(edge, soln)
    soln += edge
```

Weighted Head Partition

Given a weighted, directed graph, find an independent subset with maximum total weight.

- Edge Strategy: Sort edges by decreasing weight. Starting with highest weight edge, add each edge if it does not conflict, else skip.
- Node Strategy: Go through nodes in any order and select the heaviest edge pointing to the given node

Partition

- Let E be a finite set. π is a partition of E if it is a collection of disjoint subsets of E such that the subsets collectively cover E.
- $E = \{e_1, \ldots, e_8\}$, $\pi = \big\{\{e_1\}, \{e_2, e_3\}, \{e_4, e_5\}, \{e_6, e_7, e_8\}\big\}$
- $\pi.weights = \big\{\{5\}, \{3, 4\}, \{7, 8\}, \{2, 6, 1\}\big\}$
- A subset of E is **Independent** if no two elements come from the same component of π
- $\{e_1, e_4\}$ and $\{e_1, e_4, e_3\}$ are independent
- $\{e_1, e_2, e_3, e_4\}$ is not independent.
- Goal: given E and π, find an independent subset of maximum weight.
- Strategy 1: from each component, select the largest element.
- Same as node strategy for head partition!

Maximum Weighted Matching

- Given a weighted graph, find a matching with the maximum total
- Simple greedy algorithm doesn't work!
- Simple algorithm will choose b,d, a,c, ignoring a,b, c,d

Matching

- Given: a weighted graph and a matching M on the graph
- Edges in M are **matched edges**
- Edges not in M are **free edges**
- Nodes not adjacent to matched edges are **exposed**
- Nodes that are adjacent to matched edges are **matched**

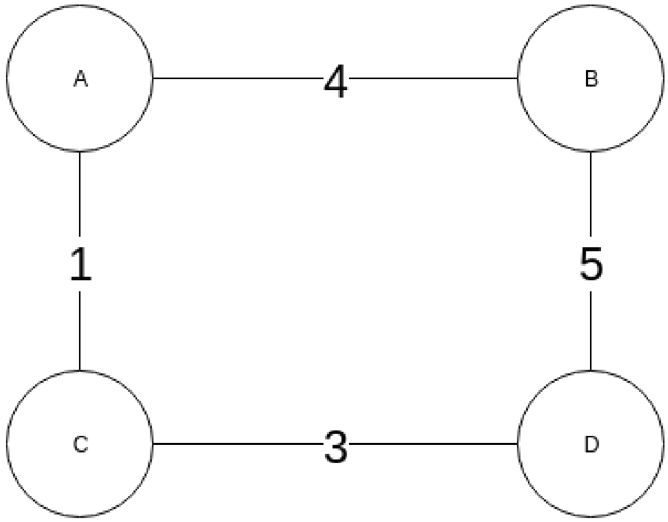

Figure 1: Weighted Matching Example

- **Augmenting Path**: a simple path in the graph beginning and ending at exposed nodes, and alternating between crossing matched and free edges.
- Augmenting Path example in 2: $(v_1, v_4, v_5, v_6, v_8, v_7, v_{10}, v_9)$
- Augmenting Path length always odd because start and end must be on exposed nodes.
- Finding longest augmenting path same as finding largest matching
- Given an augmenting path, swapping edges membership in M along the path creates a valid matching with length one longer.

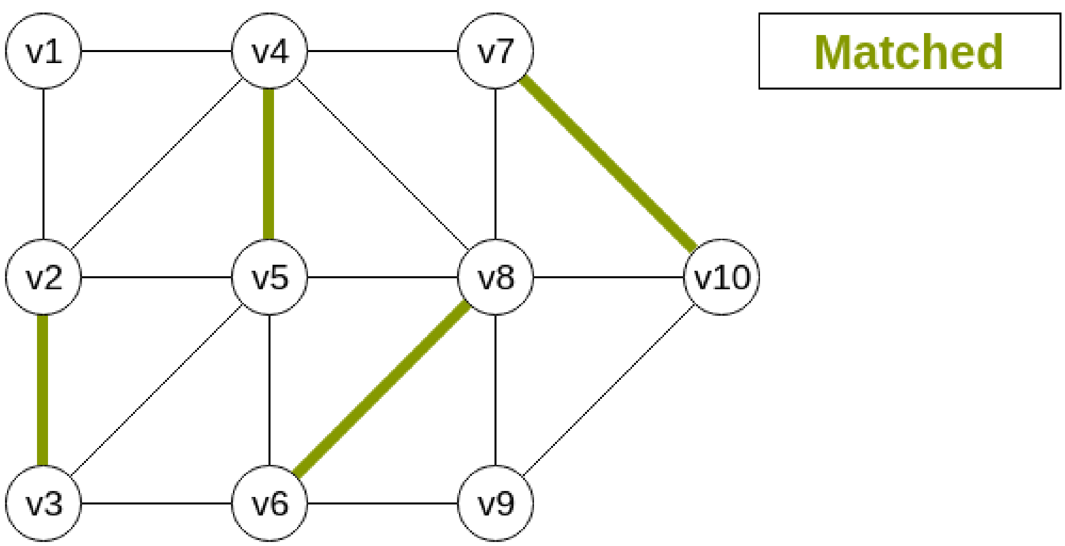

Figure 2: Matching Example