Midterm Crib Sheet

1.1 Definitions

Connected: $\forall u, v \in V(G) : \exists u, v -$ path
Component: maximal connected subgraph
Cut-edge: edge that when deleted disconnects some component
Cut-vertex: vertex that when deleted disconnects some component

Length: usually measured in edges traversed
Subgraph: subset of vertices and edges of some graph G
Induced: given vertex subset, subgraph containing those vertices and all edges between those vertices
Spanning: a subgraph that contains all vertices of graph G
Decomposition: list of subgraphs such that every edge in some G appears exactly once in some subgraph in the list
Complement: $(\overline{G})$ Use $V(G)$ and add edges between vertices that aren't adjacent in G

Isomorphism: there exists a bijection between vertex sets of some G and G' such that edge relationship are preserved
Automorphism: there exists a bijection within vertex set of some G and G' such that the edge list is preserved

Degree Sequence: list of degrees for some real or hypothetical G
Graphic Sequence: a degree sequence for some simple graph G (it can *realize* G)

Diameter: length of longest shortest path
Girth: length of shortest cycle
Eccentricity: length of shortest longest path from some u
Center: subgraph induced on vertices of minimum eccentricity
Distance: defined between u, v – length of shortest u, v-path

Degree Sum Formula: $\sum_{v \in V(G)} d(v) = 2m$
Degree Sum Formula (digraphs): $\sum_{v \in V(G)} d^+(v) = |E(G)| = \sum_{v \in V(G)} d^-(v)$
Cayley's Formula: n^{n-2} possible trees

Match: set of non-loop edges with no shared endpoints

Saturated: vertex with matched edge incident
Perfect Match: all vertices saturated in some G
M**-alternating Path**: path that alternates between edges in match M and not in M
M**-augmenting Path**: M-alternating path that starts and ends at unsaturated vertices
Stable Matching: matches between two bipartite sets, where each member of each set has an ordered preference for potential matches to the opposite set, and there are no unstable pairs – pair of vertices x, a that are not currently matched, but both vertices have a higher preference for match (x, a) over their current match partner

Vertex Cover: a vertex set that contains at least one endpoint on all $e \in E(G)$
Edge Cover: an edge set that has as at least as one endpoint of all $v \in V(G)$
Independent Set: set of vertices on a graph G are not connected by an edge
Dominating Set: set of vertices on a graph G such that all vertices of G are either in S or have a neighbor in S

1.2 Graph Classes

Simple: no multi-loops or self edges
Loopy: can have loops
Multigraph: can have multi-edges
Empty: no edges and some number of vertices
Trivial: no edges and a single vertex
Null: no vertices or edges
Complete: has all possible edges
Bipartite: union of two disjoint independent vertex sets
Digraph: directed graph, has directed edges

Tree: connected, undirected, acyclic (has no cycles)
Forest: undirected, acyclic
Leaf: vertex of degree-1

Path: (P_n) doesn't repeat vertices or edges
Trail: can repeat edges
Walk: can repeat vertices and edges
Closed P/T/W: P/T/W that starts and ends at same vertex

Cycle: (C_n) closed path
Triangle: cycle of length 3

Star: (S_n) connected n-vertex graph with $n-1$ vertices of degree-1

k-regular: has all vertices with degree of k
Eulerian: has closed trail containing all edges (Euler Tour)
Graceful: graph with graceful labeling – all n vertices and m edges labeled uniquely with $0 \dots m$, such that each edge has a unique value computed as absolute difference of its endpoints' labels

1.3 Theorems

Havel-Hakimi: A sequence $S = \{d_1, d_2, \dots, d_n\}$ is a graphic sequence iff sequence $S' = \{d_2 - 1, \dots, d_{d_1+1} - 1, d_{d_1+2}, \dots, d_n\}$ is a graphic sequence, where $d_1 \geq d_2 \geq \dots \geq d_n$ and $n \geq 2$ and $d_1 \geq 1$
Berge: match M is maximum iff G has no M-augmenting paths
Hall: X, Y-bipartite graph G has a matching that saturates X if and only if $|N(S)| \geq |S|$ for all possible $S \subseteq X$
Tutte: graph G with a perfect match satisfies the inequality $\forall S \subseteq V(G) : o(G-S) \leq |S|$
König-Egerváry: if G is a bipartite graph, then the size of a maximum matching in G equals the minimum size of a vertex cover

1.4 Algorithms

```
procedure CREATEEULERTOUR(Graph G)
    T ← ∅                                              ▷ Initialize Eulerian circuit
    G' ← G
    Start at any vertex v
    while G' ≠ ∅ do
        Select at edge e to travel along, where (G' − e) is not disconnected
        T ← e
        G' ← (G' − e)
    return T
```

procedure createPrufer(Tree T with vertex set S)
- $a \leftarrow \emptyset$ ▷ Initialize Prüfer code to null
- **for** $i = 1 \ldots (n-2)$ **do**
  - $l \leftarrow$ least remaining leaf in T
  - $T \leftarrow (T - l)$
  - $a_i \leftarrow$ remaining neighbor of l in T
- **return** a

---

procedure recreateTree(Prüfer code a created with vertex set S)
- $V(T) \leftarrow S$ ▷ Tree has vertex set S
- $E(T) \leftarrow \emptyset$ ▷ Initialize tree edges as empty
- initialize all vertices in S as unmarked
- **for** $i = 1 \ldots (n-2)$ **do**
  - $x \leftarrow$ least unmarked vertex in S not in $a_{i \ldots (n-2)}$
  - mark x in S
  - $E(T) \leftarrow (x, a_i)$
- $x, y \leftarrow$ remaining unmarked vertices in S
- $E(T) \leftarrow (x, y)$
- **return** T

---

procedure Kruskal(Graph $G = \{V(G), E(G), W(G)\}$)
▷ Note: $W(G)$ is a numeric *weight* for each edge in $E(G)$
- $V(T) \leftarrow V(G)$ ▷ Spanning tree T will have all vertices of G
- $E(T) \leftarrow \emptyset$ ▷ Edge set of T initially null
- sort $W(G)$ and correspondingly $E(G)$ by nondecreasing values in $W(G)$
- **for all** $w \in W(G), e \in E(G)$ **do**
  - **if** numComponents$(T + e) <$ numComponents(T) **then**
    - $E(T) \leftarrow E(T) + e$
  - **if** numComponents$(T) = 1$ **then**
    - **break**
- **return** T

---

procedure Prim(Graph $G = \{V(G), E(G), W(G)\}$)
▷ Note: $W(G)$ is a numeric *weight* for each edge in $E(G)$
- $V(T) \leftarrow \emptyset$ ▷ Spanning tree's vertices initially null
- $E(T) \leftarrow \emptyset$ ▷ Edge set of T initially null
- $V(T) \leftarrow$ randomSelect$(V(G))$ ▷ Randomly select one vertex from G
- **while** $V(T) \neq V(G)$ **do**
  - $e \leftarrow \min(W(u, v)) \in E(G) : u \in V(T), v \notin V(T)$
    ▷ Minimum weight edge in G with only one vertex in T
  - $E(T) \leftarrow E(T) + e$
  - $V(T) \leftarrow V(T) + v$
- **return** T

```
procedure Dijkstra(Graph G = {V(G), E(G), W(G)}, vertex u)
                                              ▷ Finding all distances from u
    for all v ∈ V(G) do
        D(v) ← ∞                              ▷ Distances from u initially infinity
    D(u) ← 0
    S ← V(G)                                  ▷ Unvisited set
    while S ≠ ∅ do
        w ← min(D(v), v ∈ S)
                        ▷ Current vertex considered has minimum distance in unvisited set
        for all x ∈ N(w), x ∈ S do
            t ← W(w, x)                       ▷ Weight of edge between w and x
            if D(w) + t < D(x) then
                D(x) ← D(w) + t
        S ← S − w                             ▷ Remove w from unvisited set
    return D
```

```
procedure MatchBipartite(X, Y-bigraph G)
    M ← ∅                                     ▷ M initially empty
    do
        P ← AugPathAlg(G, M)                  ▷ New augmented path found with M, G
        M ← MΔP                               ▷ Symmetric difference between M, P
    while P ≠ ∅
    return M
```

```
procedure AugPathAlg(X, Y-bigraph G and matching M = (V_M, E_M))
    G' ← G
    Orient G' : ∀e ∈ E_M : e(x_i, y_j) = e(y_j → x_i); ∀e ∉ E_M : e(x_i, y_j) = e(x_i → y_j)
    Add vertex s to G' with edges ∀x_i ∈ X, x_i ∉ V_M : (s → x_i)
    Add vertex t to G' with edges ∀y_j ∈ Y, y_j ∉ V_M : (y_j → t)
    P ← ShortestPathBFS(G', s, t)             ▷ Use BFS to find shortest path from s to t
    return P − {e(s, x_i), e(y_j, t)}         ▷ Return path without added edges
```

```
procedure GaleShapley(n men and n women and preference lists for each)
    while not done do
        Each man proposes to their highest preference woman
        who has not yet rejected them
        if each woman gets exactly 1 proposal then
            return these proposals as a stable match
        else
            Women with 2 or more proposals reject all proposals
            except their highest preference
```