Lecture Notes: Graph Algorithms

Transcribed from Handwritten Notes

October 15 & October 22, 2025

1 Bipartite Matching (Oct 15)

1.1 Definitions and Hall's Condition

Let $G = (V, E)$ be a bipartite graph. For a set of vertices $S \subseteq V$, let $N(S)$ denote the set of neighbors of S.

- **Hall's Marriage Theorem (Contrapositive):** If $|S| > |N(S)|$, then there is no perfect matching in the graph.
- If a matching is not perfect, there exists an unmatched vertex.
- To find a matching, we start from an unmatched vertex and perform a Breadth-First Search (BFS) to find an augmenting path.

1.2 Primal-Dual Relationship

The problem considers the relationship between the Primal problem (Matching) and the Dual problem (Vertex potentials).

- **Weak Duality:** The sum of dual variables is related to the edge weights. For any matching, the sum of the duals of the vertices is less than or equal to the corresponding edge weights (depending on minimization/maximization).
- **Strong Duality:** Dual Optimal = Primal Optimal.

2 Min-Weight Perfect Matching (Oct 22)

2.1 The Algorithm (Primal-Dual Method)

Given a graph G with edge costs, we wish to find the minimum weight perfect matching. We assign dual weights (potentials) to vertices, say u_i and v_j.

2.1.1 Initialization

Initialize all dual weights to 0 (or a feasible assignment).

2.1.2 The Procedure

1. **Construct Tight Graph:** Construct a subgraph using only the **tight edges**. An edge is tight if the dual constraints are met with equality.

2. **Check for Matching:** If the graph of tight edges contains a perfect matching, we are done. Output the matching.

3. **Update Duals:** If the tight edge graph does not have a perfect matching, we must adjust the vertex weights.

   - Find a set S (visited in the Hungarian tree) such that $|S| > |N(S)|$.
   - Calculate ϵ based on the slack of non-tight edges.
   - **Update Rule:**
     - Add ϵ to all vertices in S.
     - Subtract ϵ from all vertices in $N(S)$.
   - This process keeps edges between S and $N(S)$ tight, while allowing new edges to become tight (enter the admissible subgraph).

2.1.3 Cost Analysis

The total change in the dual objective function is calculated as:

$$\text{Change} = \epsilon|S| - \epsilon|N(S)| = \epsilon(|S| - |N(S)|)$$

Since $|S| > |N(S)|$, the dual objective strictly increases, moving strictly closer to the optimum.

3 Network Flow

3.1 Basics

- **Flow Conservation:** For any internal node (junction), Incoming Flow = Outgoing Flow. Junctions cannot create, store, or destroy flow.

- **Source and Sink:** Flow originates at Source (S) and terminates at Sink (T).

- **Capacity:** The maximum flow allowed through an edge.

3.2 Residual Graphs and Optimal Flow

A greedy approach does not guarantee a maximum flow.

- **Example:** Consider a "diamond" graph structure. If we greedily choose path $S \to a \to b \to T$, we might saturate the middle edge $a \to b$, resulting in a total flow of 1, whereas the optimal flow might be 2.

- **Correction Strategy:** We define residual edges to allow "undoing" a decision.

- When traversing *along* an edge, we increase flow. When traversing *opposite* to an edge (backward), we decrease flow.