# Theory Assignment-5: ADA Winter-2024

Nikhil Kumar (2022322)   Nikhil (2022321)

23 April 2024

## 1 Formulating the Problem as a Flow Network Problem:

We approch this problem by modeling it as a flow network. In this case, we build a directed graph G, in which every box i is represented by two vertices, ui and vi. If and only if box i can be inserted within box j, there is an edge ui to vj. The relationships between boxes that allow one box to fit inside another are represented by a graph.

This problem reduces to bipartite matching problem and we have done it using Ford-Fulkerson algorithm. We construct a bipartite graph $G$ with vertex set $U \cup W$ as follows:

- Each box is a node in both U and W
- If $u$ can be rotated so that it fits inside $v$, then for every pair of nodes where $u$ is from set $U$ and $v$ is from set $W$, we generate an edge $e = (u, v)$ in the bipartite graph $G$.

### 1.1 Problem Constraints:

One box may only accommodate exactly one other box; that is, two boxes cannot be placed inside the same box at the same time because this make it a NP-Hard Problem.

## 2 Explanation for Maximum Flow or Minimum Cut

In order to solve the box nesting problem, we utilize minimal cut or maximum flow ideas in a flow network since they enable us to determine the ideal box arrangement. We are able to think of this as a type of matching in a graph when we consider how boxes fit together. One box is directly within the other if there is an edge matching between the two boxes.

We can convert this information into a maximum matching in graph $G$ by knowing the maximum number of nested boxes and comprehending how they contain one another. How to achieve it is as below:

- We select edge $e = (u, v)$ in graph $G$ if box $u$ contains box $v$, where $u$ comes from set $U$ and $v$ comes from set $W$.
- We establish the concept of one box being immediately within another in a legitimate nesting of boxes. This indicates that box $i$ is situated inside box $j$, and that box $i$ and box $j$ are not nested inside any other boxes. A distinct hierarchy and structure are guaranteed by this direct nesting relationship in the box layout.
- Each box can now be either visible (i.e., not within any other box and plainly visible from the outside) or directly inside exactly one other box in any legitimate nesting scenario. This distinction is important because it establishes the relationship between the boxes.
- Any permissible box nesting configuration under these circumstances leads directly to a matching in graph $G$. If and only if box $i$ is situated exactly within box $j$, then an edge $u_i v_j$ is included in this matching. Box arrangement analysis and optimization may be done in an organized and formalized manner thanks to this mapping from nesting connections to graph matching

Now let's generalize,

- Assume that graph $G$ has a matching $M$. Based on the edges in $G$, this matching essentially couples boxes so that each box is directly nested within at most one other box. For example, if $G$ has an edge $u_iv_j$, it means that box $i$ is matched to be inside box $j$ exactly.
- The important thing to remember is that a valid nesting arrangement is guaranteed by the matching $M$. The smallest dimension (height, breadth, or depth) of box $i$ must be strictly less than the smallest dimension of box $j$ for each edge $u_iv_j$ in $G$. Because it ensures that box $i$ can fit inside box $j$ without any conflicts or overlaps, this condition is very important.
- Let's now explore why matching $M$ results in a nesting configuration that is valid and does not contain any cycles of nested boxes. Think about the assignment that matching $M$ causes. The dimension comparison rule would be broken if there were a cycle of nested boxes (such as box A within B, B inside C, and C inside A). In particular, in a cycle, one of the boxes would have to be bigger than the box it is intended to be inside, defying our first theory that size compatibility permits legitimate nesting. Consequently, matching $M$ assures a legitimate nesting arrangement free from nesting conflicts or cycles since it guarantees that each box is directly nested within at most one other box while adhering to the size limitations.

## 3 Time Complexity Analysis

### 1. Creating the Flow Network $G$:

Creating the graph involves determining if each pair of boxes is compatible, which takes $O(n^2)$ time. This is because you have a total of $2n$ vertices (each box represented by two vertices) and determining compatibility between each pair of boxes results in $O(n^2)$ comparisons.

### 2. Using the Ford-Fulkerson Algorithm:

- The Ford-Fulkerson algorithm runs in $O(\text{value(flow)}(|V| + |E|))$ time.
- Here, $|V|$ denotes the number of vertices and $|E|$ denotes the number of edges in the graph.
- Given that each box is represented by two vertices, and the number of edges $|E|$ is related to the number of connections between the boxes, which is $O(n^2)$.
- Thus, $|V| = 2n$ and $|E| = n^2$.
- So, the time complexity becomes $O(\text{value(flow)}(2n + n^2)) = O(\text{value(flow)}n^2)$.

### 3. Time Complexity Drop:

Since $|V| = 2n$ and $|E| = n^2$, the time complexity drops to $O(n^3)$ when substituted into the Ford-Fulkerson algorithm's time complexity equation.
In summary, the overall time complexity for constructing the flow network and running the Ford-Fulkerson algorithm is dominated by the Ford-Fulkerson algorithm itself, which results in a time complexity of $O(n^3)$.

## 4 Algorithm Description

1. We make a bipartite graph having vertex set $U$ and $W$. Each vertex is in both $U$ and $W$.
2. Now we connect the vertex $u$ from Vertex Set $U$ to vertex $v$ from Vertex Set $W$ if and only if box $v$ can be contained in box $u$, and we mark the edge capacity as 1.
3. Now we connect all the vertices of set $U$ with a source vertex $s$ with edge capacity as 1. Similarly, all the vertices of set $W$ are connected with the sink vertex $t$ with edge capacity as 1.
4. Finally, we can run the Ford-Fulkerson's algorithm directly on this, and this will return us the maximum flow, which will be the minimum number of boxes that are visible.

## 5 References Taken From Web

### Studied some examples of the Max Flow and Min Cut theorem

Source: `https://www.scaler.com/topics/data-structures/maximum-flow-and-minimum-cut/`

### Reference for Time Complexity

Source: `https://math.stackexchange.com/questions/361685`