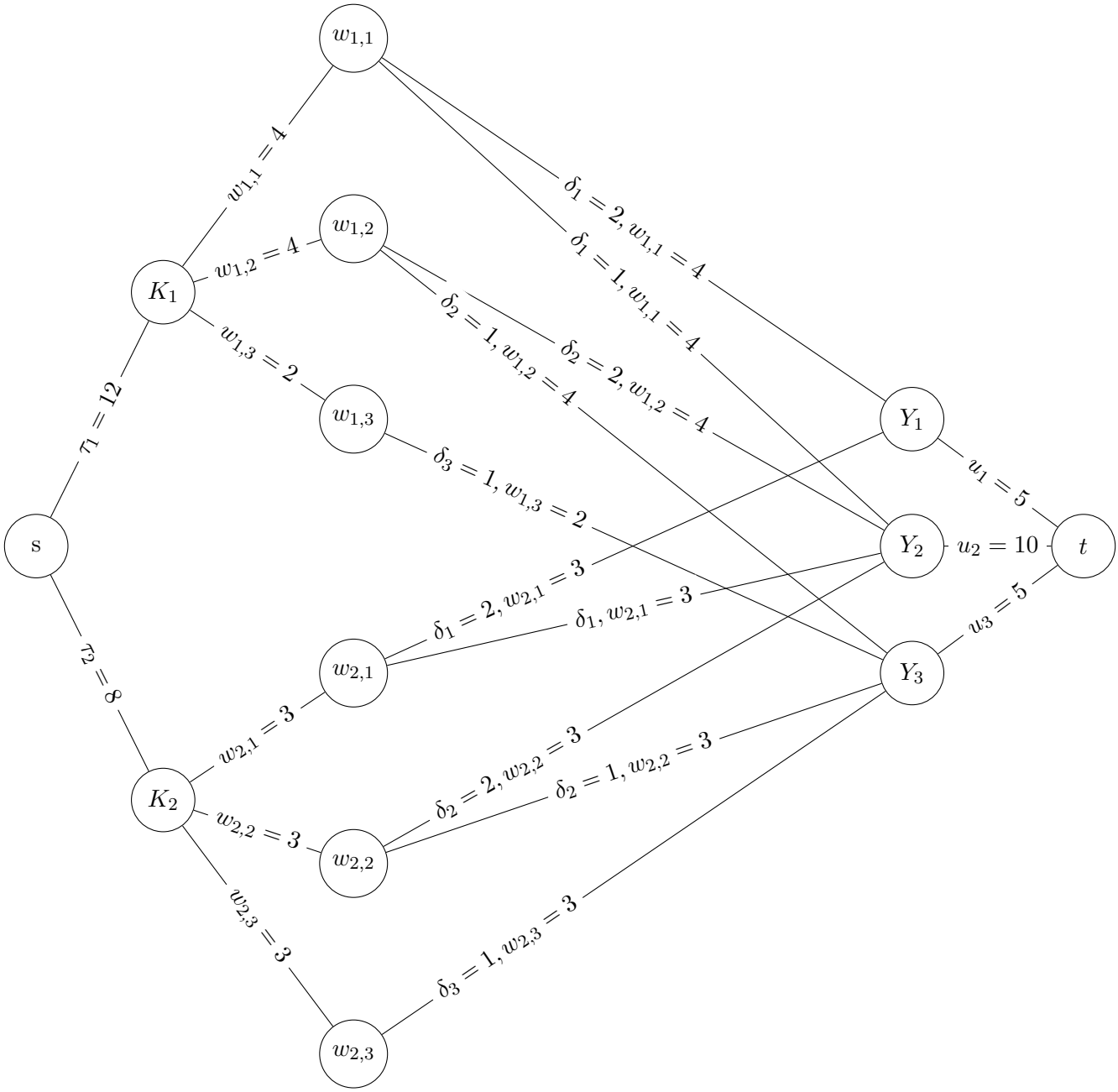
COMP2007 - Assignment 4

Matthew Watson
SID: 440267858

1. **Consider the case when $Y = 3, k = 2, \delta_1 = \delta_2 = 2$ and $\delta_3 = 1$**

   (a) **Formulate the problem of determining a schedule with maximum number of Christmas trees sold as a network flow problem.**

   Y=year
   K=forest
   $w_{i,j}$=maximum number of trees that can be harvested from forest i in year j.



(b) **Argue why your algorithm is correct.**

In this case, we have several constraints which will contribute to the design of our flow network, these are:

- k: the number of forests, in this case, 2.
- Y: the number of years, in this case, 3.
- τ_i (added as per overall specification): the maximum number of trees that can be harvested from forest K_i over the entire Y years. In this case, we say $\tau_1 = 12$ and $\tau_2 = 8$, meaning that we are restricted over the entire period to harvesting 12 trees from forest K_1 and 8 trees from forest K_2.
- $w_{i,j}$ (added as per overall specification): the maximum number of trees that can be harvested from forest K_i in year Y_j. In this case we have $w_{1,1} = w_{1,2} = 4, w_{1,3} = 2$ and $w_{2,1} = w_{2,2} = w_{2,3} = 3$.
- δ_j: the lifetime of a tree harvested in year Y_j, such that $\delta_1 = 3$ means a tree harvested in year 1 can only be sold in year 1,2 or 3. In this case, we have $\delta_1 = \delta_2 = 2$ and $\delta_3 = 1$
- u_i (added as per overall specification): The maximum number of trees that can be sold in year Y_i. This does not necessarily restrict the number of trees harvested, however, given for that year the δ_i value is greater than 1. In this case, we have $u_1 = u_2 = 5$ and $u_2 = 10$.

Given the above, we can construct this as a network flow problem in the following order from source s to sink t.

i. From our source, for every forest i, add an edge to a respective k_1,...,k_i with weight τ_i. This enforces the maximum overall harvest for each forest represented by τ_k. Given this is a restraint on our entire harvest across all years, we place this as the first constraint from the source to our first set of edges to ensure no more than this amount of trees can enter our network flow.

ii. From each k_i node (each i forest), add an edge to a respective $w_{i,j}$ node for each year j with edge weight $w_{i,j}$: the number of Christmas trees maturing in forest i, year j. This step ensures that we are bounded by the maximum tree harvest for each forest in each year whilst not exceeding for any forest, the maximum amount of harvest across all years.

iii. From each $w_{i,j}$ node, add an edge to a respective Y_i node with edge weight $w_{i,j}$. This concentrates our maximum yield from a given field and year to their respective years.

iv. From each $w_{i,j}$ node, for its given δ_i value z, add an edge to its respective $Y_{i+(z-1)}$ node with edge weight $w_{i,j}$. That is, for a given $w_{i,j}$ node with δ_i value 2, add an edge to Y_{i+1} with edge weight $w_{i,j}$. For this example, we add an edge from $w_{1,1}$ to Y_2, $w_{1,2}$ to Y_3, $w_{2,1}$ to Y_2 and $w_{2,2}$ to Y_3. This allows any excess harvest from year i that was held off market to prevent flooding it, to be used in following years bounded by their expected lifetime given by δ_i, such that excess from year 1 can be used in year 2 and excess from year 2 can be used in year 3. We then have the following final constraint:

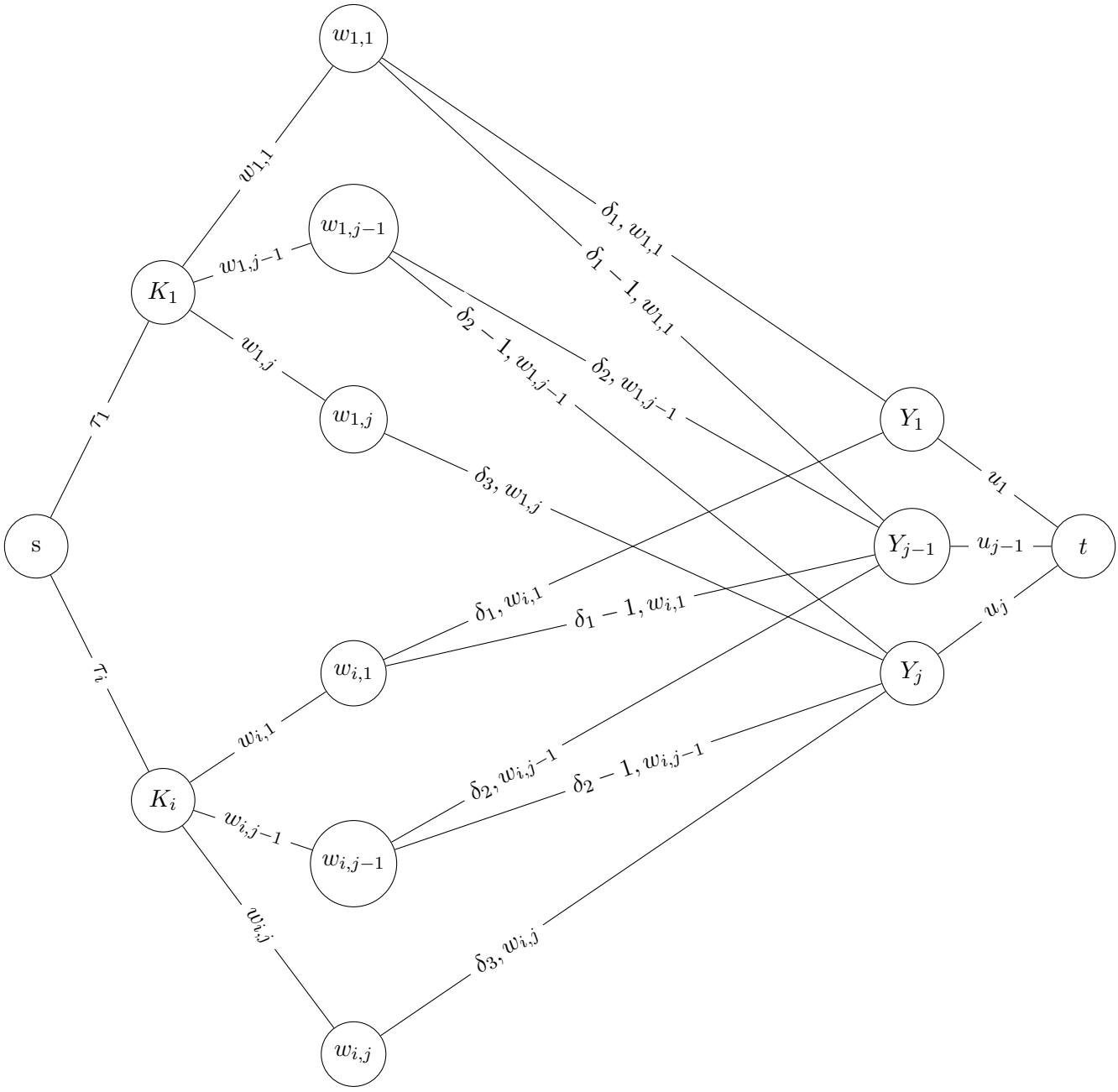
v. For each Y_i node, add an edge to t with edge weight u_i. This bound ensures the final constraint before going to market that no more than u_i trees are sold in any given year Y_i to prevent flooding the market. Given the previous step, however, any excess can be redistributed into later years to achieve a maximum number of trees sold schedule.

With the above network flow complete, running Ford-Fulkerson's algorithm to find the max flow will return the maximum flow of this network and thus the value of an optimal cutting schedule with maximum trees sold over the entire Y years.

2. **Generalise your solution to k forests, Y years and variable tree lifespans.**

   (a) **Formulate the problem of determining a schedule with maximum profit (maximum number of Christmas trees sold) as a network flow problem for a given Y, k and $\delta 1, ..., \delta Y$.**

   j=year number
   i=forest number
   Y=year
   K=forest
   $w_{i,j}$=maximum number of trees that can be harvested from forest i in year j.
   $\delta_i - 1$=representation of adding an edge from $w_{i,j}$ to Y_{i+1} for each year in δ_{i-1}



(b) **Argue why your formulation is correct.**

In this case, we have several constraints which will contribute to the design of our flow network, these are:

- i: the number of forests
- j: the number of years
- τ_i: the maximum number of trees that can be harvested from forest K_i over the entire j years.
- $w_{i,j}$: the maximum number of trees that can be harvested from forest K_i in year Y_j.
- δ_j: the lifetime of a tree harvested in year Y_j, such that $\delta_1 = 3$ means a tree harvested in year 1 can only be sold in year 1,2 or 3.
- u_i: The maximum number of trees that can be sold in year Y_i. This does not necessarily restrict the number of trees harvested, however, given for that year the δ_i value is greater than 1.

Given the above, we can construct this as a network flow problem in the following order from source s to sink t.

i. From our source, for every forest i, add an edge to a respective k_1,...,k_i with weight τ_i. This enforces the maximum overall harvest for each forest represented by τ_k. Given this is a restraint on our entire harvest across all years, we place this as the first constraint from the source to our first set of edges to ensure no more than this amount of trees can enter our network flow.

ii. From each k_i node (each i forest), add an edge to a respective $w_{i,j}$ node for each year j with edge weight $w_{i,j}$: the number of Christmas trees maturing in forest i, year j. This step ensures that we are bounded by the maximum tree harvest for each forest in each year whilst not exceeding for any forest, the maximum amount of harvest across all years.

iii. From each $w_{i,j}$ node, add an edge to a respective Y_i node with edge weight $w_{i,j}$. This concentrates our maximum yield from a given field and year to their respective years.

iv. From each $w_{i,j}$ node, for its given δ_i value z, add an edge to its respective $Y_{i+(z-1)}$ node with edge weight $w_{i,j}$. That is, for a given $w_{i,j}$ node with, for example, δ_i value 3, add an edge to Y_2 and Y_3 with edge weight $w_{i,j}$. This allows any excess harvest from year i that was held off market to prevent flooding it, to be used in following years bounded by their expected lifetime given by δ_i and the following final constraint:.

v. For each Y_i node, add an edge to t with edge weight u_i. This bound ensures the final constraint before going to market that no more than u_i trees are sold in any given year Y_i to prevent flooding the market. Given the previous step, however, any excess can be redistributed into later years to achieve a maximum number of trees sold schedule.

With the above network flow complete, running Ford-Fulkerson's algorithm to find the max flow will return the maximum flow of this network and thus the value of an optimal cutting schedule with maximum trees sold over the entire Y years.

(c) **Prove an upper bound on the time complexity of your algorithm.**

i. **Building the network flow**
Our first for loop runs for every forest, i.e. k times, leading to a time complexity of $O(k)$. The for loop within that runs for every year, i.e. Y times, leading to a time complexity of $O(Y)$. The for loop within that runs for the size of delta for each year. Given delta cannot be larger than the amount of years, we say this will run at most Y times, hence a time complexity of $O(Y)$. Overall this results in a run time of $O(k) \times O(Y) \times (Y) = O(kY^2)$.

ii. **Running Ford-Fulkerson**
As per the assignment specification, this algorithm correctly returns the maximum flow of the given flow network G in $O(m^2 logC)$ time. m is the number of edges in the input flow network, which is bounded by kY^2, and C is the maximum flow in G. Therefore, this results in an overall time of $O(kY^2 logC)$

iii. **Printing the max flow**
Returning a tuple value runs in $O(1)$ time.

In summary, we observe the overall time complexity as:

$$\begin{aligned} &O(kY^2) + O(kY^2 logC + O(1) \\ &= O(kY^2 logC) \end{aligned}$$