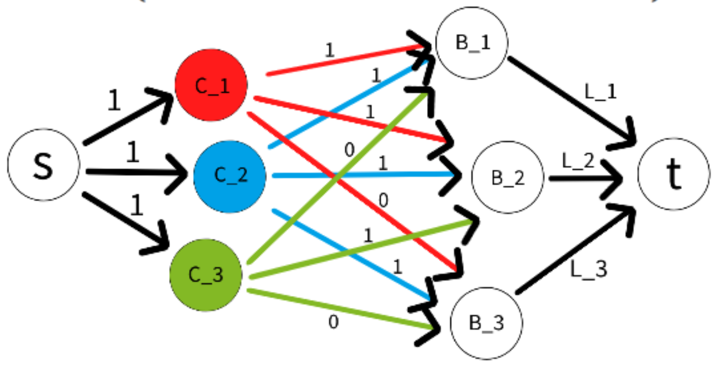
Philippe WS CSC-349

Assignment 8

$$u_{c_i,b_j} = \left\{ \begin{array}{ll} 1, & if \sqrt{(b_{j_x} - c_{j_x})^2 + (b_{j_y} - c_{j_y})^2} \leq R \\ 0, & else \end{array} \right\}$$



1.) Testing Base Station Coverage

a. To begin, the first layer, or source node, is a necessary transformation to allow the shortest augmenting path algorithm to perform operations on the graph as if it were a flow network.

The edges (S, C_i), from source S to any client node C_i, with edge weights one represent the constraint that a client will be subscribing to only one tower.

The nodes C_i represent each described client.

The edges (C_i, B_j) from any client C_i to any tower B_j, are defined by the piecewise function described in the above image. To clarify, the function is describing calculating the euclidian distance between the client C_i and tower B_j. If this distance is less than the maximum tower coverage, described as R, then it is possible for a client to connect to said tower. Thus, the edge capacity should be one on edge (C_i, B_j).
If the calculated distance is greater than R, then it is not possible for the client to connect to the said tower, and thus the edge weight should be zero, or the edge may be omitted from the flow network entirely.

The nodes B_j represent each described tower.

The edges (B_j, t) from any tower B_j to the sink t, with edge weights L_j, are representative of the maximum allowed clients at each tower. The capacity L_j prevents more than the allotted number of clients from connecting to tower B_j.

b. To determine which mobile clients are assigned to each base station since the problem has been transformed into a flow network, we can utilize the shortest augmenting path algorithm to complete the flows on the transformed network. Once this has been completed, we can inspect the flow on all edges (C_i, B_j) and if the flow is one on any edge, then client C_i is assigned to tower B_j.

2.) IT Department Holiday Scheduling

A.) Problem w/o Constraint #3

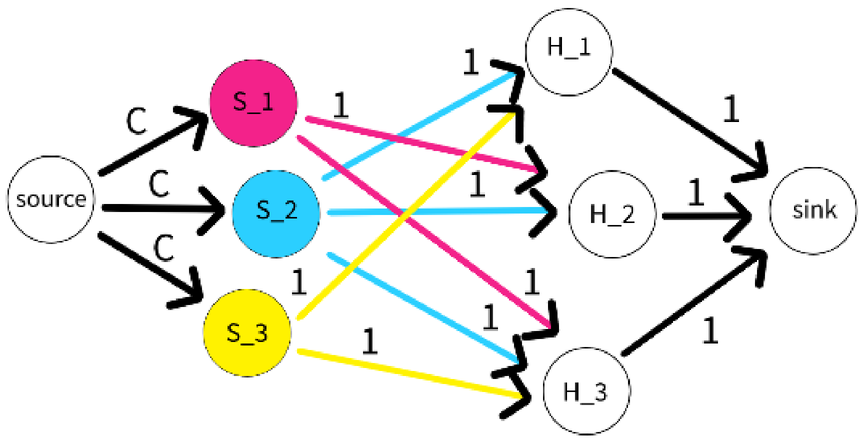
Diagram Key:

H_i: A holiday i where a supervisor may be scheduled. Also utilized as node labels.

S_x: The subset set of holidays a supervisor x wishes to be assigned to. Also utilized as node labels.

C: The maximum number of holidays any supervisor may be assigned to.

S_1 = { H_2, H_3 }, S_2 = { H_1, H_2, H_3 }, S_3 = { H_1, H_3 }



a.1.) The maximum flow obtained does not solve the problem in and of itself, but rather, is found in the flow along specific edges, obtained by performing the shortest augmenting path algorithm. The goal is to determine which supervisor is assigned to which holiday, while following the defined constraints, excluding one. By constructing the graph as shown, it is possible to ensure that each constraint is met and a solution is obtainable, should it exist. The constraints are ensured by a few key attributes of the network.

First, it is ensured that no supervisor is assigned to more than the maximum allowed holidays by the edge capacities C, on the edges from the source node to any supervisor node S_x.

Secondly, it is ensured that each supervisor S_x only can be assigned a holiday H_i by the edges (S_x, H_i), which only exists if said holiday H_i is an element of subset S_i.

Lastly, it is ensured that one and only one supervisor is assigned to each holiday by the edge capacities from the holiday nodes H_i to the sink t, which are one.

a.2.) Utilizing the list of holidays each supervisor S_x is willing to work on, we can construct the transformed flow network and perform the shortest augmenting path algorithm upon it. This will return to us a maximum flow *F*, which is equivalent to the count of holidays that supervisors have been scheduled for. Since it is a requirement that no holiday is left unaccounted for, if a flow *F* would be less than m, where m is the number of holidays, this would indicate that not every holiday is covered. Thus, there would be no solution for that collection of supervisors.

a.3.) After performing the shortest augmenting path algorithm on the transformed flow network, we can inspect the flows along each of the edges (S_x, H_i). If an edge contains a flow of one, then a supervisor S_x has been assigned to work on holiday H_i.

B.) Problem with constraint #3

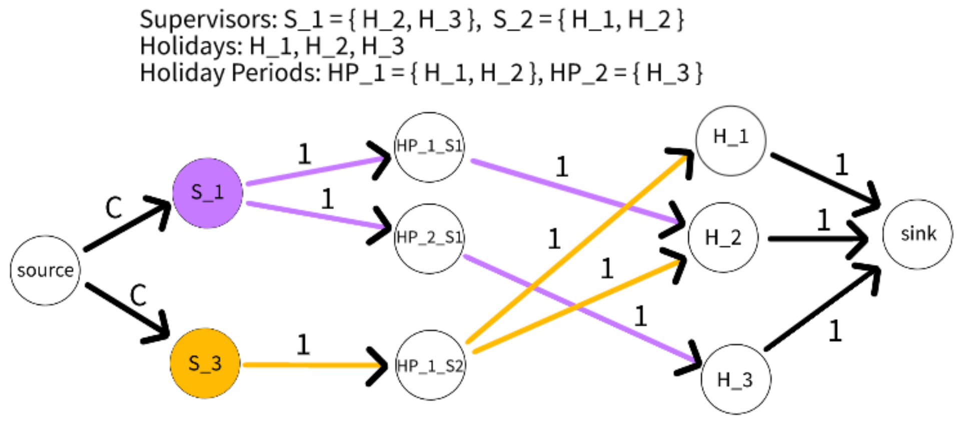
Diagram Key:

H_i: A holiday i where a supervisor may be scheduled. Utilized as node labels.

$HP_{j,\, S\_x}$: The subset of holiday periods that contain the holidays H_i, were $H_i \in S_x$.

S_x: The subset set of holidays a supervisor x wishes to be assigned to. Utilized as node labels.

C: The maximum number of holidays any supervisor may be assigned to.


Supervisors: S_1 = { H_2, H_3 }, S_2 = { H_1, H_2 }
Holidays: H_1, H_2, H_3
Holiday Periods: HP_1 = { H_1, H_2 }, HP_2 = { H_3 }

b.1.) As stated earlier, the maximum flow obtained does not solve the problem in and of itself, but rather, the solution is present in the flow along specific edges, obtained by performing the shortest augmenting path algorithm. The goal is to determine which supervisor is assigned to which holiday, while following the defined constraints, but now including the third constraint. By constructing the graph as shown, it is possible to ensure that each constraint is met and a solution is obtainable should it exist. The constraints are ensured by a few key attributes of the network.

First, it is ensured that no supervisor is assigned to more than the maximum allowed holidays by the edge capacities C, on the edges from the source node to any supervisor node S_x.

Secondly, it is ensured that no supervisor is assigned to more than one holiday period HP_j, by the set of edges (S_x, $HP_{j,\, S\_x}$), connecting a supervisor S_x to a holiday period $HP_{j,\, S\_x}$. The node $HP_{j,\, S\_x}$, and edge (S_x, $HP_{j,\, S\_x}$) will exist for $\{ \forall H_i \in S_x : H_i \in HP_j \}$. The capacity of the said edge is one since the supervisor should only be assigned once to each holiday period.

Thirdly, it is ensured that each supervisor S_x only can be assigned a holiday H_i contained within their subset S_x, by the edges ($HP_{j,\, S\_x}$, H_i), which exist if for holiday H_i, $\{ H_i : H_i \in S_x \land H_i \in HP_j \}$. The capacity of the said edge is one since the supervisor should only be assigned once to each holiday.

Lastly, it is ensured that one and only one supervisor, in general, is assigned to each holiday by the edge capacities from the holiday nodes H_i to the sink t, which are one.

b.2.) Utilizing the list of holidays each supervisor S_x is willing to work on, we can construct the transformed flow network and perform the shortest augmenting path algorithm upon it. This will return to us a maximum flow *F*, which is equivalent to the count of holidays that supervisors have been scheduled for. Since it is a requirement that no holiday is left unaccounted for, if a flow *F* would be less than m, where m is the number of holidays, this would indicate that not every holiday is covered. Thus, there would be no solution for that collection of supervisors.

b.3.) After performing the shortest augmenting path algorithm on the transformed flow network, we can inspect the flows along each of the edges ($HP_{j,\, S\_x}$, H_i). If an edge contains a flow of one, then a supervisor S_x has been assigned to work on holiday H_i.