1 Interval Partitioning

Definition 1 (Depth)**.** *Given a set of intervals, the* depth of this set is the maximum number of open intervals that contain a time t.

Lemma 2. *In any instance of interval partitioning we need at least depth many classrooms to schedule these intervals/courses.*

Proof This is simply because by definition of depth there is a time t and depth many courses that are all running at time t. That means that these courses are mutually in-compatible, i.e., no two of them can be scheduled at the same classroom. So, in any schedule we would need depth many classrooms. ■

Theorem 3. *In Interval Partitioning problem Greedy is optimum.*

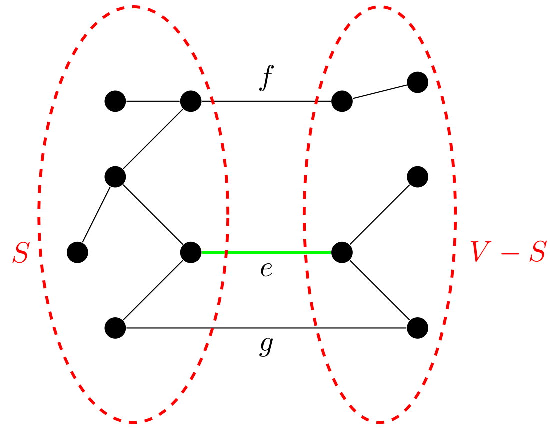
Proof Suppose that the greedy algorithm allocates d classrooms. Our goal is to prove that $d \leq depth$. Note that this is enough to prove the theorem because by the previous lemma, $depth \leq OPT$. So, putting these together we get $d \leq OPT$. On the other hand, by definition of OPT, we know $OPT \leq d$. So, we must have $d = OPT$.

To show $d \leq depth$, by definition of *depth*, it is enough to find a time t^* such that $\geq d$ open intervals contain t^*. Let t be the time that we allocate the d-th classroom. At this time we were suppose to schedule, say j-th, course but all classrooms were already occupied so greedy had to allocate the d-th classroom. The main observation is that, by description of the algorithm, every course we have schedule so far must start before $s(j)$. Furthermore, BC all classrooms are occupied at time t there must be $d-1$ courses which are still running, i.e., $d-1$ open intervals. Now, let $t^* := t + \epsilon$ where $\epsilon > 0$ is chosen small enough such that none of those $d-1$ jobs together with job j end before or at t^*. But then we have d running courses at time t^* and this implies $depth \geq d$. ■

Lemma 4 (Cut Property)**.** *Let $(S, V-S)$ be a cut in G and e be the smallest edge of this cut, then e is in every MST.*

Proof We prove by contradiction. Let T^* be a MST such that $e \notin T^*$. We want to use the *exchange argument*. Namely, find an edge $g \in T^*$ such that $g \in (S, V-S)$ and $T^* - e + g$ is all a MST. But, since e is the smallest edge of G in $(S, V-S)$ we must have $c_e < c_g$, so $c(T^* - e + g) = c(T^*) - c_e + c_g < c(T^*)$ which is a contradiction with optimality of T^*.

So, the whole question is how to find this edge g. One idea is to let g be an arbitrary edge of T^* in the cut $(S, V-S)$; note that T^* must have at least one edge because it is connected and spanning. But, we saw that this cannot work, for example in the picture below the edge e cannot be swapped with f because the resulting subgraph will be disconnected and will have a cycle.



So, to find the correct edge g and make sure that $T^* + e - g$ does not have a cycle, we first add e to T^*. $T^* + e$ has n edges so it must have a cycle, say C which has the edge e (recall that we proved any graph with n edges has a cycle). Since any cycle must cross any cut even number of times C must have another edge, call it g, such that $g \in (S, V - S)$. Now let $T := T^* + e - g$. We claim that T is a spanning tree. To check it is enough to show that T satisfies two of the following three properties of spanning trees (we said this without proof): (i) $n - 1$ edges, (ii) connected, (ii) acyclic. First since T^* has $n - 1$ edges and T has exactly $n - 1$ edges as well. Second, we show T is connected. This is because $T^* + e$ is connected and g is just an edge of the cycle C that we remove. So, after removing g the endpoints of g are still connected through the rest of C. So, $T^* + e - g$ is connected. This implies T is a spanning tree, but since $c(T) < c(T^*)$ we get a contradiction. ■