Quiz 1

Adam Karl

September 11, 2020

1 Proof by Exchange Argument

Note: for this problem I will be referring to the given algorithm as MAXLEAF.

- Assume (for later contradiction) that MAXLEAF is not correct
- thus, some input I exists such that MAXLEAF(I) is not an optimal solution
- let OPT(I) be an optimal solution that agrees with MAXLEAF(I) for the most number of steps. Here, "most number of steps" means includes the same vertices in S as MAXLEAF(I) when checking if each leaf is included from the maximum leaf value to the minimum leaf value.
- let u be the point where MAXLEAF(I) and OPT(I) first disagree (when looking at the leaves in descending order of value)
- due to the definition of MAXLEAF, we know that it always adds the leaf with the maximum value that keeps the solution feasible. Thus, MAXLEAF(I) must select the leaf at u for subset S, while
- let x be the leaf (at point u) that MAXLEAF(I) includes in S
- there are 2 options: either OPT(I) doesn't include any leaves after u in S or OPT(I) includes at least one leaf after u
- for the first case, OPT'(I) constructed of simply OPT(I) + leaf x is no worse of a solution than OPT(I) and is feasible since MAXLEAF(I) includes x and is feasible => OPT(I) both is and isn't an optimal solution that agrees with MAXLEAF(I) for the most number of steps => contradiction => our assumption that MAXLEAF isn't correct was wrong => MAXLEAF is a correct algorithm
- for the second case find leaf y on OPT(I) in the following manner
  - if leaf x has any direct siblings (children of x's parent node) that are selected in OPT(I)'s solution S AND are after u in the leaf ordering from highest profit to smallest profit, choose one of them arbitrarily as y
  - ELSE, if leaf x has any cousins (leaf nodes stemming from x's grandparent) in OPT(I)'s solution AND are after u, choose one of them arbitrarily as y
  - ELSE, if x's great-grandparent has any descendants that is in OPT(I)'s solution AND are after u, choose one of them arbitrarily as y
  - continue in this pattern (going up 1 more ancestor from x then looking for descendent leaves in OPT(I) and after u) until a leaf y is found. A leaf y MUST exist since in this case OPT(I) includes at least 1 leaf after u and we know MAXLEAF(I) includes x and is still a feasible solution
- now that we have selected y, since y comes after u (and leaf x appears at point u and we were traversing leaves in descending order when comparing MAXLEAF(I) to OPT(I)) we know $p_x \geq p_y$
- construct OPT'(I) such that it is exactly the same as OPT(I) except without y, but including x

- OPT'(I) is at least as good of a solution as OPT(I) since $p_x \geq p_y$ so the total profit of OPT'(I) is $\geq$ the total profit of OPT(I)
- for OPT(I) to be feasible, it must not exceed the capacity of any of the interior nodes
  - when x and y have the same parent, it's the easiest to see that swapping y out and x in to OPT(I) will not change how full any capacities are
  - when x and y are cousins (same grandparent), swapping y out and x in will keep the same number of leaves under the grandparent, then -1 leaf for y's parent (obviously no feasibility issue), and +1 for x's parent (possibly an issue). This is why we checked for y as a sibling to x FIRST before searching for cousins. Since no sibling to x was in OPT(I), and we know that MAXLEAF(I) was a feasible solution that included x (which OPT(I) didn't include), x's parent is guaranteed to have at least 1 excess capacity int OPT(I)
  - this pattern continues wtih grandparents and beyond, since we checked for closer related y's first, and MAXLEAF(I) is a feasible solution by definition, we have no feasibility issue since x's parent is guaranteed to have at least 1 excess capacity
- since OPT'(I) makes no changes before u, and now agrees with MAXLEAF(I) at u (since it includes x), OPT'(I) agrees with MAXLEAF(I) for more steps than OPT(I)

- thus, OPT'(I) is feasible, no worse of a solution than OPT(I), and agrees with MAXLEAF(I) for at least 1 more step than OPT(I)
- therefore, OPT(I) is NOT the optimal solution that agrees with MAXLEAF(I) for the most number of steps
- CONTRADICTION: we now have OPT(I) both is and is not the optimal solution that agrees with MAXLEAF(I) for the most number of steps
- since our only assumption was that MAXLEAF was not a correct algorithm, and we arrived at a contradiction after following logical steps, our assumption MUST be wrong
- thus, MAXLEAF is a correct algorithm