COMP3121-Ass4-Q3

z5302513, Kihwan Baek

August 2021

1. Solution

Before we solve the problem, we make a max-flow graph. Firstly, we let n-1 vertices be each squares and set their names as $S_1, S_2, ...S_{n-1}$. After that, we insert a super source and a super sink as S_0 and S_n each.

For every square i ($0 \leq i \leq n$), we connect the every square i with squares i+1, i+2,...i+k with a directed edge of infinite capacity and we set the total number of edges as $|E|$. And then, we set each vertices $S_1, S_2, ...S_{n-1}$ to have maximum capacity $A[1], A[2], ...A[n-1]$.

At this stage, we can use Edmonds-Karp algorithm because we have a super source and a super sink and our goal is getting the maximum number of children who can successfully complete the game. So, we need to maximize the flow from the super source S_0 to the super sink S_n and it takes O(|the number of vertices| $\times$ |the number of edges|^2) = O($(n+1) \times E^2$) = O(nE^2) time.

What we have to note is that if there is a square m S_m ($1 \leq m \leq n-1$), a child can jump from $S_{m_k}, ...S_{m-1}$ to S_m so, there will be 'k' directed edges to S_m. And also, S_m has the maximum capacity $A[m]$. Therefore, we need to split each squares into two vertices $s_{m,in}$ and $s_{m,out}$ and the sum of weight of edges coming into s goes into $s_{m,in}$ and $s_{m,out}$ is connected by an edge with the capacity $A[m]$ and it takes O(n) time. Hence, the total time complexity is O(nE^2) + O(n) = O(nE^2) time.