Let $OPT(j)$ denote the minimum cost of a solution on servers 1 through j, *given* that we place a copy of the file at server j. We want to search over the possible places to put the highest copy of the file before j; say in the optimal solution this at position i. Then the cost for all servers up to i is $OPT(i)$ (since we behave optimally up to i), and the cost for servers $i+1, \ldots, j$ is the sum of the access costs for $i+1$ through j, which is $0 + 1 + \cdots + (j-i-1) = \binom{j-i}{2}$. We also pay c_j to place the server at j.

In the optimal solution, we should choose the best of these solutions over all i. Thus we have

$$OPT(j) = c_j + \min_{0 \le i < j} (OPT(i) + \binom{j-i}{2}),$$

with the initializations $OPT(0) = 0$ and $\binom{1}{2} = 0$. The values of OPT can be built up in order of increasing j, in time $O(j)$ for iteration j, leading to a total running time of $O(n^2)$. The value we want is $OPT(n)$, and the configuration can be found by tracing back through the array of OPT values.

[1] ex25.372.49