Space-efficient perfect hash

Consider the two-level perfect hash tables presented in [CLRS] and discussed in class. As already discussed, for a given set of n keys from the universe U, a random universal hash function $h : U \rightarrow [m]$ is employed where $m = n$, thus creating n buckets of size $n_j \geq 0$, where $\sum_{j=0}^{n-1} nj = n$. Each bucket j uses a random universal hash function $h_j : U \rightarrow [m]$ with $m = n_j^2$. Key x is thus stored in position $h_j(x)$ of the table for bucket j, where $j = h(x)$. This problem asks to replace each such table by a bitvector of length $n = n_j^2$, initialized to all 0s, where key x is discarded and, in its place, a bit 1 is set in position $h_j(x)$ (a similar thing was proposed in Problem 4 and thus we can have a one-side error). Design a space-efficient implementation of this variation of perfect hash, using a couple of tips. First, it can be convenient to represent the value of the table size in unary (i.e., x zeroes followed by one for size x, so 000001 represents x = 5 and 1 represents x = 0). Second, it can be useful to employ a rank-select data structure that, given any bit vector B of b bits, uses additional o(b) bits to support in O(1) time the following operations on B:

- $rank_1(i)$: return the number of 1s appearing in the first i bits of B.
- $select_1(j)$: return the position i of the jth 1, if any, appearing in B (i.e. B[i] = 1 and $rank_1(i) = j$).

Operations $rank_0(i)$ and $select_0(j)$ can be defined in the same way as above. Also, note that o(b) stands for any asymptotic cost that is smaller than $\Theta(b)$ for $b \rightarrow \inf$.

SOLUTION