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 $h : U \to [m]$ 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 $h_j : U \to [m]$ 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 $b \to \inf$.

SOLUTION

The solution uses 4 data structure: Header [H], table to store a [A], table to store b [B], and the table in where we store the bit [T]. The table T concatenate the bit vector of each bucket n_j $(n_0, \dots, n_{m-1})$, notice the length of those bucket it can be different one of each other. We have indeed:

- $\sum_{j=1}^{m-1} n_j = n = m$
- In each bucket we use an universal hash function h_j with $m = n_j^2$, and we have already proved that it's perfect (no error) with probability $\geq \frac{1}{2}$
- $\sum_{j=1}^{m-1} n_j^2 = O(n)$ with probability $\geq \frac{1}{2}$

Now since we have concatenate all the buckets, we need to give a way to retrieve the range of the right bucket. To do so, we keep the length of each bucket in the header H. Therefore, we save the length in unary notation and we concatenate all of them. Then we have

$$H = (n_1^2)_1 \mid \dots \mid (n_{m-1}^2)_1$$
$$T = n_1 \mid \dots \mid n_{m-1}$$

Notice that the element in T are all binary, as explained in the text of the exercise, and $(\_)_1$ means the unary notation. Now, given a key x we are going to check the following:

$$j = h_{ab}(x)$$
$$k = select_1(j)\#\text{POSITION OF THE } j_{th}\ 1$$
$$t = rank_0(k)\#\text{NUMBER OF } 0\text{S BEFORE THE INDEX } k$$

Notice that t is the bucket in T select by h_{ab}, the external hash (note that we stored $\forall j\ .\ n_j^2\ 0s$ in H). Since we have this padding, we apply the correct hash function inside the bucket (using $A[j] = a'$ and $B[j] = b'$). Now, if the value in position $h_{a'b'}(x) + t$ of T is 0 the element is not present, instead if is 1 the element is present (with probability $\geq \frac{1}{2}$).
The space occupied by the data structures used here is the following:

- H: the number of $0s = \sum_{j=1}^{m-1} n_j^2 = O(n)$ and the number of $1s = \#\text{BUCKETS} = n = m$
- T: the number of element are $\sum_{j=1}^{m-1} n_j^2 = O(n)$

- A: here we have an array with $n = m$ elements, in which each $a \in [0,p] = Z_p$. Since, this a_s are used to build the hash used in the bucket, where the size of the bucket is n_j^2, then $p > n_j^2$ $((h_{ab} = (ax+b) mod\, p) mod\, n_j^2)$. Therefore, we take the first prime grater that n_j^2, that by Bertrand's postulate, is between $n_j^2 < p < 2n_j^2$. Therefore, each cell is going to use $log_2 2n_j^2 \leq 2log_2 n_j$. Now, we have $2\sum_{j=1}^{n-1} log_2(n_j) \leq 2\sum_{j=1}^{n-1} n_j = 2n$. Hence, we have $2n$ bits.
- B: here we have an array with $n = m$ elements, in which each $b \in [1,p] = Z_p^*$. By the same reasoning we have $2n$ bits.

Totally, the space used here is: $O(n) + n + O(n) + 2n + 2n = 5n + O(n)$