Przemysław Uznański May 16, 2017

Advanced Data Structures

Spring Semester 2017

Exercise Set 12

Recall from the lecture: for word length w, there is a datastructure for succesor/predecessor queries supporting $k = \mathcal{O}(w^{1/5})$ w-bit integers. Construction time: $\mathcal{O}(k^4)$ time, query time: $\mathcal{O}(1)$. (We used those as nodes of a B-tree, actual exponents do not matter as long as they are constant.)

Exercise 1:
Show dynamic version of fusion trees that supports $\mathcal{O}(\log n/\log w + \log w)$ amortized updates and the same worst-case time queries, and takes $\mathcal{O}(n)$ space.

Hint: Use indirection, having two levels:

- top level fusion tree,
- bottom level is composed of small size (polynomial in k) binary search trees.

With proper size of lower level trees, cost of explicitly rebuilding all nodes of fusion trees should amortize with number of updates need to happen to force update on top level.

Exercise 2: (Exponential trees)
Show dynamic version of fusion trees that supports $\mathcal{O}(\log n/\log w + \log\log n)$ amortized updates and the same worst-case time queries, and takes $\mathcal{O}(n)$ space.

Hint: Main ingredient is to have recursive construction, where tree is composed of:

- $\Theta(n^{1/5})$ separating values,
- $\Theta(n^{1/5})$ recursive structures, each using on $\Theta(n^{4/5})$ keys.

The separating values are stored in a static fusion tree. Useful for the analysis is the $\mathcal{O}(n^4)$ bound on cost of rebuilding of static fusion tree on n keys.