1.) Yes, that would be possible. Since you know how many subtrees there are per node, you would be able to compress it by assigning indexes for each node and subtree. Starting at 0 as the base root node, you would then assign incrementally one index higher to each subtree. For a node i, you would divide it by the order of 4 to find which level/subtree it is a part of.
2.) Theta(log n), which is the same as a binary min heap.
3.) I wouldn't expect either to be faster than the other, as insertion/removal for both implementations should still take O(log n) time in either case.
4.) Quaternary min heap, as you would need to traverse through fewer values when checking and modifying the location of different nodes based on their values. You can compare values much faster with a greater number of subtrees.