Project Deliverable 5

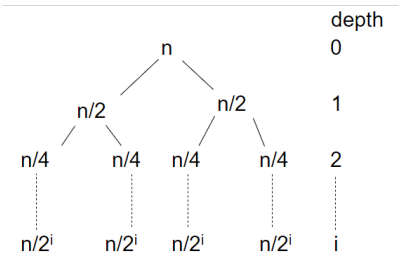
Alex Abrahamson - Mitchell Baker - Brock Ellefson - Yue Hou - Seth Severa

10 November 2017

Introspective sort is a sorting algorithm that combines the quick average time of Quicksort with the stable worst case performance of Heapsort developed by Dr. David Musser in 1997. Sorting algorithms often need to find a compromise between fast execution in the common case and a reliable performance when it comes to problematic inputs. We wanted to take this algorithm and compare its runtime to the runtime of the two algorithms it was constructed from, Quicksort and Heapsort.

We also have another goal with the project. David Musser mentions in his paper on sorting and selection algorithms the idea of using iterators rather than parring array indexes by reference. AS thi approach could speed up the algorithm, we are going to implement the use of iterators and then compare this new implementation to the older version of Introsort and its competitors. We are still in the progress of implementing this technique. Once implemented, we will compare the runtime of this iterative version of Introsort with with Quicksort and Heapsort again, and also compare it against the original implementation. Here is our progress in our runtime analysis.

Quicksort is a recursive sorting algorithm with a recurrence relation of $T(n) = 2T(n/2) + O(n)$, giving us the recurrence tree as shown below. It's average runtime is $O(nlogn)$, however, in the worst case of Quicksort, our subproblems do not increase efficiently as below, and the number of subproblems approaches n, giving us $O(n^2)$ time.



However in Introsort, we limit our depth to $2logn$, we know know that we will have at most 2^{2logn} subproblems, each being solved by the Heapsort in $nlogn$ time. Since we are at the level $2logn$, Heapsort will actually run in $(n/2^i)logn(2^i)$ time. This means that we have n^2 subproblems solved in $(1/n)log(1/n)$ time which simplifies to $nlog(1/n)$ which is $O(nlogn)$. So Introsorts runtime will always be $O(nlogn)$, regardless of the input.

Progress completed as of 11/10/17

- Preliminary time complexity analysis via Recursion Tree (seen above)
- Preliminary time complexity analysis via loop cost analysis
- Small scale actual run-time comparison of introsort, quicksort, and heapsort
- Research and early implementation of introsort using random-access iterators (pointers)

Tasks yet to be completed

- Further exhaustive proof by Recursion Tree
- Complete labled algorithm with loop cost analysis for best and worst case
- Large scale actual run-time and operation (distance, iterative, assignment) comparison between algorithms
- Optimized implementation of algorithm using iterators with comparison in run-time to standard implementation

References

[1] Musser, David. (1997). Introspective Sorting and Selection Algorithms. Software Practice and Experience. 27. . 10.1002/(SICI)1097-024X(199708)27:8<983::AID-SPE117>3.0.CO;2-#.