Lab 2 write-up

Jeremy Roy

Student no: 10092487

1. I’ve implemented the algorithm without memory. I noticed that that for the start number 100, the maximum number the algorithm traversed was 9232. For x=1000, the max number is 250504. Clearly a statically sized list of 0s as the problem suggests will be grossly inefficient on the memory front. I decided to give it a try anyways. I made an array of bools sized to 100 times the initial value x. Although I was able to find a solution for numbers 10 and 100, the program would segfault for numbers 1000 or greater. I was not able to figure out what was causing this segfault – I would imagine that I should be able to get 100kB of contiguous memory on the free store. Ah well.

After the contiguous array approach failing for large numbers, I decided to take a different approach: use the std::set container to store all values that led to the solution. The underlying structure of a std::set is a sorted tree. This would permit me to search for a previous value in time complexity (worst case). However, I every time I would insert a new value I would also loose time in sorting/re-balancing the tree. Unlike the array approach, the program didn’t segfault for x >= 1000.

1. The benchmark of the different approaches to implementing the Collatz program are presented below. Note that the times are presented in milliseconds.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Value of x** | **10** | **100** | **1000** | **10,000** | **100,000** |
| **No memory** | 0.0023 | 0.0256 | 0.4171 | 5.1925 | 81.1680 |
| **Array of bools** | 0.0080 | 0.147 | Segfault | Segfault | Segfault |
| **std::set memory** | 0.0545 | 0.4206 | 3.5468 | 35.4676 | 423.5190 |

The table above clearly shows that the fastest implementation of the Collatz program does not use memory.

1. Before a value of x can converge to zero, it must reach a number that is a power of two. Therefore, all terminating runs must pass through x=2. I was unable to determine any structure in the intermediate values of x other than the power-of-two end condition.