C. Warren Hammock

CS520

HW 4 Topic Paper

Horner Algorithm

Traditionally, to compute the value for *x* in the polynomial *f(x) = 5x4+7x3+3x2+2x+3*, you would get the value for *x* and multiply the coefficient by the variable raised to the power of the exponent. You would do this for every part of the problem separated by an addition symbol. You would add all those parts up to give you the answer. This was all until 1786 with the birth of William George Horner, theorem also known before him by Paolo Ruffini (1).

The number of multiplications needed for our original method is 5\*x\*x\*x\*x + 7\*x\*x\*x + 3\*x\*x + 2\*x + 3, which is 10 multiplications and 4 additions. Horner’s method ingeniously keeps factoring out x to cut down on the number of multiplications. This would leave the pervious problem looking like: x(x(x(5x+7)+3)+2)+3. This makes the number of multiplications 4, down from 10. The additions will not change. This method reminds me a lot of dynamic programming. Where the previous method used numerous amounts of multiplication to get an answer to be used in the next level of the problem, Horner’s method is using the previous stored calculation to answer to the next level.

Horner’s method works by moving from largest coefficient down. The problem will be solved from the left to the right. In the previous example, we will be starting at *5x4* and moving to *8*. We will store the value of *x = 3*. The algorithm starts with 0 and multiplies the first coefficient with the value of *x* which equals 15. We add that calculation with the next coefficient, 7, which gives us 22. We then multiply 27 by the value of x, which gives us 66. The number is added to the next coefficient, 3, which gives us 69. We then multiply 69 by the value of x giving us 207. The number is then added to the next coefficient, 2, giving us 209. That number is then multiplied by the value of x giving us 627. We then add the 3 giving us 630.

The benefits of the algorithm are obvious. Horner’s method cuts down the amount of multiplication actions by more than half of the previous convention method for figuring out polynomials. The asymptotic complexity for this algorithm is O(n). Horner’s algorithm will still need to compute n number of multiplications to solve the problem.

**REFERENCES**

Cargal, J. (1988). *Discrete Mathematics for Neophytes: Number Theory, Probability, Algorithms, and Other Stuff*.