Ashkan Hosseini
CS315
09/12/2012

The Algorithms

The multiplication algorithm -

This is the iterative multiplication algorithm discussed in class. This is a great algorithm when working with numbers in base binary because it is easy to implement. However it still displays complexity of $O(n^2)$:

> *Input: two numbers of n-bit digits (x, y)*
> *Output: the product of the input*
> *z = 0;*
> *for i = n-1 down to 0:*
>   *z = 2z*
>   *if y[i] = 1:*
>   *z = z+x*
> *return z*

The implementation of this algorithm is almost exactly the same to the algorithm itself. However it uses numbers stored in vectors that hold digits, so the only difference to the algorithm is we first make sure all of our inputs and outputs are of equal length before any computation is done so we don't get any out of bounds errors. It should be noted that this implementation makes use of the implementation of the addition algorithm in the first homework assignment for adding z to x. It also uses the below shift algorithm for 2z.

The shift algorithm –

This is used to compute the multiplication of a number by its base. For example if we are in base 2, this algorithm will multiply the number by 2. Unfortunately with the way we are storing our number we must use this algorithm which turns out to be $O(n)$. Theoretically we should be able to improve this to constant time since all we are really doing is appending a zero. However we must use this algorithm for now:

> *Input: a number of n-bit digits x*
> *Output: the input shifted left by one*
> *for i = 0 to n-1*
>   *x[i+1] = x[i]*
> *x[0] = 0*
> *return x*

For the implementation of the algorithm we must be careful to not hit out of bounds errors or override any previous or next values. To do this we append the last digit after the loop is complete to avoid out of bounds errors and we use two temporary placeholders for values while in the loop to avoid overriding.

Discussion of the Experimental Results

For this program we introduce a counter for all elementary operations in the hopes of understanding whether our theoretical bound holds true to experimental results. This experiment is as follows: create two random n-bit numbers, compute their product, and output the product and the number of elementary operations required for the computation. The results of the experimentation can be viewed in the Results of Experimentation section.

We expect the number of elementary operations to display $O(n^2)$ since that is our theoretical estimate for the complexity of the algorithm. In other words, the ratio $\frac{n}{\sqrt{c}}$ (where n is the number of digits and c is the number of elementary operations) should stay consistent as the number of digits increases. Looking at the Results of Experimentation section shows that ratio stays around 0.29. This indicates that the experimental results of the complexity of the algorithm agree with the theoretical estimate.

Results of Experimentation

n = 60

Number of operations:
42339

Ratio to theoretical bound:
0.292

Result:
378565053565120382726420204991644160

n = 120

Number of operations:
170834

Ratio to theoretical bound:
0.290

Result:
232288131414719079287748056155504933138972862662321416892863406244679225

n = 240

Number of operations:
666713

Ratio to theoretical bound:
0.294

Result:
269889371988186613966693850668869952490439479044248828526975374876245205425595401612
2836917866348035115149082561050048285320449491563066389450143

n = 480

Number of operations:
2741362

Ratio to theoretical bound:
0.290

Result:
2698004462610341679995842159720729092325503726172336693312058989569577440721411741586601794363013061279501233729446060420298732821875070492215611495555217967123672820009143888404319855432709552018403467414961737927524800086652449544869122300727455827905235729771705601274073869577055903356

n = 960

Number of operations:
10919210

Ratio to theoretical bound:
0.291

Result:
17882513437453700961153767342479526934941027346616794972003916266772928097151345155185383139128410625806308018928967182942843882867036269456687436744181183280783183674924314829997686247481291812084938766547140705963704849346192328798884429226895818864010568185662217413489564357860849681236063768022586682299987280869640144423875566139218779904328247124321415442199887701759343867825126132063566944545002676778729887942661582339900489540431007934402305529234758970251727675061448735864399359876117468076022614518301033991360635304790362566792057560658107445379438513057690557150

n = 1920

Number of operations:
43648578

Ratio to theoretical bound:
0.290

Result:
860582878220324250966182070734489378041093090453953378792194819264937384146994850039850680216143966215015546860877231938917445914603685317313893142504581786420683438962566668631643179651172821713411899284880411387291447246180498226925097592323017865143050136165851464226977752042163810512679466790448104884169050645949099889697728665646277447398285637569346847948838856705711607627166090284017171843969469212869466139870093088190089876444413482748460486046508995046798772478056505614240335584428047810667919073301597386921945408134459981186959056376587046638985397885402115682145241530061933440722271357531686564671972240753649250454897929195773102214886536371267475945533993399337275755113481736064987341259797346130570785775943796696266161059743259724816370142494945704918553232590762078433890135300929388681993385477983092615120949178061912399302937038904140834351803509544736782672592103114346401744240292339484983318972237809856087202410064301586164153911351119058079979026054406347275159549544014460212044482989332345514801600375528221040103712862847211025774871838174205522819731878356230926424839830198565918818189694989175601448871725640727276296