1 What Is The Overall Feeling of Writing a Thesis

I guess the overall feeling is one of abject terror. The feeling of "I must write or else!" It doesn't appear helpful. However, the feeling arises. It happens.

If I think about continuous self-improvement, then I must examine what happens in every moment, and never be disconnected from it. That seems extreme, especially for someone who has dissociated so often from the difficult present. Is it even tenable to suppose that you will do your best work if you stay present and aware, of your ass on the chair, of your fingers presssing the keyboard, of your mind thinking about Buddhism instead of your thesis?

A radical hypothesis.

2 The Plan

What I wanted to do today and the next few days was to find explicit polynomials (threshold functions) for the arithmetic functions which would lead to modular exponentiation of an exponentially large modulus. There are several things I realized.

One is that it must be modular multiplication, and it is not sufficient to just do multiple products with no modular reduction. The reason is that this would lead to different answers in different cases, which nevertheless have the same modular residue mod$m$, because modular residues are not unique. After all $\mathbb{Z}_m$ is the quotient group $\mathbb{Z}/m\mathbb{Z}$. The quantum amplitudes for all the different cosets with the same modular residue must combine, interfere, in order to get the correct result. After all, the initial register is a superposition of eigenstates for modular multiplication, and only by modular multiplication is one randomly selected by phase estimation.

The other insight is that the function EXPONENTIATION and modular reduction as given in the Siu-Bruck paper won't work for Shor's factoring algorithm because they assume a "small", i.e. polynomially bounded modulus $m < n^c$ for some $c > 0$. I may have had this insight yesterday as well.

Most of the results in this paper are that constant-depth threshold circuits for these functions exist, without giving explicit constructions. It may be hard to find such polynomials in a week, which is what I've allotted for myself. I need such circuits to find numerical constants bounding the polynomial circuit size of a constant-depth factoring algorithm. I can give myself a day of thinking about this subject before moving on. Let's say, Saturday.

> Fail softly

as a man once said, when he was trying to pass his quals exam.

So the overall plan is that I should look at POWERING, which is where you have a fixed base c, and you exponentiate it based on your input number X.

$$c^X \tag{1}$$

If c and X are n-bit numbers, then it seems like you could still do some kind of repeated or iterated squaring, even if you don't take the modular reduction at every step to keep the modular residue in a fixed size n-bit register.

EXPONENTIATION is when X is the base, and you raise it to some fixed power c, where X and c are again n-bit numbers. That seems misnamed, and in any case, it's not what we want to do.

$$X^c \tag{2}$$

After POWERING, which seems like a special case of MULTIPLE PRODUCT, we are left with an n^2-bit number. I think, by the argument in the previous skeleton thesis. Now it is time for DIVISION, to get a quotient. The quotient we MULTIPLY by the modulus m, or c in this case, which we can do in depth-4, since they are two $O(n)$-bit numbers. Then we SUBTRACT this from the result of POWERING to get the modular residue $b^{|X\rangle} \bmod m$, using two's complement representation for the number to subtract.

I think it is enough to get the total number of depth layers, and to describe the rationale behind the explicit polynomial functions for COMPARISON and ADDITION.

We can still get constants, numerical upper bounds, but the depth anyway. However, we would need the polynomials explicitly to even get asymptotic circuit size, simply to know how many terms we need (asymptotically). And therefore, probably also to bound circuit width, and circuit coherence. It would be interesting if we don't even need the explicit polynomial.

How would that even be the case? There was something about n^4 in the depth-efficient Siu-Bruck paper. Maybe that is enough.

3 How Would Multiplication Work

Let's start with multiple sum. For the addition of two n-bit numbers, we generate the carry bits in parallel, which are all functions of the n input bits $X = \{x_1, x_2, \ldots, x_n\}$. From Lemma 1 in the depth-efficient paper, any linear threshold function with polynomial weights can be implemented in a bounded depth circuit. However, if we have $n \times n$-bit numbers without the blocks, then each bit is dependent on at most n^2 bits. That is still polynomial. Why divide into blocks at all?

QUESTION.

4 How Would Division Work

I guess it's lucky that I spent a long time thinking about this stuff already. First to compute the quotient $z = x/y$, we compute y^{-1} up to some precision, let's say $\tilde{y}^{-1}$, then we multiply it by x to get z. How to get the inverse of y?

And is there some way to calculate modular inverse using a KSV style, parallel iteration of finite automata approach?

5 What is a Polynomial Threshold Function?

Seriously, I still don't know. I was supposed to devote an hour or so to editing or fact-checking pages from the past two days, but I didn't do it. It was too hard to look at them again, I guess, it was hard enough to write them in the first place.

I still suppose that it is a sum of polynomial many linear threshold functions, but how is that different from a single linear threshold function whose terms are simply the union of the terms in the original polynomial sum of linear threshold functions? Other than each linear threshold function has an easily expressible weight, or there is an easy way to partition terms into linear threshold functions.

Oh wait, no. Multinomials. I bet the terms are multinomials in a polynomial threshold function. And there are a polynomial number of them. Does that make any kind of sense? FACT-CHECK this.

6 The Real Timeline

Okay, as much as I would like to say realistically that I could write things up to a certain standard roughly in a few days. From past experience, if we learn from my fear, we know that I would probably still take a week after I thought I was done writing to really be happy with the typos and so forth.

Saturday can be figuring out explicit polynomials, or at least trying the whole day. I can lock myself away at Tracie's house in West Medford. Sunday can be writing my talk for Monday, which would just be a compilation of circuit stuff anyway.