Quantum Algorithms
Homework 2 Solutions

Patrick Canny

Due: 2019-02-05

1 Book Problems

1. Problem 2.1: Construct an Algorithm that determines whether a given set of boolean functions $\mathcal{A}$ constitutes a complete basis. (Functions are represented by tables.)

SOLUTION

We want to find all of the functions that can be represented by formulas from $\mathcal{A}$. For the algorithm, we will devise a set $\mathcal{F}$ of all constructed boolean functions from $\mathcal{A}$. To do this, two identity functions must be established and added to $\mathcal{F}$: $x(x,y) = x$ and $y(x,y) = y$.

From here, a general form for functions that must be added to $\mathcal{F}$ can be created:

$$f(a(x_1, x_2), b(x_3, x_4)...q(x_{2k-1}, x_{2k}))$$

given that $a, b, ...q \in \mathcal{F}, x_i \in \{x, y\}, and f \in \mathcal{F}]$.

This procedure of generating f from functions existing in $\mathcal{F}$ will be repeated until there cannot be another function generated. At this point, $\mathcal{F}$ will be the set of all boolean functions, implying that $\mathcal{A}$ is a complete basis, or it will not be.

2. Problem 2.9: Construct a circuit of size $\mathcal{O}(n)$ and depth $\mathcal{O}(logn)$ that tells whether two n-bit integers are equal, and if they are not, which one is greater.

SOLUTION

We went through a solution to this problem in class, so I will use that solution as a framework for my answer.

In this situation, with inputs x and y, one of three answers can occur:

- $x = y$
- $x < y$

- $y < x$

In order to come up with a sensible encoding, we will establish an equality bit and a comparison bit for the circuit output. The most significant bit will be used for equality, while the less significant bit will be used for comparison. If $y > x$, the comparison bit will be set. We can then encode these possibilities as:

- $x = y$: 10
- $x < y$: 01
- $y < x$: 00

The corresponding boolean circuit can then be built using a divide-and-conquer algorithm. For this to work, we will start with the base case, where the size of the input number is $n = 1$:

$$e = \neg(x \oplus y) \qquad c = (\neg x) \wedge y$$

The circuit complexity can be given by:

$$C_a(cmp_1) = 4 \qquad d_a(cmp_1) = 2$$

We can then extrapolate this circuit out to larger input sizes by reasoning about a general circuit. We will separate a size n input into it's upper and lower parts. These parts will then be fed into a general cmp circuit for comparing n/2 bits. The resultant comparison and equality bits can then be compared.

For the resulting equality bit, we will simply need to perform an AND operation to retrieve the final equality bit.

For the comparison bit, the work required is more involved. If the comparison bit for the upper half of the number is set, then we can simply use that bit since the upper bits are more significant. On the other hand, if it is not set we will need to compare the comparison bit from the lower half to the equality bit from the upper half. This deals with cases where the upper significant bits are the same, and the larger number is decided by comparison of the less significant bits.

From here, we can establish a general form for finding e_n and c_n:

$$e = e_u \wedge e_l \qquad c = c_u \vee (e_u \wedge c_l)$$

With general complexity:

$$C_a(cmp_n) = 2C_a(n/2) + 3 \qquad d_a(cmp_n) = d_a(n/2) + 2$$

The constants added onto the end reflect the change in depth or number of gates being added to the total circuit.

From here, we can recognize that the general form equations for C_a and d_a can be transformed into recurrence relations in order to compute the general size or depth of a circuit. It is actually possible to apply the Master Theorem, discussed in EECS 660, to solve these recurrences quickly.

First, we examine the size recurrance:

$$C_a(cmp_n) = 2C_a(n/2) + 3 \qquad C_a(cmp_1) = 4$$

By the master theorem:

$$a : 2 \quad b : 2 \quad f(n) : 3 \quad log_b(a) : 1 \quad f(n) \in \mathcal{O}(n^{1-\epsilon}) \rightarrow C_a(cmp_n) \in \theta(n)$$

Next, we can examine the depth recurrence:

$$d_a(cmp_n) = d_a(n/2) + 2 \qquad d_a(cmp_1) = 2$$

By the master theorem:

$$a : 1 \quad b : 2 \quad f(n) : 2 \quad log_b(a) : 0 \quad f(n) \in \theta(n^0) \rightarrow d_a(cmp_n) \in \theta(log(n))$$

As we have discovered, the size of the circuit is in $\theta(n)$, while the depth is in $\theta(log(n))$ as expected.

3. Problem 3.8: Prove that the predicate "x is the binary representation of a composite integer" belongs to NP.

SOLUTION

Claim 1.1. *"x is the binary representation of a composite integer" ∈ NP.*

Proof of claim. Assume that there is a "proof" that x is a composite integer. This "proof" could be a prime factorization of x. Converting x from a binary form to a base-ten form can be done in linear time in the length of the input. Then, the original "proof" can be checked in polynomial time by multiplying the primes together to form x.

This also holds true if the prime factorization is given in a binary format. The binary primes can be multiplied in polynomial time to form x.

Since this tactic shows that the proof for a given predicate can be computed in polynomial time, it proves that the predicate is in NP. •

2 Additional Problems

1. Solve **either** this problem or the two problems below.

   Give a careful and complete solution to problem 2.19.

SOLUTION

I opted to solve the 2 additional problems.

2. Find a complete Boolean basis $\mathcal{A}$ of size 1 (that is, it consists of just a single function). Prove your answer is correct.

SOLUTION

To find a complete Boolean basis of size one, we have to find a single function with which it is possible to build other functions of a complete Boolean basis.

Consider the known complete boolean basis

$$\{\vee, \wedge\}$$

For the proof, I will show that the truth table for $\vee$ and $\wedge$ can be created by building a circuit of NOR ($\downarrow$) gates.

Claim 2.1. *NOR ($\downarrow$) is a complete, single function boolean basis.*

Proof of claim. Consider the truth table for NOR:

a	b	a $\downarrow$ b
0	0	1
0	1	0
1	0	0
1	1	0

Next, consider the truth tables for $\vee$ and $\wedge$. They are as follows, respectively:

a	b	a $\vee$ b
0	0	0
0	1	1
1	0	1
1	1	1

a	b	a $\wedge$ b
0	0	0
0	1	0
1	0	0
1	1	1

A boolean circuit can be built for these two tables using only NOR $\downarrow$:

For $\wedge$: $(A \downarrow A) \downarrow (B \downarrow B) = (A \wedge B)$
With Truth Table:

a	b	$(a \downarrow a)$	$(b \downarrow b)$	$(a \downarrow a) \downarrow (b \downarrow b)$
0	0	1	1	0
0	1	1	0	0
1	0	0	1	0
1	1	0	0	1

For $\vee$: $(A \downarrow B) \downarrow (A \downarrow B) = (A \vee B)$
With Truth Table:

a	b	$(a \downarrow b)$	$(a \downarrow a) \downarrow (b \downarrow b)$
0	0	1	0
0	1	0	1
1	0	0	1
1	1	0	0

By the tables above, we can see that NOR forms a complete boolean basis. •

3. An n-ary function $f : \mathbb{B}^n \to \mathbb{B}$ is *idempotent* if

$$f(0, \ldots, 0) = 0 \qquad \text{and} \qquad f(1, \ldots, 1) = 1.$$

Find a basis $\mathcal{A}$ so that every idempotent Boolean function is representable as a circuit over $\mathcal{A}$. Prove your answer is correct. [*Hint 1: Post's Lattice.*] [*Hint 2:* ? :.]

SOLUTION

The basis { ? : } is complete over idempotent Boolean functions.

Claim 2.2. *The basis* $\{\ ?:\ \}$ *is complete over the idempotent functions.*

Proof of claim. The claim can be proven using an inductive approach. First, we establish a base case. The most simple idempotent functions are of one variable, i.e. $f : \mathbb{B}^1 \to \mathbb{B}$. The base case then becomes:

$$f(0) = 0 = 0?:0 \qquad f(1) = 1 = 1?:1$$

From here, we establish the inductive hypothesis:

$$\{f : \mathbb{B}^n \to \mathbb{B} | f \text{ is idempotent}\}$$

can be created using the basis $\{\ ?:\ \}$.

We seek to show that this claim holds for $f : \mathbb{B}^{n+1} \to \mathbb{B}$.
Consider

$$f(0_1, \ldots, 0_n, 0_{n+1}) = 0 \qquad f(1_1, \ldots, 1_n, 1_{n+1}) = 1$$

By the inductive hypothesis, the functions

$$f(0_1, \ldots, 0_n) = 0 \qquad f(1_1, \ldots, 1_n) = 1$$

hold. In order to build a circuit for the $n+1$ case, the result of the n case must be compared to the new input value in the following way:

$$f(0_1, \ldots, 0_n)?:0 \qquad \text{and} \qquad f(1_1, \ldots, 1_n)?:1$$

which results in the conditionals

$$0?:0 = 0 \qquad \text{and} \qquad 1?:1 = 1$$

which align with the base case of the induction. Therefore, ? : is a basis for the idempotent boolean functions.

•