CS3102 Theory of Computation

Warm up:

We showed that NAND gates are “Universal”. We also showed that not all things are “Computable”. How can both things be true?

Logistics

- Midterm on Thursday in class
  - Review session tomorrow evening
    - 6:30pm – 8:00pm
    - Thornton E316

Last Time

- Complexity
  - SIZE
  - Complexity Classes
  - Big-Oh

How many gates are required?

- TCS Theorem 5.3: There is a constant $\delta > 0$, such that for every sufficiently large n there is a function $f: \{0,1\}^n \to \{0,1\}$ such that $f \notin SIZE\left(\frac{\delta 2^n}{n}\right)$. That is, the shortest NAND program to compute f requires at least $\delta \cdot \frac{2^n}{n}$ gates.

How to show this

1. Count the number of n- input functions
2. Count the number of programs of size $\delta \cdot \frac{2^n}{n}$
3. Show there are more functions than programs

How many functions?

- How many functions are there of form $\{0,1\}^n \to \{0,1\}$?
- How can we count this?

How many programs?

- Bits required for an s-line program:
  - At most $3s$ variables (3 variables mentioned for each of the s lines)
    - $\log_2 3s$ bits per variable
  - 3 variables per line
    - $3 \cdot \log_2 3s$ bits per line
  - s lines total
    - $3s \log_2 3s$ bits total
- Upper bound on the number of s-line programs:
  - $2^{3s \log_2 3s}$
  - $2^{O(s \log s)}$

Fixing the Length

- If we fix the length of the programs to be $\delta \cdot \frac{2^n}{n}$ lines, how many programs are there?
- $2^{c \cdot s \log s}$ programs of length s
- $2^{\frac{c\delta 2^n}{n} \log s}$programs
- Let $\delta = \frac{1}{c}$
- $2^{\frac{2^n}{n} \log s} < 2^{2^n}$
- Some programs require more than $\delta \cdot \frac{2^n}{n}$ lines

64 bit machine

- I want to make $EVAL$ to evaluate any program for a function $f: \{0,1\}^{64} \to \{0,1\}$. How many gates do I need?
- Some functions will require at least $\delta \cdot \frac{2^n}{n}$ gates.
  - Assume $\delta = \frac{1}{10}$
- We must evaluate programs longer than: $\frac{2^{64}}{640}$ lines
- We need at least $\left(\frac{2^{64}}{640}\right)^2 \log_2\left(\frac{2^{64}}{640}\right)$ gates
  - 4.5×10^{34} gates
  - Your computer would need to be the area of the solar system

Conclusion

- A domain of 2^{64} is large enough that perhaps it’s not useful to think of the function as finite
- Let’s think of that as an infinite function instead
- We need a model of computing for infinite functions

After the exam

- A model of computing for infinite functions
- How to do simple operations over and over again to compute
  - Real computers update memory by computing “simple” functions in hardware over and over again

Major topics

- Representing things as strings
- Computing requires finite representations
  - There are more functions than finite representations of things, so some functions aren't computable!
- Boolean gates/programs as a model of computing
  - Computing with logic!
- Simple components can build complicated behavior
  - With just NAND we can do complex functions
    - ANY finite function, actually
    - Including evaluating programs
- With a model of computing we can measure efficiency of computing
  - Allows us to categoriz functions by difficulty

Is there a baby in the picture?

Computing the "Baby" function

First: Represent pictures as strings (in binary)

- Assumption: all our pictures will be scaled to be the same size

Computing the "Baby" function

Second: Define the function

$$BABY: \{0,1\}^k \rightarrow \{0,1\}$$

$$BABY(p) = \begin{cases} 1 & \text{if } p \text{ has a baby in it} \\ 0 & \text{otherwise} \end{cases}$$

Computing the “Baby” function

Third: Build a NAND-circuit/program for the function

How can we tell if that’s possible?