CS-E4530 Computational Complexity Theory, Spring 2020
Problem Set 3 **Homework Due Tue 4 Feb, 23:59**

Instructions

- Classroom Problems C3.1–C3.3 will be discussed and solved at the tutorial session on Wed 29 Jan, 14–16, Room T4 (A238). No credit is given for these problems.
- Homework Problems H3.1–H3.3 you should solve on your own, and submit your solutions via the MyCourses interface by the deadline of Tue 4 Feb, 23:59, These problems will be individually graded on a scale of 0–2 points per problem.
- In preparing your solutions to the Homework Problems:
  1. Justify your solutions, be precise, and provide sufficient detail so that it is easy to follow your reasoning.
  2. Submit your solutions as an easily readable, single pdf file, which is either typeset or written in full sentences and clean handwriting.
  3. **[Code of Conduct]** You can discuss the problems with your colleagues and the course's teaching staff, but you must write the presentations of your solutions *independently* and *individually*, without any notes from such discussions.

Classroom Problems

C3.1

(i) Design a Boolean circuit using '$\wedge$', '$\vee$' and '$\neg$' gates that implements addition for two single-bit integers. That is, describe a circuit with 2 inputs and 2 outputs that on input (x, y) outputs $z = (z_1, z_0)$, where z is the binary representation of the sum $x + y$. (Thus z_0 is the "sum bit" and z_1 is the "carry bit".) This elementary circuit is called a *half adder*.

(ii) Design a two-bit *full adder* circuit, viz. a circuit that on input (x, y, c) outputs $z = (c', z)$, where z is the binary representation of the sum $x + y + c$. (Here c and c' are the "incoming" and "outgoing" carry bits.)

(iii) Based on the designs from items (i) and (ii), present a general circuit scheme for the addition of two n-bit integers. What is the size and depth of your circuits, as a function of n? (The *size* $S(C)$ of a circuit C is the number of gates in it, and its *depth* $D(C)$ is the length of the longest path from an input gate to an output gate.)

C3.2 Present a general circuit scheme for the addition of two n-bit integers in depth $\mathcal{O}(\log n)$. (*Hint:* Recursive divide-and-conquer by halves. Precompute the sums of the high-order bits in two versions — with the low-order carry bit either 0 or 1 — and then select the correct versions when the actual low-order carry bit is known.) What is the size of your circuits in this design?

C3.3 Prove that if $\mathbf{P} = \mathbf{NP}$, then $\mathbf{EXP} = \mathbf{NEXP}$. (*Hint:* Unary notation.)

Homework Problems

H3.1 Following the design ideas of Classroom Problem C3.2, present a general circuit scheme for computing the function $f(x, y)$, where x and y are n-bit binary numbers, and $f(x, y) = 1$ if and only if the numerical value of x is strictly greater than the numerical value of y. Your circuits should have depth $\mathcal{O}(\log n)$. (*Hint:* Consider first the slightly more general $2n$-bit input/2-bit output function $g(x, y) = (a, b)$, where $a = 1$ iff $\text{val}(x) > \text{val}(y)$ and $b = 1$ iff $\text{val}(y) > \text{val}(x)$. Design a baseline circuit for g in the case that both x and y are just single bits. Then use recursion.) (2 points)

H3.2

(i) How many different Boolean functions on n variables, i.e. mappings $f : \{0, 1\}^n \to \{0, 1\}$, are there?

(ii) Give some reasonable upper bound on the number of n-variable Boolean functions computable by circuits with m gates. *Hint:* Consider each gate g, $1 \leq g \leq m$, in a circuit of size m to be of one of the following types:

  (a) $g : (x_j)$, where $1 \leq j \leq n$, or $g : (\mathbf{0})$, or $g : (\mathbf{1})$.
  (b) $g : (\neg, h)$, where $1 \leq h < g$.
  (c) $g : (\wedge, h, h')$, where $1 \leq h, h' < g$.
  (d) $g : (\vee, h, h')$, where $1 \leq h, h' < g$.

(iii) Prove that for large enough n, there exist Boolean functions on n variables that cannot be computed by circuits with $m = \frac{2^n}{2n}$ (or fewer) gates. Verify that, in fact, the fraction of functions with such small circuits goes to zero as n increases. (2 points)

H3.3 Consider the problem 2SAT, where the instance is a Boolean formula ϕ in 2cnf, i.e. in conjunctive normal form with exactly two literals per clause, and the question is whether ϕ is satisfiable or not. Show that $\text{2SAT} \in \mathbf{P}$, i.e. design an algorithm for deciding the satisfiability of 2cnf formulas in polynomial time. In addition to presenting your algorithm, explain why it (a) gives the correct answer and (b) runs in polynomial time.

(*Hint:* Let the variables appearing in ϕ be $X = \{x_1, \ldots, x_n\}$. Construct a directed graph G_ϕ, whose vertices are all the $2n$ literals over X, and for every clause $(\alpha \vee \beta)$ in ϕ, there are directed edges $\neg\alpha \to \beta$ and $\neg\beta \to \alpha$ in G_ϕ. Formulate the property of satisfiability of ϕ in terms of the properties of G_ϕ.) (2 points)