Algorithms – CS-27200/37000 Homework – January 7, 2005
Instructor: László Babai Ry-164 e-mail: `laci@cs.uchicago.edu`

ADVICE. Take advantage of the TAs' office hours on Tuesday, Wednesday, and Thursday, 5–6pm and Saturday 11am–12noon in Ryerson 255.

READING (due next class) Handouts: dynamic programming, knapsack problem. Review PSEUDOCODE conventions used in the handouts. Review all previous handouts, including "Asymptotic notation."

HOMEWORK. Please **print your name on each sheet.** Print "U" next to your name if you seek 27200 credit and "G" if you seek 37000 credit, regardless of your grad/undergrad status. Please try to make your solutions readable. Unless expressly stated otherwise, all solutions are due at the **beginning of the next class.**

Problems labeled "(U,G)" are assigned to the entire class. Problems labeled "G only" are for 37000 credit. Those who seek 27200 credit may solve the "G only" problems for bonus points.

Do not write your solutions to "G only" problems on the same sheet as the solutions to (U,G) problems. **Homework is collected in three separate piles (U, G, "G only").** Please put your solutions to "G only" problems on that pile, and your solutions to other problems on the "U" or "G" pile according to the credit you seek.

1.1 (U, G) (a) (2 points) Let $\{x_n\}$ be a sequence of real numbers. Define, using ϵ and a threshold value n_0, the statement that $\lim_{n\to\infty} x_n = 1$. Your answer should be a well-quantified, properly formed formula involving no English words. Warning: the order of the quantifiers is essential.
(b) (G only) (4 points) Let a_n, b_n, c_n, d_n be sequences of *positive* real numbers. Prove, using the definition from (a): if $a_n \sim c_n$ and $b_n \sim d_n$ then $a_n + b_n \sim c_n + d_n$. Elegance counts.

1.2 (U,G) Let x and b be positive integers, $b \geq 2$. Writing x in base b means expressing x (uniquely) as $x = \sum_{i=0}^{k} a_i b^i$ where $0 \leq a_i \leq b-1$; the a_i are the base-b digits of x. Base-2 digits are "bits." We say that x has $\leq k+1$ digits; it has exactly $k+1$ digits if $a_k \neq 0$.

(a) Determine the smallest and the largest integer which has (exactly) n digits to the base b.

(b) Let $\ell_b(x)$ denote the (exact) number of digits of the positive integer x in base b. Prove: as $x \to \infty$, we have the asymptotic relation $\ell_b(x) \sim \log_b(x)$.

(c) (Binary-decimal conversion) (2 points) Prove: $\ell_2(x) \sim c\ell_{10}(x)$. Determine the constant c.

1.3 (U,G) In this problem, "digits" are decimal, "bits" are binary. Suppose Alice holds an n-digit integer, X, and Bob holds an n-digit integer, Y (in decimal). (Initial zeros are permitted, so strictly speaking these integers have $\leq n$ digits.) p is an ℓ-digit prime number known to both Alice and Bob. The i-th digit of X is $X[i]$ and the i-th digit of Y is $Y[i]$. Alice and Bob each are infinitely powerful computers. They communicate with each other in order to determine (i) whether $X \equiv Y \pmod{p}$; and if the answer is NO then (ii) find some i such that $X[i] \neq Y[i]$. Their communication is a string of bits. For concreteness, let $n = 10^{15}$ and $\ell = 300$ (as in class). Prove:

(a) (2 points) Prove: the first task can be accomplished using less than 1000 bits of communication.

(b) (12 points) Prove: the second task can be accomplished using less than 50,000 bits of communication. Design a **deterministic** algorithm (communication protocol) (no coin flips permitted). Express your algorithm in **pseudocode**. Your algorithm should **not** be recursive (it should not call itself on smaller instances of the problem). Prove the stated complexity bound for your algorithm.

(c) (4 points) Improve the bound to 42,000 bits. (Hint. Make only a slight modification to your algorithm for (b).)

1.4 (G only) Let $\pi(x)$ denote the number of primes $\leq x$. The Prime Number Theorem asserts that $\pi(x) \sim x/\ln x$. Let $P(x)$ denote the product of all primes $\leq x$; for instance, $P(10) = 210$. Use the Prime Number Theorem to prove that $\ln P(x) \sim x$. Proceed in two parts: (a) (2 points) $\ln P(x) \lesssim x$; and (b) (8 points) $\ln P(x) \gtrsim x$.

1.5 (G only) (6 points) Let $\nu(x)$ denote the number of *distinct* prime divisors of the positive integer x. (For instance, $\nu(100) = 2$.) Prove: $\nu(x) \lesssim \ln x/\ln\ln x$.

1.6 (G only) (8 points) Suppose Alice and Bob each hold an n-digit integer, X and Y, respectively. Let $\epsilon > 0$ be given. Prove: the Rabin–Simon–Yao protocol (discussed in class) will succeed with probability $\geq 1 - \epsilon$ using a random prime number p with $O(\log n)$ digits. (Note: in class we used $n = 10^{15}$.)

1.7 (U, G) (12 points, due Wednesday, January 12) *(All-ones square problem.)* Given an $n \times n$ array A of zeros and ones, find the maximum size of a contiguous square of all ones. (You do not need to locate such a largest all-ones square, just determine its size.) Solve this problem in *linear time.* "Linear time" means the number of steps must be O(size of the input). In the present problem, the size of the input is $O(n^2)$. Manipulaing integers between 0 and n counts as one step; such manipulation includes copying, incrementing, addition and subtraction, looking up an entry in an $n \times n$ array.

Describe your solution in **pseudocode.** The solution should be *very simple,* no more than a few lines. **Elegance counts.** *Hint:* dynamic programming. Example:

$$\begin{array}{|cccccc|}
\hline
1 & 0 & 1 & 1 & 0 & 1\\
1 & 1 & 0 & 1 & 1 & 1\\
1 & 0 & 1 & 1 & 1 & 1\\
1 & 1 & 1 & 1 & 1 & 1\\
1 & 1 & 1 & 1 & 1 & 0\\
1 & 1 & 1 & 0 & 1 & 1\\
\hline
\end{array}$$

In this example, the answer is 3. There are three contiguous 3×3 square subarrays of all-ones. One is indicated below by $\underline{\text{underlines}}$, another is shown in a $\boxed{\text{box,}}$ the third one is indicated by *Italics*.

$$\begin{array}{|cccccc|}
\hline
1 & 0 & 1 & 1 & 0 & 1\\
1 & 1 & 0 & \textit{1} & \textit{1} & \textit{1}\\
\cline{3-5}
1 & 0 & \multicolumn{1}{|c}{1} & \textit{1} & \multicolumn{1}{c|}{\textit{1}} & \textit{1}\\
\underline{1} & \underline{1} & \multicolumn{1}{|c}{\underline{1}} & \textit{1} & \multicolumn{1}{c|}{\textit{1}} & \textit{1}\\
\underline{1} & \underline{1} & \multicolumn{1}{|c}{\underline{1}} & 1 & \multicolumn{1}{c|}{1} & 0\\
\cline{3-5}
\underline{1} & \underline{1} & \underline{1} & 0 & 1 & 1\\
\hline
\end{array}$$

1.8 (G only) Consider the Knapsack problem described in the last handout. The input parameters in that problem are positive reals called *weights* and *values*; and a "weight limit" W is given. It is shown in the handout how one can solve this problem in $O(nW)$ steps (arithmetic, comparison, bookkeeping) if all *weights* (including W) are *integers.*

(a) (15 points, due Wednesday, January 12) Assume now that all *values* are *integers* (but the weights are real). Let V denote the sum of the values. Solve the knapsack problem in $O(nV)$ steps under this assumption. Your solution should be a simple **pseudocode**.

(b) (15 points, due Wednesday, January 19) *(Efficient approximation algorithm.)* Assume both the weights and the values are real and assume an error parameter ϵ is given ($0 < \epsilon < 1/2$). Find the maximum knapsack value within $\pm\epsilon V$ error. (Your solution must satisfy the weight constraint exactly, not approximately; the error is in comparison with the (unknown) optimum solution.) *Complexity:* Your algorithm should use $O(n^2/\epsilon)$ steps. The steps include bookkeeping, arithmetic and comparison of $O(\log(n/\epsilon))$-digit integers, arithmetic of reals, and rounding reals to the nearest integer.