Solutions of problem 3, problem 6 of Problem Set 2

October 5, 2020

Problem 1. We find the *greatest common divisor (gcd)* of two positive integers, using divide-and-conquer.

a. Show that the following rule is true

$$\mathsf{gcd}(a, b) = \begin{cases} 2\mathsf{gcd}(a/2, b/2) & \text{if } a, b \text{ are even} \\ \mathsf{gcd}(a, b/2) & \text{if } a \text{ is odd, and } b \text{ is even} \\ \mathsf{gcd}((a-b)/2, b) & \text{if } a, b \text{ are odd} \end{cases}$$

b. Using part a, or otherwise, give an efficient divide-and-conquer algorithm for greatest common divisor.

c. Compare the efficiency of your algorithm to Euclid's algorithm if a and b are n-bit integers.

Solution 1. a. Let $k = \mathsf{gcd}(a, b)$. Then we have $a = ka'$, and $b = kb'$, where k, a', b' are positive integers, $k > 1$, and a' and b' are co-prime. We consider all the cases:

(a) If a, b are even.
The gcd k must be even in this case. We can write, $a/2 = (k/2).a'$ and $b/2 = (k/2).b'$. Since a' and b' are co-prime and $k/2$ is an integer, it is clearly the gcd of $a/2, b/2$. Hence $\mathsf{gcd}(a, b) = 2\mathsf{gcd}(a/2, b/2)$.

(b) If a is odd and b is even.
The gcd k must be odd here since 2 is not a common factor. Hence b' must be even since b is even. We can write $a = ka'$ and $b/2 = k(b'/2)$. Since $a', b'/2$ are co-prime, k is the gcd of $(a, b/2)$.

(c) If a, b are odd. (Assume $a \geqslant b$ w.l.o.g)
The gcd k, and a', b' must be odd here. Also, if a', b' are co-prime, then so must

be $a' - b', b'$, for if not they would have a common factor k' that is not 1. Then we would have $b' = k'b''$ and $a' - b' = k'c'$ for some positive integers b'', c', which gives $a' = k'(b'' + c')$. This would imply that k' is a non trivial common factor to a', b', contradicting that a', b' are co-prime.

We can write $(a - b)/2 = k(a' - b')/2$, and $b = kb'$. $(a' - b')/2$ is an integer since a', b' are odd, and $(a' - b')/2$ and b are co-prime as observed, hence k is the gcd of $(a - b)/2, b$.

b. The algorithm is as follows:

gcd(a, b)
swap (a, b) if $a < b$;
if $b == 0$ **then**
| return a;
end
else if a, b *are both even* **then**
| return($2\mathtt{gcd}(a/2, b/2)$);
end
else if a *is even,* b *is odd* **then**
| return($\mathtt{gcd}(a/2, b)$);
end
else if a *is odd,* b *is even* **then**
| return($\mathtt{gcd}(a, b/2)$);
end
else if a*,* b *are odd* **then**
| return($\mathtt{gcd}((a - b)/2, b)$);
end

c. Let a, b be m,n bits each respectively. Let $T(m+n)$ denote the time taken by algorithm to compute gcd of m,n bit numbers. In every case, at least one of either a or b is reduced by at least half, which means at least one bit is reduced in either a, or b (or both). Therefore we have:

$$T(m + n) \leqslant T(m + n - 1) + cm + dn$$

(where c and d are constants). This can easily be evaluated as:

$$T(m + n) \leqslant T(m + n - 2) + 2cn + 2dm$$

$$T(m+n) \leqslant T(m+n-3) + 3cn + 3dm$$
$$\vdots$$
$$T(m+n) \leqslant T(1) + (m+n-1)cn + (m+n-1)dn$$

The running time is therefore upper bounded by $\mathcal{O}(max(m^2, n^2))$. (Skipping a formal induction proof since the recurrence is very simple). This is the same upper bound as we get in Euclid's algorithm.

Problem 2. A *positive sequence* is a finite sequence of positive integers. *Sum of a sequence* is the sum of all the elements in the sequence. We say that a sequence A *can be embedded into another sequence* B, if there exists a strictly increasing function $\phi : \{1, 2, \ldots, |A|\} \to \{1, 2, \ldots, |B|\}$, such that $\forall i \in \{1, 2, \ldots, |A|\}, A[i] \leqslant B[\phi(i)]$, where $|S|$ denotes the length of the sequence S.

Given a positive integer n, construct a positive sequence U with sum $\mathcal{O}(n \log n)$, such that all the positive sequences with sum n, can be embedded into U.

Solution 2. We define a function $\mathsf{Seq}(n)$ that recursively computes the solution sequence as follows:

$\mathsf{Seq}(n)$
if $n == 1$ **then**
| return($[1, 1]$);
end
$S = \mathsf{Seq}(n/2)$;
Return($S.n.S$); (. is concatenation)

Proof of correctness can be done using induction. Clearly for $n = 1$, $[1, 1]$ is a trivial solution. Assume the function works correctly for $n/2$.

By induction hypothesis, the sum of sequence S is upper bounded by $c(n/2) \log n/2$, for some constant c. Then the sequence constructed for input n has sum bounded by $c(n/2) \log n/2 + n + c(n/2) \log n/2$, which is $\leqslant cn(\log n/2 + 1)$, and hence bounded by $cn \log n$ (the base for log is assumed to be 2).

To see that any sequence A of sum n can be embedded in $S.n.S$, we observe that in any sequence A of sum n, there exists an index $1 \leqslant i < |A|$, such that $A[1] + \ldots A[i-1] \leqslant n/2$ and $A[1] + \ldots A[i] \geqslant n/2$, which would imply that $A[i+1] \ldots A[|A|] \leqslant n/2$. (We abuse notation and in the case when the index i is 1 or $|A|$, $A[1] \ldots A[i-1]$ and $A[i+1] \ldots A[|A|]$ respectively denote empty sets). Then using induction hypothesis, we embed the first part, $A[1] \ldots A[i]$ in S, map $A[i]$ to n, and embed the latter part $A[i+1] \ldots A[|A|]$ in S.

Hence we can embed $|A|$ in $S.n.S$ and the sum of $S.n.S$ is bounded by $c \log n$.