Albert-Ludwigs-Universität, Inst. für Informatik
Prof. Dr. Fabian Kuhn
M. Ahmadi, O. Saukh, A. R. Molla November 11, 2015

Algorithm Theory, Winter Term 2015/16
Problem Set 3 - Sample Solution

Exercise 1: Greedy Algorithm (6 points)

In the following, a *unit fraction* is a fraction where the numerator is 1 and the denominator is some integer larger than 1. For example 1/4 or 1/384 are unit fractions.

It is well-known that every rational number $0 < q < 1$ can be expressed as a sum of pairwise distinct unit fractions, e.g., we can write $\frac{4}{13}$ as

$$\frac{4}{13} = \frac{1}{5} + \frac{1}{13} + \frac{1}{32} + \frac{1}{65}.$$

Interestingly such a decomposition into distinct unit fractions can be computed using a simple greedy algorithm.

In the following, assume that you are given two positive integers a and b such that $b > a$. Design a greedy algorithm to compute integers $0 < c_1 < c_2 < \cdots < c_k$ such that

$$\frac{a}{b} = \frac{1}{c_1} + \frac{1}{c_2} + \cdots + \frac{1}{c_k}.$$

Prove that your greedy algorithm always works and that it decomposes $\frac{a}{b}$ into at most a unit fractions.

You can assume that your algorithm can deal with arbitrarily large integer numbers. Note that for the fraction $\frac{4}{13}$, the standard greedy algorithm computes a decomposition which is different from the one given above.

Solution

Algorithm Description:

Here we introduce a greedy algorithm which solves the problem in at most a steps for any input fraction a/b. In the i^{th} step, the algorithm uses the remaining fraction from the previous step and outputs c_i and an irreducible remaining fraction a_i/b_i. For consistency, in the first step the algorithm considers the irreducible fraction equals to the input fraction given to the algorithm (i.e., a/b) as the remaining fraction of the previous step. The algorithm continues until the remaining fraction becomes a unit fraction.

To calculate c_i and a_i/b_i in each step, if the remaining fraction from the previous step is a unit fraction then the algorithm simply considers $c_i := b_{i-1}$ and stops. Otherwise, the algorithm calculates the largest unit fraction smaller than the remaining fraction from the previous step. Then, it considers the denominator of the calculated fraction as c_i. Therefore, we have

$$\frac{1}{c_i} < \frac{a_i}{b_i} < \frac{1}{c_i - 1}.$$

And it calculates the remaining fraction as follows:

$$\frac{a_i}{b_i} := \frac{a_{i-1}}{b_{i-1}} - \frac{1}{c_i},$$

where c_i is a positive integer. Note that in each step the algorithm calculates the remaining fraction, if the numerator and denominator are not co-prime, it replaces the fraction by its equal irreducible fraction.

Example:

Let us go through an example to see the algorithm more clearly. Consider the example in the exercise, $a/b = 4/13$.

1^{st} step, $a/b = 4/13$: The largest unit fraction smaller than $4/13$ is $1/4$. Then,

$$c_1 = 4, a_1/b_1 = 4/13 - 1/4 = 3/52.$$

2^{nd} step, $a_1/b_1 = 3/52$: The largest unit fraction smaller than $3/52$ is $1/18$. Then,

$$c_2 = 18, a_2/b_2 = 3/52 - 1/18 = 1/468.$$

Since the remaining fraction is a unit fraction, $c_3 = 1/468$, the algorithm stops and decomposition into sum of unit fractions is complete:

$$\frac{4}{13} = \frac{1}{4} + \frac{1}{18} + \frac{1}{468}$$

Analysis:

Now, let us show that the above proposed algorithm always stops in a finite number of steps (at most a steps) and works properly. To do this, we mention two claims.

Claim 1. In each step, the numerator of the remaining fraction is positive and strictly smaller than the numerator of the remaining fraction from previous step.

Proof. Fix some step with input fraction of x/y and the generated unit fraction of $1/c$. Then the remaining fraction is $\frac{xc-y}{yc}$. To prove the claim we should show that $xc - y$ is positive and strictly smaller than x. We assumed that $1/c$ is smaller than x/y and also it is the largest possible unit fraction. Based on the algorithm description we have

$$\frac{1}{c} < \frac{x}{y} \Rightarrow xc - y > 0 \quad (1)$$

$$\frac{1}{c-1} > \frac{x}{y} \Rightarrow xc - y < x \quad (2)$$

(1) and (2) prove claim 1. □

Claim 2. If the algorithm stops after $t \leq a$ steps there does not exist $i \neq j \in \{1, \ldots, t\}$ such that $c_i = c_j$.

Proof. Fix some step r during the execution and let us assume that for round $r-1$ the remaining fraction is a_{r-1}/b_{r-1} and the calculated integer is c_{r-1}. Then the remaining fraction for round r is

$$\frac{a_r}{b_r} = \frac{a_{r-1}}{b_{r-1}} - \frac{1}{c_{r-1}} = \frac{a_{r-1}c - b_{r-1}}{b_{r-1}c_{r-1}}.$$

Since $1/c_{r-1}$ is the largest unit fraction smaller than a_{r-1}/b_{r-1} we have

$$\begin{aligned}
\frac{1}{c_{r-1}-1} > \frac{a_{r-1}}{b_{r-1}} &\Rightarrow a_{r-1}c_{r-1} - a_{r-1} < b_{r-1} \\
&\Rightarrow a_{r-1}c_{r-1} < a_{r-1} + b_{r-1} < 2b_{r-1} \\
&\Rightarrow \frac{a_{r-1}}{b_{r-1}} - \frac{1}{c_{r-1}} < \frac{1}{c_{r-1}} \\
&\Rightarrow \frac{a_r}{b_r} < \frac{1}{c_{r-1}}.
\end{aligned}$$

Therefore, c_{r-1} never be selected for any round $r' \geq r$. □

Based on claim 1, the numerator of the remaining fractions strictly decreases after each step and it is always positive. Therefore, there exists some step with remaining fraction of numerator 1. As a result the algorithm stops after at most a steps and based on claim 2 it generates the desired sequence of integers.

Exercise 2: Matroids (6 points)

We have defined matroids in the lecture. For a matroid (E, I), a maximal independent set $S \in I$ is an independent set that cannot be extended. Thus, for every element $e \in E \setminus S$, the set $S \cup \{e\} \notin I$.

a) Show that all maximal independent sets of a matroid (E, I) have the same size. (This size is called the rank of a matroid.)

b) Consider the following greedy algorithm: The algorithm starts with an empty independent set $S = \emptyset$. Then, in each step the algorithm extends S by the minimum weight element $e \in E \setminus S$ such that $S \cup e \in I$, until S is a maximal independent set. Show that the algorithm computes a maximal independent set of minimum weight.

c) Let E be any finite subset of the natural numbers $\mathbb{N}$ and $k \in \mathbb{N}$ be any natural number. Define a collection of sets $I := \{X \subseteq E : \forall x \neq y \in X,\, x \not\equiv y \pmod{k}\}$. Show that (E, I) is a matroid.

Solution

a) Assume that there are two maximal independent sets S and T with $|S| \neq |T|$. Without loss of generality (w.l.o.g.) we assume that $|S| < |T|$. The exchange property tells us that there exists an element $x \in T \setminus S$ which can be added to S and that $S \cup \{x\}$ is still independent, which is a contradiction to the maximality of S.

b) Let $S = (e_1, \ldots, e_r)$ be the greedy solution and $T = (f_1, \ldots, f_r)$ be any other solution, where r is the rank of the matroid (note that both S and T need to have cardinality r to be maximal).
For the sake of contradiction assume that $w(T) < w(S)$, where $w(T) = \sum_{i=1}^{r} w(f_i)$ and the same for S. We also assume that e_i and f_i are already ordered by increasing weight. We now let k be the smallest index such that $w(f_k) < w(e_k)$ (note that for smaller indices equality may not hold).
Consider set $S_{k-1} := \{e_1, \ldots, e_{k-1}\}$ and $T_k := \{f_1, \ldots, f_k\}$. Using the exchange property we get that there is a $j \in [k]$ such that $S_{k-1} \cup \{f_j\}$ is independent. Since T is ordered, we know that $w(f_j) \leq w(f_k) < w(e_k)$. It means in step k the greedy algorithm skipped element f_j even though f_j has less weight than e_k and it would not have violated the independence of S_{k-1}. It contradicts the algorithm description.

c) To prove that (E, I) is matroid we need investigate the three matroid properties:

1. *Empty set is independent:* This property is trivially satisfied since there does not exist any pair of elements in the empty set to be equal modulo k.

2. *Hereditary property:* Let us assume that $A \in I$. Then there does not exist any pair of elements a and b in A such that $a \equiv b \pmod{k}$. It is clear that by removing any element from A, still, there does not exist any pair of elements with the mentioned property.

3. *Augmentation property:* First we need to notice that for any independent set $X \in I$ we have $|X| \leq k$, because for any integer k there exist k different residue classes of integers mod k.

   Now, let us assume that there exist two independent sets $A, B \in I$ such that $|B| < |A| \leq k$. We define A' the set of residue classes (mod k) of all integers in A, and B' the set of residue classes

(mod k) of all integers in B. Let $C' = A' \setminus B'$, which is not empty. Then any integer in A which belongs to one of the residue classes in C' can be added to B for its extension.