

# 15-354: Computational Discrete Math

K. Sutner

Assignment 1

Due: September 8, 2011

## Problem 1: The DASZ Operator (50 pts.)

### Background

For this problem, consider non-decreasing lists of positive integers  $A = (a_1, a_2, \dots, a_n)$ . We transform any such list into a new one according to the following simple recipe:

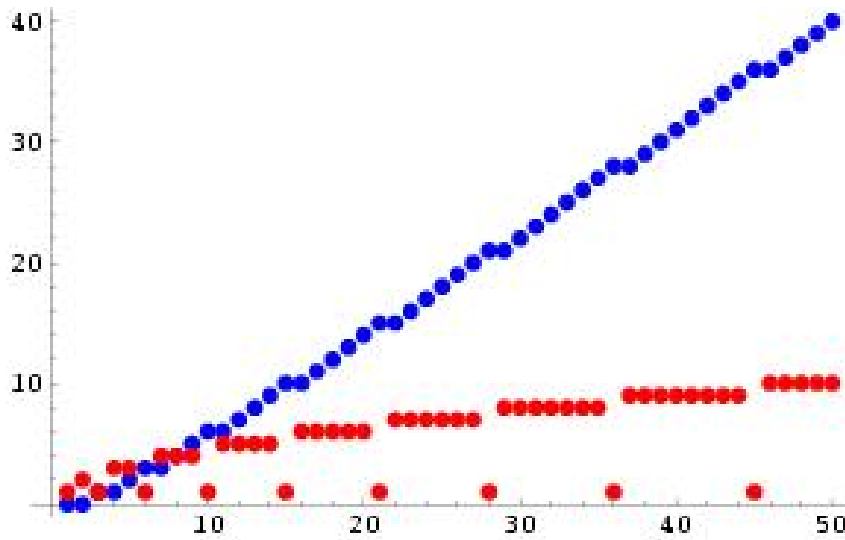
- Subtract 1 from all elements.
- Append the length of the list as a new element.
- Sort the list.
- Remove all 0 entries.

We will call this the *DASZ* operation (decrement, append, sort, kill zero) and write  $D(A)$  for the new list. For example,  $D((1, 3, 5)) = (2, 3, 4)$ ,  $D((4)) = (1, 3)$  and  $D((1, 1, 1, 1)) = (4)$ .

A single application of  $D$  is not too fascinating, but things become interesting when we apply the operation over and over again. As it turns out, no matter what the starting point  $A$  is, we always have  $D^{t+p}(A) = D^t(A)$  for some  $t \geq 0$  and  $p > 0$ . The least  $t$  and  $p$  are called the *transient* and *period* of  $A$ , respectively. For example, the transient and period of  $(1, 1, 1, 1, 1)$  are both 3:

0		1	1	1	1	1
1		5				
2		1	4			
3		2	3			
4		1	2	2		
5		1	1	3		
6		2	3			

A *fixed point* of  $D$  is any list  $A$  such that  $D(A) = A$  (i.e. the transient is 0 and the period is 1). The picture below shows the transients and the periods of all initial lists of the form  $A = (n)$  for  $n \leq 50$ . In the picture, blue indicates the transients and red the periods. The lists leading to a fixed point (corresponding to period 1) produce the red dots at the bottom of the picture.



**Task**

- A. Explain why lists must repeat after a finite number of steps (i.e., it cannot happen that  $D^i(A) \neq D^j(A)$  for all  $i < j$ ).
- B. Characterize all the fixed points of the DASZ operation.
- C. Determine which initial lists  $A = (n)$  lead to a fixed point.

**Comment**

You might find it useful to compute a few more examples. Needless to say, it is not enough to merely state the answers, you have to prove that your claim is correct.

**Problem 2: Sequence Numbers (20 pts.)**

**Background**

A polyadic function  $f : \mathbb{N}^* \rightarrow \mathbb{N}$  is called a *coding function* if there are “inverse” functions  $g : \mathbb{N} \rightarrow \mathbb{N}$  and  $h : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$  such that

$$g(f(a_1, a_2, \dots, a_n)) = n,$$

$$h(f(a_1, a_2, \dots, a_n), i) = a_i, \quad 1 \leq i \leq n$$

for all sequences  $a_1, a_2, \dots, a_n$ . Thus  $g$  determines the length of the sequence and  $h$  decodes it back into its elements. Moreover,  $h$  and  $g$  are supposed to be easily computable but let’s ignore that for the time being. Now consider the pairing function  $\pi$  defined by

$$\pi(x, y) = \frac{1}{2} (2 + 3x + x^2 + y + 2xy + y^2)$$

and define  $f$  as follows:

$$\begin{aligned} f(\text{nil}) &= 0 \\ f(a) &= \pi(0, a) \\ f(a_1, \dots, a_n) &= \pi(f(a_2, \dots, a_n), a_1) \end{aligned}$$

### Task

- Show that  $\pi$  is injective.
- Show that  $f$  is a coding function. Make sure to explain what the appropriate decoding functions  $g$  and  $h$  are.
- What would happen if we replaced  $\pi(x, y)$  by  $\pi(x, y) - 1$ ?

### Problem 3: RMs and Binary Digit Sums (30 pts.)

#### Background

Recall the register machine from class that computes the (binary) digit sum of a given number.

```
// binary digitsum of X --> Z
0:  dec X  1  4
1:  dec X  2  3
2:  inc Y  0
3:  inc Z  4
4:  dec Y  5  8
5:  inc Y  6
6:  dec Y  7  0
7:  inc X  6
8:  halt
```

In this problem you are supposed to determine the running time of this particular register machine program. Also, we will give one application of binary digit sums. For simplicity write  $\sigma(n)$  for the binary digit sum of  $n$ , so  $\sigma(10) = 2$  and  $\sigma(255) = 8$  (of course, the argument is written in decimal). We write  $\bar{x}$  for the complement of bit  $x$  and likewise for bit-sequences. Now consider the following bit-sequence  $(t_i)_{i \geq 0}$ :

$$\begin{aligned} t_0 &= 0 \\ t_{2n} &= t_n \\ t_{2n+1} &= \overline{t_{2n}} \end{aligned}$$

Let's write  $T_n$  for the binary word consisting of the first  $n$  bits of this sequence. For example,

$$T_{64} = 0110100110010110100101100110100110010110011010010110011010010110100110010110$$

Recall the shuffle operation which interleaves the elements of two sequences of equal length in a strictly alternating fashion:

$$a \parallel b = a_1b_1a_2b_2 \dots a_kb_k$$

Given shuffle we can define a sequence of binary words as follows.

$$\begin{aligned} S_0 &= 0 \\ S_n &= S_{n-1} \parallel \overline{S_{n-1}} \end{aligned}$$

As it turns out,  $\sigma(n) \pmod 2$ ,  $T$  and  $S$  all describe the same infinite sequence.

### Task

- A. Determine the time complexity of the digit sum register machine.
- B. Show that  $t_n = \sigma(n) \pmod 2$ .
- C. Show that  $T_{2^k} = S_k$ .

### Comment

For the running time first try to find an approximate answer, and then refine your solution to obtain a precise answer. Elegance counts. It is probably a good idea to take a close look at the flowgraph of the machine.