

---

## 1. Semidecidable Sets and Computable Functions (40)

---

### Background

We defined semidecidable sets as a generalization of decidable sets: on a Yes-instance the “semidecision algorithm” terminates, but on a No-instance it keeps running forever. There are many alternative characterizations that describe more directly the relationship between semidecidable sets and partial computable functions.

By an [enumeration](#) of  $A \subseteq \mathbb{N}$  we mean a partial function  $f : \mathbb{N} \rightarrow \mathbb{N}$  so that the range of  $f$  is  $A$ . For simplicity, we will assume that the support of  $f$  is either all of  $\mathbb{N}$  or some initial segment  $\{0, 1, \dots, n-1\}$ . So

$$A = \{f(i) \mid i < N\} = f(0), f(1), f(2), \dots$$

where  $N = n$  or  $N = \omega$ . Note that we allow  $n = 0$  corresponding to  $A = \emptyset$ . An enumeration is [repetition-free](#) if  $f$  is injective. A set is [recursively enumerable \(r.e.\)](#) if it can be enumerated by a computable function  $f$ .

### Task

Assume that  $A \subseteq \mathbb{N}$ . Show the following.

- A. All finite sets are recursively enumerable.
- B. The set of primes is recursively enumerable.
- C. The set of prime twins is recursively enumerable.
- D.  $A$  is semidecidable iff it is recursively enumerable.
- E.  $A$  is semidecidable iff it is recursively enumerable with a repetition-free enumeration.
- F. Suppose  $A$  is infinite. Then  $A$  is decidable iff it is recursively enumerable with a strictly increasing enumeration.

### Comment

Don’t try to argue formally in terms of register machines, just use computability in the intuitive sense, much the way you would describe a solution to a problem in an algorithms class.

Note that it is currently unknown whether there are infinitely many prime twins—but that does not affect part (C).

---

## 2. The DASZ Operator (30)

---

### Background

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

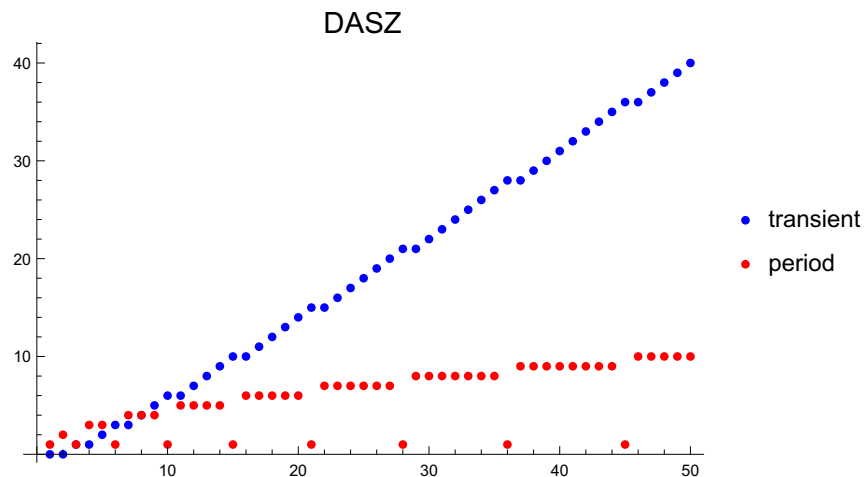
- Subtract 1 from all elements.
- Append the length of the list as a new element.
- Sort the list.
- Remove all 0 entries.

We will call this the DASZ operation (decrement, append, sort, kill zero) and write  $D(A)$  for the new list (note that  $D$  really is a function). 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 iterate the operation: as it turns out,  $D^t(A)$  always has a finite transient (and period), no matter how  $A$  is chosen. For example, the transient and period of  $(1, 1, 1, 1)$  are both 3:

	0	1	1	1	1
	1	5			
	2	1	4		
transient	3	2	3		
	4	1	2	2	
	5	1	1	3	
period	6	2	3		

Here is a plot of the transients and periods of all starting lists  $A = (n)$  for  $n \leq 50$ .



Note the fixed points  $D(A) = A$ , the few red dots at the bottom.

### Task

- Show that all transients must be finite.
- Characterize all the fixed points of the DASZ operation.
- Determine which initial lists  $A = (n)$  lead to a fixed point.

### 3. Speeding Up Iteration (30)

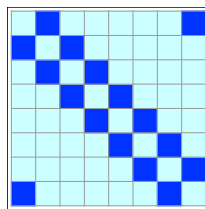
#### Background

The method of fast exponentiation can sometimes be used to speed-up the computation of  $f^t(a)$  for some endofunction  $f : A \rightarrow A$ . Here is an example, and a limitation to this speed-up effect.

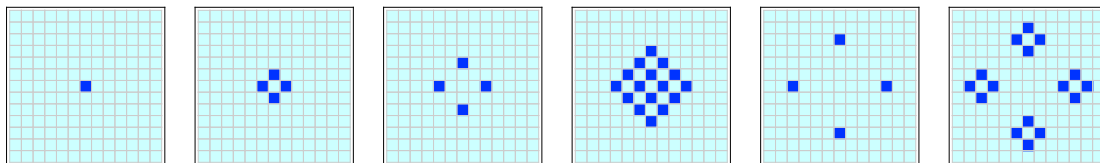
For  $n \geq 1$  let  $A = \mathbf{2}^{n \times n}$  be the set of all  $n \times n$  Boolean matrices. Define the [circulant matrix](#)  $C$  by

$$C(i, j) = \begin{cases} 1 & \text{if } j = i \pm 1 \\ 0 & \text{otherwise.} \end{cases}$$

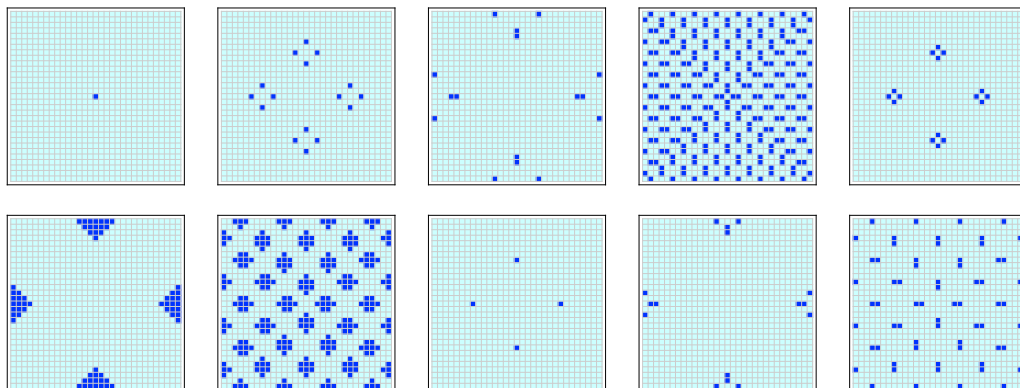
Here the indices are supposed to wrap around, so that, say,  $C_8$  has the form



Lastly, define  $f : A \rightarrow A$  by  $f(X) = C \cdot X + X \cdot C$  where for the matrix multiplication we interpret addition as logical *exclusive or* and multiplication as logical *and*. Here is the effect of applying  $f^t$  to the  $13 \times 13$  matrix with a single 1 in the center, rest all 0's, for  $t = 0, 1, \dots, 5$ .



Note how, at times 2 and 3, 4 and 5, the pictures contain 4 copies of the pictures at times 0 and 1. Similarly, the effect of  $f^t$  on the  $31 \times 31$  single-point matrix, for times  $t = 0, 10, 20, \dots, 90$ .



The patterns are rather surprising, you might want to write a program that the produces the whole orbit (and try different matrix sizes).

### Task

- A. Describe the effect of  $f$  on  $X \in A$  in geometric terms.
- B. Show how to compute  $f^t(X)$  for  $X \in A$  in time  $O(\text{pol}(n) \log t)$  where  $\text{pol}$  is a low-degree polynomial depending only on  $n$ . Make sure to explain the degree of  $\text{pol}$ .  
Hint: express  $f$  as a single matrix multiplication. You might want to look up Kronecker product.
- C. Show that  $\mathbb{P} = \mathbb{NP}$  if exponential speed-up is always possible.

### Comment

For part (C), find a way to determine satisfiability of a Boolean formula  $\phi(x_1, \dots, x_n)$  by iterating a function  $f$  defined essentially on  $2^n$ .