Problem Set 4

Due: March 23

Reading:

- Chapter 8. *Infinite Sets*, omitting 8.2. *The Halting Problem*
- Chapter 10 through 10.5. Directed Acyclic Graphs & Scheduling

Problem 1.
Describe which of the following twelve sets have bijections between them, with brief explanations.

$\mathbb{Z}$ (integers),	$\mathbb{R}$ (real numbers),
$\mathbb{C}$ (complex numbers),	$\mathbb{Q}$ (rational numbers),
$\text{pow}(\mathbb{Z})$ (all subsets of integers),	$\text{pow}(\emptyset)$,
$\text{pow}(\text{pow}(\emptyset))$,	$\{0,1\}^*$ (finite binary sequences),
$\{0,1\}^\omega$ (infinite binary sequences)	$\{\mathbf{T},\mathbf{F}\}$ (truth values)
$\text{pow}(\{\mathbf{T},\mathbf{F}\})$,	$\text{pow}(\{0,1\}^\omega)$

Problem 2.
Let $\mathbb{N}^\omega$ be the set of infinite sequences of natural numbers, and call a sequence $(a_0, a_1, a_2, \ldots) \in \mathbb{N}^\omega$ *strictly increasing* if $a_0 < a_1 < a_2 < \cdots$. Define the subset $\text{Inc} \subset \mathbb{N}^\omega$ to be the set of strictly increasing sequences. Prove that $\mathbb{N}^\omega$ bij Inc by *explicitly* constructing a bijection $f : \mathbb{N}^\omega \to \text{Inc}$. Carefully prove that

- f is a total function,
- $f(s) \in \text{Inc}$ for each $s \in \mathbb{N}^\omega$,
- f is injective,
- f is surjective.

Hint: Think sums, but remember that $0 \in \mathbb{N}$.

Problem 3.
Answer the following questions about the dependency DAG shown in Figure 1. Assume each node is a task that takes 1 second.

(a) What is the largest chain in this DAG? If there is more than one, only give one.

(b) What is the largest antichain? (Again, give only one if you find there are more than one). Prove there isn't a larger antichain.

2018, Albert R Meyer. This work is available under the terms of the Creative Commons Attribution-ShareAlike 3.0 license.

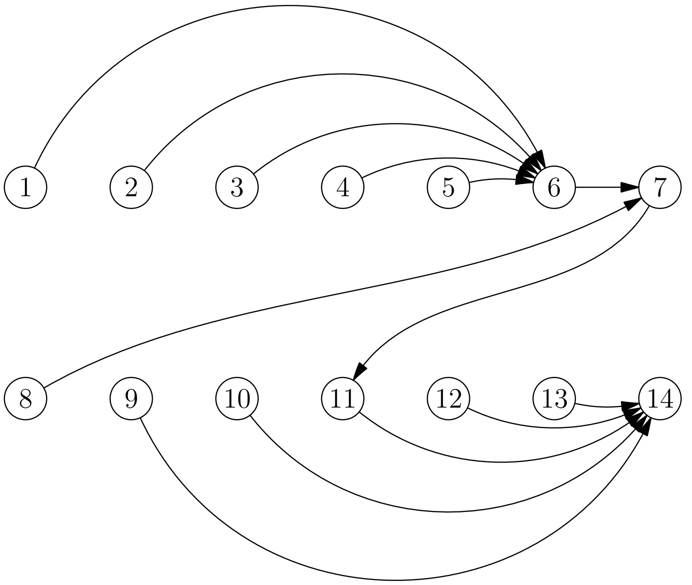

Figure 1 Task DAG

(c) How much time would be required to complete all the tasks with a single processor?

(d) How much time would be required to complete all the tasks if there are unlimited processors available.

(e) What is the smallest number of processors that would still allow completion of all the tasks in optimal time? Show a schedule proving it.