Problem Set 6

Deliverable: Submit your responses as a single, readable PDF file on the collab site before **6:29pm** on **Friday, 20 October**.

Collaboration Policy - Read Carefully

As with PS5, you should work in groups of one to three students to write-up a solution together. The rest of the collaboration policy is identical to what it was on PS5, and is not repeated here.

Preparation

This problem set focuses on Sections 6.1-6.3 of the MCS book, and Class 13 and Class 14.

Directions

Solve as many of the 8 problems as you can. For maximum credit, your answers should be correct, clear, well-written, and convincing.

Problems marked with $(\star)$ are challenging enough that it is not necessary to solve them well to get a "gold-star level" grade on this assignment (although we certainly hope you will try and some will succeed!)

State Machines

1. Describe a state machine that can be used to determine if the number of steps is divisible by 3. You should clearly define the set of states in your machine, and how each is interpreted, and the transition relation. (Don't forget to also specify q_0.)

2. Describe the set of states that are *reachable* for the "Progress" Machine defined as $M = (S, G, q_0)$:
   $S = \{(x, d) \mid x \in \mathbb{Z}, d \in \{\mathbf{F}, \mathbf{B}\}\}$
   $G = \{(x, \mathbf{F}) \to (x+1, \mathbf{B}) \mid x \in \mathbb{Z}\} \cup \{(x, \mathbf{B}) \to (x-2, \mathbf{F}) \mid x \in \mathbb{Z}\}$
   $q_0 = (0, \mathbf{F})$

3. $M = (S, G \subseteq S \times S, q_0 \in S)$ is a state machine where the cardinality of S is finite, and transition relation, G, is a total injective function. Show that it is possible for some states in S to be unreachable.

Knight's Moves

4. ($\star$) A *knight* in chess can move two squares up, down, left, or right, followed by one square in a direction perpendicular to the two squares (so, it can move two squares up or down, followed by one square left or right; or, can move two squares left or right, followed by one square up or down). We can define the possible moves of a knight on an infinite chess board as the Knight State Machine (similarly to how we defined the Bishop State Machine in Class 14):

   $S = \{(\_\_\_\_) \mid r, c \in \mathbb{N}\}$
   $G = \{(r, c) \to (r', c') \mid r, c \in \mathbb{N} \wedge ((r' = r \pm 2 \wedge c' = c \pm 1) \vee (r' = r \pm 1 \wedge c' = c \pm 2)) \wedge r' \geq 0 \wedge c' \geq 0\}$
   $q_0 = (0, 0)$

   Prove that all states in S are reachable for the Knight State Machine. (Hint: use induction, but be careful to either set up an appropriate $P(n)$ or break the proof into two separate induction proofs to show you can reach any column and can reach any rwo.)

Invariant Principle

5. MCS Problem 6.3.

6. Use the principle of induction to prove the invariant principle. (Hint: your induction predicate, $P(n)$ should use n as the number of steps.)

7. MCS Problem 6.7. (You should notice a strong similarity between this problem and the last problem on Problem Set 5!) (Part d is considered a ($\star$) problem; part e is rated ($\star\star$).) **TODO:** Which problem did you actually mean? I don't think the one that is 6.7 on p. 191? I think this was the old text https://uvacs2102.github.io/f16/docs/mcs.pdf problem, which made sense with what we did on PS5 last year, but not so much now. They seem to have removed that question from the book?

8. ($\star$) Prove the Python program below is a correct implementation of the factorial function.

```
def factorial(n):
    x = 1 # poor choices of variable names for programmers
    y = 1
    while x <= n:
        y = y * x
        x = x + 1
    return y
```

   To be *correct*, for any input $n \in \mathbb{N}$, the program should always return a result equal to $n!$. The factorial of n is defined as the product of all positive integers up to and including n. (Note that $0! = 1$.)

   A good answer will explain how you map the program to a state machine, and then prove partial correctness, and then prove termination.

