Assignment 2

Yifei Li

Problem 1:

The reason that any URM code contains the Transfer instruction can be converted to code without Transfer is the instruction Transfer can be replaced by combinations of Jumps and Increments. A Transfer instruction T(0, 1) can be represent by following instructions:

J(0, 1, 100)
S(1)
J(0, 0, 0)

This segment of code continuously adds 1 to register 1, until it equals the value of register 0. It performs the same action that T(0, 1) does, and results in copy the value in register 0 to register 1.

Problem 4:

We know that only S, Z, and T instructions modifies the value of the register. So to determine whether a specific register with index i remains the same during the process, we only need to check if there are instructions like S(i), Z(i) or T(n, i). If there aren't, the register with index i does not change. Take this segment of code as the example:

J(1, 3, 4)
S(0)
S(3)
J(0, 0, 0)
J(2, 4, 100)
S(0)
S(4)
J(0, 0, 4)

There are only S instructions for index 0, 3, and 4, so 1, 2 and other registers remain the initial state.

Problem 6:

- G : the garage door is opened
  C : there is a car in the garage
  H: there is someone at home
  $G \land \neg C \land H$

- C : temperature above 30 degerees
  H: temperature below 70 degerees
  $C \land H$

- $\neg((A \land \neg B) \lor (B \land C)) \land C$
  A:false B:false C:true

- $\neg(\neg A \lor (\neg(\neg(B \lor A) \lor C)))$
  A:true B:true C:false

Problem 7:

$\neg(\neg A \lor \neg B)$ is equivalent to $A \land B$

$\neg(\neg A \land \neg B)$ is equivalent to $A \lor B$

$A|A$ is equivalent to $\neg A$

$(A|A)|(B|B)$ is equivalent to $A \lor B$

Problem 8:

I think this is a kind of manifestation of having a "condition," just like the “if” statement in advanced programming languages. In order to make A→B true, B must be true if A is true, and else if A is false, the result is true no matter what B is. I found a good example of this in natural language for me to understand is that “if Bob got an A in school, his dad will buy him a bicycle”. Here, A is “Bob got A”, B is “dad buy bicycle” and A→B is “the promise made by Bob’s dad”. A→B is true means Bob’s dad keeps the promise well, if Bob didn’t got A, we will no longer discuss whether his dad will buy a bike for him, he will “keep” this promise anyway, because the preconditions are not met.