CS 2710/ISSP 2160 Fall 2013

Assignment 4: The Situation Calculus

REQUIRED FOR CREDIT: You must sign the following honesty and integrity statement, by hand.

I attest that I carried out all of the work below, that I did it alone, and that I did it without using resources outside of those provided by class (CS2710 ISSP 2160 Foundations of Artificial Intelligence, Fall 2013).

SIGNED:

Below are operators in the STRIPS language for representing actions and their effects. The domain is the blocks world. Imagine a child's set of building blocks on the a table top. The problem is to move the blocks from their starting configuration into some goal configuration. We will assume that each block can have only one other block directly on top of it, although blocks can be stacked to arbitrary heights. A block may be on another block, or on the table.

A robot is performing the actions. Before he can pickup a block, or stack one on top of the other, his arm must be empty.

Here are operators for this world, in STRIPS. The "P" are preconditions that must be true before you can perform that action. The "D" are things that are no longer true after performing the action. The "A" are new things that are true after performing the action.

```
stack(x,y)
  P: clear(y) and holding(x)
  D: clear(y) and holding(x)
  A: armempty and on(x,y)

unstack(x,y)
  P: on(x,y) and clear(x) and armempty
  D: on(x,y) and armempty
  A: holding(x) and clear(y)

pickup(x)
  P: clear(x) and ontable(x) and armempty
  D: ontable(x) and armempty
  A: holding(x)
```

```
putdown(x)
  P: holding(x)
  D: holding(x)
  A: ontable(x) and armempty
```

There is no explicit negation in STRIPS. There is a working memory holding the current positive true statements. If something becomes false (such as *holding(block1)*), then that is simply removed from working memory.

Your task is to represent the above operators in the Situation Calculus, specifically the version covered in the lecture notes (which are described ending on page 333 in Russell and Norvig (2nd edition)). However, if you read that description in the text, be aware that the book goes through some intermediate formulas, not just the final versions in the lecture slides.

Assume that your sentences are to be processed by a full-first-order-logic theorem prover. Thus, you have no notion of "working memory", and you **do** need to represent negation. Show an example of each type of axiom involving negation your system would need. However, you don't have to type them all out.