Sean Myers

Assignment 5

Before I begin the report, I ran all the cases that were already posted. They seem to match (other than plan number), perfectly. So I will only include in my report my STRIPS language and the logic to prove that what the program is doing is correct.

My STRIPS language:

Actions:

put\_in\_casserole:

Prereqs: clear(oven), not\_cooked(casserole)

Add: in\_oven(casserole)

Delete: clear(oven)

put\_in\_cake:

Prereqs: clear(oven), not\_cooked(cake)

Add: in\_oven(cake)

Delete: clear(oven)

put\_in\_microwave:

Prereqs: clear(microwave), not\_cooked(soup)

Add: in\_microwave(soup)

Delete: clear(microwave)

take\_out\_cake:

Prereqs: in\_oven(cake)

Add: clear(oven), cooked(cake)

Delete: in\_oven(cake), not\_cooked(cake)

take\_out\_casserole:

Prereqs: in\_oven(casserole)

Add: clear(oven), cooked(casserole)

Delete: in\_oven(casserole), not\_cooked(casserole)

take\_out\_microwave:

Prereqs: in\_microwave(soup)

Add: clear(microwave), cooked(soup)

Delete: in\_microwave(soup), not\_cooked(soup)

init: not\_cooked(soup), not\_cooked(casserole), not\_cooked(cake)

goal: cooked(soup), cooked(casserole), cooked(cake)

So here we are cooking food! Our final output should order the cooking of cake or casserole before the other since both can't be in the oven at the same time. Also, we must put something in before we take it out, so the ordering constraints will have all those.

The final output is as follows for ordering constraints:

ordering constraints (other than those with goal or init):

(0\*put\_in\_casserole < 0\*take\_out\_casserole)

(0\*put\_in\_microwave < 0\*take\_out\_microwave)

(0\*put\_in\_cake < 0\*take\_out\_cake)

(0\*take\_out\_cake < 0\*put\_in\_casserole)

The first three are the orders for putting in and taking out the item (always need to put in before you take out)

The last one is the order of putting in the oven, so that way they aren't in the oven at the same time.

As for threats, the only time there should be a threat is when both oven items are trying to put something in the oven at the same time, which appears when building the plans, but not during the final output, only because my final plan resolved the open conditions in such a way that there was never any threat. Also there seemed to be odd threats in successors where it tries to build multiple cakes or something crazy like that, but that was from the order of successors it tried before trying a different plan path:

(from output, the threat for putting in things into the oven)

threats:

(('init', 'clear(oven)', '0\*put\_in\_casserole'), '0\*put\_in\_cake')

The causal links fully constructed seems correct as well. It puts all the things in the oven at some point in time:

(init < not\_cooked(casserole) < 0\*put\_in\_casserole)

(init < not\_cooked(soup) < 0\*put\_in\_microwave)

(init < not\_cooked(cake) < 0\*put\_in\_cake)

(of course, the ordering constraints put put\_in\_casserole first)

Also the goals are solved for each:

(0\*take\_out\_casserole < cooked(casserole) < goal)

(0\*take\_out\_microwave < cooked(soup) < goal)

(0\*take\_out\_cake < cooked(cake) < goal)

The final causal links, for your perusal are here:

causal links:

(0\*take\_out\_casserole < cooked(casserole) < goal)

(0\*take\_out\_microwave < cooked(soup) < goal)

(0\*take\_out\_cake < cooked(cake) < goal)

(0\*put\_in\_casserole < in\_oven(casserole) < 0\*take\_out\_casserole)

(0\*put\_in\_microwave < in\_microwave(soup) < 0\*take\_out\_microwave)

(0\*put\_in\_cake < in\_oven(cake) < 0\*take\_out\_cake)

(0\*take\_out\_cake < clear(oven) < 0\*put\_in\_casserole)

(init < not\_cooked(casserole) < 0\*put\_in\_casserole)

(init < clear(microwave) < 0\*put\_in\_microwave)

(init < not\_cooked(soup) < 0\*put\_in\_microwave)

(init < clear(oven) < 0\*put\_in\_cake)

(init < not\_cooked(cake) < 0\*put\_in\_cake)