Name: Pratik Jasani

Course: EN605.645

**Planning Self Check**

**2a. What are S, A, T for this problem?**

**S –** a predicate

**A –**  the possible actions that can be made from the current state

**T –**  applying the action to the state causing it to change

**2b. What will the successors() function need to do? Think about Module 7 and unification**

The unification function will help determine the successors of the current state.

**2c. Solve at least part of the plan (one full step). Having worked through one full step, is there anything else that you need to have in your implementation?**

|  |  |  |
| --- | --- | --- |
| Current State | Explored | Goal stack |
| Ontable(A)  ontable(B)  on(C,B)  clear(A)  clear(C)  handempty |  |  |

**Pickup(x): ontable(x) & clear(x) & handempty**

S1 = ontable(x) & clear(x) & handempty where X can be A

**Putdown(x): holding(x) :** FAIL

**Stack(x,y): holding(x) & clear(y):** FAIL

**Unstack(x,y): on(x, y) & clear(x) & handempty**

S2 = on(x, y) & clear(x) & handempty where x = C and y = B

S1

S2

Start

|  |  |  |  |
| --- | --- | --- | --- |
| S1: current State | S1:explored | S2: Current State | S2:explored |
| ontable(B)  on(C,B)  clear(C)  holding(A) | Pickup(A) | Ontable(A)  ontable(B)  clear(A)  clear(B)  holding(C) | Unstack(c,b) |

**Pickup(x): ontable(x) & clear(x) & handempty :** FAIL

**Putdown(x): holding(x) :**

S3: holding(x) where x = A

**Stack(x,y): holding(x) & clear(y):**

S4 : holding(x) & clear(y) where x = A, y = C

**Unstack(x,y): on(x, y) & clear(x) & handempty:** FAIL

Start

S1

S4

S3

S2

|  |  |  |  |
| --- | --- | --- | --- |
| S3: current State | S3: explored | S4: Current State | S4: explored |
| Ontable(A)  ontable(B)  on(C,B)  clear(C)  clear(A)  handempty | Pickup(A)  Putdown(A) | ontable(B)  on(C,B)  on(A,C)  clear(A)  handempty | Pickup(A)  Stack(A,C) |

**Continue from S2:**

**Pickup(x): ontable(x) & clear(x) & handempty :** FAIL

**Putdown(x): holding(x) :**

S5: holding(x) where x = C

**Stack(x,y): holding(x) & clear(y):**

S6: holding(X) & clear(y) , X = C, y = A

S7 : holding(X) & clear(y) , X = C, y = B

**Unstack(x,y): on(x, y) & clear(x) & handempty:** FAIL

S4

S3

S2

S1

Start

S6

S5

S7

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| S5: Current State | S5:explored | S6: Current State | S6:explored | S7: Current State | S7:explored |
| Ontable(A)  ontable(B)  ontable(C)  clear(A)  clear(B)  clear(C)  handempty | Unstack(c,b)  Putdown(C) | Ontable(A)  ontable(B)  on(C,A)  clear(C)  clear(B)  handempty | Unstack(c,b)  Stack(C,A) | Ontable(A)  ontable(B)  on(C,B)  clear(A)  clear(C)  handempty | Unstack(c,b)  Stack(C,B) |

After looking at the above walkthrough I think we need to keep track of the path that we have taken so we do not infinitly loop on the same path. For example We don’t want to constantly pick up A and drop A and pickup A and drop A…..