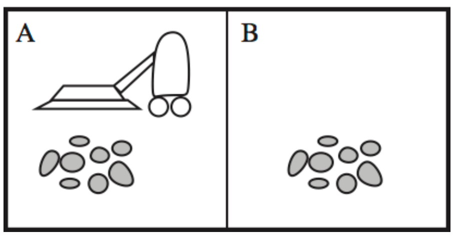
Washington State University
School of Electrical Engineering and Computer Science
Fall 2019

CptS 440/540 Artificial Intelligence
Homework 7 – Solution
Due: October 17, 2019 (11:59pm)

General Instructions: Put your answers to the following problems into a PDF document and submit as an attachment under Content → Homework 7 for the course CptS 440 Pullman (all sections of CptS 440 and 540 are merged under the CptS 440 Pullman section) on the Blackboard Learn system by the above deadline. Note that you may submit multiple times, but we will only grade the most recent entry submitted before the above deadline.

1. Recall the vacuum cleaner world depicted below from the Agents lecture (slide 4). We want to define the three actions (Left, Right, Suction) for a planner in the PDDL format. Below is the domain PDDL specification (missing the Suction action) and the problem PDDL specification.




```
(define (problem prob)
  (:domain VACUUM)
  (:objects A B)
  (:init (room A) (dirty A) (dirty B))
  (:goal (and (clean A) (clean B)))
)
```

```
(define (domain VACUUM)
  (:predicates
    (room ?r)
    (dirty ?r)
    (clean ?r)
  )

  (:action left
    :precondition (room B)
    :effect (and (not (room B)) (room A))
  )

  (:action right
    :precondition (room A)
    :effect (and (not (room A)) (room B))
  )

)
```

   a. Show the definition of the Suction action in PDDL format consistent with the domain and problem specifications above.
   b. *CptS 540 students only*: Put the above domain PDDL specification, including your Suction action, in a file called "domain.pddl". Put the above problem PDDL specification in a file called "prob.pddl". Run the Fast-Downward planner on your domain and problem using the following command:
   `./fast-downward.py domain.pddl prob.pddl --search "astar(blind())"`
   Include the output in your homework submission. The Fast-Downward planner is available at http://www.fast-downward.org. Download and installation instructions are available at http://www.fast-downward.org/ObtainingAndRunningFastDownward.

Solution:

   a. Below is the PDDL specification for the Suction action.

```
(:action suction
  :parameters (?r)
  :precondition (and (room ?r) (dirty ?r))
  :effect (and (not (dirty ?r)) (clean ?r))
)
```

b. The following is the output of Fast-Downward using the above problem PDDL and domain PDDL with Suction action from part (a). The final plan solution is highlighted.

```
INFO     Running translator.
INFO     translator stdin: None
INFO     translator time limit: None
INFO     translator memory limit: None
INFO     translator command line string: /usr/local/opt/python/bin/python3.7
/Users/holder/class/AI/systems/planner/downward/builds/release/bin/translate/translate.py
domain.pddl prob.pddl --sas-file output.sas
Parsing...
Parsing: [0.000s CPU, 0.001s wall-clock]
Normalizing task... [0.000s CPU, 0.000s wall-clock]
Instantiating...
Generating Datalog program... [0.000s CPU, 0.000s wall-clock]
Normalizing Datalog program...
Normalizing Datalog program: [0.000s CPU, 0.001s wall-clock]
Preparing model... [0.000s CPU, 0.000s wall-clock]
Generated 8 rules.
Computing model... [0.000s CPU, 0.000s wall-clock]
15 relevant atoms
2 auxiliary atoms
17 final queue length
18 total queue pushes
Completing instantiation... [0.000s CPU, 0.000s wall-clock]
Instantiating: [0.000s CPU, 0.001s wall-clock]
Computing fact groups...
Finding invariants...
6 initial candidates
Finding invariants: [0.000s CPU, 0.000s wall-clock]
Checking invariant weight... [0.000s CPU, 0.000s wall-clock]
Instantiating groups... [0.000s CPU, 0.000s wall-clock]
Collecting mutex groups... [0.000s CPU, 0.000s wall-clock]
Choosing groups...
0 uncovered facts
Choosing groups: [0.000s CPU, 0.000s wall-clock]
Building translation key... [0.000s CPU, 0.000s wall-clock]
Computing fact groups: [0.000s CPU, 0.001s wall-clock]
Building STRIPS to SAS dictionary... [0.000s CPU, 0.000s wall-clock]
Building dictionary for full mutex groups... [0.000s CPU, 0.000s wall-clock]
Building mutex information...
Building mutex information: [0.000s CPU, 0.000s wall-clock]
Translating task...
Processing axioms...
Simplifying axioms... [0.000s CPU, 0.000s wall-clock]
Processing axioms: [0.000s CPU, 0.000s wall-clock]
Translating task: [0.000s CPU, 0.000s wall-clock]
0 effect conditions simplified
0 implied preconditions added
Detecting unreachable propositions...
0 operators removed
0 axioms removed
3 propositions removed
```

```
Detecting unreachable propositions: [0.000s CPU, 0.000s wall-clock]
Reordering and filtering variables...
3 of 3 variables necessary.
0 of 3 mutex groups necessary.
4 of 4 operators necessary.
0 of 0 axiom rules necessary.
Reordering and filtering variables: [0.000s CPU, 0.000s wall-clock]
Translator variables: 3
Translator derived variables: 0
Translator facts: 6
Translator goal facts: 2
Translator mutex groups: 0
Translator total mutex groups size: 0
Translator operators: 4
Translator axioms: 0
Translator task size: 25
warning: could not determine peak memory
Writing output... [0.000s CPU, 0.000s wall-clock]
Done! [0.000s CPU, 0.004s wall-clock]
translate exit code: 0

INFO     Running search (release).
INFO     search stdin: output.sas
INFO     search time limit: None
INFO     search memory limit: None
INFO     search command line string:
/Users/holder/class/AI/systems/planner/downward/builds/release/bin/downward --search
'astar(blind())' --internal-plan-file sas_plan < output.sas
reading input... [t=0.000205852s]
done reading input! [t=0.000367983s]
Initializing blind search heuristic...
Building successor generator...done! [t=0.00152033s]
peak memory difference for successor generator creation: 0 KB
time for successor generation creation: 1.4014e-05s
Variables: 3
FactPairs: 6
Bytes per state: 4
Conducting best first search with reopening closed nodes, (real) bound = 2147483647
New best heuristic value for blind: 1
[g=0, 1 evaluated, 0 expanded, t=0.00168738s, 4272492 KB]
f = 1 [1 evaluated, 0 expanded, t=0.0016971s, 4272492 KB]
Initial heuristic value for blind: 1
pruning method: none
f = 2 [3 evaluated, 1 expanded, t=0.00173002s, 4272492 KB]
f = 3 [5 evaluated, 3 expanded, t=0.00174681s, 4272492 KB]
New best heuristic value for blind: 0
[g=3, 7 evaluated, 5 expanded, t=0.00176695s, 4272492 KB]
Solution found!
Actual search time: 6.3342e-05s [t=0.00178369s]
suction a (1)
right  (1)
suction b (1)
Plan length: 3 step(s).
Plan cost: 3
Expanded 6 state(s).
```

```
Reopened 0 state(s).
Evaluated 7 state(s).
Evaluations: 7
Generated 8 state(s).
Dead ends: 0 state(s).
Expanded until last jump: 3 state(s).
Reopened until last jump: 0 state(s).
Evaluated until last jump: 5 state(s).
Generated until last jump: 5 state(s).
Number of registered states: 7
Int hash set load factor: 7/8 = 0.875
Int hash set resizes: 3
Search time: 0.000129485s
Total time: 0.00178883s
Solution found.
Peak memory: 4272492 KB
Remove intermediate file output.sas
search exit code: 0
```

---

2. Suppose we are given the following full joint probability distribution for Halloween World, where random variable *Weather* has domain {clear, cloudy, rain}, random variable *Costume* has domain {yes, no}, and random variable *Party* has domain {yes, no}. Compute the following probabilities. Show your work.

	Weather:	clear		cloudy		rain	
	Costume:	yes	no	yes	no	yes	no
Party:	yes	0.084	0.032	0.18	0.06	0.09	0.024
	no	0.036	0.048	0.12	0.14	0.09	0.096

a. P(Weather=clear, Costume=yes, Party=yes).
b. P(Weather=cloudy, Party=no).
c. P((Costume=yes) $\wedge$ (Party=no)).
d. P((Costume=yes) $\vee$ (Party=no)).
e. P(Party=yes | Weather=rain, Costume=no).
f. P(Party=yes | Costume=yes).

Solution:

a. P(Weather=clear, Costume=yes, Party=yes) = 0.084
b. P(Weather=cloudy, Party=no) = 0.12 + 0.14 = 0.26
c. P((Costume=yes) $\wedge$ (Party=no)) = 0.036 + 0.12 + 0.09 = 0.246
d. P((Costume=yes) $\vee$ (Party=no))
   = P(Costume=yes) + P(Party=no) – P((Costume=yes) $\wedge$ (Party=no))
   = (0.084 + 0.036 + 0.18 + 0.12 + 0.09 + 0.09) + (0.036 + 0.048 + 0.12 + 0.14 + 0.09 + 0.096) – 0.246
   = 0.6 + 0.53 – 0.246 = 0.884
e. P(Party=yes | Weather=rain, Costume=no)
   = P(Party=yes, Weather=rain, Costume=no) / P(Weather=rain, Costume=no)
   = 0.024 / (0.024 + 0.096) = 0.2
f. P(Party=yes | Costume=yes)
   = P(Party=yes, Costume=yes) / P(Costume=yes)
   = (0.084 + 0.18 + 0.09) / 0.6 (from d) = 0.59

3. Suppose we are given the following information about the Boolean random variables LikeCoding and LearnAI.
   - P(LikeCoding=true | LearnAI=true) = 0.8
   - P(LikeCoding=true | LearnAI=false) = 0.6
   - P(LearnAI=true) = 0.5

   Using Bayes rule and normalization, compute **P**(LearnAI | LikeCoding=true). Note the boldfaced **P** means we want the probability *distribution* of LearnAI given LikeCoding=true. Show your work, including the value of the normalization constant α.

   Solution:

   Using Bayes Rule:
   P(LearnAI | LikeCoding=true)
   = **P**(LikeCoding=true | LearnAI) * **P**(LearnAI) / P(LikeCoding=true)
   = α **P**(LikeCoding=true | LearnAI) * **P**(LearnAI)
   = α < P(LikeCoding=true | LearnAI=true) * P(LearnAI=true),
   P(LikeCoding=true | LearnAI=false) * P(LearnAI=false) >
   = α < (0.8)(0.5), (0.6)(0.5) >
   = α < 0.4, 0.3 > $\alpha = 1 / (0.4 + 0.3) = 1.43$
   = < 0.57, 0.43 >