Assignment #3 Artificial Intelligence - CSCE 523 Due: 8:00 AM, Monday February 25, 2019 Knowledge Representation and Planning

Marvin Newlin

Turnin: E-mail me a zip file containing your typed solutions and associated files to all questions.

1. (5 points) What logic rule did the Cheshire Cat use in this argument, and

is it sound?

"To begin with," said the Cat, "a dog's not mad. You grant that?" "I suppose so," said Alice. "Well, then," the Cat went on, "you see a dog growls when it's angry, and wags its tail when it's pleased. Now I growl when I'm pleased and wag my tail when I'm angry. Therefore I'm mad."

**Sol’n:**

Let the Statement A = “Not mad” and let statement B = “Growl when Angry and Wag tail when pleased.” Now, the cat argues with the following logic

, then the cat says “Now I growl when I'm pleased and wag my tail when I'm angry. Therefore, I'm mad.” In logical terms this is . Checking our logic rules, we see that this is the following: . This rule is called *modus tollens*. This is a sound logic procedure since it only generates entailed sentences.

2. (5 points) Translate the following Lewis Carroll sentences into a

Propositional Logic Knowledge Base and derive two statements from the Knowledge Base:

All hummingbirds are richly colored.

No large birds live on honey.

Birds that do not live on honey are dull in color.

**Sol’n:**

Let H = hummingbirds

Let R = richly colored

Let L = large birds

Let B = live on Honey

Assume: dull colored = ~R, not live on honey = ~B

KB:

P1:

P2:

P3:

Via modus tollens, the first entailment we get is the contrapositive of P3, E1:

, Richly colored birds live on honey

Additionally, taking , we get the following:

(implication)

(associativity)

(satisfiability)

E2: , hummingbirds live on honey

Taking we get

= (associativity)

(Idempotency)

(and introduction)

(by entailment)

So our second entailment is that hummingbirds are not large birds

This leaves us with our two derived sentences.

S1: Hummingbirds live on honey

S2: Hummingbirds are not large birds

3. (5 points) Translate the following into First Order Logic and then convert

to Conjunctive Normal Form (CNF):

According to the Pidgeon: If little girls eat eggs, then they are a kind of serpent. Alice (who is a little girl) eats eggs. Therefore, she is a kind of serpent.

FOPC:

Conversion to CNF

Starting with statement 1

* (implication)
* (DeMorgan’s)
* (UI)

Statement 2

* (implication)
* (DeMorgan’s)

This leaves us with our two statements in CNF

4. (10 points) Determine for the following pairs of sentences if they can be

unified and if they can, given the most general unifier, if not discuss why.

1. P(x) ∨ Q(Dog, x) ⇒ R(Cat,Dog) P(Cat) ∨ Q(y,z) ⇒ R(z,y)

**Sol’n:**

1. Queen(Hearts) ^ HasProblem(Hearts, y) ⇒ Solve(y, Beheading) ^HasHeadandBody(y) Queen(x) ^ HasProblem(x, Cheshire) ⇒ Solve(y, Beheading) ^ HasHeadandBody(y)

Can be unified by the following

1. (Son(x,x) ∧ Sister(Mary,Jack)) ⇒ (Daughter(x,Mary) ∧ Brother(Jack,Mary) ) (Son(Jack,x) ∧ Sister(z,x)) ⇒ (Daughter(z,f(x)) ∧ Brother(y,z))

* These two cannot be unified because of the f(x) function. We can only replace a function without a variable which can only be done once x is substituted for Jack but at that point there is no other substitution to be made

5. (15 points) Use resolution with refutation to show that the following three queries can be inferred from the given knowledge base. At each resolution step also indicate the corresponding identifier and binding list.

KB:

* S1: Uncle(John, Jack)
* S2: Aunt( Mary, Amy)
* S3: Female(Amy)
* S4: Brother(Jack, Amy)
* S5: Brother(Bill, Jack)
* S6: Sister(x,y) ⇒ Siblings(x,y)
  + CNF: (Implication)
* S7: Brother(x,y) ⇒ Siblings(x,y)
  + CNF: (Implication)
* S8: Brother(x,y) ∧ Female(y) ⇒ Sister(y,x)
  + CNF: (Implication then DeMorgan’s)
* S9: Siblings(x,y) ∧ Uncle(z,y) ⇒ Uncle(z,x)
  + CNF: (Implication then DeMorgan’s)
* S10: Siblings(x,y) ⇒ Siblings(y,x)
  + CNF: (Implication)
* S11: Uncle(x,y) ∧ Aunt(z,y) ⇒ Married(x,z)
  + CNF: (Implication then DeMorgan’s)
* S12: Uncle(x,y) ∧ Married(z,x) ⇒ Aunt(z,y)
  + CNF: (Implication then DeMorgan’s)
* S13: Married(x,y) ⇒ Married(y,x)
  + CNF: (Implication)

Note: We use the CNF version of each statement in the resolutions below

Note: Negated goal statements are underlined and are never added to KB

1. Married(John, Mary)

**Sol’n:**

KB:

* S1: Uncle(John, Jack)
* S2: Aunt( Mary, Amy)
* S3: Female(Amy)
* S4: Brother(Jack, Amy)
* S5: Brother(Bill, Jack)
* S6:
* S7:
* S8:
* S9:
* S10:
* S11: :
* S12:
* S13:
* S14: NG = ~Married(John, Mary)
* S15:[3, 4, 8]:
* S16: [6, 15]:
* S17: [16, 9]:
* S18: [1, 18]:
* S19: [18, 2, 11]:
* S20: [19, 14]: Q.E.D

1. Aunt(Mary, Jack)

KB: (Bold statements were proved in previous part)

* S1: Uncle(John, Jack)
* S2: Aunt( Mary, Amy)
* S3: Female(Amy)
* S4: Brother(Jack, Amy)
* S5: Brother(Bill, Jack)
* S6:
* S7:
* S8:
* S9:
* S10:
* S11: :
* S12:
* S13:
* S15:
* S16:
* S17:
* S18:
* S19:
* S21: NG = ~Aunt(Mary, Jack)
* S22: [19, 13]:
* S23: [12, 22]:
* S24: [1, 23]:
* S25: [21, 24]: Q.E.D

1. ∃x(Siblings(Jack,x) ∧Uncle(John,x))

KB: (Bold statements were proved in previous part)

* S1: Uncle(John, Jack)
* S2: Aunt( Mary, Amy)
* S3: Female(Amy)
* S4: Brother(Jack, Amy)
* S5: Brother(Bill, Jack)
* S6:
* S7:
* S8:
* S9:
* S10:
* S11: :
* S12:
* S13:
* S15:
* S16:
* S17:
* S18:
* S19:
* S22:
* S23:
* S24:
* NG:
* NG-CNF S26:
* S27:[10, 16]:
* S28:[18, 26, 27]: Q.E.D

6. (10 points)Translate the knowledge base from problem 5 into a formula list for otter and use it to perform a proof by refutation for the queries from problem 5, and the two below. A copy of the otter executable and documentation can be found in the course directory. If you want to run otter on a non-Windows computer, you can access the information you will need at http://www-unix.mcs.anl.gov/AR/otter/. If a sentence cannot be proved determine what needs to be added to the knowledge base to make it provable and would this invalidate the KB. (Turn-in the otter files, and a copy of the screen output for each query)

**Note:** See files problem6a.txt – problem6e.txt for otter code for each respective subproblem and problem6a\_answer.txt – problem6e\_answer.txt for respective proof files from otter. All command line output from otter was piped into the \_answer files.

KB Put into Otter:

* S1: Uncle(John, Jack)
* S2: Aunt( Mary, Amy)
* S3: Female(Amy)
* S4: Brother(Jack, Amy)
* S5: Brother(Bill, Jack)
* S6:
* S7:
* S8:
* S9:
* S10:
* S11:
* S12:
* S13:

d. Brother(Bill, Amy)

This one cannot be proved with the knowledge base as is. We have to add the following statements to the KB to get the proof to complete (otter syntax used)

1. all x all y all z ((Siblings(x,y) & Siblings(y,z)) -> Siblings(x,z)).

2. Male(Bill).

3. Siblings(Amy, Jack).

4. all x all y (Siblings(x,y) & Male(x) -> Brother(x,y)).

None of these invalidate the KB. The first is the transitive property for sibling relationships. The second follows from Bill being Jack’s brother (given in the KB). The 3rd is proven above in part a. The 4th is also a logical conclusion.

e. Uncle(John, Bill) ^ Siblings(Bill, Jack)

Proof is successful. See problem6e.txt and problem6e\_answer.txt.

7. (25 points) For the following logic problem, a) encode the problem and have Otter solve it, and b) represent the problem as a constraint satisfaction problem and solve using the backward algorithm with forward checking.

Link, Zelda, and Ganondorf fought three different evil creatures, the Octorock, an Iron Knuckle, and a Poe. They fought them with three different weapons, a bow and arrows, magic, and a sword. Determine who fought what creature and with what weapon.

1. Ganondorf did not fight the Octorock. 2. The Iron Knuckle was not fought against with magic. 3. Zelda fought the Octorock. 4. Link has a sword and did not fight the Poe. 5. Zelda fought with the bow and arrows.

a) For otter code, text files are labeled problem7\*.txt, where \* is the subproblem. Proof files are named similarly with \_answer appended to the file name. Each otter file contains a single subproblem for otter to solve

a. See file problem7a.txt for KB and formulas and problem7a\_answer.txt for results.

* Goal for subproblem a: Ganondorf used Magic
* Proof was successful

b. See file problem7b.txt for KB and formulas and problem7b\_answer.txt for results.

* Goal for subproblem b: Ganondorf killed Poe
* Proof was successful

c. See file problem7c.txt for KB and formulas and problem7c\_answer.txt for results.

* Goal for subproblem c: Link killed Iron Knuckle
* Proof was successful

**Sol’n:** After proving these 3 subproblems, along with the initial conditions. We have that

Zelda fought the Octorock with the Bow and Arrows (constraints 3 and 5)

Link used the Sword and fought the Iron Knuckle (constraint 4 and result of subproblem c)

Ganondorf used Magic and fought Poe (subproblems a and b)

See the file problem7\_results.txt for otter code to prove these 3 statements and problem7\_results\_answers.txt for the proof.

b)

**Sol’n:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Step |  | B | M | S | I | O | P | Constraint Violation |
| 1 | L | x | x | y |  |  | x | none |
| 2 | Z | y | x | x | x | y | x | none |
| 3 | G | x | y | x | y | x |  | Violates constraint 2 |
| 4 | G | x | y | x | x | x | y | none |
| 5 | L | x | x | y | y | x | x | none |

Note: Green indicates final results. Red indicates the assignment violates a constraint. Additionally, 2 assignments are made in each step. First, the weapon and then the villain. Assignments are made in the order shown in the table, reading left to right and top to bottom.

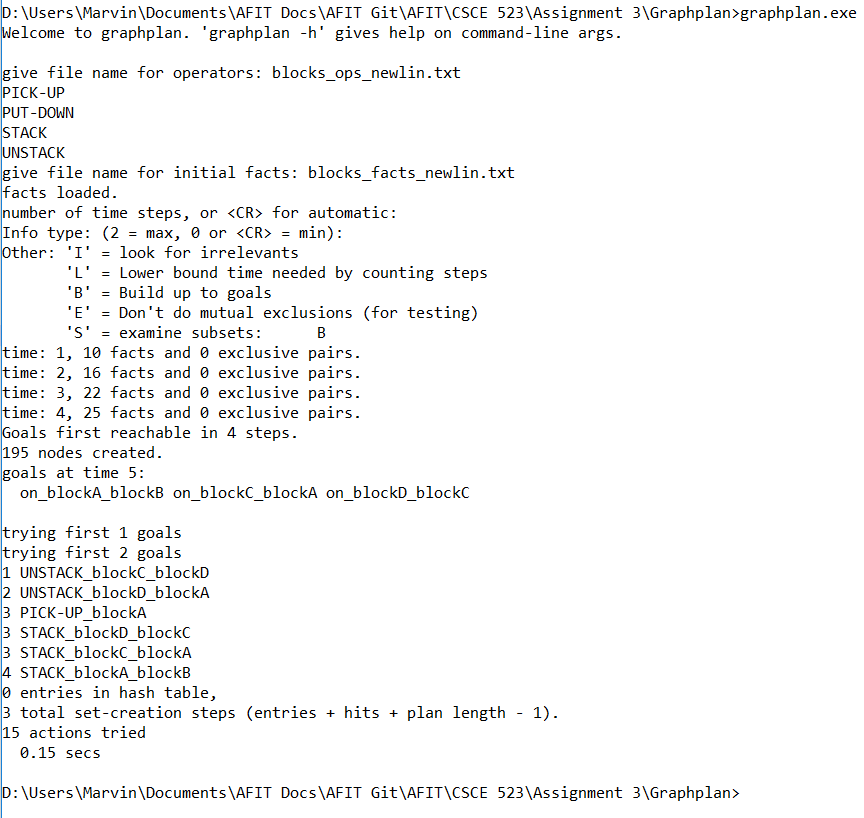
8. (25 points) Use the Graphplan planning algorithm to solve the blocks world planning problem shown in Figure 1 and the Rush Hour problems in Figures 2 and 3. The executable, fact, and operator files for the blocks domain are in the course directory. You must modify the fact file to solve blocks world planning problem shown in Figure 1. And write your own fact file and operator file for the two Rush Hour problems shown in Figures 2 and 3. Assume there are no trucks only cars of length two. Define actions as a movement of a car one square north, south, east, or west depending on orientation. During execution of Graphplan, respond to the prompts to perform automatic time steps, and information, and for other hit ‘B’ for build until goals. Note your operator file (for Rush Hour) should be the same for both problems. The only part that should be different is the initial condition in the fact file.

**Sol’n:**

**Blocks World**

The domain files for the blocks world problem are called blocks\_facts\_newlin.txt and blocks\_ops\_newlin.txt.

The results of running graph plan on the blocks world domain files is shown below.



**Rush Hour**

The domain files for Rush Hour are named rushHour1\_facts.txt, rushHour2\_facts.txt, and rushHour\_ops.txt.

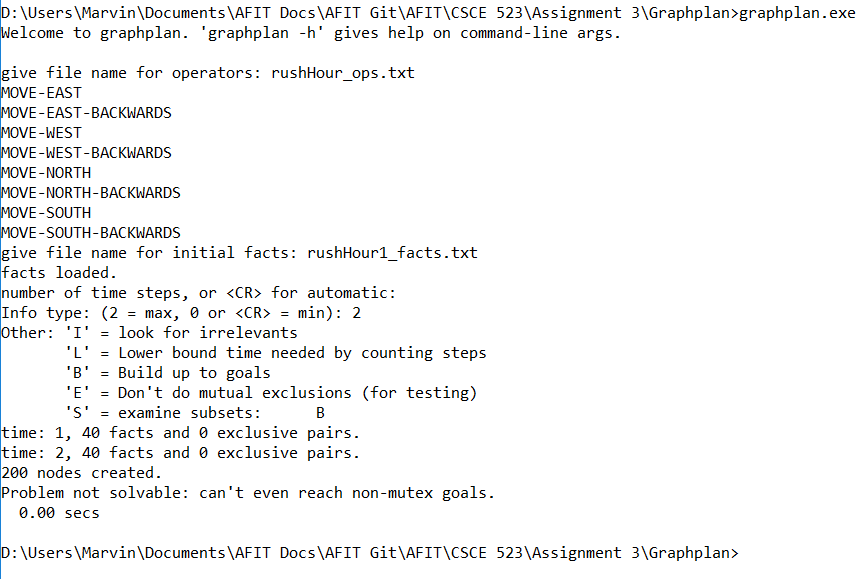
**Discussion**

I was never able to get the Rush Hour files in graphplan to run correctly. The below picture shows the results of running my files in graph plan. It appears that the facts were generated correctly but graphplan was not able to solve the problem.

Despite not getting it to run, comparing this implementation of Rush Hour to the original one we implemented in assignment 1, I would prefer the search problem over the planning problem. Most of this is due to my personal experience in dealing with both search problems and Java so I felt much more comfortable writing Java code. From a coding perspective, Graphplan is nearly impossible to debug as far as I can tell so it was much more difficult to use on this assignment having never seen it before.

Personally, I think that the search problem makes the most sense for this domain because especially as the puzzles increase in complexity, the overhead of the planning problem becomes more than the search problem so it makes more sense to use an informed search technique to solve the Rush Hour puzzle.

rushHour1\_facts.txt



rushHour2\_facts.txt

