# Problem #9.10

**A popular children’s riddle is “Brothers and sisters have I none, but that man’s father is my father’s son.” Use the rules of family domain (Section 8.3.2 on page 301) to show who that man is. You may apply any of the inference methods described in this chapter. Why do you think that this riddle is difficult?**

This problem in prenex normal form is:

This is in conjunctive normal form.

The existential quantifier can be removed by making into a function of and . Hence:

Since there are only universal quantifiers remaining, these can be dropped resulting in:

If a person is the son of person b (i.e. is true), then the relation can be rewritten:

This is still a binary expression since if a different constant other than was inside the function, then the relation may not be true. This substitution allows us to rewrite the riddle expression as:

Note the requirement of the s being is not included for brevity.

Any person only has one ; what is more, I have no siblings who could also have the same father. Hence, the outer functions can be dropped. This simplifies the equation to:

Using the previously described relation, the last clause can be written as:

Therefore, (which was the original ) is simply the my son (by substitution).

This riddle is not terribly difficult. However, it obfuscates by wrapping the object in what are

complementary operations since has no brothers.

# Problem #9.23

**From “Horses are animals,” it follows that “The head of a horse is the head of an animal.” Demonstrate that this inference is valid by carrying out the following steps:**

1. **Translate the premise and the conclusion into the language of first order logic. Use three predicates: (meaning “ is the head of ”), , and .**

The premise of this statement is “Horses are animals”. Rewritten in first-order logic with the defined predicates, this statement is:

The conclusion of this statement is:

1. **Negate the conclusion, and convert the premise and the negated conclusion into conjunctive normal form.**

By definition:

To perform refutation, negate the conclusion and show that:

The premise is already in prenex normal form so the quantifiers can be dropped resulting in:

This can be made into a single clause through implication elimination.

In the conclusion, the existential quantifier can be replaced by making a function of and (i.e. ). Hence, the conclusion becomes:

Again, since all variables are bounded by a universal quantifier, the universal quantifier(s) can be dropped making the statement:

When implication elimination is applied to this equation, the result is:

To perform resolution refutation, the conclusion is negated. This results in:

The conjunction of the premise and the negation of the conclusion is taken. It results in:

This is in CNF format.

1. **Use resolution to show that the conclusion follows from the premise.**

Unification involves applying substitutions to the clauses in an expression in order to use resolution.

**Step #1:** Apply substitution . This simplifies the expression to:

**Step #2:** The second and fourth clauses can be resolved to achieve the new clause:

**Step #3:** Apply substitution . This simplifies the expression to:

**Step #4:** The first and third clauses can be combined to achieve the new clause:

**Step #5:** The clauses from step #2 and step #5 resolve to the empty set proving this statement by resolution.

# Additional Problem #1

**Draw the planning graph for the problem in figure 10.3 in the book. Solve the problem step-by-step using the GraphPlan algorithm.**

The predicates and functions in this relation are shown below.

Predicates:

* – A predicate for whether is a block.
* – A predicate for whether block is on top of , where can be another block or the .
* – A predicate for whether there is a clear space above block where another block could be placed.

Actions:

* – Moves block from to .
* – Moves block from block to the .

# Additional Problem #2

**Briefly explain how PDDL solves the frame problem. Given some disadvantages of formulating problems in PDDL.**

As with the previous definition of a search problem, the four core items that that the Planning Domain Definition Language (PDDL) utilizes are:

1. Initial State
2. Actions available in each state
3. Result of applying an action
4. Goal Test

A state is a conjunction of fluents (i.e. facts that my change from situation to situation). The fluents are ground in that they do not rely on variables.

When performing an action, the result must explicitly define those aspects of the state that changed and those which stayed the same. The frame problem encapsulates the issue of defining what stayed the same.

By definition, classical planning focuses on those types of problems where most aspects of a state do not change when an action is performed. As such, for each action, PDDL only enumerates those aspects of the state that change. Any unmentioned aspect of the state is unchanged by the action.

The **qualification problem** in planning systems is that there are some aspects of the environment that may cause an action to fail. What is more, these implicit and necessary preconditions for the success of an action can be innumerable and unknowable for practical purposes. For example, the action in the “Air Cargo Transport” problem requires sufficient fuel in the tank and a competent pilot or the action will fail. However, the textbook’s PDDL description of this action does not capture this fact.

The **ramification problem** is another limitation of PDDL. When performing an action, there are many secondary effects that are not captured by the PDDL description. For example, when the action is performed, some of the airline’s gasoline reserve is consumed. Moreover, after a action, in addition to the movement of a package, some airline staff as well as possibly customers are moved to a new location. However, this is not captured by the PDDL description.

PDDL fluents also do not explicitly include time. While preconditions refer to a time and effects to a time , this discretized representation of time will not be sufficient for all types of problems. Scheduling problems require information about time including how long an action takes and when it occurs. For example, with the “Air Cargo Transport” problem, actions can be ordered, but the PDDL representation has no sense of things like departure and arrival times of the aircraft.