Christopher Raleigh:

Tony Ferrero:

CS 156 Assignment 4

9.10:

i = riddle teller

(¬∃b,s Brother(b, i) ∨ Sister(s, i)) ∧ (∃m Son(Father(m), Father(i)))

(∀b,s ¬Brother(b, i) ∧ ¬Sister(s, i)) ∧ (∃m Son(Father(m), Father(i)))

∀b∀s∃m ¬Brother(b, i) ∧ ¬Sister(s, i) ∧ Son(Father(m), Father(i)) **(Prenex Normal Form)**

∀b∀s∃m Male(b) ∧ Female(s) ∧ ¬Sibling(b, i) ∧ ¬Sibling(s, i) ∧ Son(Father(m), Father(i))

∀b∀s∃m ((s ≠ i) ∧ (b ≠ i)) → (Male(b) ∧ Female(s) ∧ (Father(s) ≠ Father(i)) ∧ (Father(b) ≠ Father(i)) ∧ Son(Father(m), Father(i)))

∀b∀s∃m ((s = i) ∨ (b = i)) ∨ ((Father(s) ≠ Father(i)) ∧ (Father(b) ≠ Father(i) ∧ (i = Father(m)) ∧ Son(Father(m), Father(i)) ∧ Male(b) ∧ Female(s))

∀b∀s∃m ((s = i) ∨ (b = i)) ∨ (Son(i, Father(i)) ∧ (i = Father(m)) ∧ Male(b) ∧ Female(s))

∀b∀s∃m ((s = i) ∨ (b = i)) ∨ (Son(i, Father(i)) ∧ (i = Father(m)))

Son(i, Father(i)) ∧ (i = Father(m))

i = Father(m)

Son(m, i)

The man (m) is the riddle teller's (i) son.

This riddle may confuse children, and even adults, because "That man's father is my father's son," can be confused as "That man is my father's son." Two relationships are being compared (Father(m) and Father(i)), not a person (m) and one relationship Father(i). People process direct comparisons better than indirect ones.

9.23:

a)

Premise:

Horse(x) → Animal(x)

Conclusion:

HeadOf(h, Horse(x)) → HeadOf(h, Animal(x))

b)

Premise:

Horse(x) → Animal(x)

¬(Horse(x) ∧ ¬Animal(x))

¬Horse(x) ∨ Animal(x)

Conclusion:

HeadOf(h, Horse(x)) → HeadOf(h, Animal(x))

¬HeadOf(h, Animal(x)) → ¬HeadOf(h, Horse(x)) **(Negation)**

HeadOf(h, Animal(x)) ∨ ¬HeadOf(h, Horse(x))

c)

Conclusion:

HeadOf(h, Animal(x)) ∨ ¬HeadOf(h, Horse(x))

Resolution:

(Horse(x) → Animal(x)) ∧ (HeadOf(h, Horse(x)) → HeadOf(h, Animal(x)))

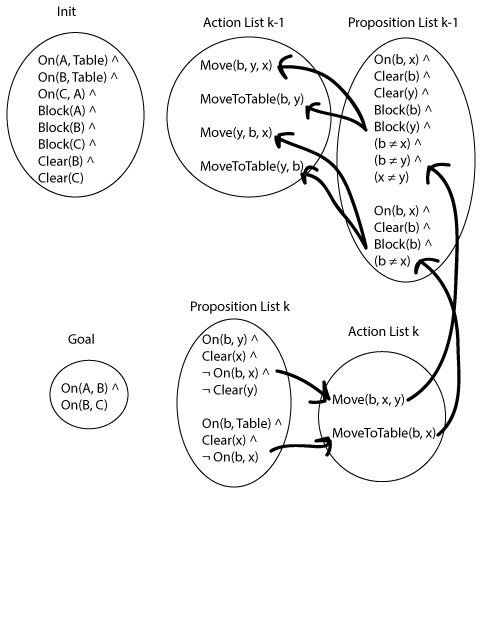
HeadOf(h, Animal(x)) → HeadOf(h, Animal(x)) (Replace "Horse(x)" with "Animal(x)", because "Horse(x) → Animal(x)".)

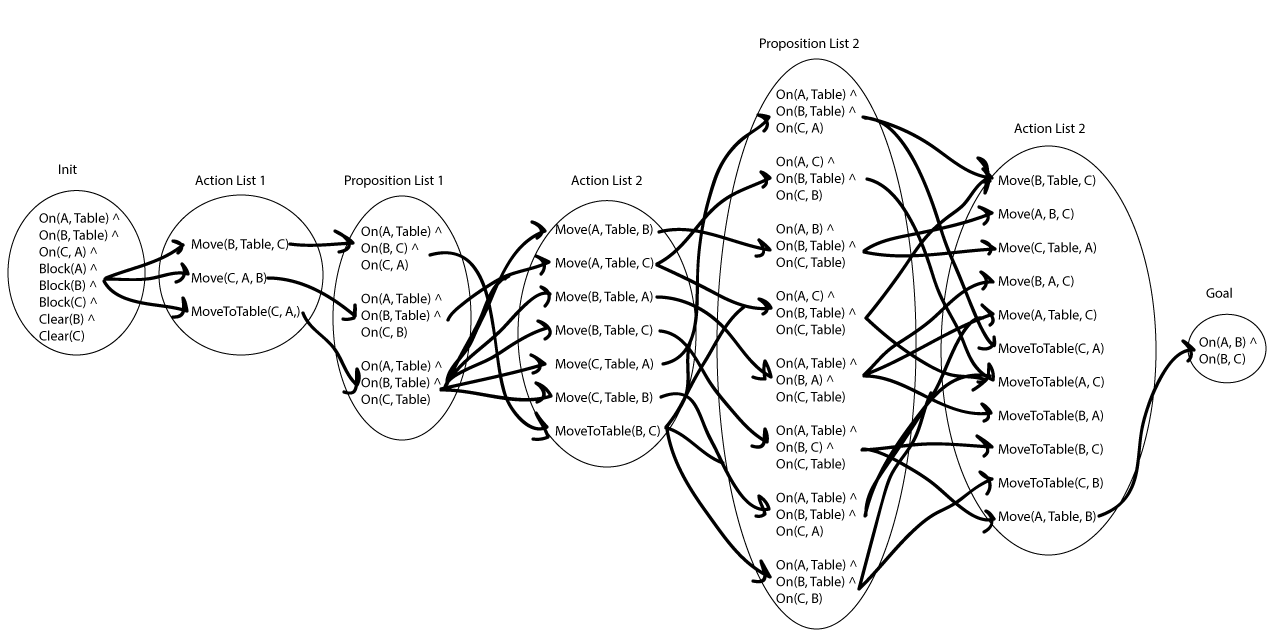
¬HeadOf(h, Animal(x)) ∨ HeadOf(h, Animal(x))

¬t ∨ t

True

Additional Problem 1





The answer, as shown in the book, is **[MoveToTable(C, A), Move(B, Table, C), Move(A, Table, B)]**.

Explanation:

The precondition for Move is:

On(b, x) ∧ Clear(b) ∧ Clear(y) ∧ Block(b) ∧ Block(y) ∧ (b ≠ x) ∧ (b ≠ y) ∧ (x ≠ y)

The precondition for MoveToTable is:

On(b, x) ∧ Clear(b) ∧ Block(b) ∧ (b ≠ x)

The set of Move's preconditions that are not MoveToTable's preconditions is:

Clear(y) ∧ Block(y) ∧ (b ≠ y) ∧ (x ≠ y)

If this complement is false, then MoveToTable must be performed instead. Otherwise, Move can be performed.

¬(Clear(y) ∧ Block(y) ∧ (b ≠ y) ∧ (x ≠ y)) → MoveToTable(b, x)

(Clear(y) ∧ Block(y) ∧ (b ≠ y) ∧ (x ≠ y)) ∨ MoveToTable(b, x)

Move is then added to the left set of conditions, which satisfy it.

(Move(b, x, y) ∧ Clear(y) ∧ Block(y) ∧ (b ≠ y) ∧ (x ≠ y)) ∨ MoveToTable(b, x)

Adding back MoveToTable's preconditions, the final formula is:

∃b∃x∃y (On(b, x) ∧ Clear(b) ∧ Block(b) ∧ (b ≠ x)) ∧ ((Move(b, x, y) ∧ Clear(y) ∧ Block(y) ∧ (b ≠ y) ∧ (x ≠ y)) ∨ MoveToTable(b, x))

Additional Problem 2:

PDDL has the notion of a state, which is a conjunction of fluents, and actions, which are described as having effects. Effects are conjunctions of literals that are asserted as true after the action takes place. PDDL solves the frame problem by mentioning what changes as a result of an action in the action's effect sentence, and assuming that nothing else about the state changes, keeping track of what doesn't change through the state PDDL mantains.

PDDL has a few disadvantages. It's not that expressive. It does not have quanitifiers like FOL, so expressing some things is either impossible or very verbose. For example, trying to express a goal involving a variable number of objects is not possible. PDDL requires that all objects involved in the problem be mentioned explicitly. There are also a lot of problems that when formulated in PDDL are NP-Hard to solve.

Additional Problem 3:

Republican(Nixon) Λ Quaker(Nixon)

Republican(x) : Warlike(x) Λ ¬Moderate(x) Λ ¬Pacifist(x)/Warlike(x) Λ ¬Moderate(x) Λ ¬Pacifist(x)

Quaker(x) : ¬Warlike(x) Λ ¬Moderate(x) Λ Pacifist(x)/¬Warlike(x) Λ ¬Moderate(x) Λ Pacifist(x)

Republican(x) Λ Quaker(x) : ¬Warlike(x) Λ Moderate(x) Λ ¬Pacifist(x)/¬Warlike(x) Λ Moderate(x) Λ ¬Pacifist(x)

Let W, M, and P be the three possible extensions of the system.

W = {Republican(Nixon) Λ Quaker(Nixon), Warlike(x) Λ ¬Moderate(x) Λ ¬Pacifist(x)}

The elements of the extension are trivially consistent with each other, as the two sentences in it share no members nor does the second element contain a contradiction by virtue of the fact that it does not contain a literal and the negation of the same literal in it.

By repeated and-elimination on default rule 1's conclusion Warlike(x) Λ ¬Moderate(x) Λ ¬Pacifist(x), we can derive Warlike(x), which is inconsistent with the justification of both default rules 2 and 3 because the literal ¬Warlike(x) appears in both. Therefore, W is a maximal set of consequences of the default rules.

M = {Republican(Nixon) Λ Quaker(Nixon), ¬Warlike(x) Λ ¬Moderate(x) Λ Pacifist(x)}

The elements of the extension are trivially consistent, as above.

Using the same kind of and-elimination on rule 2's default rules, we can derive from rule 2's conclusion ¬Warlike(x) Λ ¬Moderate(x) Λ Pacifist(x) the literal Pacifist(x). Pacifist(x) is negated in both of the justifications for the other default rules, so those default rules cannot be applied. Thus, M is a maximal set of consequences of the theory.

P = {Republican(Nixon) Λ Quaker(Nixon), ¬Warlike(x) Λ Moderate(x) Λ ¬Pacifist(x)}

The elements of the extension are trivially consistent, as above.

Repeated and-elimination on ¬Warlike(x) Λ Moderate(x) Λ ¬Pacifist(x) allows us to derive Moderate(x), which appears negated in the conjunctions that are the justifications of the other default rules. So P is a maximal set of consequences of the theory.