logical-inference-exercises

March 26, 2019

1 9. Inference in First-Order Logic

9.1 Prove that Universal Instantiation is sound and that Existential Instantiation produces an inferentially equivalent knowledge base.

9.2 From $Likes(Jerry, IceCream)$ it seems reasonable to infer $\exists\, x\ \ Likes(x, IceCream)$. Write down a general inference rule, , that sanctions this inference. State carefully the conditions that must be satisfied by the variables and terms involved.

9.3 Suppose a knowledge base contains just one sentence, $\exists\, x\ AsHighAs(x, Everest)$. Which of the following are legitimate results of applying Existential Instantiation?

1. $AsHighAs(Everest, Everest)$.
2. $AsHighAs(Kilimanjaro, Everest)$.
3. $AsHighAs(Kilimanjaro, Everest) \land AsHighAs(BenNevis, Everest)$ (after two applications).

9.4 For each pair of atomic sentences, give the most general unifier if it exists:

1. $P(A, B, B)$, $P(x, y, z)$.
2. $Q(y, G(A, B))$, $Q(G(x, x), y)$.
3. $Older(Father(y), y)$, $Older(Father(x), John)$.
4. $Knows(Father(y), y)$, $Knows(x, x)$.

9.5 For each pair of atomic sentences, give the most general unifier if it exists:

1. $P(A, A, B)$, $P(x, y, z)$.
2. $Q(y, G(A, B))$, $Q(G(x, x), y)$.
3. $Older(Father(y), y)$, $Older(Father(x), Jerry)$.
4. $Knows(Father(y), y)$, $Knows(x, x)$.

9.6 [subsumption-lattice-exercise]Consider the subsumption lattices shown in Figure Section **??** (page Section **??**).

1. Construct the lattice for the sentence $Employs(Mother(John), Father(Richard))$.

2. Construct the lattice for the sentence $Employs(IBM, y)$ ("Everyone works for IBM"). Remember to include every kind of query that unifies with the sentence.

3. Assume that indexes each sentence under every node in its subsumption lattice. Explain how should work when some of these sentences contain variables; use as examples the sentences in (a) and (b) and the query $Employs(x, Father(x))$.

9.7 [fol-horses-exercise]Write down logical representations for the following sentences, suitable for use with Generalized Modus Ponens:

1. Horses, cows, and pigs are mammals.

2. An offspring of a horse is a horse.

3. Bluebeard is a horse.

4. Bluebeard is Charlie's parent.

5. Offspring and parent are inverse relations.

6. Every mammal has a parent.

9.8 These questions concern concern issues with substitution and Skolemization.

1. Given the premise $\forall x \;\; \exists y \;\; P(x, y)$, it is not valid to conclude that $\exists q \;\; P(q, q)$. Give an example of a predicate P where the first is true but the second is false.

2. Suppose that an inference engine is incorrectly written with the occurs check omitted, so that it allows a literal like $P(x, F(x))$ to be unified with $P(q, q)$. (As mentioned, most standard implementations of Prolog actually do allow this.) Show that such an inference engine will allow the conclusion $\exists y \;\; P(q, q)$ to be inferred from the premise $\forall x \;\; \exists y \;\; P(x, y)$.

3. Suppose that a procedure that converts first-order logic to clausal form incorrectly Skolemizes $\forall x \;\; \exists y \;\; P(x, y)$ to $P(x, Sk0)$—that is, it replaces y by a Skolem constant rather than by a Skolem function of x. Show that an inference engine that uses such a procedure will likewise allow $\exists q \;\; P(q, q)$ to be inferred from the premise $\forall x \;\; \exists y \;\; P(x, y)$.

4. A common error among students is to suppose that, in unification, one is allowed to substitute a term for a Skolem constant instead of for a variable. For instance, they will say that the formulas $P(Sk1)$ and $P(A)$ can be unified under the substitution $\{Sk1/A\}$. Give an example where this leads to an invalid inference.

9.9 This question considers Horn KBs, such as the following:

$$\begin{array}{l} P(F(x)) \;\Rightarrow\; P(x) \\ Q(x) \;\Rightarrow\; P(F(x)) \\ P(A) \\ Q(B) \end{array}$$

Let FC be a breadth-first forward-chaining algorithm that repeatedly adds all consequences of currently satisfied rules; let BC be a depth-first left-to-right backward-chaining algorithm that tries clauses in the order given in the KB. Which of the following are true?

1. FC will infer the literal $Q(A)$.
2. FC will infer the literal $P(B)$.
3. If FC has failed to infer a given literal, then it is not entailed by the KB.
4. BC will return *true* given the query $P(B)$.
5. If BC does not return *true* given a query literal, then it is not entailed by the KB.

9.10 [csp-clause-exercise]Explain how to write any given 3-SAT problem of arbitrary size using a single first-order definite clause and no more than 30 ground facts.

9.11 Suppose you are given the following axioms:

1. $0 \leq 3$.
2. $7 \leq 9$.
3. $\forall x \quad x \leq x$.
4. $\forall x \quad x \leq x + 0$.
5. $\forall x \quad x + 0 \leq x$.
6. $\forall x, y \quad x + y \leq y + x$.
7. $\forall w, x, y, z \quad w \leq y \land x \leq z \;\Rightarrow\; w + x \leq y + z$.
8. $\forall x, y, z \quad x \leq y \land y \leq z \;\Rightarrow\; x \leq z$

1. Give a backward-chaining proof of the sentence $7 \leq 3 + 9$. (Be sure, of course, to use only the axioms given here, not anything else you may know about arithmetic.) Show only the steps that leads to success, not the irrelevant steps.
2. Give a forward-chaining proof of the sentence $7 \leq 3 + 9$. Again, show only the steps that lead to success.

9.12 Suppose you are given the following axioms:

1. $0 \leq 4$.
2. $5 \leq 9$.
3. $\forall x \quad x \leq x$.
4. $\forall x \quad x \leq x + 0$.
5. $\forall x \quad x + 0 \leq x$.
6. $\forall x, y \quad x + y \leq y + x$.
7. $\forall w, x, y, z \quad w \leq y \land x \leq z \;\Rightarrow\; w + x \leq y + z$.
8. $\forall x, y, z \quad x \leq y \land y \leq z \;\Rightarrow\; x \leq z$

1. Give a backward-chaining proof of the sentence $5 \leq 4 + 9$. (Be sure, of course, to use only the axioms given here, not anything else you may know about arithmetic.) Show only the steps that leads to success, not the irrelevant steps.

2. Give a forward-chaining proof of the sentence $5 \leq 4+9$. Again, show only the steps that lead to success.

9.13 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 the family domain (Section Section **??** on page Section **??**) 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?

9.14 Suppose we put into a logical knowledge base a segment of the U.S. census data listing the age, city of residence, date of birth, and mother of every person, using social security numbers as identifying constants for each person. Thus, George's age is given by $Age(\text{443-65-1282}, 56)$. Which of the following indexing schemes S1–S5 enable an efficient solution for which of the queries Q1–Q4 (assuming normal backward chaining)?

- **S1**: an index for each atom in each position.
- **S2**: an index for each first argument.
- **S3**: an index for each predicate atom.
- **S4**: an index for each *combination* of predicate and first argument.
- **S5**: an index for each *combination* of predicate and second argument and an index for each first argument.
- **Q1**: $Age(443-44-4321, x)$
- **Q2**: $ResidesIn(x, Houston)$
- **Q3**: $Mother(x, y)$
- **Q4**: $Age(x, 34) \land ResidesIn(x, TinyTownUSA)$

9.15 [standardize-failure-exercise]One might suppose that we can avoid the problem of variable conflict in unification during backward chaining by standardizing apart all of the sentences in the knowledge base once and for all. Show that, for some sentences, this approach cannot work. (*Hint*: Consider a sentence in which one part unifies with another.)

9.16 In this exercise, use the sentences you wrote in Exercise Section **??** to answer a question by using a backward-chaining algorithm.

1. Draw the proof tree generated by an exhaustive backward-chaining algorithm for the query $\exists\, h \;\; Horse(h)$, where clauses are matched in the order given.

2. What do you notice about this domain?

3. How many solutions for h actually follow from your sentences?

4. Can you think of a way to find all of them? (*Hint*: See @Smith+al:1986.)

9.17 [bc-trace-exercise]Trace the execution of the backward-chaining algorithm in Figure Section **??** (page Section **??**) when it is applied to solve the crime problem (page Section **??**). Show the sequence of values taken on by the *goals* variable, and arrange them into a tree.

9.18 The following Prolog code defines a predicate P. (Remember that uppercase terms are variables, not constants, in Prolog.)

```
P(X,[X|Y]).
P(X,[Y|Z]) :- P(X,Z).
```

1. Show proof trees and solutions for the queries P(A,[2,1,3]) and P(2,[1,A,3]).

2. What standard list operation does P represent?

9.19 The following Prolog code defines a predicate P. (Remember that uppercase terms are variables, not constants, in Prolog.)

```
P(X,[X|Y]).
P(X,[Y|Z]) :- P(X,Z).
```

1. Show proof trees and solutions for the queries P(A,[1,2,3]) and P(2,[1,A,3]).

2. What standard list operation does P represent?

9.20 This exercise looks at sorting in Prolog.

1. Write Prolog clauses that define the predicate sorted(L), which is true if and only if list L is sorted in ascending order.

2. Write a Prolog definition for the predicate perm(L,M), which is true if and only if L is a permutation of

   M.

3. Define sort(L,M) (M is a sorted version of

   L) using perm and sorted.

4. Run sort on longer and longer lists until you lose patience. What is the time complexity of your program?

5. Write a faster sorting algorithm, such as insertion sort or quicksort, in Prolog.

9.21 [diff-simplify-exercise]This exercise looks at the recursive application of rewrite rules, using logic programming. A rewrite rule (or **demodulator** in terminology) is an equation with a specified direction. For example, the rewrite rule $x + 0 \rightarrow x$ suggests replacing any expression that matches $x + 0$ with the expression x. Rewrite rules are a key component of equational reasoning systems. Use the predicate rewrite(X,Y) to represent rewrite rules. For example, the earlier rewrite rule is written as rewrite(X+0,X). Some terms are *primitive* and cannot be further simplified; thus, we write primitive(0) to say that 0 is a primitive term.

1. Write a definition of a predicate simplify(X,Y), that is true when Y is a simplified version of X—that is, when no further rewrite rules apply to any subexpression of Y.

2. Write a collection of rules for the simplification of expressions involving arithmetic operators, and apply your simplification algorithm to some sample expressions.

3. Write a collection of rewrite rules for symbolic differentiation, and use them along with your simplification rules to differentiate and simplify expressions involving arithmetic expressions, including exponentiation.

9.22 This exercise considers the implementation of search algorithms in Prolog. Suppose that successor(X,Y) is true when state Y is a successor of state X; and that goal(X) is true when X is a goal state. Write a definition for solve(X,P), which means that P is a path (list of states) beginning with X, ending in a goal state, and consisting of a sequence of legal steps as defined by successor. You will find that depth-first search is the easiest way to do this. How easy would it be to add heuristic search control?

9.23 Suppose a knowledge base contains just the following first-order Horn clauses:

$$Ancestor(Mother(x), x)$$

$$Ancestor(x, y) \land Ancestor(y, z) \implies Ancestor(x, z)$$

Consider a forward chaining algorithm that, on the jth iteration, terminates if the KB contains a sentence that unifies with the query, else adds to the KB every atomic sentence that can be inferred from the sentences already in the KB after iteration $j-1$.

1. For each of the following queries, say whether the algorithm will (1) give an answer (if so, write down that answer); or (2) terminate with no answer; or (3) never terminate.

   1. $Ancestor(Mother(y), John)$
   2. $Ancestor(Mother(Mother(y)), John)$
   3. $Ancestor(Mother(Mother(Mother(y))), Mother(y))$
   4. $Ancestor(Mother(John), Mother(Mother(John)))$

2. Can a resolution algorithm prove the sentence $\neg Ancestor(John, John)$ from the original knowledge base? Explain how, or why not.

3. Suppose we add the assertion that $\neg(Mother(x) = x)$ and augment the resolution algorithm with inference rules for equality. Now what is the answer to (b)?

9.24 Let $\mathcal{L}$ be the first-order language with a single predicate $S(p, q)$, meaning "p shaves q." Assume a domain of people.

1. Consider the sentence "There exists a person P who shaves every one who does not shave themselves, and only people that do not shave themselves." Express this in $\mathcal{L}$.

2. Convert the sentence in (a) to clausal form.

3. Construct a resolution proof to show that the clauses in (b) are inherently inconsistent. (Note: you do not need any additional axioms.)

9.25 How can resolution be used to show that a sentence is valid? Unsatisfiable?

9.26 Construct an example of two clauses that can be resolved together in two different ways giving two different outcomes.

9.27 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: $HeadOf(h, x)$ (meaning "h is the head of x"), $Horse(x)$, and $Animal(x)$.

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

3. Use resolution to show that the conclusion follows from the premise.

9.28 From "Sheep are animals," it follows that "The head of a sheep 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: $HeadOf(h, x)$ (meaning "h is the head of x"), $Sheep(x)$, and $Animal(x)$.

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

3. Use resolution to show that the conclusion follows from the premise.

9.29 [quantifier-order-exercise]Here are two sentences in the language of first-order logic:

- **(A)** $\forall x \ \ \exists y \ \ (x \geq y)$
- **(B)** $\exists y \ \ \forall x \ \ (x \geq y)$

1. Assume that the variables range over all the natural numbers $0, 1, 2, \ldots, \infty$ and that the "$\geq$" predicate means "is greater than or equal to." Under this interpretation, translate (A) and (B) into English.

2. Is (A) true under this interpretation?

3. Is (B) true under this interpretation?

4. Does (A) logically entail (B)?

5. Does (B) logically entail (A)?

6. Using resolution, try to prove that (A) follows from (B). Do this even if you think that (B) does not logically entail (A); continue until the proof breaks down and you cannot proceed (if it does break down). Show the unifying substitution for each resolution step. If the proof fails, explain exactly where, how, and why it breaks down.

7. Now try to prove that (B) follows from (A).

9.30 Resolution can produce nonconstructive proofs for queries with variables, so we had to introduce special mechanisms to extract definite answers. Explain why this issue does not arise with knowledge bases containing only definite clauses.

9.31 We said in this chapter that resolution cannot be used to generate all logical consequences of a set of sentences. Can any algorithm do this?