EXERCISES 8

FIRST-ORDER LOGIC

8.1 Representation Revisited

Exercise 8.LKNB

A logical knowledge base represents the world using a set of sentences with no explicit structure. An **analogical** representation, on the other hand, has physical structure that corresponds directly to the structure of the thing represented. Consider a road map of your country as an analogical representation of facts about the country—it represents facts with a map language. The two-dimensional structure of the map corresponds to the two-dimensional surface of the area.

a. Give five examples of *symbols* in the map language.

b. An *explicit* sentence is a sentence that the creator of the representation actually writes down. An *implicit* sentence is a sentence that results from explicit sentences because of properties of the analogical representation. Give three examples each of *implicit* and *explicit* sentences in the map language.

c. Give three examples of facts about the physical structure of your country that cannot be represented in the map language.

d. Give two examples of facts that are much easier to express in the map language than in first-order logic.

e. Give two other examples of useful analogical representations. What are the advantages and disadvantages of each of these languages?

This question will generate a wide variety of possible solutions. The key distinction between analogical and sentential representations is that the analogical representation automatically generates consequences that can be "read off" whenever suitable premises are encoded. When you get into the details, this distinction turns out to be quite hard to pin down—for example, what does "read off" mean?—but it can be justified by examining the time complexity of various inferences on the "virtual inference machine" provided by the representation system.

a. Depending on the scale and type of the map, symbols in the map language typically include city and town markers, road symbols (various types), lighthouses, historic monuments, river courses, freeway intersections, etc.

b. Explicit and implicit sentences: this distinction is a little tricky, but the basic idea is that when the map-drawer plunks a symbol down in a particular place, he says one explicit thing (e.g. that Coit Tower is here), but the analogical structure of the map representation means that many implicit sentences can now be derived. Explicit sentences: there is a monument called Coit Tower at this location; Lombard Street runs (approximately) east-west; San Francisco Bay exists and has this shape. Implicit sentences: Van Ness is longer than North Willard; Fisherman's Wharf is north of the Mission District; the shortest drivable route from Coit Tower to Twin Peaks is the following . . ..

c. Sentences unrepresentable in the map language: Telegraph Hill is approximately conical and about 430 feet high (assuming the map has no topographical notation); in 1890 there was no bridge connecting San Francisco to Marin County (map does not represent changing information); Interstate 680 runs either east or west of Walnut Creek (no disjunctive information).

d. Sentences that are easier to express in the map language: any sentence that can be written easily in English is not going to be a good candidate for this question. Any *linguistic* abstraction from the physical structure of San Francisco (e.g. San Francisco is on the end of a peninsula at the mouth of a bay) can probably be expressed equally easily in the predicate calculus, since that's what it was designed for. Facts such as the shape of the coastline, or the path taken by a road, are best expressed in the map language. Even then, one can argue that the coastline drawn on the map actually consists of lots of individual sentences, one for each dot of ink, especially if the map is drawn using a digital plotter. In this case, the advantage of the map is really in the ease of inference combined with suitability for human "visual computing" apparatus.

e. Examples of other analogical representations:

- Analog audio tape recording. Advantages: simple circuits can record and reproduce sounds. Disadvantages: subject to errors, noise; hard to process in order to separate sounds or remove noise etc.
- Traditional clock face. Advantages: easier to read quickly, determination of how much time is available requires no additional computation. Disadvantages: hard to read precisely, cannot represent small units of time (ms) easily.
- All kinds of graphs, bar charts, pie charts. Advantages: enormous data compression, easy trend analysis, communicate information in a way which we can interpret easily. Disadvantages: imprecise, cannot represent disjunctive or negated information.

8.2 Syntax and Semantics of First-Order Logic

Exercise 8.KNBT

Consider a knowledge base containing just two sentences: $P(a)$ and $P(b)$. Does this knowledge base entail $\forall x\ P(x)$? Explain your answer in terms of models.

The knowledge base does not entail $\forall x\ P(x)$. To show this, we must give a model where $P(a)$ and $P(b)$ but $\forall x\ P(x)$ is false. Consider any model with three domain elements, where a and b refer to the first two elements and the relation referred to by P holds only for those two elements.

Exercise 8.VALD

Is the sentence $\exists\, x, y \;\; x = y$ valid? Explain.

The sentence $\exists\, x, y \;\; x = y$ is valid. A sentence is valid if it is true in every model. An existentially quantified sentence is true in a model if it holds under any extended interpretation in which its variables are assigned to domain elements. According to the standard semantics of FOL as given in the chapter, every model contains at least one domain element, hence, for any model, there is an extended interpretation in which x and y are assigned to the first domain element. In such an interpretation, $x = y$ is true.

Exercise 8.SENO

Write down a logical sentence such that every world in which it is true contains exactly one object.

$\forall\, x, y \;\; x = y$ stipulates that there is exactly one object. If there are two objects, then there is an extended interpretation in which x and y are assigned to different objects, so the sentence would be false. Some students may also notice that any unsatisfiable sentence also meets the criterion, since there are no worlds in which the sentence is true.

Exercise 8.SENT

Write down a logical sentence such that every world in which it is true contains exactly two objects.

$\exists\, x, y \;\; x \neq y \land \forall\, z \;\; x = z \lor y = z$ stipulates that there are exactly two objects.

Exercise 8.MCNT

Consider a symbol vocabulary that contains c constant symbols, p_k predicate symbols of each arity k, and f_k function symbols of each arity k, where $1 \leq k \leq A$. Let the domain size be fixed at D. For any given model, each predicate or function symbol is mapped onto a relation or function, respectively, of the same arity. You may assume that the functions in the model allow some input tuples to have no value for the function (i.e., the value is the invisible object). Derive a formula for the number of possible models for a domain with D elements. Don't worry about eliminating redundant combinations.

We will use the simplest counting method, ignoring redundant combinations. For the constant symbols, there are D^c assignments. Each predicate of arity k is mapped onto a k-ary relation, i.e., a subset of the D^k possible k-element tuples; there are 2^{D^k} such mappings. Each function symbol of arity k is mapped onto a k-ary function, which specifies a value for each of the D^k possible k-element tuples. Including the invisible element, there are $D + 1$ choices for each value, so there are $(D + 1)^{D^k}$ functions. The total number of possible combinations is therefore

$$D^c \cdot \left(\sum_{k=1}^{A} 2^{D^k}\right) \cdot \left(\sum_{k=1}^{A} (D+1)^{D^k}\right) .$$

Two things to note: first, the number is finite; second, the maximum arity A is the most crucial complexity parameter.

Exercise 8.VALS

Which of the following are valid (necessarily true) sentences?

a. $(\exists x\ x=x) \ \Rightarrow\ (\forall y\ \ \exists z\ y=z)$.
b. $\forall x\ \ P(x) \lor \neg P(x)$.
c. $\forall x\ \ Smart(x) \lor (x=x)$.

Validity in first-order logic requires truth in all possible models:

a. $(\exists x\ x=x) \ \Rightarrow\ (\forall y\ \ \exists z\ y=z)$.
Valid. The LHS is valid by itself—in standard FOL, every model has at least one object; hence, the whole sentence is valid iff the RHS is valid. (Otherwise, we can find a model where the LHS is true and the RHS is false.) The RHS is valid because for every value of y in any given model, there is a z—namely, the value of y itself—that is identical to y.

b. $\forall x\ \ P(x) \lor \neg P(x)$.
Valid. For any relation denoted by P, every object x is either in the relation or not in it.

c. $\forall x\ \ Smart(x) \lor (x=x)$.
Valid. In every model, every object satisfies $x=x$, so the disjunction is satisfied regardless of whether x is smart.

Exercise 8.EMPT

Consider a version of the semantics for first-order logic in which models with empty domains are allowed. Give at least two examples of sentences that are valid according to the standard semantics but not according to the new semantics. Discuss which outcome makes more intuitive sense for your examples.

This version of FOL, first studied in depth by Mostowski (1951), goes under the title of **free logic** (Lambert, 1967). By a natural extension of the truth values for empty conjunctions (true) and empty disjunctions (false), every universally quantified sentence is true in empty models and every existentially quantified sentence is false. The semantics also needs to be adjusted to handle the fact that constant symbols have no referent in an empty model.

Examples of sentences valid in the standard semantics but not in free logic include $\exists x\ \ x=x$ and $[\forall x\ \ P(x)] \ \Rightarrow\ [\exists x\ \ P(x)]$. More importantly, perhaps, the equivalence of $\phi \lor \exists x\ \ \psi$ and $\exists x\ \ \phi \lor \psi$ when x does not occur free in ϕ, which is used for putting sentences into CNF, does not hold.

One could argue that $\exists x \;\; x = x$, which simply states that the model is nonempty, is not naturally a valid sentence, and that it ought to be possible to contemplate a universe with no objects. However, experience has shown that free logic seems to require extra work to rule out the empty model in many commonly occurring cases of logical representation and reasoning.

Exercise 8.TORF

True or false? Explain.

a. $\exists x \;\; x = \mathit{Rumpelstiltskin}$ is a valid (necessarily true) sentence of first-order logic.

b. Every existentially quantified sentence in first-order logic is true in any model that contains exactly one object.

c. $\forall x, y \;\; x = y$ is satisfiable.

a. True. In every model, the constant symbol $\mathit{Rumpelstiltskin}$ must have a referent; the extended interpretation in which x is assigned to that referent satisfies the existential sentence.

b. False. We can easily write an existentially quantified sentence that forces there to be at least two objects: $\exists x, y \;\; x \neq y$. We can also write unsatisfiable things about one object: $\exists x \;\; P(x) \land \neg P(x)$. Finally, even the simple sentence $\exists x \;\; P(x)$ is false in the one-object-model where P is the empty relation.

c. True. $\forall x, y \;\; x = y$ is true in any one-object model.

8.3 Using First-Order Logic

Exercise 8.JIMG

Does the fact $\neg \mathit{Spouse}(\mathit{George}, \mathit{Laura})$ follow from the facts $\mathit{Jim} \neq \mathit{George}$ and $\mathit{Spouse}(\mathit{Jim}, \mathit{Laura})$? If so, give a proof; if not, supply additional axioms as needed. What happens if we use Spouse as a unary function symbol instead of a binary predicate?

The fact $\neg \mathit{Spouse}(\mathit{George}, \mathit{Laura})$ does not follow. We need to assert that at most one person can be the spouse of any given person:

$$\forall x, y, z \;\; \mathit{Spouse}(x, z) \land \mathit{Spouse}(y, z) \;\Rightarrow\; x = y \,.$$

With this axiom, a resolution proof of $\neg \mathit{Spouse}(\mathit{George}, \mathit{Laura})$ is straightforward.

If Spouse is a unary function symbol, then the question is whether $\neg \mathit{Spouse}(\mathit{Laura}) = \mathit{George}$ follows from $\mathit{Jim} \neq \mathit{George}$ and $\mathit{Spouse}(\mathit{Laura}) = \mathit{Jim}$. The answer is yes, it does follow. They could not both be the value of the function applied to the same argument if they were different objects.

Exercise 8.MAPC

This exercise uses the function $\mathit{MapColor}$ and predicates $\mathit{In}(x,y)$, $\mathit{Borders}(x,y)$, and $\mathit{Country}(x)$, whose arguments are geographical regions, along with constant symbols for various regions. In each of the following we give an English sentence and a number of candidate logical expressions. For each of the logical expressions, state whether it (1) correctly expresses the English sentence; (2) is syntactically invalid and therefore meaningless; or (3) is syntactically valid but does not express the meaning of the English sentence.

a. Paris and Marseilles are both in France.

(i) $\mathit{In}(\mathit{Paris} \land \mathit{Marseilles}, \mathit{France})$.

(ii) $\mathit{In}(\mathit{Paris}, \mathit{France}) \land \mathit{In}(\mathit{Marseilles}, \mathit{France})$.

(iii) $\mathit{In}(\mathit{Paris}, \mathit{France}) \lor \mathit{In}(\mathit{Marseilles}, \mathit{France})$.

b. There is a country that borders both Iraq and Pakistan.

(i) $\exists c \;\; \mathit{Country}(c) \land \mathit{Border}(c, \mathit{Iraq}) \land \mathit{Border}(c, \mathit{Pakistan})$.

(ii) $\exists c \;\; \mathit{Country}(c) \;\Rightarrow\; [\mathit{Border}(c, \mathit{Iraq}) \land \mathit{Border}(c, \mathit{Pakistan})]$.

(iii) $[\exists c \;\; \mathit{Country}(c)] \;\Rightarrow\; [\mathit{Border}(c, \mathit{Iraq}) \land \mathit{Border}(c, \mathit{Pakistan})]$.

(iv) $\exists c \;\; \mathit{Border}(\mathit{Country}(c), \mathit{Iraq} \land \mathit{Pakistan})$.

c. All countries that border Ecuador are in South America.

(i) $\forall c \;\; \mathit{Country}(c) \land \mathit{Border}(c, \mathit{Ecuador}) \;\Rightarrow\; \mathit{In}(c, \mathit{SouthAmerica})$.

(ii) $\forall c \;\; \mathit{Country}(c) \;\Rightarrow\; [\mathit{Border}(c, \mathit{Ecuador}) \;\Rightarrow\; \mathit{In}(c, \mathit{SouthAmerica})]$.

(iii) $\forall c \;\; [\mathit{Country}(c) \;\Rightarrow\; \mathit{Border}(c, \mathit{Ecuador})] \;\Rightarrow\; \mathit{In}(c, \mathit{SouthAmerica})$.

(iv) $\forall c \;\; \mathit{Country}(c) \land \mathit{Border}(c, \mathit{Ecuador}) \land \mathit{In}(c, \mathit{SouthAmerica})$.

d. No region in South America borders any region in Europe.

(i) $\neg[\exists c, d \;\; \mathit{In}(c, \mathit{SouthAmerica}) \land \mathit{In}(d, \mathit{Europe}) \land \mathit{Borders}(c, d)]$.

(ii) $\forall c, d \;\; [\mathit{In}(c, \mathit{SouthAmerica}) \land \mathit{In}(d, \mathit{Europe})] \;\Rightarrow\; \neg\mathit{Borders}(c, d)]$.

(iii) $\neg\forall c \;\; \mathit{In}(c, \mathit{SouthAmerica}) \;\Rightarrow\; \exists d \;\; \mathit{In}(d, \mathit{Europe}) \land \neg\mathit{Borders}(c, d)$.

(iv) $\forall c \;\; \mathit{In}(c, \mathit{SouthAmerica}) \;\Rightarrow\; \forall d \;\; \mathit{In}(d, \mathit{Europe}) \;\Rightarrow\; \neg\mathit{Borders}(c, d)$.

e. No two adjacent countries have the same map color.

(i) $\forall x, y \;\; \neg\mathit{Country}(x) \lor \neg\mathit{Country}(y) \lor \neg\mathit{Borders}(x, y) \lor$
$\neg(\mathit{MapColor}(x) = \mathit{MapColor}(y))$.

(ii) $\forall x, y \;\; (\mathit{Country}(x) \land \mathit{Country}(y) \land \mathit{Borders}(x, y) \land \neg(x = y)) \;\Rightarrow$
$\neg(\mathit{MapColor}(x) = \mathit{MapColor}(y))$.

(iii) $\forall x, y \;\; \mathit{Country}(x) \land \mathit{Country}(y) \land \mathit{Borders}(x, y) \land$
$\neg(\mathit{MapColor}(x) = \mathit{MapColor}(y))$.

(iv) $\forall x, y \;\; (\mathit{Country}(x) \land \mathit{Country}(y) \land \mathit{Borders}(x, y)) \;\Rightarrow\; \mathit{MapColor}(x \neq y)$.

a. Paris and Marseilles are both in France.

(i) $\mathit{In}(\mathit{Paris} \land \mathit{Marseilles}, \mathit{France})$.
(2) Syntactically invalid. Cannot use conjunction inside a term.

(ii) $\mathit{In}(\mathit{Paris}, \mathit{France}) \land \mathit{In}(\mathit{Marseilles}, \mathit{France})$.

(1) Correct.

(iii) $In(Paris, France) \lor In(Marseilles, France)$.
(3) Incorrect. Disjunction does not express "both."

b. There is a country that borders both Iraq and Pakistan.

(i) $\exists\, c \quad Country(c) \land Border(c, Iraq) \land Border(c, Pakistan)$.
(1) Correct.

(ii) $\exists\, c \quad Country(c) \;\Rightarrow\; [Border(c, Iraq) \land Border(c, Pakistan)]$.
(3) Incorrect. Use of implication in existential.

(iii) $[\exists\, c \quad Country(c)] \;\Rightarrow\; [Border(c, Iraq) \land Border(c, Pakistan)]$.
(2) Syntactically invalid. Variable c used outside the scope of its quantifier.

(iv) $\exists\, c \quad Border(Country(c), Iraq \land Pakistan)$.
(2) Syntactically invalid. Cannot use conjunction inside a term.

c. All countries that border Ecuador are in South America.

(i) $\forall\, c\;\; Country(c) \land Border(c, Ecuador) \;\Rightarrow\; In(c, SouthAmerica)$.
(1) Correct.

(ii) $\forall\, c\;\; Country(c) \;\Rightarrow\; [Border(c, Ecuador) \;\Rightarrow\; In(c, SouthAmerica)]$.
(1) Correct. Equivalent to (i).

(iii) $\forall\, c\;\; [Country(c) \;\Rightarrow\; Border(c, Ecuador)] \;\Rightarrow\; In(c, SouthAmerica)$.
(3) Incorrect. The implication in the LHS is effectively an implication in an existential; in particular, it sanctions the RHS for all non-countries.

(iv) $\forall\, c\;\; Country(c) \land Border(c, Ecuador) \land In(c, SouthAmerica)$.
(3) Incorrect. Uses conjunction as main connective of a universal quantifier.

d. No region in South America borders any region in Europe.

(i) $\neg[\exists\, c, d\;\; In(c, SouthAmerica) \land In(d, Europe) \land Borders(c, d)]$.
(1) Correct.

(ii) $\forall\, c, d\;\; [In(c, SouthAmerica) \land In(d, Europe)] \;\Rightarrow\; \neg Borders(c, d)]$.
(1) Correct.

(iii) $\neg\forall\, c\;\; In(c, SouthAmerica) \;\Rightarrow\; \exists\, d\;\; In(d, Europe) \land \neg Borders(c, d)$.
(3) Incorrect. This says there is some country in South America that borders every country in Europe!

(iv) $\forall\, c\;\; In(c, SouthAmerica) \;\Rightarrow\; \forall\, d\;\; In(d, Europe) \;\Rightarrow\; \neg Borders(c, d)$.
(1) Correct.

e. No two adjacent countries have the same map color.

(i) $\forall\, x, y\;\; \neg Country(x) \lor \neg Country(y) \lor \neg Borders(x, y) \lor$
$\neg(MapColor(x) = MapColor(y))$.
(1) Correct.

(ii) $\forall\, x, y\;\; (Country(x) \land Country(y) \land Borders(x, y) \land \neg(x = y)) \;\Rightarrow$
$\neg(MapColor(x) = MapColor(y))$.
(1) Correct. The inequality is unnecessary because no country borders itself.

(iii) $\forall\, x, y\;\; Country(x) \land Country(y) \land Borders(x, y) \land$
$\neg(MapColor(x) = MapColor(y))$.
(3) Incorrect. Uses conjunction as main connective of a universal quantifier.

(iv) $\forall x, y\ \ (Country(x) \land Country(y) \land Borders(x,y)) \ \Rightarrow\ MapColor(x \neq y)$.
(2) Syntactically invalid. Cannot use inequality inside a term.

Exercise 8.BOSS

Consider a vocabulary with the following symbols:

$Occupation(p, o)$: Predicate. Person p has occupation o.
$Customer(p1, p2)$: Predicate. Person $p1$ is a customer of person $p2$.
$Boss(p1, p2)$: Predicate. Person $p1$ is a boss of person $p2$.
$Doctor$, $Surgeon$, $Lawyer$, $Actor$: Constants denoting occupations.
$Emily$, Joe: Constants denoting people.

Use these symbols to write the following assertions in first-order logic:

a. Emily is either a surgeon or a lawyer.
b. Joe is an actor, but he also holds another job.
c. All surgeons are doctors.
d. Joe does not have a lawyer (i.e., is not a customer of any lawyer).
e. Emily has a boss who is a lawyer.
f. There exists a lawyer all of whose customers are doctors.
g. Every surgeon has a lawyer.

a. $O(E, S) \lor O(E, L)$.
b. $O(J, A) \land \exists p\ \ p \neq A \land O(J, p)$.
c. $\forall p\ \ O(p, S) \ \Rightarrow\ O(p, D)$.
d. $\neg\exists p\ \ C(J, p) \land O(p, L)$.
e. $\exists p\ \ B(p, E) \land O(p, L)$.
f. $\exists p\ \ O(p, L) \land \forall q\ \ C(q, p) \ \Rightarrow\ O(q, D)$.
g. $\forall p\ \ O(p, S) \ \Rightarrow\ \exists q\ \ O(q, L) \land C(p, q)$.

Exercise 8.DOGS

In each of the following we give an English sentence and a number of candidate logical expressions. For each of the logical expressions, state whether it (1) correctly expresses the English sentence; (2) is syntactically invalid and therefore meaningless; or (3) is syntactically valid but does not express the meaning of the English sentence.

a. Every cat loves its mother or father.

(i) $\forall x\ \ Cat(x) \ \Rightarrow\ Loves(x, Mother(x) \lor Father(x))$.
(ii) $\forall x\ \ \neg Cat(x) \lor Loves(x, Mother(x)) \lor Loves(x, Father(x))$.
(iii) $\forall x\ \ Cat(x) \land (Loves(x, Mother(x)) \lor Loves(x, Father(x)))$.

b. Every dog who loves one of its brothers is happy.

(i) $\forall x\ \ Dog(x) \land (\exists y\ Brother(y, x) \land Loves(x, y)) \ \Rightarrow\ Happy(x)$.

(ii) $\forall x, y \;\; Dog(x) \wedge Brother(y, x) \wedge Loves(x, y) \;\Rightarrow\; Happy(x)$.
(iii) $\forall x \;\; Dog(x) \wedge [\forall y \;\; Brother(y, x) \;\Leftrightarrow\; Loves(x, y)] \;\Rightarrow\; Happy(x)$.

c. No dog bites a child of its owner.

(i) $\forall x \;\; Dog(x) \;\Rightarrow\; \neg Bites(x, Child(Owner(x)))$.
(ii) $\neg \exists x, y \;\; Dog(x) \wedge Child(y, Owner(x)) \wedge Bites(x, y)$.
(iii) $\forall x \;\; Dog(x) \;\Rightarrow\; (\forall y \;\; Child(y, Owner(x)) \;\Rightarrow\; \neg Bites(x, y))$.
(iv) $\neg \exists x \;\; Dog(x) \;\Rightarrow\; (\exists y \;\; Child(y, Owner(x)) \wedge Bites(x, y))$.

d. Everyone's zip code within a state has the same first digit.

(i) $\forall x, s, z_1 \;\; [State(s) \wedge LivesIn(x, s) \wedge Zip(x) = z_1] \;\Rightarrow$
$[\forall y, z_2 \;\; LivesIn(y, s) \wedge Zip(y) = z_2 \;\Rightarrow\; Digit(1, z_1) = Digit(1, z_2)]$.
(ii) $\forall x, s \;\; [State(s) \wedge LivesIn(x, s) \wedge \exists z_1 \;\; Zip(x) = z_1] \;\Rightarrow$
$[\forall y, z_2 \;\; LivesIn(y, s) \wedge Zip(y) = z_2 \wedge Digit(1, z_1) = Digit(1, z_2)]$.
(iii) $\forall x, y, s \;\; State(s) \wedge LivesIn(x, s) \wedge LivesIn(y, s) \;\Rightarrow\; Digit(1, Zip(x) = Zip(y))$.
(iv) $\forall x, y, s \;\; State(s) \wedge LivesIn(x, s) \wedge LivesIn(y, s) \;\Rightarrow$
$Digit(1, Zip(x)) = Digit(1, Zip(y))$.

a. Every cat loves its mother or father.

(i) $\forall x \;\; Cat(x) \;\Rightarrow\; Loves(x, Mother(x) \vee Father(x))$.
(2) Syntactically invalid. Cannot have a disjunction inside a term.
(ii) $\forall x \;\; \neg Cat(x) \vee Loves(x, Mother(x)) \vee Loves(x, Father(x))$.
(1) Correct. (Rewrite as implication with disjunctive consequence.)
(iii) $\forall x \;\; Cat(x) \wedge (Loves(x, Mother(x)) \vee Loves(x, Father(x)))$.
(3) Incorrect. Use of $\wedge$ with $\forall$ means that everything is asserted to be a cat.

b. Every dog who loves one of its brothers is happy.

(i) $\forall x \;\; Dog(x) \wedge (\exists y \; Brother(y, x) \wedge Loves(x, y)) \;\Rightarrow\; Happy(x)$.
(1) Correct.
(ii) $\forall x, y \;\; Dog(x) \wedge Brother(y, x) \wedge Loves(x, y) \;\Rightarrow\; Happy(x)$.
(1) Correct. Logically equivalent to (i).
(iii) $\forall x \;\; Dog(x) \wedge [\forall y \;\; Brother(y, x) \;\Leftrightarrow\; Loves(x, y)] \;\Rightarrow\; Happy(x)$.
(3) Incorrect. States that dogs are happy if they love all of, and only, their brothers.

c. No dog bites a child of its owner.

(i) $\forall x \;\; Dog(x) \;\Rightarrow\; \neg Bites(x, Child(Owner(x)))$.
(3) Incorrect. Uses *Child* as a function instead of a relation.
(ii) $\neg \exists x, y \;\; Dog(x) \wedge Child(y, Owner(x)) \wedge Bites(x, y)$.
(1) Correct.
(iii) $\forall x \;\; Dog(x) \;\Rightarrow\; (\forall y \;\; Child(y, Owner(x)) \;\Rightarrow\; \neg Bites(x, y))$.
(1) Correct. Logically equivalent to (ii).
(iv) $\neg \exists x \;\; Dog(x) \;\Rightarrow\; (\exists y \;\; Child(y, Owner(x)) \wedge Bites(x, y))$.
(3) Incorrect. Uses $\Rightarrow$ with $\exists$.

d. Everyone's zip code within a state has the same first digit.

(i) $\forall\, x, s, z_1\ \ [State(s) \land LivesIn(x, s) \land Zip(x) = z_1] \;\Rightarrow$
$[\forall\, y, z_2\ \ LivesIn(y, s) \land Zip(y) = z_2 \;\Rightarrow\; Digit(1, z_1) = Digit(1, z_2)]$.
(1) Correct.

(ii) $\forall\, x, s\ \ [State(s) \land LivesIn(x, s) \land \exists\, z_1\ \ Zip(x) = z_1] \;\Rightarrow$
$[\forall\, y, z_2\ \ LivesIn(y, s) \land Zip(y) = z_2 \land Digit(1, z_1) = Digit(1, z_2)]$.
(2) Syntactically invalid. Uses z_1 outside scope of its quantifier. Also uses $\land$ as the main connective in the universally quantified RHS.

(iii) $\forall\, x, y, s\ \ State(s) \land LivesIn(x, s) \land LivesIn(y, s) \;\Rightarrow\; Digit(1, Zip(x) = Zip(y))$.
(2) Syntactically invalid. Cannot use equality within a term.

(iv) $\forall\, x, y, s\ \ State(s) \land LivesIn(x, s) \land LivesIn(y, s) \;\Rightarrow$
$Digit(1, Zip(x)) = Digit(1, Zip(y))$.
(1) Correct. Since *Zip* is a function, there is no need to define additional variables to name the zip codes.

Exercise 8.LAND

Complete the following exercises about logical sentences:

a. Translate into *good, natural* English (no xs or ys!):

$$\forall\, x, y, l\ \ SpeaksLanguage(x, l) \land SpeaksLanguage(y, l) \;\Rightarrow\; Understands(x, y) \land Understands(y, x).$$

b. Explain why this sentence is entailed by the sentence

$$\forall\, x, y, l\ \ SpeaksLanguage(x, l) \land SpeaksLanguage(y, l) \;\Rightarrow\; Understands(x, y).$$

c. Translate into first-order logic the following sentences:

(i) Understanding leads to friendship.
(ii) Friendship is transitive.

Remember to define all predicates, functions, and constants you use.

a. People who speak the same language understand each other.

b. Suppose that an extended interpretation with $x \rightarrow A$ and $y \rightarrow B$ satisfy

$$SpeaksLanguage(x, l) \land SpeaksLanguage(y, l)$$

for some l. Then from the second sentence we can conclude $Understands(A, B)$. The extended interpretation with $x \rightarrow B$ and $y \rightarrow A$ also must satisfy

$$SpeaksLanguage(x, l) \land SpeaksLanguage(y, l)\ ,$$

allowing us to conclude $Understands(B, A)$. Hence, whenever the second sentence holds, the first holds.

c. Let $Understands(x, y)$ mean that x understands y, and let $Friend(x, y)$ mean that x is a friend of y.

(i) It is not completely clear if the English sentence is referring to mutual understanding and mutual friendship, but let us assume that is what is intended:
$\forall x, y \;\; Understands(x, y) \land Understands(y, x) \Rightarrow (Friend(x, y) \land Friend(y, x))$.

(ii) $\forall x, y, z \;\; Friend(x, y) \land Friend(y, z) \;\Rightarrow\; Friend(x, z)$.

Exercise 8.PEAN

Rewrite the first two Peano axioms in Section 8.3.3 as a single axiom that defines $NatNum(x)$ so as to exclude the possibility of natural numbers except for those generated by the successor function.

This exercise requires a rewriting similar to the Clark completion of the two Horn clauses:

$$\forall n \;\; NatNum(n) \;\Leftrightarrow\; [n = 0 \lor \exists m \;\; NatNum(m) \land n = S(m)] \,.$$

Exercise 8.WUMD

Equation (8.4) on page 289 defines the conditions under which a square is breezy. Here we consider two other ways to describe this aspect of the wumpus world.

a. We can write **diagnostic rules** leading from observed effects to hidden causes. For finding pits, the obvious diagnostic rules say that if a square is breezy, some adjacent square must contain a pit; and if a square is not breezy, then no adjacent square contains a pit. Write these two rules in first-order logic and show that their conjunction is logically equivalent to Equation (8.4).

b. We can write **causal rules** leading from cause to effect. One obvious causal rule is that a pit causes all adjacent squares to be breezy. Write this rule in first-order logic, explain why it is incomplete compared to Equation (8.4), and supply the missing axiom.

a. The two implication sentences are

$$\begin{array}{l}\forall s \;\; Breezy(s) \;\Rightarrow\; \exists r \;\; Adjacent(r, s) \land Pit(r) \\ \forall s \;\; \lnot Breezy(s) \;\Rightarrow\; \lnot\exists r \;\; Adjacent(r, s) \land Pit(r) \,.\end{array}$$

The converse of the second sentence is

$$\forall s \;\; \exists r \;\; Adjacent(r, s) \land Pit(r) \;\Rightarrow\; Breezy(s)$$

which, combined with the first sentence, immediately gives

$$\forall s \;\; Breezy(s) \;\Leftrightarrow\; \exists r \;\; Adjacent(r, s) \land Pit(r) \,.$$

b. To say that a pit causes all adjacent squares to be breezy:

$$\forall s \;\; Pit(s) \;\Rightarrow\; [\forall r \;\; Adjacent(r,s) \;\Rightarrow\; Breezy(r)] \, .$$

This axiom allows for breezes to occur spontaneously with no adjacent pits. It would be incorrect to say that a non-pit causes all adjacent squares to be non-breezy, since there might be pits in other squares causing one of the adjacent squares to be breezy. But if *all* adjacent squares have no pits, a square is non-breezy:

$$\forall s \;\; [\forall r \;\; Adjacent(r,s) \;\Rightarrow\; \neg Pit(r)] \;\Rightarrow\; \neg Breezy(s) \, .$$

Exercise 8.KINS

Write axioms describing the predicates $Grandchild$, $Greatgrandparent$, $Ancestor$, $Brother$, $Sister$, $Daughter$, Son, $FirstCousin$, $BrotherInLaw$, $SisterInLaw$, $Aunt$, and $Uncle$. Find out the proper definition of mth cousin n times removed, and write the definition in first-order logic. Now write down the basic facts depicted in the family tree in Figure **??**. Using a suitable logical reasoning system, TELL it all the sentences you have written down, and ASK it who are Elizabeth's grandchildren, Diana's brothers-in-law, Zara's great-grandparents, and Eugenie's ancestors.

Make sure you write definitions with $\Leftrightarrow$. If you use $\Rightarrow$, you are only imposing constraints, not writing a real definition. Note that for aunts and uncles, we include the relations whom the OED says are more strictly defined as aunts-in-law and uncles-in-law, since the latter terms are not in common use.

$$\begin{array}{l}
Grandchild(c,a) \;\Leftrightarrow\; \exists\, b \;\; Child(c,b) \land Child(b,a) \\
Greatgrandparent(a,d) \;\Leftrightarrow\; \exists\, b,c \;\; Child(d,c) \land Child(c,b) \land Child(b,a) \\
Ancestor(a,x) \;\Leftrightarrow\; Child(x,a) \lor \exists\, b \;\; Child(b,a) \land Ancestor(b,x) \\
Brother(x,y) \;\Leftrightarrow\; Male(x) \land Sibling(x,y) \\
Sister(x,y) \;\Leftrightarrow\; Female(x) \land Sibling(x,y) \\
Daughter(d,p) \;\Leftrightarrow\; Female(d) \land Child(d,p) \\
Son(s,p) \;\Leftrightarrow\; Male(s) \land Child(s,p) \\
FirstCousin(c,d) \;\Leftrightarrow\; \exists\, p_1,p_2 \;\; Child(c,p_1) \land Child(d,p_2) \land Sibling(p_1,p_2) \\
BrotherInLaw(b,x) \;\Leftrightarrow\; \exists\, m \;\; Spouse(x,m) \land Brother(b,m) \\
SisterInLaw(s,x) \;\Leftrightarrow\; \exists\, m \;\; Spouse(x,m) \land Sister(s,m) \\
Aunt(a,c) \;\Leftrightarrow\; \exists\, p \;\; Child(c,p) \land [Sister(a,p) \lor SisterInLaw(a,p)] \\
Uncle(u,c) \;\Leftrightarrow\; \exists\, p \;\; Child(c,p) \land [Brother(a,p) \lor BrotherInLaw(a,p)]
\end{array}$$

There are several equivalent ways to define an mth cousin n times removed. One way is to look at the distance of each person to the nearest common ancestor. Define $Distance(c,a)$ as follows:

$$\begin{array}{l}
Distance(c,c) = 0 \\
Child(c,b) \land Distance(b,a) = k \;\Rightarrow\; Distance(c,a) = k+1 \, .
\end{array}$$

Thus, the distance to one's grandparent is 2, great-great-grandparent is 4, and so on. Now we have

$$
\begin{aligned}
&\mathit{MthCousinNTimesRemoved}(c,d,m,n) \;\Leftrightarrow \\
&\qquad \exists\, a\;\; \mathit{Distance}(c,a) = m+1 \land \mathit{Distance}(d,a) = m+n+1\;.
\end{aligned}
$$

The facts in the family tree are simple: each arrow represents two instances of Child (e.g., $\mathit{Child}(\mathit{William},\mathit{Diana})$ and $\mathit{Child}(\mathit{William},\mathit{Charles})$), each name represents a sex proposition (e.g., $\mathit{Male}(\mathit{William})$ or $\mathit{Female}(\mathit{Diana})$), each "bowtie" symbol indicates a Spouse proposition (e.g., $\mathit{Spouse}(\mathit{Charles},\mathit{Diana})$). Making the queries of the logical reasoning system is just a way of debugging the definitions.

Exercise 8.COMM

Write down a sentence asserting that + is a commutative function. Does your sentence follow from the Peano axioms? If so, explain why; if not, give a model in which the axioms are true and your sentence is false.

Commutativity of $+$ is asserted by

$$\forall\, m,n\;\; \mathit{NatNum}(m) \land \mathit{NatNum}(n) \;\Rightarrow\; +(m,n) = +(n,m)\;.$$

The sentence follows from the Peano axioms and can be proved by double induction over m and n. For example, the base case for $m=0$ is proved as follows: Omitting all references to NatNum for clarity, we must prove

$$\forall\, n\;\; +(0,n) = +(n,0)\;.$$

- Base case $n=0$: the assertion reduces to $+(0,0) = +(0,0)$, which is trivially true.
- Inductive step: given $\forall\, n\;\; +(0,n) = +(n,0)$, prove $\forall\, n\;\; +(0,S(n)) = +(S(n),0)$. We have

$$
\begin{aligned}
+(S(n),0) &= S(+(n,0)) && \text{by the second axiom for addition}\\
&= S(+(0,n)) && \text{by the inductive hypothesis}\\
&= S(n) && \text{by the first axiom for addition}\\
&= +(0,S(n)) && \text{by the first axiom for addition.}
\end{aligned}
$$

The inductive step for m proceeds similarly.

Exercise 8.SETM

Explain what is wrong with the following proposed definition of the set membership predicate $\in$:

$$
\begin{aligned}
&\forall\, x,s\;\; x \in \{x|s\}\\
&\forall\, x,s\;\; x \in s \;\Rightarrow\; \forall\, y\;\; x \in \{y|s\}\;.
\end{aligned}
$$

Although these axioms are sufficient to prove set membership when x is in fact a member of a given set, they have nothing to say about cases where x is not a member. For example, it is not possible to prove that x is not a member of the empty set. These axioms may therefore be suitable for a logical system, such as Prolog, that uses negation-as-failure.

Exercise 8.LIST

Using the set axioms as examples, write axioms for the list domain, including all the constants, functions, and predicates mentioned in the chapter.

Here we translate $List?$ to mean "proper list" in Lisp terminology, i.e., a cons structure with Nil as the "rightmost" atom.

$$\begin{array}{l}
List?(Nil)\\
\forall x, l \;\; List?(l) \;\Leftrightarrow\; List?(Cons(x,l))\\
\forall x, y \;\; First(Cons(x,y)) = x\\
\forall x, y \;\; Rest(Cons(x,y)) = y\\
\forall x \;\; Append(Nil, x) = x\\
\forall v, x, y, z \;\; List?(x) \;\Rightarrow\; (Append(x,y) = z \;\Leftrightarrow\; Append(Cons(v,x),y) = Cons(v,z))\\
\forall x \;\; \neg Find(x, Nil)\\
\forall x \;\; List?(z) \;\Rightarrow\; (Find(x, Cons(y,z)) \;\Leftrightarrow\; (x = y \lor Find(x,z))
\end{array}$$

Exercise 8.ADJX

Explain what is wrong with the following proposed definition of adjacent squares in the wumpus world:

$$\forall x, y \;\; Adjacent([x,y],[x+1,y]) \land Adjacent([x,y],[x,y+1])\ .$$

There are several problems with the proposed definition. It allows one to prove, say, $Adjacent([1,1],[1,2])$ but not $Adjacent([1,2],[1,1])$; so we need an additional symmetry axiom. It does not allow one to prove that $Adjacent([1,1],[1,3])$ is false, so it needs to be written as

$$\forall s_1, s_2 \quad \Leftrightarrow \quad \ldots$$

Finally, it does not work as the boundaries of the world, so some extra conditions must be added.

Exercise 8.WUML

Write out the axioms required for reasoning about the wumpus's location, using a constant symbol $Wumpus$ and a binary predicate $At(Wumpus, Location)$. Remember that there is only one wumpus.

We need the following sentences:

$$\begin{aligned}&\forall s_1\ \ Smelly(s_1) \ \Leftrightarrow\ \exists s_2\ \ Adjacent(s_1, s_2) \land In(Wumpus, s_2)\\ &\exists s_1\ \ In(Wumpus, s_1) \land \forall s_2\ \ (s_1 \neq s_2)\ \Rightarrow\ \neg In(Wumpus, s_2)\ .\end{aligned}$$

Exercise 8.JOAN

Assuming predicates $Parent(p, q)$ and $Female(p)$ and constants $Joan$ and $Kevin$, with the obvious meanings, express each of the following sentences in first-order logic. (You may use the abbreviation $\exists^1$ to mean "there exists exactly one.")

a. Joan has a daughter (possibly more than one, and possibly sons as well).
b. Joan has exactly one daughter (but may have sons as well).
c. Joan has exactly one child, a daughter.
d. Joan and Kevin have exactly one child together.
e. Joan has at least one child with Kevin, and no children with anyone else.

a. $\exists x\ \ Parent(Joan, x) \land Female(x)$.
b. $\exists^1 x\ Parent(Joan, x) \land Female(x)$.
c. $\exists x\ \ Parent(Joan, x) \land Female(x) \land [\forall y\ \ Parent(Joan, y)\ \Rightarrow\ y = x]$.
(This is sometimes abbreviated "$Female(\iota(x)Parent(Joan, x))$".)
d. $\exists^1 c\ Parent(Joan, c) \land Parent(Kevin, c)$.
e.

$$\begin{aligned}&\exists c\ \ Parent(Joan, c) \land Parent(Kevin, c) \land \forall d, p\ \ [Parent(Joan, d) \land Parent(p, d)]\\ &\quad\Rightarrow\ [p = Joan \lor p = Kevin]\end{aligned}$$

Exercise 8.ARTH

Arithmetic assertions can be written in first-order logic with the predicate symbol $<$, the function symbols $+$ and $\times$, and the constant symbols 0 and 1. Additional predicates can also be defined with biconditionals.

a. Represent the property "x is an even number."
b. Represent the property "x is prime."
c. Goldbach's conjecture is the conjecture (unproven as yet) that every even number is equal to the sum of two primes. Represent this conjecture as a logical sentence.

a. $\forall x \;\; Even(x) \;\Leftrightarrow\; \exists y \;\; x = y + y.$

b. $\forall x \;\; Prime(x) \;\Leftrightarrow\; \forall y, z \;\; x = y \times z \;\Rightarrow\; y = 1 \lor z = 1.$

c. $\forall x \;\; Even(x) \;\Rightarrow\; \exists y, z \;\; Prime(y) \land Prime(z) \land x = y + z.$

Exercise 8.EQWA

In Chapter 5, we used equality to indicate the relation between a variable and its value. For instance, we wrote $WA = red$ to mean that Western Australia is colored red. Representing this in first-order logic, we must write more verbosely $ColorOf(WA) = red$. What incorrect inference could be drawn if we wrote sentences such as $WA = red$ directly as logical assertions?

If we have $WA = red$ and $Q = red$ then we could deduce $WA = Q$, which is undesirable to both Western Australians and Queenslanders.

Exercise 8.KEYS

Write in first-order logic the assertion that every key and at least one of every pair of socks will eventually be lost forever, using only the following vocabulary: $Key(x)$, x is a key; $Sock(x)$, x is a sock; $Pair(x, y)$, x and y are a pair; Now, the current time; $Before(t_1, t_2)$, time t_1 comes before time t_2; $Lost(x, t)$, object x is lost at time t.

$$\begin{array}{l}
\forall k \;\; Key(k) \;\Rightarrow\; [\exists t_0 \;\; Before(Now, t_0) \land \forall t \;\; Before(t_0, t) \;\Rightarrow\; Lost(k, t)] \\
\forall s_1, s_2 \;\; Sock(s_1) \land Sock(s_2) \land Pair(s_1, s_2) \;\Rightarrow \\
\qquad [\exists t_1 \;\; Before(Now, t_1) \land \forall t \;\; Before(t_1, t) \;\Rightarrow\; Lost(s_1, t)] \lor \\
\qquad [\exists t_2 \;\; Before(Now, t_2) \land \forall t \;\; Before(t_2, t) \;\Rightarrow\; Lost(s_2, t)] \;.
\end{array}$$

Notice that the disjunction allows for both socks to be lost, as the English sentence implies.

Exercise 8.ENGL

For each of the following sentences in English, decide if the accompanying first-order logic sentence is a good translation. If not, explain why not and correct it. (Some sentences may have more than one error!)

a. No two people have the same social security number.

$$\neg\exists x, y, n \;\; Person(x) \land Person(y) \;\Rightarrow\; [HasSS\#(x, n) \land HasSS\#(y, n)].$$

b. John's social security number is the same as Mary's.

$$\exists\, n \;\; HasSS\#(John, n) \land HasSS\#(Mary, n).$$

c. Everyone's social security number has nine digits.

$$\forall\, x, n \;\; Person(x) \;\Rightarrow\; [HasSS\#(x, n) \land Digits(n, 9)].$$

d. Rewrite each of the above (uncorrected) sentences using a function symbol $SS\#$ instead of the predicate $HasSS\#$.

a. "No two people have the same social security number."

$$\neg\exists\, x, y, n \;\; Person(x) \land Person(y) \;\Rightarrow\; [HasSS\#(x, n) \land HasSS\#(y, n)].$$

This uses $\Rightarrow$ with $\exists$. It also says that no person has a social security number because it doesn't restrict itself to the cases where x and y are not equal. Correct version:

$$\neg\exists\, x, y, n \;\; Person(x) \land Person(y) \land \neg(x = y) \land [HasSS\#(x, n) \land HasSS\#(y, n)]$$

b. "John's social security number is the same as Mary's."

$$\exists\, n \;\; HasSS\#(John, n) \land HasSS\#(Mary, n).$$

This is OK.

c. "Everyone's social security number has nine digits."

$$\forall\, x, n \;\; Person(x) \;\Rightarrow\; [HasSS\#(x, n) \land Digits(n, 9)].$$

This says that everyone has every number. $HasSS\#(x, n)$ should be in the premise:

$$\forall\, x, n \;\; Person(x) \land HasSS\#(x, n) \;\Rightarrow\; Digits(n, 9)$$

d. Here $SS\#(x)$ denotes the social security number of x. Using a function enforces the rule that everyone has just one.

$$\begin{array}{l}
\neg\exists\, x, y \;\; Person(x) \land Person(y) \;\Rightarrow\; [SS\#(x) = SS\#(y)] \\
SS\#(John) = SS\#(Mary) \\
\forall\, x \;\; Person(x) \;\Rightarrow\; Digits(SS\#(x), 9)
\end{array}$$

Exercise 8.ENGT

Translate into first-order logic the sentence "Everyone's DNA is unique and is derived from their parents' DNA." You must specify the precise intended meaning of your vocabulary terms. (*Hint*: Do not use the predicate $Unique(x)$, since uniqueness is not really a property of an object in itself!)

Let $DNA(x)$ be a function denoting (a string representation of) a person's DNA and $DerivedFrom(d, d_1, d_2)$ be true if string d is derived from strings d_1 and d_2. (Technically speaking, this predicate is a bit vague as the amount of allowed recombination and mutation is not specified.) Then we have

$$\forall x, y \;\; Person(x) \land Person(y) \land x \neq y \;\Rightarrow\; DNA(x) \neq DNA(y)$$
$$\forall x \;\; Person(x) \;\Rightarrow\; DerivedFrom(DNA(x), DNA(Mother(x)), DNA(Father(x)))$$

Exercise 8.ENGG

For each of the following sentences in English, decide if the accompanying first-order logic sentence is a good translation. If not, explain why not and correct it.

a. Any apartment in London has lower rent than some apartments in Paris.

$$\forall x \;\; [Apt(x) \land In(x, London)] \;\Rightarrow\; \exists y \;\; ([Apt(y) \land In(y, Paris)] \;\Rightarrow\; (Rent(x) < Rent(y)))\ .$$

b. There is exactly one apartment in Paris with rent below \$1000.

$$\exists x \;\; Apt(x) \land In(x, Paris) \land \forall y \;\; [Apt(y) \land In(y, Paris) \land (Rent(y) < Dollars(1000))] \;\Rightarrow\; (y = x).$$

c. If an apartment is more expensive than all apartments in London, it must be in Moscow.

$$\forall x \;\; Apt(x) \land [\forall y \;\; Apt(y) \land In(y, London) \land (Rent(x) > Rent(y))] \;\Rightarrow\; In(x, Moscow).$$

a. Any apartment in London has lower rent than some apartments in Paris.

$$\forall x \;\; [Apt(x) \land In(x, London)] \;\Rightarrow\; \exists y \;\; ([Apt(y) \land In(y, Paris)] \;\Rightarrow\; (Rent(x) < Rent(y)))\ .$$

This uses $\Rightarrow$ with $\exists$; replacing it with $\land$ makes the sentence correct:

$$\forall x \;\; [Apt(x) \land In(x, London)] \;\Rightarrow\; \exists y \;\; ([Apt(y) \land In(y, Paris)] \land (Rent(x) < Rent(y)))\ .$$

b. There is exactly one apartment in Paris with rent below \$1000.

$$\begin{aligned}&\exists x \;\; Apt(x) \land In(x, Paris) \land \\ &\quad \forall y \;\; [Apt(y) \land In(y, Paris) \land (Rent(y) < Dollars(1000))] \;\Rightarrow\; (y = x).\end{aligned}$$

This sentence would hold in a world with no apartments below \$1000. We need to say that x has the required property:

$$\begin{aligned}&\exists x \;\; Apt(x) \land In(x, Paris) \land (Rent(x) < Dollars(1000)) \land \\ &\quad \forall y \;\; [Apt(y) \land In(y, Paris) \land (Rent(y) < Dollars(1000))] \;\Rightarrow\; (y = x).\end{aligned}$$

c. If an apartment is more expensive than all apartments in London, it must be in Moscow.

$$\begin{aligned}&\forall x \;\; Apt(x) \land [\forall y \;\; Apt(y) \land In(y, London) \land (Rent(x) > Rent(y))] \;\Rightarrow \\ &\quad In(x, Moscow).\end{aligned}$$

This uses $\land$ with $\forall$. Replacing it with an implication fixes the problem:

$$\begin{aligned}&\forall x \;\; Apt(x) \land [\forall y \;\; Apt(y) \land In(y, London) \;\Rightarrow\; (Rent(x) > Rent(y))] \;\Rightarrow \\ &\quad In(x, Moscow).\end{aligned}$$

Exercise 8.FOLV

Represent the following sentences in first-order logic, using a consistent vocabulary (which you must define):

a. Some students took French in spring 2001.

b. Every student who takes French passes it.

c. Only one student took Greek in spring 2001.

d. The best score in Greek is always higher than the best score in French.

e. Every person who buys a policy is smart.

f. No person buys an expensive policy.

g. There is an agent who sells policies only to people who are not insured.

h. There is a barber who shaves all men in town who do not shave themselves.

i. A person born in the UK, each of whose parents is a UK citizen or a UK resident, is a UK citizen by birth.

j. A person born outside the UK, one of whose parents is a UK citizen by birth, is a UK citizen by descent.

k. Politicians can fool some of the people all of the time, and they can fool all of the people some of the time, but they can't fool all of the people all of the time.

l. All Greeks speak the same language. (Use $Speaks(x, l)$ to mean that person x speaks language l.)

In this exercise, it is best not to worry about details of tense and larger concerns with consistent ontologies and so on. The main point is to make sure students understand connectives and quantifiers and the use of predicates, functions, constants, and equality. Let the basic vocabulary be as follows:
$Takes(x, c, s)$: student x takes course c in semester s;
$Passes(x, c, s)$: student x passes course c in semester s;
$Score(x, c, s)$: the score obtained by student x in course c in semester s;
$x > y$: x is greater than y;
F and G: specific French and Greek courses (one could also interpret these sentences as referring to *any* such course, in which case one could use a predicate $Subject(c, f)$ meaning that the subject of course c is field f;
$Buys(x, y, z)$: x buys y from z (using a binary predicate with unspecified seller is OK but less felicitous);
$Sells(x, y, z)$: x sells y to z;
$Shaves(x, y)$: person x shaves person y
$Born(x, c)$: person x is born in country c;
$Parent(x, y)$: x is a parent of y;
$Citizen(x, c, r)$: x is a citizen of country c for reason r;
$Resident(x, c)$: x is a resident of country c;
$Fools(x, y, t)$: person x fools person y at time t;
$Student(x)$, $Person(x)$, $Man(x)$, $Barber(x)$, $Expensive(x)$, $Agent(x)$, $Insured(x)$, $Smart(x)$, $Politician(x)$: predicates satisfied by members of the corresponding categories.

a. Some students took French in spring 2001.
$\exists\, x \;\; Student(x) \land Takes(x, F, Spring2001)$.

b. Every student who takes French passes it.
$\forall\, x, s \;\; Student(x) \land Takes(x, F, s) \;\Rightarrow\; Passes(x, F, s)$.

c. Only one student took Greek in spring 2001.
$\exists\, x \;\; Student(x) \land Takes(x, G, Spring2001) \land \forall\, y \;\; y \neq x \Rightarrow \neg Takes(y, G, Spring2001)$.

d. The best score in Greek is always higher than the best score in French.
$\forall\, s \;\; \exists\, x \;\; \forall\, y \;\; Score(x, G, s) > Score(y, F, s)$.

e. Every person who buys a policy is smart.
$\forall\, x \;\; Person(x) \land (\exists\, y, z \;\; Policy(y) \land Buys(x, y, z)) \;\Rightarrow\; Smart(x)$.

f. No person buys an expensive policy.
$\forall\, x, y, z \;\; Person(x) \land Policy(y) \land Expensive(y) \;\Rightarrow\; \neg Buys(x, y, z)$.

g. There is an agent who sells policies only to people who are not insured.
$\exists\, x \;\; Agent(x) \land \forall\, y, z \;\; Policy(y) \land Sells(x, y, z) \;\Rightarrow\; (Person(z) \land \neg Insured(z))$.

h. There is a barber who shaves all men in town who do not shave themselves.
$\exists\, x \;\; Barber(x) \land \forall\, y \;\; Man(y) \land \neg Shaves(y, y) \;\Rightarrow\; Shaves(x, y)$.

i. A person born in the UK, each of whose parents is a UK citizen or a UK resident, is a UK citizen by birth.
$\forall\, x \;\; Person(x) \land Born(x, UK) \land (\forall\, y \;\; Parent(y, x) \Rightarrow ((\exists\, r \;\; Citizen(y, UK, r)) \lor Resident(y, UK))) \;\Rightarrow\; Citizen(x, UK, Birth)$.

j. A person born outside the UK, one of whose parents is a UK citizen by birth, is a UK citizen by descent.

$$\begin{array}{l}\forall x\;\; Person(x) \land \neg Born(x, UK) \land (\exists y\;\; Parent(y, x) \land Citizen(y, UK, Birth)) \\ \qquad \Rightarrow\; Citizen(x, UK, Descent).\end{array}$$

k. Politicians can fool some of the people all of the time, and they can fool all of the people some of the time, but they can't fool all of the people all of the time.

$$\begin{array}{l}\forall x\;\; Politician(x)\;\Rightarrow \\ \quad (\exists y\;\; \forall t\;\; Person(y) \land Fools(x, y, t)) \land \\ \quad (\exists t\;\; \forall y\;\; Person(y)\;\Rightarrow\; Fools(x, y, t)) \land \\ \quad \neg(\forall t\;\; \forall y\;\; Person(y)\;\Rightarrow\; Fools(x, y, t))\end{array}$$

l. All Greeks speak the same language.

$$\begin{array}{l}\forall x, y, l\;\; Person(x) \land [\exists r\;\; Citizen(x, Greece, r)] \land Person(y) \land [\exists r\;\; Citizen(y, Greece, r)] \\ \quad \land Speaks(x, l)\;\Rightarrow\; Speaks(y, l)\end{array}$$

Exercise 8.NAPD

Write a general set of facts and axioms to represent the assertion "Wellington heard about Napoleon's death" and to correctly answer the question "Did Napoleon hear about Wellington's death?"

This is a very educational exercise but also highly nontrivial. Once students have learned about resolution, ask them to do the proof too. In most cases, they will discover missing axioms. Our basic predicates are $Heard(x, e, t)$ (x heard about event e at time t); $Occurred(e, t)$ (event e occurred at time t); $Alive(x, t)$ (x is alive at time t).

$$\begin{array}{l}\exists t\;\; Heard(W, DeathOf(N), t) \\ \forall x, e, t\;\; Heard(x, e, t)\;\Rightarrow\; Alive(x, t) \\ \forall x, e, t_2\;\; Heard(x, e, t_2)\;\Rightarrow\; \exists t_1\;\; Occurred(e, t_1) \land t_1 < t_2 \\ \forall t_1\;\; Occurred(DeathOf(x), t_1)\;\Rightarrow\; \forall t_2\;\; t_1 < t_2\;\Rightarrow\; \neg Alive(x, t_2) \\ \forall t_1, t_2\;\; \neg(t_2 < t_1)\;\Rightarrow\; ((t_1 < t_2) \lor (t_1 = t_2)) \\ \forall t_1, t_2, t_3\;\; (t_1 < t_2) \land ((t_2 < t_3) \lor (t_2 = t_3))\;\Rightarrow\; (t_1 < t_3) \\ \forall t_1, t_2, t_3\;\; ((t_1 < t_2) \lor (t_1 = t_2)) \land (t_2 < t_3)\;\Rightarrow\; (t_1 < t_3)\end{array}$$

Exercise 8.BEAT

Consider a first-order logical knowledge base that describes worlds containing people, songs, albums (e.g., "Meet the Beatles") and disks (i.e., particular physical instances of CDs). The vocabulary contains the following symbols:

$CopyOf(d, a)$: Predicate. Disk d is a copy of album a.
$Owns(p, d)$: Predicate. Person p owns disk d.
$Sings(p, s, a)$: Album a includes a recording of song s sung by person p.
$Wrote(p, s)$: Person p wrote song s.

$McCartney$, $Gershwin$, $BHoliday$, Joe, $EleanorRigby$, $TheManILove$, $Revolver$: Constants with the obvious meanings.

Express the following statements in first-order logic:

a. Gershwin wrote "The Man I Love."

b. Gershwin did not write "Eleanor Rigby."

c. Either Gershwin or McCartney wrote "The Man I Love."

d. Joe has written at least one song.

e. Joe owns a copy of *Revolver*.

f. Every song that McCartney sings on *Revolver* was written by McCartney.

g. Gershwin did not write any of the songs on *Revolver*.

h. Every song that Gershwin wrote has been recorded on some album. (Possibly different songs are recorded on different albums.)

i. There is a single album that contains every song that Joe has written.

j. Joe owns a copy of an album that has Billie Holiday singing "The Man I Love."

k. Joe owns a copy of every album that has a song sung by McCartney. (Of course, each different album is instantiated in a different physical CD.)

l. Joe owns a copy of every album on which all the songs are sung by Billie Holiday.

a. $W(G,T)$.

b. $\neg W(G,E)$.

c. $W(G,T) \lor W(M,T)$.

d. $\exists\, s\;\; W(J,s)$.

e. $\exists\, x\;\; C(x,R) \land O(J,x)$.

f. $\forall\, s\;\; S(M,s,R) \;\Rightarrow\; W(M,s)$.

g. $\neg[\exists\, s\;\; W(G,s) \land \exists\, p\;\; S(p,s,R)]$.

h. $\forall\, s\;\; W(G,s) \;\Rightarrow\; \exists\, p,a\;\; S(p,s,a)$.

i. $\exists\, a\;\; \forall\, s\;\; W(J,s) \;\Rightarrow\; \exists\, p\;\; S(p,s,a)$.

j. $\exists\, d,a,s\;\; C(d,a) \land O(J,d) \land S(B,T,a)$.

k. $\forall\, a\;\; [\exists\, s\;\; S(M,s,a)] \;\Rightarrow\; \exists\, d\;\; C(d,a) \land O(J,d)$.

l. $\forall\, a\;\; [\forall\, s,p\;\; S(p,s,a) \;\Rightarrow\; S(B,s,a)] \;\Rightarrow\; \exists\, d\;\; C(d,a) \land O(J,d)$.

8.4 Knowledge Engineering in First-Order Logic

Exercise 8.ADDR

Extend the vocabulary from Section 8.4 to define addition for n-bit binary numbers. Then encode the description of the four-bit adder in Figure **??**, and pose the queries needed to verify that it is in fact correct.

There are three stages to go through. In the first stage, we define the concepts of one-bit and n-bit addition. Then, we specify one-bit and n-bit adder circuits. Finally, we verify that the n-bit adder circuit does n-bit addition.

- One-bit addition is easy. Let Add_1 be a function of three one-bit arguments (the third is the carry bit). The result of the addition is a list of bits representing a 2-bit binary number, least significant digit first:

$$\begin{aligned}
&Add_1(0,0,0) = [0,0]\\
&Add_1(0,0,1) = [0,1]\\
&Add_1(0,1,0) = [0,1]\\
&Add_1(0,1,1) = [1,0]\\
&Add_1(1,0,0) = [0,1]\\
&Add_1(1,0,1) = [1,0]\\
&Add_1(1,1,0) = [1,0]\\
&Add_1(1,1,1) = [1,1]
\end{aligned}$$

- n-bit addition builds on one-bit addition. Let $Add_n(x_1, x_2, b)$ be a function that takes two lists of binary digits of length n (least significant digit first) and a carry bit (initially 0), and constructs a list of length $n+1$ that represents their sum. (It will always be exactly $n+1$ bits long, even when the leading bit is 0—the leading bit is the overflow bit.)

$$\begin{aligned}
&Add_n([],[],b) = [b]\\
&Add_1(b_1,b_2,b) = [b_3,b_4] \;\Rightarrow\; Add_n([b_1|x_1],[b_2|x_2],b) = [b_3|Add_n(x_1,x_2,b_4)]
\end{aligned}$$

- The next step is to define the structure of a one-bit adder circuit, as given in the text. Let $Add_1Circuit(c)$ be true of any circuit that has the appropriate components and connections:

$$\begin{aligned}
&\forall c \;\; Add_1Circuit(c) \;\Leftrightarrow\\
&\quad \exists x_1,x_2,a_1,a_2,o_1 \;\; Type(x_1) = Type(x_2) = XOR\\
&\qquad \wedge Type(a_1) = Type(a_2) = AND \wedge Type(o_1) = OR\\
&\qquad \wedge Connected(Out(1,x_1), In(1,x_2)) \wedge Connected(In(1,c), In(1,x_1))\\
&\qquad \wedge Connected(Out(1,x_1), In(2,a_2)) \wedge Connected(In(1,c), In(1,a_1))\\
&\qquad \wedge Connected(Out(1,a_2), In(1,o_1)) \wedge Connected(In(2,c), In(2,x_1))\\
&\qquad \wedge Connected(Out(1,a_1), In(2,o_1)) \wedge Connected(In(2,c), In(2,a_1))\\
&\qquad \wedge Connected(Out(1,x_2), Out(1,c)) \wedge Connected(In(3,c), In(2,x_2))\\
&\qquad \wedge Connected(Out(1,o_1), Out(2,c)) \wedge Connected(In(3,c), In(1,a_2))
\end{aligned}$$

  Notice that this allows the circuit to have additional gates and connections, but they won't stop it from doing addition.

- Now we define what we mean by an n-bit adder circuit, following the design of Figure 8.6. We will need to be careful, because an n-bit adder is not just an $n-1$-bit adder plus a one-bit adder; we have to connect the overflow bit of the $n-1$-bit adder to the carry-bit input of the one-bit adder. We begin with the base case, where $n = 0$:

$$\begin{array}{l} \forall\, c \;\; Add_n Circuit(c, 0) \;\Leftrightarrow \\ Signal(Out(1, c)) = 0 \end{array}$$

Now, for the recursive case we specify that the first connect the “overflow” output of the $n-1$-bit circuit as the carry bit for the last bit:

$$\begin{array}{l} \forall\, c, n \;\; n > 0 \;\Rightarrow\; [Add_n Circuit(c, n) \;\Leftrightarrow \\ \quad \exists\, c_2, d \;\; Add_n Circuit(c_2, n-1) \wedge Add_1 Circuit(d) \\ \qquad \wedge\, \forall\, m \;\; (m > 0) \wedge (m < 2n - 1) \;\Rightarrow\; In(m, c) = In(m, c_2) \\ \qquad \wedge\, \forall\, m \;\; (m > 0) \wedge (m < n) \;\Rightarrow\; \wedge\, Out(m, c) = Out(m, c_2) \\ \qquad \wedge\, Connected(Out(n, c_2), In(3, d)) \\ \qquad \wedge\, Connected(In(2n-1, c), In(1, d)) \wedge Connected(In(2n, c), In(2, d)) \\ \qquad \wedge\, Connected(Out(1, d), Out(n, c)) \wedge Connected(Out(2, d), Out(n+1, c)) \end{array}$$

- Now, to verify that a one-bit adder *circuit* actually adds correctly, we ask whether, given any setting of the inputs, the outputs equal the sum of the inputs:

$$\begin{array}{l} \forall\, c \;\; Add_1 Circuit(c) \;\Rightarrow \\ \quad \forall\, i_1, i_2, i_3 \;\; Signal(In(1, c)) = i_1 \wedge Signal(In(2, c)) = i_2 \wedge Signal(In(3, c)) = i_3 \\ \qquad \Rightarrow\; Add_1(i_1, i_2, i_3) = [Out(1, c), Out(2, c)] \end{array}$$

If this sentence is entailed by the KB, then every circuit with the $Add_1 Circuit$ design is in fact an adder. The query for the n-bit can be written as

$$\begin{array}{l} \forall\, c, n \;\; Add_n Circuit(c, n) \;\Rightarrow \\ \quad \forall\, x_1, x_2, y \;\; InterleavedInputBits(x_1, x_2, c) \wedge OutputBits(y, c) \\ \qquad \Rightarrow\; Add_n(x_1, x_2, y) \end{array}$$

where $InterleavedInputBits$ and $OutputBits$ are defined appropriately to map bit sequences to the actual terminals of the circuit. [*Note*: this logical formulation has not been tested in a theorem prover and we hesitate to vouch for its correctness.]

Exercise 8.CIRR

The circuit representation in the chapter is more detailed than necessary if we care only about circuit functionality. A simpler formulation describes any m-input, n-output gate or circuit using a predicate with $m + n$ arguments, such that the predicate is true exactly when the inputs and outputs are consistent. For example, NOT gates are described by the binary predicate $NOT(i, o)$, for which $NOT(0, 1)$ and $NOT(1, 0)$ are known. Compositions of gates are defined by conjunctions of gate predicates in which shared variables indicate direct connections. For example, a NAND circuit can be composed from ANDs and NOTs:

$$\forall\, i_1, i_2, o_a, o \;\; AND(i_1, i_2, o_a) \wedge NOT(o_a, o) \;\Rightarrow\; NAND(i_1, i_2, o)\ .$$

Using this representation, define the one-bit adder in Figure 8.6 and the four-bit adder in Figure **??**, and explain what queries you would use to verify the designs. What kinds of queries are *not* supported by this representation that *are* supported by the representation in Section 8.4?

Strictly speaking, the primitive gates must be defined using logical equivalences to exclude those combinations not listed as correct. If we are using a logic programming system, we can simply list the cases. For example,

$$AND(0,0,0) \qquad AND(0,1,0) \qquad AND(1,0,0) \qquad AND(1,1,1)\ .$$

For the one-bit adder, we have

$$\begin{aligned}&\forall\, i_1, i_2, i_3, o_1, o_2\ \ \mathit{Add_1Circuit}(i_1, i_2, i_3, o_1, o_2)\ \Leftrightarrow\\ &\quad \exists\, o_{x1}, o_{a1}, o_{x2}\ \ XOR(i_1, i_2, o_{x1}) \land XOR(o_{x1}, i_3, o_1)\\ &\qquad \land AND(i_1, i_2, o_{a1}) \land AND(i_3, o_{x1}, o_{a2})\\ &\qquad \land OR(o_{a2}, o_{a1}, o_2)\end{aligned}$$

To form a verification query, we need some independent definition of one-bit binary addition. Let us assume this is supplied by a predicate Add_1. Then we need to prove

$$\forall\, i_1, i_2, i_3, o_1, o_2\ \ \mathit{Add_1Circuit}(i_1, i_2, i_3, o_1, o_2)\ \Rightarrow\ Add_1(i_1, i_2, i_3, o_1, o_2)$$

Add_1 itself can be defined by a lookup table or through the builtin arithmetic functions of the theorem prover.

The simplified representation cannot support queries about whether particular terminals are connected in a given circuit, since the terminals are not reified (nor is the circuit itself). Thus, it cannot be used to debug a faulty circuit; nor is it easy to extend with timing information, non-Boolean signal values, etc.

Exercise 8.PASS

Obtain a passport application for your country, identify the rules determining eligibility for a passport, and translate them into first-order logic, following the steps outlined in Section 8.4.

The answers here will vary by country. The two key rules for UK passports are given above.