UNIVERSIDADE TÉCNICA DE LISBOA
INSTITUTO SUPERIOR TÉCNICO

Exercises for the practical classes of Knowledge Representation and Reasoning

Ana Cardoso-Cachopo

2011/2012

Contents

Preface

This document contains a collection of exercises for the subject "Knowledge Representation and Reasoning" of the *Master Degree (MSc) in Information Systems and Computer Engineering*, based on the book *Knowledge representation and reasoning*, by Ronald Brachman and Hector Levesque.

Some of these exercises are taken from the book, some are available in the internet and some I created to use in the practical classes or exams.

1 Introduction

Exercise 1.1 (new)
Explain why an AI system needs knowledge representation and reasoning.

Answer:

An AI system needs knowledge in order to have information about the world, and the information needs to be represented in some way. Reasoning is useful for the system to be able to draw new conclusions from the information that it has initially stored.

Exercise 1.2 (new)
Explain the need for reasoning in a knowledge-based system.

Answer:

See page 9. Knowledge-based systems represent information about a particular domain. Reasoning is used to infer new knowledge from the one that was explicitly represented. If there was no reasoning, we woud have a knowledge *base* and not a knowledge-*based system*. One should note that usually the knowledge that is explicitly represented is a lot less than the knowledge that can be inferred from it.

Exercise 1.3 (new)
Comment the following statement: "The use of meaningful names to represent terms and predicates in a logic representation makes inference easier for the reasoning mechanism.".

Answer:

The statement is wrong. Meaningful names help a human that reads a representation, but make no difference for the reasoning mechanism.

Exercise 1.4 (new)
Explain why logic is such an important language for the area of knowledge representation and reasoning.

Answer:

The reason logic is relevant to knowledge representation and reasoning is simply that, at least according to one view, logic is the study of entailment relations: languages, truth conditions, and rules of inference. It must be stressed, however, that FOL itself is also just a starting point. We will have good reason in what follows to consider subsets and super- sets of FOL, as well as knowledge representation languages quite different in form and meaning. Just as we are not committed to understanding reasoning as the computation of entailments, even when we do so we are not committed to any particular language.

2 The Language of First-Order Logic

Exercise 2.1 (new)
Distinguish between *function symbols* and *predicate symbols* in first-order logic. Give one suitable example for each of them.

Answer:

Exercise 2.2 (new)
In propositional logic an interpretation is composed only by its interpretation function (that is, it does not have a domain). Explain why.

Answer:

Exercise 2.3 (new)
Explain why it does not make sense to consider an equality function in propositional logic.

Answer:

Exercise 2.4 (new)
Is first order logic an adequate language to represent a knowledge base about time (time points, time intervals, etc)? Explain why.

Answer:

Exercise 2.5 (Ch 2, Ex 1)
For each of the following sentences, give a logical interpretation that makes that sentence false and the other two sentences true:

1. $\forall x \forall y \forall z[(P(x,y) \wedge P(y,z)) \supset P(x,z)]$
2. $\forall x \forall y[(P(x,y) \wedge P(y,x)) \supset (x = y)]$
3. $\forall x \forall y[P(a,y) \supset P(x,b)]$

Answer:

An interpretation is a pair $\langle \mathcal{D}, \mathcal{I} \rangle$ where $\mathcal{D}$ is the domain and $\mathcal{I}$ is the interpretation mapping. This is what we must provide to answer this exercise.

1. For this sentence to be false, we need a non-transitive relation. We can get that with $\mathcal{D} = \{A, B, C, D\}$ and $\mathcal{I}(a) = A$, $\mathcal{I}(b) = B$, $\mathcal{I}(c) = C$, $\mathcal{I}(d) = D$, $\mathcal{I}(P) = \{(B, C), (C, D)\}$. With this interpretation for P, the relation is not transitive because the tuple (B, D) is not in the relation. The second sentence is true because there is no variable assignment that makes the antecedent of the entailment true, so the entailment is always true and so is the universally quantified formula. The third sentence is true for the same reason (in this case there is no tuple in $\mathcal{I}(P)$ such that A is its first element).

2. For this sentence to be false we need a non-symmetric but reflexive relation. We can get that with $\mathcal{D} = \{A, B, C, D\}$ and $\mathcal{I}(a) = A$, $\mathcal{I}(b) = B$, $\mathcal{I}(c) = C$, $\mathcal{I}(d) = D$, $\mathcal{I}(P) = \{(B, C), (C, B), (B, B), (C, C)\}$. With this, the antecedent of the second sentence is true for variable assignment $\mu(x) = B, \mu(y) = C$, but $B \neq C$, so the sentence is not true for all x and y. The first sentence is true because for the variable assignments that make the antecedent of the entailment true, its consequent is also true, so the entailment is always true and so is the universally quantified formula. The third sentence is true because there is no variable assignment that makes its antecedent true, because there is no tuple in $\mathcal{I}(P)$ such that A is its first element.
3. It is hard to find a meaning for this sentence... It is false for $\mathcal{D} = \{A, B, C, D\}$ and $\mathcal{I}(a) = A$, $\mathcal{I}(b) = B$, $\mathcal{I}(c) = C$, $\mathcal{I}(d) = D$, $\mathcal{I}(P) = \{(A, C)\}$, because the antecedent is true for variable assignment $\mu(y) = C$, but the consequent is false, so the sentence is not true for all x and y. The first two sentences are true because their antecedents are false for all variable assignments.

Exercise 2.6 (Ch 2, Ex 4)
In a certain town, there are the following regulations concerning the town barber:

- Anyone who does not shave himself must be shaved by the barber
- Whomever the barber shaves, must not shave himself.

Show that no barber can fulfill these requirements. That is, formulate the requirements as sentences of FOL and show that in any interpretation where the first regulation is true, the second one must be false. (This is called the barber's paradox and was formulated by Bertrand Russel.)

Hint: introduce a constant $barber$ for the *unique* barber and a binary predicate $Shaves(x, y)$ meaning that x shaves y.

Answer:

Translation:

1. $\forall x[\neg Shaves(x, x) \supset Shaves(barber, x)]$
2. $\forall x[Shaves(barber, x) \supset \neg Shaves(x, x)]$

Suppose some interpretation $(\mathcal{D}, \mathcal{I})$ satisfies both 1 and 2. There is an object in $\mathcal{D}$ which is the meaning of 'barber' in that interpretation, $\mathcal{I}(barber)$. Let us call it b. This b either shaves himself or not, or in logic, the pair (b, b) is either in the interpretation $\mathcal{I}(Shaves)$ of the predicate $Shaves$, or not.

First suppose that $(b, b) \in \mathcal{I}(Shaves)$. Then, because the second sentence is true in $(\mathcal{D}, \mathcal{I})$, we have that, for every assignment v, $(\mathcal{D}, \mathcal{I}), v \models Shaves(barber, x) \supset \neg Shaves(x, x)$, in particular for v which assigns b to x. In plain English: because the sentence is true for all x, it must be true for the barber. So we have $(\mathcal{D}, \mathcal{I}), v \models \neg Shaves(barber, barber)$, which means that $(b, b) \notin \mathcal{I}(Shaves)$. A contradiction with our initial fact.

So suppose that $(b, b) \notin \mathcal{I}(Shaves)$. Because the first sentence is true in $(\mathcal{D}, \mathcal{I})$, for every assignment v, $(\mathcal{D}, \mathcal{I}), v \models \neg Shaves(x, x) \supset Shaves(barber, x)$; in particular this holds for v with $v(x) = b$. Because $(b, b) \notin \mathcal{I}(Shaves)$, $(\mathcal{D}, \mathcal{I}), v \models \neg Shaves(x, x)$ and also because the implication is true, $(\mathcal{D}, \mathcal{I}), v \models Shaves(barber, barber)$ but the latter means that $(b, b) \in \mathcal{I}(Shaves)$. A contradiction again.

Because we get a contradiction whether $(b, b) \in \mathcal{I}(Shaves)$ or $(b, b) \notin \mathcal{I}(Shaves)$, we cannot make both sentences true for the barber.

Exercise 2.7 (from `http://www.cs.nott.ac.uk/~nza/G53KRR/`)
Consider the following set of sentences:

S1 Andrew is the father of Bob.

S2 Bob is the father of Chris.

S3 Every grandfather is someone´s father.

S4 Andrew is a grandfather of Chris.

1. Translate these sentences into first-order logic, using binary predicates $Father$ and $Grandfather$ and constants a, b, c for Andrew, Bob and Chris.
2. Show semantically (by reasoning about interpretations) that **S1**-**S3** do not logically entail **S4**.
3. Write in first-order logic an additional sentence that defines a general property of grandfathers, and show that **S1**-**S3** together with this new sentence entail **S4**.

Answer:

1. The sentences are:

   S1 $Father(a, b)$

   S2 $Father(b, c)$

   S3 $\forall x[\exists y[Grandfather(x, y) \supset \exists z[Father(x, z)]]]$

   S4 $Grandfather(a, c)$

2. To show that **S1**-**S3** do not logically entail **S4** we must provide an interpretation which makes **S1**-**S3** true and **S4** false. Consider an interpretation $M = (D, I)$ where $D = \{Andrew, Bob, Chris\}$, the constants are interpreted in the obvious way, $I(Father) = \{(Andrew, Bob), (Bob, Chris)\}$, and $I(Grandfather)$ is empty. Then in M, **S1**-**S3** are true and $Grandfather(a, c)$ is false.
3. The sentence is: $\forall xyz[(Father(x, y) \wedge Father(y, z)) \supset Grandfather(x, z)]$
   Now every interpretation which satisfies $Father(a, b)$, $Father(b, c)$, that is, where $\{(Andrew, Bob), (Bob, Chris)\} \subseteq I(Father)$, has to have $\{(Andrew, Chris)\} \subseteq I(Grandfather)$, so has to satisfy $Grandfather(a, c)$.

Exercise 2.8 (Ch 2, Ex 1)
Consider the following set of sentences:

S1 Five is larger than three.

S2 Three is larger than one.

S3 For every number there is another number that is larger than it (the first number).

S4 Five, three and one are numbers.

S5 Five is larger than one.

1. Translate these sentences into first-order logic.
2. Show semantically (by reasoning about interpretations) that **S1**-**S4** do not logically entail **S5**.
3. Write in first-order logic an additional sentence that defines a general property of numbers, and show that **S1**-**S4** together with this new sentence entail **S5**.

Answer:

Exercise 2.9 (new)
Consider the following set of sentences:

S1 Jack is Rob's brother.

S2 Rob is Mike's brother.

S3 If one person is someone else's brother, then this second person also is a brother to the first one.

S4 Jack, Rob and Mike are people.

S5 Jack is Mike's brother.

1. Translate these sentences into first-order logic.
2. Show semantically (by reasoning about interpretations) that **S1**-**S4** do not logically entail **S5**.
3. Write in first-order logic an additional sentence that defines a general property of brothers, and show that **S1**-**S4** together with this new sentence entail **S5**.

Answer:

Exercise 2.10 (new)
Consider the following set of sentences:

S1 Neil likes every person that is nice.

S2 Mike is not annoying.

S3 If a person is not annoying, then that person is nice.

S4 If a person is nice, then there is some other person that likes her.

S5 Neil is a person.

S6 Neil likes Mike.

1. Translate these sentences into first-order logic.
2. Show semantically (by reasoning about interpretations) that **S1**-**S5** do not logically entail **S6**.
3. Write in first-order logic an additional sentence, and show that **S1**-**S5** together with this new sentence entail **S6**.

Answer:

3 Expressing Knowledge

Exercise 3.1 (Ch 3, Ex 1)
Consider the following piece of knowledge:

> Tony, Mike and John belong to the Alpine Club. Every member of the Alpine Club who is not a skier is a mountain climber. Mountain climbers do not like rain, and anyone who does not like snow is not a skier. Mike dislikes whatever Tony likes, and likes whatever Tony dislikes. Tony likes rain and snow.

1. Prove that the given sentences logically entail that there is a member of the Alpine Club who is a mountain climber but not a skier.
2. Suppose we had been told that Mike likes whatever Tony dislikes, but we had not been told that Mike dislikes whatever Tony likes. Prove that the resulting set of sentences no longer logically entails that there is a member of the Alpine Club who is a mountain climber but not a skier.

Answer:

1. One possible solution consists in using the following predicates and constants:

   $MemberAC$ unary predicate meaning a member of the Alpine Club
   $Skier$ unary predicate meaning a skier
   $Climber$ unary predicate meaning a mountain climber
   $Likes$ binary predicate where $Likes(x, y)$ means that x likes y
   constants: $tony$, $mike$, $john$, $rain$, $snow$

   Translation into first order logic, giving sentences names so that it is easy to refer to them later.

   > Tony, Mike and John belong to the Alpine Club.

   S1 $MemberAC(tony)$
   S2 $MemberAC(mike)$
   S3 $MemberAC(john)$

   It is also possible to translate them as one conjunction
   $MemberAC(tony) \wedge MemberAC(mike) \wedge MemberAC(john)$

   > Every member of the Alpine Club who is not a skier is a mountain climber.

   S4 $\forall x[(MemberAC(x) \wedge \neg Skier(x)) \supset Climber(x)]$

   > Mountain climbers do not like rain.

   S5 $\forall x[Climber(x) \supset \neg Like(x, rain)]$

   > Anyone who does not like snow is not a skier.

   S6 $\forall x[\neg Like(x, snow) \supset \neg Skier(x)]$

Mike dislikes whatever Tony likes and likes whatever Tony dislikes.

S7 $\forall x[Like(tony, x) \supset \neg Like(mike, x)]$

S8 $\forall x[\neg Like(tony, x) \supset Like(mike, x)]$

Tony likes rain and snow.

S9 $Like(tony, rain)$

S10 $Like(tony, snow)$

Proving that S1-S10 logically entail $\exists x[MemberAC(x) \wedge Climber(x) \wedge \neg Skier(x)]$.

Consider any interpretation (D, I) where S1-S10 are true. We have to show that $(D, I) \models \exists x[MemberAC(x) \wedge Climber(x) \wedge \neg Skier(x)]$.

The way to do this is to prove that there is some object $d \in D$ such that if an assignment v assigns d to x, then
$(D, I), v \models MemberAC(x) \wedge Climber(x) \wedge \neg Skier(x)$
in other words, find an object $d \in D$ such that
$d \in I(MemberAC), d \in I(Climber), d \notin I(Skier)$

Our only hold on which objects exist in D is that we know that D contains interpretations of $tony$, $mike$, $john$, $rain$ and $snow$: $I(tony) \in D$, $I(mike) \in D$, and so on. We know of some of the properties of those objects because (D, I) satisfies the sentences S1-S10.
For example from S1, $(D, I) \models MemberAC(tony)$
from the truth conditions we know that $I(tony) \in I(MemberAC)$ (the object which is called 'tony' in (D, I) belongs to the set of things which are considered Club Members in (D, I)).

For one of those objects, we need to prove that it is in $I(MemberAC)$, in $I(Climber)$, and is not in $I(Skier)$. Clearly $john$, $rain$ and $snow$ are non-starters.

Let us check if it could be true that $I(tony) \in I(Climber)$. From S9, we know that $(I(tony), I(rain)) \in I(Like)$. If $I(tony)$ were in $I(Climber)$, then there is an assignment v which assigns $I(tony)$ to x such that $(D, I), v \not\models Climber(x) \supset \neg Like(x, rain)$ because $Climber(x)$ is true under v and $\neg Like(x, rain)$ is false. But this contradicts sentence S5 being true (it says that $Climber(x) \supset \neg Like(x, rain)$ is true for all assignments). So Tony can't be a climber.

Our last hope is Mike. From S10 we know that $(I(tony), I(snow)) \in I(Like)$. From S7 we know that for every assignment v, $(D, I), v \models Like(tony, x) \supset \neg Like(mike, x)$ in particular if $v(x) = I(snow)$, we get that $(D, I), v \models \neg Like(mike, x)$. So $(I(mike), I(snow)) \notin I(Like)$. From S6 we know that for every v, $(D, I), v \models \neg Like(x, snow) \supset \neg Skier(x)$ in particular if $v(x) = I(mike)$ this should also be true. So $I(mike) \notin I(Skier)$. Finally, from S4 we know that the set of members who are not skiers is included in the set of climbers, so since $I(mike) \in I(MemberAC)$ and $I(mike) \notin I(Skier)$ then $I(mike) \in I(Climber)$. We have found an object with desired properties: if $v(x) = I(mike)$,
$(D, I), v \models MemberAC(x) \wedge Climber(x) \wedge \neg Skier(x)$
so
$(D, I) \models \exists x[MemberAC(x) \wedge Climber(x) \wedge \neg Skier(x)]$

OR

We need to prove: $\exists x[MemberAC(x) \wedge Climber(x) \wedge \neg Skier(x)]$
From **S7, S10** get: $\neg Like(mike, snow)$ — **S11**
From **S11, S6** get: $\neg Skier(mike)$ — **S12**
From **S2, S12** get: $Climber(mike)$ — **S13**
From **S2, S12, S13** get: $MemberAC(mike) \wedge Climber(mike) \wedge \neg Skier(mike)$ — **S14**
From **S14** get: $\exists x[MemberAC(x) \wedge Climber(x) \wedge \neg Skier(x)]$

2. Suppose we do not have S7, only S1-S6 and S8-S10. Prove that $\exists x[MemberAC(x) \wedge Climber(x) \wedge \neg Skier(x)]$ no longer follows. To do this, we have to produce an interpretation where S1-S6 and S8-S10 are true and the last sentence is false. The interpretation could be like this (there are other possible ones): $D = \{t, m, j, s, r\}$ Interpretation:
$I(tony) = t$, $I(mike) = m$, $I(john) = j$, $I(snow) = s$, $I(rain) = r$

$I(MemberAC) = \{t, m, j\}$
$I(Skier) = \{t, m, j\}$
$I(Climber) = \{\}$
$I(Like) = \{(t, s), (t, r), (m, s), (m, r), (m, m), (m, t), (m, j), (j, s)\}$ (that is, Tony likes rain and snow as before, Mike likes every single object in the universe, John likes snow, and rain and snow don't have any feelings about things).

Now S1-S3 are obviously true. S4 is trivially true because there is no member who is not a skier. S5 is also trivially true because there are no climbers. S6 is true because the only skiers we have are t, m, j and they all like snow. S8 is true because Mike likes everything. S9 and S10 are true because we included (t, s), (t, r) in $I(Like)$. Finally, the sentence $\exists x[MemberAC(x) \wedge Climber(x) \wedge \neg Skier(x)]$ is false because there are no climbers so we cannot find an assignment to x which would make $Climber(x)$ true.

4 Resolution

Exercise 4.1 (new)
There is one possible refinement to resolution that consists in eliminating pure clauses from the initial set of clauses (pure clauses are the ones that contain some literal p such that $\neg p$ does not appear anywhere). Explain why this refinement does not change the results obtainable by resolution.

Answer:

When there is a pure literal in one clause, if its negation doesn't appear anywhere else, it will be impossible to eliminate the literal. This way, we will never be able to achieve a contradiction using this clause. So we may simply not use this clause because, if there is a way to reach a contradiction, it will not use this clause.

Exercise 4.2 (new)
Define what is a formula in conjunctive normal form. Give one example.

Answer:

Exercise 4.3 (new)
Explain what it means to say that resolutions is not complete, but is refutation complete. State the consequences of this fact for the proofs made using resolution.

Answer:

Exercise 4.4 (new)
Explain why automated reasoning systems based on logic (propositional or first order logic) make proofs using resolution and not the logic's semantic system.

Answer:

Exercise 4.5 (Ch 4, Ex 1)
Determine whether the following sentence is valid using resolution:

$$\exists x \forall y \forall z((P(y) \supset Q(z)) \supset (P(x) \supset Q(x)))$$

Answer:

To do this we need to check if from the negation of the sentence we can derive an empty clause (a contradiction). So, the formula that needs to be transformed to clausal form is the negation of the original formula:

$\neg\exists x \forall y \forall z[(P(y) \supset Q(z)) \supset (P(x) \supset Q(x))]$
$\neg\exists x \forall y \forall z[\neg(\neg P(y) \vee Q(z)) \vee (\neg P(x) \vee Q(x))]$
$\forall x \exists y \exists z \neg[\neg(\neg P(y) \vee Q(z)) \vee (\neg P(x) \vee Q(x))]$
$\forall x \exists y \exists z[\neg\neg(\neg P(y) \vee Q(z)) \wedge \neg(\neg P(x) \vee Q(x))]$
$\forall x \exists y \exists z[(\neg P(y) \vee Q(z)) \wedge (\neg\neg P(x) \wedge \neg Q(x))]$
$\forall x \exists y \exists z[(\neg P(y) \vee Q(z)) \wedge P(x) \wedge \neg Q(x)]$
$\forall x[(\neg P(f(x)) \vee Q(g(x))) \wedge P(x) \wedge \neg Q(x)]$
$\{[\neg P(f(x)), Q(g(x))], [P(x)], [\neg Q(x)]\}$

Clauses:

C1 $[\neg P(f(x)), Q(g(x))]$

C2 $[P(x)]$

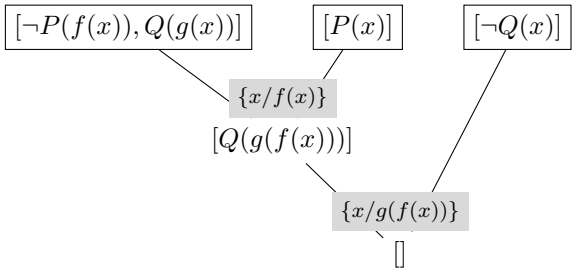
C3 $[\neg Q(x)]$

Proof:
(1) $[Q(g(f(x)))]$ from C1 and C2, $x/f(x)$
(2) [] from (1) and C3, $x/g(f(x))$

or

1	$[\neg P(f(x)), Q(g(x))]$	Prem
2	$[P(x)]$	Prem
3	$[\neg Q(x)]$	Prem
4	$[Q(g(f(x)))]$	Res, (1, 2), $\{x/f(x)\}$
5	[]	Res, (3, 4), $\{x/g(f(x))\}$

or



Exercise 4.6 (Ch 4, Ex 2, follow-up of Ch 3, Ex 1)
Use resolution to prove that there exists a member of the Alpine club who is a climber but not a skier.

Answer:

Translation into first order logic.

S1 $MemberAC(tony)$

S2 $MemberAC(mike)$

S3 $MemberAC(john)$

S4 $\forall x[MemberAC(x) \wedge \neg Skier(x) \supset Climber(x)]$

S5 $\forall x[Climber(x) \supset \neg Like(x, rain)]$

S6 $\forall x[\neg Like(x, snow) \supset \neg Skier(x)]$

S7 $\forall x[Like(tony, x) \supset \neg Like(mike, x)]$

S8 $\forall x[\neg Like(tony, x) \supset Like(mike, x)]$

S9 $Like(tony, rain)$

S10 $Like(tony, snow)$

S11 $\exists x[MemberAC(x) \wedge Climber(x) \wedge \neg Skier(x)]$

Same in clausal form, but negating the conclusion S11:

C1 $[MemberAC(tony)]$

C2 $[MemberAC(mike)]$

C3 $[MemberAC(john)]$

C4 $[\neg MemberAC(x), Skier(x), Climber(x)]$

C5 $[\neg Climber(x), \neg Like(x, rain)]$

C6 $[Like(x, snow), \neg Skier(x)]$

C7 $[\neg Like(tony, x), \neg Like(mike, x)]$

C8 $[Like(tony, x), Like(mike, x)]$

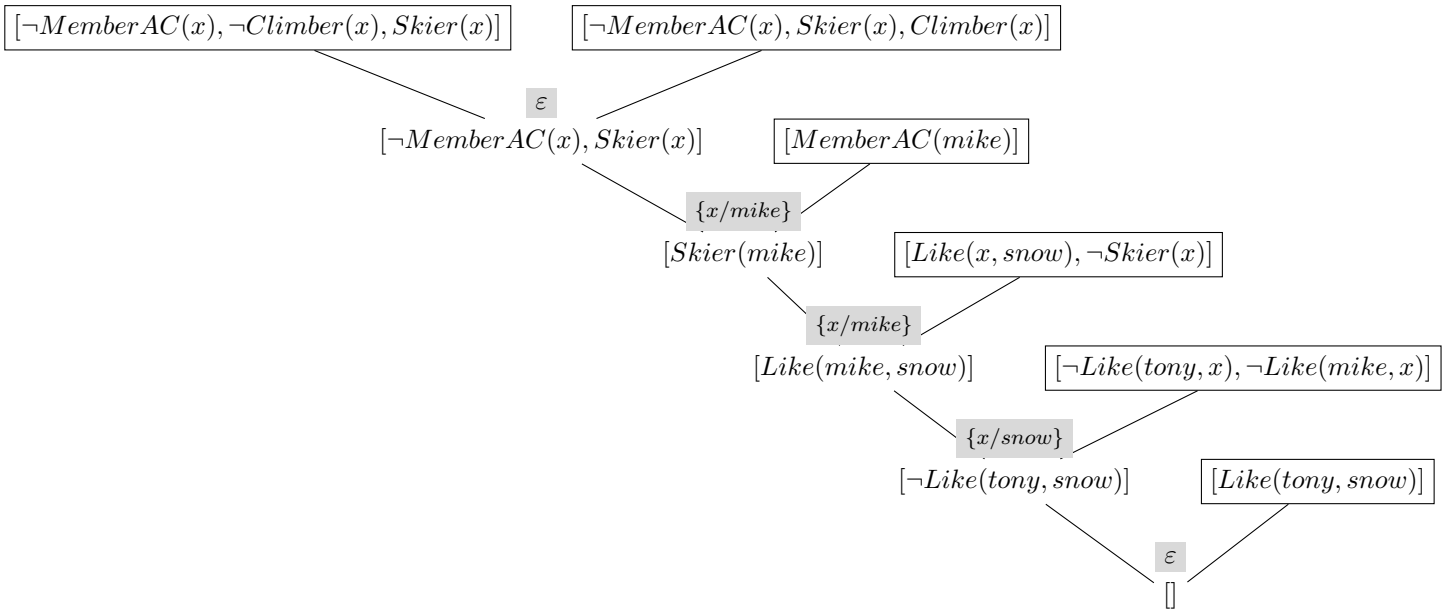
C9 $[Like(tony, rain)]$

C10 $[Like(tony, snow)]$

C11 Negation of S11: $\forall x[\neg MemberAC(x) \vee \neg Climber(x) \vee Skier(x)]$
$[\neg MemberAC(x), \neg Climber(x), Skier(x)]$

Proof that together C1-C11 are inconsistent:
(1) $[\neg Like(mike, snow)]$ from C10 and C7
(2) $[\neg Skier(mike)]$ from (1) and C6
(3) $[\neg MemberAC(mike), Climber(mike)]$ from (2) and C4
(4) $[Climber(mike)]$ from (3) and C2
(5) $[\neg MemberAC(mike), Skier(mike)]$ from (4) and C11
(6) $[Skier(mike)]$ from (5) and C2
(7) [] from (6) and (2)

or



Exercise 4.7 (new)
Consider the following set of sentences.
Using resolution, prove that $Mammal(winnie)$.

S1 $\forall x[(Animal(x) \wedge HasHair(x)) \supset Mammal(x)]$

S2 $\forall x[Bear(x) \supset (Animal(x) \wedge HasHair(x))]$

S3 $\forall x[Rabbit(x) \supset Mammal(x)]$

S4 $Bear(winnie)$

S5 $Rabbit(bugsbunny)$

S6 $Animal(sylvester) \wedge HasHair(sylvester)$

Answer:

To answer the question, we need to have each of the formulas corresponding to the problem's premisses in clausal form.

C1 $[\neg Animal(x), \neg HasHair(x), Mammal(x)]$

C2, C3 $\{\neg Bear(x) \vee (Animal(x) \wedge HasHair(x))\}$
$[\neg Bear(x), Animal(x)], [\neg Bear(x), HasHair(x)]$

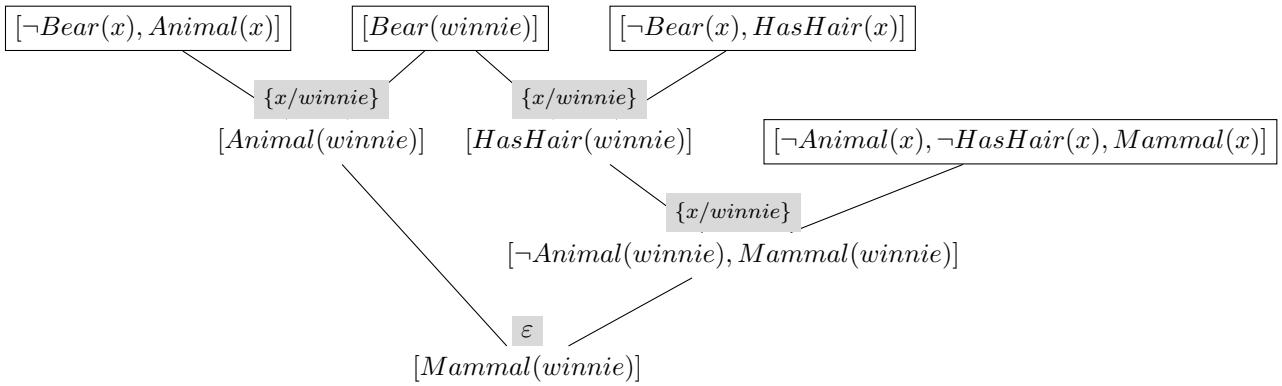
C4 $[\neg Rabbit(x), Mammal(x)]$

C5 $[Bear(winnie)]$

C6 $[Rabbit(bugsbunny)]$

C7, C8 $[Animal(sylvester)], [HasHair(sylvester)]$

To find out if Winnie is a mammal, we need to answer a True/False question. In this case, we want to know if the clause $Mammal(winnie)$ can be derived from the initial set of clauses. Because not all initial clauses are necessary, only the ones needed for the proof will be represented.



We can also show a resolution proof, instead of a resolution tree:

1	$[\neg Bear(x), Animal(x)]$	Prem
2	$[Bear(Winnie)]$	Prem
3	$[\neg Bear(x), HasHair(x)]$	Prem
4	$[\neg Animal(x), \neg HasHair(x), Mammal(x)]$	Prem
5	$[Animal(winnie)]$	Res, (1, 2), $\{x/winnie\}$
6	$[HasHair(winnie)]$	Res, (2, 3), $\{x/winnie\}$
7	$[\neg Animal(winnie), Mammal(winnie)]$	Res, (4, 6), $\{x/winnie\}$
8	$[Mammal(winnie)]$	Res, (5, 7), $\{\}$

Obviously, these are only examples of possible proofs, we could also perform refutation proof, by negating what we are trying to prove and reaching a contradiction.

Exercise 4.8 (new)
Consider the following set of sentences.
Using resolution, prove that $Thick(lordOfTheRings) \wedge \neg Thick(time)$.

S1 $\forall x[(Book(x) \wedge ManyPages(x)) \supset Thick(x)]$

S2 $\forall x[Magazine(x) \supset (\neg Thick(x) \wedge \neg Book(x))]$

S3 $Book(lordOfTheRings) \wedge Magazine(time)$

S4 $ManyPages(lordOfTheRings)$

Answer:

Exercise 4.9 (new)
Consider the following set of sentences. Using resolution, prove that $R(a, f(f(a))$.

S1 $\forall x, y, z[(R(x, y) \wedge R(y, z)) \supset R(x, z)]$

S2 $\forall x[R(x, f(x)) \supset R(f(x), f(f(x)))]$

S3 $R(a, f(a))$

Answer:

Exercise 4.10 (new)
Using resolution, prove that the formula

$$((P \supset \neg R) \wedge (\neg P \supset R)) \supset (\neg(P \wedge R) \wedge \neg(\neg P \wedge \neg R))$$

is a theorem.

Answer:

5 Reasoning with Horn Clauses

Exercise 5.1 (new)
Explain what is a Horn clause. In particular, explain its difference to a (general) clause.

Answer:

A clause is a disjunction of literals or their negations. A Horn clause is a clause that has at most um positive literal, which is the head of the clause.

Exercise 5.2 (new)
Explain how the formulas $A \rightarrow B$ and $A \vee B$ would look like as Horn clauses.

Answer:

Exercise 5.3 (new)
Explain the reason why it is necessary to use at least one positive clause when performing resolution with Horn clauses. In particular, explane why it is not possible to use two negative clauses when applying resolution with Horn clauses.

Answer:

Exercise 5.4 (new)
Explain why resolution using Horn clauses is more efficient than general resolution (using any types of clauses).

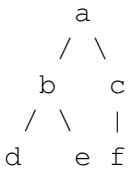
Answer:

Exercise 5.5 (from `http://www.cse.psu.edu/~catuscia/teaching/cg428/exercises/`)
Assume given a set of facts of the form `father(name1,name2)`, meaning that `name1` is the father of `name2`.

1. Define a predicate `brother(X,Y)` which holds iff `X` and `Y` are brothers.
2. Define a predicate `cousin(X,Y)` which holds iff `X` and `Y` are cousins.
3. Define a predicate `grandson(X,Y)` which holds iff `X` is a grandson of `Y`.
4. Define a predicate `descendant(X,Y)` which holds iff `X` is a descendant of `Y`.
5. Consider the following genealogical tree:

```
father(a,b).  % 1
father(a,c).  % 2
father(b,d).  % 3
father(b,e).  % 4
father(c,f).  % 5
```

   whose graphical representation is:



Say which answers, and in which order, are generated by your definitions for the queries, assuming that you ask for all the solutions (using ;).

```
?- brother(X,Y).
?- cousin(X,Y).
?- grandson(X,Y).
?- descendant(X,Y).
```

Answer:

1. `brother(X,Y) :- father(Z,X), father(Z,Y), not(X=Y).`
2. `cousin(X,Y) :- father(Z,X), father(W,Y), brother(Z,W).`
3. `grandson(X,Y) :- father(Z,X), father(Y,Z).`
4. ```
   descendent(X,Y) :- father(Y,X).
   descendent(X,Y) :- father(Z,X), descendent(Z,Y).
   ```
5. ```
   ?- brother(X,Y).
   X = b  Y = c ;
   X = c  Y = b ;
   X = d  Y = e ;
   X = e  Y = d ;
   No

   ?- cousin(X,Y).
   X = d  Y = f ;
   X = e  Y = f ;
   X = f  Y = d ;
   X = f  Y = e ;
   No

   ?- grandson(X,Y).
   X = d  Y = a ;
   X = e  Y = a ;
   X = f  Y = a ;
   No

   ?- descendent(X,Y).
   X = b  Y = a ;
   X = c  Y = a ;
   X = d  Y = b ;
   X = e  Y = b ;
   X = f  Y = c ;
   X = d  Y = a ;
   X = e  Y = a ;
   X = f  Y = a ;
   No
   ```

Exercise 5.6 (from `http://www.cse.psu.edu/~catuscia/teaching/cg428/exercises/`)
Define a predicate `mylength(L,N)` which holds iff `N` is the length of the list `L`.

Answer:

```
%   :- redefine_system_predicate(length(_,_)).
mylength([],0).
mylength([_|L],N) :- mylength(L,M), N is M+1.
```

Exercise 5.7 (from `http://www.cse.psu.edu/~catuscia/teaching/cg428/exercises/`)
Define a predicate `sumlist(L,N)` which, given a list of integers `L`, returns the sum `N` of all the elements of `L`.

Answer:

```
sumlist([],0).
sumlist([X|L],N) :- sumlist(L,M), N is M+X.
```

Exercise 5.8 (from `http://www.cse.psu.edu/~catuscia/teaching/cg428/exercises/`)
Define a predicate `occurrences(X,L,N)` which holds iff the element `X` occurs `N` times in the list `L`.

Answer:

```
occurrences(_,[],0).
occurrences(X,[X|L],N) :- occurrences(X,L,M), N is M+1.
occurrences(X,[Y|L],N) :- not(X=Y), occurrences(X,L,N).
```

Exercise 5.9 (from `http://www.cse.psu.edu/~catuscia/teaching/cg428/exercises/`)
Define a predicate `occurs(L,N,X)` which holds iff `X` is the element occurring in position `N` of the list `L`.

Answer:

```
occurs([X|_],1,X).
occurs([_|L],N,X) :- N > 1, M is N-1, occurs(L,M,X).
```

Exercise 5.10 (from `http://www.cse.psu.edu/~catuscia/teaching/cg428/exercises/`)
Define a predicate `mymerge(L,K,M)` which, given two ordered lists of integers `L` and `K`, returns an ordered list `M` containing all the elements of `L` and `K`.

Answer:

```
%:- redefine_system_predicate(merge(_,_,_)).
mymerge([X|L],[Y|K],[X|M]) :- X < Y, mymerge(L,[Y|K],M).
mymerge([X|L],[Y|K],[Y|M]) :- X >= Y, mymerge([X|L],K,M).
mymerge(L,[],L).
mymerge([],K,K).
```

using cut

```
%:- redefine_system_predicate(merge(_,_,_)).
mymerge([X|L],[Y|K],[X|M]) :- X < Y, !, mymerge(L,[Y|K],M).
mymerge([X|L],[Y|K],[Y|M]) :- X >= Y, !, mymerge([X|L],K,M).
mymerge(L,[],L) :- !.
mymerge([],K,K) :- !.
```

Exercise 5.11 (new)
Define a Prolog predicate `invertList(List,Tsil)`, which is true if `Tsil` corresponds to `List` with its elements reversed. You can generate either a recursive or iterative process.

Answer:

Exercise 5.12 (new)
Define a Prolog predicate `remove(Xs,X,Ys)`, which is true if `Ys` is the result of removing all occurrences of `X` from list `Xs`. You can generate either a recursive or iterative process.

Answer:

```
/* remove(Xs,X,Ys) :- is true if Ys results from
   removing all occurrences of X from list Xs. */
remove([X|Xs],X,Ys) :- remove(Xs,X,Ys).
remove([X|Xs],Z,[X|Ys]) :- X \= Z, remove(Xs,Z,Ys).
remove([],_,[]).
```

Exercise 5.13 (new)
Explain why the following program for `minimum3` does not produce the expected results.

```
/*
  minimum3(X,Y,Min) :- Min is the minimum of numbers X and Y.
*/
minimum3(X,Y,X) :- X =< Y, !.
minimum3(X,Y,Y).
```

Answer:

All that we have to do is to use the goal `minimum3(2,5,5)`. This goal succeeds, even though `5` i snot the minimum between its two other arguments. The problem is that this program was made thinking that the third argument would be a variable, and in this case it would work. However, when the third argument is known, we can get wrong results. The correct definition would be:

```
/*
  minimum3a(X,Y,Min) :- Min is the minimum of numbers X and Y.
*/
minimum3a(X,Y,X) :- X =< Y, !.
minimum3a(X,Y,Y) :- X > Y, !.
```

Conclusion: it is not always possible to eliminate seemingly redundant tests, because we can get wrong results if we call the program with arguments different from what was initially expected.

Exercise 5.14 (new)
Explain the problem with Prolog´s cut in the following program:

```
/*
  member3(X,L) :- X is a member of L.
*/
member3(X,[X|_]) :- !.
member3(X,[_|Ys]) :- member3(X,Ys).
```

Note: think about what happens in two different situations: when we need to know if a given element is a member of a list; and when we need to generate all the members of a given list.

Answer:

The problem is that this is a red cut, that changes the meaning of the program. When we try to use predicate `member3` to find out the elements of a list, we will only get the first one. This happens because the cut eliminates the possibility of backtracking and finding the other elements.

The following interaction illustrates some of the problems:

```
?- member3(2, [1,2,3]).
Yes
?- member3(X, [1,2,3]).
X = 1
?- member3(X, [1,2,3]), X=2.
No
?- X=2, member3(X, [1,2,3]).
X = 2
?-
```

The first goal works as expected.

The second goal only has one solution, corresponding to the list's first element, when it should have three solutions.

In the third goal the answer is `No`, when there is the solution `X = 2`.

In the fourth goal, we get that solution, because when we call `member3` the `X` is already unified with `2` and only in the recursive call to `member3` will it unify with the second element of the list.

6 Procedural Control of Reasoning

Exercise 6.1 (new)
Explain the difference between the following two Prolog rules for identifying a person´s american cousins:

R1 `americanCousin(X,Y) :- american(X), cousin(X,Y).`

R2 `americanCousin(X,Y) :- cousin(X,Y), american(X).`

Answer:

Exercise 6.2 (new)
Explain why variables in Prolog should be unified as soon as possible.

Answer:

Exercise 6.3 (new)
Explain why the order of the clauses in a Prolog program can influence its efficiency.

Answer:

7 Rules in Production Systems

Exercise 7.1 (new)
A *production system* is a forward-chaining reasoning system that uses rules of a certain form called *production rules* as its representation of general knowledge. Its basic operation cycle consists of three steps (1-recognize; 2-resolve conflict; 3-act) that are repeated until no more rules are applicable to the WM, at which point the system halts. Briefly explain each of the three steps.

Answer:

In page 119.

1. Recognize — find out wich rules are applicable, that is, which rules' antecedents are satisfied by the current WM.
2. Resolve conflict — from the rules determined in the previous step, choose the ones that must be executed.
3. Act — change the WM, by executing the consequents of the rules selected in the second step.

Exercise 7.2 (new)
Explain why conflict resolution is necessary in production systems. Name and explain two different conflict resolution strategies used by these systems.

Answer:

Exercise 7.3 (new)
In production systems, production rules have an antecedent and a consequent. Describe what can appear in each component of a production rule and give an example of one production rule.

Answer:

Exercise 7.4 (new)
Production systems use forward-chaining or backward-chaining? Explain why.

Answer:

Exercise 7.5 (from `http://www.cs.nott.ac.uk/~nza/G53KRR`)
Explain the notions of rule matching, rule instance, conflict set, and conflict resolution strategy in rule-based systems. Give two examples of common conflict resolution strategies. Illustrate your answers on the following example of rules and working memory elements. State what the conflict set is for the current state of the working memory and which rules will be fired first under each conflict resolution strategy. You can also refer to the conflict set at the next cycle, after the selected rules are fired.

F1 $animal(tiger)$
F2 $animal(cat)$
F3 $large(tiger)$

F4 $eatsMeat(tiger)$
F5 $eatsMeat(cat)$
R1 $\forall x[(animal(x) \wedge large(x) \wedge eatsMeat(x)) \supset dangerous(x)]$
R2 $\forall x[animal(x) \supset breathesOxygen(x)]$
R3 $\forall x[dangerous(x) \supset runAwayNow]$

Answer:

In rule-based systems, patterns in the body of each rule are matched against working memory elements. Each successful matching (a unification which makes the patterns identical with the working memory elements) is a rule instance. The conflict set contains all rule instances applicable for the current state of the working memory. The conflict resolution strategy is used to determine which of the rule instances in the conflict set will actually be fired. Most conflict resolution strategies pick a single rule instance based on specificity of the rules (which rule has a more specific pattern in the body), the order in which rules appear in the program, the order in which facts were added to working memory (for example, depth first where rule instances involving more recent facts are preferred) etc.

In this example, the conflict set is: **R1** with $x/tiger$ (matching **F1**, **F3**, **F4**), **R2** with $x/tiger$ (matching **F1**) and x/cat (matching **F2**). Under the rule order and most specific rule first, the first rule instance will be selected to assert $dangerous(tiger)$ into the working memory. This adds a new matching rule instance and the conflict set is **R2** with $x/tiger$ (matching **F1**) and x/cat (matching **F2**), **R3** with $x/tiger$. Now under the order of rules strategy the next fact to be fired will be an instance of **R2**, for example and under more recent first strategy, the instance of **R3**, because it matches the most recent fact.

Exercise 7.6 (from `http://www.cs.nott.ac.uk/~nza/G53KRR`)
Explain how decision tables can be used for knowledge elicitation and designing a rule-based expert system.

Answer:

Decision tables can be used to structure the knowledge before producing a rule-based expert system. First we need to decide on all conditions relevant for making a decision, and all possible actions. A decision table consists of a list of possible relevant conditions, relevant actions, condition alternatives (yes, no, and - for not relevant, for each condition) and action entries (which action is to be taken if specified conditions hold). Each column in the table corresponds to a rule (what to do if the pattern of conditions applies).

Exercise 7.7 (from `http://www.cs.nott.ac.uk/~nza/G53KRR`)
Suppose that all you have to work with in designing a rule-based expert system for recognising spam email is the following set of correctly classified messages. Produce a decision table based on this set of examples. Do not include irrelevant checks in the rules.

Message1 Properties: has an attachment, does not contain images, sender is in the receiver's address book, subject line contains "Prize". Decision: spam.
Message2 Properties: no attachments, contains images, sender is not in the receiver's address book, subject line contains "Goods". Decision: spam.
Message3 Properties: has an attachment, contains images, sender is in the receiver's address book, subject line contains "Prize". Decision: spam.

Message4 Properties: no attachments, does not contain images, sender is not in the receiver's address book, subject line does not contain "Prize"or "Goods". Decision: not spam.

Message5 Properties: has an attachment, does not contain images, sender is not in the receiver's address book, subject line contains "Prize". Decision: spam.

Message6 Properties: has no attachments, contains images, sender is in the receiver's address book, subject line contains "Goods". Decision: not spam.

Message7 Properties: has no attachments, does not contain images, sender is not in the receiver's address book, subject line contains "Goods". Decision: spam.

Message8 Properties: has no attachments, contains images, sender is not in the receiver's address book, subject line does not contain "Prize"or "Goods". Decision: not spam.

Answer:

We want to make as few tests as possible, but still correctly classify all the examples given in the text.

	1	2	3	4	5	6	7	8	1+3+5	2+7	4+8	6
Has attachment	Y	N	Y	N	Y	N	N	N	Y	-	N	N
Contains images	N	Y	Y	N	N	Y	N	Y	-	-	-	-
Sender in address book	Y	N	Y	N	N	Y	N	N	-	N	-	Y
Subject contains "Prize"	Y	N	Y	N	Y	N	N	N	-	-	-	-
Subject contains "Goods"	N	Y	N	N	N	Y	Y	N	-	Y	N	-
Spam	X	X	X		X		X		X	X		
Not spam				X		X		X			X	X

8 Object-Oriented Representation

Exercise 8.1 (new)
Explain the difference between **if-added** and **if-needed** procedures in frames.

Answer:

Exercise 8.2 (new)
Explain the difference between frame systems and object oriented programming

Answer:

A procedural frame system shares the advantages of a conventional OOP system: Definition is done primarily by specialization of more general classes, control is localized, methods can be inherited, encapsulation of abstract procedures is possible, and so on. The main difference is that frame systems tend to have a centralized, conventional control regime, whereas OOP systems have objects acting as small, independent agents sending each other messages. Frame systems tend to work in a cycle: Instantiate a frame and declare some slot fillers, inherit values from more general frames, trigger appropriate forward-chaining procedures, and then, when quiescent, stop and wait for the next input. OOP systems tend to be more decentralized and less patterned.

Exercise 8.3 (new)
Explain the similarities between frame systems and object oriented programming.

Answer:

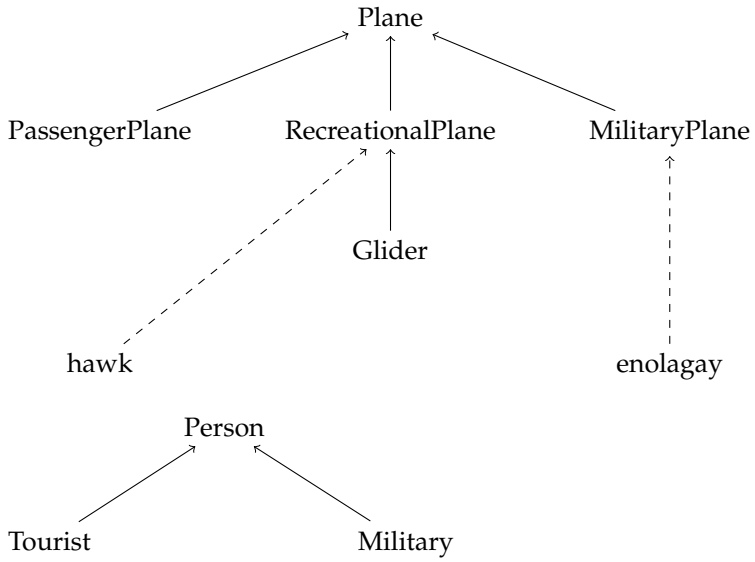
Frame-based representation languages and OOP systems were developed concurrently, and share many of the same intuitions and techniques. A procedural frame system shares the advantages of a conventional OOP system: definition is done primarily by specialization of more general classes, control is localized, methods can be inherited, encapsulation of abstract procedures is possible, and so on.

Exercise 8.4 (new)
Consider the following information.

> There are several types of planes: passenger, recreational and military planes. Different types of planes are distinguished according to the people that they transport: people in general, tourists or military personnel, respectively. Each plane can have zero or more motors. Gliders are recreational planes without motor, while passenger planes generally have two motors. The total weight of a plane can be estimated by summing the plane´s weight to the weight of its passengers and its cargo. "Hawk" is a recreational plane and "Enolagay" is a military plane.

1. Represent the hierarchy implicit in this information.
2. Design a set of frames and slots to represent this information.
3. Write in English pseudo-code the **if-added** or **if-needed** procedures that would appear in your representation.

Answer:

1. 


2. ```
(Plane
  <:PassengerList List of Person>
  <:NumMotors PositiveInteger>
  <:TotalWeight PROCEDURE>
  <:OwnWeight PositiveInteger>
  <:CargoWeight PositiveInteger>)
(PassengerPlane
  <:IS-A Plane>
  <:NumMotors 2>)
(RecreationalPlane
  <:IS-A Plane>
  <:PassengerList List of Tourist>)
(MilitaryPlane
  <:IS-A Plane>
  <:PassengerList List of Military>)
(Glider
  <:IS-A RecreationalPlane>
  <:NumMotors 0>)
(hawk
  <:INSTANCE-OF RecreationalPlane>)
(enolagay
  <:INSTANCE-OF MilitaryPlane>)

(Person
  <:Weight PositiveInteger>)
(Tourist
  <:IS-A Person>)
(Military
  <:IS-A Person>)
```

3. The only procedure is an **if-needed** procedure that is called when the plane's total weight needs to be determined. This procedure will iterate over the plane's passenger list and sum everyone's weight and then sum the plane's own weight and the plane's cargo weight to get the total weight.
```

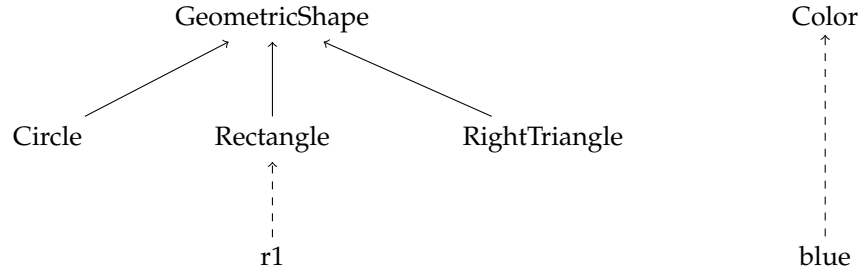
Exercise 8.5 (new)
Consider the following information.

> We need to represent information about several kinds of geometric shapes, namely how many sides they have, their color, area and perimeter. We want to represent circles, rectangles and right triangles. For each of these types of geometric shapes we want to be able to calculate its area and its perimeter. R1 is a blue rectangle that measures 10cm by 5cm.

1. Represent the hierarchy implicit in this information.
2. Design a set of frames and slots to represent this information.
3. Write in English pseudo-code the **if-added** or **if-needed** procedures that would appear in your representation.

Answer:

1.



2.
```
(GeometricShape
  <:NumSides PositiveInteger>
  <:Color Color>
  <:Area PositiveReal>
  <:Perimeter PositiveReal>)
(Circle
  <:IS-A GeometricShape>
  <:NumSides 1>          ;;; would also accept 0; INFINITY; UNDEFINED
  <:Area [IF-NEEDED ProcAreaCircle]>
  <:Perimeter [IF-NEEDED ProcPerimeterCircle]>
  <:Radius PositiveReal>)
(Rectangle
  <:IS-A GeometricShape>
  <:NumSides 4>
  <:Area [IF-NEEDED ProcAreaRectangle]>
  <:Perimeter [IF-NEEDED ProcPerimeterRectangle]>
  <:Length PositiveReal>
  <:Width PositiveReal>)
(RightTriangle
  <:IS-A GeometricShape>
  <:NumSides 3>
  <:Area [IF-NEEDED ProcAreaRightTriangle]>
  <:Perimeter [IF-NEEDED ProcPerimeterRightTriangle]>
  <:Base PositiveReal>
  <:Height PositiveReal>)
```

```
(r1
  <:INSTANCE-OF Rectangle>
  <:Color blue>
  <:Length 10>
  <:Width 5>)
```

3. All the procedures are **if-needed** procedures that are called when the figure's area and perimeter need to be determined.

```
ProcAreaCircle
  return 3,14 * SELF:Radius * SELF:Radius

ProcPerimeterCircle
  return 2 * 3,14 * SELF:Radius

ProcAreaRectangle
  return SELF:Length * SELF:Width

ProcPerimeterRectangle
  return 2 * (SELF:Length + SELF:Width)

ProcAreaRightTriangle
  return (SELF:Base * SELF:Height) / 2

ProcPerimeterRightTriangle
  return SELF:Base + SELF:Height + sqrt(sqr(SELF:Base) + sqr(SELF:Height))
```

Exercise 8.6 (new)
We are interested in representing information about several kinds of people, namely their hair color, their body mass index (BMI), and the interpretation of this index. Someone's BMI is calculated as the person's weight in kilos divided by the square of the person's height in meters. The interpretation of BMI is as follows: BMI less than 18,5 — underweight; BMI between 18,5 and 25 — normal weight; BMI between 25 and 29,9 — overweight; BMI over 30 — obesity. Jack is a person with brown hair.

1. Represent the hierarchy implicit in this information.
2. Design a set of frames and slots to represent this information.
3. Invent the values that are needed to calculate and interpret Jack's BMI.
4. Write in English pseudo-code the **if-added** or **if-needed** procedures that would appear in your representation.

Answer:

Exercise 8.7 (new)
We are interested in representing information about several types of insurances, that are differentiated according to the type of object that is insured. In particular, we are interested in representing insurance for the filling of the house, insurance for the walls of the house and auto insurance. There are also multi-risk home insurances, which include both insurance for the filling of the house and insurance for the walls of the house. The prize for each type of insurance is calculated as a percentage of the insured value: 0.5% for the walls, 1% for the filling and 2% for auto insurance. GingerbreadHouse is a house and has a multi-risk insurance. Herbie is an automobile.

1. Represent the hierarchy implicit in this information.
2. Design a set of frames and slots to represent this information.
3. Write in English pseudo-code the **if-added** or **if-needed** procedures that would appear in your representation.
4. Invent the values that are needed to calculate the value of the insurance for GingerbreadHouse, represent them, and calculate the value of the insurance.

Answer:

Exercise 8.8 (Ch 8, Ex 1)
Consider a possible frame-based application for a classroom scheduler.
We want to build a program that helps schedule rooms for classes of various sizes at a university, using the sort of frame technology (frames slots and attached procedures) discussed in the text. Slots of frames might be used to record when and where a class is to be held, the capacity of a room, and so on, and **if-added** and other procedures might be used to encode constraints as well as to fill in implied values when the KB is updated. In this problem, we want to consider updating the KB in several ways: (1) asserting that a class of a given size is to be held in a given room at a given time; the system would either go ahead and add this to its schedule or alert the user that it was not possible to do so; (2) asserting that a class of a given size is to be held at a given time, with the system providing a suitable room (if one is available) when queried; (3) asserting that a class of a given size is desired, with the system providing a time and a place when queried.

1. Design a set of frames and slots to represent the schedule and any ancillary information needed by the assistant.
2. For all slots of all frames, write in English pseudo-code the **if-added** or **if-needed** procedures that would appear there. Annotate these procedures with comments explaining why they are there (e.g., what constraints they are enforcing).
3. Briefly explain how your system would work (what procedures would fire and why they do) on concrete examples of your choosing, illustrating each of the three situations mentioned in the description of the application.

Answer:

9 Structured Descriptions

Exercise 9.1 (new)
What is the point of description logics (DL)? Why don't knowledge representation professionals use first-order logic for everything?

Answer:

The main reason for using resticted ontology languages is that restriction in expressive power makes reasoning in them much more efficient than in first order logic. Another reason is that hierarchies of concepts, and entity-relationship diagrams are familiar to many users and are considered more intuitive.

Exercise 9.2 (new)
Explain how concept classification can be considered as a form of reasoning in description logics.

Answer:

Exercise 9.3 (new)
Explain in which situations a formula $a \rightarrow b$ is true in description logics.

Answer:

Exercise 9.4 (new)
Explique quando é que uma fórmula do tipo $d1 \sqsubseteq d2$ tem o valor verdadeiro nas lógicas descritivas.

Answer:

Exercise 9.5 (from `http://www.cs.nott.ac.uk/~nza/G53KRR`)
Assume that you have an atomic concept `Woman`, roles `Child` and `Employer`, and a constant `ist` for Instituto Superior Técnico. Define the following concepts:

1. Extra-busy is a working mother employed by Instituto Superior Técnico.
2. Someone all of whose children only have female children themselves (that is a person who only has granddaughters, if he or she has any grandchildren).
3. Someone who has children, and all of whose children have children.

Answer:

1. `ExtraBusy` $\doteq$ `[AND Woman [EXISTS 1 :Child] [FILLS :Employer ist]]`
2. `[ALL :Child Woman]` describes someone all of whose children are female, and we want to say that someone's children are described by this concept, so we say `[ALL :Child [ALL :Child Woman]]`.

3. To say that someone has children we can use `[EXISTS 1 :Child]` and to describe someone all of whose children have children we can use `[ALL :Child [EXISTS 1 :Child]]`. Since we want the concept to satisfy both properties, we say `[AND [EXISTS 1 :Child] [ALL :Child [EXISTS 1 :Child]]]`.

Exercise 9.6 (from `http://www.cs.nott.ac.uk/~nza/G53KRR`)
Answer the following questions:

1. Do $d1 \sqsubseteq d2$ and $d2 \sqsubseteq d3$ entail $d1 \sqsubseteq d3$?
2. Do $c \rightarrow d1$ and $d2 \sqsubseteq d1$ entail $c \rightarrow d2$?
3. Do $c \rightarrow d1$ and $d1 \sqsubseteq d2$ entail $c \rightarrow d2$?

Answer:

1. Yes: if $d1 \sqsubseteq d2$ and $d2 \sqsubseteq d3$ are true it means that $I(d1) \subseteq I(d2)$ and $I(d2) \subseteq I(d3)$ so $I(d1) \subseteq I(d3)$, and the latter means that $d1 \sqsubseteq d3$ is true.
2. No: Consider I such that $I(c) \in I(d1)$ and $I(d2)$ is empty. Then the first two sentences are true but $c \notin I(d2)$ so $c \rightarrow d2$ is false.
3. Yes: If the first two sentences are true, then $I(c) \in I(d1)$ and $I(d1) \subseteq I(d2)$ so it has to hold that $I(c) \in I(d2)$ which means $c \rightarrow d2$ is true.

Exercise 9.7 (from `http://www.cs.nott.ac.uk/~nza/G53KRR`)
Consider a description logic with the following definition of a concept (note that it is slightly different from the one in the textbook, namely the first concept constructor is new and the forth concept constructor is different from `[EXISTS n r]`):

- $\top$ is a special atomic concept which describes any object (it is a property which is trivially true for everything).
- An atomic concept is a concept.
- If r is a role and b is a concept, then `[ALL r b]` is a concept (describing objects all of whose r-successors are described by b).
- If r is a role and b is a concept, then `[EXISTS r b]` is a concept (describing objects which have at least one r-successor which is described by b).
- If r is a role and c is a constant, then `[FILLS r c]` is a concept (describing objects which have an r-successor denoted by c).
- If $b_1, \ldots, b_n$ are concepts, `[AND` b_1 `...`b_n`]` is a concept (describing objects which are described by all of $b_1, \ldots, b_n$).

and the following definition of a sentence:

- If b_1 and b_2 are concepts then $b_1 \sqsubseteq b_2$ is a sentence (all b_1s are b_2s).
- If b_1 and b_2 are concepts then $b_1 \doteq b_2$ is a sentence (b_1 is equivalent to b_2).

- If c is a constant and b a concept then $c \rightarrow b$ is a sentence (the individual denoted by c satisfies the description expressed by b).

1. Given the atomic concepts `Female`, `Male`, `Person` roles `:Child`, `:Sibling` and constant `alice`, define in the description logic above the following concepts:
   (a) "Mother of Alice" (someone female whose child is Alice).
   (b) "Parent" (someone who has a child).
   (c) "Uncle" (someone male who has a sibling who has a child).
2. Using the same atomic concepts, translate the following sentences into description logic:
   (a) Every grandparent is a parent.
   (b) Alice is a grandmother.

Answer:

1. (a) `[AND Female [FILLS :Child alice]]`
   (b) `[EXISTS :Child Person]`
   (c) `[AND Male [EXISTS :Sibling [EXISTS :Child Person]]]`
2. (a) `[EXISTS :Child [EXISTS :Child Person]]` $\sqsubseteq$ `[EXISTS :Child Person]`
   (b) `alice` $\rightarrow$ `[AND Female [EXISTS :Child [EXISTS :Child Person]]]`

Exercise 9.8 (from `http://www.cs.nott.ac.uk/~nza/G53KRR`)
Recall the description logic DL given in the textbook.
Concepts:

- An atomic concept is a concept.
- If r is a role and b is a concept, then `[ALL` r b`]` is a concept (e.g. `[ALL :Child Girl]` describes someone all of whose children are girls).
- If r is a role and n is a positive integer, then `[EXISTS` n r`]` is a concept (e.g. `[EXISTS 2 :Child]` describes someone who has at least 2 children).
- If r is a role and c is a constant, then `[FILLS` r c`]` is a concept (e.g. `[FILLS :Child john]` describes someone whose child is John).
- If $b_1, \ldots, b_n$ are concepts, `[AND` b_1 $\ldots b_n$`]` is a concept.

Sentences:

- If b_1 and b_2 are concepts then $b_1 \sqsubseteq b_2$ is a sentence (all b_1s are b_2s).
- If b_1 and b_2 are concepts then $b_1 \doteq b_2$ is a sentence (b_1 is equivalent to b_2).
- If c is a constant and b a concept then $c \rightarrow b$ is a sentence (the individual denoted by c satisfies the description expressed by b).

1. Express the following concepts and sentences in DL using constants `john`, `krr`, roles `:Module` and `:Supervision` and atomic concepts `Academic`, `Lecturer`, `Compulsory`:

   C1 Concept of an academic who has some project students (supervises the students).

   C2 Concept of an academic who teaches at least two modules.

   C3 Concept of an academic who teaches only compulsory modules.

   C4 Concept of someone who teaches KRR.

   S1 A lecturer is an academic who has at least 8 project students and teaches at least 2 modules.

   S2 John teaches at least 3 modules and they are all compulsory.

2. At the moment the logic does not contain concept negation `NOT`. It also cannot say that there exists some individual connected by a role which is in a concept b (namely, we have `[ALL` r b`]` but no `[EXISTS` r b`]`). If we add concept negation `NOT`, with the obvious meaning that `[NOT` b`]` is a concept containing all individuals which are not in b, explain how we can then define `[EXISTS` r b`]`.

Answer:

1. **C1** `[AND Academic [EXISTS 1 :Supervision]]`

   C2 `[AND Academic [EXISTS 2 :Module]]`

   C3 `[AND Academic [ALL :Module Compulsory]]`

   C4 `[FILLS :Module krr]`

   S1 `Lecturer` $\doteq$ `[AND Academic [EXISTS 8 :Supervision] [EXISTS 2 :Module]]`

   S2 `john` $\rightarrow$ `[AND [EXISTS 3 :Module] [ALL :Module Compulsory]]`

2. `[EXISTS` r b`] = [NOT [ALL` r `[NOT` b`]]]`

Exercise 9.9 (new)
Express the following concepts and sentences in DL using the constants, roles and atomic concepts that you find the most useful.

> Computers have at least one input device and one output device, which are input devices and output devices, respectively. Keyboards and mice are different types of input devices. Screens and columns are different types of output devices. C1 is a computer whose keyboard is K1.

What can be inferred about K1?

Answer:

```
Computer ⊑
   [AND [ALL :InputDev InputDevice]
        [EXISTS 1 :InputDev]
        [ALL :OutputDev OutputDevice]
        [EXISTS 1 :OutputDev]]
```

```
Keyboard ⊑ InputDevice

Mouse ⊑ InputDevice

Screen ⊑ OutputDevice

Column ⊑ OutputDevice

c1 → [AND Computer [FILLS :InputDev k1]]

k1 → Keyboard
```

About K1 we know that it is a keyboard and that it is C1's input device. It is possible to infer that it is an input device.

Exercise 9.10 (new)
Express the following concepts and sentences in DL using the constants, roles and atomic concepts that you find the most useful.

> There are several types of drinks: water, alcoholic drinks and fruit drinks. Drinks are described by their ingredients, which are edible stuff. Alcoholic drinks are also described by their alcohol contents, which is an integer. W1 is a wine whose alcohol contents is 12. F2 is a fruit drink containing water, pineapple juice and coconut juice.

What can be inferred about F2?

Answer:

```
Drink ⊑ [ALL :Ingredients EdibleStuff]

Water ⊑ Drink

AlcoholicDrink ⊑ [AND Drink [ALL :AlcoholContents Integer]]

FruitDrink ⊑ Drink

Wine ⊑ AlcoholicDrink

w1 → [AND Wine [FILLS :AlcoholContents 12]]

f2 →
   [AND FruitDrink
        [FILLS :Ingredients PineappleJuice]
        [FILLS :Ingredients Water]
        [FILLS :Ingredients CoconutJuice]]

PineappleJuice ⊑ Drink

CoconutJuice ⊑ Drink

Drink ⊑ EdibleStuff
```

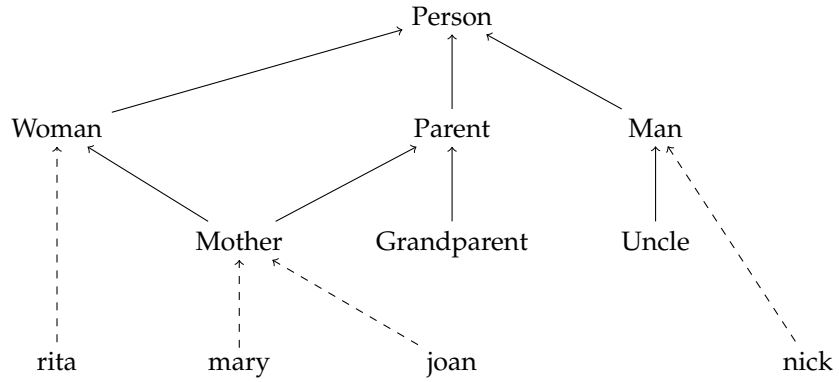
About F2 we can infer that it is a drink and that it is edible stuff.

Exercise 9.11 (new)
Express the following concepts and sentences in DL using the constants, roles and atomic concepts that you find the most useful. Start by drawing the hierarchy that is implicit in your representation. In the end, indicate what can be inferred from the information that you represented.

> We want to represent information about people and the relationships between them. Mothers are women with children. Uncles are men whose siblings have children. Grandparents are people whose children have children. Mary is the mother of Rita. Nick is Mary's brother. Joan is the mother of Mary and Nick.

Answer:

Despite the fact that "Parent" does not appear in the text, the concept of someone with children does, so this concept will also be represented.




```
Woman ⊑ Person

Parent ≐
  [AND Person
       [EXISTS 1 :child]
       [ALL :child Person]]

Man ⊑ Person

Mother ≐ [AND Woman Parent]

Uncle ≐
  [AND Man
       [EXISTS 1 :siblingWithChild]
       [ALL :siblingWithChild Parent]]

Grandparent ≐
  [AND Parent
       [EXISTS 1 :childWithChild]
       [ALL :childWithChild Parent]]
```

```
mary → [AND Mother [FILLS :child rita]]

rita → Woman

nick → [AND Man [FILLS :siblingWithChild mary]]

joan → [AND Mother [FILLS :child mary] [FILLS :child nick]]
```

We can infer that:

- Mary is a Woman, a Parent and a Person.
- Rita is a Person.
- Nick is a Person and an Uncle of Rita.
- Joan is a Grandparent of Rita. She is also a Woman, a Parent and a Person.

Exercise 9.12 (new)
Express the following concepts and sentences in DL using the constants, roles and atomic concepts that you find the most useful. Explicitly state what can be inferred from this information.

> Trees have a trunk and several branches and roots (which are trunks, branches and roots, respectively). Fruit trees are trees that grow fruits. Orange trees are trees that grow oranges. O1 is an orange tree with trunk T1 that grew orange O2.

Answer:

```
Tree ⊑
  [AND [ALL :trunk Trunk]
       [EXISTS 1 :trunk]
       [ALL :branches Branch]
       [ALL :root Root]
       [EXISTS 1 :root]]

Fruittree ⊑
  [AND Tree
       [ALL :grows Fruit]]

Orange ⊑ Fruit

OrangeTree ⊑
  [AND Fruittree
       [ALL :grows Orange]]

O1 →
  [AND Fruittree
       [FILLS :trunk T1]
       [FILLS :grows O2]]

T1 → Trunk

O2 → Orange
```

We can infer that O1 is a Fruittree and a Tree and that O2 is a Fruit.

Exercise 9.13 (new)
Express the following concepts and sentences in the description logic presented in the book, using the constants, roles and concepts that you find the most useful. Start by drawing the hierarchy implicit in this information. In the end, explicitly state what can be inferred from your representation.

> We need to represent information about electronic devices (ED) and its use. Tablets are EDs with a touch-sensitive screen. Computers are EDs with at least one processor and a (non-touch-sensitive) screen. "c1"is a computer and "t1" is a tablet.

Answer:

Exercise 9.14 (new)
Express the following concepts and sentences in the description logic presented in the book, using the constants, roles and concepts that you find the most useful. Start by drawing the hierarchy implicit in this information. In the end, explicitly state what can be inferred from your representation.

> It is intended to represent information about logic gates: AND, OR and NOT gates. Logic gates have at least one input and exactly one output, all of which are logical values. The possible logical values are TRUE and FALSE. AND gates have two inputs and one output. A1 is an AND gate with inputs TRUE and FALSE.

Answer:

10 Inheritance

Exercise 10.1 (new)
Explain the difference between *strict inheritance* and *defeasible inheritance*. Explain the main problem that can arise when using defeasible inheritance.

Answer:

Exercise 10.2 (new)
Explain why the concept of inheritance is useful. Explain the difference between single and multiple inheritance and state a problem that can arise when using multiple inheritance.

Answer:

Exercise 10.3 (new)
Name and explain the two initial strategies for defeasible inheritance discussed in the book.

Answer:

The *shortest path* heuristic says that we should prefer conclusions resulting from shorter paths in the network. The *inferential distance* says that we should prefer conclusions resulting from nodes that are closer in the inheritance network. A node a is considered closer to node b than to node c according to inferential distance if and only if there is a path from a to c through b, regardless of the actual length of any paths from a to b and to c.

Exercise 10.4 (new)
Give an example of an inheritance network where the inheritance mechanisms studied in chapter 10 of the book can find the desired extension but default logic cannot. Explain why this happens.

Answer:

Exercise 10.5 (new)
Under what circumstances do a credulous reasoner and a skeptical reasoner believe in exactly the same conclusions?

Answer:

Whenever there is a single preferred extension.

Exercise 10.6 (Ch 10, Ex 1)
Consider the following collection of assertions:

George is a Marine.

George is a chaplain.

A Marine is typically a beer drinker.

A chaplain is typically not a beer drinker.

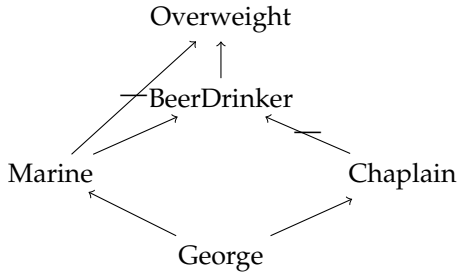
A beer drinker is typically overweight.

A Marine is typically not overweight.

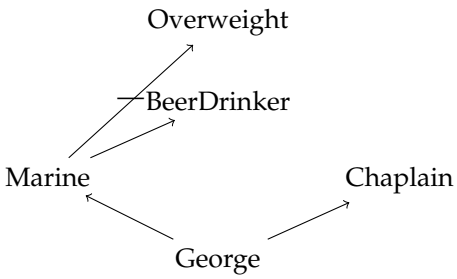
1. Represent the assertions in an inheritance network.
2. What are the credulous extensions of the network?
3. Which of them are preferred extensions?
4. Give a conclusion that a credulous reasoner might make but that a skeptical reasoner would not.

Answer:

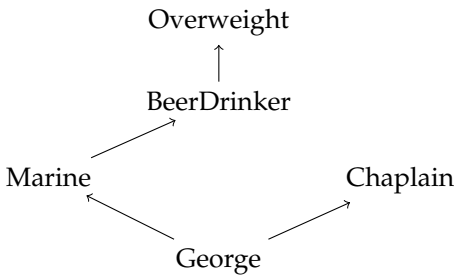
1.



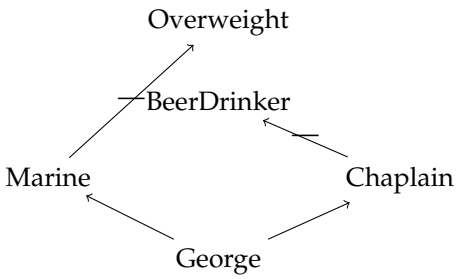
2. A credulous extension of Γ wrt node a is a maximal unambiguous a-connected subhierarchy of Γ wrt a.



(1)



(2)



(3)

3. A credulous extension is a preferred extension if there is no other extension that is preferred to it. One credulous extension is preferred over another if it has the "most specific" edges. In this case, (1) is preferred over (2) because the edge BeerDrinker · Overweight is in (2) and not in (1). Moreover, (3) is preferred over no other and no other is preferred over it. So, (1) and (3) are the preferred extensions.
4. Remember the three types of reasoning:
   - Credulous reasoning: choose a preferred extension and believe all the conclusions supported.
   - Skeptical reasoning: believe the conclusions from any path that is supported by all preferred extensions.

- Ideally skeptical reasoning: believe the conclusions that are supported by all preferred extensions. Note: ideally skeptical reasoning cannot be computed in a path-based way (conclusions may be supported by different paths in each extension).

In this case, a credulous reasoner would believe in BeerDrinker (if it chose (1)), but a skeptical reasoner would not, because it is in (1) but not in (3).

Exercise 10.7 (new)
Consider the following collection of assertions:

a is a B.

a is a C.

Bs are typically Es.

Cs are typically not Es.

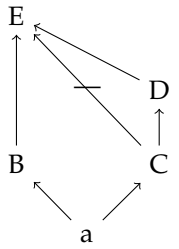
Cs are typically Ds.

Ds are typically Es.

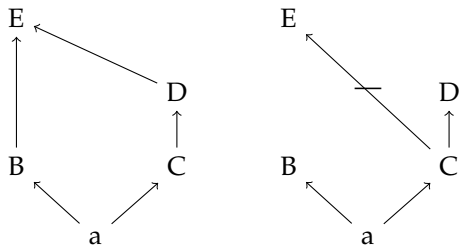
1. Represent the assertions in an inheritance network.
2. What are the credulous extensions of the network?
3. Which of them are preferred extensions?
4. Give a conclusion that a credulous reasoner might make but that a skeptical reasoner would not.

Answer:

1.



2.



(1) (2)

3. In this case no extension is preferred over any other. So, (1) and (2) are the preferred extensions.

4. In this case, a credulous reasoner would believe in E (if it chose (1)), but a skeptical reasoner would not, because it is in (1) but not in (2).

Exercise 10.8 (Ch 10, Ex 1)
Consider the following collection of assertions:

Dick is a Quaker.

Dick is a Republican.

Quakers are typically pacifists.

Republicans are typically not pacifists.

Republicans are typically promilitary.

Pacifists are typically not promilitary.

Promilitary (people) are typically politically active.

Pacifists are typically politically active.

1. Represent the assertions in an inheritance network.

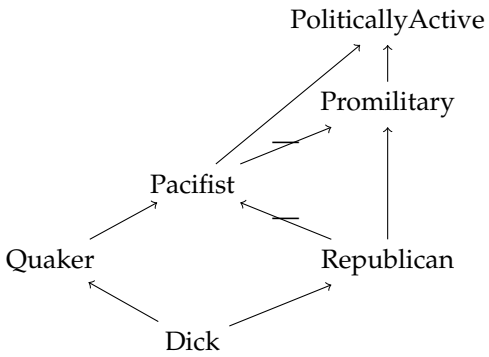
2. What are the credulous extensions of the network?

3. Which of them are preferred extensions?

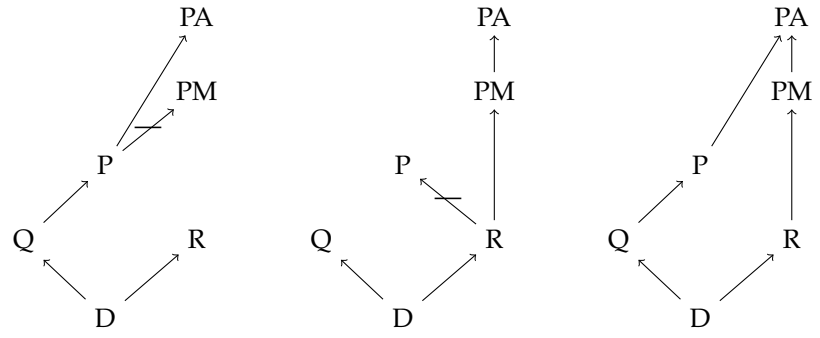
4. Give a conclusion that a credulous reasoner might make but that a skeptical reasoner would not.

Answer:

1.



2. There are two conflicts, so there are at most 2^2 extensions



(1) (2) (3)

3. In this case, (3) is preferred over (1) because the edge R · PM is in (1) and not in (3). There is no preference between (1) and (2). So, (2) and (3) are the preferred extensions.
4. In this case, a credulous reasoner would believe in Not Pacifist (if it chose (2)) or Pacifist (if it chose (3)), but a skeptical reasoner would not.

Exercise 10.9 (new)
Consider the following collection of assertions:

a is a C.

a is a D.

b is a E.

Cs are typically Es.

Ds are typically not Es.

Es are typically not Fs.

1. Represent the assertions in an inheritance network.
2. What are the credulous extensions of the network?
3. Which of them are preferred extensions?
4. Give a conclusion that a credulous reasoner might make but that a skeptical reasoner would not.

Answer:

11 Defaults

Exercise 11.1 (new)
Explain the need for default reasoning. Use a suitable example.

Answer:

Exercise 11.2 (new)
Explain the difference between a generic rule and a universal rule. Give a suitable example for each of them.

Answer:

Exercise 11.3 (new)
Explain what the closed world assumption is and why it can be considered as a non-monotonic reasoning mechanism.

Answer:

In general terms, the assumption here, called the closed-world assumption (CWA), is the following: Unless an atomic sentence is known to be true, it can be assumed to be false. Note that expressed this way, the CWA can be seen to involve a form of default reasoning. A sentence assumed to be false could later be determined in fact to be true.

Exercise 11.4 (new)
Say if each of the following default rules makes sense, from a representational point of view. Justify your answer.

1. $\frac{A(x) : B(x) \wedge C(x)}{B(x)}$
2. $\frac{A(x) : B(x)}{A(x)}$
3. $\frac{A(x) : B(x)}{B(x)}$
4. $\frac{A(x) : B(x)}{C(x)}$
5. $\frac{: \neg A(x)}{A(x)}$
6. $\frac{: A(x)}{A(x)}$
7. $\frac{A(x) :}{B(x)}$

Answer:

O objectivo deste exercício é analizar a forma das regras de omissão que escrevermos quando estivermos a representar conhecimento, para nos dar a capacidade de vermos se elas fazem ou não sentido. Do ponto de vista da LOR, todas elas podem ser usadas, à excepção da última (como a justificação é vazia, não está sintacticamente correcta).

1. $\frac{A(x):B(x)}{B(x)}$ — Esta é uma regra de omissão normal. Estas são as regras de omissão mais usadas, para tirar partido das propriedades das teorias de omissão normais, como por exemplo a existência de pelo menos uma extensão.
2. $\frac{A(x):B(x)\wedge C(x)}{B(x)}$ — Esta é uma regra de omissão semi-normal. Estas regras de omissão também são muito usadas, apesar de as teorias de omissão que usam regras com esta forma já não terem garantia de extensão.
3. $\frac{A(x):B(x)}{C(x)}$ — As regras com esta forma não são tão usadas como as regras com as duas formas anteriores, mas podem perfeitamente fazer sentido, do ponto de vista da Representação do Conhecimento.
4. $\frac{A(x):B(x)}{A(x)}$ — Não faz sentido: se temos que saber $A(x)$ para podermos aplicar a regra, não adianta nada aplicá-la para voltarmos a concluir $A(x)$.
5. $\frac{:\neg A(x)}{A(x)}$ — Não faz sentido: se for consistente assumir $\neg A(x)$, não faz sentido concluir $A(x)$, até porque $\neg A(x)$ pode já estar na base de conhecimento.
6. $\frac{:A(x)}{A(x)}$ — Faz sentido. É uma regra de omissão normal, mas em que a pré-condição é vazia.
7. $\frac{A(x):}{B(x)}$ — Não faz sentido: uma vez que a justificação da regra é vazia, deveríamos ter usado uma regra universal $\forall(x)[A(x) \supset B(x)]$. Esta regra é a única que não está sintacticamente correcta do ponto de vista da LOR.

Exercise 11.5 (Ch 11, Ex 1)
Although the inheritance networks of Chapter 10 are in a sense much weaker than the other formalisms considered in this chapter for default reasoning, they use default assertions more fully. Consider the following assertions:

> Canadians are typically not francophones.
>
> All Quebecois are Canadians.
>
> Quebecois are typically francophones.
>
> Robert is a Quebecois.

Here is a case where it seems plausible to conclude by default that Robert is a francophone.

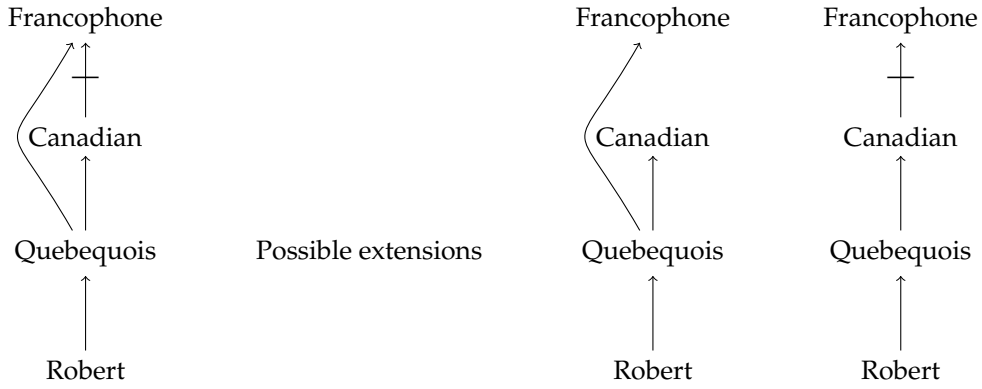
1. Represent these assertions in an inheritance network (treating the second assertion as defeasible), and argue that it unambiguously supports the conclusion that Robert is a francophone.
2. Represent these assertions in first-order logic using two abnormality predicates, one for Canadians and one for Quebecois, and argue that, as it stands, minimizing abnormality would not be sufficient to conclude that Robert is a francophone.
3. Show that minimizing abnormality would work if we add the assertion
   "All Quebecois are abnormal Canadians",
   but will not work if we only add
   "Quebecois are typically abnormal Canadians".

4. Repeat the exercise in default logic. Represent the assertions as two facts and two normal default rules, and argue that the result has two extensions. Eliminate the ambiguity using a non-normal default rule.

5. Write a variable-free version of the assertions in autoepistemic logic, and show that the procedure described in the text generates two stable expansions. How can the unwanted expansion be eliminated?

Answer:

1.



(1) (2)

This inheritance network supports the conclusion that Robert is a francophone because he is a Quebequois and Quebequois are francophones. This rule is more specific than the one that states that Canadians are not francophones, so it takes precedence over it, which is why extension (1) is preferred over extension (2).

2. In first-order logic we have:

   S1 $\forall x[(Canadian(x) \wedge \neg Ab_1(x)) \supset \neg Francophone(x)]$

   S2 $\forall x[Quebequois(x) \supset Canadian(x)]$

   S3 $\forall x[(Quebequois(x) \wedge \neg Ab_2(x)) \supset Francophone(x)]$

   S4 $Quebequois(robert)$

   The goal here is to minimize abnormality.
   Because there are two abnormality predicates and one constant, we have four possibilities of extensions:

   - $PE1 \models \{\neg Ab_1(robert), \neg Ab_2(robert)\}$ this one would generate an inconsistent extension, because it would also have to satisfy $\{Francophone(robert), \neg Francophone(robert)\}$. So this is NOT an extension.
   - $PE2 \models \{\neg Ab_1(robert), Ab_2(robert)\}$ in this case, $PE2 \models \{\neg Francophone(robert)\}$ and everything is consistent. So, this IS one possible extension.
   - $PE3 \models \{Ab_1(robert), \neg Ab_2(robert)\}$ in this case, $PE3 \models \{Francophone(robert)\}$ and everything is consistent. So, this IS one possible extension.
   - $PE4 \models \{Ab_1(robert), Ab_2(robert)\}$ this is NOT an extension because it is not minimal regarding the abnormality predicates, because it contains $PE2$ (it also contains $PE3$, but it only needs to contain one other extension not to be minimal).

   So, here we have two extensions and no way of preferring one over the other.

3. All Quebecois are abnormal Canadians.

   S5 $\forall x[Quebequois(x) \supset Ab_1(x)]$

   Quebecois are typically abnormal Canadians.

   S6 $\forall x[(Quebequois(x) \wedge \neg Ab_3(x)) \supset Ab_1(x)]$

   Using S5, we have $Ab_1(robert)$, for sure, so we cannot get $\neg Francophone(robert)$. The only possible extension would be $PE3$.
   Using S6, we have two possible extensions, one containing $\{\neg Ab_1(robert), Ab_2(robert), Ab_3(robert), \neg Francophone(robert)\}$ and another containing $\{Ab_1(robert), \neg Ab_2(robert), \neg Ab_3(robert), Francophone(robert)\}$.
   We have no way of preferring one extension over the other.
   Once more, we cannot have all $\neg Ab_i(robert)$ in the same extension, because it would be inconsistent. All the other combinations would not be minimal or would be inconsistent.

4. In default logic, we have:

   D1 $\frac{Canadian(x) : \neg Francophone(x)}{\neg Francophone(x)}$

   S2 $\forall x[Quebequois(x) \supset Canadian(x)]$

   D3 $\frac{Quebequois(x) : Francophone(x)}{Francophone(x)}$

   S4 $Quebequois(robert)$

   In default logic extensions are constructed by adding the conclusions of **all** the default rules that are applicable. Extensions must be consistent and all conclusions must be supported. In this case, we have four possibilities of extensions:

   - $PE1 \models \{S2, S4\}$ in this case there are default rules that could have been applied and were not. So this is NOT an extension.
   - $PE2 \models \{S2, S4, Francophone(robert)\}$ this is consistent and all the applicable default rules were applied. So, this IS one possible extension.
   - $PE3 \models \{S2, S4, \neg Francophone(robert)\}$ this is consistent and all the applicable default rules were applied. So, this IS one possible extension.
   - $PE4 \models \{S2, S4, Francophone(robert), \neg Francophone(robert)\}$ this is NOT an extension because it is not consistent.

   So we have two extensions, one containing $\{Francophone(robert)\}$ and the other containing $\{\neg Francophone(robert)\}$, corresponding to the application of D3 or D1 respectively.
   A way to eliminate the undesired extension is to replace D1 with a non-normal default rule:

   D1nn $\frac{Canadian(x) : \neg Quebequois(x) \wedge \neg Francophone(x)}{\neg Francophone(x)}$

   This rule is not applicable to $robert$ because we have $Quebequois(robert)$, so the only extension left is the one containing $\{Francophone(robert)\}$, as desired.

5. In autoepistemic logic we have:

   A1 $\forall x[(Canadian(x) \wedge \neg \mathbf{B} Francophone(x)) \supset \neg Francophone(x)]$

   S2 $\forall x[Quebequois(x) \supset Canadian(x)]$

   A3 $\forall x[(Quebequois(x) \wedge \neg \mathbf{B} \neg Francophone(x)) \supset Francophone(x)]$

   S4 $Quebequois(robert)$

   Variable-free version:

   A1f $(Canadian(robert) \wedge \neg \mathbf{B} Francophone(robert)) \supset \neg Francophone(robert)$

   S2f $Quebequois(robert) \supset Canadian(robert)$

   A3f $(Quebequois(robert) \wedge \neg \mathbf{B} \neg Francophone(robert)) \supset Francophone(robert)$

S4f $Quebequois(robert)$

To find the stable expansions, replace each $\mathbf{B}\alpha$ by TRUE or by FALSE, and for each combination determine if the expansion satisfies positive and negative introspection (that is, if $\mathbf{B}\alpha$ was replaced by TRUE $KB^0 \models \alpha$, if $\mathbf{B}\alpha$ was replaced by FALSE $KB^0 \not\models \alpha$). In this case, there are two possibilities:

- $\mathbf{B}Francophone(robert)$ is true. In this case, the expansion will contain $\{Francophone(robert)\}$ and will satisfy positive and negative introspection.
- $\mathbf{B}Francophone(robert)$ is false. In this case, the expansion will contain $\{\neg Francophone(robert), \mathbf{B}\neg Francophone(robert)\}$ and will also satisfy positive and negative introspection.

This theory generates two stable expansions, one containing $\{Francophone(robert)\}$ and the other containing $\{\neg Francophone(robert)\}$.
It is possible to eliminate the undesirable expansion if we replace A1f by

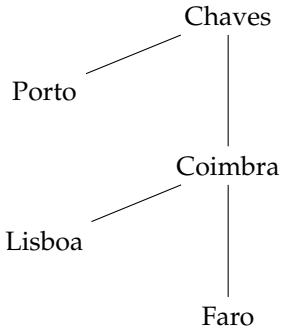
A1f2 $(Canadian(robert) \wedge \neg \mathbf{B}Quebequois(robert) \wedge \neg \mathbf{B}Francophone(robert)) \supset \neg Francophone(robert)$

Exercise 11.6 (from `http://www.cs.nott.ac.uk/~nza/G53KRR`)
Consider the following knowledge base:

$$\begin{aligned} KB = \{&NorthOf(coimbra, faro),\\ &NorthOf(chaves, porto),\\ &NorthOf(coimbra, lisboa),\\ &NorthOf(chaves, coimbra),\\ &\forall x \forall y \forall z[(NorthOf(x, y) \wedge NorthOf(y, z)) \supset NorthOf(x, z)]\} \end{aligned}$$

1. Does it hold that $KB \models_{CWA} NorthOf(chaves, faro)$? Explain why.
2. Does it hold that $KB \models_{CWA} \neg NorthOf(chaves, faro)$? Explain why.
3. Does it hold that $KB \models_{CWA} NorthOf(porto, faro)$? Explain why.
4. Does it hold that $KB \models_{CWA} \neg NorthOf(porto, faro)$? Explain why.

Answer:



1. $KB \models_{CWA} NorthOf(chaves, faro)$ because $NorthOf(chaves, coimbra)$, $NorthOf(coimbra, faro)$ and $\forall x \forall y \forall z[(NorthOf(x, y) \wedge NorthOf(y, z)) \supset NorthOf(x, z)]$ classically entail $NorthOf(chaves, faro)$.
2. $KB \not\models_{CWA} \neg NorthOf(chaves, faro)$ because there is an interpretation satisfying KB^+ and $NorthOf(chaves, faro)$ (in fact all interpretations that satisfy KB^+ also satisfy $NorthOf(chaves, faro)$, because this follows from KB in the classical sense).

3. $KB \nvDash_{CWA} NorthOf(porto, faro)$ because $NorthOf(porto, faro)$ does not follow classically from KB, so its negation is in KB^+ and since KB^+ is consistent, $NorthOf(porto, faro)$ is not entailed by it.
4. $KB \models_{CWA} \neg NorthOf(porto, faro)$ because $NorthOf(porto, faro)$ does not follow classically from KB, so its negation is in KB^+.

Exercise 11.7 (from `http://www.cs.nott.ac.uk/~nza/G53KRR`)
For the following KB:

$$\begin{aligned} KB = \{&SouthOf(milan, paris),\\ &SouthOf(milan, london),\\ &SouthOf(milan, moscow),\\ &paris \neq london,\\ &london \neq moscow,\\ &paris \neq moscow,\\ &\neg WarmerThan(milan, paris) \vee \neg WarmerThan(milan, london),\\ &\forall x[(SouthOf(milan, x) \wedge \neg Ab(x)) \supset WarmerThan(milan, x)]\} \end{aligned}$$

state whether the following sentences are minimally entailed, and explain why:

1. $WarmerThan(milan, moscow)$
2. $WarmerThan(milan, london)$

Answer:

In circumscription the goal is to minimize abnormality. Here, we have one Ab predicate and three constants, so there are eight possible combinations of AB predicate applied to each constant. The possible minimal models will be:

- $PMM1 \models \{\neg Ab(paris), \neg Ab(london), \neg Ab(moscow)\}$ this is NOT an acceptable minimal model, because it is inconsistent, because it also satisfies $\{WarmerThan(milan, paris), \neg WarmerThan(milan, paris), WarmerThan(milan, london), \neg WarmerThan(milan, london)\}$.
- $PMM2 \models \{Ab(paris), \neg Ab(london), \neg Ab(moscow)\}$ this IS an acceptable minimal model, because it is consistent, and it only satisfies one Ab predicate.
- $PMM3 \models \{\neg Ab(paris), Ab(london), \neg Ab(moscow)\}$ this IS an acceptable minimal model, because it is consistent, and it only satisfies one Ab predicate.
- $PMM4 \models \{\neg Ab(paris), \neg Ab(london), Ab(moscow)\}$ this is NOT an acceptable minimal model, because it is inconsistent, because it also satisfies $\{WarmerThan(milan, paris), \neg WarmerThan(milan, paris), WarmerThan(milan, london), \neg WarmerThan(milan, london)\}$.
- All the other four proposed minimal models will contain the positive Ab predicates in $PMM2$ or $PMM3$ and as a consequence not minimize abnormality.

So, this KB has two minimal models: $PMM2$ and $PMM3$.

1. $WarmerThan(milan, moscow)$ is minimally entailed if it is entailed by all the minimal models of KB. In both minimal models it is possible to obtain $WarmerThan(milan, moscow)$ in the classical sense, because $(I(milan), I(moscow))$ is in $I(SouthOf)$ and $I(moscow)$ is not in $I(Ab)$, so $(I(milan), I(moscow))$ is in $I(WarmerThan)$.
   This means that $WarmerThan(milan, moscow)$ is minimally entailed by KB.

2. $WarmerThan(milan, london)$ is minimally entailed if it is entailed by all the minimal models of KB. In $PMM2$ it is possible to obtain $WarmerThan(milan, london)$ in the classical sense, because $(I(milan), I(london))$ is in $I(SouthOf)$ and $I(london)$ is not in $I(Ab)$, so $(I(milan), I(london))$ is in $I(WarmerThan)$. However, in $PMM3$ it is no longer possible to obtain $WarmerThan(milan, london)$, because we have $Ab(london)$.
This means that $WarmerThan(milan, london)$ is NOT minimally entailed by KB.

Exercise 11.8 (from `http://www.cs.nott.ac.uk/~nza/G53KRR`)
Consider the following knowledge base:

S1 Cats usually don't attack people.

S2 Wild cats are cats.

S3 Wild cats when threatened attack people.

S4 a is a cat.

S5 b is a wild cat and is different from a.

S6 b is threatened.

1. Translate this knowledge base into first-order logic, using the circumscription approach to translate the default rule **S1**. Translate **S2** and **S3** as normal first order implications, which are true without exceptions. Use unary predicates C for cat, W for wild cat, A for attack people, T for being threatened.
2. Does this knowledge base minimally entail $\neg A(a)$ (a does not attack people)? Why?
3. Does this knowledge base minimally entail $\neg A(b)$ (b does not attack people)? Why?
4. Translate this knowledge base into default logic.
5. What can you conclude about a and b using default logic?
6. Translate this knowledge base into autoepistemic logic.
7. What can you conclude about a and b using autoepistemic logic? Why?

Answer:

1. **S1** $\forall x[(C(x) \wedge \neg Ab(x)) \supset \neg A(x)]$
   S2 $\forall x[W(x) \supset C(x)]$
   S3 $\forall x[(W(x) \wedge T(x)) \supset A(x)]$
   S4 $C(a)$
   S5 $W(b) \wedge b \neq a$
   S6 $T(b)$
2. Yes, $KB \models_{\leq} \neg A(a)$. In circumscription we want to minimize abnormality, and we have nothing that implies $Ab(a)$, so we can assume $\neg Ab(a)$. In conjunction with S4 and S1 we can prove $\neg A(a)$.
   More formally:
   We need to show that for any M , if $M \models KB$, then either $M \models \neg A(a)$, or we can find $M' < M$, that is, a model with a strictly smaller extension of Ab, such that $M' \models KB$.
   Note that in order to satisfy KB, any interpretation $M = (D, I)$ should have the following properties:

S1' all elements of D which are in $I(C)$ and not in $I(Ab)$ should not be in $I(A)$

S2' $I(W) \subseteq I(C)$

S3' $I(W) \cap I(T) \subseteq I(A)$

S4' $I(a) \in I(C)$

S5' $I(b) \in I(W), I(a) \neq I(b)$

S6' $I(b) \in I(T)$

From **S5'** and **S6'**, $I(b) \in I(W) \cap I(T)$, so by **S3'**, $I(b) \in I(A)$.
From **S5'** and **S2'**, $I(b) \in I(C)$. So from **S1'**, $I(b) \in I(Ab)$. In other words, in any interpretation which satisfies KB, the element denoted by b has to be abnormal.
On the other hand, the element denoted by a, $I(a)$, is different from $I(b)$ by **S5'** and can always be removed from $I(Ab)$ (provided that we also remove it from $I(A)$), and KB will still be satisfied in the resulting interpretation. So any interpretation where $I(a) \in I(Ab)$ is not minimal and can be ignored for the purposes of minimal entailment.
If $I(a) \notin I(Ab)$, then from **S4'** and **S1'**, $I(a)$ is not in $I(A)$, in other words all such interpretations satisfy $\neg A(a)$.

3. No, $KB \not\models_{\leq} \neg A(b)$. In fact, $KB \models A(b)$ in the usual classical sense of entailment (S5, S6, S3), so it also holds that $KB \models_{\leq} A(b)$ (if something is classically entailed, it is also minimally entailed).

4. In default logic (change the first rule, keep the rest):

   D1 $\frac{C(x) : \neg A(x)}{\neg A(x)}$

   S2 $\forall x[W(x) \supset C(x)]$

   S3 $\forall x[(W(x) \wedge T(x)) \supset A(x)]$

   S4 $C(a)$

   S5 $W(b) \wedge b \neq a$

   S6 $T(b)$

5. In default logic extensions are constructed by adding the conclusions of **all** the default rules that are applicable. Extensions must be consistent and all conclusions must be supported. In this case, we have two possibilities of extensions, because $D1$ is only applicable to a ($A(b)$ is a consequence in the classical sense, so $D1$ is not applicable to b):

   - $PE1 \models \{S2, S3, S4, S5, S6, A(b)\}$ in this case $D1$ could have been applied to a and was not. So this is NOT an extension.
   - $PE2 \models \{S2, S3, S4, S5, S6, A(b), \neg A(a)\}$ this is consistent and all the applicable default rules were applied. So, this IS one possible extension.

   So, we can conclude $A(b), \neg A(a)$ in the only possible extension.

6. In autoepistemic logic (change the first rule, keep the rest):

   A1 $\forall x[(C(x) \wedge \neg \mathbf{B}A(x)) \supset \neg A(x)]$

   S2 $\forall x[W(x) \supset C(x)]$

   S3 $\forall x[(W(x) \wedge T(x)) \supset A(x)]$

   S4 $C(a)$

   S5 $W(b) \wedge b \neq a$

   S6 $T(b)$

7. To find the stable expansions, replace each $\mathbf{B}\alpha$ by TRUE or by FALSE, and for each combination determine if the expansion satisfies positive and negative introspection (that is, if $\mathbf{B}\alpha$ was replaced by TRUE $KB^0 \models \alpha$, if $\mathbf{B}\alpha$ was replaced by FALSE $KB^0 \not\models \alpha$). In this case, we have four possible expansions, but only one is stable:

$\mathbf{B}A(a)$ $\mathbf{B}A(b)$	$\mathbf{B}A(a)$ $\neg\mathbf{B}A(b)$	$\neg\mathbf{B}A(a)$ $\mathbf{B}A(b)$	$\neg\mathbf{B}A(a)$ $\neg\mathbf{B}A(b)$
$\times$ because $KB \nvDash A(a)$	$\times$ because $KB \models A(b)$	✓	$\times$ because $KB \models A(b)$

So, we can conclude $\neg A(a)$ and $A(b)$ in the single stable expansion.

Exercise 11.9 (new)
Consider the following knowledge base:

S1 Birds usually fly

S2 Penguins are birds

S3 Parrots are birds

S4 Penguins do not fly

S5 Bob is a penguin

S6 Carl is a parrot

S7 Carl is different from Bob

1. Translate this knowledge base into first-order logic, using the circumscription approach to translate the default rule **S1**.
2. Does this knowledge base minimally entail that Bob flies? Justify your answer.
3. Does this knowledge base minimally entail that Carl flies? Justify your answer.
4. Translate this knowledge base into default logic.
5. What can you conclude about Bob and Carl using default logic? Justify your answer.
6. Translate this knowledge base into autoepistemic logic.
7. What can you conclude about Bob and Carl using autoepistemic logic? Justify your answer.

Answer:

Exercise 11.10 (new)
Consider the following knowledge base:

S1 Mollusks usually have a shell.

S2 Cephalopods are mollusks.

S3 Cephalopods usually do not have a shell.

S4 All nautilus are cephalopods.

S5 Nautilus usually have a shell.

S6 Nau is a nautilus.

1. Translate this knowledge base into first-order logic, using the circumscription approach to translate the default rules.
2. Does this knowledge base minimally entail that Nau has a shell? Justify your answer.
3. Translate this knowledge base into default logic.
4. What can you conclude about Nau using default logic? Justify your answer.
5. Translate this knowledge base into autoepistemic logic.
6. What can you conclude about Nau using autoepistemic logic? Justify your answer.

Answer:

Exercise 11.11 (new)
Consider the following knowledge base:

S1 Eggs usually have cholesterol.

S2 Normal eggs are eggs.

S3 Brudy eggs are eggs.

S4 Brudy eggs do not have cholesterol.

S5 O1 is a normal egg.

S6 B1 is a Brudy egg.

S7 O1 and B1 are different from each other.

1. Translate this knowledge base into first-order logic, using the circumscription approach to translate the default rules.
2. Does this knowledge base minimally entail that O1 and B1 have cholesterol? Justify your answer.
3. Translate this knowledge base into default logic.
4. What can you conclude about O1 and B1 using default logic? Justify your answer.
5. Translate this knowledge base into autoepistemic logic.
6. What can you conclude about O1 and B1 using autoepistemic logic? Justify your answer.

Answer:

12 Vagueness, Uncertainty, and Degrees of Belief

Exercise 12.1 (new)
Explain the notion of a *vague predicate*. Give an example of a vague predicate along with its *degree curve*.

Answer:

Exercise 12.2 (new)
Explain how conjunctions and disjunctions are handled in fuzzy logics.

Answer:

Exercise 12.3 (new)
Explain the meaning of the two values used in Dempster-Shafer theories to represent degrees of belief.

Answer:

Instead of using a single number to represent a degree of belief, Dempster-Shafer representations use two-part measures, called belief and plausibility. These are essentially lower and upper bounds on the probability of a proposition. For a coin known to be perfectly unbiased, we have .5 belief and .5 plausibility that the result is heads; but for the mystery coin, we have 0 belief that the result is heads (meaning we have no reason to give it any credence) and 1 plausibility (meaning we have no reason to disbelieve it either). The "value" of a propositional variable is represented by a range, which we might call the possibility distribution of the variable.

Exercise 12.4 (new)
Explain the need for fuzzy reasoning systems, that is, explain why we sometimes need to consider that something is not 100% true nor 100% false.

Answer:

Exercise 12.5 (Ch 12, Ex 2)
Consider the following example:

> Metastatic cancer (a) is a possible cause of a brain tumor (c) and is also an explanation for an increased total serum calcium (b). In turn, either of these could cause a patient to fall into occasional coma (d). Severe headache (e) could also be explained by a brain tumor (c).

1. Represent these causal links in a belief network. Let a stand for 'metastatic cancer', b for 'increased total serum calcium', c for 'brain tumor', d for 'occasional coma', and e for 'severe headaches'.
2. Give an example of an independence assumption that is implicit in this network.
3. Suppose the following probabilities are given:

   $$Pr(a) = 0.2$$

$Pr(b|a) = 0.8$

$Pr(b|\neg a) = 0.2$

$Pr(c|a) = 0.2$

$Pr(c|\neg a) = 0.05$

$Pr(e|c) = 0.8$

$Pr(e|\neg c) = 0.6$

$Pr(d|b \wedge c) = 0.8$

$Pr(d|b \wedge \neg c) = 0.8$

$Pr(d|\neg b \wedge c) = 0.8$

$Pr(d|\neg b \wedge \neg c) = 0.05$

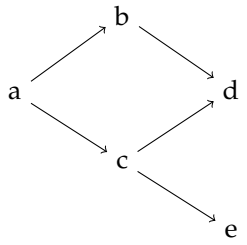
and assume that it is also given that some patient is suffering from severe headaches (e) but has not fallen into a coma ($\neg d$). Calculate joint probabilities for the eight remaining possibilities (that is, according to whether a, b, and c are true or false).

4. According to the numbers given, the a priori probability that the patient has metastatic cancer is 0.2. Given that the patient is suffering from severe headaches but has not fallen into a coma, are we now more or less inclined to believe that the patient has cancer? Explain.

Answer:

Remember that:
Negation: $Pr(\neg a) = 1 - Pr(a)$
Joint probability: $J(A1, \ldots, An) = Pr(A1|Parents(A1)) * \ldots * Pr(An|Parents(An))$
Conditional probability: $Pr(a|b) = \frac{Pr(a \wedge b)}{Pr(b)}$
Bayes's rule: $Pr(a|b) = \frac{Pr(a) * Pr(b|a)}{Pr(b)}$

1.



2. It is assumed that each propositional variable in the belief network is conditionally independent from the non-parent variables given the parent variables. In this case, there are several independence assumptions:

   - $Pr(c|a \wedge b) = Pr(c|a)$, $Pr(c|\neg a \wedge b) = Pr(c|\neg a)$, etc.
     This means that b and c are independent.
   - $Pr(d|a \wedge b \wedge c) = Pr(d|b \wedge c)$
   - $Pr(e|a \wedge b \wedge c \wedge d) = Pr(e|c)$

3. I spell out the computation of the probability of the first conjunction in more detail, after that I will skip the chain rule and use the negation rule without mentioning it.

(a) $Pr(a \wedge b \wedge c \wedge \neg d \wedge e)$ =
(using the normal chain rule)
$Pr(a) * Pr(b|a) * Pr(c|a \wedge b) * Pr(\neg d|a \wedge b \wedge c) * Pr(e|a \wedge b \wedge c \wedge \neg d)$ =
(substituting conditional probabilities using independence assumptions of the network)
$Pr(a) * Pr(b|a) * Pr(c|a) * Pr(\neg d|b \wedge c) * Pr(e|c)$ =
(using the negation rule $Pr(\neg d|b \wedge c) = 1 - Pr(d|b \wedge c)$)
$Pr(a) * Pr(b|a) * Pr(c|a) * (1 - Pr(d|b \wedge c)) * Pr(e|c)$ =
$0.2 * 0.8 * 0.2 * 0.2 * 0.8 = 0.00512$

(b) $Pr(a \wedge b \wedge \neg c \wedge \neg d \wedge e)$ =
$Pr(a) * Pr(b|a) * (1 - Pr(c|a)) * (1 - Pr(d|b \wedge \neg c)) * Pr(e|\neg c)$ =
$0.2 * 0.8 * 0.8 * 0.2 * 0.6 = 0.01536$

(c) $Pr(a \wedge \neg b \wedge c \wedge \neg d \wedge e)$ =
$Pr(a) * (1 - Pr(b|a)) * Pr(c|a) * (1 - Pr(d|\neg b \wedge c)) * Pr(e|c)$ =
$0.2 * 0.2 * 0.2 * 0.2 * 0.8 = 0.00128$

(d) $Pr(a \wedge \neg b \wedge \neg c \wedge \neg d \wedge e)$ =
$Pr(a) * (1 - Pr(b|a)) * (1 - Pr(c|a)) * (1 - Pr(d|\neg b \wedge \neg c)) * Pr(e|\neg c)$ =
$0.2 * 0.2 * 0.8 * 0.95 * 0.6 = 0.01824$

(e) $Pr(\neg a \wedge b \wedge c \wedge \neg d \wedge e)$ =
$(1 - Pr(a)) * Pr(b|\neg a) * Pr(c|\neg a) * (1 - Pr(d|b \wedge c)) * Pr(e|c)$ =
$0.8 * 0.2 * 0.05 * 0.2 * 0.8 = 0.00128$

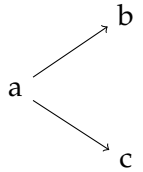
(f) $Pr(\neg a \wedge b \wedge \neg c \wedge \neg d \wedge e)$ =
$(1 - Pr(a)) * Pr(b|\neg a) * (1 - Pr(c|\neg a)) * (1 - Pr(d|b \wedge \neg c)) * Pr(e|\neg c)$ =
$0.8 * 0.2 * 0.95 * 0.2 * 0.6 = 0.01824$

(g) $Pr(\neg a \wedge \neg b \wedge c \wedge \neg d \wedge e)$ =
$(1 - Pr(a)) * (1 - Pr(b|\neg a)) * Pr(c|\neg a) * (1 - Pr(d|\neg b \wedge c)) * Pr(e|c)$ =
$0.8 * 0.8 * 0.05 * 0.2 * 0.8 = 0.00512$

(h) $Pr(\neg a \wedge \neg b \wedge \neg c \wedge \neg d \wedge e)$ =
$(1 - Pr(a)) * (1 - Pr(b|\neg a)) * (1 - Pr(c|\neg a)) * (1 - Pr(d|\neg b \wedge \neg c)) * Pr(e|\neg c)$ =
$0.8 * 0.8 * 0.95 * 0.95 * 0.6 = 0.34656$

4. We are asked whether $Pr(a|\neg d \wedge e)$ is greater or smaller than $Pr(a)$.
$Pr(a|\neg d \wedge e) = Pr(a \wedge \neg d \wedge e)/Pr(\neg d \wedge e)$ (conditional probability definition). We need to compute $Pr(a \wedge \neg d \wedge e)$ and $Pr(\neg d \wedge e)$, and to do that we use the probabilities we computed above. They describe all 8 possible states of the world given that $\neg d$ and e are true, and they are all disjoint. We are using $Pr(X) = Pr(X \wedge Y) + Pr(X \wedge \neg Y)$, or that the probability of the union of disjoint events equals to the sum of probabilities of those events.
So $Pr(a \wedge \neg d \wedge e) = Pr(a \wedge b \wedge c \wedge \neg d \wedge e) + Pr(a \wedge b \wedge \neg c \wedge \neg d \wedge e) + Pr(a \wedge \neg b \wedge c \wedge \neg d \wedge e) + Pr(a \wedge \neg b \wedge \neg c \wedge \neg d \wedge e)$, that is, the sum of the first four values above:
$Pr(a \wedge \neg d \wedge e) = 0.00512 + 0.01536 + 0.00128 + 0.01824 = 0.04$
and
$Pr(\neg d \wedge e)$ is the sum of all 8 numbers above, that is:
$Pr(\neg d \wedge e) = 0.04 + 0.00128 + 0.01824 + 0.00512 + 0.34656 = 0.4112$
so,
$Pr(a|\neg d \wedge e) = 0.04/0.4112$ which is approximately 0.1.
So the probability got smaller.

Exercise 12.6 (from `http://www.cs.nott.ac.uk/~nza/G53KRR`)
Given the following belief network:



And the following probabilities:

$Pr(a) = 1/5$

$Pr(b|a) = 2/3$

$Pr(b|\neg a) = 1/6$

$Pr(c|a) = 1/6$

$Pr(c|\neg a) = 2/3$

1. Give an example of an independence assumption that is implicit in this network.
2. What is the probability that a, b and c are all true?
3. What is the probability that a, b and c are all false?
4. What is the probability of $b \wedge c$?
5. Is $b \wedge c$ more probable given that a is true or given that a is false?

Answer:

1. $Pr(c|a) = Pr(c|a \wedge b)$
2. $Pr(a \wedge b \wedge c) = Pr(a) * Pr(b|a) * Pr(c|a) = 1/5 * 2/3 * 1/6 = 2/90 = 1/45$
3. $Pr(\neg a \wedge \neg b \wedge \neg c) = (1 - Pr(a)) * (1 - Pr(b|\neg a)) * (1 - Pr(c|\neg a)) = 4/5 * 5/6 * 1/3 = 20/90 = 2/9$
4. $Pr(b \wedge c) = Pr(a \wedge b \wedge c) + Pr(\neg a \wedge b \wedge c) = 1/45 + (1 - Pr(a)) * Pr(b|\neg a) * Pr(c|\neg a) = 1/45 + 4/5 * 1/6 * 2/3 = 1/45 + 8/90 = 1/45 + 4/45 = 5/45 = 1/9$
5. $Pr(b \wedge c|a) = Pr(a \wedge b \wedge c)/Pr(a) = 1/45 * 5 = 1/9$
   $Pr(b \wedge c|\neg a) = Pr(\neg a \wedge b \wedge c)/Pr(\neg a) = (4/45)/(4/5) = (4/45) * (5/4) = 20/180 = 1/9$ — No difference.

Exercise 12.7 (new)
Consider that a influences c and d. b also influences d. The following probabilities are given:

$Pr(a) = 1/3$

$Pr(b) = 1/6$

$Pr(c|a) = 2/3$

$Pr(c|\neg a) = 1/6$

$Pr(d|a \wedge b) = 1/6$

$Pr(d|a \wedge \neg b) = 2/6$

$Pr(d|\neg a \wedge b) = 1/6$

$Pr(d|\neg a \wedge \neg b) = 3/6$

1. Represent these causal links in a belief network.
2. Give an example of an independence assumption that is implicit in this network.
3. What is the probability that a, b and c are all true?
4. What is the probability that a, b and d are all false?
5. What is the probability of $c \wedge \neg d$?

Answer:

1.
2.
3. $Pr(a \wedge b \wedge c) = Pr(a \wedge b \wedge c \wedge d) + Pr(a \wedge b \wedge c \wedge \neg d) = Pr(a) * Pr(b) * Pr(c|a)$
4. $Pr(\neg a \wedge \neg b \wedge \neg d) = Pr(\neg a \wedge \neg b \wedge c \wedge \neg d) + Pr(\neg a \wedge \neg b \wedge \neg c \wedge \neg d) = Pr(\neg a) * Pr(\neg b) * Pr(\neg d|a)$
5.

Exercise 12.8 (new)
Consider the following information:
Michael usually has a headache (ha) if he has a cold (c) or if he worked late the previous night (lpn). If Michael has a headache, he will probably be grumpy (gr).

1. Represent these causal links in a belief network.
2. Give an example of an independence assumption that is implicit in this network.
3. The following probabilities are given:

   $Pr(c) = 0.1$
   $Pr(lpn) = 0.3$
   $Pr(ha|c \wedge lpn) = 0.9$
   $Pr(ha|c \wedge \neg lpn) = 0.7$
   $Pr(ha|\neg c \wedge lpn) = 0.6$
   $Pr(ha|\neg c \wedge \neg lpn) = 0.01$
   $Pr(gr|ha) = 0.6$
   $Pr(gr|\neg ha) = 0.1$

   What is the probability that ha, c e gr are all true?
4. What is the probability that lpn, ha e gr are all false?
5. What is the probability of $lpn \wedge \neg gr$?

Answer:

Exercise 12.9 (Adapted from Ch 12, Ex 4)
Consider the following information:
Sore elbows (soe) and sore hands (soh) can be the result of arthritis (a). Arthritis is also a possible cause of tennis elbow (tel), which in turn may cause sore elbows. Ultra-dry hands (dh) can also cause sore hands.

1. Represent these causal links in a belief network.
2. Give an example of an independence assumption that is implicit in this network.
3. The following probabilities are given:

   $Pr(a) = 0.001$
   $Pr(dh) = 0.01$
   $Pr(tel|a) = 0.0001$
   $Pr(tel|\neg a) = 0.01$
   $Pr(soh|a \wedge dh) = Pr(soe|a \wedge tel) = 0.1$
   $Pr(soh|a \wedge \neg dh) = Pr(soe|a \wedge \neg tel) = 0.99$
   $Pr(soh|\neg a \wedge dh) = Pr(soe|\neg a \wedge tel) = 0.99$
   $Pr(soh|\neg a \wedge \neg dh) = Pr(soe|\neg a \wedge \neg tel) = 0.00001$

   What is the probability that a, dh, tel and soh are all true?
4. What is the probability that a, dh, tel and soe are all false?
5. What is the probability of $a \wedge \neg dh$?

Answer:

13 Explanation and Diagnosis

Exercise 13.1 (new)
Explain why we can say that, in some sense, abductive reasoning is the converse of deductive reasoning. Include in your explanation one example application for each type of reasoning.

Answer:

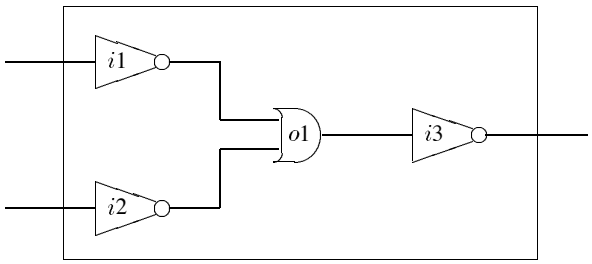
Exercise 13.2 (new)
Explain the difference between *abductive diagnosis* and *consistency-based diagnosis*.

Answer:

Exercise 13.3 (new)
Explain the use of abductive reasoning. Give an example of a situation where it would be useful.

Answer:

Exercise 13.4 (Ch 13, Ex 3)
Consider the binary circuit for logical AND depicted in this figure, where $i1$, $i2$ and $i3$ are logical inverters, and $o1$ is an OR gate.



1. Write sentences describing this circuit: its components, connectivity, and normal behaviour.

2. Write a sentence for a fault model saying that a faulty inverter has its output the same as its input.

3. Assuming the above fault model and that the output is 1 given inputs of 0 and 1, generate the three abductive explanations for this behaviour.

4. Generate the three consistency-based diagnoses for this circuit under the same conditions.

5. Compare the abductive and consistency-based diagnoses and explain informally why they are different.

Answer:

1. Logic description of the circuit:
   Components

   $$\forall x[Gate(x) \equiv (OrGate(x) \vee InvGate(x))]$$
   $$OrGate(o1)$$
   $$InvGate(i1)$$
   $$InvGate(i2)$$
   $$InvGate(i3)$$
   $$AndCircuit(c)$$

   Connectivity

   $$in(i1) = in1(c)$$
   $$in(i2) = in2(c)$$
   $$in1(o1) = out(i1)$$
   $$in2(o1) = out(i2)$$
   $$in(i3) = out(o1)$$
   $$out(i3) = out(c)$$

   Truth tables

   $$or(0,0) = 0, or(0,1) = 1, or(1,0) = 1, or(1,1) = 1$$
   $$not(0) = 1, not(1) = 0$$

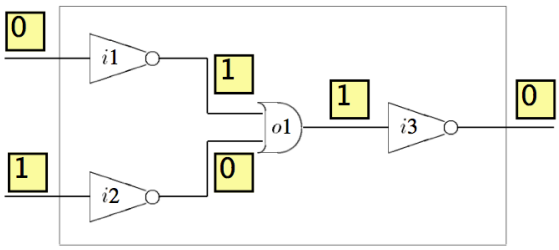
   Normal behaviour

   $$\forall x[(OrGate(x) \wedge \neg Ab(x)) \supset out(x) = or(in1(x), in2(x))]$$
   $$\forall x[(InvGate(x) \wedge \neg Ab(x)) \supset out(x) = not(in(x))]$$

2. Fault model:

   $$\forall x[(InvGate(x) \wedge Ab(x)) \supset out(x) = in(x)]$$

3. Expected behaviour for the circuit:



   Observation: output is 1 given inputs of 0 and 1.

   All possible explanations for this behaviour:

	$Ab(o1)$	$Ab(i1)$	$Ab(i2)$	$Ab(i3)$	entails?	consistent?
1-8	T	T,F	T,F	T,F	N	Y
9	F	T	T	T	Y	Y
10	F	T	T	F	N	N
11	F	T	F	T	N	N
12	F	T	F	F	Y	Y
13	F	F	T	T	Y	Y
14	F	F	T	F	N	N
15	F	F	F	T	Y	Y
16	F	F	F	F	N	N

- Lines 1-8: In all these cases, we have $Ab(o1)$. Because we do not have a fault model for when an OrGate is faulty, we cannot determine the actual output of the circuit. So, in these cases the observations are not entailed, but they are consistent.
- Line 9: $\neg Ab(o1) \supset Out(o1) = 1$
  $Ab(i1) \supset Out(i1) = 0$
  $Ab(i2) \supset Out(i2) = 1$
  $Ab(i3) \supset Out(i3) = Out(c) = 1$
- Line 10: $\neg Ab(o1) \supset Out(o1) = 1$
  $Ab(i1) \supset Out(i1) = 0$
  $Ab(i2) \supset Out(i2) = 1$
  $\neg Ab(i3) \supset Out(i3) = Out(c) = 0$
- Line 11: $\neg Ab(o1) \supset Out(o1) = 0$
  $Ab(i1) \supset Out(i1) = 0$
  $\neg Ab(i2) \supset Out(i2) = 0$
  $Ab(i3) \supset Out(i3) = Out(c) = 0$
- Line 12: $\neg Ab(o1) \supset Out(o1) = 0$
  $Ab(i1) \supset Out(i1) = 0$
  $\neg Ab(i2) \supset Out(i2) = 0$
  $\neg Ab(i3) \supset Out(i3) = Out(c) = 1$
- Line 13: $\neg Ab(o1) \supset Out(o1) = 1$
  $\neg Ab(i1) \supset Out(i1) = 1$
  $Ab(i2) \supset Out(i2) = 1$
  $Ab(i3) \supset Out(i3) = Out(c) = 1$
- Line 14: $\neg Ab(o1) \supset Out(o1) = 1$
  $\neg Ab(i1) \supset Out(i1) = 1$
  $Ab(i2) \supset Out(i2) = 1$
  $\neg Ab(i3) \supset Out(i3) = Out(c) = 0$
- Line 15: $\neg Ab(o1) \supset Out(o1) = 1$
  $\neg Ab(i1) \supset Out(i1) = 1$
  $\neg Ab(i2) \supset Out(i2) = 0$
  $Ab(i3) \supset Out(i3) = Out(c) = 1$
- Line 16: $\neg Ab(o1) \supset Out(o1) = 1$
  $\neg Ab(i1) \supset Out(i1) = 1$
  $\neg Ab(i2) \supset Out(i2) = 0$
  $\neg Ab(i3) \supset Out(i3) = Out(c) = 0$

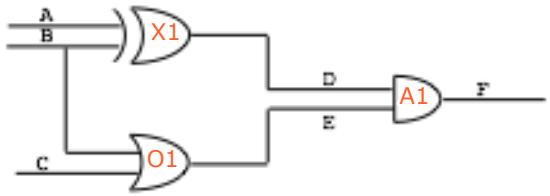
Abductive explanations: given a Knowledge Base and some input settings of the circuit, explain some output observations of the circuit, in the language of Ab. We want minimal conjunctions that **entail** the observations.
For this, only the lines where the observations are entailed matter. In this case, lines 9, 12, 13 and 15:
Line 9: $\neg Ab(o1) \wedge Ab(i1) \wedge Ab(i2) \wedge Ab(i3)$
Line 12: $\neg Ab(o1) \wedge Ab(i1) \wedge \neg Ab(i2) \wedge \neg Ab(i3)$
Line 13: $\neg Ab(o1) \wedge \neg Ab(i1) \wedge Ab(i2) \wedge Ab(i3)$
Line 15: $\neg Ab(o1) \wedge \neg Ab(i1) \wedge \neg Ab(i2) \wedge Ab(i3)$
The abductive explanations are:
Join 9 and 13: $\neg Ab(o1) \wedge Ab(i2) \wedge Ab(i3)$
Join 13 and 15: $\neg Ab(o1) \wedge \neg Ab(i1) \wedge Ab(i3)$
Line 12: $\neg Ab(o1) \wedge Ab(i1) \wedge \neg Ab(i2) \wedge \neg Ab(i3)$

4. Consistency-based diagnoses: we look for minimal sets of assumptions of abnormality that are consistent with the settings and observations. That is, consider only positive Ab, ignore $\neg Ab$ and eliminate non-minimal sets.
   For this, only the lines where the observations are consistent matter. In this case, lines 1-8, 9, 12, 13 and 15:

Lines 1-8: $Ab(o1) \rightsquigarrow \{o1\}$
Line 9: $\neg Ab(o1) \wedge Ab(i1) \wedge Ab(i2) \wedge Ab(i3) \rightsquigarrow \{i1, i2, i3\}$
Line 12: $\neg Ab(o1) \wedge Ab(i1) \wedge \neg Ab(i2) \wedge \neg Ab(i3) \rightsquigarrow \{i1\}$
Line 13: $\neg Ab(o1) \wedge \neg Ab(i1) \wedge Ab(i2) \wedge Ab(i3) \rightsquigarrow \{i2, i3\}$
Line 15: $\neg Ab(o1) \wedge \neg Ab(i1) \wedge \neg Ab(i2) \wedge Ab(i3) \rightsquigarrow \{i3\}$
The abductive explanations are the minimal sets:
$\{o1\}$, $\{i1\}$ and $\{i3\}$.

5. Compare the abductive and consistency-based diagnoses and explain informally why they are different.

Exercise 13.5 (new)
Consider the binary circuit depicted in this figure, using the usual notation for logical gates.



1. Write sentences describing this circuit: its components, connectivity, and normal behaviour.

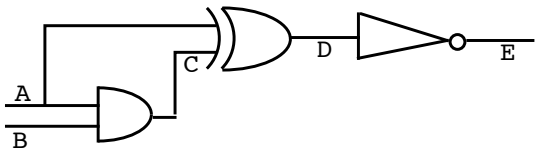
2. Write sentences for a fault model saying that a faulty OR has its output the same as its first input and that a faulty XOR has its output the same as its second input.

3. Assuming the above fault model and that the output is 0 given inputs of $A = 1$, $B = 0$ and $C = 1$, generate the abductive explanations for this behaviour.

4. Generate the consistency-based diagnoses for this circuit under the same conditions.

Answer:

Exercise 13.6 (new)
Consider the binary circuit depicted in this figure, using the usual notation for logical gates.



1. Write sentences describing this circuit: its components, connectivity, and normal behaviour.

2. Write sentences for a fault model saying that a faulty AND has its output the same as its first input and that a faulty XOR has its output the same as its second input.

3. Assuming the above fault model and that the output is 0 given inputs of $A = 1$ and $B = 1$, generate the abductive explanations and the consistency-based diagnoses for this behavior.

Answer:

14 Actions

Exercise 14.1 (new)
Explain why the *frame axioms* are needed in situation calculus. Give an illustrating example.

Answer:

Exercise 14.2 (new)
Explain what are the *fluents* in situation calculus. Give an illustrating example.

Answer:

Exercise 14.3 (new)
Explain two of the limitations of situational calculus.

Answer:

- single agent: there are no unknown or unobserved exogenous actions performed by other agents, and no unnamed events;
- no time: we have not talked about how long an action takes, or when it occurs;
- no concurrency: if a situation is the result of performing two actions, one of them is performed first and the other afterward;
- discrete actions: there are no continuous actions like pushing an object from one point to another or filling a bathtub with water;
- only hypotheticals: we cannot say that an action has occurred in reality, or will occur;
- only primitive actions: there are no actions that are constructed from other actions as parts, such as iterations or conditionals.

Exercise 14.4 (new)
Comment the following statement: "situational calculus uses as a representation language a fragment of first order logic, namely propositional logic".

Answer:

The statement is false. Situational calculus needs at least one constant to represent the initial situation $s0$. Constants do not exist in propositional logic.

Exercise 14.5 (Ch 14, Ex 1)
In the exercises below, and in the follow-up exercises of Chapter 15, we consider three application domains where we would like to be able to reason about action and change:

Pots of water: Consider a world with pots that may contain water. There is a single fluent $Contains$, where $Contains(p, w, s)$ is intended to say that a pot p contains w litres of water in situation s. There are only two possible actions, which can always be executed: $empty(p)$ which discards all the water contained in the pot p, and $transfer(p, p')$, which pours as much water as possible without spilling from pot p to p', with no change when $p = p'$. To simplify the formalization, we assume that the usual arithmetic constants, functions and predicates are also available. (You may assume that axioms for these have already been provided or built-in.)

1. Write the precondition axioms for the actions.
2. Write the effect axioms for the actions.
3. Show how successor state axioms for the fluents would be derived from these effect axioms.
4. Show how frame axioms are logically entailed by the successor state axioms.

Answer:

1. Pots of water
   - Fluent: $Contains(p, w, s)$
   - Actions: $empty(p)$ and $transfer(p, p')$

   Precondition axioms for the actions (in this case, it is always possible to execute both actions, as stated in the problem description):
   - $Poss(empty(p), s) \equiv True$
   - $Poss(transfer(p, p'), s) \equiv True$
2. Effect axioms for the actions:
   - $Contains(p, 0, do(empty(p), s))$
   - $(Contains(p, w, s) \wedge Contains(p', w', s)) \supset$
     $(Contains(p, max(w - (capacity(p') - w'), 0), do(transfer(p, p'), s)) \wedge$
     $Contains(p', min(capacity(p'), w + w'), do(transfer(p, p'), s)))$
   - $Contains(p, w, s) \supset Contains(p, w, do(transfer(p, p), s))$
3. Successor state axioms:
   - $Contains(p, wf, do(a, s)) \equiv$
     $(a = empty(p) \wedge wf = 0) \vee$
     $(a = transfer(p, p') \wedge Contains(p, w, s) \wedge Contains(p', w', s) \wedge$
     $wf = max(w - (capacity(p') - w'), 0)) \vee$
     $(a = transfer(p', p) \wedge Contains(p, w, s) \wedge Contains(p', w', s) \wedge$
     $wf = min(capacity(p), w + w')) \vee$
     $(Contains(p, wf, s) \wedge a \neq empty(p) \wedge \forall p'[a \neq transfer(p, p')] \wedge \forall p'[a \neq transfer(p', p)])$
4. Frame axioms: in this case, we have no frame axioms, because both actions affect the single fluent. So, the single fluent that can change changes with every action.

Exercise 14.6 (from `http://www.ime.usp.br/~liamf/cursoLegolog`)
Consider a world consisting of way stations connected by pathways that can run north, east, south and west; at any location, there may not be pathways leading in all of the directions. You wish to navigate a robot around this world. The state of the robot is governed by two fluents:

- $Location(x, s)$ — the robot is located at way station x in situation s
- $Direction(x, s)$ — the robot is facing direction x ($north$, $east$, $south$, $west$) in situation s

The robot is capable of performing the following actions:

- $forward$ which takes it to the next station in the direction it is facing
- $turnClockwise$ changes its direction 90 degrees by turning in a clockwise direction

You may also assume the following relations:

- $Connected(x, y, direction)$ — the robot can go from location x to location y by moving forward if it is facing $direction$
- $Clockwise(x, y)$ — when facing direction x a clockwise turn by 90 degrees will make it face direction y

There are also constant symbols for each of the way stations but for our purposes here it is sufficient to distinguish only two of them: $home$ and $depot$.

1. Write the precondition axioms for the actions.
2. Write the effect axioms for the actions.
3. Write the successor state axioms for $Location$ which can be derived from the previous axioms.

Answer:

1. Precondition axioms for the actions:
   - $Poss(turnClockwise, s) \equiv True$
   - $Poss(forward, s) \equiv Location(x, s) \wedge Direction(d, s) \wedge Connected(x, y, d)$
2. Effect axioms for the actions:
   - $(Direction(x, s) \wedge Clockwise(x, y)) \supset Direction(y, do(turnClockwise, s))$
   - $(Location(x, s) \wedge Direction(d, s) \wedge Connected(x, y, d)) \supset Location(y, do(forward, s))$
3. Successor state axioms for $Location$:
   - $Location(y, do(a, s)) \equiv$
     $(Location(x, s) \wedge Direction(d, s) \wedge Connected(x, y, d) \wedge a = forward) \vee$
     $(Location(y, s) \wedge a \neq forward)$
     This includes $turnClockwise$, which doesn't change the robot's location.

Exercise 14.7 (new)
Consider a situation where you want a robot to make a simple eggnog (egg yolk mixed with sugar). The robot can have or not have a bowl, have or not have egg yolks, have or not have sugar. The state of the robot is controlled in the world by the following fluents:

- $HasBowl(x, s)$ — the robot has bowl x in situation s.
- $HasYolk(x, s)$ — the robot has yolk x in situation s.
- $HasSugar(x, s)$ — the robot has sugar x in situation s.
- $HasYolkInBowl(x, y, s)$ — the robot has yolk x in bowl y in situation s.

- $HasSugarInBowl(x, y, s)$ — the robot has sugar x in bowl y in situation s.
- $HasEggnogInBowl(x, y, s)$ — the robot has eggnog x in bowl y in situation s.

The robot can perform the following actions:

- $grabBowl$ the robot grabs a bowl.
- $putYolk$ the robot puts the yolk in the bowl.
- $putSugar$ the robot puts the sugar in the bowl.
- $makeEggnog$ the robot mixes the yolk with the sugar in the bowl.

We can also assume that:

- The robot can always grab a bowl (and he has a bowl after that).
- If the robot has a yolk and has a bowl, he can put the yolk in the bowl.
- If the robot has sugar and has a bowl, he can put the sugar in the bowl.
- If the yolk and the sugar are in the bowl, the robot can mix them together.

1. Write the precondition axioms for the actions.
2. Write the effect axioms for the actions.
3. Write the successor state axioms for $HasSugarInBowl$ which can be derived from the previous axioms.

Answer:

15 Planning

Exercise 15.1 (new)
Explain the difference between *progressive planning* and *regressive planning*.

Answer:

Exercise 15.2 (new)
Explain why it is not practical to use resolution theorem proving over the situation calculus for planning.

Answer:

Exercise 15.3 (new)
Explain what is the *STRIPS assumption*, that is, what assumptions the STRIPS system is based on.

Answer:

In STRIPS, we assume that the world we are trying to deal with satisfies the following criteria: (1) only one action can occur at a time; (2) actions are effectively instantaneous; (3) nothing changes except as the result of planned actions. In this context, this has been called the STRIPS assumption, but it clearly applies just as well to our version of the situation calculus. What really distinguishes STRIPS from the situation calculus is that knowledge about the initial state of the world is required to be complete, and knowledge about the effects and noneffects of actions is required to be in a specific form.

Exercise 15.4 (new)
Explain why in *STRIPS* it is not necessary to have situations as an argument for the operators.

Answer:

Exercise 15.5 (Ch 15, Ex 1)
This exercise is a continuation of exercise 14.5. For each application, we consider a planning problem involving an initial setup and a goal.

Pots of water: Imagine that in the initial situation, we have two pots, a 5-litre one filled with water, and an empty 2-litre one. Our goal is to obtain 1 litre of water in the 2-litre pot.

1. Write a sentence of the situation calculus of the form $\exists x.\alpha$ which asserts the existence of the final goal situation.
2. Write a ground situation term e (that is, a term that is either $S0$ or of the form $do(a, e')$ where a is a ground action term and e' is itself a ground situation term) such that e denotes the desired goal situation.
3. Explain how you could use Resolution to automatically solve the problem for any initial state: how would you generate the clauses, and assuming the process stops,

how would you extract the necessary moves? Explain why you need to use the successor state axioms, and not just effect axioms.

4. Suppose we were interested in formalizing the problem using a STRIPS representation. Decide what the operators should be, and then write the precondition, add list, and delete list for each operator. You may change the language as necessary.

5. Consider the database corresponding to the initial state of the problem. For each STRIPS operator, and each binding of its variables such that the precondition is satisfied, state what the database progressed through this operator would be.

6. Consider the final goal state of the problem. For each STRIPS operator, describe the bindings of its variables for which the operator can be the final action of a plan, and in those cases, what the goal regressed through the operator would be.

Answer:

Initial state: $Contains(p5, 5, s0)$ and $Contains(p2, 0, s0)$.

1. Existence of the goal situation:
   $\exists s[Contains(p2, 1, s) \wedge Legal(s)]$
2. Description of the goal situation:
   $do(transfer(p5, p2), do(empty(p2), do(transfer(p5, p2), do(empty(p2), do(transfer(p5, p2), s0)))))$
3. We could use resolution to automatically solve the problem for any initial state by starting with the negated goal, the successor state axiom and the description of the initial state and applying resolution until we get the empty clause. The plan would be obtained by unification of the variables used during the proof.
4. Initial world model in STRIPS:

   - $Contains(p5, 5)$
   - $Contains(p2, 0)$
   - $capacity(p5) = 5$
   - $capacity(p2) = 2$

   Operator $Empty(p)$

   - Preconditions: $Contains(p, w)$
   - Delete list: $Contains(p, w)$
   - Add list: $Contains(p, 0)$

   Operator $Tansfer(p, p')$

   - Preconditions: $Contains(p, w), Contains(p', w'), p \neq p'$
   - Delete list: $Contains(p, w), Contains(p', w')$
   - Add list: $Contains(p, max(w-(capacity(p')-w'), 0)), Contains(p', min(capacity(p'), w+w'))$

5. For each STRIPS operator, and each binding of its variables such that the precondition is satisfied, state what the initial database progressed through this operator would be.
   $Empty(p2)$

   - $Contains(p5, 5)$
   - $Contains(p2, 0)$
   - $capacity(p5) = 5$

- $capacity(p2) = 2$

$Empty(p5)$

- $Contains(p5, 0)$
- $Contains(p2, 0)$
- $capacity(p5) = 5$
- $capacity(p2) = 2$

$Tansfer(p2, p5)$

- $Contains(p5, 5)$
- $Contains(p2, 0)$
- $capacity(p5) = 5$
- $capacity(p2) = 2$

$Tansfer(p5, p2)$

- $Contains(p5, 3)$
- $Contains(p2, 2)$
- $capacity(p5) = 5$
- $capacity(p2) = 2$

$Tansfer(p2, p2)$ and $Tansfer(p5, p5)$ are not considered because the preconditions are not satisfied ($p \neq p'$).

6. For each STRIPS operator, describe the bindings of its variables for which the operator can be the final action of a plan, and in those cases, what the goal regressed through the operator would be.
Each operator is applicable if its Delete list $\cap Goal = \{\}$.
New Goal = Goal + Preconditions - Add list
Goal: $Contains(p2, 1)$
Apply $transfer(p5, p2)$
= $Contains(p2, 1) +$
$+ Contains(p5, w) + Contains(p2, w') + p5 \neq p2 -$
$- Contains(p5, max(w - (capacity(p5) - w'), 0)) - Contains(p2, min(capacity(p2), w + w'))$

Exercise 15.6 (from `http://www.ime.usp.br/~liamf/cursoLegolog`)
This exercise is a continuation of exercise 14.6. In the initial situation the robot is located at $home$ facing $north$. You are to consider navigating the robot so that it ends up being located at $depot$.

1. Write a sentence of the situation calculus whose only situation term is $S0$, describing the initial situation.
2. Write a sentence of the situation calculus of the form $\exists x.\alpha$ which asserts the existence of the final goal situation.
3. Suppose that we were interested in formalizing the problem using a STRIPS representation. Decide what the operators should be, and then write the precondition, add list, and delete list for each operator. You may change the language as necessary.
4. Consider the database corresponding to the initial state of the problem. For each STRIPS operator, and each binding of its variables such that the precondition is satisfied, state what the database progressed through this operator would be.

Answer:

1. $Location(home, s0) \wedge Direction(north, s0)$
2. $\exists x[Location(depot, x) \wedge Legal(x)]$
3. Operator $turnClockwise$
   - Preconditions: $Direction(d1), Clockwise(d1, d2)$
   - Delete list: $Direction(d1)$
   - Add list: $Direction(d2)$

   Operator $forward$
   - Preconditions: $Location(x1), Direction(d), Connected(x1, x2, d)$
   - Delete list: $Location(x1)$
   - Add list: $Location(x2)$
4. Only the conditions for $turnClockwise$ are satisfied. Progressed database is: $Location(home)$, $Direction(east)$

Exercise 15.7 (new)
This exercise is a continuation of exercise 14.7. In the initial situation the robot $HasYolk$ and $HasSugar$. We want the robot to make eggnog.

1. Write a sentence of the situation calculus whose only situation term is $S0$, that describes the initial situation.
2. Write a sentence of the situation calculus of the form $\exists x.\alpha$ which asserts the existence of the final goal situation.
3. Suppose we were interested in formalizing the problem using a STRIPS representation. Decide what the operators should be, and then write the precondition, add list, and delete list for each operator.
4. Consider the database corresponding to the initial state of the problem. For **one** STRIPS operator of your choice, and **one** binding of its variables such that the precondition is satisfied, state what the database progressed through this operator would be.

Answer:

16 The Tradeoff between Expressiveness and Tractability

Exercise 16.1 (new)
Explain the meaning of the phrase: "A fundamental fact of life is that there is a tradeoff between the expressiveness of the representation language and the tractability of the associated reasoning task".

Answer:

Exercise 16.2 (new)
Explain why reasoning by cases is hard and can in some situations cause intractability.

Answer:

Exercise 16.3 (new)
Explain the need for the development of limited representation languages, instead of more generic languages. Illustrate with an example.

Answer:

Exercise 16.4 (new)
Explain the reason why the limited languages that we studied (eg. Horn clauses, description logics) do not allow for the representation of disjunctions.

Answer:

Because disjunctions are used to represent incomplete knowledge and incomplete knowledge may cause intractability, because in this case it is necessary to consider all the possible combinations of all possible cases.