Thomas Huzij
tph2109@columbia.edu

Homework 2

7.10) Decide whether each of the following sentences is valid, unsatisfiable, or neither. Verify your decisions using truth tables or the equivalence rules of Figure 7.11 (page 249).

a. Smoke ⇒ Smoke

¬Smoke ∨ Smoke [implication elimination]
Either ¬Smoke or Smoke will have to be true, so by the very definition of ∨, the sentence will have to evaluate to true. Therefore it is valid.

b. Smoke ⇒ Fire

¬Smoke ∨ Fire [implication elimination]
Either ¬Smoke or Fire can be true, but not necessarily. Therefore the sentence can evaluate to true, which is to say it is satisfiable. Therefore it is neither.

c. (Smoke ⇒ Fire) ⇒ (¬Smoke ⇒ ¬Fire)

¬(Smoke ⇒ Fire) ∨ (¬Smoke ⇒ ¬Fire) [implication elimination]
¬(¬Smoke ∨ Fire) ∨ (¬Smoke ⇒ ¬Fire) [implication elimination]
¬(¬Smoke ∨ Fire) ∨ (Smoke ∨ ¬Fire) [implication elimination]
(¬(¬Smoke) ∧ ¬Fire) ∨ (Smoke ∨ ¬Fire) [De Morgan]
(Smoke ∧ ¬Fire) ∨ (Smoke ∨ ¬Fire) [double-negation elimination]
For this sentence to evaluate to true, either the sub-sentence to the left of the first ∨ has to be true or the sub-sentence to the right of the first ∨ has to be true. For the first sub-sentence to be true, by the definition of ∧, both Smoke and ¬Fire have to be true. This is possible and therefore satisfiable. For the second sub-sentence to be true, by the definition of ∨,either Smoke or ¬Fire have to be true. This is possible and therefore satisfiable. Since either side of the first ∨ can possibly evaluate to true, it is satisfiable. Therefore it is neither.

d. Smoke ∨ Fire ∨ ¬Fire

Smoke ∨ (Fire ∨ ¬Fire) [associativity of ∨]
For this sentence to evaluate to true, either Smoke or (Fire ∨ ¬Fire) have to be true. It is possible for Smoke to be true but not necessarily. However, either Fire or ¬Fire has to evaluate to true, therefore (Fire ∨ ¬Fire) is always true. Therefore it is valid.

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

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

$\neg \exists x,y,n\ Person(x) \land Person(y) \Rightarrow [HasSS\#(x, n) \land HasSS\#(y, n)]$
This is problematic because ⇒ when used in conjunction with ∃ is ineffective. This implication sentence is true if either the premise and the conclusion are both true, or the premise is false. Additionally, the English sentence is obviously talking about two different people. The implication sentence must evaluate whether x and y are the same person. Therefore the correct translation would be:
$\neg \exists x,y,n\ \neg(x = y) \land Person(x) \land Person(y) \land [HasSS\#(x, n) \land HasSS\#(y, n)]$

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

$\exists n\ HasSS\#(John, n) \land HasSS\#(Mary, n)$
The implication sentence simply evaluates whether or not John has the same social security number as Mary. Therefore it is a good translation of the English sentence.

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

$\forall x,n\ Person(x) \Rightarrow [HasSS\#(x, n) \land Digits(n, 9)]$
This implication sentence is incorrect because it implies that every single person has all the 9-digit social security numbers. The sentence is trying to convey that everyone has a single 9-digit social security number. The correct translation would therefore be:
$\forall x,n\ Person(x) \land HasSS\#(x, n) \Rightarrow Digits(n, 9)$

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

No two people have the same social security number.
$\neg \exists x,y\ Person(x) \land Person(y) \Rightarrow SS\#(x) = SS\#(y)$
John's social security number is the same as Mary's.
$SS\#(John) = SS\#(Mary)$
Everyone's social security number has nine digits.
$\forall x\ Person(x) \Rightarrow Digits(SS\#(x), 9)$

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

a. Some students took French in spring 2001.

$\exists x$ Student(x) $\wedge$ Take(x, Class(French, Spring 2001))
where Take(a, b) means that a took b
and Class(c, d) means that c is a class during semester d
and Student(e) means e is a student

b. Every student who takes French passes it.

$\forall x,y$ Student(x) $\wedge$ Take(x, Class(French, y)) $\Rightarrow$ Pass(x, Class(French, y))
where Take(a, b) means that a takes b
and Class(c, d) means that c is a class during semester d
and Pass(e, f) means that e passed f
and Student(g) means that g is a student

c. Only one student took Greek in spring 2001.

$\exists x$ Student(x) ^ Take(x, Class(Greek, Spring 2001)) ^ ($\forall y$ $\neg$(x = y) $\Rightarrow$ $\neg$Take(y, Class(Greek, Spring 2001)))
where Take(a, b) means that a takes b
and Class(c, d) means that c was during semester d
and Student(e) means that e is a student

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

$\forall x,y$ $\exists z$ Score(z, Class(Greek, x)) > Score(y, Class(French, x))
where Score(a, b) means a is a score in b
and Class(c, d) means that c was during semester d

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

a. P(A, B, B), P(x, y, z)

Each part of the predicate filled with variables can unify with the corresponding constants in the other predicate, x corresponds to A, y corresponds to B, and z also corresponds to B.
{x/A, y/B, z/B}

b. Q(y,G(A, B)), Q(G(x, x), y)

First, unify y with G(x,x):
{y/G(x, x)}
Thus the new sentences are:
Q(G(x, x), G(A, B)), Q(G(x, x), G(x, x))

However, now G(x, x) must be able to unify with the corresponding G(A, B). This would require x to unify with both A and B. A variable cannot do this. Therefore no general unifier exists.

c. Older(Father(y), y), Older(Father(x), John)

To unify Father(y) and Father(x), x and y must unify. It is clear that John also unifies with the corresponding y. Therefore John unifies with both x and y. So the general unifier is:
{x/John, y/John}

d. Knows(Father(y), y), Knows(x, x)

First x and Father(y) must unify:
{x/Father(y)}
However, then x and y must unify. However, y would therefore have to unify with Father(y) but this is not possible because y occurs in Father(y). Therefore no general unifier exists.

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

a. Translate the premise and the conclusion into the language of first-order logic. Use three predicates: HeadOf(h, x) (meaning “h is the head of x”), Horse(x), and Animal(x).

Given the premise $\forall x\ Horse(x) \Rightarrow Animal(x)$,
we have conclusion $\forall h,x\ HeadOf(h, x) \land Horse(x) \Rightarrow \exists y\ HeadOf(h, y) \land Animal(y)$

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

Negating the conclusion produces:
$\neg\forall h,x\ HeadOf(h, x) \land Horse(x) \Rightarrow \exists y\ HeadOf(h, y) \land Animal(y)$

Premise converted to CNF:
$\forall x\ \neg Horse(x) \lor Animal(x)$ [implication elimination]
$\neg Horse(x) \lor Animal(x)$ [drop universal quantifiers]

Negated conclusion converted to CNF:
$\neg\forall h,x\ HeadOf(h, x) \land Horse(x) \Rightarrow \exists y\ HeadOf(h, y) \land Animal(y)$
$\neg\forall h,x\ \neg(HeadOf(h, x) \land Horse(x)) \lor (\exists y\ HeadOf(h, y) \land Animal(y))$ [implication elimination]
$\exists h,x\ \neg(\neg(HeadOf(h, x) \land Horse(x)) \lor (\exists y\ HeadOf(h, y) \land Animal(y)))$ [move ¬ inwards]
$\exists h,x\ \neg(\neg(HeadOf(h, x) \land Horse(x))) \land (\neg\exists y\ HeadOf(h, y) \land Animal(y))$ [De Morgan]
$\exists h,x\ (HeadOf(h, x) \land Horse(x)) \land (\neg\exists y\ HeadOf(h, y) \land Animal(y))$ [double-negation elimination]
$\exists h,x\ (HeadOf(h, x) \land Horse(x)) \land (\forall y\ \neg(HeadOf(h, y) \land Animal(y))$ [move ¬ inwards]

∃h,x (HeadOf(h, x) ∧ Horse(x)) ∧ (∀y ¬HeadOf(h, y) ∨ ¬Animal(y)) [De Morgan]
∀y∃h,x (HeadOf(h, x) ∧ Horse(x)) ∧ (¬HeadOf(h, y) ∨ ¬Animal(y)) [move universal outwards]
∀y (HeadOf(A, B) ∧ Horse(B)) ∧ (¬HeadOf(A, y) ∨ ¬Animal(y)) [skolemize]
HeadOf(A, B) ∧ Horse(B) ∧ (¬HeadOf(A, y) ∨ ¬Animal(y)) [drop universal quantifiers]

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

From the premise we have:
¬Horse(x) ∨ Animal(x)

From the conclusion we have:
Horse(B)
HeadOf(A, B)
¬HeadOf(A, y) ∨ ¬Animal(y)

First, resolve Horse(B) and ¬Horse(x) ∨ Animal(x) with unifier {x/B}:
¬Horse(B) ∨ Animal(B)
But we know ¬Horse(x) is false, so we have:
Animal(B)
Then, resolve Animal(B) and ¬HeadOf(A, y) ∨ ¬Animal(y) with unifier {y/B}:
¬HeadOf(A, B) ∨ ¬Animal(B)
But we know ¬Animal(B) is false, so we have:
¬HeadOf(A, B)
However, we already know HeadOf(A, B) so we have found a contradiction. Therefore the original inference is invalid.