Logic(s) for computer science - Week 11
Normal forms for First Order Logic
Tutorial

1. Prove that the following equivalences hold:

   (a) $P(e,x) \stackrel{S}{\equiv} P(e,f(x,x))$ where S is the Σ-structure $S = (\mathbb{N}, \{<\}, \{+, s, 0\})$.

   (b) $\neg\forall x.\varphi \equiv \exists x.\neg\varphi$;

   (c) $\neg\exists x.\varphi \equiv \forall x.\neg\varphi$

   (d) $(\forall x.\varphi_1) \wedge \varphi_2 \equiv \forall x.(\varphi_1 \wedge \varphi_2)$, if $x \notin \mathit{free}(\varphi_2)$;

   (e) $(\forall x.\varphi_1) \vee \varphi_2 \equiv \forall x.(\varphi_1 \vee \varphi_2)$, if $x \notin \mathit{free}(\varphi_2)$;

   (f) $(\exists x.\varphi_1) \wedge \varphi_2 \equiv \exists x.(\varphi_1 \wedge \varphi_2)$, if $x \notin \mathit{free}(\varphi_2)$;

   (g) $(\exists x.\varphi_1) \vee \varphi_2 \equiv \exists x.(\varphi_1 \vee \varphi_2)$, if $x \notin \mathit{free}(\varphi_2)$;

   (h) $\forall x.(P(x) \wedge Q(x)) \equiv (\forall x.P(x)) \wedge (\forall x.Q(x))$

   (i) $\forall x.(P(x) \vee Q(x)) \not\equiv (\forall x.P(x)) \vee (\forall x.Q(x))$

   (j) $\exists x.(P(x) \wedge Q(x)) \not\equiv (\exists x.P(x)) \wedge (\exists x.Q(x))$

   (k) $\exists x.(P(x) \vee Q(x)) \equiv (\exists x.P(x)) \vee (\exists x.Q(x))$

2. Let consider the substitution $\sigma : \mathcal{X} \to \mathcal{T}$ such that $\sigma(x) = i(y)$, $\sigma(y) = f(x,z)$ and $\sigma(z) = x$ for $z \in \mathcal{X} \setminus \{x, y\}$. Apply the substitution σ to the following formulae:

   (a) $\varphi = (\forall x.P(x,y)) \to P(i(y), x)$

   (b) $\varphi = P(x,y) \wedge \exists y.Q(y) \to \forall x.P(x,y)$

3. Compute one prenex normal form (PNF) for each of the following formulae:

   (a) $\varphi = \Big(\forall x.\neg\big(P(x,x) \wedge \neg\exists y.P(x,y)\big)\Big) \wedge P(x,x)$.

Solution

$$
\begin{aligned}
\varphi &= \Big(\forall x.\neg\big(P(x,x)\wedge\neg\exists y.P(x,y)\big)\Big)\wedge P(x,x)\\
&\overset{R.L.}{\equiv} \Big(\forall z.\neg\big(P(z,z)\wedge\neg\exists y.P(z,y)\big)\Big)\wedge P(x,x)\\
&\overset{1}{\equiv} \forall z.\Big(\neg\big(P(z,z)\wedge\neg\exists y.P(z,y)\big)\wedge P(x,x)\Big)\\
&\overset{6}{\equiv} \forall z.\Big(\neg\big(P(z,z)\wedge\forall y.\neg P(z,y)\big)\wedge P(x,x)\Big)\\
&\overset{1}{\equiv} \forall z.\Big(\neg\big(\forall y.(P(z,z)\wedge\neg P(z,y))\big)\wedge P(x,x)\Big)\\
&\overset{5}{\equiv} \forall z.\Big(\big(\exists y.\neg(P(z,z)\wedge\neg P(z,y))\big)\wedge P(x,x)\Big)\\
&\overset{3}{\equiv} \forall z.\exists y.(\neg(P(z,z)\wedge\neg P(z,y))\wedge P(x,x)).
\end{aligned}
$$

(b) $\forall x.(P(x,y)\wedge\exists x P(x,x))$

(c) $(\exists z.P(x,y))\vee P(z,z)$;

(d) $(\exists z.P(x,z))\wedge(\forall x.P(x,z))$;

(e) $(\exists z.P(x,z))\rightarrow P(x,x)$.

(f) $\neg(\exists x.P(x,y)\vee\forall z.P(z,z))\wedge\exists y.P(x,y)$

(g) $\forall x.\exists y.P(x,y)\rightarrow\neg\exists x.\neg\exists y.P(x,y)$

4. We will use the proover Z3 available at `https://rise4fun.com/z3` for the signature $\Sigma = (\{<,\leq,>,\geq,=\},\{+,*,0,1,2,3,4,\ldots\})$ and Σ-structure $S = (\mathbb{Z},\{<,\leq,>,\geq,=,+,*,0,1,2,3,4,\ldots\})$. We will verify if some formulae are satisfiable in S, and in the case they are satisfiable we will find an assignment that make them true.

We recall that the formula φ is satisfiable in a structure S if there is an S-assignment α with the property that $S,\alpha\models\varphi$.

For instance, the formula

$$>(x,+(y,2))\wedge=(x,+(*(2,z),10))\wedge\leq(+(z,y),100)$$

or, using the infixed notation

$$x>y+2\wedge x=2*z+10\wedge z+y\leq 100$$

is satisfiable in the structure S from above, and an assignment that makes the formula true is $\alpha:\mathcal{X}\rightarrow\mathbb{Z}$, defined by $\alpha(x)=10,\alpha(y)=0,\alpha(z)=0$. In order to test the satisfiability of the above formula, we use the code:

```
(declare-const x Int) ;; we declare the free variables
(declare-const y Int)
(declare-const z Int)
```

```
(assert (and (> x (+ y 2)) ;; we include the formula that we wish
             (= x (+ (* 2 z) 10)) ;; to test if it is true
             (<= (+ z y) 1000))) ;; using the syntax of Z3

(check-sat) ;; check if the formula is satisfiable
```

Because toe formula is satisfiable in S, we can use the following code in order to obtain an assignment in which the formula is true:

```
(get-model) ;; prints an assignment that makes the formula true
```

Model the following sentences as first order logic formula over the signature Σ defined above and use Z3 te determine if they are or not satisfiable.

(a) x is grater then 100, y is less then 42, and the product $x \times y$ is less then 10. Free variables: x, y.

(b) x is an even number grater then 11. Free variables: x. Hint: we can express "x is enen" by $\exists x'.(=(x, *(2, x')))$. In Z3, we write `(exists ((xp Int)) (= x (* 2 xp)))` corresponding to the formula $\exists x'.(=(x, *(2, x')))$.

(c) x is odd, y is even and $x + y$ is greater then 42. Free variables: x, y.

(d) x is odd, y is even and $x + y$ is even. Free variables: x, y.

(e) $x * y$ is odd, $x + y$ is even, $x > 10$ and $y < 0$. Free variables: x, y.

(f) Fhe sum of any two even numbers is an even number. Free variables: none

(g) Use Z3 to explain the following game: think about a number, add 4 to it, multiply the result by 2, substract 6 from the result, divide by 2 and in the end substract the number you thought about; the result is always 1, for any number you start with.

Fast guide for Z3:

Math	Z3
$x \times y$	(*xy)
$x + y$	(+xy)
$\varphi_1 \land \varphi_2$	(and φ_1 φ_2)
$\varphi_1 \to \varphi_2$	(=> φ_1 φ_2)
$\exists x.\varphi$	(exists ((x Int)) φ)