FOL (cont. from 'Predicate Logic' notes from IDB lects.)

SUBSTITUTION RULES

→ If P is a formula, s is a term and x is a variable, then $P[s/x]$ is the formula obtained by substituting s for all free occurrences of x

e.g. $(\exists x. P(x, y))[3/y] \equiv \exists x. P(x, 3)$ — free!

$(\exists x. P(x, y))[3/x] \equiv \exists x. P(x, y)$ — NOT free!

→ If necessary, bound var. in P must be renamed to avoid capture of free var. in s

$$(\exists x. P(x,y))[f(x)/y] = \exists z. P(z, f(x))$$

ASSIGNMENT & SATISFACTION

eg. Consider wff $\emptyset$: $R(f(x), g(y,a))$ where $x, y \in \mathcal{V}$ (variables) and a is a const.

Given interpretation I where

- domain D is set of integers $\mathbb{Z}$
- $a^I \equiv -5$
- $R^I \equiv <$ [Less-than]
- $f^I \equiv -$ [subtraction]
- $g^I \equiv +$ [addition]

and environment $s[x \mapsto 3, y \mapsto 2]$ then under this interpretation & assignment:

$$\emptyset^I \equiv -3 < (2+(-5)) \equiv -3 < -3 \text{ is not satisfied!}$$

SATISFACTION & VALIDITY

eg. Consider the ff. statement: $\forall x, y. R(x, y) \rightarrow \exists z. R(x, z) \wedge R(z, y)$

(a). Is it satisfiable?

→ YES! Domain is the real no. and R is interpreted as the $<$ relation

(b). Is it valid?

→ NO! Domain? Interpretation of R?

UNIVERSAL QUANTIFICATION

$$\frac{P[x_0/x]}{\forall x.P}\ (\text{allI})$$

Provided tht. x_0 is not free in the assumptions

$$\frac{\forall x.P}{P[t/x]}\ (\text{spec})$$

$$\frac{\forall x.P \qquad \begin{matrix}[P[t/x]]\\ \vdots \\ Q\end{matrix}}{Q}\ (\text{allE})$$

EXISTENTIAL QUANTIFICATION

$$\frac{P[t/x]}{\exists x.P}\ (\text{exI})$$

$$\frac{\exists x.P \qquad \begin{matrix}[P[x_0/x]]\\ \vdots \\ Q\end{matrix}}{Q}\ (\text{exE})$$

Provided x_0 does not occur in Q or any assumption other than $P[x_0/x]$

e.g. Prove tht. $\exists y.P(y)$ is true, given tht. $\forall x.P(x)$ holds.

$$\cfrac{\cfrac{\forall x.P(x)}{P(a)}\ (\text{spec})}{\exists y.P(y)}\ (\text{exI})$$

We implicitly use the fact tht. our domain is non-empty. It doesn't matter what a is.

e.g. Prove tht. $\forall x.Q(x)$ is true, given $\forall x.P(x)$ and $(\forall x.P(x) \to Q(x))$

$$\cfrac{\cfrac{\forall x.P(x)\to Q(x) \qquad \cfrac{\forall x.P(x) \qquad \cfrac{[P(y)\to Q(y)]_1 \qquad [P(y)]_2}{Q(y)}\ (\text{mp})}{Q(y)}\ (\text{allE}_2)}{Q(y)}\ (\text{allE}_1)}{\forall x.Q(x)}\ (\text{allI})$$

FOL SEQUENT PROOF

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{}{P(y)\vdash P(y)} \qquad \cfrac{}{P(y),Q(y)\vdash Q(y)}}{P(y)\to Q(y),\ P(y)\vdash Q(y)}\ (\text{e impE})}{P(y)\to Q(y),\ \forall x.P(x)\vdash Q(y)}\ (\text{e allE } y)}{\forall x.P(x)\to Q(x),\ \forall x.P(x)\vdash Q(y)}\ (\text{e allE } y)}{\forall x.P(x)\to Q(x),\ \forall x.P(x)\vdash \forall x.Q(x)}\ (\text{allI})$$

FOL IN ISABELLE [HOL]

→ All variables, terms & formulas have types

→ The type lang. is built using

- base types – such as *bool* and *nat*
- type constructors – such as *list* and *set* e.g. *nat list* or *nat set*
- function types – e.g. $nat \times nat \Rightarrow nat$ which is a fn taking 2 args of type nat and return an obj. of type nat
- type variables – such as 'a, 'b, etc. → give rise to polymorphic types $'a \Rightarrow 'a$

→ Consider the predicate $a = b \bmod n$

↳ definition mod :: "int ⇒ int ⇒ int ⇒ bool"

where "mod a b n ≡ ∃k. a = k * n + b"

→ Isabelle performs type inference, allowing us to write:

$\forall x\, y\, n.\ \text{mod } x\ y\ n \rightarrow \text{mod } y\ x\ n$

$$\frac{\Gamma \vdash P[x_0/x]}{\Gamma \vdash \forall x.\ P}\ (\text{allI}) \qquad \frac{\Gamma, P[t/x] \vdash Q}{\Gamma, \forall x.P \vdash Q}\ (\text{e allE t}) \qquad \frac{\Gamma, \forall x.P, P[t/x] \vdash Q}{\Gamma, \forall x.P \vdash Q}\ (\text{f spec t})$$

$$\frac{\Gamma \vdash P[t/x]}{\Gamma \vdash \exists x.P}\ (\text{r exI t}) \qquad \frac{\Gamma, P[x_0/x] \vdash Q}{\Gamma, \exists x.P \vdash Q}\ (\text{e exE}) \qquad \frac{\Gamma, \forall x.\neg P \vdash \bot}{\Gamma \vdash \exists x.P}\ (\text{exCIF})$$

- Rule prefixes: e = erule, f = frule, r = rule
- x_0 is some variable **not** free in hypotheses or conclusion. Isabelle automatically picks fresh names (to ensure soundness!)
- When *t* suffix is used above (e.g., as in "e allE t"), then the term *t* can be explicitly specified in Isabelle method using a variant of the existing method. *e.g.*, `apply (erule_tac x="t" in allE)`.
- Rule `exCIF` is a variation on the standard Isabelle rule `exCI` introduced in the `FOL.thy` file on the course webpage. It does not exist as an explicit Isabelle inference rule but can be derived (see `FOL.thy`).

Why the side conditions on `allI` and `exE`?

A (non-)proof of: $\vdash x > 5 \rightarrow \forall x.\ x > 5$:

$$\dfrac{\dfrac{\dfrac{}{x > 5 \vdash x > 5}\ (\text{assumption})}{x > 5 \vdash \forall x.\ x > 5}\ (\text{allI})}{\vdash x > 5 \rightarrow \forall x.\ x > 5}\ (\text{impI})$$

But it is clearly false that if a particular x is greater than 5, then every x is greater than 5. We have "proven" that $x > 5$, but not for an **arbitrary** x, only for the **particular** x we had already made an assumption about.

Exercise: Give a non-proof for exE.

Machine assistance: Isabelle keeps track of which variable names are allowed where, so we can only apply the rules in a sound way.