Typy a formule

- $K = (\lambda xy.x)^{\alpha \text{Ÿ} \beta \text{Ÿ} \alpha}$
- $S = \lambda xyz.xz(yz)^{(\delta \text{Ÿ} \varepsilon \text{Ÿ} \theta)\text{Ÿ} (\delta \text{Ÿ} \varepsilon)\text{Ÿ} \delta \text{Ÿ} \theta}$

Hilbertov axiomatický systém:
A Ÿ (B Ÿ A)
((A Ÿ (BŸ C)) Ÿ ((AŸ B) Ÿ (AŸ C))

Modus ponens:

$$\frac{A\text{Ÿ} B \qquad A}{B}$$

K = (λxy.x)αŸ βŸ α
S = λxyz.xz(yz)(δŸ εŸ θ)Ÿ (δŸ ε)Ÿ δŸ θ

Curry-Howardov izomorfizmus

- S K K = I

S:(δŸ εŸ θ)Ÿ (δŸ ε)Ÿ (δŸ θ)
K:αŸ βŸ α
K:űŸ ɣŸ ű

(δŸ εŸ θ)= (αŸ βŸ α)
- δ= α, ε=β

(δŸ ε)=(űŸ ɣ Ÿ ű)
- δ=ű, ε=ɣ Ÿ ű

δ=θ

S:(δŸ (ɣ Ÿ δ)Ÿ δ) Ÿ (δŸ (ɣ Ÿ δ)) Ÿ (δŸ δ)
K:(δŸ (ɣ Ÿ δ)Ÿ δ)
K:(δŸ (ɣ Ÿ δ))

K = (λxy.x)αŸ βŸ α
S = λxyz.xz(yz)(δŸ εŸ θ)Ÿ (δŸ ε)Ÿ δŸ θ

Curry-Howardov izomorfizmus

S:(δŸ (ɣ Ÿ δ)Ÿ δ) Ÿ (δŸ (ɣ Ÿ δ)) Ÿ (δŸ δ)
K:(δŸ (ɣ Ÿ δ)Ÿ δ)
K:(δŸ (ɣ Ÿ δ))

S:(AŸ (BŸ A)Ÿ A) Ÿ (AŸ (BŸ A)) Ÿ (AŸ A) K:(AŸ (BŸ A)Ÿ A)

SK:(AŸ (BŸ A)) Ÿ (AŸ A) K:(AŸ (BŸ A))

SKK:(AŸ A)

Polymorfický (druhorádový) λ-kalkul, System F_2 (Girard-Reynold)

V jednoducho-typovanom λ-kalkule doménou premenných sú funkcie,
v druho-rádovom λ-kalkule doménou premenných sú typy:

Typ

σ ::= Int | σ Ÿ σ | α | ∀α.σ - všeobecne kvantifikovaný typ

- λx.x : ∀α.αŸ α
- NOT : ∀α.αŸ α
- K=TRUE (λx.λy.x) : ∀α.∀β.αŸ βŸ α, FALSE (λx.λy.y) : ∀α.∀β.αŸ βŸ β
- 0,1,2 (λf.λx.f(f x)),… : ∀α.(αŸ α)Ÿ αŸ α

Dôsledky:

- v takomto λ-kalkule vieme otypovať aj to, čo v F_1 nevieme (ω)
- inferencia v takomto kalkule je nerozhodnuteľný problém ☹, 1970-1994
- hierarchia F_1, F_2, F_3, … Ÿ F_ω
- dependent type (t:Type, t)

System F_2

Typy

σ ::= Int | σ Ÿ σ | α | ∀α.σ

{x:α}:ΓƊ x:α	[VAR]
{x:α}:ΓƊ N:β ΓƊ (λx.N):αŸ β	[ABS]
ΓƊ M:αŸ β, ΓƊ N:α ΓƊ (M N):β	[APPL]
ΓƊ M:β ΓƊ M:∀α.β	[GEN] α not free in Γ
ΓƊ M:∀α.β ΓƊ M:β[α:θ]	[INST]

Príklady v F_2

λx.x : ∀α.αŸ α
AND : ∀α.αŸ αŸ α
0,1,2,... : ∀t.(tŸ t)Ÿ (tŸ t)

{x:α}Ɗ x:α	[VAR]	{x:Int}Ɗ x:Int	[VAR]
Ɗ(λx.x):αŸ α	[ABS]	Ɗ(λx.x):IntŸ Int	[ABS]
Ɗ(λx.x): ∀α.αŸ α	[GEN]		
Ɗ(λx.x): IntŸ Int	[INST]		

{x:∀α.αŸ α}Ɗx:∀α.αŸ α [VAR]

{x:∀α.αŸ α}Ɗx:(∀β.βŸ β)Ÿ (∀β.βŸ β) [INST] {x:∀α.αŸ α}Ɗ x:∀α.αŸ α [VAR]

{x:∀α.αŸ α}Ɗ(x x):∀α.αŸ α [APP]

Ɗ λx.(x x):(∀α.αŸ α)Ÿ (∀α.αŸ α) [ABS]

- podarilo sa nám otypovať výraz, ktorý v jednoducho-typovanom kalkule sa nedal
- dostali sme však typ (∀α.αŸ α)Ÿ (∀α.αŸ α), ktorý vnútri obsahuje kvantifikátory (deep type) na rozdiel od tých, čo ich majú len na najvyššej úrovni, napr. ∀t.(tŸ t)Ÿ (tŸ t) – shallow type

Let polymorfizmus Hindley-Millner

chceme zakázať kvantifikáciu typov vnútri typového výrazu (zakázať deep type)

Typy

σ ::= ψ | ∀α.σ

ψ ::= Int | ψ Ÿ ψ | α

premenná viazaná λ výrazom nemôže byť polymofrického typu, napr.
f (λx.x) where f = λg.[… (g 1) …. (g True) …], alebo v Haskelli:

- idd = (\x->x)
- foo = f idd where f = \g->(if (g True) then (g 2) else (g 1))

Nahradíme pravidlo [GEN] pravidlom [LET]

<u>ΓƊ M:β, Γ, x:∀α.β Ɗ N:δ</u> [LET] α in β, α not free in Γ

ΓƊ let x=M in N : δ