

A legegyszerűbb programozási nyelv

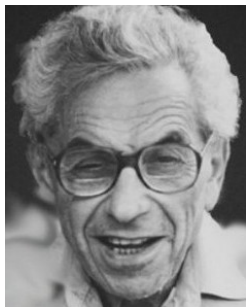
Kaposi Ambrus

Eötvös Loránd Tudományegyetem
Informatikai Kar

Kutatók Éjszakája
2021. szeptember 24.

Kombinatoristák

a program gyors legyen
teljesítmény
régi karbantartása
hacker
tartalom
C



Erdős Pál

Logisták

a program helyesen működjön
modularitás, absztrakció
új rendszer
nyelvész
forma
Haskell



Alexander Grothendieck

Formális nyelv (i)

név ::= Mózes | Alonso | Alan

alany ::= *név* | bicikli

tárgy ::= *alanyt*

minőségjelző ::= Okos | Kék

menyiségjelző ::= egy | három

állítmány ::= teker | fegyelmez

mondat ::= *minőségjelző alany mennyiségjelző tárgy állítmány*.

Mondat -e?

- ▶ Okos Mózes egy biciklit teker. IGEN
- ▶ Okos Mózes három Alant fegyelmez. IGEN
- ▶ Okos Mózes fegyelmez egy Alant. NEM
- ▶ Kék Mózes egy Alonsot teker. IGEN
- ▶ Három Alonso kék biciklit fegyelmez. NEM

Hány mondat van? $2 * (3 + 1) * 2 * 4 * 2 = 8 * 8 * 2 = 128$

Formális nyelv (ii)

név ::= Mózes | Alonso | Alan

alany ::= *név* | bicikli

tárgy ::= *alanyt*

minőségjelző ::= Okos | Kék

mennyiségjelző ::= egy | három | *mennyiségjelző* meg *mennyiségjelző*

állítmány ::= teker | fegyelmez

mondat ::= *minőségjelző alany mennyiségjelző tárgy állítmány*.

Hány mondat van?

- ▶ Okos Mózes egy Alant fegyelmez.
- ▶ Okos Mózes egy meg egy Alant fegyelmez.
- ▶ Okos Mózes egy meg egy meg egy Alant fegyelmez.
- ▶ Okos Mózes egy meg egy meg egy meg egy Alant fegyelmez.
- ▶ ...

Végtelen sok!

Csak a forma számít (i)

név ::= Imre | Kati | Marci

alany ::= *név* | trambulin

tárgy ::= *alanyt*

minőségjelző ::= Nagy | Ordítós

mennyiségjelző ::= sok | kevés | *mennyiségjelző* mínusz *mennyiségjelző*

állítmány ::= okít | előreenged

mondat ::= *minőségjelző alany mennyiségjelző tárgy állítmány*.

- ▶ Nagy Imre sok Marcit előreenged.
- ▶ Nagy Imre sok mínusz sok Marcit előreenged.
- ▶ Nagy Imre sok mínusz sok mínusz sok Marcit előreenged.
- ▶ Nagy Imre sok mínusz sok mínusz sok mínusz sok Marcit előreenged.
- ▶ ...

Csak a forma számít (ii)

$A ::= a \mid b \mid c$

$B ::= A \mid d$

$C ::= B e$

$D ::= f \mid g$

$E ::= h \mid i \mid E j E$

$F ::= k \mid l$

$G ::= m D \quad B \quad E \quad C \quad F$

► $m f a h c e l$

► $m f a h j h c e l$

► $m f a h j h j h c e l$

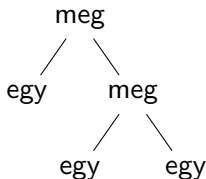
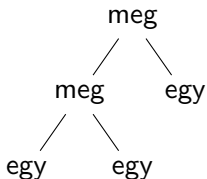
► $m f a h j h j h j h c e l$

► ...

Formális nyelv számokra (i)

$szám ::= nulla \mid egy \mid szám \text{ meg } szám$

- ▶ nulla, egy, egy meg egy, egy meg egy meg egy, ...
- ▶ (egy meg egy) meg egy \neq egy meg (egy meg egy)

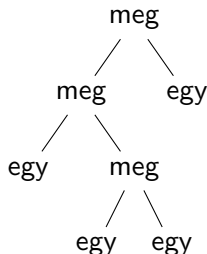


- ▶ Okos Mózes (egy meg egy) meg egy Alant fegyelmez \neq
Okos Mózes egy meg (egy meg egy) Alant fegyelmez

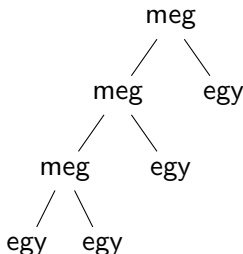
Formális nyelv számokra (ii)

$szám ::= nulla \mid egy \mid szám \text{ meg } szám$

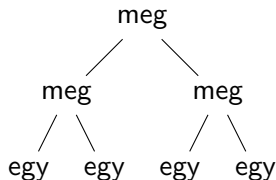
(A)



(B)



(C)



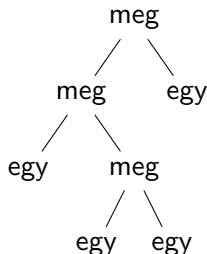
Melyik fához megyik szöveg tartozik?

1. () (egy meg egy) meg (egy meg egy)
2. () ((egy meg egy) meg egy) meg egy
3. () (egy meg (egy meg egy)) meg egy

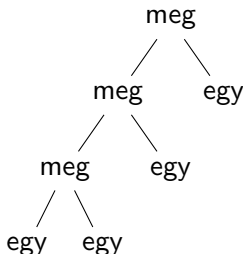
Formális nyelv számokra (ii)

$szám ::= nulla \mid egy \mid szám \text{ meg } szám$

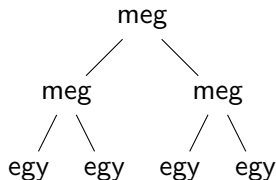
(A)



(B)



(C)



Melyik fához megyik szöveg tartozik?

1. (C) (egy meg egy) meg (egy meg egy)
2. (B) ((egy meg egy) meg egy) meg egy
3. (A) (egy meg (egy meg egy)) meg egy

Formális nyelv számokra, redundancia nélkül (i)

$szám ::= nulla \mid egy \mid szám \text{ meg } szám \mid$

$$x \text{ meg } (y \text{ meg } z) = (x \text{ meg } y) \text{ meg } z$$

$egy \text{ meg } (egy \text{ meg } egy) = (egy \text{ meg } egy) \text{ meg } egy$

$$x \text{ meg } (y \text{ meg } z) = (x \text{ meg } y) \text{ meg } z$$

$egy \text{ meg } (egy \text{ meg } (egy \text{ meg } egy)) = (egy \text{ meg } egy) \text{ meg } (egy \text{ meg } egy)$

$$x \text{ meg } (y \text{ meg } z) = (x \text{ meg } y) \text{ meg } z$$

$(egy \text{ meg } egy) \text{ meg } (egy \text{ meg } egy) = ((egy \text{ meg } egy) \text{ meg } egy) \text{ meg } egy$

$$x \text{ meg } (y \text{ meg } z) = (x \text{ meg } y) \text{ meg } z$$

Formális nyelv számokra, redundancia nélkül (ii)

$szám ::= nulla \mid egy \mid szám \text{ meg } szám \mid$

$x \text{ meg } (y \text{ meg } z) = (x \text{ meg } y) \text{ meg } z$

Most már minden szám átasszociálható ilyen alakra:

$((\dots((egy \text{ meg } egy) \text{ meg } egy)\dots) \text{ meg } egy) \text{ meg } egy$

De még mindig van egy kis redundancia:

$egy \neq egy \text{ meg } nulla \neq nulla \text{ meg } egy \neq nulla \text{ meg } nulla \text{ meg } egy$

Az lenne a jó, ha a nullák hozzáadása nem változtatni a számon.

Formális nyelv számokra, redundancia nélkül (iii)

$szám ::= nulla \mid egy \mid szám \text{ meg } szám \mid$

$x \text{ meg } (y \text{ meg } z) = (x \text{ meg } y) \text{ meg } z \mid$

$x \text{ meg } nulla = x \mid$

$nulla \text{ meg } x = x$

- ▶ A nullák eltüntethetők. (Kivéve mikor?)
- ▶ Ha már nincsenek nullák, akkor átasszociálunk.

Az összes szám:

- ▶ nulla
- ▶ egy
- ▶ egy meg egy
- ▶ (egy meg egy) meg egy
- ▶ ...

Formális nyelv számokra, redundancia nélkül (iv)

$szám ::= nulla \mid egy \mid szám \text{ meg } szám \mid$

$x \text{ meg } (y \text{ meg } z) = (x \text{ meg } y) \text{ meg } z \mid$

$x \text{ meg } nulla = x \mid$

$nulla \text{ meg } x = x$

Meg lehet-e a számokat adni egyenlőségek és redundancia nélkül?

$szám ::= nulla \mid szám \text{ megegy}$

- ▶ nulla
- ▶ nulla megegy
- ▶ (nulla megegy) megegy
- ▶ ((nulla megegy) megegy) megegy
- ▶ ...

Ezek a fák hogyan néznek ki?

Mi az, hogy program?



Moses Schönfinkel

kombinátor logika



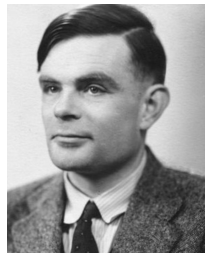
Alonzo Church

lambda kalkulus



Kurt Gödel

μ -rekurzív
függvények



Alan Turing

Turing-gép

Moses Schönfinkel formális nyelve

$$P ::= K \mid S \mid PP \mid \textcolor{red}{K}xy = x \mid \textcolor{red}{S}xyz = xz(yz)$$

Egyezményes jelölés: $xyz := (xy)z$.

Soroljuk fel a programokat!

- ▶ K, S
- ▶ KK, KS, SK, SS
- ▶ $(KK)K, (KK)S, (KS)K, (KS)S, (SK)K, (SK)S, (SS)K, (SS)S$
- ▶ $K(KK), K(KS), K(SK), K(SS), S(KK), S(KS), S(SK), S(SS)$
- ▶ $((KK)K)K, \dots, ((SS)S)S$
- ▶ $(K(KK))K, \dots, (S(SS))S$
- ▶ $(KK)(KK), \dots, (SS)(SS)$
- ▶ $K((KK)K), \dots, S((SS)S)$
- ▶ $K(K(KK)), \dots, S(S(SS))$

Moses Schönfinkel formális nyelve

$$P ::= K \mid S \mid PP \mid K_{xy} = x \mid S_{xyz} = xz(yz)$$

Egyezményes jelölés: $xyz := (xy)z$.

Soroljuk fel a programokat!

- ▶ K, S
- ▶ KK, KS, SK, SS
- ▶ SKK, SKS, SSK, SSS
- ▶ $K(KK), K(KS), K(SK), K(SS), S(KK), S(KS), S(SK), S(SS)$
- ▶
- ▶ $S(KK)K, \dots, S(SS)S$
- ▶ $(SK)(KK), \dots, (SS)(SS)$
- ▶ $K(SK K), \dots, S(SSS)$
- ▶ $K(K(KK)), \dots, S(S(SS))$

Programozás

$$P ::= K \mid S \mid PP \mid Kxy = x \mid Sxyz = xz(yz)$$

- ▶ Identitás: $Ix = x$

Definíció: $I := SKK$

Próbáljuk ki! $Ix = SKKx = Kx(Kx) = x$

- ▶ Utolsó: $Uxy = y$

Definíció: $U := KI$

Teszt: $Uxy = KIx = Iy = y$

- ▶ Duplázás: $Dx = xx$

Visszafele definíció: $xx = Ix(Ix) = SIIx$, szóval $D := SII$

- ▶ Balról asszociál: $Bxyz = x(yx)$

Definíció: $B := S(KS)K$

$Bxyz = S(KS)Kxyz = KSx(Kx)yz = S(Kx)yz = Kxz(yz) = x(yz)$

- ▶ Csere: $Cxyz = xzy$

Definíció: $C := S(S(KB)S)(KK)$

- ▶ Fordít: $Fxy = yx$

Definíció: $F := S(K(SI))(S(KK)I)$

Számok Schönfinkel nyelvén

$$P ::= K \mid S \mid PP \mid \textcolor{red}{K}xy = x \mid \textcolor{red}{S}xyz = xz(yz)$$

Eddigi programjaink tudása (library/programkönyvtár interface-e):

$$Ix = x \quad Uxy = y \quad Dx = xx \quad Bxyz = x(yz) \quad Cxyz = xzy \quad Fxy = yx$$

Számok Church-kódolása: $0fx = x$, $1fx = fx$, $2fx = f(fx)$, $3fx = f(f(fx))$

Implementáljuk! $0 := U$

Meg egy: $Mafx = f(afx)$

Visszafele: $f(afx) = Bf(af)x = SBafx$

$$M := SB$$

Megvan az összes szám: $0 := U, 1 := MU, 2 := M(MU), 3 := M(M(MU))$

Összeadás: $\ddot{O}abfx = af(bfx)$

$$af(bfx) = B(af)(bf)x = BBaf(bf)x = S(BBa)bfx = BS(BB)abfx$$

$$\ddot{O} := BS(BB)$$