# Arytmetyzacja składni

Jerzy Pogonowski

Zakład Logiki i Kognitywistyki UAM
www.kognitywistyka.amu.edu.pl
http://logic.amu.edu.pl/index.php/Dydaktyka
pogon@amu.edu.pl

12i2016

# Plan wykładu

Przygotowujemy się do przedstawienia kilku ważnych twierdzeń metalogicznych (Gödla, Rossera, Tarskiego, Löba).

- Przypomnimy *aksjomatyczną teorię arytmetyki*: PA (Arytmetykę Peana pierwszego rzędu).
- Powiemy, co to znaczy, że zbiory, relacje i funkcje rekurencyjne są *reprezentowalne* w PA.
- Pokażemy także, jak dokonuje się *arytmetyzacji składni*, czyli jak koduje się w PA pojęcia składniowe.
- Podamy definicję *hierarchii arytmetycznej*.

# Język PA

Alfabet symboli:

- stałe logiczne: $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\equiv$, $\forall$, $\exists$,
- zmienne indywidualne: $x_1, x_2, \ldots$,
- predykat (2-arg.) identyczności: $\doteq$,
- stała indywidualna (zero): $\underline{0}$,
- symbol funkcyjny (1-arg.) następnika: $\underline{s}$,
- symbole funkcyjne (2-arg.) dodawania $\underline{+}$ oraz mnożenia $\underline{\times}$,
- symbole pomocnicze: nawias lewy ( oraz nawias prawy ).

Wyrażeniem języka PA jest dowolny skończony ciąg symboli alfabetu języka PA.

Uwaga. Skorzystamy z możliwości zastąpienia notacji infiksowej przez prefiksową.

# Termy

Zbiór termów jest najmniejszym zbiorem wyrażeń języka PA takim, że:

- stała $\underline{0}$ jest termem,
- każda zmienna indywidualna $x_1, x_2, \ldots$ jest termem,
- jeśli $t$ jest termem, to $\underline{s}(t)$ jest termem,
- jeśli $t_1$ i $t_2$ są termami, to $(t_1 \underline{+} t_2)$ oraz $(t_1 \underline{\times} t_2)$ są termami.

Ze względów natury psychologicznej piszemy zwykle:

- $(t_1 \underline{+} t_2)$ zamiast $\underline{+}(t_1, t_2)$
- $(t_1 \underline{\times} t_2)$ zamiast $\underline{\times}(t_1, t_2)$.

# Formuły

Zbiór formuł jest najmniejszym zbiorem wyrażeń języka PA takim, że:

- jeśli $t_1$, $t_2$ są termami, to $t_1 \doteq t_2$ jest formułą,
- jeśli $\psi$ jest formułą, a $x_i$ zmienną indywidualną, to formułami są też $\neg\psi$, $\forall x_i \psi$, $\exists x_i \psi$,
- jeśli $\varphi$, $\psi$ są formułami, to formułami są także: $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$, $(\varphi \rightarrow \psi)$, $(\varphi \equiv \psi)$.

Stosujemy konwencje opuszczania nawiasów, znane z elementarnego kursu logiki. Dla wygody, opuszczamy też czasem indeks przy zmiennych indywidualnych.

# Aksjomaty logiczne

Aksjomatami logicznymi są wszystkie podstawienia formuł języka PA za zmienne zdaniowe w następujących formułach języka KRZ:

1) $p \to (q \to p)$
2) $(p \to (q \to r)) \to ((p \to q) \to (p \to r))$
3) $(p \to q) \to (\neg q \to \neg p)$
4) $\neg\neg p \to p$
5) $p \to \neg\neg p$
6) $(p \wedge q) \to p$
7) $(p \wedge q) \to q$
8) $(p \to q) \to ((p \to r) \to p \to (q \wedge r))$
9) $p \to (p \vee q)$
10) $q \to (p \vee q)$
11) $(p \to r) \to ((q \to r) \to ((p \vee q) \to r))$
12) $(p \equiv q) \to (p \to q)$
13) $(p \equiv q) \to (q \to p)$
14) $(p \to q) \to ((q \to p) \to (p \equiv q))$

# Aksjomaty identyczności

Aksjomatami identyczności dla PA są:

- $x \doteq x$
- $x \doteq y \to y \doteq x$
- $(x \doteq y \wedge y \doteq z) \to x \doteq z$
- $x \doteq y \to \underline{s}(x) \doteq \underline{s}(y)$
- $x \doteq y \to x \underline{+} z \doteq y \underline{+} z$
- $x \doteq y \to z \underline{+} x \doteq z \underline{+} y$
- $x \doteq y \to x \underline{\times} z \doteq y \underline{\times} z$
- $x \doteq y \to z \underline{\times} x \doteq z \underline{\times} y$

# Aksjomaty pozalogiczne

Aksjomatami specyficznymi PA są:

- (A1) $\underline{s}(x) \doteq \underline{s}(y) \rightarrow x = y$
- (A2) $\neg \underline{0} \doteq \underline{s}(x)$
- (A3) $x \underline{+} \underline{0} \doteq x$
- (A4) $x \underline{+} \underline{s}(y) \doteq \underline{s}(x \underline{+} y)$
- (A5) $x \underline{\times} \underline{0} \doteq \underline{0}$
- (A6) $x \underline{\times} \underline{s}(y) \doteq (x \underline{\times} y) \underline{+} x$
- (A7) $(\varphi(\underline{0}) \wedge \forall x(\varphi(x) \rightarrow \varphi(\underline{s}(x)))) \rightarrow \forall x \varphi(x)$

(A7) jest schematem (nieskończenie wielu) aksjomatów, zwanym *schematem indukcji matematycznej*.

# Reguły wnioskowania

Regułami wnioskowania są:

- reguła podstawiania: $\frac{\psi}{\psi(x/t)}$, o ile term $t$ jest podstawialny za zmienną $x$ w formule $\psi$,
- reguła odrywania: $\frac{\varphi \rightarrow \psi \quad \varphi}{\psi}$
- reguła generalizacji: $\frac{\psi}{\forall x \psi}$
- reguła (O$\forall$): $\frac{\varphi \rightarrow \forall x \psi}{\varphi \rightarrow \psi}$
- reguła (D$\forall$): $\frac{\varphi \rightarrow \psi}{\varphi \rightarrow \forall x \psi}$, o ile $x$ nie jest zmienną wolną w $\varphi$
- reguła (O$\exists$): $\frac{\exists x \varphi \rightarrow \psi}{\varphi \rightarrow \psi}$
- reguła (D$\exists$): $\frac{\exists x \varphi \rightarrow \psi}{\varphi \rightarrow \psi}$, o ile $x$ nie jest zmienną wolną w $\psi$.

# Semantyka

Wykorzystujemy semantykę dla teorii pierwszego rzędu, podaną w elementarnym kursie logiki.

Spośród wszystkich modeli PA (a jest ich bardzo wiele!) wyróżnia się model standardowy $\mathfrak{N}_0$, w którym:

- uniwersum stanowią wszystkie liczby naturalne $\{0, 1, 2, 3, \ldots\}$
- interpretacją stałej $\underline{0}$ jest liczba zero,
- interpretacją predykatu $\doteq$ jest relacja równości $=$,
- interpretacją symbolu $\underline{+}$ jest operacja dodawania $+$,
- interpretacją symbolu $\underline{\times}$ jest operacja mnożenia $\cdot$,
- interpretacją symbolu $\underline{s}$ jest funkcja następnika $s$: $s(n) = n + 1$.

# Arytmetyka Robinsona

- Aksjomat indukcji nie może zostać zastąpiony żadnym równoważnym mu skończonym zbiorem aksjomatów.
- Nie można zastąpić go także żadną równoważną (nawet nieskończoną) liczbą przypadków szczególnych, tj. aksjomatu indukcji dla formuł o dowolnie z góry ograniczonej liczbie kwantyfikatorów.

System w powyższym języku, o aksjomatach pozalogicznych (A1)–(A6) (a więc bez aksjomatu indukcji) nazywa się Arytmetyką Robinsona i bywa oznaczany $Q$.

- $Q$ jest istotnie słabszy od PA: można w nim dowodzić konkretnych prawd arytmetycznych (np. tego, że $2 + 2 = 4$), ale nie wielu praw ogólnych (np. tego, że dodawanie jest przemienne).

# Wybrane proste twierdzenia arytmetyczne

Jeśli $\psi$ jest twierdzeniem PA, to piszemy: $PA \vdash \psi$.
Wygodnie jest wprowadzić pewne zdefiniowane predykaty, np. *mniejszość*:

$$x \underline{\leq} y \ \equiv_{df} \ \exists z(\neg z \doteq \underline{0} \wedge z \underline{+} x \doteq y).$$

Wtedy interpretacją $\underline{\leq}$ w $\mathfrak{N}_0$ jest relacja mniejszości $<$.
Uwaga. Nie myl predykatu $\underline{\leq}$ (podkreślony symbol $<$) z predykatem nazywającym relację „mniejsze lub równe", którą zapisujemy $\leqslant$. Odróżniaj predykat identyczności $\doteq$ od relacji równości $=$, stałą $\underline{0}$ od liczby 0, symbol funkcyjny $\underline{s}$ od funkcji następnika $s$, itd.

Niektóre proste twierdzenia arytmetyczne są potrzebne w dowodach twierdzeń o reprezentowalności oraz twierdzeń dotyczących arytmetyzacji składni. Podamy wybrane z nich.

# Dygresja

Odróżniliśmy (poprzez kształt stosowanych symboli) stałe pozalogiczne arytmetyki od ich semantycznych interpretacji.

Dla pełnej (pedantycznej) poprawności trzeba byłoby również np. używać innych oznaczeń na stałe logiczne w języku przedmiotowym, a inne w tej roli w metajęzyku.
Podobnie, kształt zmiennych języka przedmiotowego powinien być różny od zmiennych metajęzykowych (przebiegających, w naszym przypadku, liczby naturalne).

Nie robimy tego. **Ufamy.** Ufamy, że kontekst użycia symbolu pozwala na uniknięcie nieporozumień w rozumieniu, co w tekście porabia ten symbol, jaki jest jego status, itd.

# Wybrane proste twierdzenia arytmetyczne

W PA można udowodnić:

- łączność i przemienność dodawania i mnożenia,
- rozdzielność mnożenia względem dodawania,
- prawa skracania dla dodawania i mnożenia,
- zgodność porządku z operacjami arytmetycznymi.

- $PA \vdash \neg x \underline{\leq} \underline{0}$.
- $PA \vdash x \underline{\leq} \underline{s}(y) \equiv (x \underline{\leq} y \vee x \doteq y)$.
- $PA \vdash x \underline{\leq} y \vee x \doteq y \vee y \underline{\leq} x$.

# Wybrane proste twierdzenia arytmetyczne

Niech formuła $\varphi'$ powstaje z formuły $\varphi$ poprzez zastąpienie niektórych wystąpień formuł $\psi_1, \ldots, \psi_n$ odpowiednio formułami $\psi'_1, \ldots, \psi'_n$ i załóżmy, że: $PA \vdash \psi_1 \equiv \psi'_1, \ldots, PA \vdash \psi_n \equiv \psi'_n$. Wtedy: $PA \vdash \varphi \equiv \varphi'$.

Niech term $t$ powstaje z termu $t'$ poprzez (poprawne!) zastąpienie niektórych wystąpień termów $t_1, \ldots, t_n$ odpowiednio termami $t'_1, \ldots, t'_n$, a formuła $\varphi'$ powstaje z formuły $\varphi$ poprzez takie samo zastąpienie wymienionych termów. Załóżmy, że: $PA \vdash t_1 \doteq t'_1, \ldots, PA \vdash t_n \doteq \beta'_n$. Wtedy: $PA \vdash t \doteq t'$ oraz $PA \vdash \varphi \equiv \varphi'$.

Powyższe twierdzenia zachodzą nie tylko dla PA, lecz także dla dowolnej teorii pierwszego rzędu.

# Liczebniki

**Definicja liczebników.**

- Term $\underline{0}$ jest liczebnikiem.
- Jeśli term $\alpha$ jest liczebnikiem, to term $\underline{s}(\alpha)$ jest liczebnikiem.
- Liczebnikami są tylko termy opisane w powyższy sposób.

Oznaczmy: $\overline{n} = \underbrace{\underline{s}(\underline{s}(\ldots\underline{s}}_{n \text{ razy}}(\underline{0})\ldots))$.

$\overline{n}$ jest zatem liczebnikiem nazywającym liczbę $n$.

Uwaga. Liczebniki są symbolami językowymi, liczby naturalne są elementami uniwersum modelu standardowego.

# Słaba reprezentowalność relacji

- Formuła $\varphi$ języka PA o $n$ zmiennych wolnych ***słabo reprezentuje*** w PA relację $R \subseteq \omega^n$, wtedy i tylko wtedy, gdy dla dowolnych liczb naturalnych $k_1, \ldots, k_n$ zachodzi równoważność:
  $R(k_1, \ldots, k_n)$ wtedy i tylko wtedy, gdy PA $\vdash \varphi(\overline{k_1}, \ldots, \overline{k_n})$.
- Relację $R \subseteq \omega^n$ nazywamy ***słabo reprezentowalną*** w PA, jeśli istnieje formuła języka PA, która słabo reprezentuje $R$.

Uwaga. Formuła $\varphi$ słabo reprezentuje $R$ w PA wtedy i tylko wtedy, gdy zachodzą implikacje:

- Jeśli $R(k_1, \ldots, k_n)$, to $\vdash \varphi(\overline{k_1}, \ldots, \overline{k_n})$.
- Jeśli $\vdash \varphi(\overline{k_1}, \ldots, \overline{k_n})$, to $R(k_1, \ldots, k_n)$.

# Mocna reprezentowalność relacji

- Formuła $\varphi$ języka PA o $n$ zmiennych wolnych ***mocno reprezentuje*** w PA relację $R \subseteq \omega^n$, wtedy i tylko wtedy, gdy dla dowolnych liczb naturalnych $k_1, \ldots, k_n$ zachodzą implikacje:
  - Jeśli $R(k_1, \ldots, k_n)$, to PA $\vdash \varphi(\overline{k_1}, \ldots, \overline{k_n})$.
  - Jeśli $\neg R(k_1, \ldots, k_n)$, to PA $\vdash \neg\varphi(\overline{k_1}, \ldots, \overline{k_n})$.
- Relację $R \subseteq \omega^n$ nazywamy ***mocno reprezentowalną*** w PA, jeśli istnieje formuła języka PA, która mocno reprezentuje $R$.

Uwaga. Każda relacja mocno reprezentowalna w PA jest też słabo reprezentowalna w PA, lecz nie na odwrót.

## Reprezentowalność relacji

Jeśli PA jest niesprzeczna oraz $R$ jest mocno reprezentowana w PA przez formułę $\varphi$, to zachodzą następujące równoważności:

- $R(k_1, \ldots, k_n)$ wtedy i tylko wtedy, gdy PA $\vdash \varphi(\overline{k_1}, \ldots, \overline{k_n})$.
- $\neg R(k_1, \ldots, k_n)$ wtedy i tylko wtedy, gdy PA $\vdash \neg\varphi(\overline{k_1}, \ldots, \overline{k_n})$.

Na mocy powyższego twierdzenia, relacja $R$ jest mocno reprezentowalna w PA wtedy i tylko wtedy, gdy istnieje formuła $\varphi$ języka PA taka, że:

- $R$ jest słabo reprezentowana przez $\varphi$,
- $\neg R$ jest słabo reprezentowana przez $\neg\varphi$.

# Reprezentowalność funkcji

- Formuła $\varphi$ języka PA o $n+1$ zmiennych wolnych ***reprezentuje*** w PA funkcję $f : \omega^n \to \omega$ wtedy i tylko wtedy, gdy dla dowolnych liczb naturalnych $k_1, \ldots, k_n$:
  PA$\vdash \forall y(\varphi(\overline{k_1}, \ldots, \overline{k_n}) \equiv (y \doteq \overline{f(k_1, \ldots, k_n)}))$.

- Funkcję $f : \omega^n \to \omega$ nazywamy ***reprezentowalną*** w PA wtedy i tylko wtedy, gdy istnieje formuła $\varphi$ języka PA o $n+1$ zmiennych wolnych taka, że $\varphi$ reprezentuje $f$ w PA.

# Reprezentowalność

- Relacja identyczności $=$ jest mocno reprezentowana w PA przez formułę $x_1 \doteq x_2$.
- Funkcja dodawania jest reprezentowana w PA przez formułę $x_1 \underline{+} x_2 \doteq x_3$.
- Funkcja mnożenia jest reprezentowana w PA przez formułę $x_1 \underline{\times} x_2 \doteq x_3$.
- Relacja mniejszości jest mocno reprezentowana w PA przez formułę $x_1 \underline{\leq} x_2$.
- Relacja $R \subseteq \omega^n$ jest mocno reprezentowalna w PA wtedy i tylko wtedy, gdy jej funkcja charakterystyczna jest reprezentowalna w PA.

# Twierdzenie o reprezentowalności

Dla dowolnej formuły $\varphi$ języka PA i dowolnej liczby naturalnej $n$:
$\mathrm{PA} \vdash (\varphi(0) \wedge \varphi(\overline{1}) \wedge \ldots \wedge \varphi(\overline{n-1}) \wedge x \leq \overline{n}) \rightarrow \varphi(x)$.

Dla dowolnej formuły $\varphi$ języka PA i dowolnej liczby naturalnej $n$, jeżeli dla każdego $i < n$, $\mathrm{PA} \vdash \neg\varphi(\overline{i})$ oraz $\mathrm{PA} \vdash \varphi(\overline{n})$, to:
$\mathrm{PA} \vdash (\varphi(x) \wedge \forall y (y \leq x \rightarrow \neg\varphi(y))) \equiv (x \doteq \overline{n})$.

**Twierdzenie o reprezentowalności.**

- Każda funkcja rekurencyjna jest reprezentowalna w PA.
- Każda relacja rekurencyjna jest mocno reprezentowalna w PA.

# Twierdzenie o reprezentowalności

Dowód twierdzenia o reprezentowalności jest dość łatwy, choć nieco żmudny. Dowodzi się mianowicie kolejno, że:

- funkcje proste są reprezentowalne,
- funkcja powstająca przez złożenie z funkcji reprezentowalnych jest reprezentowalna,
- funkcja powstająca przez schemat rekursji prostej z funkcji reprezentowalnych jest reprezentowalna,
- funkcja powstająca przez zastosowanie operacji minimum efektywnego do funkcji reprezentowalnej jest reprezentowalna.

W dowodzie wykorzystuje się (metasystemową) zasadę indukcji matematycznej.

# Arytmetyzacja składni

Możliwość kodowania wyrażeń języka PA przez liczby naturalne pozwala na „mówienie" o arytmetyce w niej samej, i to bez popadania w paradoksy pomieszania języka przedmiotowego i metajęzyka.
Dokładniej, owo „mówienie" w PA o PA umożliwione jest dodatkowo przez dwa fakty: to, że pojęciom metalogicznym odpowiadają funkcje i relacje rekurencyjne oraz to, że funkcje i relacje rekurencyjne są reprezentowalne w PA.

Są różne możliwości (jednoznacznego i efektywnego) kodowania wyrażeń (skończonych ciągów symboli), np.:

- użycie rozkładu liczb naturalnych na czynniki pierwsze;
- zastosowanie funkcji kodującej Cantora;
- zastosowanie funkcji kodującej $\beta$ Gödla;
- konkatenację w bazie dziesiętnej, itd.

# Wybór funkcji kodującej ciągi

**Lemat (funkcja $\beta$ Gödla).** *Istnieje 2-argumentowa funkcja rekurencyjna $\beta$ taka, że:*

- *dla dowolnych $a$ oraz $i$: $\beta(a, i) \leqslant a \dot{-} 1$*
- *dla dowolnych $a_0, a_1, \ldots, a_{n-1}$ istnieje $a$ taka, że $\beta(a, i) = a_i$, dla $i < n$.*

- Dla dowolnego ciągu liczb naturalnych $a_1, \ldots, a_n$ niech: $\langle a_1, \ldots, a_n\rangle^\beta = \mu x\ (\beta(x, 0) = n \wedge \beta(x, 1) = a_1 \wedge \ldots \wedge \beta(x, n) = a_n)$.
- $lh^\beta(a) = \beta(a, 0)$.
- $(a)_i^\beta = \beta(a, i + 1)$.
- $seq^\beta(a) \ \equiv\ \forall x < a\ (lh^\beta(x) \neq lh^\beta(a) \vee \exists i < lh^\beta(a)\ (x)_i^\beta \neq (a)_i^\beta)$.
- Dla każdej $n$, funkcja $(a_1, \ldots, a_n) \mapsto \langle a_1, \ldots, a_n\rangle^\beta$ jest rekurencyjna. Funkcje $lh^\beta$ i $(\ )_i^\beta$ są rekurencyjne. $seq^\beta$ jest relacją rekurencyjną.

# Wybór funkcji kodującej ciągi

Następujące funkcje są rekurencyjne (co widać z definicji):

- $In^{\beta}(a, i) = \mu x \ (lh^{\beta}(x) = i \wedge \forall j < i \ (x)_j^{\beta} = (a)_j^{\beta})$.
- $a *^{\beta} b = \mu x \ (lh^{\beta}(x) = lh^{\beta}(a) + lh^{\beta}(b) \wedge \forall i < lh^{\beta}(a) \ (x)_i^{\beta} = (a)_i^{\beta} \wedge$
  $\wedge \forall i < lh^{\beta}(b) \ (x)_{lh^{\beta}(a)+i}^{\beta} = (b)_i^{\beta})$.

- Wcześniej poznaliśmy funkcje podobne do powyższych (kodujące i odkodowujące) – funkcję pary Cantora oraz funkcję kodowania wykorzystującą rozkład na czynniki pierwsze.
- W dalszym ciągu będziemy pomijać indeks górny $\beta$ przy wprowadzonych wyżej symbolach.

# Dowód lematu

## Zarys dowodu lematu.

- Przypomnijmy, że relacja podzielności *div* jest rekurencyjna: $div(a, b) \ \equiv \ \exists z \leqslant a \, (a = b \cdot z)$.
- Rekurencyjna jest też symetryczna relacja *rp* zdefiniowana wzorem: $rp(a, b) \ \equiv \ a \neq 0 \wedge b \neq 0 \wedge \forall x \leqslant a \cdot b \, (div(a \cdot x, b) \rightarrow div(x, b))$, która zachodzi między $a$ i $b$ dokładnie wtedy, gdy $a$ i $b$ są względnie pierwsze.
- $(*)$ Dla dowolnych różnych od 0 i od 1 liczb $a_1, \ldots, a_n, b_1, \ldots, b_m$ takich, że $rp(a_i, b_j)$ dla wszystkich $i$ oraz $j$ istnieje liczba $c$ taka, że $div(c, a_i)$ dla $i = 1, \ldots, n$, natomiast nie zachodzi $div(c, b_j)$ dla $j = 1, \ldots, m$.
- Wprost z definicji relacji *div* oraz *rp* mamy: $(\dagger) \ \ (k \neq 0 \wedge z \neq 0 \wedge div(z, k)) \rightarrow rp(1 + (j + k) \cdot z, 1 + j \cdot z)$.

# Dowód lematu

- Funkcja $op(a,b) = (a+b)\cdot(a+b)+a+1$ jest rekurencyjna. Nadto: (‡) $op(a,b) = op(c,d) \rightarrow a = c \wedge b = d$.
- Definiujemy funkcję $\beta$:

  $\beta(a,i) = \mu x < a \dot{-} 1 \ (\exists y < a \exists z < a \ (a = op(y,z) \wedge div(y, 1 + (op(x,i)+1)\cdot z)))$.
- Z definicji wynika, że $\beta$ jest rekurencyjna oraz że $\beta(a,i) \leqslant a \dot{-} 1$.
- Dla danych liczb $a_0, a_1, \ldots, a_{n-1}$ trzeba znaleźć liczbę $a$ taką, że $\beta(a,i) = a_i$ dla $i < n$.
- Niech $c$ będzie największą z liczb $op(a_i, i) + 1$ dla $i < n$, natomiast $z$ liczbą podzielną przez wszystkie $x$ takie, że $x < c$.
- Na mocy (†) dla $k = r - j$ mamy: jeśli $j < r < c$, to $rp(1 + j \cdot z, 1 + r \cdot z)$.

# Dowód lematu

- Na mocy $(*)$ istnieje $y$ taka, że dla $j < c$: $div(y, 1 + j \cdot z)$ dokładnie wtedy, gdy $j$ jest postaci $op(a_i, i) + 1$.
- Niech $a = op(y, z)$. Wtedy $a_i < y < a$ oraz $z < a$.
- Dla dowodu, że $\beta(a, i) = a_i$ dla $i < n$ wystarczy pokazać, że: $a_i = \mu x \; div(y, 1 + (op(x, i) + 1) \cdot z)$.
- To wynika z faktu, że: jeśli $x < a_i$, to $op(x, i) < c$ oraz $op(x, i)$ nie jest postaci $op(a_j, j)$ dla $j < n$.
- Na mocy $(\ddagger)$ mamy jednak: $op(x, i) \leqslant op(a_i, i) < c$ oraz $op(x, i) \neq op(a_j, j)$, co kończy dowód.

# Funkcja-pamięć

- Dla dowolnej funkcji $f : \omega^n \to \omega$ jej ***ściągnięciem*** nazywamy funkcję: $\lceil f \rceil(a) = f((a)_0, \ldots, (a)_{n-1})$
- Dla dowolnej relacji $R \subseteq \omega^n$ jej ***ściągnięciem*** nazywamy relację: $\lceil R \rceil(a) \ \equiv \ R((a)_0, \ldots, (a)_{n-1})$.
- Dla dowolnej funkcji $f : \omega^n \to \omega$ jej ***funkcją-pamięcią*** nazywamy funkcję: $\widehat{f}(a_1, \ldots, a_{n-1}, b) =$
  $= \langle f(a_1, \ldots, a_{n-1}, 0), f(a_1, \ldots, a_{n-1}, 1), \ldots, f(a_1, \ldots, a_{n-1}, b-1) \rangle$.

- $f(a_1, \ldots, a_n) = \lceil f \rceil(\langle a_1, \ldots, a_n \rangle)$.
- $R(a_1, \ldots, a_n) \ \equiv \ \lceil R \rceil(\langle a_1, \ldots, a_n \rangle)$.

Funkcja-pamięć może też zostać określona przy użyciu innych (pierwotnie rekurencyjnych) funkcji kodujących.

# Funkcja-pamięć

- Funkcja $f$ jest rekurencyjna dokładnie wtedy, gdy rekurencyjna jest jej funkcja-pamięć $\widehat{f}$.

**Dowód.**

- Jeśli $f$ rekurencyjna, to $\widehat{f}$ rekurencyjna, ponieważ:
$$\widehat{f}(a_1, \ldots, a_n, b) = \mu x\ (seq(x) \wedge lh(x) = b \wedge \forall i < b\ ((x)_i) = f(a_1, \ldots, a_n, b)).$$
- Jeśli $\widehat{f}$ rekurencyjna, to $f$ rekurencyjna, ponieważ:
$$f(a_1, \ldots, a_n, b) = (\widehat{f}(a_1, \ldots, a_n, b+1))_b.$$

Funkcja-pamięć pozwala zastąpić pewne definicje przez schematy rekursji definicjami używającymi tylko funkcji pierwotnie rekurencyjnych oraz efektywnego operatora minimum:

# Funkcja-pamięć

Jeśli $g$ oraz $h$ są funkcjami rekurencyjnymi, to funkcja $f$ określona następującym schematem rekursji:

- $f(a_1, \ldots, a_n, b) = g(a_1, \ldots, a_n, )$, gdy $b = 0$,
- $f(a_1, \ldots, a_n, b) = h(\widehat{f}(a_1, \ldots, a_n, b), a_1, \ldots, a_n, b)$, gdy $b > 0$

także jest rekurencyjna.

Mamy bowiem:

$$H(a_1, \ldots, a_n, b) = \mu x \; (seq(x) \wedge lh(x) = b \wedge (x)_0 = g(a_1, \ldots, a_n) \wedge \\ \forall i < b \; (i > 0 \to (x)_i) = h(In(x, i), a_1, \ldots, a_n, i)).$$

A zatem $H$ jest rekurencyjna. Nadto, jest równa funkcji $f$. Rekurencyjność $f$ jest teraz oczywista:

- dla $b = 0$ funkcja $f$ jest równa rekurencyjnej funkcji $g$,
- dla $b > 0$ mamy: $f(a_1, \ldots, a_n, b) = h(H(a_1, \ldots, a_n, b), a_1, \ldots, a_n, b)$.

# Funkcja-pamięć

W konsekwencji, jeśli $g$ i $h$ są rekurencyjne, to funkcja $f$ określona warunkami:

- $f(a_1, \ldots, a_n, 0) = g(a_1, \ldots, a_n, )$
- $f(a_1, \ldots, a_n, b+1) = h(f(a_1, \ldots, a_n, b), a_1, \ldots, a_n, b)$

również jest rekurencyjna.

Wyrażenie schematów rekursji przez funkcję-pamięć pozwala lepiej zrozumieć działanie tych schematów.

Własności operacji ściągnięcia ukazują natomiast, że funkcje (i relacje) wieloargumentowe (na liczbach naturalnych) możemy zastąpić przez równe im funkcje (i relacje) jednoargumentowe.

# Numery symboli

Każdemu symbolowi $X$ języka PA przyporządkowujemy ***numer*** $sn(X)$, np. w taki sposób:

| $X$ | $\neg$ | $\vee$ | $\wedge$ | $\rightarrow$ | $\equiv$ | $\exists$ |
|---|---|---|---|---|---|---|
| $sn(X)$ | 3 | 5 | 7 | 9 | 11 | 13 |

| $X$ | $\forall$ | $\underline{s}$ | $\underline{+}$ | $\underline{\times}$ | $\underline{0}$ | $\doteq$ |
|---|---|---|---|---|---|---|
| $sn(X)$ | 15 | 17 | 19 | 21 | 23 | 25 |

Zmiennej $x_i$ przyporządkowujemy liczbę $2i$: $sn(x_i) = 2i$.

# Kodowanie wyrażeń

Zakładamy, że każdy term i każda formuła języka PA są zapisane w notacji ***prefiksowej*** (***polskiej***), czyli: funktor przed swoim argumentami. Tak więc, każdy term jest ciągiem o postaci $vv_1 \ldots, v_n$, gdzie $v$ jest symbolem funkcyjnym, a $v_1 \ldots, v_n$ są termami. Każda formuła jest ciągiem o postaci $vv_1 \ldots, v_n$, gdzie: albo $v$ jest funktorem prawdziwościowym, a $v_1 \ldots, v_n$ są formułami lub termami, albo $v$ jest kwantyfikatorem, a $v_1 \ldots, v_n$ są formułami lub termami. Notacja prefiksowa nie wymaga stosowania nawiasów.

- Każdemu termowi lub formule $u$ o postaci $vv_1 \ldots v_n$ przyporządkowujemy jego/jej ***numer gödlowski*** $\ulcorner u \urcorner$, zdefiniowany następująco:
  $\ulcorner u \urcorner = \langle sn(v), \ulcorner v_1 \urcorner, \ldots \ulcorner v_n \urcorner \rangle$.
- Powyższa definicja może też być „rozpisana" w sposób wyraźny, dla każdego rodzaju termu lub formuły.

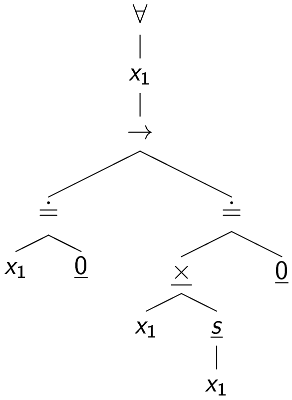
# Kodowanie wyrażeń

Dla przykładu, numer gödlowski formuły $\forall x_1\ (x_1 \doteq \underline{0} \rightarrow (x_1 \underline{\times} \underline{s}(x_1)) \doteq \underline{0})$ obliczamy następująco:

- Przekształcamy formułę do postaci prefiksowej:
  $\forall x_1 \rightarrow \doteq x_1 \underline{0} \doteq \underline{\times} x_1 \underline{s} x_1 \underline{0}$ (i w znany sposób odszukujemy jej podformuły).
- $\ulcorner \underline{s}(x_1) \urcorner = \langle 17, 2 \rangle$
- $\ulcorner x_1 \underline{\times} \underline{s}(x_1) \urcorner = \langle 21, 2, \langle 17, 2 \rangle \rangle$
- $\ulcorner (x_1 \underline{\times} \underline{s}(x_1) \doteq \underline{0}) \urcorner = \langle 25, \langle 21, 2, \langle 17, 2 \rangle \rangle, 23 \rangle$
- $\ulcorner x_1 \doteq \underline{0} \urcorner = \langle 25, 2, 23 \rangle$
- $\ulcorner x_1 \doteq \underline{0} \rightarrow (x_1 \underline{\times} \underline{s}(x_1) \urcorner = \langle 9, \langle 25, 2, 23 \rangle, \langle 25, \langle 21, 2, \langle 17, 2 \rangle \rangle, 23 \rangle \rangle$
- $\ulcorner \forall x_1\ (x_1 \doteq \underline{0} \rightarrow (x_1 \underline{\times} \underline{s}(x_1)) \doteq \underline{0}) \urcorner =$
  $\langle 15, 2, \langle 9, \langle 25, 2, 23 \rangle, \langle 25, \langle 21, 2, \langle 17, 2 \rangle \rangle, 23 \rangle \rangle \rangle$

# Kodowanie wyrażeń

Ten sposób kodowania staje się jasny, gdy spojrzymy na drzewo syntaktyczne rozważanej formuły (w istocie kodujemy właśnie to drzewo):



# Kodowanie wyrażeń

Omówione kodowanie wyrażeń jest:

- jednoznaczne (każdy term lub formuła otrzymuje dokładnie jeden numer gödlowski);
- efektywne (przypisanie numerów realizowane jest za pomocą funkcji rekurencyjnych).

Numery gödlowskie termów i formuł są oczywiście dość dużymi liczbami. W praktyce nie ma jednak żadnej potrzeby, aby je obliczać. Wystarczy możliwość ich jednoznacznego, efektywnego otrzymywania.

# Kodowanie wyrażeń

Ponadto, dla dowolnej liczby naturalnej $a$ można w sposób efektywny rozstrzygnąć, czy jest ona numerem gödlowskim termu lub formuły. Wystarczy w tym celu:

- sprawdzić, czy zachodzi $seq(a)$;
- jeśli tak, to wyznaczyć $lh(a)$ oraz $(a)_i$ dla $i = 0, 1, \ldots, lh(a)$;
- dla tak „rozłożonej" $a = \langle a_0, a_1, \ldots a_n \rangle$ sprawdzić, czy $a_0 = sn(X)$ dla jakiegoś symbolu $X$ alfabetu języka PA;
- jeśli tak, to procedurę tę powtórzyć dla $a_i$, gdzie $i = 0, 1, \ldots, lh(a)$;
- ponieważ $a_i < a$, więc po skończonej liczbie kroków procedura ta się zakończy (i uzyskamy odpowiedź czy $a$ jest numerem gödlowskim termu lub formuły).

# Kodowanie zmiennych

$\mathrm{Vble}(a) \; \equiv \; a = \langle (a)_0 \rangle \wedge \exists y \leqslant a \; ((a)_0 \doteq 2y)$

Vble($a$) zachodzi, gdy $a$ jest numerem gödlowskim pewnej zmiennej $x_i$, czyli gdy $a = \ulcorner x_i \urcorner$.

Rekurencyjność Vble wynika z faktu, że $(a)_0 < a$.

Zauważmy, że numer gödlowski zmiennej to nie to samo, co numer przypisany jej przez funkcję $sn$.

# Kodowanie termów

Formuła Term($a$) jest równoważna:

| formule | jeśli |
|---|---|
| $0 = 0$ | $a = \langle sn(\underline{0})\rangle$ |
| Term($(a_1)$) | $a = \langle sn(\underline{s}), (a)_1\rangle$ |
| Term($(a)_1$) $\wedge$ Term($(a)_2$) | $a = \langle sn(\underline{+}), (a)_1, (a)_2\rangle$ |
| Term($(a)_1$) $\wedge$ Term($(a)_2$) | $a = \langle sn(\underline{\times}), (a)_1, (a)_2\rangle$ |
| Vble($a$) | w p.p. |

Term($a$) czytamy: $a$ jest numerem gödlowskim termu, czyli $a = \ulcorner t \urcorner$ dla pewnego termu $t$. Skrót „w p.p." czytamy: „w pozostałych przypadkach." Rekurencyjność Term dostajemy z: faktu, iż $(a)_1 < a$ oraz twierdzeń o definiowaniu warunkowym i definiowaniu przez schemat rekursji.

# Kodowanie formuł

$\mathrm{AtForm}(a) \equiv$

$$a = \langle (a)_0, (a)_1, (a)_2 \rangle \wedge (a)_0 = sn(\dot{=}) \wedge \mathrm{Term}((a)_1) \wedge \mathrm{Term}((a)_2).$$

Formuła $\mathrm{Form}(a)$ jest równoważna:

| formule | jeśli |
|---|---|
| $\mathrm{Form}((a)_1)$ | $a = \langle sn(\neg), (a)_1 \rangle$ |
| $\mathrm{Form}((a)_1) \wedge \mathrm{Form}((a)_2)$ | $a = \langle sn(\vee), (a)_1, (a)_2 \rangle$ |
| $\mathrm{Form}((a)_1) \wedge \mathrm{Form}((a)_2)$ | $a = \langle sn(\wedge), (a)_1, (a)_2 \rangle$ |
| $\mathrm{Form}((a)_1) \wedge \mathrm{Form}((a)_2)$ | $a = \langle sn(\rightarrow), (a)_1, (a)_2 \rangle$ |
| $\mathrm{Form}((a)_1) \wedge \mathrm{Form}((a)_2)$ | $a = \langle sn(\equiv), (a)_1, (a)_2 \rangle$ |
| $\mathrm{Vble}((a)_1) \wedge \mathrm{Form}((a)_2)$ | $a = \langle sn(\exists), (a)_1, (a)_2 \rangle$ |
| $\mathrm{Vble}((a)_1) \wedge \mathrm{Form}((a)_2)$ | $a = \langle sn(\forall), (a)_1, (a)_2 \rangle$ |
| $\mathrm{AtForm}(a)$ | w p.p. |

$\mathrm{Form}(a)$ zachodzi, gdy $a$ jest numerem gödlowskim formuły języka PA.

# Kodowanie operacji syntaktycznych

Funkcja $\mathrm{Sub}(a, b, c)$:

| ma wartość | jeśli |
|---|---|
| $c$ | $\mathrm{Vble}(a) \wedge a = b$ |
| $\langle (a)_0, \mathrm{Sub}((a)_1, b, c)\rangle$ | $a = \langle (a)_0, (a)_1\rangle$ |
| $\langle (a)_0, \mathrm{Sub}((a)_1, b, c), \mathrm{Sub}((a)_2, b, c)\rangle$ | $a = \langle (a)_0, (a)_1, (a)_2\rangle \wedge$ <br> $(a)_0 \neq sn(\exists) \wedge (a)_0 \neq sn(\forall)$ |
| $\langle (a)_0, (a)_1, \mathrm{Sub}((a)_2, b, c)\rangle$ | $(a = \langle sn(\exists), (a)_1, (a)_2\rangle \vee$ <br> $a = \langle sn(\exists), (a)_1, (a)_2\rangle) \wedge (a)_1 \neq b$ |
| $a$ | w p.p. |

Dla termów $t$, $t_1$, $t_2$, zmiennej $x$ oraz formuły $\psi$ mamy:
$\mathrm{Sub}(\ulcorner t_1 \urcorner, \ulcorner x \urcorner, \ulcorner t_2 \urcorner) = \ulcorner t_1(x/t_2) \urcorner$ (podstawienie $t_2$ za $x$ w $t_1$)
$\mathrm{Sub}(\ulcorner \psi \urcorner, \ulcorner x \urcorner, \ulcorner t \urcorner) = \ulcorner \psi(x/t) \urcorner$ (podstawienie $t$ za $x$ w $\psi$).

# Kodowanie operacji syntaktycznych

Formuła $\mathrm{Fr}(a, b)$ jest równoważna:

| formule | jeśli |
|---|---|
| $a = b$ | $\mathrm{Vble}(a)$ |
| $\mathrm{Fr}((a)_1, b)$ | $a = \langle (a)_0, a_1 \rangle$ |
| $\mathrm{Fr}((a)_1, b) \vee \mathrm{Fr}((a)_2, b)$ | $a = \langle (a)_0, a_1, a_2 \rangle \wedge$ $(a)_0 \neq sn(\exists) \wedge (a)_0 \neq sn(\forall)$ |
| $\mathrm{Fr}((a)_2, b) \wedge (a)_1 \neq b$ | w p.p. |

Dla formuły $\psi$ oraz zmiennej $x$ zachodzi relacja $\mathrm{Fr}(\ulcorner \psi \urcorner, \ulcorner x \urcorner)$ dokładnie wtedy, gdy $x$ jest zmienną wolną w $\psi$.

# Kodowanie operacji syntaktycznych

Formuła $\mathrm{Subtl}(a, b, c)$ jest równoważna:

| formule | jeśli |
|---|---|
| $\mathrm{Subtl}((a)_1, b, c)$ | $a = \langle (a)_0, (a)_1 \rangle$ |
| $\mathrm{Subtl}((a)_1, b, c) \wedge \mathrm{Subtl}((a)_2, b, c)$ | $a = \langle (a)_0, (a)_1, (a)_2 \rangle \wedge$ $(a)_0 \neq sn(\exists) \wedge (a)_0 \neq sn(\forall)$ |
| $\mathrm{Subtl}((a)_2, b, c) \wedge$ $\wedge (\neg \mathrm{Fr}((a)_2, b) \vee \neg \mathrm{Fr}(c, (a)_1))$ | $(a = \langle sn(\exists), (a)_1, (a)_2 \rangle \vee$ $a = \langle sn(\forall), (a)_1, (a)_2 \rangle) \wedge$ $(a)_1 \neq b$ |
| $0 = 0$ | w p.p. |

$\mathrm{Subtl}(\ulcorner \psi \urcorner, \ulcorner x \urcorner, \ulcorner t \urcorner)$ zachodzi dokładnie wtedy, gdy term $t$ jest podstawialny za zmienną $x$ do formuły $\psi$.

# Kodowanie aksjomatów logicznych

Formuła LogAx($a$) jest alternatywą 14 warunków:

- $\exists x < a \exists y < a(\mathrm{Form}(x) \wedge \mathrm{Form}(y) \wedge a = \langle 9, x, \langle 9, y, x\rangle\rangle)$
- $\exists x < a \exists y < a \exists z < a(\mathrm{Form}(x) \wedge \mathrm{Form}(y) \wedge \mathrm{Form}(z) \wedge$
  $a = \langle 9, \langle 9, x, \langle 9, y, z\rangle\rangle, \langle 9, \langle 9, x, y\rangle, \langle 9, x, z\rangle\rangle\rangle)$
- $\exists x < a \exists y < a(\mathrm{Form}(x) \wedge \mathrm{Form}(y) \wedge a = \langle 9, \langle 9, x, y\rangle, \langle 9, \langle 3, y\rangle, \langle 3, x\rangle\rangle\rangle)$
- $\exists x < a(\mathrm{Form}(x) \wedge a = \langle 9, \langle 3, \langle 3, x\rangle\rangle, x\rangle)$
- $\exists x < a(\mathrm{Form}(x) \wedge a = \langle 9, x, \langle 3, \langle 3, x\rangle\rangle\rangle)$
- $\exists x < a \exists y < a(\mathrm{Form}(x) \wedge \mathrm{Form}(y) \wedge a = \langle 9, \langle 7, x, y\rangle, x\rangle)$
- $\exists x < a \exists y < a(\mathrm{Form}(x) \wedge \mathrm{Form}(y) \wedge a = \langle 9, \langle 7, x, y\rangle, y\rangle)$

# Kodowanie aksjomatów logicznych

- $\exists x < a \exists y < a(\mathrm{Form}(x) \wedge \mathrm{Form}(y) \wedge \mathrm{Form}(z) \wedge$
  $a = \langle 9, \langle 9, x, y\rangle, \langle 9, \langle 9, x, z\rangle, \langle 9, x, \langle 7, y, z\rangle\rangle\rangle\rangle)$
- $\exists x < a \exists y < a(\mathrm{Form}(x) \wedge \mathrm{Form}(y) \wedge a = \langle 9, x, \langle 5, x, y\rangle\rangle)$
- $\exists x < a \exists y < a(\mathrm{Form}(x) \wedge \mathrm{Form}(y) \wedge a = \langle 9, y, \langle 5, x, y\rangle\rangle)$
- $\exists x < a \exists y < a(\mathrm{Form}(x) \wedge \mathrm{Form}(y) \wedge \mathrm{Form}(z) \wedge$
  $a = \langle 9, \langle 9, x, z\rangle, \langle 9, \langle 9, y, z\rangle, \langle 9, \langle 5, x, y\rangle, z\rangle\rangle\rangle)$
- $\exists x < a \exists y < a(\mathrm{Form}(x) \wedge \mathrm{Form}(y) \wedge a = \langle 9, \langle 11, x, y\rangle, \langle 9, x, y\rangle\rangle)$
- $\exists x < a \exists y < a(\mathrm{Form}(x) \wedge \mathrm{Form}(y) \wedge a = \langle 9, \langle 11, x, y\rangle, \langle 9, y, x\rangle\rangle)$
- $\exists x < a \exists y < a(\mathrm{Form}(x) \wedge \mathrm{Form}(y) \wedge a = \langle 9, \langle 9, x, y\rangle, \langle 9, \langle 9, y, x\rangle, \langle 11, x, y\rangle\rangle\rangle)$

LogAx($a$) zachodzi dokładnie wtedy, gdy $a$ jest numerem gödlowskim aksjomatu logicznego PA.

# Kodowanie aksjomatów identyczności

Formuła EqAx($a$) jest alternatywą 8 warunków:

- $\exists x < a\ (a = \langle 25, 2x, 2x\rangle)$
- $\exists x < a\ \exists y < a\ (a = \langle 9, \langle 25, 2x, 2y\rangle, \langle 25, 2y, 2x\rangle\rangle)$
- $\exists x < a\ \exists y < a\ \exists z < a\ (a = \langle 9, \langle 7, \langle 25, 2x, 2y\rangle, \langle 25, 2y, 2z\rangle, \langle 25, 2x, 2z\rangle\rangle)$
- $\exists x < a\ \exists y < a\ \exists z < a\ (a = \langle 9, \langle 25, 2x, 2y\rangle, \langle 25, \langle 17, 2x\rangle, \langle 17, 2y\rangle\rangle\rangle)$
- $\exists x < a\ \exists y < a\ \exists z < a\ (a = \langle 9, \langle 25, 2x, 2y\rangle, \langle 25, \langle 19, 2x, 2z\rangle, \langle 19, 2y, 2z\rangle\rangle\rangle)$
- $\exists x < a\ \exists y < a\ \exists z < a\ (a = \langle 9, \langle 25, 2x, 2y\rangle, \langle 25, \langle 19, 2z, 2x\rangle, \langle 19, 2z, 2y\rangle\rangle\rangle)$
- $\exists x < a\ \exists y < a\ \exists z < a\ (a = \langle 9, \langle 25, 2x, 2y\rangle, \langle 25, \langle 21, 2x, 2z\rangle, \langle 21, 2y, 2z\rangle\rangle\rangle)$
- $\exists x < a\ \exists y < a\ \exists z < a\ (a = \langle 9, \langle 25, \langle 25, 2x, 2y\rangle, \langle 25, \langle 21, 2z, 2x\rangle, \langle 21, 2z, 2v\rangle\rangle\rangle)$

EqAx($a$) zachodzi dokładnie wtedy, gdy $a$ jest numerem gödlowskim jednego z aksjomatów identyczności PA.

# Kodowanie aksjomatów pozalogicznych

Formuła NLAx($a$) jest alternatywą następujących warunków:

- $\exists x < a\, \exists y < a\, ((\mathrm{Vble}(x) \wedge \mathrm{Vble}(y)) \wedge a = \langle 9, \langle 25, \langle 17, x\rangle, \langle 17, y\rangle\rangle, \langle 25, x, y\rangle\rangle)$
- $\exists x < a\, (\mathrm{Vble}(x) \wedge a = \langle 3, \langle 25, 23, \langle 17, x\rangle\rangle\rangle)$
- $\exists x < a\, (\mathrm{Vble}(x) \wedge a = \langle 25, \langle 19, x, 23\rangle, x\rangle)$
- $\exists x < a\, \exists y < y\, ((\mathrm{Vble}(x) \wedge \mathrm{Vble}(y)) \wedge a = \langle 25, \langle 19, x, \langle 17, y\rangle\rangle, \langle 17, \langle 19, x, y\rangle\rangle\rangle)$
- $\exists x < a\, (\mathrm{Vble}(x) \wedge a = \langle 25, \langle 21, x, 23\rangle, 23\rangle)$
- $\exists x < a\, \exists y < a\, ((\mathrm{Vble}(x) \wedge \mathrm{Vble}(y)) \wedge a = \langle 25, \langle 21, x, \langle 17, y\rangle\rangle, \langle 19, \langle 21, x, y\rangle, x\rangle\rangle)$
- $\exists x < a\, \exists y < a\, ((\mathrm{Form}(x) \wedge \mathrm{Vble}(y) \wedge \mathrm{Fr}(x, y)) \wedge$
  $a = \langle 9, \langle 7, \mathrm{Sub}(x, y, 23), \langle 15, y, \langle 9, x, \mathrm{Sub}(x, y, \langle 17, y\rangle)\rangle\rangle\rangle, \langle 15, y, x\rangle\rangle)$.

NLAx($a$) zachodzi dokładnie wtedy, gdy $a$ jest numerem gödlowskim aksjomatu pozalogicznego PA.
Zauważmy, że ostatni z powyższych warunków odpowiada nieskończonemu zbiorowi aksjomatów.

# Kodowanie reguł wnioskowania

Relacja sub$(a, b)$ zachodzi dokładnie wtedy, gdy istnieją $x < b$ oraz $y < b$ takie, że:

- Term$(x)$
- Vble$(y)$
- Form$(a)$
- Form$(b)$
- Subtl$(a, y, x)$
- $b = \mathrm{Sub}(a, y, x)$.

sub$(\ulcorner\varphi\urcorner, \ulcorner\psi\urcorner)$ zachodzi dokładnie wtedy, gdy formuła $\psi$ powstaje z formuły $\varphi$ przez podstawienie w $\varphi$ pewnego termu za pewną zmienną.

# Kodowanie reguł wnioskowania

Relacja $\mathrm{MP}(a, b, c)$ zachodzi, gdy:

- $\mathrm{Form}(a)$
- $\mathrm{Form}(b)$
- $\mathrm{Form}(c)$
- $a = \langle 9, (a)_1, (a)_2 \rangle$
- $b = (a)_1$
- $c = (a)_2$.

$\mathrm{MP}(a, b, c)$ zachodzi dokładnie wtedy, gdy formuła o numerze gödlowskim $c$ jest wnioskiem reguły odrywania, gdzie przesłankami są formuły o numerach gödlowskich $a$ oraz $b$.

# Kodowanie reguł wnioskowania

Relacja GR$(a, b)$ zachodzi, gdy istnieje $x < b$ taka, że:

- Form$(a)$
- Form$(b)$
- Vble$(x)$
- $(b)_0 = sn(\forall)$
- $(b)_1 = x$
- $(b_2) = a$.

GR$(a, b)$ zachodzi dokładnie wtedy, gdy gdy formuła o numerze gödlowskim $b$ jest wnioskiem reguły generalizacji, gdzie przesłanką jest formuła o numerze gödlowskim $a$.

# Kodowanie reguł wnioskowania

Relacja $\mathrm{OA}(a, b)$ zachodzi, gdy istnieją $x < a$, $y < a$ oraz $z < b$ takie, że:

- $\mathrm{Form}(x)$
- $\mathrm{Form}(y)$
- $\mathrm{Vble}(z)$
- $a = \langle 9, x, \langle 13, z, y \rangle \rangle$
- $b = \langle 9, x, y \rangle$.

Relacja $\mathrm{OA}(a, b)$ zachodzi dokładnie wtedy, gdy $b$ jest numerem gödlowskim wniosku reguły opuszczania kwantyfikatora generalnego, gdzie przesłanka ma numer gödlowski $a$.

# Kodowanie reguł wnioskowania

Relacja $\mathrm{DA}(a, b)$ zachodzi, gdy istnieją $x < a$, $y < a$ oraz $z < b$ takie, że:

- $\mathrm{Form}(x)$
- $\mathrm{Form}(y)$
- $\mathrm{Vble}(z)$
- $a = \langle 9, x, y\rangle$
- $b = \langle 9, x, \langle 13, z, y\rangle\rangle$
- $\neg\mathrm{Fr}(x, z)$.

Relacja $\mathrm{DA}(a, b)$ zachodzi dokładnie wtedy, gdy $b$ jest numerem gödlowskim wniosku reguły dołączania kwantyfikatora generalnego, gdzie przesłanka ma numer gödlowski $a$.

# Kodowanie reguł wnioskowania

Relacja $\mathrm{OE}(a, b)$ zachodzi, gdy istnieją $x < a$, $y < a$ oraz $z < b$ takie, że:

- $\mathrm{Form}(x)$
- $\mathrm{Form}(y)$
- $\mathrm{Vble}(z)$
- $a = \langle 9, \langle 13, z, x \rangle \rangle$
- $b = \langle 9, x, y \rangle$.

Relacja $\mathrm{OE}(a, b)$ zachodzi dokładnie wtedy, gdy $b$ jest numerem gödlowskim wniosku reguły opuszczania kwantyfikatora egzystencjalnego, gdzie przesłanka ma numer gödlowski $a$.

# Kodowanie reguł wnioskowania

Relacja DE$(a, b)$ zachodzi, gdy istnieją $x < a$, $y < a$ oraz $z < b$ takie, że:

- Form$(x)$
- Form$(y)$
- Vble$(z)$
- $a = \langle 9, x, y \rangle$
- $b = \langle 9, \langle 13, z, x \rangle \rangle$
- $\neg$Fr$(y, z)$.

Relacja DE$(a, b)$ zachodzi dokładnie wtedy, gdy $b$ jest numerem gödlowskim wniosku reguły dołączania kwantyfikatora egzystencjalnego, gdzie przesłanka ma numer gödlowski $a$.

# Kodowanie dowodów

$\mathrm{Ax}(a) \equiv \mathrm{LogAx}(a) \vee \mathrm{EqAx}(a) \vee \mathrm{NLAx}(a)$. Relacja ta zachodzi dokładnie dla liczb będących numerami gödlowskimi aksjomatów PA.

Dowody są ***ciągami*** formuł. Nadto, jak pamiętamy, każdy dowód jest też ***drzewem***: w korzeniu znajduje się dowodzona formuła, w liściach aksjomaty, a bezpośrednie poprzedniki każdego wierzchołka są przesłankami reguły wnioskowania, której wnioskiem jest właśnie ów wierzchołek.

Ciągi numerów formuł możemy kodować na tej samej zasadzie, na jakiej kodujemy ciągi symboli. Jeśli $\Delta = (\psi_1, \psi_2, \ldots, \psi_n)$ jest ciągiem formuł, to przez ***numer gödlowski*** ciągu $\Delta$ rozumiemy (wyznaczoną jednoznacznie i efektywnie) liczbę:

$$\langle \ulcorner \psi_1 \urcorner, \ulcorner \psi_2 \urcorner, \ldots, , \ulcorner \psi_n \urcorner \rangle.$$

# Kodowanie dowodów

Relację $\mathrm{Dow}(a, b)$ definiujemy przez koniunkcję warunków:

$seq(a) \wedge lh(a) \neq 0 \wedge (a)_{lh(a)\dot{-}1} = b \wedge \forall i < lh(a)\ (\mathrm{Form}((a)_i) \wedge$
$\exists j < i\ \exists k < i\ (\mathrm{sub}((a)_j, (a)_i) \vee \mathrm{MP}((a)_j, (a)_k, (a)_i) \vee \mathrm{GR}((a)_j, (a)_i) \vee$
$\mathrm{OA}((a)_j, (a)_i) \vee \mathrm{DA}((a)_j, (a)_i) \vee \mathrm{OE}((a)_j, (a)_i) \vee \mathrm{DE}((a)_j, (a)_i)))$.

Relacja $\mathrm{Dow}(a, b)$ zachodzi dokładnie wtedy, gdy $a$ jest numerem gödlowskim dowodu formuły o numerze gödlowskim $b$.

Jak bowiem pamiętamy, dowód formuły $\psi$ to ciąg formuł o ostatnim elemencie $\psi$ taki, że każdy z elementów tego ciągu jest bądź aksjomatem, bądź wnioskiem którejś z reguł wnioskowania, której przesłankami są wcześniejsze (od tego wniosku) elementy owego ciągu.

# Rekurencyjność kodowań

**Twierdzenie.**
*Wszystkie zdefiniowane powyżej funkcje i relacje są rekurencyjne.*

- Dowód tego twierdzenia wynika bezpośrednio z podanych wyraźnych rekurencyjnych definicji.
- Odwołujemy się przy tym także do faktów wspomnianych wyżej, przy kodowaniu termów.

# Twierdzenia PA

Wszystkie dotąd rozważane funkcje i relacje odpowiadające pojęciom metalogicznym okazały się rekurencyjne. A co z relacją $PA \vdash \psi$? Gdy zachodzi $PA \vdash \psi$, to $\psi$ jest twierdzeniem PA. Okazuje się, że relacja ta (dokładniej: relacja między numerami dowodów a numerami formuł) nie jest rekurencyjna, jest jedynie rekurencyjnie przeliczalna. Mamy bowiem: $PA \vdash \psi$ dokładnie wtedy, gdy ***istnieje*** dowód $\psi$ w PA. W definicji pojęcia „być twierdzeniem PA" występuje zatem jeden nieograniczony kwantyfikator egzystencjalny.

- Definiujemy: $\mathrm{Tw}(a) \equiv \exists x\, \mathrm{Dow}(x, a)$.
- Relacja Tw jest rekurencyjnie przeliczalna.

# Klasyfikacja złożoności pojęć

Jak wiemy, operacje używające kwantyfikatorów ograniczonych prowadzą od relacji rekurencyjnych do relacji rekurencyjnych. Kwantyfikatory nieograniczone już nie mają tej własności – istotnie zwiększają stopień skomplikowania pojęć. Można dokonać logicznej klasyfikacji pojęć uwzględniającej liczbę kwantyfikatorów nieograniczonych potrzebnych w ich definicjach.

Szczególnie istotne są dwie hierarchie, nazywane:

- ***hierarchią arytmetyczną*** (kwantyfikujemy tylko zmienne indywidualne);
- ***hierarchią analityczną*** (kwantyfikujemy zmienne przebiegające podzbiory uniwersum).

# Hierarchia arytmetyczna

**Definicja Hierarchii Arytmetycznej.**

- $\sum_0^0 = \prod_0^0 =$ zbiór relacji rekurencyjnych;
- Relacja $R \subseteq \omega^k$ jest klasy $\sum_{n+1}^0$ wtedy i tylko wtedy, gdy istnieje relacja $Q \subseteq \omega^{k+1}$ klasy $\prod_n^0$ taka, że
  $R(a_1, \ldots, a_k) \equiv \exists x\ Q(a_1, \ldots, a_k, x)$.
- Relacja $R \subseteq \omega^k$ jest klasy $\prod_{n+1}^0$ wtedy i tylko wtedy, gdy istnieje relacja $Q \subseteq \omega^{k+1}$ klasy $\sum_n^0$ taka, że
  $R(a_1, \ldots, a_k) \equiv \forall x\ Q(a_1, \ldots, a_k, x)$.

- Relacje klasy $\sum_1^0$ to dokładnie relacje rekurencyjnie przeliczalne.
- Relacja $R$ jest rekurencyjna wtedy i tylko wtedy, gdy $R$ oraz $\neg R$ są rekurencyjnie przeliczalne.

# Hierarchia arytmetyczna

- Jeżeli relacja $R$ jest klasy $\sum_n^0$ (odpowiednio, $\prod_n^0$) zaś $f_1, \ldots, f_k$ są funkcjami rekurencyjnymi, to relacja $P$ określona wzorem:

  $$P(\vec{a}) \equiv R(f_1(\vec{a}), \ldots, f_k(\vec{a}))$$

  jest również klasy $\sum_n^0$ (odpowiednio, $\prod_n^0$).
- Każda klasa hierarchii arytmetycznej jest zamknięta ze względu na koniunkcję i alternatywę.

Tu (i dalej) $\vec{a}$ oznacza ciąg argumentów o takiej długości, ile argumentów ma rozważana relacja lub funkcja.

Dla dowolnego zbioru $X$ relacji przez zbiór ***uzupełnień*** relacji z $X$ rozumiemy zbiór $\mathcal{C}X$ zdefiniowany następująco: $R \in \mathcal{C}X$ wtedy i tylko wtedy, gdy $\forall \vec{a}\, (R(\vec{a}) \equiv \neg P(\vec{a}))$ dla pewnej relacji $P \in X$.

# Hierarchia arytmetyczna

- Klasa $\sum_n^0$ jest identyczna z klasą uzupełnień relacji z klasy $\prod_n^0$ i *vice versa*.
- Operacja kwantyfikatora ogólnego nie wyprowadza poza klasę $\prod_n^0$ (dla $n > 0$).
- Operacja kwantyfikatora egzystencjalnego nie wyprowadza poza klasę $\sum_n^0$ (dla $n > 0$).

Prawdziwe są następujące (właściwe) inkluzje:

- $\prod_n^0 \subset \sum_{n+1}^0$,
- $\sum_n^0 \subset \prod_{n+1}^0$,
- $\prod_n^0 \subset \prod_{n+1}^0$,
- $\sum_n^0 \subset \sum_{n+1}^0$.

# Hierarchia arytmetyczna

- Dla każdej klasy $\sum_n^0$ (odpowiednio, $\prod_n^0$) ($n > 0$) istnieje w $\sum_n^0$ (odpowiednio, $\prod_n^0$) relacja uniwersalna dla wszystkich relacji tej klasy.
- Dla każdego $n > 0$: $\prod_n^0 \neq \sum_n^0$.
- Dla każdego $n$: $\sum_n^0 \neq \sum_{n+1}^0$ oraz $\prod_n^0 \neq \prod_{n+1}^0$.

- Dla $n > 0$ relacja uniwersalna dla klasy $\sum_n^0$ należy do $\sum_n^0$, ale nie należy ani do $\prod_n^0$ ani do $\sum_{n-1}^0$.
- Dla $n > 0$ relacja uniwersalna dla klasy $\prod_n^0$ należy do $\prod_n^0$, ale nie należy ani do $\sum_n^0$ ani do $\prod_{n-1}^0$.
- Jeżeli relacja uniwersalna dla relacji klasy $X$ sama należy do $X$, to $\mathcal{C}X \neq X$.

# Hierarchia arytmetyczna

**Przykład**. Pojęcie granicy ciągu jest pojęciem klasy $\prod_3^0$ (i nie jest pojęciem ani klasy $\sum_3^0$ ani $\sum_2^0$ ani $\prod_2^0$:

$$a = \lim a_n \equiv \forall k \exists m \forall n \, (n > m \rightarrow |a_n - a| < \frac{1}{k+1}).$$

**Przykład**. Jak widzieliśmy, zbiór twierdzeń Arytmetyki Peana jest klasy $\sum_1^0$, czyli jest rekurencyjnie przeliczalny (ale **nie** jest rekurencyjny!).

**Przykład**. Pojęcie ***prawdy*** nie może zostać scharakteryzowane na żadnym piętrze hierarchii arytmetycznej.

**Przykład**. Również definicja pojęcia ***dobrego porządku*** wykracza poza hierarchię arytmetyczną.

# Inne funkcje kodujące

- Zamiast funkcji $\beta$ Gödla można używać innych funkcji rekurencyjnych w arytmetyzacji składni. Przy tym, stosowane kodowanie może uwzględniać budowę składniową wyrażeń, bądź kodować po prostu ciągi symboli.
- W wielu podręcznikach używa się kodowania wykorzystującego rozkład liczb na czynniki pierwsze.

- Dla wyrażenia (termu lub formuły) $u$ o postaci: $vv_1v_2\ldots v_n$ jego numer gödlowski $gn(u)$ określamy indukcyjnie: $gn(u) = 2^{sn(v)} \cdot 3^{gn(v_1)} \cdot 5^{gn(v_2)} \cdot \ldots \cdot p_n^{gn(v_n)}$.
- Tu $p_n$ jest $n$-tą liczbą pierwszą ($p_0 = 2$, $p_1 = 3$, $p_2 = 5$, itd.)

# Inne funkcje kodujące

Dla rozważanej wcześniej formuły $\forall x_1\ (x_1 \doteq \underline{0} \rightarrow (x_1 \underline{\times} \underline{s}(x_1)) \doteq \underline{0})$ mamy (przy ustalonej uprzednio funkcji $sn$):

- $gn(\underline{s}(x_1)) = 2^{17} \cdot 3^2$
- $gn(x_1 \underline{\times} \underline{s}(x_1)) = 2^{21} \cdot 3^2 \cdot 5^{2^{17} \cdot 3^2}$
- $gn(x_1 \underline{\times} \underline{s}(x_1)) \doteq \underline{0}) = 2^{25} \cdot 3^{2^{21} \cdot 3^2 \cdot 5^{2^{17} \cdot 3^2}} \cdot 5^{23}$
- $gn(x_1 \doteq \underline{0}) = 2^{25} \cdot 3^2 \cdot 5^{23}$
- $gn(x_1 \doteq \underline{0} \rightarrow (x_1 \underline{\times} \underline{s}(x_1)) \doteq \underline{0}) = 2^9 \cdot 3^{gn(x_1 \doteq \underline{0})} \cdot 5^{gn(x_1 \underline{\times} \underline{s}(x_1)) \doteq \underline{0})}$
- $gn(\forall x_1\ (x_1 \doteq \underline{0} \rightarrow (x_1 \underline{\times} \underline{s}(x_1)) \doteq \underline{0})) = 2^{15} \cdot 3^2 \cdot 5^{gn(x_1 \doteq \underline{0} \rightarrow (x_1 \underline{\times} \underline{s}(x_1)) \doteq \underline{0})}$.

Kodowanie jest jednoznaczne. Dla dowolnej liczby $a$ można ustalić czy jest ona kodem jakiegoś wyrażenia.

# Inne funkcje kodujące

- Można też kodować wyrażenia po prostu jako ciągi symboli. Niech funkcja $\sigma$ numerująca symbole alfabetu przyjmuje wartości dodatnie. Jeśli $s_1 s_2 \ldots s_n$ jest ciągiem symboli alfabetu, to za kod tego ciągu można wziąć liczbę $2^{\sigma(s_1)} \cdot 3^{\sigma(s_2)} \cdot \ldots \cdot p_{n-1}^{\sigma(s_n)}$.
- Można kodować wyrażenia ustalając np., że symbole alfabetu kodujemy liczbami zaczynającymi się (w zapisie dziesiętnym) od cyfry 8, po której występuje pewna liczba cyfr 1 (inna dla każdego symbolu alfabetu).
- Można kodować wyrażenia używając funkcji Cantora lub jakiejkolwiek innej (rekurencyjnej) funkcji kodującej ciągi skończone.
- Wybór funkcji kodującej jest więc sprawą konwencji. Kodowanie musi być jedynie jednoznaczne i rekurencyjne. W rozważaniach metateoretycznych nigdy nie obliczamy wartości funkcji kodujących wyrażenia.

# Wykorzystywana literatura


- Cori, R., Lascar, D. 2001. *Mathematical Logic. A Course with Exercises.* Oxford University Press, Oxford.
- Hinman, P.G. 2005. *Fundamentals of Mathematical Logic.* A K Peters, Wellesley.
- Kleene, S.C. 1952. *Introduction to Metamathematics.* Wolters-Noordhoff Publishing — Groningen, North-Holland Publishing Company — Amsterdam Oxford, American-Elsevier Publishing Company, Inc. — New York.
- Ławrow, I.A., Maksimowa, L.L. 2004. *Zadania z teorii mnogości, logiki matematycznej i teorii algorytmów.* Wydawnictwo Naukowe PWN, Warszawa.
- Murawski, R. 2000$^3$. *Funkcje rekurencyjne i elementy metamatematyki. Problemy zupełności, rozstrzygalności, twierdzenia Gödla.* Wydawnictwo Naukowe UAM, Poznań.