Matematyczne podstawy informatyki
Część 1: Funkcje rekurencyjne

Antoni Kościelski

23 stycznia 2015

Spis treści

1 Funkcje pierwotnie rekurencyjne

Większość funkcji naturalnych, z którymi mamy do czynienia (także w informatyce), to funkcje pierwotnie rekurencyjne. Nawet wielu matematyków nie potrafi podać przykładów funkcji naturalnych, które nie są pierwotnie rekurencyjne, na ogół nie mają z nimi do czynienia. Teoria takich funkcji bardzo przypomina teorię funkcji rekurencyjnych, a podstawowe własności tych dwóch rodzajów funkcji są analogiczne. Z informatycznego punktu widzenia nieco ważniejsze są funkcje rekurencyjne, ale pierwotnie rekurencyjne też mają ważną interpretację: są to funkcje obliczane za pomocą programów, w których pętle są wykonywane najwyżej z góry zadaną (pierwotnie rekurencyjną) liczbę razy.

1.1 Wstępne pojęcia i oznaczenia

Pewne wprowadzane teraz pojęcia będą potrzebne dopiero wtedy, gdy będzie mowa o funkcjach rekurencyjnych.

Będziemy rozważać funkcje (także wieloargumentowe), które są określone dla argumentów będących liczbami naturalnymi, niekoniecznie wszystkich, i przyjmujące wartości naturalne. Jeżeli f jest taką funkcją, to będziemy pisać, że $f : N^k \to N$. Ten zapis ma więc znaczenie inne niż zwykle. Tak rozumiane funkcje nazywa się częściowymi. Funkcja określona dla wszystkich układów $\vec{x} \in N^k$ nazywa się całkowitą. Najczęściej dopiero z kontekstu będzie wynikać, czy mówimy o funkcjach całkowitych, czy częściowych.

Przyjmijmy, że

$$I_{n,k}(x_1, \ldots, x_k) = x_i \ \text{ oraz } \ S(x) = x + 1.$$

Funkcją charakterystyczną zbioru $R \subseteq N^k$ nazywamy całkowitą funkcję $ch_R : N^k \to N$ zdefiniowaną wzorem

$$ch_R(\vec{x}) = \begin{cases} 0 & \text{jeżeli } \vec{x} \in R, \\ 1 & \text{w przeciwnym razie.} \end{cases}$$

Zamiast różnicą posługujemy sie raczej ograniczoną różnicą zdefiniowaną wzorem

$$x - y = \max\{0, x - y\}$$

Obie różnice powinny być oznaczane różnymi symbolami, ale nie będą.

1.2 Składanie i podstawianie

Będziemy też rozważać kilka sposobów definiowania nowych funkcji. Jeżeli mamy funkcję $f : N^k \to N$ i k funkcji $g_i : N^n \to N$, to możemy zdefiniować funkcję h przyjmując, że

$$h(x_1, \ldots, x_n) = f(g_1(x_1, \ldots, x_n), \ldots, g_k(x_1, \ldots, x_n)).$$

Ten sposób definiowania nazywamy składaniem funkcji.

Składanie funkcji jest znanym i jasnym pojęciem, zwłaszcza wtedy, gdy dotyczy funkcji całkowitych. Dla funkcji częściowych definicja ta powinna zostać nieco uzupełniona. Zamiast podawać precyzyjną definicję, złożenie opiszemy intuicyjnie, jako funkcję, która jest obliczana za pomocą pewnego programu. Wyżej zdefiniowane h jest obliczane za pomocą programu

function złożenie$(f, g_1, \ldots, g_k, x_1, \ldots, x_n)$;

begin

$y_1 := g_1(x_1, \ldots, x_n)$;

$\ldots$

$y_k := g_k(x_1, \ldots, x_n)$;

złożenie $:= f(y_1, \ldots, y_k)$

end;

Wynika stąd, że złożenie h jest funkcją, która jest określona dokładnie dla tych argumentów, dla których ten program zakończy pracę, a jego wartością jest wtedy wynik obliczeń, czyli wartość funkcji *złożenie* w sensie informatycznym.

Oprócz składania rozważamy też podstawianie. Jest to operacja trudna do precyzyjnego zdefiniowania. Przykładem funkcji definiowanej przez podstawianie jest

$$h(x_1, x_2, x_3, y, z) = f(z, g(x_2, x_3), x_2, g'(x_2, z, x_1)).$$

Funkcję tę można zdefiniować za pomocą składania w następujący sposób:

$$h(x_1, x_2, x_3, y, z) =$$

$$= f(I_{5,5}(x_1, x_2, x_3, y, z), g_2(x_1, x_2, x_3, y, z), I_{5,2}(x_1, x_2, x_3, y, z), g_4(x_1, x_2, x_3, y, z)).$$

Jest to złożenie funkcji f, $I_{5,5}$, $I_{5,2}$ oraz

$$g_2(x_1, x_2, x_3, y, z) = g(I_{5,2}(x_1, x_2, x_3, y, z), I_{5,3}(x_1, x_2, x_3, y, z)),$$

$$g_4(x_1, x_2, x_3, y, z) = g'(I_{5,2}(x_1, x_2, x_3, y, z), I_{5,5}(x_1, x_2, x_3, y, z), I_{5,1}(x_1, x_2, x_3, y, z)).$$

Nietrudno zauważyć, że funkcje g_2 i g_4, a także funkcja h są zdefiniowane jako złożenia. Jeżeli możemy posługiwać się funkcjami $I_{n,k}$ i potrafimy składać funkcje, to możemy także definiować funkcje za pomocą podstawiania.

1.3 Rekursja prosta

Mając dane funkcje $f : N^{k+2} \to N$ i $g : N^k \to N$, możemy zdefiniować funkcję $h : N^{k+1} \to N$ przyjmując, że spełnia ona następujące równości:

$$h(x_1, \ldots, x_k, 0) = g(x_1, \ldots, x_k)$$

oraz

$$h(x_1, \ldots, x_k, y + 1) = f(h(x_1, \ldots, x_k, y), y, x_1, \ldots, x_k).$$

O tak zdefiniowanej funkcji h mówimy, że została zdefiniowana za pomocą rekursji prostej. Schemat rekursji prostej stosujemy również dla $k = 0$. Wtedy przyjmuje on postać

$$h(0) = c \ \text{ oraz } \ h(y + 1) = f(h(y), y),$$

gdzie c jest pewną liczbą naturalną.

W przypadku funkcji częściowych, funkcja h definiowana przez rekursję prostą powinna być obliczana przez następujący algorytm:

function rekursja$(f, g, x_1 \ldots, x_k, y)$;

begin

$z := g(x_1, \ldots, x_k)$;

for $i := 0$ to $y - 1$ do $z := f(z, i, x_1, \ldots, x_k)$;

rekursja $:= z$

end;

1.4 Operacja minimum

W końcu, dla funkcji $f : N^{k+1} \to N$ możemy zdefiniować funkcję

$$h(x_1, \dots, x_k) = \mu\, y(f(x_1, \dots, x_k, y) = 0)$$

przyjmującą dla argumentów $x_1, \dots, x_k$ jako wartość najmniejszą liczbę y taką, że $f(x_1, \dots, x_k, y) = 0$. O tej funkcji h mówimy, że została zdefiniowana za pomocą operacji minimum. Nie musi to być funkcja całkowita, nawet dla całkowitej funkcji f.

Funkcja h definiowana za pomocą operacji minimum powinna być obliczana przez następujący algorytm:

function minimum$(f, x_1 \dots, x_k)$;

begin

$y := 0$;

while $f(x_1, \dots, x_k, y) \neq 0$ do $y := y + 1$;

minimum $:= y$

end;

Operację minimum będziemy stosować w kilku szczególnych przypadkach. I tak dla relacji $R \subseteq N^{k+1}$ przyjmujemy

$$\mu y\; R(x_1, \dots, x_k, y) = \mu y\; (ch_R(x_1, \dots, x_k, y) = 0),$$

$$\mu y < z\; R(x_1, \dots, x_k, y) = \mu y\; (R(x_1, \dots, x_k, y) \vee y = z),$$

$$\mu y \leqslant z\; R(x_1, \dots, x_k, y) = \mu y\; (R(x_1, \dots, x_k, y) \vee y = z + 1).$$

Operacje minimum w dwóch ostatnich linijkach nazywamy ograniczonymi.

Będziemy zajmować się też efektywną operację minimum, a więc taką, która jest stosowana pod pewnymi warunkami, tylko do całkowitych funkcji f, które dodatkowo dla każdych $x_1, \dots, x_k$ przyjmują dla pewnego y wartość $f(x_1, \dots, x_k, y) = 0$.

1.5 Funkcje i zbiory pierwotnie rekurencyjne

Pierwsza[1] praca poświęcona funkcjom definiowanym przez rekursję ukazała się 1888 roku. R Dedekind rozważał w niej funkcje definiowane w sposób będący uogólnieniem definicji dodawania, mnożenia i potęgowania. Około 1919 roku Thoralf Skolem rozważał arytmetykę z funkcjami pierwotnie rekurencyjnymi. Od 1925 był znany przykład Wilhelma Ackermanna funkcji definiowanej przez rekursję, ale nie dającej się zdefiniować przez rekursję prostą. Taką jest funkcja Ackermanna[2] spełniająca równania

$$A(0, n) = n+1, \quad A(m+1, 0) = A(m, 1) \;\text{ oraz }\; A(m+1, n+1) = A(m, A(m+1, n)).$$

[1] Na podstawie książki Romana Murawskiego *Funkcje rekurencyjne i elementy metamatematyki.*

[2] Podana funkcja została w rzeczywistości wprowadzona przez Rózsę Péter i Raphaela Robinsona i jest prostszą wersją oryginalnej funkcji Ackermanna. Ta ostatnia jest definiowana (z dokładnością do kolejności argumentów) wzorami

$$\varphi(0, n, x) = n + x, \quad \varphi(1, 0, x) = 0, \quad \varphi(2, 0, x) = 1, \quad \varphi(m, 0, x) = x \;\text{ gdy } m > 2,$$

$$\varphi(m + 1, n + 1, x) = \varphi(m, \varphi(m + 1, n, x), x).$$

W szczególności mamy $\varphi(1, n, x) = n \cdot x$ oraz $\varphi(2, n, x) = x^n$.

Funkcje definiowane przez rekursję po raz pierwszy w istotny sposób wykorzystał Kurt Gödel.

Rozważał on najmniejszą klasę funkcji zawierającą następnik S, funkcje stałe oraz funkcje $I_{n,k}$ i wszystkie inne, które można zdefiniować używając wymienionych za pomocą składania i rekursji prostej. Klasa ta nazywa się klasą funkcji pierwotnie rekurencyjnych, a jej elementy to funkcje pierwotnie rekurencyjne. W pracy Gödla funkcje te były nazywane rekurencyjnymi, w tamtym czasie autor nie doceniał i nie potrzebował innych funkcji definiowanych przez rekursję.

Zbiór $X \subseteq N^k$ jest pierwotnie rekurencyjny, jeżeli jego funkcja charakterystyczna ch_X jest pierwotnie rekurencyjna.

1.6 Przykłady funkcji i zbiorów pierwotnie rekurencyjnych

Bez trudu dowodzimy, że dodawanie, mnożenie i potęgowanie są funkcjami pierwotnie rekurencyjnymi. Schematy rekursji prostej definiujące te funkcje są powszechnie znane. Wobec tego wszelkie wielomiany o współczynnikach naturalnych są pierwotnie rekurencyjne. Także silnia jest funkcją pierwotnie rekurencyjną.

Jeżeli A jest funkcją Ackermanna (patrz str. 4), a φ jest oryginalną funkcją rozważaną przez Ackermanna (patrz przypis na str. 4), to funkcje

$$A_m(n) = A(m, n)$$

oraz

$$\varphi_m(n, x) = \varphi(m, n, x)$$

są pierwotnie rekurencyjne.

Pierwotnie rekurencyjna są również funkcje $minus(m, n) = m - n$ (tzw. ograniczone odejmowanie) oraz $minus_1(m) = m - 1$ (ograniczone odejmowanie 1). Świadczą o tym następujące definicje:

$$minus_1(0) = 0 \text{ oraz } minus_1(m + 1) = I_{2,2}(minus_1(m), m)$$

$$minus(m, 0) = I_{1,1}(m) \text{ oraz } minus(m, n + 1) = minus_1(minus(m, n)).$$

Najprostszym przykładem relacji pierwotnie rekurencyjnej jest relacja mniejszości. Mamy bowiem

$$ch_<(m, n) = 1 - (n - m).$$

Także relacja większości jest pierwotnie rekurencyjna, gdyż

$$ch_>(m, n) = ch_<(n, m) = ch_<(I_{2,2}(m, n), I_{2,1}(m, n)).$$

Kolejnymi przykładami są słabe relacje porządkujące:

$$ch_\leqslant(m, n) = 1 - ch_>(m, n) \text{ oraz } ch_\geqslant(m, n) = 1 - ch_<(m, n).$$

Teraz można wykazać pierwotną rekurencyjność relacji równości i różności:

$$ch_=(m, n) = ch_\leqslant(m, n)) + ch_\geqslant(m, n) \text{ oraz } ch_\neq(m, n) = 1 - ch_=(m, n).$$

Lemat 1.1 (o podstawianiu) *Jeżeli relacja $R \subseteq N^3$ i funkcja $f : N^2 \to N$ są pierwotnie rekurencyjne, to relacja R' taka, że*

$$R'(m, n, k, l) \Leftrightarrow R(f(m, n), m, k)$$

też jest pierwotnie rekurencyjna. Liczby m, n, k, l można w tym lemacie zastąpić dowolnej długości układami liczb.

Dowód. Funkcją charakterystyczną relacji R' jest

$$ch_{R'}(m,n,k,l) = ch_R(f(m,n),m,k)$$

definiowana za pomocą podstawiania z funkcji ch_R i f. $\Box$

Kilka dalszych własności relacji pierwotnie rekurencyjnych zostało dowiedzione w rozdziale 1.8, po pewnym komentarzu metodologicznym.

Lemat 1.2 *Definicje warunkowe nie wyprowadzają poza klasę funkcji pierwotnie rekurencyjnych, a więc jeżeli funkcje $g : N^k \to N$ i $h : N^k \to N$ są pierwotnie rekurencyjne, a $R \subseteq N^k$ jest pierwotnie rekurencyjną relacją, to także funkcja $g : N^k \to N$ zdefiniowana wzorem*

$$f(\vec{m}) = \begin{cases} g(\vec{m}) & \textit{jeżeli } R(\vec{m}), \\ h(\vec{m}) & \textit{w pozostałych przypadkach} \end{cases}$$

jest pierwotnie rekurencyjna. Nietrudno też zauważyć, że analogiczną własność mają bardziej skomplikowane definicje warunkowe, z większą liczbą wykluczających się wzajemnie warunków.

Dowód. Zauważmy, że funkcję f można zdefiniować również równością

$$f(\vec{m}) = g(\vec{m}) \cdot (1 - ch_R(\vec{m})) + h(\vec{m}) \cdot ch_R(\vec{m}). \ \Box$$

Lemat 1.3 *Jeżeli funkcja $f : N^{k+1} \to N$ jest pierwotnie rekurencyjna, to funkcje*

$$g(\vec{m},n) = \sum_{i<n} f(\vec{m},i) \quad \textit{oraz}$$

$$h(\vec{m},n) = \prod_{i<n} f(\vec{m},i)$$

są pierwotnie rekurencyjne.

Dowód. Funkcję $g : N^{k+1} \to N$ definiuje się przez rekursję prostą następującymi równościami

$$g(\vec{m},0) = 0 \ \text{ oraz } \ g(\vec{m},n+1) = g(\vec{m},n) + f(\vec{m},n).$$

Funkcję h definiujemy podobnie używając iloczynu. $\Box$

Lemat 1.4 *Jeżeli $R \subseteq N^{k+1}$ jest relacją pierwotnie rekurencyjną, to funkcja*

$$f(\vec{m},n) = \mu x < n \; R(\vec{m},x),$$

definiowana za pomocą operacji minimum ograniczonego (patrz str. 4), też jest pierwotnie rekurencyjna.

Dowód. Aby dowieść ten lemat wystarczy zauważyć, że funkcja f spełnia następujące równości

$$f(\vec{m},0) = 0$$

oraz

$$f(\vec{m},n+1) = \begin{cases} f(\vec{m},n) & \text{jeżeli } f(\vec{m},n) < n, \\ n & \text{jeżeli } f(\vec{m},n) \geqslant n \text{ i } R(\vec{m},n), \\ n+1 & \text{w pozostałych przypadkach.} \ \Box \end{cases}$$

1.7 Definiowalność

Matematycy efekty swojej pracy formułują w postaci twierdzeń, a następnie dowodzą. Przykładem może być następujące

Twierdzenie 1.5 *Zbiór liczb pierwszych jest pierwotnie rekurencyjny.*

Czasem efektem pracy matematyków są jednak schematy twierdzeń. Są to stwierdzenia takiej na przykład postaci: jeżeli w dalszym ciągu zamiast φ wypowiem arytmetyczną formułę, tylko z kwantyfikatorami ograniczonymi, to otrzymam

Twierdzenie 1.6 *Zbiór* $\{n \in N : \varphi(n)\}$ *jest pierwotnie rekurencyjny.*

Matematyk wypowiadający twierdzenie nikogo nie dziwi. Może być jednak dziwne, że za jednym razem może wypowiedzieć nieskończenie wiele twierdzeń. Schematy to coś bardziej skomplikowanego od twierdzeń. I raczej nie możemy się bez nich obejść. W logice mamy wiele schematów aksjomatów i reguł dowodzenia. W arytmetyce mamy schemat indukcji i używając go dowodzimy na przykład inne, bardziej wygodne w użyciu schematy indukcji. Wrocławski profesor Czesław Ryll-Nardzewski wykazał, że arytmetycznego schematu indukcji nie można zastąpić skończonym zbiorem aksjomatów nie ograniczając przy tym mocy arytmetyki Peano.

Sytuacja jest jeszcze bardziej skomplikowana. Posługujemy się dwoma rodzajami formuł. Za formułę można uznać stwierdzenie „13 jest liczbą pierwszą". Ale formuły to także pewne obiekty matematyczne, które mają różne, dające się dowieść własności. Zwykle są to ciągi specjalnych obiektów – znaków. Przyjmijmy, że są to elementy zbioru $\mathcal{F}$. Dla stwierdzenia φ, takiego jak wyżej, można utworzyć obiekt $\underline{\varphi} \in \mathcal{F}$, który jest sformalizowaną wersją stwierdzenia φ. Takie obiekty będziemy nazywać standardowymi formułami.

Chcielibyśmy użyć formułę $\underline{\varphi}$ w definicji zbioru takiego, jak $\{n \in N : \varphi(n)\}$. Nie możemy tego jednak zrobić bezpośrednio pisząc $\{n \in N : \underline{\varphi}[n]\}$. Popełnilibyśmy błąd taki, jak pisząc $\{n \in N : 5\}$. Zarówno 5, jak i $\underline{\varphi}$ to przedmioty, a nie wymagane tutaj własności. Aby poradzić sobie z tym problemem musimy określić „znaczenie" $\underline{\varphi}$. Robimy to definiując pojęcie spełniania, w szczególności definiując relację $N \models \underline{\varphi}[h]$ spełniania formuły $\underline{\varphi}$ w strukturze N przy wartościowaniu h. Mając taką relację zbiór $\{n \in N : \varphi(n)\}$ możemy zdefiniować[3] jako $\{n \in N : N \models \underline{\varphi}[n]\}$.

Po takim skomplikowaniu języka możemy wykazać, że bez względu na to, jaką mam na myśli arytmetyczną formułę φ, bez kwantyfikatorów nieograniczonych, zachodzi

Twierdzenie 1.7 *Zbiór* $\{n \in N : N \models \underline{\varphi}[n]\}$ *jest pierwotnie rekurencyjny.*

Dwa ostatnie twierdzenia mówią właściwie to samo, a występujące w nich zbiory są identyczne. Wynika to z ogólnego i bardzo ważnego schematu, wyjaśniającego sytuację, stwierdzającego, że dla dowolnej arytmetycznej formuły φ zachodzi

Twierdzenie 1.8 *Własność* $\varphi(n)$ *zachodzi wtedy i tylko wtedy, gdy* $N \models \underline{\varphi}[n]$.

Dowód. □

Twierdzenie 1.7 ma jeszcze ogólniejszą postać:

[3]Pisząc $\varphi(n)$ mam na myśli własność φ, w której ściśle określoną zmienną (wolną) zastąpiłem liczbą n. Z drugiej strony w wyrażeniu $N \models \underline{\varphi}[n]$ mam na myśli spełnianie przy wartościowaniu, które tej określonej zmiennej (a właściwie jej odpowiednikowi) przyporządkowuje liczbę n.

Twierdzenie 1.9 *Dla każdej formuły arytmetycznej $\varphi \in \mathcal{F}$, bez kwantyfikatorów nieograniczonych, zbiór $\{n \in N : N \models \varphi[n]\}$ jest pierwotnie rekurencyjny.*

Precyzyjniejsze sformułowanie tego twierdzenia wraz z dowodem jest zamieszczone w następnym rozdziale. Teraz zauważmy, że jest to twierdzenie, a nie schemat, powinien wynikać z niego schemat 1.7, a wobec z twierdzenia 1.8 powinien też wynikać schemat 1.6.

Co więcej, twierdzenie 1.9 wydaje się silniejsze od 1.7. W przeciwieństwie do tego ostatniego, podaje własności dowolnych formuł, także niestandardowych, o ile takie istnieją. A nie jest jasne, czy można wykluczyć ich istnienie. Z rozważań związanych w twierdzeniem Gödla o niezupełności wynika, że często w arytmetyce mamy do czynienia z analogiczną sytuacją, w której dla dowolnej liczby naturalnej n dowodzi się własność $\varphi(\underline{n})$, ale nie można dowieść zdania $\forall x\ \varphi(x)$. Oznacza to, że w pewnym sensie nie da się wykluczyć istnienia niestandardowych liczb naturalnych.

1.8 Klasy Δ_0, Σ_1 i $\prod_1$

Symbolem Δ_0 oznaczamy najmniejszą klasę formuł języka arytmetyki zawierającą formuły atomowe i zamkniętą ze względu na łączenie formuł za pomocą spójników, poprzedzanie negacją i dopisywanie kwantyfikatorów ograniczonych $\forall x < t$ oraz $\exists x < t$, gdzie t jest termem, w którym nie występuje zmienna x.

Klasa Σ_1 jest najmniejszą klasą formuł języka arytmetyki zawierającą formuły klasy Δ_0 i zamkniętą ze względu na łączenie formuł spójnikami koniunkcji i alternatywy, a także dopisywanie kwantyfikatorów egzystencjalnych $\exists$ oraz ogólnych kwantyfikatorów ograniczonych $\forall x < t$, gdzie t jest termem, w którym nie występuje zmienna x.

Klasa $\prod_1$ jest najmniejszą klasą formuł języka arytmetyki zawierającą formuły klasy Δ_0 i zamkniętą ze względu na łączenie formuł spójnikami koniunkcji i alternatywy, a także dopisywanie kwantyfikatorów ogólnych $\forall$ oraz egzystencjalnych kwantyfikatorów ograniczonych $\exists x < t$, gdzie t jest termem, w którym nie występuje zmienna x.

Relacja $R \subseteq N^k$ jest klasy Δ_0, jeżeli jest definiowana formułą klasy Δ_0, a więc jeżeli jest zbiorem postaci[4] $\{\vec{n} \in N^k : N \models \varphi[\vec{n}]\}$ dla pewnego $\varphi \in \Delta_0$.

Analogicznie definiujemy relacje klasy Σ_1 i $\prod_1$. Relację nazywamy arytmetyczną, jeżeli jest definiowana (w powyższym sensie) pewną formułą języka arytmetyki.

Twierdzenie 1.10 *Relacje klasy Δ_0 są pierwotnie rekurencyjne.*

Dowód. Ponieważ w języku arytmetyki mamy do dyspozycji tylko symbole oznaczające dodawanie i mnożenie, więc termy języka arytmetyki definiują wyłącznie wielomiany o współczynnikach naturalnych, które są funkcjami pierwotnie rekurencyjnymi.

Stąd, z pierwotnej rekurencyjności relacji $<$ i $=$ oraz z lematu o podstawianiu otrzymujemy, że formuły atomowe języka arytmetyki definiują relacje pierwotnie rekurencyjne.

Jeżeli formuła φ definiuje pierwotnie rekurencyjną relację R, to formuła $\neg\varphi$ definiuje relację S, której funkcja charakterystyczna wyraża się wzorem

$$ch_S(\vec{x}) = 1 - ch_R(\vec{x}),$$

[4] Napis $N \models \varphi[\vec{n}]$ oznacza, że formuła φ jest spełniona w strukturze N przy wartościowaniu, które zmiennym z pewnego, ustalonego ciągu, zawierającego wszystkie zmienne wolne formuły φ, przyporządkowuje kolejno wyrazu ciągu $\vec{n}$.

która oczywiście jest pierwotnie rekurencyjna.

Jeżeli formuły φ_1 i φ_2 definiuje pierwotnie rekurencyjne relacje R_1 i R_2, to formuła $\varphi_1 \vee \varphi_2$ definiuje relację S, której funkcja charakterystyczna wyraża się wzorem

$$ch_S(\vec{x}) = ch_{R_1}(\vec{x}) \cdot ch_{R_2}(\vec{x}),$$

która też jest pierwotnie rekurencyjna.

Jeżeli φ definiuje pierwotnie rekurencyjną relację R, to formuła $\exists x < y\ \varphi$ (gdzie y nie występuje w φ) definiuje relację S, której funkcja charakterystyczna wyraża się wzorem

$$ch_S(\vec{x}, y) = \prod_{i<y} ch_R(\vec{x}, i).$$

Tak zdefiniowana funkcja ch_S także jest pierwotnie rekurencyjna.

Korzystając z przedstawionych wyżej konstrucji funkcji charakterystycznych można wykazać dowodzone twierdzenie w całej ogólności. □

1.9 Funkcja β Gödla

Twierdzenie 1.11 *Istnieją formuła β klasy Δ_0 ze zmiennymi wolnymi x, y, z oraz pierwotnie rekurencyjne funkcje $\beta : N^2 \to N$ i $f : N^2 \to N$ takie, że*

1) *$\beta(a, i) = n$ wtedy i tylko wtedy, gdy $\mathcal{N} \models \beta[a, i, n]$,*

2) *$\beta(a, i) \leqslant a - 1$ dla wszystkich $a, i \in N$,*

3) *dla każdego ciągu $a_0, \ldots, a_{n-1}$ istnieje liczba $a < f(\max\{a_0, \ldots, a_{n-1}\}, n)$ taka, że równości $\beta(a, i) = a_i$ są prawdziwe dla wszystkich $i < n$.*

Dowód. Zdefiniujmy funkcję (term) op przyjmując, że

$$op(a, b) = (a + b)(a + b) + a + 1,$$

relację (formułę) div taką, że

$$div(a, b) \Leftrightarrow \exists c < a + 1\ (a = b \cdot c)$$

(mówiącą, że a jest podzielne przez b) oraz pomocniczą formułę $\delta = \delta(x, y, z)$ równą

$$\exists u < x \exists v < x(op(u, v) = x \wedge div(u, 1 + (op(z, y) + 1) \cdot v)).$$

Przyjmimy, że $\beta = \beta(x, y, z)$ jest formułą

$$((\delta \wedge z < x - 1) \vee (z = x - 1)) \wedge \forall t < z\ \neg\delta[z \leftarrow t],$$

gdzie $x-1$ oznacza ograniczone odejmowanie, a formuły z takimi termami powinny zostać zastąpione równoważnymi, zapisanymi w języku arytmetyki. Oczywiście, jest to formuła klasy Δ_0. Najpierw sprawdzimy, że warunek 1) z tezy lematu definiuje pewną funkcję.

Łatwo przekonać się, że formuła β definiuje relację jednoznaczną. Jeżeli mamy $\mathcal{N} \models \beta[a, i, m]$ i $\mathcal{N} \models \beta[a, i, n]$ dla liczb $n \leqslant a-1$ i $m < n$, to także $\mathcal{N} \models \delta[a, i, m]$ (z pierwszego członu β i warunku $\mathcal{N} \models \beta[a, i, m]$) oraz $\mathcal{N} \models \neg\delta[a, i, m]$ (z drugiego członu i drugiego warunku). Uzyskana sprzeczność dowodzi jednoznaczności relacji definiowanej formułą β, a więc formuła ta definiuje pewną, niekoniecznie całkowitą, funkcję β.

Weźmy teraz liczby $a, i \in N$. Są możliwe dwa przypadki.

Przypadek 1: $\mathcal{N} \models (\forall t < z \ \neg\delta[z \leftarrow t])[a, i, a-1]$. W tym przypadku łatwo przekonać się, że $\mathcal{N} \models \beta[a, i, a-1]$.

Przypadek 2: $\mathcal{N} \models (\exists t < z \ \delta[z \leftarrow t])[a, i, a-1]$. Teraz możemy wziąć najmniejszą liczbę $n < a - 1$ taką, że $\mathcal{N} \models \delta[a, i, n]$. Ponieważ jest to najmniejsza taka liczba, więc $\mathcal{N} \models (\forall t < z \ \neg\delta[z \leftarrow t])[a, i, n]$. Łatwo sprawdzić, że $\mathcal{N} \models \beta[a, i, n]$.

Z przeprowadzonego rozumowania wynika, że dla każdych liczb a, i istnieje liczba $n \leqslant a - 1$ taka, że $\mathcal{N} \models \beta[a, i, n]$. Tak więc, β jest funkcją całkowitą. Pozostało dowieść, że funkcja β ma własność 3).

Przypuśćmy, że mamy (niepusty) ciąg $a_0, \ldots, a_{n-1}$ i $m = \max\{a_0, \ldots, a_{n-1}\}$. Bierzemy $b = op(m, n)$. Liczba ta dla wszystkich $i < n$ przekracza wartość $op(a_i, i)$. Tworzymy teraz iloczyn

$$c = \prod_{i<n} (1 + (op(a_i, i) + 1) \cdot b!).$$

Zauważmy, że liczba c została tak zdefiniowana, że $c < ((b+1)!)^n$ oraz własność

$$\mathcal{N} \models \delta[op(c, b!), i, a_i]$$

zachodzi dla wszystkich $i < n$.

Aby dowieść twierdzenie, należy jeszcze pokazać, że

$$\mathcal{N} \models \beta[op(c, b!), i, a_i]$$

dla wszystkich $i < n$. Gdyby nie było to prawdą, to

$$\mathcal{N} \models \delta[op(c, b!), i, d]$$

dla pewnego $i < n$ i dla pewnej liczby $d < a_i$. Z różnowartościowości op otrzymalibyśmy, że liczba $1 + (op(d, i) + 1) \cdot b!$ dzieli liczbę c.

Z drugiej strony, dla $j < b$ liczby postaci $1 + (j + 1) \cdot b!$ są parami względnie pierwsze. Liczba $op(d, i)$ jest mniejsza od b, a więc $1+(op(d, i)+1)\cdot b!$ jest względnie pierwsza z liczbami $1 + (op(a_k, k) + 1) \cdot b!$ i w konsekwencji jest względnie pierwsza z c. Jedyną liczbą dzielącą c i względnie pierwszą z c jest 1. Równość $1 = 1 + (op(d, i) + 1) \cdot b!$ nie jest jednak możliwa.

Uzyskana sprzeczność świadczy o tym, że liczba $a = op(c, b!)$ spełnia tezę twierdzenia. Określmy teraz

$$f(m, n) = op(((op(m, n) + 1)!)^n, op(m, n)!).$$

Zdefiniowaliśmy w ten sposób pierwotnie rekurencyjną funkcję, która dla $m = \max\{a_0, \ldots, a_{n-1}\}$ spełnia nierówność

$$a = op(c, b!) < op(((b+1)!)^n, b!) = op(((op(m, n) + 1)!)^n, op(m, n)!) = f(m, n).$$

Pozwala to zakończyć dowodzenie twierdzenia. $\Box$

1.10 Definiowalność funkcji pierwotnie rekurencyjnych

Wykresem funkcji $f : N^k \to N$ nazywamy relację

$$W_f = \{\langle \vec{x}, y\rangle \in N^{k+1} : f(\vec{x}) = y\}.$$

W powyższym wzorze warunek $f(\vec{x}) = y$ oznacza, że f jest określona dla argumentu $\vec{x}$ i dla tego argumentu przyjmuje wartość y.

Funkcja jest definiowalna wtedy i tylko wtedy, gdy jej wykres jest definiowalny.

Z twierdzenia 1.11 wynika, że funkcja β i jej wykres są definiowalne formułą klasy Δ_0. W dalszym ciągu tę formułę będziemy nazywać β, jak w sformułowaniu twierdzenia.

Twierdzenie 1.12 *Wykresy funkcji pierwotnie rekurencyjnych są klasy* Σ_1.

Dowód. Wykresy funkcji $I_{n,k}$, $+$ i $\cdot$ są definiowane odpowiednio formułami $x_k = y$, $x_1 + x_2 = y$ oraz $x_1 \cdot x_2 = y$.

Definiowalność wykresów bardziej skomplikowanych funkcji pierwotnie rekurencyjnych uzasadnimy jedynie w prostych przypadkach.

Przypuśćmy, że $f(x) = g(h(x))$ jest złożeniem funkcji g i h definiowanych odpowiednio formułami $\psi = \psi(x, y)$ i $\varphi = \varphi(x, y)$. Wtedy f jest definiowana formułą

$$\exists z\ \varphi[y \leftarrow z] \wedge \psi[x \leftarrow z].$$

Jest to oczywiście formuła klasy Σ_1, jeżeli formuły ψ i φ są tej klasy.

Załóżmy, że funkcja f definiowana równościami

$$f(0, y) = g(y) \quad \text{oraz} \quad f(x + 1, y) = h(f(x, y), x, y)$$

została określona za pomocą funkcji g i h definiowanych formułami $\varphi = \varphi(y, z)$ i $\psi = \psi(t, x, y, z)$ odpowiednio. Wtedy funkcja f jest definiowana formułą (zapisaną dla poprawienia czytelności bez operacji podstawiania)

$$\exists a\ [\ \beta(a,x,z) \wedge \exists v < a\ (\beta(a,0,v) \wedge \varphi(y,v)) \wedge$$
$$\wedge \forall i < x\ \exists u < a\ \exists v < a\ (\beta(a,i,u) \wedge \beta(a,i+1,v) \wedge \psi(u,i,y,v))\ ].$$

Formuła ta mówi, że a zawiera informacje o ciągu $f(0,y), f(1,y), \ldots, f(x,y)$ i z jest ostatnim wyrazem tego ciągu, albo mówi, że ciąg $\beta(a,0), \beta(a,1), \ldots, \beta(a,x)$ spełnia równości wymagane w rekurencyjnej definicji funkcji f od ciągu $f(0,y), f(1,y), \ldots, f(x,y)$. $\square$

Wniosek 1.13 *Relacje pierwotnie rekurencyjne są klasy Σ_1 i są klasy $\prod_1$.*

Dowód. Niech R będzie relacją pierwotnie rekurencyjną, a φ – formułą klasy Σ_1 definiującą wykres funkcji charakterystycznej relacji R. Tak więc

$$ch_R(\vec{x}) = y \Leftrightarrow \mathcal{N} \models \varphi[\vec{x}, y].$$

Wtedy

$$R(\vec{x}) \Leftrightarrow \mathcal{N} \models \varphi[y \leftarrow 0][\vec{x}] \Leftrightarrow \mathcal{N} \models \neg\varphi[y \leftarrow 1][\vec{x}]. \ \square$$

1.11 Twierdzenie o postaci normalnej

Innym przykładem zastosowania funkcji β jest

Twierdzenie 1.14 (o postaci normalnej) *Każda formuła klasy Σ_1 jest równoważna w modelu standardowym $\mathcal{N}$ formule z jednym nieograniczonym kwantyfikatorem egzystencjalnym postaci $\exists x \varphi$ dla pewnej formuły φ klasy Δ_0.*

Dowód. Dowodzimy ten fakt przez indukcję ze względu na budowę formuły klasy Σ_1, patrz str. 8. W kilku przypadkach wynika on z praw rachunku kwantyfikatorów pozwalających dopisać niepotrzebny kwantyfikator, przestawić kwantyfikator egzystencjalny z alternatywą bądź innym kwantyfikatorem tego samego rodzaju. Do rozważenia pozostają trzy przypadki.

Przypuśćmy, że rozważamy koniunkcję dwóch formuł klasy Σ_1, którym na mocy założenia indukcyjnego można nadać podaną postać. Taką koniunkcję można też wyrazić w następujący sposób:

$$(\exists x\ \varphi) \wedge (\exists x\ \psi) \Leftrightarrow \exists z\ \exists u, v < z\ (\beta(z,0,u) \wedge \beta(z,1,v) \wedge \varphi[x \leftarrow u] \wedge \psi[x \leftarrow v]).$$

Analogicznie postępujemy, gdy rozważamy formułę otrzymaną z formuły klasy Σ_1 przez dopisanie nieograniczonego kwantyfikatora egzystencjalnego. Wtedy

$$\exists y\ \exists x\ \varphi \Leftrightarrow \exists z\ \exists u, v < z\ (\beta(z,0,u) \wedge \beta(z,1,v) \wedge \varphi[x \leftarrow u][y \leftarrow v]).$$

Bardziej skomplikowaną sytuację mamy, gdy do formuły klasy Σ_1 dopisujemy ograniczony kwantyfikator ogólny. Wtedy

$$\forall y < t\ \exists x\ \varphi \Leftrightarrow \exists z\ \forall y < t\ \exists x < z\ (\beta(z,y,x) \wedge \varphi). \ \square$$

1.12 Relacje rekurencyjnie przeliczalne

Zbiory (relacje) rekurencyjnie przeliczalne zwykle definiujemy inaczej. Równoważnie możemy przyjąć, że relacja $S \subseteq N^k$ jest rekurencyjnie przeliczalna, jeżeli

$$S(\vec{x}) \Leftrightarrow \exists y\ R(\vec{x}, y)$$

dla pewnej pierwotnie rekurencyjnej relacji $R \subseteq N^{k+1}$. Korzystając z twierdzenia o postaci normalnej rekurencyjną przeliczalność można wyrazić w terminach definiowalności:

Wniosek 1.15 *Relacja $S \subseteq N^k$ jest rekurencyjnie przeliczalna wtedy i tylko wtedy, gdy jest definiowana formułą klasy Σ_1.*

Dowód. Z wniosku 1.13 wynika, że relacja R w powyższej definicji S jest definiowana formułą klasy Σ_1. Definicja S powstaje przez dopisanie do definicji R kwantyfikatora egzystencjalnego, a więc pozostaje w klasie Σ_1.

Z drugiej strony, jeżeli S ma definicję klasy Σ_1, to na podstawie twierdzenia o postaci normalnej ma też definicję, w której kwantyfikator egzystencjalny zostaje dopisany do pewnej formuły klasy Δ_0. Formuła ta definiuje relację pierwotnie rekurencyjną (patrz twierdzenie 1.10). Relację S można więc zdefiniować w sposób, który świadczy o tym, że jest rekurencyjnie przeliczalna. □

Jest to pierwszy i nieco niejasny wynik świadczący o tym, że pojęcia informatyczne można definiować za pomocą definiowalności.

Z twierdzenia o postaci lub z jego dowodu można wyprowadzić następujące własności relacji rekurencyjnie przeliczalnych:

Lemat 1.16 *Relacje pierwotnie rekurencyjne są rekurencyjnie przeliczalne. Klasa relacji rekurencyjnie przeliczalnych jest zamknięta ze względu na koniunkcję i alternatywę, operację rzutowania (definiowanie przez dopisywanie kwantyfikatora egzystencjalnego) oraz na definiowanie za pomocą kwantyfikatorów ograniczonych.*□

1.13 Kodowanie ciągów

Mając funkcję β Gödla możemy ciągi liczb naturalnych kodować za pomocą liczb naturalnych.

Przyjmujemy, że każda liczba koduje pewien ciąg, że długość ciągu kodowanego przez a znajdujemy obliczając $lh(a) = \beta(a, 0)$, a i-ty wyraz ciągu kodowanego przez a jest równy $(a)_i = \beta(a, i+1)$. Tak więc liczba a koduje ciąg $(a)_0, \ldots, (a)_{lh(a)-1}$.

Z własności funkcji β wynika, że każdy ciąg skończony jest kodowany przez przez pewną liczbę naturalną. Nietrudno zauważyć, że każdy ciąg jest kodowany przez nieskończonie wiele liczb.

Kodem ciągu nazywamy najmniejszą liczbę kodującą ten ciąg.

Dla ustalonego n symbolem $\langle \cdot, \ldots, \cdot \rangle$ będziemy oznaczać funkcję przyporządkowującą n-ce liczb kod ciągu złożonego z tych liczb. Tak więc

$$\langle a_0, \ldots, a_{n-1} \rangle = \mu x \; (lh(x) = n \wedge \bigwedge_{i<n} (x)_i = a_i).$$

Lemat 1.17 *Funkcje (przyporządkowujące wartości) $lh(a)$, $(a)_i$ oraz $\langle a_0, \ldots, a_{n-1} \rangle$ są pierwotnie rekurencyjne.*

Dowód. Nie budzi to wątpliwości w przypadku dwóch pierwszych funkcji. Trzecia z funkcji jest pierwotnie rekurencyjna na mocy lematu 1.4, można ją też zdefiniować wzorem

$$\langle a_0, \ldots, a_{n-1} \rangle = \mu x < f(\max\{a_0, \ldots, a_{n-1}\}, n) \; (lh(x) = n \wedge \bigwedge_{i<n} (x)_i = a_i),$$

gdzie f jest funkcją z twierdzenia 1.11 o funkcji β. □

Można zdefiniować też bardziej skomplikowane funkcje operujące na kodach ciągów, na przykład funkcję *conc*, która dwom liczbom przyporządkowuje kod konkatenacji ciągów kodowanych przez te liczby

$$conc(a,b) = \mu x\ (lh(x) = lh(a) + lh(b) \wedge$$
$$\wedge\ \forall i < lh(a)\ (x)_i = (a)_i \wedge \forall i < lh(b)\ (x)_{lh(a)+i} = (b)_i)$$

oraz funkcję c, której wartością jest kod ciągu otrzymanego z ciągu kodowanego przez pierwszy argument przez dopisanie na końcu drugiego argumentu

$$c(a,b) = \mu x\ (lh(x) = lh(a) + 1 \wedge$$
$$\wedge\ (x)_{lh(a)} = b \wedge \forall i < lh(a)\ (x)_i = (a)_i) \ = \ conc(a, \langle b \rangle).$$

Lemat 1.18 *Funkcje conc i c są pierwotnie rekurencyjne.*

Dowód. Lemat ten dowodzimy tak, jak poprzedni. W szczególności,

$$conc(a,b) = \mu x < f(\max\{m(a), m(b)\}, lh(a) + lh(b))\ (lh(x) = lh(a) + lh(b) \wedge$$
$$\wedge\ \forall i < lh(a)\ (x)_i = (a)_i \wedge \forall i < lh(b)\ (x)_{lh(a)+i} = (b)_i),$$

gdzie

$$m(a) = \mu x < a\ (\forall i < lh(a)\ (a)_i \leqslant x).\ \square$$

1.14 Funkcje pierwotnie rekurencyjne dwóch zmiennych

Ten rozdział ma charakter techniczny, będziemy potrzebować rekurencyjną charakteryzację klasy złożonej z funkcji pierwotnie rekurencyjnych dwóch zmiennych.

Symbolem P_2 będziemy oznaczać najmniejszą klasę, do której należą funkcje $I_{2,1}$, $I_{2,2}$, funkcje stałe dwóch zmiennych, $(\cdot).$, S_2 zdefiniowana wzorem $S_2(x,y) = x+1$ oraz zdefiniowana w poprzednim rozdziale funkcja c, i która jest zamknięta ze względu na złożenie oraz następujący schemat rekursji prostej:

$$f(x,0) = g(x,0) \ \text{ oraz } \ f(x,n+1) = h(\langle x,n \rangle, f(x,n)).$$

Do klasy P_2, oprócz wymienionych w definicji, należy też funkcja $\langle x,y \rangle = c(c(0,x),y)$. Aby sprawdzić podaną równość wystarczy zauważyć, że 0 jest kodem ciągu pustego. Dla wszystkich $n \in N$ klasa P_2 jest też zamknięta ze względu na definiowanie wzorem $k_n(x,y) = \langle h_1(x,y), \ldots, h_n(x,y) \rangle$ Dowodzimy to przez indukcję. Oczywiście, $k_1(x,y) = \langle h_1(x,y) \rangle = c(0, h_1(x,y))$ należy do P_2, o ile $h_1 \in P_2$. Jeżeli do P_2 należą funkcje k_n i h_{n+1}, to należy tam także funkcja k_{n+1}, gdyż $k_{n+1}(x,y) = c(k_n(x,y), h_{n+1}(x,y))$.

Mamy oczywisty fakt

Lemat 1.19 *Funkcje z klasy P_2 mają po dwie zmienne i są pierwotnie rekurencyjne.* $\square$

Niech f oznacza funkcję n zmiennych. Mając taką funkcję możemy zdefiniować funkcję $f^* : N^2 \to N$ żądając, aby

$$f^*(x,y) = f((x)_0, \ldots, (x)_{n-1})$$

(f^* tak naprawdę nie zależy od y).

Lemat 1.20 *Jeżeli f jest funkcją pierwotnie rekurencyjną, to $f^* \in P_2$.*

Dowód. Dowodzimy to przez indukcję wynikającą z definicji klasy funkcji pierwotnie rekurencyjnych.

Funkcja

$$S^*(x,y) = S((x)_0) = S_2((x)_0, y) = S_2((I_{2,1}(x,y))_{0(x,y)}, I_{2,2}(x,y))$$

należy do P_2, gdyż jest złożeniem kilku funkcji z P_2 ($0(x,y)$ to wartość funkcji dwóch zmiennych stale równej 0, korzystając dalej z wartości funkcji stałych stosujemy analogiczną notację). Dla funkcji stałych dopisanie * daję tę samą funkcję stałą, która należy do P_2. Mamy też

$$I^*_{n,k}(x,y) = I_{n,k}((x)_0, \ldots, (x)_{n-1}) = (x)_{k-1} = (I_{2,1}(x,y))_{(k-1)(x,y)},$$

a więc również $I^*_{n,k} \in P_2$.

Jeżeli na przykład $f(\vec{x}) = g(h_1(\vec{x}), h_2(\vec{x}), h_3(\vec{x}))$, to

$$f^*(x,y) = g(h^*_1(x,y), h^*_2(x,y), h^*_3(x,y)) = g^*(\langle h^*_1(x,y), h^*_2(x,y), h^*_3(x,y)\rangle, y).$$

Przypuśćmy, że $m > 0$ i $f : N^{m+1} \to N$ spełnia równości

$$f(\vec{x}, 0) = g(\vec{x}) \;\text{ oraz }\; f(\vec{x}, n+1) = h(\vec{x}, n, f(\vec{x}, n)).$$

Aby pokazać, że $f^* \in P_2$, zdefiniujmy pomocniczą funkcję dwóch zmiennych F:

$$F(x,0) = g^*(x,0) \;\text{ oraz }\; F(x, n+1) = h^*(c(\langle (x)_0, \ldots, (x)_{m-1}, n\rangle, F(x,n)), 0).$$

Jeżeli poprawimy definicję F tak, aby zamiast x i n były odpowiednie wyrażenia zależne od $\langle x, n\rangle$ (ich znalezienie nie jest to trudne), to okaże się, że funkcja F jest zdefiniowana zgodnie ze schematem rekursji z klasy P_2. Jeżeli dodatkowo będziemy wiedzieć, że funkcje g^* i h^* należą do P_2, to funkcja F też będzie z klasy P_2. O funkcji F można dowieść, że

$$F(x,y) = f((x)_0, \ldots, (x)_{m-1}, y).$$

Wiedząc, że $F \in P_2$ pokazujemy, że $f^* \in P_2$ korzystając z zależności

$$f^*(x,y) = F(x, (x)_m) = F(I_{2,1}(x,y), (I_{2,1}(x,y))_{m(x,y)}).$$

Pełny dowód lematu wymaga tylko uzupełnienia wyżej przedstawionych rozumowań i jest pozostawiony Czytelnikowi. □

Z udowodnionych lematów wynika następujące

Twierdzenie 1.21 *Klasa P_2 składa się dokładnie z pierwotnie rekurencyjnych funkcji dwóch zmiennych.*

Dowód. Dla pierwotnie rekurencyjnej funkcji $f : N^2 \to N$ mamy $f(x,y) = f^*(\langle x, y\rangle)$. Wobec tego, jeżeli $f^* \in P_2$, to także $f \in P_2$. □

1.15 Funkcja uniwersalna dla klasy P_2

Mając rekurencyjną definicję klasy P_2 możemy zdefiniowawać bardzo prosty język pozwalający programować obliczenia funkcji pierwotnie rekurencyjnych dwóch zmiennych.

Język ten będzie miał niespotykaną cechę. Programami w tym języku będą liczby naturalne (to jeszcze nic dziwnego) i to wszystkie.

Aby określić semantykę tego języka zdefiniujemy funkcję $[|\cdot|] : N \to P_2$ przyporządkowującą liczbie naturalnej n, czyli pewnemu programowi, funkcję $[|n|]$ obliczaną przez ten program. Przyjmujemy, że

$$[|n|] = \begin{cases} I_{2,1} & \text{jeżeli } (n)_0 = 0, \\ I_{2,2} & \text{jeżeli } (n)_0 = 1, \\ \text{funkcja stale równa } (n)_1 & \text{jeżeli } (n)_0 = 2, \\ (\cdot). & \text{jeżeli } (n)_0 = 3, \\ S_2 & \text{jeżeli } (n)_0 = 4, \\ c & \text{jeżeli } (n)_0 = 5, \\ \text{złożenie } [|(n)_1|] \text{ z } [|(n)_2|] \text{ i } [|(n)_3|] & \text{jeżeli } (n)_0 = 6, \\ \text{funkcja definiowana rekurencyjnie z} [|(n)_1|] \text{ i } [|(n)_2|] & \text{jeżeli } (n)_0 > 6. \end{cases}$$

Oczywiście, powyższe ustalenia wymagają doprecyzowania. Jeżeli $[|n|]$ jest złożeniem, to definiując je bierzemy $[|(n)_1|]$ jako funkcję zewnętrzną oraz $[|(n)_2|]$ i $[|(n)_3|]$ jako funkcje wewnętrzne. Jeżeli funkcję $[|n|]$ definiujemy przez rekursję, to określając jej wartość dla 0 bierzemy funkcję $[|(n)_1|]$, a w drugiej części definicji posługujemy się funkcją $[|(n)_2|]$.

Lemat 1.22 *Zbiorem wartości funkcji semantycznej* $[|\cdot|]$ *jest klasa* P_2. □

Przyjmijmy teraz, że

$$U(n,x,y) = \begin{cases} x & \text{jeżeli } (n)_0 = 0, \\ y & \text{jeżeli } (n)_0 = 1, \\ (n)_1 & \text{jeżeli } (n)_0 = 2, \\ (x)_y & \text{jeżeli } (n)_0 = 3, \\ x+1 & \text{jeżeli } (n)_0 = 4, \\ c(x,y) & \text{jeżeli } (n)_0 = 5, \\ U((n)_1, U((n)_2,x,y), U((n)_3,x,y)) & \text{jeżeli } (n)_0 = 6, \\ U((n)_1, x, 0) & \text{jeżeli } (n)_0 > 6 \wedge y = 0, \\ U((n)_2, \langle x, y-1\rangle, U(n,x,y-1)) & \text{jeżeli } (n)_0 > 6 \wedge y > 0. \end{cases}$$

Podane wzory są poprawną definicją pewnej funkcji $U : N^3 \to N$. Wynika to z odpoowiedniego twierdzenia o definiowaniu przez indukcję. Funkcja U wydaje się również obliczalna w sensie intuicyjnym. Mechaniczne urządzenie obliczające tę funkcję można uznać za interpreter rozważanego języka programowania. Po otrzymaniu programu n definiującego pewną funkcję i dwóch jej argumentów x i y urządzenie to oblicza $U(n,x,y)$ i tym samym znajduje wartość $[|n|](x,y)$. Można bowiem dowieść (zostawiamy to zainteresowanemu Czytelnikowi), że

Lemat 1.23 *Dla wszystkich* $n,x,y \in N$ *zachodzi wzór* $U(n,x,y) = [|n|](x,y)$.

Dowód. Korzystamy z indukcji ze względu na n. □

Tak więc, funkcja U po ustalenie pierwszego argumentu staje się funkcją dwóch zmiennych i należy do klasy P_2. Ponadto każdą funkcję z klasy P_2 można otrzymać odpowiednio ustalając pierwszy argument funkcji U. Funkcję U o takich własnościach nazywamy uniwesalną dla klasy P_2 (choć zwykle chcemy jeszcze, aby była ona w jakimś sensie łatwo obliczalna). Tym razem mamy jednak następujące

Twierdzenie 1.24 *Funkcja* U *nie jest pierwotnie rekurencyjna.*

Dowód. Załóżmy, że U jest pierwotnie rekurencyjna. Wtedy pierwotnie rekurencyjną jest również funkcja

$$f(x,y) = U(x,x,x) + 1.$$

Ponieważ jest to funkcja dwóch zmiennych, więc należy do klasy P_2 i dla odpowiedniego n mamy

$$f(x,y) = U(n,x,y)$$

dla wszystkich możliwych x i y. Biorąc $x = y = n$ otrzymujemy jednak równość

$$U(n,n,n) = f(n,n) = U(n,n,n) + 1$$

która nie jest prawdziwa. Otrzymana sprzeczność dowodzi tezy. □

Przedstawiona technika jest często wykorzystywana. Tym razem daje niewiele. Po pierwsze, że klasa funkcji pierwotnie rekurencyjnych nie jest dobrą formalizacją pojęcia obliczalności: jest funkcja, która wydaje się obliczalna, która nie jest pierwotnie rekurencyjna. Po drugie, znaleźliśmy schemat rekursji, którego nie da się sprowadzić do rekursji prostej. W końcu wydaje się uzasadniona hipoteza, że jeżeli definicję klasy funkcji pierwotnie rekurencyjnych wzbogacimy o bardziej skomplikowane schematy rekursji, na przykład taki, jak użyty w definicji funkcji U, to analogiczne rozumowanie pozwoli podać przykłady jeszcze bardziej skomplikowanego schematu rekursji i funkcji obliczalnej, której definicja wymaga użycia tego bardzo skomplikowanego schematu.

1.16 Rozszerzanie teorii o definicje nowych funkcji

Rozważamy teorię z równością, jej aksjomatami są więc między innymi aksjomaty równości.

Równość będziemy nazywać prostą, jeżeli jest albo postaci $f(x_1,\ldots,x_n) = y$ dla pewnego n-arnego symbolu funkcyjnego f i pewnych zmiennych $x_1,\ldots,x_n$ oraz zmiennej y różnej od $x_1,\ldots,x_n$, albo postaci $x = y$ dla różnych zmiennych, albo też postaci $c = y$ dla stałej c i zmiennej y. Formuła atomowa jest prosta, jeżeli jest prostą równością lub formułą atomową postaci $r(x_1,\ldots,x_n)$ dla pewnych zmiennych $x_1,\ldots,x_n$.

Lemat 1.25 *Dla każdej formuły $t = u$ ze zmiennymi u oraz $v_1,\ldots,v_n$ różnymi od u istnieje koniunkcja prostych równości φ, w której oprócz wymienionych występują również zmienne $\vec{x} = x_1,\ldots,x_m$, taka że w rachunku kwantyfikatorów z równością daje się dowieść równoważność*

$$t = u \Leftrightarrow \exists \vec{x}\ \varphi.$$

Dowód. Lemat dowodzimy przez indukcję ze względu na budowę termu t. Jeżeli t jest zmienną lub stałą, to za φ bierzemy formułę $t = u$.

Przypuśćmy, że $t = f(t_1,t_2)$. Z założenia indukcyjnego mamy koniunkcje prostych równości φ_1 i φ_2 takie, że

$$t_1 = u \Leftrightarrow \exists \vec{x}_1\ \varphi_1 \quad \text{oraz} \quad t_2 = u \Leftrightarrow \exists \vec{x}_2\ \varphi_2.$$

Wtedy (jeżeli odpowiednio dobierzemy zmienne $\vec{x}_1$ i $\vec{x}_2$)

$$\begin{aligned} f(t_1,t_2) = u \Leftrightarrow{}& \exists y_1,y_2\ (t_1 = y_1 \wedge t_2 = y_2 \wedge f(y_1,y_2) = u) \Leftrightarrow \\ &\Leftrightarrow \exists y_1,y_2\ (\exists \vec{x}_1\ \varphi_1[u \leftarrow y_1]) \wedge (\exists \vec{x}_2\ \varphi_2[u \leftarrow y_2]) \wedge f(y_1,y_2) = u \Leftrightarrow \\ &\Leftrightarrow \exists y_1,y_2\ \exists \vec{x}_1\ \exists \vec{x}_2\ (\varphi_1[u \leftarrow y_1] \wedge \varphi_2[u \leftarrow y_2] \wedge f(y_1,y_2) = u). \end{aligned}$$

Nietrudno zauważyć, że ostatnia formuła ma żądaną postać. □

Lemat 1.26 *Dla każdej formuły $R(t_1, \ldots, t_k)$ (atomowej lub negacji atomowej, a właściwie dowolnej) ze zmiennymi $v_1, \ldots, v_n$ istnieją różne od nich zmienne $\vec{u} = u_1, \ldots, u_k$, takie że następujące formuły są wzajemnie równoważne w rachunku kwantyfikatorów z równością*

1) $R(t_1, \ldots, t_l)$,

2) $\exists \vec{u} \left(\bigwedge_{i=1,\ldots,k} t_i = u_i \wedge R(u_1, \ldots, u_k) \right)$,

3) $\forall \vec{u} \left(\bigwedge_{i=1,\ldots,k} t_i = u_i \Rightarrow R(u_1, \ldots, u_k) \right)$.

Dowód. Jest to oczywista konsekwencja aksjomatów równości. □

Wniosek 1.27 *Dla każdej formuły $R(t_1, \ldots, t_k)$ (atomowej lub negacji atomowej, a właściwie dowolnej) ze zmiennymi $v_1, \ldots, v_n$ istnieją różne od nich zmienne $\vec{u} = u_1, \ldots, u_k$ oraz $\vec{x} = x_1, \ldots, x_m$, a także koniunkcja prostych równości φ, takie że następujące formuły są wzajemnie równoważne w rachunku kwantyfikatorów z równością*

1) $R(t_1, \ldots, t_l)$,

2) $\exists \vec{u}\, \exists \vec{x}\, (\varphi \wedge R(u_1, \ldots, u_k))$,

3) $\forall \vec{u}\, \forall \vec{x}\, (\varphi \Rightarrow R(u_1, \ldots, u_k))$.

Dowód. Jest to oczywista konsekwencja dwóch ostatnich lematów. □

Wniosek 1.28 *W rachunku kwantyfikatorów z równością każda formuła jest równoważna formule, w której występują tylko proste formuły atomowe.* □

Przypuśćmy, że w teorii T z równością daje się dowieść formułę $\forall \vec{x}\, \exists y^1\, \varphi(\vec{x}, y)$, gdzie $\exists^1$ oznacza kwantyfikator „istnieje dokładnie jeden" (istnieje y o własności φ i każde dwa takie y-ki są równe). Taką formułę można wykorzystać do wprowadzenia nowego symbolu funkcyjnego f. W tym celu roszerzamy teorię T o nowy aksjomat lub definicję

$$\forall \vec{x}\, \varphi(\vec{x}, f(\vec{x})),$$

albo o równoważne w teorii T stwierdzenie

$$\forall \vec{x}\, \forall y\, (\varphi(\vec{x}, y) \Leftrightarrow f(\vec{x}) = y).$$

W otrzymanej w ten sposób teorii T_f dowolną formułę ψ można przekształcić w ψ' w następujący sposób: sprawdzamy, czy w ψ występuje symbol f; jeżeli nie występuje, to $\psi' = \psi$, w przeciwnym razie najpierw znajdujemy równoważną postać formuły ψ zbudowaną wyłącznie z prostych formuł atomowych, a następnie konstruujemy ψ' zastępując w tej postaci formuły $f(\vec{x}) = y$ formułami $\varphi(\vec{x}, y)$. Teoria T_f i to przekształcenie mają następujące własności:

Twierdzenie 1.29 *Przypuśćmy, że $T \vdash \forall \vec{x}\, \exists y^1\, \varphi(\vec{x}, y)$, a T_f jest zdefiniowana jak wyżej. Wtedy dla każdej formuły ψ zapisanej w języku teorii T rozszerzonym o symbol f w teorii T_f daje się dowieść jej równoważność z formułą ψ'. Ponadto, jeżeli formuła ψ bez symbolu f daje się dowieść w teorii T_f, to daje się też dowieść w teorii T i to bez używania symbolu f.* □

Przytoczone twierdzenie mówi, że wprowadzając poprzez definicje nowe symbole funkcyjne nie zwiększamy mocy teorii, a jedynie zmieniamy sposób wyrażania interesujących nas własności. Możemy na przykład rozszerzyć arytmetykę o potęgowanie. Jeżeli robimy coś takiego, to najczęściej podajemy definicję rekurencyjną

$$x^0 = 1 \text{ oraz } x^{n+1} = x^n \cdot x.$$

Nie jest to jednak definicja w rozważanym teraz sensie. W arytmetyce możemy definiować liczby naturalne, związki między liczbami, nawet związki wyrażane za pomocą funkcji, ale nie możemy definiować funkcji. Wobec twierdzenia o definiowaniu przez indukcję, przytoczona definicja jest poprawną, teoriomnogościową definicją pewnego obiektu (zbioru), który jest funkcją. Możemy się nią posługiwać, możemy nawet dodać ją do arytmetyki. Dodając możemy jednak zwiększyć moc arytmetyki i na razie nie potrafimy tego zwiększenia wykluczyć.

Potęgowanie można zdefiniować w inny sposób, przyjmując, że

$$x^n = y \Leftrightarrow \varphi(x, n, y),$$

dla formuły $\varphi(x, n, y)$ postaci

$$\exists a\ (\beta(a, 0, 1) \wedge \forall i < n\ \forall z < a\ (\beta(a, i, z) \Rightarrow \beta(a, i + 1, z \cdot x)) \wedge \beta(a, n, y)$$

($\beta(a, n, y)$ to formuła definiująca funkcję β Gödla, formuła φ stwierdza istnienie liczby a kodującej ciąg $x^0, x^1, \ldots, x^n$ i uznaje y za ostatni wyraz tego ciągu). Aby do podanej definicji dało się zastosować twierdzenie 1.29, trzeba jeszcze w arytmetyce dowieść, że

$$\forall x\ \forall n\ \exists^1 y\ \varphi(x, n, y).$$

Jest to możliwe dopiero w dostatecznie silnej arytmetyce, na przykład w arytmetyce Peano. Tak więc arytmetykę Peano można bez zwiększania jej mocy rozszerzyć o potęgowanie. W podobny sposób można rozszerzyć arytmetykę Peano o wiele innych funkcji rekurencyjnych (choć nie o wszyskie).

Zauważmy jeszcze, że mnożenia nie można zdefiniować za pomocą dodawania. Dokładniej, gdyby w arytmetyce Presburgera dało się zdefiniować mnożenie, to na mocy twierdzenia 1.29 każde zdanie arytmetyczne miałoby swój odpowiednik w arytmetyce Presburgera i teoria Q byłaby częścią arytmetyki Presburgera, która jest zupełna. Przeczy to jednak twierdzeniu Gödla o niezupełności arytmetyki, które zostanie dowiedzione w następnym rozdziale.

2 Funkcje rekurencyjne

2.1 Trochę historii

Twierdzenie Gödla zostało ogłoszone w 7 września 1930 roku podczas konferencji w Królewcu. Jest opublikowane w pracy ze stycznia 1931 roku.

Także w 1931 roku Herbrand pisze list do Gödla, w którym zwraca uwagę na bardziej ogólne funkcje rekurencyjne. Herbrand ginie w Alpach w lipcu 1931.

W latach 1932 – 1935 Alonzo Church i jego uczeń Stephen Kleene rozwijali λ-rachunek i zdefiniowali pojęcie funkcji λ-definiowalnych. Wiele spostrzeżeń i uzyskiwanych wyników podpowiadało, że udało się im sformalizować pojęcie obliczalności. Przez pewien czas argumentem przeciw były kłopoty z wykazaniem λ-definiowalności funkcji $f(n) = n - 1$. Udało się ten fakt dowieść Kleene'emu w 1932 roku.

Gödel chyba został zaproszony do współpracy nad obliczalnością, ale odnosił się sceptycznie do tego projektu. Jednak podczas wykładu w Princeton wiosną 1934 roku zaproponował zgodnie z sugestiami Herbranda pewne uogólnienie pojęcia funkcji pierwotnie rekurencyjnych, patrz str. 23. W ten sposób pojawiła się definicja tzw. funkcji ogólnie rekurencyjnych, nazywanych dzisiaj rekurencyjnymi według Herbranda i Gödla. Sam Gödel nie był chyba przekonany, że da się dowieść oczekiwane własności tak definiowanych funkcji.

W 1936 ukazały się prace Churcha i Kleene'ego, w których zostało dowiedzione, że klasy funkcji λ-definiowalnych i ogólnie rekurencyjnych są równe. Przy okazji Church opublikował tezę mówiącą, że naturalne funkcje obliczalne (w sensie potocznym lub intuicyjnym) są λ-definiowalne i – równoważnie – ogólnie rekurencyjne. W tych pracach została też wykazana ich zamkniętość ze względu na operację minimum. Dało to możliwość zdefiniowania μ-rekurencyjności, dzisiaj najczęściej wykorzystywanej definicji rekurencyjności. Jednak przez pewien czas pojęcie rekurencyjności rozumiano jako ogólną rekurencyjność. W *Introduction to Metamathematics* Klenne'ego z 1952 roku klasa funkcji μ-rekurencyjnych jest wyraźnie zdefiniowania i zostało dowiedzione, że jest równa klasie funkcji ogólnie rekurencyjnych.

W 1936 roku, w momencie ukazania się pracy Churcha, Alan Turing miał już przygotową swoją pracę o maszynach Turinga. Praca była bogata w różne treści, pokazywała możność przeprowadzenia na maszynach Turinga rozmaitych obliczeń matematycznych, zawierała konstrukcję uniwersalnej maszyny Turinga, a także dowód nierozstrzygalności pewnej teorii. Po krótkim zamieszaniu dotyczącym oryginalności pracy, została ona opublikowana pod koniec roku. W 1937 roku ukazała się praca Turinga z dowodem równoważności λ-definiowalności i obliczalności na maszynach Turinga oraz z tezą analogiczną do tezy Churcha.

Także w 1936 roku ukazała się praca Emila Posta zawierająca definicję urządzenia podobnego do maszyny Turinga i przeświadczenie autora, że jego pomysł okaże się równoważny rekurencyjności.

Pionierem badań nad obliczalnością jest prawdopodobnie Emil Post. Jednak swoje wyniki z lat 1920-22 opublikował dopiero w 1943 roku. Proponował pojęcie obliczenia wzorowane na obliczeniach algebraicznych, patrz str. 26.

2.2 Definicje

Klasa funkcji rekurencyjnych jest najmniejszą klasą częściowych funkcji naturalnych zawierającą wszystkie funkcje $I_{n,k}$ i zamkniętą ze względu na złożenie, rekursję prostą i operację minimum. Funkcję nazywamy rekurencyjną, jeżeli należy do klasy funkcji rekurencyjnych.

Zbiór (relacja) $X \subseteq N^k$ jest rekurencyjny, jeżeli jego funkcja charakterystyczna ch_X jest rekurencyjna.

Lemat 2.1 *Funkcje pierwotnie rekurencyjne są rekurencyjne.*

Dowód. Mając funkcje $I_{n,k}$ łatwo przez rekursję wydefiniować funkcje stałe, ograniczone odejmowanie 1 i ograniczone odejmowanie. Wiemy, że $ch_<(m,n) = 1-(n-m)$. Teraz możemy zdefiniować następnik przyjmując $S(m) = \mu n\ (ch_<(m,n) = 0)$. Następnik pozwala wydefiniować wszystkie pozostałe funkcje pierwotnie rekurencyjne. □

Przykładem częściowej funkcji rekurencyjnej może być

$$f(m) = \mu n\ (n * n = m).$$

Przytoczona definicja funkcji rekurencyjnych jest najogólniejsza. Odpowiada pojęciu obliczalności. Przez dłuższy czas pojęcie rekurencyjności było ograniczane do funkcji całkowitych. My też możemy się już ograniczyć do funkcji całkowitych.

Najprościej klasą całkowitych funkcji rekurencyjnych można zdefiniować jako zbiór tych funkcji rekurencyjnych (np. w podanym sensie), które są całkowite.

Inna definicja mówi, że klasa (całkowitych) funkcji rekurencyjnych jest najmniejszą klasą funkcji naturalnych zawierającą wszystkie funkcje $I_{n,k}$ i zamkniętą ze względu na złożenie, rekursję prostą i efektywną operację minimum.

Najczęściej przyjmuje się, że klasa (całkowitych) funkcji rekurencyjnych jest najmniejszą klasą funkcji naturalnych zawierającą wszystkie funkcje $I_{n,k}$, funkcje stałe, dodawanie i mnożenie, a także funkcję charakterystyczną nierówności $ch_<$ i zamkniętą ze względu na złożenie i efektywną operację minimum.

Funkcje rekurencyjne zgodnie z ostatnią definicją określa się czasem jako μ-rekurencyjne. Dowodzi się, że trzy ostatnie definicje są równoważne.

2.3 Najważniejsze własności funkcji rekurencyjnych

W rozdziale 1.6 jest podane wiele przykładów funkcji pierwotnie rekurencyjnych, które w związku z tym są rekurencyjne. Także rezultaty tego rozdziału pozostają prawdziwe, gdy zamiast pierwotnej rekurencyjności będzie w nich mowa o rekurencyjności. Jednak w lemacie o podstawianiu teraz istotne jest niejawne założenie o całkowitości podstawianej funkcji.

Nawet jeżeli nie mamy wiedzy o zamkniętości klasy funkcji rekurencyjnych na rekursję prostą (patrz ostatnia definicja klasy funkcji rekurencyjnych) możemy posłgiwać się funkcją β Gödla. Co prawda, w dowodzie lematu o β korzystaliśmy z silni, ale nie korzystaliśmy z rekurencyjności tej funkcji.

Lemat 2.2 *Klasa funkcji rekurencyjnych jest zamknięta ze względu na rekursję prostą.*

Dowód. Przypuśćmy, że funkcja f jest definiowana przez rekursję prostą wzorami

$$f(\vec{x},0) = g(\vec{x}) \text{ oraz } f(\vec{x},n+1) = h(\vec{x},n,f(\vec{x},n))$$

dla pewnych rekurencyjnych (i całkowitych) funkcji g i h. Zauważmy, że równość $f(\vec{x},n) = y$ zachodzi wtedy i tylko wtedy, gdy

$$\exists a\, (\beta(a,0) = g(\vec{x}) \wedge \forall i < n\, (\beta(a,i+1) = h(\vec{x},i,\beta(a,i))) \wedge \beta(a,n) = y).$$

Z lematu o podstawianiu otrzymujemy, że

$$f(\vec{x},n) = y \Leftrightarrow \exists a\, R(\vec{x},n,y,a)$$

dla pewnej rekurencyjnej relacji R. Funkcja f jest rekurencyjna, ponieważ

$$f(\vec{x},n) = (\mu z\, R(\vec{x},n,(z)_0,(z)_1))_0. \ \Box$$

Podobnie dowodzimy

Lemat 2.3 *Funkcja Ackermanna jest rekurencyjna.*

Dowód. Dowód jest podobny do poprzedniego. Polega na zdefiniowaniu relacji mówiącej, że a koduje dostatecznie dużo informacji o funkcji Ackermanna, aby można było ustalić, że $A(m,n)=k$. Umówmy się, że a koduje informacje o funkcji Ackermanna w taki sposób, że dla niezbędnych i oraz j mamy

$$((a)_i)_j = A(i,j).$$

Przypomnijmy, że funkcja Ackermanna spełnia równości

$$A(0,n)=n{+}1,\quad A(m{+}1,0)=A(m,1)\ \text{ oraz }\ A(m{+}1,n{+}1)=A(m,A(m{+}1,n)).$$

Przyjmijmy, że napis $R(m,n,k,a)$ oznacza, że

$$\begin{aligned}
&((a)_m)_n = k\ \wedge\ lh(a) > m\ \wedge\ lh((a)_m) > n\ \wedge\\
&\wedge\ \forall j < lh((a)_0)\ (((a)_0)_j = j+1)\ \wedge\\
&\wedge\ \forall i < m\ (lh((a)_i) > 1\ \wedge\ ((a)_{i+1})_0 = ((a)_i)_1)\ \wedge\\
&\wedge\ \forall i < m\ \forall j < lh((a)_{i+1})\ (lh((a)_i) > ((a)_{i+1})_j\ \wedge\ ((a)_{i+1})_{j+1} = ((a)_i)_{((a)_{i+1})_j}.
\end{aligned}$$

Można dowieść, że R oznacza relację rekurencyjną taką, że

$$A(m,n)=k \Leftrightarrow \exists a\ R(m,n,k,a).$$

Jeżeli tak, to funkcję Ackermanna można też zdefiniować wzorem

$$A(m,n) = (\mu x\ R(m,n,(x)_0,(x)_1))_0.$$

Jest to więc funkcja rekurencyjna □

2.4 Definiowalność

Funkcje i relacje rekurencyjne są definiowane za pomocą formuł o analogicznych własnościach do formuł definiujących funkcje i relacje pierwotnie rekurencyjne. Tym razem jednak na ogół można dowieść równoważność, a rekurencyjność można charakteryzować za pomocą definiowalności.

Twierdzenie 2.4 *Funkcja jest rekurencyjna wtedy i tylko wtedy, gdy jest definiowalna formułą klasy* Σ_1*.*

Dowód. Istotny fragment dowodu to zamkniętość klasy funkcji definiowalnych formułami klasy Σ_1 ze względu na operację minimum. Przypuśćmy więc, że funkcja $f : N^{k+1} \to N$ jest definiowana formułą $\varphi \in \Sigma_1$, formuła $\beta \in \Delta_0$ definiuje funkcję Gödla β i

$$h(\vec{x}) = \mu y\ (f(\vec{x},y)=0).$$

Funkcja h jest definiowana formułą

$$\psi(\vec{x},z) = \varphi(\vec{x},z,0) \wedge \exists t\ \forall i < z\ \exists u < t\ (\beta(t,i,u) \wedge u > 0 \wedge \varphi(\vec{x},i,u)).$$

Jeżeli natomiast funkcja $f : N^k \to N$ jest definiowana formułą klasy Σ_1, to dla pewnej relacji pierwotnie rekurencyjnej $R \subseteq N^{k+2}$ mamy

$$f(\vec{x}) = y \Leftrightarrow \exists z\ R(\vec{x},y,z).$$

Zauważmy, że wtedy

$$f(\vec{x}) = (\mu t\ (R(\vec{x},(t)_0,(t)_1)))_0.\ \square$$

Lemat 2.5 *Zbiór jest rekurencyjny wtedy i tylko wtedy, gdy jest definiowany zarówno formułą klasy Σ_1, jak i $\prod_1$*

Dowód.

Wniosek 2.6 *Zbiór X jest rekurencyjnie przeliczalny wtedy i tylko wtedy, gdy jest definiowalny formułą klasy Σ_1.*

3 Inne formalizacje obliczalności

3.1 Funkcje ogólnie rekurencyjne

W tym rozdziale będziemy rozważać prymitywny język bez spójników logicznych i kwantyfikatorów, z jednym symbolem relacyjnym $=$, stałą 0, zmiennymi i wieloma symbolami funkcyjnymi, wśród których jest potencjalnie nieskończenie wiele symboli każdej arności. Termy w tym języku definiujemy w zwykły sposób. Jednoargumentowy symbol funkcyjny S będzie miał szczególne znaczenie, w szczególności jest wykorzystywany do reprezentowania liczb naturalnych. Przyjmujemy, że

$$\underline{0} = 0 \quad \text{oraz} \quad \underline{n+1} = S(\underline{n}).$$

Term $\underline{n}$ będziemy uważać za reprezentację liczby naturalnej n.

W razie potrzeby i zależnie od kontekstu symbole funkcyjne będziemy dzielić na stałe, o ustalonym znaczeniu i zmienne, bez określonego znaczenia. Symbol S zawsze będzie miał określone znaczenie. Intuicyjnie oznacza on operację następnika i zawsze będzie występował w tej roli.

Równością bądź równaniem nazywamy napis postaci

$$f(t_1, \ldots, t_n) = t,$$

gdzie f jest n-arnym symbolem funkcyjnym bez określonego znaczenia (a więc różnym od S), a $t_1, \ldots, t_n, t$ są dowolnymi termami rozważanego języka. Zauważmy, że tak rozumiana równość nie ma własności symetrii. Jej intuicyjna interpretacja jest następująca: obliczenie wartości funkcji f dla argumentów $t_1, \ldots, t_n$ jest równoważne obliczeniu wartości t.

Układem równości (lub równań) nazywamy dowolny ciąg równości. Symbolem głównym układu nazywamy pierwszy zapisany w nim symbol funkcyjny.

Przykładami (pięcioma) układów równań są

$$f(x_1, \ldots, x_n) = g(h_1(x_1, \ldots, x_n), \ldots, h_k(x_1, \ldots, x_n))$$

złożony z jednej równości, zwykle przyjmyjemy, że symbole g, $h_1 \ldots, h_k$ mają określone znaczenie,

$$f(x, 0) = g(x) \quad \text{oraz} \quad f(x, S(y)) = h(x, y, f(x, y)),$$

(dla g i h o określonym znaczeniu),

$$f(0) = S(0) \quad \text{oraz} \quad f(S(x)) = S(S(0)),$$

$$f(0) = S(0) \quad \text{oraz} \quad f(x) = S(S(0)),$$

$$f(S(x)) = S(S(0)).$$

Pierwszy układ złożony z jednego równania jest używany zwykle do określenia złożenia i będzie rzeczywiście definiować złożenie. Drugi – jest standardową definicją przez prostą rekursję. Trzeci jest przykładem definicji warunkowej, powinien definiować funkcję przyjmującą wartość 1 dla 0 i 2 dla pozostałych argumentów. Kolejny – jest przykładem definicji, która będzie uważana za niepoprawną. Ostatni definiuje funkcję częściową określoną dla liczb różnych od 0.

Z tego typu definicje najczęściej jakoś i to dobrze rozumiemy. Dowodzenie ich własności wymaga jednak sprecyzowania ich znaczenia. Z układów równań będziemy wyprowadzać równości – wnioski. W tym celu przyjmiemy trzy reguły wnioskowania:

$$\frac{}{f(\underline{n_1},\ldots,\underline{n_k})=\underline{m}},\quad \frac{r}{r[x\leftarrow\underline{n}]},\quad \frac{r[x\leftarrow f(\underline{n_1},\ldots,\underline{n_k})],\ f(\underline{n_1},\ldots,\underline{n_k})=\underline{m}}{r[x\leftarrow\underline{m}]},$$

przy czym pierwszą z tych reguł stosujemy tylko wtedy, gdy symbol f oznacza pewną funkcję, która argumentom $n_1,\ldots,n_k$ przypisuje wartość m. Druga reguła stwierdzą, że z równania można wywnioskować równanie powstające przez zastąpienie zmiennej x dowolnym termem reprezentującym liczbę naturalną. Ostatnia reguła mówi, że dowolne wystąpienie termu $f(\underline{n_1},\ldots,\underline{n_k})$ można zastąpić przez jego wartość $\underline{m}$ pod warunkiem, że równość $f(\underline{n_1},\ldots,\underline{n_k})=\underline{m}$ też daje się wyprowadzić z rozważanego układu.

Przyjmijmy, że U jest układem równań. Symbolem U^* bedziemy oznaczać najmniejszy zbiór równań zawierający U i zamknięty ze względu na podane reguły wnioskowania. Symbol $U\vdash r$ oznacza, że równanie r daje się wyprowadzić z układu U, a więc, że $r\in U^*$. Każdy układ równań wyznacza pewną relację

$$W_U=\{\langle n_1,\ldots,n_k,m\rangle\in N^{k+1}:U\vdash g_U(\underline{n_1},\ldots,\underline{n_k})=\underline{m}\},$$

gdzie g_U jest głównym symbolem układu U.

Lemat 3.1 *Dla każdego układu równań U relacja W_U jest przeliczalnie rekurencyjna. Wobec tego, jeżeli relacja W_U jest jednoznacza, to jest wykresem pewnej funkcji rekurencyjnej.* □

Zdarza się, że relacja $W_U\subseteq N^{k+1}$ nie jest jednoznaczna. Takie układy nie są interesujące i nie będziemy ich rozważać. Jeżeli jednak W_U jest jednoznaczna, to jest wykresem pewnej funkcji $F_U:N^k\to N$ zdefiniowanej wzorem

$$F_U(n_1,\ldots,n_k)=m\Leftrightarrow\langle n_1,\ldots,n_k,m\rangle\in W_U.$$

Teraz znowu są możliwe dwie sytuacje: albo w układzie występują, oprócz S, inne symbole funkcyjne o ustalonym znaczeniu, albo nie. W pierwszym przypadku o funkcji F_U mówimy, że jest definiowana układem U z funkcjami (które w konkretnych sytuacjach należałoby wymienić i podać odpowiedniość między wymienionymi funkcjami i użytymi symbolami). W drugim, o funkcji F_U mówimy, że jest definiowana układem U (bez funkcji pomocniczych).

Funkcje definiowane układami równań (bez funkcji pomocniczych) nazywamy ogólnie rekurencyjnymi lub rekurencyjnymi według Herbranda-Gödla, a wszystkie takie funkcje tworzą klasę funkcji ogólnie rekurencyjnych lub rekurencyjnych według Herbranda-Gödla.

Zachodzi dość oczywisty

Lemat 3.2 *Klasa funkcji ogólnie rekurencyjnych jest zamknięta ze względu na definiowanie układami z funkcjami. W szczególności, klasa ta jest zamknięta ze względu na złożenie i rekursję prostą.* □

Analogicznie, analizując zwykły układ równości definiujący funkcję Ackermanna pokazuje się

Wniosek 3.3 *Funkcja Ackermanna jest ogólnie rekurencyjna.* □

Wiemy już, że funkcje ogólnie rekurencyjne są (μ)-rekurencyjne. Zachodzi też twierdzenie odwrotne:

Twierdzenie 3.4 *Funkcje μ-rekurencyjne są ogólnie rekurencyjne. W szczególności, funkcje definiowane za pomocą operacji minimum są definiowane układami równości z funkcjami.*

Dowód. Aby dowieść całe twierdzenie wystarczy dowieść jego drugą część.

Przyjmijmy, że

$$F(x) = \mu y\ G(x, y) = 0$$

zakładając dla uproszczenia notacji, że funkcja F jest jednoargumentowa. Zdefiniujemy teraz dwa układy definiujące funkcję F, z symbolem głównym f i symbolem g odpowiadającym funkcji G.

Pierwszy z tych układów ma następującą postać:

$$f(x) = t(i(x, y), g(x, y), y), \quad t(S(x), 0, y) = y, \quad i(x, 0) = S(0),$$

$$i(x, S(y)) = m(i(x, y), g(x, y)),$$

a występujący w nim symbol m odpowiada zwykłemu mnożeniu. Symbole t oraz i oznaczają funkcje pomocnicze w dostatecznym stopniu definiowane przez podany układ.

Drugi układ, też ciekawy, z pomocniczym symbolem h składa się z równań

$$f(x) = h(x, 0, g(x, 0)), \quad h(x, y, 0) = y, \quad h(x, y, S(z)) = h(x, S(y), g(x, S(y))).$$

Uzupełnienie szczegółów w dowodzie pozostawiam jako ćwiczenie. □

3.1.1 Kilka końcowych uwag

Układy równości, które pojawiły się w dowodzie poprzedniego twierdzenia, mało przypominają schematy rekurencyjne. Na przykład, za ogólnie rekurencyjną musimy także uznać funkcję zdefiniowaną równością

$$t(S(x), 0, y) = y$$

lub układem

$$h(y, 0) = y, \quad h(y, S(z)) = h(S(y), g(S(y))).$$

Do wykładu Gödla w Pricenton w 1934 roku chyba nie pojawiają się żadne dziwnie funkcje definiowane rekurencyjnie. Znane było kilka przykładów funkcji z definicjami przypominającymi definicję funkcji Ackermanna. Podczas wykładu sam Gödel podał taki oto przykład mało typowej definicji funkcji rekurencyjnej

$$f(x, 0) = g_1(x), \quad f(0, y + 1) = g_2(y), \quad f(1, y + 1) = g_3(y)$$

$$f(x + 2, y + 1) = h(f(x, y + 2), f(x, f(x, y + 2))),$$

który w najlepszym razie może służyć za wzór definicji rekursji „drugiego stopnia" i nie pokazuje nieznanych możliwości tworzenia nawet zwykłych schematów rekursji. Gödel zdawał sobie sprawę z trudności związanych z podaniem bardzo ogólnego schematu rekurencyjnego (lub ogólnych schematów). Zdecydował się na zaproponowanie niesłychanie ogólnej definicji schematu rekursji, zgodnej z wcześniejszą, bardzo dojrzałą propozycją Herbranda, po jej po doprecyzowaniu. Herbrand zaś okazał się niesłychanie przenikliwym matematykiem o doskonałej intuicji. Nie jest jasne, jaką rolę Herbrand przypisywał swojej definicji, i czy przewidywał jej zastosowanie do formalizacji obliczalności. Zdaniem Gödla raczej nie myślał o obliczalności w dzisiejszym rozumieniu, w tym o obliczalności określanej jako mechaniczna.

Definicja rekurencyjności według Herbranda-Gödla może być rozpatrywana jako koncepcja Prologu funkcyjnego (w przeciwieństwie do Prologu rzeczywistego, który jest predykatywny). W każdym razie, zamiast o funkcji f możemy mówić o jej wykresie W_f i dzięki temu równości możemy przekształcić w klauzule. Na przykład równość

$$f(g(x), S(y)) = h(f(x, S(y)), f(y, x))$$

można zamienić na klauzulę

$$W_f(a, b, w) \ :- \ W_g(x, a),\ W_S(y, b),\ W_h(c, d, w),\ W_f(x, b, c),\ W_f(y, x, d).$$

Jeżeli w ten sposób przekształcimy układ równości definiujący (dwuagumentową) funkcję f, to przynamniej teoretycznie powinniśmy otrzymać program Prologowy, który pozwala testować hipotezy, czy $f(x, y) = z$. Spostrzeżenie to można wykorzystać w dowodzie, że w teoretycznym Prologu daje się przeprowadzić wszelkie możliwe obliczenia.

3.2 Systemy Posta

Emil Post jest prawdopodobnie pionierem badań nad obliczalnością. Urodził się w Białymstoku i w pierwszych latach XX wieku wyemigrował z rodzicami do Stanów Zjednoczonych. Na początku lat dwudziestych zajmował się logiką matematyczną. Jest jednym z logików, którzy dowiedli twierdzenie o pełności dla rachunku zdań. Chciał stworzyć system, który pozwalałby na sformalizowane generowanie praw rachunku logicznego (np. rachunku zdań). Efektem tych prac jest pojęcie systemu Posta. Mimo że prace nad swoimi systemami Post prowadził w latach dwudziestych, podstawowe wyniki zostały opublikowane dopiero w 1943 roku. Systemy Posta mogą być uważane za formalizację obliczalności.

Intuicje Post czerpał z rachunków algebraicznych. Jeżeli przekształcamy duże wyrażenie algebraiczne, to skupiamy się na pewnym fragmencie wyrażenia i przekształcamy ten fragment. Na przykład wyrażenie może mieć postać

$$S_1(x + y)(x - y)S_2,$$

gdzie S_1 i S_2 to fragmenty tego wyrażenia, początkowy i końcowy, które w tym momencie nie mają istotnego znaczenia. Przekształcając, wyrażenie to zastępujemy innym, np. postaci

$$S_1(x^2 - y^2)S_2.$$

System Posta składa się z alfabetu Σ, skończenie wielu aksjomatów i skończenie wielu reguł przekształcania. Operuje na słowach nad swoim alfabetem. Aksjomaty systemu są wybranymi słowami nad jego alfabetem. Reguły są to napisy postaci

$$u_0 S_1 u_1 \ldots S_m u_m \to w_0 S_{i_1} w_1 \ldots S_{i_n} w_n,$$

gdzie $u_0, \ldots, u_m, w_0, \ldots, w_n \in \Sigma^*$ są słowami nad alfabetem Σ, $S_1, \ldots, S_m$ są różnymi zmiennymi (specjalnymi symbolami bez określonego znaczenia), a liczby

$i_1, \ldots, i_n$ są dowolnymi indeksami z przedziału od 1 do m. Taka reguła wyznacza relację $R \subseteq (\Sigma^*)^2$ taką, że

$$\langle x, y\rangle \in R \Leftrightarrow \exists s_1, \ldots s_m \in \Sigma^* \; (x = u_0 s_1 u_1 \ldots s_m u_m \wedge y = w_0 s_{i_1} w_1 \ldots s_{i_n} w_n).$$

Intuicyjnie, relacja R składa się z par słów x i y takich, że słowo x można przekształcić w słowo y zgodnie z regułą wyznaczającą R.

Przypuśćmy, że rozważamy regułę $aS_1bS_2 \to bS_2aS_2$. Wtedy do relacji R odpowiadającej tej regule należą pary $\langle abba, baaa\rangle$ oraz $\langle abba, bbaaba\rangle$. Pary te przedstawiają wszystkie możliwe sposoby przekształcania słowa $abba$. Słowo to może zostać przedstawione w postaci aS_1bS_2 na dwa sposoby, tak aby: $S_1 = b$ i $S_2 = a$ oraz tak, aby $S_1 = \varepsilon$ i $S_2 = ba$.

Każdy system Posta wyznacza pewien zbiór słów X nad swoim alfabetem, a mianowicie najmniejszy zbiór X, do którego należą wszystkie aksjomaty systemu, i do którego wraz ze słowem $x \in X$ należą wszystkie słowa y takie, że $\langle x, y\rangle \in R_i$ dla pewnego i, a R_i oznacza relację wyznaczoną przez i-tą regułę systemu. Zbiór X będziemy nazywać generowanym przez system.

Na przykład, system Posta nad alfabetem $\Sigma = \{a, b\}$ z aksjomatami ε, a i b oraz z regułami $S \to aSa$ oraz $S \to bSb$ generuje zbiór wszystkich palindromów nad alfabetem Σ.

Aby wyjaśnić związek systemów Posta z gramatykami rozważmy gramatykę G generującą również zbiór palindromów. Ta gramatyka korzysta ze zbioru $\Sigma_T = \{a, b\}$ symboli terminalnych, posługuje się symbolami ze zbioru $\Sigma = \Sigma_T \cup \{E\}$, gdzie E jest jedynym, wykorzystywanym w tej gramatyce symbolem nieterminalnym, także symbolem startowym. Aby generować zbiór palindromów potrzebne jest pięć produkcji: $E \to \varepsilon$, $E \to a$, $E \to b$, $E \to aEa$ oraz $E \to bEb$. Tę gramatykę można przekształcić w system Posta nad alfabetem Σ. Jedynym aksjomatem tego systemu jest słowo E. System korzysta z pięciu reguł przekształcania:

$$S_1ES_2 \to S_1S_2, \;\; S_1ES_2 \to S_1aS_2, \;\; S_1ES_2 \to S_1bS_2,$$

$$S_1ES_2 \to S_1aEaS_2, \;\; S_1ES_2 \to S_1bEbS_2.$$

Nietrudno zauważyć, że zbiór X generowany przez ten system Posta składa się z wszystkich słów, które dają się wyprowadzić z symbolu startowego, zawierających także symbole nieterminalne, czyli wszystkich palindromów nad Σ, w których litera E występuje najwyżej jeden raz. Język $L(G)$ generowany przez gramatykę G jest więc równy $X \cap \Sigma_T^*$.

Przedstawiony sposób zamiany gramatyki na system Posta jest bardzo ogólny i pozwala dowieść

Lemat 3.5 *Dla dowolnej gramatyki G z alfabetem terminalnym Σ_T jest taki system Posta, generujący zbiór X, że język*

$$L(G) = X \cap \Sigma_T^*. \;\; \square$$