Motywacja

Poznaliśmy pewien język logiczny zwany rachunkiem zdań. Pozwala on wyrażać fakty proste i złożone (ze spójnikami logicznymi), a dzięki procedurom dowodowym możemy stwierdzić, czy przy zachodzeniu faktów z danego zbioru możemy mieć pewność, że zachodzi również jakiś inny fakt. Na przykład:

$$\text{wygrał-Prokom-Trefl-BOT-Turów} \Rightarrow \text{mistrzostwo-Prokom-Trefl}$$
$$\neg\ \text{wygrał-Prokom-Trefl-BOT-Turów}$$
$$\text{wygrał-BOT-Turów-Prokom-Trefl}$$
$$\text{wynik-BOT-Turów-Prokom-Trefl-86-71}$$

Jednak wyrażanie faktów w ten sposób nie jest wygodne. Konieczne jest wprowadzanie dużej liczby symboli zdaniowych, i nie da się wyrazić żadnych ogólnych zależności. To co byłoby potrzebne, to możliwość wyrażania faktów z argumentami, na przykład:

$$\text{wygrał}(\text{BOT-Turów}, \text{Prokom-Trefl})$$
$$\text{wynik}(\text{BOT-Turów}, \text{Prokom-Trefl}, 86, 71)$$
$$\forall x, y \;\; \text{wygrał}(x, y) \Rightarrow \neg\text{wygrał}(y, x)$$

Rachunek predykatów — dziedzina i predykaty

Definicja 1. (Dziedzina rachunku predykatów)
W *rachunku predykatów* używamy *zmiennych indywiduowych* pochodzących z nieskończonego zbioru $\mathcal{V} = \{x, y, z, x_1, x_2, \ldots\}$, *symboli funkcyjnych* $\mathcal{F} = \{f, g, f_1, f_2, \ldots\}$ i *symboli relacyjnych* $\mathcal{P} = \{p, q, p_1, p_2, \ldots\}$. Symbole relacyjne nazywamy również *predykatami*.

Z każdym symbolem funkcyjnym i z każdym predykatem związujemy liczbę naturalną, która określa liczbę jego argumentów (*arność*). Symbole funkcyjne 0-argumentowe nazywamy *symbolami stałych*. Predykaty można uważać za uogólnienie zmiennych zdaniowych z rachunku zdań (zmienne zdaniowe, to 0-argumentowe predykaty).

Przykłady:

wygrał(.,.)	-	dwuargumentowy symbol relacyjny (predykat)
PLK-sezon-zakończony	-	0-argumentowy predykat (formuła rachunku zdań)
kapitan(.)	-	1-argumentowy symbol funkcyjny (określa osobę)
ASCO-Śląsk	-	0-argumentowy symbol funkcyjny

Rachunek predykatów — termy

Definicja 2. *Termy* są napisami postaci x, $f(x)$, $g(f_1(x), f_2(x))$ itp. Formalnie:

1. Każda zmienna indywiduowa jest termem.
2. Każdy symbol funkcji 0-argumentowej (stałej) jest termem.
3. Jeśli $t_1, \ldots, t_n$ są termami, a f jest n-argumentowym $(n > 0)$ symbolem funkcyjnym, to $f(t_1, \ldots, t_n)$ jest termem.
4. Każdy term można zbudować przy pomocy reguł 1–3.

Przykłady:

x	-	zmienna jest termem
Andrej-Urlep	-	stała też jest termem (konkretnym)
kapitan(x)	-	term z symbolem funkcyjnym (określa jakąś osobę)
kapitan(ASCO-Śląsk)	-	inny term funkcyjny (określa konkretną osobę)

Rachunek predykatów — formuły atomowe

Definicja 3. *Formuły atomowe* są napisami postaci $p(t_1)$, $q(t_1, t_2)$ itp., gdzie t_1 i t_2 są termami. Formalnie:

1. Symbole $\perp$ i $\top$ są formułami atomowymi.
2. Każdy 0-argumentowy predykat jest formułą atomową.
3. Jeśli $t_1, \ldots, t_n$ są termami, a p jest n-argumentowym $(n > 0)$ predykatem, to $p(t_1, \ldots, t_n)$ jest formułą atomową.
4. Każdą formułę atomową można otrzymać przy pomocy reguł 1–3.

Przykłady:

PLK-sezon-zakończony	-	0-argumentowy predykat
wygrał(Prokom-Trefl,BOT-Turów)	-	konkretna formuła (prawdziwa?/fałszywa?)
wygrał(Prokom-Trefl,x)	-	formuła *otwarta*

Rachunek predykatów — kwantyfikatory i formuły

Definicja 4. *Formuły* rachunku predykatów (oznaczane $\varphi, \psi, \ldots$) budujemy z formuł atomowych za pomocą spójników logicznych w sposób podobny, jak w rachunku zdań. Ponadto możemy używać kwantyfikatorów $\forall$ i $\exists$. Formalnie:

1. Każda formuła atomowa jest formułą rachunku predykatów.
2. Jeżeli φ_1 i φ_2 są formułami rachunku predykatów, to są nimi także: $(\neg\varphi_1)$, $(\varphi_1 \wedge \varphi_2)$, $(\varphi_1 \vee \varphi_2)$, $(\varphi_1 \Rightarrow \varphi_2)$ i $(\varphi_1 \Leftrightarrow \varphi_2)$.
3. Jeżeli x jest zmienną indywiduową, a φ — formułą rachunku predykatów, to formułami rachunku predykatów są też $(\forall x\, \varphi)$ i $(\exists x\, \varphi)$.
4. Każdą formułę rachunku predykatów można otrzymać przy pomocy reguł 1–3.

Dla zwiększenia czytelności opuszczamy niektóre nawiasy w podobny sposób, jak w rachunku zdań.

Przykład:

$\forall x, y$ wygrał$(x,y) \Rightarrow \neg$ wygrał(y,x)

Rachunek predykatów — zmienne wolne i związane

Definicja 5. Mówimy, że w formule $\forall x\,\varphi$ (lub $\exists x\,\varphi$) kwantyfikator $\forall$ (lub $\exists$) *wiąże* wystąpienia zmiennej x w formule φ, oraz że wystąpienia zmiennej x w formule φ są *związane* przez ten kwantyfikator. Wystąpienia zmiennych, które nie są związane w danej formule, są w niej *wolne*. Formalnie zbiór $\mathrm{FV}(\varphi)$ zmiennych, które występują jako wolne w formule φ definiujemy następująco:

$$\begin{aligned}
\mathrm{FV}(x) &= \{x\} \\
\mathrm{FV}(f(t_1,\ldots,t_n)) &= \mathrm{FV}(t_1) \cup \ldots \cup \mathrm{FV}(t_n) \\
\mathrm{FV}(\bot) &= \emptyset \\
\mathrm{FV}(\top) &= \emptyset \\
\mathrm{FV}(p(t_1,\ldots,t_n)) &= \mathrm{FV}(t_1) \cup \ldots \cup \mathrm{FV}(t_n) \\
\mathrm{FV}(\neg\varphi) &= \mathrm{FV}(\varphi) \\
\mathrm{FV}(\varphi \circ \psi) &= \mathrm{FV}(\varphi) \cup \mathrm{FV}(\psi), \quad \text{gdzie } \circ \in \{\vee, \wedge, \Rightarrow, \Leftrightarrow\} \\
\mathrm{FV}(\forall x\,\varphi) &= \mathrm{FV}(\varphi) \setminus \{x\} \\
\mathrm{FV}(\exists x\,\varphi) &= \mathrm{FV}(\varphi) \setminus \{x\}
\end{aligned}$$

Podobnie definiujemy zbiór zmiennych występujących w formule jako związane.

Przykład

Zastosujemy rachunek predykatów do zapisania pewnych faktów i pojęć z teorii mnogości, czyli zbiorów i ich elementów. Symbole zmiennych $A, B, C, \ldots$ będą oznaczały zbiory, a symbole $x, y, z, \ldots$ elementy tych zbiorów. Wszystkie symbole zmiennych są termami. Operacje wykonywane na zbiorach, takie jak $\cup$ albo $\cap$ są w rzeczywistości funkcjami w dziedzinie termów, określającymi dla dowolnych dwóch termów jakiś inny (nowy) term. Natomiast symbole $\in$ i $\subseteq$, oznaczające relacje należenia elementu do zbioru, i zawierania się zbiorów, są predykatami.

Tradycyjne zapisy $A \cap B$, i $A \subseteq B$ możemy traktować jako skróty zapisów $\cap(A, B)$, i $\subseteq (A, B)$. Zatem przekrój jest dwuargumentową funkcją termową, a zawieranie jest dwuargumentowym predykatem.

Poniższą formułę możemy uważać za definicję relacji zawierania się zbiorów:

$$\forall A \forall B \;\; (\forall x \; x \in A \Rightarrow x \in B) \Leftrightarrow A \subseteq B$$

Natomiast kolejna formuła jest definicją funkcji tworzenia przekroju zbiorów:

$$\forall x \forall A \forall B \;\; (x \in A \;\wedge\; x \in B) \Leftrightarrow x \in A \cap B$$

Możemy też wprowadzić relację równości $=$ i symbol stały $\emptyset$ (formalnie traktowany jako 0-argumentowa funkcja). Rozważmy formułę:

$$\forall A \quad A = \emptyset$$

Formuła wydaje się w oczywisty sposób fałszywa, ale jak możemy to stwierdzić?

Interpretacje

Podobnie jak w przypadku rachunku zdań wprowadzamy pojęcie *interpretacji* I, która dla zbioru formuł U z symbolami relacyjnymi $\{p_1, p_2, ...p_k\} \subseteq \mathcal{P}$, i symbolami funkcyjnymi $\{f_1, f_2, ...f_l\} \subseteq \mathcal{F}$, jest trójką:

$$I = (D, \{R_1, R_2, ...R_k\}, \{F_1, F_2, ...F_l\})$$

gdzie D jest pewną niepustą *dziedziną interpretacji* I, każde R_i jest relacją określoną na D przyporządkowaną symbolowi relacyjnemu $p_i \in \mathcal{P}$, a F_i jest funkcją określoną na D przyporządkowaną symbolowi funkcyjnemu $f_i \in \mathcal{F}$.

Interpretacje pozwalają nam związać symbole występujące w rachunku predykatów z konkretnymi obiektami z pewnej dziedziny. Na przykład, pewna interpretacja I_1 mogłaby przyporządkować termowi BOT-Turów pewną drużynę koszykarską należącą do Polskiej Ligi Koszykówki. Jednak pamiętajmy, że możliwe są również inne interpretacje, np. interpetacja I_2 mogłaby przyporządkować termowi BOT-Turów wykładowcę niniejszego kursu, albo Kaczora Donalda, albo jeszcze coś innego.

Przykłady interpretacji

przykład 5.11 z B-A, str.117

Wartościowanie dla termów

Mając pojęcie interpretacji możemy wprowadzić związane z nią wartościowanie termów. Odróżnimy tutaj symbole funkcyjne 0-argumentowe, którym wartościowanie danej interpretacji przyporządkowuje obiekty z dziedziny D:

$$\sigma_I(f_i) = d_i$$

od symboli funkcyjnych 1- lub więcej argumentowych, którym wartościowanie przyporządkowuje rzeczywiste wyrażenia funkcyjne określone w dziedzinie interpetacji:

$$\sigma_I(f_i(t_1, \ldots, t_n)) = F_i(\sigma_I(t_1), \ldots, \sigma_I(t_n))$$

Wartościowanie dla formuł

Dla formuł wprowadzamy funkcję wartościowania formuł w_σ przyporządkowującą dowolnej formule wartość 1 lub 0 zależnie od tego, czy relacje w dziedzinie interpretacji, odpowiadające predykatom występującym w formule są spełnione dla elementów z dziedziny odpowiadających termom z formuły, czy nie.

Definicja wartościowania formuł jest podobna jak w przypadku formuł rachunku zdań, to znaczy określa wartość formuły złożonej dla wszystkich symboli spójników logicznych, i dodatkowo dla formuł z kwantyfikatorami.

Jednak nie określamy wartościowania dla formuł otwartych, czyli zawierających niezwiązane zmienne. Ich wartość może po prostu zależeć od wartościowania tych zmiennych, więc w ogólnym przypadku nie może być określona.

Rachunek predykatów — inny przykład

Teraz zastosujemy rachunek predykatów do zapisu faktów z arytmetyki liczb całkowitych. Będziemy stosować symbole zmiennych x, y, z... na oznaczenie liczb całkowitych. Zbiorem symboli funkcyjnych będzie $F = \{+, -, *, /\}$. Zgodnie z przyjętą konwencją możemy stosować symbole stałych, takie jak: **0, 1, -1, . . .** , ale są one traktowane jako 0-argumentowe funkcje. Natomiast zbiorem symboli relacyjnych będzie $P = \{=, <, \leq, >, \geq\}$.

Wszystkie symbole zmiennych są oczywiście termami. Są nimi również wyrażenia typu $x + y$, które jest w rzeczywistości skrótowym zapisem termu $+(x, y)$. Zauważmy, że w tej drugiej notacji nie ma problemu priorytetów operatorów i stosowania dodatkowych nawiasów, ponieważ term $x * y + z$ jest zapisem niejednoznacznym, i ma dwie wersje: $+(x, *(y, z))$ i $*(x, +(y, z))$.

Natomiast wyrażenia typu: $x \leq y$ są skróconym zapisem formuł atomowych z odpowiednim symbolem relacyjnym, w tym przypadku: $\leq (x, y)$. Zauważmy, że formuła atomowa nie może zawierać więcej niż jednego symbolu relacyjnego. To znaczy, zapis typu: $x < y < z$ musimy traktować jako formułę złożoną (nieatomową), będącą skrótem zapisu: $< (x, y) \wedge < (y, z)$. Ponieważ zaś nasz zbiór symboli relacyjnych nie zawiera nierówności, więc formułę: $x \neq y$ musimy zapisywać jako: $\neg(= (x, y))$.

Możemy zapisywać formuły rachunku predykatów:

$$\forall x\ \forall y\ \exists z\ x + y = z$$
$$\forall x\ \forall y\ (x < y) \Leftrightarrow (2 * x < 2 * y)$$
$$\forall x\ x = 5$$

Interpretując symbole występujące w tych zapisach jako odpowiadające symbolom z normalnej arytmetyki liczb całkowitych, możemy stwierdzić, że pierwsze dwie formuły są prawdziwe, a ostatnia jest fałszywa. Bardziej precyzyjnie, należy stwierdzić, że pierwsze dwie formuły są spełnione przez tę interpretację, a ostatnia formuła nie.

W ogólności jednak, żadna z tych formuł nie jest ani niespełnialna, ani nie jest tautologią, ponieważ można podać inne interpretacje dla nich, dające inne wartościowanie termów i formuł.

Tautologie i modele

Będziemy więc posługiwali się w rachunku predykatów pojęciami spełnialności, niespełnialności, tautologii, i modelu, analogicznie jak w rachunku zdań.

Definicja 6. Formuła zamknięta φ jest *spełniona* w interpretacji I, jeśli $w_{\sigma_I}(\varphi) = 1$, co oznaczamy $I \models \varphi$. Mówimy wtedy, że interpretacja I jest modelem formuły φ.

Przyklady: B-A 5.17, 5.19, str 119

Równoważność logiczna i wynikanie logiczne

Podobnie jak w rachunku zdań będziemy stosowali pojęcia równoważności logicznej i wynikania logicznego.

Definicja 7. Jeśli φ i ψ są formułami zamkniętymi i $w_{\sigma_I}(\varphi) = w_{\sigma_I}(\psi)$ dla każdej interpretacji I, to φ i ψ są *logicznie równoważne*, co zapisujemy $\varphi \equiv \psi$.

Jeśli U jest zbiorem formuł zamkniętych φ_i, a ψ jest formułą zamkniętą, i dla wszystkich interpretacji I, dla których wszystkie formuły $\varphi_i \in U$ mają wartość logiczną $w_{\sigma_I}(\varphi_i) = 1$, formuła ψ ma również wartość logiczną $w_{\sigma_I}(\psi) = 1$, to mówimy, że formuła ψ wynika logicznie ze zbioru formuł U, co zapisujemy $U \models \psi$.

Dowodzenie prawdziwości formuł rachunku predykatów

Fakt:

$$\varphi \equiv \psi \quad \text{wtw, gdy} \quad \models \varphi \Leftrightarrow \psi$$

$$U \models \psi \quad \text{wtw, gdy} \quad \models (\varphi_1 \wedge \ldots \wedge \varphi_n) \Rightarrow \psi$$

Pierwsza z powyższych zależności pozwala nam, w celu wykazania, że jakaś formuła φ jest prawdziwa, wykazać, że inna formuła ψ jest prawdziwa, o ile tylko $\varphi \Leftrightarrow \psi$ jest tautologią.

Tautologie rachunku predykatów: B-A rysunek 5.2, str. 121

Dowodzenie prawdziwości formuł: B-A przyklad 5.23, str. 122

Postać klauzulowa formuł

Dążymy do przedstawienia formuł rachunku predykatów w ujednoliconej postaci normalnej, pozwalającej na skonstruowanie maksymalnie automatycznej procedury dowodzenia, podobnie jak w rachunku zdań. Taką postacią będzie koniunkcyjna postać normalna CNF, w której formuła przedstawiana jest jako koniunkcja klauzul, które same są alternatywami literałów (formuł atomowych z negacją lub bez).

Procedura eliminacji spójników i przekształcania formuły może tu być podobna jak w przypadku rachunku zdań, jednak w rachunku predykatów dodatkowym elementem są kwantyfikatory.

Przekształcanie formuł do postaci prenex

- przemianowanie nazw zmiennych związanych kwantyfikatorami
- eliminacja spójników logicznych poza $\neg, \vee, \wedge$
- przesunięcie negacji do symboli predykatów
- wyciągnięcie kwantyfikatorów na początek formuły
- przekształcenie do koniunkcyjnej postaci normalnej (CNF)
- skolemizacja

Przykłady: B-A w algorytmie 7.13 str. 162-163, i przykład 7.14

Twierdzenie (Skolem): formuła Φ', otrzymana w wyniku powyższego przekształcenia formuły Φ, jest spełnialna wtedy i tylko wtedy gdy spełnialna jest formuła Φ.

Podstawienia zmiennych

Podstawienie – zbiór par uporządkowanych definiujących termy podstawiane pod poszczególne zmienne; wszystkie wystąpienia jednej zmiennej w wyrażeniu muszą być jednakowo podstawione i podstawienie nie może zawierać podstawianej zmiennej

Złożenie $s_1 s_2$ podstawień s_1 i s_2 – podstawienie uzyskane przez zastosowanie podstawienia s_2 do termów z s_1, oraz dopisanie do s_1 wszystkich par z s_2 ze zmiennymi nie wystepującymi w s_1

$$Ls_1 s_2 = (Ls_1)s_2$$

$$s_1(s_2 s_3) = (s_1 s_2)s_3$$

Unifikacja

Unifikacja – znajdowanie podstawienia termów pod zmienne dla uzyskania identycznych formuł.

Unifikator zbioru formuł – podstawienie redukujące zbiór do jednej formuły.

Najogólniejszy unifikator (mgu) – mozna z niego uzyskać dowolny inny

Algorytm unifikacji – oparty na dopasowywaniu lecz musi pozwalać na obecność zmiennych w obu dopasowywanych wyrażeniach.

Rezolucja — przypadek klauzul podstawionych

Rezolucja dla dwóch klauzul podstawionych: gdy istnieje wspólny literal mający w dwóch klauzulach przeciwne znaki, to rezolucja tworzy nową klauzulę będącą połączeniem tamtych dwóch z wyłączeniem tego literala:

$$P \vee Q(A) \text{ oraz } \neg S \vee \neg Q(A) \quad \rightsquigarrow \quad P \vee \neg S$$

Ciekawe przypadki szczególne:

$$\begin{array}{rll}
P \text{ oraz } \neg P \vee Q & \rightsquigarrow Q & \text{modus ponens} \\
P \vee Q \text{ oraz } \neg P \vee Q & \rightsquigarrow Q & \\
P \vee Q \text{ oraz } \neg P \vee \neg Q & \rightsquigarrow P \vee \neg P & \text{tautologia} \\
P \vee Q \text{ oraz } \neg P \vee \neg Q & \rightsquigarrow Q \vee \neg Q & \text{-''-} \\
P \text{ oraz } \neg P & \rightsquigarrow NIL & \text{sprzeczność} \\
\neg P \vee Q \text{ oraz } \neg Q \vee R & \rightsquigarrow \neg P \vee R & \text{przechodniość} \\
(P \Rightarrow Q) \qquad (Q \Rightarrow R) & \quad (P \Rightarrow R) & \text{implikacji}
\end{array}$$

Rezolucja — przypadek ogólny

Rezolucja w ogólnym przypadku: gdy dla dwóch klauzul (zbiorów literali) $\{L_i\}$ i $\{M_i\}$ istnieją ich podzbiory $\{l_i\}$ i $\{m_i\}$ takie, że zbiór $\{l_i\} \cup \{\neg m_i\}$ daje się zunifikować i s jest jego mgu, wtedy ich rezolwentą jest zbiór $[\{L_i\} - \{l_i\}]s \cup [\{M_i\} - \{m_i\}]s$

Moga istnieć różne rezolwenty danych klauzul dla różnych wyborów podzbiorów ich literali. Na przykład, rozważmy następujące klauzule:

$$P[x, f(A)] \vee P[x, f(y)] \vee Q(y) \text{ oraz } \neg P[z, f(A)] \vee \neg Q(z)$$

Wybierając $\{l_i\} = \{P[x, f(A)]\}$ oraz $\{m_i\} = \{\neg P[z, f(A)]\}$, dla $s = \{x/z\}$ otrzymujemy rezolwentę:

$$P[z, f(y)] \vee Q(y) \vee \neg Q(z)$$

Natomiast dla $\{l_i\} = \{P[x, f(A)], P[x, f(y)]\}$, $\{m_i\} = \{\neg P[z, f(A)]\}$, oraz $s = \{x/z, y/A\}$ otrzymujemy:

$$Q(A) \vee \neg Q(z)$$

Dowodzenie twierdzeń — przykład

Rozważmy przykład z matematyki. Chcemy udowodnić, że przekrój dwóch zbiorów zawiera się w dowolnym z nich. Zaczynamy od wypisania aksjomatów, z których rozumowanie będzie musiało korzystać. W tym przypadku są to definicje pojęć przekroju i zawierania się zbiorów.

$$\forall x \forall s \forall t \;\; (x \in s \;\wedge\; x \in t) \Leftrightarrow x \in s \cap t$$
$$\forall s \forall t \;\; (\forall x \; x \in s \Rightarrow x \in t) \Leftrightarrow s \subseteq t$$

Formuła do udowodnienia:

$$\forall s \forall t \;\; s \cap t \subseteq s$$

Po przetworzeniu do postaci klauzul otrzymujemy:

1.	$\{x \notin s, x \notin t, x \in s \cap t\}$	z definicji przekroju
2.	$\{x \notin s \cap t, x \in s\}$	z definicji przekroju
3.	$\{x \notin s \cap t, x \in t\}$	z definicji przekroju
4.	$\{F(s,t) \in s, s \subseteq t\}$	z definicji zawierania się
5.	$\{F(s,t) \notin t, s \subseteq t\}$	z definicji zawierania się
6.	$\{A \cap B \not\subseteq A\}$	z zaprzeczenia tezy

Zauważmy funkcje Skolema w klauzulach 4 i 5, oraz stałe Skolema s klauzuli 6. Dalej następuje wywód prowadzący dosyć prosto do klauzuli pustej.

7.	$\{F(A \cap B, A) \in A \cap B\}$	z klauzul 4. i 6.
8.	$\{F(A \cap B, A) \notin A\}$	z klauzul 5. i 6.
9.	$\{F(A \cap B, A) \in A\}$	z klauzul 2. i 7.
10.	$\{\}$	z klauzul 8. i 9.

To koniec dowodu. Cel osiągnięty. Trochę trudno poczuć satysfakcję jaka zwykle towarzyszy osiągnięciu tradycyjnego dowodu matematycznego. Trudno też prześledzić rozumowanie i np. je sprawdzić.

Nierozstrzygalność rachunku predykatów

Twierdzenie (Alonzo Church, 1936): rachunek predykatów jest nierozstrzygalny, to znaczy, nie istnieje procedura pozwalająca stwierdzić, czy dane zdanie jest prawdziwe.

Co to oznacza w praktyce, skoro dowodzenie twierdzeń jest możliwe? Procedura dowodowa działa w pętli, w każdym przebiegu tej pętli generuje nowe formuły, i zatrzymuje się dopiero wtedy, gdy wygeneruję fałsz (pustą klauzulę). I po prostu nie ma gwarancji, że ta procedura kiedykolwiek się zatrzyma.

Twierdzenie (Gödel): Teoria liczb naturalnych, czyli rachunek predykatów ze stałą 0, pojedynczy symbol funkcyjny s (funkcja następnika), i pojedynczy symbol relacyjny $=$, nie jest zupełna.

Definicja 8. Teoria $\mathcal{T}(\mathcal{U})$ jest *zupełna* wtw. gdy $U \vdash A$ lub $U \vdash \neg A$ dla każdej zamkniętej formuły A