czywistych, a zwykły platonizm jest odpowiedni dla geometrycznej teorii kontinuum. Stanowisko takie jest szczególnie widoczne w *Sur le platonisme en mathématique*, gdzie pojawia się również podsumowanie i ocena sporów filozoficznych o naturę matematyki w pierwszej połowie XX wieku. W większości zebranych tekstów Bernays wskazuje na konieczność akceptacji różnorodnych punktów widzenia w procesie fundowania poszczególnych dziedzin matematyki.

Metanaukowe stanowisko P. Bernaysa jest koncepcją nadal aktualną dobrze wpisującą się we współczesne dyskusje w ramach filozofii matematyki. Interesuje go nie tylko kontekst uzasadnienia, ale też zagadnienie praktyki badawczej w matematyce. Uwzględnia aspekt historyczny matematyki oraz dostrzega jej cechy wspólne z naukami przyrodniczymi. Rozpatruje *quasi*-empiryczny punkt wyjścia w procesie konstytuowania się nauk matematycznych i przyrodniczych.

Prezentowana francuskojęzyczna publikacja jest pozycją niezwykle cenną ze względu na kontekst formalnych i ideowych związków jej autora z czołowymi przedstawicielami frankofońskiej filozofii nauki.

Jerzy Kaczmarek
Katedry Filozofii Przyrody Nieożywionej KUL

Mordechai B e n - A r i, *Logika matematyczna w informatyce* [oryg. *Mathematical Logic for Computer Science*], tł. z ang. M. Miłkowska, Warszawa: Wydawnictwa Naukowo-Techniczne 2005, ss. 343. Seria: Klasyka Informatyki. ISBN 83-204-2972-2.

Logika formalna znajduje szerokie zastosowanie w różnych dziedzinach informatyki. Dwuwartościowy klasyczny rachunek zdań wykorzystywany jest do opisu budowy komputerów, w szczególności zaś do projektowania układów cyfrowych za pomocą bramek logicznych. Rachunek predykatów stosowany jest do formalnego opisu semantyki języków programowania. Swoje miejsce znajduje również w specyfikacji i weryfikacji programów komputerowych. Jest to ważna i istotna rola systemów sformalizowanych, gdyż inaczej nie można stwierdzić, że napisany przez programistę program komputerowy wykonuje dokładnie to, co opisuje algorytm. Innym istotnym polem badań logiczno-informatycznych jest programowanie w logice i automatyczne dowodzenie twierdzeń. Badania w tej dziedzinie sztucznej inteligencji doprowadziły do opracowania rezolucji – nowej metody dowodzenia twierdzeń rachunku predykatów. Pewne odmiany rezolucji wykorzystano jako podstawę do stworzenia języków programowania, z których najbardziej popularnym jest Prolog.

Logiki nieklasyczne również odgrywają istotną rolę w inżynierii oprogramowania. Szczególnie stosuje się logiki temporalne i modalne do opisu sytuacji w programach

reaktywnych, takich jak systemy operacyjne oraz systemy czasu rzeczywistego, które nie generują jednej odpowiedzi, ale działają ciągle. Istotną rolą logiki temporalnej jest opis dynamicznego zachowania się układów cyfrowych.

Niewiele spośród dostępnych podręczników logiki dla informatyków łączy podejście ściśle matematyczne z podejściem informatycznym. Zadanie połączenia tych ujęć samo w sobie jest trudne i wymaga dużej wiedzy z obu dziedzin oraz wytrwałości w znalezieniu odpowiedniej formy dydaktycznej, ułatwiającej zrozumienie materiału przez studenta. Odrębny kurs logiki dla studentów informatyki wydaje się być konieczny z powodu jej licznych i podstawowych zastosowań w informatyce oraz z powodu często niezauważanego, a jednak istotnego, mianowicie podejścia samych studentów do przedmiotów formalnych. Duża część studentów podchodzi do informatyki jako dziedziny czysto rzemieślniczej. Wydaje się im, że umiejętność pisania kodu w kilku językach programowania jest wystarczająca do „bycia" informatykiem, a nauki formalne nie są potrzebne. Jest to oczywisty przesąd związany z brakiem solidnych podstaw matematyki w szkole średniej, a w konsekwencji brakiem rozumienia zagadnień matematycznych.

Wyraźny brak odpowiedniego podręcznika do nauki logiki próbuje wypełnić recenzowana publikacja. Jej Autor twierdzi, że przeznaczona jest ona dla studentów pierwszego roku studiów. Jego zdaniem praca z recenzowanym podręcznikiem – poza znajomością kilku podstawowych pojęć teorii mnogości – nie wymaga uprzedniej wiedzy z zakresu matematyki. Wymagana jest natomiast wiedza informatyczna dotycząca grafów, języków programowania i programów. Autor w swoim wykładzie proponuje metodę tablic semantycznych jako najbardziej zrozumiałą i przystępną dla studenta. Czy rzeczywiście podręcznik spełnia wymagania polskiego czytelnika i jest adekwatny dla kursu logiki na pierwszym roku studiów informatycznych? Można spróbować odpowiedzieć na to pytanie tylko przez wgląd w treść i metodę wykładów.

Podręcznik podzielony jest na dwanaście rozdziałów zgrupowanych wokół pięciu tematów: rachunek zdań, rachunek predykatów, programowanie w logice, specyfikacja i weryfikacja programów oraz logiki temporalne. Ogólna struktura przedstawiania tematów jest czteroczłonowa: składnia i semantyka omawianego rachunku, metoda tablic semantycznych, systemy dowodzenia oraz algorytmy implementujące przedstawiane metody w programie komputerowym.

Rozdział pierwszy stanowi wprowadzenie w tematykę podręcznika. Na samym początku zawiera on wstęp historyczny, któremu warto poświęcić parę słów. Autor wspomina, że logika bierze swój początek w starożytnej Grecji, nie wspomina natomiast, kto był „ojcem" logiki. W tym kontekście zostaje zamieszczony następujący przykład reguły wnioskowania, zwany przez autora „sylogizmem".

Przesłanka: Wszyscy ludzie są śmiertelni.
Przesłanka: *X* jest człowiekiem.
Wniosek: A zatem *X* jest śmiertelny.

Warto zauważyć, że powyższy sylogizm nie jest sylogizmem w systemie zawartym w *Analitykach Pierwszych.* Takie sylogizmy wprowadził dopiero W. Ockham, a jak powszechnie wiadomo, ten autor nie tworzył w starożytności[1]. Wspomnieć należy, że przykład złego „sylogizmu" jest następujący:

Przesłanka: Jakieś samochody terkoczą.
Przesłanka: Mój samochód jest jakimś samochodem.
Wniosek: A zatem mój samochód terkocze.

Ten przykład w ogóle nie powinien być nazwany „sylogizmem". Na takie niuanse w podręcznikach należy zwrócić uwagę, gdyż tego typu błędy wydają się być powszechne w pozycjach wydawanych w krajach anglosaskich. Pomimo faktu, że podręcznik nie jest pisany dla humanistów, warto poświęcić więcej uwagi i ścisłości kwestii historii logiki. Poza uwagami natury historycznej autor omawia poszczególne działy podręcznika.

Rozdział drugi i trzeci omawia klasyczny rachunek zdań. Jest on wykładany nietypowo, gdyż w niektórych zagadnieniach autor posługuje się narzędziami zaczerpniętymi z informatyki. Na początku wykładu klasycznego rachunku zdań zostaje sformułowane pojęcie wyrażenia poprawnie zbudowanego. Do tego celu używana jest gramatyka bezkontekstowa, podobna do notacji BNF[2]. Wykorzystuje się również drzewa wywodu i drzewa struktury do wyprowadzenia i reprezentacji wyrażeń poprawnie zbudowanych. Następnie zostaje wprowadzona metoda matrycowa sprawdzania wyrażeń, jak również wprowadzone są ważniejsze twierdzenia o rachunku zdań. W dalszej części rozdziału drugiego czytelnik zapoznany jest z metodą konstruowania tabel semantycznych i dowodem pełności. Pokazana jest również implementacja metody matrycowej i tablic semantycznych w języku Prolog.

W rozdziale trzecim wprowadzone są kolejno gentzenowski i hilbertowski sytem dowodzenia oraz zaprezentowany jest program komputerowy weryfikujący poprawność dowodów w systemie hilbertowskim. Następnie wymienione są różne odmiany powyższych systemów.

Rozdział czwarty omawia zagadnienia postaci normalnych i klauzulowych wyrażeń rachunku zdań. W dalszej części wprowadza się regułę rezolucji i metodę tworzenia drzew semantycznych oraz ich implementację w programie komputerowym. Następnie wprowadza się czytelnika w tematykę diagramów binarnych decyzji, które operują na strukturach badanych wyrażeń. Podaje się również algorytmy optymalizujące wyżej wspomnianą metodę. Na końcu rozdziału czytelnik znajduje wpro-

[1] Por. M. T k a c z y k, *Metoda dowodzenia przez ektezę u Arystotelesa i Wilhelma Ockhama*, „Lignum Vitae" 6 (2006), s. 222-223.

[2] Ang. *Backus-Naur Form* – system zapisu gramatyki języka programowania Algol 60.

wadzenia do problematyki złożoności obliczeniowej, zawierającą informacje o obliczeniowych ograniczeniach systemów opartych na regule rezolucji.

W rozdziale piątym omówiony jest rachunek predykatów. Sposób prezentacji tej gałęzi logiki jest analogiczny jak klasycznego rachunku zdań. Wprowadzone zostaje pojęcie modelu skończonego i nieskończonego. Nowym omawianym zagadnieniem jest problem rozstrzygalności rachunku predykatów. Autor przedstawia w nim dowód twierdzenia Churcha o nierozstrzygalności tego rachunku.

Rozdział szósty zawiera szczegółowe omówienie systemów dowodzenia dla rachunku predykatów, jak również ich implementację w Prologu. Wprowadzone zostaje pojęcie teorii zupełnych oraz rozstrzygalnych.

Rozdział siódmy w sposób szeroki omawia zagadnienia związane z regułą rezolucji dla rachunku predykatów. Pokazany jest algorytm skolemizacji wyrażeń oraz dowód twierdzenia Skolema o spełnialności dwóch formuł. Następnie czytelnik zapoznawany jest z modelami Herbranda. Na końcu rozdziału omówiona jest ogólna reguła rezolucji dla rachunku predykatów. Przedstawione są również pojęcia podstawiania termów za zmienne i uzgadniania klauzul o nieustalonych termach.

Kolejne trzy rozdziały traktują o praktycznych zastosowaniach logiki w informatyce. Rozdział ósmy omawia implementację rachunku predykatów opartego na regule rezolucji w deklaratywny język programowania w logice. Wprowadzone zostaje pojęcie wyrażenia rachunku predykatów jako programu do wykonania. Reguła rezolucji zostaje zmodyfikowana do postaci tzw. SLD-rezolucji. Rezultatem implementacji wyżej wspomnianej reguły jest język programowania Prolog, w którym programista nie tyle określa, co program ma wykonać, ale ustala konieczne związki logiczne między danymi wejściowymi a wyjściowymi. Stworzenie algorytmu rozwiązującego dany problem sprowadza się do jego logicznego opisu (formalizacji). W ramach programowania w logice omówiono programowanie współbieżne, mające na celu przyspieszenie wykonania obliczeń poprzez podział zadania na kilka części i przetworzenia ich na kilku procesorach.

W rozdziale dziewiątym przedstawiony został formalny system definiowania semantyki języka programowania. Omówiono również formalne pojęcie poprawności programu. Wprowadzono też system aksjomatyczny stosowany w dowodzeniu poprawności programów komputerowych. Rozdział dziesiąty omawia język *Z*, który jest używany do wyrażania specyfikacji programu, czyli określania tego, jak program powinien działać i jakie dane wyjściowe uzyskać.

Rozdziały jedenasty i dwunasty zawierają wykład logiki temporalnej. Wprowadzone zostają funktory temporalne logiki zdaniowej: „zawsze”, „kiedyś”, „następny” oraz zostaje przedstawiona ich interpretacja. Następnie buduje się logikę czasu liniowego metodą tabel semantycznych oraz jej implementację w programie komputerowym.

Rozdział dwunasty przedstawia różne systemy logiki temporalnej pod kątem ich zastosowań w technice komputerowej. Jako pierwsza zostaje wprowadzona logika *L* z pierwotnymi funktorami „zawsze” i „następny”. Następnie czytelnik zostaje zapo-

znany z innymi systemami definiującymi relacje poprzedzania „*p* zachodzi przed *q*" i przeszłości „odkąd" oraz „wstecz" (ang. *back-to*). Prezentowana jest również logika czasu rozgałęzionego. Po zapoznaniu czytelnika z wyżej wspomnianymi systemami Autor przechodzi do omówienia specyfikacji i weryfikacji programów współbieżnych, specyfikacji i weryfikacji układów sprzętowych oraz implementacji tych metod w Prologu.

Na końcu podręcznika znajduje się dodatek przedstawiający ważniejsze pojęcia teoriomnogościowe. Podane zostały również dalsze lektury z różnych dziedzin logiki i informatyki.

W tym miejscu należy dać odpowiedź na pytanie postawione na początku tego tekstu. Będzie ona niestety negatywna, gdyż w polskich warunkach podręcznik ten nie jest odpowiedni dla studentów pierwszego roku studiów informatycznych. Dzieje się tak z dwóch powodów. Pierwszym i najpoważniejszym z nich jest stopień trudności przedstawianego materiału. Autor we wprowadzeniu twierdzi, że do lektury podręcznika nie jest potrzebna wiedza matematyczna. Niestety do zrozumienia dowodów twierdzeń zawartych już na pierwszych kartach wymagana jest podstawowa wiedza z zakresu teorii systemów dedukcyjnych. Drugim mankamentem podręcznika jest założenie, że czytelnik zna kilka dziedzin informatycznych. Niestety i tutaj student pierwszego roku zderzy się z przysłowiową ścianą, gdyż niewielu jest takich, którzy potrafią programować w Prologu już na pierwszym roku studiów. Podobnie ma się rzecz ze znajomością gramatyki bezkontekstowej. W tym miejscu należy zwrócić uwagę, że definicja wyrażenia poprawnie zbudowanego jest bardziej zrozumiała, gdy przedstawi się ją w sposób tradycyjny, a nie jako „przepis" produkcji formuł z gramatyki.

Omawiana publikacja okazuje się być dobrą lekturą uzupełniającą dla studentów wyższych lat informatyki i nauk z nią spokrewnionych. To, co należy zaliczyć do plusów podręcznika, to udane połączenie logiki i informatyki. Ważne są również klarowne przykłady zastosowań poszczególnych systemów formalnych do informatyki – i odwrotnie. Dla tych, którzy potrafią programować w Prologu, książka stanowi doskonałe źródło nie tylko samych algorytmów, ale także gotowych programów dostępnych ze strony internetowej Autora (http://stwww.weizmann.ac.il/G-CS/BENARI/). W sposób przystępny dla zaawansowanego czytelnika zostało przedstawione przejście od logiki predykatów do programowania w logice, a w konsekwencji do języka programowania Prolog.

Dużą zaletą podręcznika jest przejrzysta szata graficzna. Nie sposób nie wspomnieć także o bardzo dobrym poziomie językowym polskiego przekładu, co jest niewątpliwą zasługą Tłumaczki, Mirosławy Miłkowskiej.

Jakub Krysiewicz
Katedra Podstaw Informatyki KUL