dr Maciej Bendkowski

Rozmowa z autorem pracy:
„Quantitative aspects and generation of random lambda and combinatory logic terms".

Dlaczego, według Pana, warto nie tylko uczyć się matematyki, ale też rozwijać ją jako dziedzinę?

Matematyka jest językiem współczesnej nauki bez względu na to, czy weźmiemy pod uwagę inżynierię, nauki przyrodnicze, medycynę, ekonomię czy nawet nauki humanistyczne. Daje możliwość układania myśli, formułowania ścisłych argumentów oraz wydobywania sedna rozpatrywanych problemów. W konsekwencji, matematyka pełni kluczową rolę w staraniach ludzkości, aby zrozumieć otaczający nas świat. Bez postępów w takich dziedzinach jak informatyka teoretyczna, teoria liczb czy kombinatoryka żylibyśmy w całkowicie innym świecie, bez większości współczesnych osiągnięć technologicznych, na których polegamy na co dzień, jak na przykład bez Internetu. Matematyka jest nie tylko językiem nauki, jest także przyszłością nauki.

Co przekonało Pana do poświęcenia rozprawy doktorskiej logice kombinatorycznej i powiązanym zagadnieniom?

W czasie studiów przypadły mi do gustu logika i teoria obliczalności wykładane przez prof. Marka Zaionca na Uniwersytecie Jagiellońskim. Powiązane metody automatycznego rozumowania czy pytania dotyczące rozstrzygalności pewnych problemów decyzyjnych na tyle mnie wciągnęły, że zdecydowałem kontynuować tę ścieżkę pod kierunkiem prof. Zaionca, który później został moim promotorem.

Jakie największe przeszkody pojawiły się w czasie badań i jak je Pan przezwyciężył?

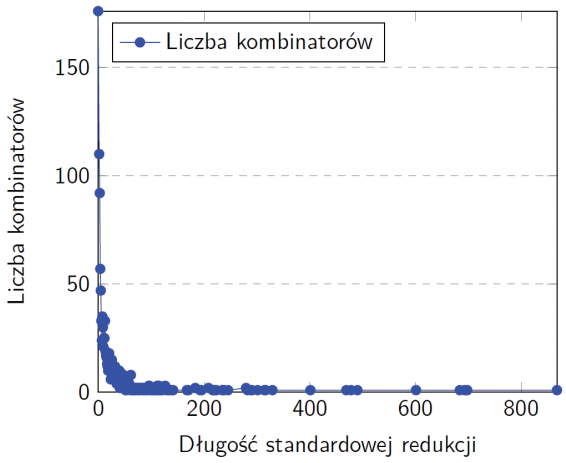
Ze względu na nietypową strukturę kombinatoryczną niektórych z badanych właściwości termów, takich jak normalizacja czy typowalność, nie udało się zastosować klasycznych metod badawczych, a zamiast tego trzeba było znaleźć alternatywne podejście. Jako przykład, rozważmy następujący problem. Mając dany program, jakie są szanse, że ostatecznie zakończy on swoje obliczenia? Czy prawdą jest, że prawie wszystkie programy kończą swoje działania, a tylko zaniedbywalnie mały odsetek programów kontynuuje obliczenia bez końca? Razem z moimi współautorami musieliśmy zaproponować alternatywne podejście do tak postawionego problemu i w efekcie przybliżać nierozstrzygalne właściwości za pomocą ich rozstrzygalnych pod- i nadzbiorów. Nasze zadanie okazało się być niezmiernie trudne. Na szczęście, pewnego dnia uświadomiłem sobie, że możliwe jest efektywne badanie programów kończących swoje obliczenia w z góry ograniczonym czasie. Co więcej, odkryłem możliwość automatyzacji mojej techniki i jej implementacji w postaci programu komputerowego. W następstwie, udało mi się zaproponować skuteczny schemat aproksymacji powyższego problemu i wyjaśnić nie tylko strukturę normalizowalnych termów logiki kombinatorycznej, ale również wytłumaczyć powód nieodłącznej trudności rozważanego problemu.

W jaki sposób infrastruktura Cyfronetu pomogła w przezwyciężaniu tych trudności?

Chociaż z wykorzystaniem mojej metody można automatycznie poprawiać oszacowanie prawdopodobieństwa, że wyrażony w języku logiki kombinatorycznej program zakończy swoje działanie, to ze względu na ogromną ilość zaangażowanych zasobów obliczeniowych, empirycznie otrzymane dolne i górne ograniczenia są wciąż bardzo odległe. Aby sprawdzić, jak dobry jest opracowany schemat przybliżenia, musiałem użyć alternatywnej metody szacowania szukanego prawdopodobieństwa. W tym celu zdecydowałem odwołać się do klasycznych technik *Monte Carlo* oraz wykorzystać do obliczeń superkomputer Prometheus. Eksperyment z założenia był dość prosty. Wygenerowałem duże, jednostajnie losowe (warunkowane rozmiarem) termy kombinatoryki logicznej i normalizowałem je przez z góry ustaloną liczbę kroków redukcji. Co więcej, przyjąłem, że jeśli obliczenia zajmą zbyt dużo czasu, dane obliczenie nigdy się nie zakończy. Oczywiście, to założenie nie jest zawsze spełnione, jednak dla celów eksperymentalnych takie uproszczenie było wystarczające. Powyższy schemat *Monte Carlo* został zaimplementowany w języku programowania Haskell oraz zrównoleglony na kilkudziesięciu rdzeniach CPU. Bez tak potężnej paralelizacji niezbędne obliczenia zajęłyby kilka miesięcy. Na szczęście, dzięki Prometheusowi ich zakończenie było możliwe w ciągu kilku dni. Eksperymenty *Monte Carlo* ukazały sporą przepaść pomiędzy obecnie najlepszymi oszacowaniami teoretycznymi a przypuszczalną prawdą.

Jakie mogą być następstwa Pana badań w przyszłości?

W ostatnich latach obserwujemy rosnące zainteresowanie wykorzystaniem losowości w procesie testowania oprogramowania. W tym podejściu programiści nie tworzą testów jednostkowych jako takich, a raczej wyszczególniają konkretne niezmienniki i właściwości, które ich kod ma za zadanie spełniać. Zadaniem komputera jest wygenerowanie losowych testów jednostkowych oraz próba dowiedzenia nieprawdziwości twierdzeń programisty. Wykazano, że taka metoda jest skuteczna i dostarcza mocnych argumentów na to, że testowany kod spełnia zadeklarowane własności. Razem z moimi współautorami odkryliśmy, że przyjęta reprezentacja syntaktyczna programów silnie wpływa na właściwości typowych obliczeń. Mam nadzieję, że wyniki moich badań zostaną wykorzystane w doskonaleniu narzędzi testujących i pomogą programistom w codziennym pisaniu poprawnego i bezpiecznego kodu.



Empiryczny rozkład długości redukcji losowych SK-kombinatorów. Dane uzyskane przy użyciu superkomputera Prometheus

Jakie ma Pan rady dla osób, które dopiero zaczynają studia doktoranckie?

Współczesna nauka jest bardziej międzynarodowa i nastawiona na współpracę niż kiedykolwiek wcześniej. Dzięki Internetowi proces współpracy znacznie się uprościł, a granice między państwami i fizyczny dystans przestają być prawdziwą przeszkodą dla prowadzenia badań naukowych. Chciałbym zachęcić młodszych kolegów i koleżanki, aby aktywnie szukali nowych możliwości i okazji oraz kontaktowali się z naukowcami spoza swojego najbliższego kręgu badawczego. Nauka jest globalnym przedsięwzięciem i, jako taka, wymaga globalnego wysiłku.