prof. dr hab. inż. Maciej Drozdowski
Instytut Informatyki
Politechnika Poznańska
ul. Piotrowo 2, 60-965 Poznań

Poznań, 29. czerwca 2023r.

Recenzja rozprawy doktorskiej
mgr. Pawła Żuka, pt.
Resource allocation methods for serverless cloud computing platforms

Niniejsza recenzja została opracowana na wniosek Rady Naukowej Dyscyplin Matematyka i Informatyka Uniwersytetu Warszawskiego.

Tematyka rozprawy doktorskiej

Praca doktorska mgr. Pawła Żuka dotyczy szeregowania obliczeń w środowiskach typu *funkcja jako usługa* (ang. Function as a Service, FaaS). Jest to jeden z nowoczesnych trybów wykonywania obliczeń w systemach chmurowych (ang. cloud computing). Umowna "funkcja" jest tu zwykle pewnym relatywnie krótkim sekwencyjnym programem, który koduje zwykle bezstanową funkcję w jednym z typowych współczesnych języków programowania.

Inne, powstałe wcześniej koncepcje używania systemów chmurowych, takie jak infrastruktura jako usługa (IaaS), platforma jako usługa (PaaS), oprogramowanie jako usługa (SaaS), nakładają na potencjalnych użytkowników (klientów) większe wymagania inwestycyjne związane z utworzeniem środowiska obliczeń, jego architekturą, rozmiarami oraz długotrwałością utrzymania. W tych trzech starszych paradygmatach użytkownik określa zwykle swoje zapotrzebowania zasobowe, np.: typ i liczbę potrzebnych instancji systemów komputerowych, platformę systemu operacyjnego, bazy danych, serwera www. Aby uzyskać pożądane gwarancje jakości świadczonych usług użytkownik systemu chmurowego musi być bardzo świadomy wymagań wydajnościowych dla świadczonych przez siebie usług, wydajności wykorzystywanego systemu chmurowego, a także musi zarządzać tą wydajnością. Zarządzanie wydajnością oznacza tu zrozumienie związku między jakością usług świadczonych użytkownikom końcowym, a ilością i typem zasobów zakupionych od dostawcy systemu chmurowego.

W porównaniu z wcześniejszymi podejściami, przetwarzanie w paradygmacie FaaS zdejmuje z użytkownika obowiązek zarządzania infrastrukturą i wydajnością. Platforma obliczeniowa zostaje wyabstrahowana do postaci środowiska uruchomieniowego, w którym wykonuje się "funkcja". Wydaje się, że skoro funkcje w FaaS są krótkimi programami, to cechują się one małymi wymaganiami zasobowymi i łatwo można nimi zarządzać. Ta wizja daje użytkownikowi przekonanie, a może wiarę, że system FaaS umożliwi automatyczne elastyczne skalowanie rozmiarów infrastruktury do obciążeń generowanych przez strumień żądań wykonania obliczeń. W rzeczywistości, to podejście jest rozwiązywaniem problemu przez przesunięcie go w inne miejsce. W przypadku FaaS problem zarządzania zasobami zostaje przeniesiony na właściciela infrastruktury chmurowej, który musi w większym stopniu administrować zleconymi obliczeniami. Praca doktorska mgr. Pawła Żuka dotyczy właśnie tego zagadnienia – określenia jak właściciel systemu chmurowego ma zarządzać wykonaniem obliczeń w paradygmacie FaaS, aby jakość świadczonych usług była jak najwyższa przy jak najniższych nakładach zasobowych.

Zawartość rozprawy

Praca doktorska składa się z 7 rozdziałów, z czego oryginalne kontrybucje Autora zawarte są w rozdziałach od 3. do 6. Problemy szeregowania wywołań funkcji są tu formułowane i rozwiązywane na różne sposoby, stosowne do modeli wykonywanych funkcji i dostępnej platformy obliczeniowej. Każdy z rozważanych problemów jest opatrzony obszernym przeglądem literatury.

W dalszej części recenzji zgłoszenia żądań wykonania funkcji będę nazywać zadaniami, a jednostki obliczeniowe na których wykonywane są zadania (rdzenie lub całe CPU) będę określać jako procesory. W pracy zakłada się, że algorytmy szeregowania muszą działać online, co w tym kontekście oznacza krótki czas wykonania przy ograniczonej informacji o przybywających zadaniach. Przyjmuje się, że wymagania zasobowe zadań, takie jak zapotrzebowanie na pamięć, czasy wykonania lub obciążenie procesora, mogą zostać określone z dużą dokładnością na podstawie zapisów z poprzednich wykonań zadania. Praktyczna stosowalność modeli szeregowania przyjętych w pracy jest weryfikowana przez odniesienie do rozwiązań systemu FaaS OpenWhisk.

Rozdział 3. pracy jest poświęcony problemowi szeregowania zadań i zarządzania środowiskami wykonania funkcji. Uruchomienie środowiska wykonania trwa długo i może trwać nawet dłużej niż wykonanie samego zadania. Ponadto, między zdaniami istnieją zależności kolejnościowe, które można wykorzystać jako wskazówki do uprzedzającego utworzenia środowiska wykonania zadań. Wyzwanie polega tu na określeniu uszeregowania (harmonogramu) zadań na procesorach, uszeregowania akcji uruchamiania środowisk wykonania na procesorach, usuwania zbędnych środowisk wykonania z procesorów, przy ograniczonej ilości pamięci dla środowisk wykonania, w celu minimalizacji średniego czasu przepływu zadań. Autor zaproponował szereg szybkich heurystyk o charakterze zachłannym, których skuteczność przebadał na danych syntetycznych i na danych wykorzystujących Microsoft Azure Trace, tj. zapis z działania produkcyjnego systemu FaaS. Praktyczne rekomendacje wynikające z przeprowadzonych badań to: a) uruchamianie najpierw zadań które mają już załadowane środowiska wykonania, b) dołączanie do kolejki zadań do wykonania następników uruchomionych właśnie zdań bo daje to czas na uruchomienie dodatkowych środowisk wykonania, c) należy raczej poczekać na zakończenie wykonywanej właśnie funkcji wykorzystującej potrzebne środowisko wykonania niż tworzyć nową kopię takiego środowiska.

W rozdziale 4. rozważany jest problem równoważenia obciążeń i minimalizacji liczby procesorów w systemie, w którym np. środowiska wykonania FaaS są utrzymywane w trwały (statyczny) sposób. Zadania (np. środowiska FaaS) są reprezentowane jako 2-wymiarowe wektory, których składowymi są: 1) zapotrzebowanie na pamięć oraz 2) zapotrzebowanie na moc obliczeniową wyrażane np. w liczbie procesorów, lub średnim wykorzystaniu procesorów. Problem optymalizacyjny jest tu szczególnym przypadkiem problemu pakowania wektorów (ang. vector packing problem). Przy tym uruchomienie zadania zawsze oznacza pobranie pewnej stałej ilości pamięci, niezależnie od intensywności ruchu (np. od intensywności żądań wykonania funkcji wyrażonej liczbą potrzebnych procesorów). Dodawane do siebie wektory zadań muszą się mieścić w pudłach opisujących zasoby procesorów aby nie dopuścić do ich przeciążenia i utraty jakości świadczonych usług. Autor rozważa dwie alternatywne funkcje celu: 1) minimalizację najgorszego obciążenia procesora przy ograniczeniu na liczbę procesorów i 2) minimalizację liczby procesorów przy ograniczeniu na obciążenie pojedynczego procesora. W obu przypadkach ogranicza się też ilość pamięci na każdym procesorze. Dla obu przypadków Autor wykazał silną NP-trudność, zaproponował algorytmy wielomianowe rozwiązujące szczególny przypadek gdy zadania mają takie same żądania pamięciowe i procesorowe. Następnie analizowane były przypadki, gdy zadanie może być przydzielone tylko do jednego procesora, albo alternatywnie, do wielu procesorów o ile całkowite żądania na moc obliczeniową są zaspokojone. Autor wykazał, że w najgorszym przypadku rozwiązania dopuszczające wykonanie zadania przez wiele procesorów mogą mieć dwukrotnie mniejszą liczbę potrzebnych procesorów niż gdy zadanie może być przydzielone tylko do jednego procesora. Dla przypadku ogólnego zaproponowano algorytmy typu zachłannego, które przetestowano na zbiorze danych pochodzącym z systemu produkcyjnego (Azure Public Dataset V2).

Rozdział 5. jest poświęcony szeregowaniu wywołań funkcji, które już zostały przydzielone do konkretnego wieloprocesorowego komputera, w celu optymalizacji jakości ich obsługi. Rozważane tu problemy, w trójpolowej notacji problemów szeregowania, można zapisać jako $P|olt,p_i{\sim}\boldsymbol{P_i}|X$. Napis ten oznacza szeregowanie na procesorach identycznych (pierwsza litera "P"), zadań które przybywają w czasie i należy je szeregować online tzn. bez deterministycznej wiedzy o czasie wykonania zadania (napis "olt"), zadania typu "i" mają czasy wykonania ze znanej dystrybuanty (napis $p_i{\sim}\boldsymbol{P_i}$). Kryterium jakości rozwiązań (oznaczone tu jako "X"), przyjmuje jedną z alternatywnych postaci, np.: średni czas przepływu, średnie spowolnienie (ang. stretch), 99. percentyl czasu przepływu, 99. percentyl

spowolnienia. W przypadku gdy dopuszcza się podzielność zadań, w środkowym polu notacji dodaje się napis "pmtn" ($P|olt,pmtn,p_i \sim P_i|X$). Dla problemów tych Autor zaproponował heurystyki typu zachłannego i przetestował ich skuteczność na danych pochodzących z systemu produkcyjnego (Azure Function Trace). Uzyskane wyniki pokazują, że heurystyki SERPT (dla szeregowania podzielnego) i SEPT, FC (dla szeregowania niepodzielnego) działają w zadowalający sposób.
W rozdziale 6. Autor opisuje eksperyment polegający na implementacji algorytmów z rozdziału 5. w rzeczywistym systemie wykorzystującym oprogramowanie OpenWhisk. Obciążeniem testowym w tych badaniach był benchmark SeBS (SeBS: A Serverless Benchmark Suite for Function-as-a-Service Computing, referencja [54]). Uzyskane wyniki pokazują, że zaproponowane algorytmy SEPT, FC działają w lepszy sposób niż referencyjny algorytm FIFO.

Uwagi krytyczne

W rozdziale 3 używa się określenia "family". Pierwszy raz zostało ono użyte na str. 19. Wydaje się, że pojęcie "family" nigdzie nie zostało zdefiniowane. Z kontekstu można zgadywać, że jest to typ wywoływanej funkcji. Sądzę, że to pojęcie należało wprost zdefiniować.
Na str. 23, użyto oznaczenia C(e), gdzie e oznacza środowisko wykonania. Chociaż zapis jest zrozumiały, to jest on nieelegancki bo oznaczenie C() na str. 18 wprowadzono w odniesieniu do zadań, a nie środowisk wykonania.
W rozdziale 3.3.4, przy omawianiu wyników z wykresu 3.3 pominięto korzystny, w moim przekonaniu, wpływ reguły "wait". Podobnie postąpiono w rozdziale 3.3.6 przy prezentacji wyników z wykresu 3.5.
W rozdziale 4.1 Autor deklaruje, że "we formally define the optimization problem . . . as a general integer linear program (ILP)", co nie jest poprawne gdyż ograniczenie (4.2) na str. 48 zawiera iloczyn dwóch zmiennych.
W rozdziale 4. Autor używa symbolu "÷" bez określenia jego znaczenia. Tradycyjnie jest to symbol dzielenia. Ale jakiego? Z kontekstu wynika, że chodzi o dzielenie bez reszty. Jednak Autor używa równolegle symboli dzielenia z zaokrągleniem w górę (sufit) i w dół (podłoga), np. w Algorytmie 3. (str. 57). Uważam że taka prezentacja jest niespójna i nieelegancka.
W rozdziale 4.5.2 przy omawianiu wyników z wykresu 4.5 należałoby pokusić się o jakieś wyjaśnienie dlaczego algorytm random ma dobre wyniki dla pamięci do 128GB, a reguły cpu-inc i ratio-inc są zawsze najgorsze.
W rozdziale 5.1 (str.71) Kandydat wprowadza miarę "Average function-aggregated stretch", której celowość objaśnia następująco: "Our aim is to measure the fairness of a schedule based on the average values of the flow time or stretch within the sets of invocations of the same functions." Ta miara nie jest dobrym wskaźnikiem bezstronności (ang. fairness) bo nie pokazuje różnicy w dostępności do zasobu dla różnych typów funkcji, a jedynie ważone centralne tendencje. W moim przekonaniu lepsze byłyby wszelkie miary pokazujące rozrzut wskaźników jakości usługi dla poszczególnych typów funkcji.
W odniesieniu do wykresu 5.2 (str. 73), o ile zgadzam się z wnioskiem że względna różnica maleje z intensywnością zgłoszeń, to mam wątpliwości czy użycie regresji liniowej jest uzasadnione. Punkty na wykresie nie wydają się układać wzdłuż linii. Rekomendowałby sprawdzenie innych zależności, w tym potęgowej, i wybranie takiej która ma najlepszy współczynnik determinacji R^2.
Uprzejmie proszę o wyjaśnienie jak można wejść do linii 6 w Algorytmie 5 (str.78). Czy NOC_j może być ujemne?
W opisie wykresów 6.3 i 6.4 (na str. 102, koniec rozdziału 6.5 i w podpisach pod wykresami) stwierdza się że liczba rdzeni jest taka sama w kolumnach, a intensywność zgłoszeń w wierszach. Jednak podpisy składowych wykresów (w szczególności c i d) nie zgadzają się z tą deklaracją.
Powyższe uwagi krytyczne nie podważają ogólnej poprawności pracy, poprawności metodologicznej prowadzenia badań, uzyskanych wyników, ani prawidłowości wyciągniętych końcowych wniosków. Rad bym jednak aby Kandydat ustosunkował się do powyższych uwag.

W tekście zauważyłem również pewną liczbę niedostatków stylistycznych i językowych. Na przykład, brakujące "a/the", użycie "addiditonal overhead . . . added", " opimisations", użycie niewłaściwej liczby

gramatycznej "all datasets contains". Nie przytaczam tu ich pełnej listy, gdyż nie mają one wpływu na poprawność wyników badań.

Ocena pracy

Doktorant zbadał trzy oryginalne problemy z dziedziny informatyki, dotyczące zarządzania wykonaniem obliczeń w systemach chmurowych. Kandydat określił klasę złożoności obliczeniowej tych problemów przez podanie dowodów NP-trudności, albo przez wskazanie znanych NP-trudnych problemów szeregowania, które są szczególnymi przypadkami rozważanych tu zagadnień. Dla prostych przypadków tych problemów podał algorytmy wielomianowe. Dla przypadków ogólnych przedstawił szybkie algorytmy zachłanne, które są praktyczne w tym sensie że mają realizowalne wymagania co do potrzebnych danych wejściowych i co do złożoności obliczeniowej. Algorytmy te poddano testom w trybie symulacyjnym jak i walidacji w rzeczywistym systemie obliczeniowym. Zwłaszcza walidacja w rzeczywistym systemie jest godna uznania. Do testów symulacyjnych wykorzystał logi z działania produkcyjnych systemów komputerowych, co jest uważane za lepsze rozwiązanie niż testowanie algorytmów na danych syntetycznych. Dlatego uważam, że merytoryczne wyniki uzyskane przez mgr. Pawła Żuka są bezsporne i istotne, a metodologia ich uzyskania nie budzi zastrzeżeń.
Publikacyjny dorobek Kandydata na który składa się 5 prac, z tego jedna w czasopiśmie (Journal of Parallel and Distributed Computing, 100pkt. MEiN) i cztery na konferencjach (SBAC-PAD 70pkt, 2*CCGrid 140pkt, IEEE Cluster 140pkt.), należy uznać za bardzo dobry.

Wniosek końcowy

Uważam, że przedmiotem przedstawionej rozprawy doktorskiej jest rozwiązanie oryginalnego problemu naukowego, że prezentuje ona ogólną wiedzę teoretyczną Kandydata w dyscyplinie informatyka i potwierdza Jego umiejętność samodzielnego prowadzenia pracy naukowej. W moim przekonaniu zarówno wymogi ustawowe jak i zwyczajowe wobec pracy doktorskiej zostały spełnione. Dlatego wnoszę o dopuszczenie mgr. Pawła Żuka do obrony.

M. Drozdowski