Prof. dr hab. inż. **Valeriy Rogoza**
Zachodniopomorski Uniwersytet
Technologiczny w Szczecinie,
Wydział Informatyki,
ul. Żołnierska 49, 71-210 Szczecin
wrogoza@wi.zut.edu.pl

Szczecin, 11 czerwca 2012 r.

Recenzja

rozprawy habilitacyjnej i dorobku naukowego dr inż. Mieczysława Drabowskiego

Rozprawa habilitacyjna: dr inż. **Mieczysław Drabowski**
„Parallel Synthesis of Computer Systems"
(Par-synteza systemów komputerowych),
Politechnika Krakowska im. Tadeusza Kościuszki,
Wydział Inżynierii Elektrycznej i Komputerowej

Zakres i cel pracy

Rozwój systemów komputerowych pewnie skierowany jest w stronę zjednoczenia zasobów sprzętowych w jednolitą strukturę z bardzo potężną mocą obliczeniową. Tendencja owa była spostrzegana co raz wyraźniej w miarę rozwoju sprzętu i oprogramowania. Zjawienie na rynku stosunkowo tanich mikroprocesorów spowodowało rozwój technologii systemów komputerowych zawierających duże zbiory procesorów, które posługiwując wspólną pamięcią w sposób równoległy rozwiązują złożone problemy obliczeniowe. A wynalezienie szybkich sieci kopmuterowych umożliwiło łączenie w obrębie firm, miast, krajów i całego świata setek i tysięcy komputerów w taki sposób, że informacje i wyniki oblizeń mogą być przesyłane między komputerami w czasie kilku mikrosekund. Obecnie aktywnie się rozwija technologie GRID- i CLOUD-komputingu, tworząc jakościowo nową platformę do rozwiązania bardzo skomplikowanych zagadnień w ekonomii, inżynierii, socjologii i innych naukach fundamentalnych i stosowanych. A więc, analiza stanu i tendencji rozwoju sprzętu i oprogramowania w systemach komputerowych zawierających duże ilości jednostek zasobów komputerowych (procesorów, komputerów) działających równoległe wskazuje na to, że konieczność budowy modeli architektur systemów komputerowych pozwalających zmniejszyć koszt i czas obliczeń jest aktualnym zagadnieniem współczesnych technologii informatycznych.

W tych warunkach prawdopodobnie jednym z centralnych zagadnień jest optymalizacja procesów obliczeniowych, poprawny podział zasobów między konkretnymi zadaniami w celu

minimalizacji czasu i kosztu obliczeń, zrównoważenie rozpodziału obciążenia jednostek sprzętowych między zadanimi i zachowanie wysokiej wiarygodności rozwiązania poszczególnych problemów. Jak świadczy praktyka, omówione warunki mogą być zasadniczo spełnione, jeżeli w procesie planowania obliczeń będą przyjęte pod uwagę zarówno specyficzne cechy zadań ulegających rozwiązaniu, jak i możliwości sprzętu, na którym zadania te będą rozwiązywane.

Niestety w dotychczasowych metodach skierowanych na optymalizację obliczeń w systemach zawierających wiele procesorów części sprzętowe były opracowywane i optymalizowane oddzielnie od części programowych. Później po optymalizacji części sprzętowej i części programowej one były łączone razem w procesie tzw. co-syntezy końcowego systemu. Chociaż techniki co-syntezy często doprowadziły do ulepszenia właściwości procesów obliczeniowych, ale w wielu konkretnych sytuacji było zrozumiało, że rozwiązania leżą daleko od optymalnych. A więc, co-synteza jest w tym sensie paliatywnym, lecz nie najlepszym rozwiązaniem omówionego zagadnienia.

Celem recenzowanej pracy jest tworzenie nowego podejścia do rozwiązywania problemów wyboru, szeregowania i alokacji zadań i zasobów komputerowych nazwaną par-syntezą (ang. parallel synthesis), czyli syntezą koherentną, która zasadniczo odróżnia się od metody co-syntezy w tym, że tworzenie i optymalizacja modelu obliczeniowego jest rozpatrywane jako proces łączący w siebie części sprzętowe i programowe. Z teoretycznego punktu widzenia jest to najbardziej uogólnająca postać rozwiązania zagadnienia optymalizacji podziału zasobów między zadaniami. Właśnie w takiej postaci został sformulowany cel badań w danej pracy.

W ten sposób, sformulowane w pracy zagadnienie jest aktualne w ujęciu teoretycznym, a jego rozwiązanie ma istotne znaczenie dla praktyk pobudowy optymalnych modeli obliczeń w złożonych środowiskach komputerowych.

Jak prawidłowo wskazuje autor rozprawy (patrz Rozd. 2 monografii oraz tekst Autoreferatu, str.2), „punktem wyjścia do opracowania algorytmów podziału funkcji systemu i jego optymalizacji jest określenie ogólnego modelu dla problemów wyboru zasobów – realizujących ów podział i wpływających na koszt systemu – i szeregowania zadań alokowanych na wybranych zasobach – istotnie wpływających na szybkość systemu." W monografii autor proponuje nowe podejście, które bazuje na modelu dla problemów szeregowania zadań.

Synteza dowolnego modelu, w tym modelu obliczeń, zyskuje sens, jeżeli określone są pewne kryteria optymalności, czyli podział funkcji projektowanego systemy obliczeniowego na części architektury sprzętowej i oprogramowania powinien być optymalizowany. Jako czynniki ulegające optymalizacji w pracy zostały wybrane: minimalny koszt implementacji zadań i korzystania z zasobów, minimalizacja poboru energii podczas realizacji funkcji, oraz minimalizacja czasu reakcji systemu na wykonywanie zadań przez przydzielone zasoby. Jako dodatkowy faktor ulegający sprawdzaniu, autor formuluje również wymaganie, że realizacja wyboru zasobów, jak i uszeregowań zadań powinna również zapewniać niezawodne działanie systemu.

A więc, można wnioskować, że autor rozprawy słusznie łączy do zakresu pracy procesy optymalizacyjne, czyli innymi słowy, rozważa proces par-syntezy modelu obliczeniowego jako proces łączący w siebie też składnik optymalizacyjny.

Zawartość rozprawy, najważniejsze osiągnięcia naukowe autora rozprawy

Tekst rozprawy przedsatwiono w postaci monografii napisaną w języku angielskim, która wydana jest przez Wydawnictwa AGH w serii Rozprawy, Monografie w roku 2011, numer 225, 208 str. (przewodniczący Komitetu Naukowego Wydawnictw AGH – prof. Tomasz Szmuc, recenzenci monografii – prof. Andrzej Piegat i prof. Mieczysław Kłopotek). Poza wstępem („1. Introduction") i zakończeniem („8. Résumé, conclusions, further research") rozprawa sklada się z szeszciu rozdziałów (przenumerowanych jako rozdziały 2 – 7).

We Wstępie do pracy („1. Introduction") oraz w rozdziale drugim ("2. Coherent synthesis of computer systems – model and method" – Koherentna synteza systemów komputerowych – model i metoda) autor wprowadza Czytelnika do problematyki badanej w pracy, a mianowicie – określa, na czym polega zasada łącznego rozwiązania problemu wyboru zasobów i szeregowania zadań i co to jest par-synteza. W interpretacji autora system komputerowy jest przedsatwiony w postaci formalnej triady (str.20 monografii; *w recenzji w tym miejscu i dalej, jeżeli nie będzie omówiono coś inne, domyślnie uważa się, że podane strony dotyczą stron monografii*):

$$\Sigma = \{ \mathbf{R}, \mathbf{T}, \mathbf{C} \},$$

gdzie **R** – mnóstwo zasobów (sprzęt i oprogramowanie); **T** – zbiór zadań ulegających rozwiązaniu; **C** – zbiór kryteriów optymalizacyjnych określających zachowanie systemu i jego strukturę.

Według autora, mnóstwo zasobów zawierają zbiór procesorów $P = \{ P_1, P_2, \dots , P_m\}$ oraz zbiór dodatkowych zasobów $A = \{ A_1 , A_2, \dots , A_p \}$, <u>których sens i przeznaczenie autor nie skomentował</u>. Zbiór zadań składa się z n zadań, które ma być podzielone przez m procesorów. Do zbioru kryteriów optymalności autor zaliczył minimalizację wartości rozwiązania zadań, maksymalizację szybkości rozwiązania zadań i minimalizację energii. Krótki drugi rozdział monografii zamykany jest przedstawieniem schematu par-syntezy systemu komputerowego. Schematy te przedsatwione są na rys. 2.3 (str. 23 monografii) i na rys. 4 (str. 5 autoreferatu). Niestety, w wersji angielskiej (czyli w monografii) i wersji polskiej (czyli w autoreferacie) <u>rysunki te nieco różne, chociaż opisują ten sam proces par-syntezy.</u>

Autor deklaruje tezę (patrz np. str. 3 referatu), że <u>„problemy syntezy i ich optymalizacje są obliczeniowo złożone (NP-trudne)"</u>, a dlatego do budowy metod syntezy systemów komputerowych wybrane były podejścia meta-heurystyczne inspirowane naturą, a mianowicie: algorytmy genetyczne, symulowane wyżarzanie, poszukiwania tabu, sieci neuronowe oraz kolonie mrówek. Kolejne rozdziały monografii poświęcone są rozwoju i stosowaniu idei wspomnianych rozdziałów sztucznej inteligencji do budowy odpowiednich algorytmów par-syntezy.

W trzecim rozdziale monografii („3. Evolutionary method for coherent synthesis of computer systems" – Metoda ewolucyjna syntezy koherentnej systemów komputerowych) tym celom posłużyła teoria metody ewolucyjnej połączonej z symulowanym wyżarzeniem. Autor korzysta z parametru „globalna temperatura algorytmu" (str.28, *„parameter global temperature"*) jako jednego z parametrów, za pomocą którego można sterować procesem wyszukiwania optymalnych rozwiązań (np. powolne „ochlodzenie" algorytmu może zapobiec zbyt szybkiej zbieżności algorytmu do lokalnych (ale nie najlepszych) optymów, co jest dosyć charakterystycznym problemem algorytmów genetycznych. Algorytm szuka takie rozwiązanie, które odpowiada mniejszej temperaturze i większej przestrzeni rozwiązań. Ogólna zasada zastosowania metody ewolucyjnej do rozwiązania problemu podziały zasobów, która jest rozwijana w danym rozdziale, polega na klasteryzacji zasobów i budowy pewnej hierarchii ich wykorzystania (patrz algorytm ASAP, str. 33 – 34; rys. 3.7., str. 34). Zasoby przedsatwione są w postaci symbolicznej hyper-sfery, a proces ich podziału – jako wybór pewnych hyper-płaszczyzn (podrozdziały 3.2 i 3.3) dzielących hyper-sferę na części (klastry) według wspomnianych wyżej kryteriów optymalności. Wybór ten realizowany jest przez posługiwanie się w pewny sposób operatorami genetycznymi krzyżowania i mutacji.

Istotną uwagę w tym rozdziale przydzielono opisu eksperymentów i porównaniu niekoherentnego i koherentnego podejść do rozwiązania tych samych problemów podziału zasobów i zadań. W szczególności, z eksperymentów z rozd. 3 można wnioskować, że podejście koherentne wyjawiają pewne przewagi jedynie na stosownie dużej ilości zadań (np. jeżeli ilośc zadań jest większa niż 50 – 60), natomiast w razie małej ich ilości obydwa podejścia są praktycznie identyczne stosownie do czasu rozwiązania problemu oraz wartości tego rozwiązania.

Podobny wniosek można zrobić również w razie posługiwania się wielokryterialną optymalizacją (patrz rezultaty badań w podrozd. 3.5.4). W rezultacie wykorzystania proponowanego algorytmu, projektant otrzymuje mnóstwo optymalnych rozwiązań w sensie Pareto, z których może wybrać rozwiązanie jaknajbardziej odpowiadające jego celom i dodatkowym ograniczeniom.

Czwarty rozdział monografii ("4. Tabu search method of koherent synthesis of computer systems" – Metoda poszukiwania tabu do syntezy koherentnej systemów komputerowych) zawiera badanie i proponowany algorytm rozwiązania koherentnego problemu optymalizacji zarówno podziału zasobów, jak i szeregowania zadań w postaci wiadomego w teorii metod sztucznej inteligencji zagadnienia poszukiwania tabu. Idea owej metody została opisana na poziomie opisu metody i odpowiadającego jej algorytmu (patrz algorytm TS, str. 70). Metoda określa dodatkowo takie składniki, jak: ruch, sąsiedztwo oraz dążenie (ang. movement, neibourhood, aspiration). Zagadnienie jest sformalizowane jako problem optymalizacji, a mianowicie (podano w terminologii autora): należy zminimalizować funkcję celu $ext\{f(z):z \in Z\}$, gdzie Z – to zbiór wszystkich dopuszczalnych rozwiązań. Ruch v - to funkcja przecinająca elementy zbioru $Z(v)$ w Z, przy czym $Z(v)$ – to zbiór dopuszczalnych rozwiązań, do których można zastosować parametr ruchu. W rozdziale pokazano, w jaki sposób można pokonać problem wielokrotnego poszukiwania tych samych sekwencji rozwiązań, przeanalizowano rosnącą zależność ilości iteracji od ilości zadań (Rys. 4.2, str. 77). Przedsatwiono eksperymenty komputerowe wskazujące na pewne przewagi metody koherentnej w porównaniu z metodą niekoherentną, która wyjawja się mniej więcej

wyraźnie na stosunkowo dużej ilości zadań (Rys. 4.3 i 4.4, str. 78 – 79). Ciekawa również jest analiza wpływu na cenę obliczeń takich cech zagadnienia, jak ilośćć zadań, ilości iteracji, wykorzystanie dodatkowych zasobów oraz posługiwanie się ograniczeniami (rozd. 4.5, str. 78 - 83). Eksperymentalnie udowodniono, że, po pierwsze, algorytm poszukiwania tabu w interpretacji autora pracy może być zasadniczo stosowany do rozwiązania zagadnienia szeregowania zadań i podziały zasobów, a po drugie, właśnie podejście koherentne efektywnie ulepsza rozwiązanie owych zagadnień przede wszystkim stosownie minimalizacji kosztu obliczeń.

W piątym rozdziale („5. Neural method for coherent synthesis of computer system" – Metoda neuronowa do syntezy koherentnej systemu komputerowego) autor rozwija teorię koherentnego szeregowania zadań i podziały zasobów z wykorzystaniem sztucznych sieci neuronowych. Część zadań przedsatwiona jest w postaci trójwymiarowego sześcianu w przestrzeni zmiennych „zadanie-zasób-kwant_czasowy" (patrz np. Rys. 5.4., str. 89), a część procesorowa – w przestrzeni dwówymiarowej „typ_procesora-numer_procesora" (patrz np. Rys. 5.8 i 5.9). W ramach tych współrzędnych autor opisuje algorytmy szeregowania zadań, podziału zasobów i proponowany przez autora algorytm koherentny. Jak widać z eksperymentów, zastosowanie tego ostatniego wymaga budowy sieci z olbrzymą ilością neuronow (str. 106, 107, 108).

W rozdziale przeanalizowano również sytuację, w której przewiduje się wykorzystanie dodatkowych zasobów, przy czym w tej sytuacji może istotnie zwiększyć się czas obliczeń (Rys. 5.1. i 5.2). Eksperymenty komputerowe wskazują również na to, że algorytm koherentny ma nieistotne przewagi przed algorytmem niekoherentnym z punktu widzenia ekonomii kosztu obliczeń (10 – 15%), i wyjawia te przewagi, jeżeli ilość zadań jest wystarczająco wysoka (np. zbliża się do 100), natomiast algorytmy te doprowadzą praktycznie do tych samych kosztów w razie małej ilości zadań (Rys. 5.3, 5.4). Ale, jak wynika z eksperymentów (patz Tab. 5.1 i 5.2), algorytm koherentny nawet istotnie przegrywa przed algorytmem niekoherentnym w czasie obliczeń, co nie mówi na korzyść pierwszego. Ciekawe są wyniki eksperymentów dotyczących analizie wpływu ilości uniwersalnych i wyspecjalizowanych zasobów będących w dyspozycji projektanta na koszt obliczeń (Tab. 5.6 – 5.11). Wyniki te mogą być wykorzystane w praktyce podczas budowy optymalnej architektury systemu komputerowego.

Szósty rozdział („6. Ant Colony and Branch & Bound methods for coherent synthesis of computer systems" – Metody kolonii komurek oraz podziału i ograniczeń do syntezy koherentnej systemów komputerowych) jest poświęcony badaniom dwóch różnych podejść określonych w tytule rozdziału. Podejście te są badane łącznie, bowiem, jak stwierdza autor, pozwalają znaleźć optymalne (lub bliskie do optymalnego) rozwiązanie, a także porównać jego z innymi ewentualnymi rozwiązaniami. Jak też słusznie stwierdza autor (str. 119), porównawcza analiza pozwala przeanalizować, który z algorytmów jest szybczy, który z nich jest więcej tolerantny do modyfikacji parametrów systemu, i wreszcie, który jest więcej elastyczny do adaptacji do zmieniających się wymóg. W rozdziale zostały badane następujące parametry algorytmów: rozmiar i koszt pamięci operacyjnej, rozmiar i koszt pamięci masowej, ilość procesorów i koszt potęgi komputerowej oraz czas potrzebny do roswiązania problemu szeregowania zadań.

W podrozdziale 6.1 przedsatwiono zmodyfikowany przez autora algorytm Ant Colony Optimization (ACO) do rozwiązania zagadnienia syntezy systemów komputerowych. Do oceny rozwiązań wygenerowanych przez algorytm autor skorzystał z ważonej średniej – czyli wszystkie parametry były oceniane z pewnymi wagami. Jeżeli sumeryczna ocena nowego rozwiązania jest lepsza niż poprzedniego, to będzie wybrane te nowe rozwiązanie. W algorytmie ACO każda „mrówka" jest rozptarywana jako „agent", a kolonia mrówek – jako system wieloagentowy. W proponowanym algorytmie ACO agenci są wykorzystane do wyszukiwania następujących informacji: (1) najlepszego podziału zadań między procesorami; (2) najlepszego szeregowania zadań; (3) wyszukiwania najlepszego podziału zadań dla danego podziału zasobów. Zasady działania agentów, oceny drogi przesuwania agentów od jednego rozwiązania do innego oraz działania proponowanego algorytmu zostały opisane słownie i przedsatwione w postaci odpowiednich schematów (patrz Rys. 6.1, str. 121; Rys. 6.2, str. 122 i Rys. 6.3, str. 123).

Opis algorytmu podziału i ograniczeń Branch & Bound (B&B) w pracy zajmuje pówtory strony (str. 123, 124). Autor przypomina, że są dwie wersje tego algorytmu – Best First Search (DFS) oraz Depth First Search (DFS), chociaż odsyłaczy do źródeł literackich autor nie przedstawił.

Od siebie recenzent dodaję, że tak jest w rzeczywistości, a opis obu tych wersji można znaleźć np. w książce *Stuart J. Russel, Peter Norvig, „Artificial Intelligence. A Modern Approach." – Prentice Hall, 2003.* Po krótkiemu opisu istoty algorytmów, resztę rozdziału autor poświęcił analizie danych eksperymentalnych. Przedstawiono kilka przykładów, na których autor demonstruje pewne przewagi metody koherentnej przed metodą niekoherentną (podrozd. 6.3.1), chociaż przewagi te nie są zbyt przekonujące. W podrozdziałach 6.3.2 i 6.3.3 autor demonstruje proste przykłady posługiwania się algorytmami ACO i B&B, i wnioskuje, że algorytmy te są wystarczająco proste do używania i szybko doprowadzą do rozwiązania problemów podziały zasobów i szeregowania. W tej części monografii algorytmy ACO i B&B są porównywane ze sobą na przykładach. Z tych porównań powstaje niektóre wnioski służące jako pewne rekomendacje dla projektantów. Np. chociaż szeregowanie zadań wykonane z wykorzystaniem obu algorytmów doprowadzi do tych samych rezultatów, ale czas rozwiązania zadania szeregowania jest istotnie różny (str. 137). Część końcowa tego rozdziału jest poświęcona badaniom optymalizacji czasu rozwiązania problemu szeregowania zadań i podziału zasobów, kosztu i nakładu mocy systemu komputerowego (podrozd. 6.3.4 – 6.3.8), które są interesujące z praktycznego punktu widzenia, bowiem służą pewną wskazówką dla projektantów do rozwiązania owówionych problemów z wykorzystaniem metod ACO i B&B.

W rozdziale 7 („7. The comparison of coherent synthesis algorithms" – Porównanie algorytmów syntezy koherentnej) autor przeprowadzi porównania tych algorytmów, które były opisane i badane w poprzednich rozdziałach. Owe porównanie ma szczególną wartość, bowiem podstawą dla takiego porównania posłużyły rzeczywiste zasoby uniwersalnego przeznaczenia i zasoby wyspecjalizowane, dla których określone są szybkość, koszt i zużycie energii. Niektóre wnioski są przedestynowane rezultatami badań w poprzednich rozdziałach, natomiast niektóre inne są dosyć niespodziewane. Np. podcas analizy zagadnienia minimalizacji kosztu autor uzasadnił wniosek, że tańsze struktury wygenerowane są przez

cztery algorytmy „Tabu search", „Neural", „Ant colony" oraz „Genetic" mogą być również więcej szybkie.

Z tych porównań można zrobić ciąg ciekawych wniosków mających praktyczne znaczenie. Np. algorytm „Genetic" ma lepsze albo przynajmniej nie gorsze parametry, niż inne algorytmy (patrz Rys. 7.1 – 7.20, str. 150 - 159). Dokładnie mówiąc, konkurencyjnymi do „Genetic" mogą być i inne podejścia, np. algorytm „Tabu search" wymaga mniejszego czasu niż „Genetic" w razie zależnych zadań, jeżeli ilość tych zadań jest nie większa niż 25 (patrz Rys. 7.18, str. 158), tak samo wskaźnik czasu jest lepszy dla algorytmu „Any colony" dla zadań zależnych, jeżeli ich ilość leży w zakresie od 25 do 40 (patrz Rys. 7.16). Z porównania algorytmów „Ant colony" i „Genetic" (patrz Rys. 7.21 i 7.22) wynika, że w szerokim zakresie ilości procesorów (od 1 do 10) parametr czasu ma praktycznie jednakowe wartości, natomiast parametr kosztu algorytmu „Ant colony" przegrywa przed tym parametrem w razie posługiwania się algorytmem „Genetic" za wyjątkiem może być dwóch przypadków (gdy ilość procesorów jest równa 5 i 9). Po wykonaniu porównawczej analizy tych algorytmów stosownie parametru kosztu pamięci, autor słusznie wnioskuje („7.1.7. Conclusions", str. 164), że algorytmy „Genetic" i „Ant colony" mają przewagi przed innymi heurestycznymi algorytmami, jeżeli chodzi o takich wskaźnikach, jak koszty oraz czas rozwiązania problemu.

Aby zilustrować zbieżność heurestycznych algorytmów do optymalnych rozwiązań, autor wykonał porównawczą analizę (na konkretnych przykładach zagadnień) wiadomych algorytmów wielomianowych takich, jak: McNaughton Algorithm, Coffman-Graham Algorithm, Hu Algorithm, Muntz-Coffman Algorithm, Baer Algorythm i Garey-Johnson Algorithm. Porównanie skuteczności tych algorytmów z algorytmami heurestycznymi opracowanymi przez autora do rozwiązania niewielomianowych problemów wskazują na to, że wszystkie proponowane w pracy algorytmy (oprócz jednego z problemów, gdzie algorytm „Tabu search" przegrawał) doprowadziły do praktycznie tych samych rezultatów.

W części końcowej rozdziału siódmego („7.3. Coherent resource partition and task scheduling" – Koherentny podział zasobów i szregowania zadań) autor skupił uwagę na porównaniu heurestycznych algorytmów rozwiązania zagadnienia koherentnej syntezy podziału zasobów i szeregowania zadań z ciągiem wiadomych algorytmów niekoherentnych, a mianowicie: Eles-Peng-Kuchcinski-Doboli Algorithm, Bianco-Auguin-Pegatoquest Algorithm, Dave-Lakshminarayana-Jha Algorithm, Yen-Wolf Algorithm, Eles-Peng-Kuchcinski-Doboli-Pop Algorithm, Dick-Jha Algprithm oraz Prakash-Parker Algprithm. Przedsatwione w postaci tabeli (patrz Tab. 7.13, 7.14, 7.15, 7.16, 7.17 oraz 7.18, str. 179 - 184) dane wskazuja na to, że algorytmy heurestyczne opracowane przez autora albo mają lepsze wskaźniki, albo co najmnej w większości przypadków są nie gorsze. Ponadto, na przykładach znów pokazano, jakie przewagi mają te czy inne algorytmy heurestyczne (patrz Rys. 7.48 – 7.57 oraz Tab. 7.19, 7.20 , str. 185 – 190) w sytuacjach, gdy mamy do czynienia z zadaniami zależnymi i niezależnymi, przy czym w ciągu praktycznych zagadnień szeregowanie zadań może być faktorem dominującym w poszukiwaniu optymalnych rozwiązań omówionych zagadnień.

Więc, zgłębainie przedsawionych materiałów daje podstawy zrobić następujące wnioski.

Wybrany temat badań jest aktualnym naukowym zagadnieniem, którego rozwiązanie może mieć istotny wpływ na tworzenie efektywnych systemów komputerowych dla równoległych i rozproszonych środowisk informatycznych, w tym takich systemów, jak np. sieci GRID.

Nowatorstwo pracy polega na tym, że autor proponuje rozpatrywać zagadnienia optymalnego podziały zasobów komputerowych i szeregowania zadań w systemach zawierających wiele zasobów obliczeniowych (procesorów, komputerów) jako nieodłączne części wspólnego procesu określonego przez autora terminem „par-synteza" (od ang. „parallel synthesis").

Autor przekonujące pokazuje na licznych przykładach, że wspomniane wyżej łączenie doprowadzi do ulepszenia wszystkich wskaźników systemów komputerowych.

Dosyć charakterystycznym jest również wybór przez autora pracy heurestycznych metod z teorii sztucznej inteligencji jako podstawy teoretycznej do budowy technik par-syntezy. Właśnie owe metody, jak udowodnia autor, posiadają wszystkie wyraziste narzędzia matematyczne do budowy technik par-syntezy, których możliwości stosownie danej problematyki, prawdopodobnie, jeszcze do końca zostały niezbadane i oczekiwuja na dalszy rozwój.

Z formalnej strony, można stwierdzać, że faktycznie autor stworzył matematyczną i zasadniczą podstawę do rozwoju nowego kierunku w tej części technologii informatycnzych, która dotyczy problemów racjonalnej budowy równoległych środowisk obliczeniowych.

Osobnej uwagi zasługiwują liczne praktyczne przykłady przedsatwione w pracy, potwierdzające teoretyczne wnioski, których duża ilość wskazuje na wykonanie przez autora olbrzymej objętości badań eksperymentalnych. Oprócz potwierdzenia wyników teoretycznych, dane eksperymentalne mają jeszcze drugą wartość – służą jako pewne wskazówki wytyczne dla projektantow systemów komputerowych, którzy chcieliby skorzystać z tych materiałów w swojej codziennej pracy.

Uwagi krytyczne, pytania

Niestety, w trakcie realizacji pracy autor nie ustrzegł się pewnych niedociągnięć, stwierdzeń kontrowersyjnych (z których część prawdopodobnie może służyć tematem dyskusji naukowej) oraz usterek.

1. W pracy zostały przedstawione wyniki badań dotyczących problematyki optymalnego podziały zasobów podczas rozwiązania szeregu zadań. Mówiąc ogólnie, nie jest to jednak nowe zagadnienie w technologiach informatycznych, jeżeli rozpatrywać zagadnienie te z punktu widzenia optymalizacji obliczeń w równoległych (w systemach wieloprocesorowych) i rozproszonych (w systemach zawierających wiele komputerów) systemach kopmuterowych. Dopiero tworzone są wyspecjalizowane kompilatory optymalizujące podział zasobów w systemach równoległych, a także platformy dla obliczeń równoległych (np. Open MP) i rozproszonych (klasyczne przykłady – platformy CORBA, COM+, Java RMI, Microsoft.NET i inne; patrz np.

Andrew S. Tanenbaum, Maarten van Steen, Distributed Systems. – Prentice Hall RTR, 2002). Ponadto istnieje dobrze rozwinięta teoria procesów wielowątkowych i wielostrumieniowych (patrz np. *Gregory R. Andrews, Foundations of Multithreaded, Parallel, and Distributed Programming, - Addison-Wesley, 2000*). W tych pracach opisana teoria zrównoleglenia procesów obliczeniowych z dekompozycją grubo- i drobno-ziarnistą na poziomach procesów, zadań, funkcji i operacji. A więc, w każdej nowej pracy dotyczącej tej problematyki oczywiście ma być zrobiono porównanie z istniejącymi metodami i algorytmami i udowodniono, że proponowane metody, techniki i algorytmy mają przewagi w stosunku do istniejących. Niestety w danej monografii autor ograniczył badania porównaniem swego podejścia kohernetnej syntezy jedynie z metodami syntezy niekoherentnej.

2. Str. 17. *"... a set of interactive tasks (processes, operations and functions) ... "*. Autor błędnie łączy razem różne pojęcia – procesy, operacje i funkcje – w ogólnym terminie „zadania".

3. Str. 20, góra. „(<u>The</u>) *resource set consists of P processots ... and <u>additional resources</u> A ...*" Autor nie tłumaczy, co to jest *"dodatkowe zasoby"* (czy to procesory, czy pamięć, czy też komputery).

4. Autor w różnych kontekstach operuje terminami *„optymalna (sub-optymalna) struktura"* , *„optymalne szeregowanie"* (patrz np. str. 23: *„Iterative calculations are executed till satisfactory results are obtained – i.e. optimal (or sub-optimal) system structure and schedule."* Powstaje jednak pytanie, w jaki sposób można ocenić, że udało się znaleźć owe <u>optymalne</u> (lub bliskie do optymalnego) rozwiązanie? W tym metodach, które dalej proponuje autor, brak jawnych sposobów oceny optymalności rozwiązań. Możemy ocenić optymalność jedynie w sposób niejawny porównując rozwiązania otrzymane przez wykorzystanie różnych metod.

5. Str. 26, Example. Skąd wiemy, ile i jakich zasobów będzie zawierał system, jeżeli struktura tego systemu będzie tworzona w trakcie syntezy?

6. Co oznaczają liczby wskazane na grafie na Rys. 3.5 , str. 31 i na Rys. 3.6, str. 32?

7. W tekście trafiają się deklaratywne stwierdzenia, które niekiedy wymagają więcej uważnego i szczególnego tłumaczenia, bowiem niektóre z nich mają zasadnicze znaczenia do lepszego rozumienia przewag proponowanych algorytmów. Np. na str. 35 czytamy: *„The likelihood of using either of the genetic operators is defined by the algorithm parameters." – Prawdopodobieństwo wykorzystania dowolnego z operatorów genetycznych jest określone przez parametry algorytmu.* Z tym deklaratywnym stwierdzeniem można całkiem się zgodzić, ale ono nic nie mówi o tym, w jaki sposób działa algorytm proponowany przez autora monografii, tym więcej, że, jak widać dalej z tekstu, wybór tych parametrów odróżnia się od zwykłego wyboru. W razie algorytmów genetycznych prawdopodobieństwa posługiwania się operatorami krzyżowania i mutacji ma zasadnicze znaczenie. Sprawa w tym, że jak udowodniono w licznych publikacjach z teorii algorytmów genetycznych (patrz, np. *David E. Goldberg, Algorytmy genetyczne i ich zastosowania. – Wyd-wo Naukowo-*

Techniczne, 2003, a także *Tomasz Dominik Gwiazda, Algorytmy genetyczne, Kompendium, tom 1: Operator krzyżowania dla problemów numerycznych. – MIKOM, 2007* i *tom 2: Operator mutacji dla problemów numerycznych. – MIKOM, 2007*) stosunek prawdopodobieństwa operacji krzyżowania do prawdopodobieństwa operacji mutacji w zwykłych realizacjach wynosi gdieś 1000:1 albo nawet więcej. Jeżeli stosunek ten będzie za mały, to znikają przewagi algorytmu genetycznego jako algorytmu skierowanego poszukiwania i algorytm ten praktycznie nie będzie odróżniać się od metody poszukiwania losowego. W pracy natomiast prawdopodobieństwa te mają ten sam rząd (patrz str. 47, *„the probability of crossbreeding = 0.7, and the possibility of mutation = 0.3 "*), co jest rozwiązaniem sprzecznym, wymagającym szczególnego wytłumaczenia. Co więcej, we wniosku na str. 62 czytamy: *„It has also been noticed that a bigger probability of mutation helps to look for a better system structure, whereas a bigger probability of crossing improves optimization of time criterion."* Czy nie doprowadzi taki wybór do degeneracji algorytmu genetycznego? W świetle tych faktów, wybór ten, co najmniej, ma być uzasadniony.

8. Str. 38: *„The equation of the hyper-plane cutting the hyper-sphere is calculated and the obtained system of linear equations is solved."* O jakie układy równań liniowych chodzi? W tekście monografii tych równań brak.

9. Nieco budzi wątpliwość wynik obliczeń przedstawiony w Tab. 3.9, str. 59. Ilość zadań rośnie, a parametr czasu dla średnich ilości zadań zostaje praktycznie taki sam, a dla 60 zadań jest nawet dwa razy mniejszy niż powiedzmy dla ilości zadań 40. Podobne relacje można spostrzegać również w Tab. 3.11, 3.12, str. 60. Jak można objaśnić taką sytuację?

10. Str. 44. Po co zostały wybrane właśnie te zadania dla wymiany? Czy jest w tym jakaś logika, albo taki wybór jest czysto losowy?

11. Niektóre diagramy w tekście są źle czytelnie albo w ogóle nieczytelne. Np. diagramy przedsatwione na Rys. 3.15, 3.16, 3.17, 3.18 na str. 64, 65 źle czytelne, a diagramy na Rys. 6.5 - 6.11 na str. 128 – 134 są nieczytelne. Nie zrozumiało też co oznaczają liczby na osiach odciętych na Rys.6.6. i 6.7, str. 145.

12. W rozdziale 4.3 autor analizuje problem szeregowania zadań i pomiędzy innym stwierdza (str. 74), że *„In case when we want to transfer a selected task (or <u>its part</u>), the tasks (or <u>their parts</u>) will be swapped."* – "W przypadku, gdy chcemy przekazać wybrane zadanie (lub jego część) zadania (lub ich częśći) będą wymienione." Czyli prawdopodobnie chodzi o dekompozycji zadań(?!) Ale ani w tej części tekstu, ani gdziekolwiek w innych miejscach monografii autor nie opisuje procesy dekompozycji zadań, a więc nie zrozumiało, o jakie części chodzi?

13. Wiele wykresów, tabeli i diagramów w tekście zostały przedstawione bez objaśnienia, skąd biorą się te rezultaty. Np. podrozdział 4..3 na str. 77 zaczyna się z przedstawienia Rys. 4.2. zależności ilości iteracji od ilości zadań. Skąd pobrano te dane, czy są to jakieś dane uśrednione czy też dotyczą jakiegoś konkretnego eksperymenu?

Dokładnie te same pytanie dotyczy wielu innych wykresów i diagramów (np Rys. 4.3 i 4.4 na str. 78. 79, itd.).

14. W opisach algorytmów autor często korzysta ze wspomnienia o specyfikacji systemu jako danych wejściowych (patrz np. opis algorytmu na str. 86: *„After entering the input data (the sysem specification), the algorithm constructs a neural network."*). Warto byłoby jednak zilustrować, w jakiej postaci autor przewiduje wspomnianą specyfikację systemu, ponieważ od tego, jakie dane są na wejściu zagadnienia i w jakiej postaci dane te są przedsawione, byłoby latwiej ocenić specyficzne cechy tych czy inncyh algorytmów.

15. Kolejne pytanie dotyczy strategii posługiwania się sztucznymi sieciami neuronowymi. Jak wiadomo, sieci neuronowe – to klasyczny klasyfikator obiektów, który może być kształcony dla klasyfikacji pewnych klasów obiektów. Kształcenie te w ten czy inny sposób ma być oparte na zasadach statystycznej teorii nauczania (patrz np. *Simon Haykin, Neural Networks. A Comprehensive Foundation. – Prentice Hall, 1998*). Są dwie podstawowe strategii kształcenia sieci neuronowych – „z nauczycielem" (ang. *supervised learning*) i „bez nauczyciela" (ang. *unsupervised learning*). O ile można zrozumieć z tekstu monografii, w tej jej części, która dotyczy sieci neuronowych, jest używana druga strategia. Jeżeli to tak jest, to tak czy inaczej ma być określona miara odległości między obiektami, ocena stopnia kształcenia sieci i inne parametry niezbędne do rozumienia sposobu nauczania sieci. Niestety te ważne dla rozumienia proponowanych algorymów szczególy w monografii są pomijane, co komplikuje obiektywną ocenę rzeczywistego wniosku autora w tę część teorii rozwijanej w pracy.

16. W algorytmach opracowanych w pracy nie przyjmuje się pod uwagę czas wykonania operacji i niejednakowość zadań w sensie czasu ich wykonania. Niestety, w większości sytuacji czas wykonania zadań zasadniczo nawet nie może być dokładnie przewidywany. Przekonują nas w tym wiele praktycznych przykładów. Np. bardzo czężko przewidywać czas rozwiązania problemów poszukiwania albo problemów powiązanych z realizacją procesów iteracyjnych. Sytuacja kompikuje się również tym, że w razie NP-trudnych problemów czas rozwiązania rośnie wykładniczo do złożoności problemu. Niemniej aby poprawnie zrealizować podział zadań wśród obecnych zasobów ze względu na minimalizację czasu rozwiązania problemów, ekonomii kosztów i innych kryteriów deklarowanych w pracy, ma być wiadomo apriori, jaki czas wymaga rozwiązanie każdego konkretnego zadania. Stąd powstają dwa pytania. Po pierwsze, czy nie jest określenie zadań na wejściu każdego algorytmu jako apriori równowartościowych zadań we wspomnianym sensie, nieco sztucznym? I po drugie, w jaki sposób można ocenić przed wykonaniem ciągu zadań, czy są zadania te zależne (niezależne) od siebie? (Tym wiećej, że zależność albo niezależność zadań jest przewidywana w niektórych algorytmach.) Czyli, innymi słowy, aby zoptymalizować procesy podziału zadań wśród zbioru procesorów ma być wiadome cechy tych zadań, bowiem zadania mogą być zbyt różne z punktu widzenia czasochłonności ich wykonania, wymaganej pamięci i wymaganej mocy obliczeniowej procesorów.

17. W tekście monografii w niektórych miejscach brak odsyłaczy do źródeł literackich. Autor niekiedy deklaruje pewne tezy bez niezbędnego ich uzasadneinia. Np. wyrażenie (5.1) na str. 92 (podobnie wyrażenie (5.2) na str. 104) albo ma być udowodnione, albo ma zawierać odsyłacz do źródła, z którego było zapożyczone.

18. Autor nieco wolno operuje wynikami swoich badań teoretycznych, nie zamyślając się nad ich praktyczną wartością. Charakterystycznym przykładem mogą służyć wyniki badań wymaganej ilości sztucznych neuronów w sieci neuronowej przedsatwione w podrozd. 5.5.2 na str.106 – 108. Np. autor odważnie pisze (patrz dane w Tabelach na wspomnianych stronach), że według ocen teoretycznych dla 200 jednostek obliczeniowych ilość neuronów w sieci wynosi 80520, a ilośc łączeń – więcej niż 5 milionów! Ciekawo, w jaki sposób, według autora, o tyle złożona sieć może być praktycznie zbudowana w rzeczywistości?

19. W różnych miejscach tekstu monografii (patr np. Tab. 3.13, str. 63), autor korzysta z następującej terminologii: uniwersalne procesory nazwane są po angielsku „general processors" (a w Tab. 5.6., str. 114 jako „universal processors), a wyspecjalizowane procesory – „dedicated processors". Słowo „general" zwykle używane jest w sensie „ogólny", a słowo „dedicated" dosłownie oznacza „samozapomniany". Natomiast więcej używanym terminem angielskim w sensie „uniwersalny" procesor jest słowo „general-purpose", a w sensie „wyspecjalizowany" procesor – angielskie słowo „special-purpose''. Co więcej, *special-purpose* procesory zawsze orientowane są na skuteczne wykonanie pewnych operacji lub funkcji (klasyczny przykład – arytmetyczne *co-processors*). Natomiast na str. 114, 115 w Tab. 5.6 i 5.7 autor porównia architektury systemów z *general-purpose* procesorami i *special-purpose* procesorami. O jakie konkretnie *special-purpose* procesory chodzi?

20. Nieco nie zrozumiało, na czym polega autorska adaptacja klasycznych algorytmów Ant Colony Optimization (ACO) i Branch&Bound (B&B) opisanych w rozdizle szóstym.

21. Tekst monografii zawiera dużą ilość błędów w języku angielskim, co niekiedy istotnie komplikuje rozumienie sensu zdań. Niżej przedsatwione są kilka charakterystycznych przykładów owych błędów:

(1) Str.17, pierwsze zdanie: *„ ... i.e. systems of type the complex of resources and operations..."* (?).

(2) Str.17: *"This monographs shows ... "* (?)

(3) Str.17: *"Set of tasks..."* . Ma być *"The set of tasks..."*

(4) Str.17: *"... representing precedence constraints."* Ma być *"... representing the precedence of constraints."*

(5) Str. 17, dół: *" ... may be optimizes in terms of costs."* Ma być: *" ... may be optimized in terms of costs."*

(6) Str. 18, Fig.2.2. *"Multicriteria optimization time-cost"* (?), *"Multicriteria optimization power consumption-cost"*(?)

(7) Str.22, góra: *"... which can be executed parallel."* Ma być: *"... which can be executed in parallel."* lub *"... which can be executed concurrently."*

(8) Str. 25: *"Number of solutions generation in cluster"*(?) (Może być *"The number of solutions generated in the cluster"* ?).

"... parameter of algorithm" (Może być: *"parameter of the algorithm"* ?).

(9) Str. 25: *"Application of the same resources clusters aims at separating solutions..."* (?)

(10) Str. 26: *"... the allocated resources in frames of solution cluster."* Ma być: *"... in the solution cluster."*

(11) Str. 27, piersze zdanie: *"This structure describes behavior of tasks in system (attributing and schedule)..."* (?). Może być ma być tak: *"This structure describes behavior of tasks in the system (its attributes and schedule)..."*, chociaż struktura sama w siebie nie może opisywać zachowania systemu (?!)

(12) Str. 27: *„... the relation which occur between the main data structure in algorithm."* Ma być: *„... the relation which occurs between the main data structure in the algorithm."* I kolejne zdanie: *"Cluster solutions describe the structure architecture of the system..."*. W jaki sposób rozwiązania mogą opisywać architekturę systemu i jaki sense tu ma połączenie wyrazowe *„the structure architecture"* ?

(13) Str. 27, dół: *„The structure contains the information about of the graph describing the functional requirements of system."* Może być ma być tak: *„The structure contains information on the graph describing functional requirements on the system."*

(14) Str. 27, dół. Skrót *BFS* podano bez objaśnienia. Objaśnienie te podano w tekście później.

(15) Str.28, dół: *"Parameter global temperature is control parameter for execution of algorithm."* Prawdopodobnie ma być tak: *"The parameter of global temperature is a control parameter for execution of the algorithm."*

(16) Str. 29: *"Each gene in chromosome..."*. Ma być: *"Each gene in the chromosome..."*.

"...a vector describing resource allocation and vector describing...". Ma być: *"...and a vector describing..."*.

"Resource processor is described...". Ma być: *"The resource processor is described..."*.

"Skilful performance of evolutionary algorithm...". Ma być: *"Skilful performance of the evolutionary algorithm..."*

(17) Str. 34, góra: "... form the starting nod to the extreme nod...". Co oznacza słoso nod? Autor ma na myśli *"node"* – węzeł ?

(18) Str. 37: *„Each resource is described with help of..."*. Ma być: *„Each resource is described with the help of..."*.

(19) Str. 40, góra: *"which refers to the criterion of minimization the total tasks completion time..."*. Ma być: *"which refers to the criterion of minimization of the total tasks completion time..."*.

(20) Str. 41: *" ... by the task level in graph determined by..."*. Ma być: *" ... by the task level in the graph determined by..."*.

"We choose the position in frame of determined boundaries...". Ma być: *"We choose the position in the frame of determined boundaries..."*.

(21) Str. 76: *„In the presented algorithm are data structures with chapter 3..."* (?) Jaki sense tego zdania? Chyba ma być tak: *"The data structures used in the presented algorithm are the same as those in chapter 3..."* ?

(22) Str. 85, dół: *"Analogical process applies to operation T_2."* Raciej ma być tak: *"The similar process is applied to operation T_2."*

(23) Str. 185. Autor odsyła Czytelnika do Rysunku 7.47, które w tekście brak.

Lista ta mógłaby być istotnie rozszerzona, bowiem niestety drobne usterki są na większości stron monografii.

Ocena dorobku naukowego

Dane ogólne

Dorobek publikacyjny dr inż. Mieczysława Drabowskiego po doktoracie w roku 1986 obejmuje wraz z monografią habilitacyjną 74 pozycji, z czego 38 pozycje są opracowaniami indywidualnymi.

Statystyka publikacji po doktoracie według aktualnej punktuacji MNiSW i łączna liczba uzyskanych punktów z uwzględnieniem procentowego udziału współautorów jest następująca. Wybrany przez habilitanta podzbiór 25 jego prac punktowanych przez MniSW, to 11 prace indywidualne i 14 prac zostało napisanych we wspólautorstwie, przy czym razem liczba punktów z uwzględnieniem udziału wynosi 131,76 od doktoratu.

Pracy powstałe w okresie ostatnich lat, wg. Scholar Google, ma następujące cytowania:

- Scheduling multiprocessor tasks to minimize schedule length – cytowane przez 200
- Scheduling independent 2-processor tasks to minimize schedule length – cytowane przez 18
- Coherent concurrent tasks scheduling and resources assignment in dependable computer systems design – cytowane przez 5
- Szeregowanie zadań w systemach wielomikroprocesorowych – cytowane przez 1
- Coherent synthesis of heterogeneous system – a Tabu search approach – cytowane przez 2
- Coherent synthesis of real time systems – neural approach – cytowane przez 2
- Coherent concurrent tasks scheduling and resources assignment in dependable system design – cytowane przez 2
- Task scheduling in coherent co-synthesis of computer system – cytowane przez 3
- Minimizing Cost and Minimizing Schedule Length in Synthesis of Fault Tolerant Multiprocessors Systems – cytowane przez 2
- A genetic method for hardware-software par-synthesis – cytowane przez 2
- Information technology for recognition, protection and management of cultural heritage – cytowane przez 1
- Multiprocessor task scheduling with single resource constraints – cytowane przez 1
- Par-synthesis of multiprocessors parallel systems – cytowane przez 2.

Charakterystyka dorobku naukowego

Pracy naukowe dr inż. Mieczysława Drabowskiego powiązane są z problematyką systemów operacyjnych, badań operacyjnych, a w szczególności z problematyką szeregowania zadań i rozdziałem zasobów w systemach komputerowych i problemami optymalnego projektowania wspomnianych systemów w różnorodnych zastosowaniach. W ramach tego głównego nurtu zainteresowań habilitanta dorobek naukowy jest tematycznie bardzo spójny, bowiem większość publikacji po doktoracie były poświęcone właśnie tej tematyce. Badania w dziedzinie teorii systemów operacyjnych autor słusznie uważą jako ważną metodologiczną podbudowę dla rozwoju intuicji i doświadczenia projektanta systemów komputerowych.

W różnych latach po doktoracie habilitant wykonywał badania naukowe jako kierownik (jeden projekt), główny wykonawca (jeden grant naukowy) lub wykonawca projektów badawczych i badawczo-rozwojowych (14 projektów). Ma sześć wyróżnień wynikających z prowadzenia badań naukowych lub prac rozwojowych. Jest członkiem naukowych komitetów solidnych konferencji na poziomie światowym, takich jak IADIS Multi Conference on Computer Science and Information Systems oraz IASTED International Association of Science and Technology for Development. Jest członkiem komitetu wydawniczego czasopism International Journal of Computer Science and Emerging Technologies i Scientific Journal International, recenzentem Journal Software i członkiem PTI.

Dr inż. Mieczysław Drabowski jest autorem lub współautorem 3 monografii, 1 podręcznika akademickiego, rozdziałów w 9 pozycjach książkowych (w tym 2 zagranicznych), 22 artykułów w czasopismach, 9 artykułów w materiałach konferencyjnych krajowych i 35 międzynarodowych oraz 11 publikacji na temat oryginalnych rozwiązań produktów informatycznych.

Na oddzielną uwagę zasługiwuje działalność dydaktyczna dr inż. Mieczysława Drabowskiego. Od roku 1996 zajmując się dydaktyką w Politechnice Krakowskiej, prowadzi przedmioty ściśle powiązane z tematyką pracy naukowej. Wśród tych przedmiotów są następujące:

- przedmioty kursowe (obowiązkowe): Systemy operacyjne, Inżynieria oprogramowania, Przetwarzanie równoległe i rozproszone oraz Sztuczna inteligencja.
- przedmioty wybieralne: Zaawansowane bazy danych, Systemy odporne na błędy oraz Systemy gridowe.

Był promotorem 102 prac dyplomowych, w tym 70 prac magisterskich (8 wyróżnionych) i 32 inżynierskich (1 wyróżniona).

Na innych uczelniach (AGH - Wydział Elektrotechniki, Automatyki i Elektroniki; Politechnika Świętokrzyska; Wyższa Szkoła Europejska; Papierska Akademia Teologiczna; Wyższa Szkoła Ekonomii i Administracji w Kielcach) prowadził zajęcia z Systemów operacyjnych, Systemów multimedialnych, Architektury komputerów i Programowania.

Ocena końcowa dorobku naukowego

Dorobek publikacyjny, a także działalność dydaktyczna dr inż. Mieczysława Drabowskiego są ściśle powiązane z tematyką rozprawy habilitacyjnej. Dotyczy on aktualnie ważnych

problemów inżynierii oprogramowania i jest udokumentowany oryginalnymi publikacjami w renomowanych wydawnictwach krajowych i zagranicznych.

Łącznie cały dorobek – publikacyjny i dydaktyczny – stanowi świadectwo dojrzałości i samodzielności badawczej habilitanta.

Podsumowanie

Oceniając pozytywnie rozprawę habilitacyjną i dorobek naukowy pana dr inż. Mieczysława Drabowskiego stawiam wniosek o dopuszczenie jego do dalszych etapów postępowania kwalifikacyjnego.