dr hab. inż. Lesław Gniewek, prof. PRz
Politechnika Rzeszowska
Wydział Elektrotechniki i Informatyki
Katedra Informatyki i Automatyki
35-959 Rzeszów, al. Powstańców Warszawy 12
tel.: 17 8651613
e-mail: lgniewek@prz.edu.pl

Rzeszów, 16.09.2020 r.

RECENZJA ROZPRAWY DOKTORSKIEJ MGRA INŻ. JERZEGO BIERNACKIEGO ZATYTUŁOWANEJ „ZASTOSOWANIE PARADYGMATU FUNKCYJNEGO DO FORMALNEJ ANALIZY SYSTEMÓW MODELOWANYCH W JĘZYKU ALVIS"

Podstawa formalna wykonania recenzji

Niniejsza recenzja została opracowana na podstawie uchwały Rady Dyscypliny Informatyki Technicznej i Telekomunikacji Akademii Górniczo-Hutniczej.

1. Aktualność i znaczenie tematyki oraz cel rozprawy

Inżynieria oprogramowania to dział informatyki stosowanej obejmujący szeroki zakres działań podejmowanych w procesie wytwarzania oprogramowania. Obejmuje ona cały ten proces rozpoczynając od przygotowania specyfikacji wymagań, przez projektowanie oprogramowania, aż do wdrożenia i utrzymania projektu informatycznego. Bardzo istotnym zagadnieniem w procesie wytwarzania oprogramowania jest jego weryfikacja na poszczególnych etapach produkcji. Wychwycenie błędów na wczesnym etapie znacznie obniża koszty ich późniejszej eliminacji. Dlatego też badania prowadzone nad efektywnymi sposobami weryfikacji wytwarzanego oprogramowania są aktualną i istotną aktywnością naukową. Rozprawa doktorska przygotowana przez mgra inż. Jerzego Biernackiego doskonale wpisuje się w ten trend badawczy. Ukierunkowana jest na formalną analizę modeli systemów przygotowanych z wykorzystaniem języka Alvis. Jest kolejnym, istotnym elementem większego projektu, jakim jest – opracowany i rozwijany w Katedrze Informatyki Stosowanej na Wydziale Elektrotechniki, Automatyki, Informatyki i Inżynierii Biomedycznej AGH – język Alvis. Projekt ten od wielu lat realizowany jest przez grupę badawczą pod kierownictwem Promotora, prof. dr. hab. Marcina Szpyrki. Środowisko Alvis przeznaczone jest do modelowania dyskretnych systemów, w których występuje współbieżność, z możliwością uwzględnienia elementów czasowych.

L.G.

Podstawowym celem rozprawy doktorskiej było opracowanie sposobów umożliwiających skuteczną formalną weryfikację modeli w języku Alvis, ze szczególnym nastawieniem na wykorzystanie postaci pośredniej modelu w języku Haskell. Ponadto, zaproponowane podejścia powinny pozwalać na weryfikację opartą o stan modelu. Cel rozprawy przedstawiono w sposób klarowny. Opracowane sposoby weryfikacji powinny być zintegrowane z istniejącymi narzędziami modelowania w języku Alvis, rozszerzając w ten sposób funkcjonalność tego środowiska o nowe możliwości weryfikacji sporządzonego modelu.

Reasumując, podjęta w rozprawie doktorskiej tematyka jest ważna i aktualna z naukowego punktu widzenia, ma duże praktyczne znaczenie i prowadzi do wzbogacenie możliwości środowiska modelowania systemów współbieżnych, jakim jest język Alvis.

2. Zakres i zawartość pracy

Przedłożona rozprawa doktorska mgra inż. Jerzego Biernackiego składa się z 8 rozdziałów, w tym wstęp i podsumowanie, zawiera stosowną bibliografię oraz 3 dodatki z listingami kodu. W bibliografii zamieszczono 122 pozycje, w tym 8 publikacji, których autorem lub współautorem jest mgr inż. Jerzy Biernacki (poz. 15, 17, 103-105, 108-109 i 115). Ponadto, 2 odwołania kierują na stronę projektu Alvis language (poz. 16, 113), w tym do 67 stronicowej instrukcji języka pt. „Alvis modelling language", której Doktorant jest współautorem.

Rozdział pierwszy stanowi wprowadzenie do poruszanej w rozprawie tematyki. Autor uzasadnił potrzebę weryfikacji wytwarzanego oprogramowania, przedstawił dotychczasowe możliwości weryfikacji modeli przygotowanych w języku Alvis oraz postawił następującą tezę:

„Możliwe jest opracowanie skutecznych algorytmów formalnej weryfikacji własności systemów współbieżnych rozwijanych w języku Alvis z wykorzystaniem paradygmatu funkcyjnego i pośredniej reprezentacji modelu wyrażonej w języku Haskell".

Teza została sformułowana w sposób przemyślany, a Autor w dalszej części pracy w konsekwentny sposób stara się ją udowodnić.

W rozdziale drugim Autor opisał środowisko Alvis oraz podstawy modelowania z wykorzystaniem podsystemów zwanych agentami. Przedstawił również sposób reprezentacji stanu modelu. Opisał etykietowany system przejść, zwany grafem LTS, który w rozdziale 5 będzie stanowić punkt wyjścia do weryfikacji modelu za pomocą narzędzia nuXmv. Na końcu Autor wskazał na dotychczasowe możliwości weryfikacji modelu przy wykorzystaniu rachunku μ i porównał język Alvis do wybranych formalnych sposobów modelowania (sieci Petriego, algebry procesów, automaty czasowe) oraz języka Statechart i środowiska PRISM, wskazując na jego zalety.

W rozdziale trzecim Autor wymienił zalety funkcyjnego języka programowania Haskell, uzasadniając w ten sposób jego wybór jako języka implementacji postaci pośredniej modelu Alvis. Postać ta jest generowana za pomocą narzędzia Alvis Compiler. Autor przedstawił najważniejsze elementy postaci pośredniej i funkcji pozwalających na generowanie grafów LTS (w wersji czasowej lub też nie). Postać pośrednia modelu

J.G.

wyrażona w języku Haskell będzie wykorzystana w rozdziale 7 do stworzenia języka zapytań Alvis Query Language.

Rozdział czwarty poświęcono weryfikacji modelowej, która polega na przeprowadzeniu analizy formalnej wcześniej opracowanego modelu badanego systemu i stwierdzeniu, czy spełnia on wymagania określone w specyfikacji. Jest ona zwykle wymagana w systemach krytycznych, gdzie testowanie wytworzonego oprogramowania jest uważane za niewystarczające. Autor opisał zalety, wady i ograniczenia weryfikacji modelowej, a następnie przedstawił niezbędne elementy potrzebne do takiej weryfikacji. Najpierw zdefiniował i opisał systemy tranzycyjne, które są wykorzystywane do tworzenia modelu systemu. Następnie omówił wybrane logiki temporalne (LTL CTL, RTCTL, rachunek μ), wykorzystywane do jednoznacznej specyfikacji własności analizowanego systemu. Przedstawił również rozwój technik pozwalających na wykorzystanie metod formalnych do przeprowadzania weryfikacji modelowej systemów, a na koniec opisał narzędzia wykorzystywane do tego celu, takie jak: SPIN, NuSMV, nuXmv, CADP, PRISM, UPPAAL.

W rozdziałach drugim, trzecim i czwartym zamieszczono więc niezbędne informacje potrzebne do umiejscowienia prac badawczych podjętych przez Autora. Kolejne trzy rozdziały szczegółowo opisują sposoby, w jaki osiągnął on postawiony cel badawczy.

W rozdziale piątym Autor uzasadnił wybór środowiska nuXmv jako narzędzia do weryfikacji modeli opisanych za pomocą języka Alvis. Przedstawił i szczegółowo opisał uproszczoną formę oryginalnego algorytmu pozwalającego na przekształcenie grafu LTS do języka SVM, akceptowalnego przez nuXmv. Algorytm ten zaimplementował jako moduł w narzędziu Alvis2ModelChecker, umożliwiając w ten sposób weryfikację modeli Alvis z użyciem logik temporalnych obsługiwanych przez nuXmv. Wprowadził również opcjonalną możliwość ograniczenia generowanego modelu. Poprawność zaproponowanego podejścia przetestował między innymi na dwóch modelach rzeczywistych systemów krytycznych (zwrotnicy tramwajowej i centrali sygnalizacji pożarowej). Należy w tym miejscu postawić pytanie, czy testowanie jest wystarczającą metodą potwierdzenia poprawności tego podejścia (patrz pkt. 6 niniejszej recenzji). Otrzymane wyniki wykazały znaczący wzrost czasu ładowania modelu i wielkości wymaganej pamięci RAM, wraz ze wzrostem złożoności modelu (liczby stanów). Autor dokonał optymalizacji algorytmu przekształcania, uzyskując lepsze wyniki wydajnościowe. Niestety, wymagania sprzętowe potrzebne do weryfikacji modelowej bardziej złożonych systemów uniemożliwiają wykorzystanie standardowego sprzętu PC. Rozwiązanie tego problemu Autor opisał w kolejnym rozdziale.

W rozdziale szóstym Autor przedstawił możliwości chmur obliczeniowych, wskazując na zalety ich wykorzystania. Niestety, nie wspomniał o wadach. Zaproponował, aby model procesu utworzyć w środowisku lokalnym, a obliczenia (zarówno przekształcenie do SMV, jak i weryfikację) przenieść do chmury. Do tych obliczeń zaproponował wykorzystanie dwóch odrębnych maszyn obliczeniowych, dobierając ich parametry osobno dla procesu przekształcenia i osobno dla weryfikacji. Praktyczne testy, które Autor przeprowadził na Google Cloud Platform, potwierdziły, że

d.G.

wykorzystując chmury obliczeniowe można przeprowadzić weryfikację modelową nawet bardzo złożonych systemów o liczbie stanów przekraczających 10^5.

W rozdziale siódmym Autor zaproponował jeszcze inny sposób weryfikacji, który nie wymaga generacji grafu LTS, co bardzo przyspiesza ten proces. Pomysł wykorzystania postaci pośredniej w języku Haskell poprzez bezpośrednie użycie funkcji filtrujących wymagałoby od użytkownika bardzo dobrej znajomości zarówno tego języka, jak i wewnętrznej reprezentacji stanów. Byłoby to sprzeczne z założeniami stawianymi środowisku Alvis, które ma być łatwe do opanowania przez inżyniera informatyka. Dlatego też Autor opracował zestaw funkcji pozwalający na weryfikację wybranych własności modelu w języku Alvis z użyciem kompilatora GHC. Wykorzystanie tych funkcji, tworzących Alvis Query Language, jest oryginalnym sposobem uzupełniającym metody weryfikacji modelowej. Liczba możliwych do przebadania własności jest obecnie ograniczona liczbą zapytań, ale wymaga znacznie mniejszych zasobów sprzętowych, co potwierdziły przeprowadzone testy.

Rozdziały piąty, szósty i siódmy są wiec najważniejszymi z perspektywy postawionego celu badawczego i prowadzą do weryfikacji postawionej na początku tezy.

W ostatnim rozdziale Autor dokonał krótkiego podsumowania wyników swoich badań, wymieniając najważniejsze z nich. Zaproponował też możliwe do podjęcia kolejne prace zmierzające do dalszego usprawniania weryfikacji modelowej w środowisku Alvis

3. Ocena pracy

Rozprawa doktorska przygotowana przez mgra inż. Jerzego Biernackiego skonstruowana jest poprawnie, a jej struktura ma logiczne uzasadnienie. Autor na początku wprowadza czytelnika w tematykę pracy, jasno określa jej cel i stawia przemyślaną tezę. Następnie opisuje środowisko Alvis, którego dotyczą badania i określa aktualne jego możliwości związane z weryfikacją modeli, które zgodnie z postawionym wcześniej celem chce poszerzyć. Już informacje zamieszczone w początkowych rozdziałach wskazują na dobre przygotowanie merytoryczne Autora do podjęcia trudnego tematu jakim jest wzbogacenie możliwości weryfikacji modelowej środowiska Alvis. Następnie Autor przedstawia oryginalny algorytm, który pozwala na przekształcenie reprezentacji przestrzeni stanów do postaci SMV, akceptowanej przez narzędzie nuXmv. Algorytm ten implementuje w środowisku Alvis jako narzędzie Alvis2ModelChecker i przeprowadza testowanie swojego rozwiązania. W ten sposób Autor potwierdza zarówno swoją zdolność do sformalizowania własnych pomysłów badawczych, jak i praktyczne umiejętności programistyczne. Wyniki testów wydajnościowych wskazują na ograniczenia otrzymanego rozwiązania. Autor nie zniechęca się takimi wynikami i podejmuje próbę znalezienia rozwiązania, aby jego metoda mogła jednak znaleźć szersze zastosowanie. Potwierdza to dobre predyspozycje Autora do pracy badawczej, która mimo włożenia dużego wysiłku nie zawsze przynosi oczekiwane efekty. Skierowanie swojej uwagi na aktualnie rozwijane możliwości dostępu do efektywnych maszyn obliczeniowych, jakie dostarczają rozwiązania chmurowe, pozwala Autorowi na znalezienie rozwiązania napotkanych trudności i osiągnięcie zmierzonego celu. Jeszcze innym, oryginalnym sposobem na ograniczenie zasobów sprzętowych wymaganych

J.G.

podczas obliczeń, który Autor zaproponował, jest wykorzystanie przygotowanego przez niego zestawu funkcji, które pozwalają na pominięcie procesu generowania grafu LTS i weryfikację wybranych własności modelu bezpośrednio z jego postaci pośredniej w języku Haskell. Powstała w ten sposób koncepcja języka zapytań Alvis Query Language może w przyszłości zostać rozbudowana o wiele nowych funkcji, co w połączeniu z ich integracją z Alvis Compilerem może przynieść nową jakość w sposobie weryfikacji modeli w środowisku Alvis.

Podsumowując, praca stanowi interesującą całość, logicznie prowadzącą do zamierzonego celu badawczego, jakim jest opracowanie sposobów umożliwiających skuteczną formalną weryfikację modeli w języku Alvis. Do realizacji tego wymagającego zadania Autor wykorzystał aktualne możliwości dostępu do dużych zasobów sprzętowych. Za najważniejsze wyniki rozprawy można uznać: (1) opracowanie metody weryfikacji modeli przygotowanych w środowisku Alvis poprzez przekształcenie grafów LTS do formatu SMV i użycie narzędzia nuXmv oraz wykorzystanie chmur obliczeniowych do zwiększenia możliwości weryfikacji bardziej skomplikowanych modeli; (2) opracowanie koncepcji formalnej analizy modeli na podstawie postaci pośredniej w języku Haskell, z wykorzystaniem zestawu funkcji tworzących język zapytań AQL. Ważnym elementami pracy, które podnoszą jej wartość, są: praktyczna implementacja zaproponowanych metod poprzez integrację ich ze środowiskiem Alvis oraz porównawcze analizy wydajności zaproponowanych podejść z wykorzystaniem modeli rzeczywistych systemów krytycznych.

4. Spostrzeżenia i uwagi dyskusyjne

Lektura pracy nasunęła mi kilka uwag i pytań o różnym charakterze:

1) W pracy nie zamieszczono spisu użytych symboli, stąd też pytania dotyczące opisu grafu LTS, który zamieszczono na str. 17. „Dla modelu Alvis, graf LTS jest czwórką: $LTS = (\Re(S_0), T, \rightarrow, S_0)$ ", gdzie relację przejść oznaczono symbolem $\rightarrow$.

   A. W wyrażeniu definiującym relację przejść $\rightarrow$ ponownie pojawia się symbol $\rightarrow$. Czy znaczenie tego symbolu jest takie samo?

   B. W tym wyrażeniu użyto również „S – t". Co oznacza symbol „–" ?

2) Na str. 21 autor napisał: „W języku Alvis każde przejście może mieć indywidualnie przypisany czas trwania. Ponieważ wartości nie są związane z globalnym zegarem, grafy LTS są zwykle skończone." Nasuwa się pytanie, kiedy grafy są nieskończone i co będzie w tym przypadku?

3) Definicję 4.2.3 Autor zatytułował „Ścieżka częściowa". Czy nie jest to jednak definicja skończonej ścieżki częściowej? Jeżeli ścieżka częściowa z definicji miałaby być skończona, to czy wtedy definicja 4.2.4 byłaby poprawna?

4) Definicję 4.2.4 Autor zatytułował „Maksymalna ścieżka skończona". Czy nie jest to definicja maksymalnej ścieżki częściowej?

J.G.

5) W definicji 4.3.2 relacja spełniania „|=" spełnia sześć warunków. Czy w świetle warunku czwartego, warunek trzeci nie powinien trochę inaczej być opisany?

6) Opisując weryfikację modelową w podpunkcie 4.1.3 Autor wskazał jej silne i słabe strony, a opisując chmury obliczeniowe wskazał w podpunkcie 6.1.4 tylko na zalety tych rozwiązań. Czy rozwiązania chmurowe nie mają wad?

7) W rozdziale 5 Autor zaproponował oryginalny algorytm przekształcania grafu LTS do języka SVM. Jeżeli ten algorytm miałby być ważnym elementem wykorzystywanym w weryfikacji modelowej, używanej głównie w systemach krytycznych, gdzie testowanie uważane jest za niewystarczające, to czy za wystarczające można uznać testowanie algorytmu, który ma być używany podczas takiej weryfikacji. Czy nie jest konieczne formalne wykazanie poprawności takiego algorytmu?

8) W podsumowaniu Autor napisał, że do najważniejszych wyników rozprawy należy zaliczyć między innymi „implementację języka zapytań Alvis Query Language, umożliwiającego analizę modeli systemów zawierających powyżej 10^6 stanów...". Czy to oznacza, że nie jest możliwa analiza modeli o mniejszej liczbie stanów? Sformułowanie „mogących zawierać powyżej 10^6 stanów" nie budziłoby wątpliwości.

9) W podsumowaniu Autor jako przyszłe, potencjalne kierunki rozwoju języka i narzędzi modelowania proponuje zestaw dostępnych funkcji AQL rozszerzyć o funkcje pozwalające na jeszcze dokładniejszą weryfikację modelu. Jaka jest w takim razie obecna dokładność tej weryfikacji z wykorzystaniem funkcji AQL?

5. Uwagi edycyjne

Praca pod względem edycyjnym i językowym jest nadzwyczaj niestaranna. Można znaleźć w niej bardzo liczne błędy językowe, interpunkcyjne i tzw. „literówki", co potwierdzają poniższe przykłady:

A) „Literówki": „kojenych" (str. 2), „poleaga" (str. 3), „furmuły" (str. 47), „wspera" (str. 51), „zastaw narzędzi" (str. 71), „noże" (str. 74).

B) Błędy językowe:

- „... uzależnione ich prawidłowego funkcjonowania." (str. 1),
- „... systemu dostarcza za pomocą model checkingu daje matematyczną ..." (str. 3),
- „... wprowadzono koncepcję chmur obliczeniowych oraz jakie prezentują zalety ..." (str. 8),
- „... sieci Petriego są bardzo szybko się rozrastają, ..." (str. 21),
- „... nie jest możliwa niejawne przekształcenie typu." (str. 25),
- „...umożliwia również na wykorzystanie ..." (str. 25),
- „...do weryfikacji w z góry narzuconych narzędziach." (str. 26),
- „..., to wszystkie opierają algorytmach które ..." (str. 47),

A.G.

- „... zostały wykorzystane są funkcje ..." (str. 47),
- „... zwiększa możliwości rozmiary możliwych ..." (str. 51),
- „... jedynie za wykorzystane w zasoby ..." (str. 72),
- „... , to wybranej maszynie można przeprowadzać ..." (str. 77),
- „Innym przykładem może być przykładowo ..." (str. 82).

Braki interpunkcyjne nie przytaczam, ze względu na ich ogromną liczbę.

6. Wniosek końcowy

Rozprawa doktorska przygotowana przez mgra inż. Jerzego Biernackiego potwierdza jego szeroką wiedzę teoretyczną i praktyczną w dyscyplinie Informatyka Techniczna i Telekomunikacja. Autor wykazał się dobrą znajomością aktualnych narzędzi wykorzystywanych do weryfikacji modelowej, umiejętnością praktycznej implementacji zaproponowanego przez siebie algorytmu, czy też języka zapytań AQL. Opracowane przez niego sposoby weryfikacji modeli przedstawionych w języku Alvis stanowią oryginalne rozwiązanie problemu naukowego i potwierdzają umiejętności Autora do samodzielnego prowadzenia pracy naukowej. Większość zastrzeżeń do pracy wynika z niestaranności Autora w redagowaniu swojej rozprawy i braku skrupulatności w opisie swoich dokonań.

Podsumowując, rozprawa doktorska przedłożona przez mgr. inż. Jerzego Biernackiego pt. „Zastosowanie paradygmatu funkcyjnego do formalnej analizy systemów modelowanych w języku Alvis" spełnia wymagania Ustawy z dnia 14 marca 2003 r. o stopniach naukowych i tytule naukowym oraz o stopniach i tytule w zakresie sztuki (Dz. U. nr 65, poz. 595 z późn. zm) i wnioskuję o jej dopuszczenie do publicznej obrony.

Gustaw Gwiela