General and Professional Education
4/2014 pp. 95-102
ISSN 2084-1469

EDUKACYJNE I EKONOMICZNE ASPEKTY ZASTOSOWANIA CYKLU HAMILTONA W PROJEKTOWANIU I TESTOWANIU OPROGRAMOWANIA

EDUCATIONAL AND ECONOMIC ASPECTS OF THE USE OF THE HAMILTON CYCLE IN THE DESIGN SOFTWARE AND SOFTWARE TESTING

Marek Żukowicz
Politechnika Rzeszowska
Wydział Elektroniki I Informatyki
ul. Wincentego Pola 2, 35-959 Rzeszów
e–mail: bobmarek@o2.pl

Abstract: The main purpose of this article is to show how Hamiltonian graphs and their property in the software designing and software testing can be used. In the early chapters we describe regression testing, portability tests, for which the application of Hamiltonian graphs and basic knowledge of graph theory is shown. The main problem of using Hamiltonian graphs is viewed on the example of Polish construction companies, which use auctions. The last chapter shows the advantages of using Hamiltonian cycle in software development and software testing.

Keywords: Hamiltonian graphs, Hamiltonian cycle, regression testing, portability tests, software testing.

Wprowadzenie

Testowanie aplikacji nie jest oczywistą i prostą pracą. W praktyce bardzo często brakuje czasu na to, aby „optymalnie" przetestować aplikację, zanim trafi ona do klienta. Z tym problemem spotyka się sporo firm informatycznych oraz programistycznych. Ostatnimi czasy testowanie oprogramowania stało się bardzo gorącym tematem. Jest to problem, który próbują rozwiązać potężne umysły. Testowanie nigdy nie jest w stanie wykryć wszystkich błędów danego oprogramowania, jednak może dostarczyć informacji o stabilności oprogramowania, zgodności z wymaganiami klienta, czy też zgodności z oczekiwaniami klienta. Trzeba pamiętać, że testowanie nie sprawdza pod kątem wszelkich możliwych warunków początkowych, lecz jedynie z wybranymi warunkami. Testowanie może we wczesnych fazach projektu wykryć defekty oprogramowania. Wczesne wykrycie defektu jest ważne z ekonomicznego punktu widzenia ponieważ gwarantuje niskie koszty naprawy. Tak więc testowanie oprogramowania sprowadza się również do wstępnej analizy wymagań. Oczywiste jest to, że nikt nie może przetestować całej aplikacji od początku do końca, ponieważ czas testów wielokrotnie przekraczałby długość życia oprogramowania lub byłoby to nieopłacalne. Nawet testy automatyczne nie są w stanie zapewnić pełnego pokrycia wszystkich możliwych kombinacji funkcjonalności oraz danych wejściowych w programie. Pojawia się pytanie: Jak zaprojektować system i jak napisać przypadki testowe, aby najbardziej optymalnie sprawdzić, czy wszystkie funkcje w aplikacji działają prawidłowo? Jak optymalnie sprawdzić przenaszalność danych i wiele inny atrybutów w systemie?

Sformułowanie problemu

Podczas projektowania systemów w firmach informatycznych pojawiają się różne propozycje implementowania systemu, klasy, modułu oraz interfejsów pod kątem późniejszych testów. Jednak często okazuje się, że po implementacji, pewna funkcjonalność *X* nie jest użyteczna na tyle, ile wymaga tego użytkownik (użyteczność rozumiemy jako ilość czynności potrzebnych do wykonania konkretnej ścieżki). Pojawia się pytanie: Dlaczego użytkownik nie jest zadowolony z funkcji, która działa? Dlaczego testerzy niechętnie wykonują testy regresji obszaru *X* czy *Y*? Odpowiedź na to pytanie jest następująca: Bo trzeba dużo razy „kliknąć", żeby przetestować właśnie tą funkcję i wymaga to sporo czasu.

W matematyce wprowadzona jest teoria grafów Hamiltona. Okazuje się, że wykorzystanie własności takich grafów oraz ich zastosowanie w systemach informatycznych zwiększa użyteczność oraz ulepsza testowanie pewnych atrybutów funkcjonalnych oraz nie funkcjonalnych. Przedstawione w artykule podejście można stosować na każdym poziomie implementacji oraz na każdym poziomie testów np. poprzez wykorzystanie modelu *V* (fazy testów są powiązane z fazami projektowania oraz wytwarzania oprogramowania).

Regresja i testowanie regresyjne

Regresja to zjawisko utraty konkretnej funkcjonalności powstałe w nowej wersji programu i zwykle skutkujące komunikatem o błędzie, błędem logicznym lub brakiem działania. Do regresji dochodzi wskutek wprowadzania zmian w jakiejś części kodu programu. Skutkiem tych zmian jest błędne działanie innej funkcji programu, która w poprzednich wersjach działała prawidłowo. Aby wyłapać takie defekty, wprowadza się **testowanie regresyjne** podczas stabilizacji wersji. Jest to testowanie, które ma na celu znalezienie błędów regresyjnych. Zwykle wykonywanie testów regresyjnych związane jest z ponownym uruchomieniem zestawu testów, które wcześniej kończyły się poprawnie. Ma ono na celu ujawnienie potencjalnych problemów powstałych na skutek dokonanych zmian.

Przenaszalność danych

Przenaszalność danych to jeden z atrybutów niefunkcjonalnych w systemie. Jest to bardzo ważny atrybut, ponieważ nie może istnieć w systemie zjawisko utraty danych podczas ich przepływu przez różne stany. Ta cecha musi być dobrze przetestowana zanim ktoś zdecyduje się powiedzieć, że dana wersja programu jest już stabilna. W przypadku systemów, które służą do zarządzania kontem bankowym przez przeglądarkę internetową, zjawisko utraty danych poprzez przejście do innego stanu (np. przelew na konto oszczędnościowe) miałoby za sobą ogromne konsekwencje, bardzo niekorzystne dla banku.

Grafy Hamiltona

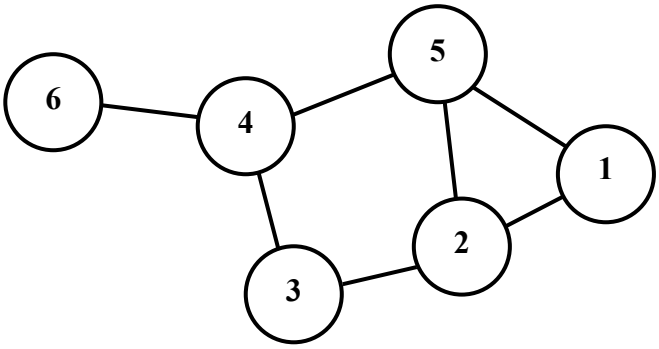
Zaprezentujmy niezbędne wiadomości, które należy poznać w celu zrozumienia najważniejszego problemu opisywanego w następnym rozdziale, a mianowicie zastosowania teorii grafów Hamiltona podczas projektowania systemów informatycznych oraz przypadków testowych. Zaczniemy od cykli Hamiltona:

Definicja 1. *Cykl Hamiltona* to taki cykl w grafie, w którym każdy wierzchołek grafu przechodzony jest tylko jeden raz (oprócz pierwszego wierzchołka) [5]. ***Ścieżka Hamiltona*** (ang. *Hamiltonian path*) jest taką ścieżką w grafie, która odwiedza każdy jego wierzchołek dokładnie jeden raz [5].

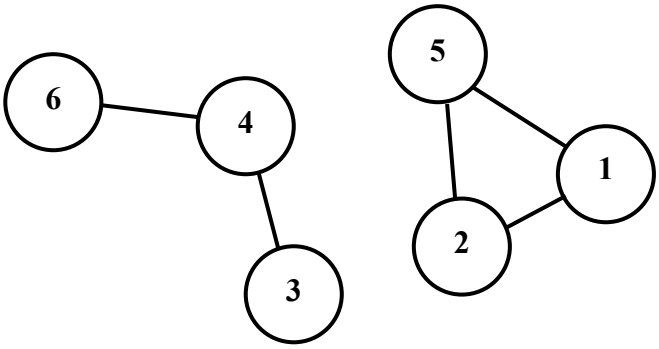
Grafy zawierające cykl Hamiltona nazywamy **hamiltonowskimi**. Graf hamiltonowski to graf rozważany w teorii grafów zawierający ścieżkę (drogę) przechodzącą przez każdy wierzchołek dokładnie jeden raz zwaną ścieżką Hamiltona. W szczególności grafem hamiltonowskim jest graf zawierający cykl Hamiltona, tj. zamkniętą ścieżkę Hamiltona.

Definicja 2. *Grafem spójnym* nazywamy graf spełniający warunek, że dla każdej pary wierzchołków istnieje ścieżka, która je łączy [5].

Graf (rys. 1) jest spójny, więc zgodnie z definicją ma jedną spójną składową. Po usunięciu krawędzi jak na rys. 2 graf ten nie jest już spójny.



Rys. 1. Graf spójny



Rys. 2. Graf niespójny

W celu łatwiejszego zrozumienia dalszej treści, należy znać dwie poniżej przedstawione definicje:

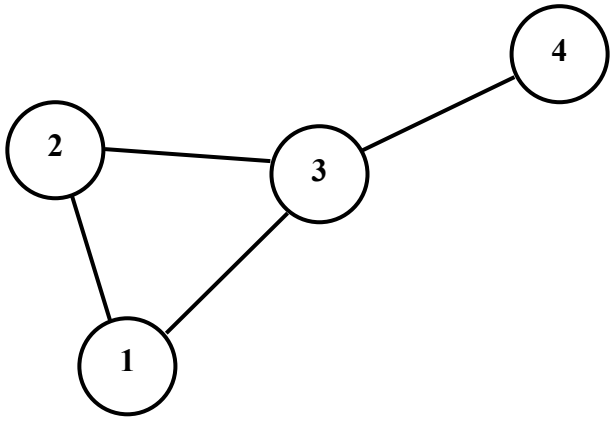
Definicja 3. *Spójną składową* grafu nieskierowanego *G* jest spójny podgraf grafu *G*, nie zawarty w większym podgrafie spójnym grafu *G* [4]. Innymi słowy *spójna składowa grafu* jest to taki podgraf, który można wyróżnić z całego grafu bez usuwania krawędzi. Graf spójny ma jedną *spójna składową*.

Definicja 4. *Grafem prostym* nazywamy graf bez pętli własnych i krawędzi wielokrotnych [3]. Często określenie *graf* (bez przymiotników) oznacza graf prosty.

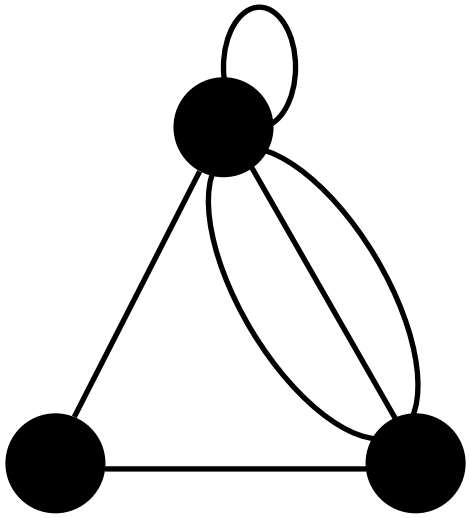
Na rys. 3 przedstawiono graf prosty. Natomiast graf pokazany na rys. 4 nie jest prosty, ponieważ zawiera krawędzie wielokrotne.

Łatwo zauważyć, że graf który posiada cykl Hamiltona, **musi być spójny**. W grafie niespójnym istnieją wierzchołki, pomiędzy którymi brak ścieżki, zatem nie można ich odwiedzić.

Istnieją twierdzenia pozwalające na podstawie cech grafu, dostępnych w czasie liniowym, stwierdzić jednoznacznie, że dany graf jest hamiltonowski. Musimy zwracać jednak uwagę, jaka jest postać twierdzenia, szczególnie, gdy jest to implikacja jednostronna. Okazuje się, że istnieje nieskończenie wiele grafów hamiltonowskich, które nie mają założeń żadnego twierdzenia, które mówi przy jakich warunkach graf jest hamiltonowski. Twierdzenia te są matematycznym obrazem dość naturalnej obserwacji dotyczącej własności grafów - jest logiczne, że im więcej jest krawędzi w grafie, tym "większe są szanse" na znalezienie wśród nich drogi Hamiltona. W skrócie (i nieformalnie), poniższe twierdzenia mówią, że graf jest hamiltonowski, jeżeli tylko ma on odpowiednio dużo krawędzi w stosunku do ilości wierzchołków.



Rys. 3. Graf prosty



Rys. 4. Multigraf (z wieloma krawędziami)

Przedstawimy kilka twierdzeń w postaci implikacji jednostronnej, które określają czy graf jest hamiltonowski. Niech *G* oznacza graf, *V(G)* zbiór jego wierzchołków, *E(G)* zbiór krawędzi, |A| moc zbioru, u_i pojedynczy (w tym przypadku *i-ty*) wierzchołek grafu a *deg(v)* stopień wierzchołka (liczbę kończących się w nim krawędzi). Tradycyjnie oznacza się *V(G)=n* oraz *E(G)=m*, zapisu {*v, u*} będącego zbiorem dwuelementowym wierzchołków, używa się do oznaczenia krawędzi między *v* i *u*.

Twierdzenie 1 (O wierzchołkach w grafie) [5]. Jeśli graf prosty o *n*wierzchołkach ma co najmniej *m* krawędzi, gdzie *m*- ½(*n*-1)(*n*-2)+2 to jest hamiltonowski.

Twierdzenie 2 (Ore) [5]. Jeżeli w grafie prostym *G* o *n* wierzchołkach, *n*>2 zachodzi następująca nierówność: *deg(v)+deg(u)≥n*-1, dla każdej pary **niepołączonych bezpośrednio** krawędzią wierzchołków *u* i *v*, to graf *G* posiada cykl Hamiltona.

Twierdzenie 3 (O grafie nadrzędnym) [2]. Niech *G* będzie grafem o *n* wierzchołkach, a *C(G)* oznacza jego nadgraf zbudowany według reguły mówiącej, że dla każdej pary {*u. v*} **nie połączonych** bezpośrednio krawędzią wierzchołków takich, że: *Deg(u)+Deg(v)≥n* dodaje się krawędź {*u, v*}. Graf *G* jest hamiltonowski wtedy, i tylko wtedy, gdy *C(G)* jest hamiltonowski.

Pierwsze dwa twierdzenia są bardzo podobne do siebie. Idea tych twierdzeń jest taka, że „graf jest hamiltonowski, jeżeli ma odpowiednio dużą liczbę krawędzi". Twierdzenie ostatnie jest nieco inne od trzech pierwszych, ponieważ przedstawia bardzo ciekawą własność związaną z rozszerzaniem grafów hamiltonowskich. Mówi ono, że jeśli wewnątrz grafu znajdziemy

cykl Hamiltona, a pozostałe wierzchołki i krawędzie będą spełniały założenia twierdzenia 3, to nasz rozbudowany graf też jest grafem Hamiltona.

Uwaga 2. Istnieją jednak grafy, które nie spełniają założeń żadnego z powyższych twierdzeń, a zawierają cykl lub ścieżkę Hamiltona.

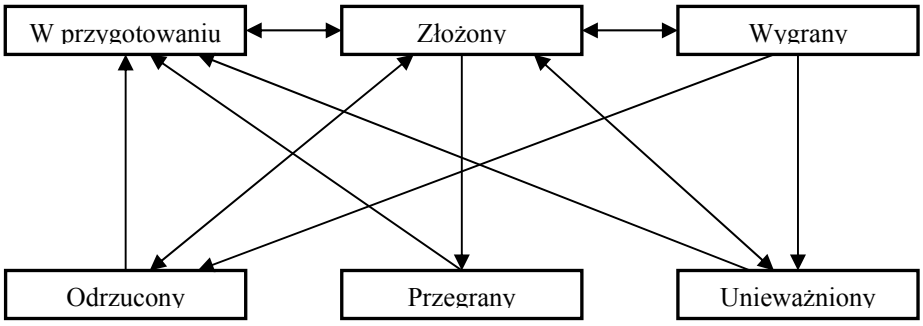
Zastosowanie teorii w praktyce

Wykorzystanie twierdzenia o wierzchołkach grafu w projektowaniu funkcji w systemie na przykładzie polskich firm budowlanych.

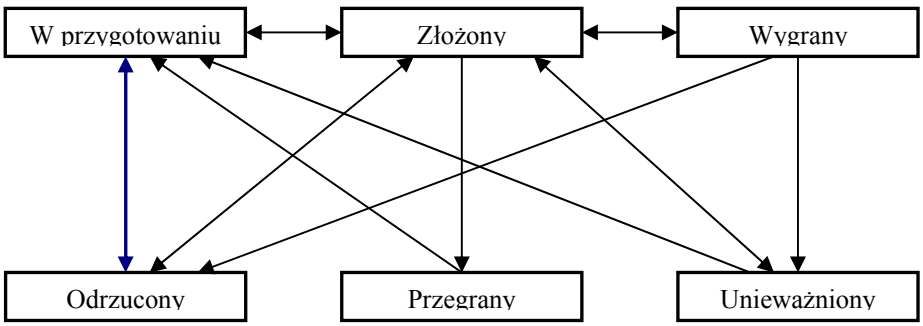
W Polsce, zanim firma budowlana rozpocznie budowę dowolnego obiektu, przygotowuje się do przetargu. Załóżmy, że chcemy napisać program do obsługi przetargów dla polskiej firmy budowlanej, który służy do obsługi przetargów. Zgodnie z prawem muszą być zastosowane następujące zasady:

1) Przetarg dodajemy do systemu ręcznie lub importujemy go ze strony internetowej,
2) Z przetargu zakłada się „projekt w przygotowaniu”,
3) Z projektu w przygotowaniu zakładamy „projekt złożony”,
4) Projekt złożony możemy cofnąć do projektu w przygotowaniu,
5) Projekt złożony może zostać „odrzucony”, możemy go wygrać – czyli staje się „projektem wygranym”, możemy go również przegrać, może ten projekt zostać odrzucony lub unieważniony,
6) Projekt wygrany może zostać odrzucony lub unieważniony,
7) Projekt unieważniony, przegrany lub odrzucony możemy przenieść do projektu w przygotowaniu.

Graf przedstawiający działanie systemu, który opisuje zmianę statusów przetargów, wygląda tak, jak na rys. 5 (przetarg i projekt w przygotowaniu traktujemy jako jeden węzeł, ponieważ nie ma powrotu z projektu do przetargu).



Rys.5. Graf zmiany statusów przetargów



Rys. 6. Graf zmiany statusów przetargów

Graf ten nie jest grafem hamiltonowskim. **Nie jest jednak niezgodne z prawem**, aby umożliwić **„odrzucanie projektu w przygotowaniu”.** Jeśli dopuścimy taką możliwość, to graf zmiany statusów przetargów przyjmie postać jak na rys. 6.

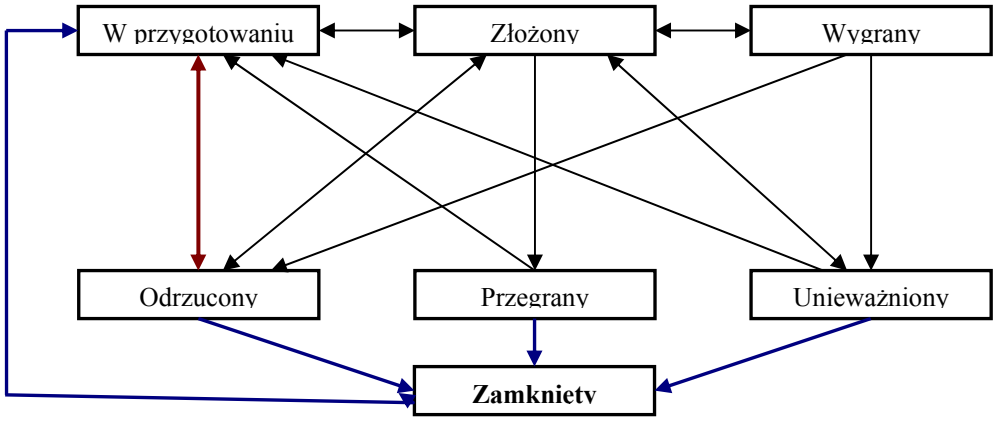
Sprawdzimy teraz, czy graf na rys. 6 spełnia założenia twierdzenia 1. Graf jest prosty.

Ponieważ istnieje w nim zawiera sześć wierzchołków, powinien zawierać 12 wierzchołków, ponieważ: m=1/2·4+2=12.
Graf ma rzeczywiście 12 wierzchołków, a więc z twierdzenia 1 wiemy, że istnieje w nim cykl Hamiltona. Ten cykl to następujący ciąg czynności:

Przegrany → W przygotowaniu → Odrzucony → Złożony → Wygrany → Unieważniony

Załóżmy teraz, że pewna firma działa w taki sposób, że w przypadku rezygnacji z przetargu chce mieć możliwość zamknięcia tego przetargu. Zamknięcie przetargu oznacza, że zostaje o nim informacja w bazie danych, ale nic z nim już nie robimy. W przyszłości jednak możemy otworzyć przetarg i wrócić do projektu w przygotowaniu. Możemy np. dodać zamykanie projektu w przygotowaniu zamykanie projektu odrzuconego, przegranego lub unieważnionego. Wtedy graf przepływu danych przyjmie postać jak na rys. 7.
Okazało się, że zostały dodane ścieżki, które spełniają założenia twierdzenia. Nasz rozbudowany graf nadal pozostanie grafem Hamiltona. Cykl Hamiltona może być np. taki:

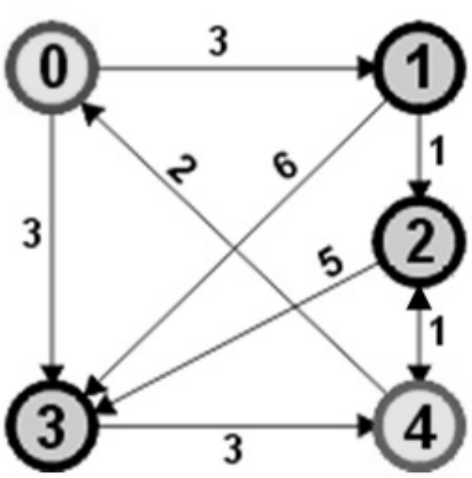
Przegrany → W przygotowaniu → Odrzucony → Złożony → Wygrany → Unieważniony→ → Zamknięty



Rys. 7. Graf zmiany statusów przetargów

Ważone grafy hamiltonowskie

Zastanówmy się, co można zrobić w przypadku, gdy w stanach w systemie lub pomiędzy przepływem danych istnieje więcej niż jeden cykl Hamiltona? W praktyce, zawsze istnieje w aplikacji ścieżka, którą klient wykonuje częściej, niż inne czynności. Problem wyboru ścieżki możemy rozwiązać za pomocą wprowadzenia wag dla krawędzi. Załóżmy, że mamy do zaprojektowania system informatyczny, który działa przenosi dane pomiędzy stanami tak jak pokazano na rys. 8.
Patrząc na twierdzenie 2, jesteśmy pewni, że graf jest hamiltonowski. Interesuje nas w tym przypadku suma wag krawędzi. Kwestia, czy suma wag ma być największa, czy najmniejsza, zależy od kontekstu. Istnieje bardzo dużo algorytmów, które wyszukują cykl Hamiltona w grafie. Jeżeli mamy do czynienia z grafem ważonym, to do algorytmu, który znajduje cykl Hamiltona należy dodać warunek sprawdzania sumy wag krawędzi. Problem sumy wag krawędzi jest analogiczny do **problemu komiwojażera** znanego już od lat.
Jeżeli chcemy najmniejszą sumę wag, to wybierzemy ścieżkę: **1 → 2 → 3 →4 →0,** gdzie suma wag wynosi 7.
Jeśli chcemy największą sumę wag, to wybierzemy ścieżkę: **0 → 1 → 3 → 4 → 2,** gdzie suma wag jest równa 13.



Rys. 8. Graf prosty ważony

Podsumowanie

Ekonomiczne zalety zastosowaniu cykli Hamiltona w projektowaniu i testowaniu

Projektowanie aplikacji pod kątem istnienia w niej cykli Hamiltona w pewnych obszarach czy modułach pozwoli zaoszczędzić czas (którego często brakuje) podczas testowania. Do takich testów należą między innymi testy regresji oraz testy przenaszalności danych (przepływy informacji) między stanami oraz testów regresji w aplikacji. Ten fakt wynika z tego że tester musi wykonać mniej ruchów, jeżeli może wykonać cykl Hamiltona podczas testów, zgłasza manualnych. Sporo testów na całym świecie wykonuje się manualnie, więc oszczędność czasu jest niezmiernie ważną rzeczą. Jeśli wynik testu przepływu danych okaże się pozytywny, to znaczy, że dane przenoszą się poprawnie i nie ma luki ani defektu pomiędzy funkcjami przejścia przez stany. Zmniejszenie czasu wykonywania czy projektowania testów oraz zredukowanie ilości tych testów jest cenione w firmach, które kładą duży nacisk na testowanie.
Takie podejście jest również bardzo korzystne przy projektowaniu i pisaniu testów automatycznych, ponieważ sprawdzanie poprawności przenoszenia danych pisane jest można napisać jeden raz, już po „przejściu cyklu Hamiltona" a dzięki temu również skrócony jest czas pisania takiego testu (przypadku testowego). W cyklu życia oprogramowania pewne funkcje mogą zostać nieco zmienione. W takiej sytuacji testy regresji, które są automatyczne również trzeba zmodyfikować. Ale istnienie ścieżek Hamiltona w przechodzeniu poprzez różne stany również zaoszczędzić czas na zmodyfikowanie skryptów z kodem automatycznych testów. W firmach informatycznych zwiększenie szybkości przeprowadzenia testów pozwala zwiększyć czas związany z zarządzaniem poprawy defektów oraz przez to zmniejszyć ryzyko projektowe (np. niedostarczenie produktu do klienta na czas).
Twierdzenie 3, które podane zostało w rozdziale *Grafy Hamiltona* można zastosować przy dokładaniu nowych funkcjonalności do systemu lub oprogramowania. Warto dołożyć nowe funkcje w taki sposób, że ścieżki Hamiltona będą istniały po ich zaimplementowaniu w oprogramowaniu. Wtedy takie oprogramowanie będzie miało zalety taki, jakie są opisane w dwóch poprzednich akapitach.

Edukacyjne walory wykorzystania cykli Hamiltona w projektowaniu i testowaniu

Dlaczego możemy mówić o edukacyjnych wartościach używania cykli Hamiltona w aplikacji i w testowaniu? Odpowiedź brzmi: szkoła, nauka, a zwłaszcza prace naukowe są po to, aby przygotować człowieka do życia, pomóc mu znaleźć pracę oraz odnaleźć siebie w świecie, w którym nie wszystko jest proste. Osoby, które są związane zawodowo lub prywatnie z testowaniem, projektowaniem systemów lub projektowaniem testów mogą skorzystać z podejścia przedstawionego w

niniejszej pracy, aby ułatwić pracę sobie lub innym osobom. Edukacja testowania ułatwia zmniejszyć koszty, zmniejszyć ryzyko projektowe, zwiększyć czas na zarządzanie poprawą defektów.

Bibliografia

1. Ron Patton., *Testowanie oprogramowania*, Mikom Warszawa 2002.
2. Harrary F., *Graph theory*, Addison – Wesley, 1969.
3. Korzan B., *Elementy teorii grafów i sieci*, WNT, Warszawa 1987.
4. Wilson R., *Wprowadzenie do teorii grafów*, PWN, Warszawa 2000.
5. Włoch I. Włoch A., *Matematyka dyskretna*, Oficyna Wyd. Politechniki Rzeszowskiej, Rzeszów 2008.