Krzysztof Nowak

Algorytmy wywłaszczania ścieżek w sieciach MPLS

Rozprawa doktorska

Promotor:
dr hab. inż. Sylwester Kaczmarek, prof. nadzw. PG
Wydział Elektroniki, Telekomunikacji i Informatyki
Politechnika Gdańska

Gdańsk, 2011
Mojej żonie Agnieszce
1. WSTĘP

1.1. Istota problemu

Poruszane w tej pracy zagadnienie wywłaszczania jest istotnym narzędziem inżynierii ruchu. Korzysta się z niego w sieciach telekomunikacyjnych, w których utworzenie nowego połączenia lub ściężki wymaga dokonania rezerwacji pasma, w sytuacjach, gdy w danym momencie brakuje wolnego pasma. Wywłaszczanie polega na usunięciu, lub inaczej wywłaszczeniu, jednego lub więcej istniejących połączeń o niższym priorytecie, w celu zwolnienia pasma brakującego do utworzenia nowego połączenia o wyższym priorytecie. Usunięte połączenia nie muszą zostać stracone – zostają w miarę dostępnych zasobów utworzone ponownie na alternatywnych trasach.

Wywłaszczanie jest techniką opcjonalną, ułatwiającą gospodarowanie zasobami sieciowymi, zwłaszcza w sieciach o dużej liczbie połączeń, w których niezbędne jest zautomatyzowanie procesu tworzenia ściężek. Nie jest to mechanizm nowy, gdyż znany był już w tradycyjnych sieciach telekomunikacyjnych z komutacją kanałów. Dopiero jednak wraz z wprowadzeniem do nowoczesnych sieci transportowych kanałów wirtualnych o definiowanej przepustowości, wywłaszczanie stało się zagadnieniem nieporównanie bardziej złożonym.

W niniejszej pracy zagadnienie wywłaszczania jest omawiane w powiązaniu z technologią wieloprotoekołowej komutacji etykietowej, czyli Multiprotocol Label Switching (MPLS) [30,96]. Została ona opracowana w pierwszej połowie lat dziewięćdziesiątych, jako uniwersalny mechanizm transportu przeznaczony szczególnie dla sieci opartych na protokole IP. W sieciach IP/MPLS strumienie pakietów są transportowane w ściężkach, identyfikowanych za pomocą etykiet, którymi są identyfikatory liczbowe, umieszczone w dodatkowym nagłówku dołączanym do pakietu. Șcieżkom przyporządkowuje się priorytety, a często również rezerwuje określone pasmo na całej drodze połączeniowej. Z tego względu w dalszej części pracy mówimy o wywłaszczaniu ściężeków.


1 Przyjęto się używać słowa pasmo dla określenia przepływności; zostało to wprowadzone wraz z technologią ATM.
niekontrolowanego czasu wykonania i jest przez to nie do zaakceptowania w jakichkolwiek praktycznych realizacjach. W praktyce wykorzystuje się więc algorytmy nieoptymalne, których zaletą jest przewidywalny i akceptowalny czas wykonania.

Wywłaszczanie może być przeprowadzane w sposób lokalny lub globalny. W pierwszym przypadku decyzja o wyborze ściężek przeznaczonych do wywłaszczenia podejmowana jest zwykle w lokalnym węźle, natomiast w drugim przypadku decyzją zajmuje się zwykle wydzielony blok w sieci. Większość ze znanych algorytmów to rozwiązania lokalne [20,70,72,82] bazujące tylko na informacjach dostępnych lokalnie w każdym węźle. Alternatywne algorytmy globalne [35,71] do wyboru ściężek wykorzystują informacje zebrane z całej zarządzanej domeny MPLS. Podejście globalne umożliwia dokonanie potencjalnie lepszego wyboru, ale wymaga zebrania i analizy większej ilości danych. Wśród pytań, na które nie udzielono dotąd odpowiedzi jest to, czy powszechnie wykorzystywane algorytmy lokalne wywłaszczania dają wyniki porównywalne z algorytmami globalnymi.

W niniejszej pracy zebrano wyniki badań nad algorytmami wywłaszczania w warunkach zbliżonych do tych, jakie występują w rzeczywistych sieciach, przy utrzymującym się wysokim poziomie rezerwacji pasma łączy. Podjęto próbę określenia zysku, jaki może zostać osiągnięty poprzez zastosowanie metod globalnych wywłaszczania w stosunku do metod lokalnych. W tym celu zaproponowano konkurencyjny algorytm globalny i sprawdzono, w jakim stopniu osiągane za jego pomocą efekty pozwalają poprawić jakość wywłaszczania. Uzyskanie odpowiedzi na to pytanie jest niezwykle istotne, gdyż każde wywłaszczenie ściężki może spowodować chwilową lub długotrwałą utratę części komunikacji w sieci. Im lepiej działa algorytm wywłaszczania, tym te straty są mniejsze. W praktyce przekłada się to na jakość usług oferowanych przez dostawcę, rzutuje na jego wiarygodność a pośrednio także na jego wyniki finansowe.

1.2. Cel i teza pracy

Celem pracy jest analiza i praktyczna weryfikacja jakości wywłaszczania oferowanej przez istniejące, w większości lokalne, algorytmy i porównanie ich z wynikami osiąganymi przy użyciu zaproponowanego algorytmu globalnego. Pod pojęciem oceny jakości rozumieć tu porównanie wyników pomiarów wielkości takich jak średnia liczba wywłaszczonych ściężek lub średnia ilość wywłaszczonego pasma.

Aby zrealizować ten cel konieczne jest przybliżenie zagadnienia wywłaszczania, naświecenie problemu badawczego i pokazanie sposobu realizacji wywłaszczania w sieciach MPLS, z uwzględnieniem używanych do tego celu protokołów oraz procedur. W celu przeprowadzenia porównania i dokonania prawdziwej oceny algorytmów należy omówić istniejące rozwiązania, przeanalizować ich sposób działania i dokonać wstępnej analizy. Bazą do tego powinno być formalne zdefiniowanie istotnych wielkości, dzięki czemu uzyska się jednolity i jednoznaczny opis, a w dalszej części ocenę jakościową algorytmów.
Kluczowym elementem pracy jest zaproponowany algorytm wywłaszczania, który używa heurystyki do wyboru możliwe najkorzystniejszego zbioru ścieżek, z punktu widzenia zadanej funkcji celu. Jest to algorytm globalny, korzystający z danych o ścieżkach utworzonych w całej domenie, co potocjalnie daje możliwość dokonania lepszego wyboru niż oferują to algorytmy lokalne. Projektując algorytm dążono do uzyskania jednocześnie wysokiej jakości wyboru ścieżek jak też akceptowalnej złożoności obliczeniowej. Ta ostatnia cecha przekłada się na czas wykonania i jest krytycznym elementem każdego realizowanego algorytmu, gdyż bez udowodnienia akceptowalnego czasu wykonania dla najgorszego przypadku nie można wprowadzić algorytmu do pracy w rzeczywistych sieciach z gwarancją jakości usług.

Aby przeprowadzić wiarygodne badania algorytmów, należy dysponować odpowiednim narzędziem analitycznym lub pomiarowym. W odniesieniu do wywłaszczania, ze względu na duże rozmiary badanych sieci zdecydowanie lepszym rozwiązaniem jest wykorzystanie metody symulacyjnej. Zdecydowano się na implementację własnego symulatora umożliwiającego wszechstronne badanie mechanizmów inżynierii ruchu. W procesie jego projektowania i implementacji szczególne nacisk położono na przenośność, łatwość rozbudowy i szybką interpretację wyników.

Przyjęto metodę badawczą polegającą na przeprowadzeniu badań symulacyjnych ogólne dostępnych algorytmów, przede wszystkim w zakresie liczby powodowanych wywłaszczeń i ilości traconego pasma. Jednym z najważniejszych celów było przy tym określenie, jak zmieniają się wyniki uzyskane przy użyciu różnych algorytmów wywłaszczania w różnych warunkach. Oczekiwany efektem badań powinno być określenie z dużym prawdopodobieństwem, jak zachowują się poszczególne algorytmy w warunkach najbardziej zbliżonych do rzeczywistych.

Celem głównym pracy jest potwierdzenie prawdziwości postawionej tezy, która brzmi następująco:

Zaproponowany przez autora algorytm globalny umożliwia bardziej efektywne wywłaszczanie niż umożliwiają to inne dostępne obecnie algorytmy, w tym najbardziej popularne algorytmy lokalne, uznawane za wysoce efektywne.

Aby osiągnąć ten cel główny postawiono następujące cele pomocnicze:

1. Ocena dostępnych algorytmów wywłaszczania:
   − dokonanie przeglądu istniejących algorytmów,
   − wybór do dalszej analizy kilku najpopularniejszych algorytmów,
   − przeprowadzenie analizy sposobu działania wybranych algorytmów,
   − sprawdzenie możliwości użycia algorytmów w sieciach MPLS i ewentualna modifikacja pod tym kątem,
   − ujednolicenie i sformalizowanie opisu wybranych algorytmów,
   − dokonanie niezależnej analizy złożoności obliczeniowej wybranych algorytmów.

2. Opis zaproponowanego algorytmu wywłaszczania:
– prezentacja zasady działania algorytmu,
– przeprowadzenie analizy porównawczej własnego i istniejących algorytmów,
– analiza złożoności obliczeniowej.

3. Przygotowanie środowiska pomiarowego:
– określenie założeń do środowiska symulacyjnego,
– utworzenie programu symulującego sieci MPLS,
– implementacja wybranych algorytmów,
– implementacja graficznego środowiska do tworzenia scenariuszy badań,
– dokonanie charakterystyki utworzonych aplikacji,
– weryfikacja wiarygodności symulatora.

4. Przeprowadzenie symulacji i analiza wyników pomiarów:
– określenie scenariuszy badań symulacyjnych,
– zdefiniowane kryteriów porównawczych,
– przeprowadzenie symulacji,
– zebranie i interpretacja wyników pomiarów.

Realizacja tak postawionych celów wraz z niezbędnym wprowadzeniem do tematu wymagała odpowiedniej struktury pracy, przedstawionej w kolejnej części pracy.

1.3. Struktura pracy

Wszystkie postawione cele znalazły swoje odzwierciedlenie w strukturze niniejszej pracy, która jest w dalszej części skonstruowana następująco.

W rozdziale drugim przedstawiono ogólne zasady, na których oparta jest architektura sieci MPLS. Rozpoczęto od genezy powstania koncepcji przełączania etykietowego, po czym omówiono te jej cechy, które są istotne dla zrozumienia zasady działania rutera IP/MPLS i sposobu tworzenia ścieżek. Szczególny akcent położono na przedstawienie protokołów tworzenia ścieżek i wyjaśnienie zasad komutacji pakietów. Na tej bazie możliwe było wprowadzenie w kolejnym rozdziale pojęcia wywłaszczania.

Trzeci rozdział poświęcono procedurom wywłaszczania i realizującym je algorytom. Omówiono w nim ogólny schemat realizacji wywłaszczania, a następnie bardziej szczegółowo przedstawiono procedury postępowania towarzyszące algorytom wywłaszczania. Wprowadzono klasyfikację metod na lokalne i globalne oraz przedstawiono różne w obu koncepcjach. Wymieniono najważniejsze parametry jakościowe, niezbędne dla porównania różnych algorytmów. Przedstawiono także stan standaryzacji związanej bezpośrednio z wywłaszczaniem i wprowadzono niezwykle istotne pojęcie priorytetu ściężki.

Dalsza część trzeciego rozdziału została poświęcona istniejącym algorytom. Rozpoczęto od zdefiniowania istotnych wielkości i parametrów opisujących algorytmy, po czym przedstawiono i scharakteryzowano poszczególne ogólne dostępne algorytmy. Opisy uzupełniono
o ujednolicony pseudokod umożliwiający lepsze zrozumienie zasady działania i ewentualną niezależną implementację. Rozdział zamyka wstępną analiza porównawcza algorytmów.

W rozdziale czwartym omówiono opracowany przez autora algorytm wywłaszczania. Wyjaśniono jego cechy i sposób działania. Przedstawiono jego pseudokod w formie zgodnej z pozostałymi algorytmami i dokonano jego wstępnej analizy, w oparciu o wcześniej wprowadzone, jednolite kryteria ocen.

Po dokonaniu prezentacji algorytmów możliwe było przedstawienie środowiska badawczego, którym jest zrealizowany przez autora program symulacyjny. Temu poświęcony jest rozdział piąty. Omówiono w nim założenia do modelu symulacyjnego a następnie przedstawiono sposób, w jaki je zrealizowano, tworząc uniwersalny symulator mechanizmów inżynierii ruchu w sieciach MPLS. Omówiono schemat blokowo-funkcjonalny programu oraz wymieniono zaimplementowane metryki służące do oceny jakości algorytmów. Na zakończenie przedyskutowano przeprowadzone analizy potwierdzające wiarygodność programu, a szczególnie poprawność uzyskiwanych z jego pomocą wyników.

Badaniom algorytmów poświęcono rozdział szósty. Przedstawiono warunki, w jakich prowadzono symulacje i zaprezentowano użyte topologie sieciowe. Następnie, w zasadniczej części rozdziału dokonano analizy uzyskanych wyników badań oraz postawiono wypływające z nich istotne wnioski. Na zakończenie podsumowano rezultaty badań i odniesiono się do tezy postawionej w pierwszym rozdziale pracy.

W ostatnim, siódmym rozdziale przedstawiono wnioski końcowe, podsumowano dokonania i nakreślono kierunki przyszłych badań. Pracę uzupełniono o załączniki zawierające istotne materiały, których nie umieszczono w zasadniczej części pracy. Dołączono bibliografię oraz spis symboli i listę akronimów użytych w tekście.
2. TECHNOLOGIA MPLS

Niniejszy rozdział zawiera omówienie najważniejszych aspektów technologii MPLS, niezbędnych dla zrozumienia specyfiki działania metod wywłaszczania. Przedstawiono czynniki, jakie wpłynęły na ewolucję sieci IP w kierunku IP/MPLS a następnie omówiono najważniejsze założenia architektury MPLS. Omówiono także sposób zestawiania ścieżek oraz dokonano porównania przeznaczonych do tego protokołów.

2.1. Historia

Przełom lat osiemdziesiątych i dziewięćdziesiątych ubiegłego wieku w dziedzinie techniki komputerowej przyniósł gwałtowny rozwój sieci Internet, w którym roczny wzrost przesyłanego ruchu przekraczał 100% [37]. Jego szybkość i skala zaskoczyły nawet twórców związanym z nim technologii, a jednocześnie były wyzwaniem dla producentów sprzętu sieciowego. Równolegle ze wzrostem ilości serwerów sieciowych pojawiały się coraz to nowe aplikacje i usługi, które wymagały coraz większego pasma. Aby sieć była w stanie temu podobać, przeprowadzano rozbudowę sieci szkieletowych oraz zwiększano przepływność łącz. Przewidywano wówczas, że istniejące routery mogą wkrótce nie sprostać prognozowanej szybkości rozwoju sieci i związanemu z tym wzrostowi rozmiaru tablic rutingu. W tej sytuacji poszukiwano rozwiązań umożliwiających przyspieszenie przełączania pakietów IP poprzez zmianę zasady kierowania ruchem. W połowie lat dziewięćdziesiątych szczególne nadzieje pokładano w technikach grupowania pojedynczych połączeń IP w większe strumienie, które mogłyby komutować zbiorczo i dzięki temu znacznie ograniczyć rozmiary tablic kierowania ruchem oraz przyspieszyć proces komutacji [75].

Odpowiednie rozwiązania istniały już w tym czasie w sieciach telekomunikacyjnych, gdzie z powodzeniem stosowano technikę Asynchronous Transfer Mode (ATM) [81,90], w której przesyłane dane były dzielone na komórki opatrzone identyfikatorami VCI i VPI. Odmienne od adresów IP, które pozostają niezmienne w czasie transportu pakietu, wartości VCI/VPI podlegają zmianie na kolejnych łączach, dzięki czemu nie muszą być unikalne w skali sieci. Wartości identyfikatorów są ustalane lokalnie przez przełączniki ATM, co umożliwia optymalizację ich przydziału pod kątem szybkości przełączania i zmniejszenia rozmiarów tablic kierowania ruchem. Technika ATM doczekała się wielu udanych implementacji, a mimo tego nie zyskała pełnej popularności w sieciach IP ze względu na wyższą cenę urządzeń, większy stopień komplikacji, a także niewielki rozmiar komórki, wynoszący 53 bajty, z czego 48 przypadało na dane. Ta ostatnia cecha powodowała konieczność dzielenia pakietów IP na małe kawałki, co zwiększało czas przetwarzania i prawdopodobieństwo strat pakietów w sieci. Niemniej jednak na przełącznikach ATM oparto pierwsze rozwiązania sprzętowe zmierzające do przyspieszenia komutacji IP, do których należały Cell Switching Router (CSR) firmy Toshiba, IP Switching firmy Ipsilon i Aggregate Router-based IP Swi-
tching (ARIS) firmy IBM [30].

Opracowane w pierwszej połowie lat dziewięćdziesiątych rozwiązanie Tag Switching [93] firmy Cisco uznaje się za pierwowzór dla architektury MPLS, z którego zapożyczono jej podstawowe zasady. Pakiet IP na wejściu do sieci został poprzedzony dodatkowym nagłówekiem (tagiem), który określił przynależność do zbiorczego strumienia i był używany do kierowania pakietów, z pominięciem adresu IP. Ciekawą cechą tego rozwiązania było to, że w przypadku zastosowania do transportu przełączników ATM, tag mógł być umieszczony w nagłówku komórki w polach VCI i VPI.


Tymczasem równolegle z rozwojem techniki MPLS nastąpił znaczący postęp w szybkości działania ruterów, wynikający z zastosowania procesorów nowej generacji. W rezultacie główny czynnik, jaki wymusił rozpoczęcie prac nad MPLS, tj. przyspieszenie komutacji, okazał się nieaktualny. Niegdyś przy okazji odkryto jednak, że nowa technika daje niedostępne dotąd w sieciach IP możliwości zarządzania ruchem. W sieciach opartych na MPLS dużo łatwiej tworzyć się sieci wirtualne a ponadto możliwe jest zarządzanie ruchem poprzez rezerwację pasma i przydzielanie ścieżek priorytetów. Rozwój MPLS przyniósł kolejne zaawansowane narzędzia, jak mechanizmy szybkiego przełączania ruchu na wypadek awarii (ang. fast rerouting) lub koncepcja użycia MPLS jako uniwersalnej platformy transportowej dla popularnych technologii warstwy średniej (Ethernet, ATM, Frame Relay). Technologia MPLS stała się szeroko akceptowana i została wdrożona w sieciach wielu operatorów telekomunikacyjnych oraz dostawców usług sieciowych. Obecnie MPLS jest niemal synonimem inżynierii ruchu w sieciach IP [76].

Od kilku lat w rozwój architektury MPLS zaangażowana jest organizacja ITU-T, czego efektem jest powstanie wymagań i standardów opisujących architekturę określoną wcześniej jako Transport MPLS (T-MPLS) [88], na podstawie której wspólnie z IETF opracowano architekturę nazywaną obecnie MPLS Transport Profile (MPLS-TP) [66,67]. Zalecenia ITU-T z jednej strony upraszczają architekturę sieci poprzez ograniczenie ilości opcji i wariantów, a z drugiej strony rozszerzają ją w zakresie m.in. wykrywania uszkodzeń lub separacji sieci sygnaлизacji i danych. Umożliwia to dostosowanie architektury MPLS do standardów obecnych w tradycyjnych sieciach transportowych, opartych na technologii takich jak SDH/SONET lub ATM.
2.2. Architektura MPLS

Technologia wieloprotokołowego przełączania etykietowego MPLS (ang. Multiprotocol Label Switching) jest przeznaczona w ogólności do transportu różnego typu pakietów (stąd określenie „wieloprotokołowe”). Od początku swojego istnienia najważniejszym jej przeznaczeniem było wykorzystanie w sieciach IP, choć ostatnio sporą popularność zdobywają także techniki emulacji łącza PWE3 (ang. Pseudo Wire Emulation Edge-to-Edge) [24,25,106], w których sieci MPLS używane są do transportu ruchu typu ATM, Frame Relay a nawet Ethernet. Ta ostatnia możliwość jest szczególnie interesująca, gdyż umożliwia łatwe tworzenie wirtualnych sieci LAN łączących oddalone od siebie lokalizacje.

Już we wcześniejszych specyfikacjach odnoszących się do warstwy łącza, w ramach architektury MPLS zdefiniowano sposób jej implementacji w technologiach Ethernet i PPP (ang. Point to Point Protocol), a także ATM i Frame Relay (rys. 2.1). W ten sposób niezależnie od posiadanego przez operatora technologii transportowej, możliwa była stosunkowo łatwa migra- cja do MPLS bez konieczności wymiany całej infrastruktury sieciowej.

![Diagram](image)

Rys. 2.1. Model warstwowowy współpracy MPLS z protokołami warstw łącza i sieci.

Każe przesylany w sieci MPLS pakiet IP zostaje wyposazony w nagłówek, który przy- pisuje go do ścieżki LSP. Na łączach typu Ethernet lub PPP dedykowany nagłówek MPLS obejmuje 4 bajty, które zostają „wciśnięte” pomiędzy nagłówek ramki w warstwie łącza a nagłówek datagramu IP – stąd pochodzi angielska nazwa *shim header* (*shim* oznacza klin). Powstały w ten sposób pakiet MPLS został przedstawiony na rys. 2.3. Inaczej tworzy się pakiet MPLS, gdy warstwa łącza zbudowana jest w technologii ATM lub Frame Relay. Nie wprowadza się wówczas dodatkowego nagłówka, ale wykorzystuje istniejące pola nagłówków ATM lub Frame Relay do przenoszenia etykiety. Dokładny opis pól i ich rola zawarte są w RFC 3032 [95] wraz ze zmianami i uzupełnieniami [1,94].

Dedykowany nagłówek MPLS zawiera następujące pola:
- etykieta (20 bitów), określająca przynależność do ścieżki,
- klasa ruchu (TC, 3 bity), do której należy pakiet,
- spód stosu (S, 1 bit), będący wskaźnikiem ostatniej etykiety na stosie oraz
- czas życia pakietu (TTL, 8 bitów), zawierający wartość zmniejszaną o jeden w każdym ruterze; po osiągnięciu wartości zero pakiet jest kasowany z powodu podejrzenia powstania pętli.

Rys. 2.3. Struktura pakietu MPLS w sieci Ethernet (liczby określają rozmiar pól w bitach).

Etykieta (ang. label) jest identyfikatorem liczbowym, który jest podstawową informacją adresową i jest używany do kierowania pakietu MPLS wewnątrz domeny. Etykiety, w przeciwieństwie do adresów IP, mają znaczenie lokalne, to znaczy podlegają zmianie przez rutery na każdym łączu na trasie ścieżki. O tym, jakie konkretne wartości etykiet przypisuje się pakietem, decydują wyłącznie sąsiednie rutery na etapie tworzenia ścieżki. Przestrzeń możliwych wartości etykiet to $10^{20}$ na jednym łączu lub w całej domenie (ang. per-interface lub per-platform), zależnie od wybranej strategii przydziału etykiet. Zatem na jednym łączu lub w domenie teoretycznie może istnieć około miliona ścieżek, choć nie zawsze cały zakres jest dostępny. W przypadku implementacji sieci MPLS na urządzeniach ATM lub Frame Relay pole etykiety jest krótsze niż 20 bitów i zakres etykiet jest przez to odpowiednio mniejszy. Poza tym niezależnie od używanej technologii niektóre implementacje ograniczają liczbę dostępnych etykiet. Dodatkowo etykiety o wartościach od 0 do 15 zostały przez IETF zarezerwowane do celów specjalnych. Część z nich została już zdefiniowana [69,92,95], a inne pozostają do wykorzystania w przyszłości.

Pole TC umożliwia oznaczenie klasy ruchu, do której należy pakiet. We wcześniejszych opracowaniach pole to było oznaczane jako eksperymentalne (EXP), jednak dokument RFC 5462 [4], ustala znaczenie tego pola jako Traffic Class. Jest to z jednej strony zgodne z początkowymi zamierzeniami projektantów architektury MPLS, a z drugiej strony jest usank-
Pole TTL (ang. Time To Live) w nagłówku MPLS ma podobne przeznaczenie, jak jego odpowiednik w nagłówku IP [85]. Jego najważniejszym celem jest wykrywanie powstania zapętleń i eliminowanie pakietów zbyt długo przebywających w sieci. Sposób przetwarzania i kontroli pola TTL z nagłówka MPLS może się różnić od analogicznego postępowania w sieciach IP i dlatego został objęty osobnymi regulacjami [1].

Architektura MPLS zapewnia wsparcie dla sieci hierarchicznych, tzn. umożliwia tworzenie ścieżek wewnątrz ścieżek (tuneli), co m. in. ułatwia kierowanie ruchem oraz budowanie sieci wydzielonych. Funkcjonalność ta jest oparta na mechanizmie stosu etykiet (ang. label stack). Tworzenie stosu polega na umieszczeniu w pakiecie MPLS następujących po sobie kilku nagłówków MPLS (rys. 2.4). Tylko nagłówek ze szczytu stosu, znajdujący się zaraz za nagłówkiem ramki warstwy łącza jest brany pod uwagę przy określaniu drogi, a kolejne wykorzystuje się dopiero, gdy same staną się szczytem stosu po usunięciu etykiet z wyższych poziomów stosu. Gdy pozostaje ostatnia etykieta ze stosu to wartość pola S równa się 1. Po jej usunięciu pakiet przestaje być pakietem MPLS i wówczas zwykle opuszcza domenę MPLS.

Rys. 2.4. Implementacja stosu etykiet poziomu N z użyciem nagłówków dedykowanych.

Standard definiujący architekturę MPLS [96] ze względów historycznych opisuje różne tryby tworzenia ścieżek, z których obecnie najczęściej wykorzystywany jest pulled-conditional (rys. 2.5), który zapewnia utworzenie kompletnej ścieżki. Tworzenie ścieżki inicjowane jest w ruterze wejściowym (ang. ingress), skąd żądanie utworzenia kolejnych części ścieżki jest wpierw kierowane krok po kroku w kierunku ruteru wyjściowego (ang. egress), a następnie w drodze powrotnej akceptowane aż do osiągnięcia rutera wejściowego. Procedura taka ma szereg zalet. Po pierwsze, umożliwia narzucenie drogi, jaką kierowana jest ścieżka, a tym samym jawne jej kontrolowanie. Po drugie, w drodze powrotnej możliwe jest zarezerwowanie pasma dla ścieżki. Po trzecie, potwierdzenie utworzenia pełnej drogi połączeniowej trafia do ruteru wejściowego. W ten sposób w ruterze wejściowym, który inicjuje tworzenie ścieżki, jest gwarancja, że ścieżka została utworzona od początku do końca, a na każdym łączu zarezerwowane jest żądaną pasmo. Charakterystyczna dla oryginalnej architektury MPLS jest zasada tworzenia ścieżki tylko w jednym kierunku [8]. Aby uzyskać komunikację dwu-
kierunkową, konieczne jest utworzenie drugiej ścieżki skierowanej w przeciwną stronę. Zasada ta została zmieniona w nowszych zaleceniach dotyczących rozszerzonych architektur opartych na MPLS, wśród których najważniejsze to GMPLS [12,14] i MPLS-TP [67,88]. Wymaga się w nich, aby ścieżka została utworzona w obu kierunkach.

![Diagram](attachment:diagram.png)

Rys. 2.5. Sposób tworzenia ścieżki w trybie *pulled-conditional*. Cyfry określają sekwencję poszczególnych wiadomości.

Za wymianę wiadomości sygnałizacyjnych związanych z utworzeniem nowej ścieżki jest odpowiedzialny wybrany przez operatora protokół tworzenia ścieżek, zwany protokołem dystrybucji etykiet. Niezależnie od użytego protokołu, tworzenie ścieżki odbywa się w podobny sposób. Ruter wejściowy określa najpierw trasę ścieżki i wysyła żądanie przydzielenia etykiet do następnego ruter. Ten określa kolejny ruter na drodze połączonym i przekazuje do niego podobne żądanie, aż wiadomość dotrze do ruter wyjściowego znajdującego się najbliżej docelowej sieci. Ruter wyjściowy określa wartość etykiety i informuje o niej ruter poprzedni. Podobnie każdy ruter w kierunku powrotnym podejmuje decyzję o wartości etykiety i informuje o niej sąsiada w kierunku do ruter wyjściowego. Jeśli zażądano także rezerwacji pasma, wówczas przed odesłaniem odpowiedzi każdy z ruterów dokonuje stosownej rezerwacji na własnym łączu na drodze ścieżki. Po zakończeniu procedury utworzona w ten sposób ścieżka jest zarejestrowana w tablicach kierowania ruchem poszczególnych ruterów (rys. 2.6). Stąd ruter posiada informację, że przed przesłaniem pakietu do następnego ruteru należy mu przydzielić otrzymaną od sąsiada wartość etykiety.

Komutacja pakietów odbywa się tak, jak pokazano w przykładzie pokazanym na rys. 2.7. Ruter wejściowy po otrzymaniu pakietu IP na podstawie docelowego adresu IP i ewentualnie innych parametrów przydziela go do klasy Forwarding Equivalence Class (FEC). Choć w ogólności możliwe jest do tego celu wykorzystanie dowolnych informacji z pakietu IP, to
zgodnie z najnowszymi zaleceniami, w celu przyspieszenia komutacji, opcjonalne pola nagłówka IP są w tym procesie ignorowane [99]. Po określeniu klasy FEC, na podstawie odpowiedniego wpisu w tablicy FTN (FEC-to-NHLFE, NHLFE = Next Hop Label Forwarding Entry) nadaje się pakietowi etykietę i kieruje go do odpowiedniego portu wyjściowego w kierunku kolejnego ruteru. Ten po otrzymaniu zaetykietowanego pakietu odszukuje parę (etykieta, port wejściowy) w tablicy ILM (Incoming Label Map), na podstawie której dokonuje wymiany etykiety na inną i przekazuje pakiet dalej. Taka procedura jest powtarzana, aż pakiet dotrze do ruteru wyjściowego, który odczyta z własnej tablicy ILM polecenie usunięcia etykiety (etykieta zerowa), po czym prześle pozbawiony etykiety pakiet dalej zgodnie z tablicą rutingu IP. Taki sposób transportu danych przez domenę MPLS jest niewidoczny dla urządzeń końcowych, choć może spowodować zmniejszenie dopuszczalnego rozmiaru pakietu MTU (ang. Maximum Transfer Unit) o wielokrotność czterech bajtów, zależnie od największego rozmiaru stosu etykiet.

![Diagram showing the mechanism of creating forwarding tables in LSR routers.](image)

Rys. 2.6. Mechanizm tworzenia tablic kierowania ruchem w ruterach LSR.

Należy dodać, że w najnowszych zaleceniach ITU-T określających profil transportowy MPLS (MPLS-TP) [21,66,67] przedstawiono wymagania na sieci transportowe MPLS stawiane przez operatorów telekomunikacyjnych:
- tworzenie ścieżek dwukierunkowych o jednakowych lub różnych pasmach w obu kierunkach,
- fizyczną separatcję warstwy transportowej i sterowania,
- niezależność od użytych protokołów sygnaлизacyjnych (patrz rozdz. 2.3),
- możliwość rezygnacji z protokołów sygnaлизacyjnych (sterowanie przez systemy zarządzania),
- tworzenie ścieżek poprzez kilka domen (homogenicznych lub heterogenicznych),
– wsparcie dla różnych typów sieci klienckich, w tym IP, MPLS, FR i ATM.
Wiele z wymienionych wymagań zostało już zrealizowanych w ramach architektury GMPLS [16,33,57,58,64]. Można oczekiwać, że architektura MPLS-TP wyznaczy jeden z ważnych kierunków rozwoju technologii MPLS w ciągu najbliższych kilku lat.

![Diagram](image)

**Rys. 2.7.** Sposób przełączania etykiet w ruterach LSR.

### 2.3. Protokoły sygnalizacyjne

W każdej praktycznej realizacji architektura MPLS bazuje na wielu protokołach, wśród których najważniejsze to protokół rutingu i protokół zestawiania ścieżek, zwany protokołem dystrybucji etykiet.

Protokoły rutingu w sieciach MPLS odpowiadają za rozgłoszenie topologii sieci wraz z informacjami dotyczącymi dostępnych na łączach zasobów. Aktualnie taką rolę spełniają protokoły OSPF-TE [55], IS-IS-TE [98] oraz BGP-TE [77], które bazują na standardowych protokołach znanych z sieci IP, rozbudowanych o mechanizmy rozsyłania dodatkowych informacji o łączach, koniecznych dla wsparcia inżynierii ruchu (TE).

Protokoły zestawiania ścieżek umożliwiają rozgłaszanie etykiet i rezerwację pasma. Zależnie od decyzji operatora i możliwości oprogramowania używanego w ruterach, jest to jeden z protokołów: RSVP-TE, LDP lub CR-LDP. W dalszej części zostaną omówione cechy poszczególnych protokołów.

Oryginalny protokół RSVP (Resource Reservation Protocol) [22] został opracowany w połowie lat 90-tych, w celu zapewnienia mechanizmu rezerwacji zasobów dla poszczególnych strumieni IP. Nigdy w tej postaci nie stał się szeroko rozpowszechniony, ale dzięki otwartej definicji możliwe było jego rozszerzenie o mechanizmy tworzenia ścieżek LSP.

W celu zestawienia nowej ścieżki ruter wejściowy wysyła do rutera wyjściowego wiadomość PATH, zawierającą między innymi żądanie Label_Request umieszczone w obiekcie SESSION. Opcjonalne obiekty to: EXPLICIT_ROUTE (ERO), zawierający zapis żądanej drogi, RECORD_ROUTE (RRO), nakazujący rejestrację listy węzłów pośrednich i zwrócenie jej do rutera wejściowego, oraz SESSION_ATTRIBUTE, zawierający między innymi priorytety ścieżki.

Ruter wyjściowy po otrzymaniu wiadomości PATH, odysła z powrotem wiadomość RESV, która zawiera między innymi obiekt LABEL i opcjonalnie RECORD_ROUTE. Obiekt LABEL zawiera etykietę, która zostaje przydzielona dla ścieżki na danym łączu, natomiast RECORD_ROUTE listę węzłów pośrednich.

Protokół RSVP-TE jest oparty na RSVP, który był w zamierzeniu przeznaczony do obsługi krótkich sesji. Jest oparty bezpośrednio na warstwie IP i nie zapewnia wiarygodnej komunikacji z sąsiadami. W konsekwencji w sieciach MPLS, gdzie ścieżki są z założenia połączeniami długookresowymi, wymagane jest periodyczne odświerzanie informacji o utworzonych ścieżkach. Jeśli w określonym przedziale czasu nie nastąpi odświeżenie, wówczas przyjmuje się, że ścieżka została usunięta. Model taki w MPLS określany jest mianem *soft-state*. Jego wadą jest duża ilość przesyłanych danych sygnalizacyjnych, proporcjonalna do liczby ścieżek i częstości odświeżania. Rodzi to wątpliwości dotyczące skalowalności protokołu RSVP-TE. Niemniej jednak zaproponowano pewne usprawnienia, które spowodowały zmniejszenie ilości informacji niezbędnej do utrzymania ścieżek [15].


Utworzenie ścieżki w LDP odbywa się następująco. Ruter wejściowy wysyła żądanie przydzielenia etykiety w wiadomości Label_Request, zawierającej aktualną długość ścieżki (*hop count*) i opcjonalnie listę ruterów dla wykrywania pętli. Ruter wyjściowy odysła informację zwrotną w wiadomości Label Mapping, zawierającą wymaganą etykietę i definicję ścieżki, obejmującą prefiksy docelowych adresów IP oraz ewentualnie inne dane umożliwiające zakwalifikowanie pakietu IP do klasy FEC skojarzonej z utworzoną ścieżką.

Wszystkie protokoły: RSVP-TE, LDP i CR-LDP mają wiele cech wspólnych, gdyż wykorzystują podobne zasady tworzenia ścieżek. Są jednak trzy obszary, w których ujawniają się podstawowe różnice pomiędzy nimi.

1. **Sposób wyboru drogi.** Protokół LDP zawsze wybiera drogę zgodnie z informacją uzyskaną od protokołu rutingu. Protokoły RSVP-TE i CR-LDP pozwalają natomiast na autorytatywne ustalenie drogi lub wykorzystanie udoskonalonych protokołów rutingu, które w procesie wyboru drogi biorą pod uwagę dostępne pasmo.

2. **Warstwy komunikacyjne.** Protokół RSVP-TE jest oparty bezpośrednio na warstwie IP, używając „zawodnej” komunikacji bez potwierdzeń, natomiast LDP i CR-LDP wykorzystują wiarygodne połączenia TCP do utrzymywania sesji związanego ze stanem ścieżek oraz UDP (multicast) do nawiązania kontaktu z sąsiadami.


### 2.4 Podsumowanie

Koncepcja sieci MPLS została opracowana w celu przyspieszenia przelaczania pakietów w ruterach IP. Pomimo, że we współczesnych ruterach szybkość przelaczania nie jest ograniczeniem, to sieci MPLS posiadają szereg zalet, które predestynują je do zastosowania jako platforma do wprowadzania mechanizmów inżynierii ruchu dla IP, takich jak rezerwacja pasma, sterowanie trasą i mechanizmi odtwarzania po awarii.

Typowym sposobem enkapsulacji pakietu IP w pakiet MPLS jest umieszczenie dodatkowego nagłówka „shim” o rozmiarze czterech bajtów pomiędzy nagłówkami ramki Ethernet a nagłówkiem datagramu IPv4 lub IPv6, choć w warstwie łącza przewidziano również możli-
wość współpracy z protokołami PPP, ATM i Frame Relay. Najważniejszym polem nagłówka MPLS jest etykieta, która przyporządkowuje zbiorczy strumień IP do klasy FEC, która z kolei wiąże go z określoną ścieżką LSP. Etykieta ma formę identyfikatora liczbowego, który ma znaczenie lokalne, gdyż jego wartość zmienia się na każdym kolejnym łączu na drodze ścieżki. Z punktu widzenia warstwy IP sieć MPLS dostarcza tuneli i zapewnia przeźroczysty transport pakietów.

Budowanie ścieżki odbywa się za pomocą wybranego protokołu dystrybucji etykiet. Najpopularniejszym obecnie protokołem jest RSVP-TE, który powstał w wyniku rozszerzenia protokołu RSVP o mechanizmy dystrybucji etykiet. Zapewnia on nie tylko utworzenie ciągłej ścieżki od rutera węźlowego (ang. ingress) do wyjściowego (ang. egress), ale też określenie z góry drogi połączeniowej i rezerwację pasma. Jego wadą jest zasada soft-state, która wymaga periodycznego odświeżania informacji o wszystkich utworzonych ścieżkach, choć w tym zakresie wprowadzono udoskonalenia w celu zmniejszenia ilości wymienianej informacji sygnałizacyjnej. Pozostałe rzadziej używane protokoły dystrybucji etykiet to LDP i jego udoskonalona wersja CR-LDP. Niezależnie od używanego protokołu ścieżka tworzona jest tylko w jedną stronę, co dla zapewnienia dwukierunkowej komunikacji wymaga utworzenia na każdej drodze połączeniowej dwóch przeciwnie skierowanych ścieżek.

W kolejnym rozdziale pokazane zostanie, jak implementuje się wywłaszczanie w sieciach MPLS i jak w tym celu wykorzystuje się priorytet ścieżki. Przedstawione zostaną też ogólnie dostępne algorytmy wywłaszczania i co wynika z ich porównania. Zagadnienia te są bezpośrednio związane z tematem i celem pracy.
3. MECHANIZMY I ALGORYTMY WYWŁASZCZANIA W SIECIACH MPLS

W poprzednim rozdziale omówiono architekturę sieci MPLS w zakresie umożliwiającym teraz na znacznie szersze omówienie jednego z mechanizmów inżynierii ruchu w sieciach MPLS, jakim jest wywłaszczanie. Ten rozdział rozpoczęto od ogólnego omówienia techniki wywłaszczania, po czym dokładnie scharakteryzowano typowe procedury z nią związane. Omówiono najważniejsze regulacje dotyczące wywłaszczania zawarte w dokumentach RFC, a następnie przedstawiono zbiór pojęć ułatwiających opis algorytmów. W zasadniczej części tego rozdziału szczegółowo omówiono znane algorytmy heurystyczne oraz algorytmy optymalne. Na zakończenie dokonano porównania omawianych algorytmów w oparciu o zdefiniowane kryteria.

3.1. Wprowadzenie

W obrębie sieci MPLS połączenia IP są grupowane w strumienie i kierowane utworzonymi wcześniej ścieżkami LSP. Przez każde łącze fizyczne może przebiegać wiele logicznie niezależnych od siebie ścieżek. Jednak mimo tej niezależności ruch transportowany w ścieżkach rywalizuje o dostęp do wspólnego łącza fizycznego, a to, podobnie jak w tradycyjnej sieci IP, wprowadza zjawiska takie jak straty pakietów i zmienne opóźnienie pakietów. O ile w ogólności nie jest możliwe pełne uniknięcie strat, to należy dążyć do ich ograniczania, a jednym ze sposobów jest rezerwacja określonego pasma dla ścieżki.

Rezerwacja pasma oznacza wydzielenie dla ścieżki części wolnego pasma łącza, a zatem po każdej rezerwacji zmniejsza się ilość pasma dostępnego dla ścieżek tworzonych później. W konsekwencji w pewnym momencie może zabraknąć pasma dla nowych zapotrzebowań i będą one odrzucane. Szczególnym problemem jest to wtedy, gdy brakuje pasma dla ścieżki o wysokim priorytecie, której utworzenie jest konieczne. Możliwym rozwiązaniem w takiej sytuacji jest wywłaszczanie, które polega na usunięciu jednej lub więcej utworzonych wcześniej ścieżek o niższym priorytecie, aby zwolnić pasmo dla ścieżki o wyższym priorytecie. W niewielkich sieciach możliwe jest wykonanie tego ręcznie przez interwencję administratora po przeanalizowaniu dostępnych ścieżek. Może to być jednak zdejmutym źródłem narażenia na błędy, a w większych sieciach wręcz niemożliwe. Niezbędna jest zatem automatyzacja tego procesu i tym właśnie zajmują się algorytmy wywłaszczania.

Na rys. 3.1 przedstawiono na przykładzie zasadę działania wywłaszczeń w sieci MPLS. Dla uproszczenia założono, że dla istniejącej ścieżki utworzonej na drodze A-B-C zarezerwowano całe pasmo dostępne na łączach A-B i B-C, tak że nie ma możliwości dokonania na tej samej trasie żadnej innej rezerwacji (1). Dodatkowo założono, że istniejąca ścieżka ma umiarkowane wymagania na wielkość opóźnienia i może w razie potrzeby być przeniesiona

Rys. 3.1. Istota wywłaszczania. Przyjęcie nowej ścieżki o wyższym priorytecie (fioletowa) kosztem istniejącej ścieżki (niebieska).
Mechanizm wywłaszczania związany jest z pojęciem priorytetów ścieżek. Aby dana ścieżka mogła wywłaszczyć inną, musi mieć od niej wyższy priorytet. O przypisaniu priorytetów do ścieżek decyduje dostawca sieci na podstawie kontraktu z klientem.

Dość oczywistym warunkiem wywłaszczania jest to, że na drodze wybranej dla nowej ścieżki przynajmniej na jednym z łącznych wolne pasmo jest niewystarczające. Nie stosuje się wywłaszczania, gdy pasmo jest wystarczające, tzn. w celu jedynie optymalizacji zasobów, gdyż każde przesunięcie ścieżki na nową drogę może być odczuwalne przez użytkowników sieci i objawiać się choćby chwilową utratą komunikacji.

Alternatywą dla wywłaszczania jest podjęcie próby wyboru alternatywnej drogi dla nowej ścieżki, zwykle dłuższej od wybranej w pierwszej kolejności [40,49]. Korzyścią jest oczywiście zmniejszenie liczby wywłaszeń, ale takie podejście może skutkować nieoptymalnym wykorzystaniem zasobów. Stanie się tak, jeśli nowa ścieżka przenosząca ruch wymagający najlepszej możliwej trasy zostanie przyjęta bez wywłaszczania na dłuższej trasie, gdyż najkrótsze trasy będą zajęte przez ścieżki o niższym priorytecie, dla których sposób wyboru trasy może nie mieć istotnego znaczenia.

Można powiedzieć, że rolą wywłaszczania jest uporządkowanie dostępu do zasobów w zależności od wymagań określonych dla poszczególnych ścieżek, minimalizując wpływ kolejności ich tworzenia. Jest to istotna korzyść i ważny aspekt inżynierii ruchu, dlatego wywłaszczanie jest dziś udostępniane przez najważniejszych dostawców sprzętu sieciowego. Należy jednak być świadomym, że dzieje się to kosztem zakłóceń obsługi w ruchu o niższym priorytecie.

3.2. Procedury wywłaszczania


Warto zauważyć, że ponowne utworzenie wywłaszczonych ścieżek może doprowadzić do wywłaszczania kolejnych ścieżek. Mamy wówczas do czynienia ze zjawiskiem kaskady wywłaszeń, które zostało opisane w rozdziale 3.4.6.
Procedura selekcji kandydatów ma szansę zakończyć się powodzeniem pod warunkiem, że na wszystkich łączach, gdzie konieczne jest zwolnienie brakującego pasma, istnieje przy najmniej jeden zbiór ścieżek zapewniający jego odzyskanie. Jeśli na którymś łączu suma

Rys. 3.2. Ogólna procedura utworzenia ścieżki z uwzględnieniem wywłaszczania.
pasm zarezerwowanych dla wszystkich potencjalnych kandydatów jest na to zbyt mała, wówczas nawet wywłaszczenie wszystkich dostępnych ścieżek nie doprowadziłoby do zwolnienia wymaganego pasma. Jeśli rozpoczęto by wywłaszczenie kandydatów zanim algorytm zakończy się sukcesem, to przeprowadzone wywłaszczenia spowodowałyby niepotrzebną uciążliwość dla użytkowników sieci. Ryzyko wystąpienia takiej sytuacji minimalizuje się poprzez wyposażenie algorytmu ruitingu w informacje o poziomach pasma dostępnego dla ścieżek z podziałem na priorytety, tzn. uwzględniając pasmo wszystkich potencjalnych kandydatów do wywłaszczenia. Nie można takiej sytuacji jednak zupełnie wykluczyć, gdy w praktyce informacje w bazie algorytmu ruitingu mogą nie być aktualne, a wówczas algorytm może wskazać drogę, która w rzeczywistości już nie posiada odpowiedniej ilości pasma. Dodatkową przyczyną niepowodzenia może być fakt braku kandydatów na innych łączach wymagających wywłaszczenia. Ewentualność taką można ograniczyć przez użycie globalnych algorytmów wywłaszczenia, ale nie da się jej uniknąć jeśli wykorzystujemy algorytm lokalny, który nie ma informacji o sytuacji na dalszych łączach na drodze połączeniowej.

O ile ogólny schemat działań prowadzących do wywłaszczenia jest dobrze zdefiniowany [7,84,91] i nie wymaga pogłębionych badań, o tyle sposób wyboru kandydatów jest zagadnieniem złożonym, którym różną się poszczególne algorytmy. Trudność polega na tym, że pasma poszczególnych ścieżek zwykle różnią się od siebie, a przez to nie można z góry określić ani tego, ile ścieżek należy wybrać, ani jakie pasma powinny mieć wybrane ścieżki. łatwo bowiem wyobrazić sobie sytuację, w której kilka mniejszych ścieżek lepiej spełni wymaganie na brakujące pasmo niż jedna większa ścieżka, która zwolniłaby zbyt dużo pasma.

Większość z istniejących algorytmów działa według schematu przedstawionego na rys. 3.3. Procedura rozpoczyna się od pustej listy W kandydatów, do której w każdym przebiegu pętli dodaje się wybraną ścieżkę p o pasmie b_p, która najlepiej pasuje do zapotrzebowania określonego przez nową ścieżkę o pasmie b_q. Po każdej takiej operacji oblicza się bilans pasma, tj. określa się, jak zmieni się wolne pasmo, jeśli się wywłaszczy wszystkich kandydatów z listy W. Wprowadzono tu symbole: B na oznaczenie zebranego pasma i c_j na oznaczenie wolnego pasma na łączu. Jeśli bilans jest dodatni, tzn. pasmo po dokonaniu wywłaszczeń będzie wystarczające dla przyjęcia nowej ścieżki, wówczas pętla zostaje zakończona a lista kandydatów jest kompletna, co pozwala zakończyć działanie algorytmu. Najważniejsza i najtrudniejsza zarazem do rozstrzygnięcia kwestia to wybranie najodpowiedniejszego w danej sytuacji kandydata do wywłaszczenia.

Istnieją dwa odmienne podejścia do problemu wyboru ścieżek przeznaczonych do wywłaszczenia, wynikające z zasięgu algorytmu. Zasięg określa to, jak szeroką wiedzą na temat zarządzanej domeny posługuje się algorytmem:

- algorytm lokalny obejmuje pojedyncze łące,
- algorytm globalny obejmuje całą domenę.
Algorytm lokalny przeprowadza analizę w oparciu o stan na bieżącym łączu bez uwzględniania wpływu podejmowanej decyzji na pozostałe łącza ścieżki. Metoda tego typu nie wymaga dostępu do informacji o długości wybieranych ścieżek oraz o ścieżkach dostępnych na innych łączach. Ograniczenie zasięgu do pojedynczego łąca umożliwia znaczne...
uproszczenie algorytmu, jednak kosztem zmniejszenia efektywności rozwiązania w skali całej domeny. Dane wymagane do realizacji metody lokalnej obejmują:

- pasma zarezerwowane dla dostępnych ścieżek,
- priorytety dostępnych ścieżek,
- wolne pasmo na danym łączu (porcie).

Algorytm globalny bierze pod uwagę szerszy kontekst, gdyż „widzi” całą domenę i podejmuje decyzję o wyborze kandydatów po przeanalizowaniu stanu na całej drodze połączeniowej. Takie podejście potencjalnie umożliwia osiągnięcie lepszych rezultatów w sytuacji braku pasma na więcej niż jednym łączu, jednak kosztem zwiększenia złożoności obliczeniowej. W stosunku do metody lokalnej następujące dane mogą być dodatkowobrane pod uwagę:

- trasy dostępnych ścieżek,
- wolne pasmo na łączach na wybranej drodze.

Przedstawiony podział jest analogiczny do tego, jaki wprowadzono do klasyfikacji protokołów rütgungu IP. Odpowiednikiem algorytmów lokalnych są metody typu distance-vector (np. RIP) natomiast odpowiednikiem metod o zasięgu domeny są protokoły typu link-state (np. IS-IS, OSPF).

Podział algorytmów na lokalne i globalne może sugerować, że pierwsze z nich są realizowane jako metody zdecentralizowane, a drugie jako centralizowane. W rzeczywistości zasięg i sposób realizacji są od siebie niezależne. To sposób realizacji określa, gdzie w sieci wykonywany jest algorytm. W przypadku metod centralizowanych jest to dedykowany blok, w którym określa się ścieżki przeznaczone do wywłaszczania. Z kolei w metodach zdecentralizowanych każdy ruter podejmuje decyzję niezależnie. Algorytm globalny może być elementem zarówno metody centralizowanej, jak i zdecentralizowanej. Różnica wynika z tego, czy wszystkie wymagane przez algorytm informacje są dostępne w każdym ruterze, czy tylko w jednym, dedykowanym urządzeniu w domenie. Z kolei do realizacji algorytmu lokalnego naturalnym podejściem jest metoda zdecentralizowana. Podejście centralizowane, choć teoretycznie możliwe do zrealizowania, byłoby tu jednak nielogiczne.

Warto zauważyć, że w kontekście ostatnich prac IETF nad architekturą Path Computation Element (PCE) [34] algorytmy globalne zyskują na znaczeniu. Zgodnie z koncepcją PCE, zadania związane z wyznaczeniem tras w domenach MPLS i GMPLS mogą zostać wyniesione poza ruter brzegowy, do dedykowanego serwera w domenie. Ten na podstawie określonych przez ruter brzegowy wymagań zwraca informację o trasie. Umożliwia to wykonywanie złożonych czasowo zadań bez angażowania mocy obliczeniowej procesorów rutera. Architektura PCE wydaje się być odpowiednia także do wykonywania obliczeń związanych z wywłaszczaniem, analogicznie do roli bloków centralnych w projekcie serwera sterowania połączeniami [54].

Implementując dowolny algorytm, niezależnie od zasięgu, definiuje się w nim w sposób
jawny lub niejawny miarę porównawczą dla oceny poszczególnych zbiorów kandydatów. Chodzi o wybranie najlepszego zbioru kandydatów spośród wszystkich dostępnych ścieżek. Ocena zależy od wybranego kryterium porównawczego, którym może być:

- minimalizacja liczby wywłaszczonych ścieżek,
- minimalizacja sumy pasm wywłaszczonych ścieżek,
- minimalizacja sumy pasm sieciowych wywłaszczonych ścieżek (iloczynu pasm i długości ścieżek),
- maksymalizacja (w sensie liczbowym) sumy priorytetów wywłaszczonych ścieżek,
- minimalizacja długości wywłaszczonych ścieżek.

Lista powyższa nie jest zamknięta i możliwe jest zdefiniowanie innych kryteriów. Dodatkowo zwykle możliwe jest łączenie kilku kryteriów, np. minimalizacja liczby ścieżek, a w przypadku dostępności kilku równoważnych zbiorów, minimalizacja pasma. Należy dodać, iż niektóre metody umożliwiają użytkownikowi dokonanie wyboru preferowanego kryterium.

To, na ile algorytm globalny umożliwi wybranie lepszego zbioru kandydatów niż algorytm lokalny, zależy od liczby łączy, na których nastąpić ma wywłaszczenie. Jeśli dotyczy to tylko jednego łącza, wtedy algorytm lokalny poradzi sobie równie dobrze jak globalny. W przypadku, gdy łącza wymagające wywłaszczenia jest więcej, wówczas dobry algorytm globalny w ogólności powinien dawać lepsze rezultaty.


3.3. Wywłaszczanie w świetle standardów

Przed omówieniem poszczególnych dostępnych algorytmów, zostanie przedstawiony stan wymagań i zaleceń określających sposób przeprowadzenia wywłaszczeń.

W odniesieniu do sieci MPLS, najciekawsze, choć częściowo już nieaktualne założenia dotyczące wywłaszczenia ścieżek zostały zawarte w dokumencie RFC 2702 [9], w którym zdefiniowano cztery atrybuty ścieżek: preemtior enabled (1), non-preemptor (2), preemptable (3) oraz non-preemptable (4). Właściwość (1) oznacza możliwość spowodowania wywłaszczenia innych ścieżek natomiast (2) brak takiej możliwości. Z kolei (3) odpowiada za
możliwość zostania wywłaszczonym przez inną ścieżkę, a (4) wyklucza taką ewentualność. Za domyślnie uznaje się atrybuty (2) i (4), które wspólnie wykluczają powstanie wywłaszczeń.

W tym samym dokumencie określił się, że ścieżki posiadają priorytet, który w połączeniu z odpowiednim atrybutem może spowodować wywłaszczenie. Określono także pięć niezbędnych warunków wywłaszczenia ścieżki $p_2$ przez ścieżkę $p_1$:

1. $p_1$ ma wyższy priorytet niż $p_2$,
2. $p_1$ domaga się zasobów zajętych przez $p_2$,
3. zasobów jest zbyt mało, aby przyjąć jednocześnie ścieżki $p_1$ i $p_2$,
4. $p_1$ posiada atrybut *preemptor enabled* oraz
5. $p_2$ posiada atrybut *preemptable*.

W świetle późniejszych rekomendacji atrybuty typu *preemptor* i *preemptable* należy uznać za nieaktualne. Ich rolę zajęły regulacje określone w standardzie RSVP-TE [7], który jest najpopularniejszym narzędziem używanym do rezerwacji pasma w sieciach MPLS. Zdefiniowano w nim dwa typy priorytetów, deklarowanych w procesie tworzenia ścieżki:

- priorytet utworzenia (*setup priority*), oraz
- priorytet utrzymania (*holding priority*).

Znaczenie tych parametrów jest następujące. Priorytet utworzenia określa priorytet ścieżki na etapie jej tworzenia, tzn. określa zdolność do wywłaszczenia innych ścieżek. Z kolei priorytet utrzymania określa priorytet przypisany ścieżce po jej utworzeniu, tzn. decyduje, w jakim stopniu dana ścieżka jest „odporna” na wywłaszczenie przez inne ścieżki. Oba priorytety określone są wartościami liczbowymi z zakresu od 0 do 7, przy czym 0 oznacza najwyższy priorytet a 7 – najniższy. Wymaga się, aby priorytet utworzenia nie był wyższy niż priorytet utrzymania, gdyż w przeciwnym wypadku grozi to powstaniem nieskończonych pętli wzajemnych wywłaszczeń. W niektórych praktycznych realizacjach jako domyślny uznaje się priorytet utworzenia równy 7 oraz priorytet utrzymania równy 0, co gwarantuje brak możliwości spowodowania wywłaszczeń jak też pełną odporność na wywłaszczenia ze strony innych ścieżek niezależnie od ich priorytetu.

Dla porządku warto dodać, iż koncepcja wywłaszczania istniała już w oryginalnym protokole RSVP zaproponowanym wcześniej niż powstała technologia MPLS, gdzie zdefiniowano parametry *Preemption Priority* oraz *Defending Priority* [41]. Oba priorytety mogły przyjmować wartości z zakresu liczba 16-bitowych bez znaku, przy czym wyższa wartość odpowiadała wyższemu priorytetowi. W kontekście sieci MPLS specyfikację tę należy uznać za nieaktualną na rzecz standardu RSVP-TE.

W odniesieniu do samej procedury wywłaszczania standard RSVP-TE określa, że ścieżka, która podlega wywłaszczeniu, zostaje natychmiast skasowana. Obecnie podejście takie określone jest jako *hard preemption*, w przeciwieństwie do nowszej koncepcji, tzw. *soft preemption*, zdefiniowanego w RFC 5712 [65]. W tym ostatnim podejściu ścieżka przeznaczona do wywłaszczenia nie jest od razu kasowana, ale zostaje skierowana na nową drogę. Ścieżka
na starej drodze pozostaje jeszcze przez pewien czas aktywna, dzięki czemu nie zostają stracone pakiety, które w momencie wywłaszczenia znajdują się na starej trasie. Możliwość taka jest już udostępniona przez czołowych producentów ruterów MPLS [3,36]. Należy jednak podkreślić, iż wykorzystywanie techniki soft preemption nie daje żadnej gwarancji uniknięcia przerw w ruchu. Mogą one wciąż się zdarzyć z powodu zaburzenia kolejności dostarczania pakietów [29] lub przejściowego powstania nadmiernych strat.

Mechanizm wywłaszczania opisano także w specyfikacji protokołu CR-LDP [43]. Tutaj również użyto analogicznych do RSVP-TE pojęć setupPriority i holdingPriority o wartościach z zakresu od 0 do 7, gdzie 0 oznacza najwyższy priorytet ścieżki. Dokument zawiera zalecenie, aby domyślne wartości obu parametrów wynosiły 4.

Przedstawione zalecenia mają swoje odzwierciedlenie w zasadzie działania algorytmów wywłaszczania w sieciach MPLS. Można je podsumować następująco.

1. Każda ścieżka posiada dwa typy priorytetów: priorytet utworzenia (setup) i priorytet utrzymania (holding).
2. Priorytetem są liczbami całkowitymi z zakresu od 0 do 7, tzn. jest osiem różnych poziomów priorytetów. Wartość 0 oznacza najwyższy, a 7 najniższy priorytet.
3. Priorytet utrzymania nie może być niższy (liczbowo wyższy) od priorytetu utworzenia (ale może być równy).
4. Aby ścieżka $p_1$ mogła dokonać wywłaszczenia ścieżki $p_2$ muszą być spełnione trzy warunki:
   - priorytet utworzenia $p_1$ musi być wyższy (liczbowo niższy) od priorytetu utrzymania $p_2$,
   - $p_1$ domaga się zasobów zajętych przez $p_2$, oraz
   - nie ma możliwości przydzielenia zasobów dla $p_1$ i $p_2$ jednocześnie.

### 3.4. Definicje

Jednoznaczny opis algorytmów nie byłby możliwy bez omówienia i formalnego zdefiniowania używanych pojęć i wielkości. W niniejszej części zebrano te definicje, które są niezbędne do jednoznacznego opisu algorytmów wywłaszczania i ich jakości.

#### 3.4.1. Jakość algorytmów

Oceniając jakość algorytmów oceniamy tu zdolność do generowania takich zbiorów kandydatów do wywłaszczenia, dla których koszt wywłaszczenia jest jak najmniejszy.

Definicja kosztu wywłaszczenia jest zależna w dużej mierze od technologii, w jakiej algorytm jest zaimplementowany. Jeśli będzie to sieć, w której usunięcie ścieżek jest kosztowne z technologicznego lub biznesowego punktu widzenia, wówczas koszt będzie prawdopodobnie zdefiniowany jako liczba wywłaszczeń. W innej sytuacji może to być priorytet wywłaszczonych ścieżek. Najczęściej spotykane, choć nie jedyny, definicje kosztów to:
− liczba wywłaszczeń, w którym dąży się do minimalizacji liczby wybranych kandydatów,
− wywłaszczone pasmo, w którym dąży się do minimalizacji sumarycznego pasma kandydatów,
− wywłaszczone pasmo sieciowe, w którym dąży się do minimalizacji liczby włączonego pasma kandydatów,
− priorytety wywłaszczonej ścieżki, w którym dąży się do wyboru w pierwszej kolejności ścieżek o najniższym priorytecie.

Jeśli algorytm pozwala na wybór definicji kosztów, wówczas decyzję podejmuje zwykle administrator sieci, zwany tutaj użytkownikiem, bazując na znajomości technologii i otoczenia biznesowego w ramach zarządzanej domeny MPLS. Mówimy wówczas, że użytkownik ma możliwość podania kryterium wyboru kandydatów.

Zdefiniowana w ten sposób jako duża część algorytmu wynika w największej części ze sposobu wyboru ścieżek przeznaczonych do wywłaszczenia, a więc tego, w jaki sposób algorytm został zaprojektowany. Niezależnym od algorytmu czynnikiem wpływającym na jego jakość może być sposób definicji kosztu przez użytkownika, o ile algorytm daje taka możliwość, a jednocześnie przełożenie kosztu na sposób konfiguracji algorytmu nie jest oczywiste. Na jakość może mieć również wpływ sposób implementacji, o ile dokonano zmiany w stosunku do oryginalnej specyfikacji algorytmu.

Jakość algorytmów jest trudna do oszacowania w drodze teoretycznej analizy jego zasady działania, ale może być zbadana np. przy użyciu metod symulacyjnych. Pewną trudność w tym przypadku sprawia określenie, czy osiągany wynik świadczy o dobrej czy złej jakości, gdyż nie dysponujemy wartościami odniesienia. Możliwe jest jednak określenie jakości algorytmów poprzez badania porównawcze. W ten sposób określiła się, że algorytm, który osiągnął niższy średni koszt wywłaszczenia, jest lepszy.

### 3.4.2. Sieć i ścieżki

Sieć jest opisana przez graf \((V, E, C)\):

− \(V\) jest zbiorem węzłów sieci; \(V = \{v_1, v_2, \ldots, v_n\}\), gdzie \(N = |V|\) jest liczbą węzłów,
− \(E\) jest zbiorem łączy w sieci; \(E = \{e_1, e_2, \ldots, e_l\}\), gdzie \(L = |E|\) jest liczbą łączy; wszystkie łącza są jednokierunkowe,
− \(C\) jest zbiorem pasm (przepływności) łączących fizycznych w sieci; \(C = \{c_1, c_2, \ldots, c_L\}\); jeśli przepływności wszystkich łączy są równe to oznaczamy je symbolem \(c\).
Gęstość $d$ sieci jest to stosunek liczby łącz fizycznych (jednokierunkowych) do liczby węzłów sieci.

$$d = \frac{L}{N} \quad (3.1)$$

Długość $l$ ścieżki jest rozumiana jako liczba łącz pośrednich, z których składa się droga (trasa) ścieżki LSP, od rutera wejściowego do wyjściowego.

Podobnie zdefiniowana jest długość drogi. Jest to liczba łącz pośrednich, z których składa się ścieżka $LSP$, od rutera wejściowego do wyjściowego. Pojęcie długości drogi może dotyczyć konkretniej ścieżki, ale może też być użyte bez powiązania ze ścieżką.

Ścieżka $p$ jest opisana przez cztery wielkości $P_p = (R_p, b_p, s_p, h_p)$:

- $R_p$ jest listą łącz, na których ścieżka została utworzona, $R_p = [e_p^{(1)}, e_p^{(2)}, \ldots, e_p^{(l_p)}]$, gdzie $l_p = |R_p|$ jest długością ścieżki, wszystkie ścieżki są jednokierunkowe,
- $b_p$ jest pasmem (przepływnością) zarezerwowanym dla ścieżki,
- $s_p$ jest priorytetem utworzenia (setup) ścieżki, $s_p \in \{0, 1, \ldots, 7\}$, przy czym 0 oznacza najwyższy priorytet,
- $h_p$ jest priorytetem utrzymania (holding) ścieżki, $h_p \in \{0, 1, \ldots, 7\}$, przy czym 0 oznacza najwyższy priorytet.

Zbiór $P$ zawiera ścieżki utworzone w sieci $P = \{P_1, P_2, \ldots, P_Z\}$, gdzie $Z$ jest liczbą ścieżek.

Aby uprościć zapis pewnych definicji, zdefiniujemy ogólną funkcję testującą $\sigma()$, która dla danej funkcji logicznej $f()$ przyjmię wartość jeden wtedy, gdy funkcja $f()$ będzie prawdziwa, a zero w przeciwnym wypadku, tj. gdy warunek w funkcji $f()$ nie będzie spełniony.

$$\sigma(f()) = \begin{cases} 1 & \text{gdy } f() \text{ jest prawdziwe} \\ 0 & \text{gdy } f() \text{ jest nieprawdziwe} \end{cases} \quad (3.2)$$

### 3.4.3. Wolne pasmo i dostępne pasmo

Wolne pasmo $A_f$ w sieci: $A_f = [a_f^{(1)}, a_f^{(2)}, \ldots, a_f^{(L)}]$ oznacza pasma na poszczególnych łączach, jakie nie zostały jeszcze zarezerwowane i są dostępne dla nowej ścieżki bez potrzeby wywłaszczenia.

Zajęte pasmo $A_{oc}$ w sieci, $A_{oc} = [a_{oc}^{(1)}, a_{oc}^{(2)}, \ldots, a_{oc}^{(L)}]$ jest pasmem zarezerwowanym dla ścieżek na poszczególnych łączach.

$$a_{oc}^{(e)} = \sum_{p \in P} \sigma(\epsilon \in R) b_p \quad (3.3)$$

Dostępne pasmo $A$ w sieci określa pasma dostępne na poszczególnych łączach: $A = [A^{(1)}, A^{(2)}, \ldots, A^{(L)}]$, gdzie $A^{(e)}$ to pasmo dostępne na łączu $e$, $e \in \{1, \ldots, L\}$. Dostępne pasmo jest ilo-
ścią pasma, jakie może zostać przydzielone nowej ścieżce przy uwzględnieniu wywłaszczania. Pasmo dostępne jest tablicą ośmioelementową \( A^{(e)} = [a_0^{(e)}, a_1^{(e)}, \ldots, a_7^{(e)}] \). Na tej bazie definiuje się pasmo \( a_i^{(e)} \) dostępne na łączu \( e \) dla ścieżki o priorytecie utworzenia \( s \), które jest równa wartości wolnego pasma \( a_i^{(e)} \) na łączu \( e \) powiększonej o sumę pasm ścieżek o priorytecie utrzymania \( h_p \) niższym (liczbowo wyższym) niż \( s \):

\[
a_i^{(e)} = a_i^{(e)} + \sum_{p \in P} \sigma(e \in R_p) \sigma(h_p > s) b_p ,
\]

gdzie \( s \in \{0, 1, \ldots, 7\} \).

Ponieważ każda ścieżka o priorytecie utworzenia równym \( s \) może wywłaszczyć każdą ścieżkę o priorytecie utrzymania \( h \) większym liczbowo od \( s \), zatem pasmo dostępne dla ścieżek o priorytecie \( s \), dla \( s < 7 \), jest zawsze większe lub równe pasmu dostępnemu dla ścieżek o priorytecie \( s+1 \), zatem:

\[
a_0^{(e)} \geq a_1^{(e)} \geq \ldots \geq a_7^{(e)} .
\]

Nowa ścieżka o najniższym priorytecie utworzenia \( s = 7 \) nie może spowodować wywłaszczenia żadnej ścieżki, więc pasmo dostępne dla takiej ścieżki jest równe z definicji wolnemu pasmu, tj. \( a_7^{(e)} = a^{(e)} \).

Pasmo \( b_p^{(e)} \) zajęte przez ścieżkę \( p \) na łączu \( e \) jest równe pasmu ścieżki \( p \), jeśli ścieżka istnieje na łączu \( e \), lub zero, jeśli ścieżka \( p \) nie przechodzi przez łącze \( e \). Zakładamy tu, że pasmo ścieżki na całej jej długości jest jednakowe.

\[
\forall e \in R_p \Rightarrow b_p^{(e)} = \begin{cases} b_p & \text{dla } e \in R_p \\ 0 & \text{dla } e \notin R_p \end{cases}
\]

### 3.4.4. Wywłaszczanie

Niech nowa ścieżka \( q \), która może spowodować wywłaszczania, jest opisana przez \( P_q = (R_q, b_q, s_q, h_q) \).

Lista \( \hat{R} = \{\hat{e}_1, \hat{e}_2, \ldots\} \) zawiera te łącza należące do \( R_q \), na których wolne pasmo jest mniejsze niż pasmo \( b_q \), tzn. na których niezbędne jest wykonanie wywłaszczeń. Liczbę z łączy należących do \( \hat{R} \) określa się jako liczbę łączę z wywłaszczaniem.

\[
z = |\hat{R}|
\]

Zbiór \( K \subset P \) potencjalnych kandydatów do wywłaszczania obejmuje te ścieżki ze zbioru \( P \) dla których spełnione są warunki:

- priorytet utrzymania \( h \) jest liczbowo wyższy priorytetowi utworzenia \( s_q \) nowej ścieżki,
– dzielą przynajmniej jedno łącze z nową ścieżką \( q \) oraz
– posiadają zarezerwowane niezerowe pasmo.

Można to formalnie opisać następująco:

\[
K = \{ p : p \in P \land h_p > s_q \land R_p \cap \hat{R} \neq \emptyset \land b_p > 0 \}. \tag{3.8}
\]

Dla danego łącza \( e \) zbiór \( K_e \subseteq K \) zawiera te ścieżki ze zbioru potencjalnych kandydatów, które przechodzą przez łącze \( e \).

\[
K_e = \{ p : p \in K \land e \in \hat{R} \}. \tag{3.9}
\]

Zbiór \( W \subseteq K, W = \{ w_1, w_2, \ldots \} \), kandydatów do wywłaszczenia zawiera, na zakończenie algorytmu wywłaszczenia, ścieżki przeznaczone do wywłaszczenia po to, aby można było przyjąć nową ścieżkę \( q \) i zarezerwować dla niej żadaną ilość pasma \( b_q \).

Liczba \( M \) wywłaszeń to liczba kandydatów, tj. ścieżek, które należy wywłaszczyć, aby przyjąć nową ścieżkę, zgodnie z wynikiem działania danego algorytmu.

\[
M = |W|. \tag{3.10}
\]

Wywłaszczone pasmo \( B \) jest sumą pasm kandydatów, tj. ścieżek przeznaczonych do wywłaszczenia.

\[
B = \sum_{p \in W} b_p. \tag{3.11}
\]

Wywłaszczone pasmo sieciowe \( B_n \) jest sumą iloczynów pasm i długości kandydatów (długość jest określona jako liczba łączy \( l_p \), przez które przechodzi ścieżka-kandydat).

\[
B_n = \sum_{p \in W} b_p l_p. \tag{3.12}
\]

\[
l_p = |\hat{R}_p|. \tag{3.13}
\]

Bilans pasma \( S = [S_1, S_2, \ldots] \) określa, w trakcie lub po zakończeniu działania algorytmu, różnicę pomiędzy pasmem jakie zostanie lub zostało wywłaszczone a minimalnym pasmem wymaganym do przyjęcia nowej ścieżki \( q \):

\[
S_e = \sum_{p \in W} \beta(p,e) + a_f^{(c)} - b_q, \tag{3.14}
\]

gdzie \( e \in \hat{R} \), natomiast \( \beta(p,e) \) jest funkcją zwracającą pasmo ścieżki \( p \) w przypadku, gdy ścieżka \( p \) przechodzi przez łącze \( e \), lub zero w przeciwnym wypadku:
\[
\beta(p,e) = \begin{cases} 
\beta_p & \text{gdy } e \in R_p \\
0 & \text{gdy } e \notin R_p.
\end{cases}
\] (3.15)

Globalny bilans pasma \( S_g \) osiąga wartość nieujemną tylko wtedy, gdy bilans pasma \( S \) jest nieujemny dla każdego łącz \( e \in \hat{R} \), tzn. gdy zbiór kandydatów zapewnia pasmo niezbędne do przyjęcia nowej ścieżki (jest to warunek na sukces procedury wywłaszczania). Jest obliczany poprzez wybór najmniejszej wartości bilansu pasma \( S_e \) na łączach wymagających wywłaszczania na drodze połączeniowej:

\[
S_g = \min_{e \in R} S_e. \] (3.16)

Poszukiwane pasmo \( B_o = [B_o(1), B_o(2), \ldots] \) określa wartość pasma niezbędną dla przyjęcia nowej ścieżki \( q \), na początku działania algorytmu:

\[
B_o^{(e)} = \max\left(0, b_q - a_f^{(e)}\right), \] (3.17)

gddie \( e \in \hat{R} \).

Wymagane pasmo \( B_r = [B_r(1), B_r(2), \ldots] \) określa, w trakcie działania algorytmu, ile pasma na poszczególnych łączach należy jeszcze pozyskać w drodze powiększenia zbioru \( W \) o kolejnych kandydatów:

\[
B_r^{(e)} = \max\left(0, -S_e\right), \] (3.18)

gddie \( e \in \hat{R} \).

W oparciu o tablicę \( B_r \) można zdefiniować wymagane pasmo sieciowe \( B_{net} \) jako sumę wymaganych pasm na wszystkich łączach należących do drogi połączeniowej, wybranej dla ścieżki.

\[
B_{net} = \sum_{e \in R} B_r^{(e)}. \] (3.19)

Nadmiarowe (stracone) pasmo \( B_x = [B_x(1), B_x(2), \ldots] \) oznacza, po zakończeniu działania algorytmu, pasmo, które zostanie lub zostało wywłaszczone ponad wymagane minimum, na wszystkich łączach \( e \) należących do drogi połączeniowej ścieżki \( p \), tzn. również tych, na których nie brakuje wolnego pasma. Nadmiarowe pasmo jest wynikiem niedokładności algorytmu i/lub braku zbioru ścieżek, które dokładnie pokrywałyby się z zapotrzebowaniem na brakujące pasmo na wszystkich łączach:

\[
B_x^{(e)} = \max\left(0, \sum_{p \in W} \beta(p,e) + a_f^{(e)} - b_q\right), \] (3.20)

gddie \( e \in R_p \).
Podobnie jak dla wymaganego pasma, zdefiniować można nadmiarowe (stracone) pasmo sieciowe jako sumę pasm straconych na wszystkich łączach należących do drogi połączeniowej wybranej dla nowej ścieżki.

\[ B_{\text{net}} = \sum_{e \in R_p} B_x^{(e)} \quad (3.21) \]

Ponieważ algorytmy wywłaszczania zwykle nie zajmują się łączami, na których pasmo jest wystarczające, pasmo stracone sieciowe może nie oddawać w pełni jakości algorytmów w zakresie minimalizacji wywłaszczonego pasma. Z tego powodu zdefiniowano dodatkowo nadmiarowe pasmo lokalne \( B_{\text{loc}} \), obejmujące tylko te łąca, na których pasmo jest niewystarczające dla przyjęcia nowej ścieżki.

\[ B_{\text{loc}} = \sum_{e \in K} B_x^{(e)} \quad (3.22) \]

W praktyce przy ocenie algorytmów wywłaszczania przydatne jest zdefiniowanie wielkości względnych: pasma straconego \( b_x^{(e)} \) na łączu \( e \), pasma straconego sieciowego \( b_{\text{net}} \) i pasma straconego lokalnego \( b_{\text{loc}} \). Są to wartości odniesione do wymaganego pasma \( B_o^{(e)} \) lub wymaganego pasma sieciowego \( B_{\text{net}} \).

\[ b_x^{(e)} = \frac{B_x^{(e)}}{B_o^{(e)}} \quad (3.23) \]

\[ b_{\text{net}} = \frac{B_{\text{net}}}{B_{\text{net}}} \quad (3.24) \]

\[ b_{\text{loc}} = \frac{B_{\text{loc}}}{B_{\text{net}}} \quad (3.25) \]

Do celów oceny wpływu algorytmu na warunki ruchowe w sieci zdefiniowano poziom rezerwacji pasma \( b_b^{(e)} \) dla łącza \( e \), jako stosunek ilości pasma zarezerwowanego dla wszystkich ścieżek przechodzących przez łącze \( e \) do pasma tego łącza.

\[ b_b^{(e)} = \frac{a_{\text{loc}}^{(e)}}{c_e} \quad (3.26) \]

Analogicznie zdefiniować można globalny poziom \( B_b \) rezerwacji pasma jako stosunek sumy pasm zarezerwowanych na wszystkich łączach do sumy pasm łączy.
co w przypadku identycznych pasm łączy w sieci różnych \( e \) daje następującą zależność:

\[
B_e = \frac{\sum_{e \in E} a_{oc}^{(e)} }{\sum_{e \in E} c_e },
\]

(3.27)

Do opisu niektórych algorytmów przydatne jest zdefiniowanie tablicy \( B_{hold} \) zawierającej sumę pasm zajętych przez wszystkich potencjalnych kandydatów, dla których priorytet utrzymania \( h_p \) jest równy danemu \( h \):

\[
B_{hold}[h] = \sum_{p \in K} \sigma(h_p = h) b_p ,
\]

(3.29)

gdzie \( h \in \{0, 1, \ldots, 7\} \).

Podobnie, można zdefiniować tablicę pasma \( B_{pr} \) możliwego (dostępnego) do wywłaszczania, zdefiniowaną jako sumę pasm ścieżek (potencjalnych kandydatów), które mogą być wywłaszczone przez ścieżkę o danym priorytecie:

\[
B_{pr}[h] = \sum_{h' = h+1}^{7} B_{hold}[h'] = \sum_{p \in K} \sigma(h_p > h) b_p ,
\]

(3.30)

gdzie \( h \in \{0, 1, \ldots, 7\} \).

Z definicji wywłaszczania wynika, że \( B_{pr}[7] = 0 \), tzn. pasmo dostępne do wywłaszczania dla ścieżki o najniższym priorycie utworzenia równym 7 musi być równe 0, gdyż ścieżka o najniższym możliwym priorycie utworzenia nie może wywłaszczyć żadnej innej ścieżki, w tym także innej ścieżki o priorycie utrzymania równym 7.

3.4.5. Warunki poprawności wywłaszczania

Aby dowolna ścieżka \( q \) mogła wywłaszczyć dowolną inną ścieżkę \( p \) muszą być jednocześnie spełnione następujące trzy warunki.

1. Istnieje przynajmniej jedno łącze \( e \) należące do obu ścieżek, na którym brakuje wolnego pasma dla ścieżki \( q \), co możemy zapisć następująco:

\[
a_f^{(e)} < b_q ,
\]

(3.31)

dla dowolnego \( e \in (R_q \cap R_p) \).
2. Priorytet utworzenia $s_q$ ścieżki $q$ jest wyższy (liczbowo mniejszy) od priorytetu utrzymania $h_p$ ścieżki $p$.

$$s_q < h_p$$ (3.32)

3. Dodatkowym warunkiem jest pozytywne zakończenie algorytmu wywłaszczania. Warunek ten oznacza, że musi istnieć przynajmniej jeden zbiór $W$ kandydatów taki, że wywłaszczenie ścieżek należących do $W$ umożliwi odzyskanie wolnego pasma w ilości wymaganej do przyjęcia ścieżki $q$. Formalnie to ujmując dla każdego łącza $e$, na którym brakuje pasma do przyjęcia ścieżki $q$, musi być spełniona następująca nierówność.

$$a_f^{(e)} + \sum_{w \in W} b_n \beta(w, e) \geq b_q,$$ (3.33)

dla każdego $e \in \hat{R}$.

3.4.6. Kaskada i łańcuchy wywłaszczeń

Ze zjawiskiem kaskady wywłaszczeń mamy do czynienia wtedy, gdy wywłaszczona ścieżka podczas jej ponownego utworzenia na nowej drodze powoduje kolejne wywłaszczenia innych ścieżek. Te z kolei ścieżki mogą spowodować kolejne wywłaszczenia, itd. W efekcie może w sieci dojść do niekontrolowanego, masowego wywłaszczenia wielu ścieżek. Jest to zjawisko wysoce niekorzystne, gdyż powoduje nie tylko utratę dużych ilości ruchu z powodu przeniesienia ścieżek, ale także gwałtowny wzrost ilości ruchu sygnalizacyjnego w sieci.


Należy pamiętać, iż każda ścieżka może potencjalnie wywłaszczyć kilka innych ścieżek. Zatem kaskada wywłaszczeń może mieć postać drzewa, podobnego jak na rys. 3.4, a liczba ścieżek biorących udział w kaskadzie może być bardzo duża. Oczywiście o kaskadzie mówimy tylko wtedy, gdy poziom wywłaszczań osiągnął wartość dwa lub więcej.

Ze zjawiskiem kaskady wywłaszczeń wiąże się bezpośrednio pojęcie łańcucha wywłaszczeń, który odpowiada pojedynczej gałęzi w kaskadzie. Łańcuch wywłaszczeń jest to sekwencja wywłaszczeń spowodowanych pośrednio lub bezpośrednio przez ścieżkę inicjującą. Długość łańcucha jest o jeden mniejsza od liczby ścieżek w łańcuchu, np. jeśli nowo tworzo-
na ścieżka powoduje wywłaszczenie, a wywłaszczona ścieżka już nie powoduje kolejnych wywłaszczeń, wówczas mamy najkrótszy możliwy łańcuch o długości jeden, składający się z dwóch ścieżek. Bazując na pojęciu łańcucha, można zdefiniować długość kaskady wywłaszczeń jako długość najdłuższego łańcucha wywłaszczeń w kaskadzie. Ostatnią z istotnych definicji jest liczba ścieżek w kaskadzie wywłaszczeń, równa liczbie wszystkich ścieżek wywłaszczonych bezpośrednio lub pośrednio przez ścieżkę inicującą.

Dla zilustrowania wprowadzonych pojęć, na rys. 3.4 przedstawiono schematicznie przykład kaskady wywłaszczeń. Ścieżka inicjująca wywłaszczenie o priorytecie utworzenia równym 0 powoduje wywłaszczenie trzech ścieżek o priorytetach 3, 3 i 1. Wywłaszczona ścieżka o priorytecie 1 przy próbie ponownego utworzenia powoduje kolejne wywłaszczenia dwóch ścieżek o priorytetach 3 i 7, z których pierwsza również wywłaszcza trzy inne ścieżki. Na tym przykładzie widać, że ścieżka inicjująca spowodowała wywłaszczenie w sumie dziewięciu ścieżek, mimo, że tylko trzy z nich zostały wywłaszczone bezpośrednio. Liczba ścieżek w kaskadzie jest tu równa dziewięć, natomiast długość kaskady wynosi trzy. Przykładowy łańcuch obejmujący ścieżki o priorytetach 0, 1 i 7 ma długość dwa.

Rys. 3.4. Zjawisko kaskady wywłaszczeń – przykład; strzałki oznaczają kolejne wywłaszczenia, a numery w węzłach drzewa – priorytety ścieżek. Kolorem niebieskim wyróżniono przykładowy łańcuch wywłaszczeń.

Zbiór $D^{(p)}$ ścieżek w kaskadzie wywłaszczeń spowodowanej przez ścieżkę $p$ zawiera wszystkie ścieżki, jakie zostały bezpośrednio lub pośrednio wywłaszczone w efekcie utworzenia ścieżki $p$.

$$D^{(p)} = X^{(p)} \cup \sum_{l \in X^{(p)}} D^{(l)}$$ (3.34)

Liczba $H^{(p)}$ ścieżek w kaskadzie wywłaszczeń zapoczątkowanym przez ścieżkę $p$ jest określona jako liczność zbioru $D^{(p)}$.

$$H^{(p)} = |D^{(p)}|$$ (3.35)

Zdefiniujemy funkcję poprzednika $\text{prev}(p)$ ścieżki $p$, zwracającą bądź jednoelementowy zbiór zawierający ścieżkę $q$, która spowodowała wywłaszczenie ścieżki $p$, bądź zbiór pusty, gdy ścieżka $p$ nie została wywłaszczena.

$$\text{prev}(p) = \begin{cases} \{q\} & \text{gdy } \exists q \in X^{(q)} \\ \emptyset & \text{w przeciwnym wypadku} \end{cases}$$ (3.36)

Można teraz zdefiniować poziom $m^{(p)}$ wywłaszczenia ścieżki $p$ jako (dotychczasową) długość łańcucha wywłaszczeń, określoną tuż po wywłaszczeniu ścieżki $p$.

$$m^{(p)} = \begin{cases} 0 & \text{dla } \text{prev}(p) = \emptyset \\ m^{(\text{prev}(p))} + 1 & \text{dla } \text{prev}(p) \neq \emptyset \end{cases}$$ (3.37)

Długość $g^{(p)}$ kaskady wywłaszczeń spowodowanej przez ścieżkę $p$ definiujemy jako największy poziom wywłaszczenia wśród ścieżek biorących udział w kaskadzie wywłaszczeń.

$$g^{(p)} = \max_{l \in D^{(p)}} (m^{(l)})$$ (3.38)

### 3.5. Dostępne algorytmy

Wywłaszczanie jako metoda udostępniania pasma w sytuacji braku wolnych zasobów jest znane w sieciach telekomunikacyjnych od dawna. Dopóki jednak nie zaczęto implementować sieci pakietowych z kanałami o różnej przepustowości, dopóty nie było potrzeby prowadzenia bardziej zaawansowanych badań w tym kierunku. Dopiero na początku lat dziewięćdziesiątych pojawiły się pierwsze opracowania szerzej omawiające ten temat.
W tej części pracy omówione zostaną znane algorytmy heurystyczne, które kosztem nie-
optymalnego wyboru oferują akceptowalną złożoność obliczeniową. Dla uzupełnienia zostaną
przedstawione i omówione algorytmy optymalne, które wprawdzie z powodu nadmiernej zło-
żoności obliczeniowej nie mogą być praktycznie zastosowane, ale pozwalają uzyskać pełniej-
szy obraz możliwych rozwiązań.

Omówione algorytmy heurystyczne zaczepnięto z najczęściej cytowanych publikacji do-
tyczących wywłaszczania. Najpierw krótko je scharakteryzowano, aby następnie opisać je
dokładniej. Przyjęto konwencję nazewniczą w postaci skrótów nazwisk autorów.

1. **GarGop** (Garay-Gopal, 1992) [35]. Jest to pierwsze ogólnie dostępne opracowanie
dające wywłaszczania połączeń w sieciach pakietowych. Zawarto w nim dwa nie-
zależne algorytmy globalne przeznaczone do minimalizacji liczby wywłaszczonych
ścieżek i do minimalizacji wywłaszczonego pasma, bez uwzględnienia priorytetów
ścieżek. Algorytmy zostały opracowane dla sieci ATM, ale mogą być zaadoptowane
do wykorzystania w sieciach MPLS.

2. **Pey** (Peyravian, 1994) [82]. W tym artykule przedstawiono pojedynczy, stosunkowo
prosty algorytm lokalny, przeznaczony do minimalizacji zarówno liczby wywłasz-
czeń jak i wywłaszczonego pasma.

3. **OliSco** (de Oliveira-Scoglio, 2002) [72]. W artykule zawarto jeden z pierwszych al-
gorytmów przeznaczonych z założenia dla sieci MPLS. Do tej publikacji odnoszą się
autorzy większości późniejszych prac dotyczących wywłaszczania. Jest to prosta,
uniwersalna i szybka metoda lokalna oparta na sortowaniu ścieżek-kandydatów. Al-
gorytm ten został później upowszechniony jako dokument informacyjny RFC 4829
[70].

4. **BlaMeL** (Blanchy-Mélon-Leduc, 2003) [20]. W artykule opisano prosty algorytm lo-
alny, w którym nadrzędnym kryterium wyboru jest priorytet kandydatów. W dalszej
kolejności minimalizuje się liczbę ścieżek a w ostatniej kolejności stracone pasmo.

W kilku kolejnych podrozdziałach dokładnie zaprezentowano i poddano analizie wy-
mienione algorytmy. Kod programów je realizujących, zaczerpnięty z oryginalnych artyku-
lów, został ujednolicony pod względem użytych oznaczeń i zapisany w języku pseudo-pascal,
w którym niektóre instrukcje i procedury zostały dla zachowania przejrzystości zapisane
w języku naturalnym. Dodatkowym uproszczeniem w stosunku do języka programowania
Pascal jest zastosowanie notacji matematycznej dla zapisania operacji na zbiorach i sekwen-
cjach. Dla poprawy jakości opisów zachowano oznaczenia wprowadzone w rozdziale 3.4
a linie kodu zostały ponumerowane.

### 3.5.1. Algorytm GarGop (Garay-Gopal)

Pierwsze szeroko dostępne analityczne ujęcie zagadnienia wywłaszczania zawarto
w [35]. Artykuł ten został wprawdzie napisany pod kątem zastosowania w sieciach ATM, ale
dzięki podobieństwu obu technik wiele treści pozostaje aktualnych również dla sieci IP/MPLS.


Algorytm minimalizacji liczby ścieżek został przedstawiony na rys. 3.5. Główna procedura \textit{algorithm\_GarGop} działa następująco. Po inicjalizacji (linie 2-4) uruchamia się główną pętlę (linie 5-11), która jest powtarzana tak długo, aż brakujące pasmo $B_{net}$ osiągnie wartość zero. W linii 6 następuje sprawdzenie, czy w zbiorze $K$ potencjalnych kandydatów znajduje się przynajmniej jedna ścieżka. Jeśli nie, wówczas algorytm jest przerywany z powodu braku wystarczającej liczby potencjalnych kandydatów.

```plaintext
1  procedure algorithm_GarGop ()
2      W := ∅  # wyzerowanie zbioru kandydatów
3      K := cand_preselect_on_route ()  # wyznaczenie zbioru # potencjalnych kandydatów
4      calculate_bw_balance ()  # wyliczenie początkowego # bilansu pasma
5      repeat
6         if K = ∅ then return ∅  # zakończ, jeśli nie ma więcej # potencjalnych kandydatów
7             p := best_path (K)  # wybierz kandydata z listy,
8             W := W ∪ {p}  # dodaj do listy
9             K := K \ {p}  # i usuń z listy wyrób
10            update_bw_balance (p)  # sprawdź ile pasma jeszcze brakuje
11            until B_{net} = 0  # zakończ gdy brakujące pasmo jest zerowe
12            return W  # zwróć listę kandydatów
13       end
14
15  procedure cand_preselect_on_route ()
16      K := ∅
17      for each p in P do  # dla każdej ścieżki
18         if h_p > s_q  # która może zostać wywłaszczona
19             and path_on_route (R_p, p) then
```

45
procedure calculate_bw_balance ()

B_{net} := 0  # zainicjowanie brakującego pasma sieciowego

for each e in \( \hat{\mathcal{R}} \) do  # dla każdego łącza na wybranej drodze
    S_e := a_j^{(e)} - b_q  # oblicz bilans pasma na łączu
    B_r^{(e)} := \max (0, -S_e)  # uaktualnij brakujące pasmo na łączu e
    B_{net} := \min (B_{net}, B_r^{(e)})  # uaktualnij brakujące pasmo sieciowe
end for

end

function best_path (K)

max_sum := 0  # wyzeruj wskaźnik najlepszej ścieżki

for each p in K  # dla każdego potencjalnego kandydata
    sum := 0  # wyzeruj sumę dla ścieżki
    for each e in \( \hat{\mathcal{R}} \)
        sum := sum + \min [B_r^{(e)}, \beta(p,e)b_p]  # na łączach gdzie występuje ścieżka zsumuj pasmo ścieżki
        # ograniczone do wymaganego pasma
    end for
    if sum > max_sum then  # jeśli aktualna suma jest największa
        max_sum := sum  # uaktualnij wskaźnik najlepszej ścieżki
        best_p := p  # zapamiętaj najlepszą ścieżkę
    end if
end for
return best_p  # zwróć zapamięтанą najlepszą ścieżkę

end

procedure update_bw_balance (p)

for each e in \( \hat{\mathcal{R}} \)
    S_e := S_e + \beta(p,e)b_p  # uaktualnij bilans pasma i brakujące
    B_r^{(e)} := \max (0, -S_e)  # pasmo o pasmo wybranej ścieżki
end for

end

Rys. 3.5. Algorytm GarGop/RC dla minimalizacji liczby wywłaszczonych ścieżek.
Najważniejszymi krokami algorytmu są przypisanie zmiennej $p$ najlepszej ścieżki (linia 7), dodanie jej do listy kandydatów (linia 8) i usunięcie z listy potencjalnych kandydatów (linia 9). Po tym następuje uaktualnienie bilansu pasma (linia 10), tzn. pasmo wybranej ścieżki zostaje dodane do bilansu pasma $S_e$ na każdym łączu $e$, przez które przechodzi ścieżka, i na którym brak jest wolnego pasma. Gdy wartość brakującego pasma spadnie do zera (linia 11) następuje wyjście z pętli i na zakończenie zbiór $W$ kandydatów zostaje zwrócony jako wynik działania algorytmu.

O wyborze ścieżek do wywłaszczania decyduje kluczowa dla algorytmu funkcja $best\_path$. W omawianym algorytmie za najlepszego kandydata do wywłaszczania uznaje się ścieżkę, dla której wartość metryki $\alpha(p)$ (wzór 3.39) jest największa. Aby ją obliczyć, na każdym łączu wymagającym wywłaszczania, na którym ścieżka $p$ jest obecna, wybiera się mniejszą z dwóch wartości: brakujące pasmo $B_r^{(e)}$ i pasmo ścieżki $b_p$, i sumuje po tych łączach.

$$\alpha(p) = \sum_{e \in K} \min(B_r^{(e)}, b_p) \quad (3.39)$$

Procedura $best\_path$ rozpoczyna się od wyzerowania zmiennej $max\_sum$ (linia 33), po czym dla każdej ścieżki z listy potencjalnych kandydatów $K$ oblicza się (linie 35-38) metykę $\alpha(p)$ przechowywana w zmiennej $sum$. Obliczona wartość jest następnie porównywana z dotychczasową największą wartością $max\_sum$. Jeśli nowa metryka jest większa, wówczas następuje uaktualnienie $max\_sum$ oraz przypisanie aktualnej ścieżki $p$ do zmiennej $best\_p$ (linie 40-41). Po przejściu wszystkich potencjalnych kandydatów zwrócona zostaje ścieżka o najwyższej wartości metryki $\alpha(p)$ (linia 44).

Druga postać algorytmu GarGop, utworzona dla przypadku minimalizacji ilości wywłaszczonego pasma, różni się od poprzedniej jedynie inną definicją procedury $best\_path$. Tym razem minimalizacji podlega metryka $\chi(p)$ (wzór 3.40). Odpowiednio zmodyfikowana procedura została pokazana na rys. 3.6.

$$\chi(p) = \frac{\sum_{e \in K} \max(b_p - B_r^{(e)}, 0)}{\sum_{e \in K} \min(B_r^{(e)}, b_p)} \quad (3.40)$$

Z racji przeznaczenia dla sieci ATM, oryginalna metoda nie uwzględnia priorytetów ścieżek. Nie jest to problemem, gdyż z racji użycia zbioru $K$ ograniczono wybór kandydatów do tych ścieżek, które mają priorytet utrzymania niższy niż priorytet utworzenia nowej ścieżki $q$. Tak zostało to z resztą określone w ogólnej definicji zbioru $K$ (wzór 3.8).

Algorytm GarGop jest algorytmem zachłannym, który raz wybranej ścieżki nie usuwa już z listy kandydatów. Jest to cecha, która umożliwia uruchamianie procedury usuwania ścieżek już w trakcie działania algorytmu, jeśli tylko się zagwarantuje, że wybór zakończy się sukcesem. Może to mieć znaczenie w praktycznych realizacjach, jeśli czas potrzebny do zakończe-
nia działania algorytmu jest nie do pominięcia, np. z powodu długiego czasu dostępu do danych o ścieżkach.

```plaintext
function best_path (K)
    min_div := \infty  # inicjalizuj wskaźnik najlepszej ścieżki
    for each p in K  # dla każdego potencjalnego kandydata
        sum1 := sum2 := 0  # wyzeruj wskaźniki sum
        for each e in \hat{R}  # na każdym łączu ścieżki
            # zsumuj nadwyżkę pasma (sum1)
            # oraz pasmo wykorzystane (sum2)
            sum1 := sum1 + max (b_p - B^e_r, 0)
            sum2 := sum2 + min (B^e_r, b_p)
        end for
        div := sum1 / sum2  # oblicz metrykę ścieżki
        if div < min_div then  # jeśli aktualna metryka jest najmniejsza
            min_div := div  # uaktualnij wskaźnik najlepszej ścieżki
            best_p := p  # zapamiętaj aktualną ścieżkę
        end if
    end for
    return best_p  # zwróć ścieżkę o najmniejszej metryce div
end
```

Rys. 3.6. Algorytm GarGop/BW dla minimalizacji ilości wywłaszczonego pasma.

Dokładniejsza analiza wariantu GarGop/RC pozwala zauważyć, że algorytm nie bierze pod uwagę pasma ścieżek w sytuacji, gdy to pasmo jest większe niż pasmo wymagane. Nie są przez to rozróżniane ścieżki o dużym i małym pasmie, co ma negatywny wpływ na wywłaszczone pasmo. Może się zdarzyć, że potrzebne jest zwolnienie małej ilości pasma, a algorytm wybierze wtedy dowolną, być może największą ścieżkę. Takie działanie jest uzasadnione tym, że wariant ten jest nastawiony na minimalizację liczby wywłaszczeń. Algorytm wykonuje więc to, do czego został przeznaczony ale nic więcej. Nie jest tu możliwe nałożenie dodatkowych wymagań.

Odnosnie wariantu GarGop/BW daje się zauważyć, że również tutaj zdolność do wyróżniania ścieżek jest ograniczona. W przypadku, gdy na każdym z łącz pasmo ścieżki jest mniejsze niż pasmo wymagane, metryka \(\chi(p)\) ma wartość zero, która jest wartością minimalną. Użyte kryterium wyboru powoduje, że wybierana jest jedna z takich ścieżek bez względu na pasmo. To ma z kolei przełożenie na pewną nieokreśloność tej metody i zależność wyniku od kolejności analizy ścieżek.

Niewątpliwą zaletą metody GarGop w obu wariantach jest prostota i niewielka złożoność
obliczeniowa. Przekłada się to na stosunkowo niewielkie wymagania na moc procesora, a to ma znaczenie w przypadku sieci o dużej intensywności żądań utworzenia ścieżek.

3.5.2. Algorytm Pey (Peyravian)


Odpowiednia procedura została przedstawiona na rys. 3.7. Jest to algorytm lokalny, dla tego zadaniem głównej pętli for (linie 3-17) jest powtórzenie procedury wyboru zbioru kandydatów na każdym łączu, na którym brakuje wolnego pasma. W pierwszym kroku w pętli następuje obliczenie bilansu pasma na bieżącym łączu (linie 4-5), wybór potencjalnych kandydatów (linia 6), a następnie uruchomienie pętli while odpowiedzialnej za wybór kompletu kandydatów na danym łączu. Pętla jest powtarzana tak długo, dopóki brakuje pasma.

```plaintext
1 procedure algorithm_Pey ()
2     W := ∅  # wyzerowanie zbioru kandydatów
3     for each e in R  # dla każdego łącza na drodze ścieżki
4         S_e := a_(f(e)) - b_q  # oblicz bilans pasma
5         B_r^{(e)} := max (0, -S_e)  # oblicz wymagane pasmo
6     Ke := cand_preselect_on_link (e)  # wyznaczenie lokalnego zbioru
7         # potencjalnych kandydatów
8     while B_r^{(e)} ≠ 0 do  # powtarzaj, dopóki brakuje pasma
9         if Ke = ∅ then return ∅  # wyjdź, jeśli brakuje kandydatów
10        p := best_path (Ke)  # wybierz najlepszą ścieżkę
11        W := W ∪ {p}  # dodaj ścieżkę do listy kandydatów
12        Ke := Ke \ {p}  # usuń z listy potencjalnych kandydatów
13        S_e := S_e + b_p  # uaktualnij bilans pasma
14        B_r^{(e)} := max (0, -S_e)  # oraz wymagane pasmo
15     end while
16 end for
17 return W
e
18
19 function best_path (Ke)
20     best_bw := ∞  # zainicjuj metrykę najlepszej ścieżki
```
Rys. 3.7. Algorytm Pey.

Po wejściu do pętli while następuje sprawdzenie, czy w zbiorze potencjalnych kandydatów \( K_e \) znajdują się ścieżki (linia 8), po czym wywołana zostaje funkcja \( best\_path \), aby wybrać najlepszą ścieżkę z listy \( K_e \) (linia 9). Po tym następuje aktualizacja listy ścieżek do wywłaszczenia, listy potencjalnych kandydatów oraz bilansu pasma (linie 10-13). Struktura pętli while jest podobna jak w przypadku algorytmu globalnego, jednak w tym przypadku lista potencjalnych kandydatów ogranicza się do ścieżek dostępnych na aktualnym łączu, podczas gdy w algorytmach globalnych bierze się pod uwagę ścieżki na wszystkich łączach.
należących do \( \hat{R} \). Po zagwarantowaniu wystarczającego pasma następuje opuszczenie pętli \textit{while} i w razie potrzeby powtórzenie wyboru dla kolejnego łącza w pętli \textit{for}. Na zakończenie następuje zwrócenie zbioru \( W \) zawierającego ścieżki wybrane na wszystkich łączach (linia 16).


Omówiona postać procedury \textit{best_path} jest zgodna z oryginalnym opracowaniem [82]. Jak łatwo zauważyć, można poprawić szybkość działania tego algorytmu poprzez użycie pojedynczej pętli z bardziej złożonym warunkiem. To może mieć znaczenie wtedy, gdy w danej implementacji przeglądanie ścieżek zajmuje stosunkowo dużo czasu. Odpowiednio zmodyfikowana procedura \textit{best_path_mod} została przedstawiona na rys. 3.8.

```
1 function best_path_mod (K_e)
2   best_bw := 0
3   best_p := ∅
4   for each p in K_e
5     if (b_p >= B_r(e) and (best_bw < B_r(e) or b_p < best_bw) )
6       or (b_p < B_r(e) and b_p > best_bw) then
7         best_bw := b_p
8         best_p := p
9     end if
10   end for
11 return best_p
12 end
```

Rys. 3.8. Modyfikacja algorytmu Pey z pojedynczą pętlą wyboru najlepszego kandydata.

Należy zwrócić uwagę, iż ogólna procedura realizująca algorytm Pey z rys. 3.7 w liniach 1-17 została przedstawiona w taki sposób, aby dać lepsze porównanie z algorytmem globalnym GarGop. W praktyce jednak najbardziej prawdopodobne jest zaimplementowanie algorytmu lokalnego jako metody zdecentralizowanej. Wówczas pętla \textit{for} odpowiadająca za obsługę kolejnych łącz (linie 3-15) nie będzie istniała w sposób jawny. Jej funkcję spełni uruchomienie lokalnych instancji algorytmu w kolejnych ruterach na drodze połączeniowej, przy czym musi być ono wykonane jedno po drugim. Jest to ważne, aby algorytm w danym ruterze bazował na wywłaszczeniach przeprowadzonych na poprzednich łączach. Może się
bowiem zdarzyć, że jedna ze ścieżek wywłaszczenych na wcześniejszym łączu zwolniła przy okazji wystarczającą ilość pasma na aktualnym łączu, a wtedy rozpoczynanie kolejnego wywłaszczania jest już niepotrzebne.

Zaletą algorytmu Pey jest jego prostota. Jego działanie polega na minimalizacji liczby wywłaszczenych ścieżek przy jednoczesnym ograniczaniu wywłaszczonego pasma. Określa-jąc kolejnych kandydatów do wywłaszczenia daży on przy tym do wyboru ścieżek o paśmie najbardziej zbliżonym do wymaganego, kosztem jak najmniejszej liczby ścieżek. W ten pro- sty sposób algorytm osiąga bardzo dobrą wydajność pod warunkiem, że wymagane kryterium to liczba wywłaszczeń. Dokładniejsza analiza prowadzi do wniosku, że algorytm Pey jest w istocie optymalny w ramach pojedynczego łącza w zakresie minimalizacji liczby wywłaszczeń.

Algorytm Pey nie posiada wariantów, zatem nie wymaga też konfiguracji. To może być jego zaletą, ale też jest oczywistym ograniczeniem, bo nie umożliwia dostosowanie go do indywidualnych wymagań.

3.5.3. Algorytm OliSco (de Oliveira, Scoglio)

Jedną z pierwszych publikacji dotyczących wywłaszczenia dla sieci MPLS jest [72]. W artykule tym opisano prosty algorytm o zasięgu lokalnym oparty na sortowaniu ścieżek. Metoda polega na obliczaniu dla każdej ze ścieżek odpowiednio zdefiniowanej metryki liczbowej. Następnie tworzona jest lista dostępnych ścieżek posortowana zgodnie z rosnącymi wartościami ich metryk. Wybór kandydatów do wywłaszczenia polega na pobieraniu z czoła listy ścieżek tak długo, aż zebrane zostanie brakujące pasmo. Ponieważ jest to algorytm lokalny, zatem w razie potrzeby procedura jest powtarzana na kolejnym łączu na drodze połączeniowej w tych ruterach, gdzie wciąż brakuje pasma dla nowej ścieżki.

Algorytm jest oparty na uniwersalnej metryce $m(p, e)$, rozumianej jako koszt wywłaszczzenia danej ścieżki $p$ na łącze $e$. Jest ona zdefiniowana w oparciu o cztery parametry $X_1$, $X_2$, $X_3$ i $X_4$ o wartościach zależnych od zadanego kryterium wyboru kandydatów:

$$m(p, e) = X_1 \cdot (8 - h_p) + X_2 \cdot \frac{1}{b_p} + X_3 \cdot \left(b_p - B_o^{(e)}\right)^2 + X_4 \cdot b_p,$$

(3.41)

gdzie $X_1$, $X_2$, $X_3$ i $X_4$ są nieujemnymi współczynnikami służącymi do wyboru kryterium. Jeśli któryś ze współczynników ma wartość zero, wówczas dany parametr nie jest w ogóle brany pod uwagę przy wyborze kandydatów do wywłaszczenia. Rola poszczególnych współczynników jest następująca:

- $X_1$ odpowiada za minimalizację priorytetu ścieżki,
- $X_2$ odpowiada za minimalizację liczby wywłaszczeń,
- $X_3$ odpowiada za minimalizację wywłaszczonego pasma,
- $X_4$ odpowiada za minimalizację wywłaszczeń dużych ścieżek.
Na rys. 3.9 przedstawiono implementację algorytmu w pseudokodzie. Po wyzerowaniu zbioru kandydatów (linia 2) następuje zainicjowanie listy L, przechowującej wpisy (ścieżka p, metryka m, pasmo b). Lista jest sortowana według rosnącej wartości metryki m, a w razie istnienia dwóch takich samych wartości metryk według rosnącego pasma. Dalej w fazie inicjalizacji następuje obliczenie początkowego bilansu pasma i wymaganego pasma (linie 4-5), oraz wyznaczenie zbioru potencjalnych kandydatów na danym łuczu. Następnie dla każdej ścieżki z listy \( K_e \) następuje obliczenie metryk (linia 8) i dodanie ścieżki do listy L (linia 9).

```plaintext
1   procedure algorithm_OliSco ()
    # algorytm lokalny, dla uproszczenia podano procedurę dla
    # pojedynczego węzła, powtarzaną w razie potrzeby w następnych węzłach
2      W := ∅    # inicjuj listę kandydatów
3      L := ∅    # inicjuj listę potencjalnych kandydatów
4      # sortowaną po metryce m,
4      # w razie identycznych m po b_p
4      S_e := a_f(e) – b_q    # oblicz początkowy bilans pasma
5      B_r(e) := max (0, –S_e)    # oraz brakujące pasmo
6      K_e := cand_preselect_on_link (e)    # wyznaczenie lokalnego zbioru
6      # potencjalnych kandydatów
7   for each p in K_e    # dla każdego potencjalnego kandydata
8      m := X_1*(8 – h_p) + X_2*(1/b_p) + X_3*sqr (b_p – B_o(e)) + X_4*b_p    # wylicz metrykę ścieżki
9      L ← (p, m, b_p)    # i dodaj do listy sortowanej
10  end for
11  B_orig := B_r(e)    # zapamiętaj brakujące pasmo
12  while B_r(e) ≠ 0 do    # dopóki brakuje pasma
13    if L = ∅ then return ∅    # wyjdź jeśli brak potencjalnych kandydatów
14    m := L[0].m    # pobierz metrykę ścieżki z czoła listy
15    if |L|=1 or m < L[1].m then    # jeśli pojedyncza najmniejsza metryka
16      p := L[0].p    # to zapamiętaj ścieżkę z czoła listy
17    else begin    # w przypadku równych metryk
18      L' := ∅    # utwórz zbiór ścieżek o jednakowej metryce
19      for each p in L such that L[].m = m do    # zachowaj ścieżki o jednakowej metryce
20        L' := L' ∪ {p}    # i sortuj według pasma rosnąco
21      for each p in L' do    # dla każdej ścieżki o jednakowej metryce
```
Główna pętla while (linie 12-40) jest wykonywana tak długo, aż pasmo zajęte przez kandydatów będzie wystarczające do przyjęcia nowej ścieżki. W ramach pojedynczego przebiegu pętli najpierw następuje sprawdzenie (linia 13), czy w zbiorze $L$ znajdują się jeszcze ścieżki do wyboru, co jest warunkiem znalezienia wystarczającego pasma. Następnie zostaje pobrana pierwsza ścieżka z listy $L$ i dodana do listy $W$ kandydatów, o ile jej metryka jest unikalna (linie 15-16). Jeśli natomiast na czele listy znajdują się przynajmniej dwie ścieżki o identycznych metrykach, wówczas następuje dodanie wszystkich ścieżek o tej samej metryce do oddzielnej listy $L'$ (linie 19-20) i analiza całego zbioru $L'$. Jeśli znajdzie się ścieżka o pasmie większym od pasma wymaganego na samym początku (zmienna $B_{\text{orig}}$), wówczas tylko ta jedna ścieżka jest zwracana jako zbiór kandydatów (linie 21-23). W dalszej kolejności (przy jednakowych metrykach) poszukiwana jest w zbiorze $L'$ ścieżka o pasmie większym do aktualnego pasma wymaganego. Jeśli taka zostanie znaleziona (linie 24-26), wówczas jest
dodawana do listy kandydatów, a jeśli nie, to ścieżki z listy $L'$ dodawane są do kandydatów w kolejności od największego pasma (linie 27-32), za każdym razem sprawdzając, czy nie zebrano już wymaganego pasma (warunek w linii 31). Po każdym dodaniu ścieżki do zbioru $W$ kandydatów, następuje usunięcie go z listy $L$ i uaktualniane wartości bilansu pasma. Po określении kompletnej listy $W$ kandydatów algorytm zostaje zakończony.

Rozwinięciem tego algorytmu jest metoda adaptacji pasma [72], w której zanim rozpocznę się procedura wywłaszczania, następuje sprawdzenie, czy jest możliwość zredukowania pasma istniejących ścieżek, korzystając np. z możliwości elastycznego przydziału pasma dla klas AF mechanizmu DiffServ [19]. W takim przypadku zmniejszenie zajętego pasma odbywa się proporcjonalnie dla wszystkich ścieżek. Jest to podejście bardziej „łagodne” od wywłaszczania, a jego zaletą – zmniejszenie liczby wywłaszczeń w sieci.

W oparciu o algorytm OliSco opracowano również metodę globalną wywłaszczania o nazwie PREPATH [71]. Celem algorytmu jest odszukanie ścieżek, których droga najbardziej pokrywa się z drogą nowej ścieżki. Algorytm przyjmuje stałe kryterium wyboru, którym jest w pierwszej kolejności priorytet ścieżki, następnie liczba wywłaszczeń, a w ostatniej kolejności wywłaszczone pasmo. Przedstawiona implementacja jest jednak kłopotliwa w realizacji i cechuje ją duża złożoność obliczeniowa. Należy dodać, że algorytm PREPATH przestał być rozwijany przez autorów [74].

Algorytm OliSco jest bowiem najbardziej znanym algorytmem z racji wielu odniesień w publikacjach naukowych. Jego najważniejszą zaletą jest możliwość dostosowania do indywidualnych wymagań obejmujących pasmo ścieżek, liczbę wywłaszczeń oraz priorytety wybranych ścieżek. Algorytm ten został również opublikowany w formie dokumentu IETF jako RFC 4829 [70]. Należy jednak zaznaczyć, że posiadając status „informational” nie jest i nie pretendsuje do roli żadnego standardu IETF, a ma znaczenie jedynie informacyjne [23].

Złożoność obliczeniowa algorytmu OliSco nie jest duża, choć jego implementacja może być nieco trudniejsza od GarGop i Pey. Wynika to z dodatkowych procedur wykonywanych w przypadku napotkania ścieżek o tych samych metrykach, włączając w to konieczność budowania i analizy dodatkowej listy $L'$. Oczywiście możliwe jest zaimplementowanie algorytmu w prostszej postaci, z pominięciem dodatkowych sprawdzeń dla przypadku jednakowych metryk, co jednak spowoduje pogorszenie osiąganych rezultatów.

3.5.4. Algorytm BlaMeL (Blanchy, Mélön, Leduc)

Implementację algorytmu BlaMeL pokazano na rys. 3.10. Po zainicjowaniu zbioru kandydatów \( W \) i obliczeniu początkowego bilansu pasma (linie 3-4) następuje zainicjowanie (linie 5-8) i obliczenie (linie 10-13) tablicy \( B_{hold} \) zawierającej sumy pasm kandydatów z podziałem na ich priorytety utrzymania oraz związane z nią listy \( K_p \) ścieżek o określonym priorytecie. Zatem zamiast pojedynczej listy kandydatów \( K_e \), wyznaczonej w linii 9, operuje się tu zestawem ośmiu list potencjalnych kandydatów. W linii 14 wprowadzono zmienną \( r \) określającą aktualnie analizowany priorytet. Ścieżki wybierane są w kolejności malejącego priorytetu, zatem jego początkowa wartość wynosi 7.

```
1 procedure algorithm_BlaMeL ()
2 # algorytm lokalny, dla uproszczenia podano procedurę dla
3 # pojedynczego węzła, powtarzaną w razie potrzeby w następnych węzłach
4  \( W := \emptyset \) \# inicjuj listę kandydatów
5  \( S_e := a_l(e) - b_q \) \# oblicz początkowy bilans pasma
6  \( B_{r}^{(e)} := \max (0, -S_e) \) \# oraz brakujące pasmo
7  for \( h \) in [0..7] do \# dla każdego priorytetu utrzymania
8     \( B_{hold}[h] := 0 \) \# wyzeruj pasmo dostępne dla wywłaszczenia
9     \( K_p[h] := \emptyset \) \# inicjuj zbiór potencjalnych kandydatów
10    end \# pogrupowany według priorytetów
11  \( K_e := \text{cand_preselect_on_link}(e) \) \# wyznaczenie lokalnego zbioru
12     potencjalnych kandydatów
13  for each \( p \) in \( K_e \) do \# dla każdego potencjalnego kandydata
14     \( B_{hold}[h_p] := B_{hold}[h_p] + b_p \) \# uaktualnij pasmo ścieżek
15     \( K_p[h_p] := K_p[h_p] \cup \{ p \} \) \# dodaj do listy ścieżek,
16     \# posortowanej według pasma malejąco
17  end
18  \( r := 7 \) \# rozpoczęj od najniższego priorytetu
19  while \( B_{r}^{(e)} \neq 0 \) do \# dopóki brakuje pasma
20     if \( r \leq s_q \) then \# jeśli za mało dostępnego pasma
21        return \( \emptyset \) \# wtedy wywłaszczenie jest nieudane
22     if \( B_{r}^{(e)} \geq B_{hold}[r] \) then \# jeśli suma pasm dla danego priorytetu \( r \)
23        \( W := W \cup K_p[r] \) \# dodaj wszystkie ścieżki o priorytecie \( r \)
24        \( S_e := S_e + B_{hold}[r] \) \# uaktualnij bilans pasma
25     \# nie przekracza brakującego pasma
26     end
27     \( r := r - 1 \) \# przejdź do niższego priorytetu
28  end
```
zbior kandydatów.

Algorytm BlaMeL jest prostym algorytmem o zasięgu lokalnym ukierunkowanym na minimalizację priorytetów wywłaszczonych ścieżek a w dalszej kolejności liczbę wywłaszczeń i wywłaszczonego pasma. Należy zauważyć, że taki wybór podstawowego kryterium umożliwia łatwe zbudowanie algorytmu optymalnego w ramach pojedynczego łącza. Prostota konceptu algorytmu została osiągnięta kosztem pewnej kompleksji wykorzystywanych struktur danych, gdyż w ogólności wymaga utworzenia ośmiu niezależnych list potencjalnych kandydatów.

Słabością metody BlaMeL jest zupełny brak możliwości sterowania kryterium wyboru kandydatów. W konsekwencji, jeśli użytkownik nie stawia priorytetu ścieżek na pierwszym miejscu, ten algorytm nie będzie spełniał jego oczekiwań.

3.5.5. Algorytm dokładny (optymalny)

Dokładny lub optymalny algorytm wywłaszczania to taki, który mając daną drogę przeznaczoną dla nowej ścieżki, w oparciu o ilość wolnego pasma i listę ścieżek utworzonych na poszczególnych łączach, wybiera najlepszy możliwy zbiór ścieżek przeznaczonych do wywłaszczenia. Rozstrzygnięcie tego, który zbiór jest najlepszy, opiera się na zadanej funkcji kosztu wywłaszczania, którym może być np. liczba wywłaszczonych ścieżek lub ilość wywłaszczonego pasma. Algorytm optymalny zawsze zwraca zbiór kandydatów, dla którego zadany koszt jest najmniejszy. Z tego powodu zwykle nie istnieje jeden idealny zbiór kandydatów do wywłaszczenia, ale jest tyle zbiórów ile możliwych kryteriów wyboru. Tym samym możliwe jest zdefiniowanie wielu różnych algorytmów optymalnych, zależnie od wybranego kryterium. Niezależnie od tego różny może być zasięg algorytmu, tzn. wybór optymalny może dotyczyć całej domeny lub tylko pojedynczego łącza. W pierwszym przypadku będzie to zwykle metoda realizowana w sposób scentralizowany, a w drugim jako metoda zdecentralizowana.

Prostszym w analizie przypadkiem są optymalne algorytmy lokalne, które są wykonywane krok po kroku na kolejnych łączach wymagających wywłaszczenia, a decyzja podejmowana jest wyłącznie w oparciu o stan na pojedynczym łączu. Dwa przykłady takich algorytmów dla sieci ATM przedstawiono w [83]. Pierwszy przyjmuje za cel minimalizację liczby wywłaszczených ścieżek, drugi natomiast ma na celu minimalizację ilości wywłaszczonego pasma. Oba algorytmy zostaną tu omówione w oparciu o ich wersje zaadoptowane przez autora do zastosowania w sieciach MPLS.

Algorytm optymalny dla minimalizacji liczby wywłaszczenych ścieżek został przedstawiony na rys. 3.11. Po zainicjowaniu zbioru \( W \) kandydatów oraz obliczeniu bilansu pasma (linia 2-4) wyznaczona zostaje lista \( K_e \) potencjalnych kandydatów na danym łączu (linia 5). Następnie uruchamiana jest pętla while (linia 6-22) wykonywana tak długo, aż brakujące pasmo \( B_r(e) \) zmiaże do zera. Przebieg pętli odbywa się w dwóch etapach. Najpierw wyszukuje się pojedynczą ścieżkę o najmniejszym paśmie nie mniejszym od aktualnie wymaganego (li-

Złożoność obliczeniowa tego algorytmu to $O(|K_e|^2)$ na każde łącze wymagające wywłaszczania, gdzie $|K_e|$ to liczba potencjalnych kandydatów. Jest to unikalny przykład algorytmu optymalnego, który ma akceptowalną złożoność obliczeniową dzięki temu, że jest algorytmem lokalnym i nie zakłada minimalizacji pasma w przypadku istnienia kliku możliwych zbiorów o tej samej liczbie ścieżek.

Porównanie omówionego algorytmu z algorytmem Pey prowadzi do wniosku, że te algorytmy są w istocie bardzo podobne. Pey jest algorytmem optymalnym lokalnie pod względem liczby wywłaszczonych ścieżek. Jego jakość jest więc dobrym poziomem odniesienia przy interpretacji wyników badań pozostałych algorytmów.

```plaintext
1  procedure algorithm_OptRC_local ()
   # algorytm lokalny, dla uproszczenia podano procedurę dla
   # pojedynczego węzła, powtarzaną w razie potrzeby w następnych węzłach
2      W := ∅               # inicjować listę kandydatów
3      S_e := α_f(e) − β_q    # oblicz początkowy bilans pasma
4      B_r(e) := max (0, −S_e) # oraz brakujące pasmo
5      K_e := cand_preselect_on_link (e) # wyznaczenie lokalnego zbioru
       # potencjalnych kandydatów
6      while $B_r(e) \neq 0$ do # dopóki brakuje pasma
7         i := 0              # oznaczenie pierwszego przebiegu pętli for
8         for each $k$ in $K_e$ # dla każdego potencjalnego kandydata
9           if $i = 0$ and $b_k \geq B_r(e)$ then $p := k$ # zapamiętaj pierwszą ścieżkę ze zbioru $K_e$
10          if $i > 0$ and $b_k \geq B_r(e)$ and
11              ($b_k < b_p$ or ($b_k = b_p$ and $h_k > h_p$)) then $p := k$
12     end for
13     if $i = 0$ then # jeśli brak pojedynczej ścieżki
14         $i := 1$, $p := K_e[0]$ # o pasmie większym niż wymagane
```

59
for each $k$ in $K_e$ do  # wybierz największą dostępną ścieżkę
    if $b_k > b_p$ or ($b_k = b_p$ and $h_k > h_p$)) then $p := k$
end if

$K_e := K_e \setminus \{p\}$  # usuń ścieżkę z listy
$W := W \cup \{p\}$  # dodaj ścieżkę do kandydatów
$S_e := S_e + b_p$  # uaktualnij bilans pasma
$B_r(e) := \max (0, -S_e)$  # oblicz brakujące pasmo
while
return $W$  # zwróć zbiór kandydatów
end

Rys. 3.11. Algorytm optymalny (lokalny) dla minimalizacji liczby wywłaszczeń.

Optymalny algorytm lokalny dla minimalizacji wywłaszczonego pasma został przedstawiony na rys. 3.12. Dokonuje on sprawdzenia wszystkich możliwych kombinacji w zbiorze potencjalnych kandydatów, poczynając od pojedynczej ścieżki, następnie analizując wszystkie pary, potem wszystkie 3-elementowe kombinacje itd., a kończąc na pojedynczym zbiorze zawierającym wszystkie dostępne ścieżki. Kolejne kombinacje są porównywane z najlepszym znalezionym dotychczas zbiorem, dla którego suma pasm jest najmniejsza, ale jednocześnie większa lub równa brakującemu pasmu. Złożoność obliczeniowa tego algorytmu jest bardzo duża i wynosi $O(|K_e|^2|K_e|^{r-1})$, co tym samym uniemożliwia jego praktyczne zastosowanie.

procedure algorithm_OptBw_local ()
  # algorytm lokalny, dla uproszczenia podano procedurę dla
  # pojedynczego węzła, powtarzaną w razie potrzeby w następnych węzłach
  $K_e := \text{cand\_preselect\_on\_link} (e)$  # wyznaczenie lokalnego zbioru
  # potencjalnych kandydatów
  $k := |K_e|$  # liczba potencjalnych kandydatów
  $\text{min\_sum} := \text{min\_prio} := \infty$  # inicjuj metryki
  $\text{length}(\text{best\_set}) := 0$  # wyzeruj najlepszy zbiór kandydatów
  for $r := 1$ to $k$ do  # dla każdej możliwej liczności
    $\text{for } n := 0$ to $r-1$ do  # dla każdej kombinacji r ścieżek
      $\text{set}[n] := n + 1$  # inicjuj tablicę kolejnymi liczbami
    end
  end
  for $i := 1$ to $k!/(k-r)!r!$ do
    # dla każdej kombinacji r ścieżek
sum_bw := sum_prio := 0  # wyzeruj sumy
for n := 0 to r-1 do
    p := K[set[n]-1]  # przypisz ścieżkę należącą do zbioru
    sum_bw := sum_bw + b_p  # uaktualnij sumę pasma
    sum_prio := sum_prio + h_p  # uaktualnij sumę priorytetów
end for
if sum_bw < min_sum and sum_bw ≥ B_r^{(e)} then  # jeśli suma pasm ścieżek jest wystarczająca
    # i mniejsza od dotychczas znalezionych
    best_set := set  # zapamiętaj ten zbiór ścieżek jako najlepszy
if sum_bw = min_sum and sum_bw ≥ B_r^{(e)} and sum_prio > min_prio then  # jeśli suma pasm ścieżek równa dotychczas
    # najlepszej ale sumarycznie niższe priorytety
    best_set := set  # zapamiętaj tez zbiór jako lepszy
    # generuj kolejną kombinację ścieżek
    pos := r
while pos > 0 and set[pos] = k – r + pos + 1 do
    pos := pos – 1
if set[pos] < k – r + pos + 1 then
    set[pos] := set[pos] + 1
    spos := pos + 1
for pos := spos to r do
    set[pos] := set[pos-1] + 1
end for
end for
W := ∅  # inicjuj listę kandydatów
for pos := 0 to length (best_set)-1 do
    W := W ∪ K[set[pos]-1]  # przepisz ścieżki ze zbioru set do listy kandydatów
return W  # zwróć listę kandydatów
end

Rys. 3.12. Algorytm optymalny lokalny dla minimalizacji wywłaszczonego pasma.

Algorytmy o zasięgu domeny są znacznie bardziej złożone, gdyż wymagają sprawdzenia wszystkich ścieżek obecnych na wszystkich łączach na drodze połączeniowej, na których występuje niedostatek pasma. W dostępnej literaturze nie natrafiono na przykład takiego algorytmu. Implementacja opracowana przez autora została przedstawiona na rys. 3.13. Zastosowano tu parametr variant określający kryterium optymalnego wyboru: MIN_RC ozna-
cza minimalizację liczby wywłaszczeń, a MIN_BW minimalizację wywłaszczonego pasma. Algorytm działa na zasadzie wygenerowania wszystkich możliwych kombinacji kandydatów i porównaniu ich, po czym najlepszy wybór zostaje zwrócony jako lista kandydatów do wywłaszczenia.

```plaintext
procedure algorithm_Opt_domain (variant: {MIN_RC, MIN_BW})
W := ∅  # inicjuj listę kandydatów
K := cand_preselect_on_route ()  # wybierz potencjalnych kandydatów
k := |K|  # liczba potencjalnych kandydatów
for r := 0 to k-1 do  # dla każdej możliwej liczby kandydatów
    if W ≠ ∅  # znaleziono przynajmniej jeden zbiór
        and variant = MIN_RC then  # metoda minimalizacji
            break  # procedura zakończona sukcesem
        pos := 0
        set[0] := -1  # inicjalizacja zbioru kandydatów
    repeat
        spos := set[pos]
        while pos ≤ r do  # wygeneruj pierwszy lub kolejny zbiór
            spos := spos + 1
            set[pos] := spos
            pos := pos + 1
        end while
        if set[r] < k then
            Y := ∅  # aktualnie analizowany zbiór kandydatów
            for pos := 0 to r-1 do
                Y := Y ∪ K[set[pos]-1]  # utwórz zbiór kandydatów
            end for
            if is_balance_positive (Y)  # zbiór zapewnia brakujące pasmo
                and found_better_set (W, Y, variant) then
                    W := Y  # zapamiętaj aktualny zbiór
                end if
            pos := r  # rozpoczęń od ostatniej pozycji
        end if
```
if set[pos] ≥ k-1 then # osiągnięto największy numer ścieżki
# na pozycji pos
pos := pos – 1 # przejdź na poprzednią pozycję
while pos > 0 and set[pos] ≥ k – (r – pos) – 1 do # przejdź do pozycji, na której
# można zwiększyć numer ścieżki
pos := pos – 1 # przejdź do pozycji, na której
end if
until set[0] ≥ k – r # zakończ gdy na pierwszej pozycji
# osiągnięto największy możliwy numer ścieżki
return W # zwróć listę kandydatów
end

function is_balance_positive (Y)
for each e in R do # oblicz bilans początkowy
S_e := a_f(e) – b_q # ile pasma brakuje na łączu
# dodaj pasmo jakie można uzyskać z wywłaszczenia
for each p in Y do # dla każdej ścieżki ze zbioru Y
for each e in R do # na każdym łączy na wybranej drodze
if e ∈ R_p then # ścieżka-kandydat zajmuje łącz e
S_e := S_e + b_p # powiększ bilans o pasmo ścieżki p
# sprawdź obliczony bilans na każdym łączu
for each e in R do # na każdym łączu na wybranej drodze
if S_e < 0 then
return FALSE # błąd: zbiór Y zwalnia za mało pasma
return TRUE # sukces: zbiór Y zwalnia # wystarczającą ilość pasma
end
end

function found_better_set (W, Y, variant)
better := FALSE # jest to pierwszy akceptowalny zbiór
if W = ∅ then

return TRUE
if variant = MIN_RC then # minimalizuj liczbę wywłaszczeń
if |Y| < |W| then # Y zawiera mniej ścieżek do wywłaszczenia
return TRUE
if |Y| = |W| then # jeśli oba zbiory zawierają taką samą liczbę kandydatów
if sum_bw(Y) < sum_bw(W) then # decyduje mniejsze pasmo
return TRUE
if sum_bw(Y) = sum_bw(W) then # w przypadku jednakowych pasm
and sum_prio(Y) > sum_prio(W) then # decydują niższe priorytety
return TRUE
end if
else begin # minimalizuj wywłaszczone pasmo
if sum_bw(Y) < sum_bw(W) then # zbiór Y zawiera mniejsze pasmo
return TRUE
if sum_bw(Y) = sum_bw(W) then # jeśli sumy pasm są identyczne
if |Y| < |W| then # decyduje mniejsza liczba wywłaszczeń
return TRUE
if |Y| = |W| then # jeśli liczby wywłaszczeń są jednakowe
and sum_prio(Y) > sum_prio(W) then # decydują niższe priorytety
return TRUE
end if
end if
return FALSE # zbiór W pozostaje lepszy od aktualnego
end

Rys. 3.13. Uniwersalny algorytm optymalny o zasięgu domeny.

Po zainicjowaniu zmiennych i zbiorów globalnych (linie 2-4) rozpoczyna się główna pętla wyboru kandydatów (linie 5-33). Każde powtórzenie tej pętli oznacza zwiększenie liczności zbioru kandydatów, rozpoczynając od zbiorów zawierających pojedyncze ścieżki, a kończąc na pojedynczym zbiorze zawierającym wszystkie ścieżki ze zbioru potencjalnych kandydatów K.

Każdy przebieg pętli rozpoczyna się od sprawdzenia (linie 6-7), czy w poprzednim przebiegu odnaleziono zbiór kandydatów. Jeśli tak jest i zadany kryterium wyboru była minimalizacja liczby wywłaszczeń, wówczas algorytm zostaje zakończony z sukcesem, gdyż kolejne powtórzeń spowodowałyby wygenerowanie zbioru o większej liczbie kandydatów, a więc gorszego w świetle wybranego kryterium. Jeśli warunek nie jest spełniony, wówczas następuje kontynuacja przebiegu wewnątrz pętli.
Pętla w liniach 11-32 odpowiada za wygenerowane i sprawdzenie jakości kolejnego zbioru kandydatów o tej samej liczności, zadanej przez zewnętrzną pętłę for (linie 5-33) w zmiennej \( r \). W pętli while (linie 13-17) następuje wygenerowanie listy wskaźników do elementów ze zbioru \( K \), a ścieżek numerów pozycji w zbiorze. Przykładowo, mając danych dziesięciu potencjalnych kandydatów i generując zbiór \( set \) o liczności \( 5 (r=4) \), generowane są kolejno ciągi, począwszy od \([0 1 2 3 4]\), następnie \([0 1 2 3 5]\) aż po \([5 6 7 8 9]\). W kolejnym etapie następuje utworzenie dodatkowego zbioru \( Y \) (linie 19-21), zawierającego ścieżki wybrane zgodnie z aktualną zawartością zbioru \( set \). Jeśli dany zbiór \( Y \) jest lepszy od poprzedniego (sprawdzenie w liniach 22-23), wówczas zostaje zapamiętany jako aktualny zbiór kandydatów \( W \). Na koniec następuje przygotowanie do wygenerowania kolejnego zbioru \( set \) (linia 27-31).

Po wyjściu z obu pętli, tj. po sprawdzeniu wszystkich możliwych kombinacji potencjalnych kandydatów, następuje zwrócenie najlepszego znalezionego zbioru \( W \) kandydatów (linia 34). Ponieważ jest to algorytm globalny, więc obliczenie i sprawdzenie bilansu pasma w funkcji \( is\_balance\_positive \) jest wykonywane na każdym łączu wymagającym wywłaszczania.

3.5.6. Inne opracowania

Zupełnie odmienne podejście do wywłaszczania połączeń (w sieciach ATM) przedstawiono w artykule [13]. Jedynym kryterium wyboru ścieżek-kandydatów jest w tym przypadku czas, który upłynął od ich utworzenia. Przedstawiono szeroką analizę teoretyczną szeregu algorytmów opartych na tej zasadzie. Wydaje się jednak, że podejście takie nie mieści się w koncepcji sieci MPLS, która z założenia jest siecią szkieletową a ścieżki połączeniami długoterminowymi, bez określonego rozkładu czasów trwania.

Podobne podejście zaproponowano w [2], określając dla każdej ścieżki czas, jaki upłynął pomiędzy jej utworzeniem a wywłaszczeniem, i na tej podstawie wyznaczając oczekiwany stopień zadowolenia klienta. Bazuje się tu na przykładowych czasach trwania transmisji dla połączeń określonego typu, która jednak, jak przyznają autorzy, jest znana jedynie użytkownikowi końcowemu. Niewątpliwie satysfakcja użytkownika jest bardzo istotnym aspektem wywłaszczania, choć dyskusyjne są założenia o określonym czasie wystarczającym na wykonanie pojedynczego transferu danych w ścieżce.

Autorzy artykułu [102] z kolei zaproponowali udoskonalenie algorytmu CSPF (ang. *Constraint Shortest Path First*) w taki sposób, aby ten w procesie wyboru drogi dla nowej ścieżki brał pod uwagę pasmo zajęte przez ścieżki o niższym priorytecie i obsługiwał mechanizm wywłaszczania. Jako algorytmu wywłaszczania autorzy używają omówionego wcześniej algorytmu optymalnego dla minimalizacji liczby wywłaszczeń (rys. 3.11). W artykule omówiono również Istotne dla wywłaszczania elementy protokołów RSVP-TE, CR-LDP i OSPF, a także przedstawiono szereg własności metryk i algorytmów zaproponowanych dla CSPF.

Nieco podobną zasadę zastosowano w [56]. Tu również już w procesie wyboru trasy bierze się pod uwagę pasmo dostępne dla ścieżki o określonym priorytecie, uwzględniając ewen-
tualne wykonywanie wywłaszczeń. Dodatkowo modyfikuje się funkcję kosztu łączy tak, aby uwzględnić priorytety potencjalnych kandydatów do wywłaszczenia oraz pasmo niezbędne do wywłaszczenia.

Zagadnienie rutingu CSPF świadomego istnienia wywłaszczeń opisano także w [108], w którym dodatkowo zaproponowano wariant algorytmu V-PREPT zaadoptowany z [73]. Wyniki symulacyjne uzupełniono aspektami praktycznej implementacji metody z użyciem protokołu RSVP-TE.

Jedne z najprostszych algorytmów wywłaszczania zaproponowano w [100,101]. Oba algorytmy są oparte na zasadzie losowego wyboru ścieżek do wywłaszczenia z dostępnego zbioru potencjalnych kandydatów. Nie ma możliwości wpływu na sposób wyboru ścieżek.

Bazując na oczywistym stwierdzeniu, że kolejność żądań utworzenia ścieżek ma wpływ na liczbę wywłaszczeń, zaproponowano [27] sortowanie żądań według pasma a następnie tworzenie ścieżek w odpowiedniej kolejności. Zaprezentowano wyniki symulacyjne liczby odrzuconych zgłoszeń i maksymalnego obciążenia łączy, bazując na pomiarach jednej wybranej topologii. W konkluzji podkreślono, że sortowanie kolejności żądań według pasma umożliwia zmniejszenie liczby odrzuczeń. W praktyce takie podejście jest ograniczone do specyficznych zastosowań, w których z góry znane są wszystkie żądania.

Istnieje kilka opracowań, w których autorzy dążą do minimalizacji liczby wywłaszczeń poprzez taki przydział zasobów, aby w ogóle unikać wywłaszczeń lub minimalizować ich liczbę. W sytuacji braku wolnego pasma jednym ze sposobów jest podjęcie próby zmniejszenia przydzielonego pasma tym ścieżkom, które przenoszą ruch elastyczny i w ten sposób uniknięcie wywłaszczeń [72,105]. Innym sposobem na ograniczenie liczby wywłaszczeń jest unikanie mocno obciążonych łącz przy wybieraniu trasy dla nowej ścieżki [10,32].

Dwa proste algorytmy wywłaszczania ścieżek ukierunkowane na sieci ze wsparciem dla technologii Differentiated Services (DiffServ) [19] zaprezentowano w [97]. Zakładają one wykorzystanie jednego z dwóch modeli Maximum Allocation Bandwidth Constraints Model (MAM) [11,63] lub Russian Dolls Bandwidth Constraints Model (RDM) [62], w których pula pasma, jakie może być zarezerwowane dla ścieżek o danym priorytecie, jest administracyjnie ograniczona.

Jednym z praktycznych problemów związanych z wywłaszczaniem jest niebezpieczeństwo zbyt częstego wywłaszczania ścieżek o niskim priorytecie utrzymania. Jedną z propozycji rozwiązania tego problemu jest ograniczenie częstości wywłaszczenia za pomocą żetonów (tokenów) przydzielanych ścieżkom i odbieranych w momencie wywłaszczenia [26].

Zagadnienie wywłaszczania w kontekście praktyki operatora telekomunikacyjnego omówiono w [38], gdzie w oparciu o istnienie czterech klas jakości obsługi: złotej (gold), srebrnej (silver), ekonomicznej (economy) i bez gwarancji (best-effort) wprowadzono dodatkowe ograniczenia na możliwość wzajemnego wywłaszczania pomiędzy ścieżkami z poszczególnych klas, np. unikanie wywłaszczeń ścieżek z klasy ekonomicznej przez ścieżki z klasy
srebrnej.
Należy również nadmienić, że podejmowano próby rozwiązania problemu wywłaszczenia za pomocą mechanizmów logiki rozmytej i algorytmów genetycznych [31,104].

3.6. Złożoność obliczeniowa


3.6.1. Algorytm GarGop

Algorytm GarGop/RC (rys. 3.5) redukuje się do postaci przedstawionej na rys. 3.14. Usunięto z niego wszystkie instrukcje znajdujące się poza główną pętlą. Dla łatwiejszej analizy włączono niezbędne części kodu wywoływanych funkcji.

```plaintext
1 repeat
2    # best_path ()
3    for each p in K
4        for each e in Ė
5            end for
6    end for
7    # update_bw_balance (p)
8    for each e in Ė
9        end for
10 until B_r = 0
```


Już nawet pobieżna analiza pozwala poczynić dalsze uproszczenia. Mianowicie, instrukcje zawarte w funkcji update_bw_balance można pominać, gdyż wykonywane są w szeregu z bardziej złożoną pętlą w funkcji best_path. Zatem ostateczna, zredukowana postać algorytmu GarGop/RC jest taka, jak na rys. 3.15.

Pętla C algorytmu GarGop/RC będzie więc wykonywana tyle razy, ile wynosi iloczyn liczby kandydatów (pętla A), liczby potencjalnych kandydatów (pętla B) i liczby łączących wymagających wywłaszczenia (pętla C). Ograniczeniem górnym dla tych trzech wielkości są odpowiednio liczba ścieżek Z, liczba ścieżek Z i liczba węzłów N. Zatem złożoność obliczeniowa algorytmu ma postać $O(NZ^2)$. 
Warto dodać, że w praktyce w sieciach złożonych z kilkudziesięciu węzłów liczba łącz wymagających wywłaszczenia zwykle nie przekracza dwóch [46] a zatem czynnik $N$ jest do pominięcia w tym i pozostałych analizowanych algorytmach.

Algorytm GarGop/BW (rys. 3.6) różni się od GarGop/RC obliczeniami zawartymi w funkcji best_path. Liczba przejściów pętli $C$ jest w obu algorytmach identyczna, a zatem złożoność algorytmu GarGop/BW jest taka sama jak dla GarGop/RC, tzn. $O(NZ^2)$.

### 3.6.2. Algorytm Pey

Algorytm Pey (rys. 3.7) na potrzeby analizy złożoności został zredukowany do docelowej postaci pokazanej na rys. 3.16.

Na tej podstawie natychmiast otrzymujemy liczbę wykonań wewnętrznej pętli $C$ równą iloczynowi liczby łącz wymagającej wywłaszczenia (pętla $A$), liczby kandydatów (pętla $B$) i liczby potencjalnych kandydatów (pętla $C$). Biorąc górne ograniczenia dla tych wielkości jako, odpowiednio, liczbę węzłów $N$, liczbę ścieżek $Z$ i liczbę ścieżek $Z$ otrzymujemy wartość złożoności obliczeniowej w postaci $O(NZ^2)$. 

```plaintext
1 for each $e$ in $\hat{R}$ # A: wykonywana dla każdego łącza w wywłaszczeniach
2 while $B_r(e) \neq 0$ do # B: wykonywana aż zebrany zostanie zbiór kandydatów
3 # best_path ()
4 for each $p$ in $K_e$ # C: wykonywana dla każdego potencjalnego kandydata
5 end for # C
6 end while # B
7 end for # A
```

Rys. 3.16. Analiza złożoności obliczeniowej algorytmu Pey.

3.6.3. Algorytm OliSco

Dekonstrukcja algorytmu OliSco (rys. 3.9) została przeprowadzona w dwóch etapach. Analizę rozpoczęto od postaci rozwiniętej, zawierającej wszystkie warianty oryginalnej procedury (rys. 3.17).

Rys. 3.17. Analiza złożoności obliczeniowej algorytmu OliSco, krok 1/2.

W przypadku ścieżek o różnych metrykach wewnętrzna pętla B będzie wykonywana tyle razy, ile wynosi iloczyn liczby łączy z wywłaszczaniem (pętla A) i liczby kandydatów (pętla B). Natomiast w przypadku jednakowych metryk ścieżek redukujemy cztery pętle C1…C4 do pojedynczej pętli C, gdyż w analizie złożoności obliczeniowej pomija się współczynniki całkowite [60]. Otrzymujemy w ten sposób postać algorytmu przedstawioną na rys. 3.18.

Rys. 3.18. Analiza złożoności obliczeniowej algorytmu OliSco, krok 2/2.

Łatwo jest zauważyć, że liczba przejść w pętli C powoduje odpowiednio zmniejszenie liczby przejść w pętli B (odpowiednio większa liczba ścieżek dodanych do listy kandydatów). Zatem wynik dla jednakowych metryk jest taki sam jak dla różnych metryk, tj. iloczyn liczby
łączy z wywłaszczaniem i liczby kandydatów. Daje to złożoność algorytmu równą \( O(NZ) \).

### 3.6.4. Algorytm BlaMeL

Algorytm BlaMeL (rys. 3.10) został zredukowany na potrzeby analizy złożoności obliczeniowej do postaci przedstawionej na rys. 3.19.

```plaintext
for each \( e \) in \( \hat{R} \)  # A: wykonywana dla każdego łąca z wywłaszczaniem 
  while \( B_r^{(e)} \neq 0 \) do  # B: wykonywana dla każdego priorytetu 
    if \( B_r^{(e)} \geq B_{hold[r]} \) then
      end
    else
      while \( B_r^{(e)} < 0 \) do  # C: wykonywana dla ścieżek 
        # o tym samym priorycie
        while \( i < [K_p[r]] - 1 \) and \( K_p[r][i+1] > B_r^{(e)} \) do  # D: przeszukiwana lista 
          end while
        # dla priorytetu \( r \)
      end while
    end if
  end while  # B
end for  # A
```

Rys. 3.19. Analiza złożoności obliczeniowej algorytmu BlaMeL, krok 1/2.

Pętla C może być w rozważaniach pominięta, gdyż każde jej przejście powoduje zmniejszenie liczby przejść pętli B, w przeciwieństwie do pętli D, która ma bezpośredni wpływ na czas wykonania. W najgorszym przypadku, gdy wszystkie ścieżki-kandydaci mają identyczny priorytet, otrzymujemy postać algorytmu przedstawioną na rys. 3.20.

```plaintext
for each \( e \) in \( \hat{R} \) do  # A: wykonywana dla każdego łąca z wywłaszczaniem 
  while \( B_r^{(e)} \neq 0 \) do  # B: wykonywana dla każdego priorytetu 
    # aż zebrany zostanie zbiór kandydatów
    while \( i < [K_p[r]] - 1 \) and \( K_p[r][i+1] > B_r^{(e)} \) do  # D: przeszukiwana lista 
      # dla priorytetu \( r \)
    end while
  end while
end for  # A
```

Rys. 3.20. Analiza złożoności obliczeniowej algorytmu BlaMeL, krok 2/2.

Zatem liczba powtórzeń pętli D algorytmu BlaMeL jest iloczynem liczby łączy z wy-
właszczeniem (pętla $A$), liczby kandydatów (pętla $B$) i liczby potencjalnych kandydatów (pętla $D$). To daje złożoność obliczeniową równą $O(NZ^2)$.

3.7. Analiza porównawcza

Mając podbudowę w postaci charakterystyki poszczególnych algorytmów możliwe jest dokonanie porównania ich cech i możliwości, bez czego owa charakterystyka byłaby w istocie niepełna. W tym celu wybrano cztery cechy, według których zdecydowano się dokonać oceny opisanych algorytmów.

1. **Zasięg algorytmów** określa, czy jest to algorytm lokalny czy globalny. W pierwszym przypadku decyzja podejmowana jest w oparciu o informacje dostarczone na pojedynczym łączu wyjściowym, a w drugim bierze się pod uwagę dane ze wszystkich łącz na trasie ściężki, w granicach domeny administracyjnej. Z uwagi na jakość osiąganych rezultatów większe możliwości mają algorytmy o zasięgu globalnym, co odbywa się kosztem większej ilości wymaganych informacji oraz większej złożoności implementacyjnej i obliczeniowej.

2. **Elastyczność kryteriów wyboru** umożliwia określenie celu algorytmu zależnie od preferencji użytkownika, choćby w ograniczonym zakresie. Niektóre metody na sztywno definiują funkcję kosztu i nie pozostawią żadnych możliwości konfiguracyjnych, podczas gdy inne zapewniają możliwość wpływu na kryteria wyboru ścieżek.

3. **Złożoność obliczeniowa** jest szacunkową, względną miarą czasu wykonywania algorytmu, rozważaną dla najgorszego możliwego przypadku. Określa ona to, czy dana metoda nadaje się do zastosowania w rzeczywistej sieci i jak zmienia się jej szybkość w zależności od rozmiaru danych wejściowych, np. liczby potencjalnych kandydatów. Przyjmuje się, że algorytmy o złożoności potegowej ($n$, $n^2$, $n^3$, itd.) są akceptowalne, natomiast algorytmy o złożoności wykładniczej ($a^n$, gdzie $a$ jest stałą, zwykle równą 2) już nie. Oczywiście im mniejsza złożoność (mniejszy wykładnik potęgi), tym lepiej.

4. **Jakość wyboru kandydatów** przejawia się w zdolności do minimalizacji kosztu wylaszczania, zgodnie z definicjami z rozdz. 3.4.1. Dodatkową cechą, która opracowanie może być nieistotna w wielu implementacjach, ale jest ważna omówienia, jest świadomość priorytetów, czyli taki sposób wyboru kandydatów, w którym ścieżki o wyższym priorytecie są wybierane rzadziej. O ile dany algorytm posiada tę cechę, to albo uzna przydzielone ścieżki za najważniejszy czynnik wyboru kandydatów (algorytm BlaMeL) lub pozwala na uwzględnienie go w parametrach algorytmu (algorytm OliSco).

Odpowiednie zestawienie oparte na trzech pierwszych cechach i świadomości priorytetów zawarto w tabeli 3.1. Czwarta cecha, określająca jakość wyboru, zostanie zweryfikowana w rozdziale szóstym, w którym opisano wyniki badań symulacyjnych. Z informacji zawartych w tabeli wynikają następujące wnioski. Z wyjątkiem algorytmu GarGop pozostałe mają za-
sięg lokalny, a zatem biorą pod uwagę tylko stan na bieżącym łączu. Z kolei algorytmy GarGop i Pey nie biorą pod uwagę priorytetu wywłaszczanych ścieżek, choć trzeba zaznaczyć, iż w dwóch pozostałych algorytmach różnie podchodzi się do wagi priorytetów, bo w BlaMeL jest to najważniejsze kryterium, a w OliSco konfigurowalne. Należy zaznaczyć, iż nawet jeśli algorytm nie bierze pod uwagę priorytetów, jego implementacja w sieci MPLS musi wykluczać z listy potencjalnych kandydatów ścieżki o wyższym priorytecie utrzymania niż priorytet utworzenia nowej ścieżki. Jeśli chodzi o elastyczność kryteriów, to tylko metoda OliSco oferuje taką możliwość, a na dodatek umożliwia dowolne ich określenie z użyciem kilku zmiennych. Można tu dodać, że oba omówione warianty algorytmu GarGop oparte są na dwóch różnych procedurach, które można zastosować zależnie od potrzeby, jednak trudno to traktować jako podejście elastyczne z punktu widzenia definicji kryteriów. W odniesieniu do złożoności obliczeniowej, dla wszystkich opisywanych metod jest ona akceptowalna, a szczególnie wyróżnia się tu algorytm OliSco, który cechuje najmniejszą złożoność.

Tabela 3.1. Porównanie algorytmów wywłaszczania.

<table>
<thead>
<tr>
<th>Algorytm</th>
<th>Zasięg</th>
<th>Świadomość priorytetów</th>
<th>Elastyczność kryteriów wyboru</th>
<th>Złożoność obliczeniowa</th>
</tr>
</thead>
<tbody>
<tr>
<td>GarGop</td>
<td>Globalny</td>
<td>Nie</td>
<td>Nie</td>
<td>$O(NZ^2)$</td>
</tr>
<tr>
<td>Pey</td>
<td>Lokalny</td>
<td>Nie</td>
<td>Nie</td>
<td>$O(NZ^2)$</td>
</tr>
<tr>
<td>OliSco</td>
<td>Lokalny</td>
<td>Tak</td>
<td>Tak</td>
<td>$O(NZ)$</td>
</tr>
<tr>
<td>BlaMeL</td>
<td>Lokalny</td>
<td>Tak</td>
<td>Nie</td>
<td>$O(NZ^2)$</td>
</tr>
</tbody>
</table>

Warto dodać, iż wszystkie omówione algorytmy heurystyczne należą do klasy algorytmów zachłannych [28], gdyż wybór pada zawsze na tą ścieżkę, która na danym etapie algorytmu ma największą wartość zadanej metryki. W efekcie ścieżka dodana do listy kandydatów nie jest już z niej usuwana. Jedynie algorytm OliSco dopuszcza taką sytuację, ale tylko w przypadku, gdy wśród rozpatrywanych potencjalnych kandydatów są ścieżki o identycznej metryce i pasmo jednej z nich przekracza pasmo wymagane w momencie rozpoczęcia działania algorytmu (patrz rys. 3.9, linie 21-23). Jak zobaczymy w rozdziale 4, algorytm może zupełnie odejść od tej zasady i dowolnie usuwać kandydatów wcześniej dodanych do listy.

Przedstawione zestawienie pozwala zauważyć trywialny fakt, że liczba dostępnych i jednocześnie uniwersalnych algorytmów nie jest duża, co w połączeniu z wysokim stopniem skomplikowania zagadnienia daje możliwości poszukiwania doskonalszych rozwiązań. Bazując na przedstawionym zestawieniu żaden z algorytmów nie jest zdecydowanie lepszy od innych. Bliski wysokiej ocenie jest algorytm OliSco ze względu na świadomość priorytetów i elastyczność kryteriów wyboru, choć ma zasięg lokalny, a jego prostota pozostawia wątpli-
wości odnośnie jakości generowanych rozwiązań.

Przedstawioną charakterystykę można podsumować stwierdzeniem, że wprawdzie wszystkie omówione algorytmy wydają się być akceptowalne pod kątem praktycznego zastosowania, to jednak nie dysponujemy efektywnym obliczeniowo i elastycznym algorytmem o zasięgu domeny. Efektem prac nad spełnieniem tych wymagań jest algorytm opisany w kolejnym rozdziale.
4. PROPONOWANY ALGORYTM

W ramach badań nad mechanizmami wywłaszczenia ścieżek opracowano i przetestowano własny algorytm heurystyczny. Przeprowadzone badania symulacyjne wykazały jego wysoką jakość w sieciach o różnej wielkości i topologii. Proponowany algorytm obejmuje zasięgiem domenę i umożliwia wybór kryterium wywłaszczenia. Cechuje go złożoność obliczeniowa podobnego rzędu jak pozostałych algorytmów heurystycznych.

4.1. Opis algorytmu

W trakcie badań istniejących heurystycznych algorytmów wywłaszczenia zdecydowano się na opracowanie nowego algorytmu globalnego, który byłby konkurencyjny w stosunku do istniejących algorytmów przy zachowaniu akceptowalnej złożoności obliczeniowej. W większości znanych algorytmów kolejne ścieżki wybiera się w pętli i dodaje kolejno do zbioru kandydatów. Proponowany algorytm działa inaczej. Tu kolejne ścieżki są dodawane do listy kandydatów (bez żadnych specyficznych warunków wstępnych), ale po każdym dodaniu ścieżki sprawdzany jest bilans pasma. Jeśli jest on dodatni, cały zbiór kandydatów jest przeglądany pod kątem nadmiarowych ścieżek, tzn. takich, które można usunąć ze zbioru kandydatów przy zachowaniu zdolności do przyjęcia nowej ścieżki. Oczeka się, iż taki sposób działania umożliwi bardziej efektywny wybór kandydatów niż ma to miejsce w przypadku opisanych wcześniej algorytmów. Proponowany algorytm został nazwany KNow, dla zachowania konwencji nazewnictwa użytej dla opisanych wcześniej algorytmów heurystycznych.

Rys. 4.1. Schemat działań zasadniczej części algorytmu KNow.

\[ \exists p \in W, e \in (k - R_p), S_e \geq b_p \Rightarrow W := W \setminus \{p\} \quad (4.1) \]

Aby lepiej zrozumieć sens takiego sposobu działania algorytmu, na rys. 4.2 przedstawiono stosowny przykład. Ścieżki $p_1$ i $p_2$ zostały kolejno dodane do zbioru kandydатów $W$. Jest to stan początkowy, w którym następuje uruchomienie procedury eliminacji. Obliczono w niej, że bilans pasma, czyli wolne pasmo powiększone o sumę pasm kandydатów $p_1$ i $p_2$, jest nałącz 2 większe od pasma ścieżki $p_1$. Inaczej mówiąc, ścieżka $p_1$ powinna zostać usunięta z listy kandydатów, gdyż ścieżka $p_2$ samodzielnie zapewnia pasmo wystarczające dla przyjęcia nowej ścieżki. Zatem po zakończeniu eliminacji w zbiorze kandydатów pozostaje tylko ścieżka $p_2$, co jest zgodne z intuicyjnym wyborem.

W uzupełnieniu podanego opisu na rys. 4.3 pokazano uproszczoną postać pseudokodu dla tej części algorytmu, która jest odpowiedzialna za eliminację nadmiarowych kandydатów. W pętli (linie 1-11) sprawdzana jest każda ścieżka $p$ ze zbioru kandydатów $W$. Na każdym łańcuchu wymagającym wywłaszczania (pętla for w liniach 2-6) sprawdzany jest warunek na zachowanie ścieżki w zbiorze kandydатów, tj. czy bilans pasma $S_e$ ma mniejszą wartość niż pasmo ścieżki $b_p$ (pod warunkiem, że łańcuch $e$ należy do trasy $R_p$ ścieżki). Jeśli na którymś łańcuchu warunek jest spełniony, wówczas następuje skok do linii 10 w celu ominienia procedury usuwania kandydata i zaczyna się sprawdzanie kolejnej ścieżki. Jeśli jednak odwrotny warunek z linii 3 nie jest spełniony dla wszystkich badanych łańcuchów, ścieżka powinna być usunięta z listy kandydатów, co następuje w linii 8, po czym w linii 9 następuje uaktualnienie bilansu pasma $S_e$.

Należy zauważyć, że w procedurze eliminacji w takiej postaci założono, że każda ścieżka-kandydat przechodzi przez przynajmniej jedno łańcze wymagające wywłaszczania. Gdyby tak nie było, wówczas warunek w linii 3 nie mógłby być spełniony i ścieżka byłaby eliminowana bez względu na bilans pasma. Taka sytuacja nie wystąpi, gdyż z założenia w zbiorze kandydатów mogą znajdować się tylko te ścieżki, które przechodzą przez przynajmniej jedno łańcze wymagające wywłaszczania.
Stan początkowy: lista kandydatów obejmuje ścieżki $p_1$ i $p_2$.

Rys. 4.2. Ilustracja zasady działania procedury eliminacji w algorytmie KNow: ścieżka $p_1$ dodana wcześniej do listy kandydatów cała mieści się poniżej linii wymaganego pasma, a zatem nie powinna być wywłaszczana, ale usunięta z listy kandydatów.

Algorytm w postaci pełnego pseudokodu został przedstawiony na rys. 4.4. Lista kandydatów $W$ zostaje zainicjowana (linia 2) i wyznaczony zostaje zbiór $K$ potencjalnych kandydatów (na całej długości drogi), a lista sortowana jest zgodnie z zadawnym parametrem variant (linia 3). Następnie (linia 4) obliczony zostaje początkowy bilans pasma $S_e$ równy dla każdego łącza $e$ różnicy między wolnym pasmem $a_f^{(e)}$ a pasmem nowej ścieżki $b_q$ (por. wzór 3.14):

$$S_e = a_f^{(e)} - b_q. \quad (4.2)$$
for each $p$ in $W$ do  # sprawdź wszystkich aktualnych kandydatów
  for each $e$ in $\hat{R}$ do  # dla każdego łączego wymagającego wywłaszczania
    if $e \in R_p$ and $S_e < b_p$ then  # warunek odwrotny, na brak eliminacji
      goto check_next_candidate  # jeśli spełniony, sprawdź kolejnego kandydata
    end if
  end for
  # eliminacja: ścieżka $p$ jest nadmiarowa, zostanie usunięta z listy kandydatów
  $W := W \setminus \{p\}$  # usuń ścieżkę z listy kandydatów
  update_bw_balance()  # uaktualnij bilans pasma $S_e$
check_next_candidate:
end for

Rys. 4.3. Fragment algorytmu KNow odpowiedzialny za procedurę eliminacji.

Właściwa część algorytmu składa się z dwóch części. Pierwsza (linie 5-16), pomocnicza, jest uruchamiana tylko wtedy, gdy brakuje pasma tylko na jednym łączu (warunek w linii 5). Przeprowadza się wówczas poszukiwanie pojedynczego rozwiązania, tj. pojedynczej ścieżki o najmniejszym paśmie zapewniającym przyjęcie nowej ścieżki, o ile taka istnieje. Odbywa się to poprzez przejście wszystkich potencjalnych kandydatów w pętli for (linie 9-15). W zmiennych sgl_cand, sgl_bw i sgl_surplus zapisuje się odpowiednio wybranego kandydata, jego pasmo i osiagamą wartość straconego pasma. W drugiej części algorytmu (linie 17-33), wykonywanej niezależnie od liczby łączów wymagających wywłaszczania, następuje charakterystyczne dla algorytmu poszukiwanie najlepszego zbioru ścieżek. Dla każdej ścieżki $p$ należącej do zbioru potencjalnych kandydatów $K$ wykonuje się procedurę składającą się z dwóch kroków.

1. Ścieżka zostaje dodana do listy kandydatów (linie 18-19).
2. Nadmiarowe ścieżki zostają usunięte z listy kandydatów (linie 20-28).

W pierwszym kroku ścieżka $p$ zostaje dodana do listy kandydatów, co skutkuje zwiększeniem wartości bilansu pasma na łączach znajdujących się na trasie ścieżki $p$. Jeśli bilans na którymś z łączyc osiągnął przez to poziom większy od zera, to tak skonstruowany zbiór kandydatów $W$ spowoduje wywłaszczenie nadmiarowego pasma. To z kolei wskazuje, że zbiór kandydatów może zawierać zbyt wiele ścieżek i wyzwala krok drugi.

W drugim kroku następuje sprawdzenie każdego ścieżki $l$ dodanej do tej pory do zbioru kandydatów. Jeśli pasmo ścieżki $l$ jest mniejsze niż bilans pasma na każdym łączu wymagającym wywłaszczania (warunek w linii 23), wówczas ta ścieżka jest usuwana z listy kandydatów (linie 24-25).

Na zakończenie procedury następuje sprawdzenie, czy pasmo uzyskane w efekcie wywłaszczenia wybranych kandydatów jest wystarczające do przyjęcia nowej ścieżki (linie 30-33). Jedynym powodem niepowodzenia może tu być zbyt mały zbiór potencjalnych kandyda-
tów, tzn. suma pasm wszystkich potencjalnych kandydatów nie wystarcza do przyjęcia nowej ścieżki.

```plaintext
procedure algorithm_KNow (variant)
     W := ∅  # inicjuj listę kandydatów
     K := cand_preselect_on_route (variant)  # utwórz listę
     # potencjalnych kandydatów
     calculate_bw_balance ()  # oblicz początkowy bilans pasma
     if |K| = 1 then  # wywłaszczanie tylko na jednym łączu
          sgl_surplus := sgl_bw := 0  # inicjalizuj metryki dla
          sgl_cand := ∅  # pojedynczego rozwiązania
          for each p in K do  # dla każdego potencjalnego kandydata
               if b_p ≥ B_r(e) and (sgl_cand = ∅ or b_p < sgl_bw) then
                    sgl_cand := p  # zapamiętaj ścieżkę i jej parametry
                    sgl_bw := b_p
                    sgl_surplus := b_p - B_r(e)
               end if
          end for
     end if

     # POCZÄTEK PROCEDURY DODAWANIA I ELIMINACJI KANDYDATÓW
     for each p in K do  # dla każdego potencjalnego kandydata
          W := W ∪ {p}  # dodaj p bezwarunkowo do listy kandydatów
          update_bw_balance ()  # uaktualnij bilans pasma
          if surplus_generated () then  # występuje nadmiar zebranego pasma
               for each / in W do  # sprawdź wszystkich aktualnych kandydatów
                    if path_under_surplus (l) then  # ten kandydat jest nadmiarowy
                         W := W \ {l}  # usuń ścieżkę z listy kandydatów
                         update_bw_balance ()  # uaktualnij bilans pasma
                    end if
               end for
          end if
     end for  # wybór kandydatów zakończony

     # KONIEC PROCEDURY DODAWANIA I ELIMINACJI KANDYDATÓW
```
for each $e$ in $\hat{R}$ do  # sprawdź bilans na każdym łącze
    if $S_e < 0$ then  # procedura zakończona niepowodzeniem
        return $\emptyset$
    end for
    # procedura zakończona sukcesem
    if $\text{sgl\_cand} \neq \emptyset$  # jeśli wystarcza pojedyncza ścieżka
        and $S_g > \text{sgl\_surplus}$ then  # i zapewnia mniejszą nadwyżkę pasma
            return $\text{sgl\_cand}$  # zwróć pojedyncze rozwiązanie
        else return $W$  # zwróć listę kandydatów
    end

procedure cand_preselect_on_route (variant)

$K := \emptyset$  # lista sortowana zgodnie z variant
for each $p$ in $P$ do  # dla każdej ścieżki
    if $h_p > s$  # która może zostać wywłaszczona
        and path_on_route ($R_p$, $p$) then
            $K := K \cup \{p\}$  # dodaj ścieżkę do listy
            # potencjalnych kandydatów
        end
end

procedure calculate_bw_balance ()  # oblicz bilans początkowy

for each $e$ in $\hat{R}$ do  # na każdym łączu na wybranej drodze
    $S_e := a_f(e) - b_q$  # od wolnego pasma
        # odejmij pasmo nowej ścieżki
    $B_r(e) := \max (0, -S_e)$  # brakujące pasmo to odwrotność nadmiaru
end for

# dodaj pasmo kandydatów
for each $p$ in $W$ do  # dla każdego kandydata
    for each $e$ in $\hat{R}$ do  # na każdym łączu na wybranej drodze
        $S_e := S_e + b_p$  # dodaj do bilansu pasmo ścieżki
    end
    # uaktualnij brakujące pasmo
    for each $e$ in $\hat{R}$ do  # dla każdego łącza na wybranej drodze
        $B_r(e) := \max (0, -S_e)$  # przelicz brakujące pasmo
    end
end

procedure update_bw_balance ($p$)
for each $e$ in $\hat{R}$ do  
# dla każdego łącza na wybranej drodze  
$S_e := S_e + \beta(p,e)b_p$  
# uaktualnij bilans pasma  
$B,(e) := \max (0, -S_e)$  
# uaktualnij brakujące pasmo  
end for  

end  

procedure surplus_generated ()  
for each $e$ in $\hat{R}$ do  
# dla każdego łącza na wybranej drodze  
if $S_e > 0$ then  
# znaleziono nadmiarowe pasmo  
return TRUE  
end for  
return FALSE  
# nie ma nadmiarowego pasma  
end  

procedure path_under_surplus $(p)$  
for each $e$ in $\hat{R}$ do  
# dla każdego łącza na wybranej drodze  
if $b_p \geq S_e$ then  
# ścieżka nie jest nadmiarowa  
return FALSE  
end for  
return TRUE  
# ścieżka jest nadmiarowa na każdym łączu  
end

Rys. 4.4. Algorytm KNow.

4.2. Analiza algorytmu

Porównanie istniejących metod heurystycznych w rozdziale trzecim zakończono stwierdzeniem, że nie dysponujemy algorytmem, który posiadałby następujące cztery pożądane cechy.

1. **Efektywność obliczeniowa**, umożliwiająca zastosowanie w rzeczywistych sieciach o dużej liczbie węzłów i ścieżek.
2. **Elastyczność konfiguracyjna**, umożliwiająca określenie przez użytkownika kryteriów wyboru kandydatów (patrz rozdz. 3.4.1).
3. **Zasięg obejmujący całą domenę**, aby wykorzystać znajomość topologii sieci i położenia ścieżek do dokonania wyboru możliwie najlepszego z punktu widzenia całej domeny.
4. **Jakość**, przejawiająca się uzyskiwaniem dobrych wyników w porównaniu z innymi badanymi algorytmami heurystycznymi (patrz rozdz. 3.4.1).
W oparciu o te cechy zostanie przeprowadzona analiza zaproponowanego algorytmu.

4.2.1. Efektywność obliczeniowa

Efektywność obliczeniową charakteryzuje się poprzez badanie złożoności obliczeniowej. Złożoność wielomianowa \( (n, n^2, n^3, \text{itd.}) \) jest akceptowalna, natomiast wykładnicza \( (2^n) \) wyklucza zastosowanie algorytmu w praktyce. Dla zaproponowanego algorytmu wykonano podobną analizę, jaką zrobiono wcześniej dla omówionych algorytmów heurystycznych. Na rys. 4.5 zebrano te elementy kodu algorytmu KNow, które są istotne dla określenia złożoności, przy czym do kodu włączono pętle zawarte w wywoływanych procedurach.

```
for each p in K do # A: dla każdego potencjalnego kandydata
    # update_bw_balance ()
for each e in \( \hat{E} \) do # B: dla każdego łącza na wybranej drodze
    end for
    # surplus_generated ()
for each e in \( \hat{E} \) do # C: dla każdego łącza na wybranej drodze
    end for
    if … then
        for each l in W do # D: dla wszystkich aktualnych kandydatów
            # path_under_surplus (l)
            for each e in \( \hat{E} \) do # E: dla każdego łącza na wybranej drodze
                end for
            if … then
                # update_bw_balance ()
                for each e in \( \hat{E} \) do # F: dla każdego łącza na wybranej drodze
                    end for
                end if
            end for
        end if
        end for
    end if
end for
```

Rys. 4.5. Analiza złożoności obliczeniowej algorytmu KNow, krok 1/3.

Algorytm składa się więc aż z sześciu pętli i posiada cztery poziomy zagnieżdżenia, choć możliwe jest tu wykonanie kilku redukcji. Po pierwsze, pętla B wykonuje się w szeregu z pętlą C, więc na potrzeby wyliczeń złożoności obliczeniowej może być pominięta. Po drugie, pętla F również wykonuje się w szeregu z pętlą E, co także czyni ją zbędną w dalszej analizie. Usuwając także nieistotne warunki if w liniach 8 i 13 otrzymujemy postać pokazaną na rys. 4.6.
Można teraz zauważyć, że pętla C jest wykonywana w szeregu z pętlami D i E, zatem po jej pominięciu otrzymujemy ostateczną postać przedstawioną na rys. 4.7.

Stąd otrzymujemy następującą liczbę powtórzeń pętli: (liczba kandydatów)² · (liczba łącz z wywłaszczaniem). To daje złożoność obliczeniową \( O(NZ^2) \), identyczną jak dla większości omówionych wcześniej algorytmów, a więc jak najbardziej akceptowalną.

Analizując średnią liczbę przebiegów w realnych sytuacjach, jest ona równa iloczynowi liczby potencjalnych kandydatów, średniej liczby kandydatów i liczby łącz z wywłaszczaniem. Jeśli weźmiemy pod uwagę to, że w praktyce zwykle średnia liczba łączy z wywłaszczaniem nie przekracza 2 [46], wówczas można ten czynnik pominać. Z kolei liczba kandydatów do wywłaszczenia zwykle nie przekracza kilku ścieżek i w niewielkim stopniu zależy od warunków badań. Możemy więc podsumować, że złożoność obliczeniowa jest proporcjonalna do liczby potencjalnych kandydatów. Pozwala to z wysokim prawdopodobieństwem uznać algorytm za efektywny obliczeniowo, a przez to możliwy do zastosowania w praktycznych realizacjach.

**4.2.2. Elastyczność konfiguracyjna**

Elastyczność konfiguracyjna w przypadku proponowanego algorytmu wyraża się poprzez możliwość zmiany sposobu sortowania zbioru \( K \) potencjalnych kandydatów, zaimplemento-
wanego w postaci listy. Nie wymaga to znaczącej rozbudowy implementacji metody, a jedynie wprowadzenia wariantowości w funkcji dodającej ścieżkę do zbioru $K$. Ogólna zasada jest taka, że typy ścieżek preferowanych do wyboru jako kandydatów powinny być umieszczane na końcu listy. Przykładowo, w celu minimalizacji liczby wywłaszczeń należy wybierać ścieżki duże, a więc sortować zbiór $K$ według pasma rosnącego. Z kolei aby zmniejszyć stracone pasmo należy preferować wybór małych ścieżek, a więc sortować zbiór $K$ według pasma malejącego. Natomiast posortowanie zbioru $K$ według wartości priorytetu rosnąco spowoduje, że preferowane będzie wywłaszczenie ścieżek o wysokim liczbowo, czyli niskim priorytecie.

4.2.3. Zasięg algorytmu

Trzecim pożadanym kryterium dobrego algorytmu jest zasięg obejmujący domenę. Przewidywany algorytm z założenia posiada tę cechę, gdyż w procesie wyboru ścieżek analizuje stan na całej drodze połączeniowej, w ramach zarządzanej domeny. Dzięki temu w sytuacji gdy wywłaszczenie następuje na więcej niż jednym łączu, możliwe jest osiągnięcie lepszego rezultatu niż przy użyciu metody lokalnej.

4.2.4. Jakość algorytmu

Ostatnią, choć kluczową, cechą algorytmu jest jakość, rozumiana (zgodnie z definicją w rozdz. 3.4.1) jako zdolność do wyboru takich zbiorów kandydatów $W$, dla których średni koszt wywłaszczenia będzie niski. Na tym etapie jednak pozostawimy tę cechę bez oceny, gdyż trudno jest w przypadku tego algorytmu dokonać teoretycznej analizy jakościowej. Bardziej uzasadnione jest przeprowadzenie badań symulacyjnych poszczególnych algorytmów i porównanie ze sobą otrzymanych wyników. Temu poświęcone zostały dwa kolejne rozdziały pracy, w których opisano warunki przeprowadzonych badań a następnie zaprezentowano i omówiono wyniki.
5. ŚRODOWISKO BADAWCZE


5.1. Badane algorytmy i miary ich oceny


1. GarGop/RC: algorytm GarGop dla minimalizacji liczby wywłaszczeń (rys. 3.5).
2. GarGop/BW: algorytm GarGop dla minimalizacji wywłaszczonego pasma (rys. 3.6).
3. Pey: algorytm Pey z pojedynczą pętlą (rys. 3.7, 3.8).
4. OliSco/RC: algorytm OliSco (rys. 3.9) dla minimalizacji liczby wywłaszczeń (wzór 3.41: \( X_1=0, X_2=1, X_3=0, X_4=0 \)).
5. OliSco/BW: algorytm OliSco (rys. 3.9) dla minimalizacji wywłaszczonego pasma (wzór 3.41: \( X_1=0, X_2=0, X_3=1, X_4=0 \)).
7. KNow: algorytm KNow (rys. 4.4) z listą kandydatów sortowaną według pasma, malejąco.

Jest wiele różnych kryteriów, według których dokonuje się oceny jakościowej algorytmów wywłaszczania, ale najważniejszymi są liczba wywłaszczeń i pasmo wywłaszczonej ścieżki. W praktyce stosuje się różne modyfikacje tych wielkości, takie jak:

- średnia liczba wywłaszczeń \( \overline{M} \),
- średnie stracone pasmo \( \overline{B_1} \),
- średnie stracone pasmo sieciowe \( \overline{B_{net}} \),
- średnia wartość metryki złożonej \( Q \), która uwzględnia jednocześnie liczbę wywłaszczeń i stracone pasmo sieciowe, (zdefiniowanej wzorem 5.1),
- średnie wywłaszczone pasmo \( \overline{B} \),
- średnie wywłaszczone pasmo sieciowe \( \overline{B_{net}} \),
- średnie względne pasmo stracone \( \overline{b_1} \),
- średnie względne pasmo stracone sieciowe \( \overline{b_{net}} \).

Są jeszcze inne wielkości, jakie można mierzyć, choć zwykle należy je traktować jako wielkości obrazujące skalę zmian powodowanych w sieci poprzez wywłaszczenia, a nie jako

85
cechy określonych metod, np.:
− średni poziom $\bar{h}$ rezerwacji pasma na poszczególnych łączach,
− średnia długość $\bar{g}$ kaskady wywłaszczeń,
− średnia liczba $\bar{H}$ ścieżek w kaskadzie wywłaszczeń,
− prawdopodobieństwo $p_{pre}$ wywłaszczenia, tj. prawdopodobieństwo spowodowania wywłaszczeń przez nowo tworzoną ścieżkę (5.2),
− prawdopodobieństwo $p_{reg}$ odrzucenia żądania utworzenia ścieżki, spowodowanej brakiem zasobów (5.3).

Niektróre z wymienionych wielkości nie zostały wcześniej zdefiniowane a inne wymagają tutaj dodatkowego komentarza.

Średnia liczba wywłaszczeń $\bar{M}$ przekłada się na liczbę operacji, jakie należy wykonać, aby przyjąć nową ścieżkę. Zwykle zależy nam na minimalizacji tej wielkości, gdyż każda wywłaszczona ścieżka to przerwa w ruchu odczuwalna przez klienta. Ponadto każdą wywłaszoną ścieżkę należy ponownie utworzyć, co oznacza zwiększenie ruchu sygnalizacyjnego w sieci i dodatkowe obciążenie routerów. Cechą dobrego algorytmu jest więc dążenie do minimalizacji tej wielkości.

Średnia wartość straconego pasma $\bar{B_x}$ ma również niebagatelnne znaczenie. Im większe ścieżki zostają wywłaszczone, tym potencjalnie większy ruch zostaje zakłócony, a co za tym idzie, straty dla operatora są większe. Zależy nam zatem, aby algorytm wywłaszczania w ten sposób wybierał ścieżki, aby nie zwalniać duży więcej pasma, niż jest rzeczywiście potrzebne dla przyjęcia nowej ścieżki. Oczywiście jest, że strat pasma nie da się zupełnie wyeliminować, ale stosunek pasma wywłaszczonego do wymaganego powinien być możliwie mały.

Oba omówione parametry są dobrymi wyznacznikami jakości metody, gdyż oddają bezpośrednio to, czego od metod oczekujemy. Należy jednak pamiętać, że w praktyce ważniejsze niż konkretne wartości tych parametrów są różnice w wynikach osiąganych przez poszczególne algorytmy, które zostały zmierzone przy zachowaniu identycznych warunków pomiarowych i dla tych samych sieci.

Pojawia się pytanie, czy możliwe jest jednoczesne utrzymanie na niskim poziomie obu wielkości, tj. liczby wywłaszczeń i straconego pasma. W celu zbadania tego zagadnienia zdefiniowano złożoną metrykę $Q$, która uwzględnia zarówno liczbę wywłaszczeń jak też wywłaszczone pasmo sieciowe. Przykładową prostą definicję przedstawiono w postaci wzoru (5.1). Tak skonstruowana metryka $Q$ zawiera się w przedziale $[0;1]$, przy czym im algorytm jest lepszy w zakresie liczby wywłaszczeń i wywłaszczonego pasma, tym wyższa wartość metryki.

$$Q = \frac{B_{net}}{MB_{net}}$$ (5.1)
Komentarza wymagają także wielkości opisujące wywłaszczone pasmo sieciowe \( B_{net} \) i stracone pasmo sieciowe \( B_{xnet} \). Wychodzi się tu z założenia, że wywłaszczenie dłuższej ścieżki jest bardziej szkodliwe niż wywłaszczenie ścieżki krótszej, gdyż zajmuje więcej czasu i wymaga zaangażowania większej liczby ruterów. Aby to uwzględnić, można się pasma wywłaszczonych ścieżek przez ich długości. Ten parametr jest często pomijany przy omawianiu metod o zasięgu lokalnym, gdyż te nie biorą pod uwagę długości wywłaszczonych ścieżek, a przez to nie mają wpływu na jego wartość.

Prawdopodobieństwo \( p_{pre} \) spowodowania wywłaszczenia określa się jako stosunek liczby \( k_{prepr} \) utworzen ścieżek, które kończą się spowodowaniem wywłaszczenia, do ogólnej liczby \( k_{cr} \) utworzonych ścieżek.

\[
p_{pre} = \frac{k_{prepr}}{k_{cr}}
\]

(5.1)

Prawdopodobieństwo \( p_{req} \) odrzucenia żądania utworzenia ścieżki to stosunek liczby \( k_{req} \) odrzuconych żądań do ogólnej liczby \( k_{req} \) żądań utworzenia ścieżek.

\[
p_{req} = \frac{k_{req}}{k_{req}}
\]

(5.1)

5.2. Założenia do modelu symulacyjnego

Mając zdefiniowane kryteria oceny, można określić założenia dla środowiska, w jakim zostaną przeprowadzone badania algorytmów. Z uwagi na brak dostępu do środowiska badawczego opartego na rzeczywistych sieciach MPLS, weryfikację algorytmów przeprowadzono w oparciu o badania symulacyjne. Cechy, jakimi powinien charakteryzować się model symulacyjny to:

− budowa modułowa dla zapewnienia łatwego jego rozszerzania o nowe funkcje i algorytmy,
− możliwość badania dowolnej struktury sieci o wielkości przynajmniej 100 węzłów,
− wejście (opis sieci i parametry badań) oraz wyjście (wyniki) w postaci plików tekstowych,
− wyniki nie powinny wymagać dalszego złożonego przetwarzania statystycznego.

Najważniejszym z przyjętych założeń jest otwartość implementacji, przejawiająca się łatwym rozszerzaniem modelu o nowe funkcjonalności. W szczególności chodzi o to, aby móc stosunkowo szybko przeprowadzić badania nowych algorytmów lub rozszerzyć możliwości badawcze o nowe rodzaje pomiarów. Implementacja powinna również umożliwiać, po niezbędnych rozszerzeniach, prowadzenie badań nad innymi mechanizmami inżynierii ruchu, np.
protokołów dynamicznego rutingu. Dzięki temu wysiłek poniesiony na implementację zauważa w postaci uniwersalnego narzędzia, które będzie mogło być rozwijane i służyć badaniom wielu różnych aspektów sieci MPLS.

Zmiana struktury badanej sieci musi odbywać się poza samym narzędziem, w oparciu o plik konfiguracyjny. Program powinien w oparciu o te dane utworzyć i zasymulować strukturę sieci składającą się z zadanej liczby węzłów i źródeł, połączonych w określoną strukturę łączami o zadanym parametrach, obejmujących szybkość i opóźnienie.

Wejście i wyjście programu powinno być oparte na plikach tekstowych. Plik wejściowy powinien opisywać topologię sieci oraz parametry symulacji. Powinien podlegać kontroli składni i przynajmniej podstawowemu sprawdzeniu spójności zdefiniowanej sieci, aby zapewnić wczesne wykrycie błędów w opisach sieci. Wyniki pomiarów w postaci raportów o ujednoliconej strukturze powinny zostać skierowane do określonych w konfiguracji plików tekstowych.

Wyniki pomiarów powinny zawierać wartości średnie uzupełnione o przedziały ufności, które są niezbędne do oceny jakości uzyskanych wyników [39,61]. Aby to zapewnić, symulacja musi się składać z szeregu odcinków pomiarowych, po zakończeniu których zebrane zostaną wyniki częściowe. Program powinien dokonać ich automatycznego przetworzenia w celu wyznaczenia wartości średnich i przedziałów ufności. Liczba i długość odcinków pomiarowych oraz długość niezbędnego odcinka rozbiegowego powinny być określone w pliku wejściowym.

5.3. Realizacja modelu symulacyjnego

Mając dane założenia należało zdecydować się na wybór narzędzia symulacyjnego. Możliwy był wybór jednego z ogólnodostępnych simulatorów, np. ns2 [103] lub OMNeT++ [86]. Alternatywą było opracowanie własnego rozwiązania. Obie możliwości mają swoje zalety i wady. W przypadku wyboru istniejącej aplikacji ma się do dyspozycji gotowy produkt testowany przez dużą społeczność użytkowników. Z kolei własna implementacja umożliwia utworzenie środowiska spełniającego wszystkie postawione wymagania i łatwą do rozbudowy, a ponadto zapewniającą pełną kontrolę nad implementacją. Po rozważeniu dostępnych możliwości, zdecydowano się na zrealizowanie własnego symulatora sieci MPLS. Ważnymi argumentami przy podjęciu takiej decyzji były brak algorytmów wywłaszczania w najpopularniejszych projektach oraz wcześniejszy sukces implementacji symulatora algorytmów sterowania zjawiskiem zgłoszeń dla sieci ATM [48,68].

Powstały program o nazwie msim jest rozwojowym symulatorem sieci MPLS, umożliwiającym z założenia przeprowadzenie badań różnych mechanizmów inżynierii ruchu. Umieścił on zarówno prowadzenie szczegółowych symulacji na poziomie pakietów, jak też symulacji długookresowych zjawisk na poziomie zgłoszeń. Dzięki modułowej architekturze i wykorzystaniu mechanizmów obiektowości i dziedziczenia łatwo można go rozszerzyć.
o nowe funkcjonalności. Symulator, napisany w języku C++, został uruchomiony i przetestowany w systemach operacyjnych Microsoft Windows XP oraz Linux (dystrybucja Debian 5 „lenny”), z wykorzystaniem kompilatorów Borland C++ Builder oraz GNU gcc. Program posiada uproszczony interfejs użytkownika w postaci konsoli tekstowej, na którą kierowane są informacje o postępie symulacji i dodatkowe komunikaty, w tym informacje o wykrytych błądach (rys. 5.1).

Rys. 5.1. Typowe informacje kierowane na konsolę przez symulator msim.

Uzupełnieniem symulatora jest program o nazwie Vims z graficznym interfejsem użytkownika wspomagającym projektowanie badanych sieci i umożliwiającym automatyczne generowanie na ich podstawie plików konfiguracyjnych. Aplikacja działa w systemie operacyjnym Microsoft Windows XP i posiada interfejs okienkowy z możliwością pracy na wielu mapach sieci jednocześnie. Oferuje ona dodatkowo funkcję analizy statystycznej sieci, obejmującej rozkłady gęstości połączeń i długości najkrótszych tras (rys. 5.2).

5.3.1. Zasada działania

Program przeprowadza symulację zdarzeniową z czasem dyskretnym (ang. *descret event simulation*). Symulacja składa się odcinka rozbiegowego i zadanej przez użytkownika liczby odcinków symulacyjnych. Wyniki pomiarów po każdym odcinku symulacyjnym są zapamiętane i po wykonaniu ostatniego odcinka przeliczone w celu uzyskania wartości średniej i wariancji, co jest niezbędne do określania przedziałów ufności uzyskanych wyników.

Struktura blokowo-funkcjonalna symulatora została przestawiona na rys. 5.3. Najważniejsze obiekty wchodzące w skład programu to:

- *menedżer symulacji*, odpowiadający za właściwy przebieg procesu symulacji,
- *menedżer kolejki zdarzeń*, odpowiadający za przechowywanie i dystrybucję zdarzeń,
- *menedżer konfiguracji*, przechowujący informacje o sieci i odpowiadający za jej tworzenie,

Menedżer kolejki zdarzeń przechowuje zdarzenia generowane przez obiekty sieci i obiekty sterujące. Koordynuje również dostarczanie zarówno zdarzeń kierowanych do poszczególnych obiektów jak też zdarzeń rozgłoszeniowych, kierowanych do wszystkich obiektów symulacji. Do jego najważniejszych zadań należy też sterowanie bieżącym czasem symulacji.

Menedżer konfiguracji odczytuje plik wejściowy, który zawiera opis warunków symulacji, w tym topologię sieci, warunki ruchowe, typ i czas trwania symulacji oraz rodzaj generowanych wyników pomiarów. Wszystkie odczytane dane zostają wczytane do pamięci i udostępnione w postaci struktur danych innym obiektom symulatora. Najważniejszym jego zadaniem jest utworzenie symulowanej sieci i takie powiązanie logiczne obiektów, aby tworzyły kompletną strukturę węzłów, źródeł i łącz, zdolną do przeprowadzenia symulacji.
Rys. 5.3. Schemat blokowo-funkcjonalny symulatora \textit{msim}.

Menedżer połączeń zajmuje się tworzeniem i kasowaniem ścieżek, w czym współpracuje z blokami rutingu i wywłaszczania. Przechowuje też informacje o wszystkich ścieżkach utworzonych w sieci.

Generator raportów przechowuje w czasie trwania symulacji statystyki generowane przez symulowane obiekty. Po każdym odcinku pomiarowym przelicza je, a po zakończeniu symulacji tworzy raporty z wynikami pomiarów. W przypadku, gdy symulacja składa się z kilku powtórzeń, odpowiednio łączy wyniki z poszczególnych powtórzeń, aby utworzyć zbiorcze raporty.

Menedżer topologii zajmuje się tworzeniem i kasowaniem obiektów symulowanej sieci w czasie trwania symulacji. Poprzez kreowanie nowych źródeł powoduje generowanie żądań utworzenia ścieżek i w ten sposób umożliwia prowadzenie masowych badań algorytmów wywłaszczania. W przeciwnieństwie do pozostałych obiektów sterujących jest tworzony przez
menedżera konfiguracji, a nie menedżera symulacji, gdyż jest blokiem opcjonalnym, tworzonym tylko w razie obecności odpowiedniego parametru w pliku konfiguracyjnym.

5.3.2. Symulowana sieć

Symulowana sieć jest modelem rzeczywistej sieci IP/MPLS, na którym testowane są mechanizmy inżynierii ruchu. W programie symulacyjnym sieć składa się z dowolnej (choć ograniczonej z góry) liczby następujących obiektów:

- węzeł, który jest odpowiednikiem rutera IP/MPLS,
- źródło, odpowiadające strumieniowi pakietów wprowadzanych do sieci IP/MPLS,
- łącze, będące modelem łącza fizycznego lub logicznego (tunelu) typu punkt-punkt.

Węzeł jest modelem nieskończenie szybkiego rutera IP/MPLS z buforami na wyjściu. Każdemu łączu wyjściowemu przydzielony jest konfigurowalny system buforów, którym może być pojedyncza kolejka FIFO (ang. First In First Out) lub zestaw oddzielnych buforów dla każdej klasy ruchu z przełącznikami typu WFQ (ang. Weighted Fair Queuing) oraz PQ (ang. Priority Queuing), jak to pokazano na rys. 5.4.

Węzeł definiuje się poprzez jego zadeklarowanie w pliku konfiguracyjnym i przydzielenie mu interfejsów sieciowych. Przykładowa definicja wygląda następująco:

```plaintext
node: id=1, name=Gdansk, x=287.535, y=32, size=1, igrp=1, ogrp=1;
addport: node=1, port=1, link=1, bufset=1;
addport: node=1, port=2, link=3, bufset=1;
addport: node=1, port=3, link=35, bufset=1;
```

Rys. 5.4. Typowy system buforowy wykorzystywany w symulacjach na poziomie pakietów.
Oprócz identyfikatora (id) oraz opisu (name) zdefiniowane zostały opcjonalne współrzędne (x i y), a także względny rozmiar (size) węzła oraz jego przyporządkowanie do grup wejściowych (igrp) i wyjściowych (ogrp). Te ostatnie parametry (igrp, ogrp) są używane przez menedżera topologii i w tym przypadku oznaczają, że w danym węźle mogą się zarówno rozpoznawać, jak i kończyć ścieżki generowane losowo. Do węzła dołączone trzy porty (interfejsy), skojarzone z łączami (link) o identyfikatorach 1, 3 i 35 (zdefiniowanymi oddzielnie). Wszystkie interfejsy posiadają identyczny system buforowy (bufset) o identyfikatorze 1 (zdefiniowany oddzielnie).

źródła pełnią rolę generatorów zagregowanych strumieni ruchu. Ich działanie zależy od zdefiniowanego w pliku konfiguracyjnym poziomu symulacji. Dla poziomu zgłoszeń aktywność źródeł ogranicza się do wygenerowania żądania przydziału ścieżki od węzła dostępowego do innego węzła w sieci. Parametry żądania obejmują identyfikator jednego z predefiniowanych typów źródeł i związany z nim klasę ruchu, co umożliwia określenie wymaganego pasma i warunków jakościowych QoS, w tym dopuszczalne prawdopodobieństwo strat pakietów oraz maksymalne opóźnienie. Nawet jeśli badania prowadzone są tylko na poziomie zgłoszeń, wymagane parametry QoS mogą być używane przez blok rutingu do wyznaczenia trasy spełniającej zadeklarowane warunki.

W przypadku, gdy symulacja odbywa się na poziomie pakietów, wtedy źródło oprócz wygenerowania żądania przydzielenia ścieżki tworzy i wysyła do węzła dostępowego pakiety IP o określonych rozkładach długości i rozkładach ich generacji.

```
source: id=1, type=1, link=22, destip=10.0.0.4, gcnt=0;
srctype: id=1, cos=1, mpl=100, ldt=none,
mspd=8e6, spddt=none, intens=100, sgend=expon, sspdd=expon;
cos: id=1, tos=ef, loss=1e-6, delay=0.05, setup=1;
```

W podanym przykładzie zdefiniowano źródło skojarzone z typem (type) 1, dołączone do łącza (link) 22, generujące pakiety do adresu IP (destip) 10.0.0.4. Dodatkowo pojawił się parametr określający ile pakietów ma być wygenerowanych za jednym razem (gcnt). Jeśli ten parametr ma wartość 0, to źródło działa tylko na poziomie zgłoszeń, tzn. jego aktywność ogranicza się do wygenerowania żądania przydzielenia ścieżki. Skojarzony ze źródłem typ (srctype) zawiera numer klasy (cos), średnią długość pakietu (mpl), średnią szybkość generowanego strumienia (mspd) wyrażoną w bitach na sekundę i odpowiadające obu wielkościom typy rozkładów (ldt i spddt). Pozostałe parametry są używane przez menedżera topologii do określenia częstości tworzenia źródeł danego typu. Klasa ruchu skojarzona z tym typem zawiera między innymi definicje parametrów QoS, tj. dopuszczalne straty pakietów (loss) oraz dopuszczalne opóźnienie (delay) na całej długości ścieżki, a poza tym priorytet utworzenia ścieżki (setup). Priorytet utrzymania (holding) nie został zdefiniowany, a zatem jest w tym przykładzie określony niejawnie jako równy wartości priorytetu utworzenia.
Łące w programie symulacyjnym jest obiektem statycznym, tzn. nie generuje żadnych zdarzeń a jest jedynie strukturą danych, wykorzystywaną przez obiekty węzłów i źródeł do poznania swojego sąsiedztwa oraz do określenia szybkości transferu danych i opóźnienia propagacyjnego. Przykładowa definicja łąca wygląda następująco.

```
link: id=1, name=Gda_War, bw=1.55e+08, pt=0.001246, outn=11, outp=1;
```

Definicja łąca obejmuje identyfikator (id), opis (name), pasmo (bw) wyrażone w bitach na sekundę, czas propagacji (pt), oraz numery węzła (outn) i portu (outp) po drugiej stronie łącza. Zdefiniowane w ten sposób łąca są jednokierunkowe, co wynika z założeń opisujących architekturę MPLS [96]. W razie potrzeby definiuje się dodatkowe łące skierowane w przeciwną stronę.

### 5.3.3. Przygotowanie badań

Przygotowanie i przeprowadzenie badań wywłaszczania z użyciem symulatora składa się zwykle z następujących etapów.

1. Zdefiniowanie scenariusza badań.
2. Utworzenie pliku konfiguracyjnego.
3. Uruchomienie serii symulacji.
4. Przetwarzanie i interpretacja wyników.

W każdym z tak zdefiniowanych etapów użytkownik jest wspomagany przez aplikacje msim i Vims. W dalszej części opisano dokładniej rolę obu programów w poszczególnych etapach.

Zdefiniowanie scenariusza badań polega między innymi na zbudowaniu topologii sieci, określeniu rodzaju pomiarów, wybraniu badanych algorytmów, określeniu wystarczającego czasu trwania symulacji. Na tym etapie aplikacja wspomagająca Vims umożliwia przygotowanie topologii w jednym z trzech trybów:

- ręczny,
- automatyczny,
- poprzez import i konwersję z innego formatu.

Ręczne tworzenie sieci polega na umieszczeniu na planszy węzłów oraz źródeł i połączeniu zdefiniowanych obiektów za pomocą łączy o określonych parametrach. Jest to najbardziej czasochłonny sposób, ale umożliwiający utworzenie dowolnej topologii. Sposób automatyczny polega na wygenerowaniu losowej topologii używając jednej z zaimplementowanych odmian algorytmu Waxman’a [107], co jest szczególnie przydatne w tworzeniu dużych sieci.

Z kolei tworzenie topologii poprzez import wykorzystuje się, jeśli dysponujemy plikiem z opisem sieci utworzonej przez zewnętrzne narzędzie. Program został wyposażony w procedurę importu plików XML zawierających topologie dostępne poprzez stronę internetową instytutu ZIB (Zuse-Institut Berlin) [18].
Niezależnie od wybranego sposobu tworzenia sieci, po jego zakończeniu niezbędne jest utworzenie pliku konfiguracyjnego. Ten etap wymaga wiele uwagi, gdyż zawartość pliku kompletnie definiuje środowisko badawcze, a ewentualna pomyłka może skutkować uzyskiwaniem niewłaściwych wyników. Na tym etapie wsparcie aplikacji Vims jest kluczowe, gdyż umożliwia automatyczne wygenerowanie gotowego do użycia pliku konfiguracyjnego, przy użyciu istniejącego szablonu. Dzięki temu unika się pomyłek w opisie topologii, a czas potrzebny na utworzenie pliku jest ograniczony do minimum. W praktyce tworzenie plików opisujących złożone sieci bez takiego wsparcia byłoby bardzo uciśliwe, czasochłonne i narażone na błędy. Po wygenerowaniu, plik jest dostępny do przeglądania i edycji, więc w razie potrzeby można go dodatkowo dostosować do indywidualnych potrzeb, choć część opisująca topologię rzadko wymaga dalszych zmian. Gdy plik jest gotowy, można przystąpić do symulacji.

Wykonanie pojedynczej symulacji może być niewystarczające. Zwykle potrzebne jest porównanie kilku wyników dla oceny różnic pomiędzy określonymi algorytmami, topologiami lub warunkami ruchowymi. Program symulacyjny umożliwia automatyczne powtarzanie symulacji, w oparciu o pojedynczy plik, w którym za pomocą specjalnej składni określone parametry definiuje się w postaci serii poszczególnych wartości, używanych w kolejnych powtórzeń. Dodatkowe ułatwienie wynikające z takiego trybu pracy symulatora to automatyczne zbieranie wyników z kolejnych powtórzeń i umieszczenie ich we wspólnej tabeli w pliku raportu. Znacznie przyspiesza to przetwarzanie i porównywanie wyników.

Po wykonaniu serii symulacji należy zwykle przetworzyć wyniki, a następnie dokonać ich interpretacji. Program wspomaga przetwarzanie wyników poprzez opisany wcześniej mechanizm łączenia wyników z całej serii symulacji. Dodatkowo program oblicza przedziały ufności i odpowiednio formatuje listę wyników. Wszystko to dzieje się automatycznie przed zakończeniem pracy symulatora i dzięki temu powstałe pliki mogą być użyte bezpośrednio do generowania wykresów i tabel.

Opisana charakterystyka nie wyczerpuje możliwości i cech programów msim i Vims. Więcej informacji na ten temat można odnaleźć w [45,47,49,53].

5.3.4. Pomiary

Wybór pomiarów dokonywanych przez program symulacyjny odbywa się poprzez ich specyfikację w pliku konfiguracyjnym. Pomiary odbywają się w następujący sposób. W ramach pojedynczego odcinka symulacyjnego wszystkie obiekty zdolne do generowania pomiarów, np. węzeł lub źródło, generują związane z poszczególnymi zdarzeniami próbki liczbowe. Każda pojedyncza próbka, która jest powiązana z generującym ją obiektem i typem zdarzenia, zostaje przekazana za pośrednictwem menedżera raportów do rejestru powiązanego z konkretnym typem pomiaru. Bieżąca wartość rejestru zostaje następnie zmodyfikowana wartością próbki w sposób, jaki został zdefiniowany dla zdarzeniawiązanego z próbką. Może to być jedna lub więcej operacji typu:
- uśrednienie wartości poszczególnym próbek,
- zliczanie liczby wystąpień próbek,
- sumowanie wartości wszystkich próbek,
- określenie wartości minimalnej lub maksymalnej.

Najczęściej wykonuje się pomiary wielkości należących do dwóch pierwszych typów, np. średniej liczby wywłaszczeń koniecznych do utworzenia jednej ścieżki albo liczby ścieżek utworzonych z użyciem wywłaszczeń. Na zakończenie każdego odcinka symulacyjnego rejestrzy zawierają częściowe wyniki pomiarów zebrane w ciągu tego odcinka. Wyniki częściowe nie mogą być jeszcze użyte do interpretacji badanych sieci, gdyż bez znajomości wyników z innych odcinków nie jest możliwe określenie przedziałów ufności.

Po zebraniu wyników częściowych trafiają one do powiązanych z nimi dwóch dodatkowych rejestrów. W pierwszym przechowuje się wartość średnią wyników częściowych, a w drugim wartość średnią kwadratów wyników częściowych, która służy potem do obliczenia przedziału ufności metodą t-Studenta. Wyznaczone w ten sposób pary (wartość średnia i przedział ufności) są wynikami generowanymi przez program i mogą być użyt do interpretacji badanych zjawisk.

W programie zaimplementowano kilkadziesiąt różnych typów pomiarów, wśród których są następujące (wartości średnie oznaczają tu wartości uśrednione po zdarzeniach a nie po odcinkach):
- liczba tworzonych (zagregowanych) źródeł w sieci, co odpowiada liczbie żądań zestawienia ścieżki,
- liczba przyjętych i odrzuconych żądań zestawienia ścieżki,
- średnia liczba wywłaszczeń ścieżek M,
- średnia wartość względnego straconego pasma lokalnego \( b_x^{(e)} \) i sieciowego \( \text{b}_{\text{net}} \),
- średnia wartość metryki złożonej \( Q \),
- średni poziom rezerwacji pasma \( b_{\text{p}}^{(e)} \),
- średnia długość kaskady wywłaszczeń \( g^{(p)} \),
- liczba niepowodzeń realokacji wywłaszczenych ścieżek,
- średnia długość ścieżek \( l \),
- liczba utworzeń ścieżek powodujących powstanie kaskady wywłaszczeń,
- średnia liczba ścieżek w kaskadzie wywłaszczeń \( H^{(p)} \).

Wszystkie wielkości mierzone są zbiornie oraz z podziałem na klasy tworzonych ścieżek. Dla każdej wielkości możliwe jest uzyskanie przedziału ufności, przy zadanym poziomie ufności, który domyślnie ustalony jest na wartość 0,95.

5.3.5. Weryfikacja modelu

Samodzielnie zrealizowany model symulacyjny powinien być z założenia traktowany jako niewiarygodny i podlegać szeregowi badań weryfikujących, które umożliwią wychylenie błędów i uzasadni wysoki stopień zaufania do uzyskanych przez program wyników.
Krytyczne podejście od własnego dzieła, choć trudne do osiągnięcia, jest najlepszym sposobem na prawidłową jego weryfikację w oparciu o dobre metody testowe.

W trakcie rozwijania programu prowadzono testowanie modelu korzystając z następujących metod i narzędzi.

1. Weryfikacja funkcjonowania programu w oparciu o pliki śladu.
2. Testy regresyjne na zgodność wyników po wprowadzeniu poważniejszych zmian.
3. Subiektywna analiza generowanych wyników.

Weryfikacje w oparciu o plik śladu (log) polegały w skrócie na wnikliwej jego analizie i porównaniu zawartości z przeprowadzoną niezależnie analizą oczekiwanego zachowania programu. Znajomość danych wejściowych, takich jak pasmo ścieżek oraz priorytet ścieżek, rozbudowana o pełną wiedzę na temat implementacji algorytmów umożliwiła weryfikację, czy kroki podejmowane przez program są oczywistą konsekwencją zaprogramowanego algorytmu. Jeśli działanie programu nie pokrywało się z oczekiwanym, wówczas konieczne było wyjaśnienie, skąd wynikają różnice, a w szczególności czy nie są wynikiem błędu programu. Proces takiego sposobu badań bywa żmudny, ale jest niezbędny do uzyskania wysokiego poziomu wiarygodności programu.

Przeprowadzone testy regresyjne miały na celu wyeliminowanie wtórnych błędów, tzn. takich, które mogły się pojawić w efekcie poprawiania przetestowanych już wcześniej bloków programu. Testy te polegały na powtórzeniu badań symulacyjnych przy użyciu dokładnie tych samych parametrów symulacji i topologii sieci, co badania przeprowadzone wcześniej. Korzystano tu z opcji programu umożliwiającej start symulacji z zadaną wartoścą początkową generatora liczb losowych. W ten sposób przy zadaniu tych samych warunków program powinien wygenerować wyniki pomiarów identyczne z wynikami odniesienia zebranymi wcześniej. Takie badania mogą być przeprowadzone w dużej części automatycznie łącznie z porównaniem zawartości plików raportów. Ewentualna różnica oznacza, że identyczne warunki początkowe prowadzą do innych wyników, co zwykle oznacza błąd w programie. To, czy błąd zawarty jest w nowej wersji czy w wersji odniesienia, wymaga indywidualnego sprawdzenia. Zdarzają się jednak sytuacje, gdy takie odstępstwo nie oznacza błędu, np. użycie innego kompilatora powoduje wykorzystanie innego sposobu inicjowania generatora liczb pseudolosowych, a to prowadzi do wyników nieco się różniących od wyników odniesienia. Inną tego typu sytuacją jest zmiana częstości lub sekwencji odczytu generatora liczb losowych, np. dodatkowy odczyt generatora w czasie symulacji przez nowy kod powoduje, że sekwencja liczb odczytywana przez pozostałe funkcje zmienia się, co też prowadzi do różnic w wynikach. Jednak nawet w takich przypadkach wyniki powinny być do siebie zbliżone, w granicach uzyskanych przedziałów ufności.

Badania oparte na analizie wyników polegały między innymi na sprawdzeniu, czy uzyskiwane wyniki mieszczą się w oczekiwany przedział oraz jaki wpływ na generowane wyniki mają zmiany pewnych parametrów. Ten rodzaj testów był w dużej części oparty na
doświadczeniu osoby przeprowadzającej badania a przez to ich wynik jest subiektywny. Mimo tego badania tego typu, prowadzone w sposób ciągły stanowią niezbędne uzupełnienie omówionych wcześniej metod.

Dzięki dążeniom do wyjaśnienia przyczyn wszelkich niejasności wyeliminowano szereg błędów programu i uzyskano z czasem wysoki poziom zaufania do wyników uzyskiwanych przez program. Biorąc pod uwagę prawo Murphy’ego, takie zaufanie musi pozostać ograniczone, jednak wyniki przeprowadzonych testów dają podstawę do optymizmu w tym zakresie. Nie bez znaczenia jest też fakt, iż koncepcja modelu i wyniki wygenerowane przez program zostały pozytywnie zweryfikowane przez recenzentów i zaprezentowane na szeregu krajowych i międzynarodowych konferencji naukowych oraz opublikowane na łamach czasopism [44,45,46,47,49,50,51,52]. Program został również praktycznie wykorzystany w projekcie badawczym dotyczącym koncepcji serwerów sterowania połączeniem w sieciach MPLS i ASON/GMPLS [53].

5.4. Zastosowanie

Zaimplementowany symulator jest uniwersalnym narzędziem, umożliwiającym prowadzenie badań nad różnymi metodami inżynierii ruchu. Szczególny nacisk położono na badania algorytmów wywłaszczania.

Program był wielokrotnie aktualizowany i rozbudowywany, a jego aktualna postać umożliwia wszechstronne badanie algorytmów wywłaszczania, obejmując pomiary czasów działania, generowanie histogramów, badanie cech kaskad wywłaszczania czy pomiary obciążenia sieci.

Symulator $msim$ i aplikacja wspomagająca $Vims$ okazały się bardzo przydatne w prowadzonych badaniach. Najważniejsze rezultaty uzyskane za ich pomocą zostały przedstawione i zinterpretowane w kolejnym rozdziale.
6. BADANIA I OMÓWIENIE WYNIKÓW

6.1. Opis warunków badań

Aby uzyskać miarodajne wyniki, konieczne było zaplanowanie warunków, w jakich badane będą algorytmy, a w szczególności wybrać topologie sieci oraz określić ilość i strukturę generowanego ruchu. Należało również zdecydować, które parametry algorytmów są najbardziej istotne i powinny wchodzić w skład kryteriów porównawczych.

Zdecydowano się na przeprowadzenie badań w oparciu o jedenaście różnych topologii, które podzielono na dwie serie. Do pierwszej wybrano sześć sieci o różnych wielkościach przy zachowaniu podobnej gęstości $d$, a do drugiej pozostałe pięć sieci, różniących się gęstością.

6.1.1. Topologie sieci

Wybór określonej topologii sieci może potencjalnie mieć wpływ na uzyskane wyniki. Dla osiągnięcia wysokiego poziomu wiarygodności wyników uzasadnione jest więc przeprowadzenie badań dla różnorodnych sieci. Podjęto przy tej okazji próbę określenia zależności wyników od parametrów sieci, o ile takie zależności istnieją.

Rozmiar sieci, rozumiany jako liczba węzłów w sieci, może mieć wpływ na wyniki badań poprzez różnicę wartość średniej długości drogi. Im droga jest dłuższa, tym więcej łączy może wymagać wywłaszczeń. Zatem im większa sieć, tym większe jest prawdopodobieństwo uzyskania znaczącej różnicy w wynikach w zależności od zasięgu algorytmu, tj. od tego, czy dany algorytm bierze pod uwagę dane tylko z pojedynczego łącza (lokalny), czy z całej domeny (globalny).

Gęstość sieci $d$ określa się przez średnią liczbę łącznych wychodzących od jednego węzła (wzór 3.1). Duża gęstość oznacza, że węzeł jest statystycznie lepiej skomunikowany z sąsiadami, a przez to ma potencjalnie większe możliwości wyboru drogi obejściowej. Może to mieć wpływ na prawdopodobieństwo wystąpienia wywłaszczenia lub przywrócenia wywłaszczonych ścieżek na alternatywnych drogach.

Starano się wybrać topologie możliwie bliższe tym, jakie spotykane są w sieciach operatorów telekomunikacyjnych. Źródłem informacji była biblioteka SNDlib [18], zgromadzona i udostępniona na serwerze sieciowym instytutu Zuse w Berlinie (Zuse-Institut Berlin, ZIB). Tabela 6.1 zawiera zestawienie parametrów sieci wybranych do badań, które dodatkowo przedstawiono graficznie na rys. 6.1 i 6.2.

Wybrane topologie zgodnie z założeniami podzielono na dwie serie. Do pierwszej (A) zaliczono sześć sieci o podobnej gęstości połączeń (od 2,93 do 3,60) ale różnej wielkości (od 12 do 50 węzłów). Do drugiej (B) wybrano pięć sieci o podobnej liczbie łącz (od 82 do 102) ale różnej gęstości połączeń (od 2,93 do 9,00). Tabela 6.2 zawiera zestawienie zaplanowanych scenariuszy badań. Dla wszystkich sieci ustalono średnią długość życia ścieżki na 1 godzinę.
Wartość ta może być dyskusyjna, jednak należy zauważyć, że z punktu widzenia badań nad algorytmami wywłaszczania istotna jest intensywność strumienia żądań tworzenia ścieżek. W praktyce więc nie tyle ważna jest konkretna wartość długości życia ścieżek, ile jej relacja do intensywności zgłoszeń, a tą również wyrażono w jednostkach na godzinę. Konkretne wartości długości życia oraz pasma ścieżek wybierano losowo z rozkładem wykładniczym.

Tabela 6.1. Parametry topologii wybranych do badań.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Polska</td>
<td>12</td>
<td>36</td>
<td>155</td>
<td>3,00</td>
<td>Polska</td>
</tr>
<tr>
<td>2</td>
<td>Atlanta</td>
<td>15</td>
<td>44</td>
<td>1000</td>
<td>2,93</td>
<td>Atlanta</td>
</tr>
<tr>
<td>3</td>
<td>Francja</td>
<td>25</td>
<td>90</td>
<td>2500</td>
<td>3,60</td>
<td>france</td>
</tr>
<tr>
<td>4</td>
<td>USA</td>
<td>26</td>
<td>84</td>
<td>64</td>
<td>3,23</td>
<td>janos-US</td>
</tr>
<tr>
<td>5</td>
<td>Europa I</td>
<td>37</td>
<td>114</td>
<td>7560</td>
<td>3,08</td>
<td>cost266</td>
</tr>
<tr>
<td>6</td>
<td>Niemcy I</td>
<td>50</td>
<td>176</td>
<td>40</td>
<td>3,52</td>
<td>germany50</td>
</tr>
<tr>
<td>7</td>
<td>Europa II</td>
<td>28</td>
<td>82</td>
<td>20</td>
<td>2,93</td>
<td>nobel-eu</td>
</tr>
<tr>
<td>8</td>
<td>Telecom Austria</td>
<td>24</td>
<td>102</td>
<td>504000</td>
<td>4,25</td>
<td>ta1</td>
</tr>
<tr>
<td>9</td>
<td>Nowy Jork</td>
<td>16</td>
<td>98</td>
<td>1000</td>
<td>6,13</td>
<td>newyork</td>
</tr>
<tr>
<td>10</td>
<td>Di-Yuan</td>
<td>11</td>
<td>84</td>
<td>1</td>
<td>7,64</td>
<td>di-yuan</td>
</tr>
<tr>
<td>11</td>
<td>Niemcy II</td>
<td>10</td>
<td>90</td>
<td>80000</td>
<td>9,00</td>
<td>dfn-bwin</td>
</tr>
</tbody>
</table>

6.1.3. Przedmiot badań

Dokonanie oceny algorytmów jest możliwe po wybraniu zbioru wielkości opisujących ich jakość z uwzględnieniem różnych definicji kosztów wywłaszczania (patrz rozdz. 3.4.1). O ile zbiór możliwych wielkości jest duży (patrz rozdz. 3.5.4), to w praktyce do oceny algorytmów używa się jedynie kilku parametrów. W niezależnych publikacjach [35,83] dokonuje się oceny w oparciu o liczbę wywłaszczeń i wywłaszczone pasmo (lokalne lub sieciowe). Za bardzo istotne należy uznać również czasy działania algorytmów. Opierając się dodatkowo na wynikach wcześniejszych analiz [50,51] zdecydowano, że do oceny algorytmów wystarczające są wyniki pomiarów pięciu następujących wielkości.
1. Średnia liczba wywłaszczeń: $\overline{M}$.
2. Średni indeks pasma straconego sieciowego: $\overline{b_{\text{NET}}}$ (wzór 6.1).
3. Średni indeks pasma straconego lokalnego: $\overline{b_{\text{LOC}}}$ (wzór 6.2).
4. Średnia metryka złożona: $\overline{Q}$.
5. Rozkład czasów działania algorytmów.

Trzy pierwsze parametry, tj. średnia liczba wywłaszczeń i średnie pasma stracone (sieciowe i lokalne), reprezentowane tu w postaci indeksów pasma, to najważniejsze parametry, charakteryzujące jakość algorytmów. Średnia liczba wywłaszczeń jest brana pod uwagę, gdy wymagana jest minimalizacja liczby usuwanych ścieżek, natomiast pasmo stracone charakteryzuje zdolność algorytmu do wybierania ścieżek najbardziej odpowiadających brakującemu pasmu. Czwarty parametr, średnia metryka złożona, umożliwia ujęcie w jednym parametrze liczby wywłaszczeń i ilości straconego pasma sieciowego, co znacznie ułatwia porównywanie algorytmów.

![Topologie o różnej wielkości wybrane do badań.](image)

Ostatnia wielkość, tj. rozkład czasów działania algorytmu, wskazuje na rzeczywisty czas potrzebny na wybór listy kandydatów do wywłaszczenia. Jest to ważny parametr decydujący o możliwości praktycznej realizacji danej metody i stanowiący istotne uzupełnienie teoretycznych rozważań przeprowadzonych w oparciu o złożoność obliczeniową (patrz rozdział 3.6). Porównanie ze sobą czasów daje informację o tym, które z algorytmów wymagają
mniejszych zasobów procesora, co przekłada się na możliwość obsługi większej liczby zgło- szeń w określonym czasie lub możliwość użycia tańszej platformy sprzętowej. Analiza cza- sów musi wykraczać poza pomiar wartości średniej, gdyż rozkład czasów działania dla poszczególnych algorytmów może znacznie się różnić.

![Rys. 6.2. Topologie o różnej gęstości wybrane do badań.](image)

Do celów prezentacji wyników wprowadzono pojęcia indeksów pasma lokalnego $b_{LOC}$ i sieciowego $b_{NET}$, zdefiniowane jako odwrotność straconego pasma, odpowiednio lokalnego i sieciowego:

$$b_{LOC} = \frac{1}{b_{loc}} = \frac{B_{net}}{B_{loc}},$$  \hspace{1cm} (6.1)$$

$$b_{NET} = \frac{1}{b_{net}} = \frac{B_{net}}{B_{net}}. \hspace{1cm} (6.2)$$
Taki sposób prezentacji wyników wziął się z powodów czysto praktycznych, gdyż tylko w ten sposób można było uzyskać akceptowalne wartości przedziałów ufności. Uzyskiwane wyniki samego pasma straconego były narażone na przeklamania spowodowane sporadycznymi bardzo dużymi wartościami pasm ścieżek (generowanymi losowo), co powodowało, że niejednokrotnie przedział ufności przekraczał wartość zmiennej, powodując nieprzydatność takich rezultatów. Zdefiniowany wcześniej indeks pasma nie ma tej wady. Zatem wykresy zawierające wartość indeksu pasma należy interpretować w ten sposób, że im większa wartość, tym lepiej, tzn. tym mniejsze pasmo jest stracone w wyniku wywłaszczania.

Tabela 6.2. Scenariusze badań.

<table>
<thead>
<tr>
<th>Seria badań</th>
<th>Topologia bazowa</th>
<th>Liczba priorytetów</th>
<th>Średnie pasmo ścieżki [Mbit/s]</th>
<th>Intensywność zgłoszeń [1/godz.]</th>
<th>Średnia liczba ścieżek na lączu</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Polska</td>
<td>3</td>
<td>6,5</td>
<td>130</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>Atlanta</td>
<td>3</td>
<td>40</td>
<td>170</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td>Francja</td>
<td>3</td>
<td>99</td>
<td>330</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td>USA</td>
<td>3</td>
<td>2,5</td>
<td>250</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td>Europa I</td>
<td>3</td>
<td>316</td>
<td>320</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>Niemcy I</td>
<td>3</td>
<td>1,5</td>
<td>500</td>
<td>26</td>
</tr>
<tr>
<td>B</td>
<td>Europa II</td>
<td>3</td>
<td>0,8</td>
<td>260</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td>Telecom Austria</td>
<td>3</td>
<td>20000</td>
<td>400</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td>Nowy Jork</td>
<td>3</td>
<td>40</td>
<td>450</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td>Di-Yuan</td>
<td>3</td>
<td>0,04</td>
<td>560</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td>Niemcy II</td>
<td>3</td>
<td>3200</td>
<td>750</td>
<td>25</td>
</tr>
</tbody>
</table>

6.2. Wyniki badań - seria A

Wyniki przeprowadzonych badań przyniosły szereg ciekawych wniosków. W niniejszym podrozdziale przedstawiono i skomentowano najbardziej interesujące rezultaty uzyskane w ramach badań topologii serii A, tzn. sieci różniących się wielkością przy zachowaniu podobnej gęstości. Uzyskane wyniki przedstawiono w postaci wykresów, na których umiesz-
czono również przedziały ufności. Te uzyskano metodą t-Studenta z poziomem ufności równym 0,95 przy użyciu procedur wbudowanych w program symulacyjny.

6.2.1. Liczba wywłaszczeń

Jedną z najważniejszych, jeśli nie najważniejszą mianą jakości algorytmów jest średnia liczba wywłaszczeń $M$. Parametr ten określa, ile średnio ścieżek zostaje usuniętych w efekcie wykonania procedury wywłaszczania dla przyjęcia pojedynczej ścieżki. Oczywiście ta wielkość powinna być jak najmniejsza, gdyż każde usunięcie ścieżki skutkuje chwilową lub trwałą utratą określonej liczby połączeń.

Na rys. 6.3 przedstawiono wykres liczby wywłaszczeń dla sieci z serii A, tj. dla sieci różniących się liczbą węzłów. Już pobieżna analiza pozwala zaobserwować następujące zależności:

- różne algorytmy i ich warianty prowadzą do bardzo różnych wyników oraz
- liczba wywłaszczeń rośnie wraz ze wzrostem rozmiaru sieci.

![Wykres liczby wywłaszczeń](image)

Rys. 6.3. Zależność średniej liczby wywłaszczeń $M$ od rozmiaru sieci (seria A).

Istnienie różnic w wynikach jest oczywiste, gdyż algorytmy działają w róży sposób i w róży sposób definiują funkcję kosztu wywłaszczania. Zaskakiwać może jednak rozpiętość wyników, np. dla topologii „Polska” w zakresie od około 1,5 do około 3,1. Zauważyć przy tym można, że najgorszy wynik generuje algorytm GarGop/BW, którego priorytetem nie jest minimalizacja liczby wywłaszczeń, ale wywłaszczonego pasma. Choć słabe rezultaty w tym zestawieniu nie dziwią, to jednak warto zdawać sobie sprawę, że użycie algorytmu GarGop/BW może powodować średnio ponad dwukrotny wzrost liczby wywłaszczeń.
w stosunku do większości pozostałych algorytmów.

Zaobserwowano także, że średnia liczba wywłaszczeń zwiększa się wraz ze wzrostem rozmiaru sieci. Taka zależność nie jest zaskakująca, gdyż w większych sieciach częściej występują długie ścieżki, a to prowadzi do większej liczby łączy, na których należy przeprowadzić wywłaszczenie. Potwierdzają to wyniki przedstawione na rys. 6.4, z których wynika, że średnia liczba łączy z wywłaszczaniem rzeczywiście rośnie wraz ze wzrostem liczby węzłów sieci, osiągając w tej serii badań wartości od około 1,2 do około 1,7. Można więc w tym miejscu podsumować tą część badań następującym wnioskiem.

**Wniosek 6.1.** Średnia liczba wywłaszczeń rośnie wraz ze wzrostem liczby węzłów w sieci, przy zachowaniu podobnej gęstości sieci.

![Rys. 6.4. Średnia liczba łączy wymagających wywłaszczenia w funkcji rozmiaru sieci (seria A). Przedziały ufności mają tu pomijalne wartości.](image)

**6.2.2. Wywłaszczone pasmo**

Drugim najczęściej ocenianym kryterium porównawczym algorytmów jest wywłaszczone pasmo. Parametr ten jest o tyle istotny, że większe wywłaszczone pasmo oznacza więcej straconych pakietów danych i mniejszy ruch obsłużony przez sieć.

Przedstawione wcześniej słabe wyniki algorytmu GarGop/BW w odniesieniu do liczby wywłaszczeń pozwalają przypuszczać, że znacznie lepiej wypadnie on w zestawieniach porównujących wywłaszczone pasmo. Wyniki przedstawione na rys. 6.5 i 6.6 przeczą jednak tym przewidywaniom. Co więcej, rezultaty osiągane przez ten algorytm znajdują się wśród
dwóch najgorszych wyników. To pozwala wysunąć wniosek, że w takiej postaci algorytm GarGop/BW nie sprawdza się w żadnym z dwóch istotnych kryteriów wyboru kandydatów, tj. liczbie wywłaszczeń i ilości wywłaszczonego pasma.

![Diagram](image_url)

Rys. 6.5. Średni indeks wywłaszczonego pasma sieciowego $b_{NET}$ w funkcji rozmiaru sieci (seria A).

W tym zestawieniu zaskakujący jest fakt, że większość algorytmów minimalizujących liczbę wywłaszczeń umożliwia równocześnie minimalizowanie wywłaszczonego pasma. To pozwala wyciągnąć ważny wniosek, że minimalizacja liczby wywłaszczeń i minimalizacja wywłaszczonego pasma nie muszą być w sprzeczności. Ma to wielkie znaczenie praktyczne dla problemu wyboru najlepszego algorytmu i jego wariantu, gdyż eliminuje problem decyzji, które z dwóch kryteriów są w sieci najważniejsze – liczba wywłaszczeń czy stracone pasmo. Przeprowadzone badania wskazują, że istnieją takie algorytmy, które dobrze sprawdzają się w obu kategoriach. Ocenie metod pod tym kątem służy metryka złożona $Q$, opisana w kolejnej części.

Wniosek 6.2. Dobre algorytmy wywłaszczania umożliwiają minimalizację jednocześnie liczby wywłaszczeń i ilości wywłaszczonego pasma.

6.2.3. Metryka złożona

Metryka złożona $Q$ umożliwia ocenę zdolności algorytmów do podejmowania decyzji zapewniających osiągnięcie jednocześnie małej liczby wywłaszczeń i małego straconego pa-
sma. Jest prosta w interpretacji, gdyż większa wartość oznacza lepszą jakość wywłaszczenia w obu tych obszarach.

Rys. 6.6. Średni indeks wywłaszczonego pasma lokalnego $b_{LOC}$ w funkcji rozmiaru sieci (seria A).

Wykres wyników badanych algorytmów w tym zakresie przedstawiono na rys. 6.7. Najlepsze wyniki osiąga tu algorytm globalny KNow, a nieco gorszy jest algorytm Pey, który zapewnia rezultaty optymalne pod względem liczby wywłaszczeń w skali lokalnej. Pozostałe algorytmy generują wyraźnie gorsze wyniki, co jest szczególnie widoczne dla algorytmów GarGop/BW i OliSco/RC, odbiegających znacząco od pozostałych wyników.

6.3. Wyniki badań - seria B

Seria badań B została dobrana tak, aby reprezentować rzeczywiste sieci o różnych gęstościach przy zachowaniu podobnej liczby łączy. Podobnie jak w serii A, poziom ufności wyników jest równy 0,95.

6.3.1. Liczba wywłaszczeń

O ile badania w serii A wykazały wyraźny wzrost liczby wywłaszczeń przy wzroście rozmiaru sieci, o tyle w serii B nie otrzymano jednoznacznjej zależności od gęstości sieci. Wyniki przedstawione na rys. 6.8 wskazują na to, że charakter zmienności jest zależny od typu algorytmu. Można jednak stwierdzić, że zwykle w sieciach o większej gęstości jakość wywłaszczenia jest lepsza, tzn. zmniejsza się liczba wywłaszczeń.
Rys. 6.7. Średnia wartość metryki złożonej $Q$ w funkcji rozmiaru sieci (seria A).

Rys. 6.8. Zależność średniej liczby wywłaszczeń $M$ od gęstości sieci (seria B).

Powodem takiego charakteru zmian jest to, że dla gęstszych topologii średnia długość drogi się zmniejsza, a przy tym zmniejsza się prawdopodobieństwo tego, że wywłaszczanie wykonuje się na więcej niż jednym łączu na drodze połączeniowej. To z kolei prowadzi do
powstania mniejszej liczby wywłaszczeń. Potwierdzeniem jest wykres z rys. 6.9 przedstawiający średnią liczbę łączy z wywłasczaniem, który wykazuje bardzo podobną zmienność. Należy jednak zaznaczyć, że wybrane topologie o większej gęstości charakteryzują się mniejszą liczbą węzłów, co w świetle wyników badań w serii A, sprzyja zmniejszeniu liczby wywłaszczeń.

Rys. 6.9. Średnia liczba łączy wymagająca wywłasczania dla topologii w serii B.

Analiza jakości poszczególnych algorytmów w zakresie minimalizacji liczby wywłaszczenych ścieżek potwierdza rezultaty uzyskane wcześniej. Najlepsze wyniki uzyskuje się poprzez zastosowanie algorytmów KNow i Pey, natomiast wyniki algorytmu GarGop/BW są również w tym zestawieniu raczej nie do zaakceptowania.

6.3.2. Wywłaszczone pasmo i metryka złożona

W odniesieniu do wywłaszczonego pasma wyniki przedstawione na rys. 6.10 i 6.11 wskazują, że przy większej gęstości jakość wywłasczania dla najlepszych metod znacznie się poprawia, tzn. zdecydowanie zmniejsza się stracone pasmo. Odpowiednio większa jest też wartość metryki złożonej (rys. 6.12), której zmienność w tym przypadku zależy znacznie bardziej od pasma niż od liczby wywłaszczeń. Najlepsze rezultaty po raz kolejny osiągają algorytmy KNow i Pey, przy bardzo zbliżonych wynikach.
Rys. 6.10. Średni indeks wywłaszczonego pasma sieciowego $b_{NET}$ w funkcji gęstości sieci (seria B).

Rys. 6.11. Średni indeks wywłaszczonego pasma lokalnego $b_{LOC}$ w funkcji gęstości sieci (seria B).
6.3.3. Przypadek topologii regularnej

Wyniki przedstawione w ramach serii B odnoszą się do rzeczywistych sieci, które wybrane zostały z ograniczonego zbioru. Z tego powodu topologie te różnią się nie tylko gęstością, ale również liczbą węzłów. To może nasuwać podejrzenia, że uzyskane wyniki nie są rezultatem zmiany jedynie gęstości, ale też funkcją rozmiaru sieci, zwłaszcza, że charakterystyki zmian liczby wywłaszczeń i wywłaszczonego pasma zgodzają się z charakterystyką otrzymaną dla sieci o różnej liczbie węzłów w serii A. Aby rozstrzygnąć tę wątpliwość, przeprowadzono badania topologii regularnej składającej się z ośmiu węzłów, w której zwiększano liczbę łączy międzywęzłowych, począwszy od pierścienia (d = 2), a kończywszy na połączeniu każdy-z-każdym (full mesh, d = 7). W ten sposób uzyskano zmienność jedynie gęstości i bardziej jednoznaczne wyniki.

Uzyskane rezultaty wskazują na pewną zmienność liczby wywłaszczeń (rys. 6.13) oraz zdecydowaną poprawę indeksu straconego pasma (rys. 6.14) i metryki złożonej (rys. 6.15) wraz ze wzrostem gęstości sieci regularnej. Porównując te wyniki z wynikami z serii B można zatem uznać, iż wykazywany tam spadek liczby wywłaszczeń jest jednak efektem raczej zmniejszania się rozmiaru sieci. Można też podsumować, że zmiana gęstości sieci bez zmiany liczby węzłów nie powoduje znaczących różnic w liczbie wywłaszczeń. Na tej podstawie sformułowano wniosek określający zależności jakości wywłaszczania od zmiany gęstości sieci.
Wniosek 6.3. Wzrost gęstości sieci przy zachowaniu stałego obciążenia łączy powoduje zmniejszenie straconego pasma i brak znaczących zmian liczby wywłaszczeń.

Rys. 6.13. Średnia liczba wywłaszczeń $M$ w funkcji gęstości $d$ sieci regularnej.

Rys. 6.14. Średni indeks straconego pasma sieciowego $b_{NET}$ w funkcji gęstości $d$ sieci regularnej.
6.4. Wyniki pozostałych badań

6.4.1. Czasy działania algorytmów

Pomiary czasów działania algorytmów, a ścisłej czasów działania procedur wyboru kandydatów do wywłaszczania są uzupełnieniem rozważań teoretycznych określających złożoność obliczeniową (rozdziały 3.6 i 4.2.1) i służą ich praktycznej weryfikacji. Pomiary przeprowadzono dla dwóch najbardziej różniących się rozmiarem sieci z serii A, tzn. topologii „Polska” (12 węzłów) i „Niemcy I” (50 węzłów).

Badania wykonano na komputerze osobistym typu laptop z procesorem Intel Core i7 640M 2,8 GHz w systemie Linux Debian 5 (jadro w wersji 2.6) osadzonym na platformie wirtualnej Oracle Sun VirtualBox. Do pomiarów czasów wykorzystano dwukrotne wywołanie funkcji gettimeofday(), zwracającej czas systemowy na wejściu i na wyjściu z procedury wyboru ścieżek. Wyniki przedstawiono w postaci histogramu zależności średniej liczby próbek n od czasu wykonania t. Wartości liczbowe czasów t podano w mikrosekundach czasu rzeczywistego, choć należy pamiętać, że konkretne wartości nie mają tu większego znaczenia. Zdecydowanie bardziej istotne jest porównanie ze sobą rozkładów czasów dla poszczególnych algorytmów.

Wyniki pomiarów dla topologii „Polska” przedstawiono na rys. 6.16. Obserwowane różnice nie są duże i dla wszystkich mierzonych algorytmów zdecydowana większość wywołań kończy się przed upływem 100µs, a najczęściej pomiędzy 30µs a 60µs, w zależności od ba-
danego algorytmu. Czasy powyżej 120µs zdarzają się rzadko i stanowią od 2% do 6% wyników. Porównując poszczególne algorytmy można stwierdzić, że statystycznie nieco więcej czasu na obliczenia wymaga algorytm KNow. Pozostałe algorytmy osiągają zbliżone rezultaty.

Ciekawych informacji dostarcza przedstawione na rys. 6.17 porównanie czasów dla dużej topologii „Niemcy I”, złożonej z 50 węzłów. Tutaj czasy działania algorytmów są znacznie większe. Jedynie poniżej 10% wywołań mieści się w czasach poniżej 100µs i mimo, że większość wywołań kończy działanie w czasie do 150µs, to co dziesiąte wywołanie kończy się dopiero po upływie 400µs lub 500µs, zależenie od algorytmu. Najszybszymi algorytmami są OliSco/BW oraz GarGop/BW, choć jak pokazano wcześniej te algorytmy nie należą do najlepszych, zarówno w zakresie liczby wywłaszczeń, jak i wywłaszczonego pasma. Algorytm KNow ponownie uzyskuje jedne z najdluższych statystycznie czasów, choć jednocześnie ta metoda ma jeden z najmniejszych odsetek czasów powyżej 600µs. Oznacza to, że algorytm KNow, jak też oba warianty GarGop, a więc algorytmy o zasięgu globalnym, są najbardziej przewidywalne pod kątem maksymalnego czasu działania.

Interesującym zjawiskiem, jakie można zaobserwować dla topologii „Niemcy I” dla części algorytmów jest pojawienie się lokalnego drugiego maksimum, które przypada na przedział czasu 200-250µs. Można przypuszczać, że wynika to ze stosunkowo częstej konieczności przeprowadzania wywłaszczeń na przynajmniej dwóch łączach. Zachowanie takie obserwujemy szczególnie wyraźnie dla algorytmów Pey, BlaMeL i OliSco/RC. Są to jednocześnie algorytmy, dla których obserwujemy stosunkowo dużo przypadków czasów działania powyżej 600µs (około 5%).

Warto przywołać wyniki testów jakościowych, tzn. liczby wywłaszczeń, wywłaszczonego pasma i metryki ogólnej i zestawić je z czasami działania. Porównanie takie pozwala wnioskować, że jakościowo najlepsze metody, tj. Pey i KNow, to również te, które wykazują najdłuższe casy wykonania. Spośród nich algorytm Pey częściej kończy wyznaczanie ścieżek szybciej, ale znacznie gorzej radzi sobie z przypadkami wymagającymi dłuższych obliczeń. Algorytm KNow należy z kolei do tych, które wykazują najszybsze „wygaszanie” długich czasów wykonania.

W załączniku B zaprezentowano omówione wyniki w postaci tabeli z rozkładem procentowym dla poszczególnych czasów.

Rys. 6.16. Histogramy czasów wykonania $t$ algorytmów dla topologii Polska (skala w mikrosekundach czasu rzeczywistego).
Rys. 6.17. Histogramy czasów wykonania $t$ algorytmów dla topologii Niemcy I (skala w mikrosekundach czasu rzeczywistego).
6.4.2. Wywłaszczenie lokalne i globalne

Podział na algorytmy globalne i lokalne wynika z sposobu analizy dostępnych ścieżek i wyboru spośród nich kandydatów do wywłaszczenia. W algorytmach lokalnych bierze się pod uwagę tylko dane dostępne lokalnie w węźle, a więc pasmo i priorytet ścieżki, natomiast algorytmy globalne operują na danych z całej domeny. Są dzięki temu w stanie dokonać lepszego wyboru szczególnie w przypadku, gdy wywłaszczenie jest efektem braku pasma na więcej niż jednym łączu na wybranej drodze.

Przedstawione dotąd wyniki najlepszych algorytmów wskazują na stosunkowo niewielką przewagę algorytmu globalnego KNow nad algorytmem lokalnym Pey. Jednocześnie zaobserwowano, iż tylko dla sieci obejmujących więcej niż 30 węzłów średnia liczba łącz z wywłaszczeniem jest większa niż 1,5. Zatem w przypadku mniejszych sieci zwykle wywłaszczenie występuje tylko na jednym łączu. To z kolei uniemożliwia wykrywanie potencjału algorytmów globalnych, gdyż w przypadku gdy wywłaszczenie odbywa się tylko na jednym łączu zarówno algorytmy globalne jak i lokalne w ogólności nie różnią się w działaniu. Jedyną przewagą tych pierwszych może być wtedy wybór ścieżek w oparciu o dodatkowe atrybuty niedostępne lokalnie, jak długośc ścieżek lub typy zajętych zasobów.

W trakcie badań prowadzono kontrolne śledzenie sposobu działania symulowanych algorytmów. Odkryto wówczas możliwość istnienia dodatkowego czynnika ograniczającego rezultaty osiągane przez algorytmy globalne w większych sieciach. Otóż według tych obserwacji, w nierегулярnych sieciach drogi niezależnych ścieżek rzadko się pokrywają, a zatem małe jest prawdopodobieństwo znalezienia pojedynczej ścieżki obejmującej wszystkie łącza wymagające wywłaszczenia, a to jest warunek na uzyskanie przewagi przez algorytmy globalne. Dla sprawdzenia tego zjawiska przeprowadzono dodatkowe pomiary określające, jaka ilość spośród dostępnych kandydatów pokrywa się z łączami wymaganymi wywłaszczenia. Pominięto przy tym nieistotne dla problemu przypadki wywłaszczeń na pojedynczym łączu. Uzyskane wyniki potwierdziły uzupełnione wcześniejszej zjawisko. Rzeczywiście, dla topologii „Polska” około 84% wszystkich ścieżek obejmowało tylko pojedyncze łące z wywłaszczeniem, a dla topologii „Niemcy I” było to około 78%. Zatem jedynie około 15-20% dostępnych ścieżek może przynieść korzyść z wywłaszczenia globalnego. Przytoczone wyniki zawarto w tabeli 6.3.

Mając na uwadze wymienione czynniki ograniczające jakość wyboru algorytmów globalnych, zdecydowano się na przeprowadzenie dodatkowych badań z uwzględnieniem tylko tych wywłaszczeń, w których występowęa przynajmniej dwa łącza wymagające wywłaszczenia. Cel tych badań była ocena zasadności używania metod globalnych wywłaszczenia, a jednocześnie skonfrontowanie tezy sformułowanej we wstępie do niniejszej pracy.

Na rys. 6.18, 6.19 i 6.20 przedstawiono odpowiednio wyniki średniej liczby wywłaszeń $M$, średniego indeksu straconego pasma sieciowego $b_{NET}$ oraz wartości średnie metryki złożonej $Q$ dla tylko tych przypadków, gdy na drodze połączeniowej występują przynajmniej dwa

Tabela 6.3. Średnia liczba ścieżek przechodzących przez daną liczbę łącz wymagających wywłaszczania. Uwzględniono tylko przypadki wywłaszczania na przynajmniej dwóch łączach. Poziom ufności 0,95.

<table>
<thead>
<tr>
<th>Liczba łącz</th>
<th>Polska</th>
<th>Niemcy I</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Liczba ścieżek</td>
<td>Przedział ufności</td>
</tr>
<tr>
<td>1</td>
<td>355,095</td>
<td>±76,5206</td>
</tr>
<tr>
<td>2</td>
<td>61,68</td>
<td>±12,6943</td>
</tr>
<tr>
<td>3</td>
<td>4,3125</td>
<td>±1,0098</td>
</tr>
<tr>
<td>4</td>
<td>0,5025</td>
<td>±0,366803</td>
</tr>
</tbody>
</table>

Rys. 6.18. Zależność średniej liczby wywłaszczeń $M$ od rozmiaru sieci, dla wywłaszczeń wymaganych na więcej niż jednym łączu na drodze połączeniowej ($z>1$). Pominięto algorytm GarGop/BW, który osiągnął dużo gorsze rezultaty ($M>6$).
Rys. 6.19. Zależność średniego indeksu straconego pasma sieciowego $b_{NET}$ od rozmiaru sieci, dla wywłaszczeń wymaganych na więcej niż jednym łączu na drodze połączeniowej ($z>1$).

Rys. 6.20. Zależność wartości średniej metryki złożonej $Q$ od rozmiaru sieci, dla wywłaszczeń wymaganych na więcej niż jednym łączu na drodze połączeniowej ($z>1$).
Wniosek 6.5a. Czynnikiem ograniczającym jakość algorytmów globalnych jest mała średnia liczba łączy wymagających wywłasczania na drodze wybranej dla nowej ścieżki.

Wniosek 6.5b. Dodatkowym czynnikiem ograniczającym jakość algorytmów globalnych są rzadkie przypadki pokrywania się tras ścieżek z łączami wymagającymi wywłasczania na więcej niż jednym łączu.

6.5. Wnioski

6.5.1. Porównanie jakości wywłaszczenia

Badania symulacyjne przeprowadzone dla rzeczywistych topologii sieci o różnych wielkościach i gęstościach wykazały znaczne różnice w jakości poszczególnych algorytmów wywłaszczenia. Wykazano przy tym, że w przypadku algorytmów istniejących w różnych wariantach niezbędny jest wybór właściwego z nich, gdyż niektóre z nich mogą prowadzić do rezultatów znacznie odbiegających od średnich wyników pozostałych algorytmów. Co więcej, jeden z badanych algorytmów globalnych, GarGop/BW, wykazał swoją nieprzydatność w badanych warunkach.

Analiza wyników symulacji prowadzi do wniosków, że im większa liczba węzłów sieci, tym większa średnia liczba wywłaszczeń (patrz wniosek 6.1), natomiast wzrost gęstości sieci wpływa na zmniejszenie straconego pasma (patrz wniosek 6.3).

Jeden z najciekawszych wniosków jakie wypływają z badań to odkrycie, że najlepsze algorytmy, takie jak KNow i Pey, cechują się doskonałą uniwersalnością (patrz wniosek 6.2). Osiągają one najlepsze wyniki w zakresie dwóch najważniejszych kryteriów, tj. liczby wywłaszczeń i wywłaszczonego pasma. Dzięki temu nie ma potrzeby dobierania algorytmu i jego parametrów pod kątem wymagań sieci, o ile celem jest minimalizacja liczby wywłaszczeń lub minimalizacja straconego pasma. To z kolei zdecydowanie poprawia komfort pracy z takimi algorytmami w praktycznych realizacjach.

Biorąc pod uwagę wszystkie przeprowadzone badania najlepszym algorytmem jest zaproponowany przez autora niniejszej pracy algorytm globalny KNow. Zarówno w zakresie minimalizacji liczby wywłaszczeń jak i minimalizacji straconego pasma osiągnięte rezultaty są bądź najlepsze bądź równe najlepszemu z pozostałych algorytmów, w granicach przedziału ufności. W porównaniu z najbardziej znany algorytmem OliSco, algorytm KNow zapewnia zdecydowanie lepsze rezultaty w zakresie liczby wywłaszczeń i straconego pasma. Wyniki pomiarów wskazują na porównywalny z innymi algorytmami czas działania i rzadziej niż w przypadku innych algorytmów przypadki długich czasów wykonania. Te fakty pozwolą stwierdzić, że postawiona we wstępie teza pracy została uwodniona.

Analizując pozostałe algorytmy, bardzo dobre wyniki oferuje metoda lokalna Pey, która
została opracowana zanim powstały sieci MPLS i została przez autora zaadoptowana do tej technologii. Jest to realizacja optymalnego algorytmu lokalnego w zakresie liczby wywłaszczeń, który dodatkowo dąży do minimalizacji wywłaszczonego pasma. W porównaniu z tą metodą inne algorytmy lokalne dedykowane dla sieci MPLS wykazały dużo gorszą jakość.

6.5.2. Metody globalne i lokalne

Zgodnie z oczekiwaniami, zastosowanie zaproponowanego algorytmu globalnego spowodowało poprawę jakości wywłaszczenia, rozumianej jako zmniejszenie średniej liczby wywłaszczeń i wywłaszczonego pasma, w stosunku do najlepszego z badanych algorytmów lokalnych. Jednocześnie skala tej poprawy, rzędu 10%, nie jest tak duża jak możliwa tego oczekiwać. Wykazano, że przyczyna takiego stanu rzeczy leży w następujących dwóch czynnikach, które ograniczają zysk osiągany przez zastosowanie algorytmów globalnych.

1. Większość wywłaszczeń odbywa się na pojedynczym łączu (porównaj rys. 6.4). To oznacza, że w większości wywłaszczeń nie ma możliwości skorzystania z potencjału jaki mają algorytmy globalne. Pokazano, że po uwzględnieniu tylko tych przypadków, gdy liczba łączu z wywłaszczeniem jest większa od jednego, wyniki nie tylko jednego, ale dwóch algorytmów globalnych, KNow i GarGop/RC, są lepsze od najlepszego algorytmu lokalnego Pey (patrz wniosek 6.5a).

2. Trasy niezależnych od siebie ścieżek rzadko podążają tą samą drogą. W efekcie osłabia się efekt zastosowania algorytmu globalnego, gdyż stosunkowo rzadko występuje korzystna dla wywłaszczenia globalnego sytuacja, w której wystarczające jest wywłaszczenie pojedynczej ścieżki obejmującej wszystkie łącza wymagające wywłaszczenia (patrz wniosek 6.5b).

Co warte podkreślenia, oba te czynniki wzajemnie wzmacniają negatywny wpływ na jakość algorytmów globalnych. Wynika to z faktu, że drugi czynnik objawia się szczególnie w dużych sieciach, w których nieco mniejsze znaczenie ma czynnik pierwszy, gdyż jak pokazano, im większa sieć, tym większa średnia liczba łączyc wymagających wywłaszczenia.

Pomimo tych ograniczeń pokazano, że zaproponowany algorytm KNow radzi sobie dobrze zarówno w małych jak też dużych sieciach i osiąga najlepsze wyniki we wszystkich badanych sieciach, niezależnie od wielkości i gęstości.

6.5.3. Czasy działania

Rzeczywiste czasy działania algorytmów wykazują dużą zależność od rozmiaru sieci. Im większa sieć, tym dłuższe czasy działania, niezależnie od wybranego algorytmu. Rozrzut jednostkowych czasów działania również zwiększa się w przypadku dużych sieci.

W odniesieniu do różnic w czasach działania dla poszczególnych algorytmów, są one stosunkowo niewielkie. Najdłuższe czasy działania wykazują te algorytmy, które osiągają najlepsze wyniki jakościowe. Zatem dłuższy czas działania jest ceną za osiągnięcie lepszych wyników jakościowych (patrz wniosek 6.4).
7. PODSUMOWANIE

Celem postawionym we wstępie do niniejszej pracy było porównanie wydajności dostępnych algorytmów wywłaszczania i porównanie ich z wynikami osiąganymi przez zaproponowany algorytm heurystyczny. Postawiono tezę o lepszej efektywności zaproponowanego algorytmu niż ta, jaką oferują inne najbardziej popularne algorytmy. Cele pomocnicze pracy obejmowały:

- dokonanie oceny dostępnych algorytmów wywłaszczania,
- wykonanie opisu zaproponowanego algorytmu,
- przygotowanie środowiska pomiarowego, oraz
- przeprowadzenie badań symulacyjnych i analizy wyników.

W ramach przeprowadzonych badań nad algorytmami wywłaszczania udało się osiągnąć wszystkie zamierzone cele, a w szczególności wykonano następujące zadania.

1. Udowodniono postawioną tezę w oparciu o wyniki badań symulacyjnych.

Wyniki badań wskazują na wysoką jakość opracowanego przez autora globalnego algorytmu wywłaszczania KNow. Pokazano, iż użycie zaproponowanego algorytmu pozwała uzyskać wyniki lepsze od uzyskiwanych z użyciem pozostałych najlepszych lokalnych i globalnych algorytmów heurystycznych. Badania oparto na dwóch najważniejszych kryteriach wyboru kandydatów, tj. średniej liczby wywłaszczeń i średniego wywłaszczonego pasma. Stwierdzono, że poprawa jakości w stosunku do najlepszego z pozostałych algorytmów w większości przypadków nie przekracza 10%. W porównaniu z najpopularniejszym algorytmem dedykowanym dla sieci MPLS, w wariantach OliSco/RC i OliSco/BW, zaproponowany algorytm zapewnia około 20% mniej wywłaszczeń i około 40% poprawę indeksu wywłaszczonego pasma.

Zaproponowany przez autora algorytm wydaje się być dobrą alternatywą dla innych algorytmów heurystycznych, gdyż spełnia cztery pożądane cechy dobrego algorytmu, tj. ma zasięg globalny, pozwala na zadanie kryterium wyboru kandydatów, posiada typową i akceptowalną złożoność obliczeniową oraz wykazuje się wysoką jakością wyboru kandydatów, zwykle lepszą od wszystkich pozostałych algorytmów poddanych badaniom.

Badania potwierdzające dobre wyniki algorytmu przeprowadzono z użyciem kilkunastu różnych topologii, włączając w to dostępne topologie sieci operatorów telekomunikacyjnych jak i topologie regularne. Wybór topologii dokonano w ten sposób, aby zweryfikować wpływ różnego rozmiaru oraz różnej gęstości sieci.

2. Samodzielnie wykonano narzędzia symulacyjne i poprawnie je zweryfikowano.

Wszystkie badania przeprowadzono używając programu msim, zaprojektowanego i zaimplementowanego od podstaw przez autora. Jest to otwarty na rozszerzenia symulator ukierunkowany na prowadzenie badań mechanizmów inżynierii ruchu w sieciach MPLS. Jego najważniejsze cechy są następujące:
budowa obiektowa,
- możliwość kompilacji w systemach Windows i Linux,
- możliwość prowadzenia symulacji na poziomie połączeń lub pakietów,
- automatyzacja serii symulacji oraz zintegrowane przetwarzanie wyników.

Program msim, rozwijany przez kilka lat, podlegał ewolucji i udoskonaleniom, dzięki którym stał się narzędziem bardzo użytecznym i realizującym wszystkie zamierzone rodzaje badań. W efekcie długotrwałego procesu testowania i poprawiania programu osiągnięto wysoki poziom zaufania do wyników generowanych przez program.

Wykonano także niezmiernie pomocną aplikację Vims, umożliwiającą łatwe tworzenie i modyfikowanie badanych sieci, automatyczne generowanie plików konfiguracyjnych dla symulatora, a także wczytywanie plików topologii dostępnych w innych formatach i generowanie plików graficznych z mapami topologii.

3. Odkryto interesujące zjawiska związane z wywłaszczaniem.

W efekcie analizy wyników badań symulacyjnych odkryto kilka zaskakujących lub nieoczywistych wniosków:
- można by oczekiwać, że dwa najważniejsze kryteria wyboru kandydatów, tj. minimalizacja liczby wywłaszczeń i minimalizacja ilości wywłaszczonego pasma są wzajemnie sprzeczne, tzn. jedno uzyskuje się kosztem drugiego; tymczasem wyniki badań wskazują na to, że najlepsze algorytmy pozwalały na uzyskanie dobrych wyników jednocześnie dla obu wymienionych kryteriów,
- przewaga najlepszej metody globalnej KNow, operującej na całej drodze połączeniowej jednocześnie, nad najlepszą metodą lokalną Pey, obejmującą działaniem tylko pojedyncze łącze, nie jest tak duża, jak można by tego oczekiwać; powodem jest to, że zwykle konieczność wywłaszczania występuje na pojedynczym łączu, a drogi różnich ścieżek rzadko pokrywają się na więcej niż jednym łączu,
- nie zweryfikowano pozytywnie argumentu o tym, że przewaga algorytmów lokalnych wywłaszczania bierze się ze zdecentralizowanego charakteru sieci IP/MPLS i trudności implementacyjnych algorytmów globalnych; odkryto, iż przyczyna atrakcyjności algorytmów lokalnych nie może być raczej wynikiem niedostatecznego zysku z dodatkowych nakładów poniesionych na implementację algorytmów globalnych; siła algorytmów globalnych poza lepszą jakością wywłaszczania leży w możliwości zaimplementowania jako metoda scentralizowana w oddzielnym urządzeniu w sieci, pozwalając na odcięcie procesorów ruterów i wykorzystanie zaawansowanych kryteriów wyboru ścieżek przeznaczonych do wywłaszczania.

4. Dokonano porównania kilku niezależnych algorytmów heurystycznych.

Praca jest jedynym znanyym zestawieniem czterech niezależnych algorytmów wywłaszczania, zawierającym opis zasady działania poszczególnych algorytmów i wyniki badań symulacyjnych. Najważniejsze osiągnięcia w tym zakresie to:
− ujednolicone ścienny algorytmów, obejmująca użyte symbole i kod źródłowy,
− opracowanie metodologii określania na potrzeby badań symulacyjnych warunków ruchowych w sieciach o dowolnej strukturze,
− dokonanie niezależnej analizy złożoności obliczeniowej,
− prezentacja wyników pomiarów czasów działania algorytmów w postaci histogramów,
− dostosowanie algorytmów GarGop i Pey do pracy w technologii MPLS,
− udoskonalenie algorytmu Pey w zakresie złożoności obliczeniowej,
− zaproponowanie przykładowego algorytmu optymalnego o zasięgu domeny.

Przeprowadzone badania wypełniają zatem lukę, jaką dotąd był brak ogólnie dostępnych wyników porównawczych dostępnych algorytmów wywłaszczania. Przedstawione w niniejszej pracy rezultaty umożliwiły weryfikację jakości algorytmów w realnych scenariuszach sieciowych. Wyniki uzyskane dla różnych topologii i warunków ruchowych wykazują powtarzalność i pozwalają na stawianie wniosków obarczonych niewielkim ryzykiem błędów.

Rozszerzenie badań na algorytmy, które są starsze niż technologia MPLS, umożliwiło skorzystanie z doświadczenia naukowców badających zjawisko wywłaszczania na długo przed pojawieniem się sieci MPLS. W szczególności jeden z algorytmów, Pey, opracowany dla sieci ATM, okazał się być algorytmem lepszym od wszystkich badanych algorytmów lokalnych dedykowanych dla sieci MPLS.

Można mieć nadzieję, że niniejsze opracowanie ułatwi prowadzenie niezależnych badań porównawczych i doprowadzi do poszerzenia listy publikacji w zakresie wywłaszczania w sieciach MPLS. Sprzyja temu ujednolicony przegląd istniejących algorytmów, zbiór definicji dotyczących wywłaszczania a także opracowany przez autora sposób określania warunków ruchowych w dowolnej sieci (patrz załącznik A).

Pomimo wielu ciekawych wniosków jakie przyniosły badania, temat wywłaszczania pozostawia jeszcze wiele otwartych zagadnień. Jednym z nich jest opisane w rozdziale 3.4.6 zjawisko kaskady wywłaszczeń i jego znaczenie w praktycznych realizacjach. Niezmiernie istotny jest związany z tym problem efektu wielokrotnego wywłaszczania ścieżek o niskim priorytecie, nawet w obrębie pojedynczej kaskady wywłaszczeń. Dalszych badań wymaga opracowanie metod pozwalających na ograniczenie takich niekorzystnych zjawisk.

Innym kierunkiem dalszych badań jest poszerzenie obszaru badań na technologie MPLS-TP i Generalized MPLS (GMPLS). W pierwszej z nich sieci MPLS ewoluują w kierunku operatorских sieci transportowych, natomiast w drugiej protokoły i algorytmy opracowane dla sieci MPLS są wykorzystywane do nowoczesnego sterowania ruchem w sieciach opartych na technologiach SDH [17], ASON/DWDM [79,80] lub OTN [87,89]. Jest to również długofalowy kierunek dalszego rozwoju programu symulacyjnego.
BIBLIOGRAFIA


IETF MPLS Working Group (Grupa Robocza MPLS), http://datatracker.ietf.org/wg/mpls/charter/


[74] Oliveira J. C. de, z prywatnej korespondencji (e-mail) z autorem.

SPIS SYMBOLI I OZNACZEŃ

A tablica pasm dostępnych na poszczególnych łączach przy uwzględnieniu wywłaszczania
A^{(e)} tablica pasm dostępnych na łącze e dla ścieżek o poszczególnych priorytetach
A_f tablica ilości wolnego pasma na poszczególnych łączach
a_f^{(e)} wolne pasmo na łącze e
A_{oc} tablica pasm zarezerwowanych dla ścieżek na poszczególnych łączach
a_{oc}^{(e)} suma pasm zarezerwowanych dla ścieżek na łącze e
a_s^{(e)} pasmo dostępne dla ścieżek o priorytecie utworzenia s na łącze e
B wywłaszczone pasmo
B_b globalny (dla wszystkich łącz) poziom rezerwacji pasma
b_b^{(e)} poziom rezerwacji pasma na łącze e
B_{hold} tablica pasm zarezerwowanych dla ścieżek o poszczególnych priorytetach
b_{LOC} indeks straconego pasma lokalnego (odwrotność względnego straconego pasma lokalnego)
B_n wywłaszczone pasmo sieciowe
b_{NET} indeks straconego pasma sieciowego (odwrotność względnego straconego pasma sieciowego)
B_o tablica pasm poszukiwanych na poszczególnych łączach
B_o^{(e)} pasmo poszukiwane na łącze e
b_p pasmo zarezerwowane dla ścieżki p
b_p^{(e)} pasmo zarezerwowane dla ścieżki p na łącze e
B_{pr} tablica pasm, jakie mogą być wywłaszczone przez ścieżki o poszczególnych priorytetach
B_r tablica wymaganych pasm na poszczególnych łączach
B_r^{(e)} wymagane pasmo na łącze e
B_{met} wymagane pasmo sieciowe (suma pasm wymaganych na kolejnych łączach)
B_{tot} sumaryczne pasmo jakie może być zarezerwowane w całej sieci
B_s tablica nadmiarowych (straconych) pasm na kolejnych łączach
B_s^{(e)} nadmiarowe (stracone) pasmo na łącze e
b_p^{(e)} względną nadmiarowe (stracone) pasmo na łącze e
B_{loc} nadmiarowe (stracone) pasmo lokalne (suma pasm nadmiarowych na łączach na drodze połączeniowej, na których brakuje wolnego pasma)
b_{loc} względną nadmiarowe (stracone) pasmo lokalne
B_{net} nadmiarowe (stracone) pasmo sieciowe (suma pasm nadmiarowych na wszystkich łączach należących do drogi połączeniowej)
b_{net} względne nadmiarowe (stracone) pasmo sieciowe
\begin{itemize}
\item $b_n$: średnie pasmo zarezerwowane dla ścieżek należących do \textit{n}-tej klasy
\item $C$: tablica pasm (przepływności) łączy
\item $c$: pasmo (przepływność) łącza (przy jednakowych pasmach łączy)
\item $c_e$: pasmo (przepływność) łącza $e$
\item $c_n$: średnie pasmo (przepływność) rezerwowane dla źródeł należących do \textit{n}-tej klasy
\item $C_{net}$: sumaryczna ilość pasma (przepływności) jaką może być obciążona sieć
\item $D^{(p)}$: zbiór ścieżek w kaskadzie wywłaszczeń spowodowanej przez ścieżkę $p$
\item $d$: gęstość sieci
\item $E$: zbiór łączy jednokierunkowych w badanej sieci
\item $e$: łącze
\item $e^{(n)}_p$: $n$-te łącze, na którym utworzona jest ścieżka $p$
\item $\hat{e}_e$: $n$-te łącze na drodze ścieżki, na którym brakuje wolnego pasma
\item $g^{(p)}$: długość kaskady wywłaszczeń spowodowanej przez ścieżkę $p$
\item $H^{(p)}$: liczba ścieżek w kaskadzie spowodowanej przez ścieżkę $p$
\item $h$: priorytet utrzymania ścieżki
\item $h_p$: priorytet utrzymania ścieżki $p$
\item $K$: zbiór ścieżek-potencjalnych kandydatów do wywłaszczenia na całej wybranej drodze
\item $k_{cr}$: liczba utworzonych ścieżek
\item $k_{crpr}$: liczba ścieżek, których utworzenie spowodowało wywłaszczenie
\item $K_e$: zbiór ścieżek-potencjalnych kandydatów do wywłaszczenia na łącze $e$
\item $k_{req}$: liczba żądań utworzenia ścieżek
\item $L$: liczba łączy jednokierunkowych w sieci
\item $l_p$: długość ścieżki $p$ (liczba łączy)
\item $M$: liczba wywłaszczeń (ścieżek przeznaczonych do wywłaszczenia)
\item $m$: metryka ścieżki w algorytmie OliSco
\item $m^{(p)}$: poziom wywłaszczenia ścieżki $p$ (w kaskadzie lub łańcuchu)
\item $N$: liczba węzłów sieci
\item $n$: liczba próbek (dot. histogramów czasów wykonania)
\item $P$: zbiór ścieżek utworzonych w sieci
\item $P_p$: cztery wielkości ($R_p$, $b_p$, $s_p$, $h_p$) opisujące ścieżkę $p$
\item $p_{pre}$: prawdopodobieństwo spowodowania wywłaszczenia przez ścieżkę
\item $p_{req}$: prawdopodobieństwo odrzucenia żądania utworzenia ścieżki
\item $Q$: metryka złożona (jakości algorytmów wywłaszczenia)
\item $q_n$: względny udział \textit{n}-tej klasy w generowanym ruchu
\item $\hat{R}$: lista łączy, na których wolne pasmo jest mniejsze od pasma danej ścieżki
\item $\bar{\tau}$: średnia długość drogi w sieci
\end{itemize}
lista łącz, na których utworzona jest ścieżka $p$

tablica bilansów pasma na łączach na wybranej drodze

priorytet utworzenia ścieżki

bilans pasma na łączu $e$

globalny bilans pasma

priorytet utworzenia ścieżki $p$

zmierzony czas wykonania algorytmu

średnia długość trwania połączeń należących do $n$-tej klasy

zbiór węzłów badanej sieci

węzeł

zbiór ścieżek-kandydatów do wywłaszczenia

$i$-ta ścieżka-kandydat do wywłaszczenia

zbiór ścieżek wywłaszczonych bezpośrednio przez ścieżkę $p$

liczba ścieżek w sieci

liczba łącz na drodze połączeniowej, na których wymagane jest wywłaszczenie

intensywność żądań utworzenia ścieżek należących do $n$-tej klasy

natężenie ruchu dla ścieżek należących do $n$-tej klasy
<table>
<thead>
<tr>
<th>Wykaz Akronimów</th>
<th>Opis</th>
</tr>
</thead>
<tbody>
<tr>
<td>AF</td>
<td>Assured Forwarding Per-Hop Behavior</td>
</tr>
<tr>
<td>ATM</td>
<td>Asynchronous Transfer Mode</td>
</tr>
<tr>
<td>ARIS</td>
<td>Aggregate Router-based IP Switching</td>
</tr>
<tr>
<td>ASON</td>
<td>Automatically Switched Optical Network</td>
</tr>
<tr>
<td>BGP-TE</td>
<td>Border Gateway Protocol with Traffic Engineering extensions</td>
</tr>
<tr>
<td>BW</td>
<td>Bandwidth</td>
</tr>
<tr>
<td>CoS</td>
<td>Class of Service</td>
</tr>
<tr>
<td>CR-LDP</td>
<td>Constraint-based Routing Label Distribution Protocol</td>
</tr>
<tr>
<td>CSPF</td>
<td>Constrained Shortest Path First</td>
</tr>
<tr>
<td>CSR</td>
<td>Cell Switching Router</td>
</tr>
<tr>
<td>DLCI</td>
<td>Data Link Connection Identifier</td>
</tr>
<tr>
<td>DWDM</td>
<td>Dense Wavelength-Division Multiplexing</td>
</tr>
<tr>
<td>ERO</td>
<td>Explicit Route Object</td>
</tr>
<tr>
<td>FEC</td>
<td>Forwarding Equivalence Class</td>
</tr>
<tr>
<td>FIFO</td>
<td>First In First Out</td>
</tr>
<tr>
<td>FR</td>
<td>Frame Relay</td>
</tr>
<tr>
<td>FTN</td>
<td>FEC-To-NHLFE; Forwarding Equivalence Class To Next Hop Label Forwarding Entry</td>
</tr>
<tr>
<td>GMPLS</td>
<td>Generalized Multiprotocol Label Switching</td>
</tr>
<tr>
<td>GNU</td>
<td>GNU's Not Unix</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>IPv4</td>
<td>Internet Protocol version 4</td>
</tr>
<tr>
<td>IPv6</td>
<td>Internet Protocol version 6</td>
</tr>
<tr>
<td>IETF</td>
<td>Internet Engineering Task Force</td>
</tr>
<tr>
<td>ILM</td>
<td>Incoming Label Map</td>
</tr>
<tr>
<td>IS-IS</td>
<td>Intermediate System To Intermediate System</td>
</tr>
<tr>
<td>IS-IS-TE</td>
<td>Intermediate System To Intermediate System with Traffic Engineering extensions</td>
</tr>
<tr>
<td>LDP</td>
<td>Label Distribution Protocol</td>
</tr>
<tr>
<td>LER</td>
<td>Label Edge Router</td>
</tr>
<tr>
<td>LSP</td>
<td>Label Switched Patch</td>
</tr>
<tr>
<td>LSR</td>
<td>Label Switching Router</td>
</tr>
<tr>
<td>MAM</td>
<td>Maximum Allocation (Bandwidth Constraints) Model</td>
</tr>
<tr>
<td>MPLS</td>
<td>Multiprotocol Label Switching</td>
</tr>
<tr>
<td>MPLS-TP</td>
<td>Multiprotocol Label Switching – Transport Profile</td>
</tr>
<tr>
<td>MTU</td>
<td>Maximum Transfer Unit</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>NHLFE</td>
<td>Next Hop Label Forwarding Entry</td>
</tr>
<tr>
<td>NP</td>
<td>Non-Polynomial</td>
</tr>
<tr>
<td>OSPF</td>
<td>Open Shortest Path First</td>
</tr>
<tr>
<td>OSPF-TE</td>
<td>Open Shortest Path First with Traffic Engineering extensions</td>
</tr>
<tr>
<td>OTN</td>
<td>Optical Transport Network</td>
</tr>
<tr>
<td>PCE</td>
<td>Path Computation Element</td>
</tr>
<tr>
<td>PPP</td>
<td>Point-to-Point Protocol</td>
</tr>
<tr>
<td>PWE3</td>
<td>Pseudo Wire Emulation Edge to Edge</td>
</tr>
<tr>
<td>QoS</td>
<td>Quality Of Service</td>
</tr>
<tr>
<td>RC</td>
<td>Relocation Count</td>
</tr>
<tr>
<td>RDM</td>
<td>Russian Dolls (Bandwidth Constraints) Model</td>
</tr>
<tr>
<td>RFC</td>
<td>Request For Comments</td>
</tr>
<tr>
<td>RIP</td>
<td>Routing Information Protocol</td>
</tr>
<tr>
<td>RRO</td>
<td>Record Route Object</td>
</tr>
<tr>
<td>RSVP</td>
<td>Resource Reservation Protocol</td>
</tr>
<tr>
<td>RSVP-TE</td>
<td>Resource Reservation Protocol with Traffic Engineering extensions</td>
</tr>
<tr>
<td>SDH</td>
<td>Synchronous Digital Hierarchy</td>
</tr>
<tr>
<td>TC</td>
<td>Traffic Class</td>
</tr>
<tr>
<td>TCP</td>
<td>Transmission Control Protocol</td>
</tr>
<tr>
<td>TDP</td>
<td>Tag Distribution Protocol</td>
</tr>
<tr>
<td>TE</td>
<td>Traffic Engineering</td>
</tr>
<tr>
<td>TTL</td>
<td>Time To Live</td>
</tr>
<tr>
<td>UDP</td>
<td>User Datagram Protocol</td>
</tr>
<tr>
<td>VCI</td>
<td>Virtual Channel Identifier</td>
</tr>
<tr>
<td>VPI</td>
<td>Virtual Path Identifier</td>
</tr>
<tr>
<td>WFQ</td>
<td>Weighted Fair Queuing</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Markup Language</td>
</tr>
<tr>
<td>ZIB</td>
<td>Zuse-Institut Berlin</td>
</tr>
</tbody>
</table>
ZAŁĄCZNIK A. METODA WYZNACZANIA ILOŚCI RUCHU W SIECI

Jedno z pytań, na jakie należało odpowiedzieć przy dobieraniu warunków badań było to, czy ilość ruchu wprowadzanego do sieci, która przekłada się na częstość występowania wywłaszczeń, ma wpływ na uzyskane wyniki. Jeśli tak jest, wówczas pojawia się problem istnienia dodatkowej zmiennej, jaką należy brać pod uwagę przy interpretacji wyników badań. To może utrudniać prowadzenie badań i stawiać pod znakiem zapytania ich wiarygodność.

Aby zweryfikować to zagadnienie na drodze eksperymentalnej przeprowadzono serię badań topologii „Polska”, w których badano, jak zmieniają się rezultaty przy zwiększaniu intensywności zgłoszeń, tzn. żądań tworzenia ścieżek. Na rys. A.1 i A.2 przedstawiono wyniki tych badań, zawierające odpowiednio średnią liczbę wywłaszczeń $M$ i średni indeks straconego pasma sieciowego $b_{NET}$. O ile stracone pasmo nie zmienia się znacząco dla najlepszych algorytmów, to liczba wywłaszczeń wykazuje dość istotne zmiany, rosnąc od około 1,3 dla 90 zgł./godz. do ok. 1,8 dla 240 zgł./godz.

![Rys. A.1. Topologia „Polska” - zależność średniej liczby wywłaszczeń $M$ od intensywności zgłoszeń $\lambda$.](image-url)

Należy zatem mieć świadomość, że rezultaty osiągane przez badane algorytmy rzeczywiście zależą od ilości ruchu, jaki jest generowany w sieci. W przypadku konkretnej topologii nie jest to duży problem, gdyż wszystkie algorytmy bada się w tych samych warunkach ruchowych. Większym problemem jest przeprowadzenie wiarygodnych badań porównawczych...
dla różnych topologii. Zaproponowane rozwiązanie to opracowanie uniwersalnej procedury generowania podobnych warunków ruchowych niezależnie od topologii.

Rys. A.2. Topologia „Polska” - zmiany wartości średniej indeksu pasma straconego sieciowego $b_{NET}$ od intensywności zgłoszeń $\lambda$.

Dodatkowym powodem badań nad ilością wprowadzanego do sieci ruchu była potrzeba uzyskania warunków zapewniających powstanie masowych wywłaszczeń bez potrzeby źmudnego prowadzenia badań testowych metodą prób i błędów. Dla osiągnięcia optymalnych warunków badań konieczne jest osiągnięcie wysokiego poziomu rezerwacji pasma na łączach i utrzymanie takiego stanu przez cały czas trwania symulacji. Celem jest spowodowanie zajęcia przynajmniej części łaczy na poziomie około 0,9. Zbyt małe obciążenie skutkuje brakiem dostatecznej liczby wywłaszczeń co generuje niewiarygodne wyniki, a z kolei wygnerowanie zbyt dużego zapotrzebowania na pasmo nie jest korzystne, gdyż niemal każde utworzenie ścieżki spowoduje powstanie wywłaszczeń. To jest sytuacja spoza obszaru normalnego funkcjonowania sieci, a raczej wynikająca z przekierowań ścieżek spowodowanych powstaniem uszkodzenia w sieci, i leży poza obszarem objętym badaniami w niniejszej pracy.

Dokładne określenie zapotrzebowania sieci jest w ogólności trudne, ale z analizy topologii można uzyskać pewne oszacowane od góry wartości ruchu, jakie należy generować, a następnie w razie potrzeby zmodyfikować je eksperymentalnie. Maksymalna, teoretyczna ilość pasma, jaką można zarezerwować w sieci odpowiada stanowi, w którym wszystkie łączy są w pełni zajęte. Prowadzi to do określenia dla danej sieci obciążenia maksymalnego $C_{net}$ (A.1).

$$C_{net} = \sum_{ee \in E} C_e . \quad (A.1)$$
Biorąc pod uwagę to, że ścieżka składa się z kilku łącz, można określić sumaryczną wielkość $B_{tot}$ zapotrzebowania na pasmo ścieżki, jakie należy wygenerować, aby całkowicie obciążyć sieć:

$$B_{tot} = \frac{C_{net}}{\bar{r}},$$  \hspace{1cm}  (A.2)

gdzie $\bar{r}$ jest średnią długością drogi w sieci. W przypadku, gdy wszystkie łącza mają identyczną przeporność $c$, otrzymujemy zależność (A.3).

$$B_{tot} = \frac{C_{net}}{\bar{r}} = \frac{|E|}{\bar{r}} \cdot c = \frac{L \cdot c}{\bar{r}}.$$  \hspace{1cm}  (A.3)

Pozostaje jeszcze określić średnią długość drogi, przy czym zależnie od rozkładu źródeł ruchu w sieci bierze się pod uwagę wszystkie węzły lub tylko węzły brzegowe. Aby wyznaczyć średnią długość drogi, należy wyznaczyć wszystkie pary różnych węzłów brzegowych, a następnie zsumować długości dróg pomiędzy węzłami w parze i podzielić przez liczbę par. Zakładamy tutaj, że wszystkie węzły są ze sobą połączone bezpośrednio lub pośrednio, tzn. pomiędzy dwoma dowolnymi węzłami można odnaleźć drogę połączeniową. Zatem średnią długość drogi określa się jako:

$$\bar{r} = \frac{\sum_{e,f \in V, e \neq f} \text{dist}(e,f)}{L(L-1)},$$  \hspace{1cm}  (A.4)

gdzie dist$(e,f)$ jest długością najkrótszej drogi (liczba łącz) pomiędzy parą węzłów $e$ i $f$. W praktyce średnia długość drogi dla określonej sieci była obliczana przez program wspomagający tworzenie symulowanych sieci, a do wzoru (A.4) była podstawiana uzyskana w ten sposób wartość liczbowa.

Mając dane całkowite pasmo $B_{tot}$ należy określić parametry źródeł należących do poszczególnych klas ruchu. Średnie pasmo $b_n$ zarezerwowane w dowolnym momencie przez wszystkie źródła należące do $n$-tej klasy powinno wynosić:

$$b_n = q_n B_{tot}, \quad q_n \in (0;1), \sum_n q_n = 1$$  \hspace{1cm}  (A.5)

gdzie $q_n$ to względny udział $n$-tej klasy w całkowitym ruchu generowanym w sieci. Parametry źródeł danej klasy, tj. rezerwowane pasmo $c_n$, intensywność zgłoszeń $\lambda_n$ i średni czas trwania połączeń $t_n$, należy dobrać według zależności (A.6).

$$c_n = \frac{b_n}{\rho_n} = \frac{b_n}{\lambda_n t_n}$$  \hspace{1cm}  (A.6)
Wartości w ten sposób otrzymane są teoretycznym limitem zakładającym idealnie równomierne obciążenie wszystkich łączy, które mogą być przeszacowane i spowodować więcej wywłaszczeń, niż się oczekuje. Jest to jednak prosta metoda, dzięki której można wyznaczyć wartość odniesienia, skorygowaną później eksperymentalnie. W praktyce uzyskane w ten sposób wartości były zwykle odpowiednie do badań algorytmów wywłaszczania, a użycie opisanej procedury istotnie skróciło czas potrzebny na przeprowadzenie badań.
ZAŁĄCZNIK B. ROZKŁADY CZASÓW WYKONANIA ALGORYTMÓW WYWŁASZCZANIA

W rozdziale 6.4.1 zamieszczono histogramy czasów wykonania badanych algorytmów wywłaszczania w środowisku symulacyjnym. Dla lepszej ilustracji przedstawiono tu rozkłady tych czasów w postaci tabeli, w których określono procentowy udział próbek należących do poszczególnych przedziałów.

Dodatkowym ułatwieniem w interpretacji wyników jest oznaczenie kolorowym tłem tych przedziałów, w których znalazło się najwięcej próbek. Kwalifikacji poszczególnych wartości do odpowiedniego koloru dokonano przy użyciu wartości progowych. Dla topologii „Polska”, wybrano cztery wartości, wynoszące 3%, 5%, 8% i 16%, a dla topologii „Niemcy I”, trzy progi 3%, 5% i 8%, przy czym przekroczenie wyższego progu odpowiada bardziej intensywnemu kolorowi.

Tabela B.1. Topologia „Polska” - procentowy udział próbek w poszczególnych zakresach czasów wykonania algorytmów (kolorami o rosnącej intensywności wyróżniono zakresy zawierające odpowiednio przynajmniej 3, 5, 8 i 16% próbek).

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1,66</td>
<td>1,25</td>
<td>1,64</td>
<td>1,23</td>
<td>1,51</td>
<td>1,25</td>
<td>1,33</td>
</tr>
<tr>
<td>10</td>
<td>0,25</td>
<td>0,30</td>
<td>0,21</td>
<td>0,37</td>
<td>0,24</td>
<td>0,27</td>
<td>0,23</td>
</tr>
<tr>
<td>20</td>
<td>0,88</td>
<td>6,27</td>
<td>2,13</td>
<td>1,23</td>
<td>5,76</td>
<td>1,06</td>
<td>1,06</td>
</tr>
<tr>
<td>30</td>
<td>16,03</td>
<td>25,91</td>
<td>18,25</td>
<td>19,17</td>
<td>22,94</td>
<td>11,04</td>
<td>10,30</td>
</tr>
<tr>
<td>40</td>
<td>23,19</td>
<td>18,43</td>
<td>20,52</td>
<td>20,50</td>
<td>23,57</td>
<td>19,26</td>
<td>18,94</td>
</tr>
<tr>
<td>50</td>
<td>23,37</td>
<td>16,63</td>
<td>23,50</td>
<td>22,93</td>
<td>18,38</td>
<td>26,13</td>
<td>17,52</td>
</tr>
<tr>
<td>60</td>
<td>14,78</td>
<td>11,30</td>
<td>12,40</td>
<td>14,02</td>
<td>9,82</td>
<td>14,99</td>
<td>15,94</td>
</tr>
<tr>
<td>70</td>
<td>8,80</td>
<td>6,65</td>
<td>6,50</td>
<td>7,10</td>
<td>5,69</td>
<td>8,50</td>
<td>12,73</td>
</tr>
<tr>
<td>80</td>
<td>3,97</td>
<td>4,57</td>
<td>3,47</td>
<td>4,10</td>
<td>3,82</td>
<td>4,20</td>
<td>8,20</td>
</tr>
<tr>
<td>90</td>
<td>2,19</td>
<td>2,45</td>
<td>3,33</td>
<td>3,16</td>
<td>2,60</td>
<td>4,17</td>
<td>5,33</td>
</tr>
<tr>
<td>100</td>
<td>1,48</td>
<td>1,71</td>
<td>2,82</td>
<td>1,80</td>
<td>1,98</td>
<td>2,81</td>
<td>2,98</td>
</tr>
<tr>
<td>110</td>
<td>1,09</td>
<td>1,38</td>
<td>1,73</td>
<td>1,75</td>
<td>1,16</td>
<td>1,99</td>
<td>1,87</td>
</tr>
<tr>
<td>120</td>
<td>2,31</td>
<td>3,15</td>
<td>3,49</td>
<td>2,64</td>
<td>2,55</td>
<td>4,33</td>
<td>3,57</td>
</tr>
</tbody>
</table>
Tabela B.2. Topologia „Niemcy I” - procentowy udział próbek w poszczególnych zakresach czasów wykonania algorytmów (kolorami o rosnącej intensywności wyróżniono zakresy zawierające odpowiednio przynajmniej 3, 5 i 8% próbek).

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0,63</td>
<td>0,89</td>
<td>0,61</td>
<td>0,55</td>
<td>0,71</td>
<td>0,54</td>
<td>0,61</td>
</tr>
<tr>
<td>25</td>
<td>0,34</td>
<td>0,39</td>
<td>0,27</td>
<td>0,31</td>
<td>0,31</td>
<td>0,26</td>
<td>0,30</td>
</tr>
<tr>
<td>50</td>
<td>0,40</td>
<td>0,46</td>
<td>0,42</td>
<td>0,35</td>
<td>0,45</td>
<td>0,36</td>
<td>0,42</td>
</tr>
<tr>
<td>75</td>
<td>1,22</td>
<td>3,63</td>
<td>1,71</td>
<td>0,93</td>
<td>10,90</td>
<td>0,90</td>
<td>0,69</td>
</tr>
<tr>
<td>100</td>
<td>21,23</td>
<td>33,68</td>
<td>20,42</td>
<td>19,80</td>
<td>19,53</td>
<td>15,95</td>
<td>15,30</td>
</tr>
<tr>
<td>125</td>
<td>16,32</td>
<td>14,73</td>
<td>8,52</td>
<td>10,91</td>
<td>6,27</td>
<td>8,00</td>
<td>18,99</td>
</tr>
<tr>
<td>150</td>
<td>7,22</td>
<td>5,97</td>
<td>4,17</td>
<td>3,99</td>
<td>5,89</td>
<td>4,14</td>
<td>8,00</td>
</tr>
<tr>
<td>175</td>
<td>5,94</td>
<td>4,72</td>
<td>4,76</td>
<td>4,21</td>
<td>8,96</td>
<td>4,89</td>
<td>5,76</td>
</tr>
<tr>
<td>200</td>
<td>5,83</td>
<td>4,71</td>
<td>8,12</td>
<td>6,80</td>
<td>9,29</td>
<td>7,77</td>
<td>5,76</td>
</tr>
<tr>
<td>225</td>
<td>6,44</td>
<td>4,73</td>
<td>8,34</td>
<td>8,66</td>
<td>5,71</td>
<td>8,54</td>
<td>5,84</td>
</tr>
<tr>
<td>250</td>
<td>6,44</td>
<td>4,76</td>
<td>6,28</td>
<td>6,41</td>
<td>4,48</td>
<td>6,68</td>
<td>5,87</td>
</tr>
<tr>
<td>275</td>
<td>6,16</td>
<td>4,36</td>
<td>5,17</td>
<td>5,16</td>
<td>3,65</td>
<td>5,63</td>
<td>5,61</td>
</tr>
<tr>
<td>300</td>
<td>5,15</td>
<td>3,69</td>
<td>4,38</td>
<td>4,61</td>
<td>3,56</td>
<td>5,10</td>
<td>5,56</td>
</tr>
<tr>
<td>325</td>
<td>4,26</td>
<td>2,82</td>
<td>3,66</td>
<td>4,21</td>
<td>3,22</td>
<td>4,03</td>
<td>5,00</td>
</tr>
<tr>
<td>350</td>
<td>3,43</td>
<td>2,28</td>
<td>2,80</td>
<td>3,08</td>
<td>3,04</td>
<td>3,29</td>
<td>4,10</td>
</tr>
<tr>
<td>375</td>
<td>2,51</td>
<td>1,71</td>
<td>2,49</td>
<td>2,67</td>
<td>2,39</td>
<td>2,88</td>
<td>3,24</td>
</tr>
<tr>
<td>400</td>
<td>1,73</td>
<td>1,35</td>
<td>2,69</td>
<td>2,43</td>
<td>1,73</td>
<td>2,74</td>
<td>2,42</td>
</tr>
<tr>
<td>425</td>
<td>1,18</td>
<td>0,98</td>
<td>2,33</td>
<td>2,23</td>
<td>1,33</td>
<td>2,67</td>
<td>1,82</td>
</tr>
<tr>
<td>450</td>
<td>0,82</td>
<td>0,78</td>
<td>2,01</td>
<td>1,97</td>
<td>1,30</td>
<td>2,28</td>
<td>1,24</td>
</tr>
<tr>
<td>475</td>
<td>0,59</td>
<td>0,55</td>
<td>1,45</td>
<td>1,67</td>
<td>1,23</td>
<td>1,87</td>
<td>0,98</td>
</tr>
<tr>
<td>500</td>
<td>0,39</td>
<td>0,45</td>
<td>1,26</td>
<td>1,32</td>
<td>1,06</td>
<td>1,52</td>
<td>0,59</td>
</tr>
<tr>
<td>525</td>
<td>0,28</td>
<td>0,37</td>
<td>1,22</td>
<td>1,08</td>
<td>0,91</td>
<td>1,31</td>
<td>0,45</td>
</tr>
<tr>
<td>550</td>
<td>0,23</td>
<td>0,30</td>
<td>1,09</td>
<td>1,03</td>
<td>0,65</td>
<td>1,23</td>
<td>0,30</td>
</tr>
<tr>
<td>575</td>
<td>0,21</td>
<td>0,25</td>
<td>0,97</td>
<td>0,83</td>
<td>0,57</td>
<td>1,11</td>
<td>0,24</td>
</tr>
<tr>
<td>600</td>
<td>1,06</td>
<td>1,43</td>
<td>4,88</td>
<td>4,78</td>
<td>2,86</td>
<td>6,31</td>
<td>0,92</td>
</tr>
</tbody>
</table>