ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1999
Seria: INFORMATYKA z. 36 Nr kol. 1414

Daniel ARENDT
Politechnika Łódzka, Samodzielny Zakład Sieci Komputerowych
Piotr S. SZCZEPANIAK
Politechnika Łódzka, Instytut Informatyki

ALGORYTM GENETYCZNY W WYTYCZANIU TRAS DATAGRAMÓW IP

Streszczenie. W pracy przedstawiono problem znajdowania optymalnej trasy dla datagramów IP. Opracowany algorytm wyznaczania tras działający w oparciu o algorytmy genetyczne uwzględnia zawartość pola Type Of Service, w którym podane są żądania jakości przesyłania. Podano reprezentację tras, sposób inicjowania populacji i jej ewolucję. Przedstawiono szczegóły implementacji algorytmu.

A GENETIC ALGORITHM FOR IP DATAGRAMS ROUTING

Summary. The paper presents the routing problem for the IP datagram. The IP protocol itself allows specification of requirements concerning a desired quality of delivery in the field Type Of Service. A genetic algorithm is proposed for finding the route based on the contents of this field. Encoding of the routes, initialization and evolution are described, and the implementation details are given.

1. Wstęp

Popularność protokołów TCP/IP ciągle rośnie. Różne są przyczyny takiego stanu rzeczy. Wydaje się, że główna przyczyna tkwi w istnieniu implementacji w wielu systemach i wykorzystywaniu różnorodnych technologii sieciowych jako warstwy dostępu do sieci. Wielu upatruje jej w otwartości oraz przejrzystości specyfikacji. Inni wiążą to zjawisko z powszechnością sieci Internet. Bez względu jednak na przyczyny popularności łatwo zauważyć, że rośnie liczba zastosowań protokołu TPC/IP i to nawet w dziedzinach, w których jego przydatność jest problematyczna. Można tu wymienić bardzo popularne ostatnio

przesyłanie głosu czy spotykane także zastosowania pomiarowo-kontrolne. W dziedzinach tych tradycyjnie ważne są takie cechy protokołów, jak transmisja izochroniczna, gwarantowany czas dostarczenia danych czy priorytety dla różnych typów danych. Protokoły TCP/IP zostały natomiast pomyślane właśnie pod kątem niewrażliwości na zróżnicowane czasy wędrówki datagramów IP. Znane nam zasady dostarczania danych w sieciach IP kładą nacisk na omijanie uszkodzonych fragmentów sieci, a nie na szybkość, niezawodność czy mały bądź stały czas zwłoki. Jednak budowa datagramu IP wskazuje, że jest możliwe nadanie priorytetu oraz określenie jakości usługi przenoszenia w sieci. Obecnie typowe rutery będą ignorować takie informacje. Jednak wykorzystywanie ich może dawać korzyści i uczynić łatwiejszą realizację transmisji głosu czy danych sterujących. W niniejszej pracy przedstawiony jest genetyczny sposób reprezentacji rozwiązania zadania wytyczania trasy w sieci komputerowej przy trójkryterialnej ocenie optymalności połączenia. Ocenie podlegać mogą: czas oczekiwania, szybkość łącza lub pewność przesyłania danych, przy czym priorytety ustalane są przez użytkownika, a zatem możliwe jest uwzględnienie jednego albo łącznie dwóch, czy trzech kryteriow. Mimo stosunkowo krótkiej historii algorytmy genetyczne [1,2,3] stały się uznanym narzędziem optymalizacji i znajdują liczne zastosowania. W literaturze znane są podobne próby, przy czym jako reprezentatywne i świeżej daty warte są odnotowania dwie prace. W pierwszej [4] zastosowano algorytm genetyczny do projektowania sieci spełniającej kryterium niezawodności, a w drugiej [5] – do rozwiązania problemu wytyczania optymalnych tras dla pewnego zadania z wieloma ruterami docelowymi. Obie prace zawierają dość liczne wykazy innej literatury.

Praca składa się z dwóch głównych części. W pierwszej opisano techniczną stronę zadania, a w drugiej zaproponowano i przedstawiono rozwiązanie oparte na algorytmie genetycznym.

2. Problem rutowania datagramów o określonym typie serwisu

Postępująca integracja przekazywania głosu i obrazów w sieciach przesyłu danych powoduje zmiany w charakterze ruchu w sieciach IP. Dawny nieregularny, nieodporny na straty danych, lecz akceptujący zmienne, czasami znaczne opóźnienia ruch w sieciach zastępowany jest równomiernym odpornym na częściową utratę danych, ale wymagającym oznaczonych czasów opóźnień. Technologia, pozwalająca pogodzić różne, częściowo sprzeczne wymagania w jednym protokole, nazywa się Quality Of Service (QoS). Idea ta nie jest nowa i jest dostępna choćby w technologii ATM. Atrakcyjność protokołu TCP/IP i powszechność standardu 802.3 w sieciach lokalnych zmusza do wprowadzenia QoS także do tych rozwiązań. Same protokoły IP i TCP nie pozwalają w zasadzie implementować usług

QoS [6]. Jednak istnieje zapomniane pole w nagłówku datagramu IP stanowiące namiastkę takiej usługi. Zgodnie z RFC 791 wśród sześćdziesięciu czterech bajtów nagłówka drugi bajt nazwany Typ Serwisu ma postać jak na rys. 1.

0	1	2	3	4	5	6	7
Priorytet			O	T	R		

Rys. 1. Pole Typ Serwisu w datagramie IP
Fig. 1. Type of Service field in IP datagram

Bity 0-2 należy traktować jako cyfrę ósemkową przedstawiająca priorytet datagramu.

Bity 3, 4 i 5 są dwustanowymi znacznikami reprezentującymi pożądany sposób przekazywania datagramu. Oznaczają one:

O - Bit 3 – potrzebne małe opóźnienie,

T - Bit 4 – potrzebne przekazywanie łączem o dużej przepustowości,

R - Bit 3 – potrzebna duża niezawodność.

Obecnie stosowane rutery ignorują pole priorytet i podobnie pojedyncze bity oznaczające pożądany sposób dostarczenia. Zwykle do wyboru drogi stosowana jest zasada określania odległości jako liczby ruterów, przez które wiedzie trasa od źródła do miejsca przeznaczenia. Przy istnieniu wielu dróg wybierana jest najkrótsza. Nie znaczy to jednak, że w sieci prywatnej nie można skutecznie zastosować trasowania opartego na analizie zawartości pól O T R na rys. 1. Można wyobrazić sobie również dołączenie takiej sieci do sieci stosującej klasyczną miarę odległości. Jednak wtedy ruter czy rutery łączące powinny dokonywać standaryzowania wartości tych pól przy przekazywaniu datagramów z otoczenia zewnętrznego.

3. Genetyczny algorytm wytyczania tras

3.1. Założenia

Opracowany algorytm wyznaczania tras, działający w oparciu o algorytmy genetyczne, uwzględnia zawartość pola TOS (Type Of Service). Oznacza to, że algorytm ten ma większe możliwości niż jego poprzednicy. Przedstawiona aplikacja nie jest protokołem, a jedynie symulacją działania protokołu. Dlatego nie można jej bezpośrednio włączyć do zestawu protokołów TCP/IP. Przy budowie modelu sieci ruterów przyjęto następujące założenia:

- Każdy ruter zna topologię całej sieci i przechowuje ją w tablicy tras.
- Każdy ruter ma przyporządkowaną unikalną liczbę całkowitą będącą jego jednoznacznym identyfikatorem w sieci.
- Pole w tablicy tras ma następującą postać:
  - pole TOS (zawiera wartość liczbową dla każdego elementu składowego charakteryzując parametry łącza),
  - numer rutera źródła, czyli początku łącza,
  - numer rutera celu - końca łącza.
- Numer rutera źródła i numer rutera celu identyfikują łącze, a nie wyznaczają kierunku połączenia i można je zamieniać miejscami w obrębie tego samego pola w tablicy tras.
- Pola w tablicy tras (łącze) identyfikują połączenia jedynie bezpośrednio sąsiadujących ruterów.
- Każde łącze w tablicy tras występuje tylko jeden raz (nie ma w tablicy dwóch identycznych pól).
- Każde łącze (ścieżka) ma wszystkie trzy wartości pola TOS.
- Wszystkie rutery mają dostęp do tej samej tablicy tras zarówno fizycznej, jak i logicznej. W rzeczywistości każdy ruter posiada własną fizyczną tablicę tras, która jest identyczna dla każdego rutera.

3.2. Kodowanie, budowa genu, chromosomy

Każdy ruter w topologii ma przyporządkowany numer identyfikujący go w sposób jednoznaczny. Połączenie między ruterami jest identyfikowane przez numer rutera początku i numer rutera końca. Z punktu widzenia algorytmu genetycznego połączenia, czyli pola w tablicy tras, są traktowane jako geny. Każdy gen (rys.2) zawiera pole TOS oraz numery dwóch ruterów. Dla potrzeb tej implementacji przyjęto, że każde z podpól pola TOS ma losowo przydzieloną cyfrę z zakresu <0,2>.

Numer rutera początku połączenia	O	T	R	Numer rutera końca połączenia

Rys. 2. Ideowa reprezentacja genu
Fig. 2. Representation of a single gene

Numer rutera początku i numer rutera końca połączenia nie wskazują kierunku przesyłania datagramów między ruterami; jest to jedynie kwestią umowy. W obrębie określonego genu można zatem zamieniać je dowolnie. Pole TOS w programie zaimplementowane jest jako

klasa *Gen*, zatem powyższy rysunek nie przedstawia faktycznych pól występujących w tej klasie.

Z punktu widzenia algorytmu genetycznego pola w tablicy tras są traktowane jako geny. Żądanie wyznaczenia trasy dla datagramu składa się z numerów dwóch ruterów: numeru rutera źródła i numeru rutera celu. Tworzenie populacji bazowej polega na budowie chromosomów w oparciu o probabilistyczne metody znajdowania możliwego połączenia. Chromosomy można wyobrazić sobie jako ciąg par liczb całkowitych niosących *bagaż genetyczny*, w którym tylko dwie liczby pierwsza (numer rutera źródła) i ostatnia (numer rutera celu) nie mają pary.

bagaż genetyczny	0	1	2	0	1	2
chromosom	3-5	5-2	2-6	6-8	8-9	9-3

Chromosomy mogą mieć różne długości. Długość chromosomu zależy od liczby genów będących kolejnymi łączami na trasie przemieszczania się pakietu. Niekiedy żądanie znalezienia połączenia między określonymi ruterami jest niewykonalne ze względu na faktyczny brak takiego połączenia. W rzeczywistości stan taki odzwierciedla uszkodzenie łącza. Rutery aktywnie testują połączenia ze swymi sąsiadami i rozsyłają wyniki do sąsiadów, zatem topologia widziana przez ruter w istocie podlega dynamicznym zmianom. Zabezpieczenie przed taką sytuacją polega na deterministycznym, choć uzależnionym od liczby ruterów w sieci, określeniu ilości prób tworzenia chromosomów. Po przekroczeniu tej liczby uznajemy, że żądanie połączenia nie jest możliwe do realizacji. Każdy ruter może występować tylko jednokrotnie w określonym chromosomie z populacji. Nie ma powodu, aby datagram był przesyłany dwa lub więcej razy przez ten sam ruter. Ponadto istnienie takiej możliwości mogłoby teoretycznie spowodować niepotrzebne obciążanie sieci i krążenie datagramów nawet w nieskończoność (w rzeczywistości jest to niemożliwe, ponieważ TCP/IP ma mechanizmy chroniące przed takimi pomyłkami - skończony czas życia datagramu w sieci). Zbiór wszystkich genów dla określonej topologii nie tworzy przestrzeni ciągłej. Przestrzeń ciągła pojawi się wtedy tylko, gdy rutery są połączone każdy z każdym, co nie ma znaczenia praktycznego. Dlatego zastosowanie algorytmu genetycznego jest niepełne. Chromosomy nie mogą ulegać mutacji, ponieważ zmutowane geny nie należałyby do dziedziny genów.

3.3. Przystosowanie

Ocena przystosowania chromosomów w populacji polega na minimalizacji funkcji celu, która jest sumą *bagażu genetycznego* – wartości wybranych elementów składowych pola TOS dla poszczególnych łączy.

Funkcja przystosowania optymalizuje sumę ustawionych przez użytkownika elementów składowych pola TOS wszystkich genów danego chromosomu. Na przykład, dla ustawionych znaczników O i P funkcja przystosowania będzie wyliczana zgodnie ze wzorem:

$$Przystosowanie = \sum_{i}^{n} O_i P_i$$

gdzie: n – liczba genów w chromosomie, O_i oraz P_i wartości podpól odpowiednio O i P dla kolejnych genów chromosomu.

Jak wcześniej wspomniano, każdy gen zbudowany jest z trzech pól: numer rutera początku łącza, numer rutera końca łącza i pole TOS. Do operacji tworzenia chromosomów i operacji krzyżowania wykorzystujemy numery ruterów źródłowy i docelowy. Natomiast do tworzenia funkcji celu - jedynie wartości pola TOS, które stanowią *bagaż genetyczny*. Minimalizację można przeprowadzać ze względu na dowolną liczbę elementów składowych z pola TOS. Dzięki temu dla datagramu może zostać wybrana optymalna trasa, np. ze względu na dwa kryteria: przesyłanie szybkimi łączami i duża pewność przesyłanych danych. Inne kombinacje żądań też są możliwe, lecz nie zawsze mają sens. Gdy suma podpól pola TOS jest taka sama dla dwóch lub większej liczby chromosomów, wybierany jest chromosom krótszy. Takie uzależnienie przystosowania od długości chromosomu powoduje, że najważniejszym kryterium pozostaje pole TOS. Długość chromosomu, czyli liczba ruterów do pokonania, która jest typowym wskaźnikiem wyboru trasy we współcześnie działających ruterach, ma znaczenie drugorzędne.

3.4. Algorytm genetyczny, krzyżowanie

Algorytm genetyczny działa w oparciu o standardowy schemat [1].

Liczność populacji dobrana została eksperymentalnie i jest równa sumie połowy ruterów i liczby dwa. Zwiększenie liczby chromosomów powyżej tej wartości nie powodowało wzrostu skuteczności algorytmu genetycznego, a tylko zwiększało czas rozwiązania problemu. Liczność populacji można opcjonalnie zmniejszyć, jednak dwa chromosomy stanowią oczywiste minimum. Selekcja chromosomów następuje metodą ruletki. Krzyżowanie dowolnych chromosomów *A*, *B* przebiega w oparciu o niżej przedstawiony algorytm:

1. Sprawdzenie, czy chromosom *A* zbudowany jest z dwóch lub mniej genów. Jeżeli tak jest, to algorytm kończy swoje działanie.
2. Wygenerowanie numeru genu licząc od początku chromosomu *A* (oznaczmy numer genu przez *n*).
3. Wyłuskanie numeru rutera z genu *A*, który jest wspólny dla genu bieżącego i następnego.

4. Przeszukanie chromosomu *B* w celu znalezienia genu, w którym numer jednego z ruterów jest równy numerowi rutera znalezionego powyżej (załóżmy, że znaleziono go na pozycji *m*).
5. Jeżeli powyższe poszukiwania zakończą się porażką, to algorytm kończy swoje działanie.
6. W przeciwnym wypadku utworzone zostają dwa nowe chromosomy *C* i *D*, które powstają w następujący sposób:
   - Chromosom *C* jest połączeniem *n* pierwszych genów z chromosomu *A* i *(rb – m)* genów z chromosomu *B*, gdzie *rb* jest liczbą genów w chromosomie *B*.
   - Chromosom *D* jest połączeniem *m* pierwszych genów z chromosomu *B* i *(ra – n)* genów z chromosomu *A*, gdzie *ra* jest liczbą genów w chromosomie *A*.

Rozpatrzmy następujący przykład. Niech będą dane chromosomy A:

bagaż genetyczny	0	1	2	0	1	2
chromosom	3-5	5-2	2-6	6-8	8-9	9-4

oraz chromosom *B*:

bagaż genetyczny	0	1	2	0	1
chromosom	3-5	5-1	1-7	7-6	6-4

Linią pionową zaznaczono miejsce krzyżowania chromosomów. W wyniku krzyżowania powstaną dwa nowe chromosomy. Chromosom *C* będzie miał postać:

bagaż genetyczny	0	1	2	1
chromosom	3-5	5-2	2-6	6-4

natomiast chromosom *D* będzie wyglądał następująco:

bagaż genetyczny	0	1	2	0	0	1	2
chromosom	3-5	5-1	1-7	7-6	6-8	8-9	9-4

Przestrzeń połączeń jest nieciągła; można znaleźć dwa rutery, które nie mają ze sobą połączenia. Rodzi to dwie konsekwencje:

1. Krzyżowanie zachodzi zawsze, jeśli jest to możliwe. Nie ma żadnych ograniczeń z punktu widzenia algorytmu genetycznego. Ponieważ prawdopodobieństwo

krzyżowania przy przyjętym kodowaniu jest niewielkie, więc nie ma potrzeby dodatkowo go zmniejszać.

2. Mutacja nie ma prawa bytu, ponieważ w wyniku mutacji mógłby powstać chromosom, który reprezentowałby połączenie nie istniejące „fizycznie" w topologii. Można dokonywać jedynie mutacji przy nałożonych ograniczeniach topologii sieci, co czyni dyskusyjną jej przydatność.

3.5. Wybrane szczegóły implementacji

Tworzenie genu

1. Generowanie wartości podpól pola TOS w oparciu o generator liczb pseudolosowych.
2. Generowanie numeru rutera początku i numeru rutera końca (różnego od numeru rutera początku) łącza w oparciu o generator liczb pseudolosowych.
3. Sprawdzenie, czy wygenerowane połączenie znajduje się już w tablicy tras. Jeżeli tak jest, to algorytm przechodzi do punktu 2.
4. Sprawdzenie, czy wygenerowano zadaną przez użytkownika liczbę połączeń. Jeśli tak, to algorytm kończy działanie.

Budowa chromosomu

Budowa chromosomu przebiega w oparciu o niżej przedstawiony algorytm:

1. Przeszukiwanie tablicy tras w celu znalezienia wszystkich genów, w których jeden z ruterów jest ruterem końca budowanego chromosomu, lub jeżeli jest to początek budowy chromosomu - rutera początku połączenia (jeżeli nie znaleziono żadnego, to znaczy, że ruter nie ma „fizycznego" połączenia z resztą sieci – nie można zbudować chromosomu i algorytm kończy działanie).
2. Wybranie losowo jednego genu z puli znalezionej powyżej i wstawienie go na początek budowanego chromosomu.
3. Wyznaczenie i sprawdzenie, czy drugi ruter w wybranym powyżej genie nie jest ruterem końca połączenia. Jeśli tak jest, to budowa chromosomu kończy się pomyślnie i algorytm kończy swoje działanie.
4. Szukanie następnego genu do chromosomu – przeszukując tablicę w celu znalezienia wszystkich tych genów, w których jeden z ruterów jest taki sam jak ruter wyznaczony powyżej.
5. Jeżeli nie znaleziono żadnego takiego genu, to znaczy, że ruter nie ma połączenia z resztą sieci. Usuwamy poprzedni gen z chromosomu i przechodzimy do kroku 2. Czynność taka może być powtarzana zadaną liczbę razy lub do osiągnięcia pierwszego genu dla bieżącego chromosomu; wówczas budowa chromosomu jest anulowana,

próbę uważa się za nieważną i algorytm przechodzi do punktu 1. Jeżeli ilość nieudanych prób tworzenia chromosomów dla danej populacji przekroczy inną zadaną liczbę prób, wówczas zakłada się, że zadane połączenie między ruterami nie istnieje i algorytm kończy swoje działanie.

6. Sprawdzenie, czy wyznaczony ruter nie występuje w każdym genie w tworzonym chromosomie (nie ma sensu, aby trasa przebiegała dwa lub więcej razy przez ten sam ruter), jeśli tak jest, to przejdź do punktu 2.
7. Wybranie losowo jednego genu z puli znalezionej powyżej i wstawienie go na koniec budowanego chromosomu i przejście do punktu 3.

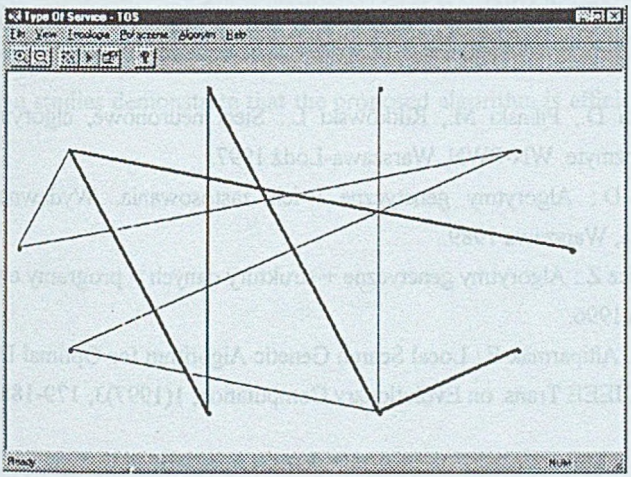
Ograniczenie liczby prób tworzenia chromosomów dotyczy tylko pierwszego chromosomu w populacji. Dla polepszenia skuteczności algorytmu ruter początku i ruter końca połączenia są cyklicznie wzajemnie zamieniane przy budowie każdego nowego chromosomu.

Symulacja

Aplikacja TOS jest programem symulującym trasowanie z uwzględnieniem próśb o rodzaj przesyłania. Tablica tras fizycznie w pamięci komputera znajduje się tylko raz, a każdy chromosom został zaimplementowany jako lista dwukierunkowa. Pozwoliło to zmniejszyć rozmiar wymaganej przez program pamięci.

Aplikacja ta posiada możliwość generowania topologii ruterów oraz możliwość podjęcia próby znalezienia połączenia pomiędzy dwoma dowolnie wybranymi ruterami – rys.3.

Możliwe jest wygenerowanie i wizualizacja istniejącej topologii oraz uzyskanie informacji o parametrach tras. Warto tu wspomnieć o zależności między liczbą ruterów a liczbą połączeń. Założono arbitralnie, że liczba połączeń nie może być mniejsza niż n-1 i nie może być większa niż liczba wszystkich przekątnych wielokąta wypukłego zbudowanego na n ruterach.



Rys.3. Przykład zestawionego połączenia
Fig. 3. The sample connection

Liczba chromosomów w populacji domyślnie jest ustawiona na maksymalną wartość odpowiadającą wygenerowanej topologii nie mniejszą jednak niż 3. Liczba prób podejmowanych przy budowie jednego chromosomu nie może być mniejsza niż 5.

4. Podsumowanie

Przeprowadzono eksperymenty dla różnych topologii sieci o liczbie ruterów od 2 do 80. Jeśli pominąć zagadnienia nakładu obliczeń, to najistotniejsza z punktu widzenia skuteczności metody jest obserwacja, że niezależnie od inicjowania algorytmu uzyskiwano wyniki powtarzalne - znajdowana była ta sama trasa optymalna. Obiecujące wyniki algorytmu wydają się być trudne do realizacji praktycznej z powodu założenia o znajomości całej topologii sieci przez wszystkie rutery. Inny problem to zakłócenia wprowadzane przez ewentualne rutery klasyczne ignorujące zawartość znaczników TOS. Wydaje się, że podstawowy cel, czyli przesyłanie danych z różną jakością w sieciach TCP/IP, będzie w najbliższym czasie rozwiązywany na różne sposoby i jedynie w stosunkowo małych sieciach będzie możliwy do realizacji jedynie w oparciu o „zapomniane" flagi w nagłówku datagramu IP. W większych sieciach będzie opracowany nowy protokół, jak na przykład RSVP, pozwalający żądać i gwarantować QoS.

Podziękowanie

Autorzy wyrażają podziękowanie mgr. inż. Karolowi Grygerowi za implementację i wykonanie obliczeń.

LITERATURA

1. Rutkowska D., Piliński M., Rutkowski L.: Sieci neuronowe, algorytmy genetyczne i systemy rozmyte. WN-PWN, Warszawa-Łódź 1997.
2. Goldberg D.: Algorytmy genetyczne i ich zastosowania. Wydawnictwa Naukowo-Techniczne, Warszawa 1989.
3. Michalewicz Z.: Algorytmy genetyczne + struktury danych = programy ewolucyjne. WNT, Warszawa 1996.
4. Dengiz B., Altiparmak F.: Local Search Genetic Algorithm for Optimal Design of Reliable Networks. IEEE Trans. on Evolutionary Computation, 1(1997)3, 179-188.

5. Leung Y., Li G., Xu Z.-B.: A Genetic Algorithm for the Multiple Destination Routing. IEEE Trans. on Evolutionary Computation, 2(1998)4,s 150-161.
6. Dutt D.: Bringing Quality of Service to IP. Byte Vol.23,No 6, 1998,s.49-50

Recenzent: Prof. dr hab. inż. Zbigniew Czech

Wpłynęło do Redakcji 26 marca 1999 r.

Abstract

With the integration of voice and video with data networking, the nature of internet traffic is about to change substantially. While burst, loss-intolerant but delay tolerant network traffic was once a norm, soon things will become oriented more toward constant and loss tolerant traffic that requires predictable delay. The IP protocol itself allows specification of requirements concerning a desired quality of delivery and a relative priority. Eight bits named Type of Service (Fig. 1) allow their specification. In this paper three of them, i.e. short transmission time, high reliability and high through-output, binary flags are used to determine the best route. To this aim the genetic algorithm has been applied. Fig. 2 shows the gene used to form the individuals (routes) of the population. The simple fitness function is given in 3.3., then genetic operators are discussed, and an initialization algorithm is proposed. The individuals are crossed-over but there is no mutation to avoid reproduction of non-existent routes. If two or more individuals have equal fitness functions, the length of chromosomes is compared. Simulation was carried out with different network topologies and with the number of routers from 2 up to 80.

The simulation studies demonstrate that the proposed algorithm is efficient and the results are reproducible.