**Universitatea Politehnica Bucuresti**

**Facultatea de Electronica, Telecomunicatii si Tehnologia Informatiei**

Algoritmi de rutare cu implicatii in protocoale de rutare

Student: Cucu Bogdan-Cristian

Grupa: 442A

Cuprins:

 1. Introducere

 2. Caracteristicile algoritmilor de rutare

 3. Metrici ale algoritmilor de rutare

 4. Clasificarea algoritmilor de rutare

 5. Prezentarea algoritmilor de rutare

 6. Rutarea in retele ad-hoc

 7. Comparatie intre algoritmii de rutare

 8. Concluzii

 9. Bibliografie

**1*. Introducere despre rutare***

***Scopul algoritmilor de rutare[[1]](#footnote-1)***

Prin rutare se intelege procesul de selectare a caii optime pe care se transmite pachetul de la sursa la destinatie in interiorul unei retele sau intre doua retele diferite.

Rutarea este utilizata in cadrul mai multor tipuri de retele: retea telefonica(comutarea de circuite), retele electronice de date(Internet) si retele de transport.

In retelele cu comutatie de pachete, fiecare pachet este rutat cu ajutorul unor noduri numite centre de comutatie pana la destinatie, fiecare pachet fiind tratat independent.

Aceste centre de comutatie pot fi: rutere, punti(„bridges”), porti(„gateways”), firewall-uri sau switch-uri.

Rutarea presupune existenta unei tabele de rutare, care mentine o evidenta a rutelor catre diferite destinatii.

Tabela de rutare contine urmatoarele campuri:

* Adresa retea(net address);
* Masca retea(netmask);
* Adresa ruterului urmator(next hop);
* Adresa interfetei de iesire(optional).

Intr-un sens mai ingust al termenului de „rutare”, acesta intra in opozitie cu notiunea de „bridging” presupunand ca adresele similare de retea apartin de aceeasi retea.

In zilele noastre, rutarea a devenit forma dominanta de adresare in Internet. Bridging-ul a ramas totusi destul de larg utilizat in cadrul unor medii locale.

***2. Caracteristici algoritmi de rutare***

Algoritmii de rutare trebuie sa prezinte urmatoarele caracteristici[[2]](#footnote-2):

* *Robustete*: functionare corecta in situatii diferite;
* *Optimalitate*: gasirea rutei optime;
* *Corectitudine*: rutarea trebuie sa asigure ca pachetele ajung la destinatia lor corecta;
* *Adaptabilitate*: gasirea unor rute alternative in cazul caderii unei legaturi;
* *Stabilitatea*;
* *Convergenta*: algoritmul trebuie sa calculeze corect tabela de rutare, deoarece pe durata timpului de convergenta se pot propaga in retea informatii incorecte privind rutele;
* *Incarcare balansata* a retelei: se vor evita congestia si legaturile incete
* *Complexitatea* cat mai redusa: pentru a se reduce costurile implementarii hardware pe rutere.
* *Simplitatea:* overhead-ul(informatia redundanta) trebuie sa fie cat mai redus.

***3. Metrici ale algoritmilor de rutare***

Tabelele de rutare contin informatia folosita de software-ul de rutare pentru determinarea caii optime. Exista mai multe tipuri de metrici ce sunt utilizate de catre algoritmii de rutare. Algoritmii de dirijare mai sofisticati pot selecta ruta optima cu ajutorul mai multor metrici, combinate sub forma unei metrici hibride. [[3]](#footnote-3)

Exemple de metrici sunt:

* Lungimea caii;
* Fiabilitatea;
* Intarzierea;
* Latimea de banda necesara;
* Incarcarea;
* Costul datorat comunicatiei.

Lungimea caii este cel mai important criteriu. Unele protocoale de rutare permit administratorilor sa asigneze costuri arbitrare fiecarei legaturi din retea. In aceasta situatie, lungimea caii devine suma costurilor asociate cu fiecare legatura traversata. Alte protocoale presupun contorizarea numarului de hop-uri, adica numarul de rutere traversate pana la destinatie.

Fiabilitatea. In contextul algoritmilor de dirijare, fiabilitatea se refera la un bit error rate (BER) cat mai scazut pe fiecare legatura.

Intarzierea se refera la timpul necesar ca un pachet sa ajunga de la sursa la destinatie. Depinde de mai multi factori, printre care latimea de banda a legaturilor intermediare, precum si distanta fizica.

Incarcarea reprezinta gradul de utilizare a unui echipament de retea, cum este ruterul. Incarcarea tine cont de mai multi factori, incluzand utilizarea UCP si numarul de pachete procesate pe secunda.

Latimea de banda se refera la capacitatea unei legaturi.

***4. Clasificarea algoritmilor de rutare[[4]](#footnote-4)***

Algoritmii de rutare se pot clasifica dupa cum urmeaza:

* Algoritmi de rutare globali: care presupun cunoasterea intregii retele, toate conexiunile si costurile asociate fiecarei legaturi.
* Algoritmi de rutare descentralizati: initial fiecare nod cunoaste doar legaturile cu vecinii sai. La schimbarea tabelei de rutare au loc calcule iterative.

O alta clasificare imparte algoritmii de dirijare in:

* Neadaptivi(statici): rutele sunt configurate manual si sunt usor de implementat in retele de mici dimensiuni. Ca dezavantaje: nu sunt fezabili decat pentru retele de mici dimensiuni; pe masura ce complexitatea retelei creste, gestionarea tabelei de rutare se dovedeste a fi consumatoare de timp . In plus, daca o legatura cade, nu se poate stabili o ruta alternativa.
	+ Shortest Path Algorithm(Algoritmul de dirijare cu cea mai scurta cale)
	+ Flooding(Inundare)
* Adaptivi(dinamici): in general, rutarea dinamica nu depinde de dimensiunea retelei si se adapteaza automat cu schimbarile de topologie a retelei. Ca dezavantaj, acest tip de rutare ofera mai putina siguranta.
	+ Distance Vector Routing(Algoritm de dirijare cu vector distanta)
	+ Link State Routing(Rutarea folosind starea legaturii)
	+ Hierarchical Routing(Rutarea ierarhica)

O subclasificare a algoritmilor dinamici iii imparte in algoritmi cu rutare de tip:

* Broadcast(Difuziune)
* Multicast(Trimitere multipla)

***5. Prezentarea algoritmilor***

***Algoritmi statici[[5]](#footnote-5)***

***Shortest Path Routing***

Legaturile dintre rutere au un cost asociat. Acesta poate fi o functie de distanta, latime de banda, trafic mediu, cost al comunicatiei, intarziere, viteza de procesare a ruterului, etc.

Algoritmul Shortest Path cauta sa gaseasca cea mai putin costisitoare cale prin retea, pe baza unei functii de cost.

Algoritmul lui Dijkstra este un astfel de exemplu. Folosind algoritmul lui Dijkstra se poate determina drumul de cost minim. Algoritmul este utilizat in ***protocolul OSPF***.

Se eticheteaza fiecare nod, eticheta reprezentand distanta de la nodul sursa pana la nodul destinatie. Initial, deoarece nu este cunoscuta o cale, toate nodurile vor avea eticheta infinit. In decursul executiei algoritmului, noi cai sunt descoperite si se actualizeaza etichetele. Eticheta pot fi de doua tipuri: permanenta si temporara. La inceput, toate sunt temporare. O eticheta devine permanenta in momentul in care s-a determinat ruta optima.

***Flooding(Inundare)***

„Flooding” este un algoritm de rutare simplu in care fiecare pachet receptionat de un ruter este trimis prin fiecare legatura mai putin cea de la care a fost receptionat. Astfel, fiecare mesaj este livrat catre toate nodurile retelei. Exista mai multe variante ale acestui algoritm.

Sursa: http://www.ijarcce.com/upload/2013/october/18-o-Rahul\_Desai-FLOODING\_-\_An\_Efficient\_Routing\_Algorithm.pdf

Fiecare nod are rol de transmitator si receptor. Fiecare nod incearca sa trimita mai departe fiecare mesaj primit catre fiecare vecin.

Desi nu constituie o varianta eficienta de transmitere a pachetelor in retea, este un algoritm destul de utilizat ce faciliteaza descoperirea de rute si topologii.

Avantajul este dat de faptul ca pachetele vor fi, cu siguranta, livrate, din moment ce se vor utiliza toate caile disponibile in retea.

Dezavantajul este acela ca se iroseste multa banda si poate periclita fiabilitatea unei retele, in cazul, spre exemplu, a unui ping flood sau atac de tip DoS(Denial of Service).

***Algoritmi dinamici***

***Link State Routing(Rutarea folosind starea legaturii)[[6]](#footnote-6)***

Este utilizat in ***protocolul OSPF***. Larg raspandit in ziua de azi, imbunatateste convergenta Vectorului Distanta prin faptul ca fiecare nod partajeaza tabela de rutare cu toate celelalte noduri.

Pasii sunt urmatorii:

* Descoperirea vecinilor, prin trimiterea unui pachet ‚HELLO’. Celelalte rutere raspund cu un mesaj continand adresa lor unica;
* Masurarea intarzierii trimiterii de pachete catre vecini;
* Impachetarea informatiei despre vecini. Se trimit informatii privind: identificatorul nodului sursa, numarul secventei si varsta pachetului;
* Trimiterea acestei informatii catre toti ruterii din subnet. Se utilizeaza un algoritm de tip flooding;
* Calculul caii celei mai scurte catre fiecare ruter cu ajutorul informatiilor primite. Se utilizeaza algoritmul lui Dijkstra, astfel, fiecare ruter putand sa calculeze distanta de la sine pana la oricare alt ruter.Memoria necesara stocarii acestor informatii este proportionala cu k\*n, unde k este numarul de vecini, iar n numarul de rutere.

***Distance Vector(Vector distanta)***

Algoritmul mai este cunoscut si sub numele de ***Bellman-Ford sau Ford-Fulkerson***, algoritm ce este utilizat in ***protocoalele*** ***RIP, BGP si IGRP***. A fost utilizat in vechiul ARPANET si in Internet sub forma RIP.

Sunt o categorie de algoritmi care presupun construirea si actualizarea automata a unei tabele locale de rutare de catre echipamentele din retea.

Astfel, oricare nod poseda o tabela cu distantele(costurile) catre toate celelalte noduri si distribuie acest „vector” tuturor vecinilor sai.

Se fac urmatoarele presupuneri:

* Fiecare nod cunoaste costul leagaturilor cu nodurile adiacente;
* O legatura cazuta are un cost infinit;

Exemplu[[7]](#footnote-7):

 1 1

 1

 1 1 1

Sursa: http://www.cs.bu.edu/fac/byers/courses/ 791/F99/scribe\_notes/cs791-notes-990923.html

|  |  |
| --- | --- |
| **Tabela de rutare a nodului..** | **Distanta(Costul) catre nodul..** |
| **A** | **B** | **C** | **D** | **E** | **F** | **G** |
| **A** | 0 | 1 | 1 | 2 | 1 | 1 | 2 |
| **B** | 1 | 0 | 1 | 2 | 2 | 2 | 3 |
| **C** | 1 | 1 | 0 | 1 | 2 | 2 | 2 |
| **D** | 2 | 2 | 1 | 0 | 3 | 2 | 1 |
| **E** | 1 | 2 | 2 | 3 | 0 | 2 | 3 |
| **F** | 1 | 2 | 2 | 2 | 2 | 0 | 1 |
| **G** | 2 | 3 | 2 | 1 | 3 | 1 | 0 |

Sursa: http://www.cs.bu.edu/fac/byers/courses/791/F99/scribe\_notes/cs791-notes-990923.html

Pe scurt, modul de functionare consta din urmatorii pasi[[8]](#footnote-8):

* Fiecare nod calculeaza distanta dintre el insusi si toate nodurile din retea si stocheaza informatia in tabela proprie de rutare;
* Fiecare nod trimite tabela sa de rutare nodurilor vecine;
* Cand un nod primeste tabela cu distante de la vecinii sai, calculeaza cea mai scurta ruta catre toate nodurile si isi actualizeaza tabela.

Dezavantajele acestui algoritm sunt urmatoarele:

* Nu este scalabil;
* Daca topologia se modifica, timpul de convergenta este destul de mare, fapt ce se va reflecta in informatii eronate privind topologia retelei ;
* Problema numararii la infinit: daca un nod/o legatura cade lasand astfel un nod izolat, atunci celelalte noduri vor cauta sa determine ruta optima si va dura destul de mult pana cand rutele neviabile vor fi eliminate.

Algoritmul mai este utilizat si in cadrul ***protocolului de*** ***rutare Babel***, care este un algoritm eficient si robust atat in retelele cablate, cat si in cele wireless cu topologie de tip mesh[[9]](#footnote-9).

***Rutarea ierarhica[[10]](#footnote-10)***

Daca retelele sunt de dimensiuni mari atunci nu este viabil a utiliza tabele de rutare de dimensiuni mari, din ratiuni ce tin de memoria necesara, timpul cautarii, si timpul de calcul. Solutia consta in ierarhizare.

Ideea acestui algoritm consta in a incloui N intrari in tabela de rutare cu N rutere cu o singura intrare pentru un cluster de N rutere. Dezavantajul este dat de faptul ca nu va mai exista notiunea de ruta optimala pentru fiecare ruter.

***Algoritmul cu actualizare difuzata DUAL(Diffusing Update Algorithm)[[11]](#footnote-11)***

Algoritmul de rutare DUAL(Diffusing Update Algorithm) este un algoritm de rutare implementat ***in cadrul protocolului CISCO EIGPR*** , protocol ce utilizeaza algoritm DV (vector distanta).

Protocolul EIGRP utilizeaza o conditie de test pentru a se asigura ca vor fi selectate doar rutele care nu contin bucle.In situatia cand nu se gaseste nicio ruta catre destinatie, algoritmul DUAL face apel la algoritmul Bellman-Form pentru a obtine o ruta.

Utilizeaza trei tabele separate pentru calculul rutei, combinand informatiile ruterelor EIGRP. Spre deosebire de protocoalele de rutare bazate pe starea legaturilor(Link State Routing), la EIGRP informatiile cuprind: rute, metrici de cost pentru fiecare ruta in parte, timer-e.

Tabelea de vecini.

Contine informatii despre nodurile vecine, fiecare linie reprezentand un vecin, cu descrierea aferenta a interfetei de retea si a adresei. Mai este prevazut si un timer care se declanseaza periodic pentru a testa daca o conexiune este in continuare viabila.

Tabela de topologie.

Cuprinde informatii despre costul tuturor rutelor din cadrul sistemului autonom. Informatiile sunt preluate din tabelele ruterelor vecine. Rutele primara(succesor) si secundara(fezabila) vor fi determinate cu ajutorul cunoasterii topologiei. Campurile continute in aceasta tabela sunt: FD(Distanta fezabila) , reprezinta metrica calculata a unei rute catre nodul destinatie, RD(Distanta raportata) reprezinta distanta catre destinatie conform cu informatiile de la nodurile vecine, Starea Rutei(RouteStatus), indicand daca o ruta este activa sau pasiva. O ruta activa este o ruta care nu este disponibila si este in proces de re-calculare. O ruta pasiva este o ruta care poate fi folosita, fiind considerata stabila.

Tabela de rutare. Contine informatii privind ruta optima din punct de vedere al costului minim. Caile optime sunt date de succesorii din tabela de topologie.

Astfel, algoritmul DUAL calculeaza pe baza datelor primite de la nodurile vecine rutele primare(succesor) si pe cele fezabile,

***Rutarea de tip broadcast***

In cazul acestui tip de rutare, se ruteaza un pachet de la sursa catre toate nodurile din retea. Posibile implementari sunt : Flooding-ul si ***protocolul Spanning Tree Routing***. Dezavantajele sunt date de existenta pachetelor duplicat si consumul de latime da banda. Un exemplu de utilizare este streaming-ul multimedia.

***Rutarea de tip Multicast***

Prin intermediul acestui tip de rutare se incearca marirea eficientei fata de unicast, dar fara a facilita accesul la informatie oricui. Ca mod de functionare, ruterul interogheaza nodurile gazda apartinand unui anumit grup. Se trimit apoi pachetele catre rutere.

***Rutarea in retele ad-hoc[[12]](#footnote-12)***

Un protocol de rutare de tip ad-hoc reprezinta un standard care controleaza modul in care un nod decide pe care ruta vor fi transmise pachetele intr-o retea mobila de tip ad-hoc.

Fiecare nod este conectat la retea wireless si actioneaza atat ca un host(gazda), cat si ca un ruter. Retelele de noduri care se afla langa orice alt nod se numesc retele de tip „***ad-hoc***” sau ***MANET***(Mobile Ad hoc NETworks).

In retelele ad-hoc, nodurile nu cunosc topoloogia retelei din care fac parte. Ruterele trebuie sa invete topologia , astfel: un nod isi anunta prezenta si asculta mesajele de tip broadcast de la celelalte rutere.

Au fost propusi mai multi algoritmi de rutare pentru retelele ad hoc, unul dintre cei mai populari fiind algoritmul AODV(Ad hoc On-demand Distance Vector), propus de catre Perkins si Royer in 1999. Este de fapt o versiune a algoritmului Distance modificata pentru a opera in medii mobile in care nodurile au banda limitata sau o durata de viata dictata de nivelul bateriei.

***Algoritmul AODV***[[13]](#footnote-13)

Algoritmul AODV cauta rute la cerere, cand un nod sursa doreste sa trimita pachete. Utilizeaza numere de secventa pentru identificarea celei mai recente cai. Nodul sursa si nodurile intermediare pastreaza informatii despre urmatorul nod din retea(next-hop) corespunzator fiecarui flux de pachete de date trimise. Intr-un protocol de rutare de tip on-demand(de rutare la cerere) nodul sursa trimite pachete de cerere a rutei (RouteRequest) sub forma unui flood atunci cand nu se cunoaste o ruta catre nodul destinatie. Dintr-o astfel de cerere de obtinere a rutei poate descoperi chiar mai multe rute . Desoebirea dintre AODV si alti algoritmi care cauta rute la cerere consta in utilizarea numarului de secventa DestSeqNum. Astfel, un nod isi actualizeazatabele de rutare doar daca numarul de secventa al pachetului curent receptionat este mai mare decat valoarea stocata.

Cererea de aflare a rutei(RouteRequest) contine urmatoarele campuri:

* SrcID, numarul de identificare al nodului sursa;
* DestID, numarul de identificare al nodului destinatie;
* SrcSeqNum, numarul de secventa al nodului sursa;
* DestSeqNum, numarul de secventa al nodului destinatie;
* BcastId, identificator de broadcast;
* TTL, timp de viata.

Avantajul major al acestui algoritm consta in faptul ca rutele se stabilesc la cerere si ca utilizarea numerelor de secventa la destinatie permit aflarea celei mai recente rute, ruta optima. Timpul de stabilire al conexiunii este scazut.

Dezavantajul este ca nodurile intermediare pot duce la rute inconsistente daca numarul de secventa al rutei este vechi, iar nodurile intermediare poseda o valoare mai mare, dar nu actualizata.

***6. Comparatie intre algoritmii de rutare***

Scopul oricarui algoritm de rutare este simplu, gasirea celei mai bune cai de transmitere a pachetelor de la sursa la destinatie. Intrucat fiecare retea are reguli diferite, procesul de rutare nu este tot simplu. Fiecare legatura are asociat un cost, cost ce se traduce prin distanta fizica.

In continuare se compara algoritmii : Link State Rotuting si Distance Vector.

Complexitatea mesajului in cazul LS(Link State) este mult mai mare, intrucat LS trebuie sa transmita costul fiecarei legaturi catre toate nodurile, pe cand DV (Distance Vector) comunica doar cu vecinii sai si doar in situatia in care au loc modificari in tabela de rutare. Pe de alta parte, pentru DV, timpul de convergenta este mare si se poate ajunge la problema numararii la infinit.

In ceea ce priveste robustetea algoritmului, LS sta mai bine din acest punct de vedere intrucat va face broadcast numai pentru schimbarile asupra unui singur nod. DV, spre deosebire, va trebui sa ajusteze toti vectorii distanta catre toate nodurile, intr-un anumit numar de iteratii.

***7. Concluzii***

In concluzie, desi scopul algoritmilor de rutare/dirijare este unul simplu si anume aflarea caii optime de trimitere a pachetelor de la un nod la altul, procesul de rutare nu este unul simplu, deoarece trebuie sa se tina cont de o serie de criterii si cerinte. Trebuie determinata ruta „cea mai ieftina” , insa retele diferite au reguli diferite.

Algoritmii pot fi statici cand sunt cunoscute topologia si capacitatea legaturilor, sau dinamici, cand ruterele iau deciziile pe baza informatiilor adunate, gasind rute alternative.

Asadar, rutarea este o functie cheie a nivelului „Retea” obiectivul fiind dat de gasirea rutei optime.

***8. Bibliografie***

[1]http://www.inforetele.com/tag/tabela-rutare/

[2]http://thor.info.uaic.ro/~adria/teach/courses/net/files/8rc\_RutareCongestie.pdf

[3]-http://docwiki.cisco.com/wiki/Routing\_Basics

[4]-Andrew Tanenbaum – "Retele de calculatoare" editia IV, Ed. Byblos 2003

[5]-Andrew Tanenbaum – "Computer Networks" editia V, Ed. Prentice Hall 2011

[6]-http://www.ijarcce.com/upload/2013/october/18-o-Rahul\_Desai-FLOODING\_-\_An\_Efficient\_Routing\_Algorithm.pdf

[7]-http://www.cs.bu.edu/fac/byers/courses/791/F99/scribe\_notes/cs791-notes-990923.html

[8]-http://www.cs.bu.edu/fac/byers/courses/791/F99/scribe\_notes/cs791-notes-990923.html

[9]-https://tools.ietf.org/html/rfc6126

[10]-http://www.cse.iitk.ac.in/users/dheeraj/cs425/lec12.html/

[11]-Cisco EIGRP official white paper, Sep 09, 2005

[12]-Perkins, C.; Belding-Royer, E.; Das, S. (July 2003). [*Ad hoc On-Demand Distance Vector (AODV) Routing*](https://tools.ietf.org/html/rfc3561). [IETF](http://en.wikipedia.org/wiki/Internet_Engineering_Task_Force). RFC 3561. Retrieved 2010-06-18.

1. http://www.inforetele.com/tag/tabela-rutare/ [↑](#footnote-ref-1)
2. http://thor.info.uaic.ro/~adria/teach/courses/net/files/8rc\_RutareCongestie.pdf [↑](#footnote-ref-2)
3. http://docwiki.cisco.com/wiki/Routing\_Basics [↑](#footnote-ref-3)
4. * Andrew Tanenbaum – "Retele de calculatoare" editia IV, Ed. Byblos 2003 [↑](#footnote-ref-4)
5. Andrew Tanenbaum – "Computer Networks" editia V, Ed. Prentice Hall 2011 [↑](#footnote-ref-5)
6. Andrew Tanenbaum – "Computer Networks" editia V, Ed. Prentice Hall 2011 [↑](#footnote-ref-6)
7. http://www.cs.bu.edu/fac/byers/courses/791/F99/scribe\_notes/cs791-notes-990923.html [↑](#footnote-ref-7)
8. * Andrew Tanenbaum – "Computer Networks" editia V, Ed. Prentice Hall 2011 [↑](#footnote-ref-8)
9. https://tools.ietf.org/html/rfc6126 [↑](#footnote-ref-9)
10. http://www.cse.iitk.ac.in/users/dheeraj/cs425/lec12.html/ [↑](#footnote-ref-10)
11. Cisco EIGRP official white paper, Sep 09, 2005 [↑](#footnote-ref-11)
12. Andrew Tanenbaum – "Computer Networks" editia V, Ed. Prentice Hall 2011 [↑](#footnote-ref-12)
13. Perkins, C.; Belding-Royer, E.; Das, S. (July 2003). [*Ad hoc On-Demand Distance Vector (AODV) Routing*](https://tools.ietf.org/html/rfc3561). [IETF](http://en.wikipedia.org/wiki/Internet_Engineering_Task_Force). RFC 3561. Retrieved 2010-06-18. [↑](#footnote-ref-13)