13. Разпределена маршрутизация с дистантен вектор

Малко история

Маршрутизиращ протокол с дистантен вектор (distance vector protocol) е използван отначало в ARPANET.

По-късно в Интернет намира широко приложение като RIP (Routing Information Protocol).

Основите на тези алгоритми са поставени от Белман (1957 г.), Форд и Фолкерсън (1962 г.).

Затова са известни като алгоритми Белман-Форд или Форд-Фолкерсън.

Cisco Systems – подобни протоколи IGRP и EIGRP

Общи положения

При маршрутизацията с дистантен вектор (distance vector routing) всеки маршрутизатор изгражда и поддържа маршрутна таблица, в която всеки ред съдържа адрес на местоназначение, адрес на следващата стъпка към това местоназначение по най-добрия известен до момента път и дължината на този път (метрика).

Маршрутизаторите разменят само с директно свързаните към тях съседни маршрутизатори съобщения с информация от маршрутните си таблици за всички възли в мрежата.

Метрика

Предполага се, че всеки маршрутизатор знае метриката на връзките до своите съседи.

Ако метриката е брой стъпки или маршрутизатори до местоназначението (хопове), разстоянието до всеки съсед е 1.

Ако метриката е натоварване на възела, разстоянието до всеки съсед е броя на пакетите в изходящата опашка към този пакет.

Ако метриката е времезакъснение, маршрутизаторът периодично изпраща “ехо” пакети до съседните му маршрутизатори и измерва закъснението на техния отговор.

Нека да предположим, че за метрика е избрано времезакъснението на пакетите.

Запълване на таблицата

Веднъж на всеки T милисекунди маршрутизаторите изпращат на своите съседи съдържанието на маршрутната си таблица.

Нека даден маршрутизатор *J* получи маршрутната таблица на съседа си *X*, като *Xi* е обявеното от *X* закъснение до маршрутизатора *i*.

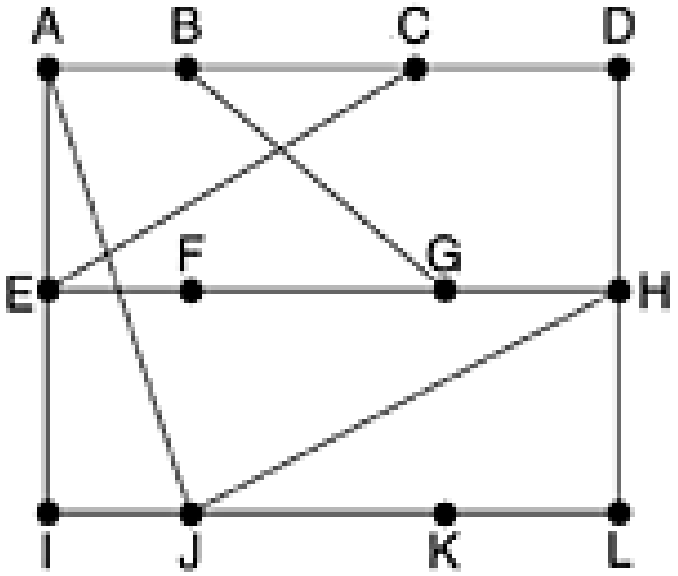
Ако закъснението от *J* до *X* е *m*, то от *J* до всеки маршрутизатор *i* има път през *X* със закъснение *Xi* + *m*.

Възможни са четири случая:

Запълване на таблицата

1. ако в маршрутната таблица на *J* няма ред за направлението *i*, то *J* добавя такъв ред и записва в него следваща стъпка *X* и закъснение *X*i + *m*;
2. ако в маршрутната таблица на *J* има ред за направлението *i* и в него е записана следваща стъпка *X*, то стойността на закъснението се актуализира с *X*i + *m* независимо дали тя е по-голяма или по-малка от предходната стойност;
3. ако в маршрутната таблица на *J* има ред за направлението *i*, в него е записана следваща стъпка различна от *X* и закъснение по-голямо от *X*i + *m*, то редът се актуализира като за следваща стъпка се записва *X*, а за закъснение *X*i + *m*;
4. ако в маршрутната таблица на *J* има ред за направлението *i*, в него е записана следваща стъпка различна от *X* и закъснение по-малко или равно на *X*i + *m*, то редът не се променя.

Примерна мрежа



Примерна мрежа

Векторите, които *J* получава от своите съседи *A*, *I*, *H*, *K* са следните:

To	A	I	H	K
A	0	24	20	21
B	12	36	31	28
C	25	18	19	36
D	40	27	8	24
E	14	7	30	22
F	23	20	19	40
G	18	31	6	31
H	17	20	0	19
I	21	0	14	22
J	9	11	7	10
K	24	22	22	0
L	29	33	9	9

Примерна мрежа

Измерените закъснения между *J* и съседите му *A*, *I*, *H*, *K* са следните:

JA	JI	JH	JK
8	10	12	6

На базата на тази информация, новият вектор на разстоянията, изчислен от *J* е следния:

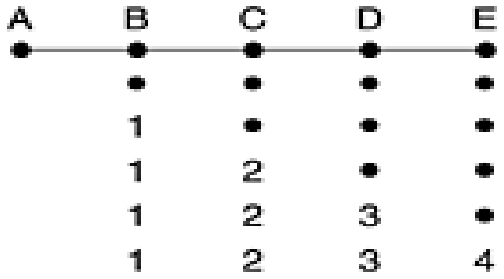
To		
A	8	A
B	20	A
C	28	I
D	20	H
E	17	I
F	30	I
G	18	H
H	12	H
I	10	I
J	0	—
K	6	K
L	15	K

Недостатък

Сериозен недостатък на маршрутизиращите алгоритми с дистантен вектор е ниската им скорост на сходимост.

Добрите новини се разпространяват бързо в мрежата, но лошите новини обикновено изискват твърде голям брой периодични съобщения за да достигнат до всички маршрутизатори.

Да разгледаме следния пример с метрика в хопове.



Добавяне на обект

Нека маршрутизаторът *A* в началото не е включен в мрежата.

Всички останали маршрутизатори знаят това - в маршрутната им таблица към направлението *A* е записано ∞ (достатъчно голямо число, трябва да е поне с единица повече от диаметъра на мрежата). Това е отразено на първия ред по-горе.

След включването на *A* останалите маршрутизатори научават за това събитие чрез няколко обмена на своите вектори на разстоянията, всеки от които се извършва едновременно между всички съседни маршрутизатори.

Добавяне на обект

При първата обмяна *B* научава от *A* за път с дължина 0 до *A* и записва в своята таблица, че *A* е на разстояние 1.

В този момент останалите маршрутизатори все още не са научили за включването на *A*. Това е отразено на втория ред по-горе.

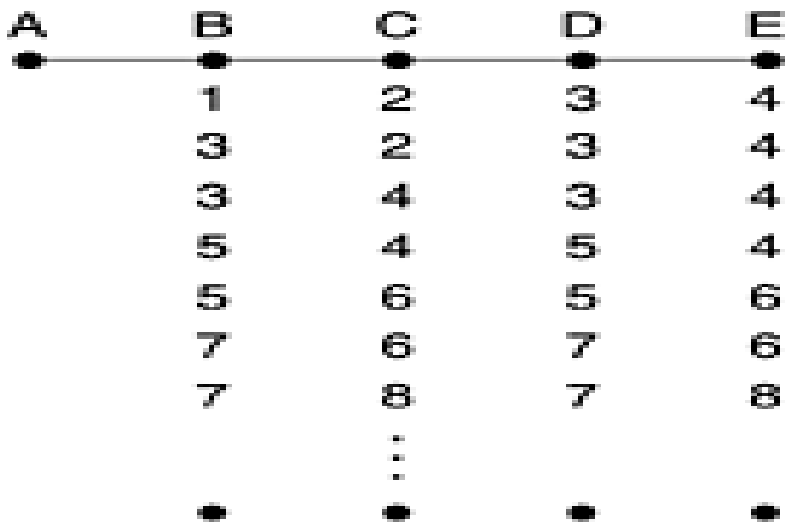
При следващия обмен *C* научава, че от *B* съществува път до *A* с дължина 1 и записва в своя вектор път до *A* през *B* на разстояние 2 и т.н.

Добавяне на обект

По-общо в мрежа с диаметър *k* хопа са необходими най-много *k* размени на съобщения за разпространяване на новината за появил се по-добър път.

Иключване на обект

Да разгледаме друг пример.



Иключване на обект

Нека всички маршрутизатори в началото са включени в мрежата.

Да предположим, че *A* спира да работи или се прекъсва връзката от *A* до *B*, което от гледна точка на *B* е същото.

При първия обмен *B* не получава информация от *A*, но получава информация от *C*, че има път до *A* с дължина 2.

B не знае, че пътя от *C* до *A* минава през него - от негова гледна точка би могъл да съществува друг независим път от *C* до *A*, затова *B* записва в таблицата си в реда за *A* път с дължина 3 и следваща стъпка *C*.

D и *E* не променят маршрутните си таблици при първия обмен на векторите на разстоянията. На следващия обмен *C* научава за два възможни пътя до *A*, и двата с дължина 4, единият през *B*, другият през *D.*

C избира и записва в маршрутната си таблица единия от тях в зависимост от реда на обработването на съобщенията от *B* и *D*.

Count to Infinity

Резултатът от продължаващия обмен е отразен в следващите редове по-горе. Той ще продължи, докато стойностите по направленията към *A* и в четирите машрутизатора не достигнат ∞.

Този проблем се нарича **броене до безкрайност (count to infinity)**.

Едно частично негово решение е т.н. **разделяне на хоризонта** (**split horizon**).

При него се въвежда ново правило - ако в маршрутната таблица на *X* в реда за *Y* е записана следваща стъпка *Z*, то *X* не изпраща към *Z* информация за маршрута към *Y*.

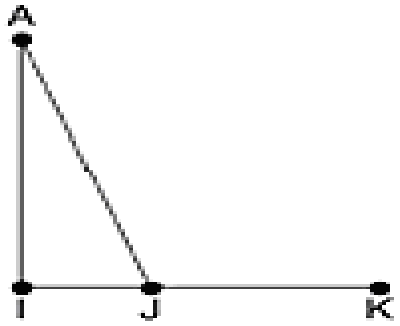
Частично решение - **split horizon**

В горния пример на втория обмен на вектори, *C* не изпраща към *B* информация за маршрута към *A*, тъй като маршрутът от *C* към *A* минава през *B*.

Въвеждането на разделяне на хоризонта не решава напълно проблема броене до безкрайност.

Да разгледаме следния пример.

split horizon



В началото *A* и *I* имат пътища с дължина 2 стъпки до *K* през *J*, а *J* има път с дължина 1 до *K*.

Да предположим, че връзката между *J* и *K* отпадне.

split horizon

Тогава на първия последвал обмен *J* няма да получи информация за нов път до *K* през *A* или *I* по правилото за разделяне на хоризонта и правилно ще заключи, че *K* е недостижим.

На следващия обмен *A* и *I* научават, че няма път до *K* през *J*, но *A* научава за път до *K* през *I* с дължина 3 и *I* научава за път до *K* през *A* с дължина 3.

Така въпреки разделянето на хоризонта, *A* и *I* ще броят до безкрайност.

Triggered Updates

Незабавното изпращане на извънредни съобщения за обновяване (Triggered Updates) намалява вероятността за поява на цикли.

Когато един маршрутизатор промени метриката за даден маршрут, той трябва веднага да изпрати извънредно съобщение.

Например, връзката J-K отпада.

Triggered Updates

J ще съобщи веднага на A и I, че:

$$d(J\text{-}K) = \infty$$

A и I записват в маршрутните си таблици, че K е недостижим и веднага съобщават това на своите съседи.

При следващо редовно (периодично) обновяване няма да възникне “броене до безкрайност” между A и I. В таблиците им няма да има остаряла информация за K.

Hold down

Triggered Updates не достига едновременно до всички маршрутизатори.

Маршрутизатор, още неполучил съобщението, би изпратил редовно периодично съобщение за обновяване с неточна информация до друг маршрутизатор, който вече има актуалната информация за променен маршрут.

Hold down

Защита срещу това: hold down таймер.

След изпращане или получаване на triggered update маршрутизаторът стартира hold down таймер.

До неговото нулиране не приемат съобщения за обновяване на променения маршрут.