الگوريتمهاي مسيريابي

اصول عملكرد

روترها از الگوريتمهاي مسيريابي،براي يافتن بهترين مسير تا مقصد استفاده مينمايند هنگامي كه ما در مورد بهترين مسير صحبت ميكنيم،پارامترهايي همانند تعداد hopها (مسيري كه يك بسته از يك روتر ديگر در شبكه منتقل ميشود).زمان تغيير و هزينه ارتباطي ارسال بسته را در نظر ميگيريم.

مبتني بر اينكه روترها چگونه اطلاعاتي در مورد ساختار يك شبكه جمع آوري مينمايند و نيز تحليل آنها از اطلاعات براي تعيين بهترين مسير،ما دو الگوريتم مسير يابي اصلي را در اختيار داريم:الگوريتم مسير يابي عمومي و الگوريتمهاي مسير يابي غير متمركز.

در الگوريتم هاي مسير يابي غير متمركز،هر روتر اطلاعاتي در مورد روترهايي كه مستقيما به آنها متصل ميباشند در اختيار دارد. در اين روش هر روتر در مورد همه روتر هاي موجود در شبكه،اطلاعات در اختيار ندارد.اين الگوريتمها تحت نام الگوريتمهاي (DV distance vector)معروف هستند.در الگوريتمهاي مسيريابي عمومي،هر روتر اطلاعات كاملي در مورد همه روترهاي ديگر شبكه و نيز وضعيت ترافيك شبكه در اختيار دارد.اين الگوريتمها تحت نام الگوريتمهاي(LS(Link state معروف هستند.ما در ادامه مقاله به بررسي الگوريتمهاي LS ميپردازيم.

الگوريتمهاي LS

در الگوريتمهاي LS ،هر روتر ميبايست مراحل ذيل را به انجام رساند:

روترهاي را كه به لحاظ فيزيكي به آنها متصل ميباشد را شناسايي نموده و هنگامي كه شروع به كار ميكند آدرسهايIP آنها بدست آورد. اين روتر ابتدا يك بسته HELLO را روي شبكه ارسال ميكند. هر روتري كه اين بسته را دريافت ميكند از طريق يك پيام كه داراي آدرس IP خود اين روتر ميباشد به پيام HELLO پاسخ ميدهد.

زمان تاخير مربوط به روترهاي مجاور را اندازه گيري نمايد(يا هر پارامتر مهم ديگري از شبكه همانند ترافيك متوسط)

براي انجام اين كار ،روترها بسته هاي echo را روي شبكه ارسال ميكنند. هر روتري كه اين بسته ها را دريافت ميكند با يك بسته echo reply به آن پاسخ ميدهد.با تقسيم زمان مسير رفت و برگشت به دو،روترها ميتوانند زمان تاخير را محاسبه كنند.(زمان مسير رفت و برگشت،سنجشي از تاخير فعلي روي يك شبكه ميباشد)توجه داشته باشيد كه اين زمان شامل زمانهاي ارسال و پردازش ميباشد.

اطلاعات خود را در مورد شبكه،براي استفاده ساير روترها منتشر نموده و اطلاعات روترهاي ديگر را دريافت كند.

در اين مرحله همه روترها دانش خود را با روتر هاي ديگر به اشتراك گذاشته و اطلاعات مربوط به شبكه را با يكديگر مبادله ميكنند.با اين روش هر روتر ميتواند در مورد ساختار و وضعيت شبكه اطلاعات كافي بدست آورد.

با استفاده از اين الگوريتم مناسب،بهترين مسير بين هر دو گره از شبكه راشناسايي كند.

در اين مرحله،روترها بهترين مسير تا هر گره را انتخاب ميكنند.آنها اين كار را با استفاده از يك الگوريتم همانند الگوريتم كوتاهترين مسير Dijkstra انجام ميدهند.در اين الگوريتم،يك روتر مبتني بر اطلاعاتي كه از ساير روترها جمع آوري نموده است،گرافي از شبكه را ايجاد مينمايد.اين گراف مكان روترهاي موجود در شبكه و نقاط پيوند آنها را به يكديگر نشان ميدهد.هر پيوند با يك شماره به نام weightياCost مشخص ميشود.اين شماره تابعي از زمان تاخير،متوسط ترافيك و گاهي اوقات تعداد hopهاي بين گره ها ميباشد.براي مثال اگر دو پيوند بين يك گره و مقصد وجود داشته باشد،روتر پيوندي با كمترين Weight را انتخاب ميكند.

الگوريتم Dijkstra داراي مراحل ذيل ميباشد:

روتر گرافي از شبكه را ايجاد نموده و گره هاي منبع و مقصد(براي مثال V1 وV2)را شناسايي ميكند.سپس يك ماتريس به نام ماتريس adjacency را ميسازد.در اين ماتريس يك مختصه مبين Weight ميباشد.براي مثال[i,j]،وزن يك پيوند بين Viو Vj ميباشد.در صورتي كه هيچ پيوند مستقيمي بين Vi وVj وجود نداشته باشد اين وزن (ويت) بصورت infinity در نظر گرفته ميشود.

روتر يك مجموعه ركورد وضعيت را براي هر گره روي شبكه ايجاد مينمايد اين ركورد داراي سه فيلد ميباشد:

فيلد Predecessor:اولين فيلدي كه گره قبلي را نشان ميدهد.

فيلد Length:فيلد دوم كه جمع وزنهاي از منبع تا آن گره را نشان ميدهد.

فيلد Label:آخرين فيلد كه وضعيت گره را نشان ميدهد.هر گره ميتواند داراي يك مود وضعيت باشد:tentative يا permanent

روتر،پارامترهاي مجموعه ركورد وضعيت براي همه گره ها را آماده سازي اوليه نموده و طول آنها را در حالت infinity و Labelآن را در وضعيت tentative قرار ميدهد.

روتر،يك گره T را ايجاد ميكند.براي مثال اگر V1 ميبايست گره T منبع باشد،روتر برچسب V1را در وضعيت permanent قرار ميدهد.هنگامي كه يك Label به حالت permanent تغيير ميكند ديگر هرگز تغيير نخواهد كرد. يك گره T در واقع يك agent ميباشد.

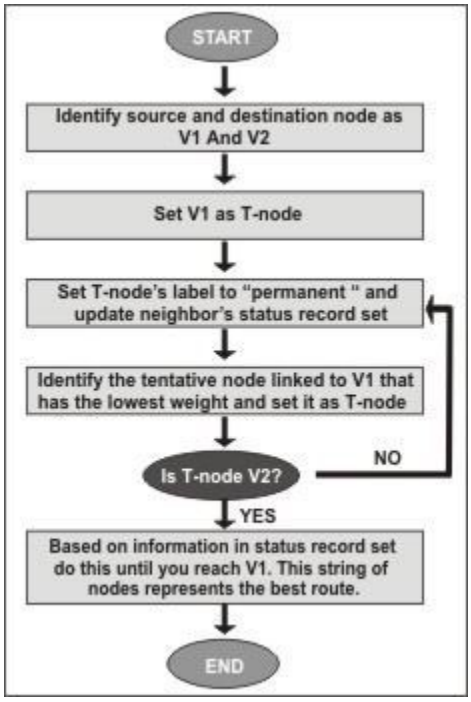
روتر،مجموع ركورد وضعيت مربوط به همه گره هاي Tentative را كه مستقيما به گره T منبع متصل هستند،روز آمد مينمايد.

روتر همه گره هاي Tentative را بررسي نموده و گرهاي را كه وزن آن تا V1 كمترين مقدار را دارد انتخاب ميكند.سپس اين گره،گره Tمقصد خواهد بود

اگر اين گره،V2 نباشد(گره مقصد)روتر به مرحله 5باز ميگردد.

اگر اين گره V2 باشد،روتر گره قبلي آن را از مجموع ركورد وضعيت استخراج نموده و اين كار را انجام ميدهد تا به V1 برسد،اين فرست از گره ها،بهترين مسير از V1تاV2را نشان ميدهد.

اين مراحل بصورت يك فلوچارت در شكل نشان داده شده است ما از اين الگوريتم بعنوان يك مثال در ادامه مقاله استفاده خواهيم نمود.

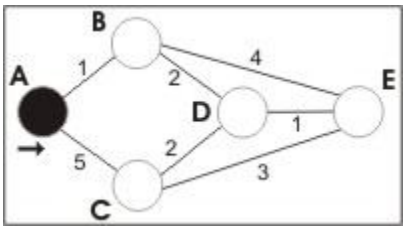


مثال

الگوريتم Dijkstra

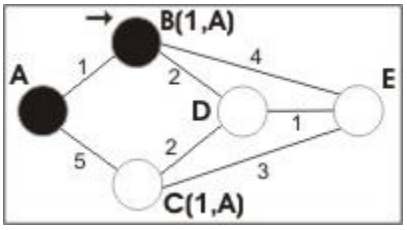
در اينجا ما ميخواهيم بهترين مسير بين گره هاي A و E را پيدا كنيم همانطور كه ميبينيد 6 مسير بين A و E وجود دارد.(ACDBE، ABDCE، ACDE، ABDE، ACE،ABE)و واضح است كه ABDEبهترين مسير ميباشد زيرا كمترين وزن را دارد اما هميشه به اين سادگي نيست و برخي موارد پيچيده وجود دارد كه در آن ما مجبوريم از الگوريتم هايي براي يافتن بهترين مسير استفاده كنيم.

همانطور كه در تصوير ذيل مشاهده ميكنيد،گره منبع(A)بعنوان گره Tانتخواب شده و بنابراين برچسب آن، Permanent ميباشد. (ما گره هاي Permanent را با دايره هاي تو پر و گره هاي Tرا با يك پيكان نشان ميدهيم)



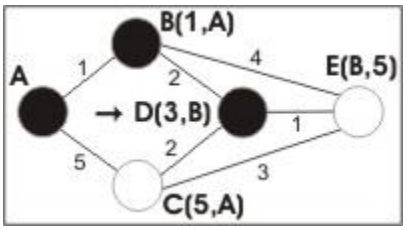
در اين مرحله شما ميبينيد كه مجموع ركورد وضعيت گره هاي Tentative كه مستقيما به گره(C) B،Tمتصل شده اند،تغيير يافته است.همچنين از آنجايي كه گره Bكمترين

وزن را دارد،بعنوان گره T انتخاب شده و برچسب آن به حالت Permanent تغيير كرده است.

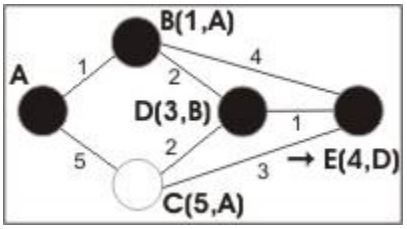


در اين مرحله همانند مرحله قبل دو مجموعه ركورد وضعيت گره هايي كه Tentative داراي اتصال مستقيم به گره T ميباشد(D،E)تغيير كرده است.همچنين از آنجايي كه گره D وزن كمتري دارد،بعنوان گره T انتخاب شده و برچسب آن به وضعيت Permanent تغيير كرده است.

در اين مرحله ما هيچ گره Tentative نداريم بنابراين فقط گره T بعدي را شناسايي ميكنيم.از آنجايي كه Eداراي كمترين وزن ميباشد بعنوان گرهTانتخاب ميشود.



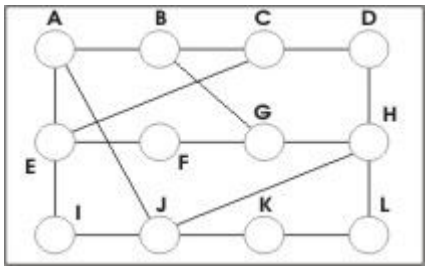
E گره مقصد بوده بنابر اين كار ما در اينجا تمام ميشود.اكنون ما كار شناسايي مسير را به انتها رسانده ايم.گره قبلي Eگره D،گره B ميباشد و گره قبلي B،گره A ميباشد.بنابر اين بهترين مسير ABDE است در اين مورد وزن كل مسير،(1+2+1)4ميباشد.



با وجودي كه اين الگوريتم بخوبي كار ميكند اما آنقدر پيچيده است كه زمان پردازش آن براي روتر طولاني بوده و راندمان شبكه را كاهش ميدهد.همچنين اگر يك روتر اطلاعات غلطي را به روترهاي ديگر بدهد،همه تصميمات مسير يابي نادرست خواهد بود.

الگوريتمهاي DV

الگوريتمهاي DVبا نامهاي الگوريتمهاي مسيريابي Bellman-Ford و ford-fulkerson نيز ياد ميشوند.در اين الگوريتمها،هر روتر داراي يك جدول مسيريابي ميباشد كه بهترين مسير تا هر مقصد را نشان ميدهد.يك گراف معمولي و جدول مسيريابي مربوط به روتر G در شكل زير نشان داده شده است.



همانطور كه در جدول مشاهده ميكنيد،اگر روتر G بخواهد بسته هايي را به روتر D ارسال كند،ميبايست آنها را به روتر H ارسال نمايد.هنگامي كه بسته ها به روتر H رسيدند،اين روتر جدول خود را بررسي نموده و روي چگونگي ارسال بسته ها به D تصميم گيري مي كند.

Line	Weight	Destination
A	8	A
A	20	B
I	28	C
H	20	D
I	17	E
I	30	F
H	18	G
H	12	H
I	10	I
-	0	J
K	6	K
K	15	L

در الگوريتمهاي DV،هر روتر ميبايست مراحل ذيل را انجام دهد:

وزن لينكهاي مستقيما متصل به آن را اندازه گرفته و اين اطلاعات را در جدول خود ذخيره كند.

در يك دوره زماني خواص،روتر جدول خود را به روترهاي مجاور ارسال نموده و جدول مسيريابي هر يك از روترهاي مجاور خود را دريافت ميكند.

مبتني بر اطلاعات بدست آمده از جداول مسيريابي روترهاي مجاور،جدول خود را روز آمدسازي مينمايد.

يكي از مهمترين مشكلات،هنگام كار با الگوريتمهاي DV،مشكل Count to infinity اجازه بدهيد اين مشكل را با ذكر يك مثال روشن كنيم.

همانطور كه در قسمت ذيل نشان داده شده است يك شبكه را در ذهن خود تصور كنيد.همانطور كه در اين جدول ميبينيد،فقط يك پيوند بين A و ساير بخشهاي شبكه وجود دارد.در اينجا شما ميتوانيد،اين گراف و جدول مسيريابي همه گره ها را مشاهده كنيد:

D	C	B	A	
3,D	2,B	1,A	0,-	A
3,D	2,C	0,-	1,B	B
1,C	0,-	1,C	2,B	C
0,-	1,D	2,C	3,B	D

اكنون تصور كنيد كه پيوند بين A و B قطع شود.در اين هنگام، B جدول خود را تصحيح ميكند بعد از يك مدت زمان خاص،روترها جداول خود را مبادله نموده و بنابراين B جدول مسيريابي C را دريافت ميكند. از آنجايي كه C نميداند چه اتفاقي براي پيوند بين A و B رخ داده است اين اطلاعات را حفظ ميكند.B اين جدول را دريافت نموده و فكر ميكند كه يك پيوند جداگانه بين Cو A وجود دارد،بنابراين جدول خود را تصحيح نموده مقدار infinity را به 3 تغيير ميدهد.به همين شكل دوباره روترها جداول خود را مبادله ميكنند.هنگامي كه C،جدول مسيريابي B را دريافت ميكند،مشاهده ميكنيد كه B وزن پيوند خود تا A را از 1به 3 تغيير داده است،بنابراين C ،جدول خود را روزآمد نموده و وزن پيوند خود تا Aرا به 4 تغيير ميدهد.اين پروسه تكرار ميشود تا همه گره ها وزن پيوند خود را تا A در وضعيت infinity قرار دهند.اين وضعيت در جدول زير نشان داده شده است.

D	C	B	
3,C	2,B	∞,A	Sum of weight to A after link cut
3,C	2,B	3,C	Sum of weight to B after 1st updating
3,C	4,B	3,C	Sum of weight to A after 2nd updating
5,C	4,B	5,C	Sum of weight to A after 3rd updating
5,C	6,B	5,C	Sum of weight to A after 4th updating
7,C	6,B	7,C	Sum of weight to A after 5th updating

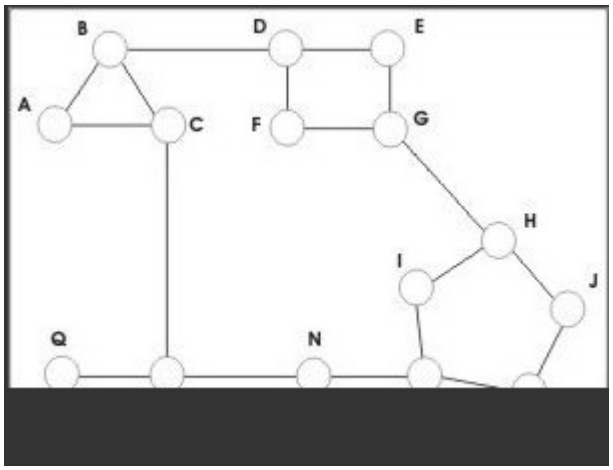
در اين روش متخصصين ميگويند،الگوريتمهاي DV داراي يك سرعت همگرايي پايين هستند.يك روش براي حل اين مشكل در مورد روترها،ارسال اطلاعات فقط به روترهايي

ميباشد كه داراي پيوند انحصاري تا مقصد نيستند.براي مثال در اين مورد،C نميبايست هيچ اطلاعاتي را به گره B در مورد A ارسال كند زيرا B فقط يك مسير تا A را در اختيار دارد.

مسيريابي سلسله مراتبي

همانطور كه شما ميبينيد،در هر دو الگوريتم LS و DV،هر روتر مجبور به ذخيره نمودن اطلاعات مربوط به روترهاي ديگر ميباشد.هنگامي كه اندازه شبكه رشد ميكند،تعداد روترهاي شبكه افزايش مي يابد در نتيجه اندازه جداول مسيريابي نيز افزايش مي يابد و روترها نميوانند ترافيك شبكه را به طور موثر كنترل كنند.ما از مسيريابي سلسله مراتبي براي برطرف كردن اين مشكل استفاده ميكنيم.اجازه بدهيد اين موضوع با ذكر يك مثال روشن كنيم:

ما از الگوريتمهاي DV براي يافتن بهترين مسير بين گره ها استفاده ميكنيم در وضعيت نشان داده شده در ذيل،هر گره از شبكه مجبور به نگهداري يك جدول مسيريابي با 17 ركورد ميباشد.در اينجا يك گراف معمولي و جدول مسيريابي مربوط به A ارائه شده است.

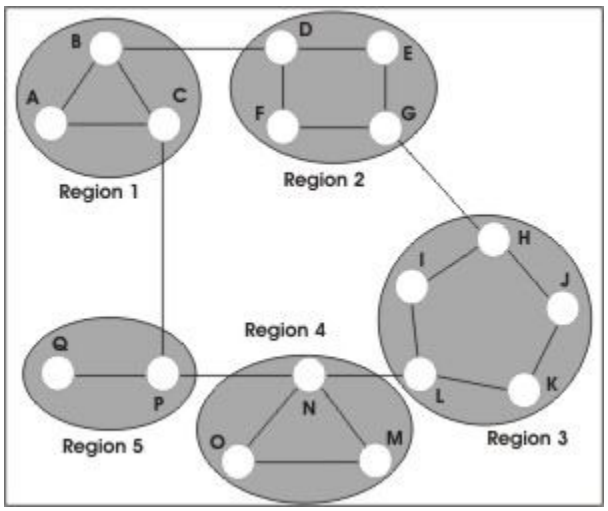


Weight	Line	Destination
-	-	A
1	B	B
1	C	C
2	B	D
3	B	E
3	B	F
4	B	G
5	B	H
5	C	I

6	C	J
5	C	K
4	C	L
4	C	M
3	C	N
4	C	O
2	C	P
3	C	Q

در مسيريابي سلسله مراتبي،روترها در گروههايي به نام regions طبقه بندي ميشوند.هر روتر داراي اطلاعاتي فقط در مورد روترهايي كه در region آنها قرار دارد در اختيار داشته و هيچ گونه اطلاعاتي در مورد region هاي ديگر ندارند.

در اين مثال ما شبكه خود را به پنج region تقسيم ميكنيم.اگر A بخواهد بسته ها را به هر روتر در region2 ارسال كند،آنها را به B ارسال ميكند و الي آخر.



Weight	Line	Destination
-	-	A
1	B	B
1	C	C
2	B	Region 2
2	C	Region 3
3	C	Region 4
4	C	Region 5

در اين نوع مسيريابي،جداول را ميتوان خلاصه نمود بنابراين راندمان شبكه بهبود مييابد.مثال بالا مسيريابي سلسله مراتبي دو سطحي را نشان ميدهد همچنين ميتوان از مسيريابي سلسله مراتبي 3 سطحي و 4 سطحي استفاده كرد.در

مسيريابي سلسله مراتبي 3سطحي،شبكه به تعدادي كلاستر تقسيم بندي ميشود.هر كلاستر متشكل از تعدادي region و هر region داراي تعدادي روتر ميباشد.مسيريابي سلسله مراتبي به طور وسيعي در مسيريابي اينترنت مورد استفاده قرار ميگيرد و استفاده از چندين پروتكل مسيريابي را ممكن مي سازد.

برگرفته از سايت : www.PooyeshR.com