شبکه‌های مخابراتی

سید حمید صفوی
دانشکده فنی و مهندسی
دانشگاه محقق اردبیلی
نیمسال دوم ۹۷-۹۸

الگوریتم‌های مسیریابی

مسیریابی

- همان‌طور که اشاره شد، مهم‌ترین وظیفه لایه شبکه، **مسیریابی** بسته‌ها از مبدأ تا مقصد است.
- در بسیاری از شبکه‌ها هر بسته پس از چندین پرش (multiple hops) به مقصد می‌رسد.
- **الگوریتم مسیریابی:** کدام لینک خروجی روتر باید استفاده شود؟
- اگر مسیریابی برای شبکه دیتاگرام است، تصمیم مسیریابی برای هر بسته به صورت مستقل گرفته می‌شود. چون ممکن است بهترین مسیر در طول زمان عوض شود.
- اگر مسیریابی برای شبکه Virtual Circuit (VC) است، تصمیم مسیریابی هربار که VC جدیدی ایجاد شد، گرفته می‌شود. از آن به بعد، بسته‌های داده، مسیری را که در VC مشخص شده‌است را دنبال می‌کنند. به این حالت session routing نیز گفته می‌شود. (هنگامی که شما وارد VPN دانشگاه می‌شوید.)

مسیریابی

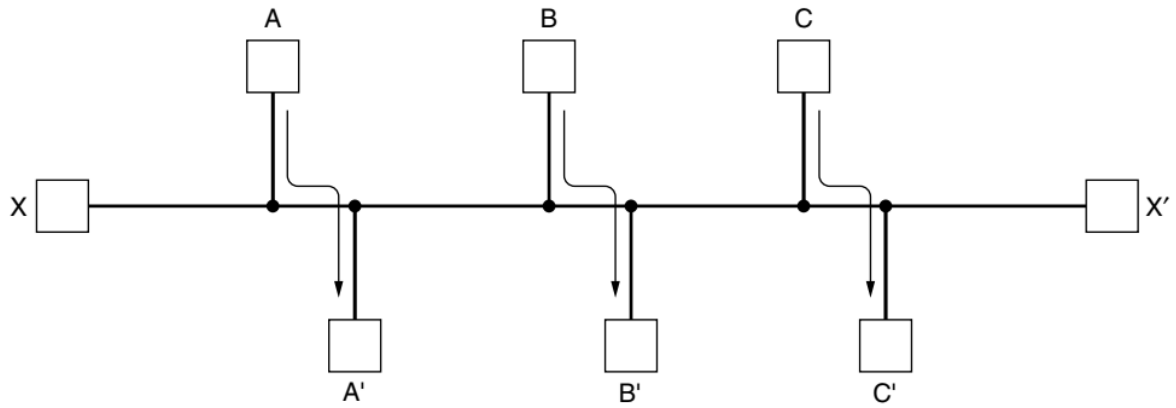
- **نیازمندی‌های الگوریتم‌های مسیریابی:**
  - **صحت (Correctness)**
  - **سادگی (Simplicity)**
  - **مقاوم بودن (Robustness):** کنارآمدن با تغییر توپولوژی شبکه
    - زمانی که شبکه ای راه‌اندازی می‌شود، معمولا انتظار می‌رود که بدون خرابی سراسری شبکه، چندین سال از آن به صورت پیوسته استفاده شود. در این مدت، قطعا مشکلات سخت افزاری و نرم افزاری جزئی برای هر روتر، لینک و هاست ها ایجاد خواهد شد. در نتیجه توپولوژی شبکه ممکن است بارها تغییر کند. الگوریتم مسیریابی باید نسبت به این تغییرات توپولوژی و ترافیک مقاوم باشد بدون آنکه نیاز به تنظیمات در همه روترها و هاست ها داشته باشد.
  - **پایداری (Stability):** سریع همگرا شود.
    - الگوریتم‌هایی وجود دارند که مجموعه مشخصی از مسیرها همگرا نمی‌شوند. فارغ از اینکه چه مدت در حال اجرا هستند. الگوریتم پایدار باید به نقطه تعادل رسیده و در همان نقطه باقی بماند. همچنین همگرایی باید سریع اتفاق بیفتد چراکه ممکن است تا زمانی که به نقطه تعادل برسیم، ارتباط قطع شود.

وظایف لایه شبکه

- **نیازمندی‌های الگوریتم‌های مسیریابی:**
  - **عدالت (Fairness)**
  - **کارآمدی (Efficiency)**
- بدیهی است که این دو معیار باید در الگوریتم مسیریابی لحاظ شوند. اما نکته اینجاست که در برخی از زمان‌ها این دو معیار مقابل هم قرار می‌گیرند و مصالحه‌ای بین این دو وجود دارد.

وظایف لایه شبکه

- **مثال:** فرض کنید ظرفیت هر کدام از لینک‌ها 100Mbps باشد.
- هنگامی که نود X به نود X’ داده ارسال نمی‌کند، نود A با بیشینه ظرفیت ممکن به A’ داده ارسال می‌کند. همین طور نود B به B’ و نود C به C’.
- حال فرض کنید نود X می‌خواهد به نود X’ با ظرفیت 50Mbps داده ارسال کند. در این حالت روترهای بین نود A و A’ تصمیم می‌گیرند 75Mbps ترافیک A را جابجا کنند و 25Mbps هم ترافیک X را. در این حالت کل ترافیک جابجاشده برابر 75+75+75+25=250Mbps است. حال آنکه در حالت اول برابر 100+100+100=300Mbps بود!
- بین **عدالت و کارآمدی** مصالحه وجود دارد.



الگوریتم‌های مسیریابی

- **روش‌های غیر وفقی (Nonadaptive Algorithms) و یا استاتیک**
  - از اندازه‌گیری‌ها و تخمین ترافیک فعلی و یا توپولوژی استفاده نمی‌کند.
  - مسیریابی استاتیک: مسیر را به صورت آفلاین پیدا می‌کند و یا با یک روش معین (Deterministic) این کار را انجام می‌دهد.
  - مثال: Flooding ارسال بسته‌ها در همه جهت‌ها

الگوریتم‌های مسیریابی

- روش‌های وفقی (Adaptive Algorithms) و یا پویا (Dynamic)
  - انتخاب مسیر به حالت شبکه بستگی دارد.
  - اطلاعات حالت شبکه به صورت منظم به روزرسانی می‌شود.
  - تصمیم مسیریابی بر اساس چه معیاری است؟
    - تعداد پرش‌ها (Hop Count)
    - فاصله (Distance)، زمان ارسال تخمینی
    - نرخ ترافیک
    - پارامترهای ازدحام

مسیریابی استاتیک

الگوریتم Flooding

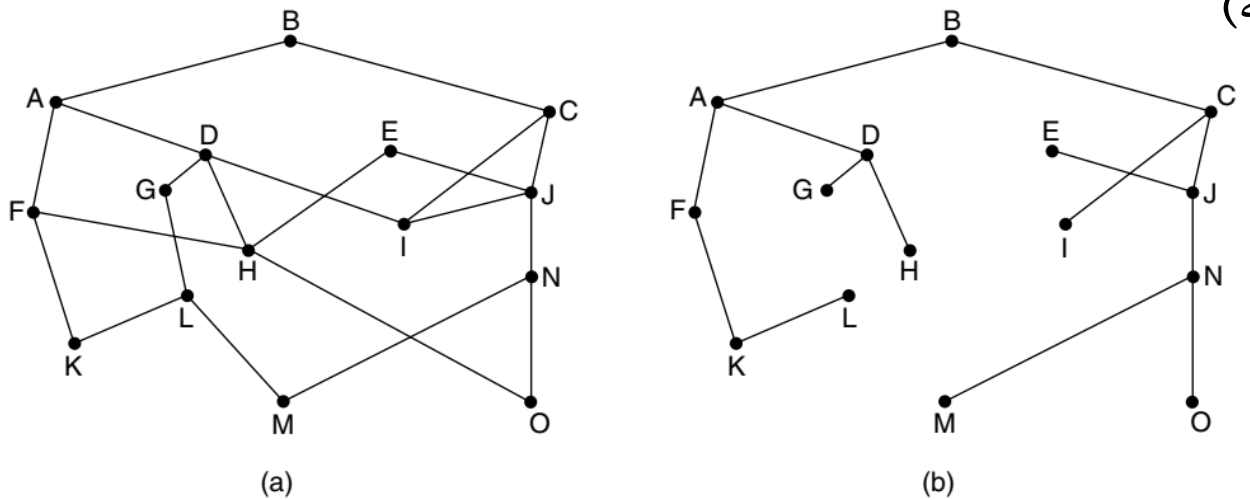
- **ایده:** بسته‌های رسیده را برای همه مسیرها به جز مسیری که آمده، فوروارد کن.
- اجتناب از Over-flooding:
  - تنظیم شمارنده پرش (Hop Counter): هر روتر که بسته را دریافت کرد، یک شماره از HC کم می‌کند و زمانی که HC=0 شد، بسته دیگر فوروارد نمی‌شود.
  - اجتناب از دوبار مسیریابی برای یک بسته یکسان: نگهداری لیستی از شمارنده بسته‌ها
  - روش flooding انتخابی: فقط در جهت‌هایی که احتمالاً درست است، flooding صورت پذیرد.

خواص الگوریتم Flooding

- در بسیاری از کاربردها عملی نیست.
- ترافیک زیادی را ایجاد می‌کند.
- مناسب برای
  - شبکه‌های پخش
  - زمانی که مقاوم بودن (Robustness) اولویت اصلی است. در کاربردهای نظامی
- استفاده به عنوان الگوریتم مرجع برای مقایسه دیگر الگوریتم‌ها
- عدم نیاز به تنظیمات زیاد. هر نود فقط باید همسایه‌های خود را بشناسد.
- **همواره کوتاه‌ترین مسیر به لحاظ تأخیر را پیدا می‌کند!!**

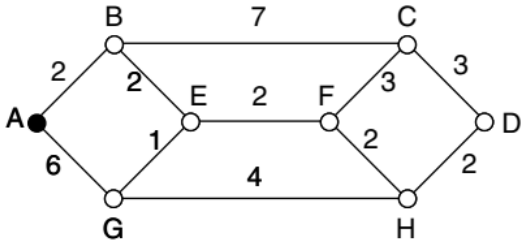
مسیریابی استاتیک

- **اصل بهینگی (Bellman, 1957)**
  - اگر روتر C مسیر بهینه از روتر I به روتر B باشد، مسیر بهینه از روتر C به روتر B نیز همان مسیر است.
- **نتیجه:**
  - همه مسیرهای بهینه از تمامی مبدأها به یک نود مشخص داده شده، تشکیل یک **Sink Tree** می‌دهند. (عدم وجود حلقه)

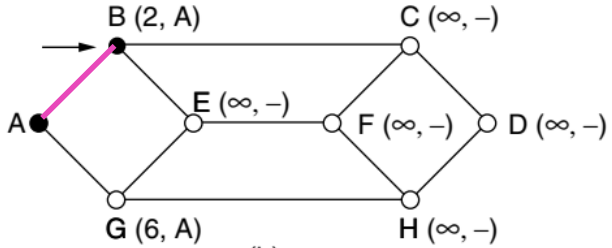


(a) A network. (b) A sink tree for router *B*.

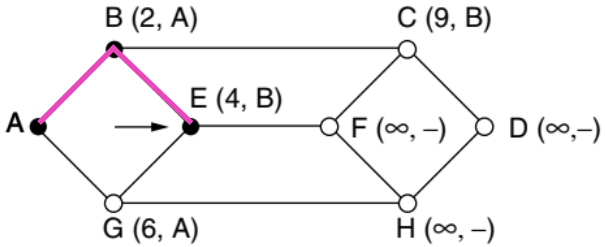
الگوریتم کوتاه‌ترین مسیر
(Shortest Path Algorithm)



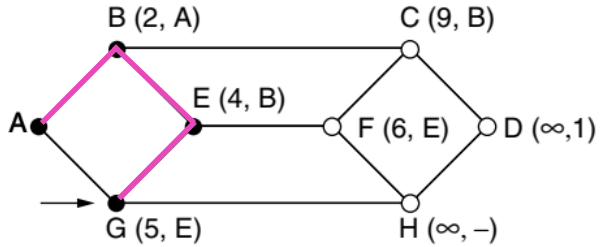
(a)



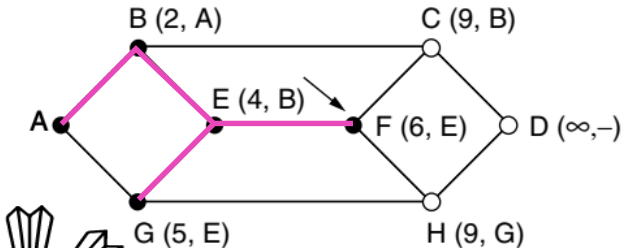
(b)



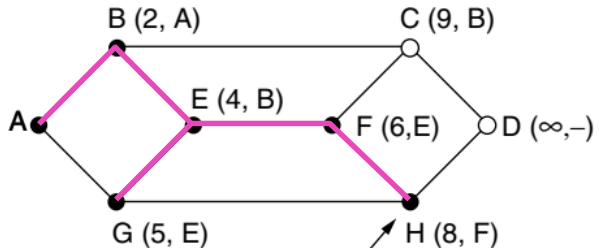
(c)



(d)



(e)



(f)

- الگوریتم کوتاهترین مسیر Djikstra

	Base Set	B	C	D	E	F	G	H
0	A	2,A	∞	∞	∞	∞	6,A	∞
1	A,B		9,B	∞	4,B	∞	6,A	∞
2	A,B,E		9,B	∞		6,E	5,E	∞
3	A,B,E,G		9,B	∞		6,E		8,F
4	A,B,E,G,F		9,B	∞				8,F
5	A,B,E,G,F,H		9,B	10,H				

الگوریتم کوتاه‌ترین مسیر
(Shortest Path Algorithm)

- **توضیح اسلاید قبلی: یافتن sink tree نود A**
- همه نودهای دیگر را در ستون‌های دیگر یادداشت می‌کنیم.
- فاصله نود A تا همسایه‌هایش را در جدول یادداشت می‌کنیم. (نودهای B و G)
- کدام یک از این دو همسایه به نود A نزدیکتر است؟ نود B. بنابراین نود B را جزو مجموعه درخت درنظر می‌گیریم.
- در قدم بعدی همسایه‌های A و B کدام نودها هستند؟ نودهای C و E و G
- کدام یک از این نودها کمترین فاصله را نسبت به نود A دارند؟ نود E فاصله ۴ از طریق B
- بنابراین نود E نیز وارد مجموعه درخت می‌شود.
- حال همسایه‌های نودهای A و B و E کدامند؟ نودهای C و F و G
- در این مرحله اتفاق جالبی برای G می افتد. مسیر کوتاهتر از طریق E جایگزین مسیر قبلی می شود.
- ...
- **این الگوریتم نیاز به توپولوژی و متریک‌ها دارد.**

مسیریابی پویا

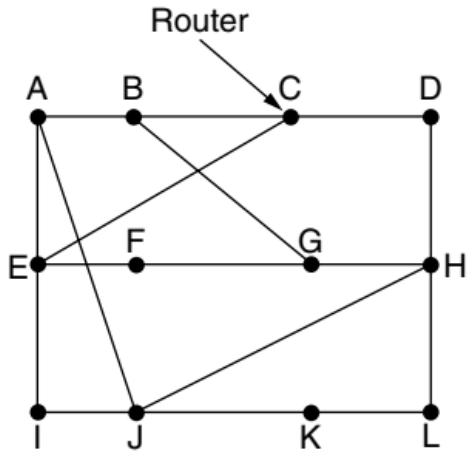
الگوریتم مسیریابی بردار فاصله
(Distance Vector Routing)

- **هدف:** به روز رسانی جدول مسیریابی در نودها

شماره پورت	آدرس مقصد

- هر یک از نودها دارای چندین پورت هستند. هدف الگوریتم مسیریابی این است که هر نود در شبکه بداند برای آنکه بسته‌ای به نودهای دیگر ارسال کند، از کدام پورت خود استفاده کند.
- **الگوریتم Distance Vector Routing**، استفاده شده در شبکه ARPANET اولیه
- در این الگوریتم، هر روتر **جدول فاصله‌ها** را با شماره‌های مقصد نگه می‌دارد.
- هر سطر شامل لینک خروجی و فاصله تخمین زده شده می‌باشد.

الگوریتم مسیریابی بردار فاصله
(Distance Vector Routing)



To	A	I	H	K	New estimated delay from J	Line
A	0	24	20	21	8	A
B	12	36	31	28	20	A
C	25	18	19	36	28	I
D	40	27	8	24	20	H
E	14	7	30	22	17	I
F	23	20	19	40	30	I
G	18	31	6	31	18	H
H	17	20	0	19	12	H
I	21	0	14	22	10	I
J	9	11	7	10	0	–
K	24	22	22	0	6	K
L	29	33	9	9	15	K
	JA delay is 8	JI delay is 10	JH delay is 12	JK delay is 6	New routing table for J	

Vectors received from J's four neighbors

(a) (b)

(a) A network. (b) Input from *A*, *I*, *H*, *K*, and the new routing table for *J*.

الگوریتم مسیریابی بردار فاصله
(Distance Vector Routing)

- **ایده این الگوریتم:**
- نود J فاصله خود را از همسایه‌های خودش با ارسال یک بسته با نام Echo اندازه‌گیری می‌کند. مدت زمان ارسال بسته Echo و دریافت پاسخ آن تقسیم بر دو، معیاری از فاصله نودهاست.
- متریک مورد نظر برای بررسی این روش، تأخیر لازم برای رسیدن بسته از مبدأ به مقصد است.
- نود J، علاوه بر این اندازه‌گیری‌ها، **جدول فاصله‌های نودهای همسایه** را نیز از آن‌ها می‌گیرد.
- بنابراین، نود J به جز فاصله خود تا همسایه‌هایش، جدول فاصله‌های آن‌ها را هم در اختیار دارد. به عبارت دیگر، به همسایه‌های خود می‌گوید که فاصله شما تا نودهای دیگر چقدر است؟

الگوریتم مسیریابی بردار فاصله
(Distance Vector Routing)

- **توضیح الگوریتم با یک مثال: (شکل دو اسلاید قبل)**
- می‌خواهیم در نود J تصمیم بگیریم که اگر به نودهای دیگر بسته‌ای بخواهیم ارسال کنیم، از کدام پورت استفاده کنیم؟
- فاصله نود J تا نود B از طریق نود A، برابر 20 است. از نود J تا نود A برابر 8 اندازه‌گیری شده است. در جدول نود A نیز نوشته شده‌است از نود A تا نود B فاصله برابر 12 است. بنابراین در مجموع فاصله نود J تا نود B برابر 20 است.
- فاصله نود J تا نود B از طریق نود I، برابر 46 است. از نود J تا نود I برابر 10 اندازه‌گیری شده است. در جدول نود I نیز نوشته شده‌است از نود I تا نود B فاصله برابر 36 است. بنابراین در مجموع فاصله نود J تا نود B برابر 46 است.

الگوریتم مسیریابی بردار فاصله
(Distance Vector Routing)

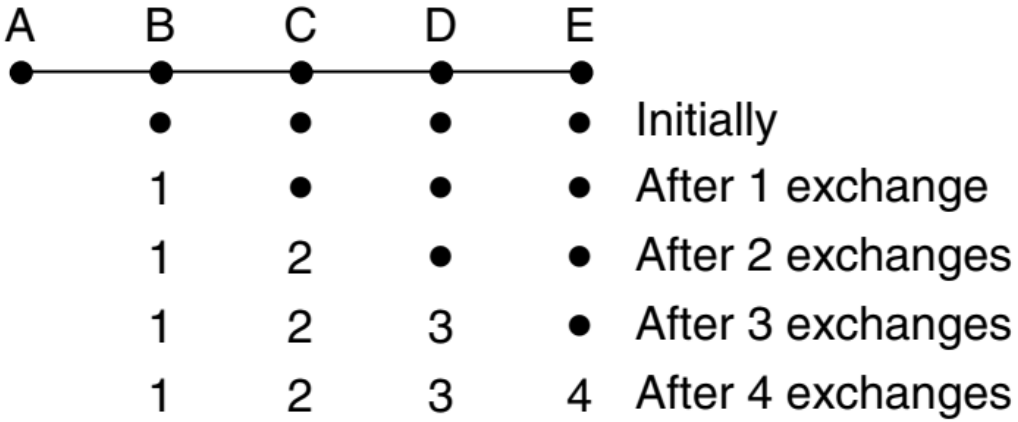
- **توضیح الگوریتم با یک مثال: (شکل دو اسلاید قبل)**
- فاصله نود J تا نود B از طریق نود H، برابر 43 است. از نود J تا نود H برابر 12 اندازه‌گیری شده است. در جدول نود H نیز نوشته شده‌است از نود H تا نود B فاصله برابر 31 است. بنابراین در مجموع فاصله نود J تا نود B برابر 43 است.
- فاصله نود J تا نود B از طریق نود K، برابر 34 است. از نود J تا نود K برابر 6 اندازه‌گیری شده است. در جدول نود K نیز نوشته شده‌است از نود K تا نود B فاصله برابر 28 است. بنابراین در مجموع فاصله نود J تا نود B برابر 34 است.

- **کدام‌یک نزدیکتر است؟ از طریق نود A**

الگوریتم مسیریابی بردار فاصله
(Distance Vector Routing)

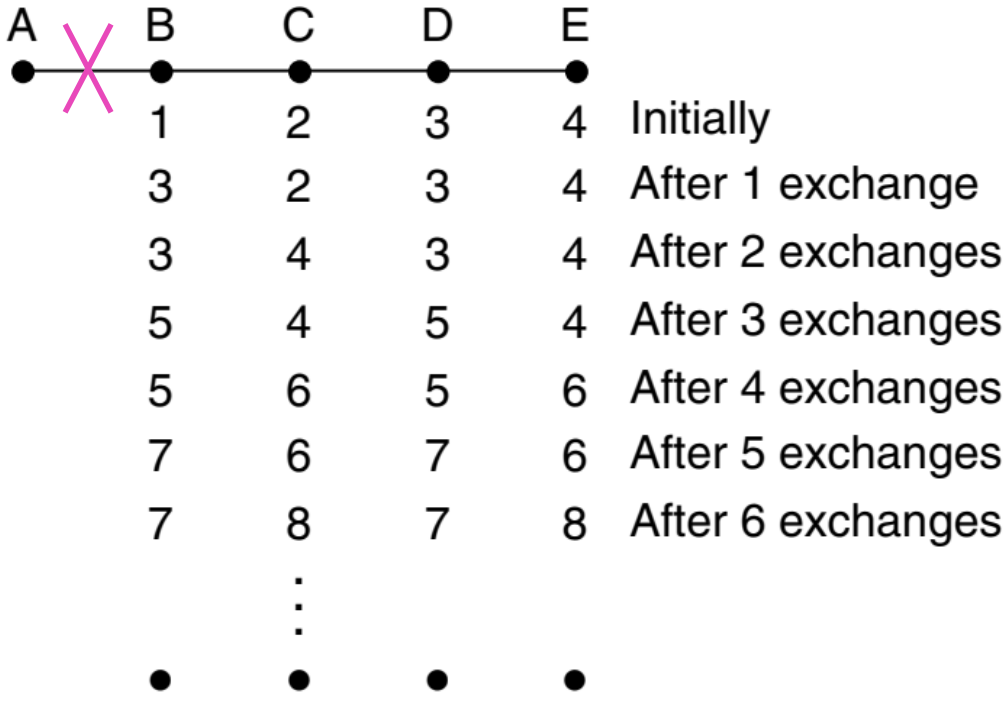
- **سوال:** اول کار که هیچ یک از نودها، هیچ اطلاعاتی از شبکه ندارند، جدول چگونه پر می‌شود؟
  - هر نود فاصله خودش تا همسایه‌های خود را می‌داند. فاصله بقیه نودها را **بی‌نهایت** می‌گذارد.
  - خیلی زود و پس از چندین تکرار، جداول به روز رسانی شده و کوتاهترین مسیر در جدول همه نودها پیدا خواهد شد.
  - اگر همه اعداد جدول بی‌نهایت شوند، بنابراین هیچ مسیری انتخاب نمی‌شود. معنی آن این است که در حال حاضر هیچ مسیری به آن نود، از طریق نود فعلی وجود ندارد.
  - به دلیل اتصال همه نودها به همدیگر، حتماً پس از چند تکرار، خبر وجود یک مسیری به نود خواسته‌شده، به همه خواهد رسید.
- مدت زمانی که طول می‌کشد تا یک تغییر (اضافه شدن نود جدید، لینک جدید و ...) در شبکه به اطلاع همه نودها برسد، به نام **سرعت همگرایی** مطرح است.

مسئله شمارش تا بی‌نهایت!



- توپولوژی ساده شکل روبرو را برای ABCDE در نظر بگیرید.
- در ابتدا هیچ یک از نودها فاصله‌ها را نمی‌داند.
- در تکرار اول، نود B فاصله خود از نود A را محاسبه می‌کند (یک پرش).
- در تکرار دوم، نود C فاصله خود از نود B را محاسبه می‌کند (یک پرش). فاصله خود از نود A را با توجه به جدول B محاسبه می‌کند (دو پرش).
- در تکرار سوم، نود C و در تکرا چهارم، نود D و در تکرار پنجم، نود E فاصله‌های خود را به روز رسانی می‌کنند.

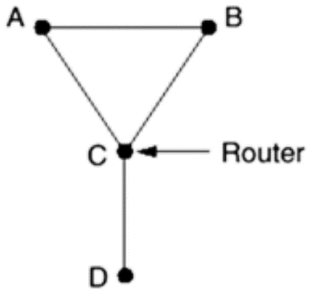
مسئله شمارش تا بی‌نهایت! (۲)



- **وقوع یک اتفاق بد: لینک نود A به نود B قطع می‌شود.**
- نود B فاصله خودش تا نود A را اندازه‌گیری می‌کند. پاسخی دریافت نمی‌کند. بنابراین نتیجه می‌گیرد که مستقیما راهی ندارد. حال جدول نود C به دستش می‌رسد. در جدول نود C فاصله تا نود A، برابر ۲ گزارش شده‌است. بنابراین به این نتیجه می‌رسد که فاصله‌اش تا نود A از طریق نود C برابر ۳ است!!!
- بدیهی است که این تصمیم اشتباه است. زیرا هیچ راهی برای رسیدن به نود A وجود ندارد.

مسئله شمارش تا بی‌نهایت! (۳)

- اعداد موجود در جدول نود C تا زمانی اعتبار داشت که نود B به نود A متصل بود.
- روش مسیریابی بردار فاصله در این مثال خاص، **صحیح (Correct)** نیست.
- این مسئله، مشکل بنیادی روش مسیریابی بردار فاصله است و به **مسئله شمارش تا بی‌نهایت** (The count-to-infinity problem) مشهور است.
- **راه حل:** روش‌های زیادی برای حل این مشکل پیشنهاد شده‌است.
- **روش Split Horizon Hack:** می‌دانیم نود C بسته‌هایی که به A باید ارسال شوند را از طریق نود B به نود A ارسال می‌کرد. حال که خود نود B از نود C می‌پرسد کمترین فاصله تا نود A چقدر است، عدد **بی‌نهایت** گزارش می‌شود.
- خبر بد: توپولوژی‌های دیگری نیز وجود دارد که این روش شکست می‌خورد!



مشکل اصلی روش مسیریابی بردار فاصله

- در روش مسیریابی بردار فاصله، **مشاهده هر نود از شبکه محدود به همسایه‌های خود است**. در این روش، نودها به تنهایی هیچ احساسی از توپولوژی شبکه ندارند.
- به همین دلیل از یک مدتی به بعد، این روش دیگر در ARPANET استفاده نشد.
- اگر توپولوژی شبکه را هر نود بداند، می‌تواند sink tree را بسازد و مشکل حل شود.

الگوریتم مسیریابی وضعیت لینک
(Link State Routing)

- هنگامی که توپولوژی شبکه تغییر کند، روش مسیریابی بردار فاصله به دلیل مسئله شمارش تا بی‌نهایت، زمان همگرایی زیادی خواهد داشت. بنابراین دنبال روش جدیدی هستیم.
- الگوریتم مسیریابی وضعیت لینک، پایه روش‌هایی است که امروزه استفاده می‌شود.
- در سال ۱۹۷۹ این روش جایگزین روش مسیریابی بردار فاصله در ARPANET شد.

- **ایده اصلی:** برای اینکه روش پیشنهادی درست و پایدار باشد، راهی به جز این نداریم که توپولوژی شبکه را همه نودها بدانند.

الگوریتم مسیریابی وضعیت لینک
(Link State Routing)

- ۵ گام اصلی در این روش:
  - شناسایی همسایه‌ها و همچنین آدرس شبکه آن‌ها
  - انتخاب معیار فاصله و یا هزینه تا هر یک از روترها
  - تولید بسته‌های وضعیت لینک
  - ارسال (دریافت) بسته‌های وضعیت لینک به (از) روترهای دیگر
  - دانستن توپولوژی و محاسبه کوتاهترین مسیر به هر روتر دیگر موجود در شبکه

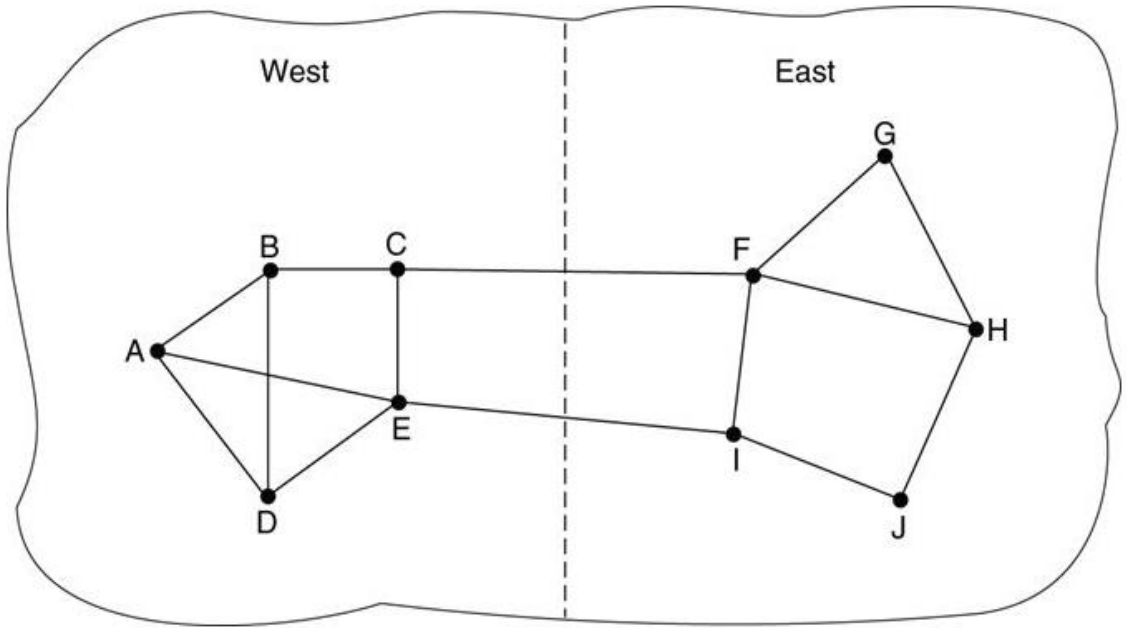
شناسایی همسایه‌ها

- اولین وظیفه هر روتر پس از روشن شدن، شناسایی همسایه‌های خود می‌باشد.
- برای رسیدن به این هدف، بسته خاصی به نام HELLO از طریق پورت‌های روتر ارسال می‌شود. انتظار می‌رود هر روتر دیگری که به این روتر متصل است، در پاسخ خود را معرفی کند.
- به منظور جلوگیری از ابهام، این نام‌ها برای هر روتر باید یکتا باشد.

انتخاب معیار فاصله

- در الگوریتم مسیریابی وضعیت لینک، هر لینک نیاز به انتخاب معیاری برای فاصله و یا هزینه دارد.
- هزینه رسیدن به روتر همسایه می‌تواند به صورت اتوماتیک تنظیم شود و یا توسط اپراتور شبکه تنظیم شود.
- انتخاب متداول این است که **هزینه به صورت معکوس با پهنای باند رابطه داشته باشد**. برای مثال لینک اترنت 1Gbps ممکن است عدد هزینه 1 داشته باشد در حالیکه لینک اترنت 100Mbps عدد هزینه 10 داشته باشد. این کار باعث می‌شود تا لینک‌های پرظرفیت، انتخاب بهتری باشند.
- اگر شبکه به لحاظ جغرافیایی گسترده باشد، **تأخیر هر لینک** می‌تواند به صورت مناسبی که لینک‌های کوتاه‌تر انتخاب بهتری باشند، به عنوان معیار هزینه در نظر گرفته شود.

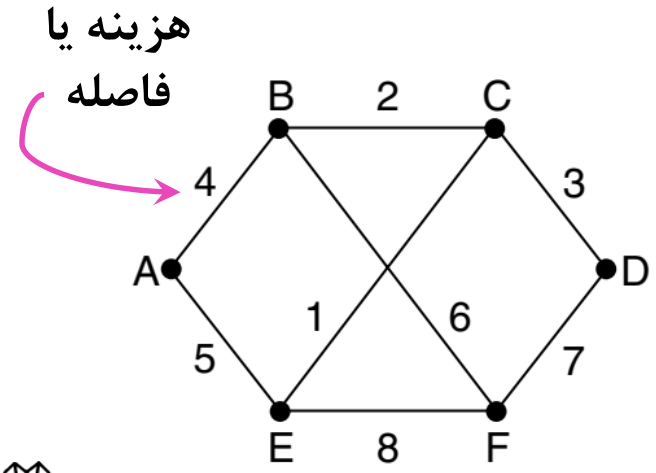
اندازه‌گیری تأخیر



- **نحوه اندازه‌گیری:**
  - ارسال بسته خاصی به نام ECHO و بلافاصله دریافت پاسخ از همسایه‌ها
  - محاسبه زمان رفت و برگشت و تقسیم آن بر ۲
  - ممکن است مقدار متوسط در چندین بار اندازه‌گیری گزارش شود.
  - برای گزارش تأخیر می‌توان از دو روش بهره جست:
    - با احتساب تأخیر صف
    - بدون احتساب تأخیر صف
  - در صورت انتخاب روش با احتساب تأخیر صف، ممکن است ترافیک لینک‌ها دچار نوسان شود. این حالت پایدار نیست.

بسته وضعیت لینک
(Link State Packet)

- نحوه تولید بسته وضعیت لینک خیلی آسان است.
- نکته مهم تصمیم درباره این است که چه زمانی این بسته را تولید کنیم؟
  - به صورت متناوب و در بازه‌های زمانی منظم
  - هر زمان که اتفاق مهمی در شبکه رخ دهد. مانند از بین رفتن و یا اضافه شدن لینک و یا همسایه



(a)

Link State Packets

A	B	C	D	E	F
Seq.	Seq.	Seq.	Seq.	Seq.	Seq.
Age	Age	Age	Age	Age	Age
B 4	A 4	B 2	C 3	A 5	B 6
E 5	C 2	D 3	F 7	C 1	D 7
	F 6	E 1		F 8	E 8

هویت فرستنده
شماره دنباله
عمر
لیست همسایه‌ها

(b)

نحوه توزیع بسته وضعیت لینک

- از روش Flooding برای توزیع بسته وضعیت لینک بین همه روترها استفاده می‌شود.
- از شماره دنباله (Sequence Number) برای کاهش ترافیک استفاده می‌شود. هر بار مقدار شمارنده یک واحد اضافه می‌شود.
- روترها شماره دنباله را برای هر مبدأ (Source) نگه می‌دارند. و بسته‌هایی با شماره پایین‌تر از شماره دنباله را دور می‌ریزند.
- **مشکلات موجود در شماره دنباله:**
  - اگر شماره دنباله چرخشی باشد، ابهام پیش می‌آید. راه حل استفاده از یک شماره دنباله ۳۲ بیتی است. با فرض یک بسته وضعیت لینک در هر ثانیه، ۱۳۷ سال طول می‌کشد تا شماره دنباله مجدداً از صفر شروع شود.
  - اگر روتر بنا به هر دلیلی از شبکه خارج شود و مجدد وارد شبکه شود، دنبال کردن شماره دنباله برایش مقدور نخواهد بود. اگر مجدداً از صفر شروع کند، بسته بعدی که ارسال خواهد کرد، چون شماره دنباله کوچکتری دارد، به عنوان کپی محسوب شده و دور ریخته خواهد شد!

نحوه توزیع بسته وضعیت لینک

- **مشکلات موجود در شماره دنباله (ادامه):**
  - با یک بیت خطا در MSB شماره دنباله، مقدار عددی شماره دنباله بزرگ می‌شود. بنابراین بسته‌هایی با شماره دنباله 5 تا 65540 دور ریخته خواهد شد چراکه شماره دنباله فعلی 65540 است.

با فرض ۱۶ بیتی بودن شماره دنباله

00 … 0100 — معادل عدد 4

۱۳ تا

10 … 0100 — معادل عدد 65536+4=65540

۱۲ تا

نحوه توزیع بسته وضعیت لینک

- راه حل همه مشکلات گفته شده، استفاده از فیلد Age است.
- هر روتر، عددی که در این فیلد نوشته شده است را در هر ثانیه یک واحد کاهش می‌دهد. هر زمان که صفر شد، اطلاعات ارسالی آن روتر از این لحظه به بعد معتبر نخواهد بود.
- همچنین از فیلد Age به منظور مدیریت شماره دنباله‌های خطادار استفاده می‌شود. روتر در هر بار Flooding، مقدار Age را یک واحد کاهش می‌دهد تا به صفر برسد. هنگامی که صفر شد، بسته دور ریخته می‌شود. این کار باعث می‌شود که بسته گم نشود و همچنین برای مدت تعریف نشده ای در شبکه وجود نداشته باشد.

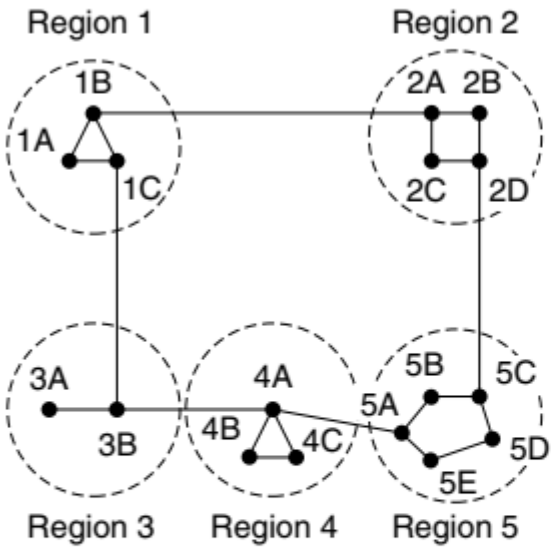
			Send flags			ACK flags			
Source	Seq.	Age	A	C	F	A	C	F	Data
A	21	60	0	1	1	1	0	0	
F	21	60	1	1	0	0	0	1	
E	21	59	0	1	0	1	0	1	
C	20	60	1	0	1	0	1	0	
D	21	59	1	0	0	0	1	1	

The packet buffer for router *B*

مشکل روش مسیریابی وضعیت لینک

- پس از توزیع بسته وضعیت لینک، حال باید کوتاهترین مسیر در هر روتر توسط الگوریتم کوتاهترین مسیر Dijkstra محاسبه شود. به عبارتی sink tree نودها ساخته شود.
- **مشکل اساسی روش مسیریابی وضعیت لینک:**
  - حافظه مورد نیاز برای ذخیره جدول به صورت O(NK)~ تغییر می‌کند. N تعداد روترها و K درجه اتصال شبکه (Network connectivity) است.

الگوریتم مسیریابی سلسله‌مراتبی (Hierarchical Routing)



(a)

Full table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2A	1B	2
2B	1B	3
2C	1B	3
2D	1B	4
3A	1C	3
3B	1C	2
4A	1C	3
4B	1C	4
4C	1C	4
5A	1C	4
5B	1C	5
5C	1B	5
5D	1C	6
5E	1C	5

(b)

Hierarchical table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2	1B	2
3	1C	2
4	1C	3
5	1C	4

(c)

- **ایده اصلی:**
  - **کاهش اندازه جدول مسیریابی و کاهش زمان پردازش مسیریابی**
- **راه حل:**
  - **استفاده از توپولوژی چند لایه‌ای**

- در مثال روبرو، جداول برای نود 1A آورده شده‌است.

الگوریتم‌های مسیریابی همه پخشی (Broadcast Routing Algorithms)

- **ارسال بسته‌های مجزا به همه روترها**
  - ترافیک بالا
  - مبدأ نیاز دارد تا آدرس همه مقصدها را بداند.
  - در حالت کلی استفاده نمی‌شود.
- **روش Flooding**
  - بسته‌های زیادی تولید می‌کند.
- **مسیریابی چند مقصده (Multi-destination Routing)**
  - هر بسته حاوی لیست مقصدهاست
  - روتر، بسته را بین همه نودهای متصل به خروجی خود پخش می‌کند و لیست را آپدیت می‌کند.
  - مجدداً **همه** آدرس مقصد ها باید دانسته شود.

الگوریتم‌های مسیریابی همه پخشی
(Broadcast Routing Algorithms)

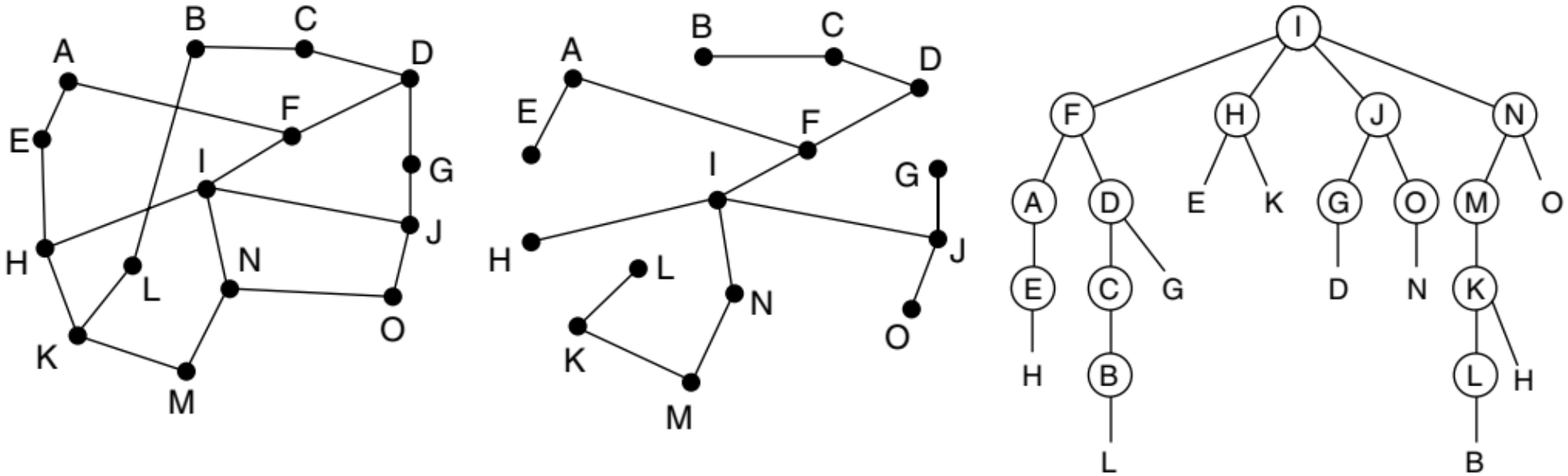
- **استفاده از Sink tree:**
  - هر بسته در عکس مسیر sink tree فرستنده ارسال می‌شود.
  - روش بسیار مناسبی برای پخش بسته‌ها برای همه روترها
  - هر روتر، sink tree همه روترهای دیگر را نمی‌داند. (در روش الگوریتم مسیریابی بردار فاصله این اطلاعات موجود نیست. اما در روش مسیریابی وضعیت لینک موجود است.)

الگوریتم‌های مسیریابی همه پخشی
(Broadcast Routing Algorithms)

- روش Reverse Path Forwarding:
  - عدم نیاز به اطلاعات شبکه
  - پیاده‌سازی آسان
  - اتمام ارسال به صورت اتوماتیک
  - کارآمد
  - عدم نیاز به لیست مقصدها
  - عدم نیاز به مکانیزم خاص برای متوقف کردن فوروارد به روترهای دیگر

الگوریتم‌های مسیریابی همه پخشی
(Broadcast Routing Algorithms)

- روش Reverse Path Forwarding:



Reverse path forwarding.
(a) A network. (b) A sink tree. (c) The tree built by reverse path forwarding.

الگوریتم‌های مسیریابی همه پخشی
(Broadcast Routing Algorithms)

- روش Reverse Path Forwarding:
  - هر روتر، مسیر **نرمال** برای ارسال به روترهای دیگر را می‌داند.
  - هنگامی که بسته‌ای از سمت یک روتر می‌رسد، چک می‌شود که آیا از مسیر نرمال آمده است؟
  - بسته‌هایی را که از مسیر نرمال آمده است، فوروارد می‌کند، اما بسته‌هایی که از مسیر **اشتباه (غیر از نرمال)** آمده‌اند را دور می‌ریزد.
  - **ایده این است که اگر بسته‌ای از مسیر اشتباه بیاید، به احتمال زیاد از مسیر کوتاهترین مسیر نیامده است.**
  - ترافیک ایجاد شده، به مقدار کمی بیشتر از مقدار مورد نیاز است. اما کارآمدی روش قابل قبول است.