CH5 网络层：控制平面

一、路由算法

1. 路由

定义：按照某种指标（传输延迟，所经过的站点数等）找到一条从源节点到目的节点的较好路径

以网络为单位进行路由：路由信息传输、计算和匹配的代价低

前提条件：一个网络所有节点地址前缀相同，且物理上聚集

路由就是：计算网络到其他网络如何走的问题

网络到网络的路由：路由器到路由器之间的路由

一个网络中路由器-主机之间的通信，链路层解决

2. 图抽象

链路的代价：可能总为1，或是链路带宽的倒数，或是拥塞情况的倒数

路由的输入：拓扑、边的代价、源节点

路由的输出：源节点的汇集树

汇集树：此节点到所有其他节点的最优路径形成的树（最优化原则）

路由选择算法就是为所有路由器找到并使用汇集树

3. 路由的原则

- 正确性：算法正确且完整
- 简单性：算法实现简单，延迟低
- 健壮性：算法能够适应通信量和网络拓扑的变化
- 稳定性：产生的路由不应该摇摆
- 公平性：对每一个站点都公平
- 最优性：某一个指标的最优，实际上获取最优代价较高，可以是次优的

4. 路由算法分类

全局：所有的路由器拥有完整的拓扑和边的代价的信息——link state算法

分布式：路由器只是到与它有物理连接关系的邻居路由器，和到相应邻居路由器的代价值——DV算法

静态：路由随时间变化缓慢——非自适应算法，不能适应网络拓扑和通信的变化，路由表是事先计算好的

动态：路由变化很快，周期性地更新，根据链路代价的变化而变化，能适应网络拓扑和通信量的变化

5. Link State算法

配置过程：各点先获得整个网络拓扑，和所有链路代价信息（属于算法和协议，与算法无关）

使用LS算法，计算本站点到其他站点的最优路径（汇集树），得到路由表。

LS路由的基本工作过程：

获取代价信息：

> 获知相邻节点的网络地址
>
> 测量到相邻节点的代价（延迟，开销）
>
> 组装一个LS分组，描述它到相邻节点的代价情况
>
> 将分组通过扩散的方式发到所有其他路由器：可以使用顺序号或年龄字段（老化算法）来解决重复和老化的问题

然后，使用Dijkstra算法找到最优路径（迭代式Dijkstra，每一步能够计算出一个节点的最短路径）

符号标记：

$c(i,j)$：节点i到j链路代价（初始状态下非相邻节点之间的链路代价为∞）

$D(v)$：从源节点到节点V的当前路径代价

$p(v)$：从源节点到节点V的路径前序节点

N'：当前已经知道最优路径的节点集合

工作原理：

- 节点标记：每一个节点使用$(D(v),p(v))$表示
- 临时节点（还没有找到最优解）、永久节点（已经找到最优解）

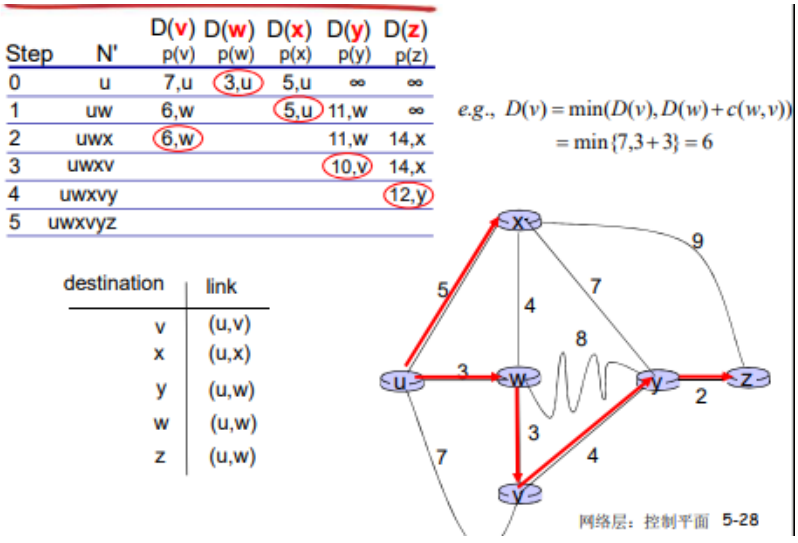
> 初始化，除源节点外所有节点为临时节点，更新临时节点的邻居的距离
>
> 迭代N次：（一共N个临时节点）
>
> > 找到D(v)最小的节点，加入永久节点
> >
> > 使用该节点更新其邻居的$D(v')$：$D(v')=\min\{D(v'),D(v)+c(v,v')\}$
> >
> > 下一轮迭代

算法分析：

复杂度：朴素$O(n^2)$，堆优化$O(n\log n)$

可能的震荡：链路代价 = 链路承载的流量

例子：



Step	N'	D(v) p(v)	D(w) p(w)	D(x) p(x)	D(y) p(y)	D(z) p(z)
0	u	7,u	3,u	5,u	∞	∞
1	uw	6,w		5,u	11,w	∞
2	uwx	6,w			11,w	14,x
3	uwxv				10,v	14,x
4	uwxvy					12,y
5	uwxvyz					

destination	link
v	(u,v)
x	(u,x)
y	(u,w)
w	(u,w)
z	(u,w)

6. DV算法（距离矢量路由选择）

动态路由算法。

各个路由器维护一张路由表：（To，Next，cost）

代价：跳数、延迟、队列长度

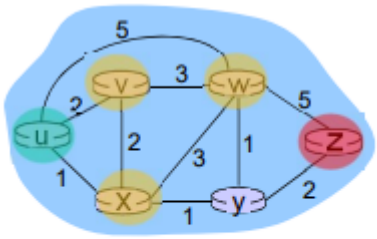
相邻节点间代价的获得：通过实测

路由信息的更新：

> 根据实测，得到本节点到相邻节点的代价（延迟）
>
> 根据各相邻节点生成它们到目标站点B的代价
>
> 计算出本站点A经过各响铃站点到目标站点B的代价
>
> 找到一个最小的代价，和相应的下一个节点Z，到达节点B经过此站点Z，并且代价为A-Z-B的代价

定期测量它到相邻节点的代价、定期交换与相邻节点交换路由表→更新路由表

Bellman-Ford方程：$d_x(y) = \min_{v \in V}\{c(x,v) + d_v(y)\}$（$d_x(y)$为从x到y的最小路径代价）



明显的，$d_v(z) = 5$, $d_x(z) = 3$, $d_w(z) = 3$

由B-F 方程得到：

$$d_u(z) = \min\{c(u,v) + d_v(z),\ c(u,x) + d_x(z),\ c(u,w) + d_w(z)\}$$
$$= \min\{2 + 5,\ 1 + 3,\ 5 + 3\} = 4$$

距离矢量：$\vec{D}_x = [D_x(y) | y \in N]$，

核心思路：

> 每个节点都将自己的距离矢量估计值传送给邻居
>
> 当x从邻居收到DV时，自己运算，更新它自己的距离矢量（使用B-F方程）
>
> $D_x(y)$估计值最终收敛于实际的最小代价值$d_x(y)$

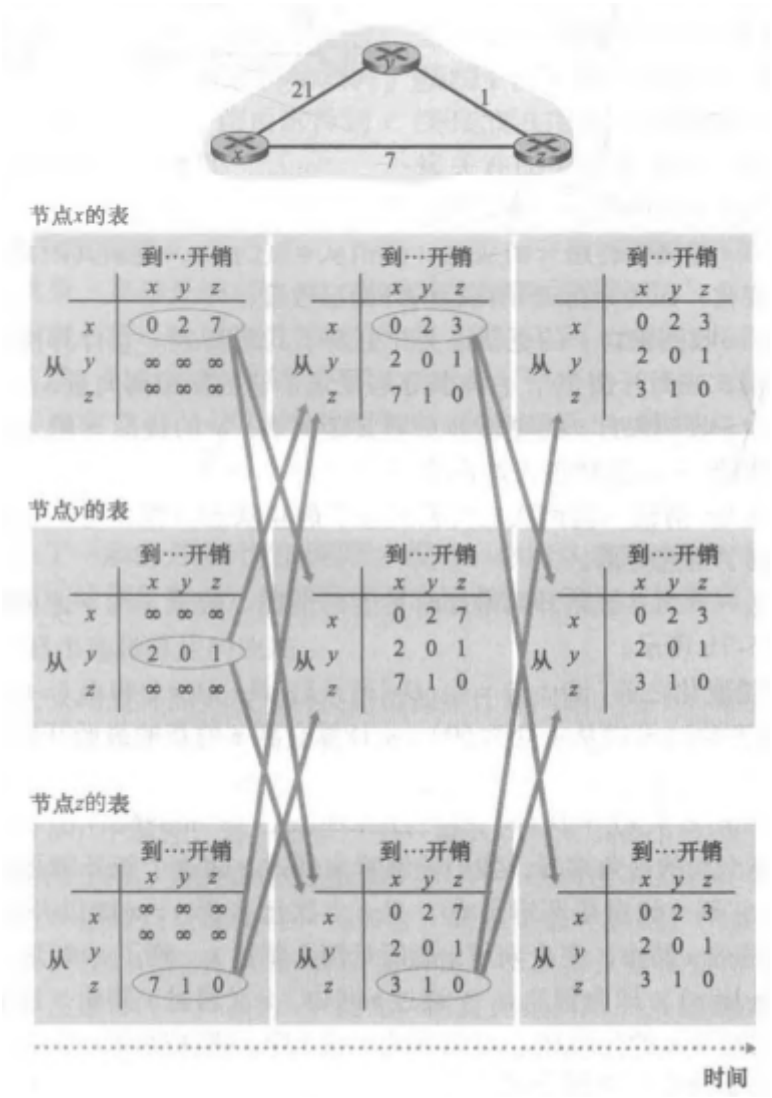
异步式（迭代）：本地链路代价变化和邻居更新DV会触发本地迭代

分布式：每个节点只是在自己的DV改变后向邻居通告

对于每个节点来说：

```
def Node():
    while 1:
        if (accept or CostChanged): # 收到DV报文或者本地链路代价变化
            BellmanFord()           # 重新计算给目标代价估计值
            if(sth changed):        # 任何目标的DV发生变化
                BroadCast(chance)   # 通告邻居
```

每个节点首先计算出自己到其他节点的距离矢量，然后广播给其他节点，其他节点直接复制收到的距离矢量，然后根据这个距离矢量更新自己到其他节点的距离矢量，一旦发生变化，再次进行广播，重复上述过程。



DV的选择环路：

当某一条边开销减少时，不会产生选择环路，并且开销减少的好消息将得到迅速传播；

当某一个条边开销突然增加时，就会产生选择环路：

考虑上述情况算法已经静止，此时$c(x,y)$突变为21，然后y检测到了这种变化：

$D_y(x)=\min\{d_y(x),d_y(z)+D_z(x)\}=\min\{21,1+3\}=4$，这显然是一个错误代价，然后进一步恶性传播至其他节点；

如果$c(x,y)$突变为∞，则会产生**无穷计算。**

增加毒性逆转：

z通过y路由选择到目的地x，则z将通告y，z到x的距离是∞，也就是将通告$D_z(x)=\infty$。只能够解决三个节点及以内的无穷计算问题。

7. LS和DV的比较

复杂度：LS发送$O(VE)$个报文（V,E分别为节点数和链路数）；DV只和邻居交换信息（win）

收敛时间：LS时间复杂性为$O(n^2)$，但是容易产生震荡（win）；DV收敛较慢，且可能存在路由环路和无穷计算问题

健壮性：LS（win）

LS会通告不正确的链路代价，且每个节点只计算自己的路由表，错误信息影响较小

DV会向全网通告不正确的路径代价，每一个节点的路由表会被其他节点使用

二、自治系统内的路由选择

1. AS

AS（Autonomous System），自治系统。

每一个AS由一组通常处在相同管理控制下的路由器组成，一个ISP中的路由器以及互联它们的链路构成一个AS。

每一个ISP既可以构成一个庞大的AS，也可以拆分为若干个互联的AS。

AS内运行的路由选择算法叫作**自治系统内部路由选择协议**。

2. OSPF

OSPF（Open Shortest Path First），开放最短路优先。

是一种链路状态协议，使用洪泛（Flooding，往所有可能连接路径上递送）链路状态信息和Dijkstra最低开销路径算法。

使用OSPF，一台路由器构建了整个AS的完整拓扑图，且向AS内所有其他路由器广播路由选择信息。

3. BGP

BGP（Broder Gateway Protocol），边界网关协议。负责在AS间路由。

BGP为每台路由器提供了完成以下任务的手段：

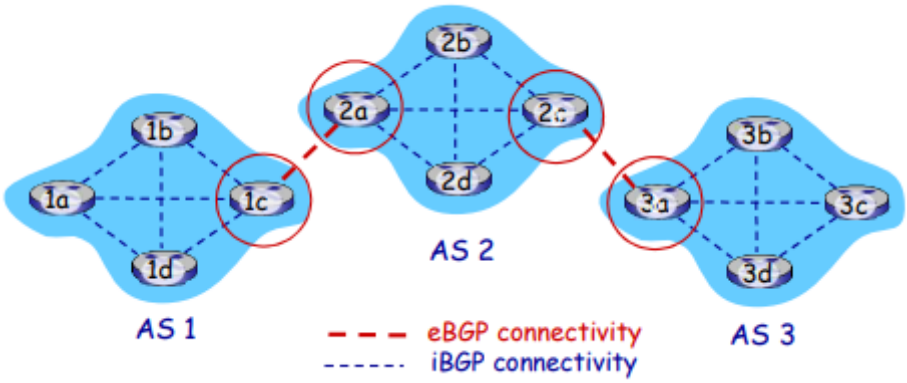
> 从邻居AS获得前缀的可达性信息（BGP允许每个子网向因特网的其余部分）
>
> 确定到该前缀的"最好的路由"

AS内的路由器或为**网关路由器**或为**内部路由器**。

eBGP：从相邻的ASes获得子网可达信息

iBGP：将获得的子网可达信息传遍AS内的所有路由器

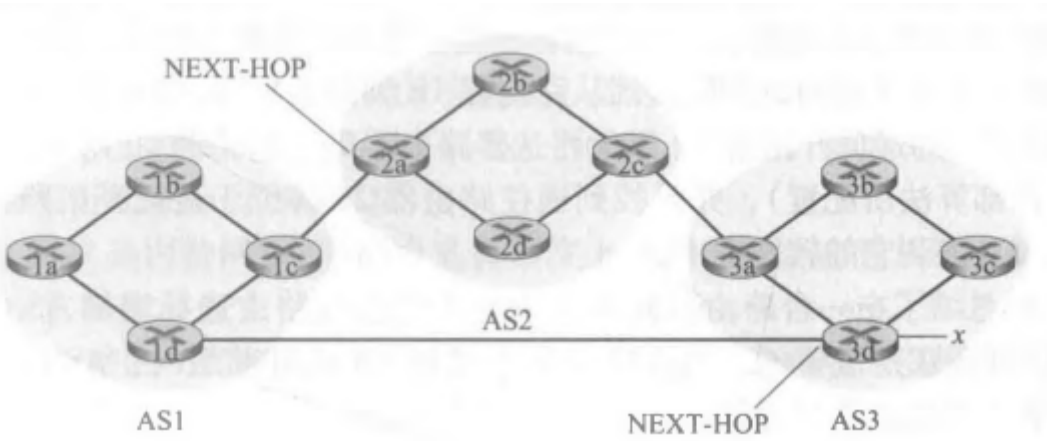
基于DV算法。



上图中，1c、2a、2c、3a是网关路由器，其余均为内部路由器。

BGP连接：2个BGP路由器在一个半永久的TCP连接上交换BGP报文，通告向不同目标子网前缀的“路径”。

BGP属性：前缀+属性称为**路由**、AS-PATH（包含了通告以及通过的AS的列表）、NEXT-HOP（AS-PATH起始的路由器接口的IP地址）



2a左侧接口的IP地址为：AS2 AS3；x

3d左侧接口的IP地址为：AS3；x

BGP报文：

> OPEN：打开TCP连接，认证发送方
>
> UPDATE：通告新路径
>
> KEEPALIVE：在没有更新时保持连接，也用于对OPEN请求确认
>
> NOTIFICATION：报告以前消息的错误，也用来关闭连接

热土豆路由选择：（贪心思路）

通过AS间协议学到经过多个网关可以到达子网x，使用来自AS内部协议的路由选择信息，以决定到达每个网关的最低开销路径的开销。

然后，选择具有最小最低开销的网关，从转发表确定通往最低开销网关的接口I，在转发表中加入(x, I)。

尽可能快地（用最低开销）将分组送出其AS，而不担心其AS外部到目的地的余下部分的开销。

通俗的说，就是试图减小在自己AS中的开销。

上图中，1b会选择通过2a转发（代价为2）而不是3d（代价为3）转发。

路由器选择算法：

对于任何给定的目的地前缀，进入BGP的路由选择算法的输入是*到达某前缀的所有路由的集合*，该前缀是已被路由器学习和接受的。

如果仅仅有一条路径，BGP选择它。

如果有多条路由，调用下述消除规则直到剩下一条路由：

> 路由被指派一个**本地偏好**值作为其属性之一（不是AS-PATH和NEXT-HOP）。由管理员决定，具有最高该值的路由将会被选择。
>
> 从余下的路由中（具有相同本地偏好最高本地偏好值），将选择具有最短AS-PATH的路由。如果该规则是路由选择的唯一规则，将使用DV算法决定路径，距离指标为AS的跳数。

> 从余下的路由中，使用热土豆路由选择。
>
> 如果仍有存留，使用BGP标识符选择路由

通过路径通告执行策略：

供应商之间可以不通告可达路径，使得其他客户感知不到某些可达路径。

内部网关协议与外部网关协议的不同

策略：内部网关（控制通信路径），外部网关（管理者，无需策略）

规模：AS间路由必须考虑规模。

性能：内部网关（策略大于性能），外部网关（关注性能）