**key question**: what is the least-cost path between u and z ?

**routing algorithm**: algorithm that finds that least cost path

**路由选择算法的分类：**

**集中式路由选择算法（centralized routing algorithm）:**

所有路由器知道网络中每条链路的开销和拓扑结构。

也称作链路（Link State)算法，LS算法

**分散式路由选择算法（decentralized routing algorithm）**

每个路由器仅有与其相邻节点的开销信息。通过迭代计算过程以及于相邻节点的信息交换，得出最低开销路径。

Distance-Vector,DV 距离向量算法

**静态路由选择算法（static routing algorithm）**

路由人工进行调整

**动态路由选择算法（dynamic routing algorithm）**

随着流量负载或拓扑发生变化而改变路由选择路径

**负载敏感算法（load-sensitive algorithm）**

链路开销会动态变化，以反映出底层链路的当前拥塞水平

**负载迟钝算法（load-insensitive）**

.......

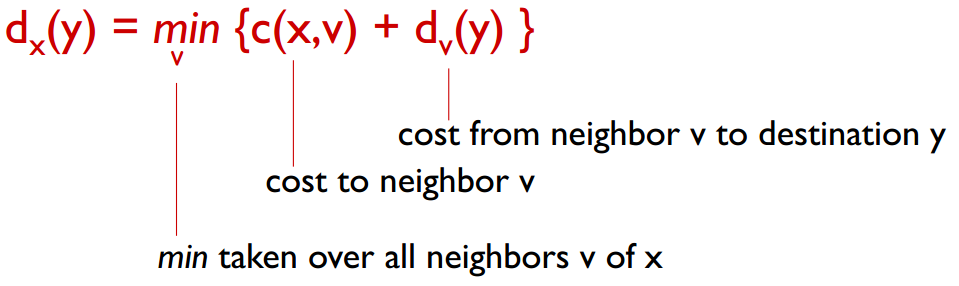
**链路状态路由选择算法**

LS算法（就是Dijikastra）

**DV算法：**

Bellman-Ford equation:

dx(y) := cost of least-cost path from x to y



Dx(y) = estimate of least cost from x to y

x maintains distance vector

Dx = [Dx(y): y є N ] 是节点x的距离向量，该向量是从x到在N中所有其他节点y的开销估计向量。

当节点x从任何一个邻居v接受到一个新距离向量，

它就使用**Bellman-Ford**方程更新自己的距离向量。

**Goal of routing protocol:**

determine “good” paths (equivalently, routes), from sending hosts to

receiving host, through network of routers

将network抽象为graph图

G = {N,E}

N = set of routers

E = set of links

**Routing algorithm classification**

global:

所有路由知道全局的状态信息，比如每条链路的开销和拓扑结构

link state algorithm

decentralized:

router knows physically connected neighbors, link costs to neighbors

iterative process of computation, exchange of info with neighbors

“distance vector” algorithms

static:

routes change slowly over time

dynamic:

periodic update

in response to link cost changes

One link state algorithm is Dijkstra

**路由选择算法**

**Link state broadcast (LS) 链路状态广播算法**

用全局信息，所有节点都具有该网络的统一、完整的视图。

例：Dijkstra算法

**Distance-Vector (DV)距离向量算法**

迭代地、异步的、分布式的算法

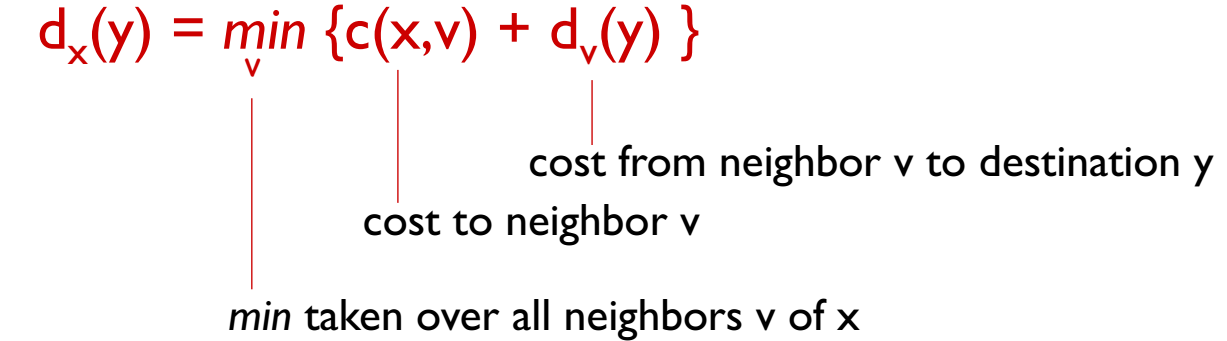
分布式：每个节点都要通过邻居接受某些信息。

迭代：此过程要一直持续到邻居之间无更多信息为止

异步：不要求所有节点步伐一致操作

Bellman-Ford 方程：

dx(y) = min{ c(x,v) + dv(y) }



每个节点x 包含了它的一个距离向量 Dx = [Dx(y): y ∈ N] ，是x到N中所有目的地y的开销估计。

x节点的所有邻居都有距离向量Dv，如果 x 从一个邻居处接收到了一个新距离向量，就用Bellman-Ford 更新自己的距离向量。Dx(y) = min{ c(x,v) + Dv(y) }