3）在 t_1 后某个时间，z 收到 y 的新距离向量，它指示了 y 到 x 的新最低费用是 6。z 知道它能以费用 1 到达 y，因此计算出到 x 的新最低费用 $D_z(x)=\min\{50+0, 1+6\}=7$。因为 z 到 x 的最低费用已增加了，于是它便在 t_2 时刻通知 y 其新费用。

4）以类似方式，在收到 z 的新距离向量后，y 决定 $D_y(x)=8$ 并向 z 发送其距离向量。接下来 z 确定 $D_z(x)=9$ 并向 y 发送其距离向量，等等。

该过程将要继续多久呢？你应认识到循环将持续 44 次迭代（在 y 与 z 之间交换报文），即直到 z 最终算出它经由 y 的路径费用大于 50 为止。此时，z 将（最终）确定它到 x 的最低费用路径是经过它到 x 的直接连接。y 将经由 z 路由选择到 x。关于链路费用增加的坏消息的确传播得很慢！如果链路费用 $c(y, x)$ 从 4 变为 10 000 且费用 $c(z, x)$ 为 9999 时将发生什么样的现象呢？由于这种情况，我们所见的问题有时被称为无穷计数（count-to-infinity）问题。

2. 距离向量算法：增加毒性逆转

刚才描述的特定循环的场景可以通过使用一种称为*毒性逆转*（poisoned reverse）的技术而加以避免。其思想较为简单：如果 z 通过 y 路由选择到目的地 x，则 z 将通告 y，它（z）到 x 的距离是无穷大，即 z 将向 y 通告 $D_z(x)=\infty$（即使 z 实际上知道 $D_z(x)=5$）。只要 z 经 y 路由选择到 x，z 就持续地向 y 讲述这个善意的谎言。因为 y 相信 z 没有到 x 的路径，故只要 z 继续经 y 路由选择到 x（并这样去撒谎），y 将永远不会试图经由 z 路由选择到 x。

我们现在看一下毒性逆转如何解决我们前面在图 4-31b 中遇到的特定环路问题。作为毒性逆转的结果，y 的距离表指示了 $D_z(x)=\infty$。当（x，y）链路的费用在 t_0 时刻从 4 变为 60 时，y 更新其表，虽然费用高达 60，仍继续直接路由选择到 x，并将到 x 的新费用通知 z，即 $D_y(x)=60$。z 在 t_1 时刻收到更新后，便立即将其到 x 的路由切换到经过费用为 50 的直接（z，x）链路。因为这是一条新的到 x 的最低费用路径，且因为路径不再经过 y，z 就在 t_2 时刻通知 y 现在 $D_z(x)=50$。在收到来自 z 的更新后，y 便用 $D_y(x)=51$ 更新其距离表。另外，因为 z 此时位于 y 到 x 的最低费用路径上，所以 y 通过在 t_3 时刻通知 z 其 $D_y(x)=\infty$（即使 y 实际上知道 $D_y(x)=51$）毒化从 z 到 x 的逆向路径。

毒性逆转解决了一般性的无穷计数问题吗？没有。你应认识到涉及 3 个或更多结点（而不只是两个直接相连的邻居结点）的环路将无法用毒性逆转技术检测到。

3. LS 与 DV 路由选择算法的比较

DV 和 LS 算法采用互补的方法来解决路由选择计算问题。在 DV 算法中，每个结点仅与它的直接相连的邻居交谈，但它为其邻居提供了从它自己到网络中（它所知道的）所有其他结点的最低费用估计。在 LS 算法中，每个结点（经广播）与所有其他结点交谈，但它仅告诉它们与它直接相连链路的费用。我们通过快速比较它们各自的属性来总结所学的链路状态与距离向量算法。记住 N 是结点（路由器）的集合，而 E 是边（链路）的集合。

- *报文复杂性*。我们已经看到 LS 算法要求每个结点都知道网络中每条链路的费用。这就要求要发送 $O(|N||E|)$ 个报文。而且无论何时一条链路的费用改变时，必须向所有结点发送新的链路费用。DV 算法要求在每次迭代时，在两个直接相连邻居之间交换报文。我们已经看到，算法收敛所需时间依赖于许多因素。当链路费用改变时，DV 算法仅当在新的链路费用导致与该链路相连结点的最低费用路径发生改变时，才传播已改变的链路费用。