3：　（1）常言道，看一个人怎样，看他有什么朋友就知道了。也就是说，一个人有着越多牛人朋友的人，他是牛人的概率就越大。将这个知识迁移到网页上就是“被越多优质的网页所指的网页，它是优质的概率就越大”

（2）当从网页 A 链接到网页 B 时，就认为“网页 A 投了网页 B 一票”，增加了网页 B 的重要性。最后根据网页的得票数评定其重要性，以此来实现排序算法的优化，而这个重要性的量化指标就是 PageRank 值。

4：下面将用一个简单的例子来讲解 PageRank。

5：（1）首先将 Web 做如下抽象：

（2）显然这个图是强联通的，即从任一节点出发都可以到达另外任何一个节点。

6：（1）被用户访问越多的网页更可能质量越高，而用户在浏览网页时主要通过超链接进行页面跳转，因此我们需要通过分析超链接组成的拓扑结构来推算每个网页被访问频率的高低。

（2）假设当一个用户停留在某页面时，跳转到页面上每个被链页面的概率是相同的。则上图中 A 页面链向 B、C、D，所以一个用户从 A 跳转到 B、C、D 的概率各为 1/3。

7：（1）由于 M 的第一行是 A、B、C、和 D 转移到页面 A 的概率，而向量 v 所包含的分别是 A、B、C、和 D 当前的 rank，因此用 M 的第一行乘以 v 的第一列，所得结果就是页面 A 最新 rank 的合理估计，使用同样的方法可以得出 A、B、C、D 的新 rank 向量 Mv。

（2）然后用 M 再乘以这个新的 rank 向量，又会产生一个更新的 rank 向量。迭代这个过程，可以证明 v 最终会收敛，即 v 约等于 Mv，此时计算停止。最终的 v 就是各个页面的 pagerank 值。例如上面的向量经过几步迭代后，大约收敛在（1/4, 1/4, 1/5, 1/4），这就是 A、B、C、D 最后的 pagerank。

8：（1）以上是在理想情况下进行 PageRank 运算的，但实际中会有很多问题。下面将简要讲到一些处理这些特殊情况的方法。

（2）上面的 PageRank 计算方法假设 Web 是强连通的，但实际上，Web 并不是强连通（甚至不是连通的）。下面看看 PageRank 算法如何处理一种叫做 Dead Ends 的情况。

（3）请问下面哪一个点是 Dead Ends？

（4）上图中的 D 页面不存在外链，因此它是一个 Dead End。

9：（1）而在这个图中，M 第四列将全为 0。有了 Dead Ends，迭代结果将最终归零（需用矩阵论知识证明，比较繁琐）。

10：（1）封闭：若干个网页相互指向对方，但不指向别的网页

（2）这种情况会使得这些网页的 rank 值在计算时不断地累加，最后使得结果不能收敛。而且当红色网页的 rank 值给绿色网页后，绿色网页就将这些 rank 值吞掉了。

11：（1）这种情况和 Rank Sink 类似，我们可以通过这种思想来解决 Rank Sink 问题。具体的做法是加入“逃脱因子”或者称为“心灵转移”（teleporting),它可以让在任一页面浏览的用户有一极小的概率瞬间转移到另外一个随机页面中，而这两个页面之间是没有超链接

的。