9.1.5 面包屑留下的痕迹

但是，记录曾经到过哪些地方并不总是一件容易的事。编写本书时，因特网上有数十亿个不同的 Web 页面，其中还不包括那些由动态网关产生的内容。

如果要爬行世界范围内的一大块 Web 内容，就要做好访问数十亿 URL 的准备。记录下哪些 URL 已经访问过了是件很具挑战的事情。由于 URL 的数量巨大，所以，要使用复杂的数据结构以便快速判定哪些 URL 是曾经访问过的。数据结构在访问速度和内存使用方面都应该是非常高效的。

数亿 URL 需要具备快速搜索结构，所以速度是很重要的。穷举搜索 URL 列表是根本不可能的。机器人至少要用到搜索树或散列表，以快速判定某个 URL 是否被访问过。

数亿 URL 还会占用大量的空间。如果平均每个 URL 有 40 个字符长，而且一个 Web 机器人要爬行 5 亿个 URL（只是 Web 的一小部分），那么搜索数据结构只是存储这些 URL 就需要 20GB 或更多的存储空间（40 字节 /URL × 5 亿个 URL = 20GB）！

这里列出了大规模 Web 爬虫对其访问过的地址进行管理时使用的一些有用的技术。

- *树和散列表*
  复杂的机器人可能会用搜索树或散列表来记录已访问的 URL。这些是加速 URL 查找的软件数据结构。

- *有损的存在位图*
  为了减小空间，一些大型爬虫会使用有损数据结构，比如存在位数组（presence bit array）。用一个散列函数将每个 URL 都转换成一个定长的数字，这个数字在 218
  数组中有个相关的“存在位”。爬行过一个 URL 时，就将相应的“存在位”置位。如果存在位已经置位了，爬虫就认为已经爬行过那个 URL 了。[3]

- *检查点*
  一定要将已访问 URL 列表保存到硬盘上，以防机器人程序崩溃。

- *分类*
  随着 Web 的扩展，在一台计算机上通过单个机器人来完成爬行就变得不太现实了。那台计算机可能没有足够的内存、磁盘空间、计算能力，或网络带宽来完成爬行任务。

注 3：由于 URL 的潜在数量是无限的，而存在位数组中的比特数是有限的，所以有出现冲突的可能——两个 URL 可能会映射到同一个存在位上去。出现这种情况时，爬虫会错误地认为它已经爬行过某个实际未爬行过的页面了。在实际应用中，使用大量的存在位就可以使这种情况极少发生。产生冲突的后果就是爬虫会忽略某个页面。