在文件创建时不得不指定最终文件的大小，这个想法被放弃了。但是，随着CD-ROM、DVD以及其他一次性写光学介质的出现，突然间连续分配又成为一个好主意。所以研究那些具有清晰和简洁概念的老式系统和思想是很重要的，因为它们有可能以一种令人吃惊的方式在未来系统中获得应用。

2. 链表分配

存储文件的第二种方法是为每个文件构造磁盘块链表，如图4-11所示。每个块的第一个字作为指向下一块的指针，块的其他部分存放数据。

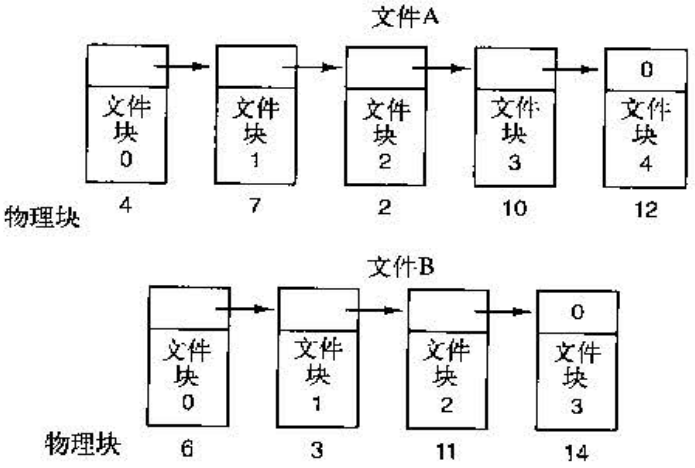

图4-11 以磁盘块的链表形式存储文件

与连续分配方案不同，这一方法可以充分利用每个磁盘块。不会因为磁盘碎片（除了最后一块中的内部碎片）而浪费存储空间。同样，在目录项中，只需要存放第一块的磁盘地址，文件的其他块就可以从这个首块地址查找到。

另一方面，在链表分配方案中，尽管顺序读文件非常方便，但是随机存取却相当缓慢。要获得块n，操作系统每一次都必须从头开始，并且要先读前面的$n-1$块。显然，进行如此多的读操作太慢了。

而且，由于指针占去了一些字节，每个磁盘块存储数据的字节数不再是2的整数次幂。虽然这个问题并不是非常严重，但是怪异的大小确实降低了系统的运行效率，因为许多程序都是以长度为2的整数次幂来读写磁盘块的。由于每个块的前几个字节被指向下一个块的指针所占据，所以要读出完整的一个块，就需要从两个磁盘块中获得和拼接信息，这就因复制引发了额外的开销。

3. 在内存中采用表的链表分配

如果取出每个磁盘块的指针字，把它放在内存的一个表中，就可以解决上述链表的两个不足。图4-12表示了图4-11所示例子的内存中表的内容。这两个图中有两个文件，文件A依次使用了磁盘块4、7、2、10和12，文件B依次使用了磁盘块6、3、11和14。利用图4-12中的表，可以从第4块开始，顺着链走到最后，找到文件A的全部磁盘块。同样，从第6块开始，顺着链走到最后，也能够找出文件B的全部磁盘块。这两个链都以一个不属于有效磁盘编号的特殊标记（如-1）结束。内存中的这样一个表格称为文件分配表（File Allocation Table，FAT）。

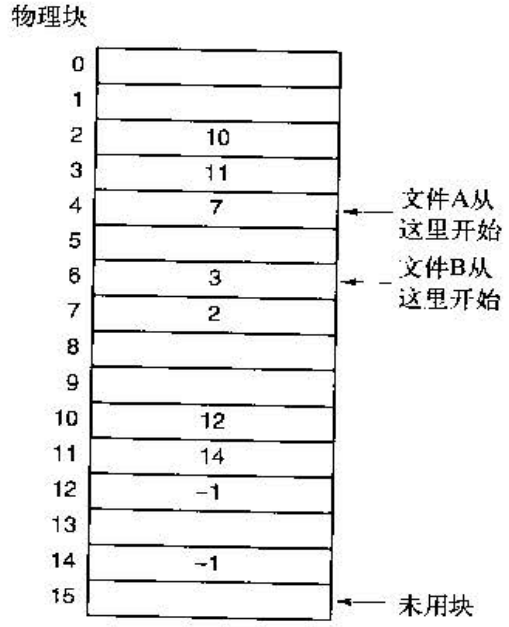

图4-12 在内存中使用文件分配表的链表分配

按这类方式组织，整个块都可以存放数据。进而，随机存取也容易得多。虽然仍要顺着链在文件中查找给定的偏移量，但是整个链表都存放在内存中，所以不需要任何磁盘引用。与前面的方法相同，不管文件有多大，在目录项中只需记录一个整数（起始块号），按照它就可以找到文件的全部块。

这种方法的主要缺点是必须把整个表都存放在内存中。对于200 GB的磁盘和1KB大小的块，这张表需要有2亿项，每一项对应于这2亿个磁盘块中的一个块。每项至少3个字节，为了提高查找速度，有时需要4个字节。根据系统对空间或时间的优化方案，这张表要占用600MB或800MB内存，不太实用。很显然FAT方案对于大磁盘而言不太合适。