COSC1112/1114: Operating Systems Principles

Tutorial 09 (week 10)

1. **The open-file table is used to maintain information about files that are currently open. Should the operating system maintain a separate table for each user or just maintain one table that contains references to files that are being accessed by all users at the current time? If the same file is being accessed by two different programs or users, should there be separate entries in the open file table?**

   <u>Answer:</u>
   By keeping a central open-file table, the operating system can perform the following operation that would be infeasible otherwise. Consider a file that is currently being accessed by one or more processes. If the file is deleted, then it should not be removed from the disk until all processes accessing the file have closed it. This check can be performed only if there is centralized accounting of number of processes accessing the file. On the other hand, if two processes are accessing the file, then two separate states need to be maintained to keep track of the current location of which parts of the file are being accessed by the two processes. This requires the operating system to maintain separate entries for the two processes.

2. **Provide examples of applications that typically access files according to the following methods:**
   - **Sequential**
   - **Random**

   <u>Answer:</u>
   - Applications that access files sequentially include word processors, music players, video players, and web servers.
   - Applications that access files randomly include databases, video and audio editors.

3. **If the operating system were to know that a certain application is going to access the file data in a sequential manner, how could it exploit this information to improve performance?**

   <u>Answer:</u>
   When a block is accessed, the file system could prefetch the subsequent blocks in anticipation of future requests to these blocks. This prefetching optimization would reduce the waiting time experienced by the process for future requests.

4. **Give an example of an application that could benefit from operating system support for random access to indexed files.**

   <u>Answer:</u>
   An application that maintains a database of entries could benefit from such support. For instance, if a program is maintaining a student database, then accesses to the database cannot be modelled by any predetermined access pattern. The access to records is random and locating the records would be more efficient if the operating system were to provide some form of tree-based index.

5. **Contrast the performance of the three techniques for allocating disk blocks (contiguous, linked, and indexed) for both sequential and random file access.**

<u>**Answer:**</u>

- Contiguous Sequential - Works very well because the file is stored contiguously.
- Contiguous Random -Works very well as you can easily determine the adjacent disk block containing the position you wish to seek to.

- Linked Sequential - Satisfactory because you are simply following the links from one block to the next.
- Linked Random - Poor as it may require following the links to several disk blocks until you arrive at the intended seek point of the file.

- Indexed Sequential - Works well as sequential access simply involves sequentially accessing each index.
- Indexed Random - Works well as it is easy to determine the index associated with the disk block containing the position you wish to seek to

6. **Consider a file system on a disk that has both logical and physical block sizes of 512 bytes. Assume that the information about each file is already in memory. For each of the three allocation strategies (contiguous, linked, and indexed), answer these questions:**
   a) **How is the logical-to-physical address mapping accomplished in this system? (For the indexed allocation, assume that a file is always less than 512 blocks long.)**
   b) **If we are currently at logical block 10 (the last block accessed was block 10) and want to access logical block 4, how many physical blocks must be read from the disk?**

<u>**Answer:**</u>

Let *Z* be the starting file address (block number).

- **Contiguous**. Divide the logical address by 512 with *X* and *Y* the resulting quotient and remainder respectively.
  a) Add *X* to *Z* to obtain the physical block number. *Y* is the displacement into that block.
  b) 1
- **Linked**. Divide the logical physical address by 511 with *X* and *Y* the resulting quotient and remainder respectively.
  a) Chase down the linked list (getting *X* + 1 blocks). *Y* + 1 is the displacement into the last physical block.
  b) 4
- **Indexed**. Divide the logical address by 512 with *X* and *Y* the resulting quotient and remainder respectively.
  a) Get the index block into memory. Physical block address is contained in the index block at location *X*. *Y* is the displacement into the desired physical block.
  b) 2