|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| File allocation method | 名稱 | 定義 | 優點 | 缺點 | 範例 |
| Contiguous allocation | 若file的大小是n個block則OS必須在disk中找到n個連續的free block，才可配置給他 | 1. 平均seek time短(連續的block大多落在同一條track) 2. 支援sequential access及random access 3. 循序存取速度快於linked allocation | 1. 有external fragmentation (解決法:磁碟重組) 2. File大小不易動態擴增 3. 建file前須事先宣告大小 |  |
| Linked allocation | 若file大小= n個block則OS只要在disk中找到n個free block(不一定連續)即可配置，且allocated block間以link串連 | 1. 無external fragmentation 2. File大小可動態擴增 3. 建file前無須事先宣告大小 | 1. 平均seek time長(block大多散落在不同track) 2. 不支援random access 3. 循序讀取的time較長(要去Disk讀取資訊) 4. 可靠度差(link斷裂，file block無法access) |  |
| Index allocation | 若file大小= n個block則OS除了在disk中找到n個free block配置外，另外需額外配置index block給他，內容紀錄allocated block的NO. | 1. 無external fragmentation 2. 支援sequential access及random access 3. File大小可動態擴增 4. 建file前無須事先宣告大小 | 1. index block占用額外空間 2. 若file block數很多，則index block無法容納所有配置區塊NO.，須有解決法 |  |

* 每個file配置法皆會有internal fragmentation

|  |  |  |  |
| --- | --- | --- | --- |
| FAT method | 是Linked allocation的變形，差別allocated block之間的linking info.並非存在disk內，而是用一個表格FAT(file allocation table)去紀錄linking info. | 改善   1. 存取block的時間快很多 2. Data block完全用於存放data 不需記錄linking info. 3. 循序存取速度也較快 |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 解決single index block無法容納所有 allocated block的NO. | 方法 | 優點 | 缺點 | 範例 |
| 使用多個index block彼此以link list串連 |  | 1. 存取任一data block的平今I/O次數增加，時間變長 2. 有些許喪失random access的感覺 |  |
| 使用multilevel index structure | 1. 任一data block的存取其I/O次數皆相同，縮短時間 2. 大型file適用 | 1.小型file不適用 |  |
| UNIX的I-node structure  (大小型file皆適用) |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 名稱 | 定義 | 分析 | 範例 |
| FCFS  (first come first service) | 最早到達的track request優先服務 | 1. 公平，不會有starvation 2. 簡單容易實施 3. 效果不好(軌道移動多seek time長) |  |
| SSTF  (shortest seek time track first) | 距離目前磁頭最近的先服務 | 1. 平均seek time較短 2. 不公平，會有starvation |  |
| SCAN | 磁頭來回掃描，提供雙向服務，磁頭到track的開端或盡頭才會折返 | 1. 效能上可接受，適用在負擔重(軌道track多的情況) 2. 磁頭必須遇到track盡頭或開端才會折返，浪費不必要的seek time(用look法則改善) 3. 在某些時刻對某些track不公平(用C-SCAN改善) |  |
| C-SCAN (C-circular) | 類似SCAN但是只提供單向服務 | 爭議：拉回track是否計算? |  |
| LOOK法則 | 和SCAN類似，差別磁頭只服務完該方向的最後一個request立即折返，不用到盡頭 |  |  |
| C-LOOK | 類似LOOK但是只提供單向服務 | 拉回要計算 |  |

* Disk scheduling法則無最佳與最差