OPERATING SYSTEMS

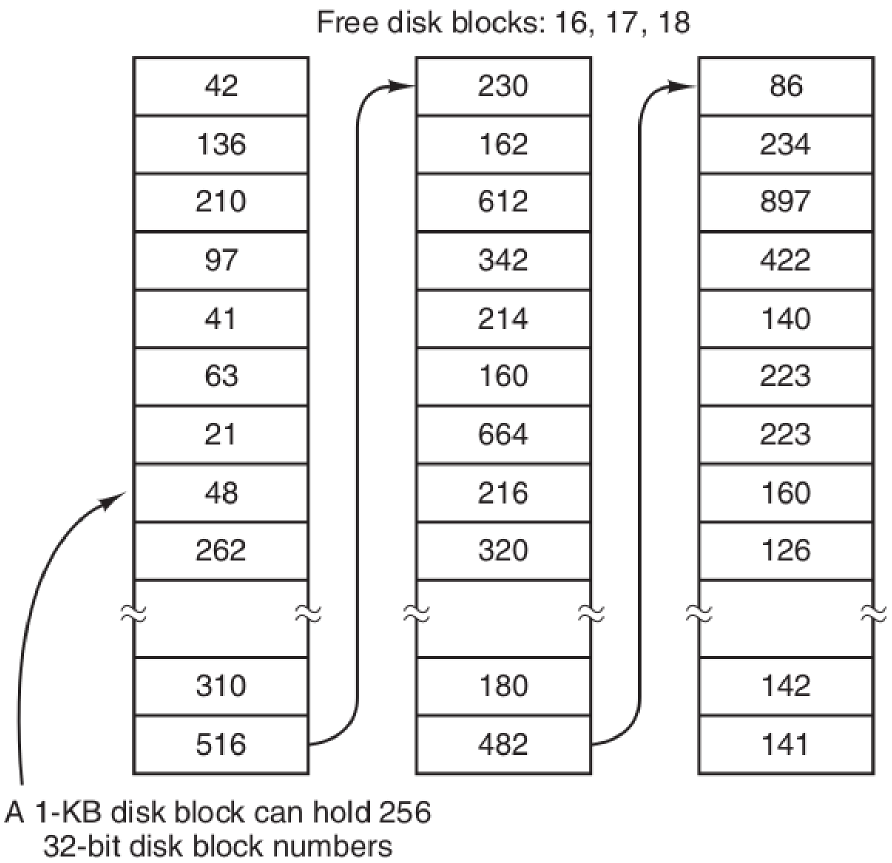
by
Marwa Yusuf

Lecture 18
Sunday 10-1-2021

Chapter 4 (4.4.1 to 4.4.3)
File Systems

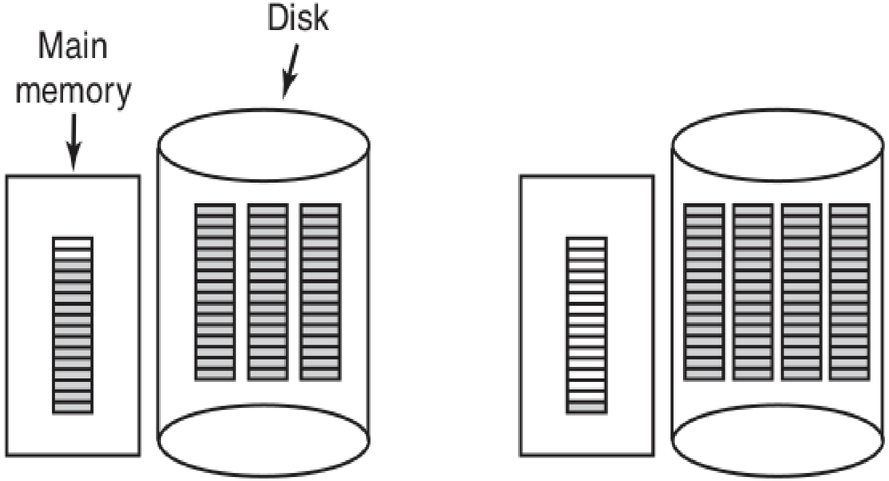
Keeping Track of Free Blocks using Linked Lists

- A linked list of disk blocks, each holding free disk blocks' numbers + a pointer to the next.
- 1KB disk block, 32 bit disk block number → each block holding 255 free blocks' numbers + a pointer.
- 1 TB disk → 1 billion blocks → about 1 million blocks for the linked list.
- Free list stored on free blocks → free storage.
- If free blocks are consecutive, the address of the first and the number could be stored (an optimization). Optimally, if disk is wholly free, only the first address and size are stored. If disk highly fragmented, it won't be efficient.



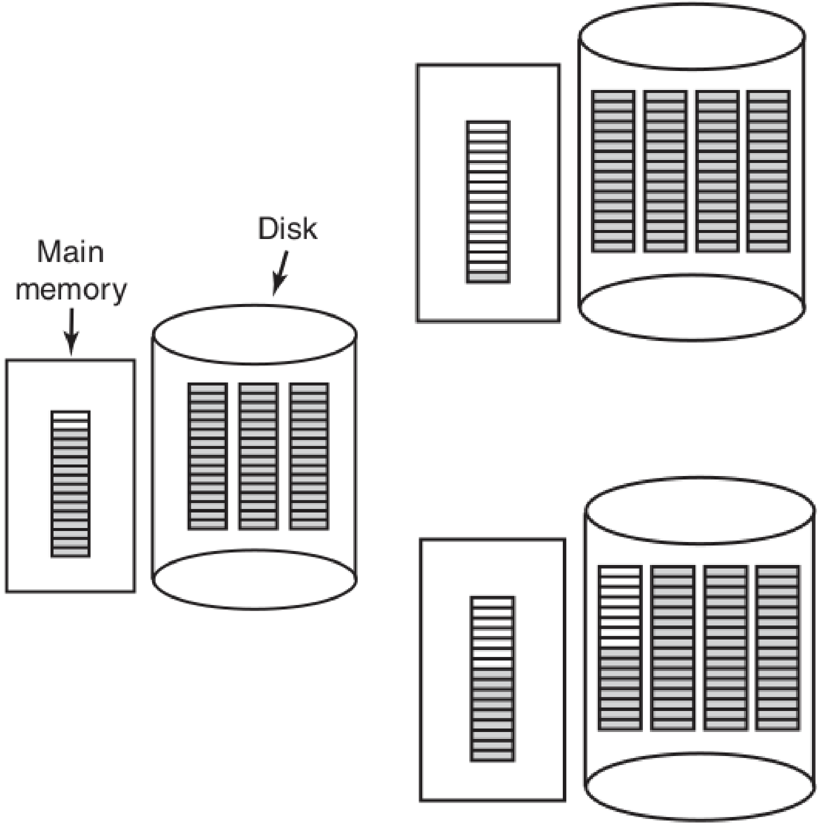
Keeping Track of Free Blocks using Linked Lists

- One block of pointers is kept in memory at a time.
  - When a block is needed, it is taken from the pointers block.
  - When it runs out a new pointer block is brought into memory.
  - When a file is deleted, its blocks' addresses are added to the current pointers' block.
  - When it fills up, it is written back.
  - When the block of pointers is almost empty, a series of short-lived temporary files can cause a lot of disk I/O



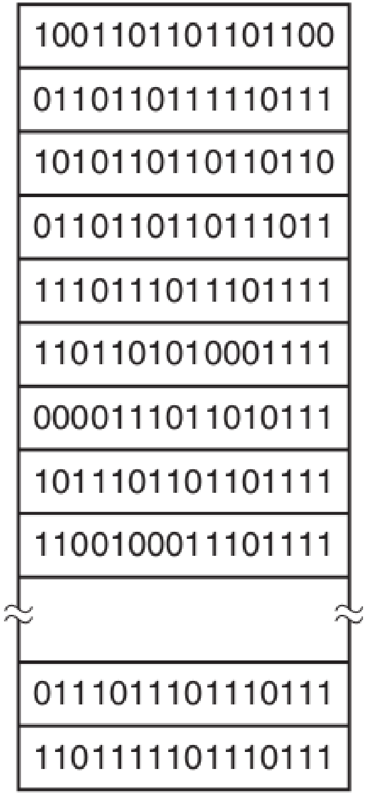
Keeping Track of Free Blocks using Linked Lists

- When the block of pointers is almost empty, a series of short-lived temporary files can cause a lot of disk I/O.
- The sol: when the pointers block fills up, it is written back, and half full block is read in.
  - The idea is to keep the one in memory (only) half full to handle file create and remove without much disk I/O.



Keeping Track of Free Blocks using bitmap

- *N* blocks disk → *n* bits map.
- 1 TB disk → 130,000 1-KB blocks for bitmap.
  - Bitmap is smaller: 1 bit vs 32 bits
  - Except when?
- Also, keep only one block in memory going for disk only when full or empty.
  - Allocated disk blocks will be close together, reducing disk head movement.
- Bit-map is fixed size (relative to disk size), it can be paged if the kernel is partially paged.



1001101101101100
0110110111110111
1010110110110110
0110110110111011
1110111011101111
1101101010001111
0000111011010111
1011101101101111
1100100011101111
0111011101110111
1101111101110111

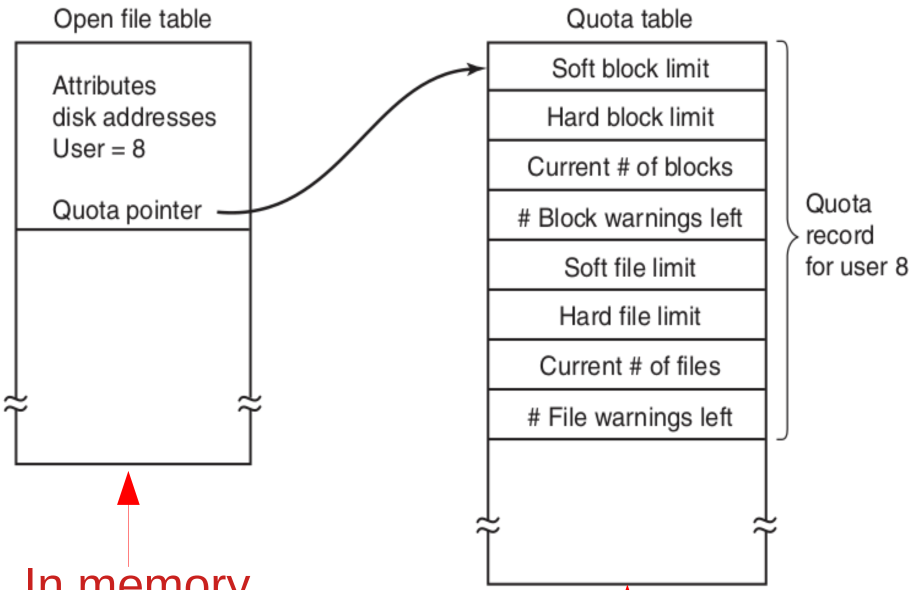
A bitmap

Keeping Track of Free Blocks

- As an OS designer which technique you will use? Based on what criteria and information?
  - Studies?

Disk Quotas

- OS may enforce specific max quota for users in a multi-user system.
- Any increase in file size is charged to the owner’s quota. (owner not opener).
  - Check against hard (cannot be exceeded) and soft (can be exceeded) limits. Trying to exceed hard limit → error.
  - A similar check is made for # of files (to protect available i-nodes).
  - At log-in, the soft limit is checked, generating warning if exceeded, decrement # of warnings.
  - If a user ignores too many warnings, he cannot login without a discussion with the admin.
  - A user can exceed soft limit in a session, but should fix it before logging out.
- When all files are closed, the record is written back to quota file.



In memory

In memory, extract from quota file on disk

File System Backups

- A lost file systems is difficult and in some times impossible.
- A lost file system may be a disaster.
- One solution offered by file system that – might – help is to create backup.
- While individuals may bypass backup – until they face the disaster – companies backup periodically (daily), usually to a tape.
- However, backup is not that easy.

File System Backups

- Backups are made to:
  - Recover from disaster: like disk crash, fire, flood … etc.
  - Recover from mistakes: like removing files and needing them again.
    - That’s why there is a trash bin. (But you always use shift-delete, right? Too bad)
- Making backup consumes time and space. Hence, efficiency is needed.

File System Backups Issues

1) Backup the entire file system? What about executable? Drivers? Temporaries?
   - Mark folders that need backup.
2) Backup everything every time? What if it is not changed?
   - **Incremental dump**: full dump (weekly or monthly), daily partial dump for changed files only, since last full dump (or last dump).
   - Recovery more complicated: recover full, then recover partials in reverse order.
3) Compress data?
   - What if there is a bad spot on the tape?
4) Can you backup active FS? It is not consistent.
   - Make FS offline at night for some hours to backup. Is it possible?
   - One sol: take snapshot of critical data, require future changes to copy, not update in place (as if the data is frozen at snapshot. Then backup later when there is time.
5) Non-technical security issue:
   - You need to guard the backup tapes.
   - You may need to store them in an off-site place (otherwise the same disaster that eats the computers will take the tapes). Then you will need to guard the other place also.
   - Fortunately, this is not our role now.

File System Backups Strategies
Physical Dump

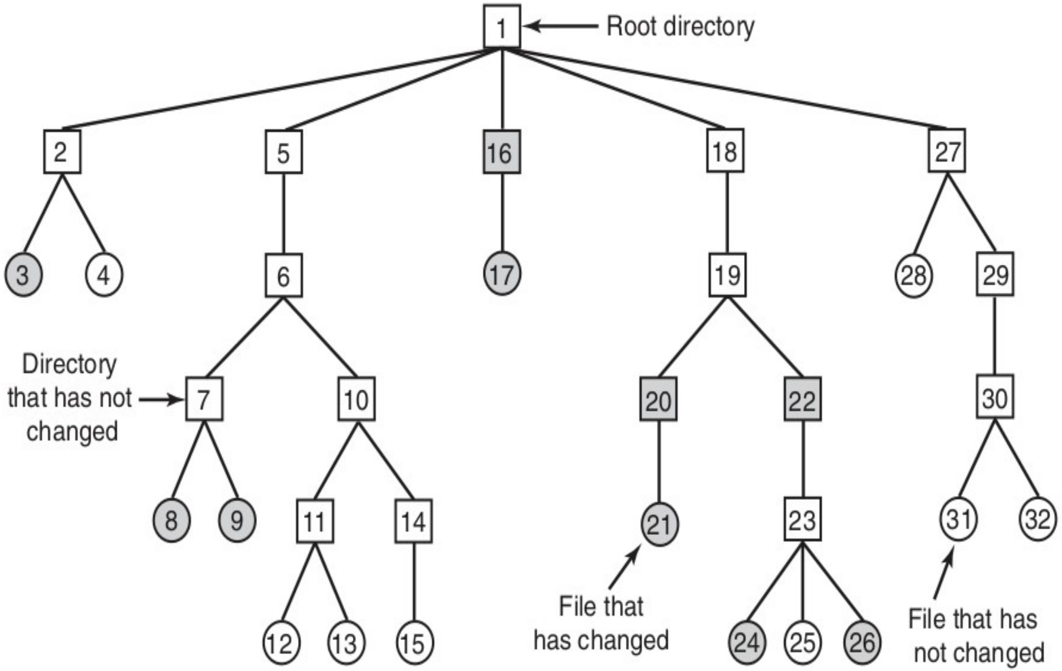
- Start at block 0 till the last block, copy everything. PERFECT!
- Advantages:
  - Simplicity.
  - Great speed.
- Problems:
  - It will copy free blocks!
    - If it reads the free blocks map, it can avoid this. But it will need block numbers in the tape.
  - It will copy bad blocks!
    - If bad blocks are handled by the disk controller, no problem. But if handled by OS (putting them in a special file), they will be copied. They must be known for the backup program to avoid disk read errors.
  - Systems may have specific data that need not backup. Like hibernation info in Windows.
  - Unable to skip selected directories, make incremental dumps and restore individual files upon request.

File System Backups Strategies Logical Dump

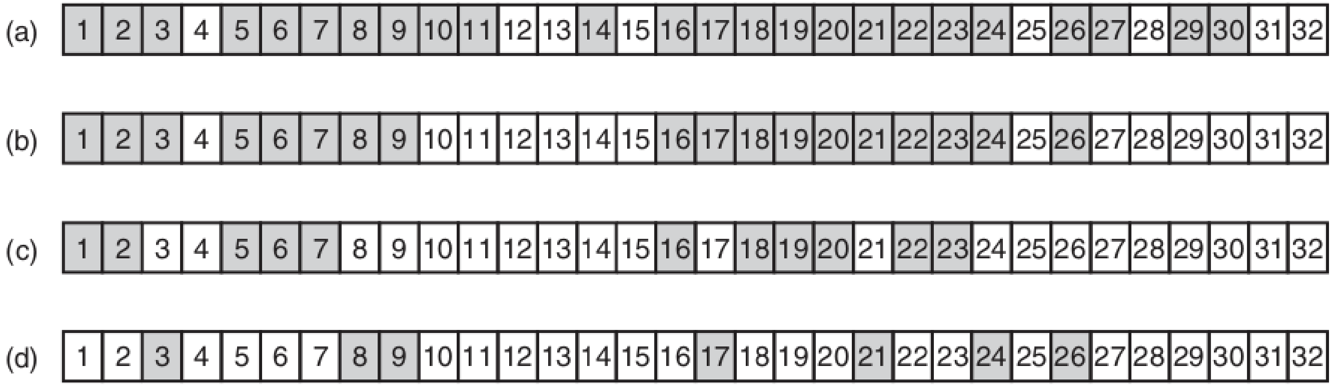
- Dump specified directories (and files) changed since some base date (installation if full dump or last dump). Hence, we can restore a specific file or directory.
- The most commonly used.

File System Backups Strategies Logical Dump

- An ex. Algorithm.
- It dumps all directories in the path of a modified file or directory:
  - To make it possible to restore to a fresh FS on a different computer (transporter).
  - To make it possible to incrementally restore. (if a file is removed and is to be restored, all its parents info must be restored)



File System Backups Strategies
Logical Dump



- Bitmap indexed by i-node #, several bits/i-node
- Phase 1: mark every directory, and every changed file.
- Phase 2: unmark directories with no changes in their children.
  - 1 & 2 can be combined for speed.
- Phase 3: dump marked directory, prefix them with attributes.
- Phase 4: dump marked files, prefix them with attributes.

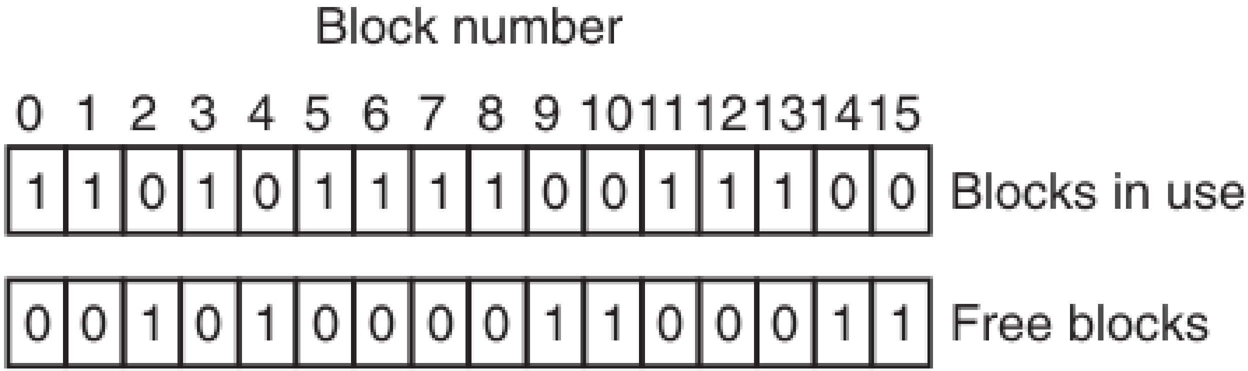
File System Backups Strategies Logical Dump

- Restoring:
  - Create empty FS on disk.
  - Start with the last full dump:
    - Restore directories → skeleton.
    - Restore files.
  - Follow with incremental dumps, starting from oldest.
- Issues:
  - Free block list not dumped: need to be created (can be done as a compliment to all the files).
  - Links: restore a linked file once, and directories refer to it.
  - In UNIX a file may have holes (may be very large like in core files). Should be handled.
  - Special files (ex: pipe) should not be dumped.

File System Consistency

- Read blocks, modify, write.
- What if a sys. crash happens in the middle → inconsistent state.
  - Specially when dealing with i-node, directory or free list blocks.
- Sol: a tool to check consistency at boot (specially after crash): fsck (UNIX) & sfc (Windows).

Fsck: blocks

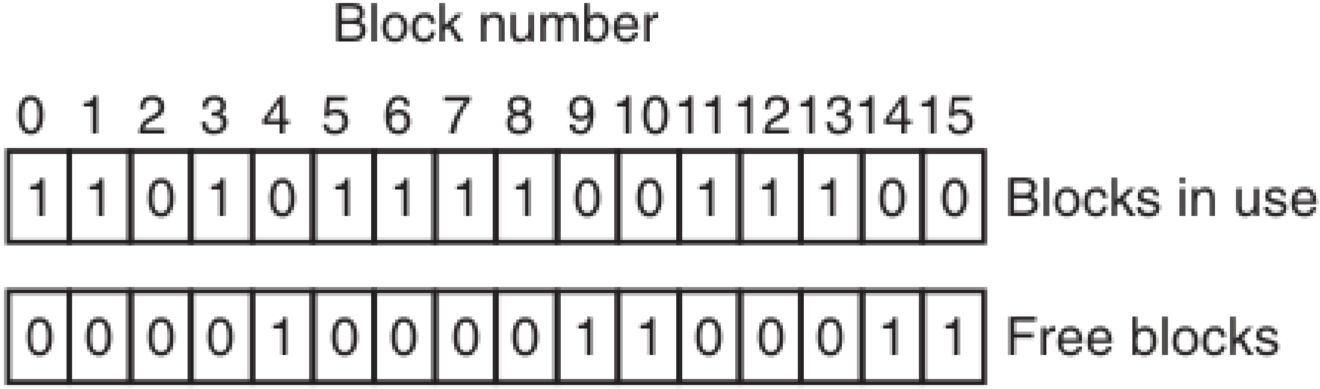


Block number	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Blocks in use	1	1	0	1	0	1	1	1	1	0	0	1	1	1	0	0
Free blocks	0	0	1	0	1	0	0	0	0	1	1	0	0	0	1	1

(a)

- Consistent State.

Fsck: blocks



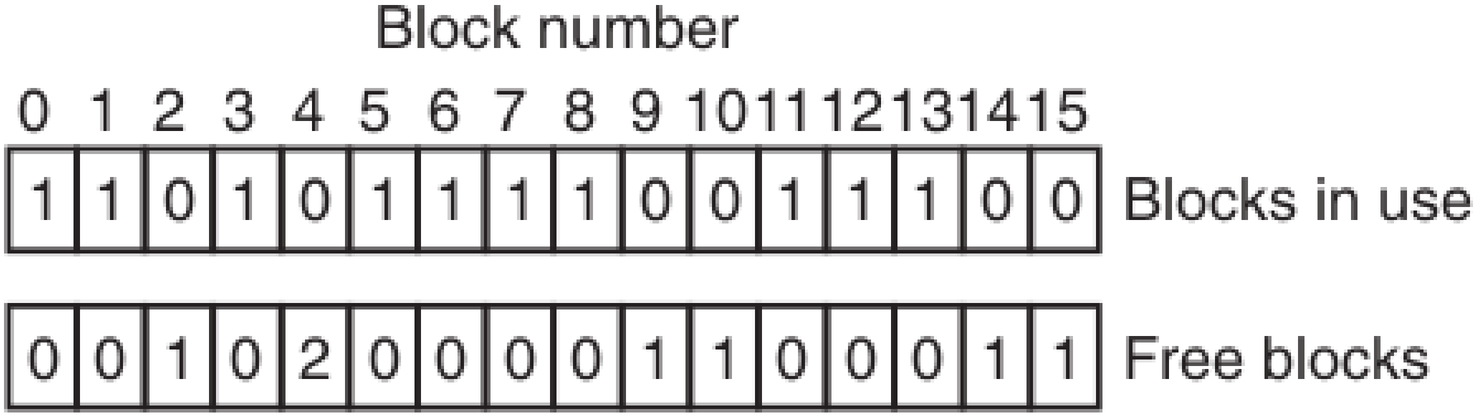
Block number

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Blocks in use	1	1	0	1	0	1	1	1	1	0	0	1	1	1	0	0
Free blocks	0	0	0	0	1	0	0	0	0	1	1	0	0	0	1	1

(b)

- Block 2 **missing block**: no real harm, wasted space, add to free list.

Fsck: blocks

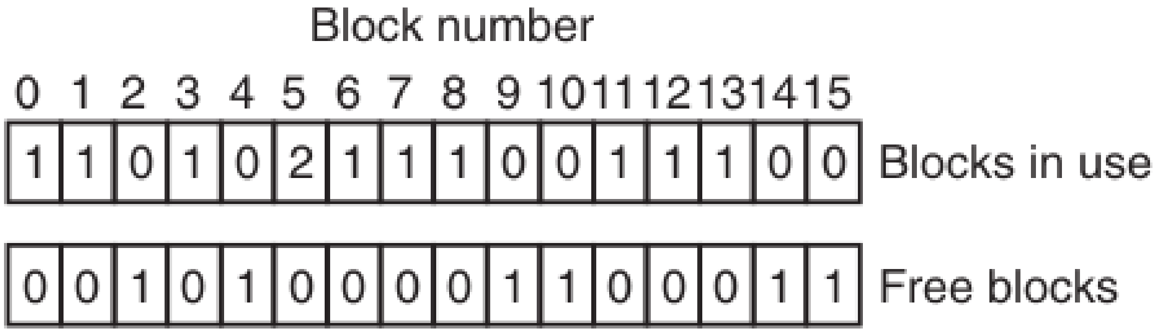


Block number

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Blocks in use	1	1	0	1	0	1	1	1	1	0	0	1	1	1	0	0
Free blocks	0	0	1	0	2	0	0	0	0	1	1	0	0	0	1	1

- Block 4 duplicate: can happen only with lists not bitmap, rebuild free list.

Fsck: blocks



Block number

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Blocks in use	1	1	0	1	0	2	1	1	1	0	0	1	1	1	0	0
Free blocks	0	0	1	0	1	0	0	0	0	1	1	0	0	0	1	1

(d)

- Block 5 in 2 files:
  - If 1 file removed, block in use and free at the same time.
  - If 2 files removed, block in free twice.
  - Copy the block to another block and add to one file. One of them is incorrect.
    - Structure is okay, report the problem to the user.

Fsck: directories

- Table of counters, counter/file.
- Start at root, recursively descend, increment file counter for each i-node found. (symbolic counts do not count).
- A list indexed by i-node number → how many directories containing file. Compare these with link number in i-node:
  - Number in i-node is larger → wasted space, correct number.
  - Number in i-node is lower → directory may point to unused i-node, correct number.

File System Consistency

- The two checks may be combined for efficiency
- Other possible checks like:
  - If i-node number larger than # of i-nodes on disk → directory is damaged.
  - 0007 permission (outsiders are more powerful than owner → at least report).
  - Directories with so many entries (>1000)??
  - Files owned by superuser located in user folders with special privileges.
- And all technically legal but suspicious situations that requires at a least reporting.
- What about errors of the user himself?
  - rm *.o vs rm * .o