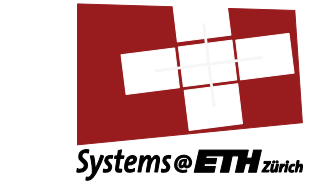


Computer Systems

Exercise 9

Last Exercsie

Systems@ETH zürich

Distributed Computing

1.1 Segmentation

Consider the following segment table:

Segment	Base	Length
0	219	600
1	2300	14
2	90	100
3	1327	580
4	1952	96

What are the physical addresses for the following logical addresses?

a) 0,430

b) 1,10

c) 2,500

d) 3,400

e) 4,112

1.) 430 < 600? ✓

2.) 219 + 430 = 649

Last Exercsie

Systems@ETH zürich

Distributed Computing

1.1 Segmentation

Consider the following segment table:

Segment	Base	Length
0	219	600
1	2300	14
2	90	100
3	1327	580
4	1952	96

What are the physical addresses for the following logical addresses?

a) 0,430

b) 1,10

c) 2,500

d) 3,400

e) 4,112

1.) 500 < 100?

Last Exercise

Systems@ETH zürich

Distributed Computing

1.2 Paging

Consider a paging system with the page table stored in memory.

a) If a memory reference takes 200 nanoseconds, how long does a paged memory reference take if there is no TLB?

b) If we add a TLB and 75% of all page-table references are found in the TLB, what is the average memory reference time when a TLB access takes 2 nanosecond?

c) A typical program has 20% memory instructions. Assume there are 5% data TLB misses, each requiring 100 cycles to handle. Furthermore each instruction requires 1 cycle to execute and each memory operation in the cache takes 1 cycle. Also, 10% of the data accesses are cache misses each of which takes 15 cycles. How long would it take to execute 1000 instructions?

a) No TLB: one access for page, one access for page table: 200 + 200 = 400

b) (2+200)*0.75 + (2+200+200)*0.25 = 252ns

c) Idea: calculate the overhead

```
# cache misses = 1000 * 20% * 10% = 20
# TLB misses = 1000 * 20% * 5% = 10

 total cycles = 1000 cycles + (cache miss overhead) + (TLB miss overhead)
              = 1000 + (20 * 14) + (10 * 99) = 1000 + 280 + 990 = 2270 cycles
```

Last exercise

Systems@ETH zürich

Distributed Computing

1.3 Virtual Memory

Consider a paged virtual address space composed of 32 pages of 2 KB each which is mapped into a 1 MB physical memory space.

a) What is the format of the logical address; i.e., which bits are the offset bits and which are the page number bits? Explain.

b) What is the length and width of the page table, disregarding the access right bits?

a) offset: each byte needs to get accessed
- 2KB = 2^{11} bytes -> 11 offset bits

page number: each page needs to get accessed
- 32 pages = 2^5 pages -> 5 page number bits

b) length of page table: for each page one entry
- 32 entries

width of page table: each entry contains one physical address
- size of physical address: 1MB = 2^{20} bytes. So 20 bits to represent address

Last exercise

Systems@ETH zürich

Distributed Computing

1.4 Page Replacement

Consider the following page access pattern:

1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6

How many page faults would occur for the following replacement algorithms, assuming one, two, three, four, five, six, or seven frames?

a) LRU replacement

b) FIFO replacement

c) Optimal replacement

1

Last exercise

Systems@ETH zürich

Distributed Computing

1.4 Page Replacement

Consider the following page access pattern:

1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6

How many page faults would occur for the following replacement algorithms, assuming one, two, three, four, five, six, or seven frames?

a) LRU replacement

b) FIFO replacement

c) Optimal replacement

Last exercise

Systems@ETH zürich

Distributed Computing

1.4 Page Replacement

Consider the following page access pattern:

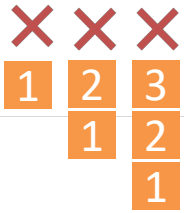
1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6

How many page faults would occur for the following replacement algorithms, assuming one, two, three, four, five, six, or seven frames?

a) LRU replacement

b) FIFO replacement

c) Optimal replacement



Last exercise

Systems@ETH zürich

Distributed Computing

1.4 Page Replacement

Consider the following page access pattern:

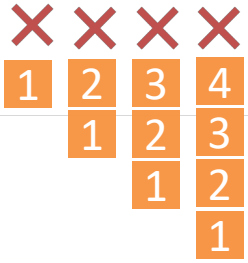
1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6

How many page faults would occur for the following replacement algorithms, assuming one, two, three, four, five, six, or seven frames?

a) LRU replacement

b) FIFO replacement

c) Optimal replacement



Last exercise

Systems@ETH zürich

Distributed Computing

1.4 Page Replacement

Consider the following page access pattern:

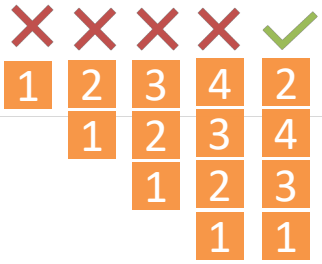
1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6

How many page faults would occur for the following replacement algorithms, assuming one, two, three, four, five, six, or seven frames?

a) LRU replacement

b) FIFO replacement

c) Optimal replacement



Last exercise

Systems@ETH Zürich

Distributed Computing

1.4 Page Replacement

Consider the following page access pattern:

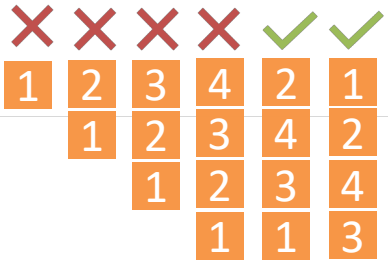
1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6

How many page faults would occur for the following replacement algorithms, assuming one, two, three, four, five, six, or seven frames?

a) LRU replacement

b) FIFO replacement

c) Optimal replacement



Last exercise

Systems@ETH Zürich

Distributed Computing

1.4 Page Replacement

Consider the following page access pattern:

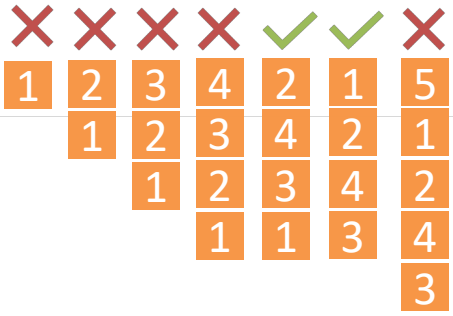
1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6

How many page faults would occur for the following replacement algorithms, assuming one, two, three, four, five, six, or seven frames?

a) LRU replacement

b) FIFO replacement

c) Optimal replacement



Last exercise

Systems@ETH zürich

Distributed Computing

1.4 Page Replacement

Consider the following page access pattern:

1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6

How many page faults would occur for the following replacement algorithms, assuming one, two, three, four, five, six, or seven frames?

a) LRU replacement

b) FIFO replacement

c) Optimal replacement

1

Last exercise

Systems@ETH zürich

Distributed Computing

1.4 Page Replacement

Consider the following page access pattern:

1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6

How many page faults would occur for the following replacement algorithms, assuming one, two, three, four, five, six, or seven frames?

a) LRU replacement

b) FIFO replacement

c) Optimal replacement

Last exercise

Systems@ETH zürich

Distributed Computing

1.4 Page Replacement

Consider the following page access pattern:

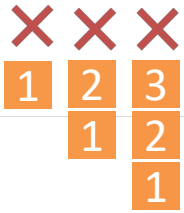
1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6

How many page faults would occur for the following replacement algorithms, assuming one, two, three, four, five, six, or seven frames?

a) LRU replacement

b) FIFO replacement

c) Optimal replacement



Last exercise

Systems@ETH Zürich

Distributed Computing

1.4 Page Replacement

Consider the following page access pattern:

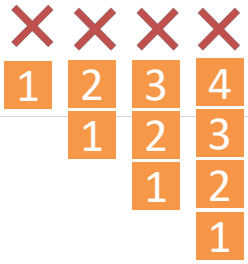
1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6

How many page faults would occur for the following replacement algorithms, assuming one, two, three, four, five, six, or seven frames?

a) LRU replacement

b) FIFO replacement

c) Optimal replacement



Last exercise

Systems@ETH Zürich

Distributed Computing

1.4 Page Replacement

Consider the following page access pattern:

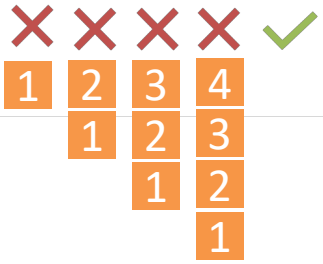
1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6

How many page faults would occur for the following replacement algorithms, assuming one, two, three, four, five, six, or seven frames?

a) LRU replacement

b) FIFO replacement

c) Optimal replacement



Last exercise

Systems@ETH zürich

Distributed Computing

1.4 Page Replacement

Consider the following page access pattern:

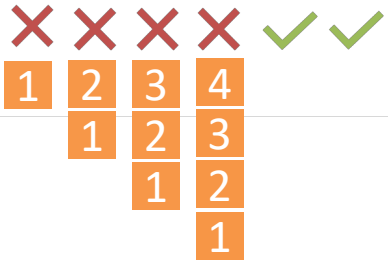
1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6

How many page faults would occur for the following replacement algorithms, assuming one, two, three, four, five, six, or seven frames?

a) LRU replacement

b) FIFO replacement

c) Optimal replacement



Last exercise

Systems@ETH zürich

Distributed Computing

1.4 Page Replacement

Consider the following page access pattern:

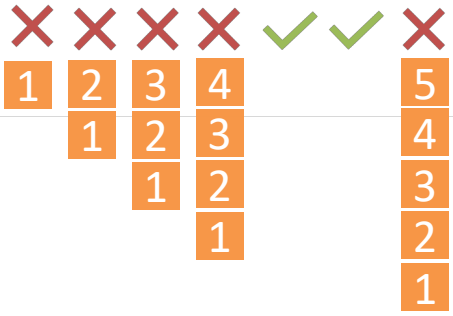
1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6

How many page faults would occur for the following replacement algorithms, assuming one, two, three, four, five, six, or seven frames?

a) LRU replacement

b) FIFO replacement

c) Optimal replacement



Last exercise

Systems@ETH zürich

Distributed Computing

1.4 Page Replacement

Consider the following page access pattern:

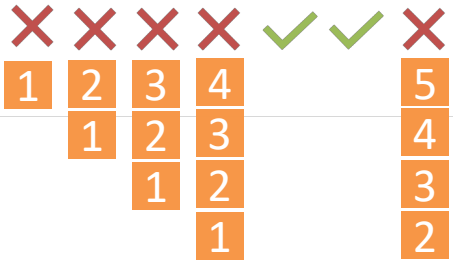
1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6

How many page faults would occur for the following replacement algorithms, assuming one, two, three, four, five, six, or seven frames?

a) LRU replacement

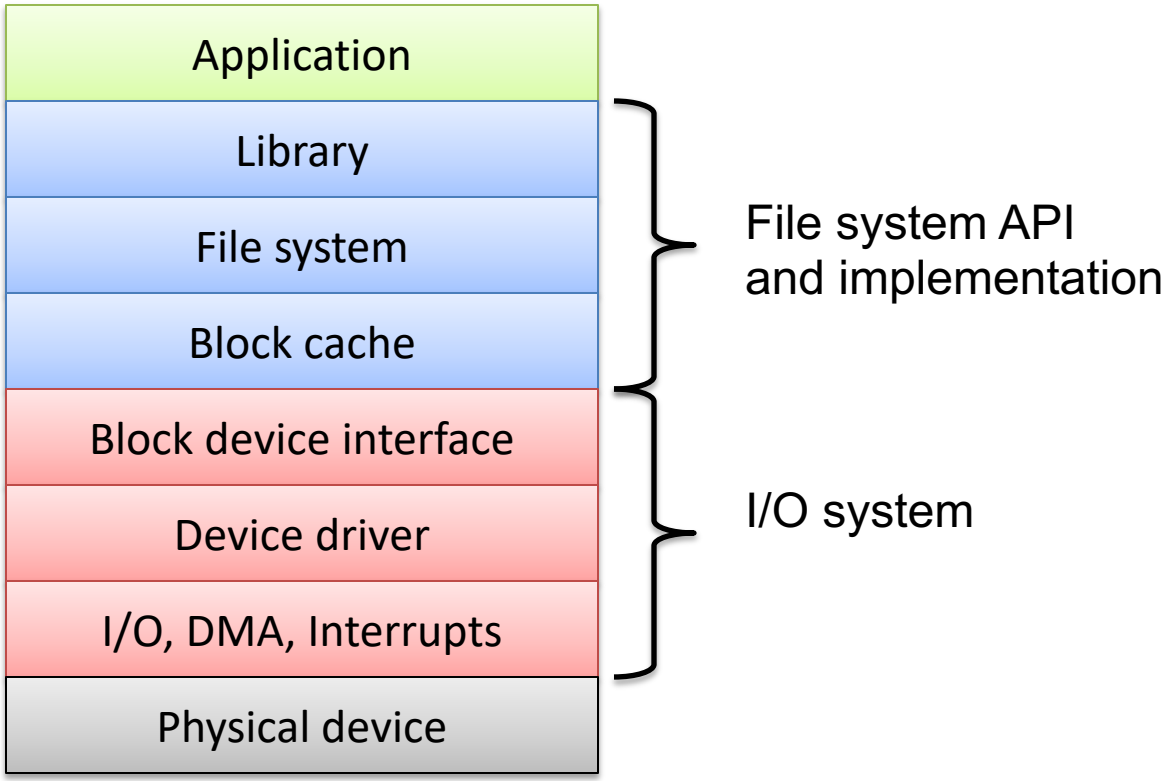
b) FIFO replacement

c) Optimal replacement



File System Abstractions

File system in context



File systems operations

Systems@ETH Zürich

Distributed Computing

- Files
  - Read
  - Write
  - Execute
- Directory
  - Link
  - Unlink
  - Rename
  - List

File metadata

Systems@ETH Zürich

Distributed Computing

- Metadata: important concept!
  - Data *about* an object, not the object *itself*
- File metadata examples:
  - Name
  - Location on disk
  - Times of creation, last change, last access
  - Ownership, access control rights File type, file structure
  - Arbitrary descriptive data (used for searching)

Protection

Systems@ETH Zürich

Distributed Computing

- File owner/creator should be able to control:
  - what can be done
  - by whom
- Types of access
  - **Read**
  - **Write**
  - **Execute**
  - **Append**
  - **Delete**
  - **List (“read” for directory)**

Access control matrix

Systems@ETH Zürich

Distributed Computing

For a single file or directory:

Objects

Principals	File1	File2	File3	File4	File5	File6	File7	…
A	rwx	r	x			rw	x	
B	r	rwx		x			x	
C	rw				rwx			
D	rw	rw	x	x				
E	r							
F	r				rw			
…								

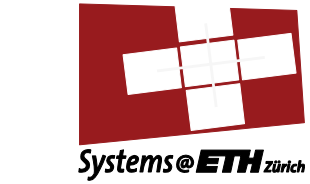
Problem: how to scalably represent this matrix?

ACLs

Systems@ETH Zürich

Distributed Computing

- Access Control Lists
  - For each file, list the principals and their rights on the object
  - Store with the file
- Good:
  - Easy to change rights quickly
  - Scales to large numbers of files
- Bad:
  - Doesn't scale to large numbers of principals



File System Implementation

Implementation aspects

Systems@ETH Zürich

Distributed Computing

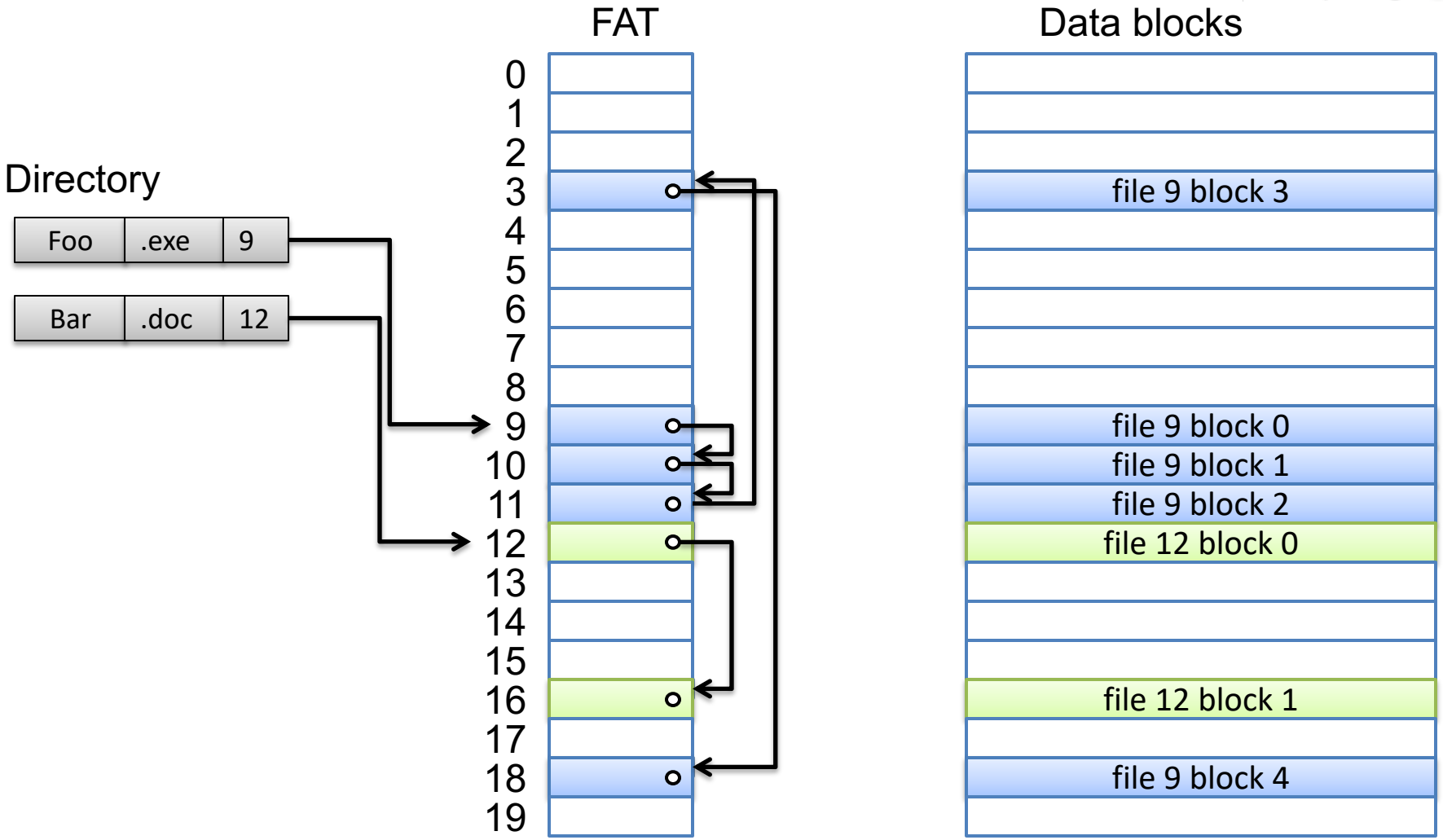
- Directories and indexes
  - Where on the disk is the data for each file?
- Index granularity
  - What is the unit of allocation for files?
- Free space maps
  - How to allocate more sectors on the disk?
- Locality optimizations
  - How to make it go fast in the common case

FAT

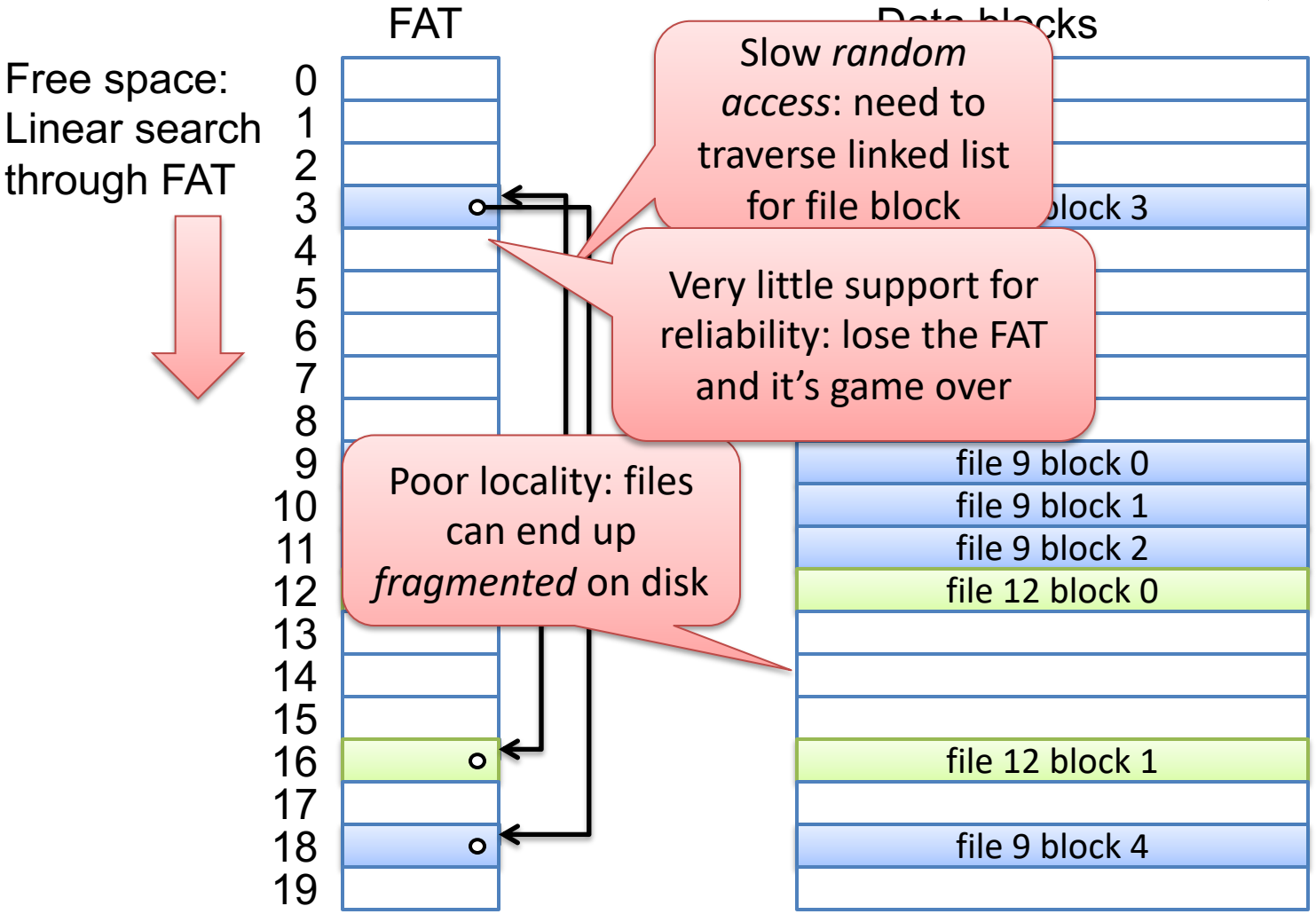
FAT background

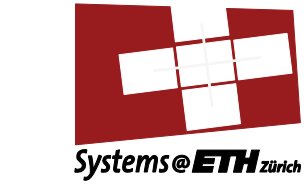
- No access control
- Very little metadata
- Limited volume size
- BUT still extensively used ☹
  - Flash devices, cameras, phones

FAT file system



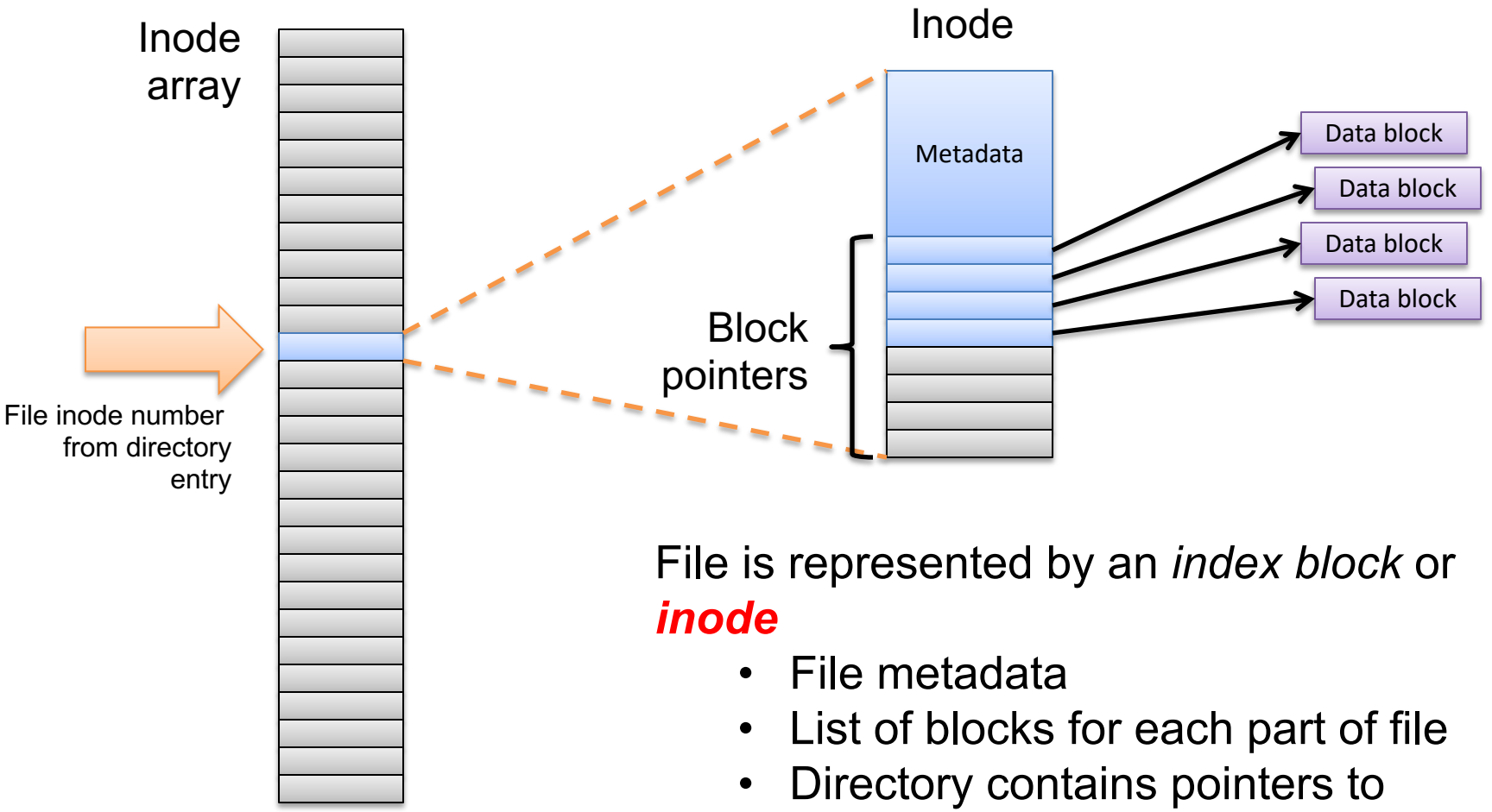
FAT file system





FFS

FFS uses indexed allocation



File is represented by an *index block* or ***inode***

- File metadata
- List of blocks for each part of file
- Directory contains pointers to inodes

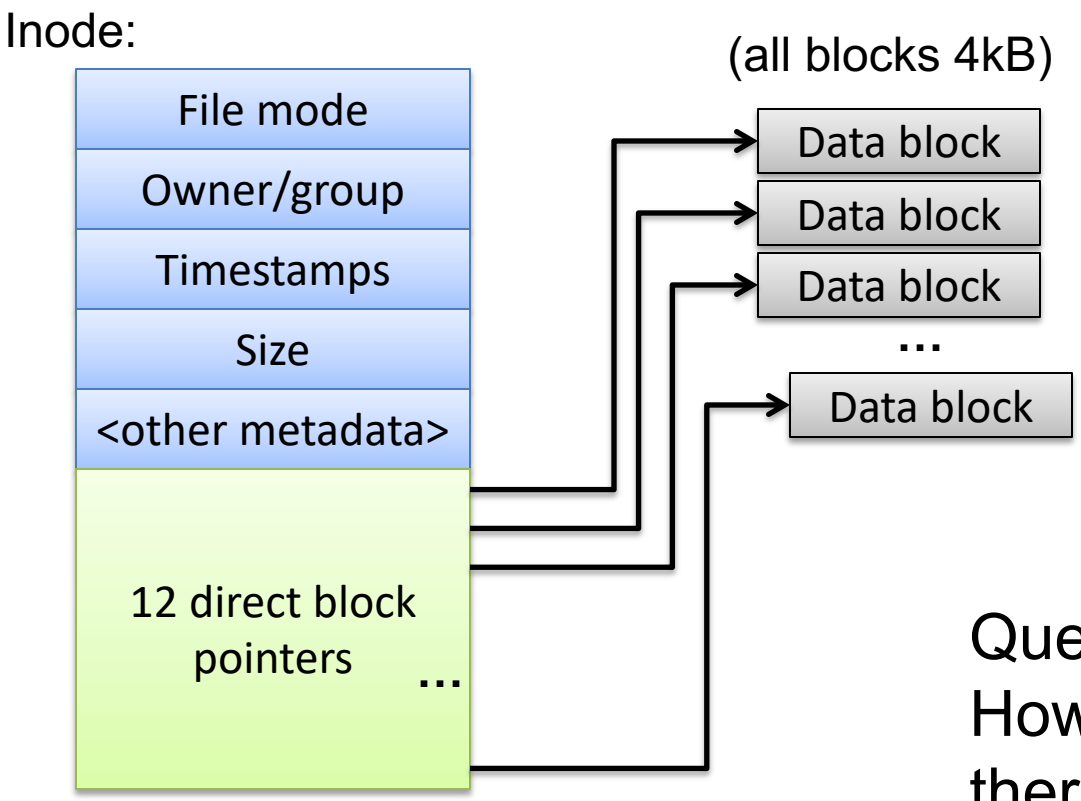
Inode and file size in FFS

Systems@ETH Zürich

Distributed Computing

- Example:
  - Inode is 1 block = 4096 bytes
  - Inode metadata = 512 bytes
  - Block addresses = 8 bytes
- Hence:
  - (4096-512) / 8 = 448 block pointers
  - 448 * 4096 = 1792kB max. file size

Unix file system inode format (simplified)

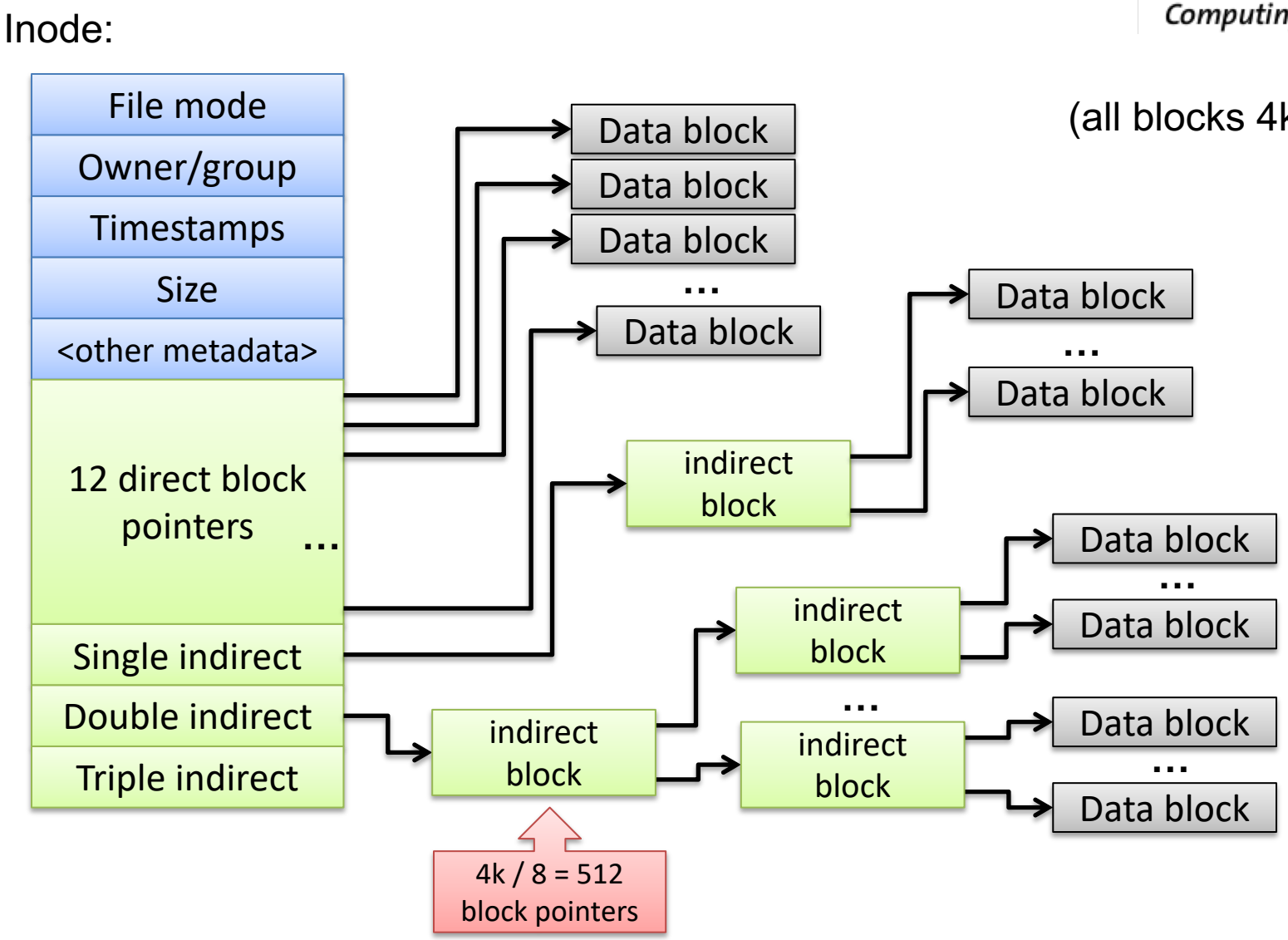


Question:
How to extend file size if there are no more block pointers in the Inode?

Unix file system inode format (simplified)

Systems@ETH zürich

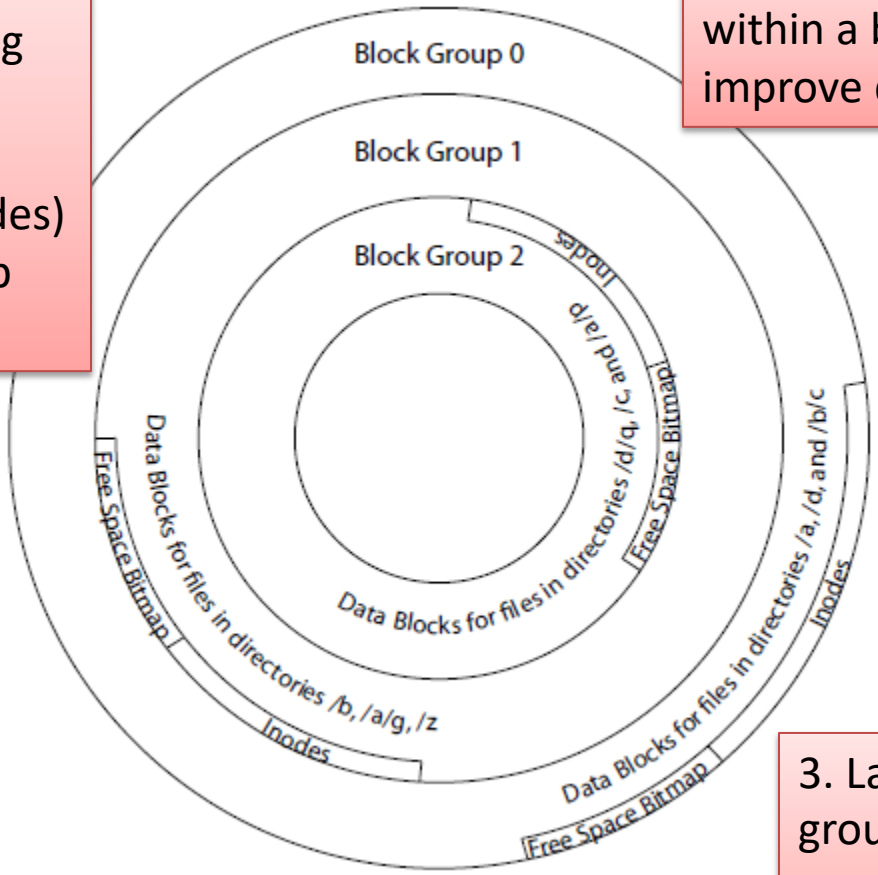
Distributed Computing



Block groups

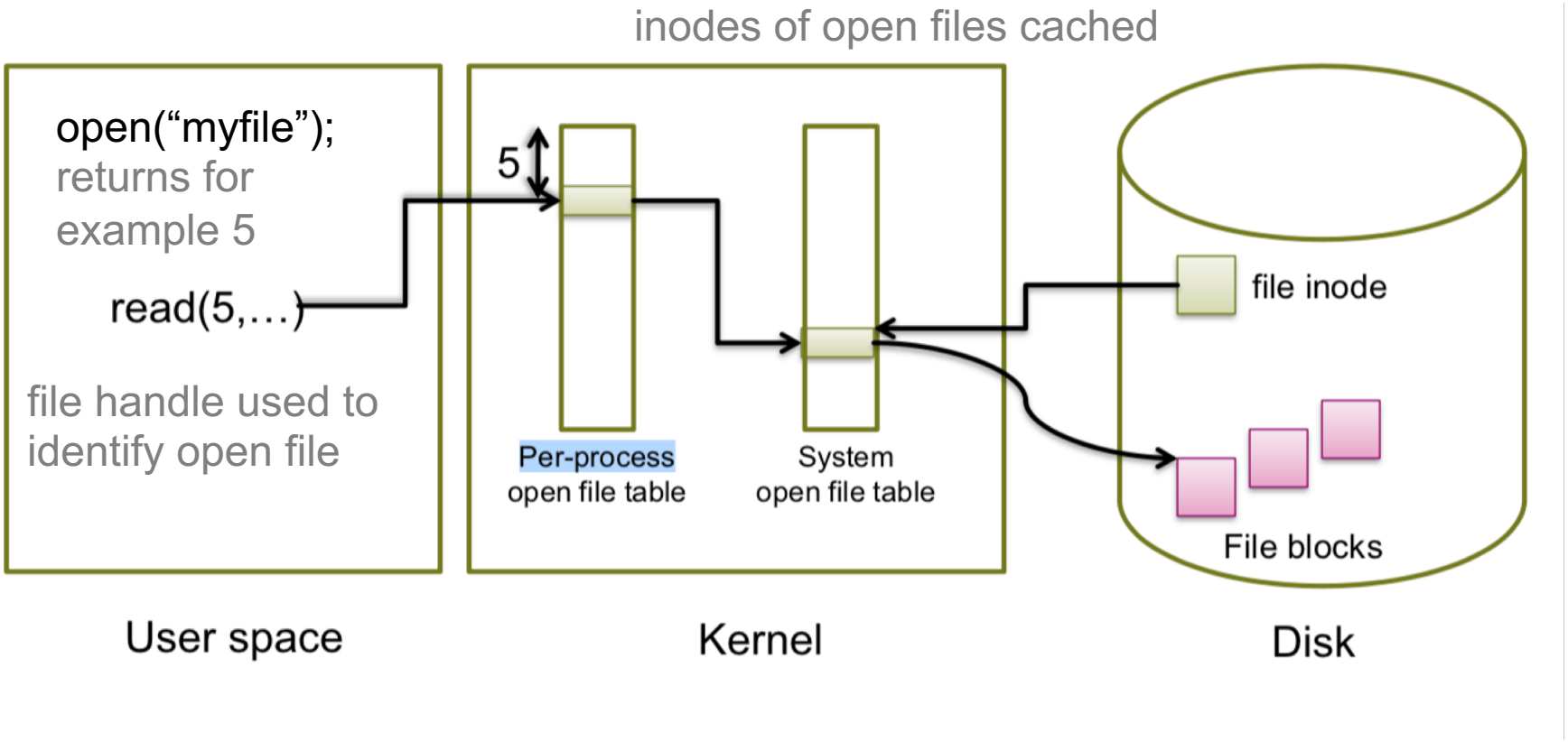
1. Optimize disk performance by keeping together related:
   - Files
   - Metadata (inodes)
   - Free space map
   - Directories

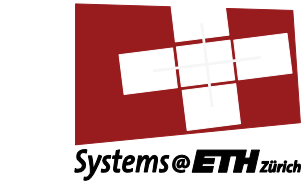
2. Use *first-fit* allocation within a block group to improve disk locality



3. Layout and block groups defined in the *superblock* (not shown); Replicated several times.

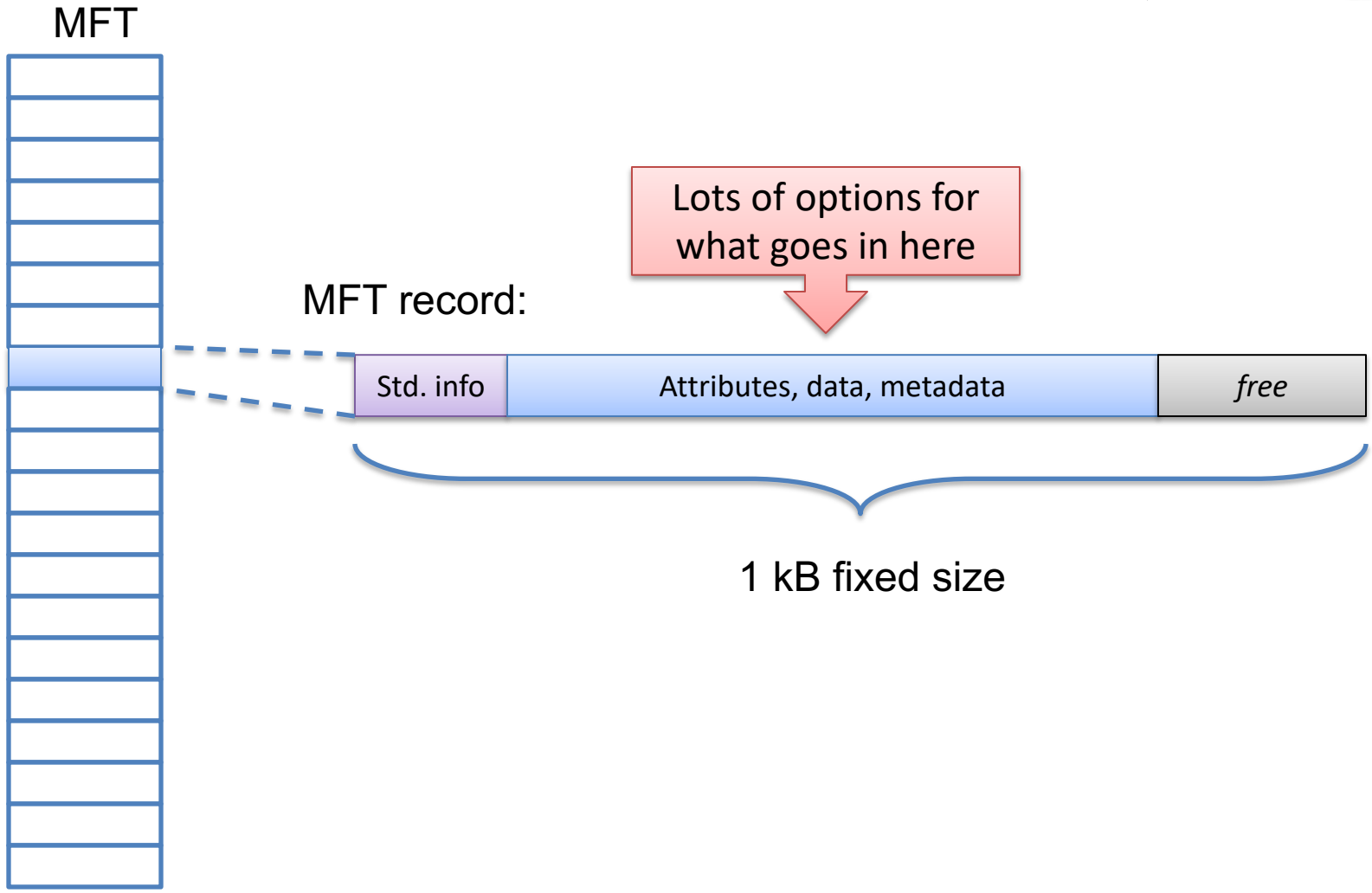
Open Files


inodes of open files cached
open("myfile");
returns for
example 5
read(5,…)
file handle used to
identify open file
5
Per-process
open file table
System
open file table
file inode
File blocks
User space
Kernel
Disk



NTFS

NFTS Master file table

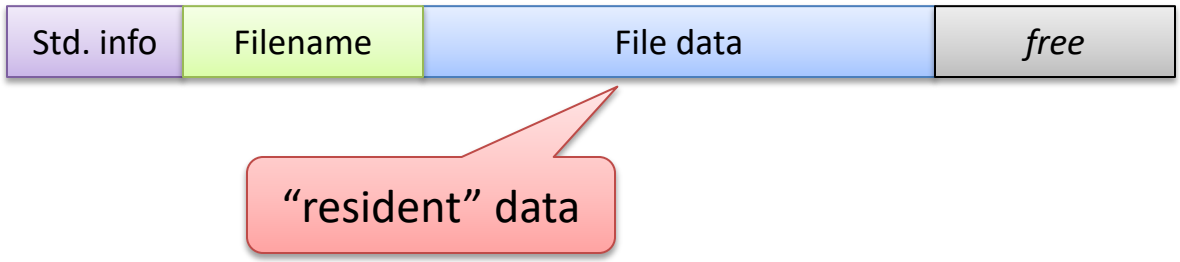


NTFS small files

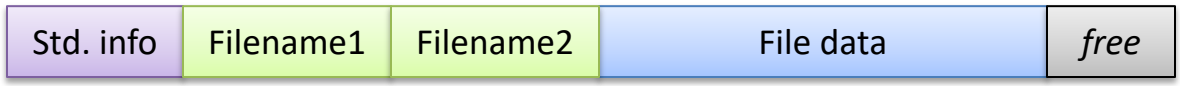
Systems@ETH zürich

Distributed Computing

- Small file fits into MFT record:



- Hard links (multiple names) stored in MFT:

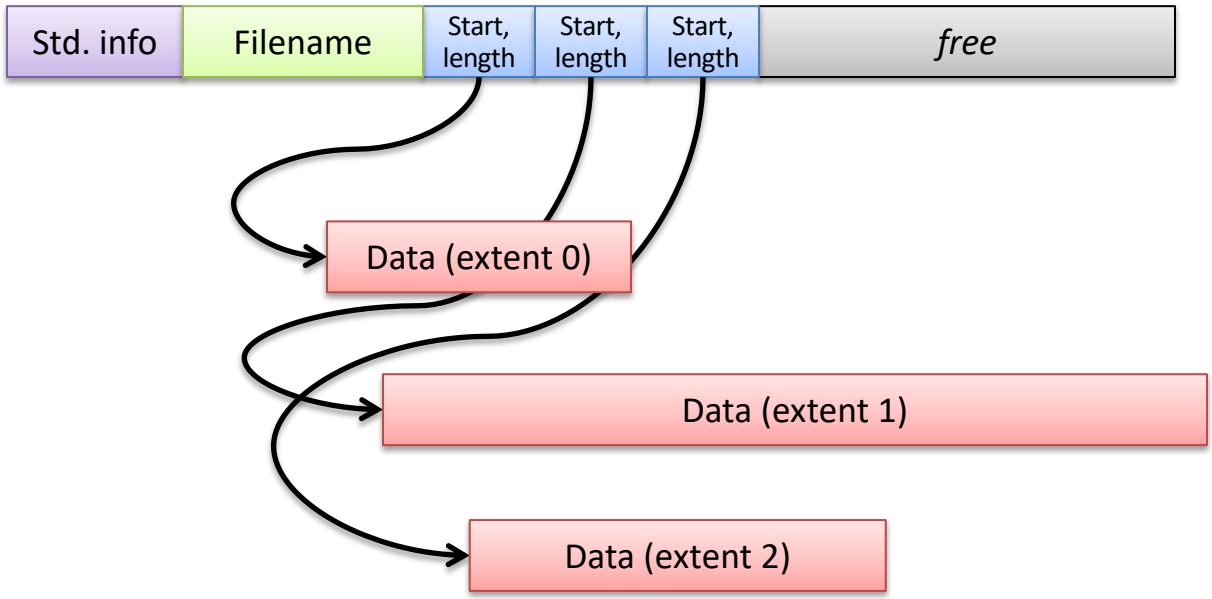


NTFS normal files

Systems@ETH Zürich

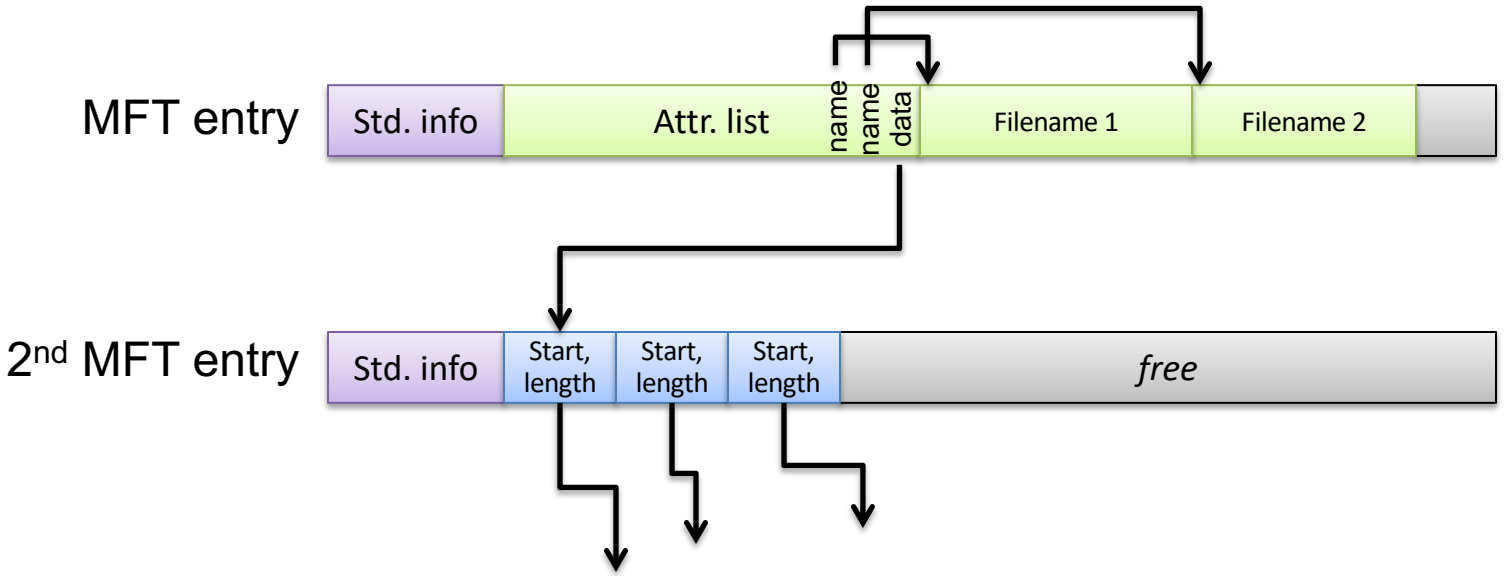
Distributed Computing

- MFT holds list of *extents*:



Too many attributes?

- Attribute list holds list of attribute locations



In addition, attributes can also be stored in extents ⇒ very large scaling

Metadata files

Systems@ETH zürich

Distributed Computing

- File system metadata in NTFS is held *in files!*

File num.	Name	Description
0	$MFT	Master file table
1	$MFTirr	Copy of first 4 MFT entries
2	$Logfile	Transaction log of FS change
3	$Volume	Volume information & metada
4	$AttrDef	Table mapping numeric IDs to a
5	.	Root directory
6	$Bitmap	Free space bitmap
7	$Boot	Volume boot record
8	$BadClus	Bad cluster map
9	$Secure	Access control list database
10	$UpCase	Filename mappings to DOS
11	$Extend	Extra file system attributes (e.g. quota)

Question:
Huh?
Where is it then?
Answer:
First sector of volume points to first block of MFT

File system implementations

Systems@ETH Zürich

Distributed Computing

	FAT	FFS	NTFS
Index structure	Linked list	Fixed, assymetric tree	Dynamic tree
Index granularity:	Block	Block	Extent
Free space management	FAT Array	Fixed bitmap	Bitmap in file
Locality heuristics	Defragmentation	Block groups, Reserve space	Best fit, Defragmentation

Feedback

- https://bit.ly/2Fuon6W