

I/O and Storage: File System Implementation

CS 571: Operating Systems (Spring 2020)

Lecture 11b

Yue Cheng

File System Implementation

File System Implementation

- On-disk structures
 - How do we represent files and directories?
- File system operations (internally)
 - How on-disk structures get touched when performing FS operations
- File system locality & data layout policies
 - How data layout impacts locality for on-disk FS?

On-Disk Structures

A Naïve Flat Persistent Store

- **Given:** big array of on-disk bytes/blocks
- **Want:** to support reads and writes

A Naïve Flat Persistent Store

- **Given**: big array of on-disk bytes/blocks
- **Want**: to support reads and writes

- Build a **flat** persistent store where each file is associated with a unique key
 - Uses a flat table to track files
 - Uses offsets for non-sequential I/O

Flat Persistent Store vs. File System

- What features does a [file system](#) provide beyond what a naïve flat persistent store would provide?

Flat Persistent Store vs. File System

- What features does a **file system** provide beyond what a naïve flat persistent store would provide?
 - Human readable string names
 - Hierarchy (names within names)
 - Changeable file sizes
 - ...

Flat Persistent Store vs. File System

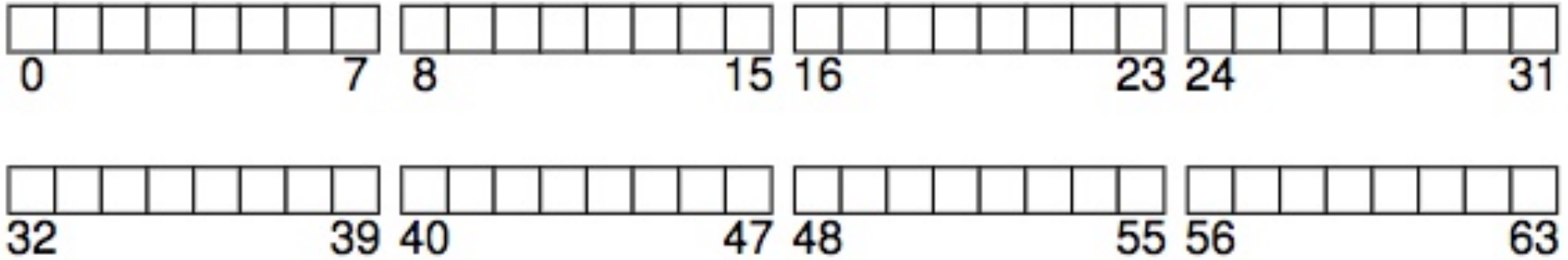
- What features does a **file system** provide beyond what a naïve flat persistent store would provide?
 - Human readable string names
 - Hierarchy (names within names)
 - Changeable file sizes
 - ...

All these features require a variety of on-disk data structures!

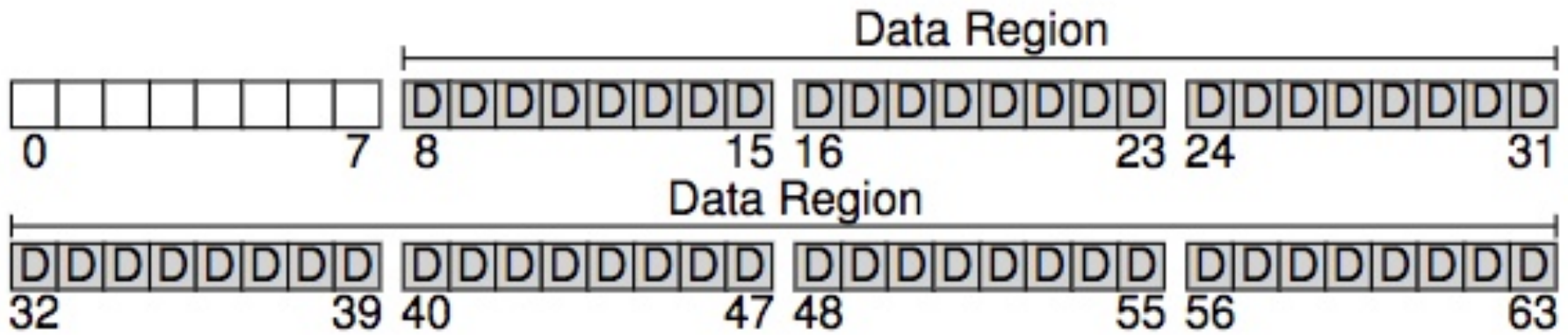
On-Disk Structures

- Common file system structures
 - Data block
 - inode table
 - Directories
 - Data bitmap
 - inode bitmap
 - Superblock

On-Disk Structure: Empty Disk



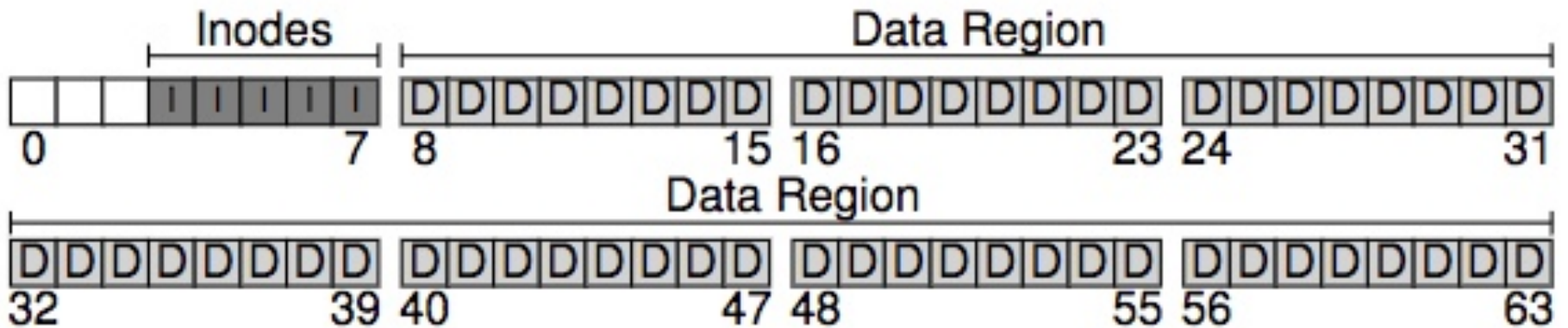
On-Disk Structure: Data Blocks



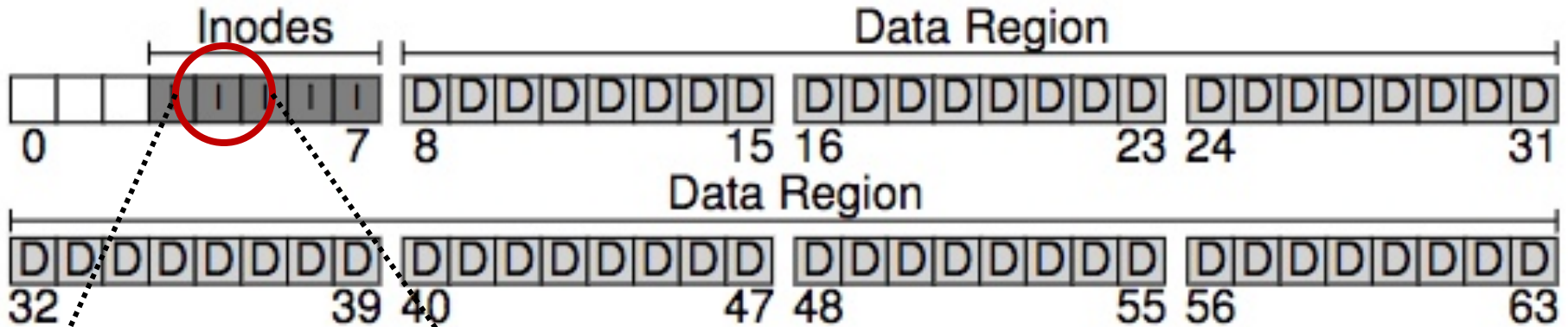
On-Disk Structures

- Common file system structures
 - Data block
 - **inode table**
 - Directories
 - Data bitmap
 - inode bitmap
 - Superblock

On-Disk Structure: Inodes



On-Disk Structure: Inodes

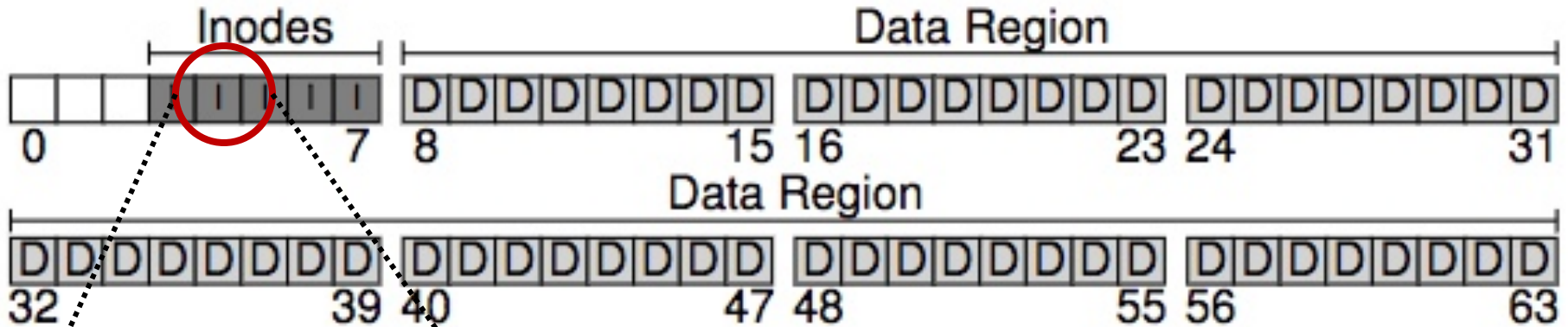


inode 16	inode 17	inode 18	inode 19
inode 20	inode 21	inode 22	inode 23
inode 24	inode 25	inode 26	inode 27
inode 28	inode 29	inode 30	inode 31

Inode Block

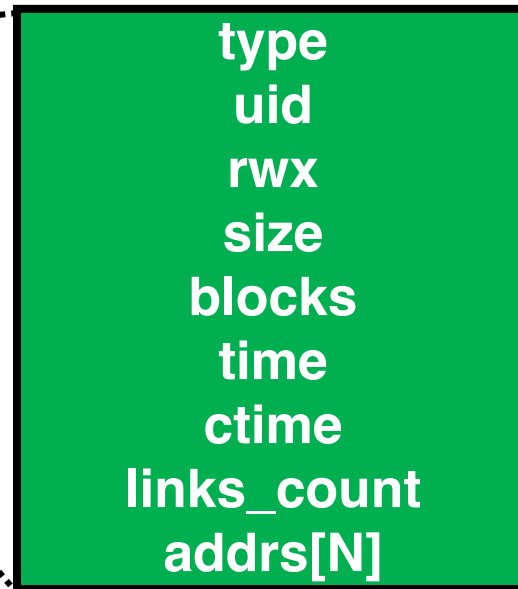
- Inodes are typically 128 or 256 bytes (depends on the file system)
 - 16—32 inodes per inode block

On-Disk Structure: Inodes



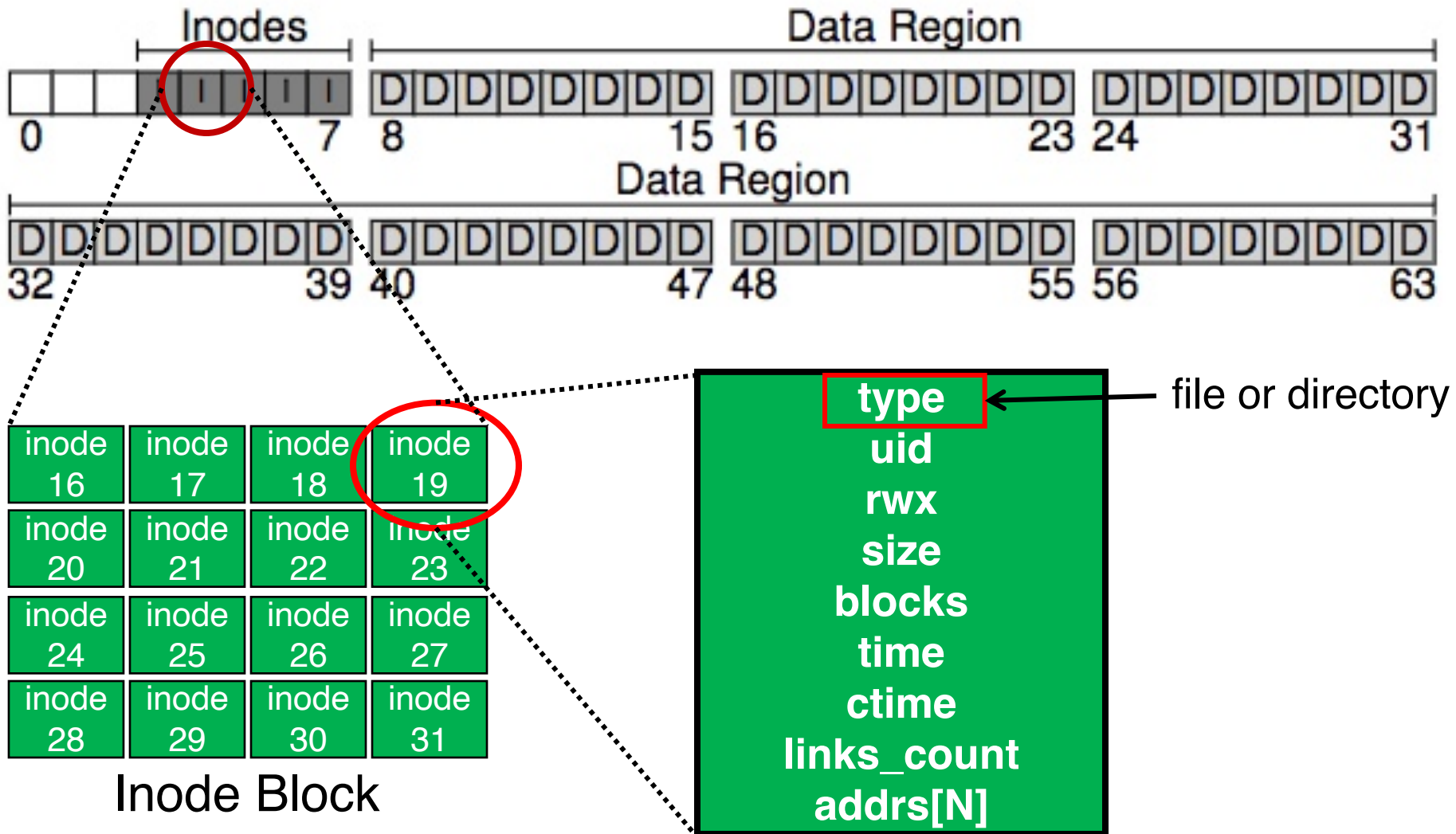
inode 16	inode 17	inode 18	inode 19
inode 20	inode 21	inode 22	inode 23
inode 24	inode 25	inode 26	inode 27
inode 28	inode 29	inode 30	inode 31

Inode Block

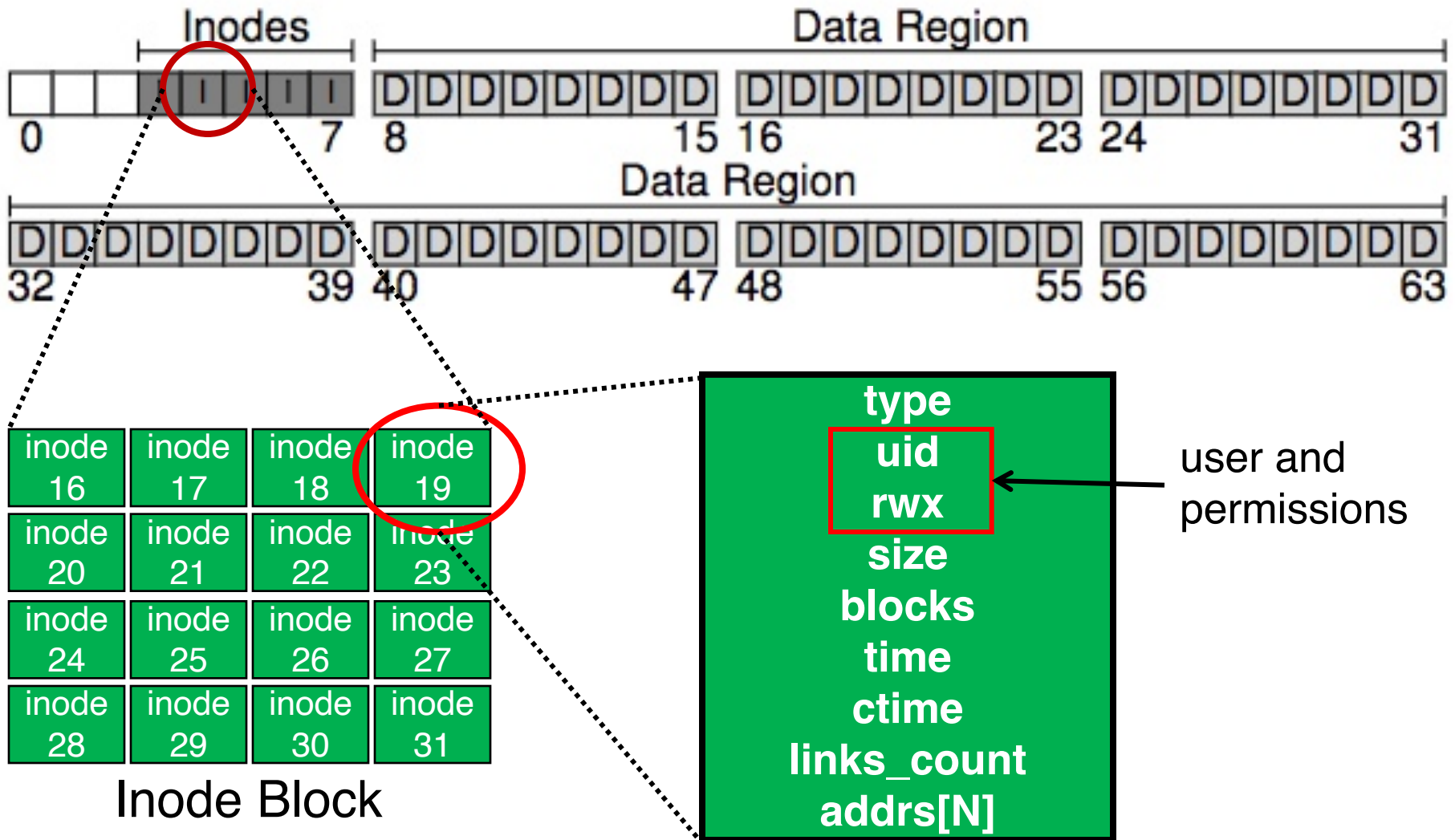


Inode

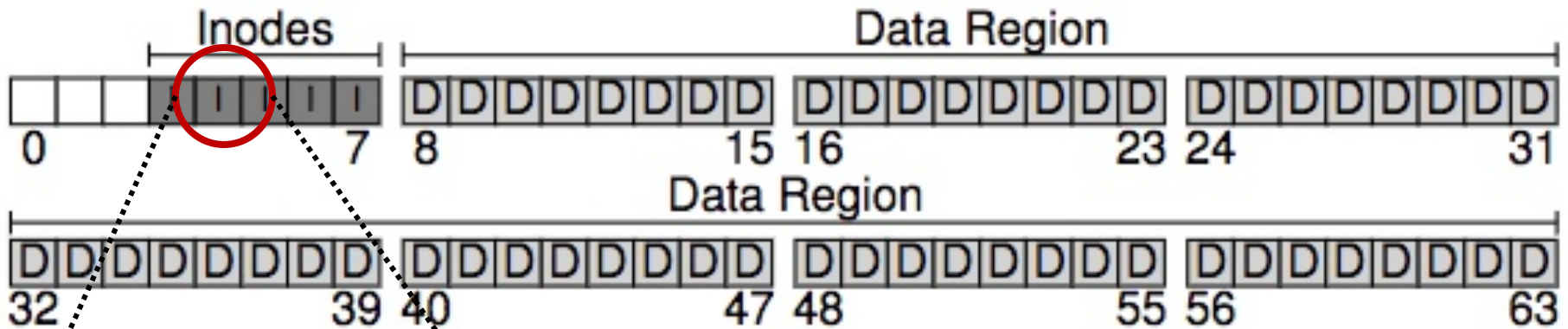
On-Disk Structure: Inodes



On-Disk Structure: Inodes

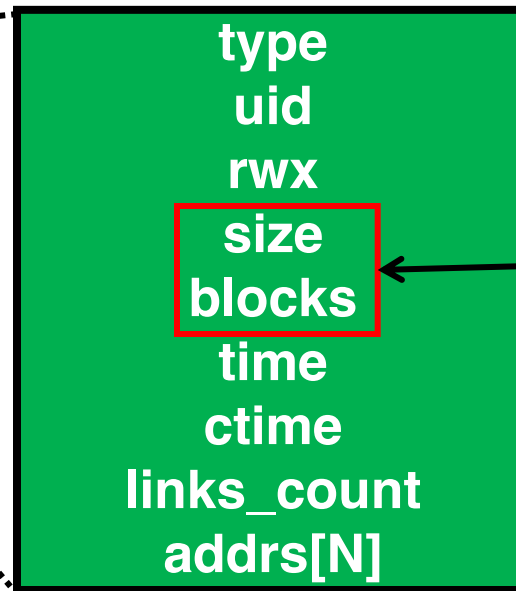


On-Disk Structure: Inodes

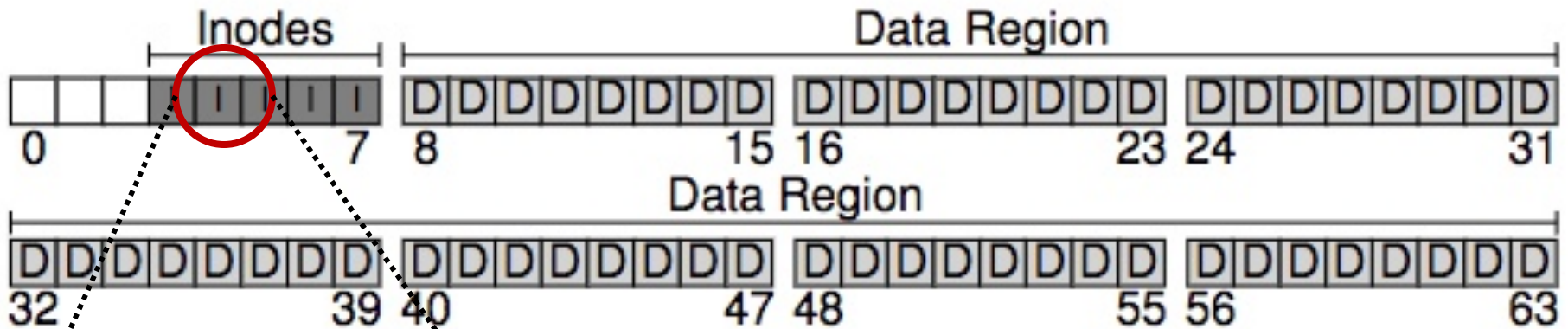


inode 16	inode 17	inode 18	inode 19
inode 20	inode 21	inode 22	inode 23
inode 24	inode 25	inode 26	inode 27
inode 28	inode 29	inode 30	inode 31

Inode Block

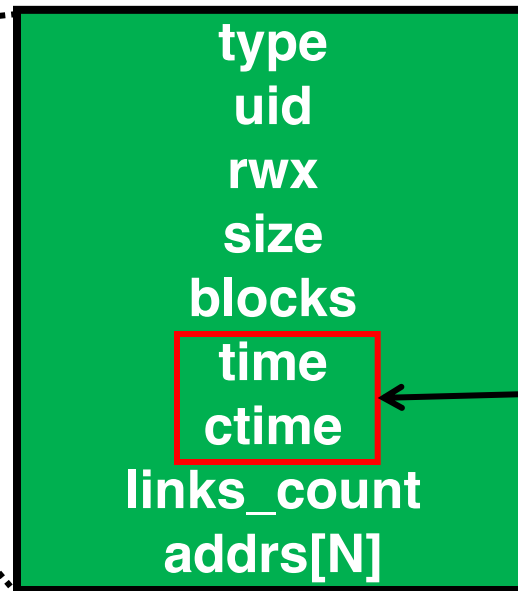


On-Disk Structure: Inodes



inode 16	inode 17	inode 18	inode 19
inode 20	inode 21	inode 22	inode 23
inode 24	inode 25	inode 26	inode 27
inode 28	inode 29	inode 30	inode 31

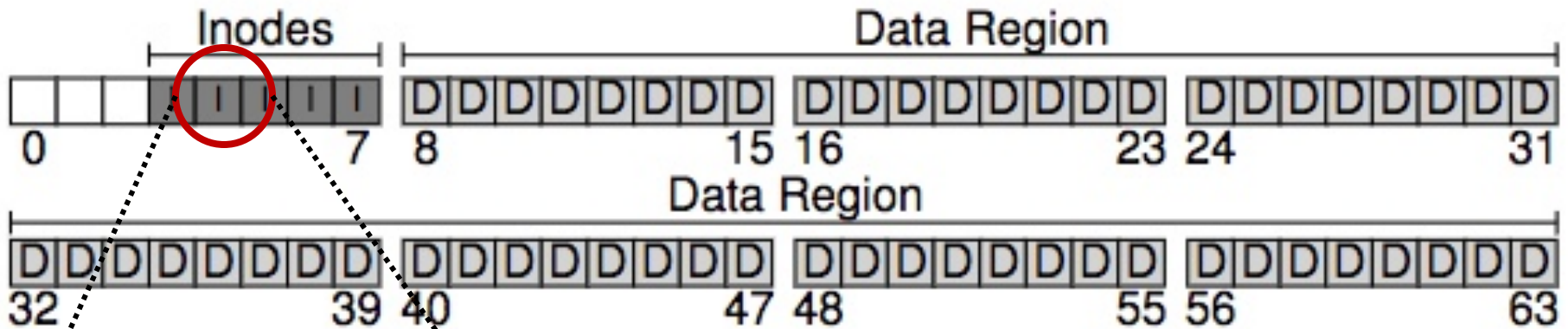
Inode Block



access time
and create time

Inode

On-Disk Structure: Inodes



inode 16	inode 17	inode 18	inode 19
inode 20	inode 21	inode 22	inode 23
inode 24	inode 25	inode 26	inode 27
inode 28	inode 29	inode 30	inode 31

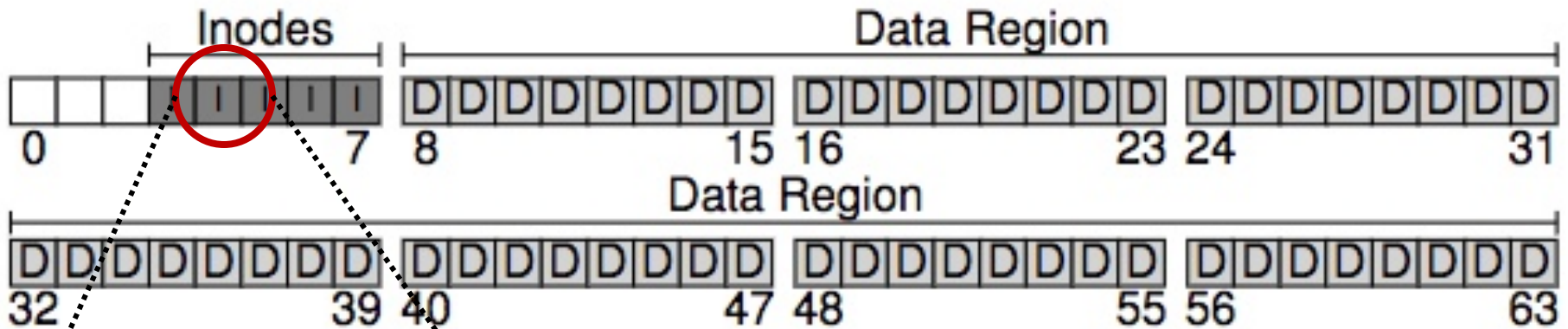
Inode Block

type
uid
rxw
size
blocks
time
ctime
links_count
addrs[N]

how many links
(directories)

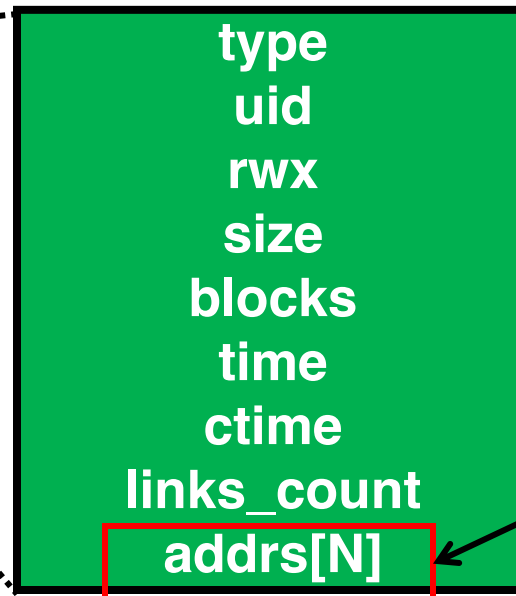
Inode

On-Disk Structure: Inodes



inode 16	inode 17	inode 18	inode 19
inode 20	inode 21	inode 22	inode 23
inode 24	inode 25	inode 26	inode 27
inode 28	inode 29	inode 30	inode 31

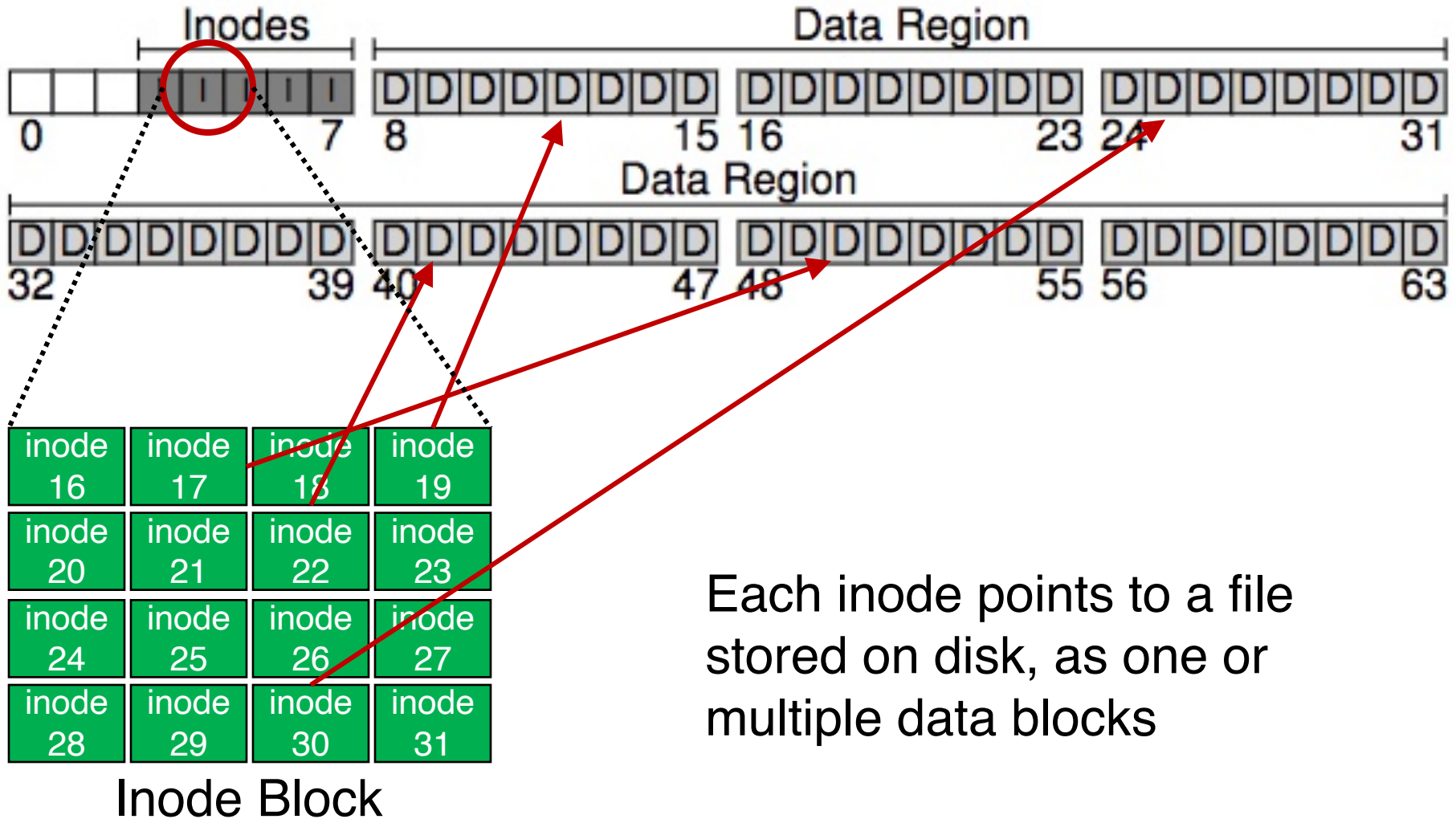
Inode Block



addrs of N data
blocks

Inode

On-Disk Structure: Inodes



On-Disk Structures

- Common file system structures
 - Data block
 - Inode table
 - Directories
 - Data bitmap
 - Inode bitmap
 - Superblock

On-Disk Structure: Directories

- Common directory design: just store directory entries in files
 - Different file systems vary
- Various data structures (formats) could be used
 - Lists
 - B-trees

On-Disk Structures

- Common file system structures
 - Data block
 - inode table
 - Directories
 - Data bitmap
 - inode bitmap
 - Superblock

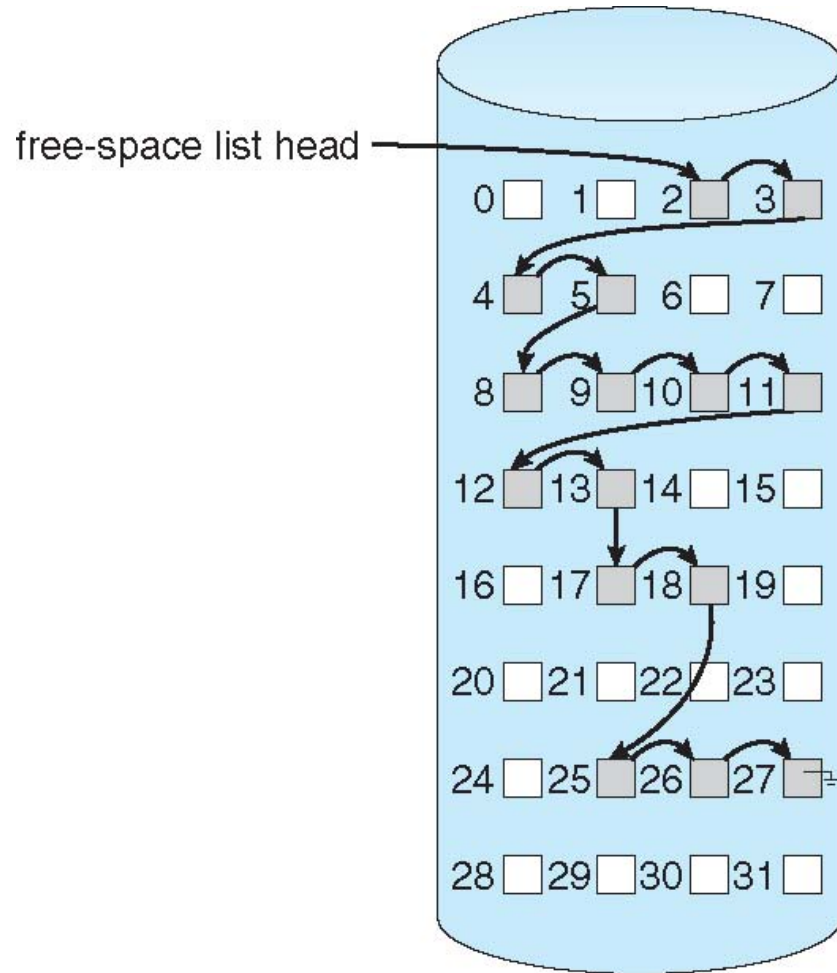
Allocation

- How does file system find free data blocks or free inodes?

Allocation

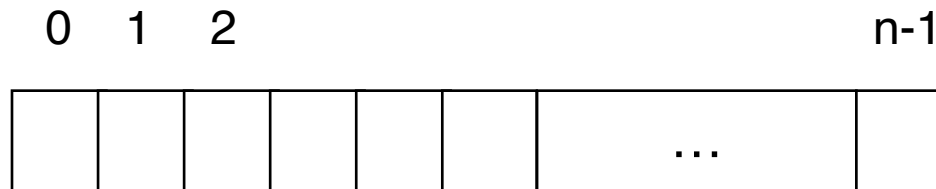
- How does file system find free data blocks or free inodes?
 - Free list
 - Bitmaps
- What are the tradeoffs?

Free List



Bitmap

Each bit of the bitmap is used to indicate whether the corresponding object/block is **free** (0) or **in-use** (1)

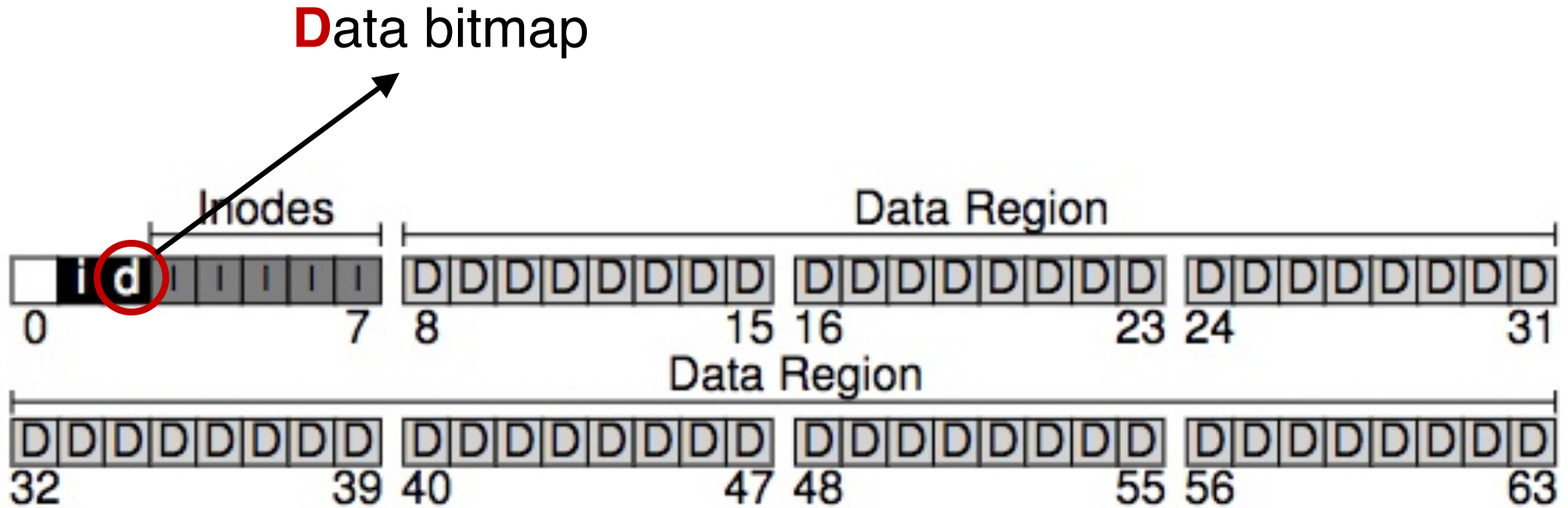


$$\text{bit}[i] = \begin{cases} 1 \Rightarrow \text{object}[i] \text{ in use} \\ 0 \Rightarrow \text{object}[i] \text{ free} \end{cases}$$

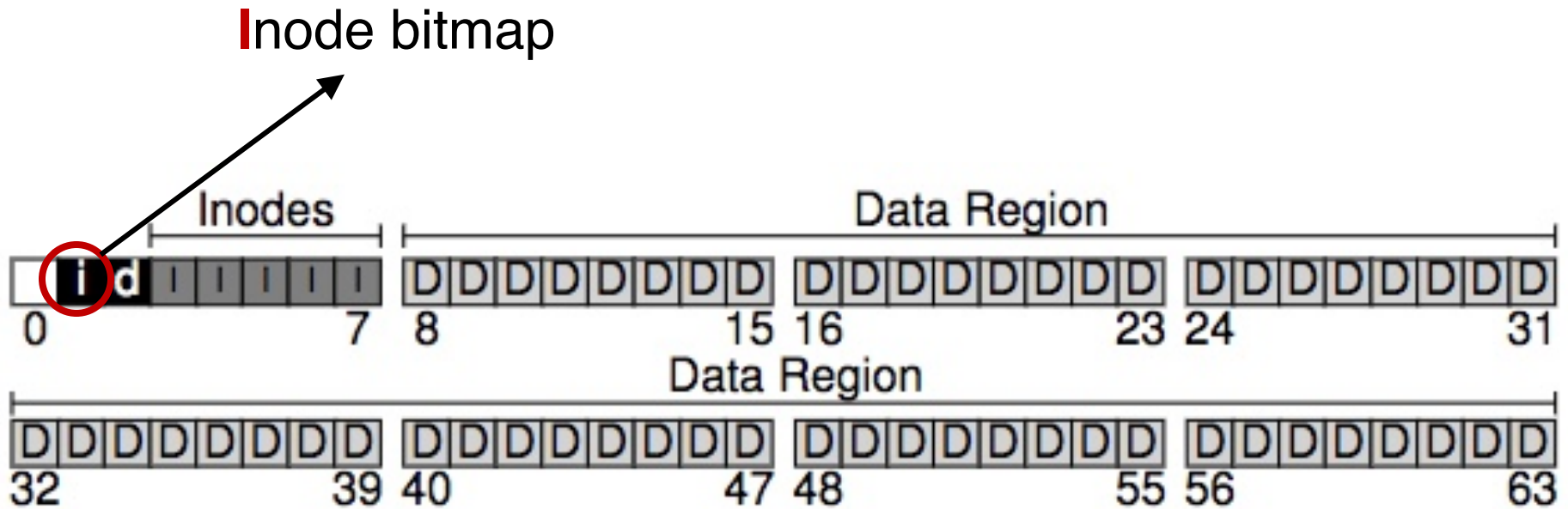
Allocation

- How does file system find free data blocks or free inodes?
 - Free list
 - Bitmaps
- What are the tradeoffs?
 - Free list: Cannot get contiguous space easily
 - Bitmap: Easy to allocate contiguous space for files

On-Disk Structure: Data Bitmaps



On-Disk Structure: Inode Bitmaps



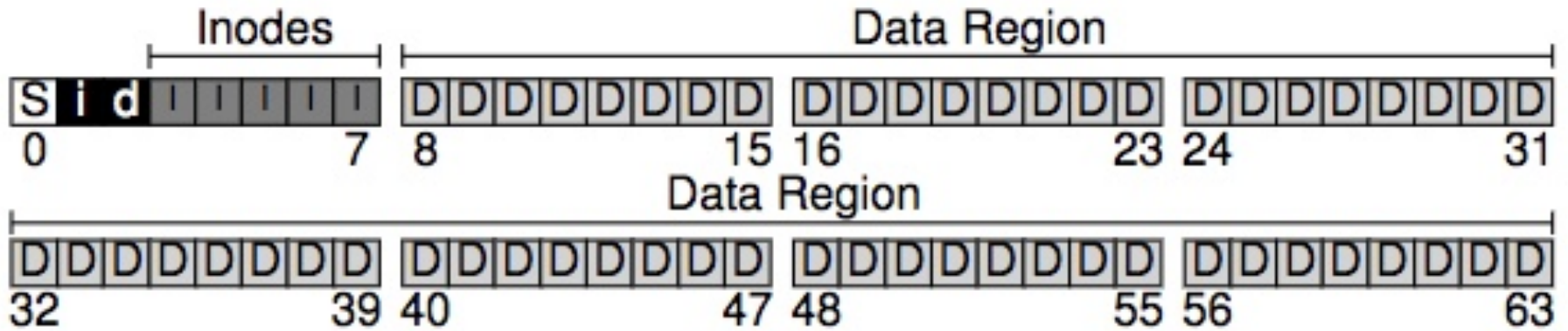
On-Disk Structures

- Common file system structures
 - Data block
 - Inode table
 - Directories
 - Data bitmap
 - Inode bitmap
 - Superblock

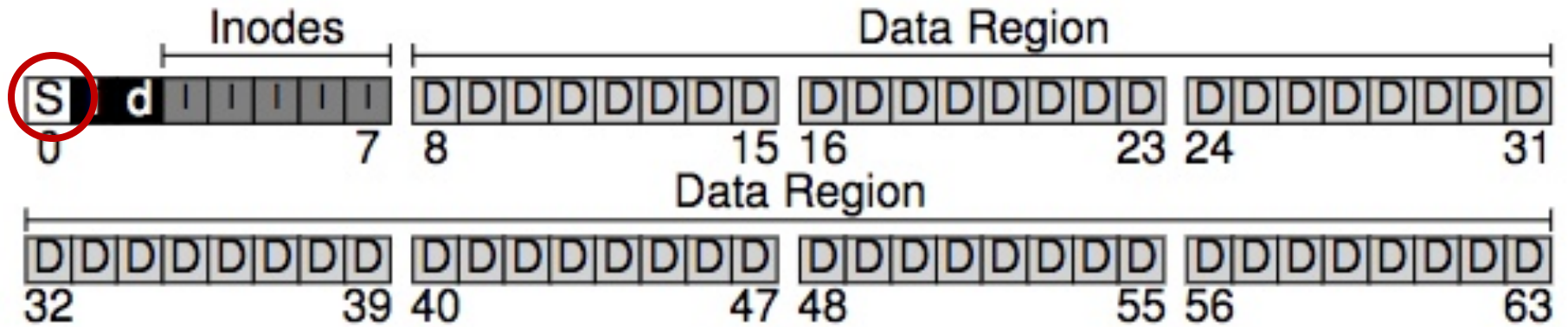
On-Disk Structure: Superblock

- Need to know basic file system configuration and runtime status, such as:
 - Block size
 - How many inodes are there
 - How much free space
- Store all these **metadata** info in a superblock

On-Disk Structure: Superblock

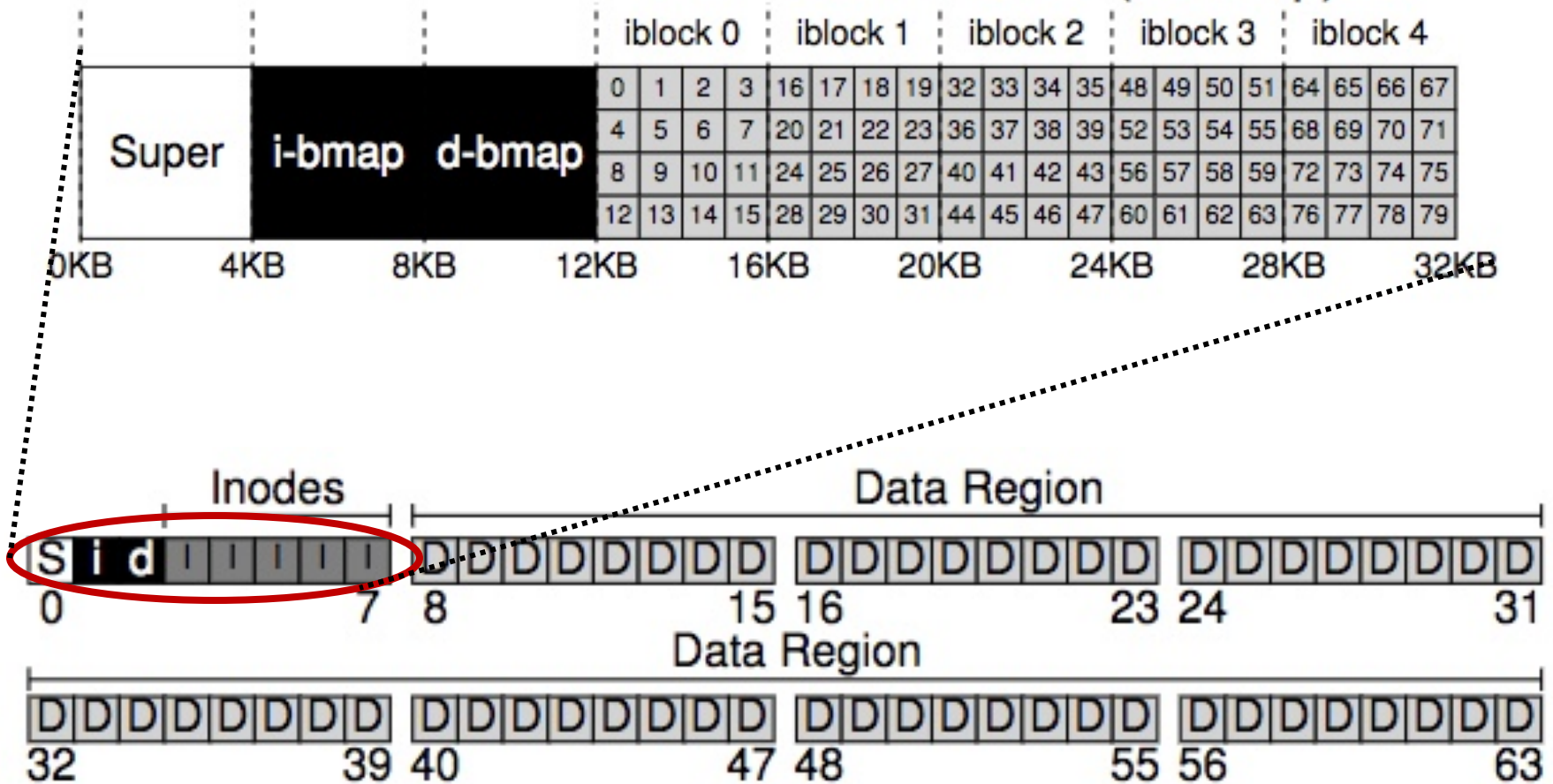


On-Disk Structure: Superblock



On-Disk Structure Overview

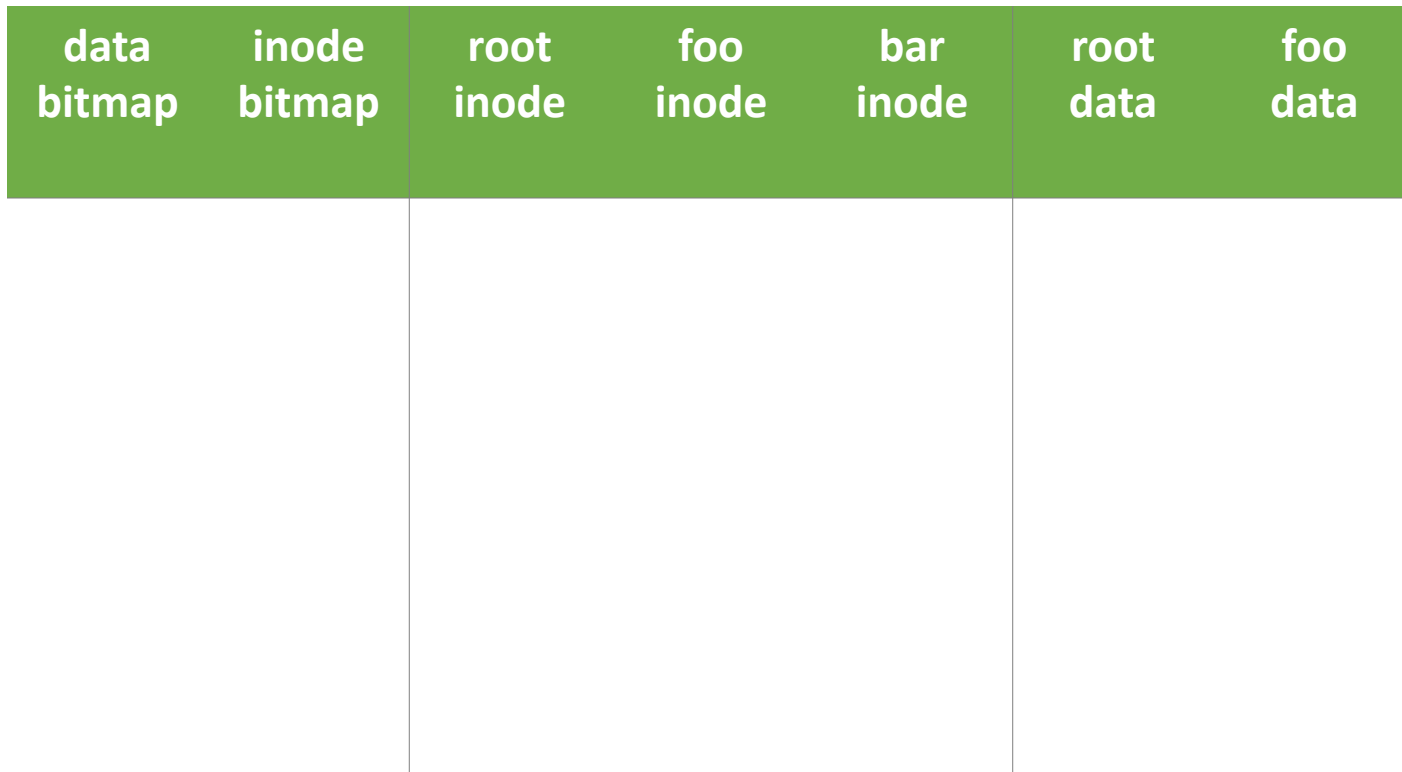
The Inode Table (Closeup)



File System Operations

Basic File System Operations

create /foo/bar



Basic File System Operations

create /foo/bar

[traverse]

data bitmap	inode bitmap	root inode	foo inode	bar inode	root data	foo data
		read			read	

Basic File System Operations

create /foo/bar

[traverse]

data bitmap	inode bitmap	root inode	foo inode	bar inode	root data	foo data
		read			read	
			read			
						read

Basic File System Operations

create /foo/bar

[traverse]

data bitmap	inode bitmap	root inode	foo inode	bar inode	root data	foo data
		read				
			read		read	
						read

foo inode: we have permission
foo data: bar doesn't exist

Basic File System Operations

create /foo/bar

data bitmap	inode bitmap	root inode	foo inode	bar inode	root data	foo data
		read				
			read		read	
						read

Basic File System Operations

create /foo/bar

[allocate inode]

data bitmap	inode bitmap	root inode	foo inode	bar inode	root data	foo data
		read				
			read		read	
	read write					read

Basic File System Operations

create /foo/bar

[populate inode]

data bitmap	inode bitmap	root inode	foo inode	bar inode	root data	foo data
		read				
			read		read	
	read write					read
				read write		

Basic File System Operations

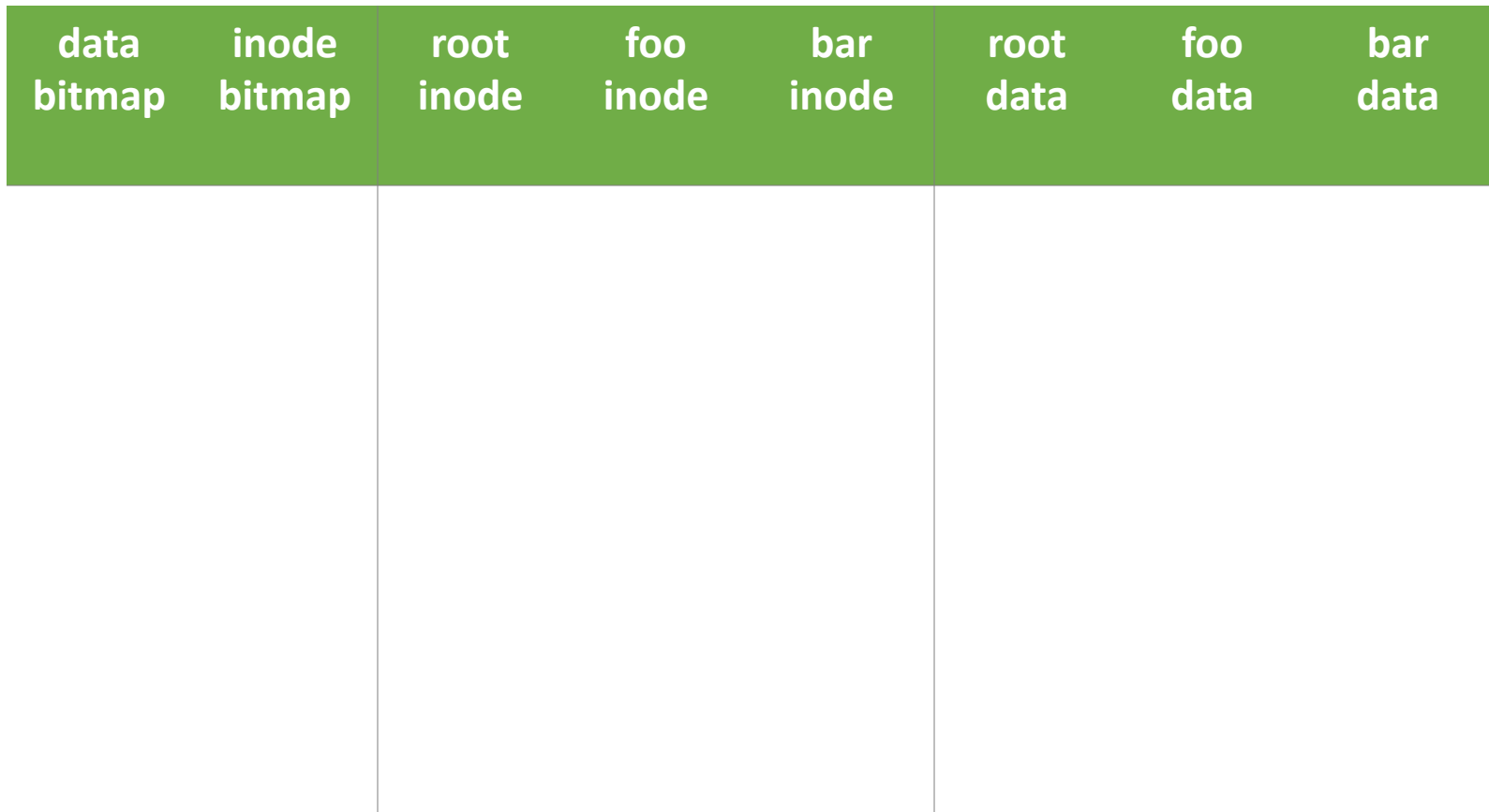
create /foo/bar

[add bar to /foo]

data bitmap	inode bitmap	root inode	foo inode	bar inode	root data	foo data
		read			read	
			read			read
	read write			read write		
			write			write

Basic File System Operations

write to /foo/bar



Basic File System Operations

write to /foo/bar

[block full? yes]

data bitmap	inode bitmap	root inode	foo inode	bar inode	root data	foo data	bar data
				read			

Basic File System Operations

write to /foo/bar

[allocate block]

data bitmap	inode bitmap	root inode	foo inode	bar inode	root data	foo data	bar data
				read			
read write							

Basic File System Operations

write to /foo/bar

[point to block]

data bitmap	inode bitmap	root inode	foo inode	bar inode	root data	foo data	bar data
				read			
read write				write			

Basic File System Operations

write to /foo/bar

[point to block]

data bitmap	inode bitmap	root inode	foo inode	bar inode	root data	foo data	bar data
				read			
read write				write			
							write

Basic File System Operations

write to /foo/bar

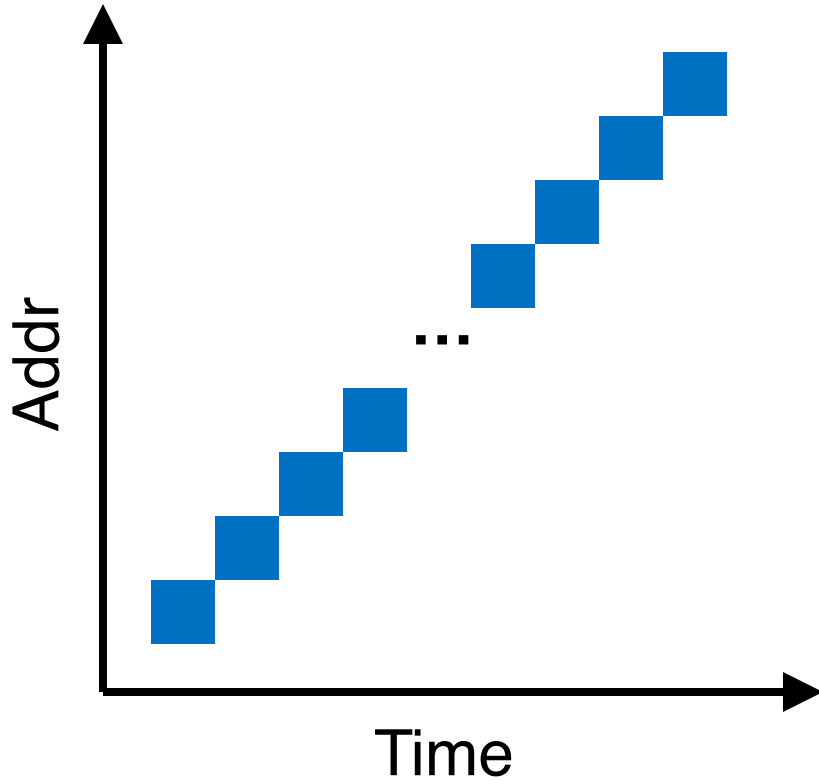
[point to block]



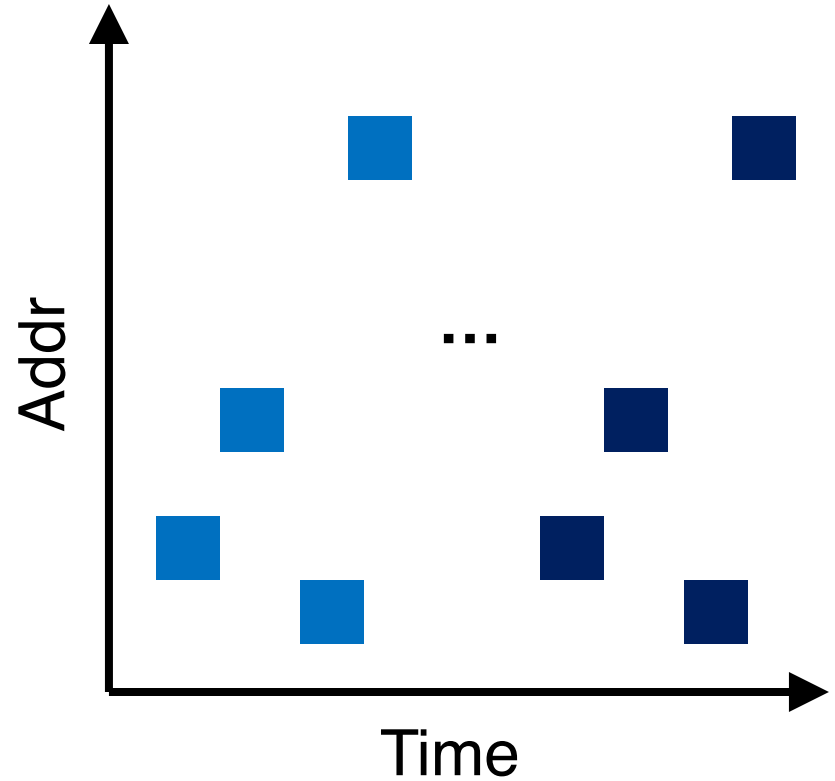
Locality & Data Layout

Review: Locality Types

Workload A

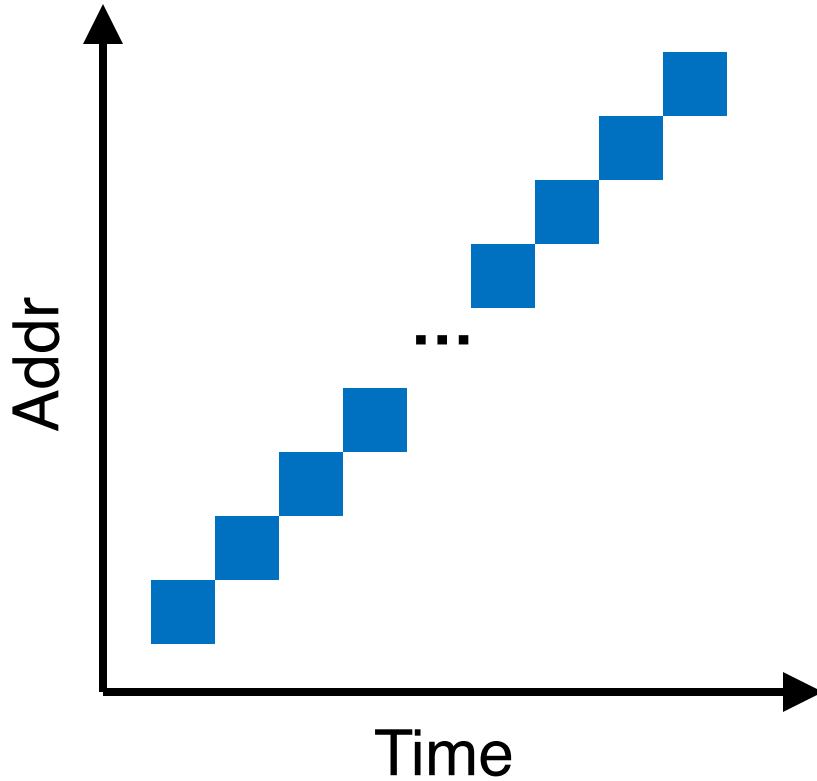


Workload B



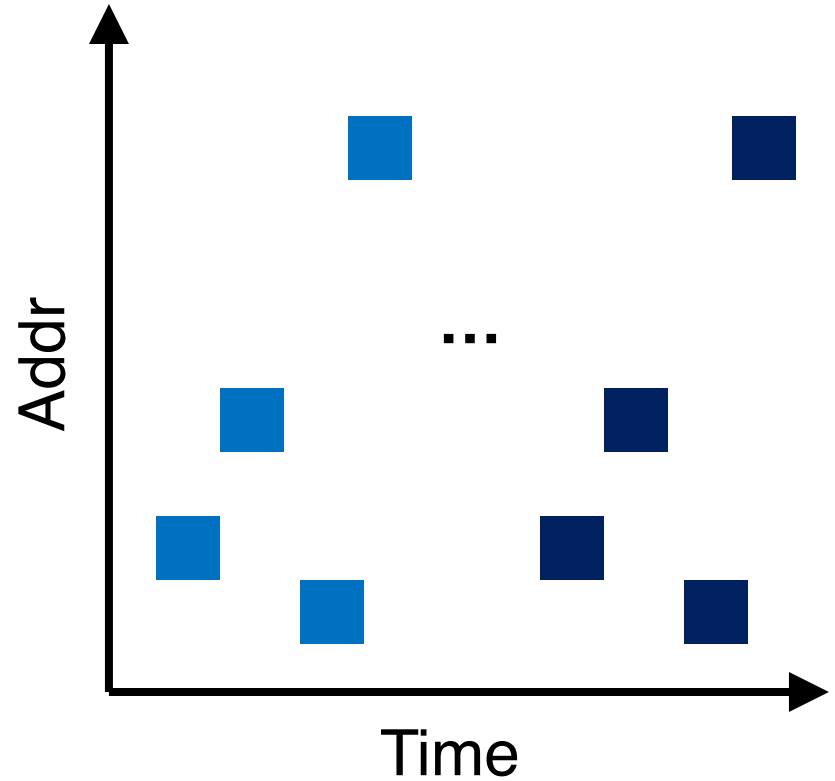
Review: Locality Types

Workload A



Spatial Locality

Workload B



Temporal Locality

Locality Usefulness in the Context of Disk-based File Systems

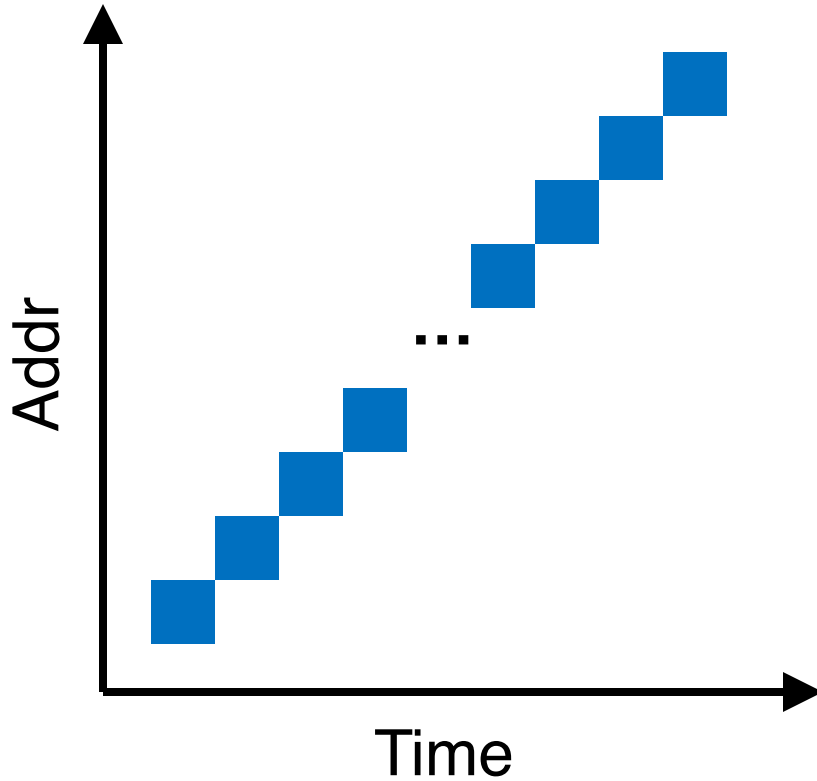
- What types of locality are useful for a [cache](#)?
- What types of locality are useful for a disk?

Locality Usefulness in the Context of Disk-based File Systems

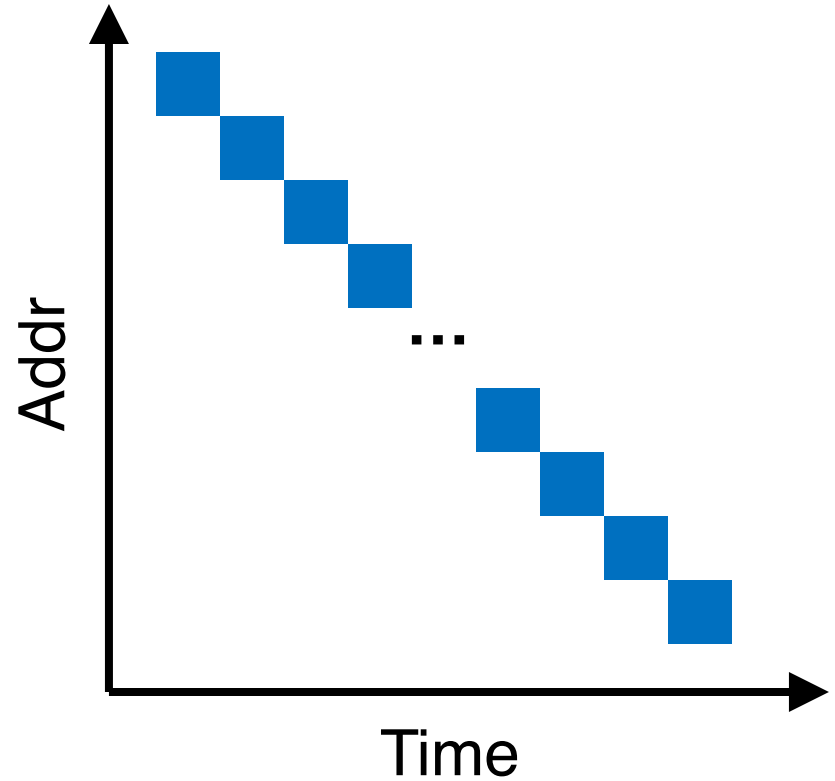
- What types of locality are useful for a **cache**?
 - Possibly, both spatial & temporal locality
- What types of locality are useful for a disk?
 - Spatial locality, since a disk sucks in random I/Os but can provide reasonably good sequential performance

Order Matters Now for FS on Disk

Workload A

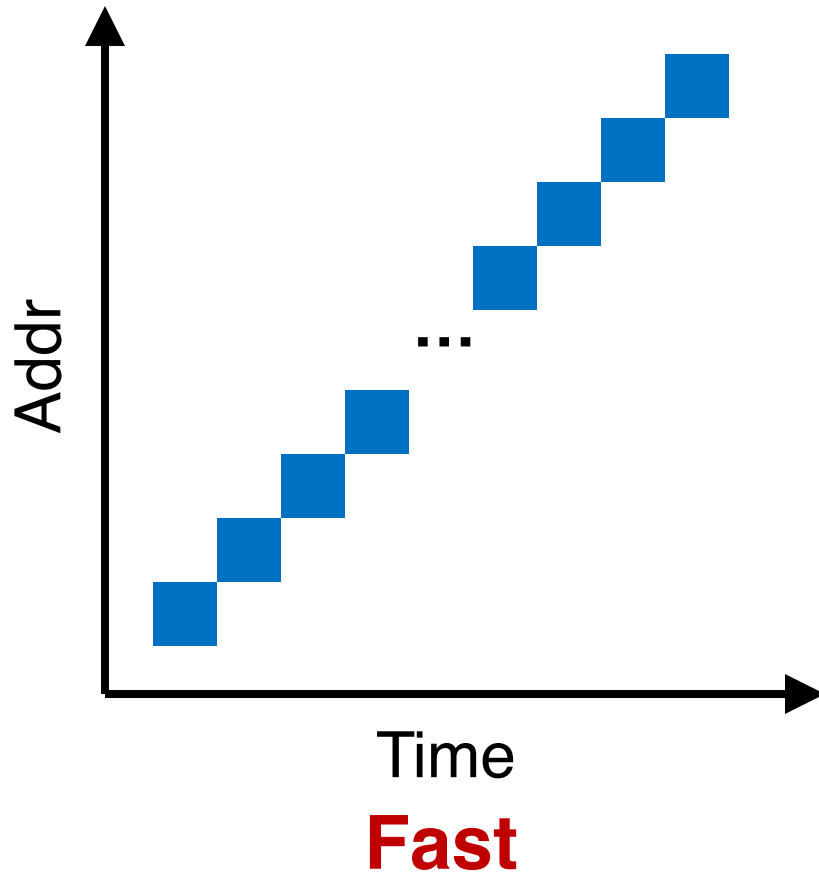


Workload B

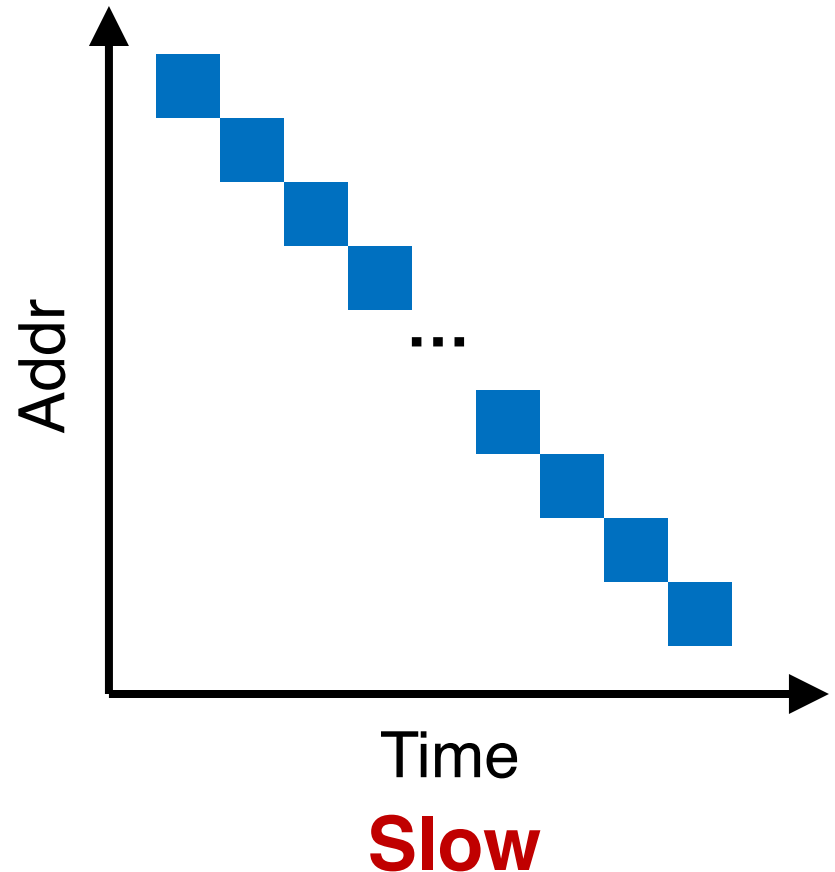


Order Matters Now for FS on Disk

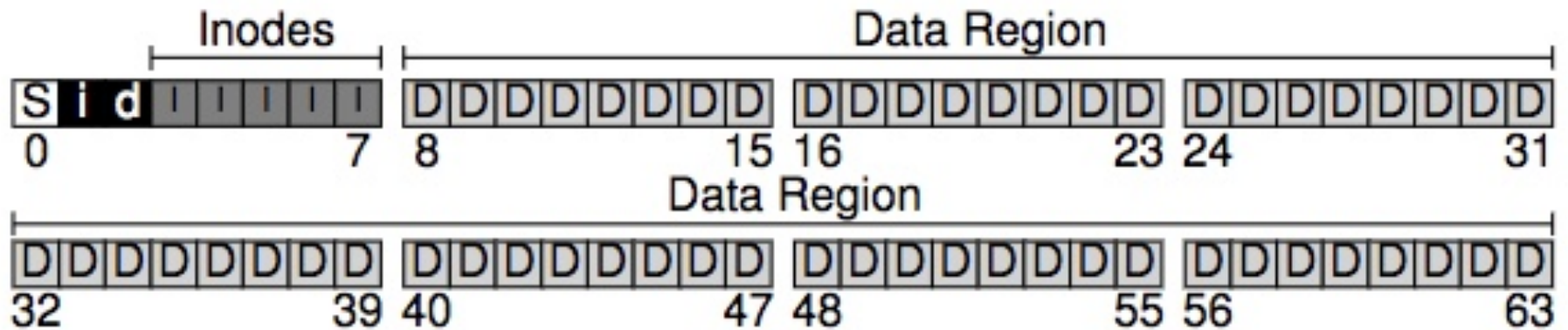
Workload A



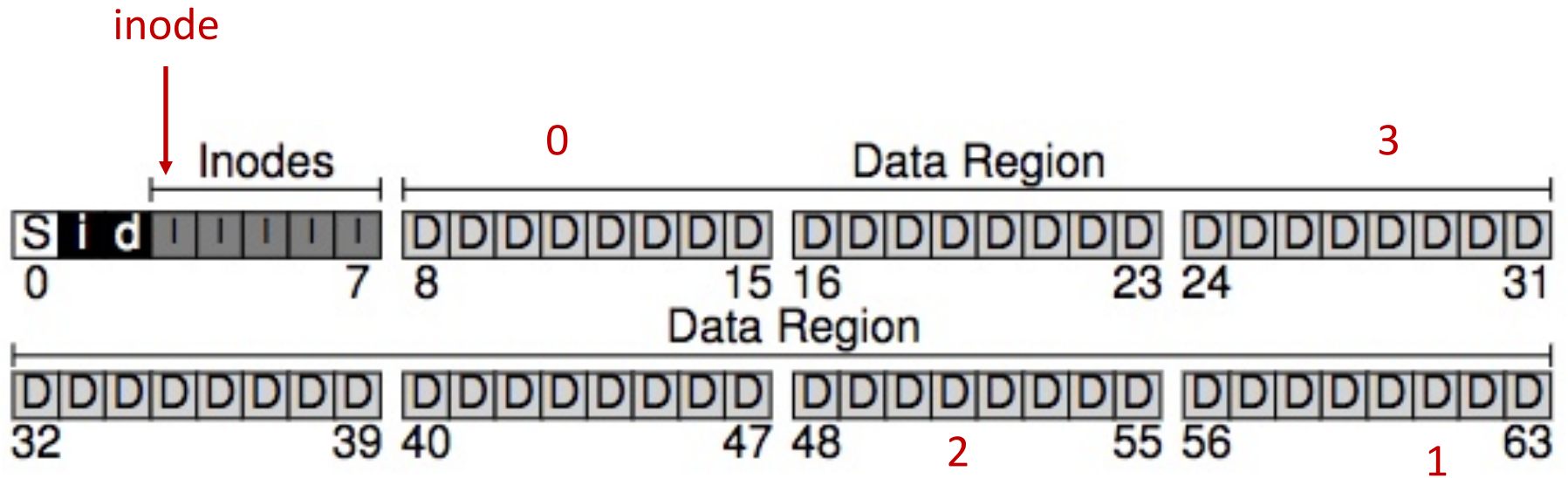
Workload B



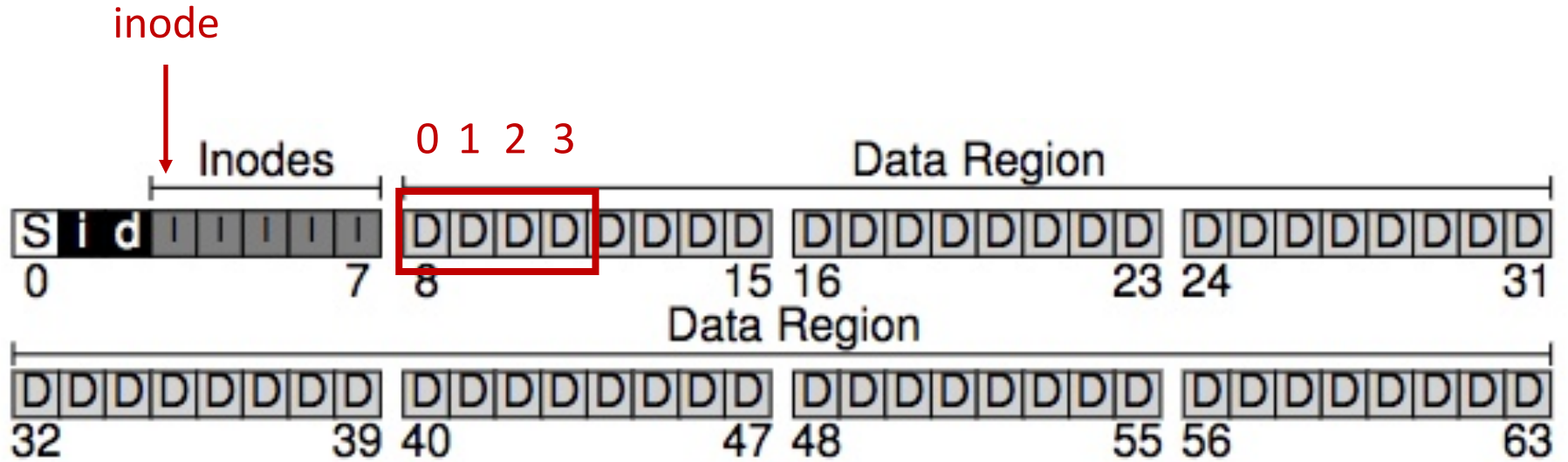
Policy: Choose Inode, Data Blocks



Bad File Layout



Better File Layout



Best File Layout

