

Basic Solution File System

Vitor de Freitas Benites
Fabio B. Rodrigues
Gustavo L. Flores

Orientador: Prof. Brivaldo Alves da Silva Jr.



Sumário

01

Introdução

02

**Trabalhos
Relacionados**

03

**Sistema de
Arquivos**

04

Implementação

05

**Resultados e
Discussão**

06

Conclusão

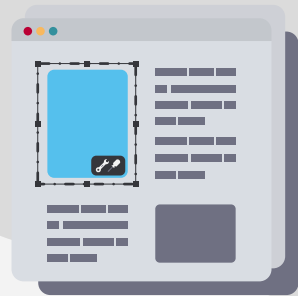
07

**Trabalhos
Futuros**



01

INTRODUÇÃO





CONTEXTO



COMPONENTE CENTRAL

Os sistemas de arquivos são parte essencial do sistema operacional, controlando como os dados são armazenados e acessados.



NOVA ERA DOS SSDs

O avanço dos SSDs trouxe novas demandas de desempenho, confiabilidade e integridade dos dados.



ORGANIZAÇÃO EFICIENTE

Eles organizam e abstraem o acesso aos dados, garantindo que o uso do armazenamento seja lógico e eficiente.



SOLUÇÕES MODERNAS

Sistemas como Ext4, Btrfs e ZFS utilizam *journaling* e *copy-on-write* para evitar corrupção e perda de dados.



CONTEXTO



LIMITAÇÕES PRESENTES

Mesmo com melhorias, ainda falta um mecanismo nativo de recuperação de arquivos excluídos, depende-se de ferramentas externas.



DESAFIO ATUAL

Persiste uma lacuna entre eficiência de armazenamento e preservação lógica dos dados.



OBJETIVOS



DESENVOLVER O BSFS

Implementar o Basic Solution File System (BSFS) em espaço de usuário, utilizando a linguagem C.



RECUPERAÇÃO NATIVA

Adicionar um mecanismo interno capaz de restaurar arquivos excluídos sem depender de ferramentas externas.



ESTRUTURAS INTERNAS

Implementar componentes como superbloco, mapa de blocos livres, árvores B e *i-nodes*.



AVALIAÇÃO COMPARATIVA

Executar testes de desempenho e integridade comparando o BSFS com os sistemas Ext4, Btrfs e ZFS.



OBJETIVOS



APRENDIZADO PRÁTICO

Mesmo com melhorias, ainda falta um mecanismo nativo de recuperação de arquivos excluídos, depende-se de ferramentas externas.



EXPERIMENTAÇÃO FLEXÍVEL

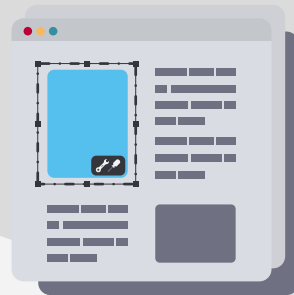
Persiste uma lacuna entre eficiência de armazenamento e preservação lógica dos dados.





02

Trabalhos Relacionados



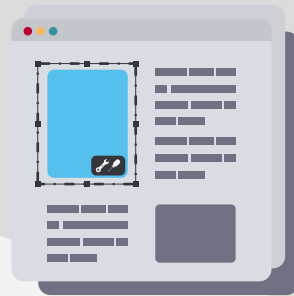
Trabalhos Relacionados

File System	Indexação de Metadados	Mecanismo de Endereçamento	Integridade em Falhas	Recuperação Nativa
Ext2	Tabela de <i>i-nodes</i> fixa	Indireção Múltipla	Depende de <i>fsck</i>	✗
Ext3	Tabela de <i>i-nodes</i> fixa + <i>HTree</i> para diretórios	Indireção Múltipla	<i>Journaling</i>	✗
Ext4	Tabela de <i>i-nodes</i> fixa + <i>HTree</i> + <i>Checksums</i>	Extents	<i>Journaling</i>	✗
ZFS	Árvores B Dinâmicas	COW + <i>Extents</i>	COW + <i>Checksums</i>	✗
BTRFS	Múltiplas Árvores B Dinâmicas	COW + <i>Extents</i>	COW + <i>Checksums</i>	✗
BSFS	Árvore B Tradicional	Bspan	✗	✓

The background on the left side of the slide features a large, light gray cloud shape. Inside this cloud, there is a smaller white cloud and a blue cloud. In the bottom left corner, there is a blue folder icon with a document icon inside it.

03

SISTEMA DE ARQUIVOS






Sistema de Arquivos

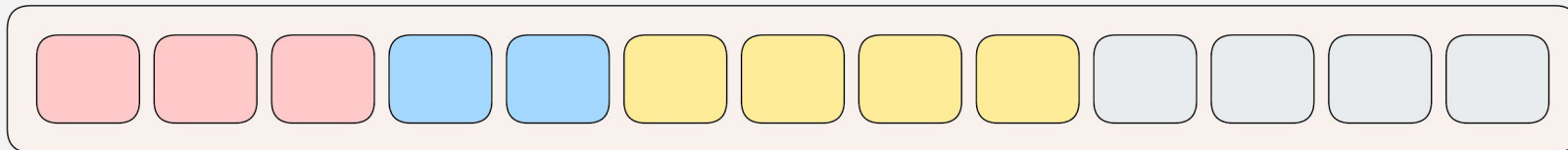
- **Definição:** Componente responsável por organizar e operar o armazenamento persistente.
- **Método:** Divisão do espaço físico em uma sequência de blocos de tamanho fixo.
- **Objetivo:** Padronizar as operações de leitura e escrita.
- **Contexto:** Base conceitual para o desenvolvimento do BSFS.






Exemplo de Sistema de Arquivos

Arquivo	Tamanho	Cor
texto.txt	12 KiB	
imagem.png	8 KiB	
audio.wav	16 KiB	

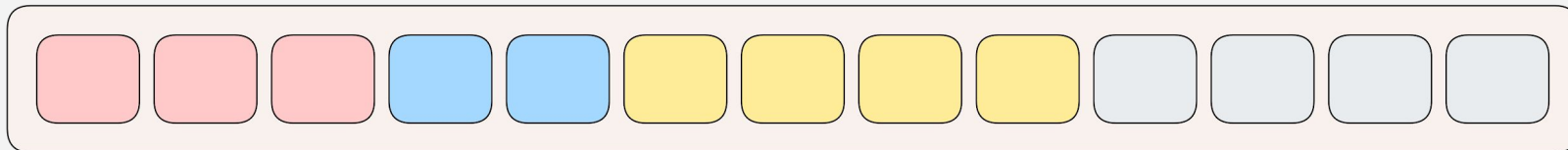
Blocos (4 KiB)






Exemplo de Sistema de Arquivos

Arquivo	Tamanho	Cor
texto.txt	12 KiB	
imagem.png	8 KiB	
audio.wav	16 KiB	

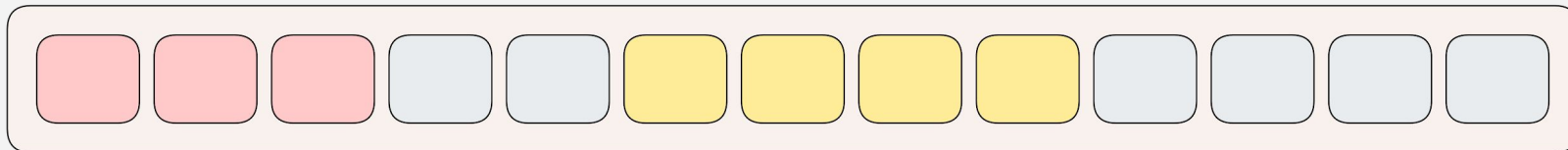
Blocos (4 KiB)





Exemplo de Sistema de Arquivos

Arquivo	Tamanho	Cor
texto.txt	12 KiB	
imagem.png	8 KiB	
audio.wav	16 KiB	

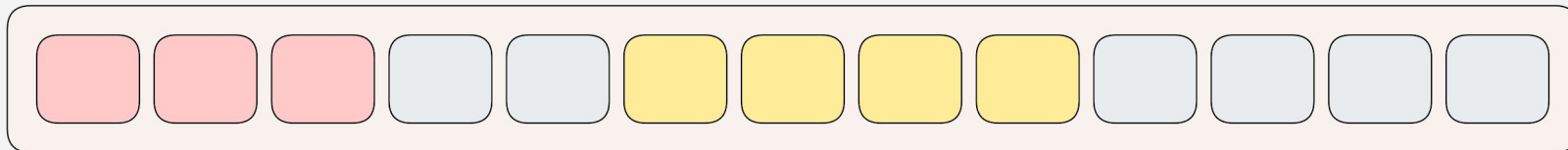
Blocos (4 KiB)






Exemplo de Sistema de Arquivos

Arquivo	Tamanho	Cor
texto.txt	12 KiB	
audio.wav	16 KiB	

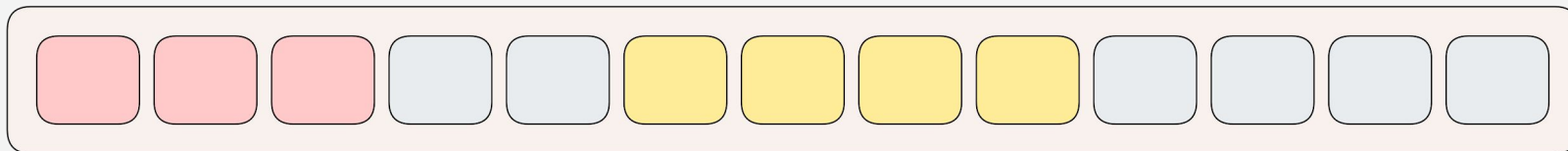
Blocos (4 KiB)






Exemplo de Sistema de Arquivos

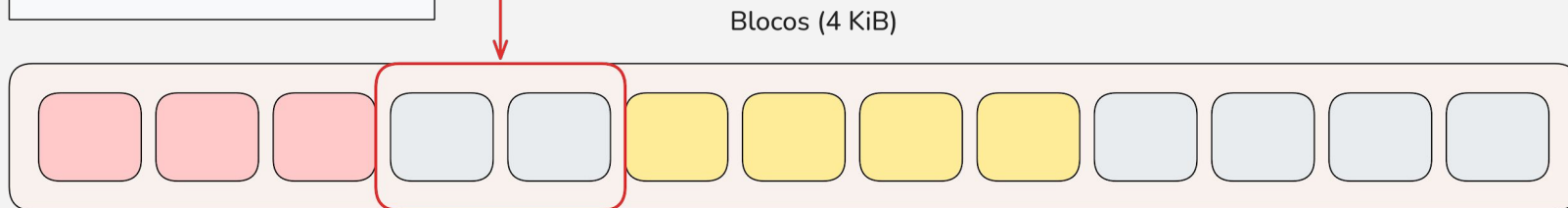
Arquivo	Tamanho	Cor
texto.txt	12 KiB	
audio.wav	16 KiB	
programa.bin	16 KiB	

Blocos (4 KiB)






Exemplo de Sistema de Arquivos

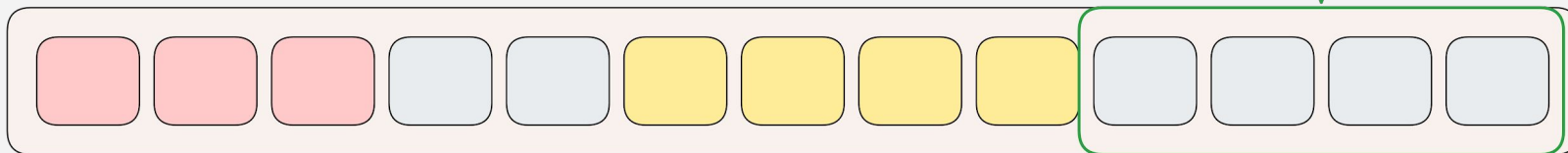
Arquivo	Tamanho	Cor
texto.txt	12 KiB	
audio.wav	16 KiB	
programa.bin	16 KiB	






Exemplo de Sistema de Arquivos

Arquivo	Tamanho	Cor
texto.txt	12 KiB	
audio.wav	16 KiB	
programa.bin	16 KiB	

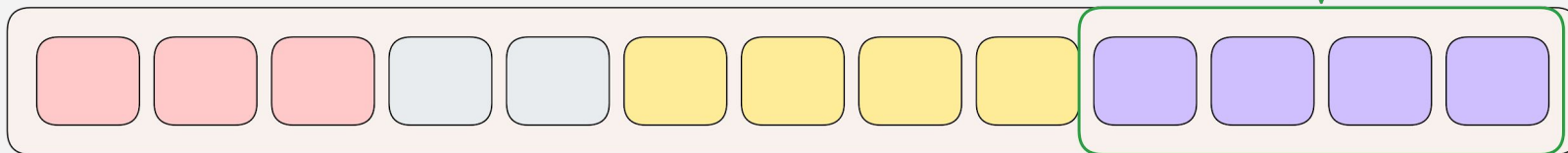
Blocos (4 KiB)







Exemplo de Sistema de Arquivos

Arquivo	Tamanho	Cor
texto.txt	12 KiB	
audio.wav	16 KiB	
programa.bin	16 KiB	

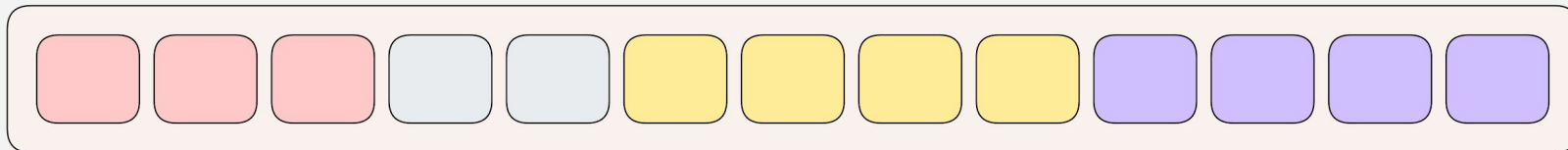
Blocos (4 KiB)







Exemplo de Sistema de Arquivos

Arquivo	Tamanho	Cor
texto.txt	12 KiB	
audio.wav	16 KiB	
programa.bin	16 KiB	
script.sh	2 KiB	

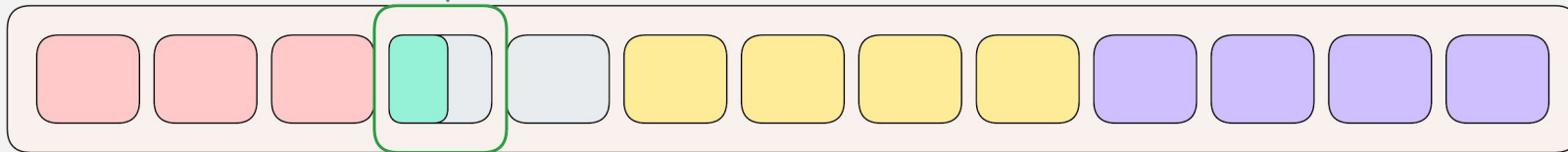
Blocos (4 KiB)



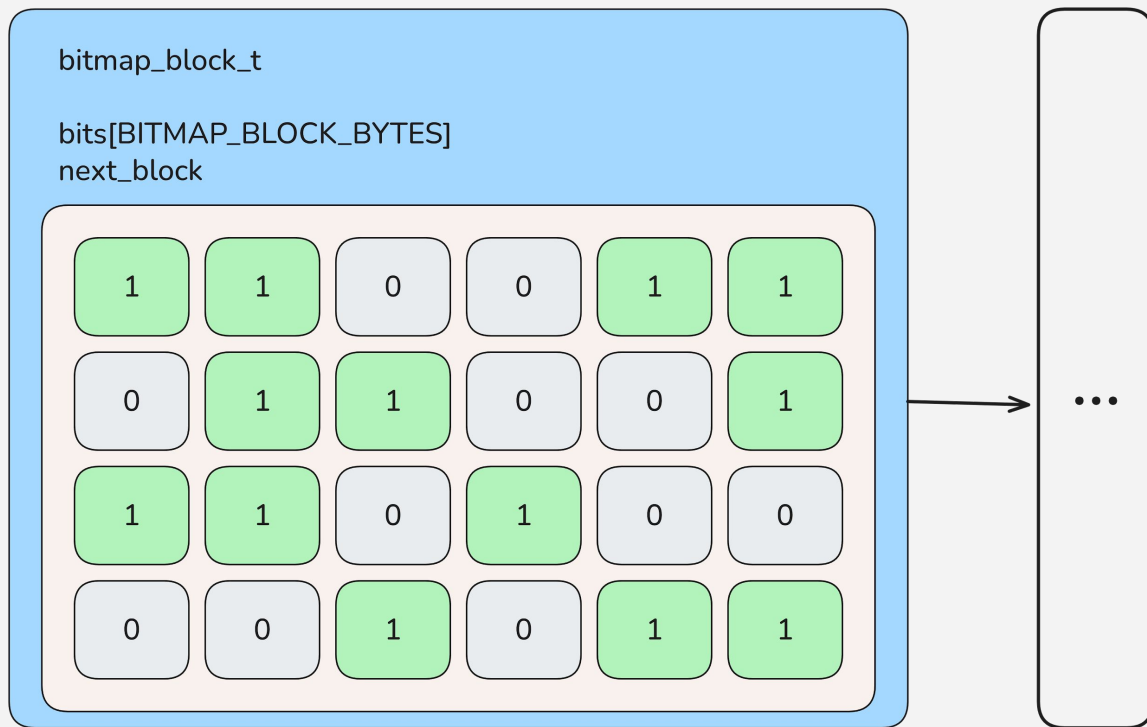
Exemplo de Sistema de Arquivos

Arquivo	Tamanho	Cor
texto.txt	12 KiB	
audio.wav	16 KiB	
programa.bin	16 KiB	
script.sh	2 KiB	

Blocos (4 KiB)



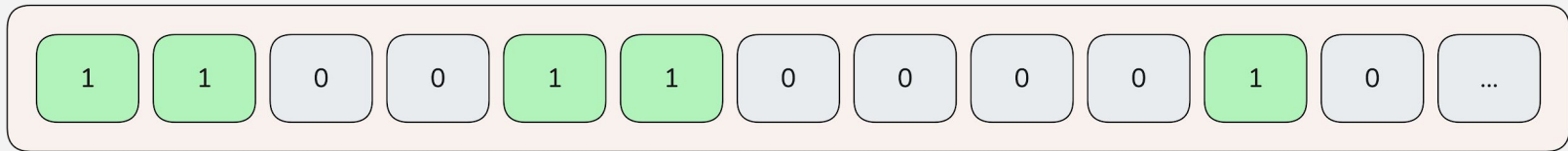
Mapeamento de Blocos



Alocação de Blocos: First-Fit

Arquivo	Tamanho
texto1.txt	8 KiB
system.log	8 KiB
anot.md	4 KiB

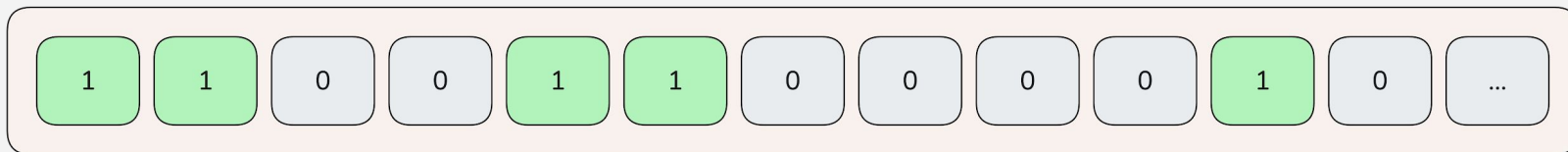
Bitmap



Alocação de Blocos: First-Fit

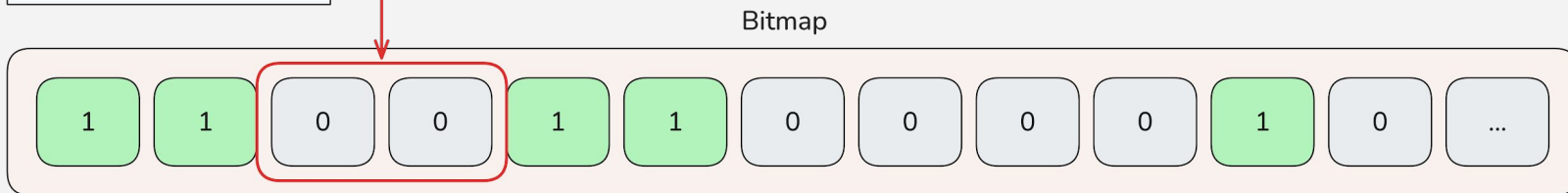
Arquivo	Tamanho
texto1.txt	8 KiB
system.log	8 KiB
anot.md	4 KiB
programa.bin	16 KiB

Bitmap



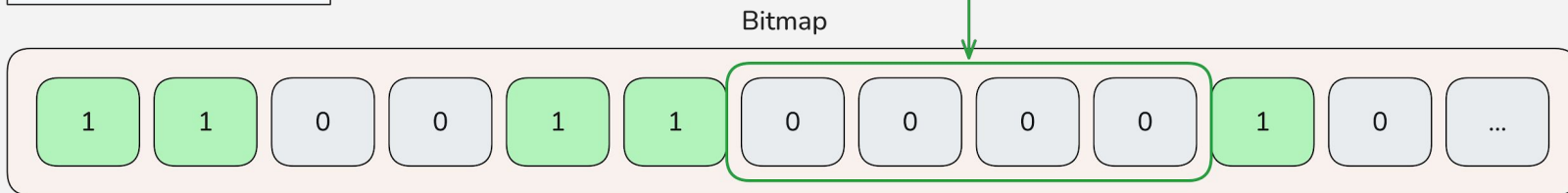
Alocação de Blocos: First-Fit

Arquivo	Tamanho
texto1.txt	8 KiB
system.log	8 KiB
anot.md	4 KiB
programa.bin	16 KiB



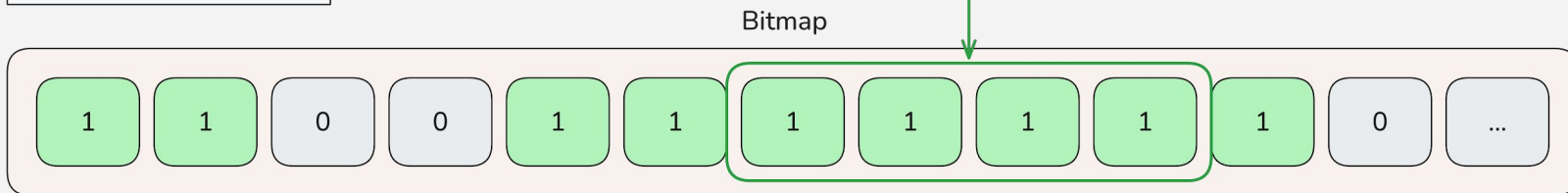
Alocação de Blocos: First-Fit

Arquivo	Tamanho
texto1.txt	8 KiB
system.log	8 KiB
anot.md	4 KiB
programa.bin	16 KiB



Alocação de Blocos: First-Fit

Arquivo	Tamanho
texto1.txt	8 KiB
system.log	8 KiB
anot.md	4 KiB
programa.bin	16 KiB



I-node

Função: Estrutura que representa o arquivo no sistema.

Conteúdo:

- Mapeamento dos blocos de dados.
- Identificador único, tamanho e permissões.
- **BSFS:** Armazenamento dinâmico em Árvore.

inode_t

inode_number
file_size
file_type
block_count
uid
gid
permissions
created_at
modified_at
accessed_at
direct_blocks
btree_root

Diretórios

- **Definição:** Arquivos especiais que mapeiam Nomes em *i-nodes*.
- **Estrutura:** Hierárquica e multinível.
- **Entrada de Diretório:**
 - Número do *i-node*
 - Tipo de Arquivo.
 - Nome do Arquivo.
 - Hash (*djb2*): Algoritmo rápido para mapear nomes em inteiros.

```
dir_entry_t
```

```
inode_number
```

```
file_type
```

```
file_size
```

```
name_hash
```

```
filename[256]
```

Diretórios

Exemplo

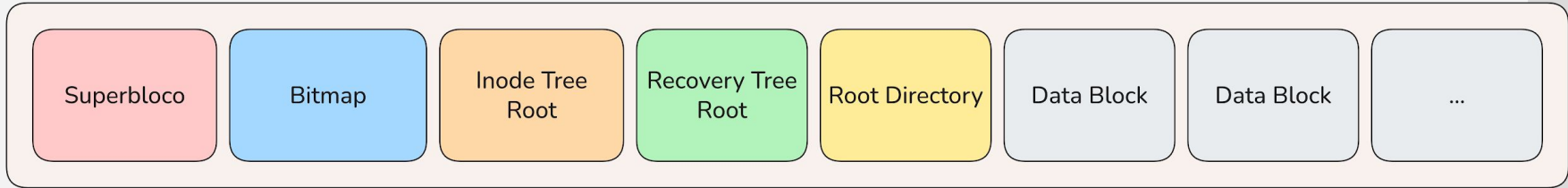
Nome	Hash	Tipo	Tamanho	<i>l-node</i>
main.c	221775931	Arquivo	12 KB	4
README.txt	241372033	Arquivo	3 KB	5
imagens	2612680009	Diretório	4 KB	6
config.sys	3704202481	Arquivo	1 KB	7
documentos	3711626278	Diretório	4 KB	8

Superbloco

- **Definição:** Estrutura central que contém a configuração global da partição.
- **Localização:** Fixo no primeiro bloco do dispositivo.
- **Dados Armazenados:**
 - **Info da Partição:** Magic Number, tamanho total, tamanho do bloco.
 - **Ponteiros:** Endereços iniciais das Árvores e dos Bitmaps.

```
superblock_t  
  
magic_number  
block_size  
fs_size  
total_blocks  
block_bitmap_start  
block_bitmap_total  
inode_root  
recovery_root  
data_block_start  
root_inode
```

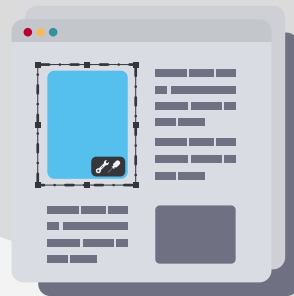
Basic Solution File System



The background features a large, light gray abstract shape on the left. Within this shape are two white clouds, one of which has a blue cloud underneath it. In the bottom left corner, there is a blue folder icon with a document inside.

04

Implementação

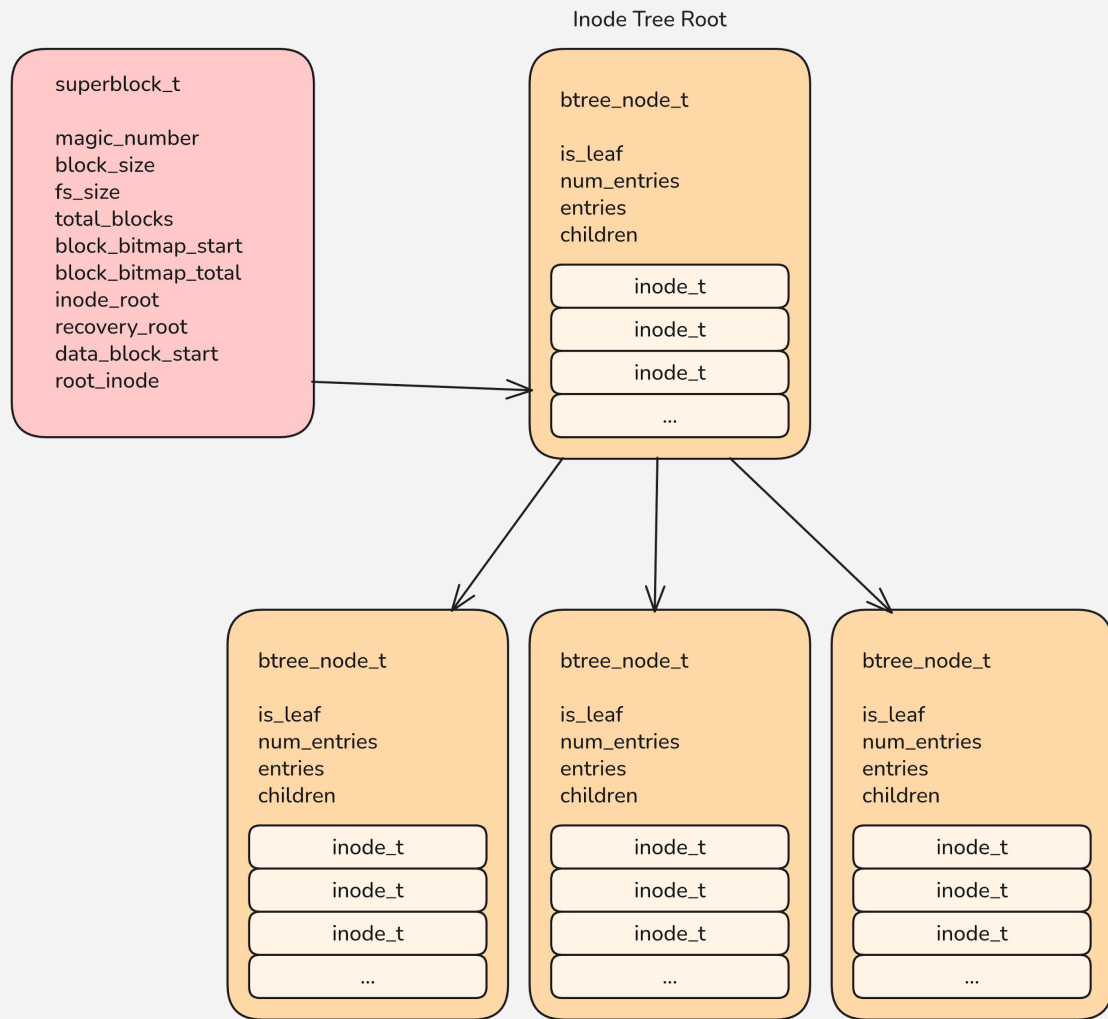


Árvore B

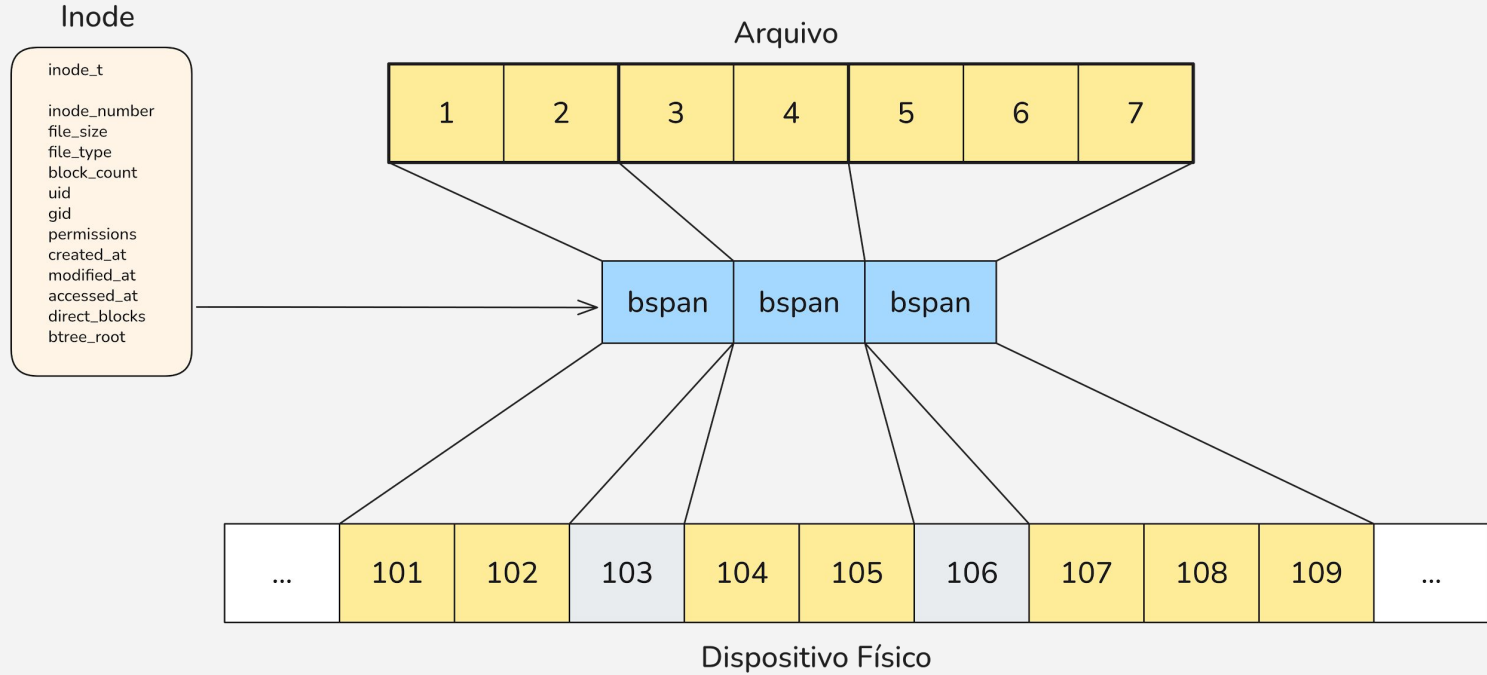
- Sistemas de Arquivos modernos utilizam árvores para gerenciamento de metadados.
- Árvores B+ contém metadados nos nós folha e índices nos nós internos.
- No BSFS foram implementadas Árvores B.
- Metadados estão todos os nós da árvore.



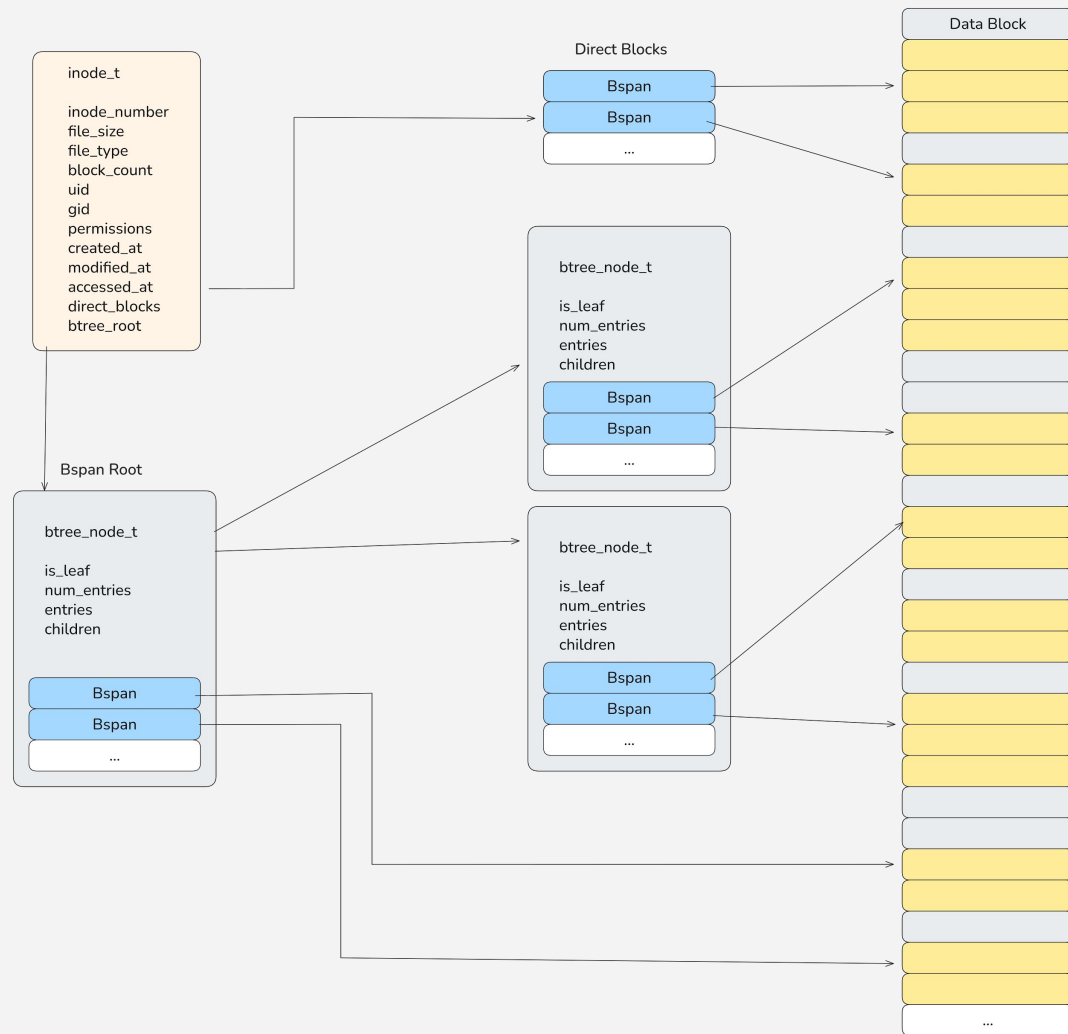
Árvore de *i-nodes*



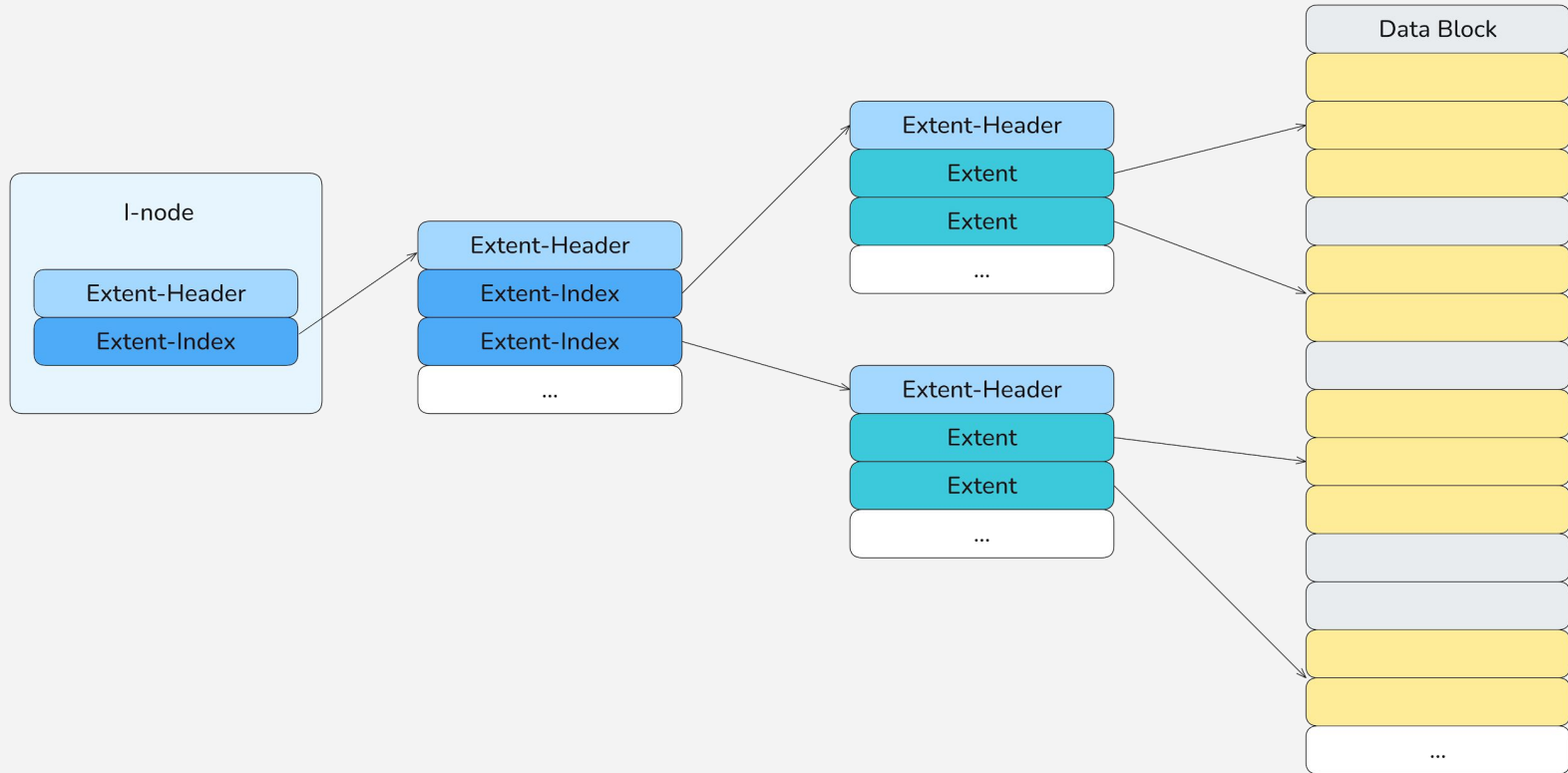
Estrutura *Bspan*



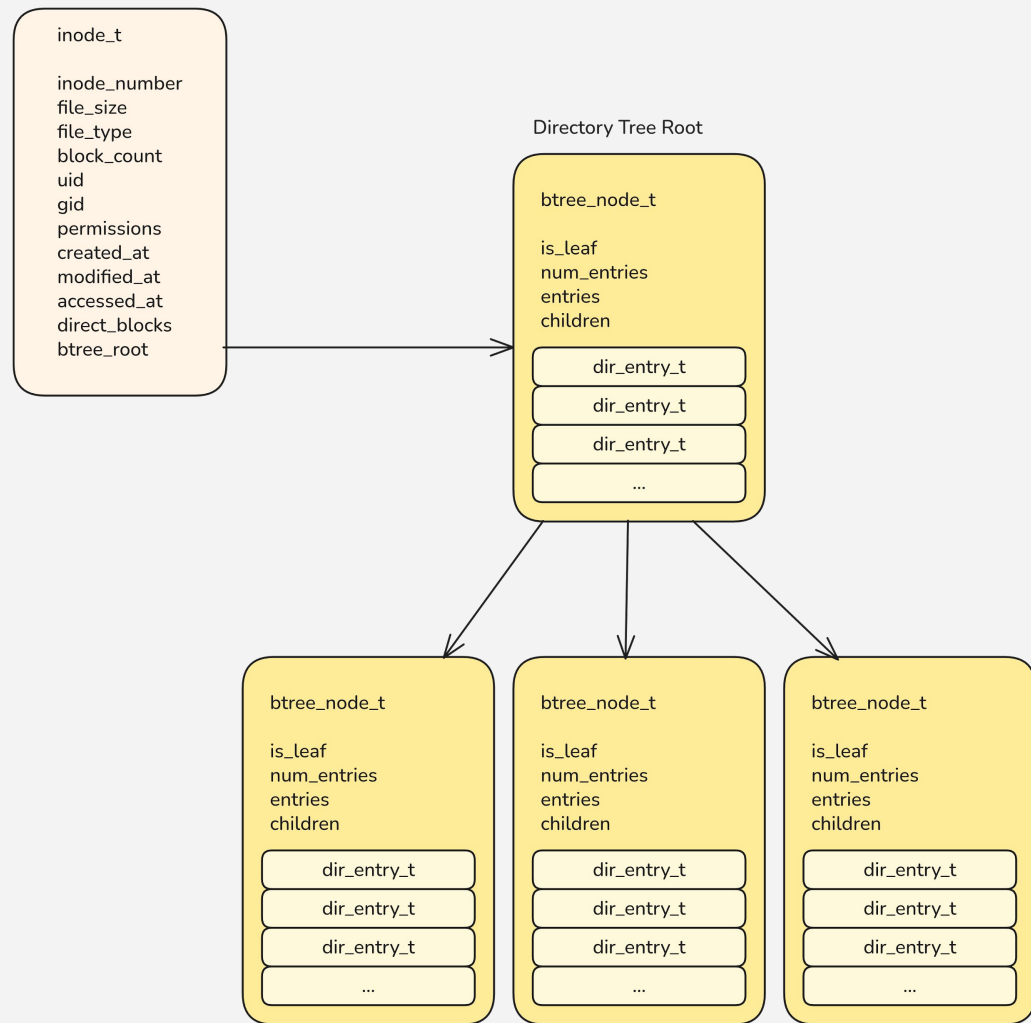
Árvore de *Bspan*



Árvore de *Extents* (Ext4)



Diretório



Remoção de Arquivos

1. Arquivo é removido pelo usuário.
2. Entrada de diretório é excluída.
3. Metadados do *i-node* são copiados para uma entrada de recuperação.
4. *I-node* é removido da árvore de *i-nodes*.
5. Entrada de recuperação é inserida na tabela de recuperação do sistema.
6. Arquivo pode ser recuperado.

rec_entry_t

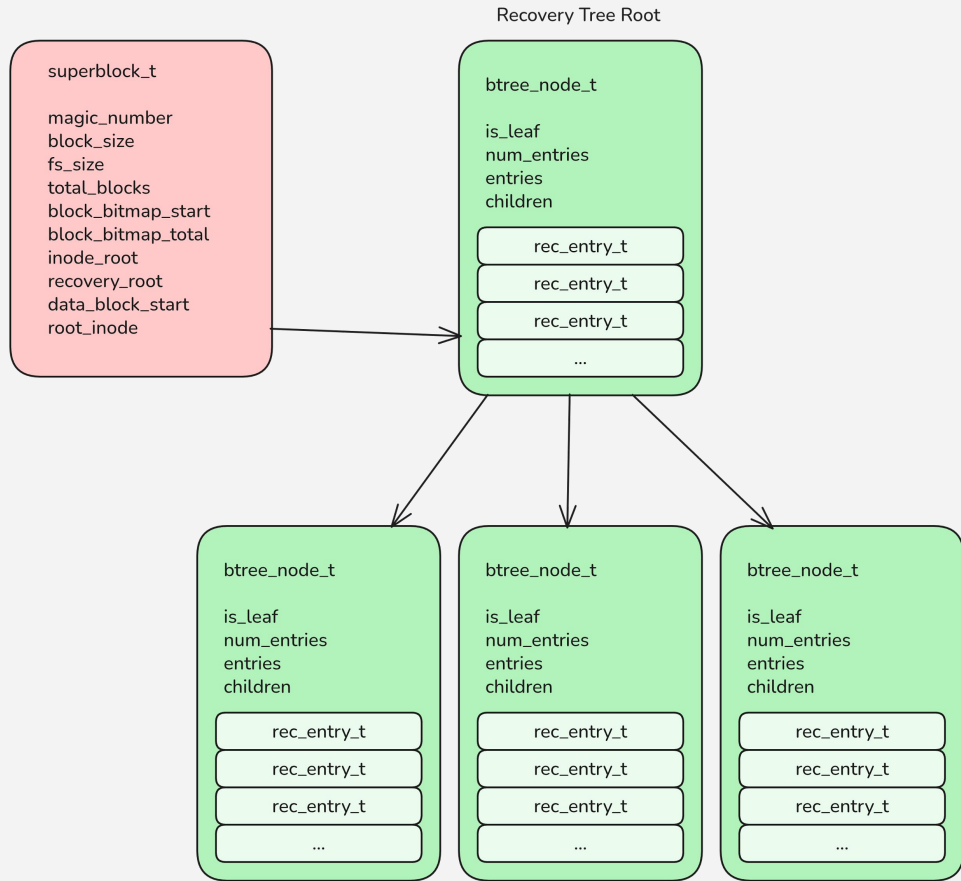
recovery_id
deleted_at
parent_ino
original_name[256]
inode_number
uid
gid
permissions
file_size
block_count
direct_blocks
btree_root
retention_until

Tabela de Recuperação

Exemplo

Identificador	Tamanho (<i>bytes</i>)	Excluído em	Nome
11	5242880	2025-12-01 22:07:01	arquivo.tar
12	2097152	2025-12-02 09:21:07	imagem.png
13	1048576	2025-12-02 14:44:10	audio.wav
14	10485760	2025-12-03 17:05:35	main.c
15	65536	2025-12-03 22:04:15	script.sh

Árvore de Recuperação



Interface de usuário

- ***mkfs.bsfs*** é utilizado para formatar dispositivos de blocos.
- ***browser*** é um *shell-like* em que o usuário pode inserir comandos.

```
> mkfs.bsfs /dev/sda1
BSFS formatted successfully
Partition size (bytes): 375244455936
Block size (bytes): 4096
Total blocks: 91612416
Block bitmap (start,total): 1,2799
Data block start: 2800
Inode root block: 2800
Root inode: 1
```

```
> browser /dev/sda1
BSFS Browser
Partition: /dev/sda1
Type "help" for available commands or "info"
for filesystem details
```

```
BSFS:/> info
Magic Number: 1397117762
Block size (bytes): 4096
File System Size (bytes): 375244455936
Total Blocks: 91612416
Block Bitmap Start: 1
Block Bitmap Total: 2799
Data Blocks Start: 2800
Inode Root Block: 2800
Recovery Root Block: 0
Root Inode: 1
```

```
BSFS:/> exit
Leaving BSFS Browser...
```

Comandos

Comandos de navegação

Comando	Função
<code>cd</code>	Altera o diretório atual.
<code>whoami</code>	Exibe o UID/GID atual e a umask.
<code>su</code>	Troca a identidade de sessão (teste).
<code>umask</code>	Exibe/define a umask.
<code>pwd</code>	Mostra o diretório de trabalho atual.
<code>ls</code>	Lista o conteúdo do diretório.
<code>list</code>	Alias para ls.

Comandos

Operações de diretório

Comando	Função
<code>mkdir</code>	Cria um novo diretório.
<code>rmdir</code>	Remove um diretório vazio.

Informações do sistema

Comando	Função
<code>info</code>	Exibe informações do sistema de arquivos.
<code>help</code>	Mostra o resumo de ajuda dos comandos.
<code>exit</code>	Encerra o browser do BSFS.

Comandos

Operações de arquivo

Comando	Função
<code>touch</code>	Cria um arquivo vazio.
<code>rm</code>	Remove um arquivo (opção <code>rm --all [dir]</code> varre arquivos com medição de tempo).
<code>cp</code>	Copia um arquivo.
<code>mv</code>	Move ou renomeia arquivo ou diretório.
<code>cat</code>	Exibe o conteúdo do arquivo.
<code>fill</code>	Cria muitos arquivos de determinado tamanho e reporta o tempo decorrido.
<code>chmod</code>	Altera permissões (octal, ex.: 755).
<code>chown</code>	Altera o proprietário (apenas root).
<code>chgrp</code>	Altera o grupo (apenas root).
<code>echo</code>	Exibe texto ou escreve em arquivo.
<code>dd</code>	Gera/duplica conteúdo com bloco e contagem.

Comandos de Recuperação

Recuperação de arquivos

Comando	Função
recovery	Lista entradas da árvore de recuperação com saída tabular.
restore	Restaura por número identificador ou por nome (com desambiguação). Verifica permissões e reconstrói o <i>i-node</i> a partir da cópia armazenada na entrada de recuperação.
purge	Limpa a árvore de recuperação: sem argumentos, remove entradas com mais de 30 dias; <i>purge [id]</i> remove uma entrada específica; <i>purge oldest [N]</i> remove as N mais antigas. Exibe a contagem removida.

Recuperação de Arquivos

1. Usuário utiliza o comando *restore* para recuperar um arquivo com seu nome ou identificador.
2. Sistema busca o arquivo na tabela de recuperação.
3. Um novo *i-node* é criado, e os metadados da entrada de recuperação são copiados.
4. *I-node* é inserido na árvore de *i-nodes*.
5. Entrada de diretório é inserida no diretório em que o arquivo havia sido previamente excluído.

Recuperação de Arquivos

```
BSFS:/> ls
```

```
Listing directory: .
```

Name	Type
----	----
.	DIR
..	DIR
system.log	FILE
file.txt	FILE
anot.md	FILE

```
BSFS:/> rm file.txt
```

```
File removed: file.txt
```

```
BSFS:/> rm anot.md
```

```
File removed: anot.md
```

```
BSFS:/> ls
```

```
Listing directory: .
```

Name	Type
----	----
.	DIR
..	DIR
system.log	FILE

```
BSFS:/> recovery
```

```
Recovery entries:
```

ID	Type	Size	Deleted At	Retain Until	ParentIno	Name
--	----	----	-----	-----	-----	-----
1	FILE	0	2025-12-01 23:04:33	2025-12-31 23:04:33	1	file.txt
2	FILE	0	2025-12-01 23:04:38	2025-12-31 23:04:38	1	anot.md

Recuperação de Arquivos

```
BSFS:/> restore file.txt
Restored 'file.txt' (id 1)
```

```
BSFS:/> restore anot.md
Restored 'anot.md' (id 2)
```

```
BSFS:/> ls
Listing directory: .
Name                               Type
----                               -
.                                   DIR
..                                  DIR
system.log                         FILE
file.txt                           FILE
anot.md                             FILE
```

```
BSFS:/> recovery
```

Recovery entries:

ID	Type	Size	Deleted At	Retain Until	ParentIno	Name
--	----	----	-----	-----	-----	----

No recovery entries.



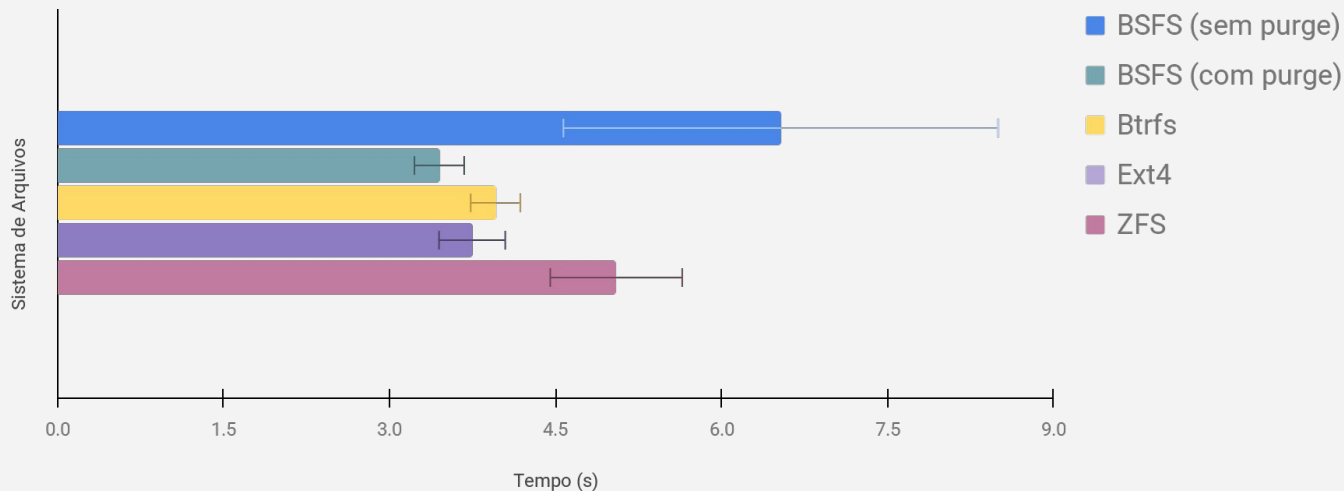
05

Resultados & Discussão

Comparativo

BSFS vs Ext4/BTrfs/ZFS

Gravação de Arquivos (100x10 MiB)

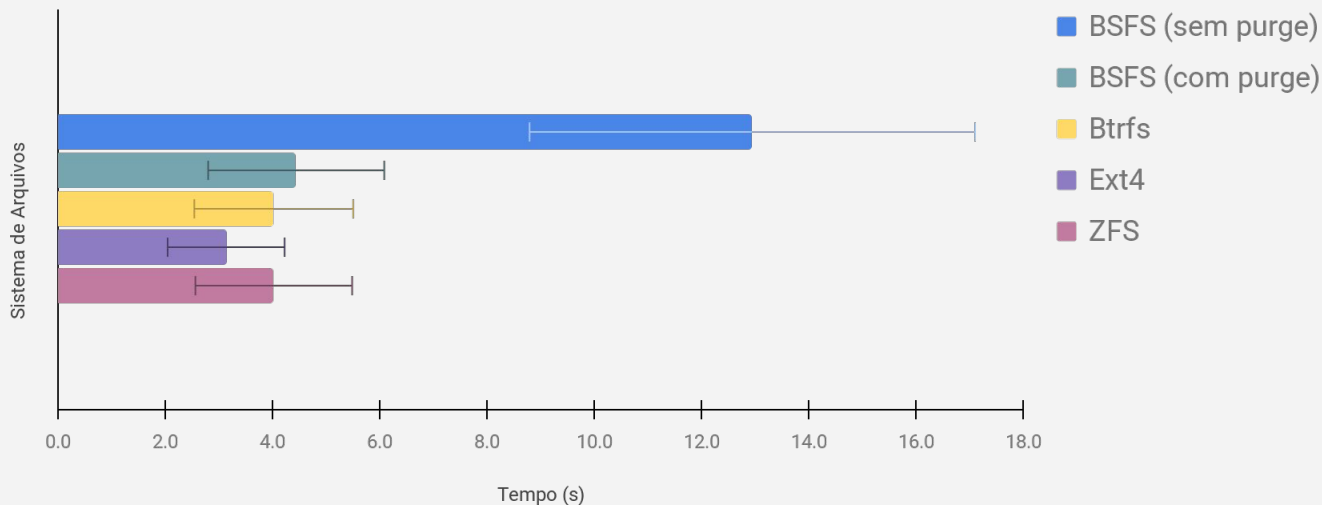


O gráfico apresenta a média de dez execuções na gravação de 100 arquivos de 10 MiB em cada um dos sistemas de arquivos.

Comparativo

BSFS vs Ext4/BTrfs/ZFS

Gravação de Arquivos (1x1 GiB)

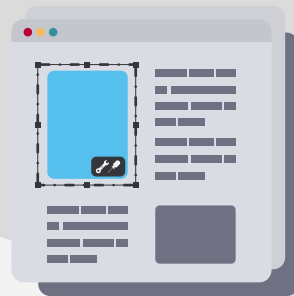


O gráfico apresenta a média de dez execuções na gravação de um arquivo de 1 GiB em cada um dos sistemas de arquivos



06

Conclusão



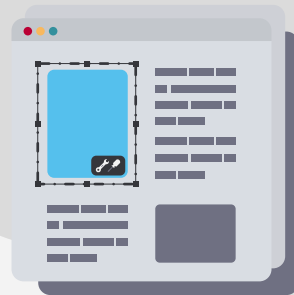
Conclusão

- Implementação de um sistema de arquivos funcional.
- Mecanismo de recuperação.
- Principais limitações estão relacionadas à estruturas de gerenciamento de espaço livre.



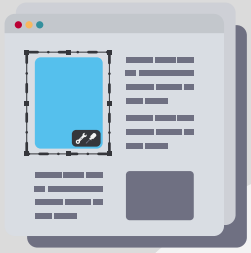
07

Trabalhos Futuros



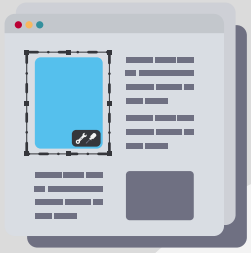
Trabalhos Futuros

- **Operação em Modo Kernel:** eliminar o *overhead* das chamadas de sistema (transições de contexto).
- **Redesign do Gerenciamento de Blocos Livres:** tratar a fragmentação e retenção de blocos pela Árvore de Recuperação.
- **Concorrência e Paralelismo:** aproveitar arquiteturas multicore para permitir o processamento simultâneo de diferentes regiões das Árvores B de metadados.
- **Journaling ou COW:** garantir a consistência da Árvore de Recuperação em falhas abruptas.



Obrigado!





Dúvidas?

