

Sistemas Operativos (Pipes)

IPC - Inter-Process Communication

- Mecanismos de comunicação inter-processo:
 - através de ficheiros “normais”
 - requer que leitores e escritores sincronizem com *muito cuidado*;
 - requer que os dados sejam escritos para disco (lento);
 - gestão de nomes, permissões, interferência de outros processos.
 - ***pipes* anónimos**
 - **entre processos relacionados (pai/filho, entre filhos do mesmo pai, ...)**
 - *pipes* com nome
 - sinais
 - *sockets*, semáforos, memória partilhada

Pipes anónimos

- Buffer em memória, com a sincronização entre produtores (escritas) e consumidores (leituras) gerida pelo kernel:
 - consumidor bloqueia na leitura se não há nada para ler.
 - produtor bloqueia na escrita se não há espaço para escrever.
- Semântica de comunicação FIFO, num só sentido
- Permitem combinar programas sem os modificar:
 - por exemplo, na *shell*, `ls | less`

Chamadas ao sistema

- Bibliotecas
 - `<unistd.h>` - definições e declarações de chamadas

Chamadas ao sistema

```
int pipe(int fildes[2]);
```

- Recebe um array de 2 inteiros no qual aloca 2 descritores de ficheiros:
 - dados escritos para o descritor `fildes[1]` podem ser lidos através do descritor `fildes[0]`.
- Devolve :
 - 0 caso a criação tenha sido bem sucedida;
 - -1 caso tenha ocorrido um erro.

Pipes anónimos

```
int pipe(int fildes[2]);
```

Processo

1123

TP

0	
1	
2	
3	
4	

fildes[0]

fildes[1]

TF

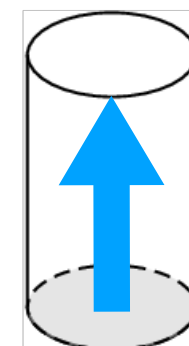
modo	r
pos	0
#ref	1
vptra	

modo	w
pos	0
#ref	1
vptra	

v-node

...	

leitura



buffer

...	

escrita

Pipes anónimos

```
int fork();
```

Processo

1123

fildes[0]

fildes[1]

TP

0	
1	
2	
3	
4	

1523

fildes[0]

fildes[1]

0	
1	
2	
3	
4	

TF

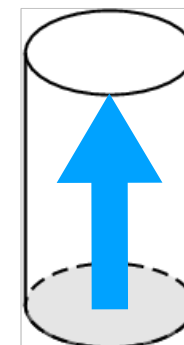
modo	r
pos	0
#ref	2
vptr	

modo	w
pos	0
#ref	2
vptr	

v-node

...	

leitura



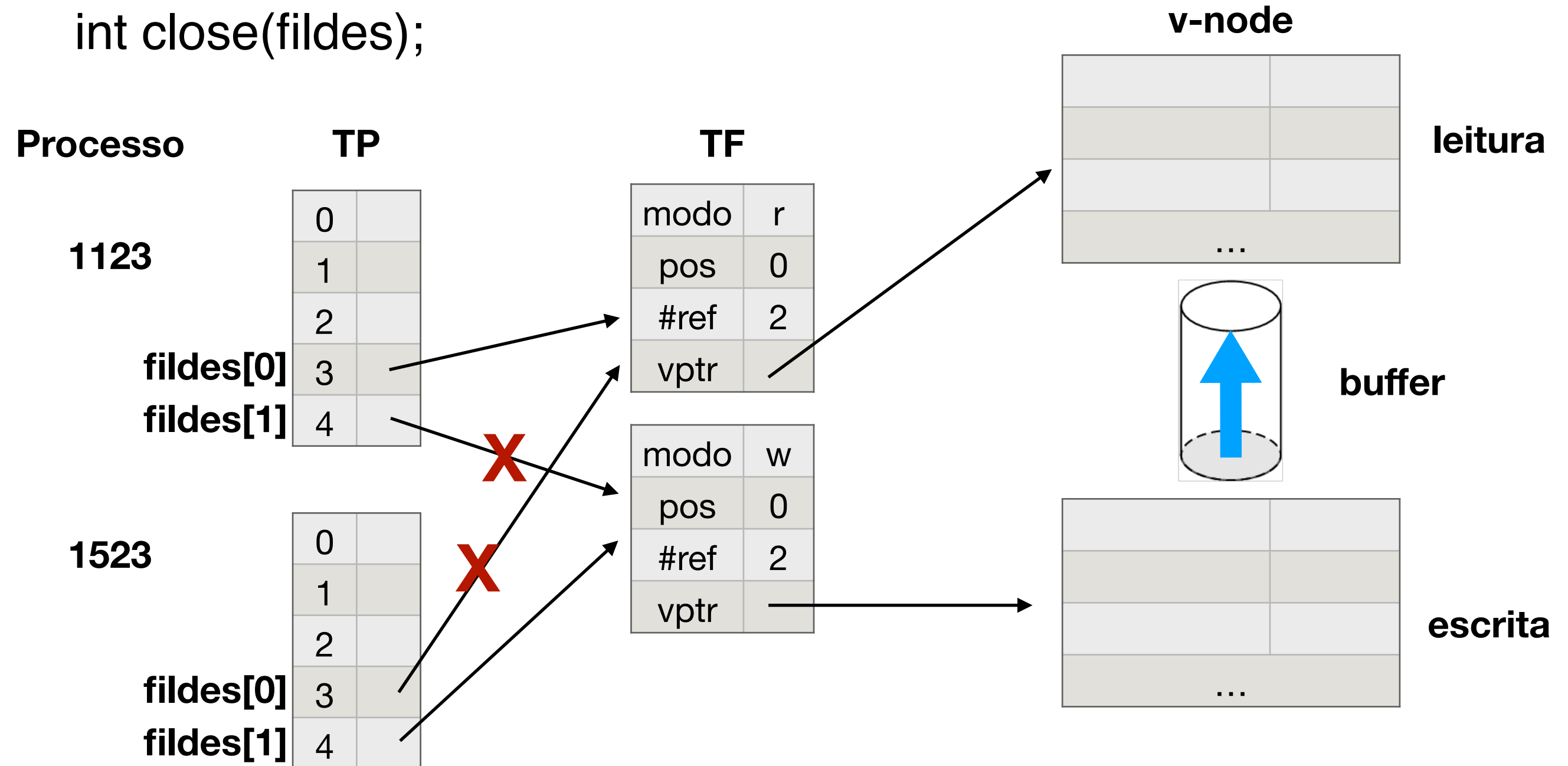
buffer

...	

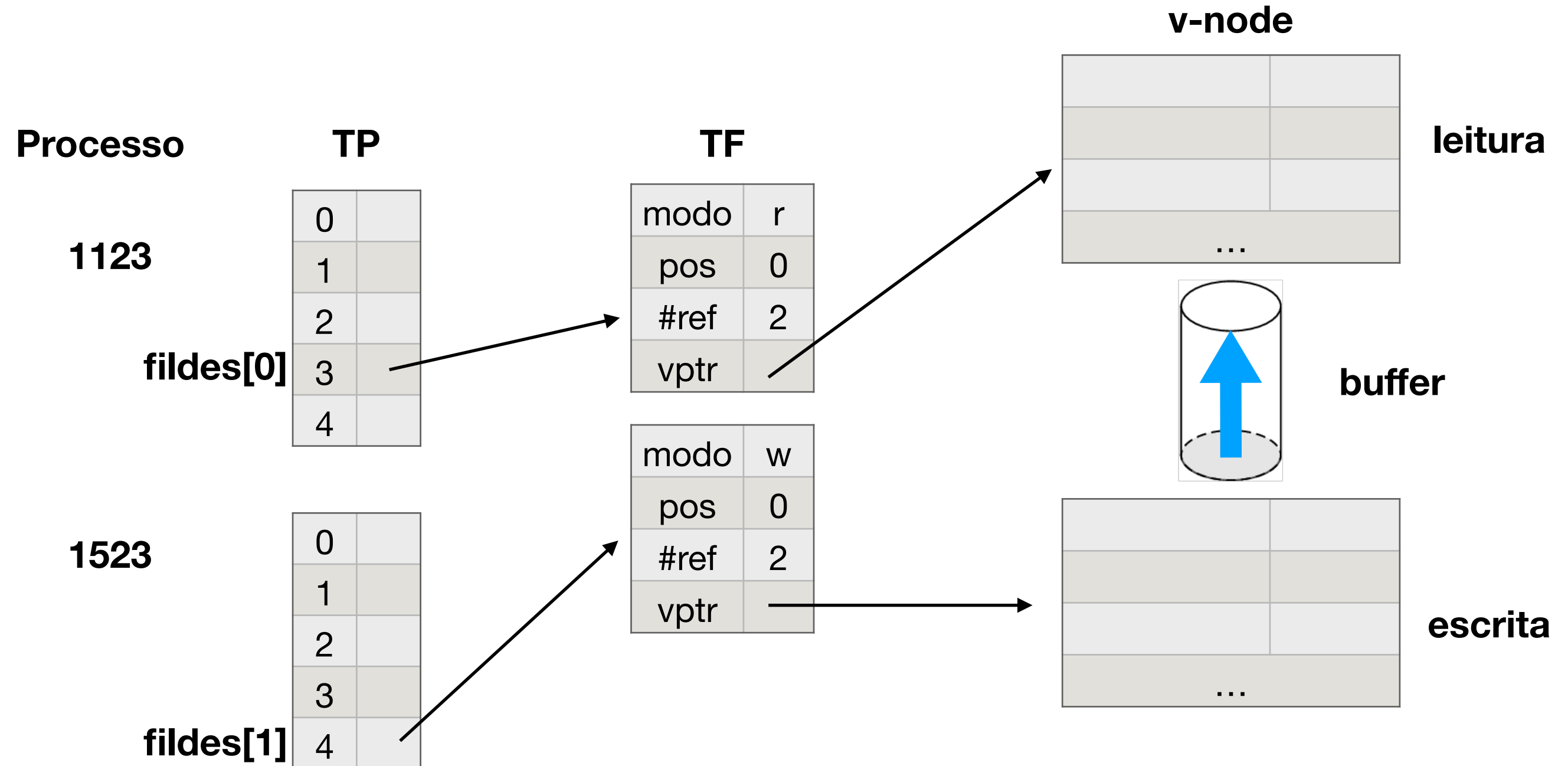
escrita

Pipes anónimos

```
int close(fildes);
```



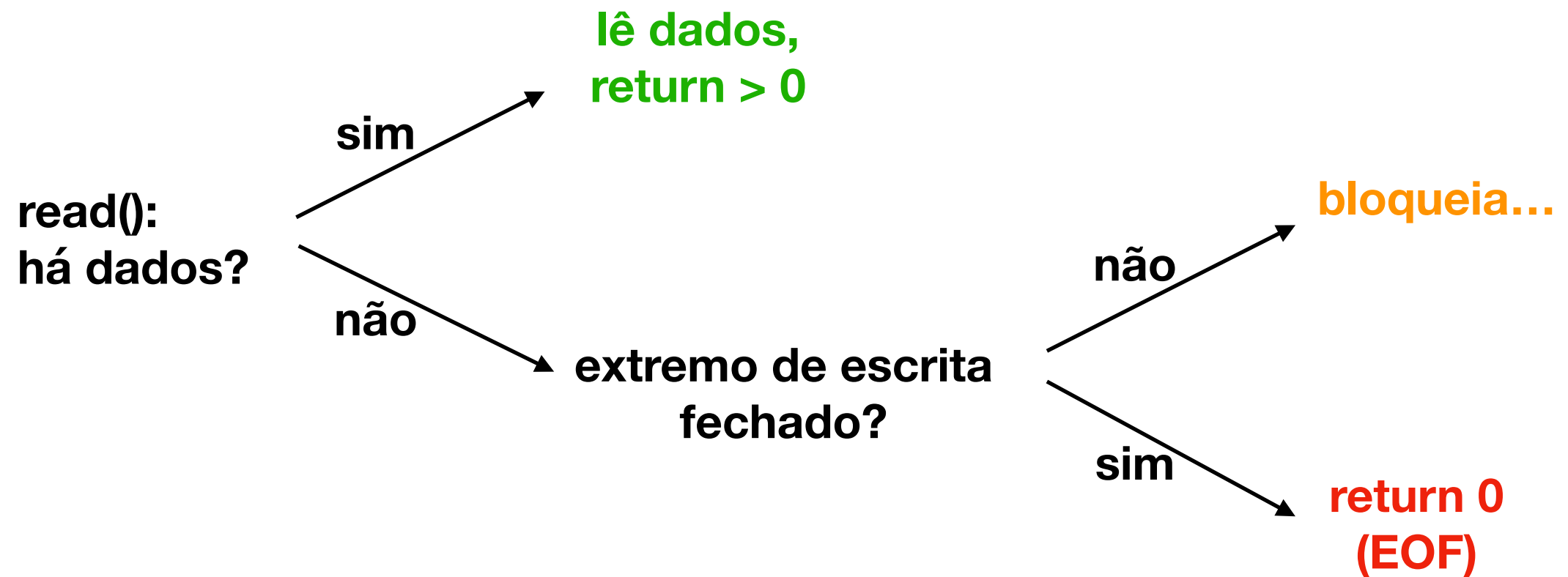
Pipes anónimos



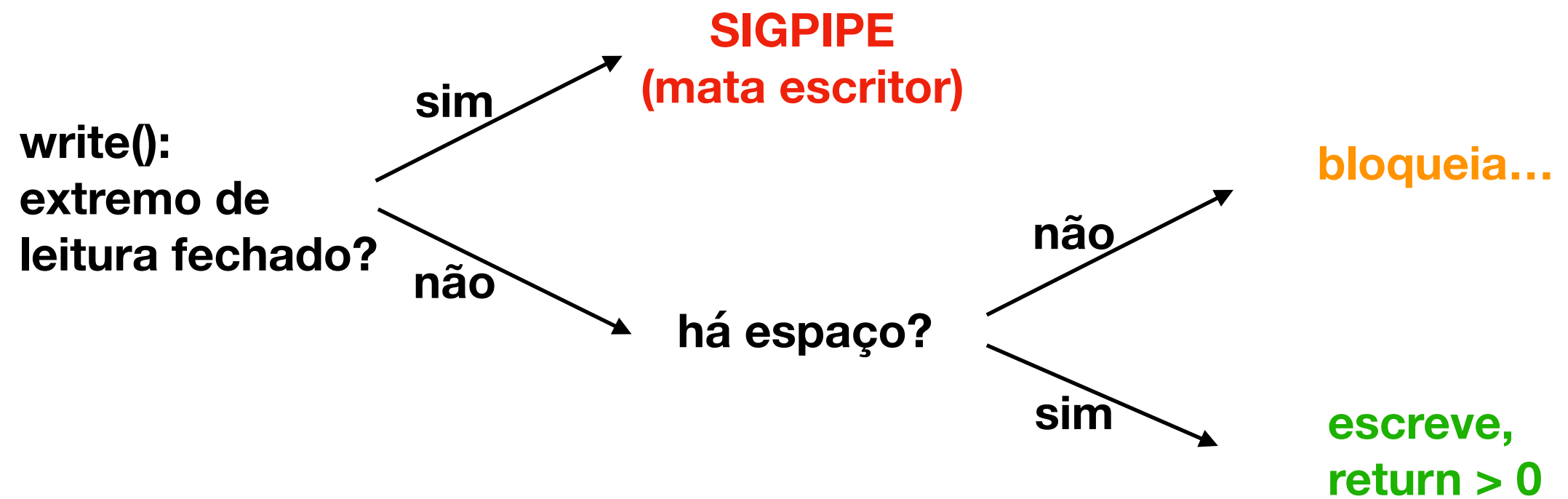
Considerações

- A leitura de um *pipe* apenas devolve EOF se todos os descritores para o extremo de escrita estiverem fechados.
 - Processos que vão ler de um *pipe* devem fechar o extremo de escrita e vice-versa.
 - Descritores abertos que deveriam estar fechados podem dar origem a *deadlocks*.
- A tentativa de escrever para um *pipe* cujo extremo de leitura esteja fechado provoca SIGPIPE.
- O tamanho do buffer varia e não se deve depender deste.
- Nunca redireccionar stdin para o extremo de escrita e vice-versa.

Leitura



Escrita



Material de Apoio

- https://www.gnu.org/software/libc/manual/html_node/Pipes-and-FIFOs.html#Pipes-and-FIFOs
- <https://www.usna.edu/Users/cs/wcbrown/courses/IC221/classes/L13/Class.html>
- <https://www.usna.edu/Users/cs/aviv/classes/ic221/s16/lec/21/lec.html>