Pai

1) cria pipe

2) fork()

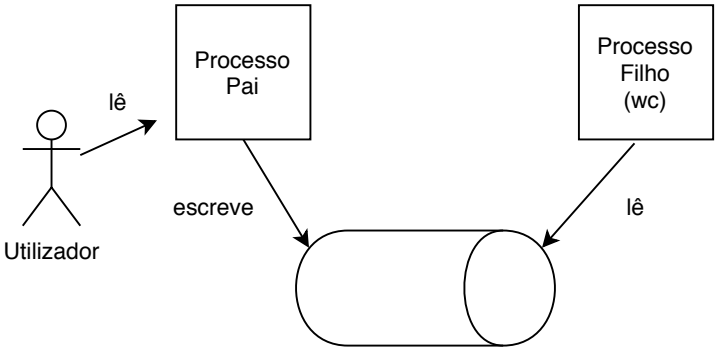
3) Ciclo de leitura do stdin e escrita do conteúdo lido para o extremo de escrita do pipe

4) espera pelo filho terminar

Filho

a) redireciona stdin para extermo leitura (ler) do pipe

b) exec(wc)



Notas

O passo 3) e os passos a) e b) são concorrentes

O exec herda descritores redirecionados

Não esquecer de fechar extremos de pipe que não estão a ser usados pelo processo

Não esquecer de fechar o descritor copiado (origem) do dup

Pai

1) cria pipe

2) fork()

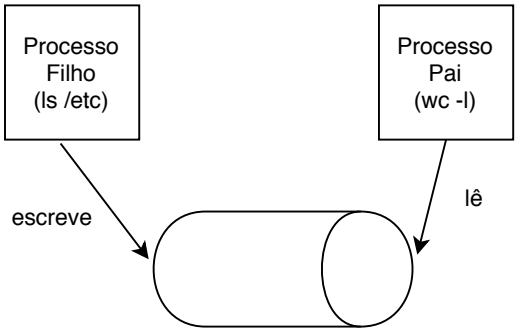
3) redireciona stdin para ler do extremo de leitura do pipe

4) exec(wc -l)

Filho

a) redireciona stdout para escrever no extremo de escrita do pipe

b) exec(ls /etc)



Notas

O passo 3) e 4) e os passos a) e b) são concorrentes

O exec herda descritores redirecionados

Não esquecer de fechar extremos de pipe que não estão a ser usados pelo processo

Não esquecer de fechar o descritor copiado (origem) dos dups

Podia ser o pai a executar o "ls" e filho o "wc"

Pai

1) cria pipe

2) fork() x 2

3) wait() x 2

Filho 0

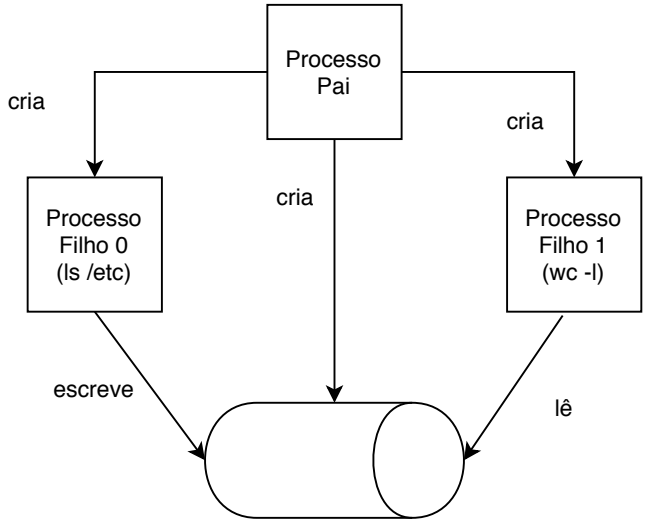
a.1) redireciona stdout para escrever no extremo de escrita do pipe

b.1) exec(ls /etc)

Filho 1

a.2) redireciona stdin para ler do extremo de leitura do pipe

b.2) exec(wc -l)



Notas

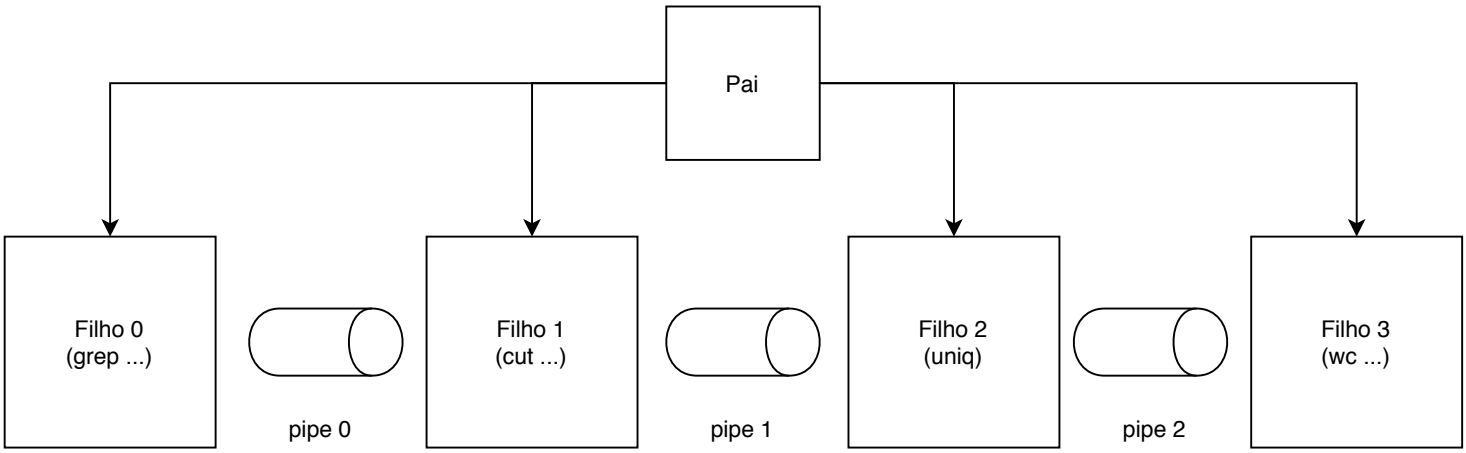
De novo concorrência entre pai e filhos

O exec herda descritores redirecionados

Não esquecer de fechar extremos de pipe que não estão a ser usados pelos processos filhos e pelo pai.

O pai só pode fechar um extremo depois de fazer fork() de forma a garantie que o filho tem uma cópia onde pode ler/escrever

Não esquecer de fechar o descritor copiado (origem) do dups



Notas

Para N comandos existem N-1 pipes

pipe[N-1][2] - array pipes

cmd[N] - array comandos

if(i==0){}

else if(i==N){}

else {}

Pai

1) While(N){

pipe() //nota se pipe fosse num while() à parte

fork()

}

Filho i=0

a.1) redireciona stdout para escrever no extremo de escrita do pipe[i][1]

b.1) exec(cmd[i])

Filho i=(1..N-1)

a.i) redireciona stdin para ler do extremo de leitura do pipe[i-1][0]

a.1) redireciona stdout para escrever no extremo de escrita do pipe[i][1]

b.i) exec(cmd[i]l)

Filho i=N

a.2) redireciona stdin para ler do extremo de leitura do pipe[i-1][0]

b.2) exec(cmd[i])

Estratégia de criação e close de pipes

Pai vai criando um filho i de cada vez, o seu pipe e fechando descritores que não necessita

- close pipe[i][1] e pipe[i-1][0] (atenção que no primeiro e ultimo filho isto muda)

Os filhos vão fechando os extremos que não usam pipe[i] [0]