Algoritmos e Estruturas de Dados III

Aula 6.2 - Hashing Dinâmico

Prof. Hayala Curto

2022

PUC Minas

Hash Dinâmico
Definições

PUC Minas

Hashing – Hash Dinâmico

- Quando o arquivo de dados cresce ou diminui com frequência (muitas inclusões e exclusões), o índice também precisará ser ajustado.

- Uma tabela hash estática, para crescer, **precisa reposicionar** todos os registros.

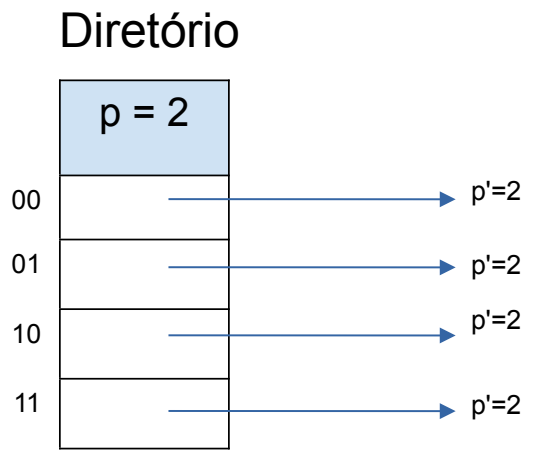
- Basicamente, refazer todo o índice!

Hashing – Hash Dinâmico

- Uma tabela hash extensível é uma tabela hash em que apenas alguns registros afetados (aqueles do bucket) precisam ser reposicionados.

- O diretório **sempre** tem tamanho equivalente a uma potência de dois

Hashing – Hash Dinâmico

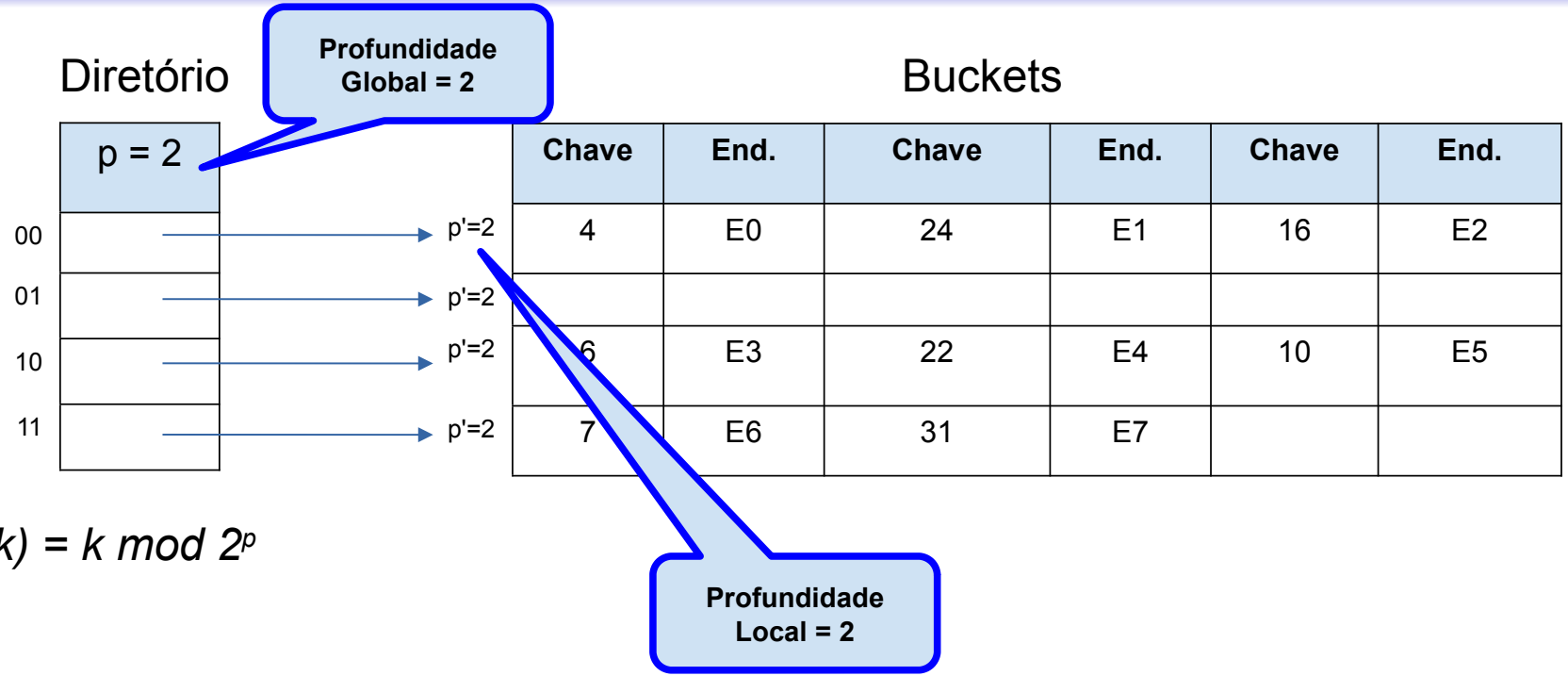


Buckets

Chave	End.	Chave	End.	Chave	End.
4	E0	24	E1	16	E2
6	E3	22	E4	10	E5
7	E6	31	E7		

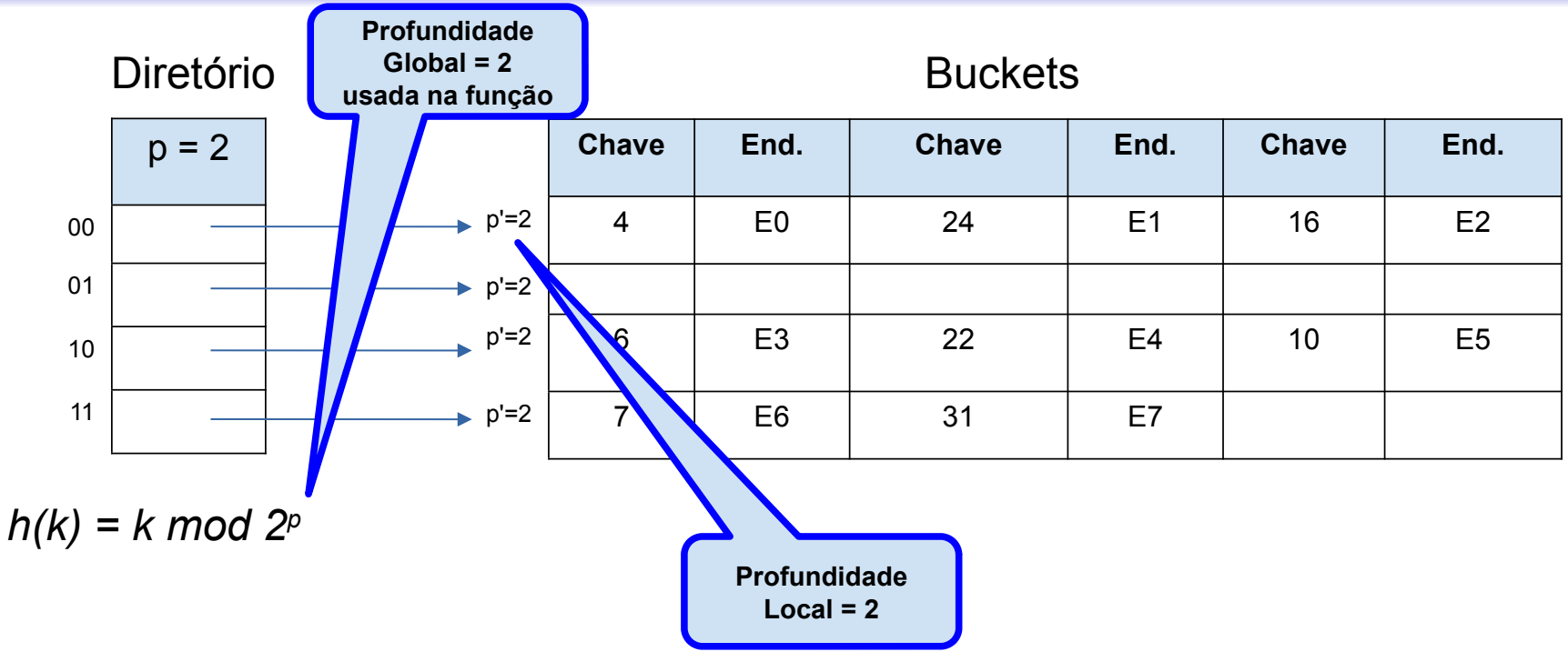
$h(k) = k \bmod 2^p$

Hashing – Hash Dinâmico



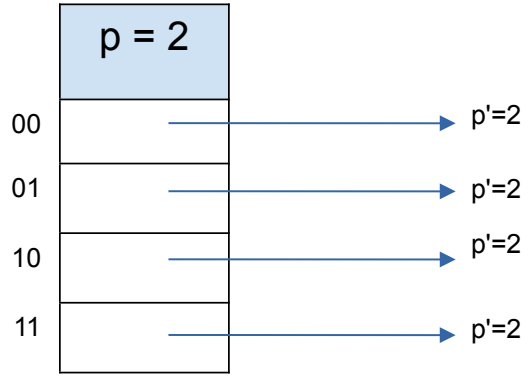
$h(k) = k \bmod 2^p$

Hashing – Hash Dinâmico



Hashing – Hash Dinâmico

Diretório



Buckets

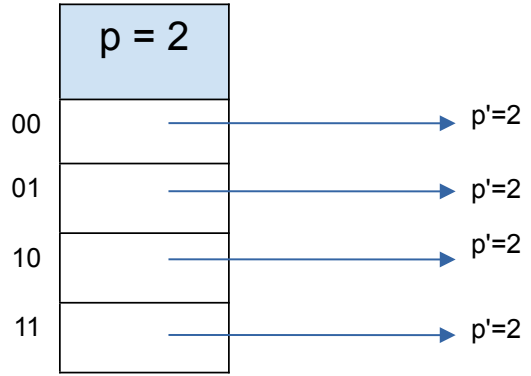
Chave	End.	Chave	End.	Chave	End.
4	E0	24	E1	16	E2
6	E3	22	E4	10	E5
7	E6	31	E7		

$h(k) = k \bmod 2^p$

Função Hash depende de p

Hashing – Hash Dinâmico

Diretório



Buckets

Chave	End.	Chave	End.	Chave	End.
4	E0	24	E1	16	E2
6	E3	22	E4	10	E5
7	E6	31	E7		

$h(k) = k \bmod 2^p$

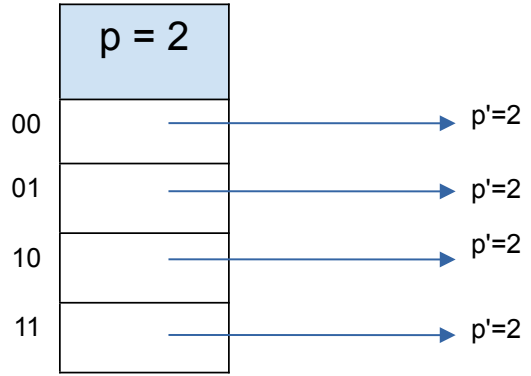
Buckets com total de elementos fixo

Inserção

PUC Minas

Inserindo chaves

Diretório



Buckets

Chave	End.	Chave	End.	Chave	End.
4	E0	24	E1	16	E2
6	E3	22	E4	10	E5
7	E6	31	E7		

$h(k) = k \bmod 2^p$

$h(9) = 9 \bmod 2^2$

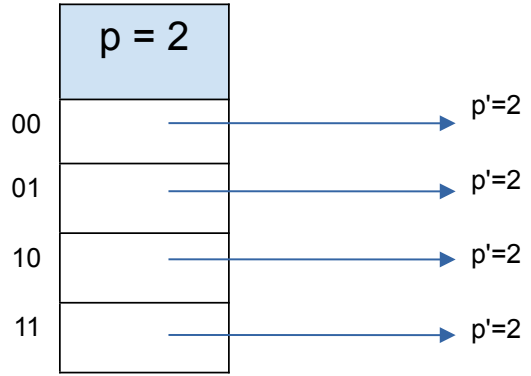
$h(9) = 9 \bmod 4$

$h(9) = 1$

Adicionar chave 9

Inserindo chaves

Diretório



Buckets

Chave	End.	Chave	End.	Chave	End.
4	E0	24	E1	16	E2
9	E8				
6	E3	22	E4	10	E5
7	E6	31	E7		

$h(k) = k \bmod 2^p$

$h(9) = 9 \bmod 2^2$

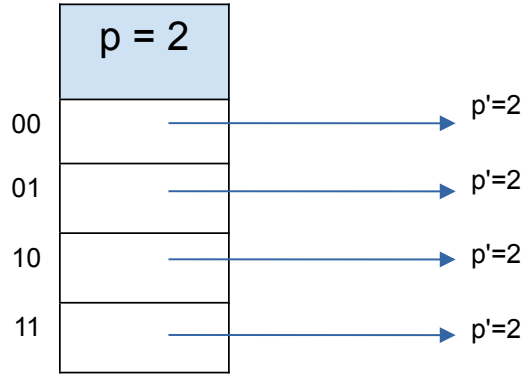
$h(9) = 9 \bmod 4$

$h(9) = 1$

Adicionar chave 9

Inserindo chaves

Diretório



Buckets

Chave	End.	Chave	End.	Chave	End.
4	E0	24	E1	16	E2
9	E8				
6	E3	22	E4	10	E5
7	E6	31	E7		

$h(k) = k \bmod 2^p$

$h(20) = 20 \bmod 2^2$

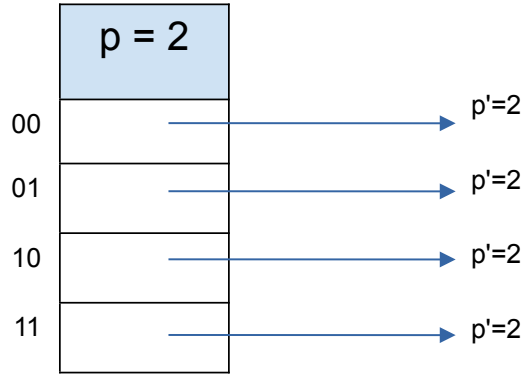
$h(20) = 20 \bmod 4$

$h(20) = 0$

Adicionar chave 20

Inserindo chaves

Diretório



Buckets

Chave	End.	Chave	End.	Chave	End.
4	E0	24	E1	16	E2
9	E8				
6	E3	22		10	E5
7	E6				

Não cabe no bucket 0!

Adicionar chave 20

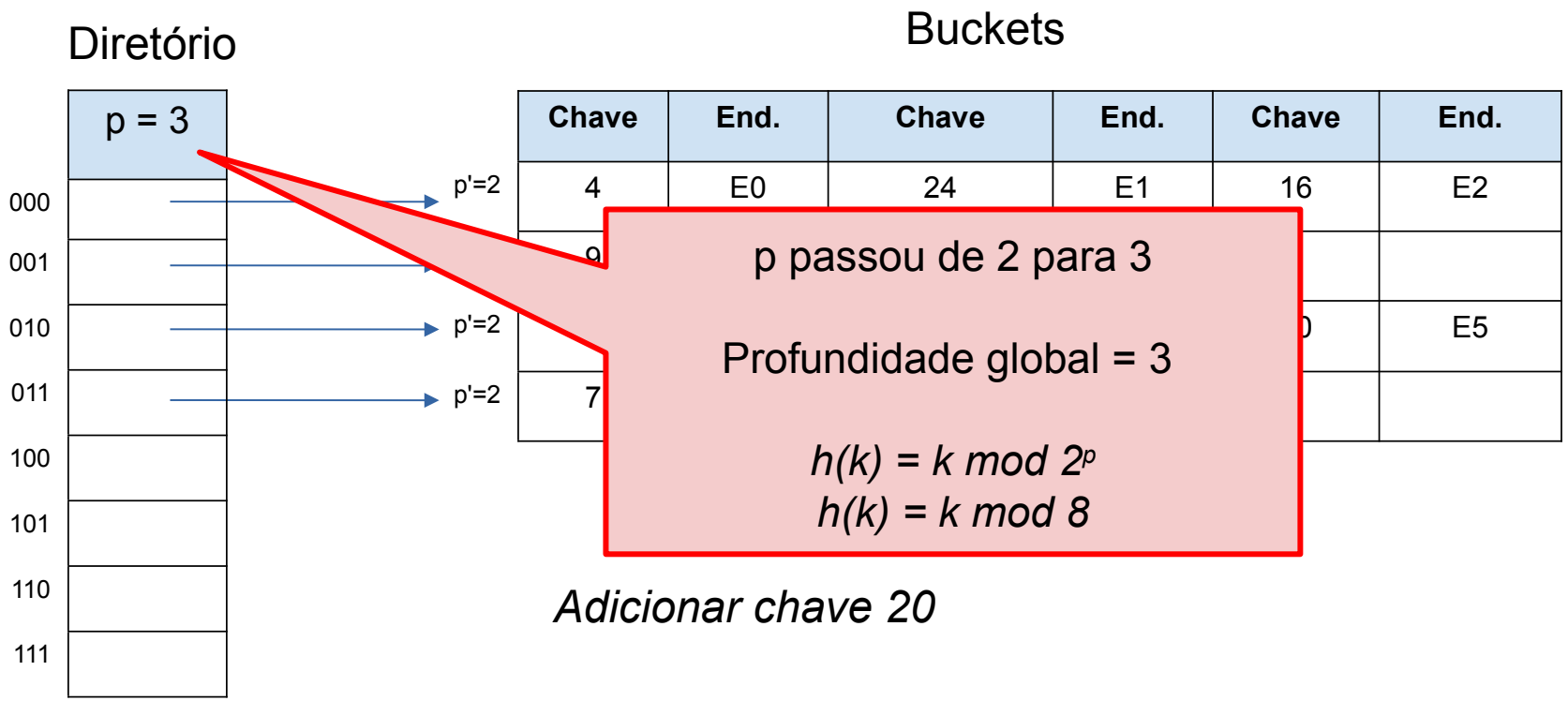
$h(k) = k \bmod 2^p$

$h(20) = 20 \bmod 2^2$

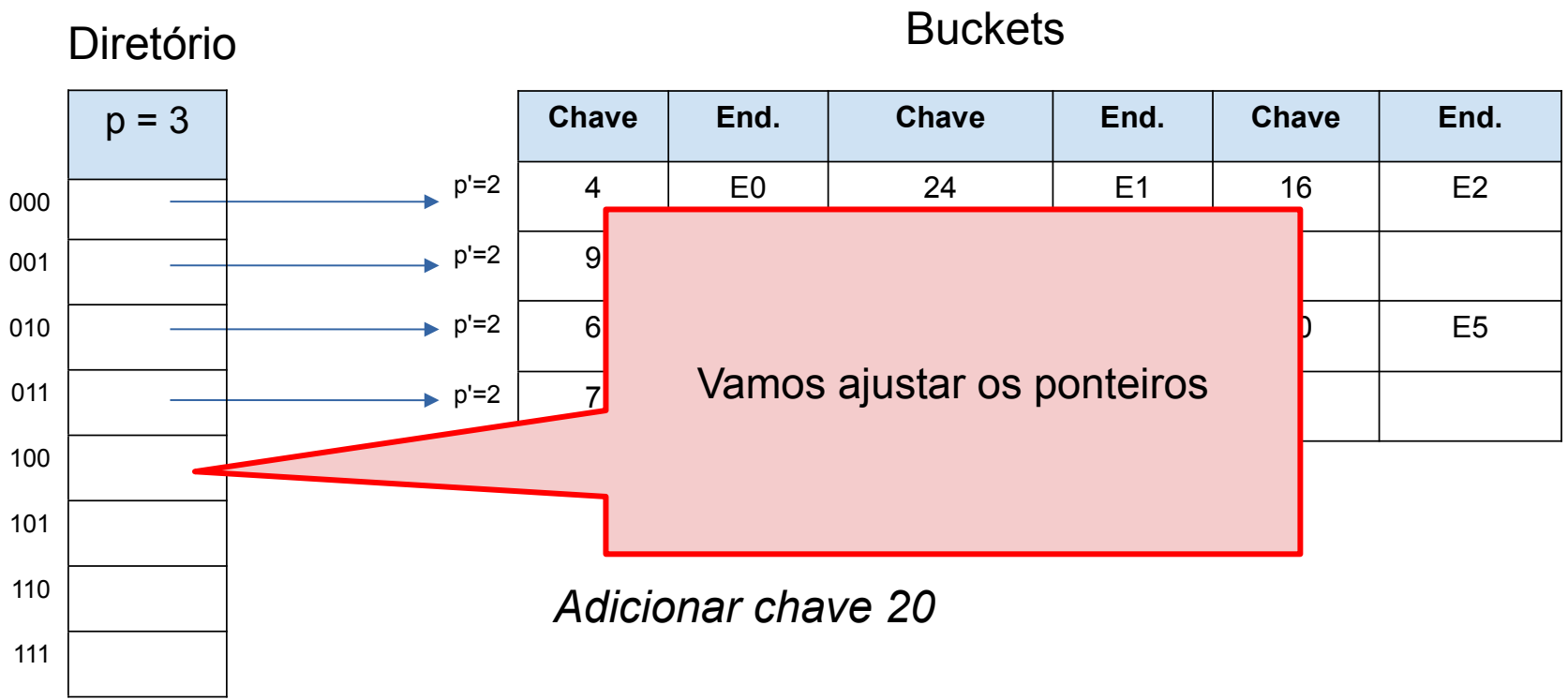
$h(20) = 20 \bmod 4$

$h(20) = 0$

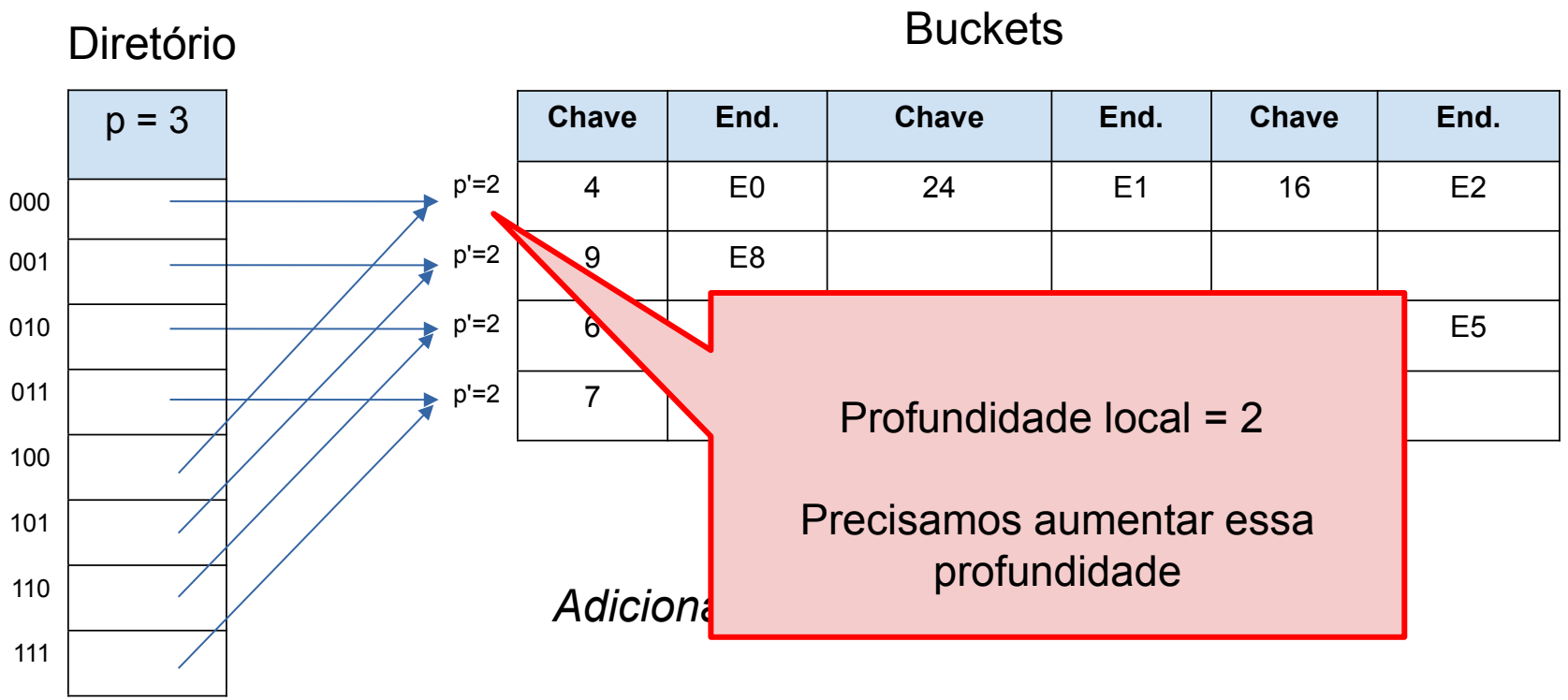
Inserindo chaves



Inserindo chaves

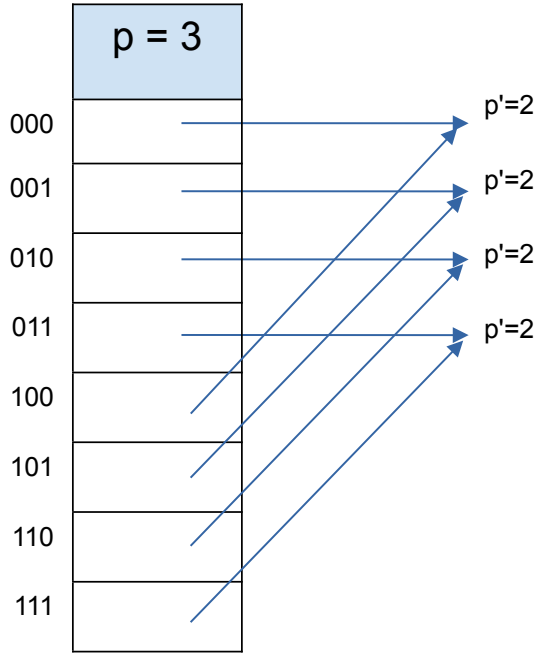


Inserindo chaves



Inserindo chaves

Diretório



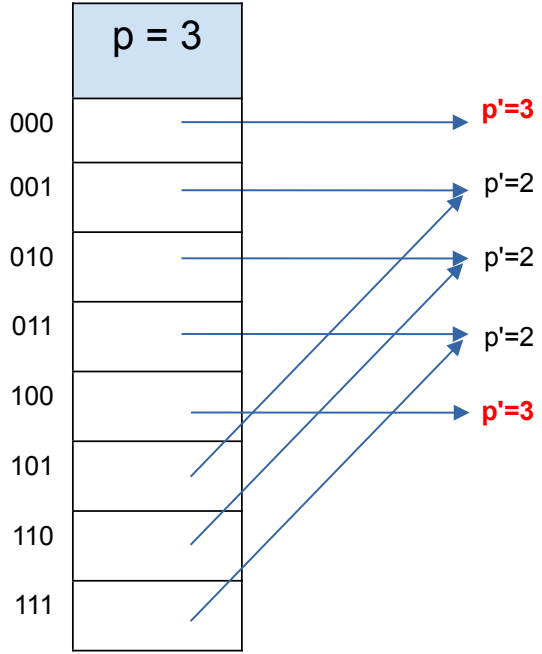
Buckets

Chave	End.	Chave	End.	Chave	End.
4	E0	24	E1	16	E2
9	E8				
6	E3	22	E4	10	E5
7	E6	31	E7		

Adicionar chave 20

Inserindo chaves

Diretório

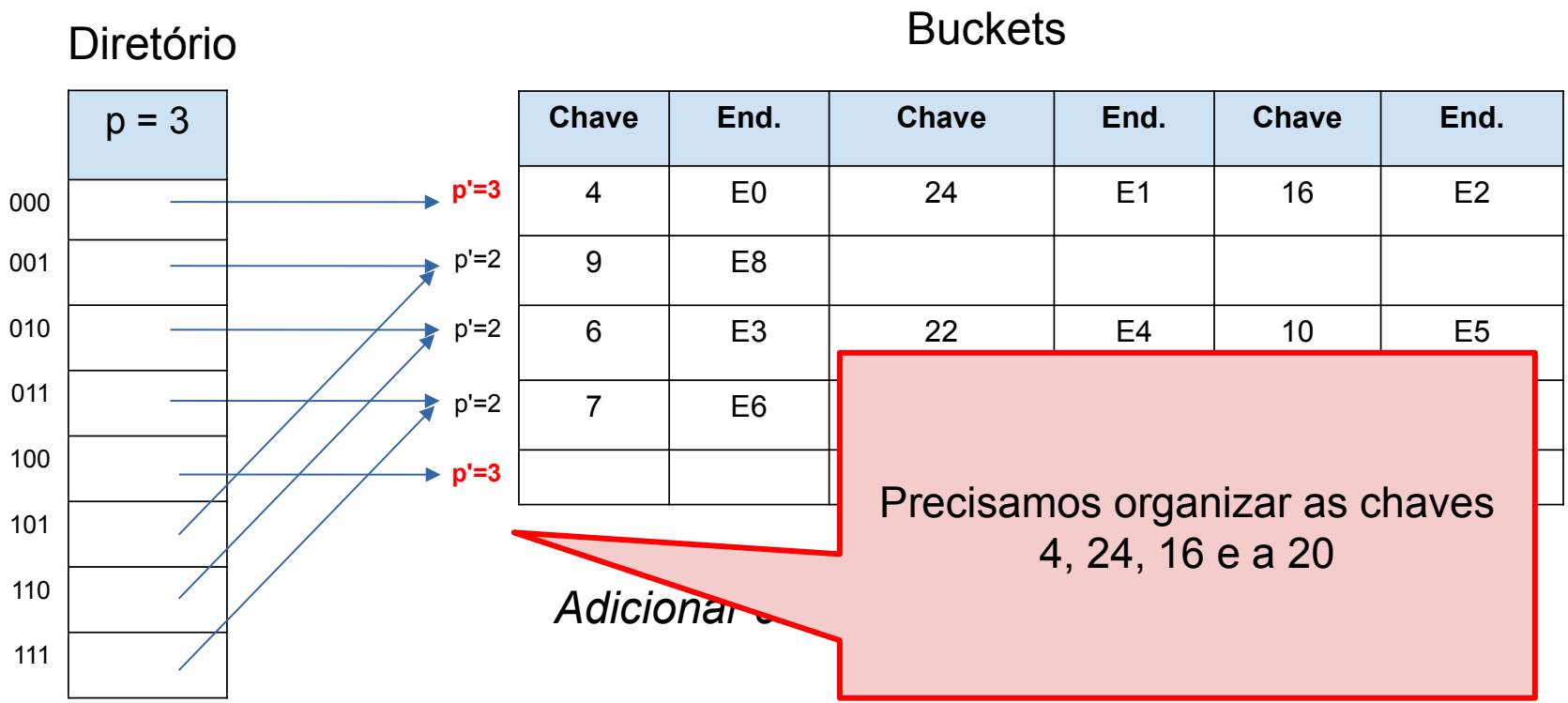


Buckets

Chave	End.	Chave	End.	Chave	End.
4	E0	24	E1	16	E2
9	E8				
6	E3	22	E4	10	E5
7	E6	31	E7		

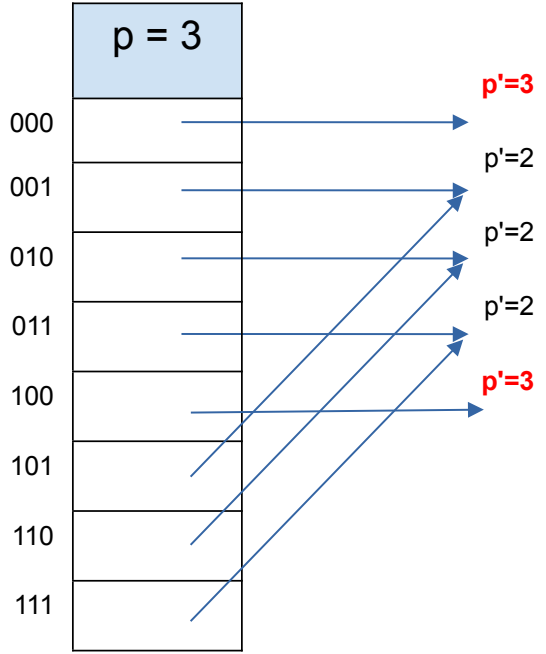
Adicionar chave 20

Inserindo chaves



Inserindo chaves

Diretório

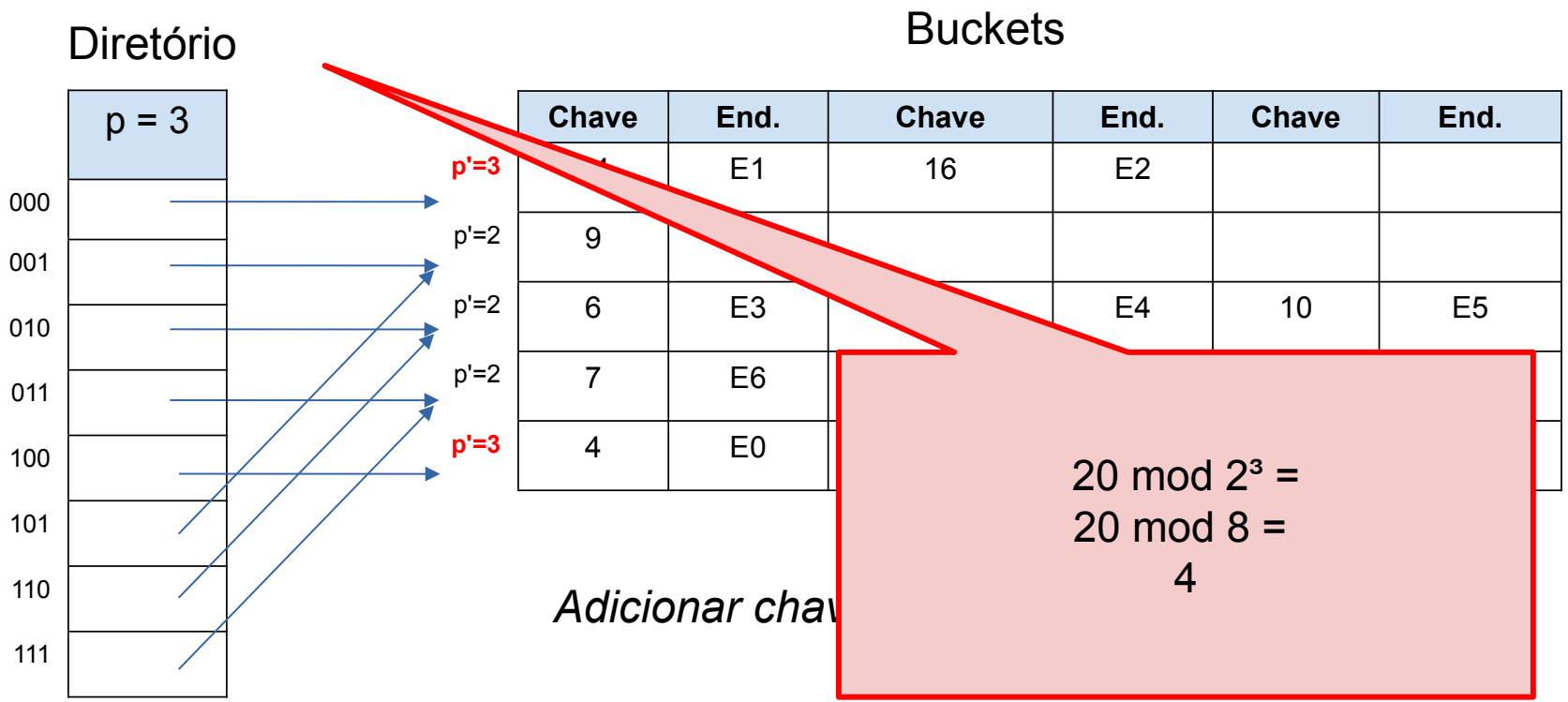


Buckets

Chave	End.	Chave	End.	Chave	End.
24	E1	16	E2		
9	E8				
6	E3	22	E4	10	E5
7	E6	31	E7		
4	E0				

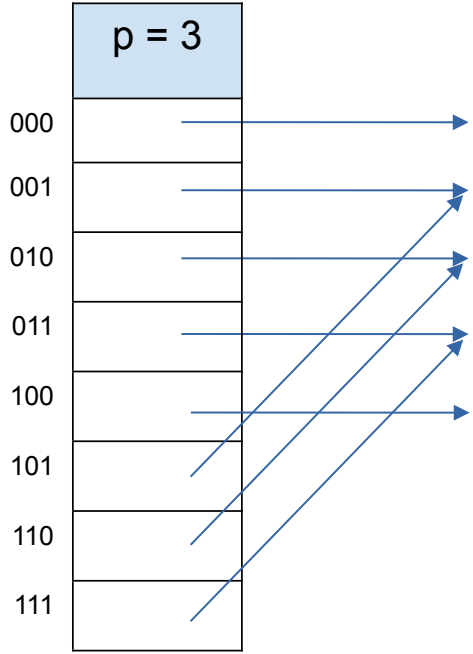
Adicionar chave 20

Inserindo chaves



Inserindo chaves

Diretório

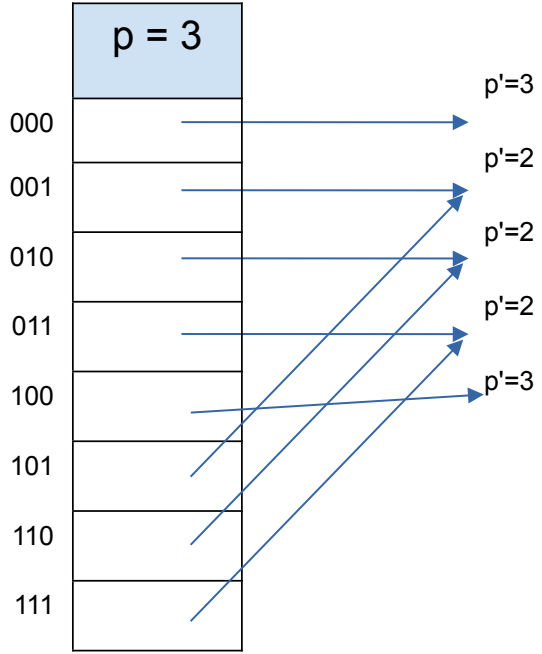


Buckets

	Chave	End.	Chave	End.	Chave	End.
p'=3	24	E1	16	E2		
p'=2	9	E8				
p'=2	6	E3	22	E4	10	E5
p'=2	7	E6	31	E7		
p'=3	4	E0	20	E9		

Inserindo chaves

Diretório



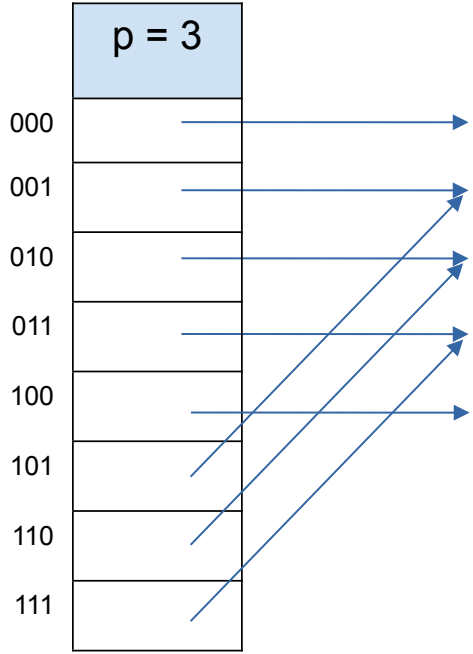
Buckets

Chave	End.	Chave	End.	Chave	End.
24	E1	16	E2		
9	E8				
6	E3	22	E4	10	E5
7	E6	31	E7		
4	E0	20	E9		

Adicionar chave 26

h(k) = 26 mod 8
h(26) = 2

Inserindo chaves



Buckets

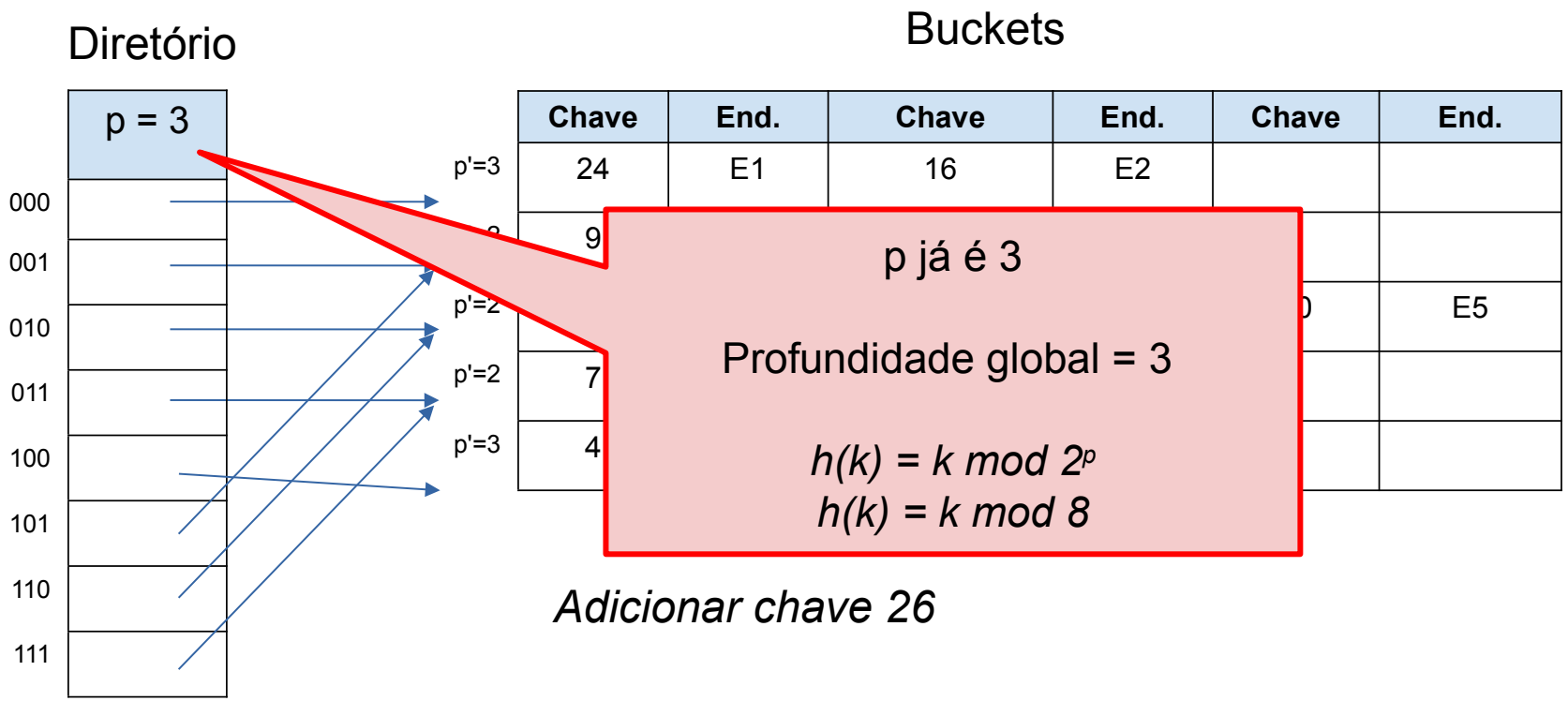
	Chave					End.
p'=3	24	E1				
p'=2	9	E8				
p'=2	6	E3	2	E4	10	E5
p'=2	7	E6	31	E7		
p'=3	4	E0	20	E9		

Não cabe no bucket 2!

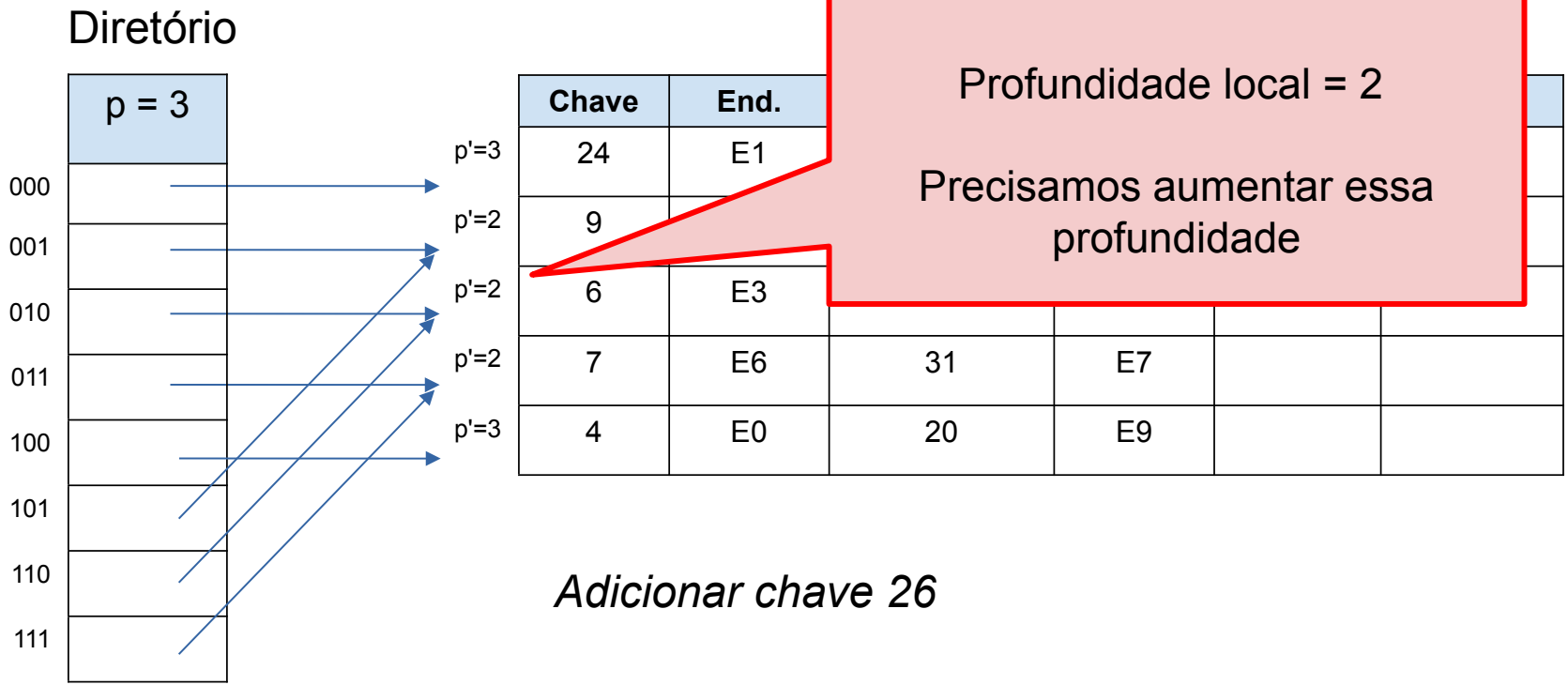
Adicionar chave 26

h(k) = 26 mod 8
h(20) = 2

Inserindo chaves



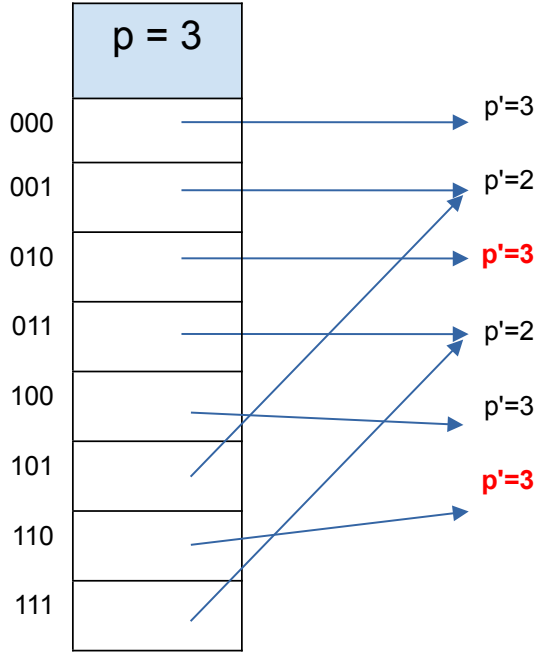
Inserindo chaves



Adicionar chave 26

Inserindo chaves

Diretório

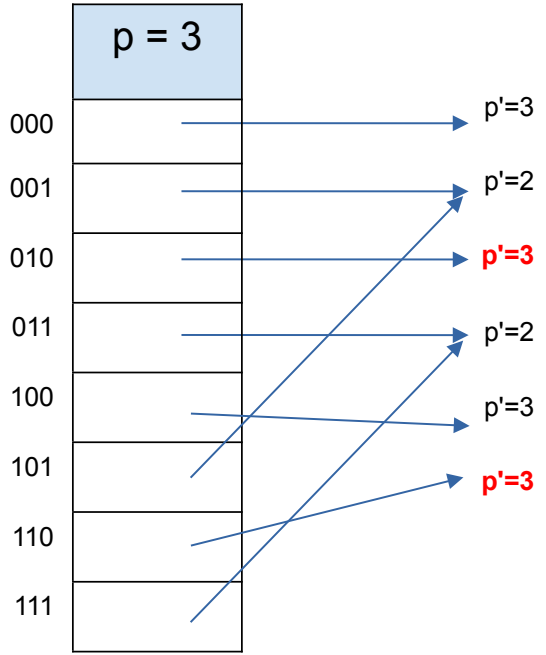


Buckets

Chave	End.	Chave	End.	Chave	End.
24	E1	16	E2		
9	E8				
6	E3	22	E4	10	E5
7	E6	31	E7		
4	E0	20	E9		

Adicionar chave 26

Inserindo chaves



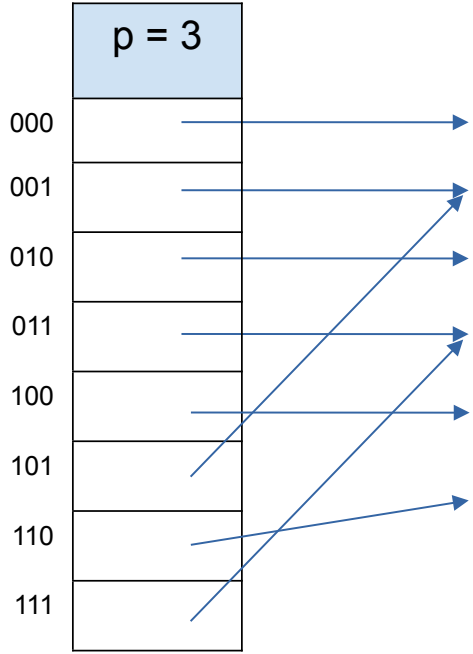
Chave	End.				
24	E1				
9					
6	E3	22	E4	10	E5
7	E6	31	E7		
4	E0	20	E9		

Precisamos organizar as chaves 6, 22, 10 e 26

Adicionar chave 26

Inserindo chaves

Diretório

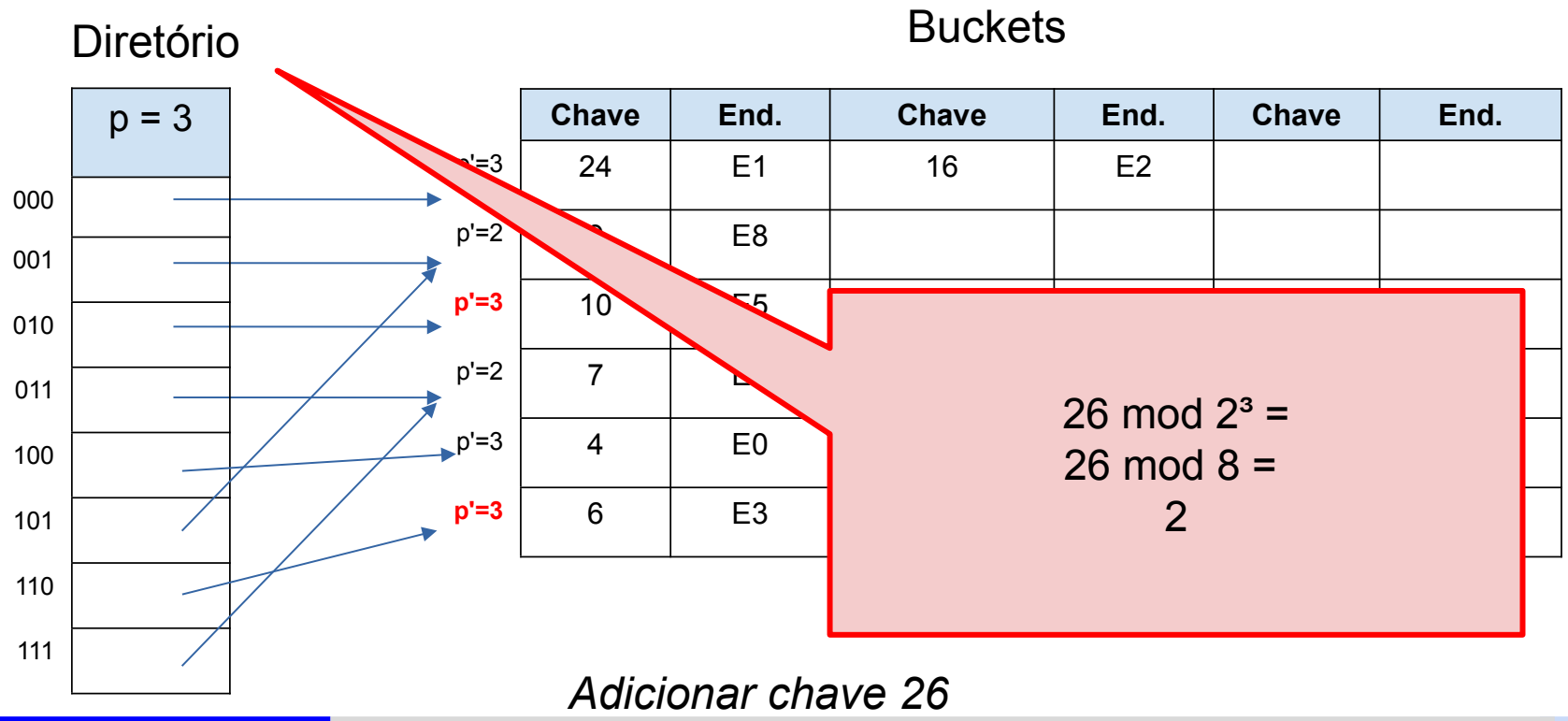


Buckets

	Chave	End.	Chave	End.	Chave	End.
p'=3	24	E1	16	E2		
p'=2	9	E8				
p'=3	10	E5				
p'=2	7	E6	31	E7		
p'=3	4	E0	20	E9		
p'=3	6	E3	22	E4		

Adicionar chave 26

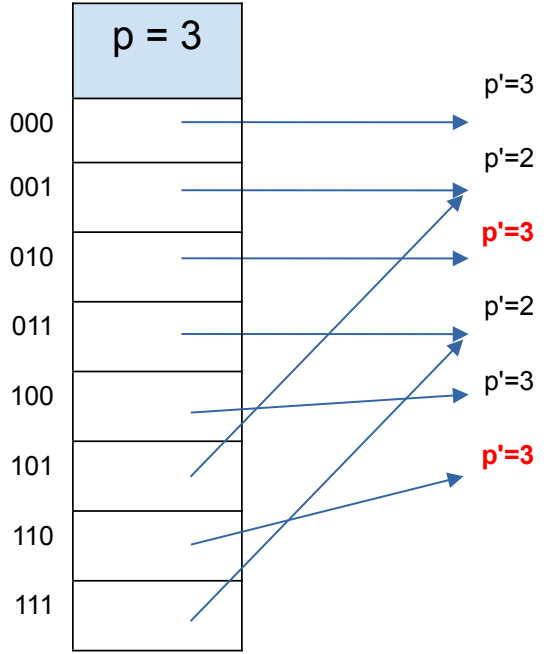
Inserindo chaves



Adicionar chave 26

Inserindo chaves

Diretório



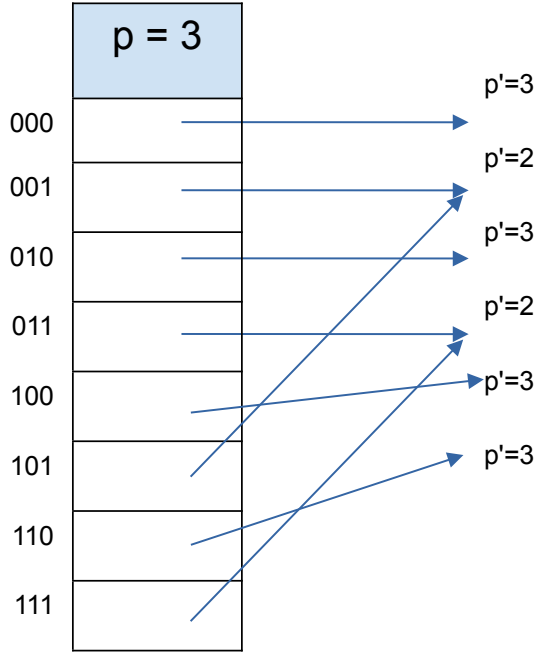
Buckets

	Chave	End.	Chave	End.	Chave	End.
p'=3	24	E1	16	E2		
p'=2	9	E8				
p'=3	10	E5	26	E10		
p'=2	7	E6	31	E7		
p'=3	4	E0	20	E9		
p'=3	6	E3	22	E4		

Adicionar chave 26

Inserindo chaves

Diretório



Buckets

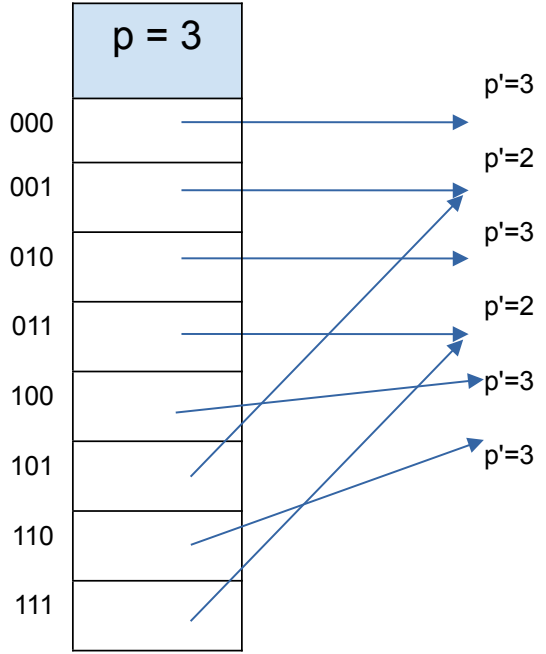
Chave	End.	Chave	End.	Chave	End.
24	E1	16	E2		
9	E8				
10	E5	26	E10		
7	E6	31	E7		
4	E0	20	E9		
6	E3	22	E4		

Deletando

PUC Minas

Deletando chaves

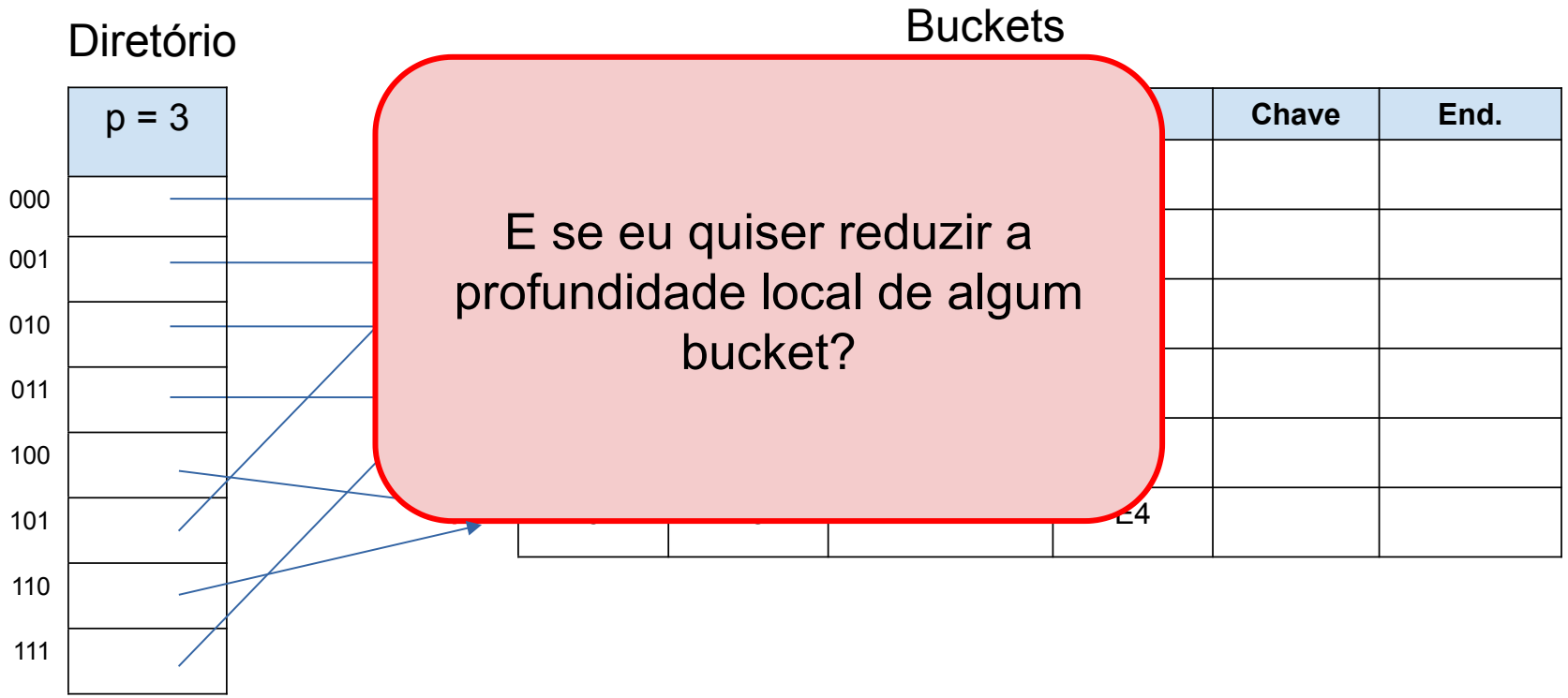
Diretório



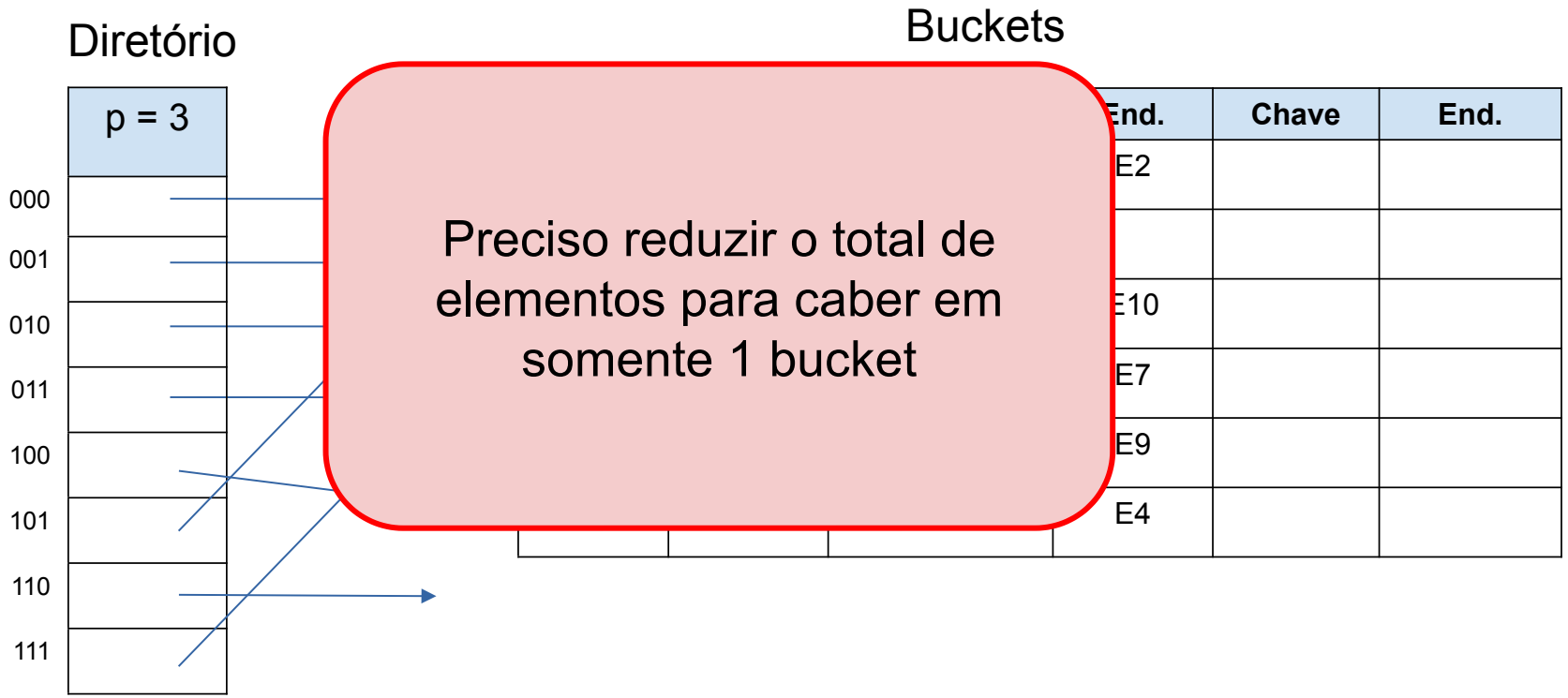
Buckets

Chave	End.	Chave	End.	Chave	End.
24	E1	16	E2		
9	E8				
10	E5	26	E10		
7	E6	31	E7		
4	E0	20	E9		
6	E3	22	E4		

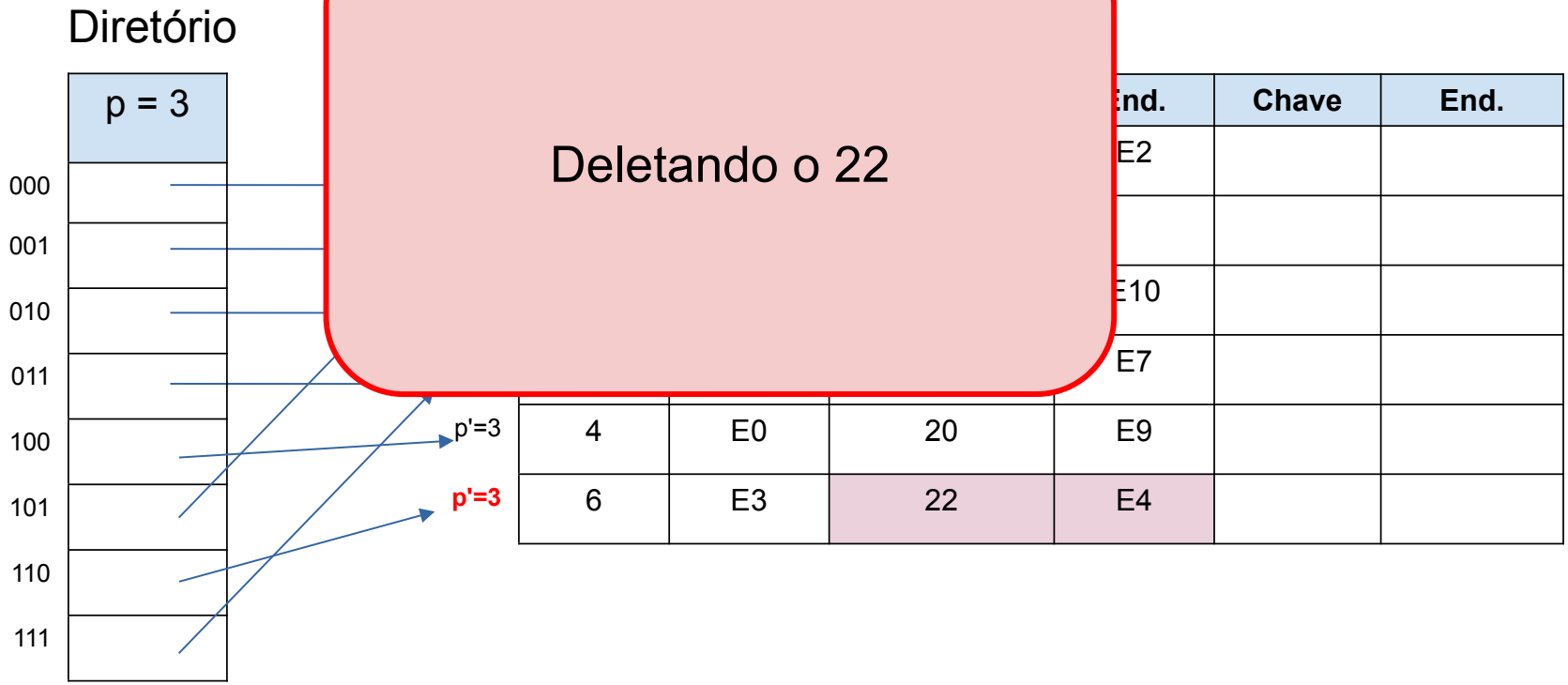
Deletando chaves



Deletando chaves

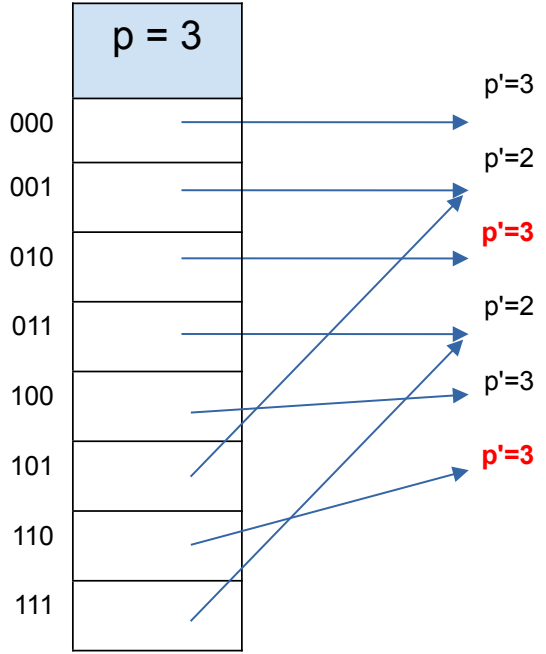


Deletando chaves



Deletando chaves

Diretório



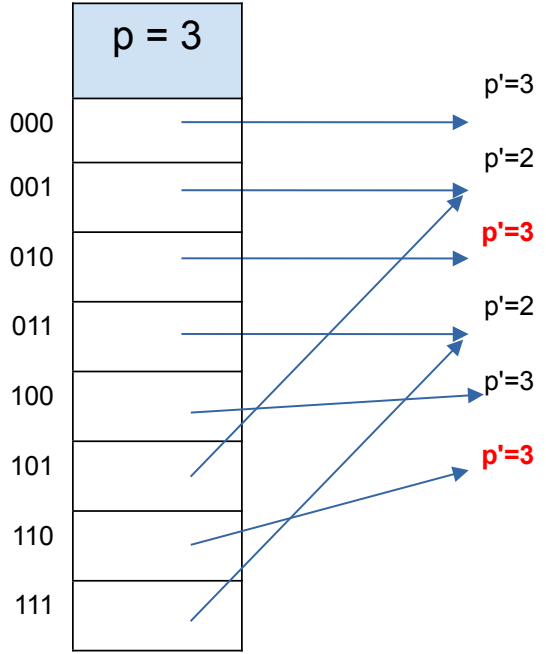
Buckets

Chave	End.	Chave	End.	Chave	End.
24	E1	16	E2		
9	E8				
10	E5	26	E10		
7	E6	31	E7		
4	E0	20	E9		
6	E3	22	E4		

Deletando a chave 22

Deletando chaves

Diretório



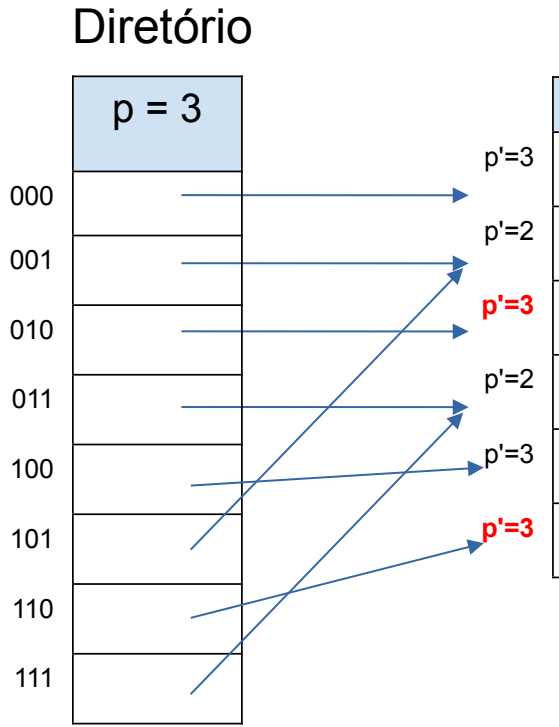
Buckets

Chave	End.	Chave	End.	Chave	End.
24	E1	16	E2		
9	E8				
10	E5	26	E10		
7	E6	31	E7		
4	E0	20	E9		
6	E3				

Deletando chaves

Juntando os buckets 2 e 5

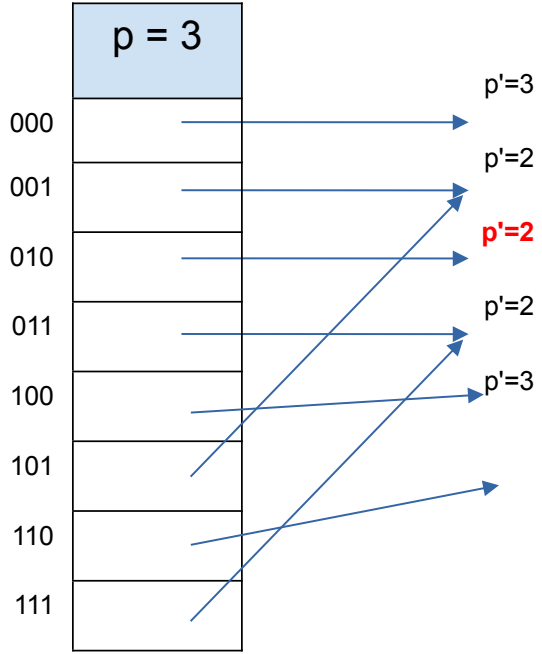
Tenho 3 elementos e o bucket cabe 3 elementos



Chave					
24					
9					
10	E5	26	E10		
7	E6	31	E7		
4	E0	20	E9		
6	E3				

Deletando chaves

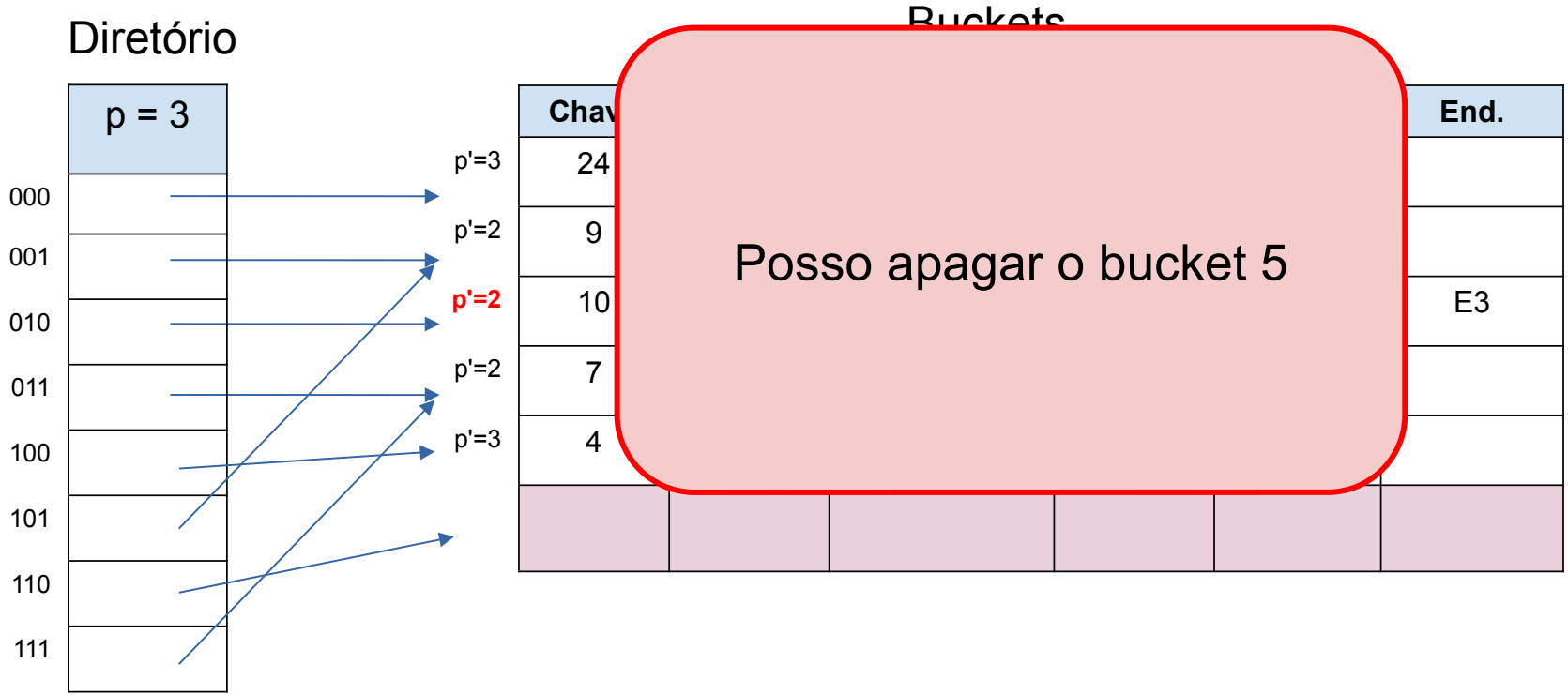
Diretório



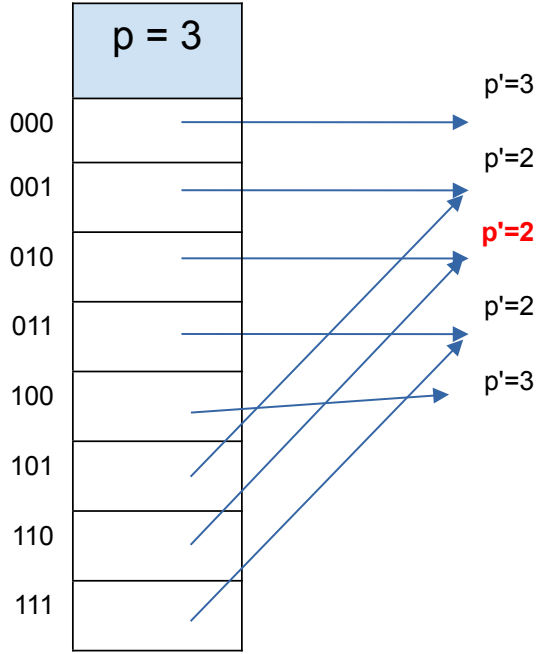
Buckets

Chave	End.	Chave	End.	Chave	End.
24	E1	16	E2		
9	E8				
10	E5	26	E10	6	E3
7	E6	31	E7		
4	E0	20	E9		

Deletando chaves



Deletando chaves

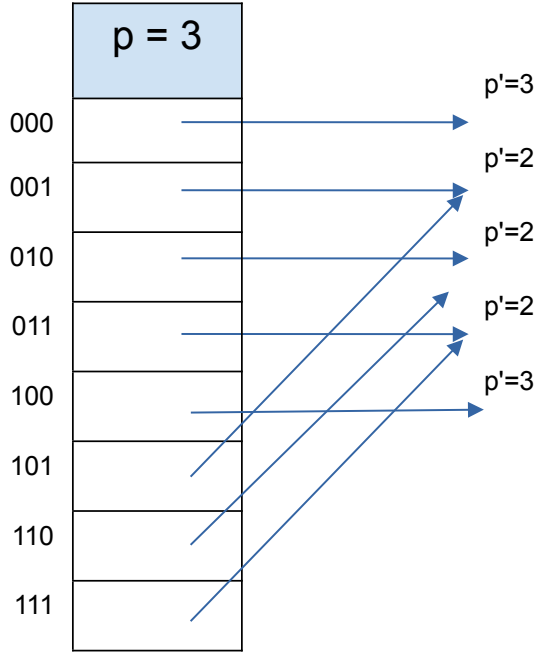


Buckets

	Chave	End.	Chave	End.	Chave	End.
p'=3	24	E1	16	E2		
p'=2	9	E8				
p'=2	10	E5	26	E10	6	E3
p'=2	7	E6	31	E7		
p'=3	4	E0	20	E9		

Deletando chaves

Diretório



Buckets

Chave	End.	Chave	End.	Chave	End.
24	E1	16	E2		
9	E8				
10	E5	26	E10	6	E3
7	E6	31	E7		
4	E0	20	E9		

Conclusões

PUC Minas

Hashing – Hash Dinâmico

- O diretório cresce, sem precisarmos reposicionar todos os registros (do índice)
- O índice (lista de buckets) cresce de acordo com a necessidade
- Como não há encadeamento dos buckets, não há perda de eficiência

Hashing – Hash Dinâmico

- Qual a melhor estrutura auxiliar para índices:

Árvore B, B+, B* ou Hash Dinâmico?

Hashing – Hash Dinâmico

- Qual a melhor estrutura auxiliar para índices?

Você só pode acessar elementos por sua chave primária em uma tabela de hash.
Isso é mais rápido do que com um algoritmo de árvore (**O(1) em vez de log(n**)), mas você não pode selecionar **intervalos** (tudo entre x e y).

Os algoritmos de árvore suportam isso em Log(n), enquanto os índices de hash podem resultar em uma varredura completa da tabela O(n).

A sobrecarga dos índices de hash geralmente é maior.
Os algoritmos de árvore geralmente são mais fáceis de manter, crescer com dados, escalar, etc.