On Disk B+ Trees

Databases Spring 2019

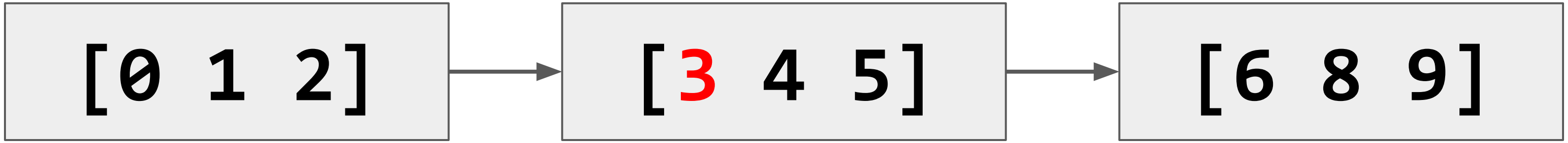
B+ Trees On Disk

- 1MB block pull from hard drive = 20ms
- B+ Tree
  - Root = 5000 indices
  - Internal Height 1 = 5000 * 5001 = 25,005,000 indices
  - Max 3 block pulls for 25,000,000 indices
  - Max 6 block pulls for lookup
- Binary Search
  - Log_2 25,000,000 = 24
  - Log_2 5,000 = 12
  - 12 block pulls

B+ Trees On Disk

Super fast range queries if blocks are sorted on disk

	Random Read 4KB Blocks (MB/s)	Streaming Reads
Hard Drive	2.1	151.6
Solid State Drive	129.4	252.6



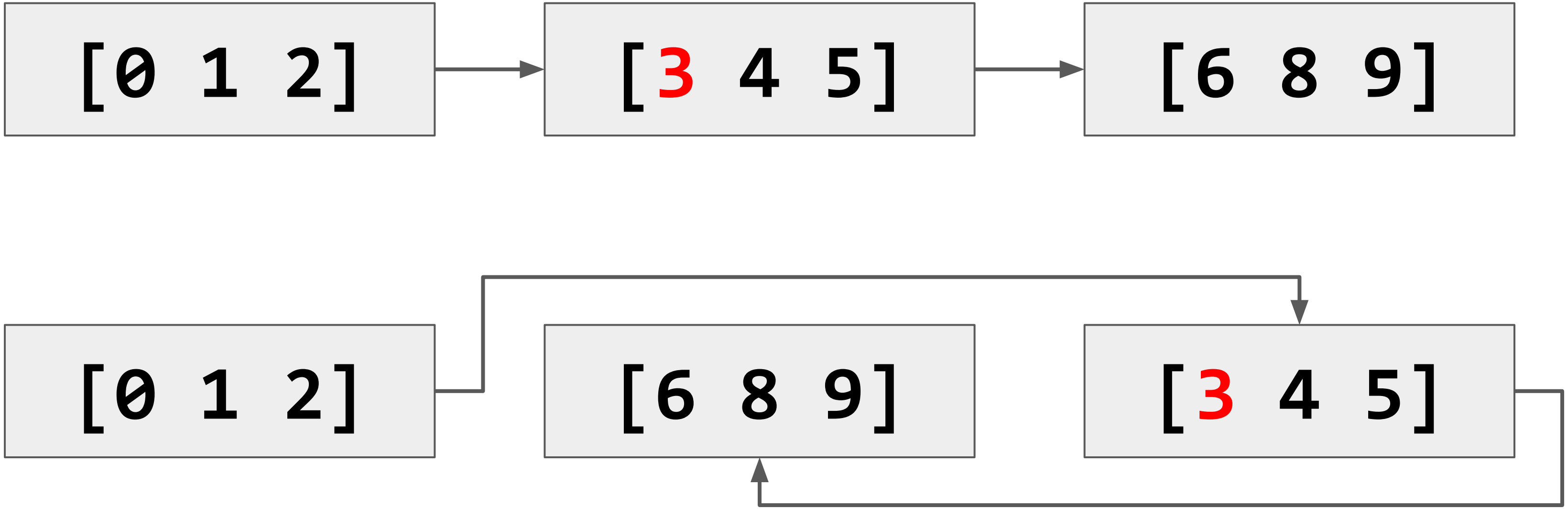
[0 1 2]

[3 4 5]

[6 8 9]

B+ Trees On Disk

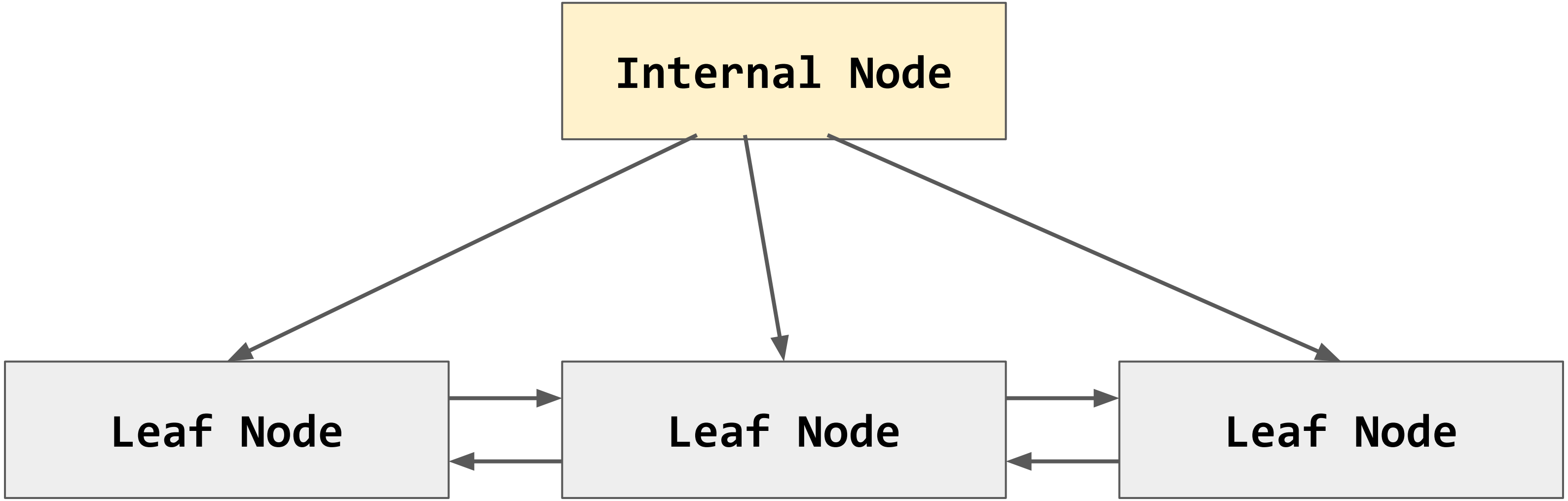
Super fast range queries if blocks are sorted on disk



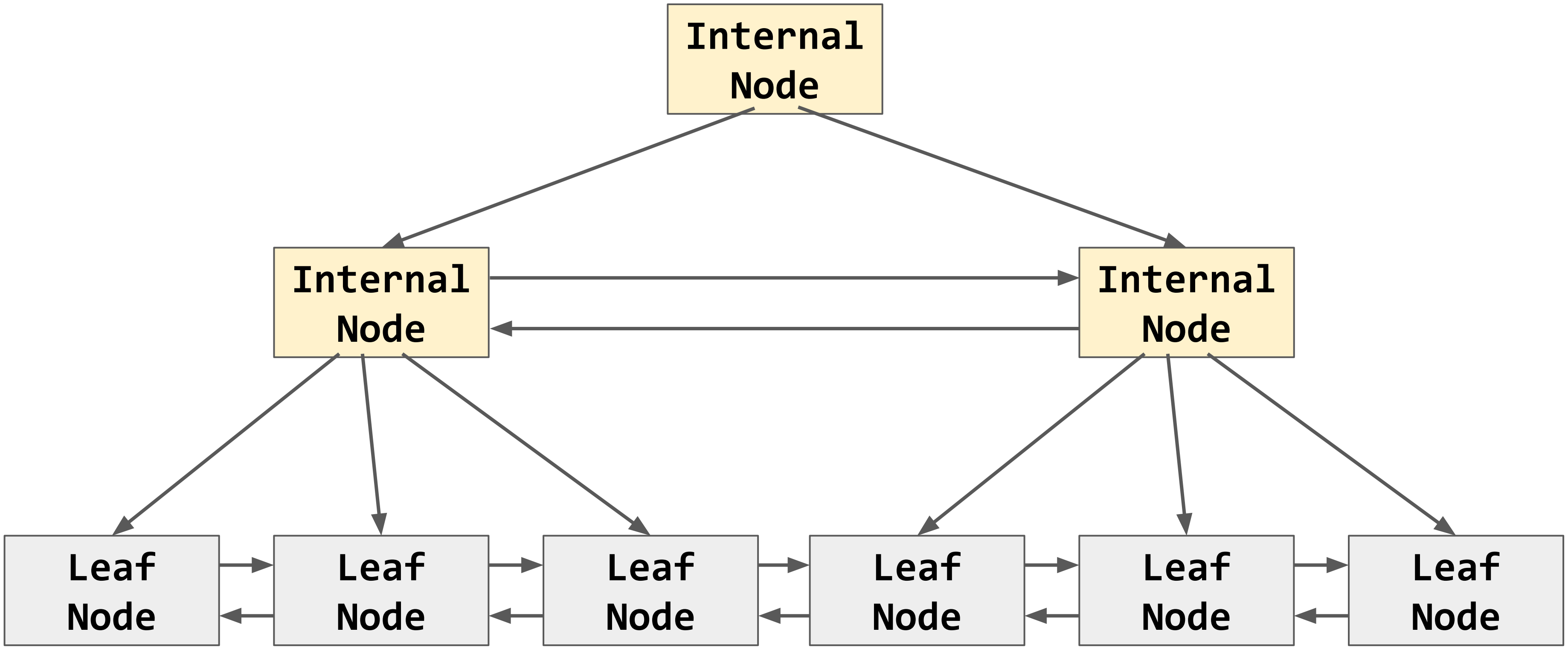
[0 1 2] [3 4 5] [6 8 9]

[0 1 2] [6 8 9] [3 4 5]

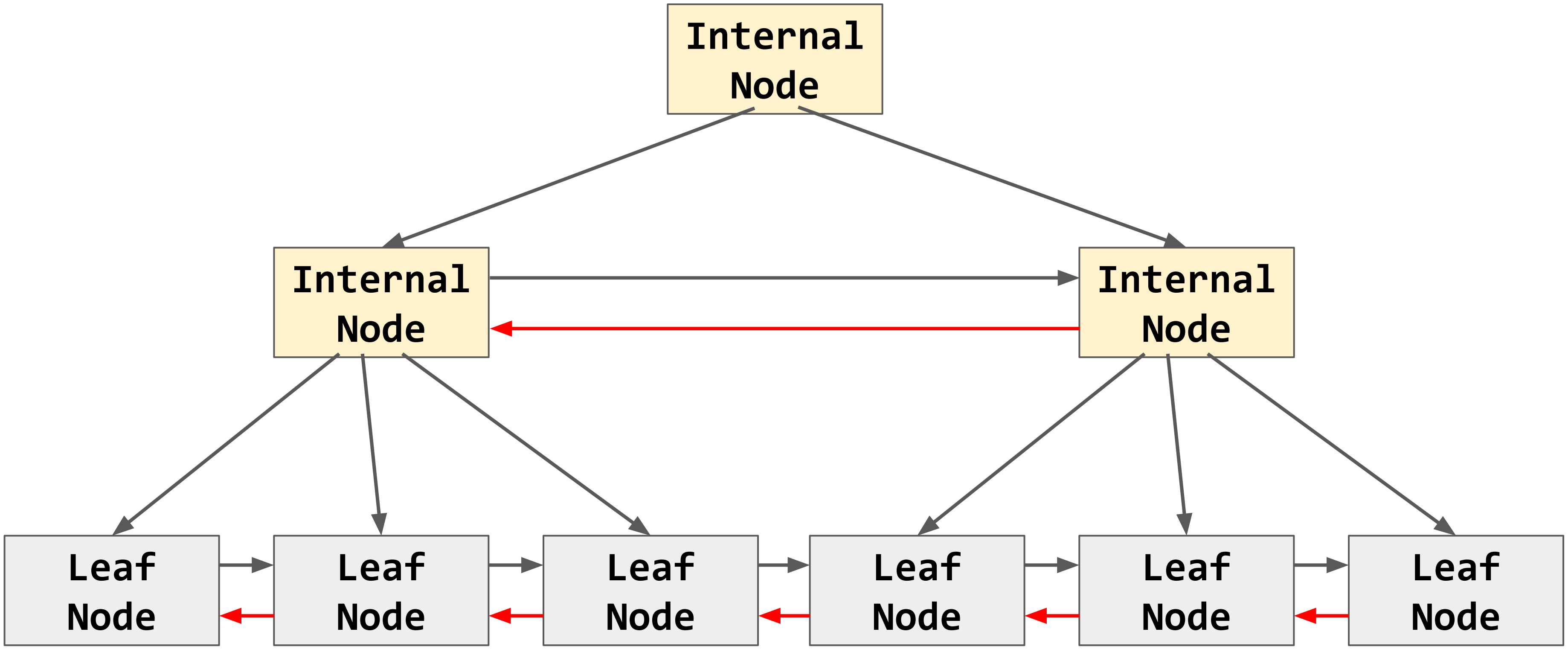
B+ Tree



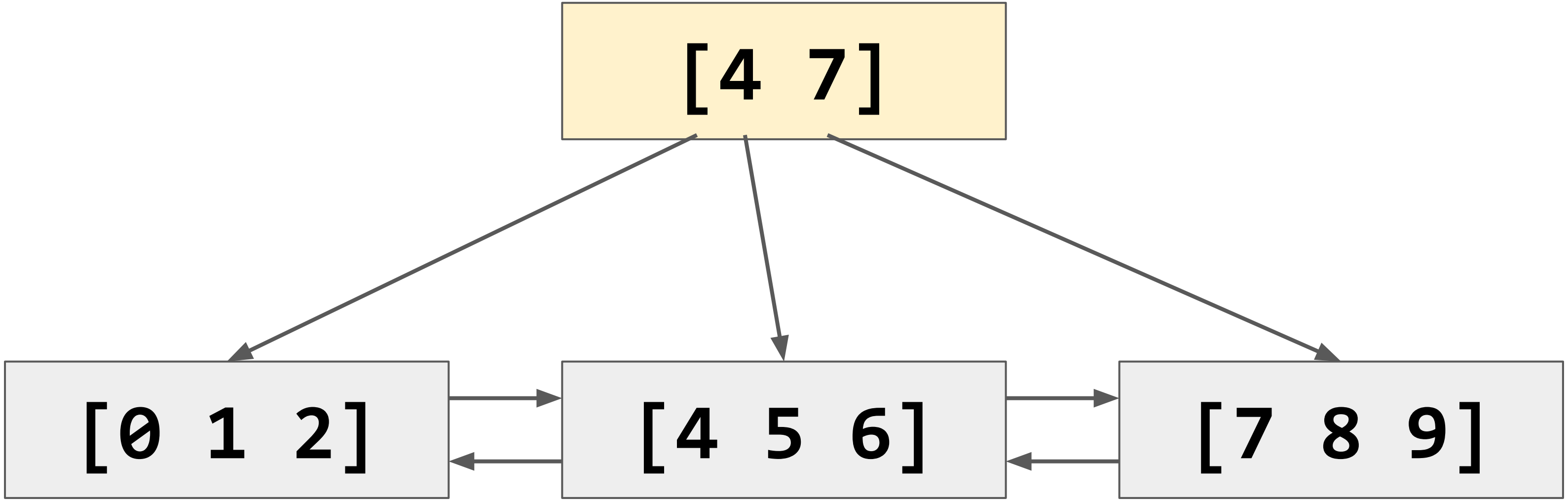
B+ Tree


Internal Node
Internal Node
Internal Node
Leaf Node
Leaf Node
Leaf Node
Leaf Node
Leaf Node
Leaf Node

B+ Tree


Internal Node
Internal Node
Internal Node
Leaf Node
Leaf Node
Leaf Node
Leaf Node
Leaf Node
Leaf Node

B+ Tree


[4 7]

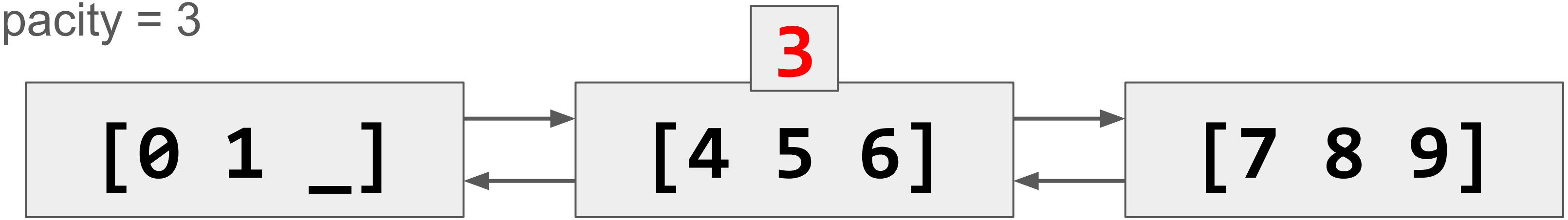
[0 1 2]

[4 5 6]

[7 8 9]

Give Left - insert 3

Capacity = 3



3

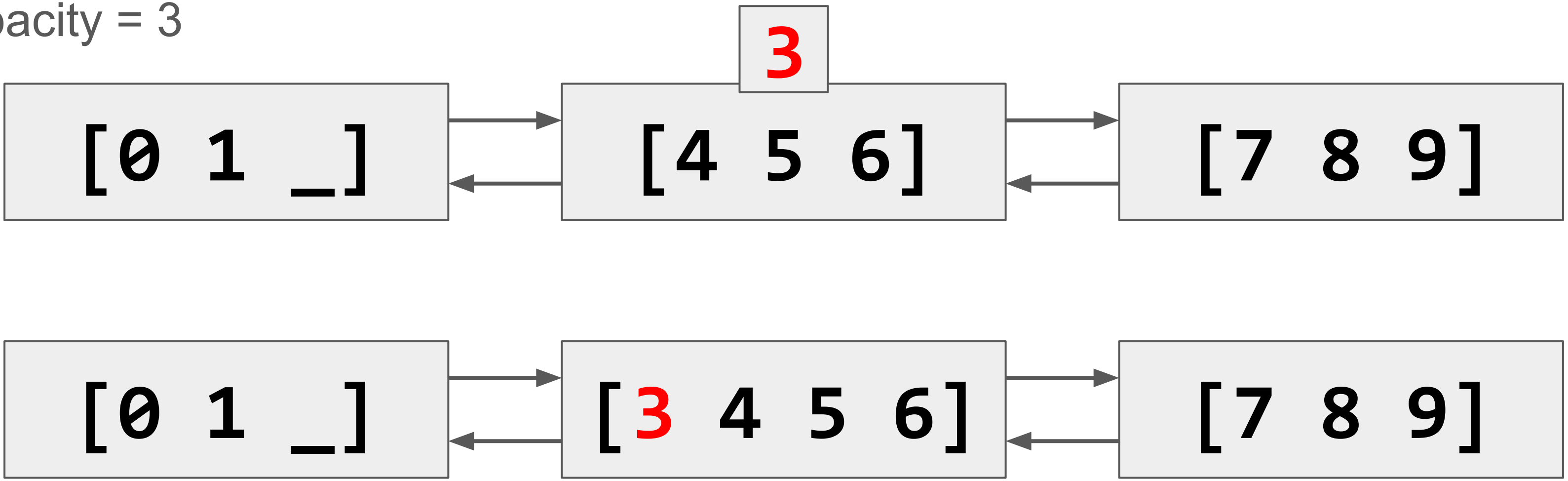
[0 1 _]

[4 5 6]

[7 8 9]

Give Left - insert 3

Capacity = 3



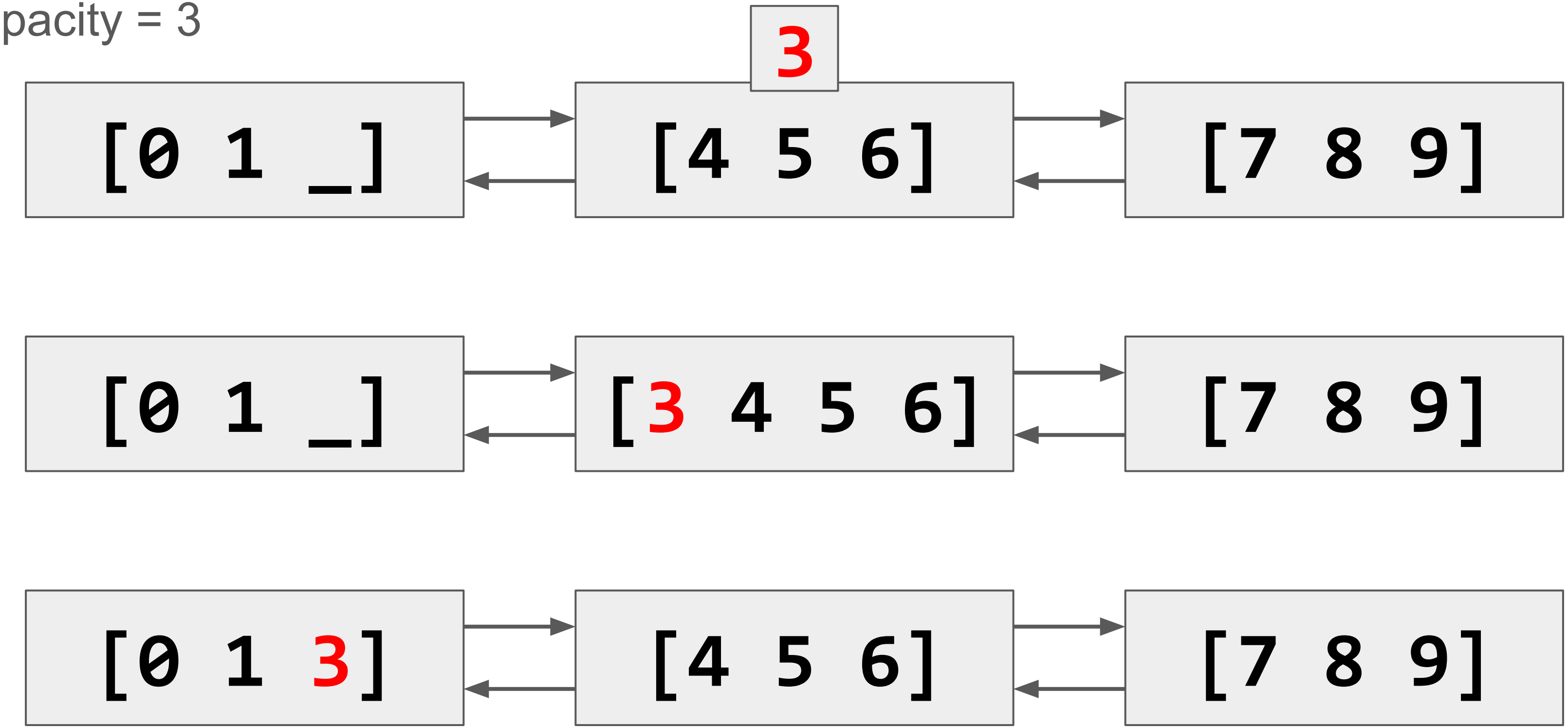
3

[0 1 _] [4 5 6] [7 8 9]

[0 1 _] [3 4 5 6] [7 8 9]

Give Left - insert 3

Capacity = 3



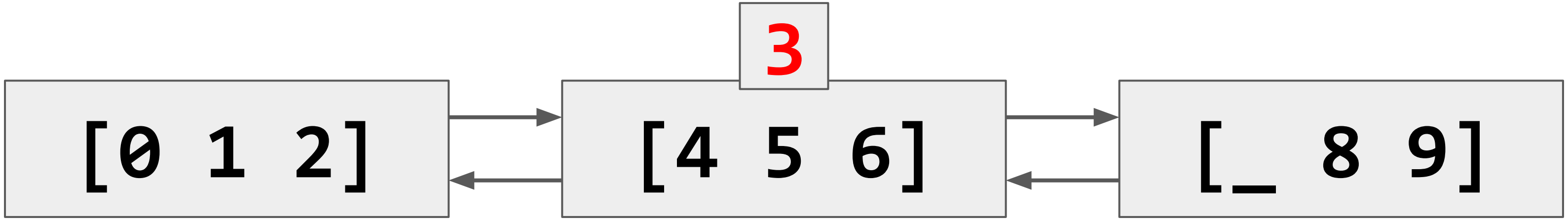
3

[0 1 _] [4 5 6] [7 8 9]

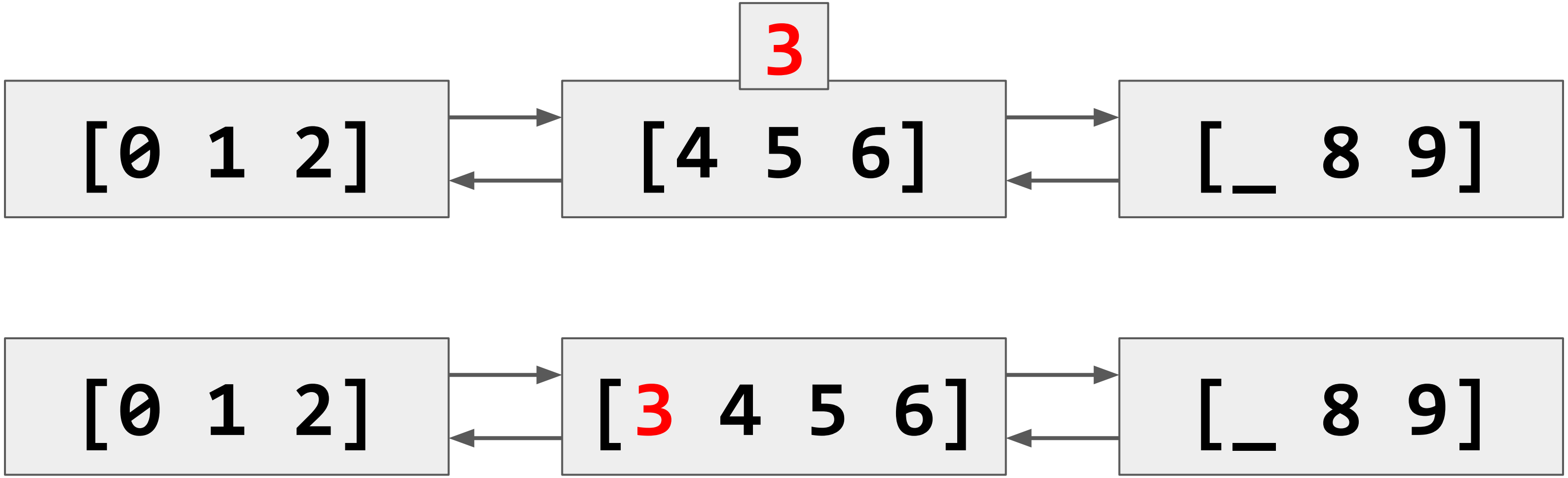
[0 1 _] [3 4 5 6] [7 8 9]

[0 1 3] [4 5 6] [7 8 9]

Give Right - insert 3



Give Right - insert 3

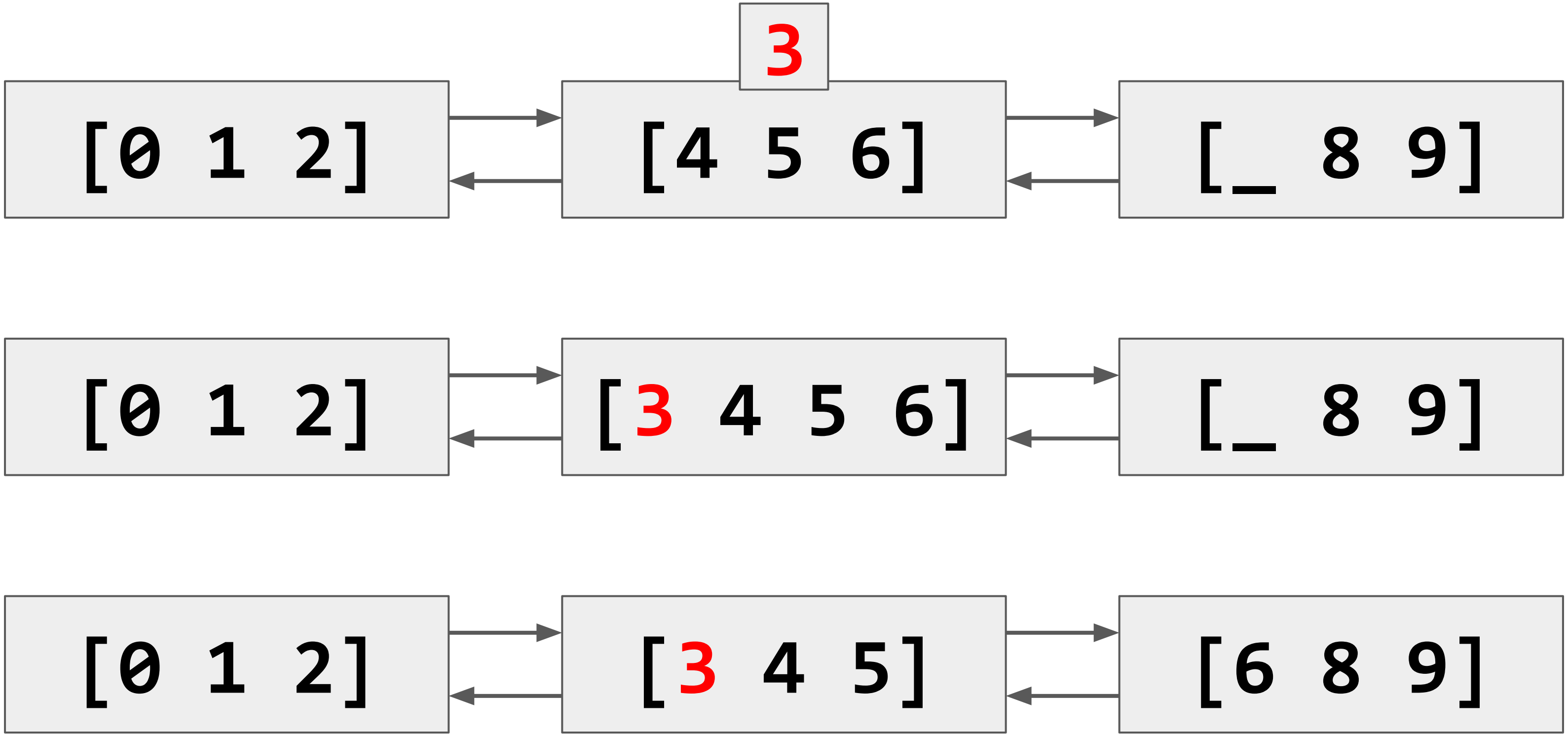


3

[0 1 2] [4 5 6] [_ 8 9]

[0 1 2] [3 4 5 6] [_ 8 9]

Give Right - insert 3



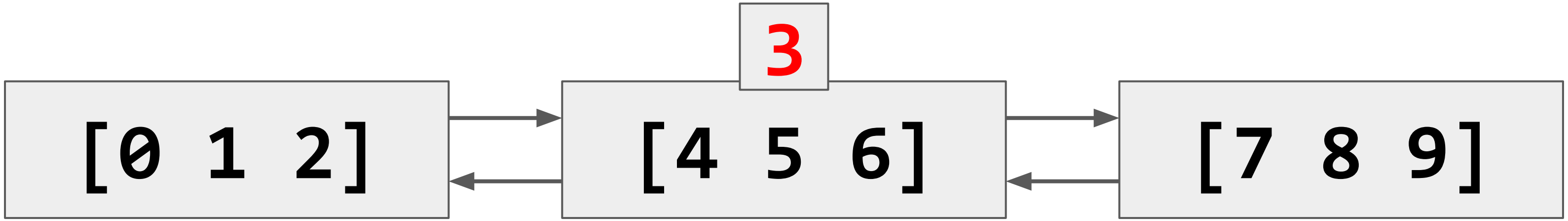
3

[0 1 2] [4 5 6] [_ 8 9]

[0 1 2] [3 4 5 6] [_ 8 9]

[0 1 2] [3 4 5] [6 8 9]

Split - insert 3



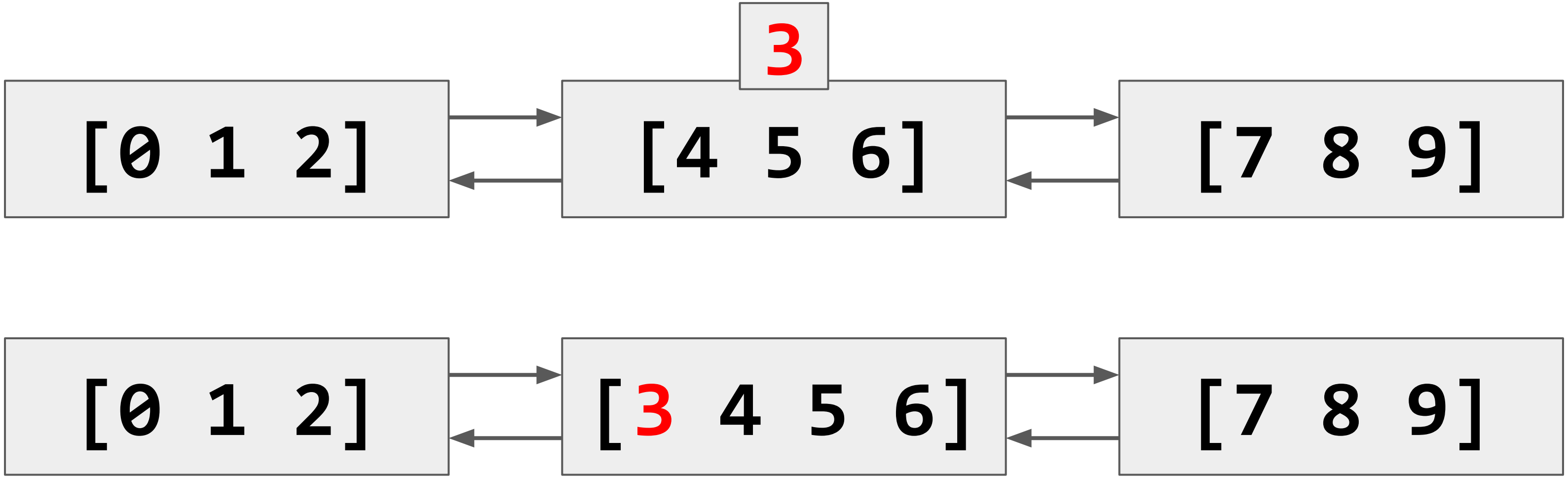
3

[0 1 2]

[4 5 6]

[7 8 9]

Split - insert 3

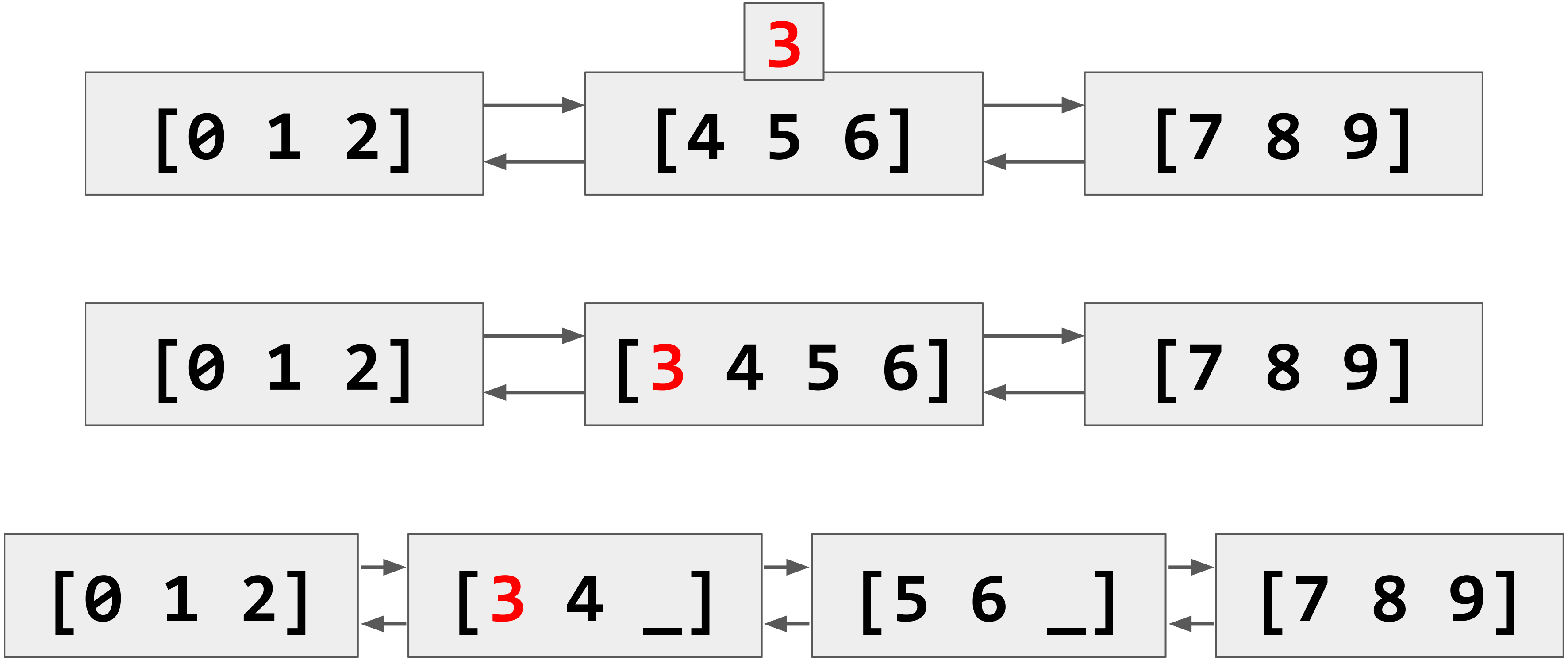


3

[0 1 2] [4 5 6] [7 8 9]

[0 1 2] [3 4 5 6] [7 8 9]

Split - insert 3



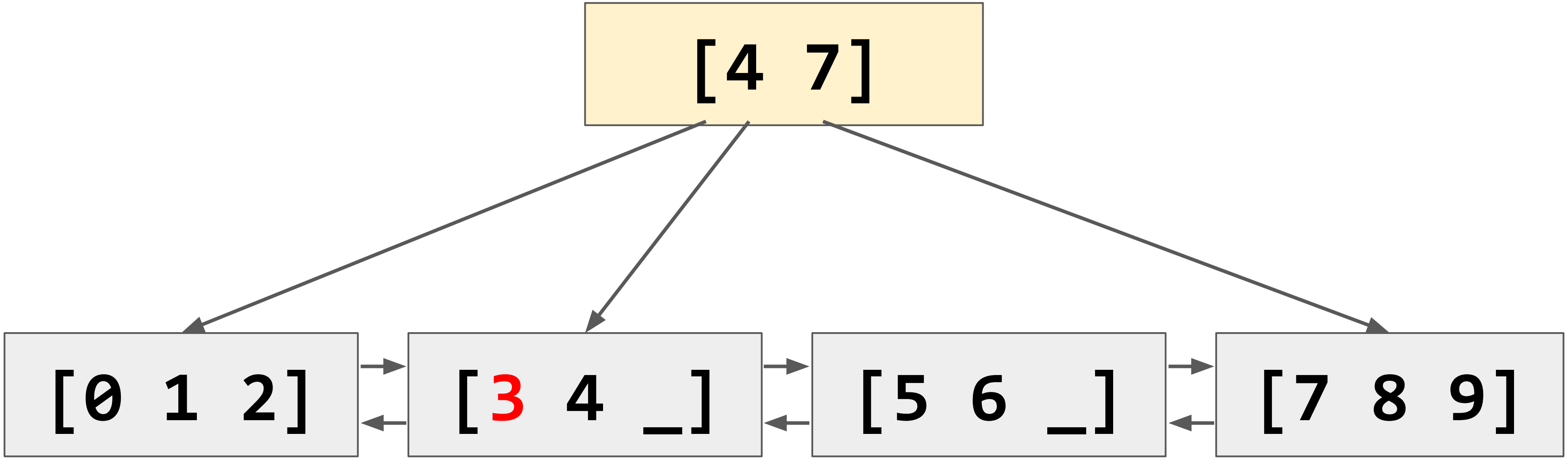
3

[0 1 2] [4 5 6] [7 8 9]

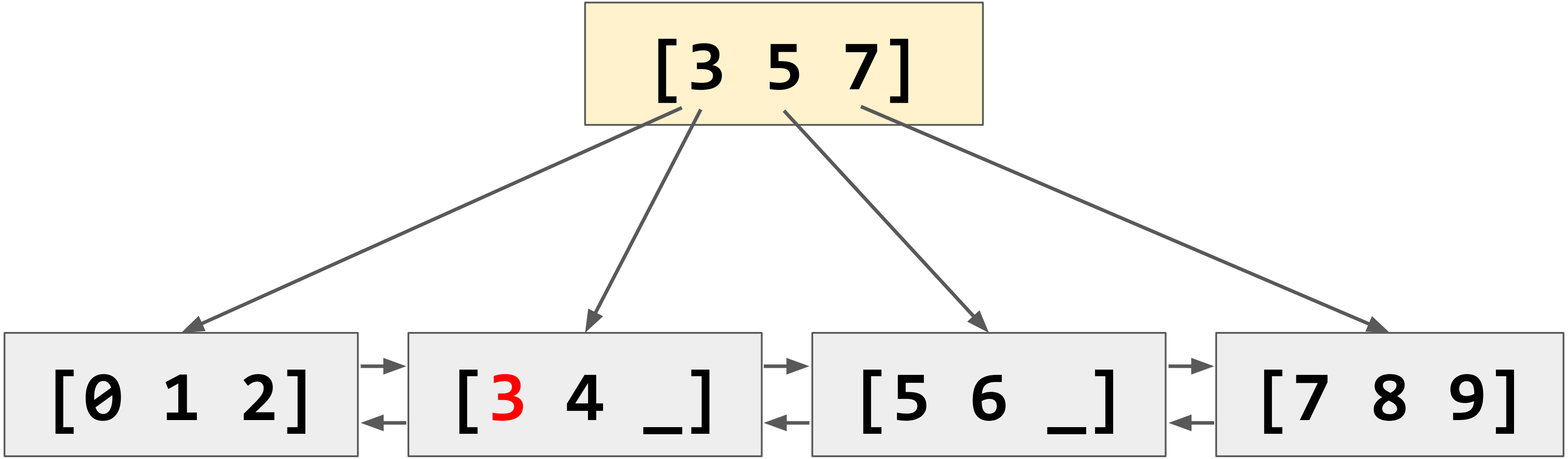
[0 1 2] [3 4 5 6] [7 8 9]

[0 1 2] [3 4 _] [5 6 _] [7 8 9]

Split - insert 3

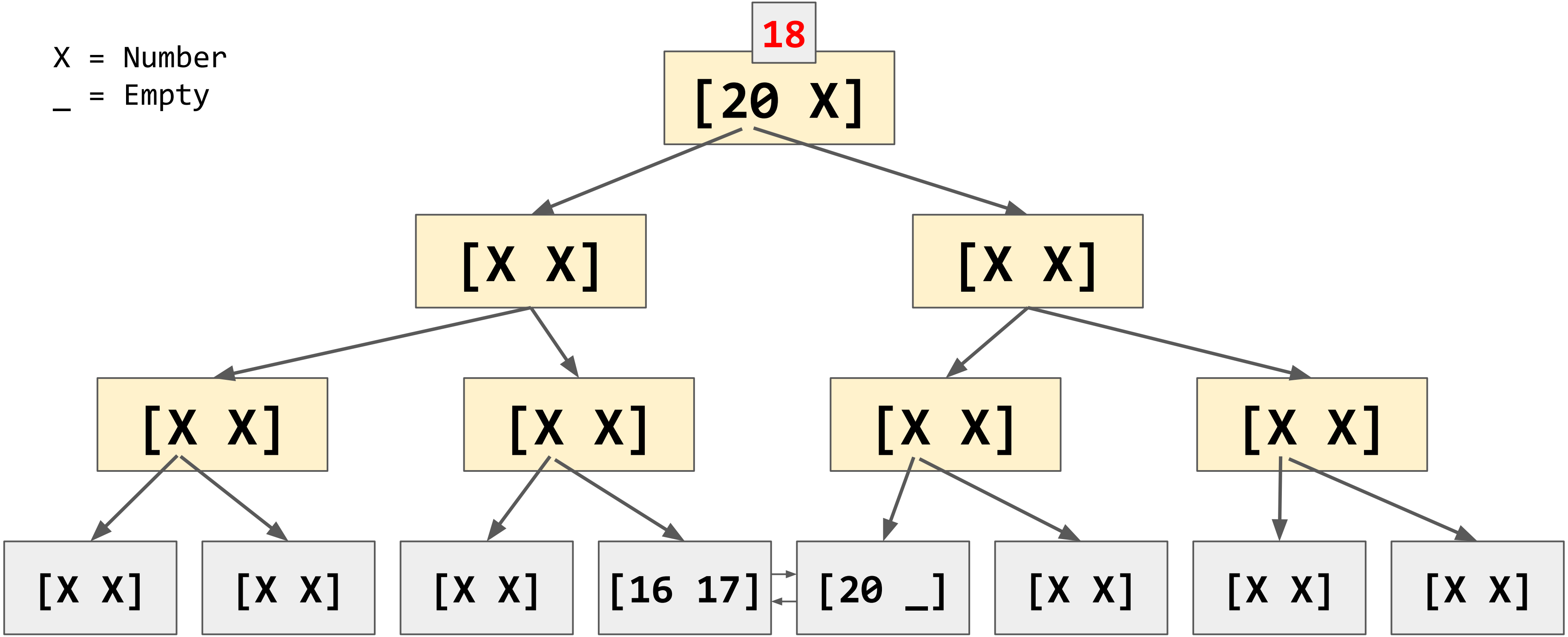


Split - insert 3

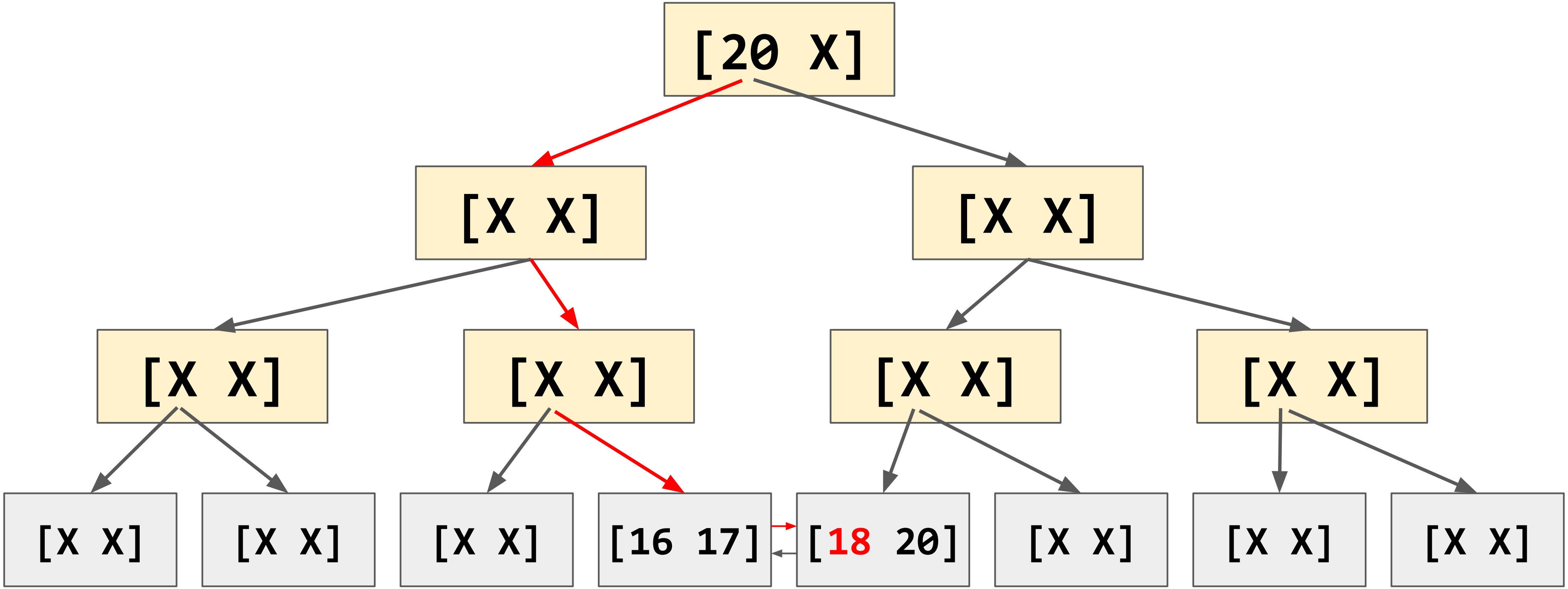


Edge cases

X = Number
_ = Empty

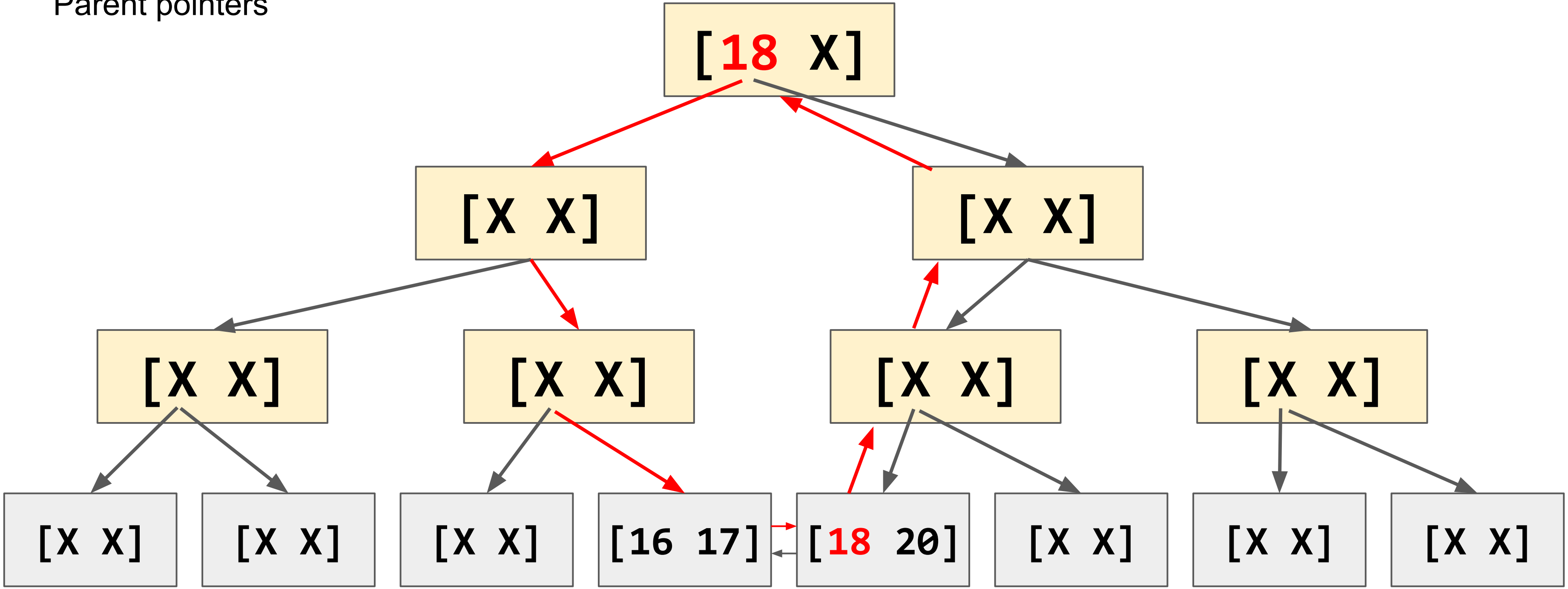


Edge cases



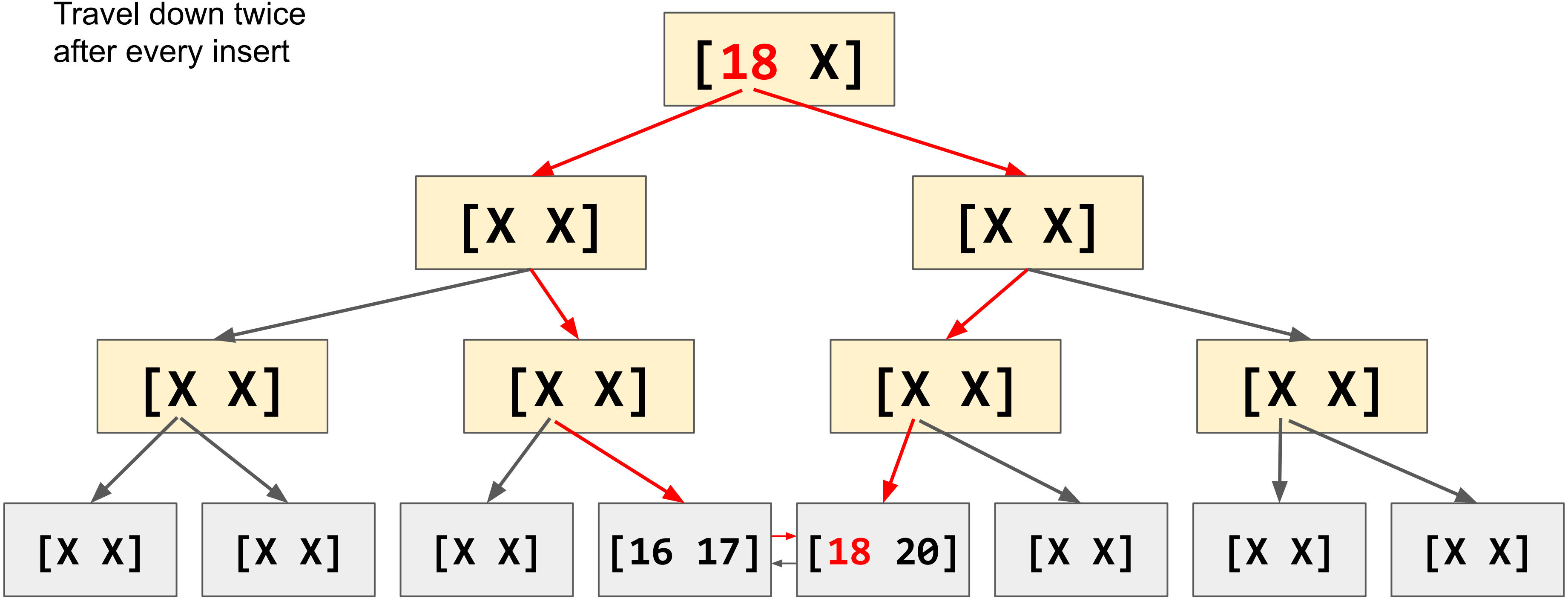
Edge cases

Parent pointers



Edge cases

Travel down twice
after every insert



Rotations for insertion

- Internal -> Leaf
  - left, right, split
- Internal -> Internal
  - left, right, split
- Root -> Internal
  - split

No Padding

```
("Good Omens", 1990, 432, "0060853980")("A Di
stant Mirror", 1972, 677, "0345349571")XXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

End Padding

```
("Good Omens", 1990, 432, "0060853980")XXXXXX
("A Distant Mirror", 1972, 677, "0345349571")
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

End Padding

```
("Good Omens", 1990, 432, "0060853980")XXXXXX
("A Distant Mirror", 1972, 677, "0345349571")
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

```
("Good Omens", 1990, 432, "0060853980")X
("A Distant Mirror", 1972, 677, "0345349
571")XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

Field Padding

```
("Good OmensXXXXXX", 1990, 432, "0060853980")
("A Distant Mirror", 1972, 677, "0345349571")
("$100000 under th", 2009, 396, "0307274939")
```

Strings vs Bits

```
("Good OmensXXXXXX", 1990, 432, "0060853980")
("A Distant Mirror", 1972, 677, "0345349571")
("$100000 under th", 2009, 396, "0307274939")
```

```
101101010101100100010101011011010101011010100
101000101001010110101010100101010101001010010
100010010101101010110101010010101110101011000
```

Python/C Structs

```
("Good OmensXXXXXX", 1990, 432, "0060853980")
("A Distant Mirror", 1972, 677, "0345349571")
("$100000 under th", 2009, 396, "0307274939")
```

```
(16 char[], int, int, 10 char[])
```

```
import struct
format_string = ‘16s i i 10s’
```

Future

Things I should do

- Finalize rotations so B+ Tree works properly
- Use a file and structs to make the B+ Tree work on disk

Cool things I could do

- Implement delete
- Binary Search Trees instead of Lists for internal nodes
- Read through an actual implementation
    - B+ Tree resources are extremely limited