LF *live*

MENTORSHIP SERIES

LF Live MENTORSHIP SERIES

The Maple Tree:

Structure and Algorithms

Liam R. Howlett,

Consulting Member of Technical Staff, Oracle

LF live MENTORSHIP SERIES

Introduction and Agenda

- Overview of the Maple Tree
- Motivation
- Node contents
- Tree example
- Searches
- Modifications

Scope and Expectations

- Fundamental understanding of tree data structures

LF Live MENTORSHIP SERIES

The Maple Tree Overview

- **B-tree**
  - Multiway branching data structure
  - Every node has a maximum number of children
  - All nodes (except root) meets the minimum number of elements
  - All leaves appear at the same level

- **Maple Tree**
  - Stores non-overlapping ranges
  - RCU-safe
  - Slots contain entries or internal nodes
  - An index is often called a key
  - A group of indices are called ranges

LF live MENTORSHIP SERIES

Motivation

- **Benefits**
  - Cache efficient
  - Faster searches
  - Reduced size for NULLs
  - Reduce or remove lock contention

- **In-development**
  - Increase the write speed
  - More node types

LPC 2019

LPC 2022

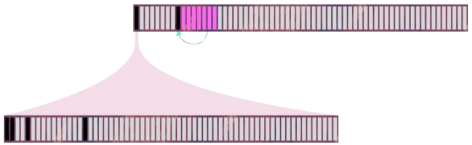
LF live MENTORSHIP SERIES

Motivation

- Radix tree
  - Reduced size for NULLs
  - Store 0,1, 4, 15, 512-1023

LPC 2022

Radix Tree



Maple Tree

Motivation: Virtual Memory Area

- **Initial released in 6.1**
  - No locking changes
- **Per-vma locks in 6.4 to 6.6**
  - Avoid locking on pagefault path
  - High thread count Android apps get ~20% faster start
- **Userfaultfd operations**
  - Uses per-vma locks
  - Android runtime Garbage Collector
    - compaction ~3s to <500ms
    - Uninterruptible sleep time >20s to ~10ms

LF Live MENTORSHIP SERIES

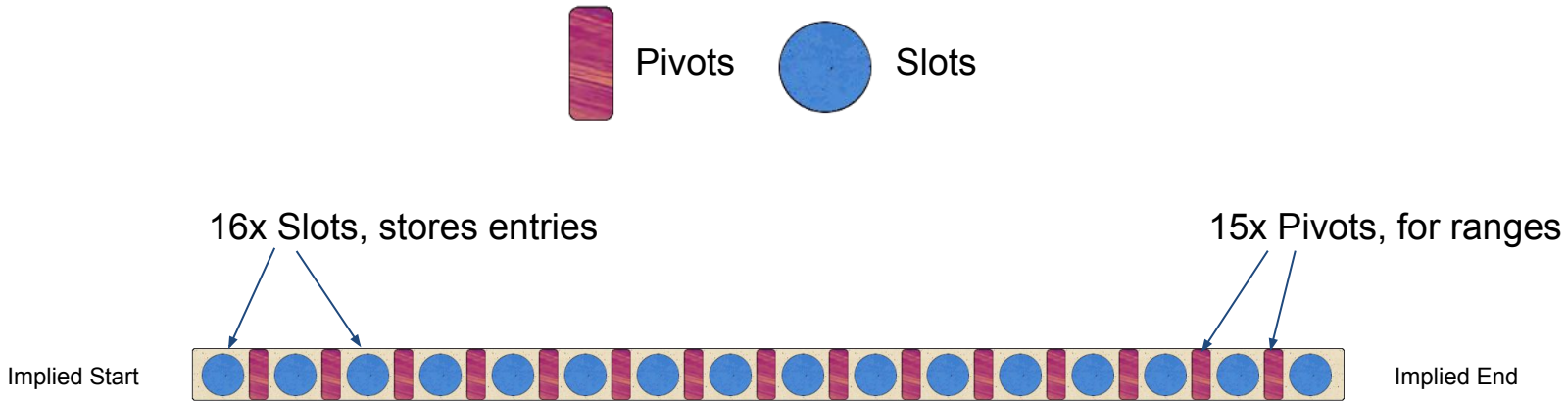
Other Linux Kernel Maple Tree Users

- mm_struct: vma tree
- arm64 kvm_host: hypercall filtering
- regmap: register cache
- btrfs: lru list
- EFI: efi_mm
- tboot: tboot_mm
- IRQ descriptors
  - Support INT_MAX IRQs instead of NR_IRQS+8192
- fs/libfs.c
  - 11.6% improvement aim9.disk_src.ops_per_sec
  - Should be improved with planned features

Libfs change

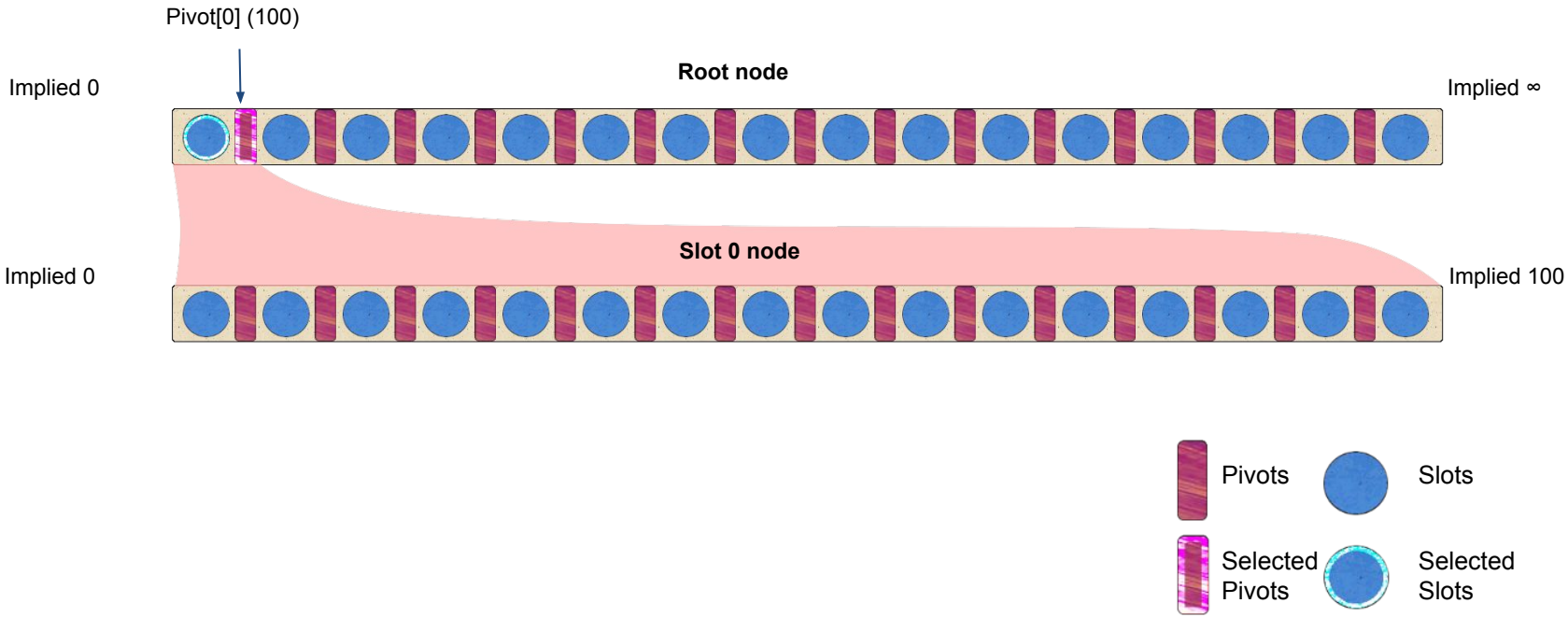
LF live MENTORSHIP SERIES

Maple Tree Node Layout



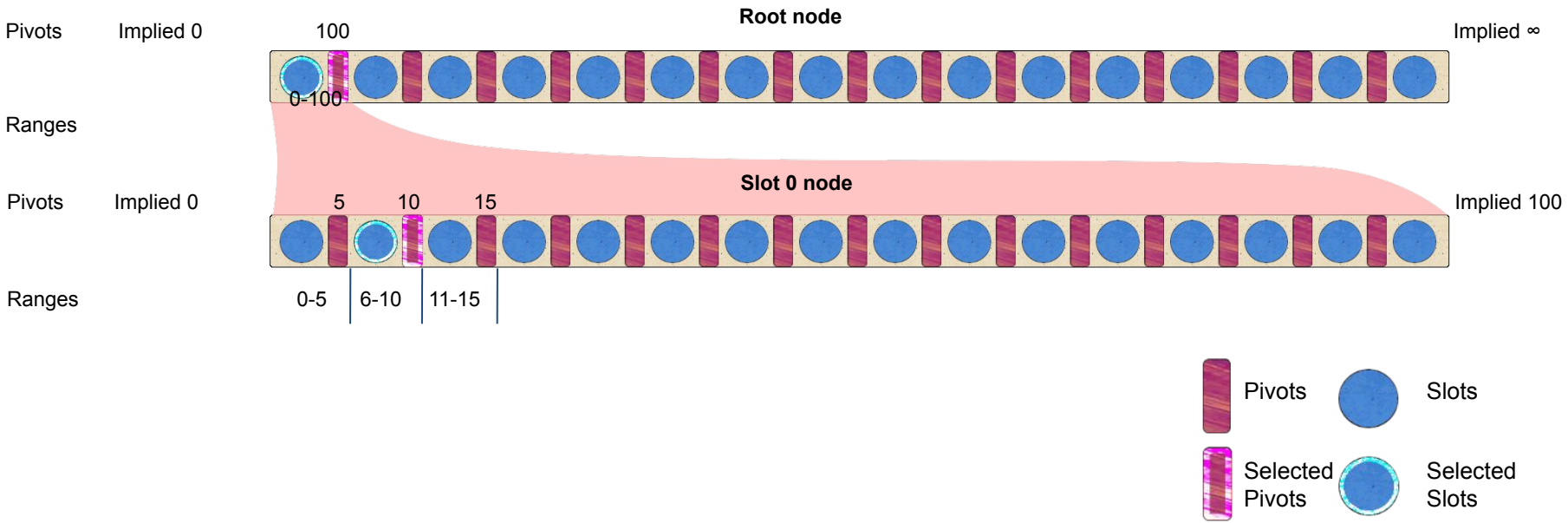
LF Live MENTORSHIP SERIES

Maple Tree Example



LF live MENTORSHIP SERIES

Maple Tree Look up



Maple Tree Write

- **More expensive than searching**
  - Tree nodes are not embedded
  - Unknown if the tree needs to expand, reduce, or remain constant
- **Different write types**
  - Direct entry replacement
  - Node replacement
  - Node Split - potentially cascading upwards
  - Node rebalance - potentially cascading upwards
  - Spanning store - write spans more than one node

LF live MENTORSHIP SERIES

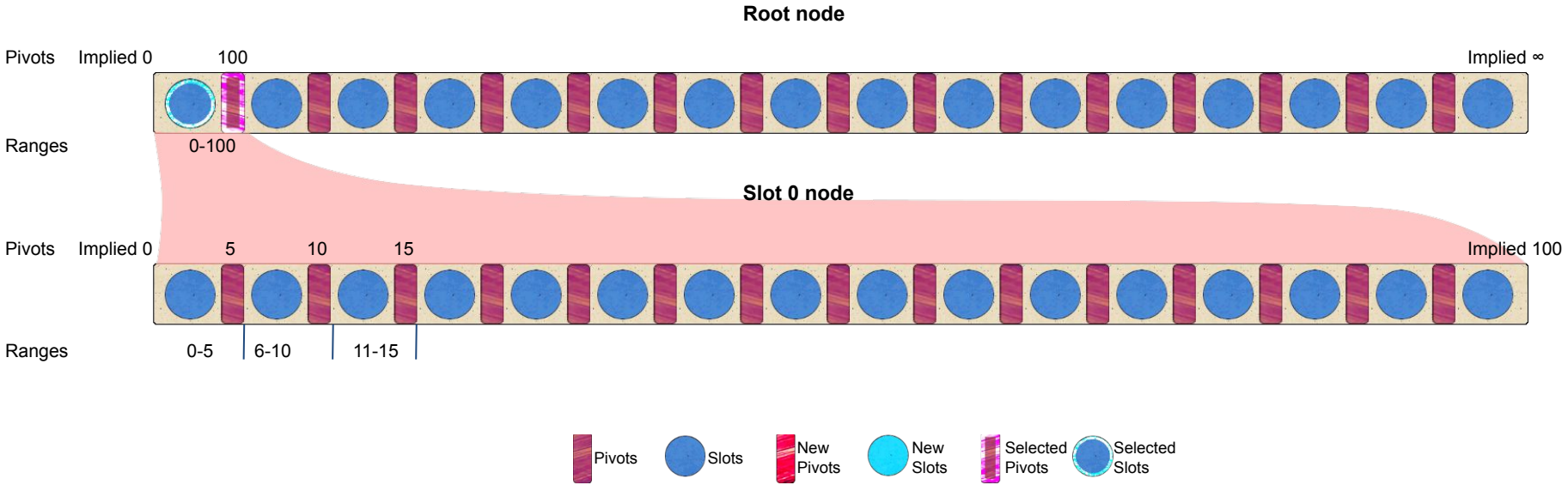
Maple Tree Write: Direct Replacement

- Fastest write
- Overwrites entry

LF live MENTORSHIP SERIES

Maple Tree Write: Direct replacement

Store [6-10]

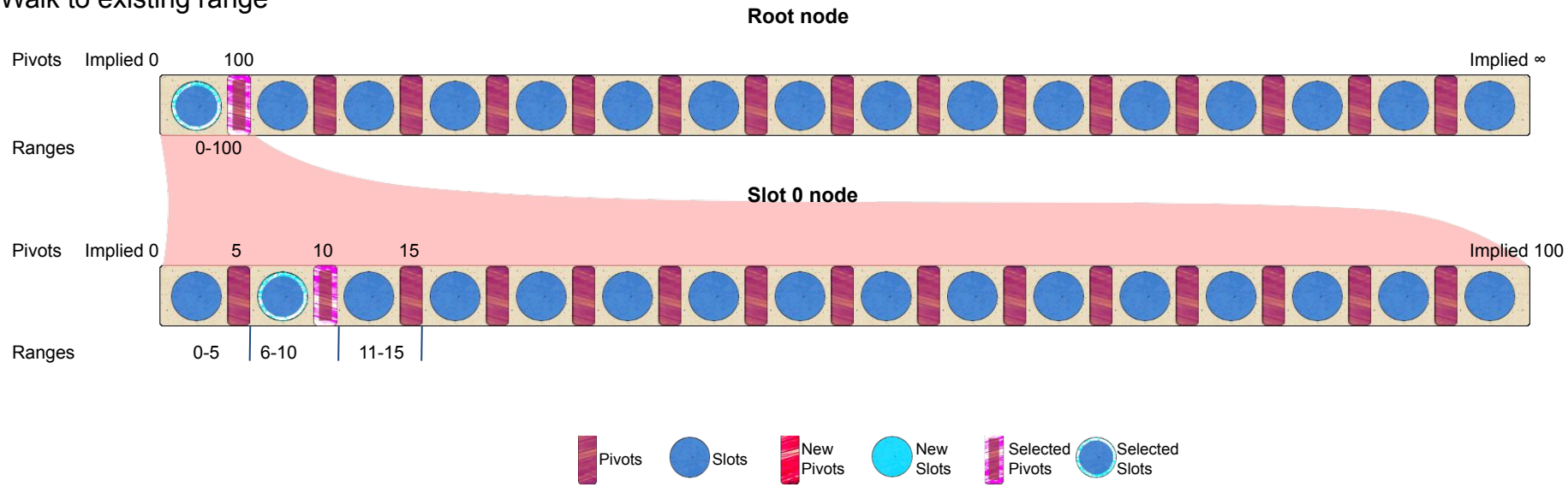


LF live MENTORSHIP SERIES

Maple Tree Write: Direct replacement

Store [6-10]

Walk to existing range

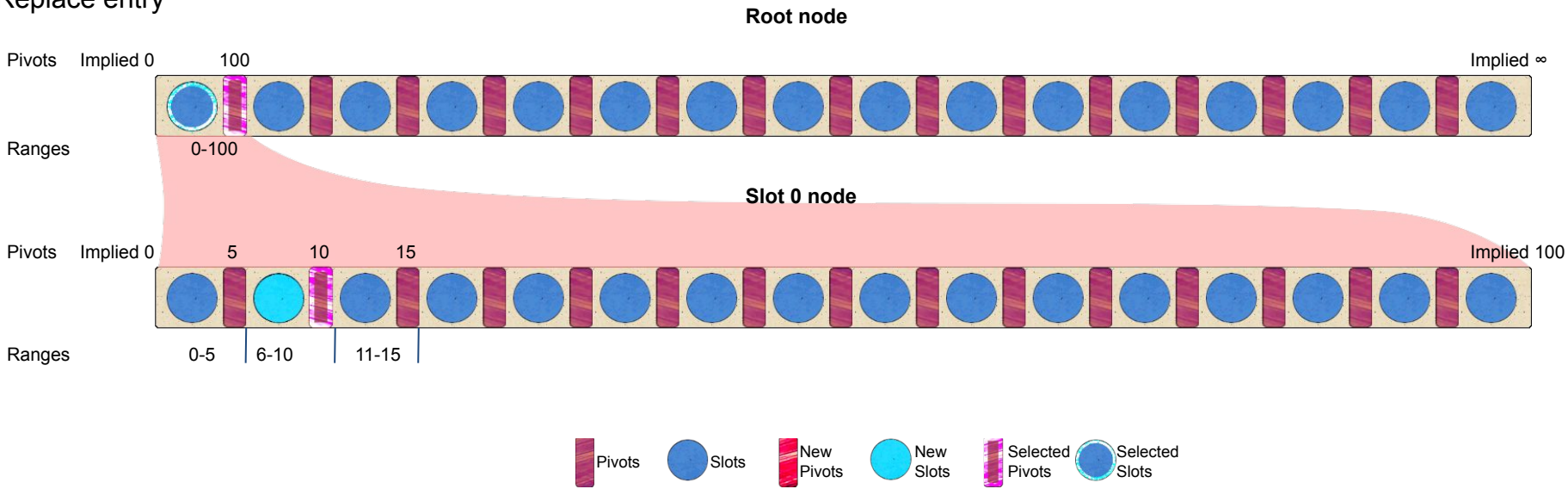


LF live MENTORSHIP SERIES

Maple Tree Write: Direct replacement

Store [6-10]

Replace entry



LF live MENTORSHIP SERIES

Maple Tree Write: Node Replacement

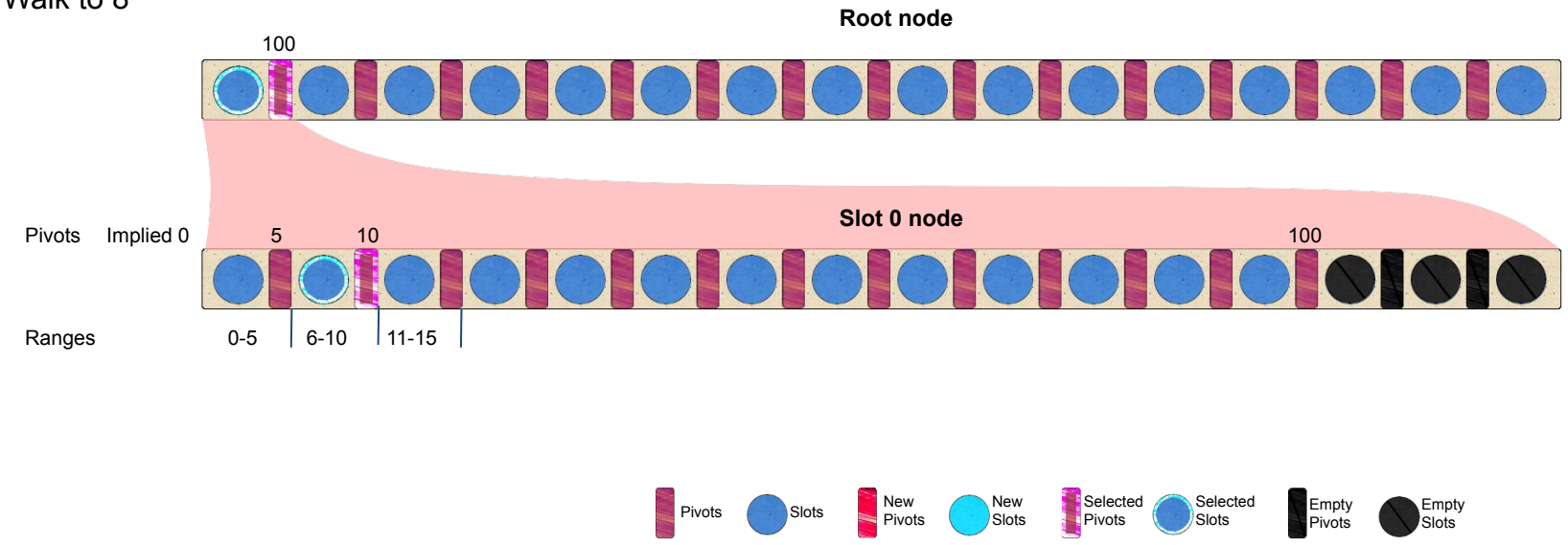
- Uses a new node
- Copy and update of RCU

LF Live MENTORSHIP SERIES

Maple Tree Write: Node Replacement

Store [8-9]

Walk to 8

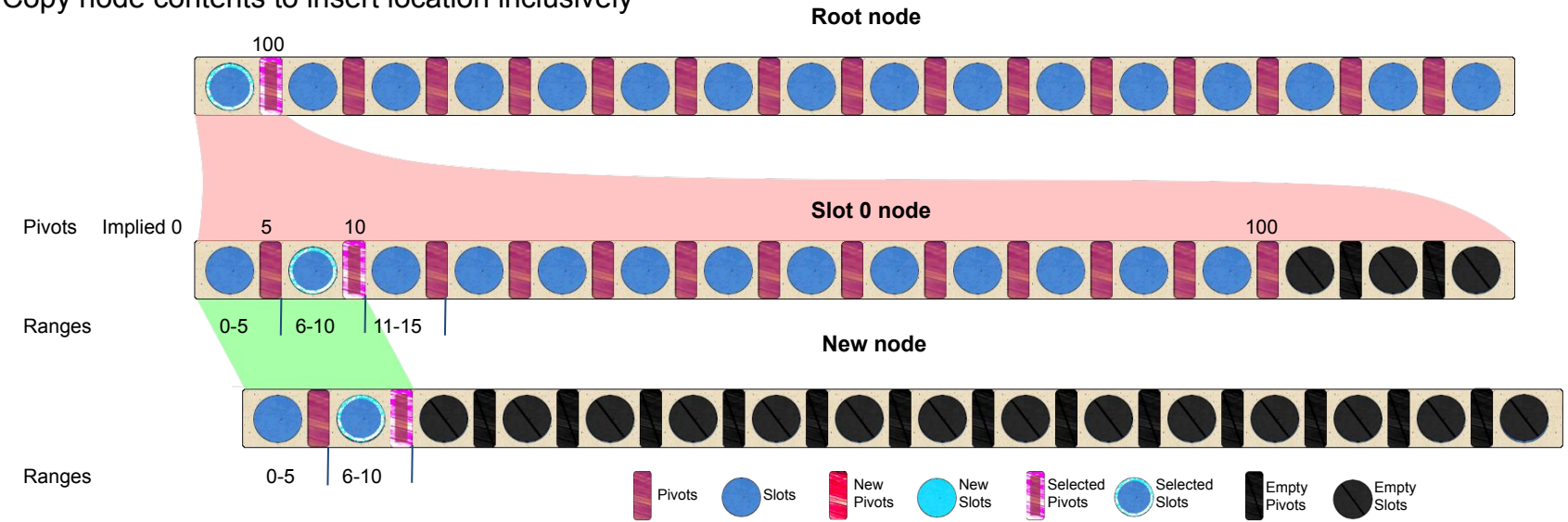


LF live MENTORSHIP SERIES

Maple Tree Write: Node Replacement

Store [8-9]

Copy node contents to insert location inclusively



LF live MENTORSHIP SERIES

Maple Tree Write: Node Replacement

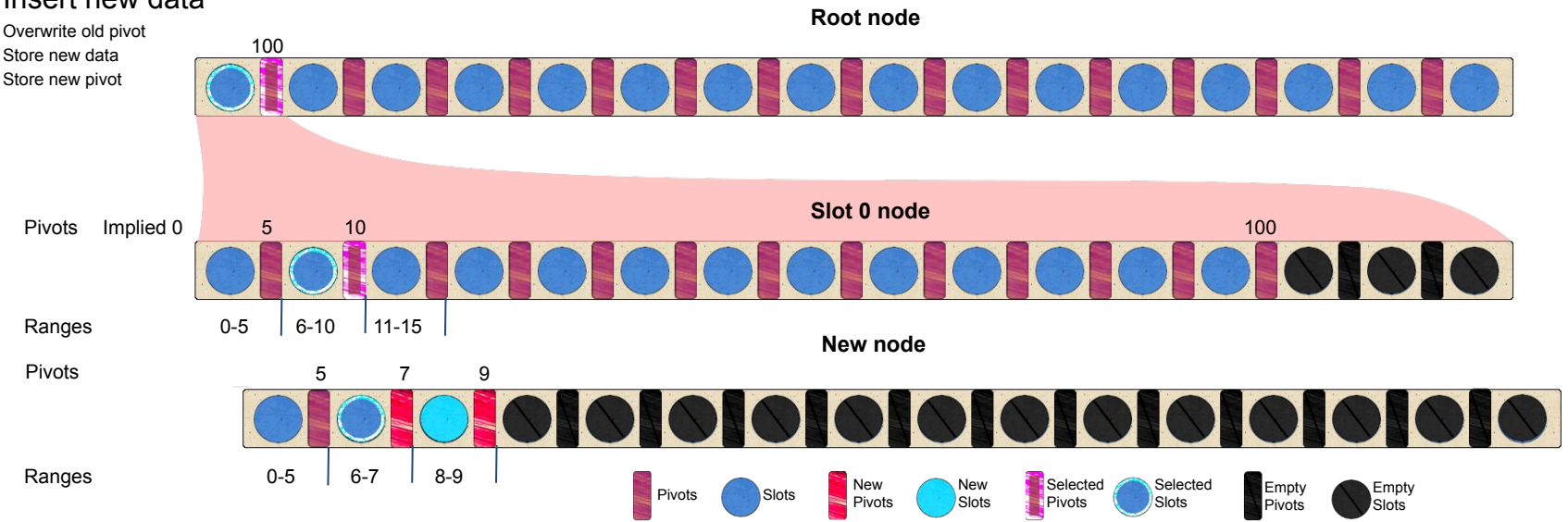
Store [8-9]

Insert new data

Overwrite old pivot

Store new data

Store new pivot



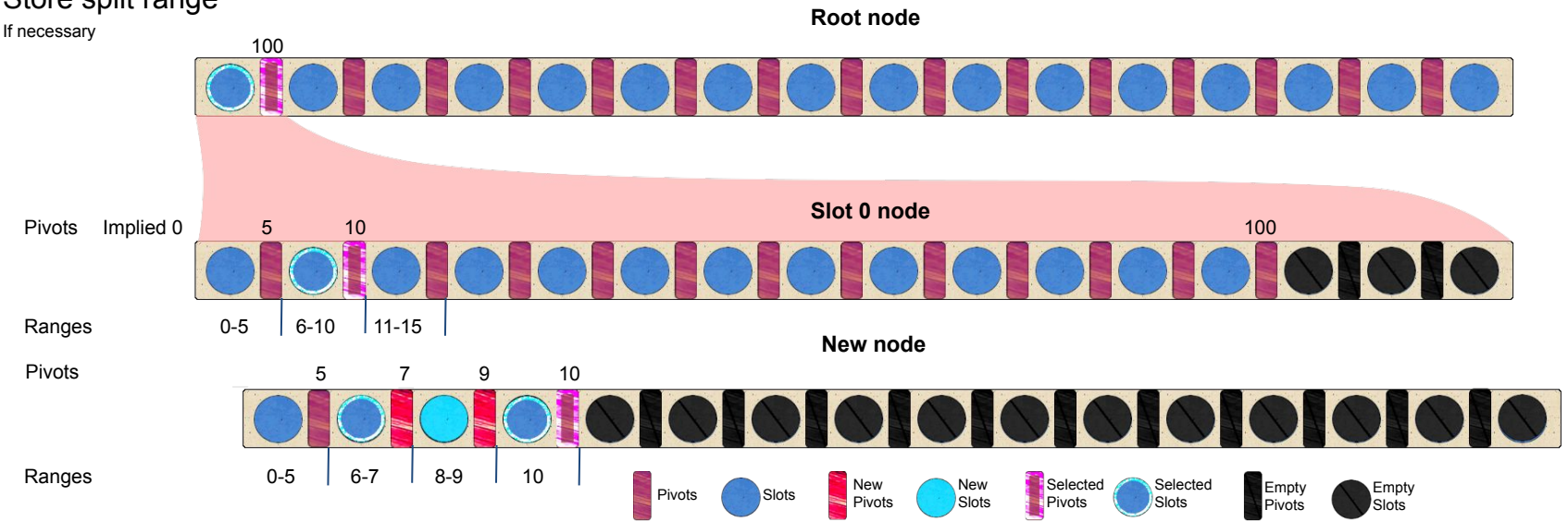
LF live MENTORSHIP SERIES

Maple Tree Write: Node Replacement

Store [8-9]

Store split range

If necessary

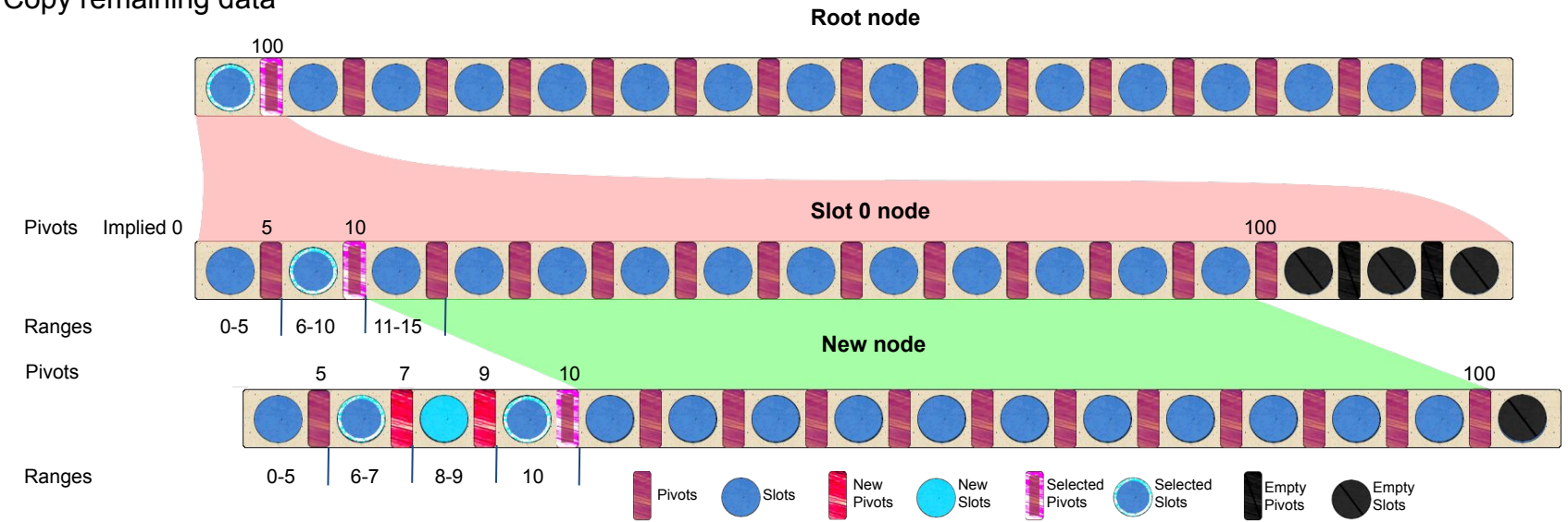


LF Live MENTORSHIP SERIES

Maple Tree Write: Node Replacement

Store [8-9]

Copy remaining data

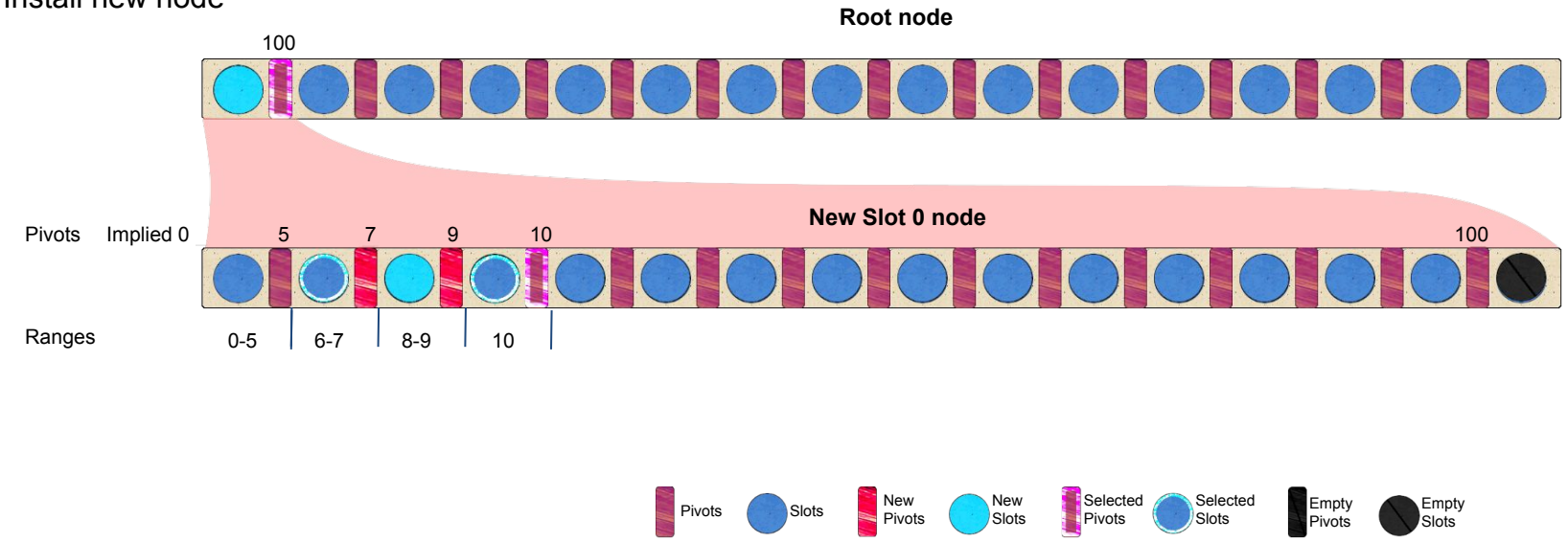


LF live MENTORSHIP SERIES

Maple Tree Write: Node Replacement

Store [8-9]

Install new node



LF live MENTORSHIP SERIES

Maple Tree Write: Splitting Nodes

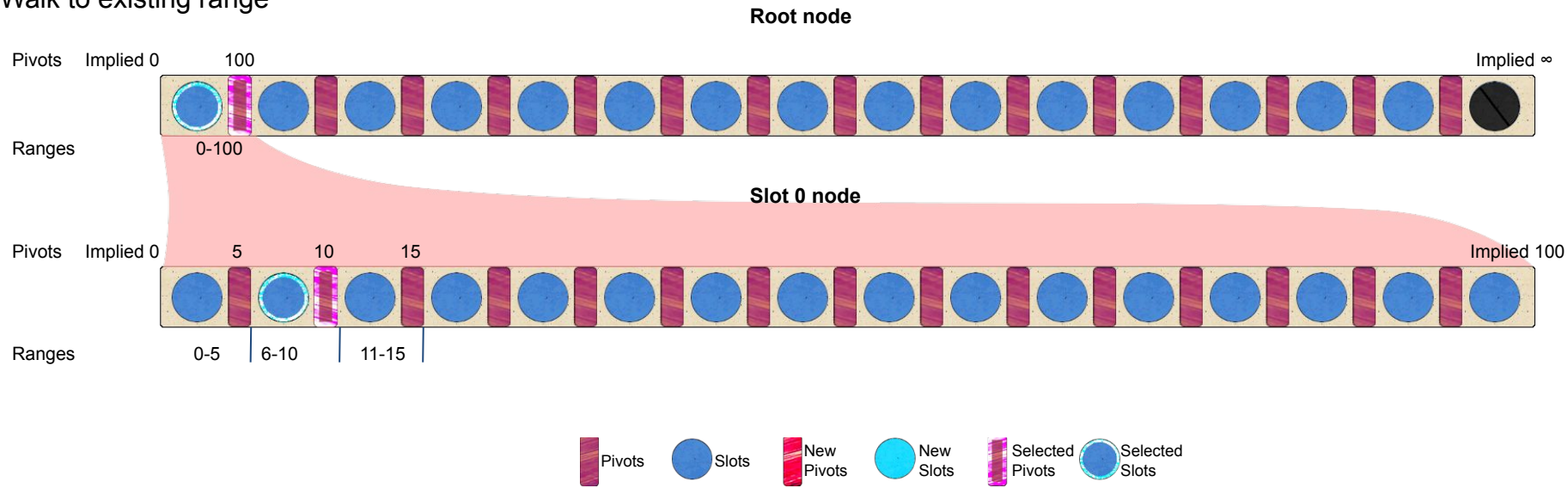
- May push data to siblings
- Split at leaves
- May propagate many levels
- May grow tree height (new root)

LF live MENTORSHIP SERIES

Maple Tree Write: Splitting Nodes

Store [8-9]

Walk to existing range

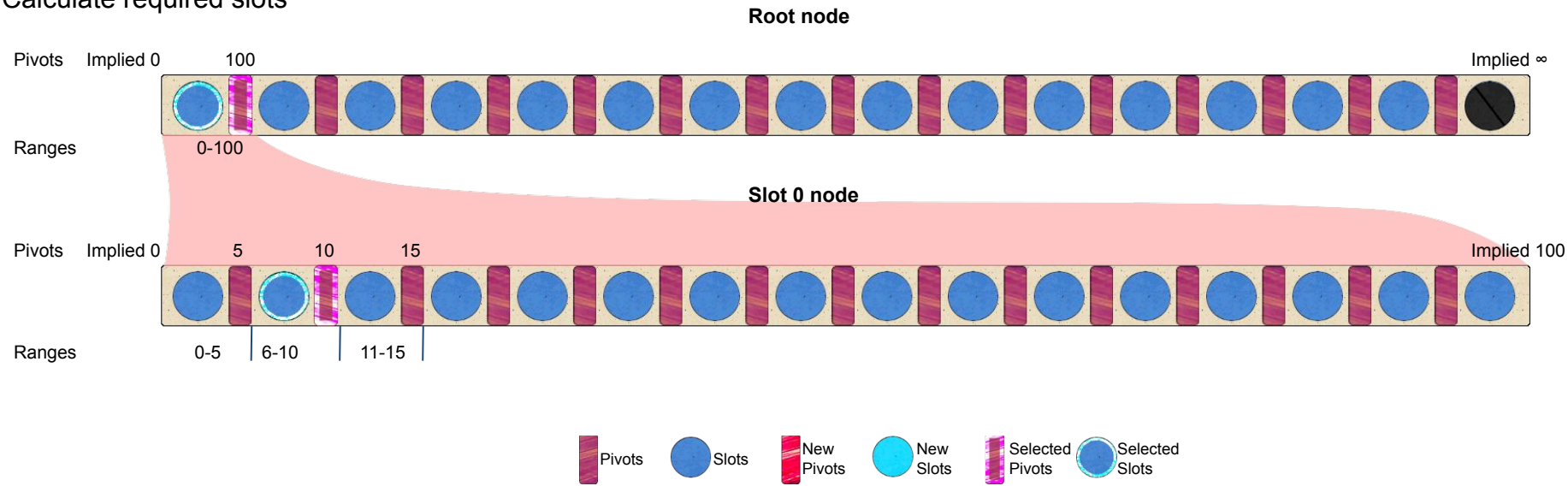


LF live MENTORSHIP SERIES

Maple Tree Write: Splitting Nodes

Store [8-9]

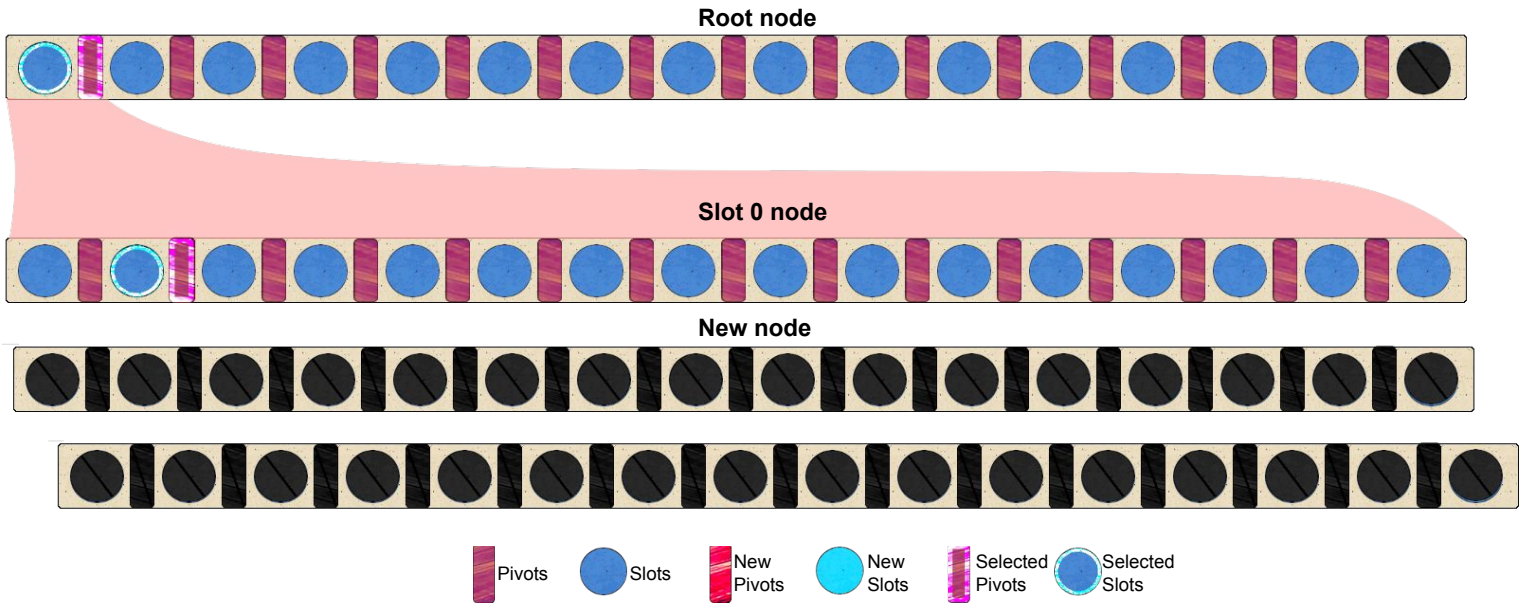
Calculate required slots



LF live MENTORSHIP SERIES

Maple Tree Write: Splitting Nodes

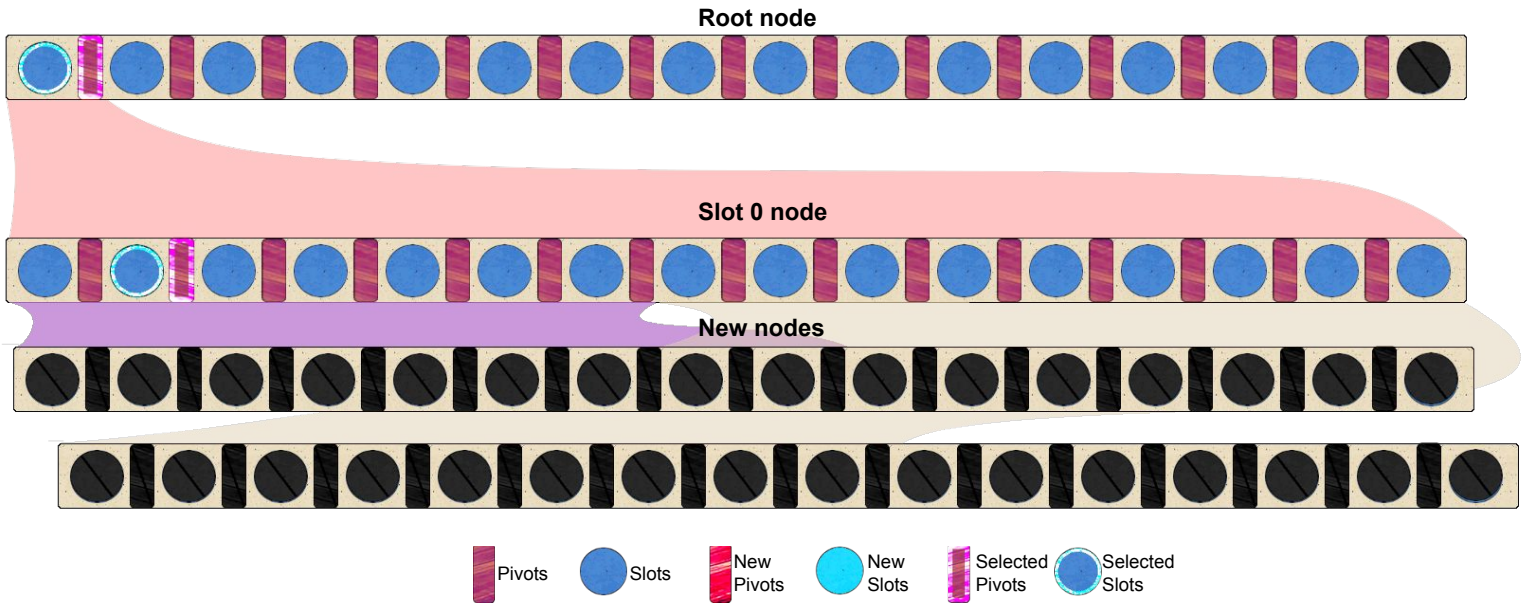
Store [8-9]: Create two new nodes



LF Live MENTORSHIP SERIES

Maple Tree Write: Splitting Nodes

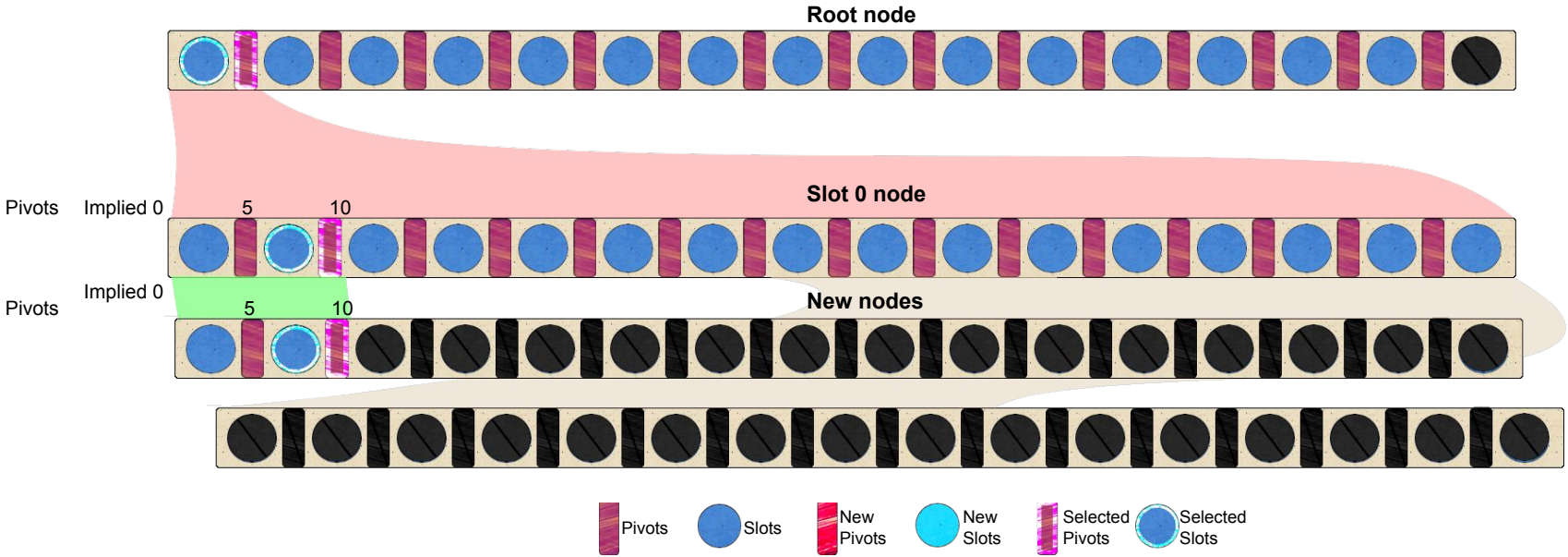
Store [8-9]: Calculate split location



LF Live MENTORSHIP SERIES

Maple Tree Write: Splitting Nodes

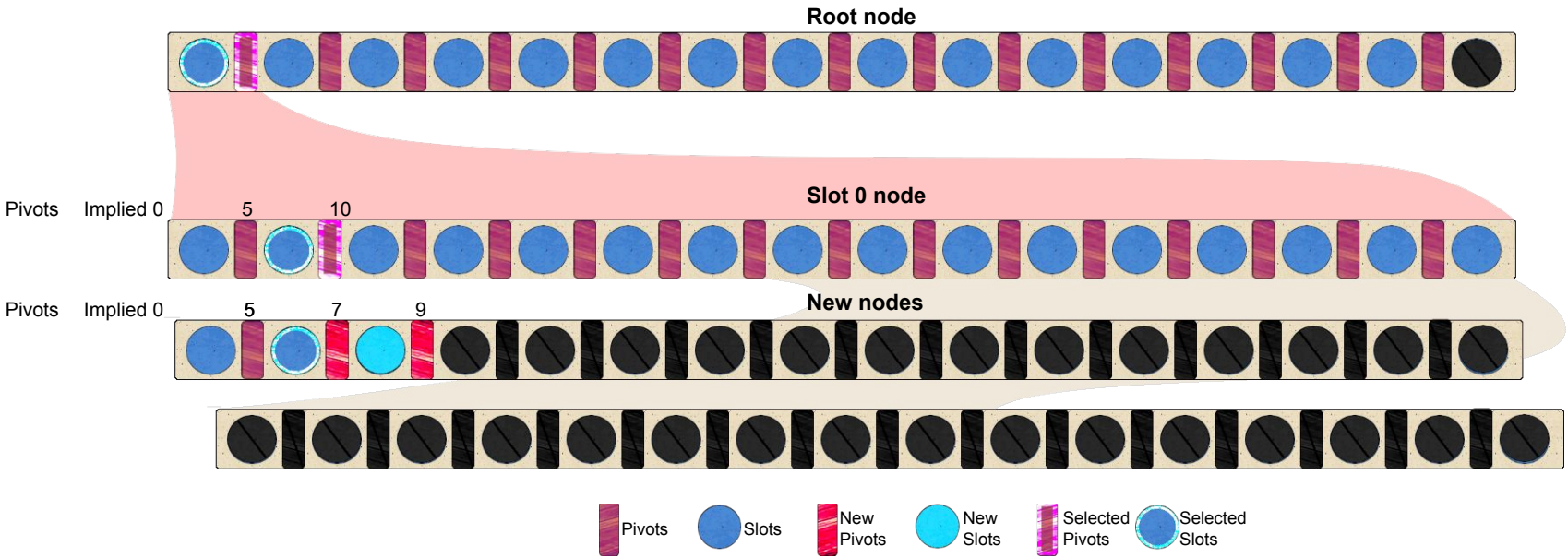
Store [8-9]: Copy node contents to insert location inclusively



LF Live MENTORSHIP SERIES

Maple Tree Write: Splitting Nodes

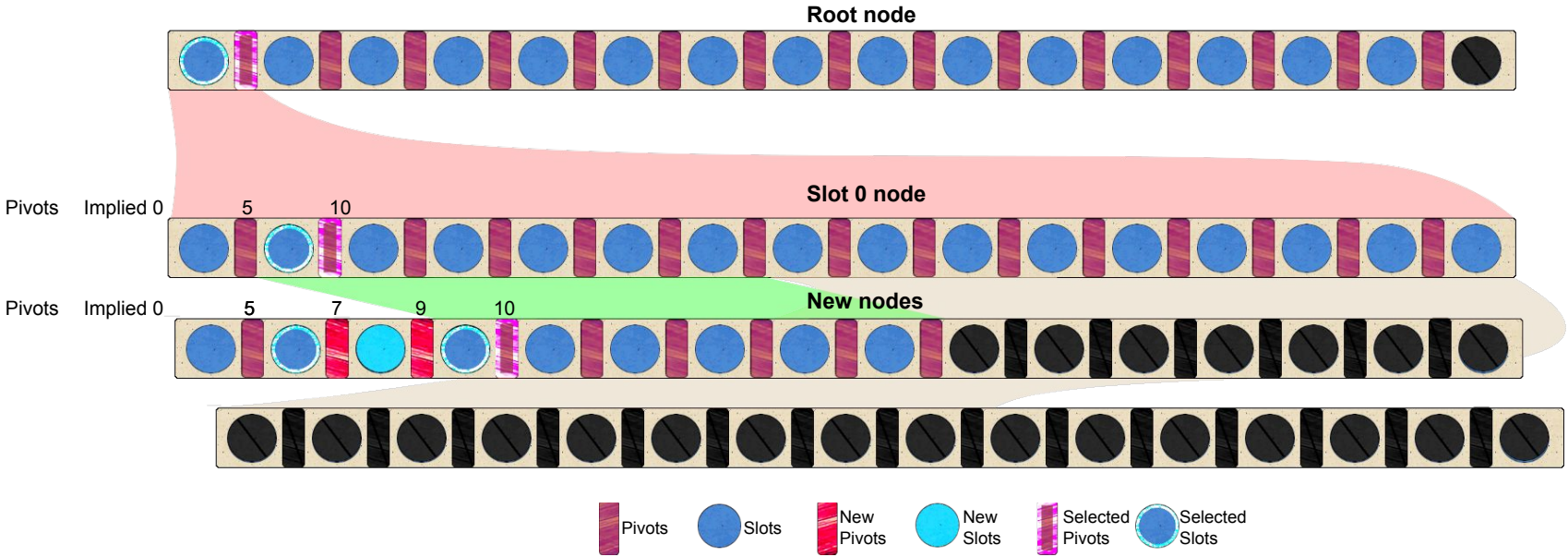
Store [8-9]: Store new entry



LF live MENTORSHIP SERIES

Maple Tree Write: Splitting Nodes

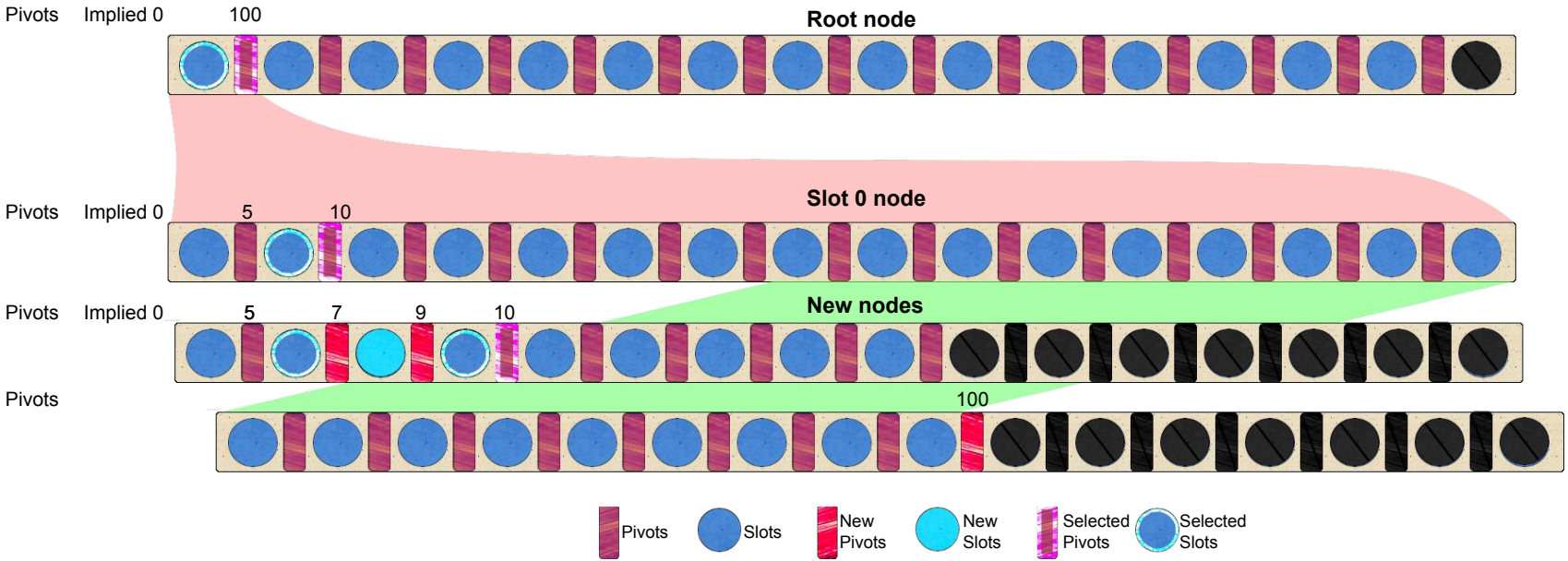
Store [8-9]: Copy data to split location



LF live MENTORSHIP SERIES

Maple Tree Write: Splitting Nodes

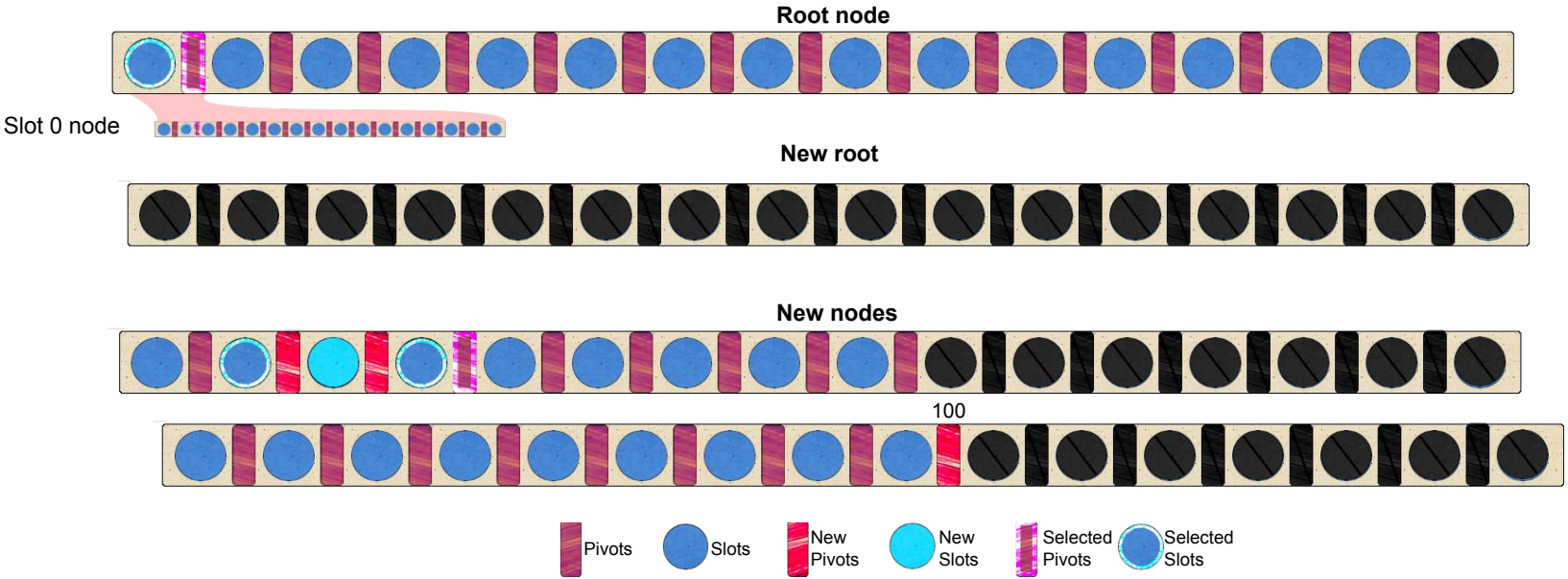
Store [8-9]: Copy remainder of data to second node



LF Live MENTORSHIP SERIES

Maple Tree Write: Splitting Nodes

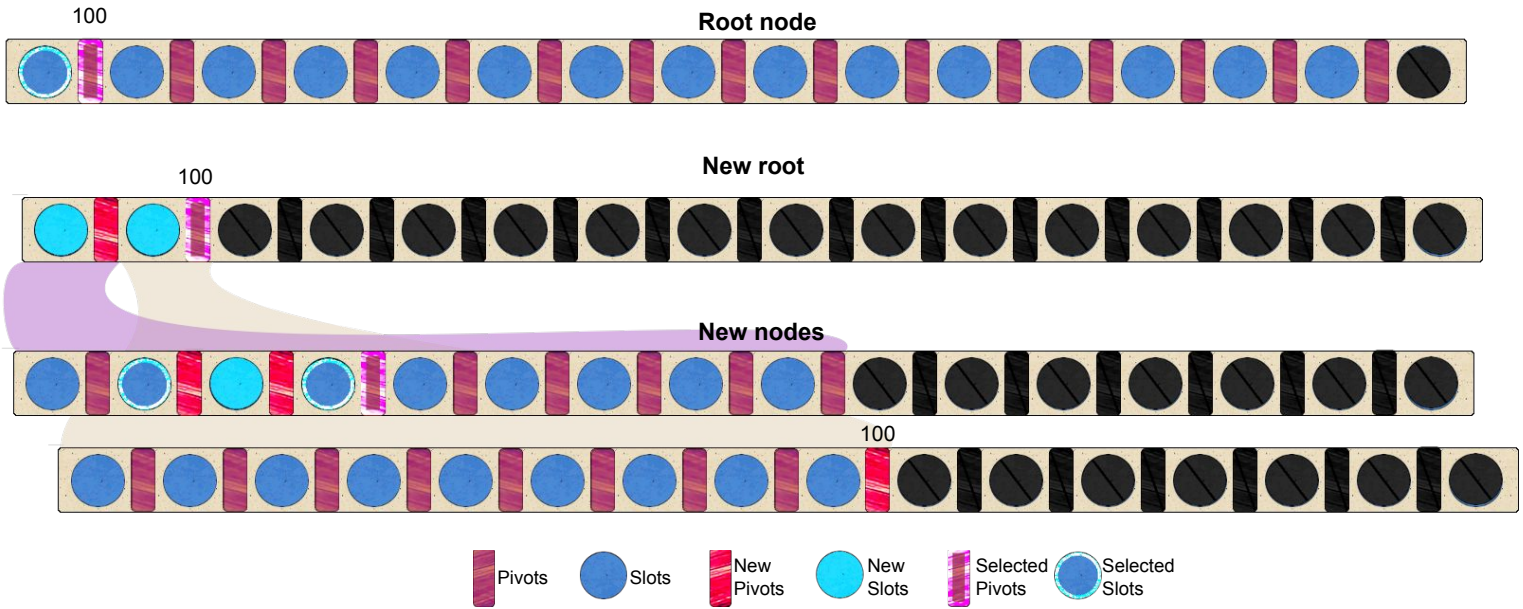
Store [8-9]: Create new root



LF Live MENTORSHIP SERIES

Maple Tree Write: Splitting Nodes

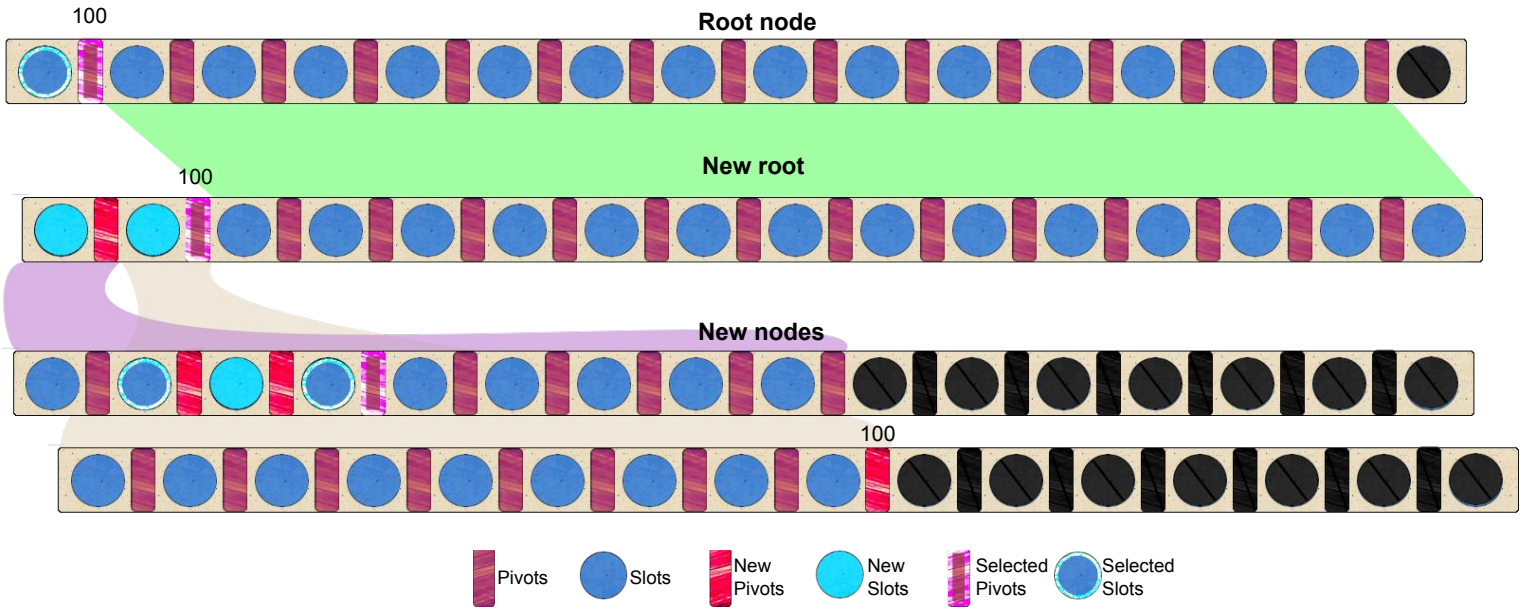
Store [8-9]: Store new nodes in new root



LF Live MENTORSHIP SERIES

Maple Tree Write: Splitting Nodes

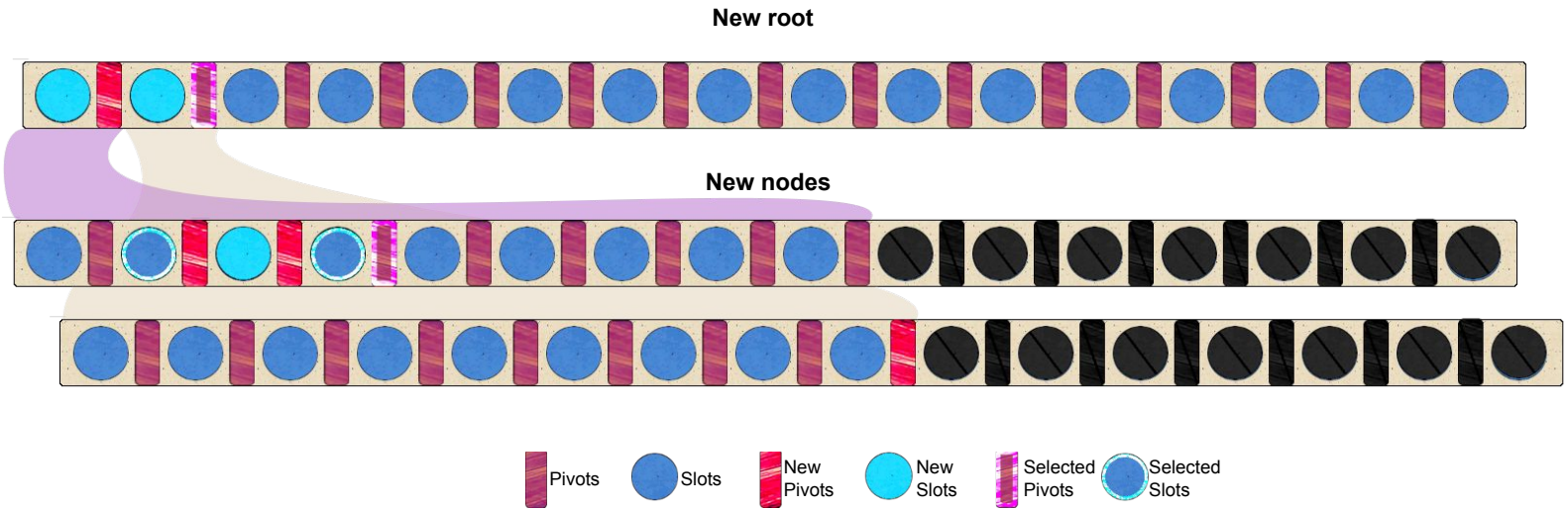
Store [8-9]: Copy remaining root data



LF live MENTORSHIP SERIES

Maple Tree Write: Splitting Nodes

Store [8-9]: Insert new root



Maple Tree Write: Rebalance

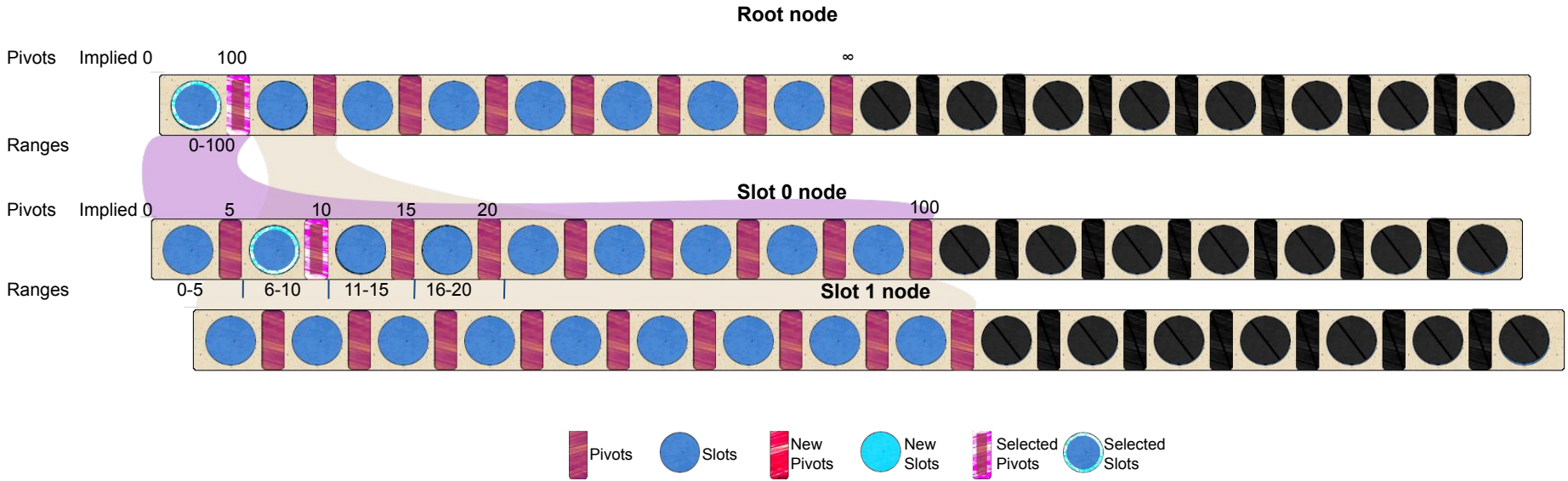
- **Avoid insufficient nodes**
- **Keep data density high**
  - Nodes rebalance between each other
  - Rebalance favour pulling data left
- **Tree height may decrease**

LF live MENTORSHIP SERIES

Maple Tree Write: Rebalance

Store [6-20]

Walk to existing range

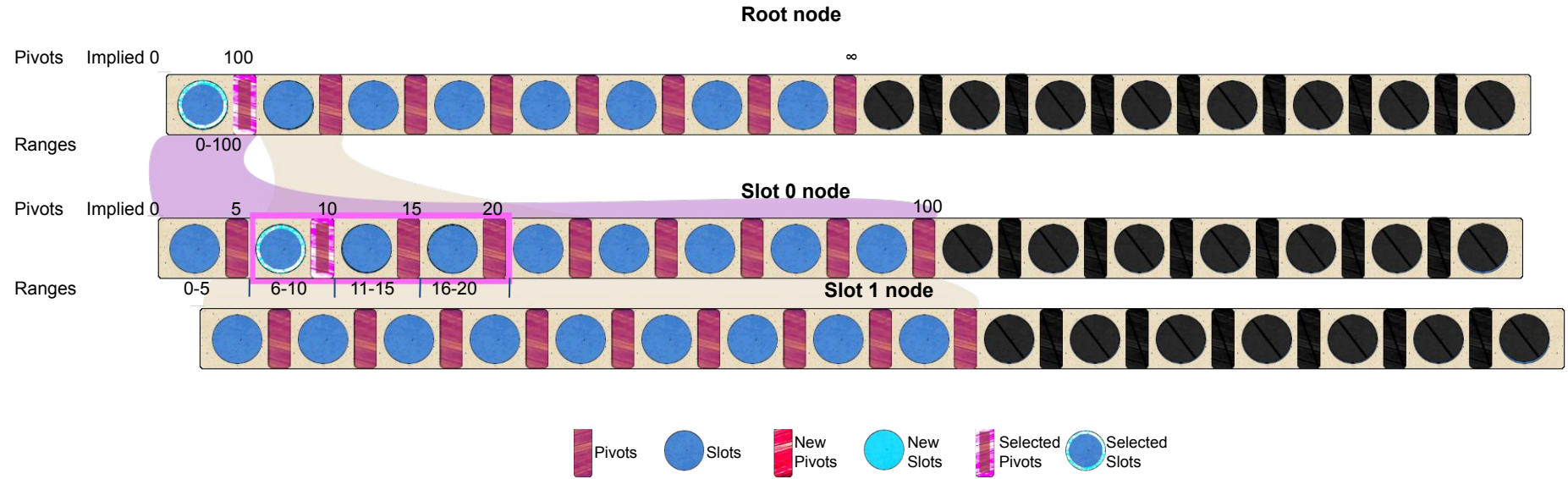


LF live MENTORSHIP SERIES

Maple Tree Write: Rebalance

Store [6-20]

Calculate the size of the new node

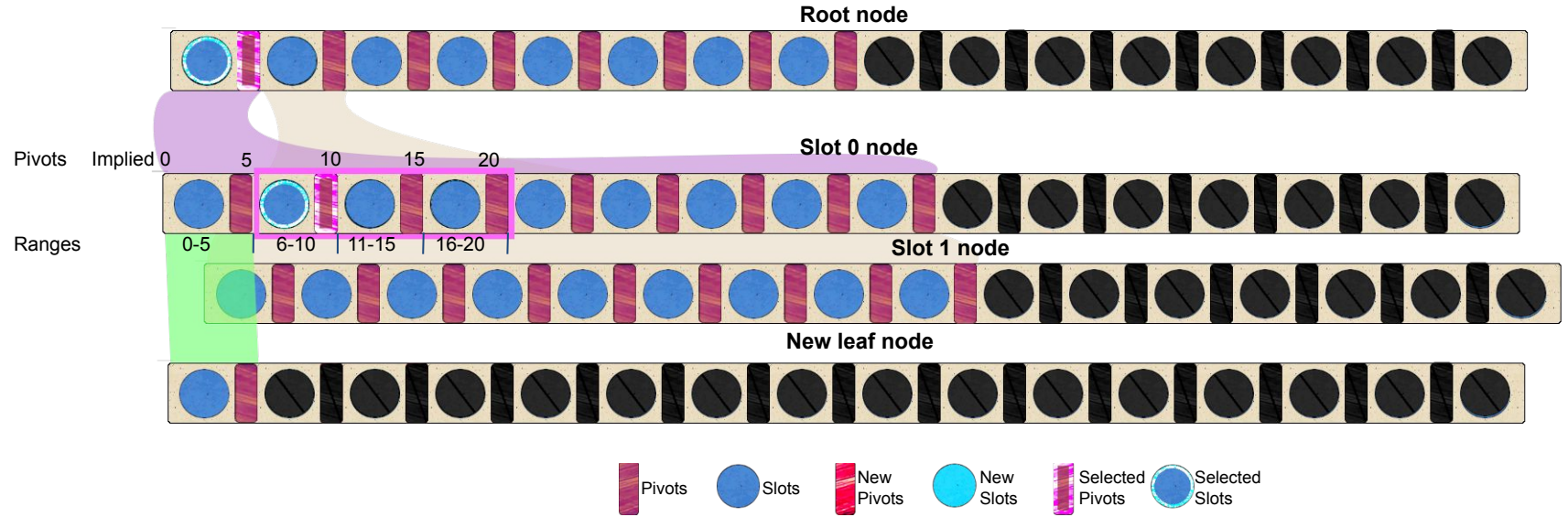


LF live MENTORSHIP SERIES

Maple Tree Write: Rebalance

Store [6-20]

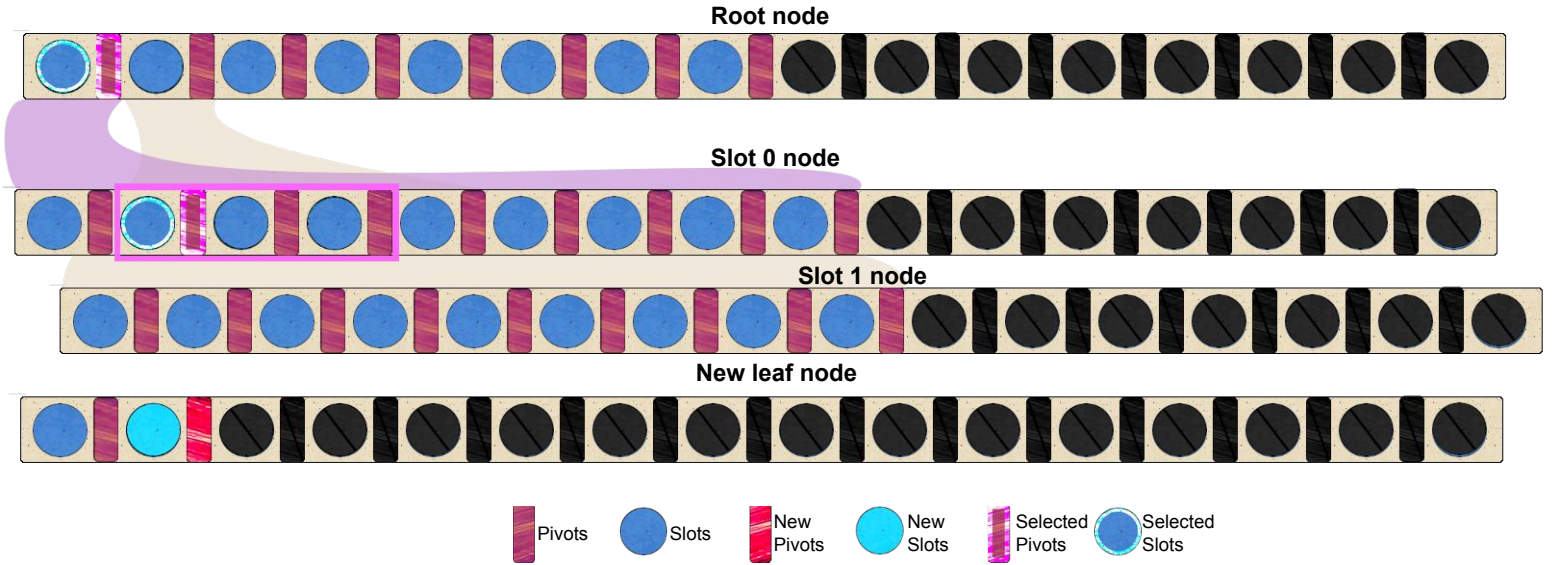
Copy data up to insert location



Maple Tree Write: Rebalance

Store [6-20]

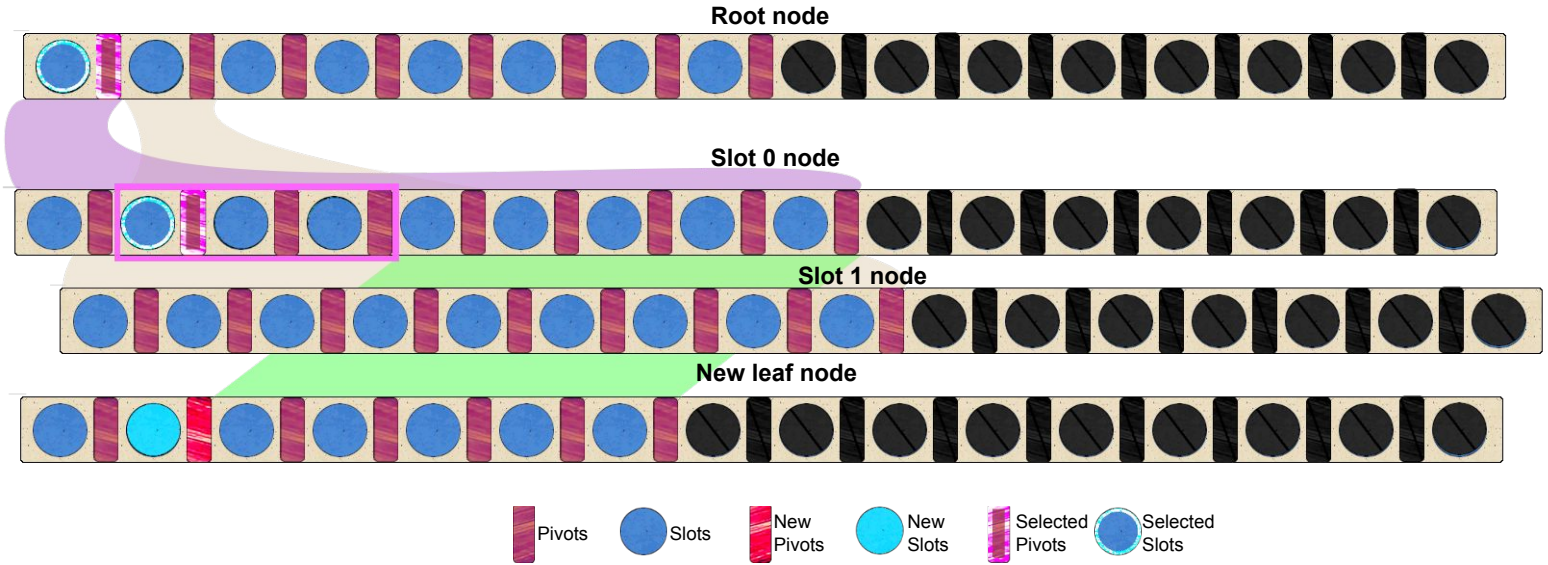
Insert the new data



Maple Tree Write: Rebalance

Store [6-20]

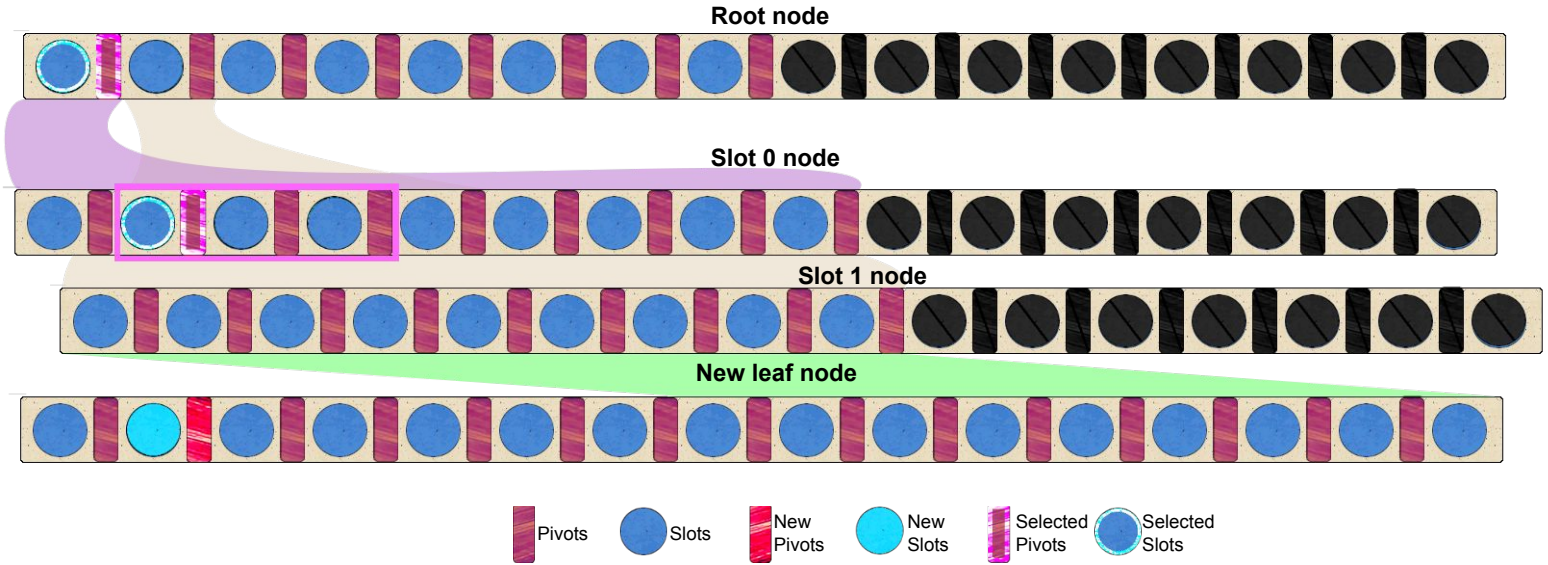
Copy remaining data from the first node



Maple Tree Write: Rebalance

Store [6-20]

Copy all the data from the second node

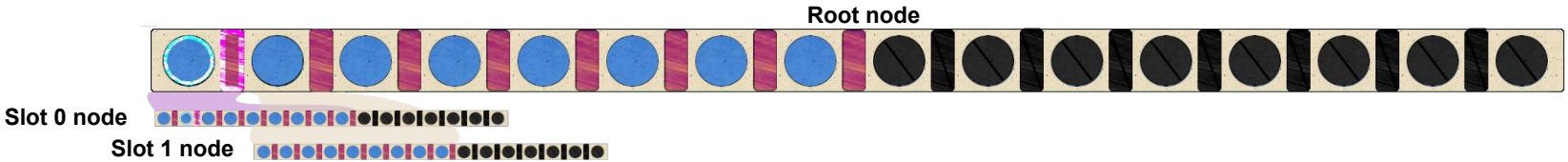


LF live MENTORSHIP SERIES

Maple Tree Write: Rebalance

Store [6-20]

Examine the parent node

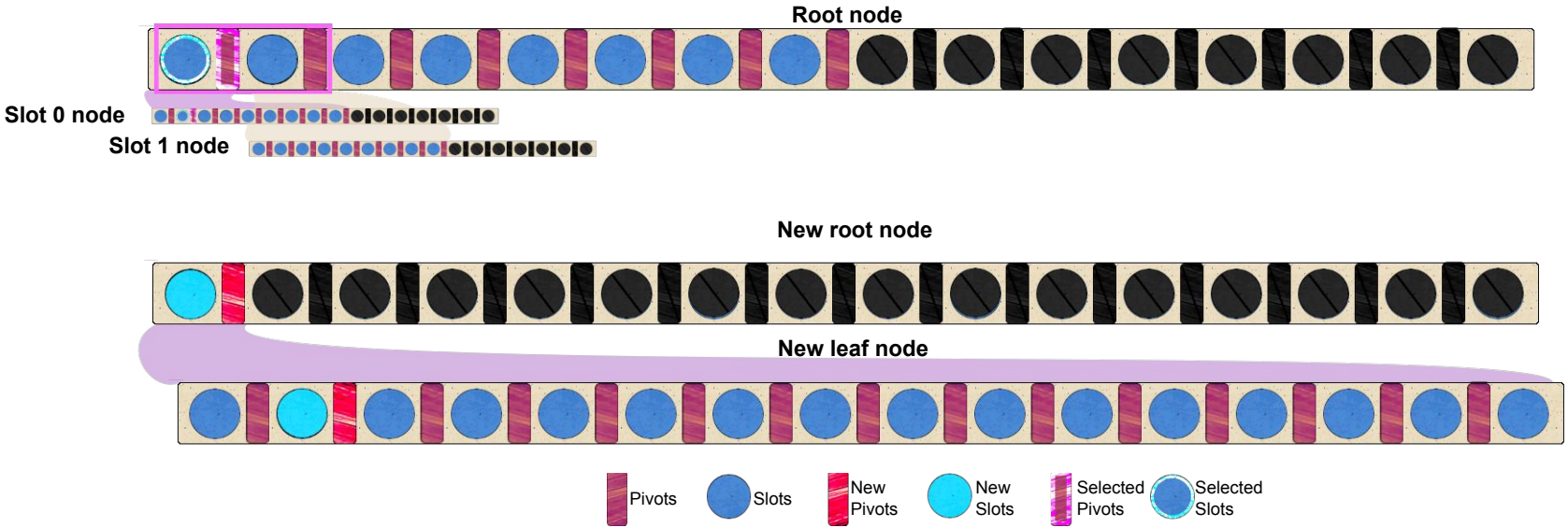


LF live MENTORSHIP SERIES

Maple Tree Write: Rebalance

Store [6-20]

Copy up to to the first node and store the replacement

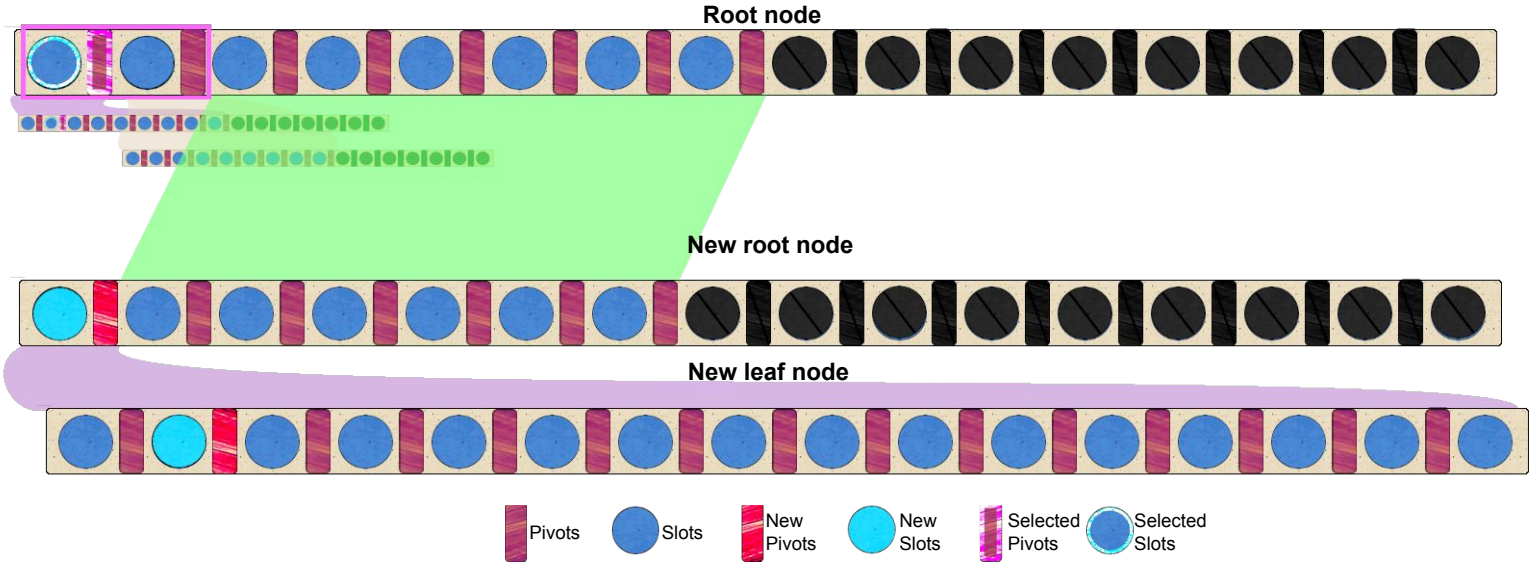


LF live MENTORSHIP SERIES

Maple Tree Write: Rebalance

Store [6-20]

Skip overwritten nodes and copy the remaining data from the parent node

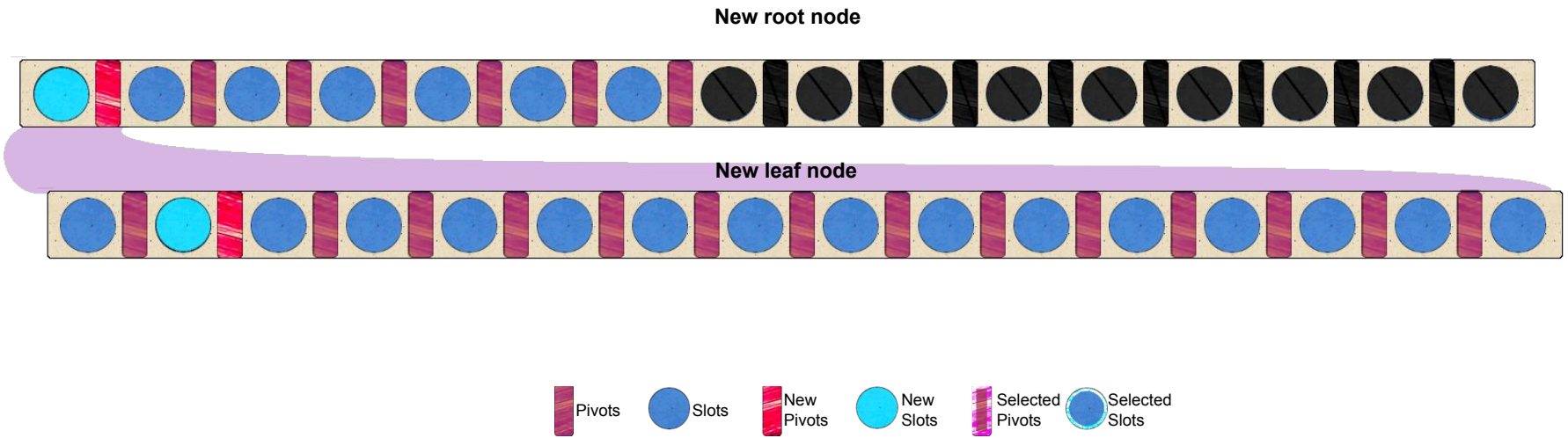


LF live MENTORSHIP SERIES

Maple Tree Write: Rebalance

Store [6-20]

Replace the parent node in the tree



LF live MENTORSHIP SERIES

Maple Tree Write: Spanning Store

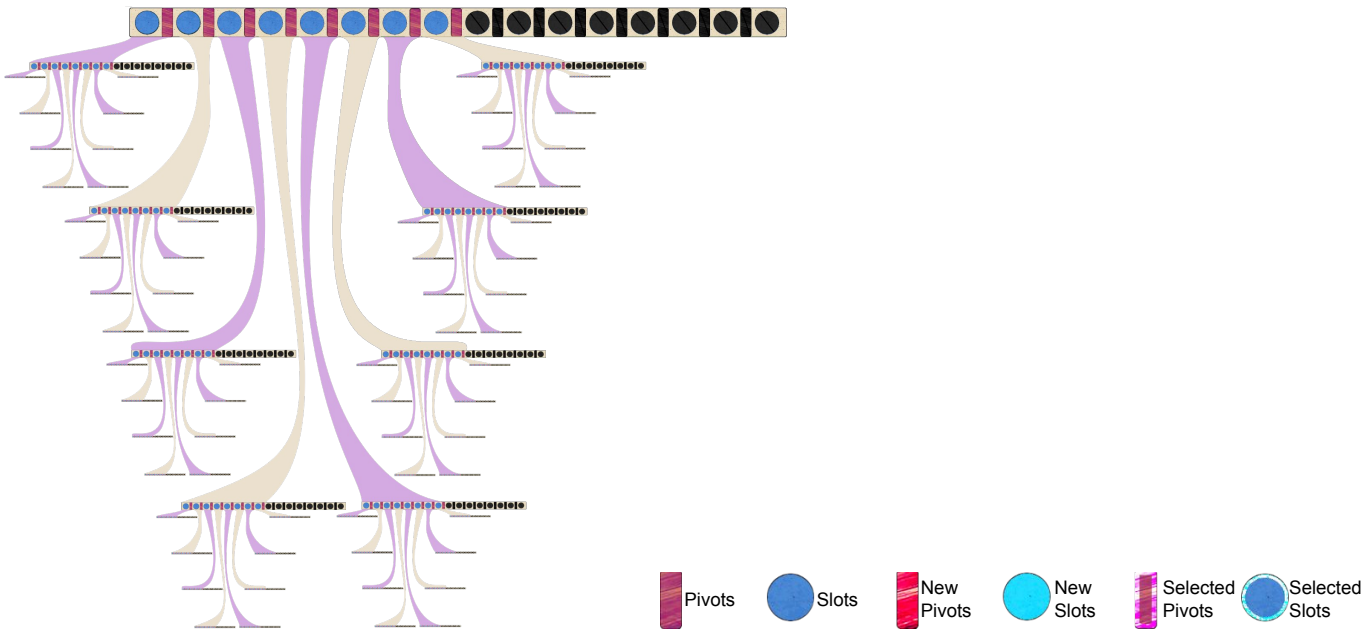
- **Writes span more than one node**
- **Complex operation**
  - Due to ranges
  - Complexity compounded by RCU support
- **Requires rebalancing of parent nodes**
  - May result in:
    - Reuse of multiple tree parts
    - Entire new tree
    - One entry tree

LF Live MENTORSHIP SERIES

Maple Tree Write: Spanning Store

Tree with a height of 3

Contains the minimum entries required to be a valid tree

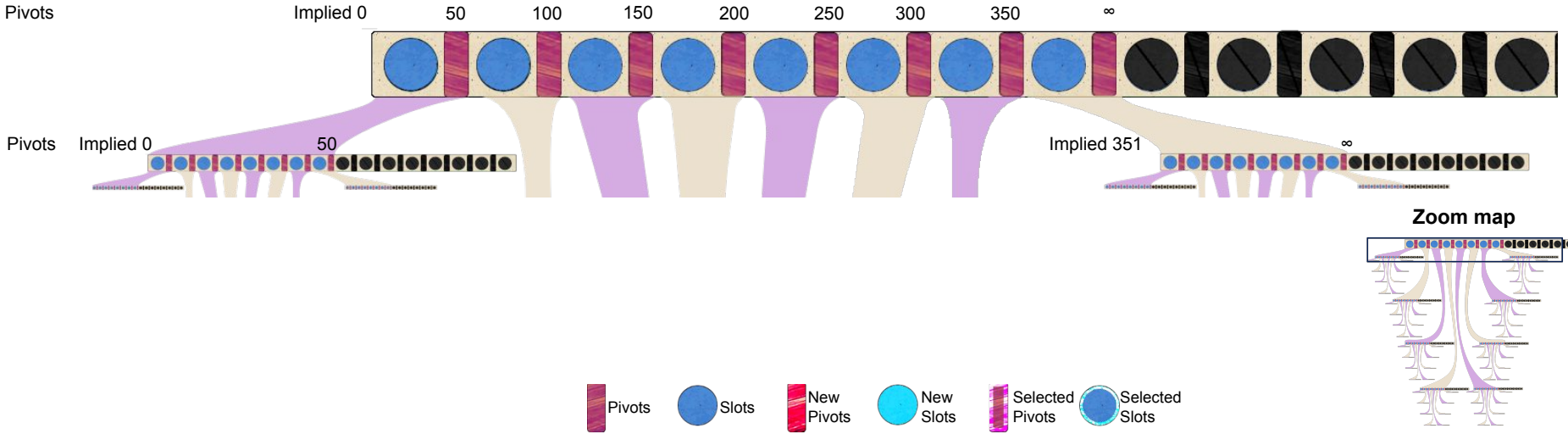


LF live MENTORSHIP SERIES

Maple Tree Write: Spanning Store

Store [125-260]

Zooming in on the larger tree for details



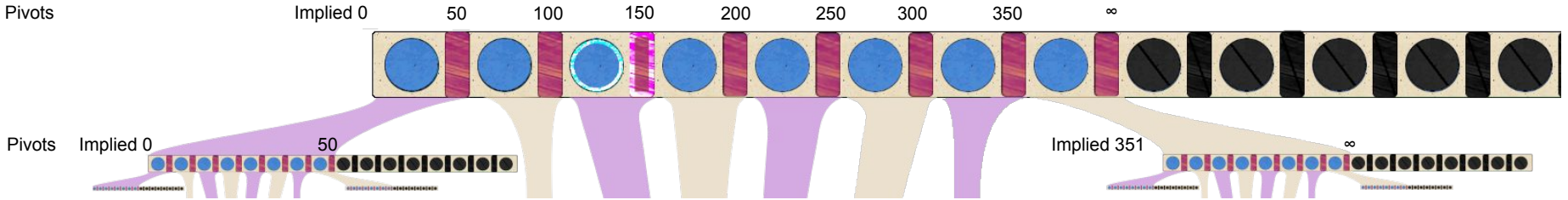
LF live MENTORSHIP SERIES

Maple Tree Write: Spanning Store

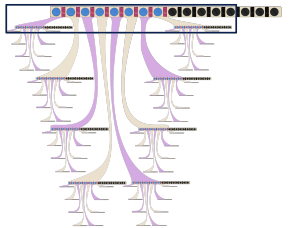
Store [125-260]

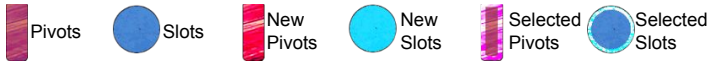
Walk to 125

Spanning store detected by the range spanning the slot



Zoom map



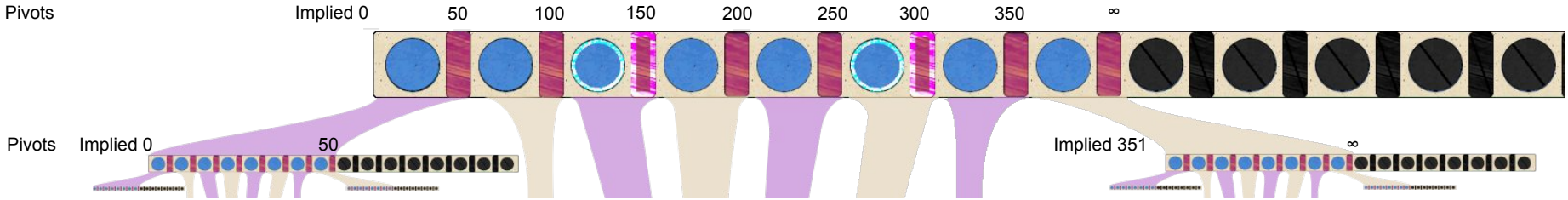


LF live MENTORSHIP SERIES

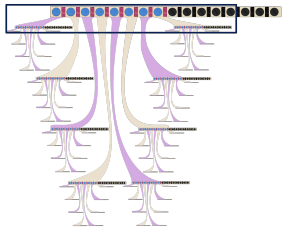
Maple Tree Write: Spanning Store

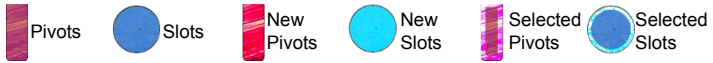
Store [125-260]

Walk to both 125 and 260



Zoom map

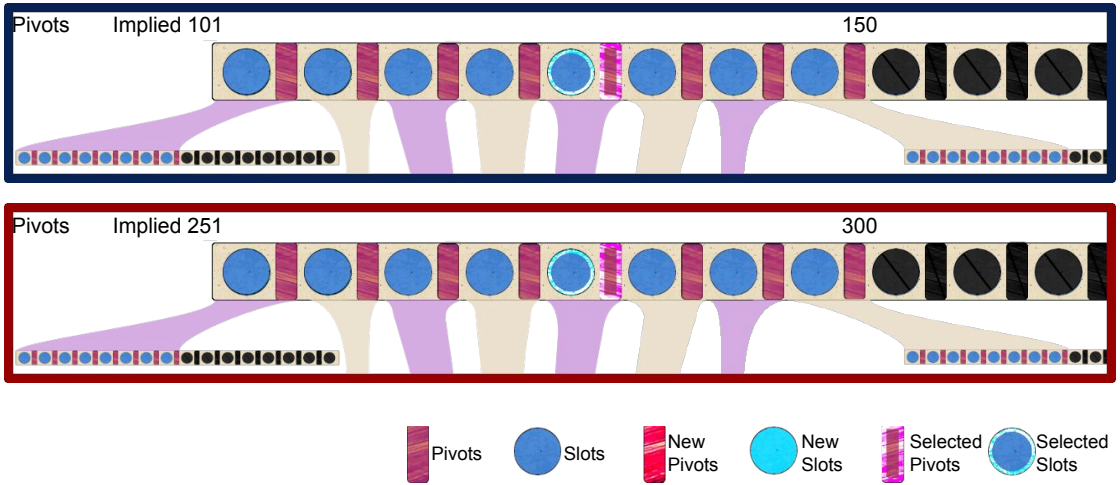




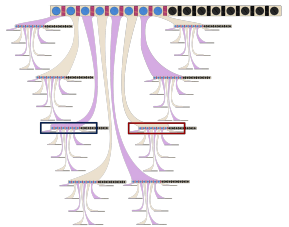
Maple Tree Write: Spanning Store

Store [125-260]

Continue to walk to 125 (in blue) and 260 (in red) down a level in the tree



Zoom map

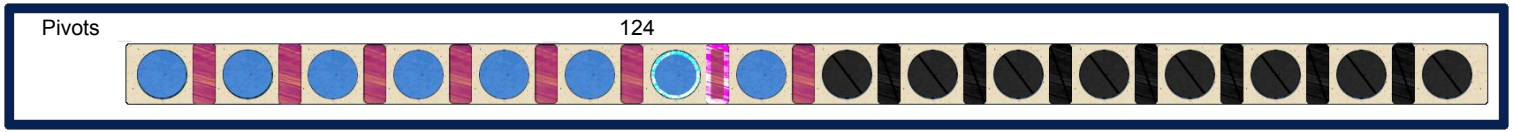


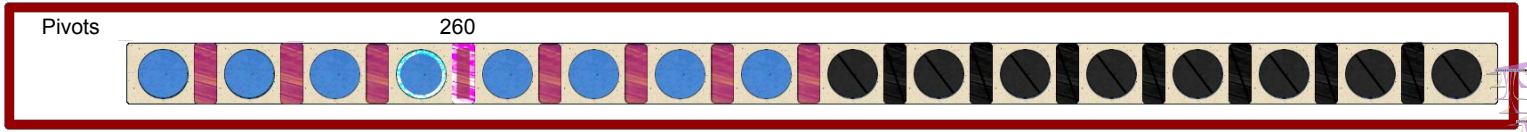
LF live MENTORSHIP SERIES

Maple Tree Write: Spanning Store

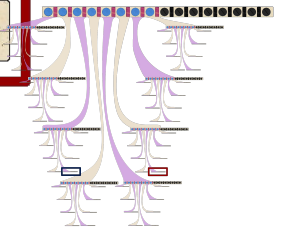
Store [125-260]

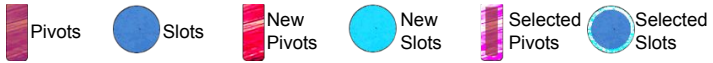
Walk the leaves to 125 (in blue) and 260 (in red) down another level





Zoom map



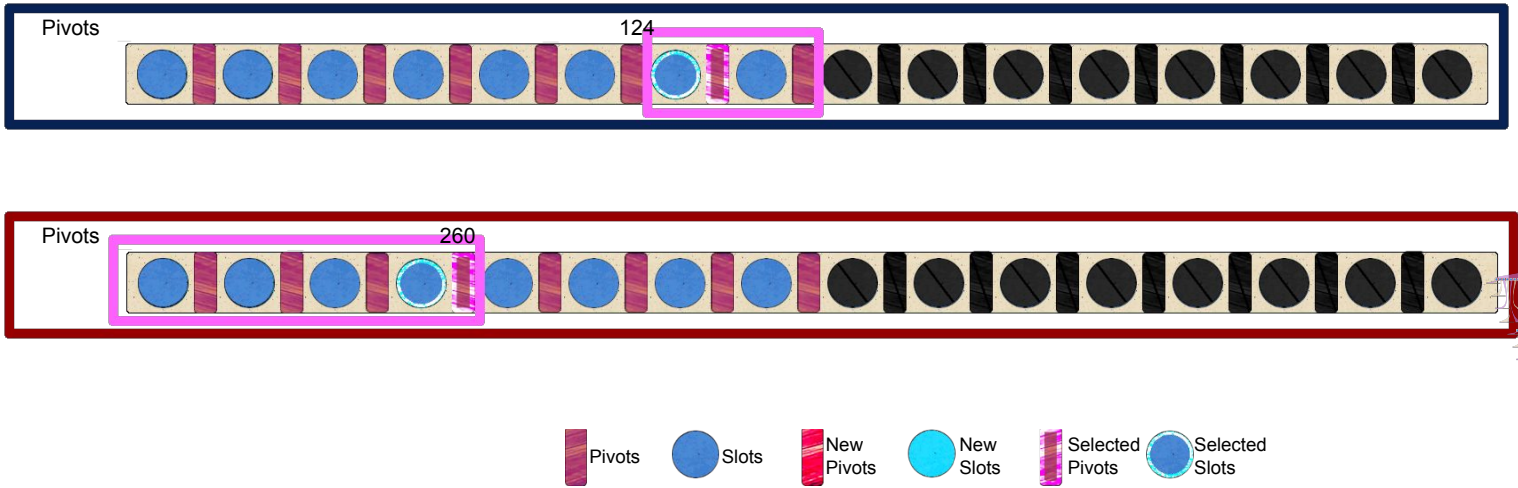


LF live MENTORSHIP SERIES

Maple Tree Write: Spanning Store

Store [125-260]

The pink highlighted data will be removed from the leaves during the operation

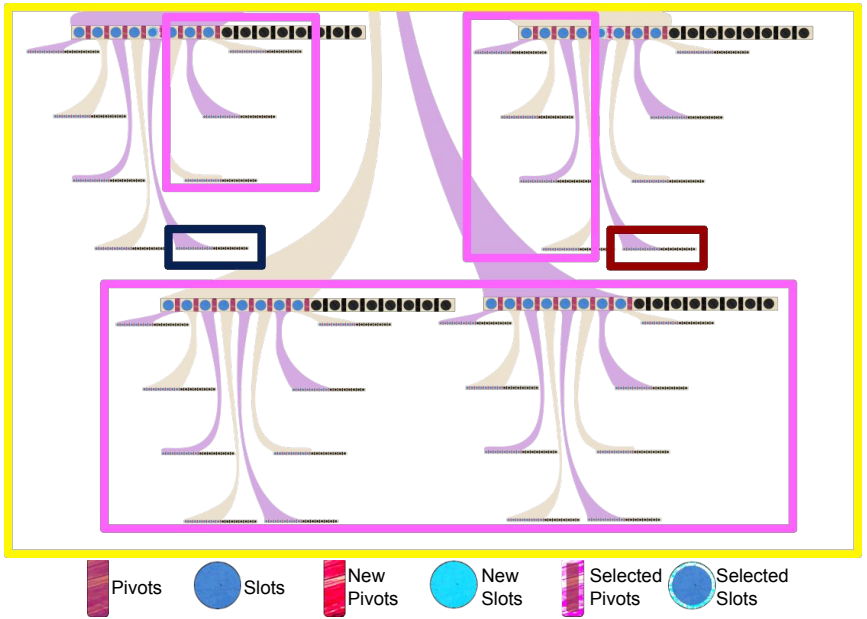


LF Live MENTORSHIP SERIES

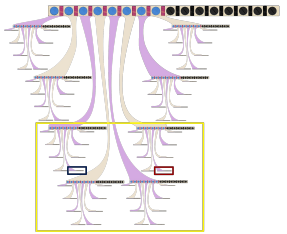
Maple Tree Write: Spanning Store

Store [125-260]

Subtrees highlighted in pink must also be removed



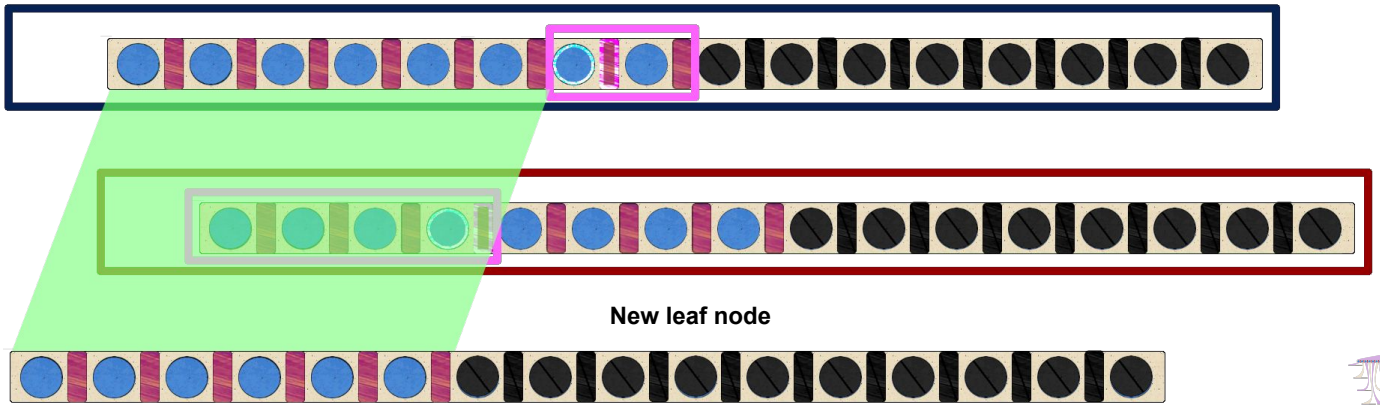
Zoom map



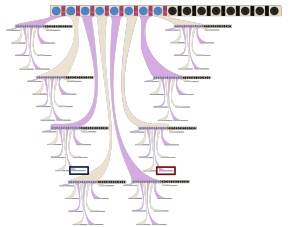
Maple Tree Write: Spanning Store

Store [125-260]

Copy left the node from 0 to the insert location into the new node



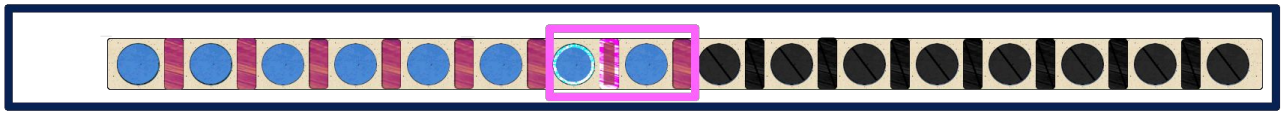
Zoom map

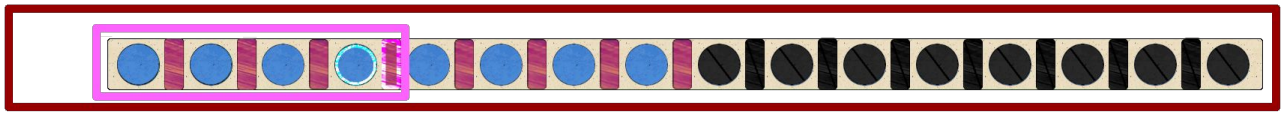


Maple Tree Write: Spanning Store

Store [125-260]

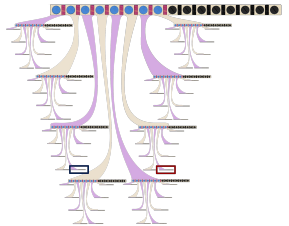
Store the new data in the new node

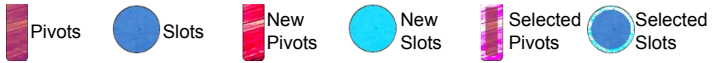




New leaf node

Zoom map

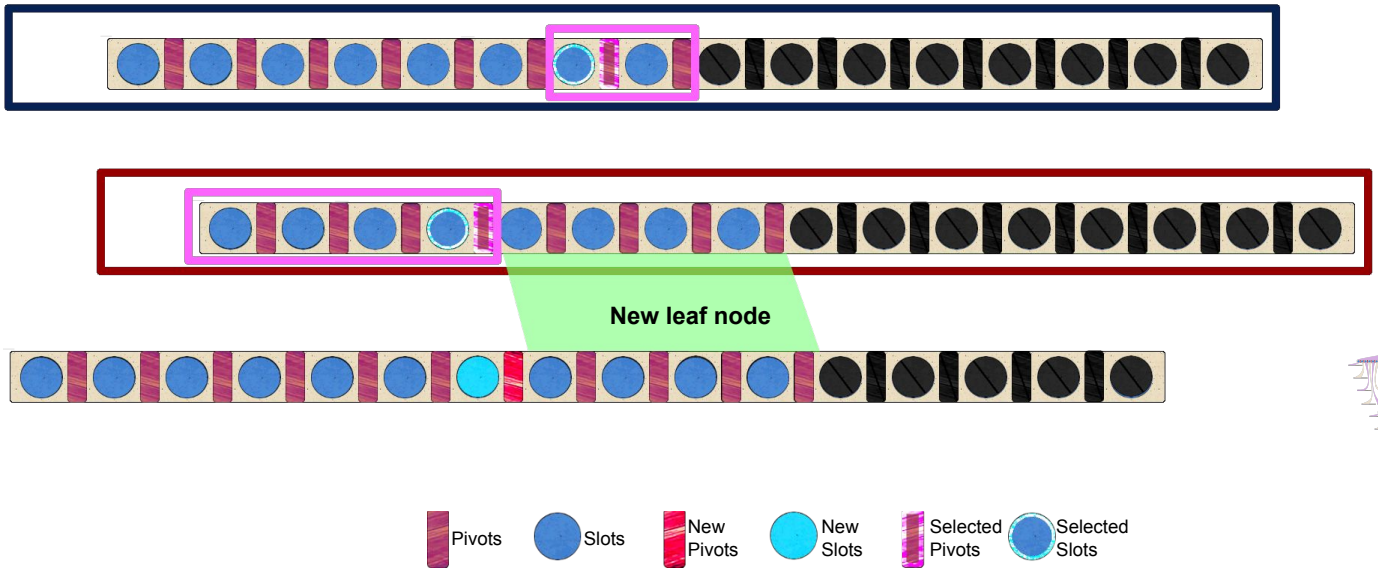




Maple Tree Write: Spanning Store

Store [125-260]

Copy the data from the right node from after the insert location to the end of the node

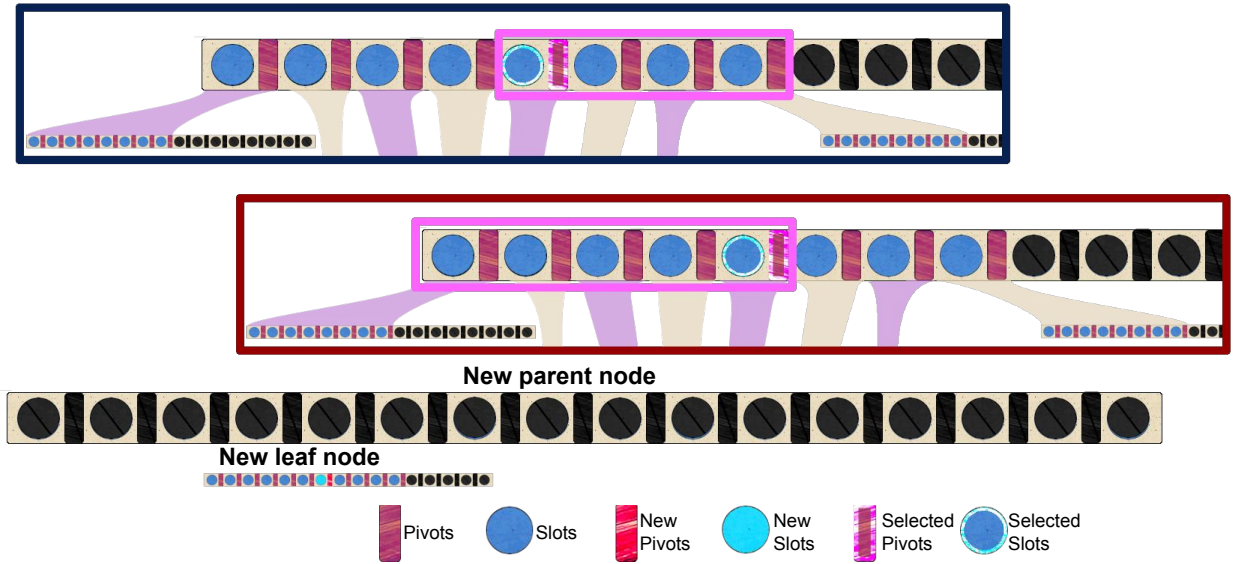


LF live MENTORSHIP SERIES

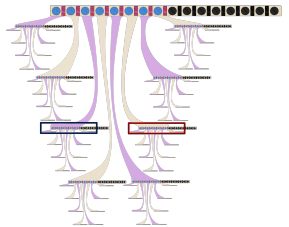
Maple Tree Write: Spanning Store

Store [125-260]

Build a new subtree; walk to left and right parents, copy data to new parent(s)



Zoom map

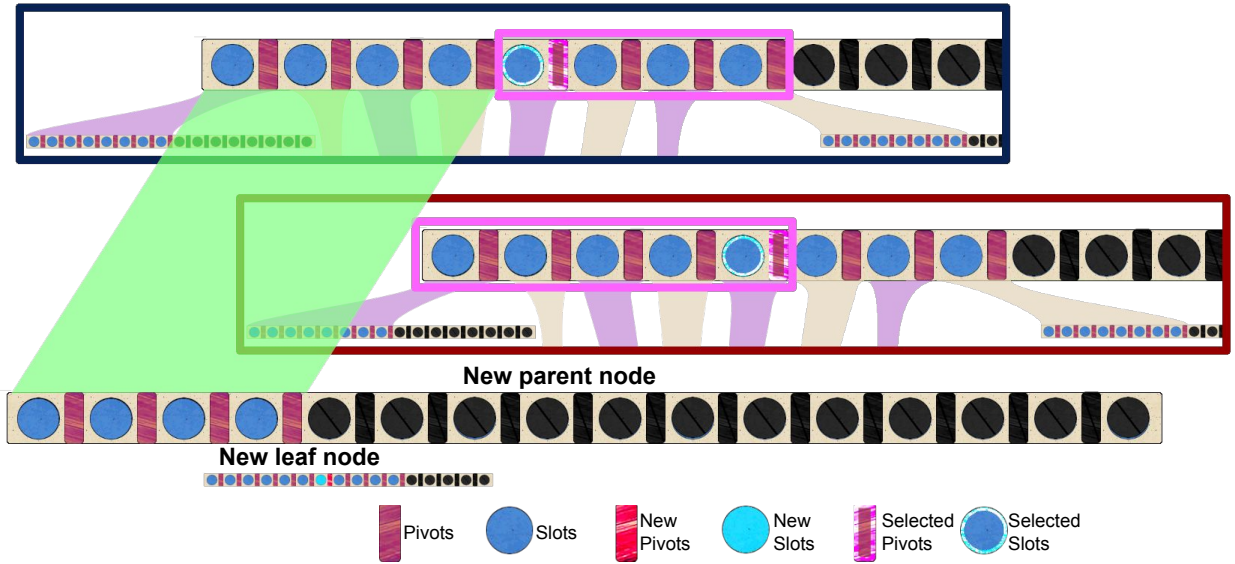


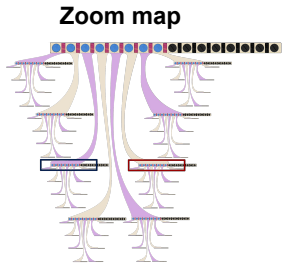
LF live MENTORSHIP SERIES

Maple Tree Write: Spanning Store

Store [125-260]

Copy the left parent data up to the insert location



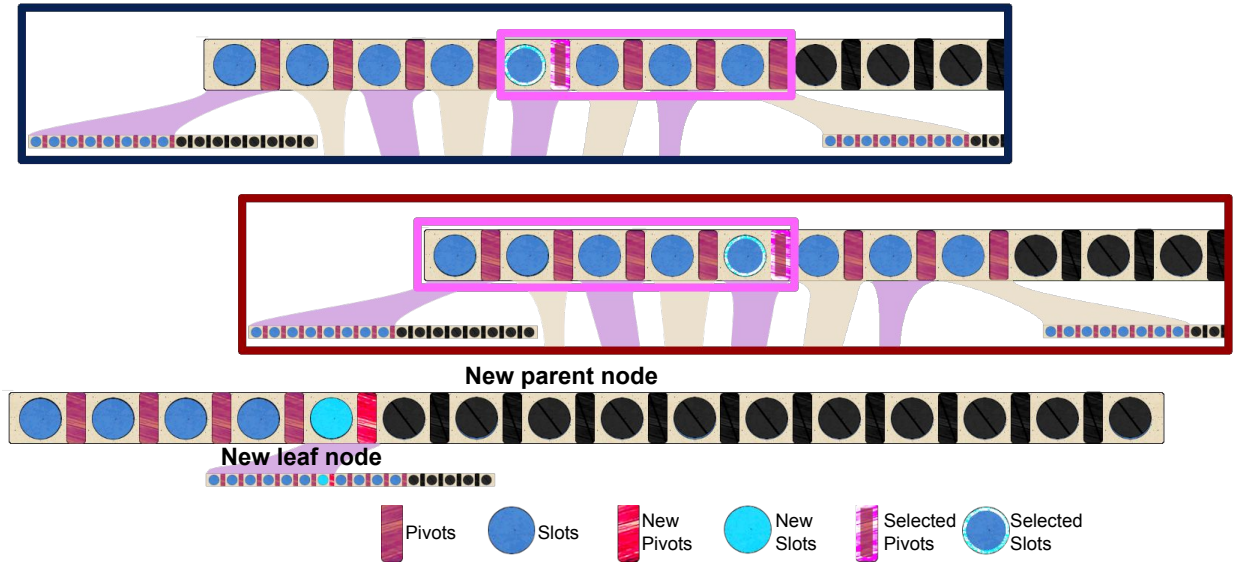


LF live MENTORSHIP SERIES

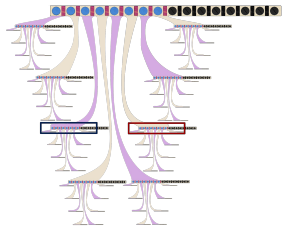
Maple Tree Write: Spanning Store

Store [125-260]

Insert new data into the new parent



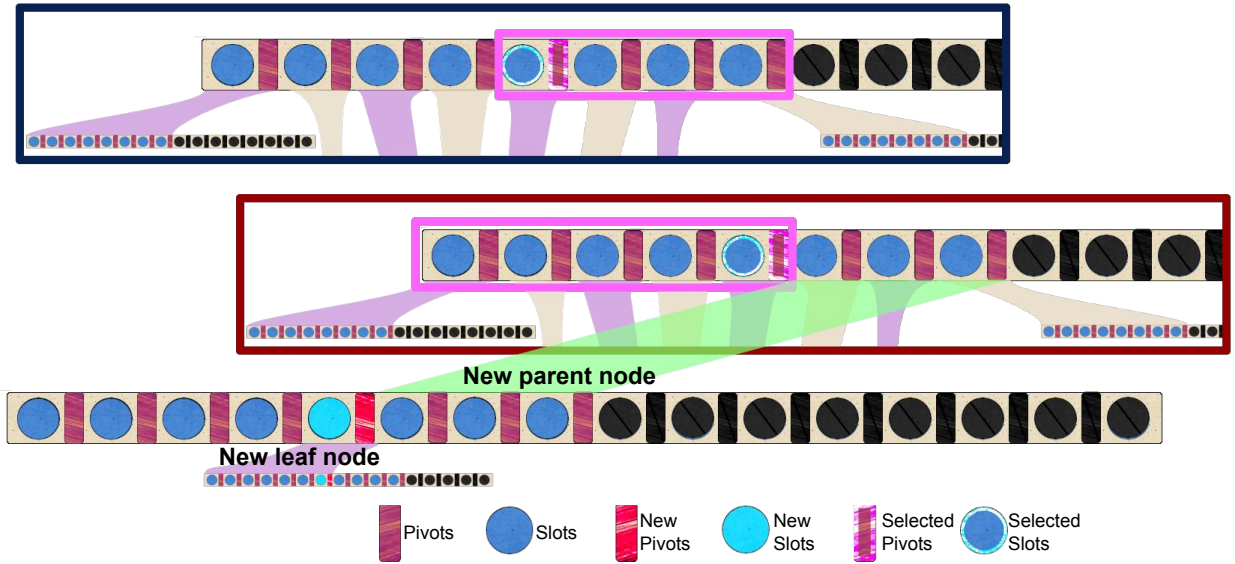
Zoom map

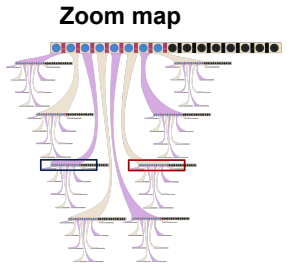


Maple Tree Write: Spanning Store

Store [125-260]

Copy the right parent data from the store location to the end of the node



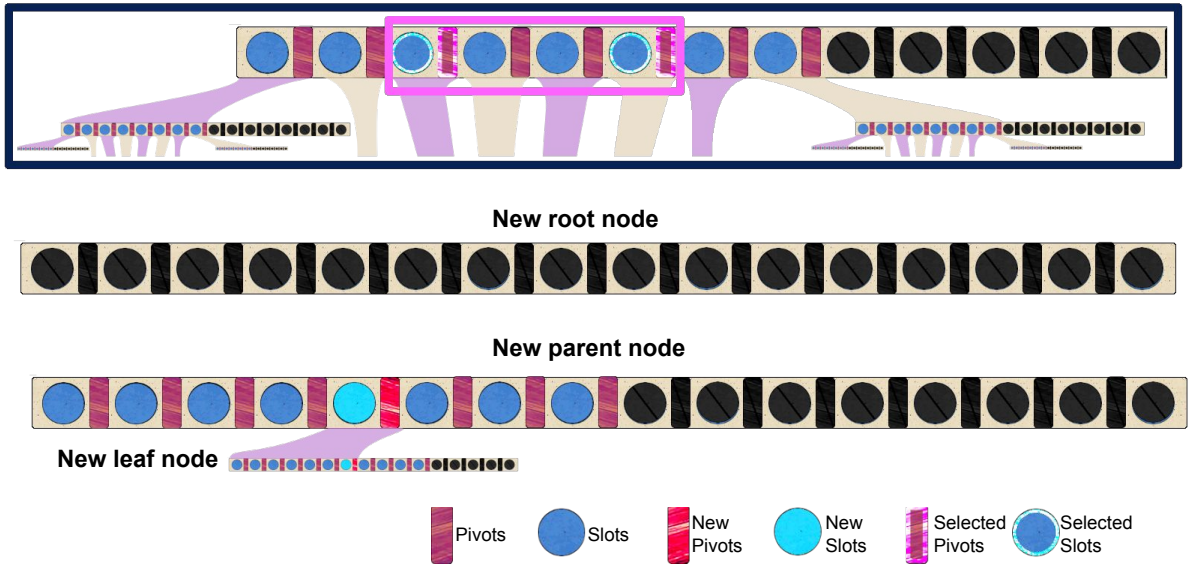


LF live MENTORSHIP SERIES

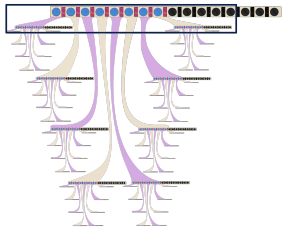
Maple Tree Write: Spanning Store

Store [125-260]

Create a new root node



Zoom map

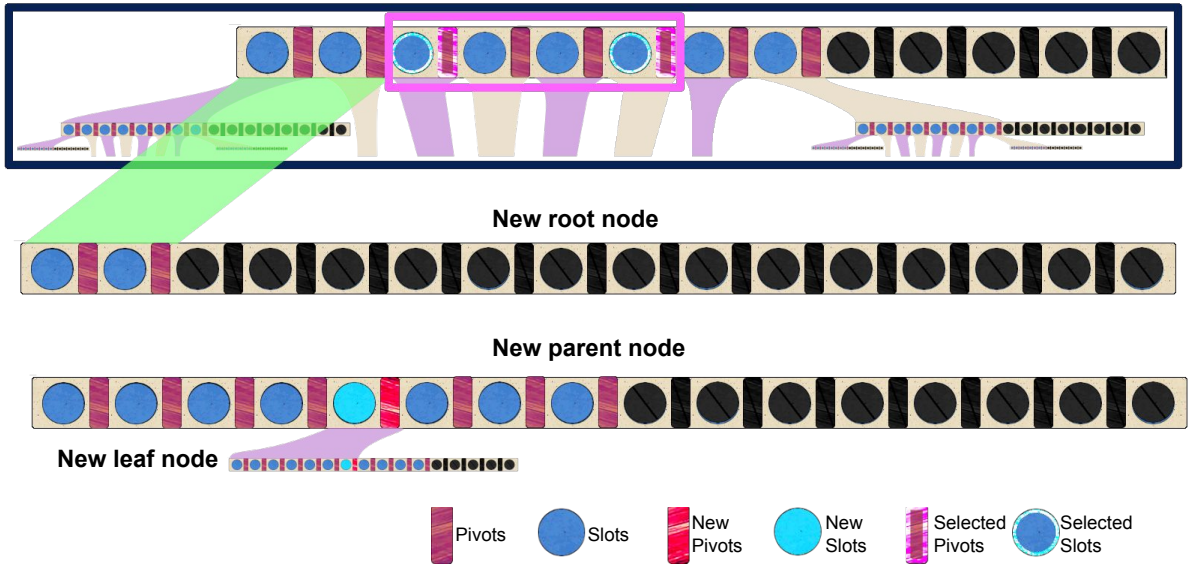


LF live MENTORSHIP SERIES

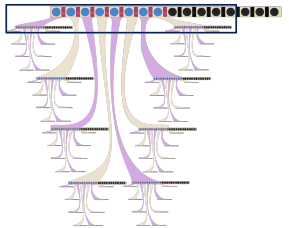
Maple Tree Write: Spanning Store

Store [125-260]

Copy the root data up to the insert location



Zoom map

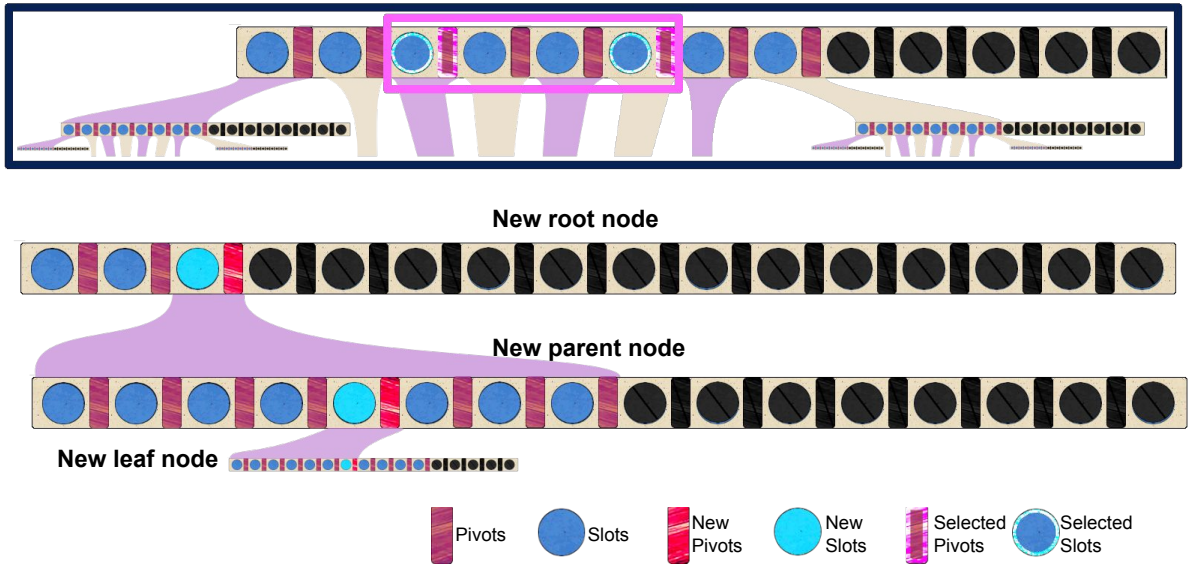


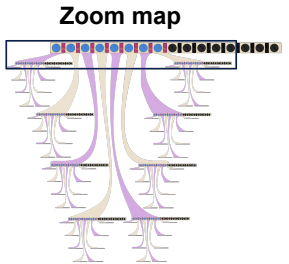
LF live MENTORSHIP SERIES

Maple Tree Write: Spanning Store

Store [125-260]

Insert the new parent into the new root

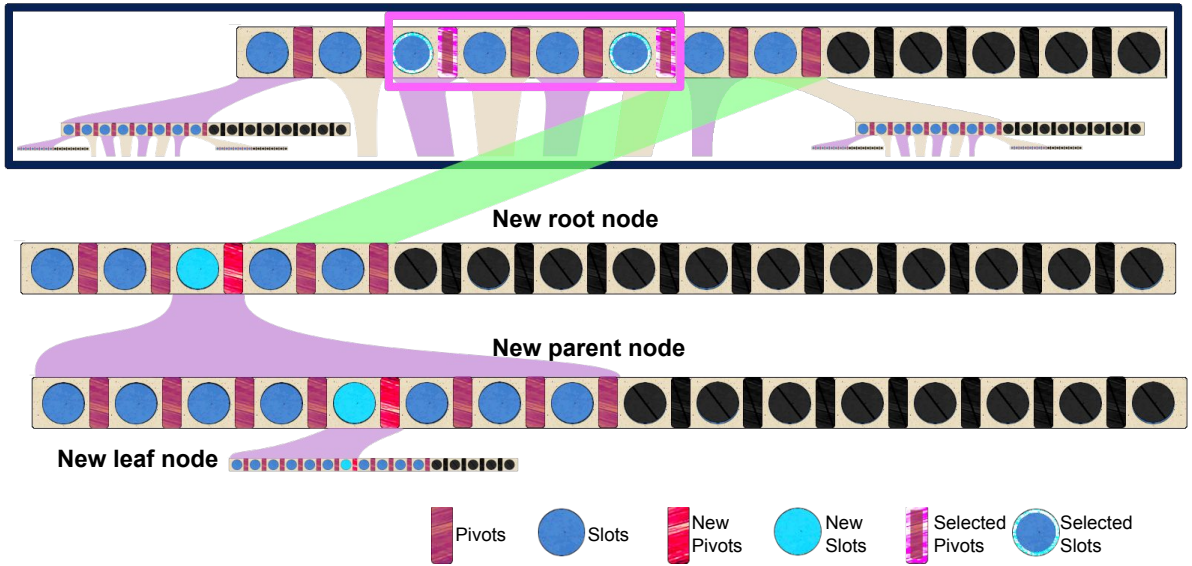




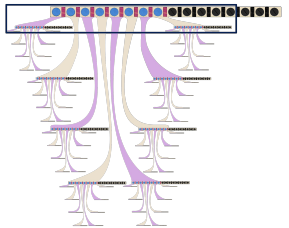
Maple Tree Write: Spanning Store

Store [125-260]

Copy the remaining data to the new root



Zoom map

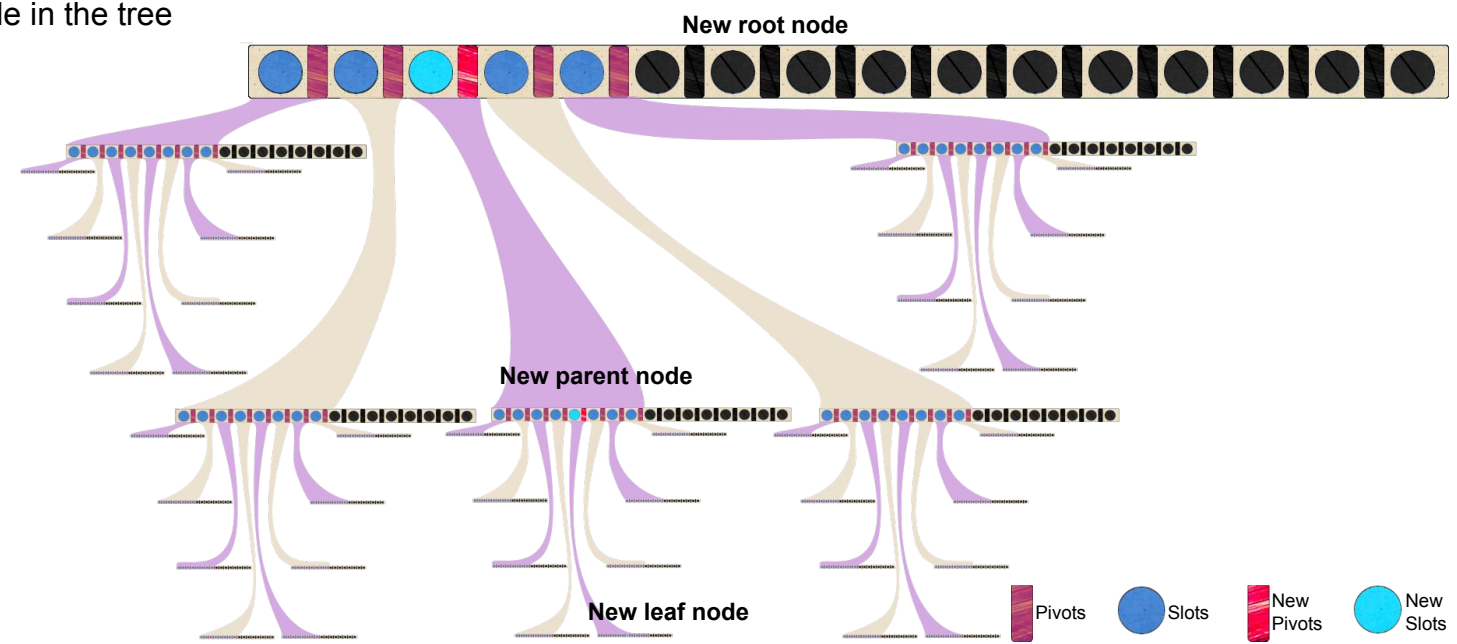


LF Live MENTORSHIP SERIES

Maple Tree Write: Spanning Store

Store [125-260]

Replace node in the tree



LF live MENTORSHIP SERIES

Thank you for joining us today!

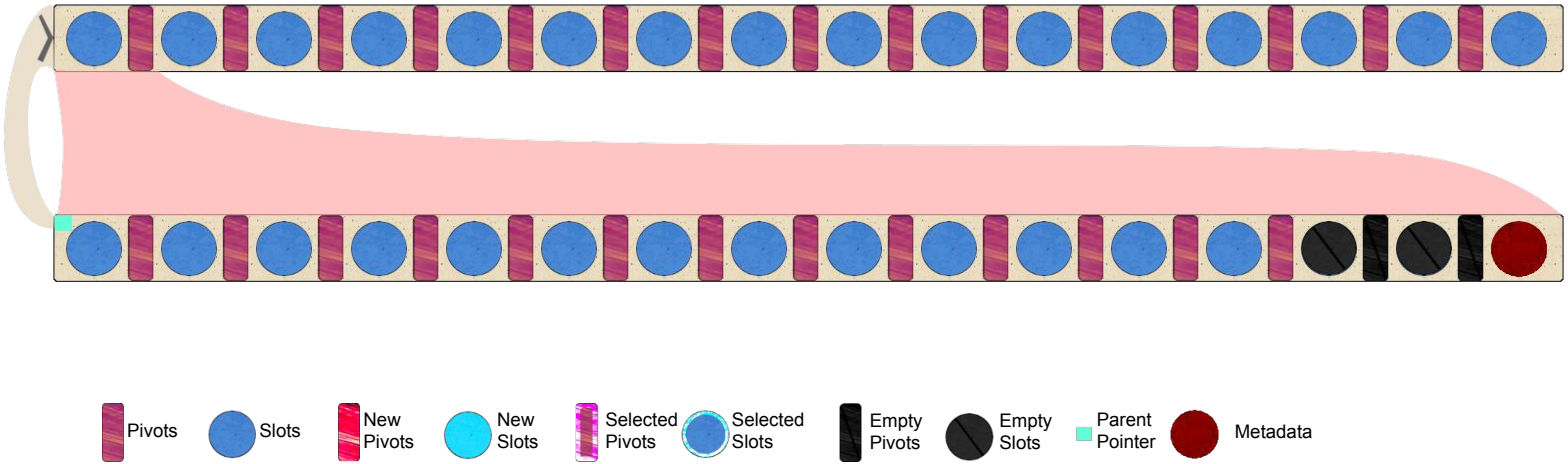
We hope it will be helpful in your journey to learning more about effective and productive participation in open source projects. We will leave you with a few additional resources for your continued learning:

- The LF Mentoring Program is designed to help new developers with necessary skills and resources to experiment, learn and contribute effectively to open source communities.
- Outreachy remote internships program supports diversity in open source and free software
- Linux Foundation Training offers a wide range of free courses, webinars, tutorials and publications to help you explore the open source technology landscape.
- Linux Foundation Events also provide educational content across a range of skill levels and topics, as well as the chance to meet others in the community, to collaborate, exchange ideas, expand job opportunities and more. You can find all events at events.linuxfoundation.org.

LF Live MENTORSHIP SERIES

Maple Tree: Enhancing the Node

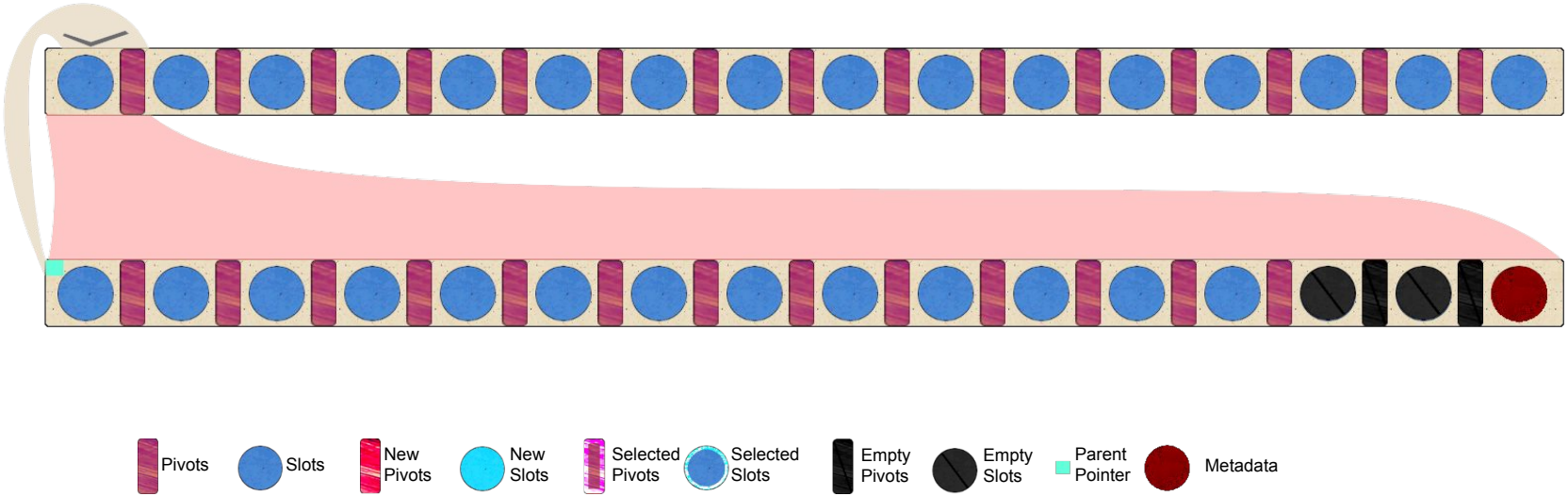
Nodes are actually more complex



LF Live MENTORSHIP SERIES

Maple Tree: Enhancing the Node

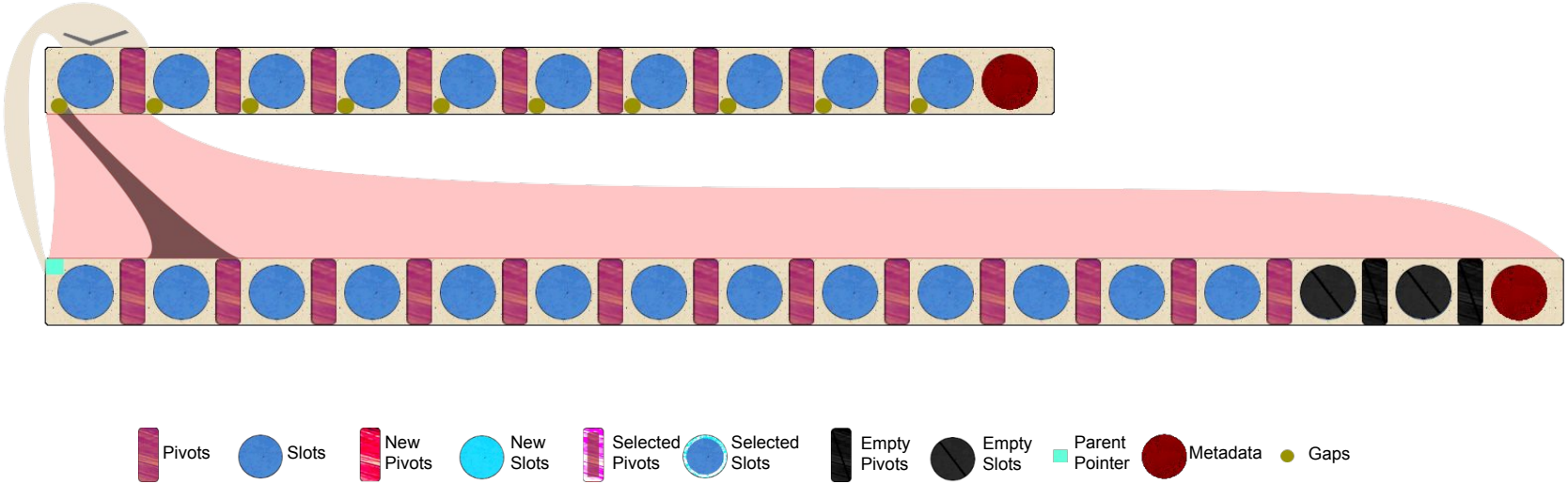
Parent pointers encodes the child's own slot in the parent



LF live MENTORSHIP SERIES

Maple Tree: Enhancing the Node

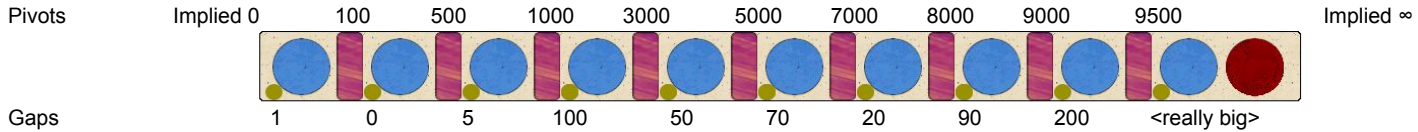
Internal nodes can track gaps



LF live MENTORSHIP SERIES

Maple Tree: Enhancing the Node

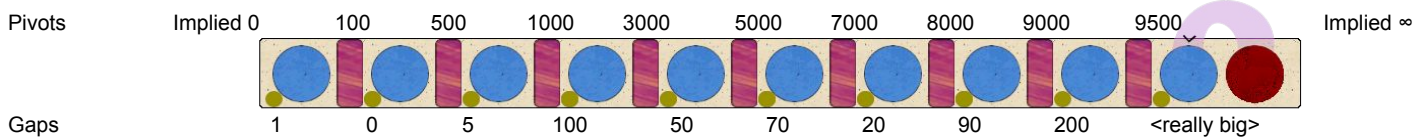
Topdown search for 30 within [500-7000]



LF Live MENTORSHIP SERIES

Maple Tree: Enhancing the Node

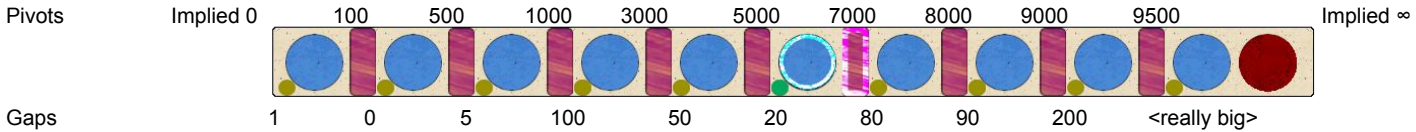
Topdown search for 30 within [500-7000]

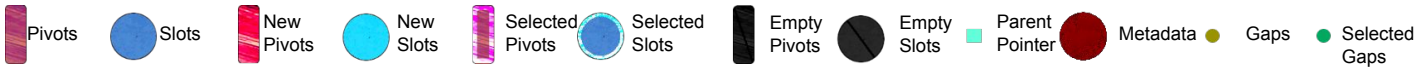


LF live MENTORSHIP SERIES

Maple Tree: Enhancing the Node

Topdown search for 30 within [500-7000]

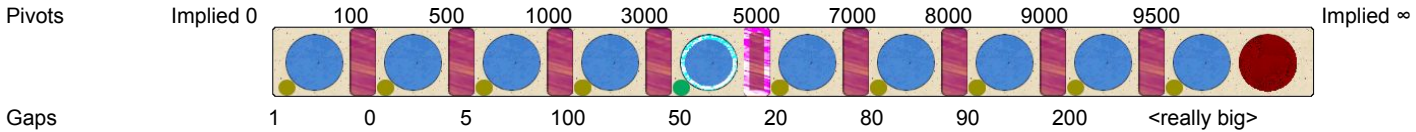


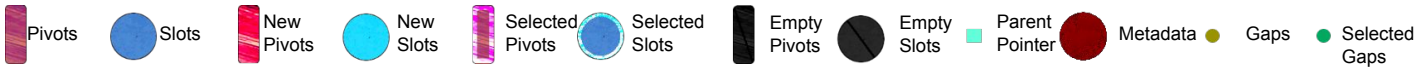


LF live MENTORSHIP SERIES

Maple Tree: Enhancing the Node

Topdown search for 30 within [500-7000]

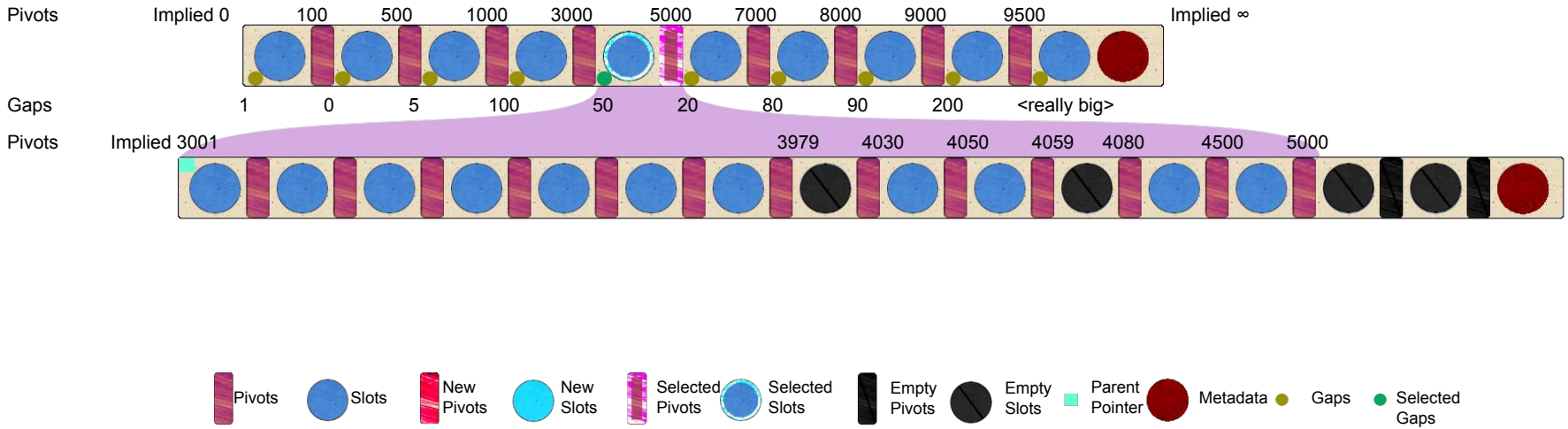




LF live MENTORSHIP SERIES

Maple Tree: Enhancing the Node

Topdown search for 30 within [500-7000]



Maple Tree Node Layout

Dense nodes

- No gap tracking, only leaves
- No pivots, just singletons with implied start and end
- Can contain multiple NULLs in a row
- Minimum data density will cause a rebalance to a range64 leaf

Maple Tree Node Layout

Node logical view is different from struct

- Struct is mostly arrays, not intermixed

Tree may be a single pointer for 0 -> entry

- Important optimisation for storing a single entry

Nodes are 256b aligned

- 7 bits for encoding information
- Parent pointers indicate parent type and slot where this node lives
- Child pointers indicate child type and if the node has any NULLs

Maple Tree Operations

Tree iterators:

- prev/next to emulate linked list operations
- prev_range/next_range to go to the next/prev slot even if it’s null
- mas_for_each(), mas_for_each_rev() to iterate across each entry

Maple Tree Future work

Add support for search marks

- New node type
- Replace gap tracking
- Reach parity of the xarray

Tree jitter reduction

- Avoid splitting and rebalance more aggressively based on NULL location

Write path of most functions

- Avoid rcu overhead; re-checking and smb_rmb()