The Maple Tree

Enhances the natural flavour of waffles

Liam R. Howlett
Matthew Wilcox

September 23rd, 2021
Talk Agenda

1. The Maple Tree Overview
2. Locking
3. The RCU Future
4. Performance
5. Other Potential Users
The Maple Tree Overview

**RCU-safe, range-optimised B-Tree variant**
- All leaves are at the same height
- Self-balancing
- Cache efficient
- Supports bulk loading

**256 Byte Node**
- 10 internal or 16 leaf slots
- Search by Index
- Find gap by size
Locking

Readers

• mmap_lock()

Writers

• mmap_lock()
• i_mmap_lock_write()
• anon_vma_lock_write()

Maple Tree Patches

• Adds mas_lock() for writes
• Adds rcu_read_lock() for reads
• Iterators may handle locking
The RCU Future

Readers
- rcu_read_lock()/rcu_read_unlock()
- VMA ref_count++/--

Writers
- Prepare for tree operations
- Mark VMA inactive
- mas_lock()/mas_unlock()
- Other locks need to be maintained, happen **before** mas_lock()

Forking
- mas_lock(), dup tree, mas_unlock()
- Iterate through VMAs, copy VMAs or delete them
Performance

Based on maple tree v2 patch set

Cache efficiency

- VMA size 200B → 152B
  - 20/page to 25/page
- MM struct size 148B to 132B

Real World Workloads

- Close to no change
- Kernbuild: user time ↓, system time ↑
  - Elapsed time worst case is +~2% or less than 1s on a 32.87s build

Trade off

- Updates can be more work, but sometimes less!
- Reads are less
Other Potential Users

**IDA/IDR**
- Dense nodes
  - Ranges of length 1 are inefficient right now
  - Encoded node types already supported

**Page cache**
- Search for marks as opposed to gaps
- Pruning of shadow entries
Thank you
ORACLE