Parallel E2fsck

Li Xi
Sept 2019
Background

- LU-8465
- 1 PB+ OST is coming
- On 1PB OST with 105M inodes, e2fsck time:
  - Pass 1: 3771
  - Pass 2: 98
  - Pass 3: 0.02
  - Pass 4: 12.94
  - Pass 5: 66.93
Need to Improve Pass1 Step

► Pass 1 takes 95% of the e2fsck time
► Why Pass 1 is slow
  • Walk through the entire inode table
  • On each inode
    o Read and check the inode attributes
    o Check the blocks used by each inode
    o A lot of inserting and searching of data structures
► How to improve
  • Fortunately, the check of each inode is almost independent
  • Different threads can check different inodes in parallel
Challenges & Solutions

► The result of Pass 1 will be used by Pass 2/3/4 too
  • Merge step is needed after threads finish

► Synchronization will be needed between threads in some cases
  • Bad blocks should be synced to avoid using them
  • Used blocks should be synced to avoid allocating them in multiple threads

► The threads of Pass 1 shouldn’t change disk at the same time
  • Lock need to be held to avoid any conflict of writing disk

► Correctness is very hard to confirm
  • Wrong e2fsck would cause/escalate data corruption
  • Need to pass all regression tests of e2fsprogs
  • Fortunately, there are already 186 regression tests
  • Strict review
Design

Start

Thread 0
Thread 1
Thread 2
Thread 3

Copy of contexts

Parallel scan of Pass1

Merge

Pass2
Pass3
Pass4

Merge of contexts
Steps towards Parallel E2fsck

► Step 1: Proof of concept: Done
  • Do not care whether the patch is clean or not
  • Get performance number to confirm the performance is improvable

► Step 2: Multiple threads run sequentially: Working on
  • Merge the pass1 results from multiple threads properly
  • All regression tests need to be passed no matter how many threads
  • Pass the tests then thread number is 1, 2, 3, ... n

► Step 3: Multiple threads run in parallel: Future
  • Threads need to sync with each other from time to time
  • Tests might not be able to be passed any more
  • Any way to pass the tests

► Step 4: Review, test and merge: Future
  • Need strict review to make sure nothing breaks
  • Codes need to be rewritten for better quality

Harder and harder
Sequential run of threads for regression tests

Output and result should be exactly the same with original e2fsck
Current status

- 40+ patches, a lot more is coming
- Speedup for more than X4 times, from 3771 seconds to 800 seconds
- More speedup is possible with better load balancing and more threads
- Bigalloc feature might help a lot too
- "libext2fs: optimize ext2fs_convert_subcluster_bitmap()" patch improves E2fsck speed a lot
- All tests can be passed with single thread, except occasional crash because of
Thought & Concerns

► E2fsck codes really need to be cleaned up
  • A lot of similar codes that could be put into shared library, e.g. binary search
  • Cleanup is hard because things can be easily broken

► E2fsck correctness is tooooo critical
  • Review of the patches needs to be really careful
  • Not able to reuse the regression tests for parallel fsck

► Any more ways to test the correctness?
  • Regression tests that already exists
  • Valgrind command to detect memory leak
  • E2fsck on huge Ext4 with hundreds of millions inodes to confirm no performance regression.
New ideas

► The parallel fsck can be only used for check
  • If any problem is found, restart to use single threads check

► Several choice to fix problem
  • Thread 0 fix all the found problems
  • Fix the problem at the thread that found it
  • Fix the problem after all threads join