metadata: directories, inodes, free block maps.

improving the write-back performance by combining multiple updates into one disk write.

* multiple updates to a single metadata component
* multiple updates to a single block of metadata

`tip`

in memory, there are not usually pointers, which may cause inconsistency.

in persistent memory, since the content stores, need to consider inconsistency for it.

If just synchronization the writes, there will be lots of disk I/Os.

`tip`

`tip`

solution: roll-back to a safe, intermediate state; and then the dependency is broken, can use two times of I/Os + 1 time of I/O. rather than 4 times of I/Os.

If u are not understand, can go back to the original slide to check the example.

`tip`

## metadata update problem and previous solution

update dependency problem

three basic rules:

* never points to a structure before it has been initialized
* never reuse a resource before nullifying all previous pointers
* never reset the last pointer to a live resource before the new pointer has been set.

Previous solutions:

* synchronous writes: bad performance
* NVRAM: hardware requirement
* Atomic updates: require write-ahead logging, that is requirement to the disk structure.
* scheduler-enforced ordering: cannot satisfy the request sequencing.

## the detailed solutions to soft updates

dependency information: per pointer or per field

promised dependencies:

* for block allocation. The contents of blocks should be written first before the pointer to the block(inode) is written.
* for block deallocation. The contents of the block should not be reused before the pointers to the blocks are cleared.
* For link addition. The inodes should be written before the differectory pointers be written.
* For link removal: The iode pointer should be written before the inode link count be decreased.

The possible file system inconsistencies:

* unused blocks not in the free space maps
* unused inodes not in the free inode maps
* inode link count exceed the actural number of directory entries.

rsck utility to clear these above inconsistencies.

## performance

## conclusion