SQLskills
PASS Data Architecture Virtual UG
August 3rd 2015

Index Fragmentation: Internals, Analysis, and Solutions

Paul S. Randal
Paul@SQLskills.com

SQLskills
immerse yourself in sql server

- **Team of world-renowned SQL Server experts:**
  - Paul S. Randal (@PaulRandal)
  - Kimberly L. Tripp (@KimberlyLTripp)
  - Glenn Berry (@GlennAlanBerry)
  - Erin Stellato (@ErinStellato)
  - Jonathan Kehayias (@SQLPoolBoy)
  - Tim Radney (@TRadney)
- **Instructor-led training: Immersion Events (US, UK, and Australia)**
- **Online training:** pluralsight http://pluralsight.com/
- **Consulting: health checks, hardware, performance, upgrades**
- **Remote DBA: system monitoring and troubleshooting**
- **Conferences: PASS Summit, SQLintersection**
- **Become a SQLskills Insider**
  - https://www.sqlskills.com/Insider

SQL intersection

2015 Immersion Events

- **Classes in Chicago, Bellevue (WA), London, Dublin, Sydney (Australia)**
  - IE0: Immersion Event for Junior/Accidental DBA
  - IEPTO1/2: Immersion Events on Performance Tuning – Parts 1 and 2
  - IEHADR: Immersion Event on High Availability and Disaster Recovery
  - IEBI: Immersion Event on Business Intelligence
- **In-depth, instructor-led, technical training for SQL Server**
- **Here's our topic list for IEPTO2: Immersion Event on Performance Tuning, Part 2**

SQL Server I/O	I/O Concepts for DBAs	SANs for DBAs
SQLOS Scheduling and CPU Performance Tuning	SQLOS Memory Management and Memory Performance Tuning	Data Collection and Baselining
Wait and Latch Statistics	Query Plan Analysis	Extended Events
Performance Issue Patterns	Statement Execution, Stored Procedures, and the Plan Cache	
Deadlock Analysis	Advanced Extended Events	

- **For more information: https://www.sqlskills.com/training/**

Author/Instructor: Paul S. Randal

- Consultant/Trainer/Speaker/Author
- CEO and Owner, SQLskills.com
  - Email: Paul@SQLskills.com
  - Blog: https://www.SQLskills.com/blogs/Paul
  - Twitter: @PaulRandal
- Contributing Editor of TechNet Magazine, author of the SQL Q&A column, articles on DBA topics, multiple whitepapers for Microsoft
- 5 years at DEC responsible for the VMS file-system and chkdsk equivalent
- Almost 9 years as developer/manager in the SQL Storage Engine team through August 2007, ultimately responsible for Core Storage Engine
  - Wrote DBCC component, other Storage Engine code, DBCC CHECKDB/repair for SQL 2005
- Regular presenter at worldwide conferences on HA/DR, administration, performance, and internals (#1 rated session and workshop at PASS Summit in 2013)
- Course author/instructor for Microsoft Certified Master qualifications
- Owner and Manager of the SQLintersection conference
- (I also like diving, electronics, robotics, Arduino, books, sheep…)

SQL intersection

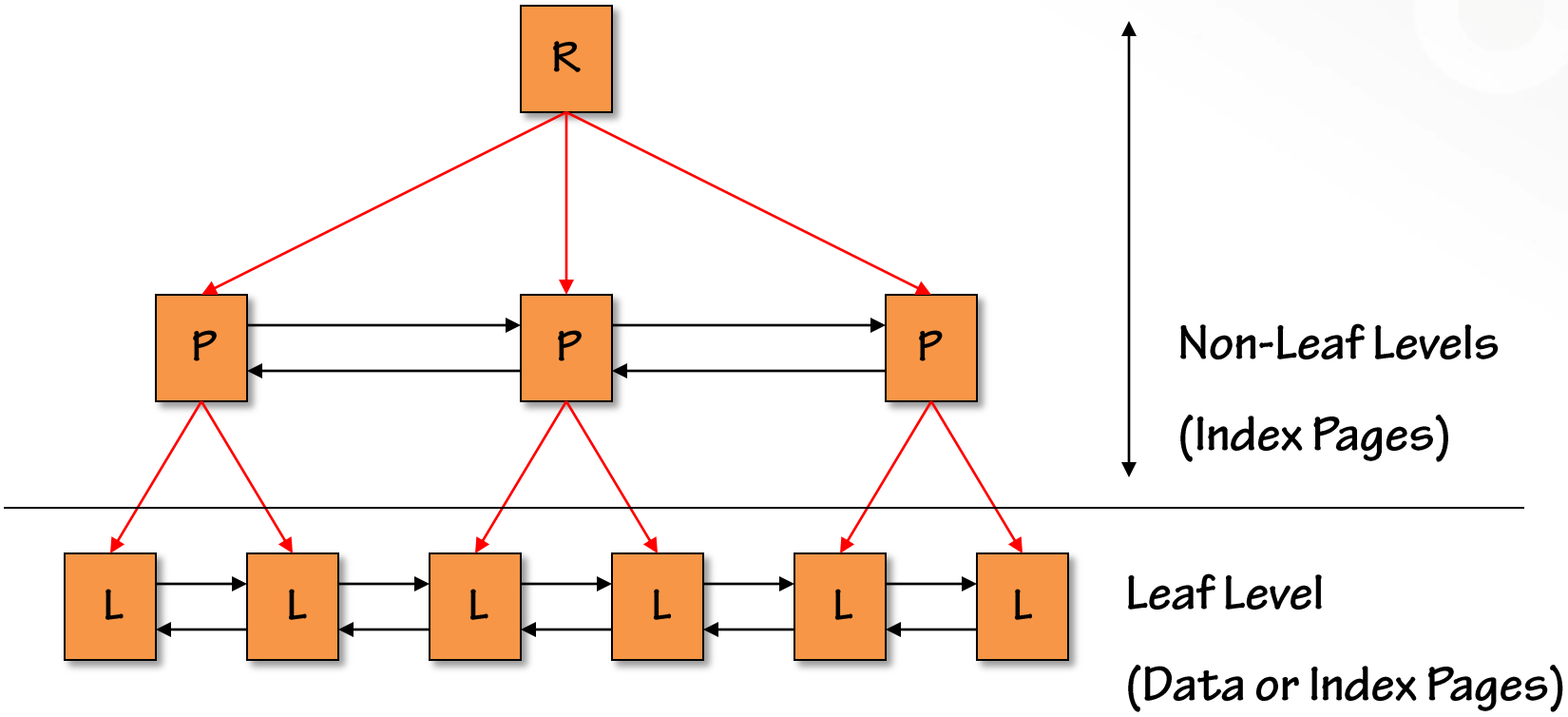
- **Email paul@SQLskills.com with the subject line: User Group Pluralsight code to get a FREE (no catches, no credit card) 30-day trial of our 120+ hours of SQLskills content on Pluralsight**
- **For example:**
  - http://www.pluralsight.com/courses/sqlserver-logging
    - 7 hours on logging, recovery, and the transaction log (Paul)
  - http://www.pluralsight.com/training/Courses/TableOfContents/sqlserver-waits
    - 4.5 hours on waits, latches, spinlocks (Paul)
  - http://www.pluralsight.com/courses/sqlserver-optimizing-stored-procedure-performance
    - 7 hours on stored procedure performance tuning (Kimberly)

Overview

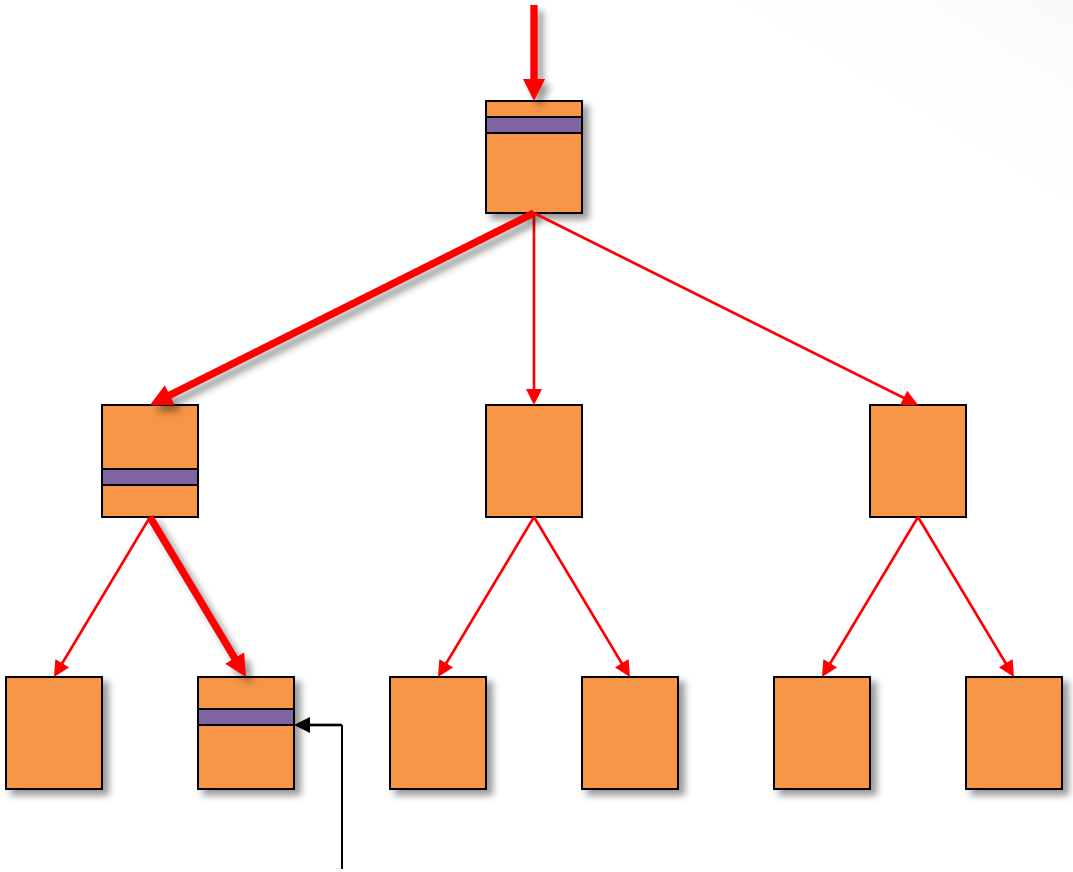
- **Index fragmentation is inescapable when you have any indexes where the index key does not match the insertion pattern**
- **Fragmentation itself is not the only problem, as we'll see**

- **We're going to cover:**
  - Data access methods
  - What is fragmentation?
  - How does fragmentation happen?
  - Detecting, mitigating, removing fragmentation

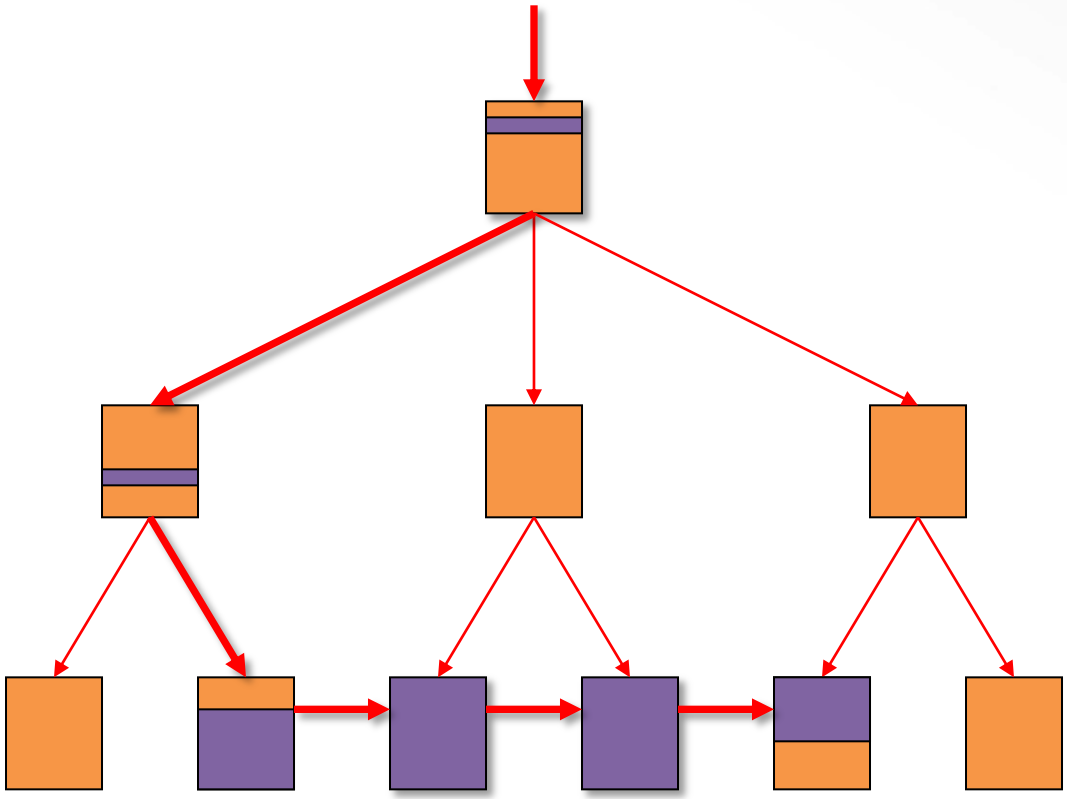
Index Structure



Singleton Lookup

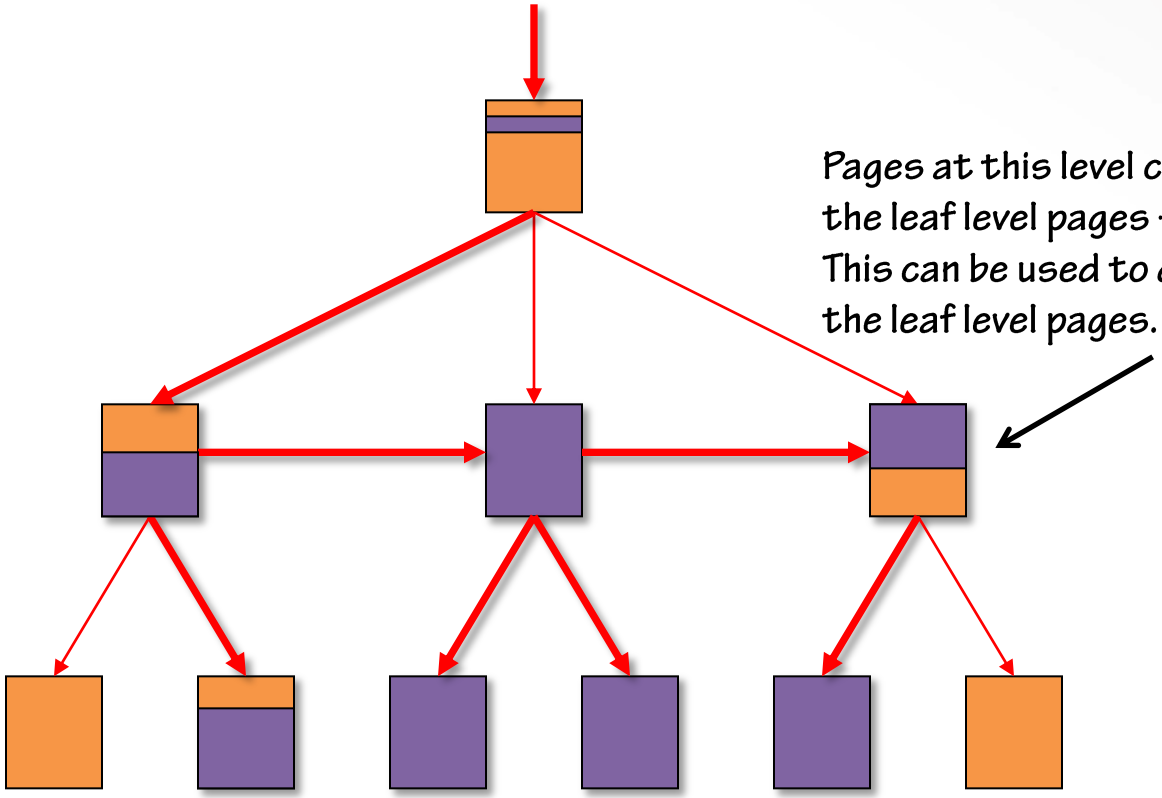


Range Scan



Matching records (in purple)

Range-Scan: Readahead (1)



Range-Scan: Readahead (2)

- **Why use readahead?**
  - Keep the CPUs busy, maximize throughput, avoid I/O waits
  - More efficient to issue 1 x 8-page I/O than 8 x 1-page I/Os
- **Feedback mechanism to determine how far ahead of the scan point to read**
- **Driven from parent level during range scans**
  - Parent level pages contain logically-ordered links to the leaf level
- **Issues 1, 8, 32, 64-page I/Os, and up to 8 parallel I/Os (i.e. 512 pages)**
  - 8, 32+ page I/Os only possible with contiguous pages
  - Better contiguity = bigger I/Os
- **Problem: fragmentation affects range scans**

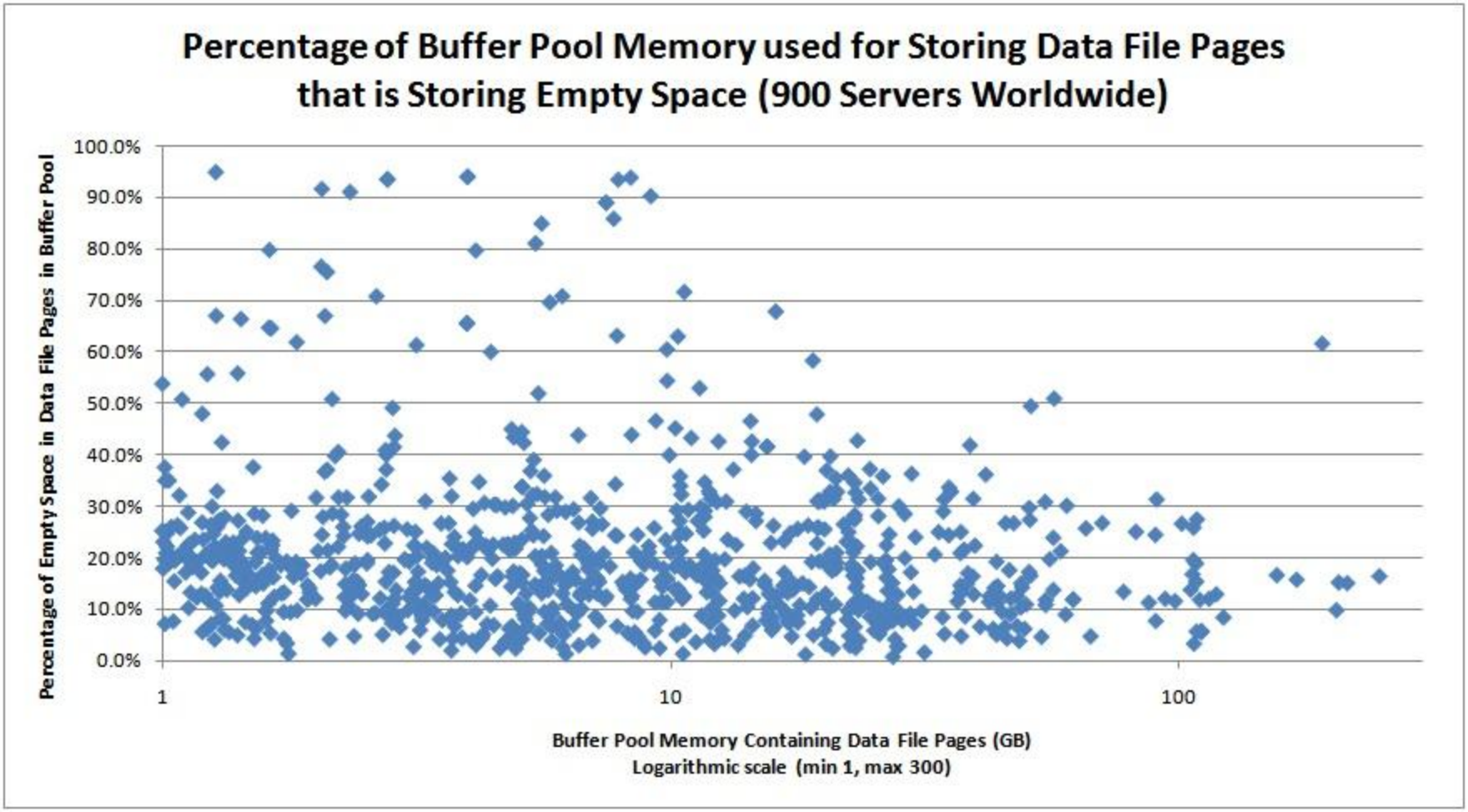
Logical Fragmentation Defined

- **(Sometimes called "external" fragmentation)**
- **Occurs when the next logical page is not the next physical page**
- **Prevents optimal readahead**
  - Reduces range scan performance
- **Does not affect pages that are already in cache**
  - Smaller indexes affected less (e.g. 1-5000 pages or less)
- **Reported as avg_fragmentation_in_percent in the sys.dm_db_index_physical_stats DMV in SQL 2005+**

Page Density Defined

- **(Sometimes called "physical" or "internal" fragmentation)**
- **Page fullness is below the optimal level so wasted space**
- **Effect is:**
  - Increased disk space (more pages required to hold the same number of rows)
  - Increased I/Os to read the same amount of data
  - Greater memory usage if most of the index is memory resident
- **Reported as avg_page_space_used_in_percent in the DMV**

Low Buffer Pool Usage

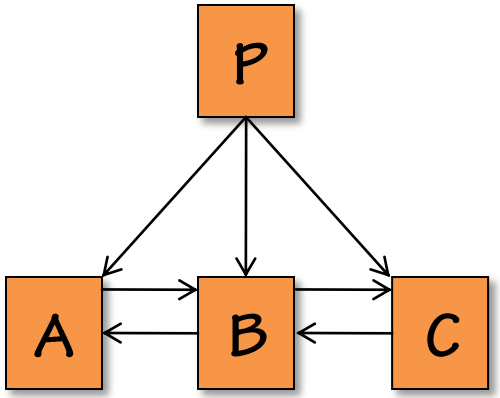


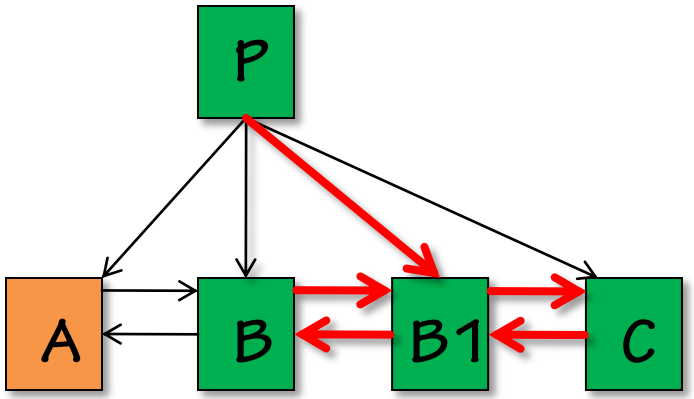
Source: my blog at http://bit.ly/10qs55H

What is a Page Split?

- **Occurs when a record must be inserted onto a specific page in the index (determined by index key value) and there is not enough space**
  - Could be caused by a new record or an updated record that is now longer than it was before
  - Could also be caused by enabling snapshot isolation, which makes updated records 14-bytes longer
- **The page has to 'split' to make room**
  - Split point is usually as close to 50/50 as possible, but may be skewed if Storage Engine can determine an obvious split point)

Updates During a Page Split





- Page B splits into B and B1
- New page B1 is unlikely to be physically adjacent to B
- Pages P, B, C must be updated to change/create page links
  - Page P might split…
- All changes are fully logged

Tracking Page Splits

- **There are 'good' and 'nasty' page splits…**
  - 'Good' split is when a page is allocated as part of an append-only insert pattern
  - 'Nasty' split is when a real page split occurs
- **Unfortunately, all documented methods of tracking page splits prior to SQL Server 2012 do not differentiate between 'good' and 'nasty' page splits**
  - Perfmon counter
  - Sys.dm_db_index_operational_stats
  - Extended event (possibly with post-processing)
- **Either use log/log backup scanning or 2012 Extended Events**
  - See my blog post at http://bit.ly/UG1SyG

Demo

Increased logging during page splits

What Causes Fragmentation?

- **Schemas/workloads that cause page splits on full pages**
  - GUID as high-order key (or any other random key)
    - Can even affect nonclustered indexes
  - Updates to variable-length columns
  - Badly configured FILLFACTOR
- **Clustered index is likely the only one you can make the key not cause fragmentation by picking an ascending order key (e.g. bigint identity)**
- **Wide schemas that only fit a few records per page**
  - E.g. a fixed-size 5000 byte row = 3000 bytes lost per page!
- **Even deletes can cause fragmentation by lowering page density**

FILLFACTOR

- **Makes the Storage Engine leave space on each leaf-level page to allow row inserts/expansions without causing page splits**
- **Specified at index creation or rebuild time or using Object Explorer**
  - NOT maintained during regular DML
- **Can specify with sp_configure for entire instance**
  - Not recommended – specify it per index
- **Use PAD_INDEX to affect the upper levels of the b-tree (uses the FILLFACTOR value)**
  - Rarely used
- **0 = 100 = default value with special meaning of 'leave no space'**
  - Excellent for data warehouse, but not ideal for OLTP
- **For OLTP, which value to use?**

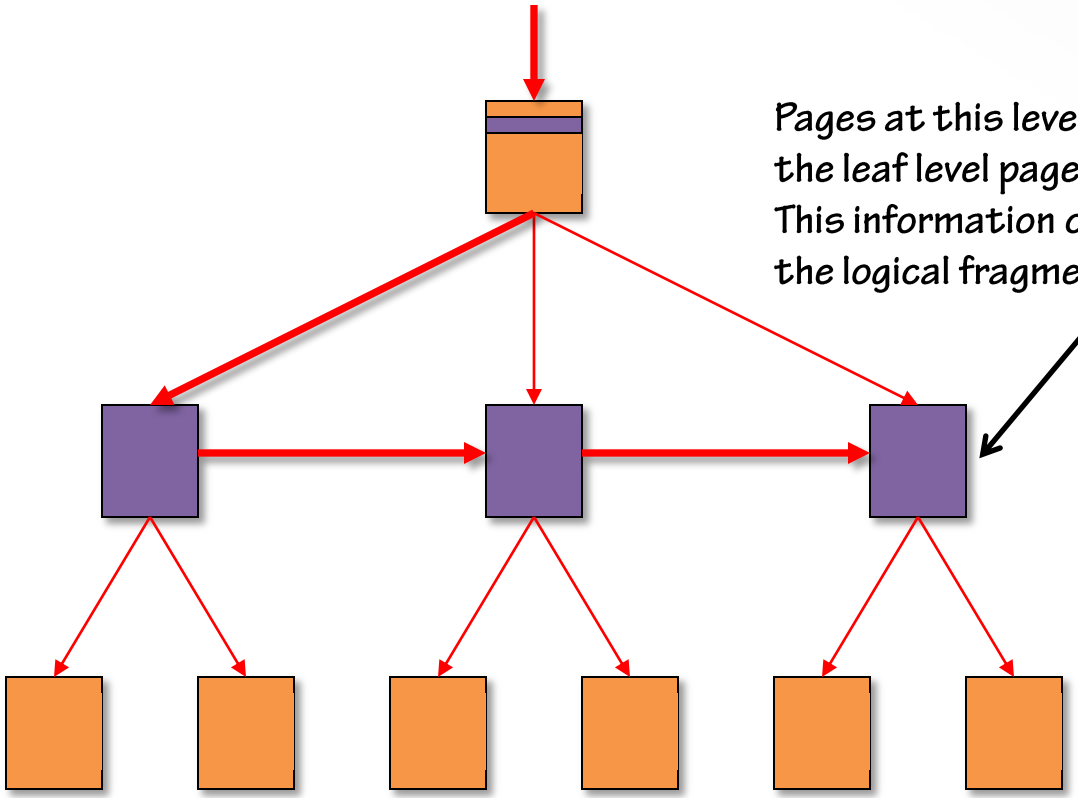
Configuring FILLFACTOR

- **Balancing act between how often page splits occur and how often you can rebuild/defrag the index**
- **What is going to cause page splits in your schema?**
  - UPDATEs to variable-width data types?
  - Random INSERTs?
    - The more volatile ⇨ lower FILLFACTOR
- **How often can you rebuild/defrag?**
  - The more frequent ⇨ higher FILLFACTOR
- **Pick a value, try it, monitor fragmentation, tweak it**
  - Use DMVs to see how fast the fragmentation increases
  - The faster fragmentation occurs ⇨ lower FILLFACTOR or decreased time between rebuilds/defrag
  - 70% is a common first guess

sys.dm_db_index_physical_stats (1)

- **Replacement for DBCC SHOWCONTIG**
  - select * from sys.dm_db_index_physical_stats (dbid, objectid, indexid, partitionid, samplemode)
- **No longer need insert/exec to analyze/process DBCC SHOWCONTIG results**
  - DMVs are programmatically "composable"
  - However, this is a DMF, not a true DMV so must do work for results
- **Ability to control how much data is read using sample mode (LIMITED, SAMPLED, DETAILED)**
  - LIMITED (default) does not read the leaf level so is fastest mode
  - SAMPLED reads 1% of the leaf-level pages if the index/partition has more than 10000 pages
  - DETAILED reads everything and is the slowest mode

sys.dm_db_index_physical_stats (2)
LIMITED Scanning Mode



sys.dm_db_index_physical_stats (3)
Output and How to Interpret it

- **Logical fragmentation**
  - avg_fragmentation_in_percent (should be low)
- **Page density**
  - avg_page_space_used_in_percent
    - Should be high for data warehouse
    - Should have some free space for OLTP

- **Other counters exist (e.g. fragments, avg. fragment size) but these were only invented to be more accessible to users – somewhat unsuccessfully**

Demo

Detecting fragmentation using sys.dm_db_index_physical_stats

How to Correct Fragmentation?

- **2 realistic choices**
  - Rebuild the index: ALTER INDEX … REBUILD
  - Defrag the index: ALTER INDEX … REORGANIZE
- **Also**
  - CREATE INDEX … WITH (DROP_EXISTING = ON)
    - Commonly used to move or (re)partition an index
- **Can also choose not to remove fragmentation**
  - If the index isn't used for range scans, and page density isn't an issue, why spend the resources?
- **Don't necessarily just rebuild everything every day**
  - Although for an involuntary DBA with enough resources, this is can be better than nothing
- **Synchronous mirroring or AGs may force REORGANIZE to be used**

ALTER INDEX … REBUILD

- **Replaces DBCC DBREINDEX in SQL Server 2005 onward**
- **Pros**
  - Atomic operation
  - No knowledge of index schema or constraints required
  - Can use multiple CPUs, and control MAXDOP
  - Rebuilds index statistics (with equivalent of full scan)
  - Can rebuild a single partition or all partitions
  - Can be done online (indexes with LOBs from 2012+, single partition from 2014+)
  - Rebuilding using the same sort order does not require a sort
  - Can be minimally-logged (but log backup will be the same size)
- **Cons**
  - Atomic operation – potentially long rollback on interrupt, all or nothing semantics
  - Requires creating complete new index before dropping old one
  - When offline – Sch-M table lock for rebuilds

ALTER INDEX … REORGANIZE

- **Replaces DBCC INDEXDEFRAG in SQL Server 2005 onward**
- **Pros**
  - ALWAYS online – only requires table IX lock
  - Interruptible with no loss of work – stops instantly
  - Has progress reporting in sys.dm_exec_requests (percent_complete)
  - Optionally compacts LOB storage (on by default)
  - Usually faster for a lightly fragmented index
  - Can reorganize one or all partitions
  - Does not require any extra disk space
- **Cons**
  - Usually slower for a heavily fragmented index
  - Always fully-logged, single CPU only, does not update statistics

Comparison Points

- **Space required**
  - This may force you to do REORGANIZE
- **Log generated**
  - This may force you to do 'staggered index maintenance' using REORGANIZE
- **Algorithm speed on amount of fragmentation**
- **Locks required (i.e. online or not)**
  - This may force you to do REORGANIZE
- **Interruptible or not**
- **Progress reporting or not**

When To Rebuild vs. Defrag

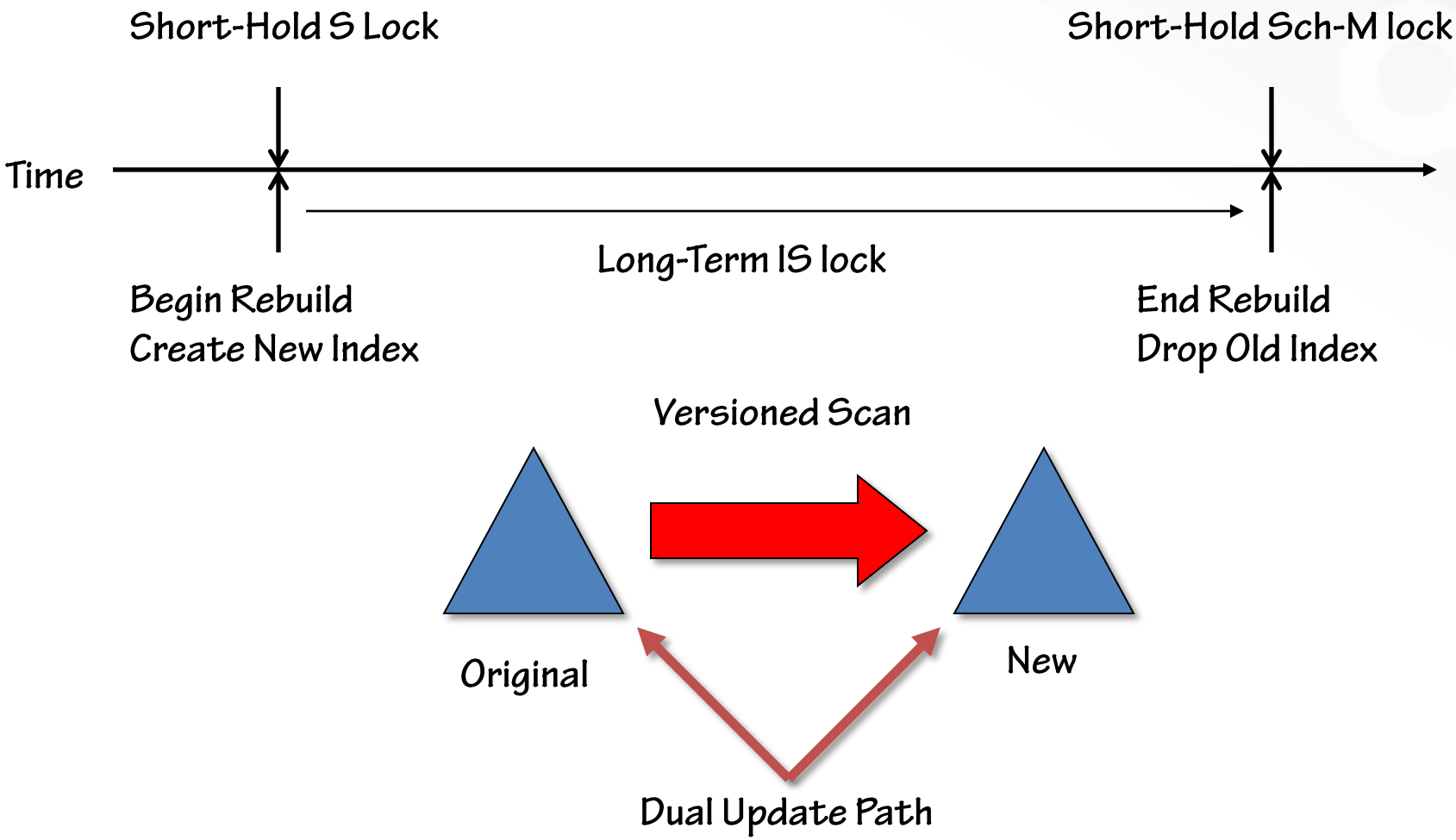
- **Much debate on this, basically it depends!**
- **I had to come up with numbers for Books Online so I chose:**
  - < 5-10% do nothing
  - 5-10% <> 30% defrag/reorganize
  - 30%+ rebuild
  - And don't do anything if the index has < 1-5000 pages
- **Your mileage may (and will) vary**

Paul's Method…

- Create a table with one row per index you want to work on
  - I call it the 'driver table'
- Call the DMV for the indexes listed in the driver table
- Use per-index fragmentation thresholds to determine whether to rebuild, reorganize, or do nothing
- Log what you decide to do for future reference
- Optional: keep a counter of how many times in succession an index is rebuilt and programmatically reduce fillfactor

- Much easier: use code someone's already written
  - http://ola.hallengren.com/

Inside Online Index Operations



Demo

Removing fragmentation

Are Your Indexes Being Used?

- **There are lots of bad practices around index strategy, including creating extra indexes**
  - E.g. an index for each column in the table
- **Extra, unused indexes waste resources as they must be maintained by DML operations**
- **Use the sys.dm_db_index_usage_stats DMV to tell if an index is being used at all during the business cycle**
  - Beware of indexes not being used but enforcing unique constraints

Summary: Prevent Fragmentation

- **As you can see, fragmentation is very expensive when it happens**
- **Many people say not to bother about fragmentation with SSDs**
  - They're WRONG!
  - Lots of wasted space on your SSD?
  - Lots of wasted buffer pool memory?
  - Lots of extra log to back up, ship, mirror, scan…
  - Performance hit of the page splits happening
- **Set appropriate FILLFACTORs for indexes that get heavily fragmented**
  - Reduce all three extra uses of space
  - Start with FILLFACTOR = 70 and tweak as needed
- **Try to set clustered index keys to avoid page splits**
  - Clustered index is the largest index to work on

Resources

- **My blog**
  - http://www.sqlskills.com/blogs/paul/category/fragmentation/
- **Pluralsight course**
  - http://www.pluralsight.com/courses/sqlserver-index-fragmentation-internals-analysis-solutions
- **Ola's scripts to automate index maintenance and more**
  - http://ola.hallengren.com/
- **Microsoft SQL Server 2000 Index Defragmentation Best Practices**
  - http://technet.microsoft.com/library/Cc966523
  - Based on SQL Server 2000, so discusses DBREINDEX vs. INDEXDEFRAG
  - Concepts translate to 2005+
- **Online Indexing Operations in SQL Server 2005**
  - http://technet.microsoft.com/library/Cc966402

Thank you!