

z/TPFDF Support for Recoup Optimized Chain Chase

—

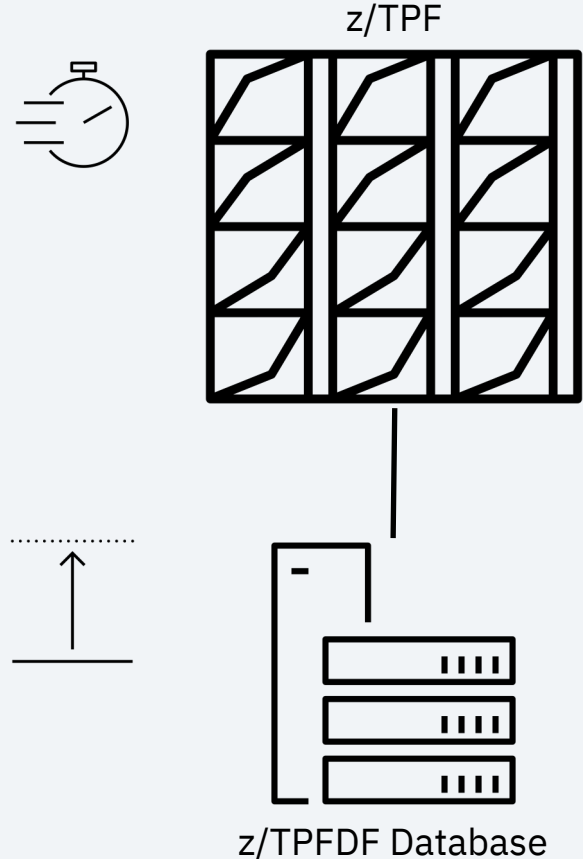
Chris Filachek

Disclaimer

Any reference to future plans are for planning purposes only. IBM reserves the right to change those plans at its discretion. Any reliance on such a disclosure is solely at your own risk. IBM makes no commitment to provide additional information in the future.

Recoup Runtime Considerations

- Allowing Recoup to run on fenced I-streams will provide additional CPU capacity
 - Help Recoup run faster – Let recoup use more CPU than previously
 - Avoid interfering with transactional traffic by running on fenced I-streams
- What if number of I/O's dominates Recoup runtime?
 - Recoup runtime increases as your databases grow larger




Problem Statement

Typically, Recoup has a time window when it can run. If Recoup exceeds the time window, other work is impacted.

As databases grow larger, it becomes difficult to keep Recoup run time within the allowed time window.

Pain Points

A balancing act is performed when Recoup is run...

- z/TPFDF databases can contain 100's of billions of records
 -  z/TPF has room to scale with 40,000 real time mods, 64K cylinders, and FARE6 addressing for both fixed and pools.
- Recoup might take over a day to read 10's of billions of records depending on your z/TPF system's IOPS capacity
- If Recoup runs past its allotted time, other utilities or business-critical processing might be impacted.
- As databases grow, will Recoup complete in its allotted time?

As-Is

z/TPFDF Fixed Record Index

| | | |
|-----------------------------|-----------------------------|-----------------------------|
| Index ordinal 0 FA = 123 | Index ordinal 1 FA = 456 | Index ordinal 2 FA = 789 |
|-----------------------------|-----------------------------|-----------------------------|

Subfile A
FA = 123

Prime block
with no
overflows

Subfile B
FA = 456

Prime block
with 2
overflows

Subfile C
FA = 789

Prime block
with 1
overflow and
an LLR

LLR Data

Recoup chain chase reads all records in the database

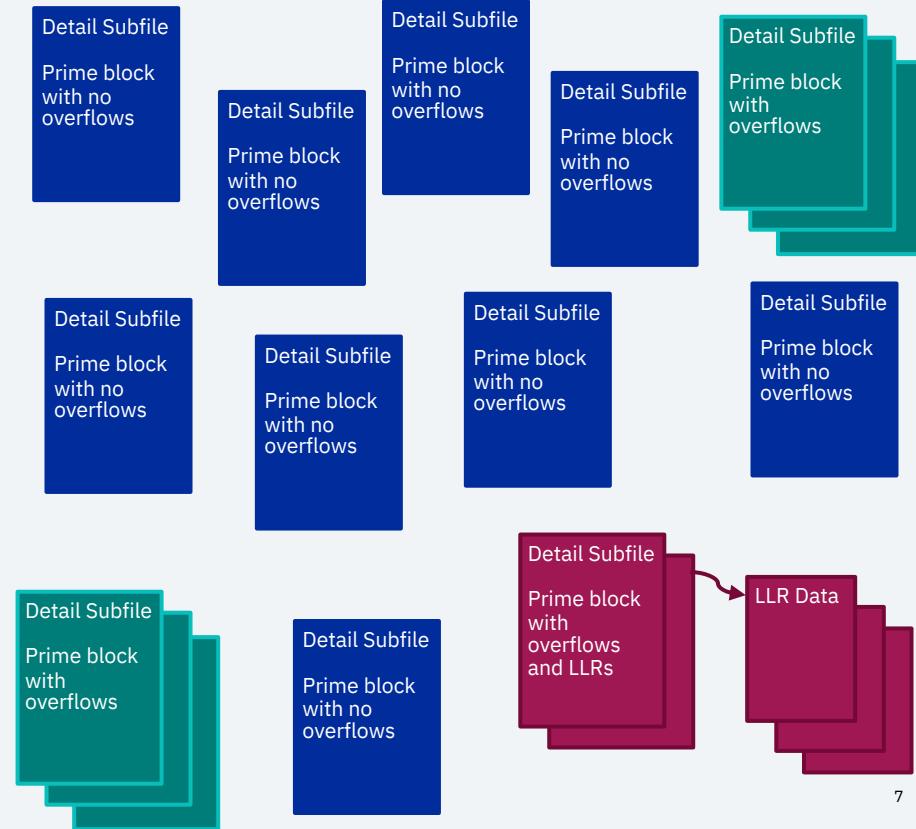
- Recoup reads Subfile A and finds the prime record
- Recoup reads Subfiles B & C and finds prime, overflow, and LLR records
- Almost 2 days to chain chase a database with 1B index ordinals & 80B detail subfiles @ 500K IOPS

Value Statement

A database administrator can use the Recoup Optimized Chain Chase (ROC) option on an indexed database with billions of single-record detail subfiles and be able to chain chase that database in a fraction of the time compared to existing recoup processing.

- For databases with a large number of detail subfiles consisting of only a prime block
- Database can still contain detail subfiles with overflows and LLRs

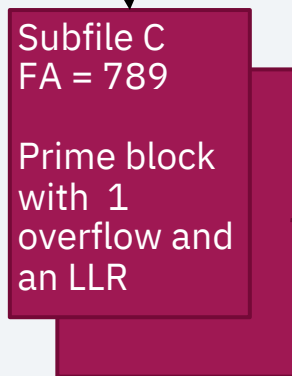
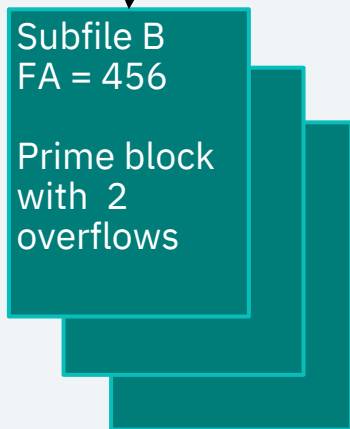
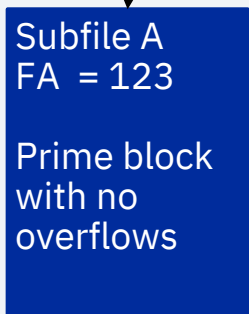
Indexed DB with mostly single-record detail subfiles
Some subfiles might have overflows or LLRs



To-Be

z/TPFDF Fixed Record Index

| | | |
|---|---|---|
| Index ordinal 0 FA = 123 ROC = NoChase | Index ordinal 1 FA = 456 ROC = Chase | Index ordinal 2 FA = 789 ROC = Chase |
|---|---|---|



Recoup chain chase conditionally reads records in the database

- ROC Flag indicates if Recoup needs to read the detail subfile in the index LREC
 - NoChase – Only 1 record. Mark pool as in-use and skip read
 - Chase – Multiple records. Read all records in chain
- **About 30 minutes** to chain chase 1B index ordinals and 80B “NoChase” subfiles @ 500K IOPS

To-Be

z/TPFDF Fixed Record Index

| | | |
|---|---|---|
| Index ordinal 0 FA = 123 ROC = NoChase | Index ordinal 1 FA = 456 ROC = Chase | Index ordinal 2 FA = 789 ROC = Chase |
|---|---|---|

Subfile A
FA = 123

Prime block
with no
overflows

Subfile B
FA = 456

Prime block
with 2
overflows

Subfile C
FA = 789

Prime block
with 1
overflow and
an LLR

LLR Data

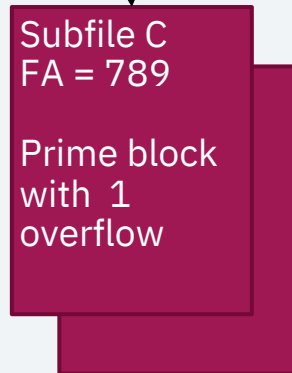
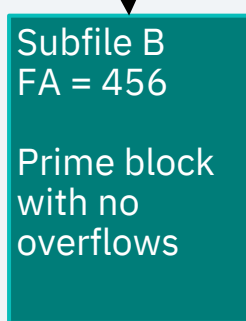
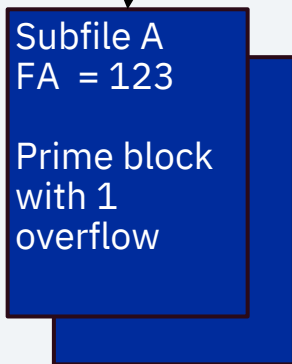
Benefits a range of database sizes...

- For example: 50M index ordinals, 2B subfiles, and 50M overflow records
 - As-Is: ~70 minutes to chain chase @ 500K IOPS
 - **To-Be: ~5 minutes** to chain chase @ 500K IOPS
- The more single-record detail subfiles, the greater the benefit

To-Be

z/TPFDF Fixed Record Index

| | | |
|---|---|--|
| Index ordinal 0 FA = 123 ROC = Chase | Index ordinal 1 FA = 456 ROC = NoChase | Index ordinal 2 FA = 789 ROC = Chase |
|---|---|--|



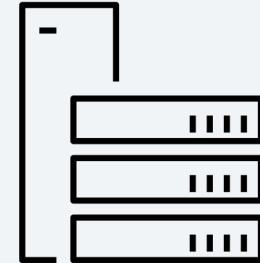
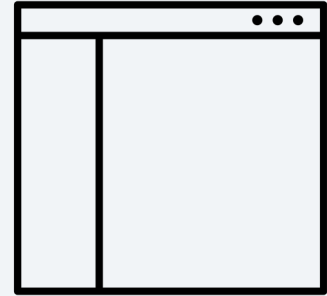
z/TPFDF automatically updates the ROC Flag as the subfile changes

- ROC Flag is updated during subfile close if the subfile changes between only 1 record and more than 1 record
 - Updated ROC Flags - Subfiles A & B changed between no overflows and having overflows
 - No ROC Flag Update - Subfile C's LLR was deleted but still has overflows
- A technical LREC (TLREC) is stored in each detail subfile with index information
 - The TLREC is used to find the index reference and update the ROC Flag

Technical Details

- No application changes are required
 - Supported with most z/TPFDF APIs
- Enabled through new DBDEF parameters on the index and detail files
 - Supported for z/TPFDF single-level indexed databases
 - Index files must be used with 8-byte forward index path references (FAL=8)
 - Detail files must be R-type files with variable length LRECs and allow TLRECs up to ID X'15'

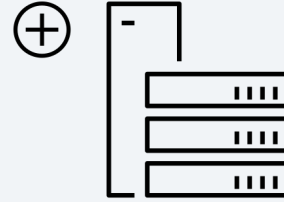
✔ z/TPFDF Application



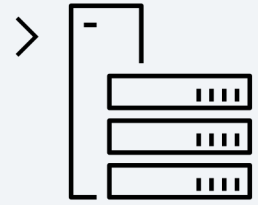
z/TPFDF Database

Technical Details

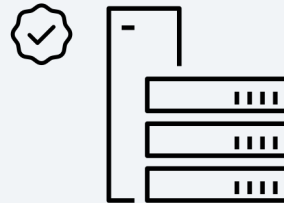
- Use with new or existing databases
 - Recoup optimized chain chase (ROC) used only for subfiles indexed after the option is enabled
- Recoup will not check for broken chains in subfiles not chased due to the ROC option
 - CRUISE VERIFY can be run as a low priority utility in the background to check for broken chains in all subfiles



Add Database



Edit Database



Verify Database

Technical Details

- Is my database a potential candidate for ROC?
 - Are z/TPFDF file requirements met?
 - Single level indexed database, R-type file, etc.?
 - Are there millions of detail subfiles with no overflows or LLRs?
 - Use Recoup or CRUISE statistics to compute the number of subfiles with no overflows
 - Subfiles with no overflows = Number of prime blocks (BLOCKS P/) -
Sum of the CHAINS counts (1/ + 2/ + 3/ + ... + M/)
 - Is there room for TLRECs without causing overflows?
 - Each TLREC is 40-300 bytes depending on the algorithm string
- Contact us if you have a potential candidate and are interested!

We want sponsor users!

These recoup performance improvements are active development projects.

If you are interested in participating as a sponsor user, please email one of the contacts below.

Contact:

Chris Filachek

filachek@us.ibm.com

Danielle Tavella

danielle.tavella@ibm.com

Q&A

Summary of Q&A from the virtual TPFUG event:

| Question | Answer |
|--|---|
| What is a pool record is erroneously available but is not chased due to this new option? | Erroneously available will be identified and protected by Recoup the same way they are today. Recoup will recognize that this pool address is in-use but is also listed as available in the pool directory. Recoup will identify this pool as erroneously available and protect the pool address in the directory (change it from available to in-use). |
| By not reading single-record subfiles, won't record ID errors be missed? | That is correct. Because Recoup skills reading the pool record for a single-record subfile, Recoup can't check the record ID for that pool and report errors. If the ROC is turned on for a database, we recommend running CRUISE VERIFY as a low-priority utility in the background to check for record ID errors. |
| Does this option support multiple indexes referencing the same subfiles. | Yes, this is supported. In addition, the ROC flag can be enable on each of the indexes or only a subset. If the ROC flag is enabled on a subset of indexes, indexes with ROC should be chain chased first by Recoup with RCI processing enabled so recoup skips those subfiles if the same subfiles are found again in indexes that do not have the ROC flag enabled. |

Q&A

Summary of Q&A from the virtual TPFUG event:

| Question | Answer |
|---|--|
| Is this supported on indexes with 4-byte addresses or indexes that were migrated to 8-byte address slots? | <p>This option is not supported for indexes with 4-byte or 8-byte migrated address slots. The index must be defined with “FAL=8”.</p> <p>If you are interested in this support with these types of indexes, please let us know so we can consider supporting these types of indexes.</p> |
| Won't CRUISE VERIFY take as long to run as RECOUP without this option? | Yes, but unlike Recoup, CRUISE VERIFY will not prevent you from running PDU or other pool utilities while it is running. |
| If the ROC Flag in the index is corrupted, could that cause lost pools? | <p>If the ROC Flag is set to “NoChase” when the subfile has overflows or LLRs, then this would lead to lost pools being reported by Recoup. Whether or not you are using ROC, customers should always review lost address reports before returning lost pools and should exclude any lost pools that are a concern or unexpected.</p> <p>In addition, this support is updating CRUISE to verify the ROC flags and TLRECs to report and fix any issues.</p> |

Thank you

© Copyright IBM Corporation 2021. All rights reserved. The information contained in these materials is provided for informational purposes only, and is provided AS IS without warranty of any kind, express or implied. Any statement of direction represents IBM's current intent, is subject to change or withdrawal, and represent only goals and objectives. IBM, the IBM logo, and ibm.com are trademarks of IBM Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available at [Copyright and trademark information](#).

