Enhanced Logical Record Cache

File Support team

Michael Shershin, TPF Development Lab

IBM **z/TPF**

April 11, 2016

Logical record cache

Logical record cache (LRC) definition

- What is logical record cache (LRC)?
  - Set of APIs and infrastructure to keep data in memory on TPF
  - Can be used to:
    - Reduce I/O needs
    - Reduce processing overhead to access data
  - Provided in TPF 4.1

LRC characteristics

- Many logical record caches can exist
  - A logical record cache can exist for specific functions
  - Examples: File system / DNS
- Use of logical record cache is not transparent to applications
  - Requires use of logical record cache APIs
    - Update data (or put data) in cache
    - Read data from cache
- Data access in a logical record cache is done with a primary key
  - Optionally a secondary key can be used
  - Both primary and secondary keys can be up to 256 bytes in size

LRC characteristics

- Records up to 4 K in size can be used in LRC
- Records in a specific cache must be the same size
- Ability to set time out values for data in the cache
  - Use cast out value
  - Prevent use of stale data
- When TPF IPLs:
  - Data is lost
  - Responsibility of the application to recreate and re-populate the cache

LRC characteristics

- Processor unique or processor shared cache
- Coupling Facility is used for processor shared cache
  - Update to data on one processor results in the data being not valid on all other processors in the complex
  - A read for the data on other processors will return “not in cache”
  - Applications responsibility to re-populate data in cache
- Intended for records with high read to write ratio

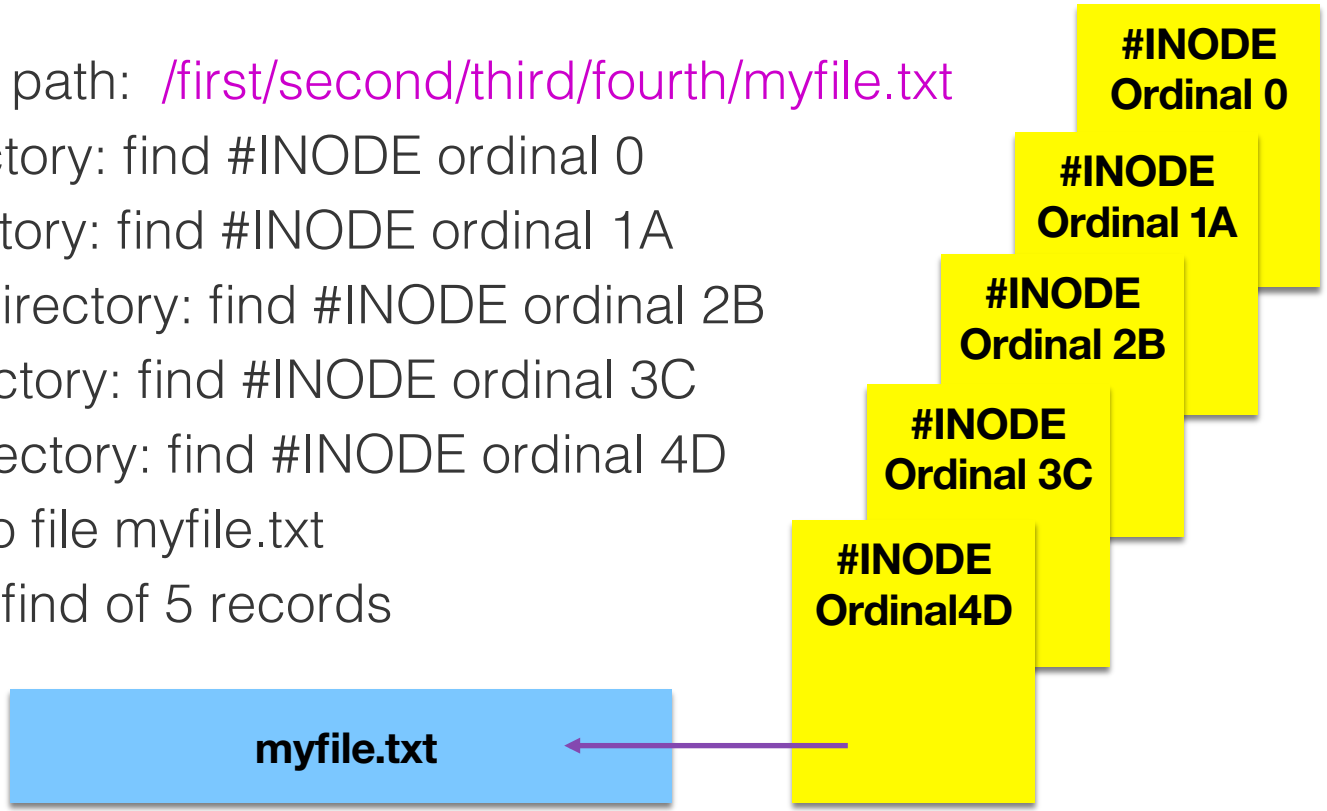
LRC vs VFA

- VFA is a cache for DASD records
  - Integrated into TPF find / file logic
  - No application changes requires to use VFA
  - One cache for all candidate records
- Why use logical record cache rather than VFA?
  - Use of keys in logical record cache can provide direct access to data compared to VFA

File system example without LRC

Open file using path: /first/second/third/fourth/myfile.txt

- For root directory: find #INODE ordinal 0
- For first directory: find #INODE ordinal 1A
- For second directory: find #INODE ordinal 2B
- For third directory: find #INODE ordinal 3C
- For fourth directory: find #INODE ordinal 4D
- Get access to file myfile.txt
- Requires the find of 5 records



File system example with LRC

Open file using path: /first/second/third/fourth/myfile.txt

- Initial access
  - Read all 5 records to get to #INODE ordinal 4D
  - #INODE ordinal 4D is put into logical record cache using primary key /first/second/third/fourth
- Subsequent accesses
  - Read primary key /first/second/third/fourth to get #INODE ordinal 4D
  - Get access to file myfile.txt
  - Requires the logical record cache access of 1 record
- If accessed frequently, significant savings can be realized

#INODE Ordinal 4D

LRC APIs

- Manage a cache
  - deleteCache()
  - flushCache()
  - newcache()
  - tpf_newcache_ext()
- Manage entries in a cache
  - deleteCacheEntry()
  - readCacheEntry()
  - tpf_updateCacheEntry_ext()
  - updateCacheEntry()

PJ42731 - Enhanced logical record cache

Problem

- Need ability to cache data entries that are larger than 4 K
- Need ability to cache data entries of different sizes

Enhancements

- PJ42731 is on PUT 13
- What is Enhanced logical record cache (ELRC)?
  - Same characteristics as LRC with the following enhancements
    - Individual entries in a cache can be larger than 4 K
    - Individual entries in a cache can be different sizes
  - Large data entries are managed as single entry
    - readCacheEntry() returns the entire entry
    - updateCacheEntry() writes the entire entry
    - If the entry is 16 K, read and update APIs treat entry as 16 K

Enhancements

- What is Enhanced logical record cache (ELRC)?
  - Same APIs as LRC
  - A cache is marked as ELRC when the cache is created
    - newCache() or tpf_newCache_ext()
    - ELRC is used when data_length parameter is greater than 4 K
  - ZCACH displays are updated
    - ZCACH DISPLAY ATTRIBUTES
    - ZCACH DISPLAY COUNTS
  - Updates to data collection / reduction
    - For both LRC and ELRC

ELRC usage

- Enhanced logical record cache (ELRC) is the underlying infrastructure for TPFDF cache
- Recommended use
  - High read to write ratio
  - Entries or objects that are greater than 4 K in size
    - readCacheEntry() will return a single entry
      - If the entry is 16 K in size, readCacheEntry() will return all 16 K
    - If a single entry is multiple 4 Ks, expect a savings over VFA due to reduced linkage costs

Example

```
==> ZCACH DISPLAY CTL
CSMP0097I 14.03.10 CPU-B SS-BSS  SSU-HPN  IS-01
CACH0002I 14.03.10 CACHE CONTROL AREA DISPLAY
NAME  IVFS_FS_DIR , ADDR  00000009F3892000
NAME  TPF_RECBUF  , ADDR  00000009F38F3000
NAME  IFFS_ICACHE , ADDR  00000009F3E42000
NAME  IPFS_ICACHE , ADDR  00000009F3E4D000  _
NAME  TPF_FS_DIR  , ADDR  00000009F3E58000
NAME  TPF_FS_INODE, ADDR  00000009F3EAC000
NAME  IPRCFF009490, ADDR  00000009F41F9000
NAME  ISYSFF005650, ADDR  00000009F5BAC000
NAME  ITPFDFFILES , ADDR  00000009F7C00000
NAME  ISYSFE01AE70, ADDR  00000009F9500000
NAME  IDNSHOSTADDR, ADDR  00000009FAEFE000  _
NAME  IDNSHOSTNAME, ADDR  00000009FC497000
DISPLAY COMPLETE+
```

Example

```
==> ZCACH DISPLAY ATTRIBUTES IVFS_FS_DIR
CSMP0097I 14.03.57 CPU-B SS-BSS  SSU-HPN  IS-01
CACH0035I 14.03.57 CACHE ATTRIBUTE DISPLAY
NAME  IVFS_FS_DIR , ADDR  00000009F3892000
CACHE ATTRIBUTES: TRADITIONAL, PROCESSOR UNIQUE, NOT CONNECTED TO CF,
                  CACHE CREATION/DELETION NOT SYNCHRONIZED
NUMBER CACHE ENTRIES DEFINED 1323 _
NUMBER HASH ENTRIES DEFINED  661
PRIMARY KEY SIZE  64  SECONDARY KEY SIZE  16
MAX ENTRY SIZE    88
CAST OUT TIME     3600
DISPLAY COMPLETE
```

Example

```
==> ZCACH DISPLAY ATTRIBUTES ITPFDFFILES
CSMP0097I 14.04.29 CPU-B SS-BSS  SSU-HPN  IS-01
CACH0035I 14.04.29 CACHE ATTRIBUTE DISPLAY
NAME  ITPFDFFILES , ADDR  00000009F7C00000
CACHE ATTRIBUTES: EXTENDED, PROCESSOR SHARED, CONNECTED TO CF,
                  CACHE CREATION/DELETION SYNCHRONIZED
NUMBER CACHE ENTRIES DEFINED 125002 _
NUMBER HASH ENTRIES DEFINED  62497
PRIMARY KEY SIZE  8  SECONDARY KEY SIZE  4
MAX ENTRY SIZE    512000000
CAST OUT TIME     0
EXTENDED CACHE INFORMATION FOR LARGE CACHES
TOTAL MEMORY USABLE BY CACHE ENTRIES     513032192
DISPLAY COMPLETE
```

Example

```
==> ZCACH DISPLAY COUNTS ITPFDFFILES
CSMP0097I 14.05.20 CPU-B SS-BSS  SSU-HPN  IS-01
CACH0036I 14.05.20 CACHE DATA COLLECTION COUNTER DISPLAY
NAME  ITPFDFFILES , ADDR  00000009F7C00000
NUMBER CACHE ENTRIES DEFINED 125002
NUMBER CACHE ENTRIES IN USE  6165
CAST OUT TIME                0
CALLS      579101445   MISSED     38521203    _
CASTOUTS   0   UPDATES    11595182   INVALIDATED    27171003
DUPLICATE HASH REFUSALS      0
EXTENDED CACHE INFORMATION FOR LARGE CACHES
TOTAL MEMORY USABLE BY CACHE ENTRIES     513032192
MEMORY ACTUALLY IN USE BY CACHE ENTRIES  124526592
AVERAGE CACHE ENTRY SIZE                 20198
DISPLAY COMPLETE
```

Summary

- Both LRC and ELRC
  - Provide ability to reduce I/Os
  - Provide ability to invalidate records in cache in loosely coupled complex using a CF
- ELRC
  - Provide ability to cache data entries that are larger than 4 K
  - Provide ability to cache data entries of different sizes
- Available for all customers (loosely coupled and non-loosely coupled)

Thank you!

Questions or comments?

Trademarks

- IBM, the IBM logo, ibm.com and Rational are trademarks or registered trademarks of International Business Machines 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 on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

Notes

- Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.
- All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.
- This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.
- All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.
- Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.
- Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.
- This presentation and the claims outlined in it were reviewed for compliance with US law. Adaptations of these claims for use in other geographies must be reviewed by the local country counsel for compliance with local laws.