IBM Education Assistance for z/OS V2R2

Item: Contractible CPOOL
Element/Component: BCP Virtual Storage Manager (VSM)

Material current as of March 2015

Agenda

- Trademarks
- Presentation Objectives
- Overview
- Usage & Invocation
- Migration & Coexistence Considerations
- Presentation Summary
- Appendix

Trademarks

- See url http://www.ibm.com/legal/copytrade.shtml for a list of trademarks.

Presentation Objectives

- Understand the new contractible CPOOL capability

Overview

- Problem Statement / Need Addressed
  - CPOOL (a VSM cell pool service) cannot be contracted because of serialization and performance requirements that dictate its algorithm. Due to potentially excessive storage utilization, some areas cannot use multi-headed CPOOL.
- Solution
  - When transactional execution is available, provide contraction capabilities under control of the pool owner.
- Benefit / Value
  - Improved usability and indirectly performance if it allows additional usage of cell pools

Overview Approach

- Constrained Transactions
  - Allows updating of free count and freeing extent without being exposed to timing scenarios that could otherwise reference freed storage (this is why current CPOOL does not allow contraction)
- On contraction, the cell is freed but the extent block is not.

Usage & Invocation – CPOOL macro

- CPOOL BUILD
  - CONTRACTIBLE=NO
  - CONTRACTIBLE=COND (based on TX availability)
    - Requires MULTIHDR=YES
    - Requires CELLSPERCPU=1 (1 cell per extent)
  - PLISTVER=0 | 1 | MAX | IMPLIED_VERSION

Usage & Invocation – CPOOL macro

- CPOOL BUILD with CONTRACTIBLE=COND(cont)
  - AUTOCONTRACT=YES | NO
    - When YES, system will initiate contraction based on user-defined criteria

When AUTOCONTRACT=YES:

- FREECELLS
  - Target number of free cells at conclusion of contraction, multiplied by number of applicable online CPUs
- FREECELLSPERCPU
  - Which CPU types are applicable (All, Standard, zAAP, zIIP)
- FREECELLSDIVISOR
  - Divisor in calculation (to provide flexibility)
- CONTINTVL
  - Number of seconds after previous contraction to begin next one

Usage & Invocation – CPOOL macro

- CPOOL BUILD with CONTRACTIBLE=COND(cont)
  - ONLY when z/OS 2.2 functions (bit CVTZOS_V2R2)
  - If TX is not available (bit PSATXC is off), contraction parameters are ignored

Usage & Invocation – CPOOL macro

- CPOOL GET
  - CONTRACTIBLE=NO
  - CONTRACTIBLE=COND
    - Requires MULTIHDR=YES or MULTIHDR=COND
    - Contractible if TX available, normal if not
    - MULTIHDR=COND
      - Use MULTIHDR=NO when no contraction available, MULTIHDR=YES when available

If you specified CONTRACTIBLE=COND on BUILD you must specify CONTRACTIBLE=COND on every GET

Usage & Invocation – CPOOL macro

- CPOOL FREE
  - CONTRACTIBLE=NO
  - CONTRACTIBLE=COND
    - Requires MULTIHDR=YES or MULTIHDR=COND
    - Contractible if TX available, normal if not
    - MULTIHDR=COND
      - Use MULTIHDR=NO when no contraction available, MULTIHDR=YES when available

If you specified CONTRACTIBLE=COND on BUILD you must specify CONTRACTIBLE=COND on every FREE

Usage & Invocation – CPOOL macro

- CPOOL CONTRACT
  - CPID
  - FREECELLS=n
  - FREECELLS=CPOOL_BUILD
  - Return code:
    - 00 (CPOOL CONTRACT, pool was defined to be contractible)
    - 04 (CPOOL CONTRACT, but pool was not defined to be contractible)

- CPOOL LIST
  - Any extent might not be valid upon return if this is a contractible pool
    - Unless contraction is prevented by holding a particular lock (the appropriate lock depends on the subpool)

Migration & Coexistence Considerations

- No Migration considerations, no Coexistence considerations
- Contraction processing, whether via CPOOL BUILD, GET, or FREE, is ignored if TX is not available.. Do not use contractible CPOOL unless the code is running on z/OS 2.2 (you can check bit CVTZOS_V2R2).

Presentation Summary

- CPOOL provides a “contraction” option, when Transactional Execution is available

Appendix

- Publications:
  - z/OS MVS Programming: Authorized Assembler Services Reference ALE-DYN