Charm++ - Bug #1442
CkLoop fixed tree limits helper recruitment
02/22/2017 12:00 PM - Jim Phillips

**Description**

CkLoop uses a tree of branching 4 when CkMyNodeSize() >= 8. This means that if you have ppn=9 and rank 0 calls CkLoop, then ranks 5-8 are only recruited if rank 1 is available. If ranks 1-4 are busy then no helpers are activated. This is unexpected and performance-limiting for the use case where CkLoop work is urgent but PEs are generally busy with lower-priority tasks of small grainsize.

A more dynamic approach would be to use messages in the node queue to push notify messages for the PEs, one message per X PEs on the node (probably large X to amortize overhead, but some limit to allow scaling to large thread counts).

**History**

**#1 - 02/22/2017 02:17 PM - Jim Phillips**

- File ckloop_node.patch added
- Target version set to 6.8.0

Proof-of-concept patch attached. With this change I see the expected behavior, with all PEs able to steal work.

Issues with this patch:
1) Had to declare CmiPushNode() in CkLoop.C, should be added to converse.h and made non-static in all machine layers.
2) No tests for CMK_NODE_QUEUE_AVAILABLE
3) Modifies srcRank in notifyMsg, which I think might be enqueued from previous entries on other PEs. This would be unsafe if int write was non-atomic, or if CkLoop is sensitive to extra notify messages being received. It is done because there is no way to tell if message came from node queue or pe queue.
4) Only modified USE_CONVERSE_NOTIFICATION branch. Do we need the other side?

Other possible improvements:
1) For numChunks << nodesize it would be better to put numChunks/numChunks-1 messages in the node queue rather than messaging every PE.
2) If no work left don't send the notifies.
3) The implementation of tree broadcast notification broke tracing by over-writing srcRank to -1.

**#2 - 03/07/2017 11:33 AM - Eric Bohm**

- Assignee set to Seonmyeong Bak

**Files**

- ckloop_node.patch 2.36 KB 02/22/2017 Jim Phillips