Realm Update

Legion Virtual Retreat - 14 June 2021

Sean Treichler

Seema Mirchandaney

Wei Wu

Realm

- High-performance asynchronous task-based runtime for distributed heterogeneous systems
  - Initially designed as a companion to Legion, now being used directly as well
- Presents a flattened abstraction of machine with uniform API for spawning tasks and accessing/transferring data
  - Does not try to solve problem of heterogenous code generation for “leaf tasks”
- Allows/expects “application” to constrain ordering of operations - dynamically schedules operation graph for best latency tolerance
  - Scales to thousands of nodes and millions of operations/node/sec
- Constant functional and performance improvements
  - Highlight a few recent/upcoming user-visible changes today

GASNet-EX Support

- Takes advantage of features in GASNet's "new" GASNet-EX API:
  - Immediate mode reduces latency in uncongested case (without stalling when congested)
  - Negotiated payload mode eliminates extra memory copy(s) on sender
  - Dynamic registration of Realm-allocated memory permits RDMA more often (GPUDirect!)
  - Local completion notification allows better pipelining of multi-hop copies
- Realm "gasnetex" network layer rewritten from scratch
  - Gets rid of a LOT of old cruft from (still-supported) "gasnet1" code
  - Set Legion_NETWORKS=gasnetex (cmake) or REALM_NETWORKS=gasnetex (make) to enable
- Current status:
  - Infiniband (ibv conduit) works in public branches, chasing a few remaining bugs
  - Aries has a few show-stopper bugs, should be fixed pretty soon
  - Expect to track GASNet-EX's support for future networks/conduits and expanded GPUDirect

AMD/HIP Support

(Primary Contact: Wei Wu, wwu@lanl.gov)

- Available in public branches
  - Legion C++ support for AMD GPU via HIP: user can create Legion GPU tasks for AMD GPU
- Usage:
  - Similar with CUDA tasks, HIP tasks are also registered as Processor::TOC_PROC
  - Cmd line enabling of HIP support is the same as existing CUDA support.
- Current restrictions:
  - If a task is launched with default HIP stream, Realm is not able to hijack the stream; therefore, users are encouraged to launch tasks with Realm managed HIP stream (hipGetTaskStream).
- Future work:
  - Kokkos support for HIP: a prototype is done, will be available soon.
  - Regent support for HIP

Aurora Support

(Primary Contact: Seema Mirchandaney, seema.mirchandaney@stanford.edu)

- Currently lives in a development branch:
  - https://gitlab.com/StanfordLegion/legion/-/tree/intel-dev
- Status:
  - Level Zero backend integrated into Realm, support for multiple GPUs (Gen9/Gen12)
  - DPC++ kernels integrated into Legion tasks
  - saxpy, stencil, circuit tests written for Aurora Legion
- Challenges:
  - Integrate regularly with newer versions of OneAPI toolkit
  - DPC++ interoperability with level zero backend (WIP)
- Future work:
  - Enable options currently disabled for new hardware features
  - Performance analysis and optimization, documentation, merge into master branch

Processor/Memory “Kind” Rework

- Current “Kind” enums for processors/memories are too restrictive
  - e.g. can’t have a Python processor with an associated GPU
  - Offers no additional information (e.g. SM architecture, AVX512 support, ...)
- Plan:
  - Provide richer ProcessorInfo/MemoryInfo objects that can be interrogated
  - Add corresponding predicates for machine model queries
  - Work out matching changes required in Legion task variant constraints
  - Maintain old kind enum (with existing limitations) for backwards-compatibility
- ETA:
  - Prototype in the fall, hopefully merged in for 21.12 release
  - Looking for folks with interesting use cases to drive design/testing

Programmatic Configuration

- Problem: command-line configuration of Realm is getting unwieldy
  - Lots and lots of options, often interacting with each other
  - Many options dependent on target machine (e.g. available CPU core count)
  - Often interferes with application's use of command line options
- Idea: perform Realm configuration in a small Lua (or Python?) script
  - Script can examine command line, system configuration, etc. and set options as desired
  - Default script provided to match current command-line parser, can be augmented/replaced
  - Allows for application- and/or target-specific scripts
  - Maybe provides a hook for dynamic reconfiguration in the future?
- ETA:
  - Also hoping for prototype in fall in order to make 21.12 release
  - Also looking for interested parties to beta test (and likely help steer functionality decisions)

CUDA Graph Support

- Realm “CUDA tasks” are host functions that happen to launch CUDA work
  - CPU overhead is often a bottleneck for faster GPUs
  - Realm can’t optimize (e.g. fuse tasks, skip CPU synchronization) these tasks
- CUDA Graphs allow multiple kernels/copies/etc to be launched in a single API call
  - Introduced in CUDA 10.0
  - Graphs describe arbitrary (acyclic) dependencies, can be queried, included in other graphs
- Plan: new CudaGraph code descriptor available for task registration
  - Provides cudaGraph_t - Realm instantiates it as needed (for concurrency)
  - Working out details for passing arguments to task launch
  - Hoping to use CUDA stream capture to let Legion tracing use CUDA graphs automatically
- ETA:
  - Prototype soon, targeting 21.09 release