Runtime and Third-party Changes

Chapel Team, Cray Inc.

Chapel version 1.16

October 5, 2017

Safe Harbor Statement

This presentation may contain forward-looking statements that are based on our current expectations. Forward looking statements may include statements about our financial guidance and expected operating results, our opportunities and future potential, our product development and new product introduction plans, our ability to expand and penetrate our addressable markets and other statements that are not historical facts. These statements are only predictions and actual results may materially vary from those projected. Please refer to Cray's documents filed with the SEC from time to time concerning factors that could affect the Company and these forward-looking statements.

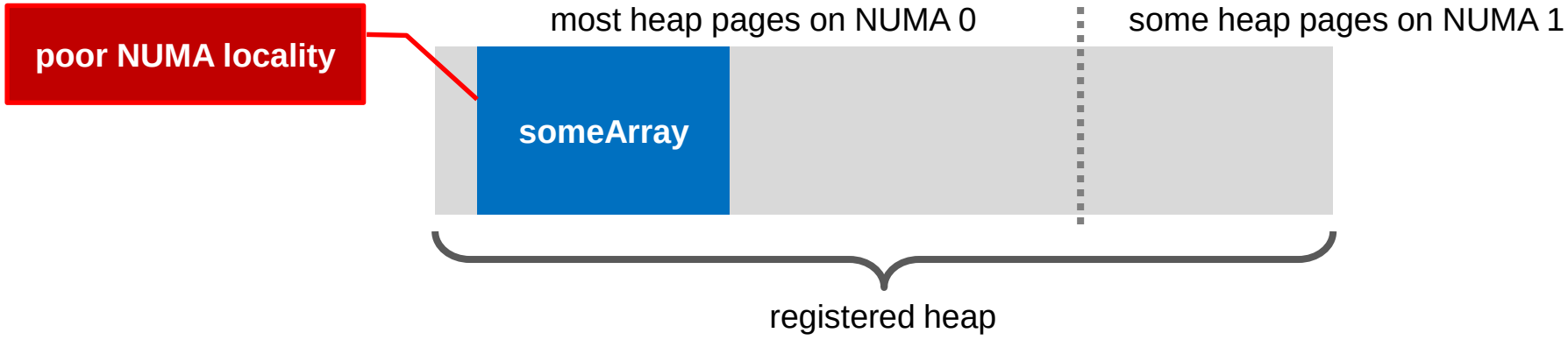
Outline

- ‘ugni’ Comm Layer: Register Arrays Dynamically
- Other ‘ugni’ Comm Layer Improvements
- ‘gasnet’ Comm Layer: Enable Multi-domain
- Other Runtime Improvements
- Other Third-party Improvements

‘ugni’ Comm Layer: Register Arrays Dynamically

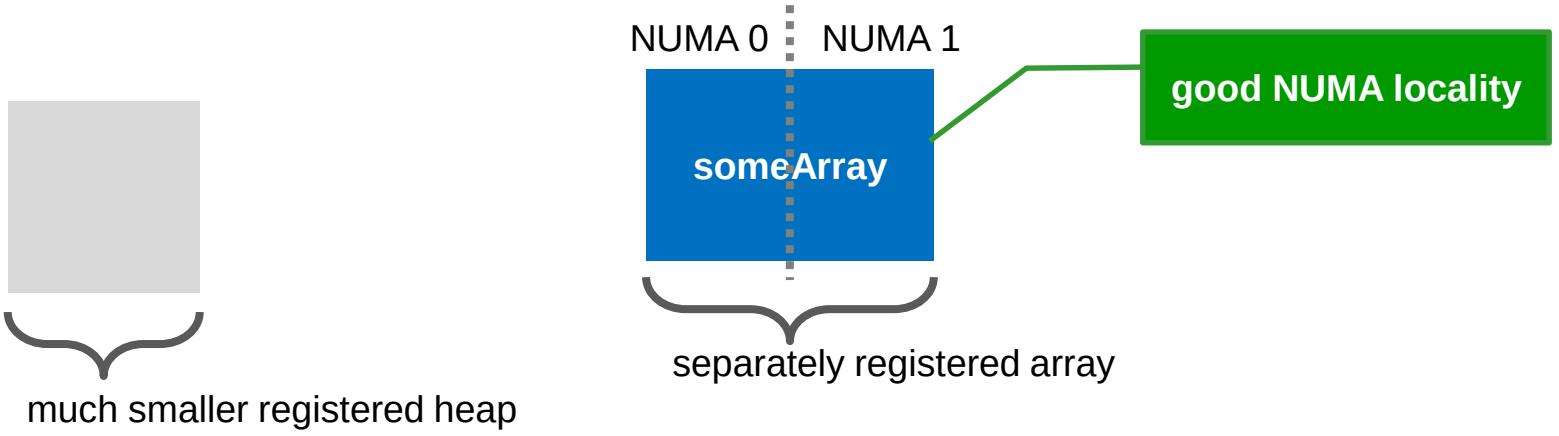
‘ugni’ Dynamic Registration: Background

- **‘ugni’ comm produced poor NUMA memory affinity:**
  1. Comm layer acquired contiguous chunk of memory to serve as heap
  2. Comm layer registered heap with NIC
     - Most/all of heap ended up on NUMA domain 0, which is closer to NIC
  3. Comm layer passed heap base+size to mem layer to manage
     - Array allocations were typically entirely on one NUMA domain

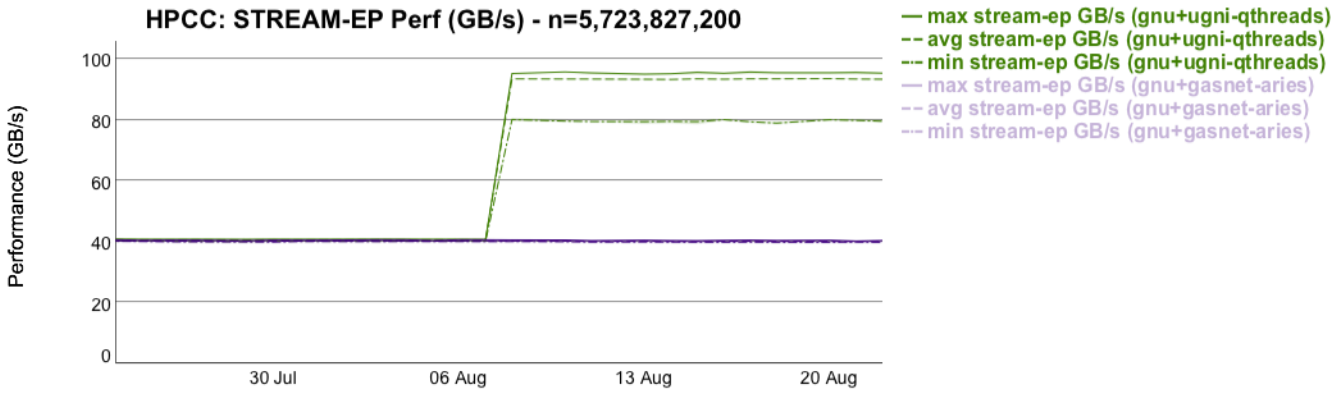


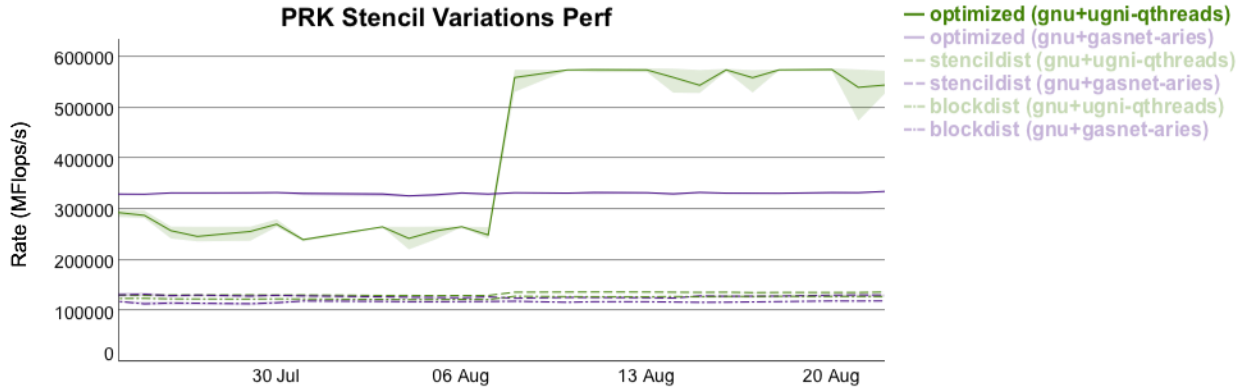
‘ugni’ Dynamic Registration: This Effort

- **Allocate arrays outside heap, register dynamically**
  1. Array allocator calls comm layer to get non-heap memory
  2. Array allocator initializes array in parallel, localizing it
     - First-touch semantics balances array localization across NUMA domains
  3. Array allocator calls comm layer again, to register the memory



‘ugni’ Dynamic Registration: Impact





‘ugni’ Dynamic Registration: Status, Next Steps

Status:

- Only applies to “big” arrays (#hugepages >= #NUMA domains)
  - Smaller arrays and other things still come from the regular heap
- Some scaling issues with large #s of registered memory regions
- Awkward: when out of memory, alloc succeeds but init gets SIGBUS

Next Steps:

- Dynamically extend and register the heap itself
  - Gets NUMA affinity for things other than big arrays
- Improve scalability for registration broadcast and lookup
- Make use of recent kernel change to avoid SIGBUS-on-OOM problem
- Investigate a full-blown registration caching implementation?

Other ‘ugni’ Comm Layer Improvements

Reduce Default Heap Size

Background: Default heap size was 2/3 node memory

- Heap had to hold everything, including arrays
- Heap was not extendable

This Effort: Reduce default heap size to 16 gb

- With dynamic registration, arrays are allocated outside the heap
- Still not really extendable, but doesn't need to be as big
- Major heap space driver: Qthreads stack pools

Impact: Much quicker program startup

- Don't have to create as many heap pages up front

Next Steps: Extend heap dynamically, on demand

- This is work in progress, which just missed the release

Use Nonblocking Ops for Strided Transfers

Background: Strided transfers under-utilized the network

- Used for array assignments that have to be done as many chunks
- Were done simply: network op, wait for done, network op, wait, etc.

This Effort: Use nonblocking technique instead

- Initiate many network ops, then wait for all
- Reduces time-to-initiate

Impact: Limited, not visible in regular nightly perf testing

- ~2x improvement on a feature-specific test, ~5% on PRK stencil

Next Steps: Probably only background efforts

- At this time few codes seem sensitive to strided transfer performance

‘gasnet’ Comm Layer: Enable Multi-domain

‘gasnet’ Multi-domain: Background and Effort

Background: ‘ugni’ comm significantly outperforms ‘gasnet’

- Especially for applications with a high degree of comm concurrency

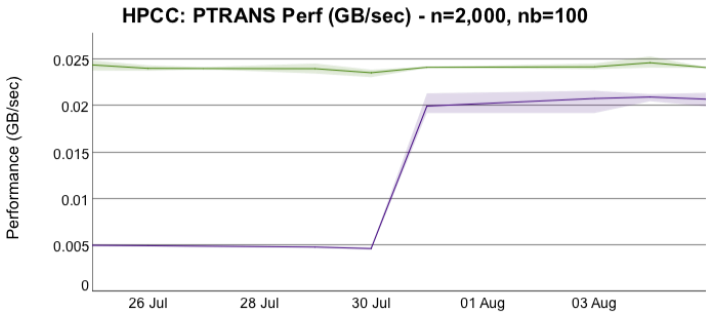
This Effort: Enable GASNet’s multi-domain feature

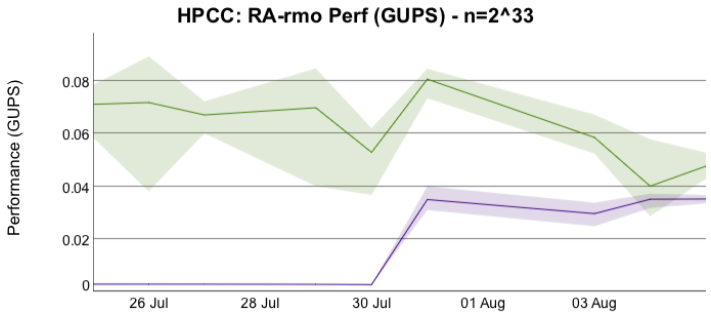
- Improves performance of parallel RDMA operations
  - aries/gemini specific feature
  - similar to what ugni does by default

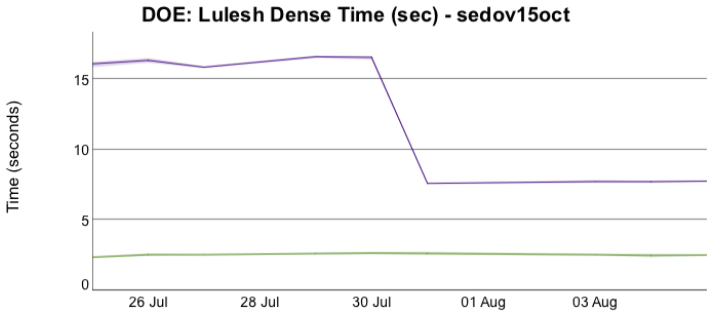
‘gasnet’ Multi-domain: Impact

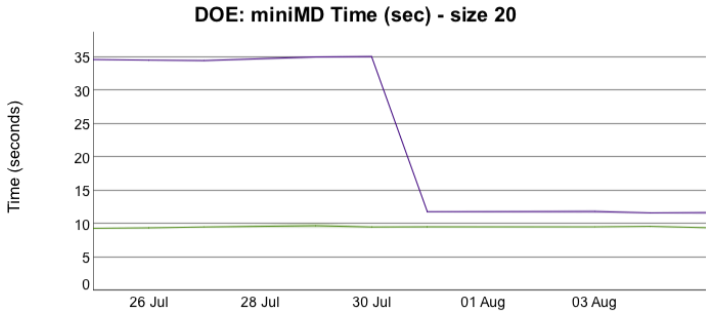
Impact: Significantly improved ‘gasnet’ aries performance

- Though still lagging behind ‘ugni’









‘gasnet’ Multi-domain: Next Steps

Next Steps: Continue to improve ‘gasnet’ comm performance

- Take advantage of dynamic registration for correct affinity
- Work with the GASNet team to explore other optimizations
- Track and explore GASNet-EX
  - add support for network atomics with GASNet when that comes online

Other Runtime Improvements

Other Runtime Improvements

- **Launcher changes:**
  - Added a gasnetrun_psm launcher for running on OmniPath
    - Contributed by Barry Moore
  - Fixed bugs in pbs-gasnetrun_ibv and slurm-based launchers
- **Retired ‘muxed’ tasking layer, deprecated in last release**

Other Third-party Improvements

Other Third-party Improvements

- Updated compiler to work with newer LLVM versions
- Switched LLVM back-end to use version 4.0.1 by default
- Updated GASNet to version 1.30.0
- Updated hwloc to version 1.11.8
- Updated GMP to version 6.1.2
- Updated RE2 to 2017-07-01
- Augmented third-party Makefiles to support auto-rebuild

Legal Disclaimer

Information in this document is provided in connection with Cray Inc. products. No license, express or implied, to any intellectual property rights is granted by this document.

Cray Inc. may make changes to specifications and product descriptions at any time, without notice.

All products, dates and figures specified are preliminary based on current expectations, and are subject to change without notice.

Cray hardware and software products may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Cray uses codenames internally to identify products that are in development and not yet publically announced for release. Customers and other third parties are not authorized by Cray Inc. to use codenames in advertising, promotion or marketing and any use of Cray Inc. internal codenames is at the sole risk of the user.

Performance tests and ratings are measured using specific systems and/or components and reflect the approximate performance of Cray Inc. products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance.

The following are trademarks of Cray Inc. and are registered in the United States and other countries: CRAY and design, SONEXION, and URIKA. The following are trademarks of Cray Inc.: ACE, APPRENTICE2, CHAPEL, CLUSTER CONNECT, CRAYPAT, CRAYPORT, ECOPHLEX, LIBSCI, NODEKARE, THREADSTORM. The following system family marks, and associated model number marks, are trademarks of Cray Inc.: CS, CX, XC, XE, XK, XMT, and XT. The registered trademark LINUX is used pursuant to a sublicense from LMI, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis. Other trademarks used in this document are the property of their respective owners.

CRAY

THE SUPERCOMPUTER COMPANY