Parcel Forwarding for AGAS in HPX

Vinay C Amatya[1,2], Bryce Adelstein-Lelbach[1], Maciej Brodowicz[1], Hartmut Kaiser[1,2]

[1]Center for Computation and Technology, [2]LSU Department of Computer Science

STE||AR
stellar.cct.lsu.edu
CENTER FOR COMPUTATION & TECHNOLOGY

Introduction

High Performance ParalleX (HPX) is the runtime implementation of ParalleX, a new model of computation targeting future generation of High Performance Computing Systems. HPX has been designed as an alternative to the conventional computation models such as MPI, attempting to overcome their limitations such as: global barriers, too coarse-grained parallelism and poor latency hiding capabilities. HPX is a modular, feature complete and performance oriented representation of ParalleX. Currently it has been developed on conventional architectures such as Linux or Windows based SMPs or Clusters. The current implementation of HPX provides the infrastructure for key ParalleX concepts: threads and their management, local control objects, parcel transport and parcel management and the active global address.

Threads and their Management: The HPX-thread managers implements a work queue based execution model, similar to prior systems like Cilk++, TBB. HPX-threads are first class objects with immutable global names, enabling remote management. Thread migration is avoided across localities as it is an expensive operation. Instead, actions are transferred to remote localities through parcels.

Local Control Objects (LCOs): An LCO is an abstraction of different functionalities for event-driven HPX-thread creation, protection of data structures from race condition and automatic event driven on-the-fly scheduling of work and synchronization of tasks.

Parcel Transport and Parcel Management: In HPX, parcels are extended form of active messages for inter-locality communication. Parcels enable invocation of action at a remote locality, take part in synchronization of remote tasks and provide means of transferring data.

Active Global Address Space (AGAS) is a fundamental concept in ParalleX execution model and is implemented as a part of the HPX runtime system. AGAS enables programming in distributed systems in a shared memory context. Each first class object in HPX is assigned a uniquely identifiable tag. AGAS maintains this tag as well as information about its locality, virtual address and component type throughout the duration of lifetime of the object. As such, core functionality of AGAS are: a) ascertain whether an object in reference is local or remote and b) to provide the global virtual address of the object in query (whether local or remote).

Parcel Forwarding: It is an incremental feature added in the HPX system that allows the AGAS service itself forward the parcels from the requesting locality to the destination locality. This feature would minimize the overall time taken for address resolution for a parcel's destination and its arrival at the destination.

Methods

In the previous implementation of AGAS, whenever there was a need for address resolution, the requesting locality would send a parcel to the AGAS server and the AGAS server would send a parcel back to the requesting locality with the resolved global virtual address (GVA). After receiving resolved address of the destination, the locality would then send the parcel to the final destination.

With Parcel Forwarding feature added, now the parcels could be sent directly to the AGAS server, and the server in turn would be able to forward the parcel to the final destination directly, hence minimizing the overall trip time of query-resolution-shipping of the parcel.

The Application of choice is ShenEOS (shen equation of state), that maintains tables of nuclear matter at finite temperature and density with various electron fractions within the relativistic mean field(RMF), in a set of three dimensional data arrays enabling high precision interpolation of 19 relevant parameters required for neutron star simulation. It is implemented in HPX, with a HPX component encapsulating the non-overlapping partitioning and distribution of the ShenEOS tables to all available localities, thus reducing the required memory footprint per locality.

Test-bed for the experiment is a heterogeneous cluster with two 48 core SMP systems and 15 quad core Xeon type workstations.
For the test, an iterative operation on the ShenEOS tables of huge size is performed. The partitioning of data would allow exchange of parcels across localities, which serves the purpose of our test.

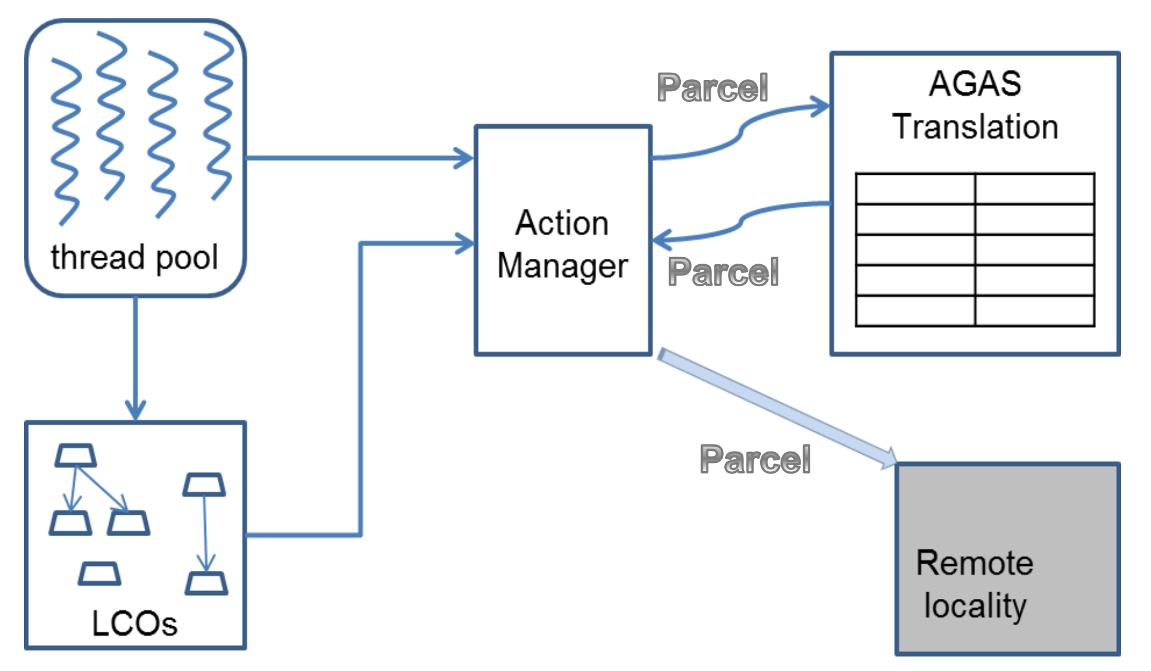

Fig. 1. Address resolution without parcel forwarding

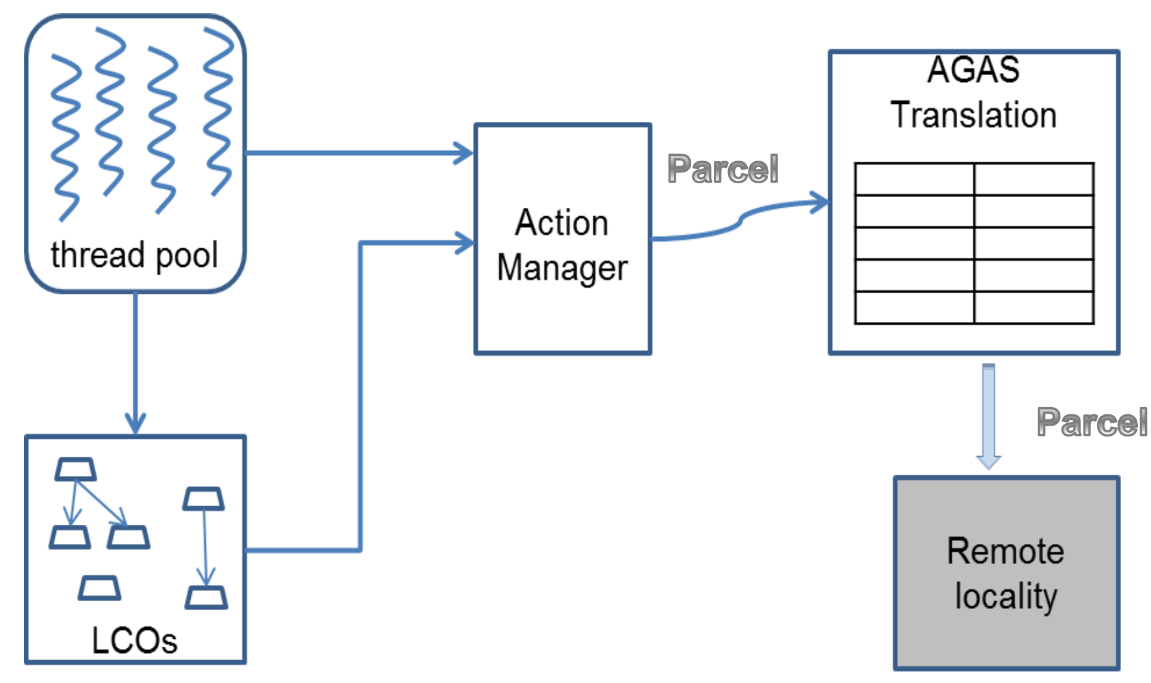

Fig. 2. Address resolution with parcel forwarding

Results

The test runs were performed with the local caching turned off so that more AGAS requests could be registered. The tests were performed for weak scaling with increasing resources as we increased work load.

The results of preliminary implementation of parcel forwarding shows slight speedup with parcel forwarding as compared with test run of the same application without parcel forwarding and every other test parameter remaining same. The results of the test runs are shown in the graphs below.

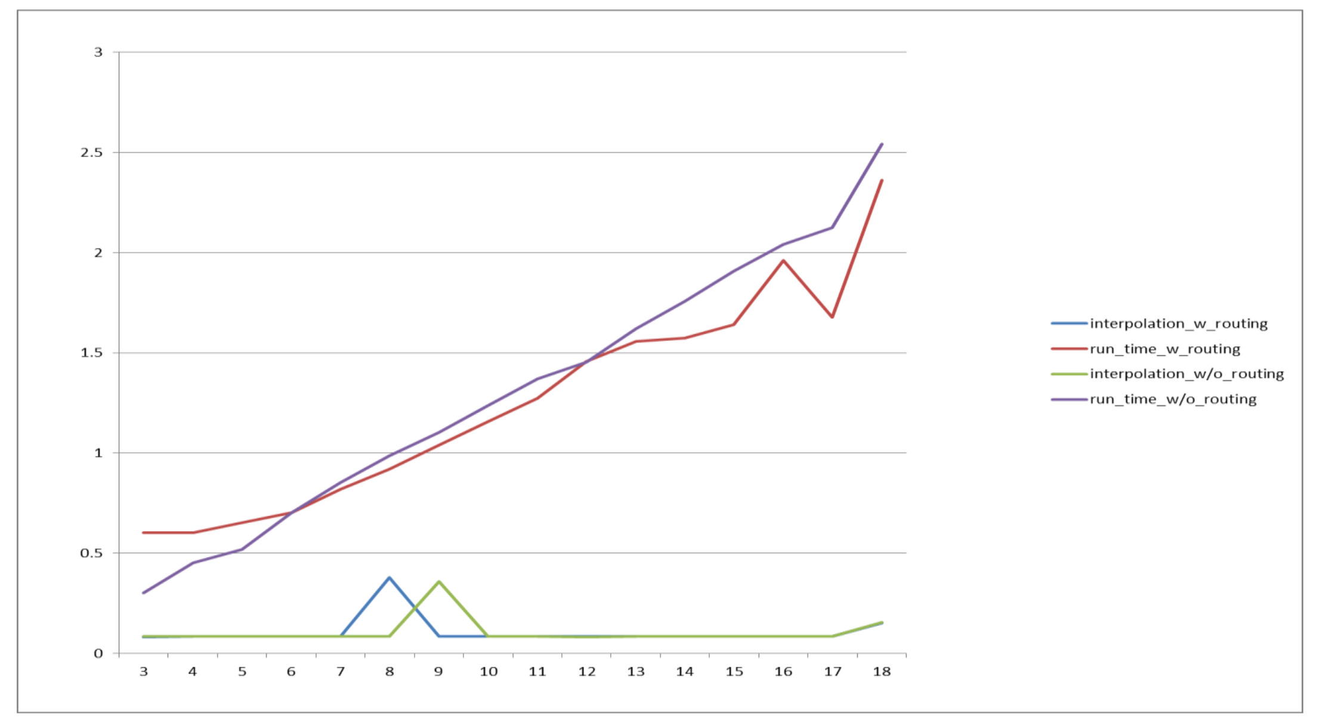

Figure 3: ShenEOS test run with 1024 runs, 4 partitions. Number of Nodes on X-axis and time in seconds in Y-axis.

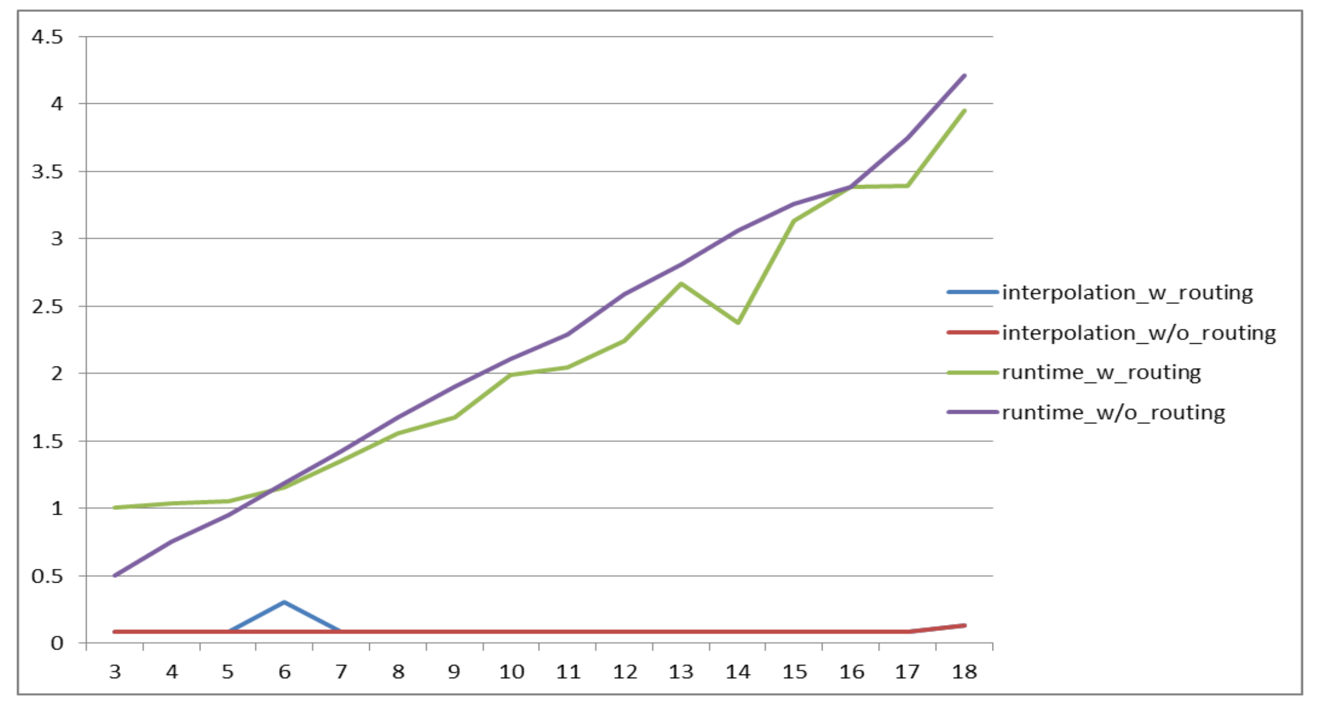

Figure 4: ShenEOS test run with 2048 runs, 4 partitions. Number of Nodes on X-axis and time in seconds in Y-axis.

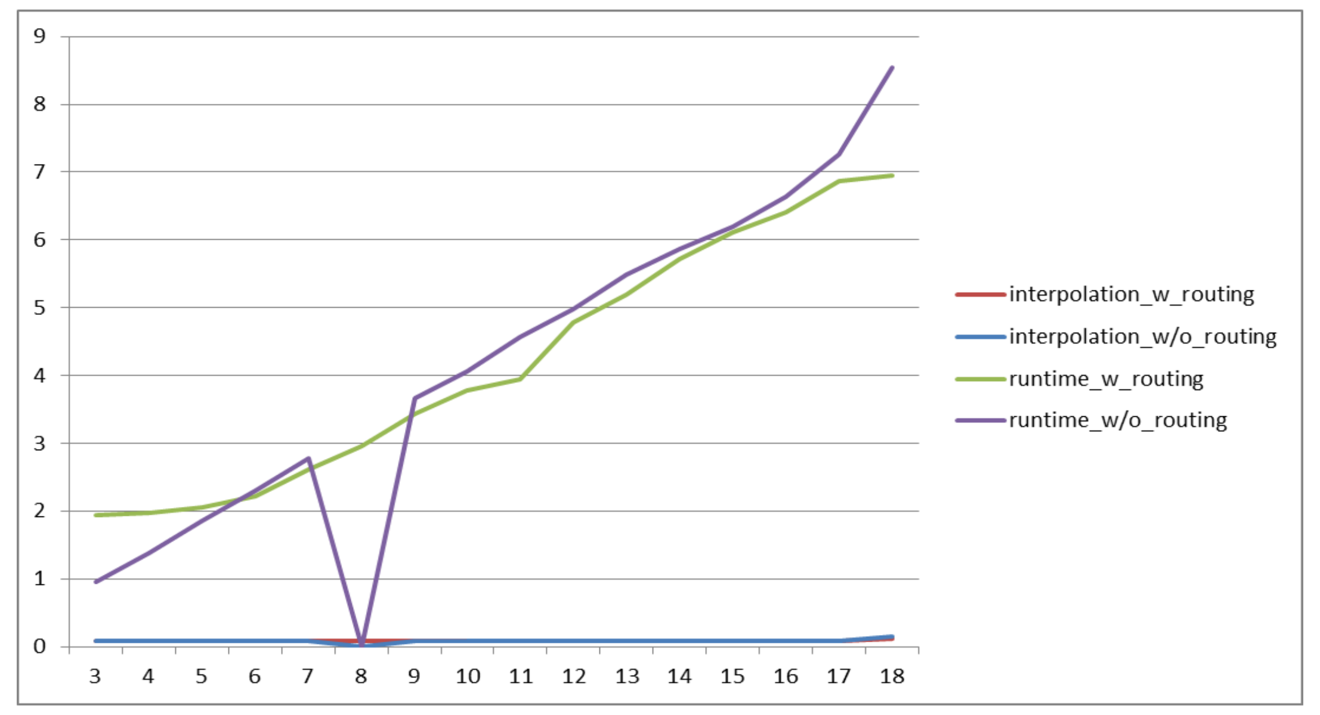

Figure 5: ShenEOS test run with 4096 runs, 4 partitions. Number of Nodes on X-axis and time in seconds in Y-axis.

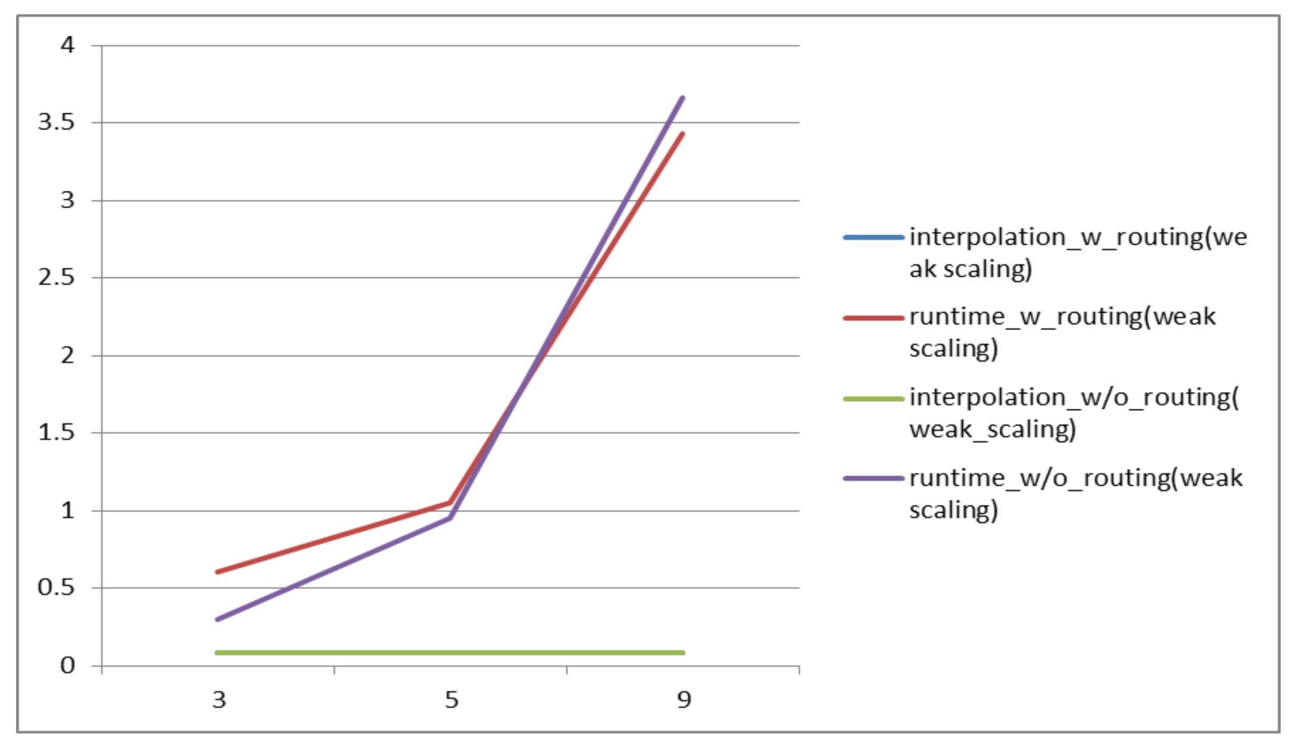

Figure 6: ShenEOS test run with two worker nodes with varying number of nodes and run iterations with fixed data partitions. Number of nodes on X-axis and time in seconds in Y-axis.

Conclusions

As a proof of concept, we can clearly see from the test results, that the ability of the AGAS service to forward parcel does minimize the round-trip time that is observed in the tests with parcel forwarding enabled. In essence, parcel forwarding cuts down the time spent to receive the parcel back to the requesting locality/node when an address resolution request is put to AGAS.

In the result, we observe an overhead with parcel forwarding enabled when there is less number of nodes involved. This overhead is due to extra cost of serialization and de-serialization of parcels at the action manager and the AGAS server side.

The current implementation does not consider the size of the parcel to be forwarded. Any parcel that needs address resolution is sent to the AGAS server, irrespective of the argument size. This may not be the optimal solution as there might be situations where parcel's size is too big for it to a) serialize and b) too huge to fit in memory of AGAS server. As an improvement to current implementation, an intelligent way of detecting parcel size before hand and thereafter triggering a faster way of transferring parcels such as RDMA could be implemented.

Acknowledgement:

- NSF Grants 1117470, 1048019, 1029161
- DARPA UHPC Funding
- LONI Allocation loni_hpx

Bibliography

1. T. Eicken, D. Culler, S. Goldstein, and K. Schauser. Active messages: A mechanism for integrated communication and computation. Computer Architecture, 1992. Proceedings., The 19th Annual International Symposium on, pages 256–266, 1992.
2. G. Gao, T. Sterling, R. Stevens, M. Hereld, and W. Zhu. Parallex: A study of a new parallel computation model. In Parallel and Distributed Processing Symposium, 2007. IPDPS 2007. IEEE International, pages 1–6, 2007. doi: 10.1109/IPDPS.2007.370484.
3. H. Kaiser, M. Brodowicz, and T. Sterling. ParalleX: An advanced parallel execution model for scaling-impaired applications. In Parallel Processing Workshops, pages 394–401, Los Alamitos, CA, USA, 2009. IEEE Computer Society. doi: http://doi.ieeecomputersociety. org/10.1109/ICPPW.2009.14.