**UPC++ at Lawrence Berkeley National Lab**

UPC++ is a library providing lightweight PGAS one-sided communication and asynchronous remote function execution with a C++ interface.

- UPC++ is a C++11 library
- No custom compiler
- Easy on-ramp and integration
  - Interoperable with MPI+OpenMP/CUDA/etc
  - Enables incremental development
  - Replace performance-critical sections with lightweight PGAS
- New extensions under development
  - Co-processor memory support, non-contiguous communication, teams, and active message interfaces

**Case 1: Easy distributed hash-table via function shipping and futures**

- **Function shipping** simplifies distributed data-structure design
  - Use a GASNet Active Message to ship updates to the key's owner, avoiding round trip communication
- **Futures** hide the latency of remote operations, naturally express overlap of independent operations

```cpp
// c++ "global" variables become rank-local state.
std::unordered_map<int, int> _dht_local;

// owner does the work, result is a future<int>
upcxx::future<int> dht_fetch_inc (int key) {
    return upcxx::ship_function(
        key % upcxx::rank_count(),
        [=]() { return atomic_fNlncri_dht_local[key]; }
    );
}
```

- **Impact**: on average, symPACK delivers a ×2.65 speedup over the best state-of-the-art sparse symmetric solver.

**Case 2: symPACK: UPC++ asynchronous task-based sparse Cholesky solver**

- **Application**: symPACK, a sparse direct linear solver for symmetric matrices.
- **Challenges**: Sparse matrix factorizations have low computational intensity and irregular communication patterns.
- **Solution**: UPC++ function shipping enables an efficient pull communication strategy and event-driven scheduling.
- **Impact**: symPACK delivers a ×2.65 speedup over the best state-of-the-art sparse symmetric solver.

UPC++’s one-sided pull strategy avoids the need for (and cost of) unexpected messages in MPI.

**Impact of communication strategy for boneS10**

**Run times for Flann 1565 on NERSC Edison**

---

UPC++ and GASNet: PGAS Support for Exascale Apps and Runtimes

**Pls: Scott B. Baden and Paul Hargrove**

1.3.1.17

© 2017, Lawrence Berkeley National Laboratory