UNIVERSITY of HOUSTON

PARALLEL SOFTWARE TECHNOLOGIES LABORATORY PSTL

Maximizing Communication-Computation Overlap through Automatic Parallelization and Run-Time Tuning of Non-blocking Collective Operations

Youcef Barigou and Edgar Gabriel
Parallel Software Technologies Laboratory, Department of Computer Science,
University of Houston, Houston, TX

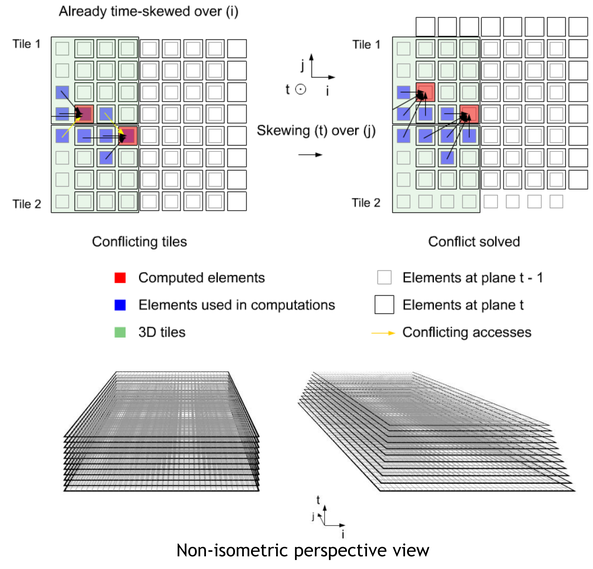
Non-blocking Collective Operations

- Non-blocking Collective Operations are essential for scalability of HPC applications by
  - Abstracting out group communication operations from implementations
  - Enabling communication-computation overlap
- **Main goal**: facilitating <u>efficient</u> integration of NBCs for end-users, while ensuring:
  - **Applicability**: the scope of targeted HPC applications
  - **Portability**: breadth of appropriate architectures
  - **Effectiveness**: potential ground for communication-computation overlap

Factors Affecting NBCs Performance

- **Overlapping code sections**
  - Producing maximal overlapping compute-intensive blocks
  → Depends on <u>parallelization strategy</u>
- **Underlying algorithms**
  - Selecting optimal sequence of underlying p2p operations
  - Run-time Auto-tuning to determine fastest algorithm
  → Auto-tuning still influenced by <u>parallelization strategy</u>
- **Progression**
  - Determining optimal frequency and placement of calls
  → Flexibility depends on <u>parallelization strategy</u>
- **Overlapping NBCs**
  - Tuning interfering NBCs as a whole (co-tuning)
  → Occurs as a result of <u>parallelization strategy</u>

PLUTO - Automatic Paralellizer and Code Generator

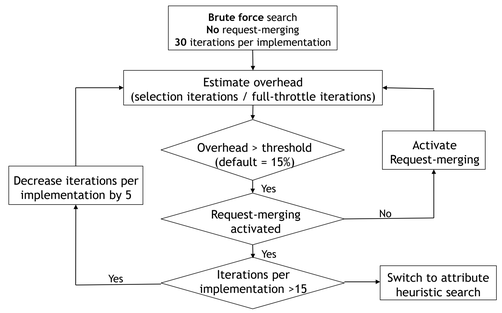


PLUTO-ADCL

```
// Preliminary declarations (buffers, ...)
ADCL_Request ...;
ADCL_Timer timer;
...
// Initialize non-blocking persistent collective operation
ADCL_Ialltoall_init (...);
...
for (...) { // Main code loop
  // Backup meta-buffers of current iteration
  ...
  //Start timer
  ADCL_Timer_start (timer);
  // Start meta-communication operation of next iteration
  ADCL_Request_init (...);
  #pragma omp parallel for ...
  for (...) { // Iterates over data-independent process tiles
      for (...) { // First dimension in tile
        ...
      }
    ...
  }
  MPI_Waitall(...);
  // Unpacking exchanged data
  ...
  #pragma omp parallel for ...
  for (...) { // Iterates over data-dependent process tiles
      for (...) { // First dimension in tile
        ...
      }
    ...
  }
  // Wait for completion of meta-communication
  ADCL_Request_wait (...);
  // Stop timer
  ADCL_Timer_end (timer);
  // Restore meta-buffers of current iteration
  ...
  // Packing routines for MPI pseudo-neighborhood communication
  ...
  for(...){
    MPI_Isend(...); // Pseudo-neighborhood send to each neighbor
    MPI_Irecv(...); // Pseudo-neighborhood receive from each neighbor
  }
}
```

Progress Calls Auto-tuning

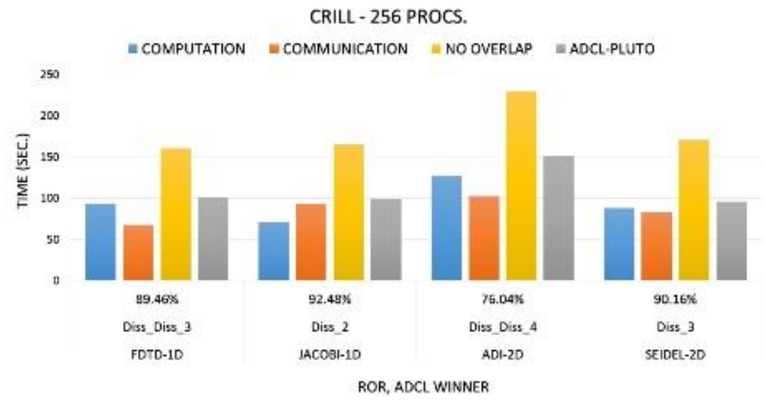
Automatic ADCL configuration



Experimental Results

- Four 2.2 GHz 12-core AMD Opteron cores per node
- 64 GB of main memory per node
- Two 4x DDR InfiniBand Host Channel Adapters per node
- Open MPI 1.8 – PLUTO 0.11 – ADCL 2.0

Benchmark	Process counts	Time iterations	Space size	% of ramp-up & -down	% of ADCL Selection	Tile-occupancy	Optimal tile size
FDTD-1D	256	$200K$	$20M$	3.86%	5.64%	4	$256 * 256$
Jacobi-1D	256	$200K$	$20M$	5.73%	0.47%	4	$256 * 256$
ADI-2D	256	$40K$	$4K * 4K$	7.32%	17.34%	3	$3 * 3 * 3$
Seidel-2D	256	$40K$	$4K * 4K$	7.33%	1.73%	3	$3 * 3 * 3$



References

[1] Maximizing Communication-Computation Overlap through Automatic Parallelization and Run-Time Tuning of Non-blocking Collective Operations - Youcef Barigou and Edgar Gabriel. *Submitted to The International Journal of Parallel Programming. March, 2016*

[2] Auto-tuning Non-blocking Collective Communication Operations - Youcef Barigou, Vishwanath Venkatesan and Edgar Gabriel. *Accepted for publication at The International Workshop on Automatic Performance Tuning (iWAPT), held in conjunction with the 29th IEEE International Parallel & Distributed Processing Symposium (IPDPS). Hyderabad, INDIA, 2015*