Sparse Non-Blocking Collectives in Quantum Mechanical Calculations

Torsten Hoefler[1], Florian Lorenzen[2]
and Andrew Lumsdaine[1]

[1]Indiana University
[2]Freie Universität Berlin

EuroPVM/MPI 2008
Dublin, Ireland

9th September 2008

Ab-Initio Calculations

- important for nano and material sciences
- model physical processes with Schrödinger equation
- DFT is computationally feasible method to solve it
- solve static and time-dependent Kohn-Sham equations:

$$H\varphi_j = \varepsilon_j \varphi_j \qquad i\frac{\partial}{\partial t}\varphi_j(t) = H(t)\varphi_j(t)$$

The Problem

- two Eigenvalue problems:

  $H\varphi_j = \varepsilon_j \varphi_j$ is solved by iterative diagonalization

  $i\frac{\partial}{\partial t}\varphi_j(t) = H(t)\varphi_j(t)$ is solved by evolving $\varphi_j(t)$ in time

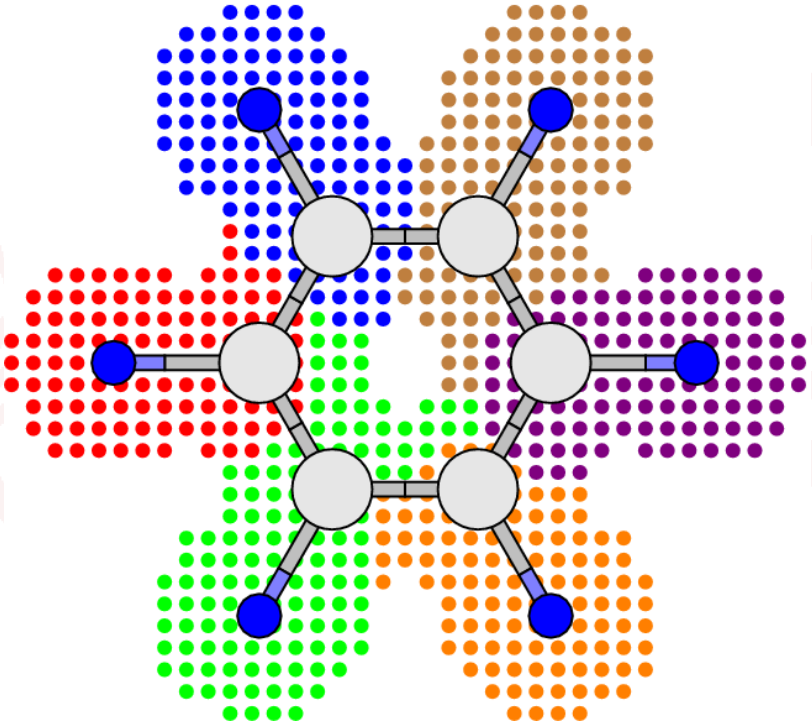
- H is the Hamiltonian operator:

$$H = -\frac{1}{2}\nabla^2 + V$$

- where V is the atomic potential

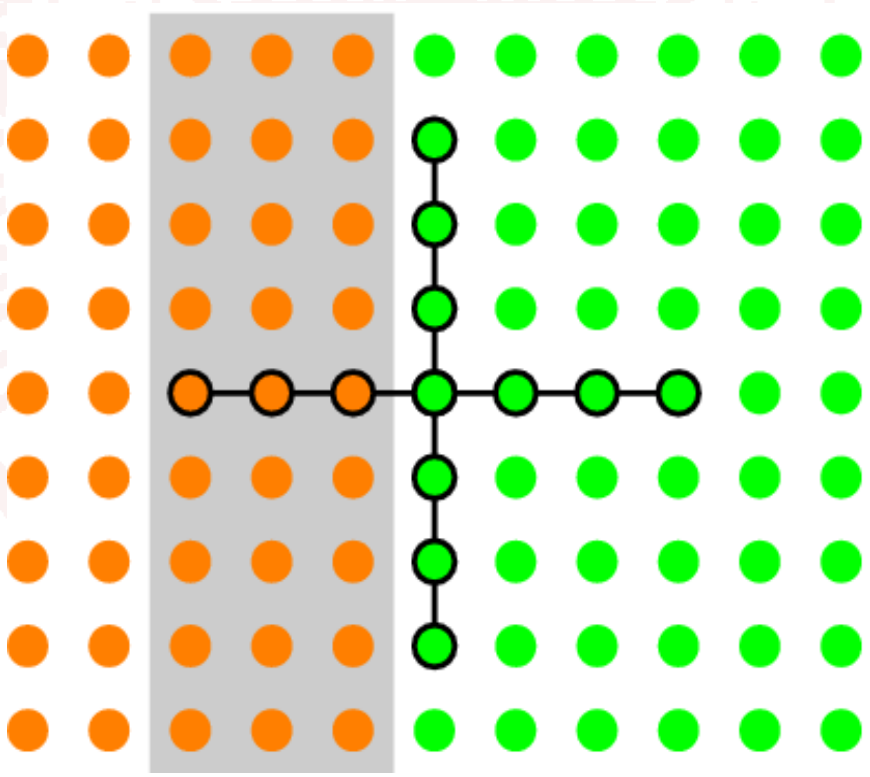
Representing the Problem

- essential operation is $H\varphi_j$
- φ is represented as finite difference grid in 3d-domain
- domain decomposition by distributing φ



Representing the Hamiltonian

- potential V is diagonal matrix, $V\varphi_j$ can be applied locally
- ∇^2 needs a finite difference stencil (exchange boundaries)



Application of $H\varphi_j$

1. exchange boundary values
2. apply kinetic energy operator $-\frac{1}{2}\nabla^2$
3. apply potential $\varphi + V\varphi$

We analyze communication of boundary values!

(Other communications (e.g. check for convergance) remain untouched!)

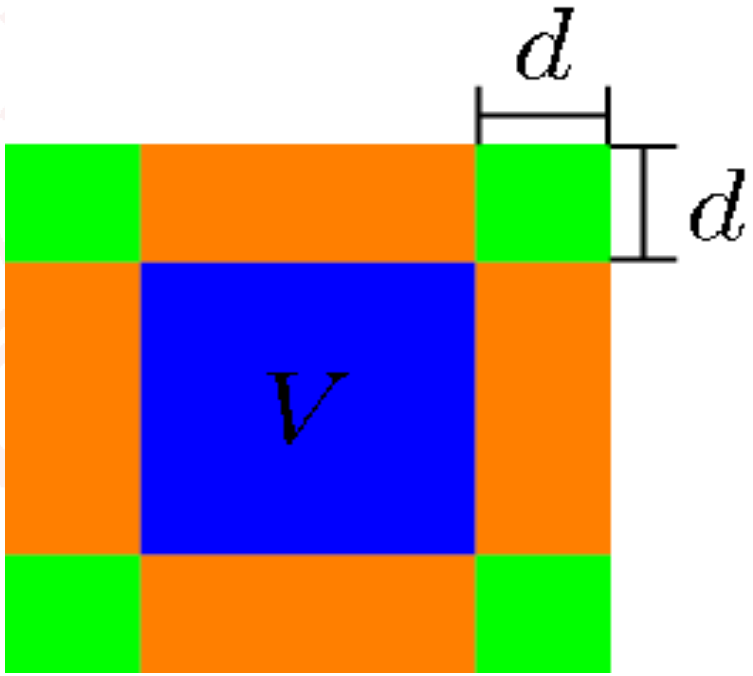
Parallel Implementation

- METIS is used to obtain ”balanced” partition
- communication overhead of $H\varphi_j$ is dominated by nearest neighbor exchange
- assume NP points, nearly optimal decomposition
- P processors - N points per process
- ”volume” per node V = N
- points to be communicated are in: $V_d - V$

$$p(P) = \alpha d^3 + \beta d^2 \sqrt{V(P)} + \gamma d \sqrt[3]{V(P)^2}$$

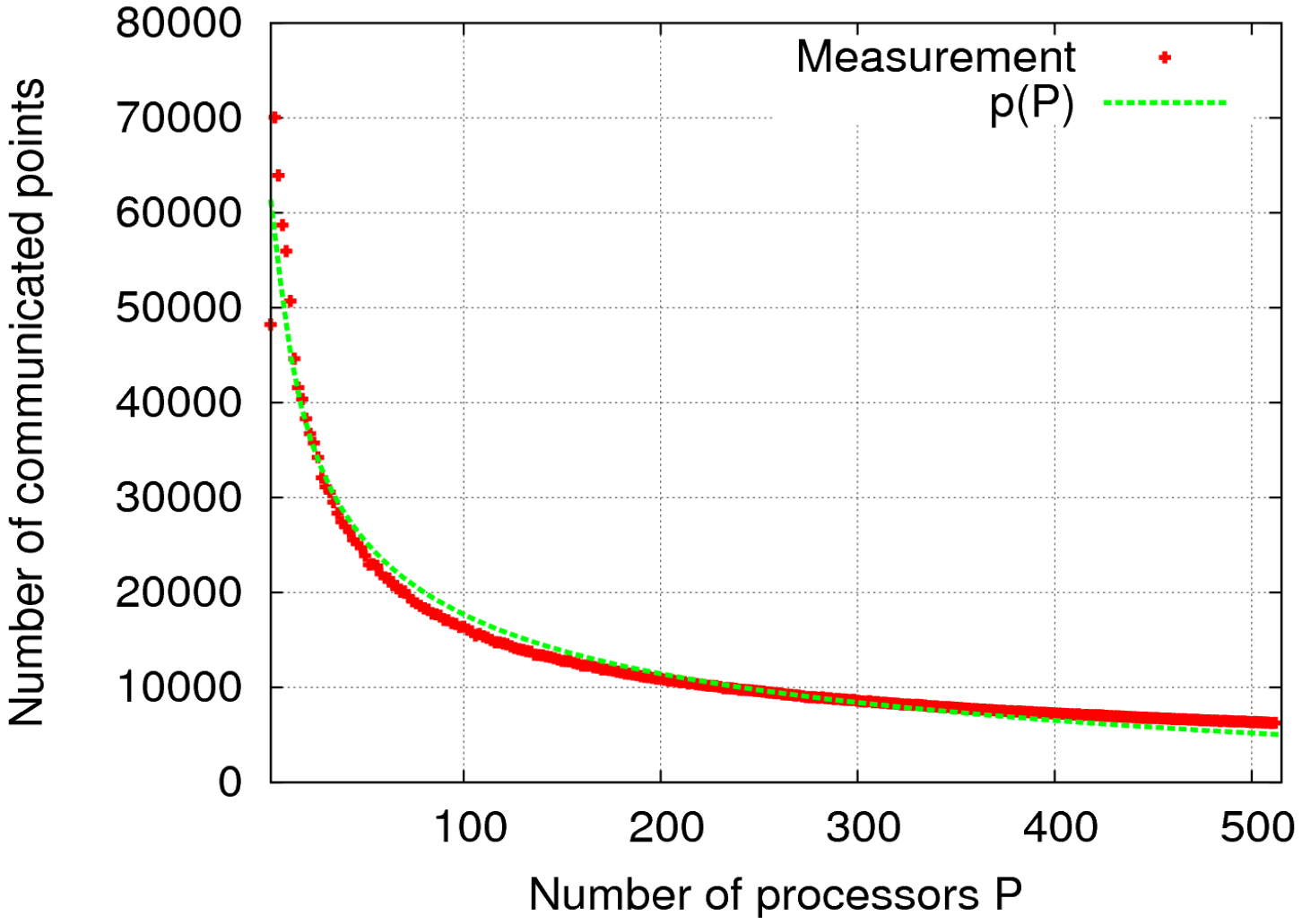
Example

- 2-dimensional simplified (square) example
- extending V by d
- proportional to d^2 (red) and $d\sqrt{V}$
- 3rd dimension would add $d\sqrt[3]{V(P)^2}$



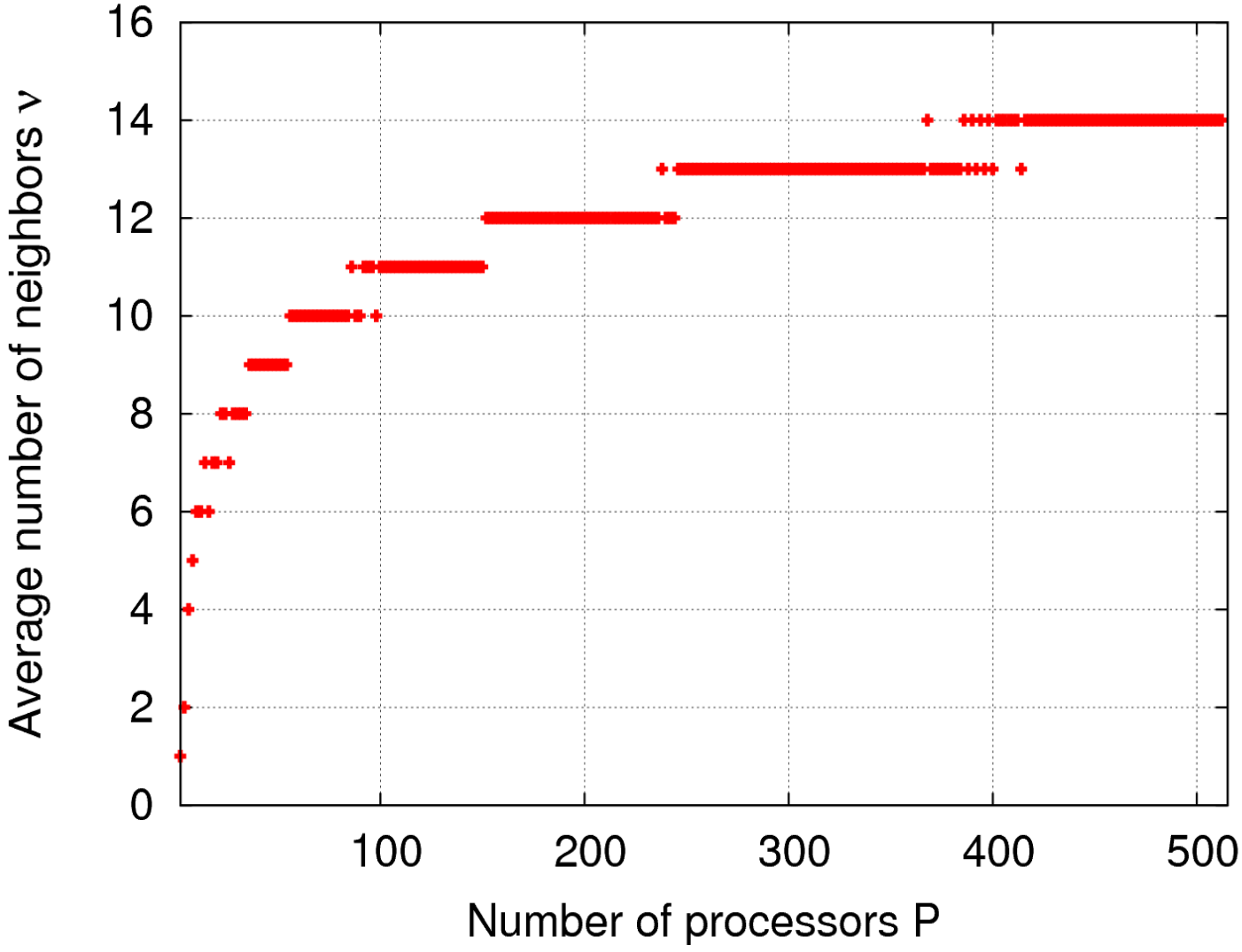
Model and Reality

- fitting Li+H measurement to equation



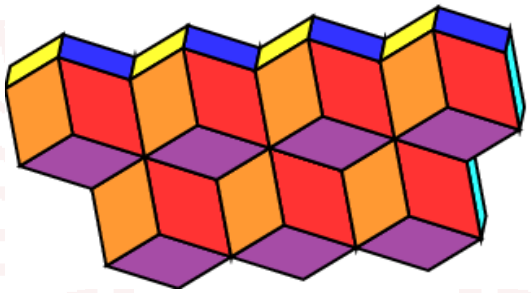
Number of Neighbors

- second important parameter: number of neighbors



Computation and Communication Complexity

- number ν stays clearly below P for large P
- for truly minimum surfaces – maximum of 12 neighbors



- applying LogGP model to assess overhead:

$$t_{comm} = L + o\nu + g(\nu - 1) + G(\nu * 8\mathrm{p}(P))$$

- computation and communication have similar scaling!

Practical Considerations

- benchmarking Octopus on 16 processors
  - 13% overhead due to 8-bytes MPI_Allreduce
  - 8.2% overhead due to MPI_Alltotallv
- what can we optimize?
  - neighbor exchange can be overlapped
  - eliminate MPI_Alltoallv but retain collectives

Non-blocking Collectives

- trivial change from MPI_Alltoallv
- move NBC_Wait as far back as possible:

1. exchange boundary values - NBC_Ialltoallv()
2. apply kinetic energy (inner points) $\varphi_j - \frac{1}{2}\nabla^2\varphi_j^{inner}$
3. apply potential $\varphi + V\varphi$
4. NBC_Wait()
5. apply kinetic energy (edge points) $\varphi_j - \frac{1}{2}\nabla^2\varphi_j^{outer}$

Problems with Alltoallv

MPI ALLTOALLV(SBUF, SCOUNTS, SDISPLS, STYPE, RBUF, RCOUNTS, RDISPLS, RTYPE, COMM, IERROR)

- maximum number of neighbors is limited
- Alltoallv assumes rather dense communication
- four P-element arrays need to be written/read
- obvious scalability/memory overhead issues
- hard to program (store/build arrays)

Topological Collectives

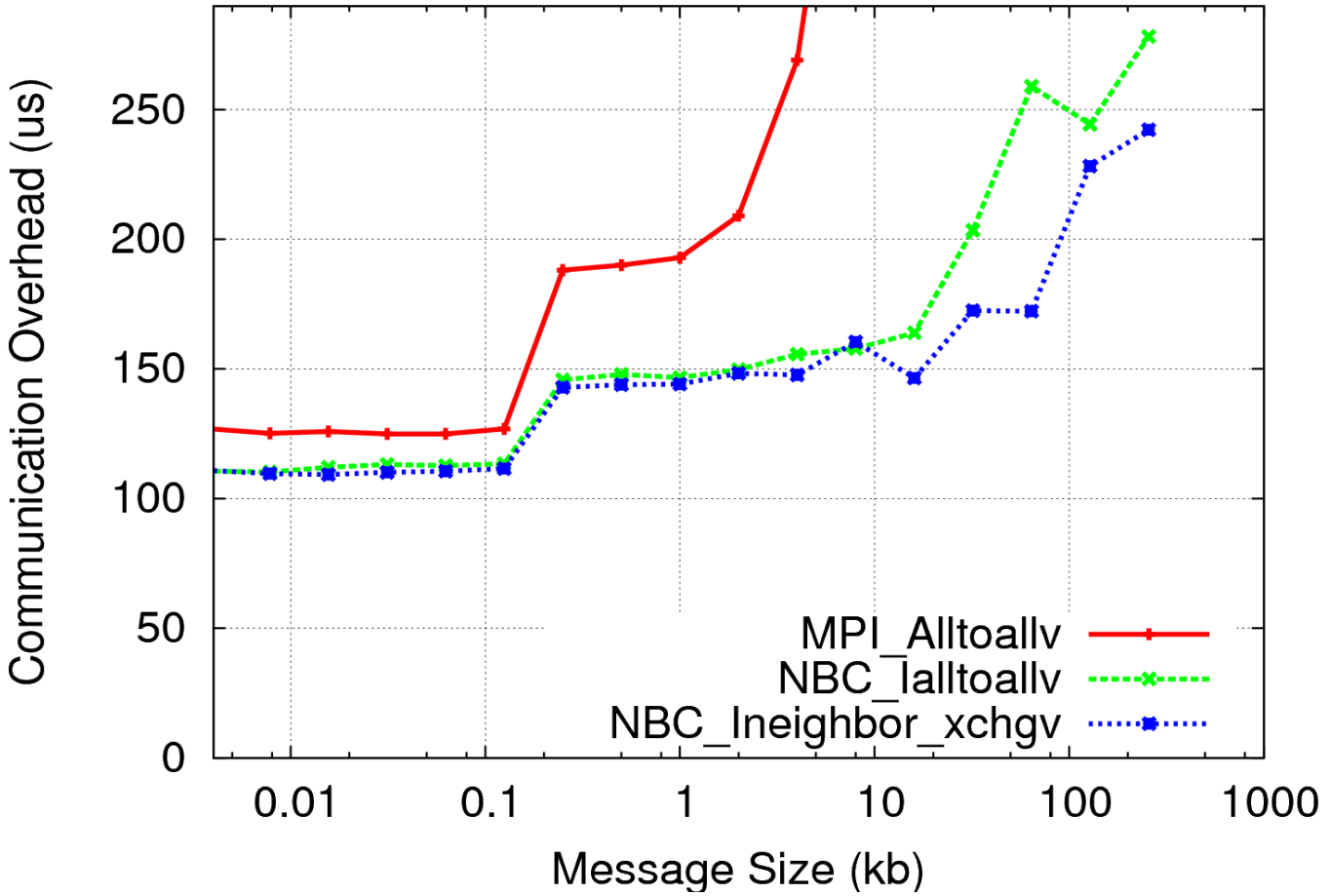
NBC_NEIGHBOR_XCHG(SBUF, SCOUNTS, STYPE, RBUF, RCOUNTS, RTYPE, COMM, IERROR)

- sends and receives data only with neighbors
- counts-array have only nneighbors elements
- no scalability problem
- needs to create graph topology (easy with METIS)
- graph enables reordering and other optimizations
- extension of a well-known MPI topology feature
- MPI-1 graph interface has scalability issues

  → simple MPI-2.2 proposal to fix them

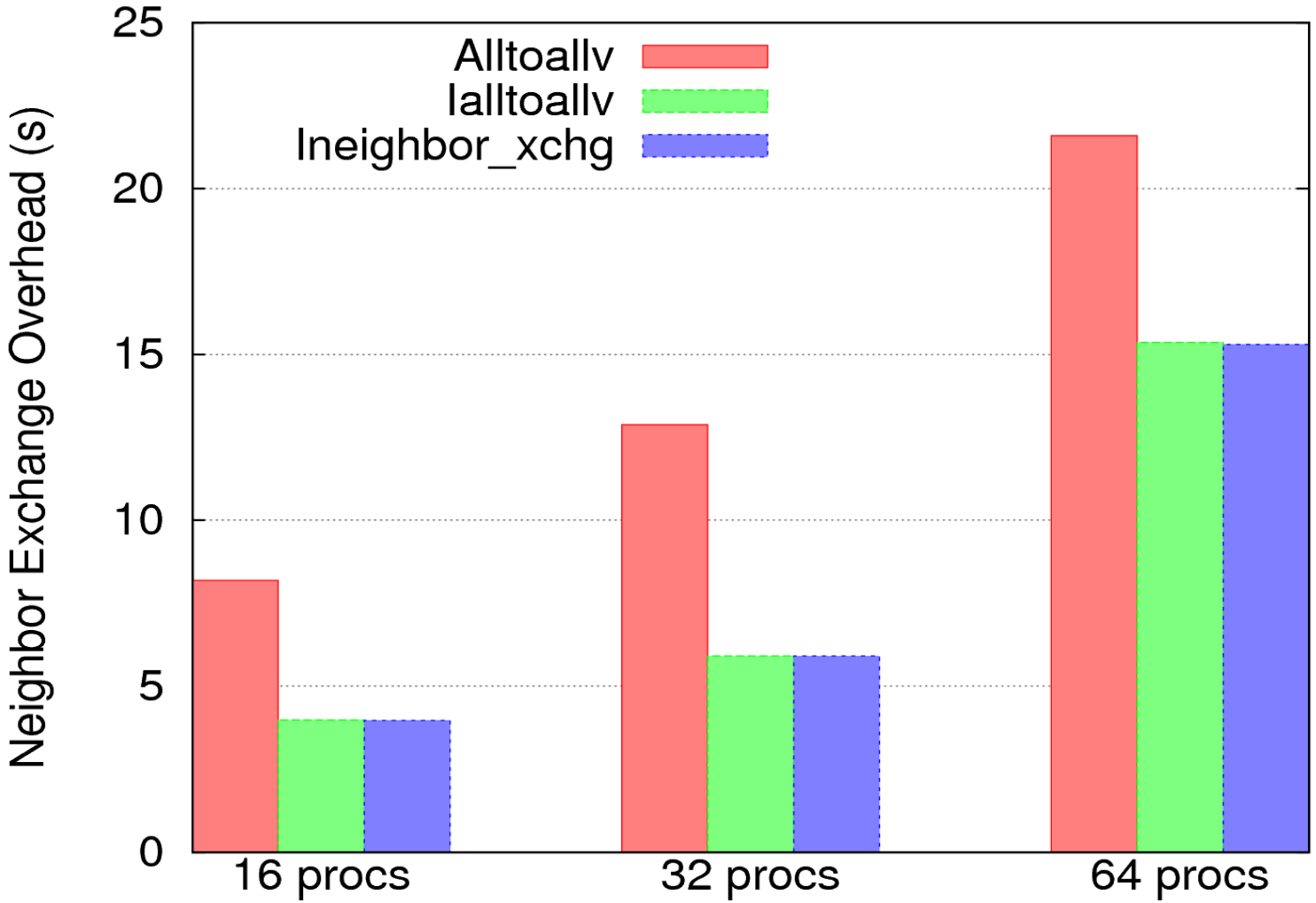
Performance - Microbenchmarks

- 64 processes on 16 nodes (4 cores per node)
- LibNBC 1.0 (IB optimized, not threaded)



Application Performance

- ground state calculation of Lithium and Hydrogen atoms
- overhead varies between 22% and 25% on 4-16 nodes



Conclusions

- application of non-blocking techniques halves neighbor exchange communication overhead
- graph interface seems like a good way to express scalable neighbor communication operations
- graph topologies have many unused optimization possibilities
- eight-byte MPI_Allreduce remains a problem in the application (dominates quickly)
- new collective techniques are promising and simplify programming