Sparse Triangular Solve in UPC

By Christian Bell and Rajesh Nishtala

Motivation

- A common but irregular mathematical operation occurring in linear algebra is the Sparse Triangular Solve (SpTS).
  - Solve for x in $Tx = b$ where T is a lower triangular sparse
  - Used after sparse Cholesky or LU factorization to solve sparse linear systems
- Irregularity arises from dependence
- Hard to parallelize
  - dependence structures only known at runtime
  - must effectively build dependence tree in parallel

Algorithm Description

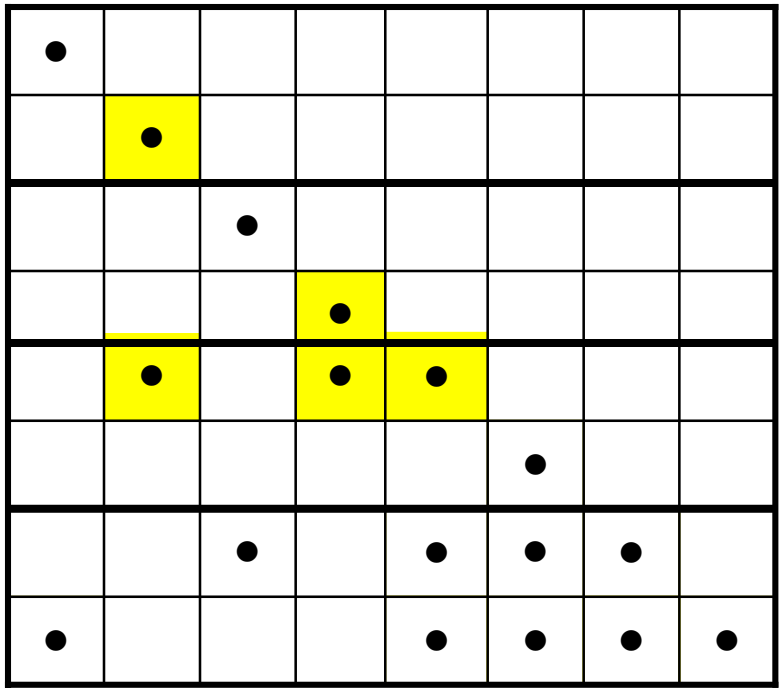
- To solve for x in Tx = b (T is lower triangular)

```
for r=1:n {
   x(r) = b(r);
   for c=1:r
      x(r) = x(r) - T(r,c)*x(c);
}
```

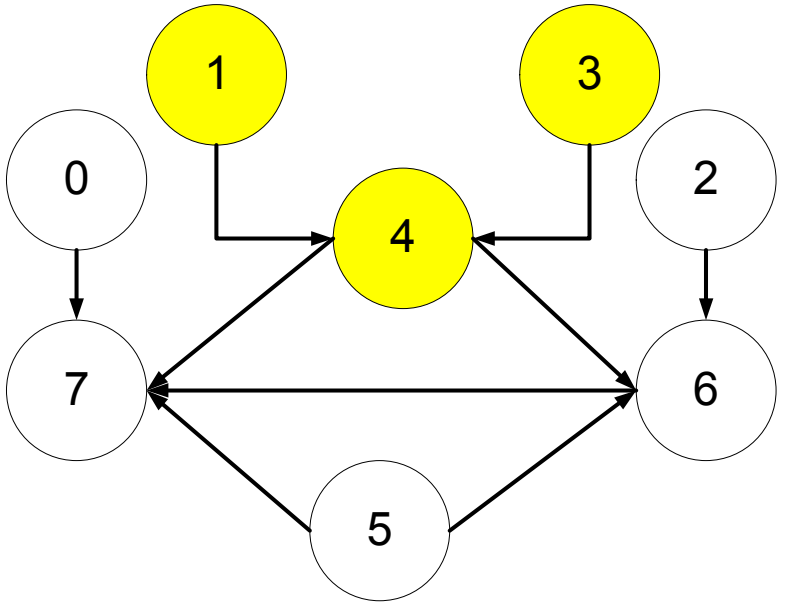
- Key takeaways
  - To solve x_r you depend on all values of x before it
  - rows can be partially solved by knowing which values of x_c are valid

Dependency Graph

Matrix



Dependence Graph



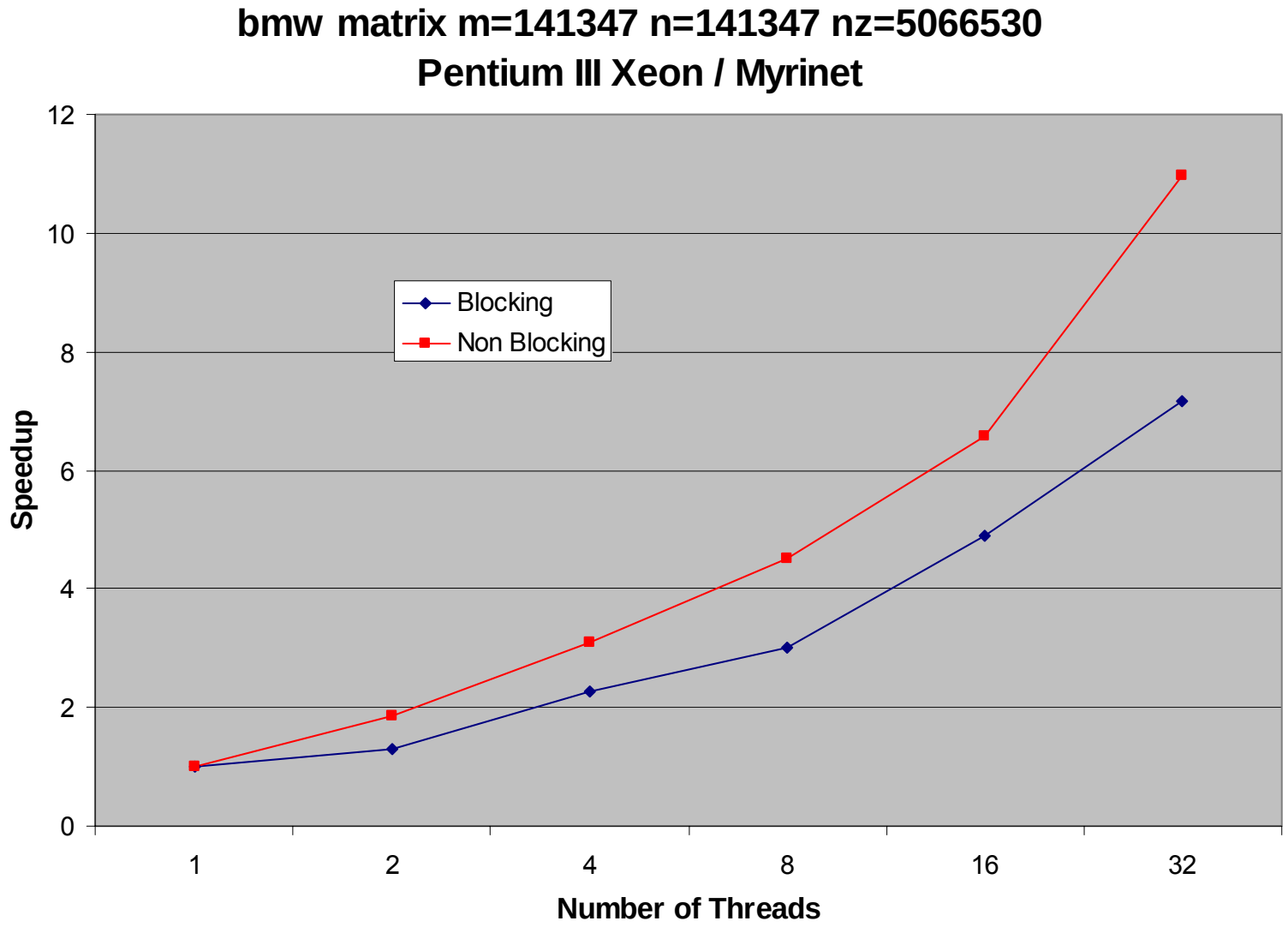
Data Structure Design

- Allow more startup time analysis of matrix so that the solve is faster
- Build the dependence graph in parallel
- Support O(1) lookup during solve time
- O(1) operations made easy by UPC

Solve Methodology

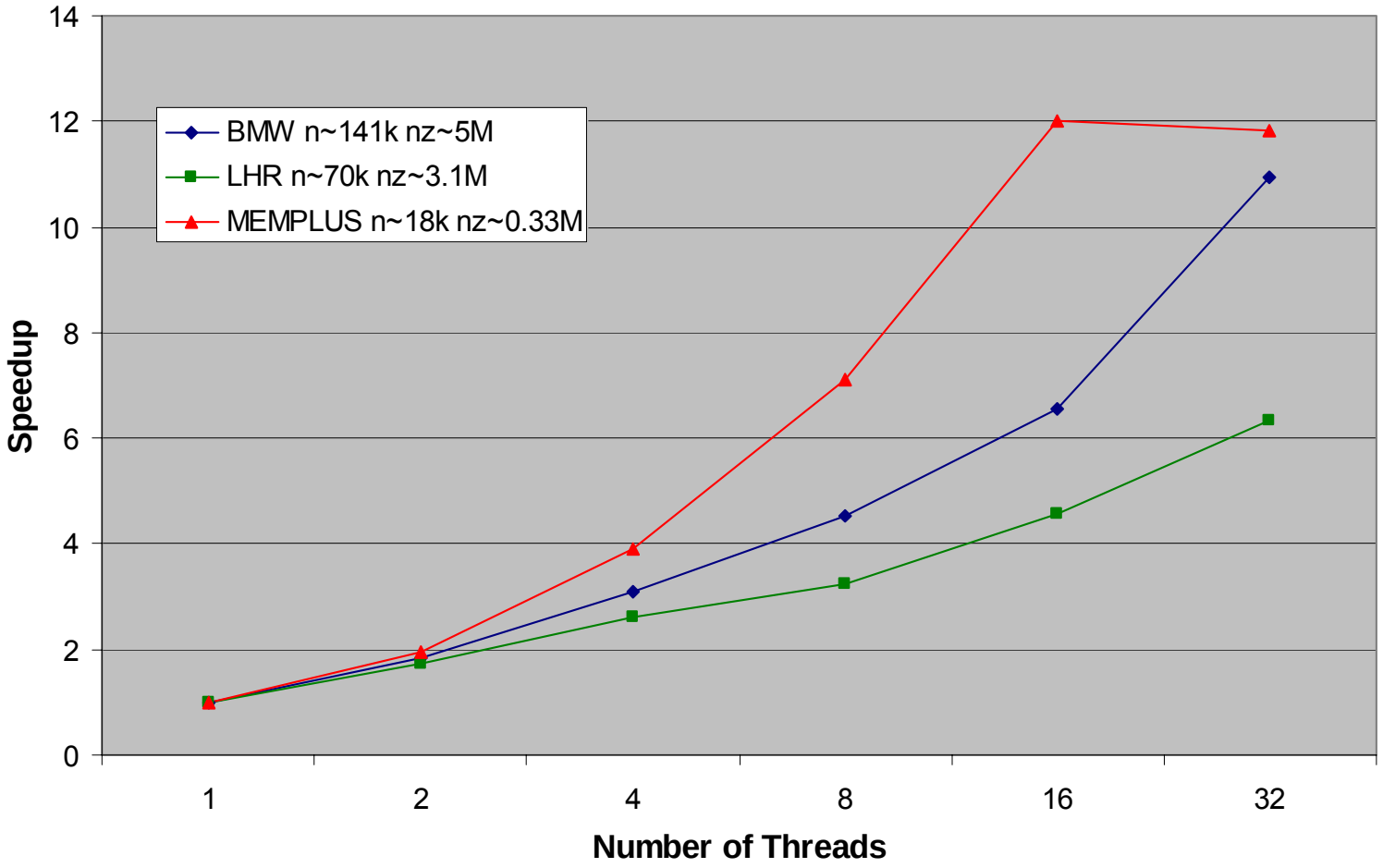
- Producer / Consumer Relationship
  - "consume" x vector in Tx = b to produce a new x_j variable.
  - "production" causes generation of signal to every processor waiting on x_j
- Difficult with two-sided model of MPI
  - allows you to effectively leverage one-sided communication available in UPC
- Avoid synchronization
  - by knowing a priori what part of other threads address space you can safely write into.
  - very difficult to get right through MPI

Performance (1)



Performance (2)

Speedups Across Matrices (Pentium III 866MHz/ Myrinet cluster)



Conclusions and Future Work

- A new style of programming for an old problem
- Leverage one-sided messaging not easily available in MPI
- Integrate into libraries such as SuperLU