Idaho National Engineering and Environmental Laboratory

Performance Analysis and Optimization of a Deterministic Radiation Transport Code on the Cray SV1

Peter Cebull

Advisory Engineer

May 20, 2004

Outline

- *Background*
- *Description of Attila*
- *Initial analysis and optimization*
- *Final analysis and optimization*
- *Speedups achieved*
- *Platform comparison & parallel scalability*
- *Conclusions*

Background

- *INEEL to become INL, research center for NE*
- *Advanced Test Reactor (ATR)*
- *Motivation*
  - *Renewed emphasis on high scientific computing*
  - *Acquisition of 3 Cray SV1s from NERSC*
  - *Need to free up compute cycles*
  - *Move appropriate applications to Crays*
  - *Attila models being run on Opteron-based PC*

Attila

- *Deterministic radiation transport code*
- *3-D discrete ordinate code*
- *Uses unstructured tetrahedral mesh*
- *Developed at LANL, marketed by Radion Technologies*
- *~45,000 lines Fortran 90*

Attila – Basic Structure

```
SUBROUTINE OUTER

DO until converged
   DO 1, NGROUPS
          .
          .
      CALL INNER
          .
          .
   END DO
END DO
```

```
SUBROUTINE INNER

DO until converged
          .
          .
   CALL SOLVE_FO
          .
          .
   CALL DSA
          .
          .
END DO
```

Initial Performance Analysis

- NEACRP benchmark case
  - 1/8th reactor core, 2 energy groups, 24 angles
  - 3,932 elements
- Initial unoptimized run with perftrace enabled
  - Only achieved 21.8 MFLOPS
  - MFLOPS/MIPS ratio of 0.27
  - 83.9% time spent in SOLVE_FO
  - 6.6% time in CGD and DSA combined

Subroutine SOLVE_FO

```
SUBROUTINE SOLVE_FO

DO over angles
  DO over sweeps in angle

    DO 1-side visible cells
      CALL LU4
    END DO

    DO 2-sides visible cells

    DO 3-sides visible cells

  END DO
END DO
```

SOLVE_FO Optimizations

- *3 inner loops not vectorizing*
  - *Subroutine calls:* `CALL LU4(ier, AMAT, IB)`
  - *Recurrences:* `PSI(4), AMAT(4,4), IB(4)`
- *Solution*
  - *Index arrays by loop counter, add* `!$CONCURRENT`
  - *Pass sections to LU4:* `AMAT(:,:,i), IB(:,i)`
- *Memory stride issues*
  - *Strides of 4, 16*
  - *Solution: make leading dimensions odd*
- *Results: SOLVE_FO went from 22.2 to 55.0 MFLOPS*

Final Performance Analysis

- *Large ATR model*
  - *4 energy groups, 24 angles*
  - *2,528,838 elements*
- *Another* perftrace *run with previous optimizations*
  - *CGD consumes 86.6% of CPU time!*
  - *Work pushed into conjugate gradient solver*
  - *SOLVE_FO uses 9.6%, DSA 1.8%*

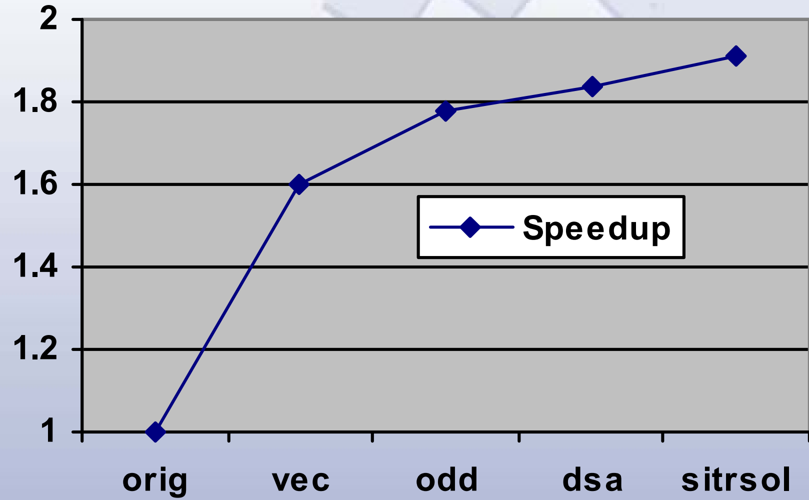
DSA and CGD Optimizations

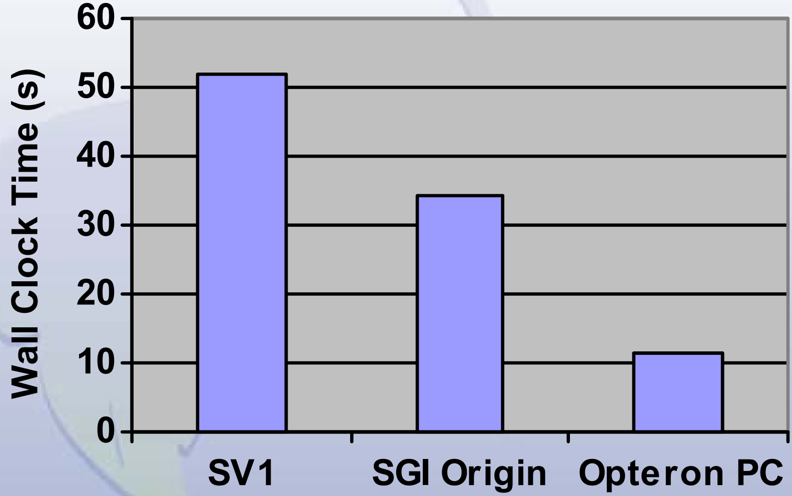
- *DSA: two loops over `ncells`, separated by CGD call*
  - *First loop inhibited by gather-scatter, subscript collisions*
    - *Split into separate loop*
  - *Second loop similar to SOLVE_FO*
    - *Add loop index to arrays with recurrences*
- *CGD: preconditioned conjugate gradient solver*
  - *Does not vectorize as written: 21.8 MFLOPS*
  - *Replace with call to SITRSOL in Scientific Library*
  - *Order of magnitude speedup!*

Final Results

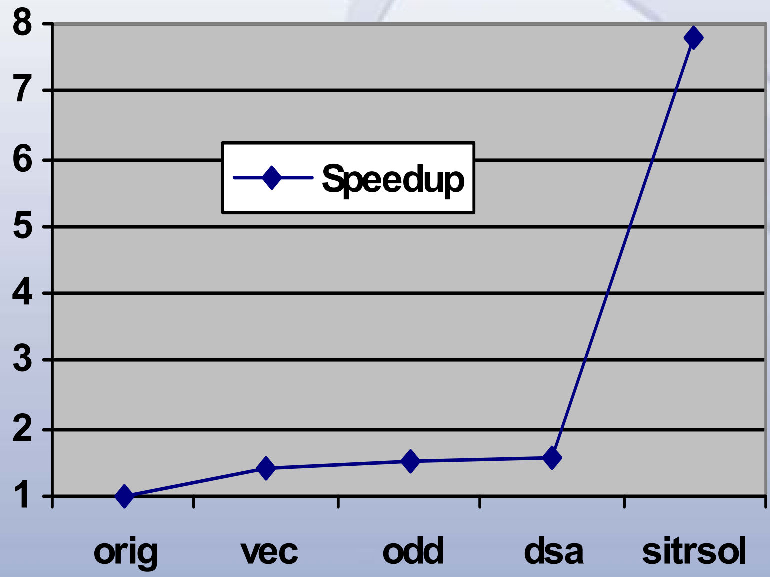
- NEACRP model shows modest speedup
  - Wall clock time: 99.3 s to 52.1 s
  - 21.8 MFLOPS to 59.5 MFLOPS
  - SGI Origin and Opteron PC still faster
- ATR model shows much better improvement
  - Wall clock time: 6.4 days to 19.8 hrs
  - 22.3 MFLOPS to 136.6 MFLOPS
  - Slightly faster than Opteron at 21 hrs
  - Not X1-level performance, but not too bad, either

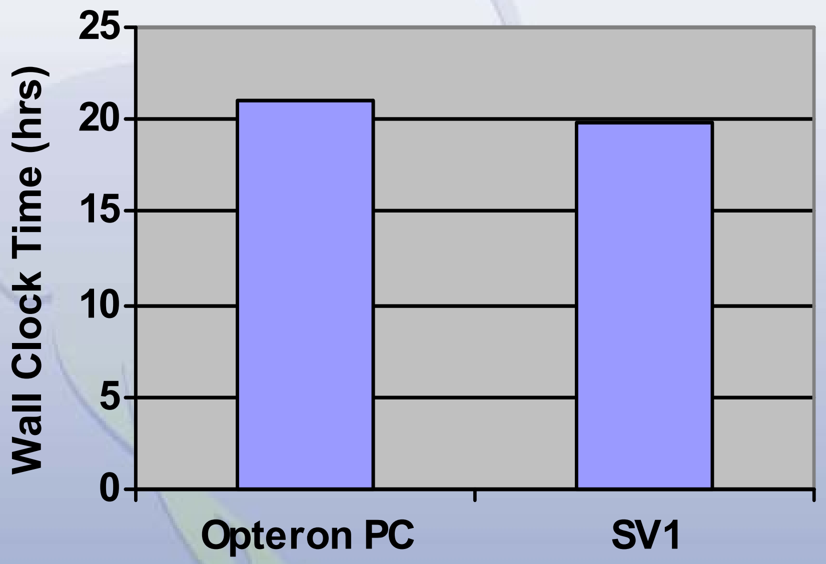
NEACRP Model Results



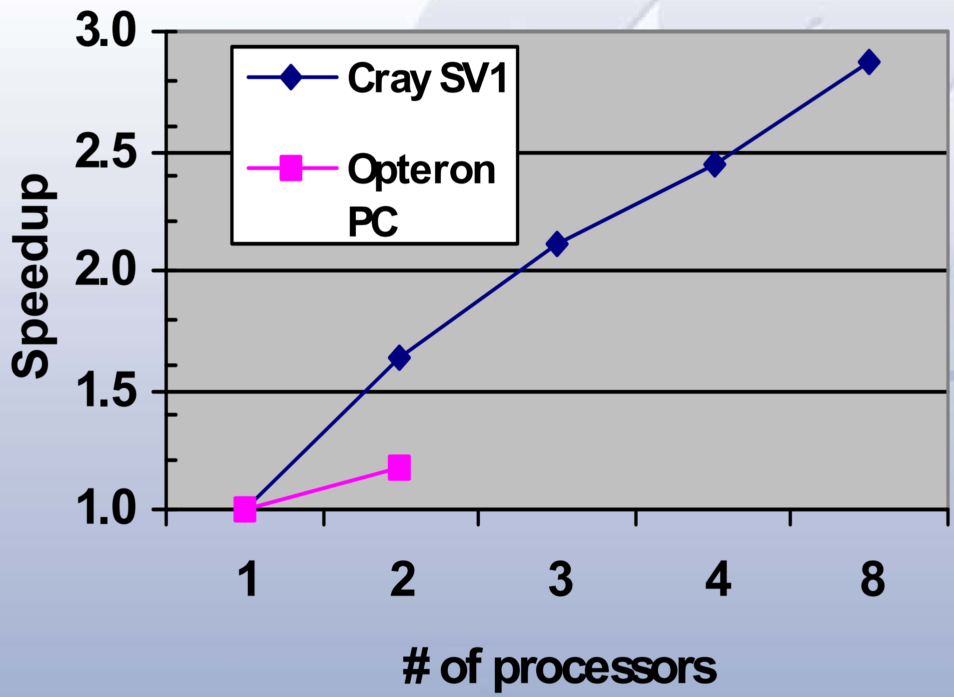


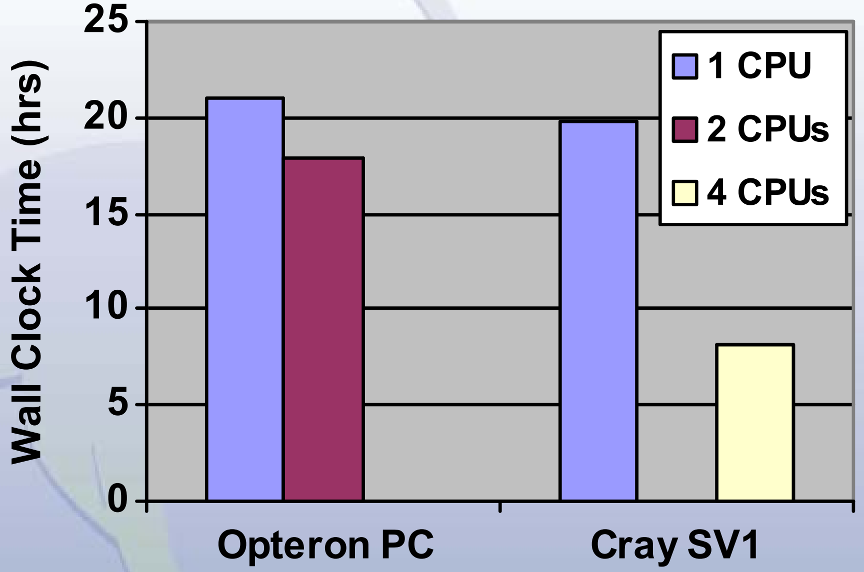
Large ATR Model Results





Parallel Performance – ATR Model





Conclusions

- *Attila can be modified to perform well on a Cray SV1*
  - *Cray Scientific Library made the difference!*
  - *Multitasking in SITRSOL improves scalability*
- *Greater productivity*
  - *Large ATR models can run in half the time as before*
  - *Multiple models can be run simultaneously*
- *More work to be done*
  - *Can we get even better performance?*
  - *More extensive V&V*