Benchmarking BGL, UPC, Checkpoint/Restart

Future Technologies Group

Lawrence Berkeley National Laboratory

Brent Gorda
October 15th, 2003
Outline

- HPC @ LBNL/NERSC
- FTG’s purpose
- Performance studies, UPC, Checkpoint/Restart
- Follow-up
DOE Office of Science flagship Computing Center

Supports open, unclassified, basic research

- ~2000 Users, ~400 Projects

- Main computational facility (Seaborg) consists of:
  - 416 16-way Power 3+ nodes
  - 6,656 CPUs – 6,080 for computation @ 1.5 Gflop/s each
  - Peak Performance of 10 Teraflop/s
  - 7.8 TB Memory, 44TB GPFS disk (+15TB local disk)
LBNL’s Future Technologies Group (FTG) is focused on performance aspects of High Performance Computing (HPC).

FTG’s focus is the 5+ year timeframe.

http://ftg.lbl.gov
Bgorda at lbl.gov
FTG seeks to understand performance of new architectures:

http://ftg.lbl.gov
Bgorda at lbl.gov
Modern Vector Evaluation

Principal Investigator: Lenny Oliker

- Study performance of SX6, X1, Earth Simulator
- Study of key factors of modern parallel vector systems: runtime, scalability, programmability, portability, and memory overhead while identifying potential bottlenecks
- Microbenchmarks, kernels, and application codes

http://ftg.lbl.gov
Bgorda at lbl.gov
Leverage current work:

- Micro benchmarks in communications, memory access issues/patterns/conflicts
- Application kernels – glimpse at performance expectations
- If able: select application codes for in-depth capability-oriented study

Can BGL enable science for the Office of Science?
Applications of interest

- **Astrophysics:**
  - **MADCAP** Microwave Anisotropy Dataset Computational Analysis Package. Analyses cosmic microwave background radiation datasets to extract the maximum likelihood angular power spectrum. [Julian Borrill](mailto:jborrill@lbl.gov) LBNL
  - **CACTUS** Direct evolution of Einstein's equations. Involves a coupled set of non-linear hyperbolic, elliptic equations with thousands of terms. [John Shalf](mailto:john@lbl.gov) LBNL

- **Climate:**
  - **CCM3** Community Climate Model [Michael Wehner](mailto:mwehner@lbl.gov) LBNL
Fusion

- **GTC** Gyrokinetic Toroidal Code. 3D particle-in-cell code to study microturbulence in magnetic confinement fusion. Stephane Ethier Princeton Plasma Physics Laboratory

- **TLBE** Thermal Lattice Boltzmann equation solver for modeling turbulence and collisions in plasma. Jonathan Carter LBNL

Material Science

- **PARATEC** PARAllel Total Energy Code. Electronic structure code which performs ab-initio quantum-mechanical total energy calculations. Andrew Canning LBNL

Molecular Dynamics

- **NAMD** Object-oriented molecular dynamics code designed for simulation of large biomolecular systems. David Skinner LBNL

http://ftg.lbl.gov
Bgorda at lbl.gov
Principal Investigator: Kathy Yelick (UCB)
Joint project between LBNL and UC Berkeley
UPC is an explicitly parallel *global address space* language with **SPMD parallelism**

- An extension of C
- Shared memory is partitioned by threads
- One-sided (bulk and fine-grained) communication through reads/writes of shared variables

UPC has a “forall” construct for distributing computation:

**Ex: Vector Addition**

```c
shared int v1[N], v2[N], v3[N];
upc_forall (i=0; i < N; i++; &v3[i] ) {
  v3[i] = v2[i] + v1[i];
}
```
Compiler Implementation

- Based on the Open64 compiler
- Source to source transformation
- Convert shared memory operations into runtime library calls
- Designed to incorporate existing optimization framework in open64
- Communicate with runtime via a standard API

http://ftg.lbl.gov
Bgorda at lbl.gov
Goals: Portability and High-Performance
Single word transfer is key

Software send overhead for 8-byte messages over time.

Not improving much over time (even in absolute terms)
• Standard C compiler (optimizer good)
• Runtime support: GaSNet
• Low latency single word get/put operations

✓ UPC Compiler is Open Source
  – V1.0 released early 2002
  – Next release for SC03
  – Strong BGL interest from UPC Team
Principal Investigator: Paul Hargrove

- DOE Scalable systems software SciDAC
- Checkpoint/restart is a part of the larger resource management picture

- System initiated
- Apps needn’t know (for the most part)
  - No recompile necessary
  - But: sockets, changing files, etc.
System Level Checkpoint facility enables:

- Resource utilization: (NERSC T3D ~70%-90+%)
- Fault tolerance for long running applications
- System Maintenance / Upgrades
- “Livermore Model”
  - Gang Scheduling – Moe Jette’s work
  - Day vs. night use; debug vs long running
  - Capability + capacity
State of C/R

- Linux Kernel 2.4 (RedHat)
- Kernel Module – no kernel source modification
- LAM MPI
- Some details:
  - Standard I/O working
  - In process: pipes, special device files, full process groups, and sessions
  - Signals (and handlers) reinstated, files reopened

- Visit LBL booth @ SC03
- Soon: Initial (open source) release
• **Checkpoint:**
  – Coordination of compute & I/O nodes
  – Save state from compute node / BLRTS
  – Messages in flight: reliable delivery – just drop them?
  – Interaction with rest of BG/L system: batch system

• **Restart:**
  – I/O node: reinstate file pointers, reacquire locks, pid, session ID, process group, etc.
  – Compute nodes: recover memory, reestablish communications end-points

✔ **Checkpoint restart is Open Source**
  – Initial release for SC03
  – C/R on BGL shouldn’t be hard
Conclusion

- FTG is interested in whether BlueGene/L is an appropriate architecture for the Office of Science
- We have applicable projects and talent to contribute to the LLNL/IBM effort

Thank you!