Optimizing IO Performance in ECHAM5-HAM

Mark Cheeseman
CSCS is Switzerland's national HPC resource

- provides resource and assistance to all educational institutions
- provide computer resource from MeteoSwiss
- support various disciplines (computational chemistry, CFD, climate)
- offer different project sizes (small, large, ALPS)

Advanced Large Projects in Supercomputing

- large amounts of computational and technical resource over 2 years
- 4 projects approved for January 2007 start
  - CFD
  - computational biology
  - computational chemistry
  - climate modelling
ALPS Climate Project

“Climate change and the hydrological cycle from global to European/alpine scales”.
- Principal Investigator: Dr. Ulrike Lohmann (ETHZ)

Project goals include:
- development of high-resolution climate modelling system
- prediction of extreme weather events

Use of ECHAM5-HAM (Version 5.3.02)
- 500 years of integration at T63L31
- 100 years of integration at T106L31
- 2.35M CPUh (out of 3.75M CPUh allotted)
- Optimizing ECHAM5-HAM is HIGHLY desirable
Code Specifics (at T63L31 on Cray XT3)

ECHAM5 with aerosol and atmos. chemistry modules added
- 36 additional tracers

normal domain decomposition used
- minimum of 32 single-core nodes (memory)
- maximum of 384 single-core nodes (discretization)

single IO node used with NetCDF output format

data nudging used for SST, VOR, STP and DIV

6-hourly output frequency
The graph shows the relationship between the number of days of integration and wall time (in minutes) for different configurations of ECHAM5-HAM. The configurations are represented by different markers: 64 (square), 128 (diamond), 256 (triangle), and 384 (inverted triangle). The wall time increases with the number of days of integration, indicating that more days of integration require more computational time. The configurations with higher numbers of days of integration show a more pronounced increase in wall time, suggesting that the computational cost grows with the complexity of the integration period.
Performance Analysis Continued...

for 1 month integration on 128 CPU (no optimizations)

- **34%** walltime is IO activity
  - concentrated in output of large diagnostic files
  - data nudging input not so significant

- **17%** walltime is MPI activity
  - barriers
  - synchronizations
  - collective communication calls

Our Goal? - *Minimize the computational significance of IO*

**How do we proceed?**

- find optimal NetCDF chunk size
- exploit Lustre filesystem
- use IOBUF
IO node strategy used

global domain is broken into number sub-domains that are distributed to worker nodes

At an “output event”, IO node will
  ▪ collect worker sub-domains
  ▪ create corresponding global domain
  ▪ output data to hard disk

simple, efficient with little interruption to compute node workload

targeted for shared memory platform with small # of nodes
  ▪ concern for large distributed memory platforms
  ▪ could add additional IO nodes (complexity, development time)

IO performance in default code

  ▪ Output per month is ~32.5GB data
    ▪ translates to average output rate of ~32.5 MB/sec
    ▪ system's maximum output rate is ~250 MB/sec
Diagnostic data output is the most computationally expensive

Diagnostic data output consists of:
- output of approx. 32.5 GB
- data written to 12 principal NetCDF files (files over 0.1 GB)
- adding a time slice of a 3D array (~ 4.4MB for T63L31)
- high frequency of these writes

Let's perform quick analytical test
- use IO pattern described above
- include 124 writes (corresponds to 31-day month)
- add some NetCDF performance tuning
  - no pre-filling of variables
  - vary size of write “chunks”
- measure average bandwidth of outputting 32.5GB in 12 files
No CRAY XT-IOBUF
- output rate increases with chunk size
- 128MB optimal chunk size

With CRAY XT-IOBUF
- huge performance increase
- inverse chunk size relation
- 0.5MB optimal chunk size

*performed on a separate, dedicated system
Lustre parallel filesystem

virtual high-speed file interface system. Composed of

- metadata server (MDS)
- object data servers (OSS)
- object storage targets (OSTs)

achieves high-throughput via

- splitting of meta-data and “real” data operations
- data striping

optimal Lustre settings are important

- application dependent
- system dependent

Conduct test

- 1 day of integration in ECHAM5-HAM on “PALU”
- vary stripe size from 64kB to 8MB
- vary OST number from 2 to 21
- possible 3-24% drop in average execution time using optimal settings
- performance drop between 4 and 8 OSTs
- 4 OSTs with 1MB stripes chosen
- Again best performance is with 4 OSTs
- settings still significant (2-13% drop in average execution time)
- IOBUF lessens impact of Lustre settings
- only 2-6% possible drop in average execution time
XT-IOBUF Library

- intercepts IO calls made on the compute nodes
- user can specify:
  - size and number of buffers
  - filetype to be buffered
  - shared/not shared
  - flush frequency
- INTENDED USE: Gather “small-block” writes to reduce flush frequency from compute nodes
- limited value for “large-block” writes

Conduct tests

- 1 model day of integration in ECHAM5-HAM
- vary number of buffers from 1 to 32
- vary buffer size from 5MB to 20MB each
- use 4 OSTs with 1MB stripes
Observations

- more virtual buffer space leads to better performance
- only 2-5% change in average execution time
Final Configurations

32kB chunk size
- 4 OSTs with 1MB stripes
- thirty-two 20MB buffers for IOBUF

512kB chunk size
- 4 OSTs with 2MB stripes
- eight 5MB buffers for IOBUF

128MB chunk size
- 4 OSTs with 256kB stripes
- no IOBUF for NetCDF output

* IOBUF used to buffer STDOUT and input nudging data in all cases
** 128 single-core CPUs used
*** 6-hourly output
### Execution times (32 days, 128 CPU)

- no aggressive opt: 149 min
- aggressive opt + Lustre config: 51 min
- optimal chunksize + IOBUF: 44 min
- remove NetCDF sync (add Massoc): 38 min

### Results achievable with “user” or “vendor” approaches

1 month of integration takes ~112 min on NEC SX-6 (8 CPU) dedicated queue
Final Performance

Total execution time (32 days, 256 CPU)
- compiler opt + Lustre config
- optimal chunksize & IOBUF 35 min
- remove NetCDF sync 28 min
  (add Massoc)
- remove IEEE compliance 26.5 min

IO Significance (1 month, 128 CPU)
- no optimizations 33%
- with compiler opt 21%
- in final configurations <11%
Communication costs are more important

- no optimizations: 17%**
- with compiler opt: 57%
- in final configurations: ~55%

Barriers and collective communication calls are most significant (~17.5% and ~12% respectively)

Communication optimization is now vital
- barrier reduction
- removal of collective communication calls

**of total wall time for run
many thanks to the following...

CSCS
  Dr. Neil Stringfellow
  Mr. Hussein Harake

Cray
  Mr. Roberto Ansaloni
  Ms. Nina Suwanphim
  Ms. Tricia Balle

MPI
  Dr. Luis Kornblueh

IES-Ispra
  Ms. Silvia Kloster