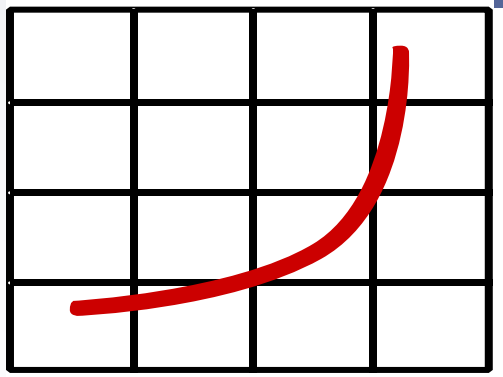
How to run SPEC benchmarks?

Sandra Wienke, Bo Wang, RWTH Aachen University

Ke Wang, University of Virginia



spec

Contents

- Hands-On: Cluster login
- Overview of System Requirements
- Configuration Files – The bad, the good & the ugly
  - SPEC ACCEL: Investigating OpenACC Performance
    - runspec & run rules
    - From base to peak runs
    - Different compilers
  - SPEC ACCEL: Adaptions for OpenCL Benchmarks
  - SPEC OMP2012: Host Benchmarking
- Hands-On: Run your benchmark

Hands-On

Cluster login

- See instructions on handouts
- Follow interactive demo

Interactive demo time!

- We present SPEC ACCEL OpenACC config files
- More slides for OpenCL and OpenMP attached
- Opportunity to follow instructions interactively (also see handouts)
- Later: run benchmarks
  - Choose from OpenACC, OpenCL or OpenMP (whatever you like)
  - MPI runs take very long – get prepared results
- Advertisement: interpret results after break

Contents

System Requirements

- Different benchmarks suites – different requirements
  - SPEC ACCEL[1], SPEC OMP2012[2], SPEC MPI2007[3]
- Supported operating systems: AIX, Linux, MacOS, Solaris, Windows (except very old Windows)
  - Please do not use Windows/Unix compatibility products
- Compatible processors
  - CPU
  - GPU
  - APU
  - Xeon Phi

[1] https://www.spec.org/accel/docs/system-requirements.html
[2] https://www.spec.org/omp2012/Docs/system-requirements.html
[3] https://www.spec.org/mpi2007/Docs/system-requirements.html

System Requirements

- Memory requirements
  - OpenMP: 28GB for the whole system
  - MPI: 1GB/rank (medium size) and 2GB (large size)
  - ACCEL: 4GB of host + 2GB of device
  - Otherwise, you are measuring your paging file, not your system
- Disk space requirements
  - OpenMP: 8GB
  - MPI: 10GB (medium), 17GB (large, big endian), 24GB (large, little endian)
  - ACCEL: 9GB
- Support of compilers
  - C99, C++98 and Fortran-95 compilers + MPI library for SPEC MPI 2007

Contents

- Hands-On: Cluster login
- Overview of System Requirements
- Configuration Files – The bad, the good & the ugly
  - SPEC ACCEL: Investigating OpenACC Performance
    - runspec & run rules
    - From base to peak runs
    - Different compilers
  - SPEC ACCEL: Adaptions for OpenCL Benchmarks
  - SPEC OMP2012: Host Benchmarking
- Hands-On: Run your benchmark

SPEC Tools - `runspec`

Tools provided to ensure consistent operation of benchmarks across variety of platforms[1]

- **`specmake`**
  - GNU make to build benchmarks
- **`runspec`**[2]
  - Primary tool in the suite
  - Used to build the benchmarks, run them, and report their results
  - Config file needed for usage (with detailed instructions about how to build and run the benchmarks)
- And more

[1]https://www.spec.org/accel/docs/tools-build.html
[2]https://www.spec.org/accel/docs/runspec.html

runspec - Run SPEC benchmarks

```
$> cd $SPEC
$> . ./shrc
$>
$> runspec --config=tutSC15-openacc-pgi --tune=base,peak 350
```

- Base & peak runs

Use config file `myopenaccconf`

- Run selected benchmark: `350`
- Or: whole suite, e.g. `openacc`

- Compile benchmarks
- Run benchmarks
- Obey to **SPEC run rules**[1]
  - Tester supplies compiler & system
  - No code modifications of provided sources allowed
  - Restrictions for base & peak runs
  - Data set sizes pre-specified (test, train, ref)[2]

[1] https://www.spec.org/accel/docs/runrules.html, https://www.spec.org/omp2012/docs/runrules.html, https://www.spec.org/mpi/docs/runrules.html

[2] For MPI2007: medium and large data sets

Config Files

- Contain instructions for
  - building benchmarks
  - running them
  - description of system under test

Key for reproducibility!

Use Case:
How to investigate OpenACC performance? The bad, the good & the ugly

- How to write a config file? – The bad
  - Often start off using a config file that someone else has previously written[1]
    - E.g. directory `$SPEC/config/` or ACCEL result submissions similar to your system
  - Write your own[2]

[1] https://www.spec.org/accel/docs/runspec.html#about_config
[2] https://www.spec.org/accel/docs/config.html

Structure of Config Files

Header section
- 1st section prior to names section
- Usually runspec flags

```
iterations    =  1
# Tuning levels: base, peak
tune          =  base
# Dataset size: test, train, ref
size          =  ref
# Environment variable will be set using "ENV_*"
env_vars      =  1
# Output format: all, pdf, text, html and so on
output_format =  text
flagsurl      =  $[top]/result/pgi_flags.xml
teeout        =  yes

# Run benchmarks according to your specific system config
# The variable "command" is the command used by spec
submit = aprun -n 1 -N 1 -q $command

############################################################
# Compiler information
############################################################
```

Named section
- Begins w/ “section marker”
- One- to four-part string of the form `benchmark[,…]=tuning[,…] =extension[,…]=machine[,…]:`

```
openacc=base=default=default:
OPTIMIZE     = -fast -Mfprelaxed
FOPTIMIZE    = -acc -ta=tesla:cc35,cuda6.5
COPTIMIZE    = -acc -ta=tesla:cc35,cuda6.5

############################################################
# Portability flags for each benchmark
# Following flag should not have any impact on performance.
############################################################
116.histo=default=default=default:
PORTABILITY  = -DSPEC_LOCAL_MEMORY_HEADROOM

118.cutcp=default=default=default:
CPORTABILITY += -D__GNUC__

359.miniGhost=default=default=default:
EXTRA_LDFLAGS += -Mnomain

############################################################
# Peak
############################################################
350.md=peak=default=default:
FOPTIMIZE     = -acc -ta=tesla:cc35,cuda5.5,maxregcount:54

363.swim=peak=default=default:
FOPTIMIZE    = -acc -ta=tesla:cc35,cuda5.5,pin

############################################################
# Hardware and software information for the machine under test.
# This information will  be extracted for a reportable run.
# An example configuration can be copied from the website
# https://www.spec.org/accel/results/accel_acc.html
############################################################
company_name    = SPEC Tutorial Company
test_sponsor    = SPEC Tutorial Sponsor
tester          = SPEC Tutorial Tester
license_num     = SPEC Tutorial License
machine_name    = SPEC Tutorial Machine
hw_vendor       = Cray
hw_avail        = Apr-2013
hw_cpu          = AMD Opteron Processor 6276
hw_cpu_mhz      = 2300
hw_cpu_max_mhz  = 3200
hw_cpu_char000  = AMD Turbo CORE Technology up to 3.2GHz,
hw_cpu_char001  = Turbo CORE off
hw_disk         = None
hw_fpu          = Integrated
hw_memory       = 32 GB (4 x 8 GB 2Rx4 PC3L-12800R-11, ECC)
hw_model        = Cray XK7
hw_ncores       = 16
hw_nchips       = 2
hw_ncoresperchip = 16
hw_nthreadspercore = 1
hw_ncpuorder    = 1 chip
hw_other        = None
hw_ocache       = None
hw_pcache       = 32 KB I + 16 KB D on chip per core
hw_scache       = 16 MB I+D on chip per chip, 2 MB shared / 2 cores
hw_tcache       = 16 MB I+D on chip per chip, 8 MB shared / 8 cores
sw_avail        = Feb-2015
sw_compiler000  = PGI Accelerator Fortran/C/C++ Server,
sw_compiler001  = Release 15.3
sw_file         = NFSv3 (IBM N5500 NAS) over Gb ethernet
sw_os000        = SUSE Linux Enterprise Server 11 (x86_64),
sw_os001        = Cray Linux Environment 4.2, Kernel
hw_os002        = 2.6.32.59-0.7.1_1.0402.7496-cray_gem_c
sw_other        = NVIDIA CUDA 5.5.20
sw_state        = Multi-user, run level 3
sw_base_ptrsize = 64-bit
hw_accel_connect = PCIe 2.0 16x
hw_accel_desc000 = NVIDIA Tesla K20m GPU, 2496 CUDA cores,
hw_accel_desc001 = 706MHz, 5 GB GDDR5 RAM
hw_accel_ecc     = yes
hw_accel_model   = Tesla K20
```

MD5 section
- Automatically-generated

```
############################################################
# MD5 section. It will be created by SPEC automatically.
# It is used by SPEC to check whether an executable if
# available is created using the current compiler and flags
# settings.
############################################################
```

SPEC ACCEL Config File (1/11)

config file: $SPEC/config/tutSC15-openacc-pgi.cfg

```
##########################################################
# The header section of the config file.  Must appear
# before any instances of "default="
##########################################################


# what to do: build, validate = build + run
action        =  validate


# Number of iterations of a test
iterations    =  1


# Tuning levels: base, peak
tune          =  base


# Dataset size: test, train, ref
size          =  ref
```

- **`build`**: compile benchmarks
- **`validate`**: benchmarks are built if necessary, run and reports are generated

- How many times to run each benchmarks
- e.g. for reportable run = 3

- **`base`**: flags for all benchmarks the same
- **`peak`**: set of optimizations individually selected for that benchmark

- Data set sizes (from small to big): `test`, `train`, `ref`
- e.g. test for debugging new set of compilation options

SPEC ACCEL Config File (2/11)

config file: $SPEC/config/tutSC15-openacc-pgi.cfg

```
##########################################################
# The header section of the config file.  Must appear
# before any instances of "default="
##########################################################


# Environment variable will be set using "ENV_*", see the next section
env_vars       =  1



# Output format: all, pdf, text, html and so on
output_format =  text
flagsurl       =  $[top]/result/pgi_flags.xml
teeout         =  yes
```

- Environment settings
- ENV_*VAR* = …
- Apply to build phase → rebuild if any changes

- Different output formats possible
- In $SPEC/results

- Description of portability & tuning options (“Flags File”)
- Information on syntax of flags and their meanings
- Needed for valid reports

Displays the build commands to screen

SPEC ACCEL Config File (3/11)

```
# How to run the benchmarks according to your specific system configuration
# The variable "command" is the command used by spec
submit = aprun -n 1 $command
```

- **How to execute the benchmarks**
- Use `$command` for SPEC command
- Preferred to assign work to processors
  - May place benchmarks on desired processors or benchmark memory on a desired memory unit
  - Especially needed for MPI runs
  - Here: run job on one node
- Can be used to change the run time environment
  `submit = export ENV_VAR=…; …`
  → no rebuild if any changes occur

SPEC ACCEL Config File (4/11)

Named section
benchmark[,…]=tuning[,…]
=extension[,…]=machine[,…]:

```
############################################################
# Software information
############################################################

# Compilers. Using PGI compiler for example
default=default=default=default:
CC                = pgcc
CXX               = pgc++
FC                = pgfortran



################################################################
# Base
################################################################

openacc=base=default=default:
OPTIMIZE     = -fast -Mfprelaxed
FOPTIMIZE    = -acc -ta=tesla:cc35,cuda5.5
COPTIMIZE    = -acc -ta=tesla:cc35,cuda5.5
```

- Define PGI compiler for OpenACC compilation
- Define PGI GPU flags

Base

- Common set of optimizations & environment settings for all benchmarks
- "baseline"
  - single set of switches
  - single-pass make process
  - high degree of portability, safety, performance
- Must adhere to strict rules
  - e.g. same compiler for all modules of a given language
  - All flags, options must be the same
  - More rules (base & peak) on names, library substitutions, data type sizes, source code changes

SPEC ACCEL Config File (5/11)

```
############################################################################
# Portability flags for each benchmark
# Following flag should not have any impact on performance.
############################################################################

116.histo=default=default=default:
PORTABILITY  = -DSPEC_LOCAL_MEMORY_HEADROOM

118.cutcp=default=default=default:
CPORTABILITY += -D__GNUC__

359.miniGhost=default=default=default:
EXTRA_LDFLAGS += -Mnomain
```

Portability flags

- Allowed if benchmark cannot be built and execute correctly w/o these flags
- Must be performance neutral
- Base rules except portability flags, i.e. flags may differ from one benchmark to another (even in base)
- Requirements
  - Provided over `PORTABILITY` flag
  - Must be approved by SPEC HPG committee

SPEC ACCEL Config File (6/11)

```
############################################################################
# Hardware and software information for the machine under test.
# This information will  be extracted for a reportable run.
# An example configuration can be copied from the website
# https://www.spec.org/accel/results/accel_acc.html
############################################################################

company_name     = SPEC Tutorial Company
test_sponsor     = SPEC Tutorial Sponsor
tester           = SPEC Tutorial Tester
license_num      = SPEC Tutorial License
machine_name     = SPEC Tutorial Machine

hw_vendor        = Cray
hw_avail         = Apr-2013
hw_cpu           = AMD Opteron Processor 6276
hw_cpu_mhz       = 2300
hw_cpu_max_mhz   = 3200
```

HW & SW description

- Needed only for reportable runs
- `runspec` tools captures information in submission file
- Very detailed information
- More: next tutorial section

Information on host configuration, e.g. CPU

SPEC ACCEL Config File (7/11)

```
hw_cpu_char      = AMD Turbo CORE Technology up to 3.2GHz, Turbo CORE off
hw_disk          = None
hw_fpu           = Integrated
hw_memory        = 32 GB (4 x 8 GB 2Rx4 PC3L-12800R-11, ECC)
hw_model         = Cray XK7
hw_ncores        = 16
hw_nchips        = 2
hw_ncoresperchip = 16
hw_nthreadspercore = 1
hw_ncpuorder     = 1 chip
hw_other         = None
hw_ocache        = None
hw_pcache        = 32 KB I + 16 KB D on chip per core
hw_scache        = 16 MB I+D on chip per chip, 2 MB shared / 2 cores
hw_tcache        = 16 MB I+D on chip per chip, 8 MB shared / 8 cores
```

SPEC ACCEL Config File (8/11)

```
sw_avail        = Feb-2015
sw_compiler     = PGI Accelerator Fortran/C/C++ Server, Release 15.3
sw_file         = NFSv3 (IBM N5500 NAS) over Gb ethernet
sw_os000        = SUSE Linux Enterprise Server 11 (x86_64), Cray Linux Environment 4.2
sw_os001        = Kernel 2.6.32.59-0.7.1_1.0402.7496-cray_gem_c
sw_other        = NVIDIA CUDA 5.5.20
sw_state        = Multi-user, run level 3
sw_base_ptrsize = 64-bit

hw_accel_connect = PCIe 2.0 16x
hw_accel_desc    = NVIDIA Tesla K20m GPU, 2496 CUDA cores, 706MHz, 5 GB GDDR5 RAM
hw_accel_ecc     = yes
hw_accel_model   = Tesla K20
hw_accel_name    = NVIDIA Tesla K20
hw_accel_type    = GPU
hw_accel_vendor  = NVIDIA
sw_accel_driver  = NVIDIA UNIX x86_64 Kernel Module 319.82
```

Information on software configuration, e.g. compilers

Information on accelerator configuration, e.g. device vendor

SPEC ACCEL Config File (9/11)

```
############################################################################
# MD5 section. It will be created by SPEC automatically.
# It is used by SPEC to check whether an executable if available is created using
# the current compiler and flags settings.
############################################################################
```

MD5 section

- Automatically generated by SPEC tools
- Used to check whether an executable is created using the current settings

SPEC ACCEL Config File (10/11)

- Modifying the config file – The good
  - Once you have a config file that runs on your system, it is easy to modify it
  - E.g. `peak` optimizations for better performance

```
##########################################################
# Peak
##########################################################

350.md=peak=default=default:
FOPTIMIZE    = -acc -ta=tesla:cc35,cuda5.5,maxregcount:54

363.swim=peak=default=default:
FOPTIMIZE    = -acc -ta=tesla:cc35,cuda5.5,pin
```

Peak
- Set of optimizations individually selected for each benchmark
  - e.g. different compilers, flags
- Called “aggressive compilation”

- `maxregcount`: sets register spilling to 54
- `pin`: pinned memory usage

- `md` benchmark: computational-intensive
- `swim` benchmark: many data transfers between CPU & GPU

SPEC ACCEL Config File (11/11)

- Rewriting the config file – The ugly
  - Major changes in a config file (e.g. compiler) need more adaption

```
# Compilers. Using Cray compiler for example
default=default=default=default:
CC           = cc
CXX          = CC
FC           = ftn
###############################################################
# Base
###############################################################
openacc=base=default=default:
OPTIMIZE     = -O2
FOPTIMIZE    = -h acc,noomp -em -fpic -dynamic
COPTIMIZE    = -h pragma=acc -h nopragma=omp -fpic -dynamic
###############################################################
# new portability flags; new peak flags
###############################################################
```

Contents

- Hands-On: Cluster login
- Overview of System Requirements
- Configuration Files – The bad, the good & the ugly
  - SPEC ACCEL: Investigating OpenACC Performance
    - runspec & run rules
    - From base to peak runs
    - Different compilers
  - SPEC ACCEL: Adaptions for OpenCL Benchmarks
  - SPEC OMP2012: Host Benchmarking
- Hands-On: Run your benchmark

SPEC ACCEL OpenCL – Device Selection

- SPEC ACCEL OpenCL benchmarks are compatible with multiple devices
- These devices include: CPU, GPU, Xeon Phi and APU
- When multiple devices co-exist, SPEC ACCEL OpenCL allows user to specify the device for running benchmarks through the runspec option `--device <device_number>`
- *device_number* can be a
  - number
  - name among CPU, GPU and ACCELERATOR

Example:

```
$> runspec --size train --platform NVIDIA --device 0 --config tutSC15-ocl-gnu.cfg lavamd
```

SPEC ACCEL OpenCL – Workgroup Sizes

- Some (not all) OpenCL benchmarks have the ability to change the workgroup size
- Changing workgroup size may improve the performance
- To accomplish this, SPEC provides define variables to change workgroup sizes
- Some of the benchmarks have multiple variables for different OpenCL kernels
- Only allowed in a peak run.

- For one benchmark:
  - `SPEC_ACCEL_WG_SIZE`
    - sets all of the workgroup sizes to this value
  - `SPEC_ACCEL_WG_SIZE_n` and `SPEC_ACCEL_WG_SIZE_n_n`
    - set specific values for a specific kernel and a specific dimension of a kernel, respecitvely

101.tpacf	0	0_0
103.stencil	0, 1	0_0, 0_1, 0_2
104.lbm	0, 1	0_0, 0_1, 0_2
112.spmv	0	0_0
114.mriq	0, 1	0_0, 1_0
116.Histo	0, 1, 2, 3	0_0, 1_0, 2_0, 3_0
117.bfs	0, 1	0_0, 1_0
120.kmeans	0, 1	0_0, 1_0
121.lavamd	0	0_0
122.cfd	0, 1, 2, 3, 4	0_0, 1_0, 2_0, 3_0, 4_0
123.nw	0	0_0
124.hotspot	0	0_0
125.lud	0	0_0
127.srad	0	0_0
128.heartwall	0	0_0

Contents

- Hands-On: Cluster login
- Overview of System Requirements
- Configuration Files – The bad, the good & the ugly
  - SPEC ACCEL: Investigating OpenACC Performance
    - runspec & run rules
    - From base to peak runs
    - Different compilers
  - SPEC ACCEL: Adaptions for OpenCL Benchmarks
  - SPEC OMP2012: Host Benchmarking
- Hands-On: Run your benchmark

SPEC OMP2012 Config File (1/4)

config file: $SPEC/config/myopenmpconf.cfg

```
############################################################
# Header section
############################################################

# what to do build
action          =       validate
# Number of iterations of a test
iterations      =       1
# Tunning levels: base, peak
tune             =      base,peak
# Dataset size: test, train, ref
size             =       train
# Number of used threads
threads         =       16
# Environment variable will be set using "E_ _", see the next section
env_vars        =       1
# Output format: all, pdf, text, html and so on
output_format   =       text
```

OMP needs # threads *OR*

```
runspec --config=myopenmpconf --threads=16 omp
```

Environment variables important for NUMA settings

SPEC OMP2012 Config File (2/4)

```
############################################################
# Compiler information
############################################################
# Compilers. Using Intel compiler for example
default=default=default=default:
CC  = icc
CXX = icpc
FC  = ifort
F77 = ifort
############################################################
# Portability flags for each benchmark
############################################################
350.md=default=default=default:
FPORTABILITY  = -free
367.imagick=default=default=default:
CPORTABILITY  = -std=c99
357.bt331=default=default=default:
PORTABILITY   = -mcmodel=medium
363.swim=default=default=default:
PORTABILITY   = -mcmodel=medium
```

Choose your OMP compiler

As always: portability flags should not impact performance

SPEC OMP2012 Config File (3/4)

```
############################################################
# Base
############################################################
default=base=default=default:
# Basic optimization flags for all benchmarksE
OPTIMIZE=-O3 -openmp -ipo -xCORE-AVX-I  -no-prec-div
# Optimization flags for benchmarks written in C
COPTIMIZE=-ansi-alias
# Optimization flags for benchmarks written in C++
CXXOPTIMIZE=-ansi-alias
# Optimization flags for benchmarks written in Fortran
FOPTIMIZE=-align
# Environment variables at runtime
ENV_KMP_AFFINITY  = compact
ENV_KMP_SCHEDULE  = static,balanced
ENV_KMP_BLOCKTIME = infinite
ENV_KMP_LIBRARY   = throughput
ENV_KMP_STACKSIZE = 500M
ENV_OMP_NESTED    = FALSE
ENV_OMP_DYNAMIC   = FALSE
```

Basic optimization flags

Environment variables, e.g. on NUMA settings or loop schedules

SPEC OMP2012 Config File (4/4)

```
############################################################
# Peak
############################################################
350.md=peak=default=default:
OPTIMIZE=-O3 -openmp -ipo -xCORE-AVX-I -ansi-alias -opt-malloc-options=1
FOPTIMIZE=-fp-model fast=2 -no-prec-div -no-prec-sqrt -align array64byte

363.swim=peak=default=default:
OPTIMIZE=-O3 -openmp -ipo -xCORE-AVX-I  -no-prec-div -ansi-alias -opt-streaming-stores always
-opt-malloc-options=4
ENV_KMP_AFFINITY=compact,0



############################################################
# Hardware information
############################################################
# MD5 section
############################################################
```

Peak runs include more aggressive optimizations

Summary - SPEC ACCEL & SPEC OMP2012

- SPEC ACCEL for accelerators
  - Including benchmarks implemented using OpenACC and OpenCL
  - Under development: benchmarks using OpenMP target
  - Don’t need to specify #threads
  - Can specify an accelerator, if there are multiple
- SPEC OMP2012 for Host CPU
  - Benchmarks implemented using OpenMP for host
  - Need to specify #threads
  - Processor/core can be chosen using environment variable, such as ENV_OMP_PLACES

Contents

- Hands-On: Cluster login
- Overview of System Requirements
- Configuration Files – The bad, the good & the ugly
  - SPEC ACCEL: Investigating OpenACC Performance
    - runspec & run rules
    - From base to peak runs
    - Different compilers
  - SPEC ACCEL: Adaptions for OpenCL Benchmarks
  - SPEC OMP2012: Host Benchmarking
- Hands-On: Run your benchmark

Hands-On

Follow the instructions on the hands-on handout!

Run your benchmark!

- OpenACC
- OpenCL
- OpenMP (2012)