Running codes with PerfExpert and MACPO

1. Running a stand-alone code (Lulesh)

Enter the node:

```
$ ~/reserve
```

Enter the LULESH example directory:

```
$ cd 4
```

1.1 Using PerfExpert

Compile both optimized and unoptimized codes:

```
$ g++ -g -O3 -fopenmp -o lulesh lulesh.cc
$ g++ -g -O3 -fopenmp -o lulesh_opt lulesh_opt.cc
```

Run both codes with PerfExpert:

```
$ OMP_NUM_THREADS=16 perfexpert lulesh 50
$ OMP_NUM_THREADS=16 perfexpert lulesh_opt 50
```

Compare the result using the PerfExpert analysis output in `perfexpert-temp-*`

1.2 Using MACPO

Compile the un-optimized code:

```
$ macpo.sh --macpo:function=CalcFBHourglassForceForElems -g
-O3 -fopenmp -o lulesh lulesh.cc
```

Run the application:

```
$ ./lulesh 50
```

Analyze the result:

```
$ macpo-analyze macpo.out
```

Compile the optimized code:

```
$ macpo.sh --macpo:function=CalcFBHourglassForceForElems -g
-O3 -fopenmp -o lulesh_opt lulesh_opt.cc
```

Run the application:

```
$ ./lulesh_opt 50
```

Analyze the result:

```
$ macpo-analyze macpo.out
```

Exit the node:

```
$ exit
```

2. Running a Makefile-based code (Rodinia BackProp benchmark)

Enter the node:

```
$ ~/reserve
```

Enter the Rodinia example directory:

```
$ cd 3
```

2.1 Using PerfExpert

Run PerfExpert using the "-m" argument:

```
$ OMP_NUM_THREADS=16 perfexpert –m –s backprop.c backprop
10000000
```

Compare the result using the PerfExpert analysis output in `perfexpert-temp-*`

2.2 Using MACPO

Compile the code by using MACPO as the C compiler:

```
$ CC="macpo.sh --macpo:function=bpnn_adjust_weights" make
clean backprop
```

Run the code as usual:

```
$ OMP_NUM_THREADS=16 ./backprop 10000000
```

Analyze the output:

```
$ macpo-analyze macpo.out
```

Exit the node:

```
$ exit
```