How to use ppOpen-AT with d-Spline

Riku Murata, Teruo Tanaka

Kogakuin University

September 1, 2014

1. An example of ppOpen-AT with d-Spline

This report is an example of how to use ppOpen-AT with d-Spline[1][2]. ppOpen-AT[3] is auto-tuning preprocessor created by Dr. Katagiri. We used ppOpen-AT version-0.2.0 at http://ppopenhpc.cc.u-tokyo.ac.jp/ (accessed on 27/05/2014).

2. Explanation of the example

This example shows unrolling middle-loop of GEMM. We tried estimating the best parameter using ppOpen-AT in parameters "n" as loop unrolling depth. Range of Matrix Size is from 1000 to 10000 and the step is 1000. Loop unrolling depth is from 1 to 16. This example is not parallelized. Following Figure 1 shows base program code.

```
For (i=0;i<n;i++) {
        for(j=0;j<n;j++) {
                for(k=0;k<n;k++) {
                        C[i*n+j]=A[i*n+k]+B[k*n+j];
                }
        }
}
```

Figure 1 Base program

Following Figure 2 shows a program code inserting ppOpen-AT pragma.

```
 1 #pragma OAT call OAT_ATset(OAT_INSTALL,OAT_InstallRoutines)
 2 #pragma OAT call OAT_ATset(OAT_STATIC,OAT_StaticRoutines)
 3 #pragma OAT call OAT_ATset(OAT_DYNAMIC,OAT_DynamicRoutines)
 4 #pragma OAT call OAT_ATset(OAT_ALL,OAT_AllRoutines)
 5 #pragma OAT OAT_NUMPROCS=1
 6 #pragma OAT OAT_STARTTUNESIZE=1000
 7 #pragma OAT OAT_ENDTUNESIZE=10000
 8 #pragma OAT OAT_SAMPDIST=1000
 9 #pragma OAT OAT_DEBUG=1
10
11 #pragma OAT call OAT_ATexec(OAT_INSTALL,OAT_InstallRoutines)
12 #pragma OAT call OAT_BPset("n")
13 #pragma OAT install unroll (j) region start
14 #pragma OAT name MatMul
15 #pragma OAT fitting dspline
16 #pragma OAT varied (j) from 1 to 16
17 #pragma OAT debug (pp)
18 for(i=0;i<n;i++){
19      for(j=0;j<n;j++) {
20              for(k=0;k<n;k++) {
21                      C[i*n+j]=A[i*n+k]+B[k*n+j];
22              }
23      }
24 }
25 #pragma OAT install unroll (j) region end
```

Figure 2 Program inserting ppOpen-AT

Pragma at line 1 to 9 are related to initialization of the program. OAT_ATexec at line 11 indicates that this program is executed in the installation phase. OAT_BPset at line 12 indicates that name of the variable that specifies size of parameter. The program between “region start” in line 13 and “region end” in line 25 specify portion of auto-tuning (AT). Variable “(j)” in lines 13, 16 and 25 specifies

performance parameter for AT. OAT fitting d-Spline at line 15 shows d-Spline is implemented with ppOpen-AT. Using d-Spline sampling points are automatically selected. In this implementation, when selected value is the same for three successive times, the selected value is determined to be the optimal value and the AT is terminated. OAT_debug (pp) at line 17 specifies output of information of performance parameter.

3. Experimental conditions

Following Table 1 is the conditions for our experiment.

Table 1 Experimental conditions

CPU	Intel Core i7-3770K@3.5GHz
Memory	32GB
OS	CentOS 6.3
gcc version	4.4.7
optimization option	-O3
ppOpen-AT version	0.2.0

- Range of Matrix Size is from 1000 to 10000 and the step is 1000 [10 patterns].
- Loop unrolling depth is from 1 to 16.
- This example is not parallelized.

We executed previous program [dspline] and for comparison, another program which searches all values of the performance parameter [all].

4. Results

Among the ten matrix size patterns, “dspline” estimated exact optimal value for the seven matrix size patterns. Figure 3 and Table 2 show an example when d-Spline estimated exact optimal value (matrix size = 4000).

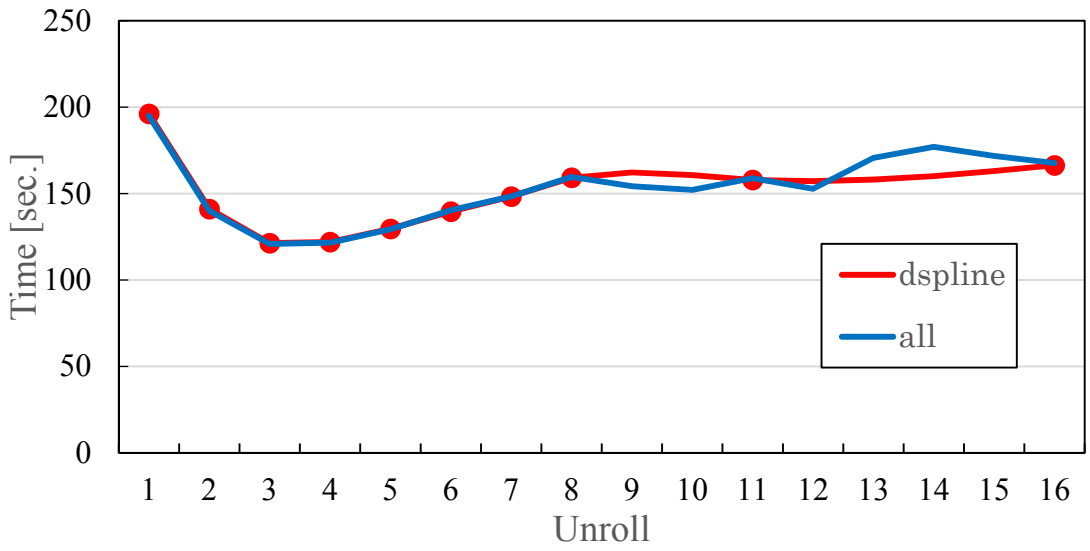

Figure 3 Execution time of unroll depth when matrix size = 4000

Table 2 Execution time for AT (matrix size = 4000)

	All	d-Spline
Execution time (AT) [sec.]	107,323	62,498

Figure 3 shows that “d-Spline” automatically determined the exact optimal value using ten sampling points out of sixteen. Table 2 shows that the execution time for AT by “d-Spine” is reduced by 42%.

Figure 4 shows the worst case when “d-Spline” failed to estimate the exact optimal value (matrix size = 3000). The difference of execution time for the optimal values selected by “d-Spline” and “all” is only 9.7%.

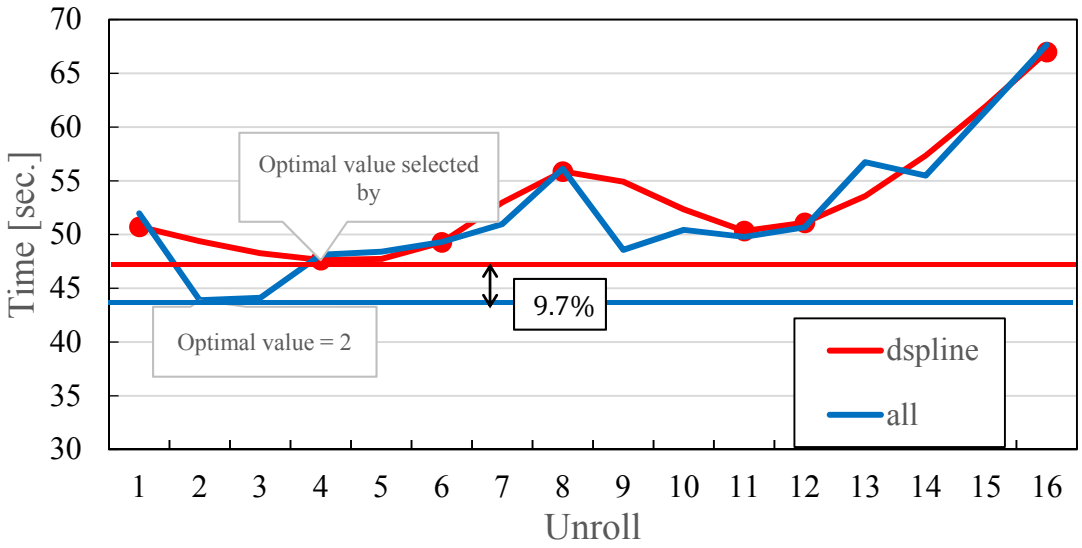

Figure 4 Execution time of unroll depth (matrix size = 3000)

Figure 5 shows performance of three programs, base program, “d-Spline” and “all”. It shows that performance of “d-Spline” and “all” with AT are improved compared with base program (the average is 1.52 times and the maximum is 1.65 times). The figure shows the three cases when “d-Spline” failed to estimate the exact optimal value. The bar graphs in yellow shows relative error between “d-Spline” and “all”.

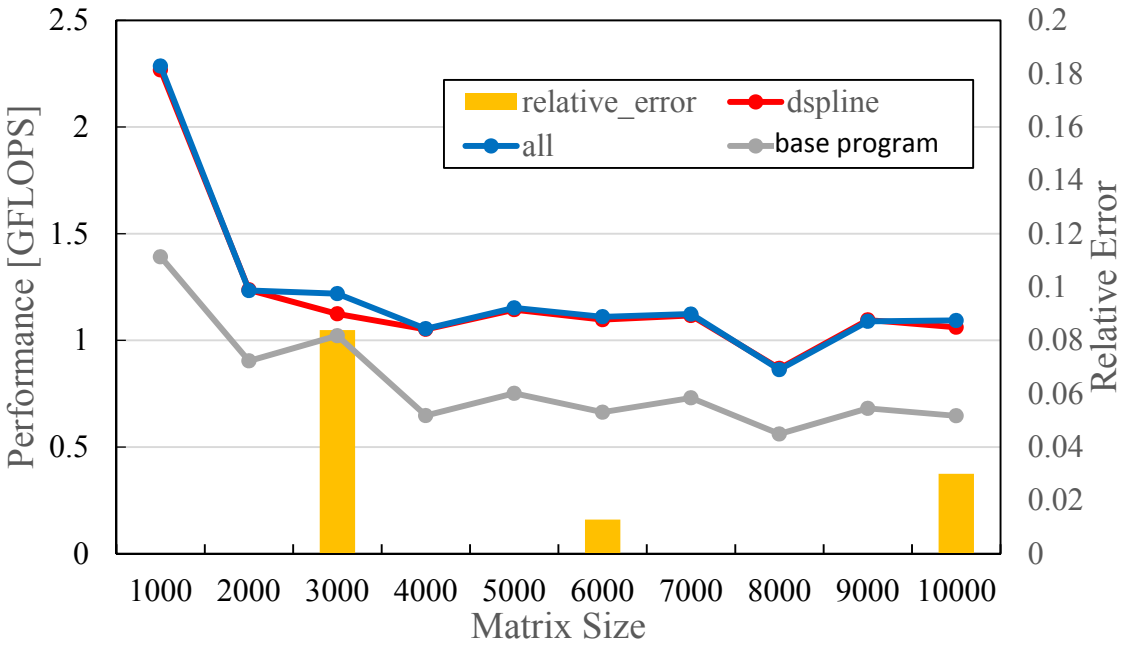

Figure 5 Effect of auto tuning

Reference

[1] T.Tanaka, T.Katagiri, T.Yuba, d-Spline Based Incremental Parameter Estimation Method in Automatic Performance Tuning, LNCS, 4699, Springer, pp.986-995, 2007.

[2] T.Tanaka, R.Otsuka, A.Fujii, T.Katagiri, T.Imamura, Implementation of d-Spline based Incremental Performance Parameter Estimation Method with ppOpen-AT, IOS Press, 2014.

[3] http://ppopenhpc.cc.u-tokyo.ac.jp/ (accessed on 27/05/2014).