

# Isocoupling: Reusing Kernel Coupling Values to Predict the Performance of Parallel Applications

Xingfu Wu, Valerie Taylor  
Dept. of Computer Science  
Texas A&M University  
College Station, TX 77843  
{wuxf, taylor}@cs.tamu.edu

Jonathan Geisler  
Department of CSS  
Taylor University  
Upland, IN 46989  
jgeisler@css.tayloru.edu

Rick Stevens  
MCS Division  
Argonne National Lab  
Argonne, IL 60439  
stevens@mcs.anl.gov

## Abstract

*Kernel coupling quantifies the interaction between adjacent and chains of kernels in an application. A kernel can be a loop, procedure or file. In our previous work, we used the kernel coupling values to identify how to combine the execution times of the individual kernels that compose the application to predict the execution time of the full application. The results of this previous work using the NAS Parallel Benchmark SP demonstrated that the use of coupling values resulted in very good predictions with average errors in the range of only 1.18% in contrast to simply summing the execution times of the kernels that resulted in average errors in the range of 20.54%. The major concern with the coupling values is the fact that values are needed for each different problem size, number of processors and machine. In this paper, however, we explore the ability to reuse coupling values. In particular, we explore the reuse in terms of the three dimensional space consisting of the following axes: number of processors, problem size and system architecture. The experimental results indicate that when considering parallel systems, with increasing number of processors and problem sizes, we found clear transitions with the coupling values resulting in the ability to reuse values. Further, reusing coupling values is feasible on classes of systems such as clusters, distributed shared memory and other distributed memory systems.*

## 1. Introduction

Kernel coupling refers to the effect that *kernel i* has on *kernel j* in relation to running each kernel in isolation. The two kernels can be adjacent kernels in the control flow of the application or a chain of three or more kernels. In our previous work we used the coupling concept to identify parts of the application that required performance improvement [GT99]. In particular, the coupling value provided insight into where further algorithm and code

implementation work were needed to reduce execution time; the focus was on the reuse of data between kernels. This work was extended in [TW01, TW02] to demonstrate how to use the coupling values of adjacent and chains of kernels to predict parallel application execution time using three NAS Parallel Benchmarks [BH95] -- BT, LU and SP -- executed on the IBM SP with up to 32 processors. For BT dataset A, the four-kernel predictor had an average relative error of 0.79 %, while the summation method had an average relative error of 21.80%. For the SP dataset A, the five-kernel predictor had an average relative error of 1.18%, while the summation method had an average relative error of 20.54%. For the LU dataset A, the three-kernel predictor had an average relative error of 1.47%, while the summation method had an average relative error of 4.56%.

In this paper, we explore the ability to reuse coupling values to predict parallel application execution time. In particular we explore the three dimensional space consisting of: (1) different system architectures, (2) different problem sizes and (3) different number of processors. The goal is to determine if coupling values can be reused, implying that coupling values from two distinct points in the space are used to predict similar execution times, hence the term isocoupling. As coupling refers to the interaction between kernels, the resources shared between kernels (namely the memory hierarchy) affect the coupling values. In previous work that focused on single processor performance, it was determined that coupling values went through a fixed number of transitions as determined by the mapping of the dataset to the memory hierarchy [GT99]. The transitions occurred as the dataset size increased from being resident in a given level of the memory hierarchy to requiring the next level. While this previous work focused on a single processor, it was expected that similar results would occur with parallel systems.

For brevity, we present the results from one NAS Parallel Application Benchmark, SP; the results from this

benchmark are representative of the results from other NAS benchmarks. The parallel systems used in the experiments include the IBM SP, Linux cluster Chiba City at Argonne National Laboratory, the SGI Origin2000 at NCSA, and the Linux Supercluster Los Lobos at University of New Mexico. The number of processors went up to 32; and the dataset sizes included small (dataset W), medium (dataset A) and large (dataset B). The results indicate that when considering parallel systems, similar systems produced similar results. Further, with increasing number of processors and dataset sizes, we found clear transitions with the coupling values and distinct areas where coupling values could be reused to produce good estimates of parallel execution time.

The remainder of the paper is organized as follows. Section 2 presents the background on the coupling concept. Section 3 extends the work to consider isocoupling. The experimental results are given in Section 4 followed by related work in Section 5. The paper is summarized in Section 6.

## 2. Background: Kernel Coupling

In this work, a kernel is a unit of computation that denotes a logical entity within the larger context of an application. The unit may be a loop, procedure, or file depending on the level of granularity of detail that is desired from the measurements. The kernel coupling parameter,  $C_{ij}$ , quantifies the interaction between adjacent kernels in the application. To compute the parameter  $C_{ij}$ , three measurements must be taken:

- $P_i$  is the execution time of *kernel i* alone,
- $P_j$  is the execution time of *kernel j* alone, and
- $P_{ij}$  is the execution time of *kernels i* and *j* (assuming *kernel i* immediately precedes *kernel j*) in the application.

These measurements are done in the sequence determined by the application. In particular, a measurement is obtained by placing a given kernel or pair of kernels into a loop, such that the loop dominates the execution time. We record the time of this loop as the time for the kernel or pair of the kernels.

We define the term  $C_{ij}$  to equal the ratio of the measured execution time of the pair of kernels to the expected execution time resulting from combining the isolated execution time of each kernel. Since  $C_{ij}$  is the measurement of interaction between kernels, we compute it as the ratio of the actual execution time of the kernels together to that of no interaction as given below:

$$C_{ij} = \frac{P_{ij}}{P_i + P_j} \quad (1)$$

Let  $W$  be the set of all kernels in the ordered chain of  $k$  kernels. Assume that  $C_W$  is the coupling value of the chain  $W$ , and  $P_W$  is the execution time of the chain  $W$ . Then, the above equation is generalized as

$$C_W = \frac{P_W}{\sum_{j \in W} P_j} \quad (2)$$

Note that for  $k=2$ , Equation 2 is equivalent to Equation 1.

In [TW02], we extended the coupling metric to estimate application execution time,  $T$ , as follow:

$$T = \sum_{i=1}^n \alpha_i N_i P_i \quad (3)$$

where  $N_i$  gives the number of times kernel  $i$  is executed,  $P_i$  is the execution time of kernel  $i$ , and  $\alpha_i$  ( $i = 1, 2, \dots, n$ ) is a weighted average of the coupling values associated with a given kernel  $i$ . Let  $Q_i$  be the set of all ordered chains of  $k$  ( $2 \leq k < n$ ) kernels involved with kernel  $i$ , where the size of the Set  $Q_i$  is  $|Q_i| = k$ . We define the coefficient  $\alpha_i$  ( $i = 1, 2, \dots, n$ ) as follows:

$$\alpha_i = \frac{\sum_{W \in Q_i} C_W P_W}{\sum_{W \in Q_i} P_W} \quad (4)$$

## 3. Isocoupling

In this section, we discuss the concept of isocoupling, or the reuse of coupling values, thereby reducing the number of coupling values that one has to generate. This discussion is based upon intuition, as the mathematical analysis of the equations for the execution time in terms of first and second derivatives with respect to  $C_W$  does not allow one to draw any clear conclusions. Recall that the coupling values quantify the interaction between kernel pairs or kernel chains. This quantity represents the amount of sharing of data between kernels. A coupling value less than one implies that the execution time of the kernel pair or chain is less than the summation of the execution times of the isolated kernels. If we consider single processor execution time, this reduction in execution time can only occur if there is a reduction in data access time as the number of computations remains fixed. Hence, when the coupling value is less than one, there is some data that remains resident in one of the levels of the memory hierarchy, for which when the kernels are executed in

isolation this is not the case. A similar argument can be made about the coupling value being larger than one, as indicating conflicts or thrashing of data when the kernels are executed together as compared to executing the kernels in isolation. Consequently, one would expect that as the size of the dataset increases, the coupling values would go through a fixed number of transitions. In particular, the transition would occur when the dataset increased such that data was no longer resident in one of the levels of the memory hierarchy. The number of transitions of the coupling values would be equal to the number of levels of the memory hierarchy, as the dataset size grows.

If we consider parallel execution time, the sharing of data occurs within the memory hierarchy of each processor as well as the communication of data between processors. There are clear distinctions between the communication mechanisms of distributed shared memory machines versus message passing machines. Further, with the message passing machines, distinctions exist with the network used to connect the processors. For example, there are clear distinctions between the network used with the IBM SP (the proprietary omega network) and the Linux Supercluster Los Lobos (the myrinet

network). Hence, considering parallel execution time and memory hierarchy, we would hypothesize that coupling values can be reused among similar classes of parallel machines with common processor memory hierarchies; the processor speed does not really affect the coupling reuse.

In the following section, we will use experimental results to address the issue about isocoupling in terms of different systems, different dataset sizes, and different number of processors.

#### 4. Case Studies

The experiments presented in this paper use one NAS Parallel Benchmark, SP. The other benchmarks, such as BT and LU, had similar results and are not included for the sake of space. The parallel systems include the Linux Supercluster Los Lobos at the University of New Mexico, the IBM SP and Linux Cluster Chiba City at Argonne National Laboratory, and the SGI Origin 2000 at NCSA. Details about the different architectures are given below in Tables 1 and 2.

**Table 1. Four different system architectures used in the case study.**

| System name        | System Type              | Number of Processors | CPU type            | Network          | Operating system |
|--------------------|--------------------------|----------------------|---------------------|------------------|------------------|
| IBM SP             | Distributed share memory | 80                   | 120 MHz P2SC        | HPS switch       | IBM AIX          |
| Linux Supercluster | Distributed memory       | 512 (dual-cpu)       | Pentium III 733 MHz | Myrinet          | Linux            |
| Linux Cluster      | Distributed memory       | 512 (dual-cpu)       | Pentium III 500 MHz | Myrinet          | Linux            |
| SGI Origin2000     | Shared memory            | 64                   | 195 MHz R10000      | Gigabit Ethernet | SGI IRIX         |

**Table 2. Memory hierarchy of the different systems.**

| System name        | Memory | CPU Type            | L1 Cache Size                   | L2 Cache Size |
|--------------------|--------|---------------------|---------------------------------|---------------|
| IBM SP             | 40GB   | 120 MHz P2SC        | I-cache: 32KB<br>D-cache: 128KB | N/A           |
| Linux Supercluster | 256GB  | Pentium III 733 MHz | 64KB                            | 256KB         |
| Linux Cluster      | 128GB  | Pentium III 500 MHz | 64KB                            | 256KB         |
| SGI Origin2000     | 16GB   | R10000 195 MHz      | 64KB                            | 4MB           |

The Linux Supercluster Los Lobos has 256 individual computing nodes; each node consists of two Intel Pentium III 733 MHz CPUs and 1GB of RAM, for a total of 512

processors. A high-speed, low latency Myrinet network, LANai7, interconnects the 256 nodes, and a Fast Ethernet network connects the cluster to file servers and the

Internet. We use MPICH-Myrinet/GNU library to compile all benchmarks with '-O3' option, and run them using one process per node. Chiba City has 256 dual-cpu Pentium III 500 MHz nodes with 512 MB of RAM and 9GB of local disk. There are eight storage nodes with 300 GB of disk, 64-bit Myrinet high performance network, and switched gigabit Ethernet management network. We also use MPICH-Myrinet/GNU library to compile all benchmarks with '-O3' option, and run them using one process per node on Chiba City.

The IBM SP has 80 SP processors, with 120 MHz of P2SC CPUs, approximately 40 GB of memory, 1TB of on-line storage and 60TB off-line storage. We use MPI library provided by IBM to compile all benchmarks with '-O3' option, and again execute one process per node. SGI Origin2000 has 64 processors with 195 MHz R10000 CPUs and 16 GB memory. We use MPI library provided by SGI to compile all benchmarks with '-O3' option.

#### 4.1 Case Study: SP Benchmark

SP (Scalar Pentadiagonal solver) is an application benchmark, which solves three sets of uncoupled systems of equations, first in the x, then in the y, and finally in the z dimension. These systems are scalar pentadiagonal. We have divided the application benchmark into eight kernels:

- 1) **INITIALIZATION:** Initializes values.
- 2) **COPY FACES:** Phase one computation of the right hand side.
- 3) **TXINVR:** Phase two computation of the right hand side.
- 4) **X SOLVE:** Solves the problem in the x dimension.
- 5) **Y SOLVE:** Solves the problem in the y dimension.
- 6) **Z SOLVE:** Solves the problem in the z dimension.
- 7) **ADD:** Updates values.
- 8) **FINAL:** Verifies the solution integrity.

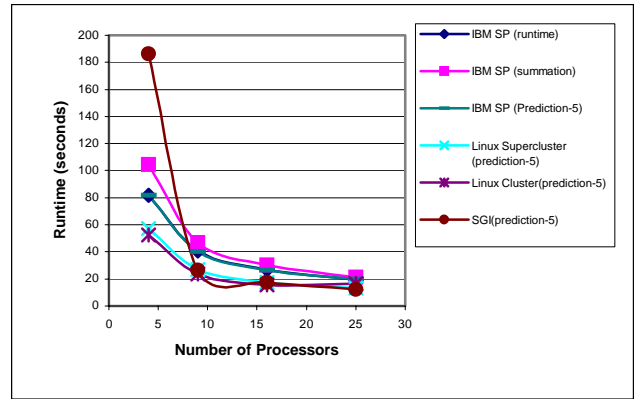
The SP control flow entails executing steps 2 through 7 in a loop. Hence, we focus on obtaining coupling values for the six main kernels. The parallel SP benchmark requires that the number of processors is a square. The datasets and the number of iterations for SP benchmark are showed in Table 3. The kernels INITIALIZATION and FINAL are called once for the three datasets and the kernels COPY FACES, TXINVR, X SOLVE, Y SOLVE, Z SOLVE, and ADD are called 400 times.

**Table 3. Datasets used with the NPB SP.**

| SP Dataset | Problem Size    | Number of Iterations |
|------------|-----------------|----------------------|
| W          | 36 x 36 x 36    | 400                  |
| A          | 64 x 64 x 64    | 400                  |
| B          | 102 x 102 x 102 | 400                  |

#### 4.2 Different System Architectures

In this subsection, we use the SP benchmark with a given data size W as the basis to explore the ability to reuse coupling values from different system architectures. For the data size A and B, the results indicated similar trends.



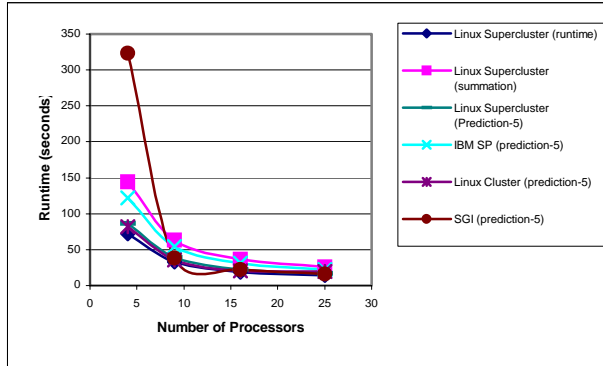
**Figure 1. Experimental and predicted execution times for the SP benchmark with data size W executed on the IBM SP.**

Figure 1 gives the experimental and predicted results of the SP benchmark with data size W executed on the IBM SP using coupling values from the other three architectures. The legend in Figure 1 has the following meaning: **IBM SP (runtime)**: actual runtime on IBM SP; **IBM SP (summation)**: prediction using the summation technique that entails summing the individual kernel execution times; **IBM SP (prediction-5)**: prediction using the coupling values for a 5-kernel chain from the IBM SP; **Linux Supercluster (prediction-5)**: prediction using the coupling values for a 5-kernel chain from the Linux Supercluster Los Lobas; **Linux Cluster (prediction-5)**: prediction using the coupling values for a 5-kernel chain from Linux Cluster Chiba City; **SGI (prediction-5)**: prediction using the coupling values for a 5-kernel chain from the SGI Origin2000.

**Table 4. Percentage of average relative errors for Figure 1.**

| Method                            | Average Relative Error |
|-----------------------------------|------------------------|
| Summation                         | 15.95%                 |
| IBM SP (prediction-5)             | 0.07%                  |
| Linux Supercluster (prediction-5) | -32.49%                |
| Linux Cluster (prediction-5)      | -33.87%                |
| SGI (prediction-5)                | 58.70%                 |

A comparison of the results is given in Table 4. The results indicate that, as expected, the best predictor is the coupling values from the IBM SP. Use of the coupling values from the Linux clusters and the SGI were significantly worse than the use of coupling values from the IBM SP as well as the summation technique. It is interesting to note that the average relative errors from both Linux clusters are very close, despite the difference in processor speed and memory size.



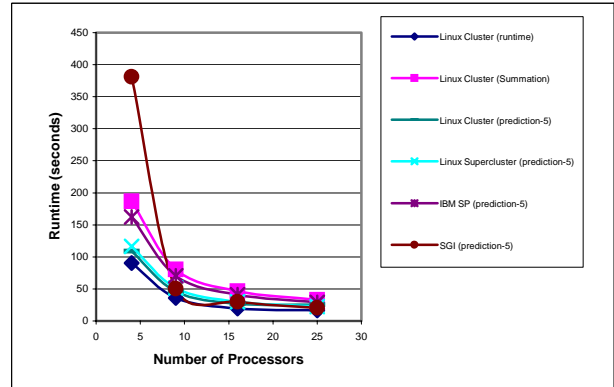
**Figure 2. Experimental and predicted execution times for the SP benchmark with data size W executed on the Linux Supercluster.**

Figure 2 gives the experimental and predicted results of the SP benchmark with data size W executed on the Linux Supercluster using coupling values from the different architectures. The legend in Figure 2 has the following meaning: **Linux Supercluster(runtime)**: actual runtime on Linux Supercluster; **Linux Supercluster (summation)**: prediction using the summation technique; **Linux Supercluster (prediction-5)**: prediction using the coupling values for a 5-kernel chain from the Linux Supercluster; **IBM SP(prediction-5)**: prediction using the coupling values for a 5-kernel chain from the IBM SP; **Linux Cluster (prediction-5)**: prediction using the coupling values for a 5-kernel chain from the Linux Cluster Chiba City; **SGI (prediction-5)**: prediction using the coupling values for a 5-kernel chain from the SGI Origin2000.

**Table 5. Percentage of average relative errors for Figure 2.**

| Method                            | Average Relative Error |
|-----------------------------------|------------------------|
| Summation                         | 93.15%                 |
| Linux Supercluster (prediction-5) | 21.00%                 |
| Linux Cluster (prediction-5)      | 19.19%                 |
| IBM SP (prediction-5)             | 65.94%                 |
| SGI (prediction-5)                | 99.87%                 |

Table 5 presents the comparison of the graph in Figure 2 in terms of average relative error. The results in Table 5 indicate that the coupling values from the two Linux clusters, again, had very similar execution time, in terms of relative error, in comparison to the other architectures; as expected these two machines provided the best predictions of the actual execution time. The use of the coupling values from the other machines resulted in a large relative error with the coupling values from the IBM SP performing better than that of the summation technique.



**Figure 3. Experimental and predicted execution times for the SP benchmark with data size W executed on the Linux Cluster.**

Figure 3 gives the experimental and predicted results of the SP benchmark with data size W executed on the Linux Cluster using coupling values from different architectures. The legend in Figure 3 has the following meaning: **Linux Cluster(runtime)**: actual runtime on the Linux Cluster; **Linux Cluster (summation)**: use of the summation technique; **Linux Cluster (prediction-5)**: prediction using the coupling values for a 5-kernel chain from the Linux Cluster; **IBM SP (prediction-5)**: prediction using the coupling values for a 5-kernel chain from the IBM SP; **Linux Supercluster (prediction-5)**: prediction using the coupling values for a 5-kernel chain from the Linux Supercluster Los Lobas; **SGI (prediction-**

5): prediction using the coupling values for a 5-kernel chain from the SGI Origin2000.

**Table 6. Percentage of average relative errors for Figure 3.**

| Method                            | Average Relative Error |
|-----------------------------------|------------------------|
| Summation                         | 117.78%                |
| Linux Cluster (prediction-5)      | 36.01%                 |
| Linux Supercluster (prediction-5) | 40.62%                 |
| IBM SP (prediction-5)             | 90.94%                 |
| SGI (prediction-5)                | 109.80%                |

Table 6 presents the comparison of the graph in Figure 2. Again similar results and trends occur, for which the coupling values from the Linux systems resulted in similar and the best predictions; the use of coupling values from the other architectures resulted in very poor predictions.

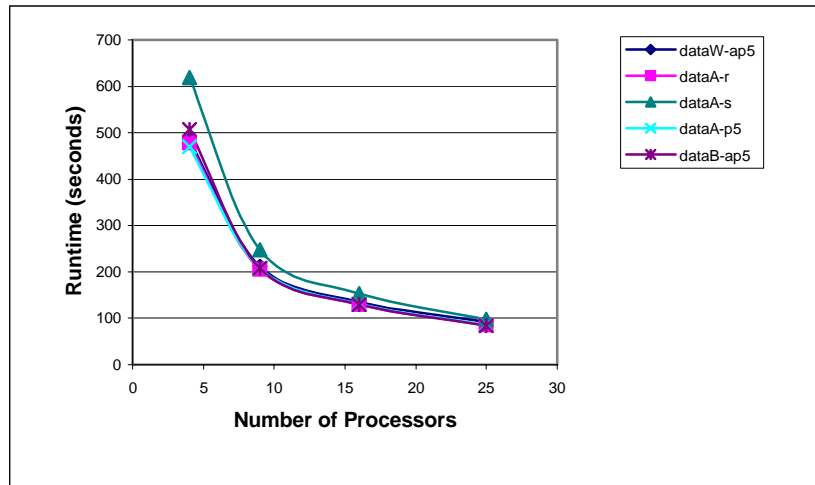
In summary, the results indicate that coupling values can be reused across similar classes of parallel machines, with the classes consisting of message-passing with clusters, message-passing with the proprietary network, and distributed shared memory.

### 4.3 Different Dataset Sizes

In this subsection, again we use the SP benchmark to explore the ability to reuse coupling values for different dataset sizes given in Table 3. We only present the results from the execution of the SP benchmark on the IBM SP and the Linux Supercluster Los Lobas for the sake of space.

#### 4.3.1 IBM SP

Figure 4 gives the experimental and predicted results of the SP benchmark with dataset size A executed on IBM SP using coupling values from the different architectures. The legend in the graph has the following meaning: **dataW-ap5**: prediction using the coupling values for a 5-kernel chain from dataset size W; **dataA-r**: actual runtime for data size A; **dataA-s**: prediction using the summation technique; **dataA-p5**: prediction using the coupling values for a 5-kernel chain from the dataset size A; **dataB-ap5**: prediction using the coupling values for a 5-kernel chain from the dataset size B.



**Figure 4. Experimental and predicted exec. times for the SP benchmark with dataset A executed on the IBM SP.**

**Table 7. Percentage of relative errors for Figure 4.**

| Method    | 4 processors: Relative Error | 9 processors: Relative Error | 16 processors: Relative Error | 25 processors: Relative Error |
|-----------|------------------------------|------------------------------|-------------------------------|-------------------------------|
| dataA-s   | 29.09%                       | 20.10%                       | 18.04%                        | 14.93%                        |
| dataA-p5  | -1.83%                       | 1.08%                        | 1.32%                         | 0.48%                         |
| dataW-ap5 | 1.42%                        | 2.76%                        | 4.26%                         | 8.13%                         |
| dataB-ap5 | 5.77%                        | 0.92%                        | 0.03%                         | -1.58%                        |

The comparison of the relative errors for the different prediction methods is given in Table 7. Recall that dataset size A is the medium size problem, with W being smaller and B being larger. When using the coupling values from the smaller dataset size, W, the relative error increases with the number of processors. This results because the coupling values for dataset size W increases with increasing number of processor. Hence for dataset size W, there is less reuse of data between kernels, as the problem size per processor gets smaller. The coupling values from the larger dataset size, B, provided a very good estimate of the execution time for dataset size A, especially for the cases of 9, 16 and 25 processor. This results because the coupling values for the larger dataset size exhibit similar characteristics to dataset size A, whereby the coupling values increase at a slow rate with increase in the number of processors.

### 4.3.2 Linux Supercluster Los Lobas

Figure 5 gives the experimental and predicted results with dataset A executed on the Linux Supercluster Los Lobas using the coupling values from the other datasets. The legend in the graph has the following meaning:

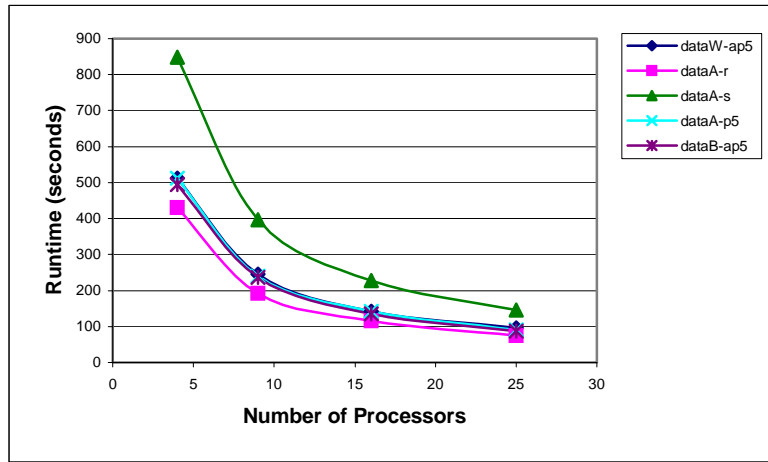


Figure 5. Experimental and predicted execution time for the SP benchmark with dataset A executed on the Linux Supercluster.

Table 8. Percentage of relative errors for Figure 5.

| Method    | 4 processors:<br>Relative Error | 9 processors:<br>Relative Error | 16 processors:<br>Relative Error | 25 processors:<br>Relative Error |
|-----------|---------------------------------|---------------------------------|----------------------------------|----------------------------------|
| dataA-s   | 97.48%                          | 106.61%                         | 95.53%                           | 93.72%                           |
| dataA-p5  | 19.41%                          | 25.18%                          | 21.78%                           | 20.68%                           |
| dataW-ap5 | 19.15%                          | 27.31%                          | 22.41%                           | 29.35%                           |
| dataB-ap5 | 14.95%                          | 23.17%                          | 15.67%                           | 16.92%                           |

**dataW-ap5**: prediction using the coupling values for a 5-kernel chain from dataset size W; **dataA-r**: actual runtime for data size A; **dataA-s**: prediction using the summation technique; **dataA-p5**: prediction using the coupling values for a 5-kernel chain from the dataset size A; **dataB-ap5**: prediction using the coupling values for a 5-kernel chain from the dataset size B.

The comparison of the relative errors for the different prediction methods is given in Table 8. The results are similar to that given with the IBM SP whereby the coupling values from the larger dataset, B, provided better predictions than the coupling values from the smaller dataset, W. It is interesting, however, that the coupling values from dataset B actually resulted in better prediction than that from the baseline dataset size, A.

In summary, the coupling values can be reused across different dataset sizes. For the case of the SP benchmark, the results indicate that coupling values from a larger dataset can be reused to provide good predictions of the execution time. For a different application, it is important to understand the trends as the coupling reuse will depend on the characteristics about the application.

#### 4.4 Different Number of Processors

In this subsection, we explore the ability to reuse coupling values across different number of processors. In particular, we use the coupling values from 4, 16, and 25 to predict the parallel application execution time on 9 processors. We only present results using the SP benchmark executed on two systems: IBM SP and Linux Supercluster Los Lobas for the sake of space.

##### 4.4.1 IBM SP

Figure 6 gives the experimental and predicted results of the SP benchmark executed on the IBM SP using coupling values from different number of processors. The legend has the following meaning: **summation-9**: prediction using the summation technique; **runtime**: actual runtime on 9 processors; **prediction-9p5**: prediction using the coupling values for a 5-kernel chain from 9 processors; **prediction-9p5**: prediction using the coupling values for a 5-kernel chain from 9 processors; **prediction-4p5**: prediction using the coupling values for a 5-kernel chain from 4 processors; **prediction-4p5**: prediction using the coupling values for a 5-kernel chain from 4 processors;

**prediction-16p5**: prediction using the coupling values for a 5-kernel chain from 16 processors; **prediction-25p5**: prediction using the coupling values for a 5-kernel chain from 25 processors.

Table 9 gives the comparison of the relative errors for the different prediction methods. The results indicate with an increase in problem size, the relative difference from using the coupling values from different number of processors decreases. For the large dataset size B, coupling values change very little with an increase in the number of processors. This results because the decrease in the problem size per processors with an increase in the number of processors does not incur major changes in the dataset size resident in the upper levels of the memory hierarchy. For the case of the small dataset size, W, the relative error is the best for the 16-processor case, and poor for the 4 and 25-processor cases. For the small dataset size, the coupling values are sensitive to the decrease in problem size per processor.

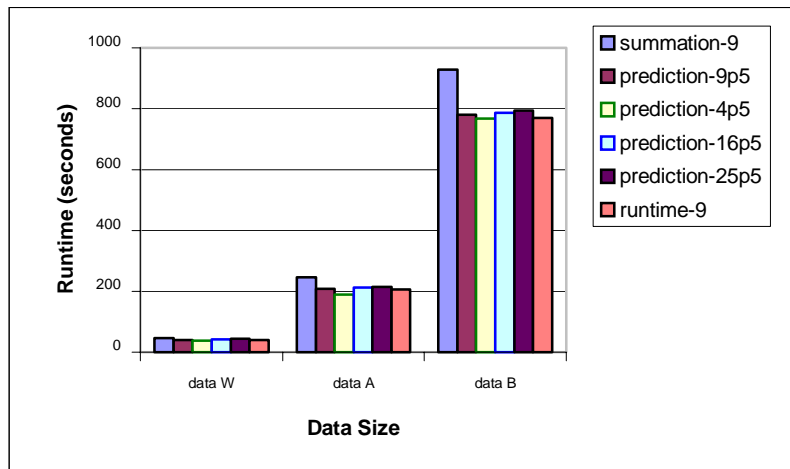


Figure 6. Experimental and predicted execution time for the SP benchmark executed on the IBM SP.

Table 9. Percentage of relative errors for Figure 6

| Method          | Data size W:<br>Relative Error | Data size A:<br>Relative Error | Data size B:<br>Relative Error |
|-----------------|--------------------------------|--------------------------------|--------------------------------|
| Summation-9     | 15.81%                         | 20.10%                         | 20.50%                         |
| Prediction-9p5  | -0.92%                         | 1.08%                          | 1.38%                          |
| Prediction-4p5  | -8.26%                         | -7.74%                         | -0.37%                         |
| Prediction-16p5 | 2.17%                          | 2.96%                          | 2.08%                          |
| Prediction-25p5 | 8.90%                          | 4.43%                          | 3.13%                          |

#### 4.4.2 Linux Supercluster Los Lobos

Figure 7 gives the experimental and predicted results of the SP benchmark executed on the Linux Supercluster using coupling values from different number of processors. The legend has the following meaning: **summation-9**: prediction using the summation technique; **runtime-9**: actual runtime on 9 processors; **prediction-9p5**: prediction using the coupling values for a 5-kernel chain from 9 processors; **prediction-4p5**: prediction using the coupling values for a 5-kernel chain from 4 processors; **prediction-16p5**: prediction using the coupling values for a 5-kernel chain from 16 processors; **prediction-25p5**: prediction using the coupling values for a 5-kernel chain from 25 processors.

Table 10 gives the comparison of the relative errors for the different prediction methods. The results are very similar to that given for the IBM SP. The results indicate with increase in the problem size, the relative difference from using the coupling values from different number of processors decreases. For the case of the small dataset size, W, the relative error is best for the 4-processor case, and poor for the 16 and 25-processor cases. For the small dataset size, the coupling values are sensitive to the decrease in problem size per processor.

In summary, the results indicate that coupling values can be reused across different number of processors. The amount of reuse is determined by the sensitivity of the coupling values to the change in problem size per processor.

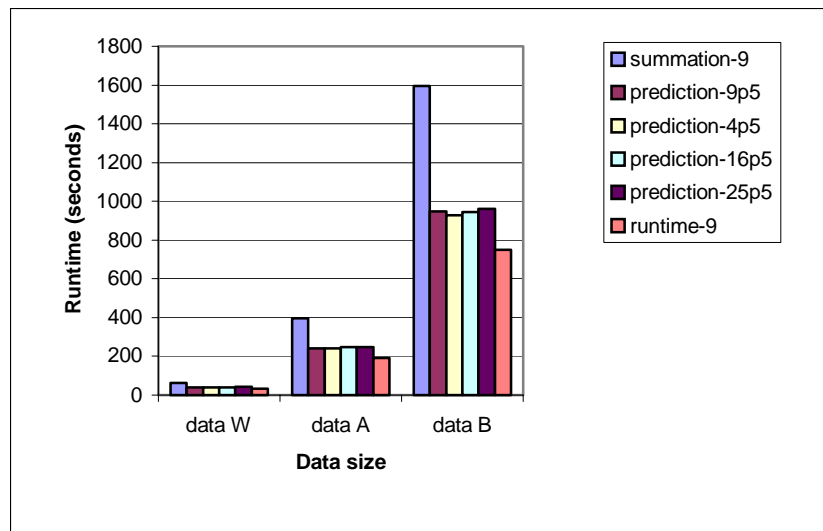


Figure 7. Experimental and predicted execution time for the SP benchmark executed on the Linux Supercluster

Table 10. Percentage of relative errors for Figure 7

| Method          | Data size W:<br>Relative Error | Data size A:<br>Relative Error | Data size B:<br>Relative Error |
|-----------------|--------------------------------|--------------------------------|--------------------------------|
| Summation-9     | 95.18%                         | 106.61%                        | 113.08%                        |
| Prediction-9p5  | 20.17%                         | 25.18%                         | 26.73%                         |
| Prediction-4p5  | 17.93%                         | 25.21%                         | 24.02%                         |
| Prediction-16p5 | 22.58%                         | 29.19%                         | 26.20%                         |
| Prediction-25p5 | 30.34%                         | 28.87%                         | 28.48%                         |

## 5. Related work

A. Snively [SC00] defined symbiosis for applications as  $\frac{TR - 1}{TM - 1}$  where TR is the throughput rate and TM is the

maximum throughput, which is similar to our coupling parameter. His work studies the interaction of separate programs on a multithreaded machine; in contrast, our work is focused on the interactions within a single application. The equation used to quantify symbiosis uses a similar ratio of measured (TR) and expected (TM)

performance. His work also indicates that dynamic scheduling of multiple, highly tuned applications produces better efficiency on the Tera MTA. We believe that our work complements this work in that we can provide the highly tuned applications to his dynamic scheduler.

R. Saavedra [SS93] did much work characterizing various benchmarks by decomposing them into high-level Fortran statements. He then counted the number of times each statement occurred in the program execution. By measuring the execution time of each statement on various target machines, he was able to predict the total execution time of the benchmarks by multiplying the statement execution times by the number of times it occurred and then summing the product over all statements. Saavedra's work demonstrated that measurements of high level constructs can result in accurate predictions (over 50% of the predictions had less than 10% error) if cache or compiler optimizations are not used. He also added terms in his model to account for cache effects. This improved the accuracy of his model to more than 80% of the predictions with less than 10% error. Our work complements Saavedra's work in quantifying and understanding the interaction between kernels. Additionally, his work is very similar to the summation method.

N. Mitchell et. al. [MH98] have studied the performance impacts of hierarchical tiling. Their technique focuses on improving a single kernel within an application, however the additional information that the coupling parameter provides indicates that the technique would be useful across kernels. The coupling parameter can indicate which cross-kernel tilings should be pursued and which should be ignored.

## 6. Summary

In this paper we explored the ability to reuse coupling values to predict parallel application execution time. In particular we explore the three dimensional space consisting of: (1) different number of processors, (2) different dataset sizes and (3) different system architectures. The goal was to determine if coupling values can be reused, implying that coupling values from two distinct points in the space are used to predict similar execution times, hence the term isocoupling. The experimental results indicated that when considering parallel systems, similar systems produced similar results.

Further, with increasing number of processors and dataset sizes, we found clear transitions with the coupling values and distinct areas where coupling values could be reused to produce good estimates of parallel execution time. Future work is focused on using isocoupling to study distributed applications on Grid computing environments.

## Acknowledgements

The authors would like to acknowledge the University of New Mexico for the use of Los Lobos, NCSA at UIUC for the use of the SGI Origin2000, and the MCS Division of Argonne National Laboratory for the use of the IBM SP and Chiba City. This work is supported in part by NSF NGS grant EIA-9974960, a NASA grant NCC 2-1363, and NSF ITR grants (ITR-0086044, ITR-0085952 and ANI-0225642).

## References

- [BH95] D. Bailey, T. Harris, et al., *The NAS Parallel Benchmarks*, Tech. Report NAS-95-020, Dec. 1995. See also <http://science.nas.nasa.gov/Software/NPB/>.
- [GT99] J. Geisler and V. Taylor, Performance coupling: A methodology for predicting application performance using kernel performance, in *Proc. of the 9<sup>th</sup> SIAM Conference on Parallel Processing for Scientific Computing*, March 1999.
- [GL94] W. Gropp, E. Lusk and A. Skjellum, *Using MPI: Portable Parallel Programming with the Message-Passing Interface*, MIT Press, Cambridge, MA, 1994.
- [MH98] N. Mitchell, K. Hogstedt, L. Carter, and J. Ferrante, Quantifying the multi-level nature of tiling interactions, *International Journal of Parallel Programming*, 1998.
- [SC00] A. Snively and L. Carter, Symbiotic job scheduling on the Tera MTA, in *Proc. of 3<sup>rd</sup> Workshop on Multi-Threaded Execution Architecture and Compilers*, January 2000.
- [SS93] R. Saavedra and A. J. Smith, *Measuring cache and TLB performance and their effect on benchmark run times*, Technical Report CSD-93-767, University of California, Berkeley, 1993.
- [TW01] V. Taylor, X. Wu, J. Geisler, X. Li, Z. Lan, M. Hereld, I. Judson, R. Stevens, Prophesy: Automating the Modeling Process (Invited Paper), in *Proc. of the 3rd International Workshop on Active Middleware Services 2001*, in conjunction with HPDC-10, August 2001.
- [TW02] V. Taylor, X. Wu, J. Geisler, and R. Stevens, Using Kernel Couplings to Predict Parallel Application Performance, in *Proc of the 11th IEEE International Symposium on High Performance Distributed Computing (HPDC2002)*, Edinburgh, Scotland, July 24-26, 2002.