

UMA System Performance Analysis

A project executed in partial fulfillment for the requirements of
Computer Systems Performance Analysis

Teacher: Prof. T. Stricker
Assisted by: Dipl.-Inf. Chr. Kurmann

F. Gramsamer

IBM Zürich Research Laboratory
Säumerstrasse 4
CH-8803 Rüschlikon
Switzerland

March 9, 2000

Contents

| | | |
|----------|---|-----------|
| 1 | Task Description | 1 |
| 2 | Approach | 2 |
| 2.1 | Definition of Factors | 2 |
| 2.2 | Platform of Experiments | 3 |
| 2.3 | Chosing the Experimental Design | 4 |
| 2.4 | Chosing the Levels | 4 |
| 2.5 | Parameters | 6 |
| 3 | Results | 7 |
| 3.1 | Preparing the Experiments | 7 |
| 3.2 | Executing the Experiments | 9 |
| 3.3 | Confounding | 11 |
| 3.4 | Validity of results | 11 |
| 4 | Conclusions | 13 |
| | Bibliography | 15 |

List of Figures

| | | |
|-----|--|---|
| 2.1 | Multiprocessor environment is scalable from 1 to n=64 processors | 2 |
|-----|--|---|

List of Tables

| | | |
|-----|--|----|
| 2.1 | DESIRED LEVELS OF FACTORS | 4 |
| 2.2 | CHOSEN LEVELS OF FACTORS | 5 |
| 3.1 | FACTORS AND LEVELS | 7 |
| 3.2 | THE EXPERIMENTS TO BE EXECUTED | 8 |
| 3.3 | THE SIGN TABLE | 8 |
| 3.4 | MEASUREMENT RESULTS | 9 |
| 3.5 | ORDER OF RELEVANCE OF FACTORS | 10 |
| 3.6 | ORDER OF RELEVANCE OF FACTORS AND CONFOUNDED EFFECTS . . . | 11 |
| 3.7 | MEASUREMENT RESULTS (COMPLETE) | 12 |

Chapter 1

Task Description

The aim of this project is two-fold,

primary to analyze system parameters in a bus-based multiprocessor environment that potentially improve or are said to have an influence to the speed-up of scientific programs.

secondary to verify that used statistical methods really do work.

While the primary aim should be clear, from its statement, the second aim needs some more comment, to understand its meaning. It means, that levels of factors have been chosen that keep simulation time low and that allow to easily verify the results gathered from the experiments. So, in case something needs to be repeated, it can be done fast and the validity of the results can easily be cross-checked.

Chapter 2

Approach

2.1 Definition of Factors

The system under investigation is a multiprocessor environment capable in scaling from 1 (uni-processor mode) to 64 processors as displayed in figure 2.1. Since scalability is a great issue in parallel computing, and since the bus as a shared medium is more and more likely to be the performance bottleneck the more processors are connected to it, one principal factor is the number of processors.

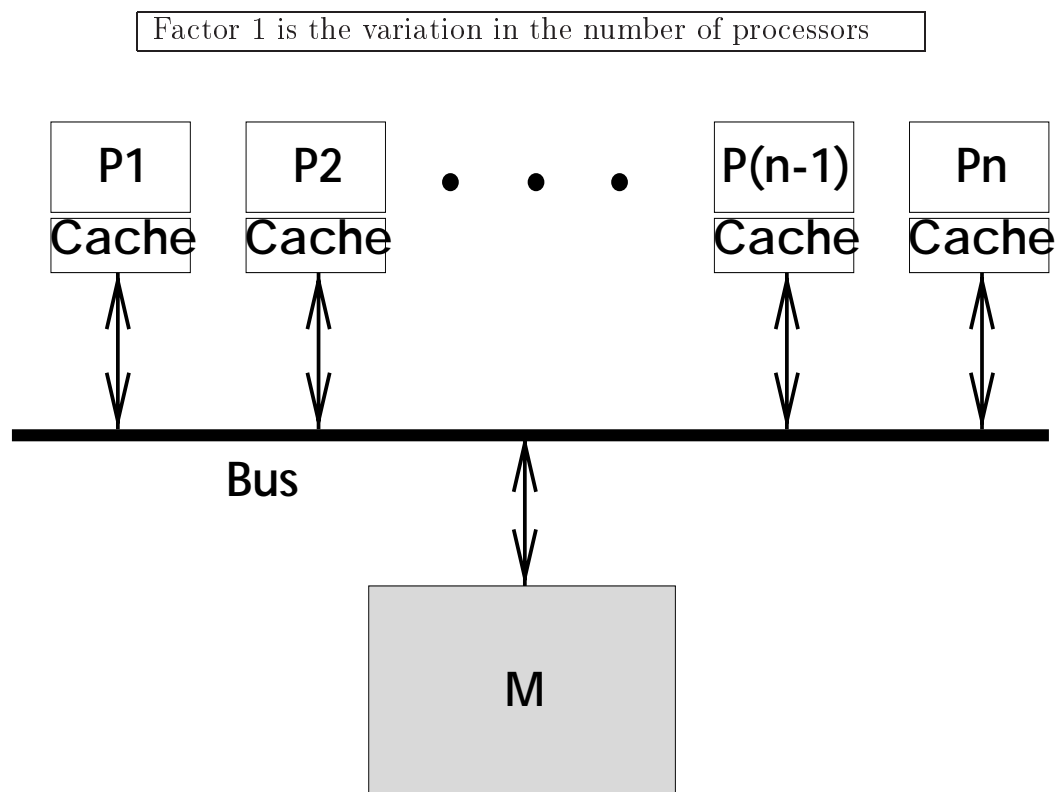


Figure 2.1: Multiprocessor environment is scalable from 1 to $n=64$ processors

Another factor of high interest in such systems is memory access latency. This can be either the latency to access a word for the first time for retrieval (read latency) or for

storage (write latency). How the system operates in detail is discussed later when the parameters are described.

Factor 2 is the variation in memory access latency

Multiprocessor systems are build to speed up the execution time of scientific algorithms or applications, to handle a high number of requests in parallel transaction systems (e.g. airline booking, database retrieval, decision support, etc.), or to be able to process huge amount of data as in some military applications (e.g. Radar image processing or the like). Short, there exists a large variety of workloads. Therefore, the workload to be exercised plays a major role. Here, it is focused on scientific workloads, which is the third factor.

Factor 3 is the variation in scientific workload

As already mentioned earlier, the processors are connected via a bus and communicate using a snoopy protocol. It has been proven in literature [4] (and it is also an intuitive assumption) that the kind of protocol also influences the results. A “smarter” protocol may reduce the number of cache misses, which in turn will reduce the bus utilisation factor which results in better performance. Hence, this factor - the type of snoopy protocol - is the fourth factor.

Factor 4 is the variation in the type of snoopy protocol

Also, an important factor in mutliprocessor system performance is the cache and its size. There are three reasons for cache misses as described in chapter 5.3 of [3]: Compulsary or cold misses, which occur whenever a data item is loaded the first time into the cache, say loading the “virgin copy”. Conflict misses, which would not occur in a fully associative cache, but do occur in directly mapped or set-associative caches. And capacity misses, which occur because the problem size does not fit into the cache. Associativity and miss rate have been studied in detail by other researchers and a 4-way set associative cache shows acceptable results (that are in the range of fully-set associative caches) [5]. Therefore, the parameter associativity is set to 4 (4-way set associative cache) and conflict misses are not further studied. However, the cache size, addressing the capacity misses, remains interesting and is therefore the fifth factor.

Factor 5 is the variation in cache size

2.2 Platform of Experiments

Since there exist only few real systems in the world that let analyze the wished factors at once (for instance the MAGIC chip of Prof. Hennessy at Stanford University is said to be configurable to run different coherency protocols, however directory protocols), the author has decided for a memory simulator - the LIMES-Simulator. This simulator is available under the GNU-license agreement and is therefore free of charge. It is downloadable under the following URL: <http://galeb.etf.bg.ac.yu/~dav0r/limes/>. The LIMES-Simulator is a result from the studies at the Belgrade University led by Prof. Davor Magdic [1].

2.3 Chosing the Experimental Design

It would be nice to measure said factors at all desired levels to get the most accurate results (full factorial design). However, this would require $10 \times 4 \times 12 \times 4 \times 4 = 7680$ experiments as can be seen in table 2.1, mentioning those levels. Out of simple reasons the author refrains from such a solution. First of all, it would simply be too time consuming to conduct all those experiments. Second, there exist statistical methods, that allow almost the same level of accuracy with much less effort. Third, this is the reason of this project. The statistical method that allows for much less experiments is the (fractional) factorial design as described in part IV of [2]. Still, a full factorial 2^5 design would require 32 experiments. In order, to reduce the number of experiments even more, a $2^{(5-2)}$ design, requiring only 8 experiments, is chosen. This means that each factor is to be limited to two levels each. Why which level was chosen is described in the next section.

| Factors | Desired Levels | # of Levels |
|-----------------|---|-------------|
| # of processors | 1, 2, 4, 8, 12, 16, 24, 32, 48, 64 | 10 |
| Latency | 3, 8, 10, 50 (cycles) | 4 |
| Workload | SPLASH-2 suite | 12 |
| Snoopy protocol | Write invalidate/update for Write through/back caches | 4 |
| Cache Size | 256kB, 512 kB, 1MB, 2MB | 4 |

Table 2.1: DESIRED LEVELS OF FACTORS

2.4 Chosing the Levels

Clearly, any $2^{(k-p)}$ fractional factorial design allows only for 2 levels per factor. The levels that finally were chosen are listed in table 2.2.

The number of processors, 4 and 16, were elected to have a small and a medium level representation. One reason, why a level above 16 was not selected, is that simulation time increases drastically, and that bus-based systems tend to saturate at 16 processors (or higher). For instance, the OCEAN application using a problem size of 258x258 grid points runs under default ¹ configuration about 1h and 20min with 16 processors, while the simulation needs about 5h and 30min with 64 processors. The 64 processor case needs 4.125 times as long as the 16 processor case. The Radix algorithm runs in its default configuration about 6min with 16 processors, while the simulation requires roughly 1h 20min for 64 processors, which is a factor of 13.33 longer!

The access time is typically within the range of 8 - 80 clock cycles according to the lecture notes of Digitaltechnik und Rechnerstrukturen of Prof.Thiele chapter 5 “Speicherhierarchie”. Therefore, an ideal level is chosen with 3 clock cycles access latency, addressing more the question, if access latency could be reduced, would it result in a performance

¹The default configuration is described in [1].

benefit? The second level, 10, is a realistic one, but also a good case. This has been selected to keep simulation time low, but still to have a good variation.

The workload chosen is the FFT (Fast Fourier Transformation) kernel and the integer radix sort kernel. The FFT exhibits all-to-all interprocessor communication, realized in this algorithm in a staggered fashion (to avoid memory hot-spotting). The data set chosen is $n = 2^{16} = 64k$ complex doubles. This results in a FFT matrix of $\sqrt{n} \times \sqrt{n} = 256 \times 256$. Each processor (or processing node) handles a submatrix of size $\frac{\sqrt{n}}{p} \times \frac{\sqrt{n}}{p}$, where p denotes the number of processors. So, for $p = 4$, each node handles 4k data points and require 64kB of (cache) storage. In the case $p = 16$, each node processes 256 points and 4kB of (cache) storage are demanded. Since not only “raw” data points are needed for the algorithm, but there is also some overhead and the roots of unity, that by the way also span a $\sqrt{n} \times \sqrt{n}$ matrix, a cache size of 1MB seems to be a good choice for the $p = 4$ case. For $p = 16$ 1MB of cache per node seems to be more than sufficient.

The radix kernel is an iterative algorithm sorting a set of r digit keys by one digit per iteration. This algorithm is interesting, since – according to [5] – the keys are communicated through writes, rather than through reads in its permutation phase. The permutation phase also requires all-to-all communication and is needed to split the global set of keys that is ordered after x digits (after the x th iteration, $x \leq r$) into new arrays of keys, one per processing node, that can be processed locally in the next iteration.

The snoopy protocol is WTI (Write Through/write Invalidate) or Berkeley, a MESI (Modified, Exclusive, Shared, Invalid). The WTI-protocol is a two state protocol (valid, invalid), where the memory is the single unit of consistency. The bus is supposed to saturate rather quickly. For instance, on every write the memory is updated, even though the issuing processor is the “exclusive” owner of a copy. Therefore, the caches have to contend for the bus more often. The MESI protocol is a four state protocol and exhibits decent performance also for a larger number of processors connected to the bus. The WTI-protocol has been chosen, to check whether its poor performance is also reflected by the statistical analysis of the system.

As already mentioned earlier, for FFT, the cache size of 1MB seems to be fine. So, this level has been chosen and an even bigger one of 2MB, that should rule out capacity misses. Increasing the cache size is expected to have very little influence for the case of $p = 16$.

| Factors | Chosen Levels | # of Levels |
|-----------------|----------------------|-------------|
| # of processors | 4, 16 | 2 |
| Latency | 3, 10 (cycles) | 2 |
| Workload | Radix, FFT | 2 |
| Snoopy protocol | WTI, MESI (Berkeley) | 2 |
| Cache Size | 1MB, 2MB | 2 |

Table 2.2: CHOSEN LEVELS OF FACTORS

2.5 Parameters

The parameters that are used throughout the experiments are

- read latency = write latency
- a 4-set associative cache
- the cacheable unit (block) is 16B, therefore a cacheline (data) is 64B
- the current model allows only 1 level of cache

Chapter 3

Results

3.1 Preparing the Experiments

Let's assign symbols, signs and indices to transform the problem to a more abstract level that allows to better conduct the steps of a $2^{(5-2)}$ design as described in chapter 19 of [2]. This is performed in table 3.1. Here, the #CPU is assigned symbol A . A "1" in sign table 3.3 (level "1") translates into index 2. Therefore, an experiment using 16 CPUs would be identified by $A2$. Similarly, an experiment using the WTI protocol would be represented by $C1$. The sign table presented in table 3.3 (which is already filled out) determines 8 experiments that need to be executed. The sign table has been established according to chapter 19 of the book [2]. All experiments are listed in table 3.2. The measured results (column "measured t_{exe} ") of this table are then filled into column y_i of table 3.3.

| Symbol | Factor | (Index 1) Level "-1" | (Index 2) Level " 1" |
|--------|----------------|-------------------------|-------------------------|
| A | #CPU | 4 | 16 |
| B | workload | FFT | RADIX |
| C | protocol | WTI | Berkeley |
| D | cache saize | 1MB | 2MB |
| E | memory latency | 3 | 10 |

Table 3.1: FACTORS AND LEVELS

| | Identification | #CPU | workload | protocol | \$ size | latency | measured t_{exe} |
|---|----------------|------|----------|----------|---------|---------|--------------------|
| 1 | $A1B1C1D2E1$ | 4 | FFT | WTI | 2MB | 3 | 65.4 |
| 2 | $A2B1C1D2E2$ | 16 | FFT | WTI | 2MB | 10 | 154.1 |
| 3 | $A1B2C1D1E2$ | 4 | RADIX | WTI | 1MB | 10 | 52.1 |
| 4 | $A2B2C1D1E1$ | 16 | RADIX | WTI | 1MB | 3 | 29.4 |
| 5 | $A1B1C2D1E2$ | 4 | FFT | Berkeley | 1MB | 10 | 25.3 |
| 6 | $A2B1C2D1E1$ | 16 | FFT | Berkeley | 1MB | 3 | 7.3 |
| 7 | $A1B2C2D2E1$ | 4 | RADIX | Berkeley | 2MB | 3 | 7.9 |
| 8 | $A2B2C2D2E2$ | 16 | RADIX | Berkeley | 2MB | 10 | 5.2 |

Table 3.2: THE EXPERIMENTS TO BE EXECUTED

Execution time t_{exe} is measured in clock cycles by the simulator. For the evaluation a cycle time of $2\text{ns} = 500\text{ MHz}$ is assumed.

| # of Exp. | I | A | B | C | AB | AC | $BC = D$ | $ABC = E$ | y_i |
|-------------------|-------|-------|--------|---------|--------|--------|----------|-----------|-------|
| 1 | 1 | -1 | -1 | -1 | 1 | 1 | 1 | -1 | 65.4 |
| 2 | 1 | 1 | -1 | -1 | -1 | -1 | 1 | 1 | 154.1 |
| 3 | 1 | -1 | 1 | -1 | -1 | 1 | -1 | 1 | 52.1 |
| 4 | 1 | 1 | 1 | -1 | 1 | -1 | -1 | -1 | 29.4 |
| 5 | 1 | -1 | -1 | 1 | 1 | -1 | -1 | 1 | 25.3 |
| 6 | 1 | 1 | -1 | 1 | -1 | 1 | -1 | -1 | 7.3 |
| 7 | 1 | -1 | 1 | 1 | -1 | -1 | 1 | -1 | 7.9 |
| 8 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 5.2 |
| Sum | 346.7 | 45.3 | -157.5 | -255.3 | -96.1 | -86.7 | 118.5 | 126.7 | |
| $\frac{Sum}{8}$ | 43.34 | 5.66 | -19.69 | -31.91 | -12.01 | -10.84 | 14.81 | 15.84 | |
| $\frac{Sum^2}{8}$ | | 32.06 | 387.60 | 1018.41 | 144.30 | 117.45 | 219.41 | 250.83 | |
| Variation | | 1.48% | 17.86% | 46.93% | 6.65% | 5.41% | 10.11% | 11.56% | |

Table 3.3: THE SIGN TABLE

SST calculates as $SST = 2170.06$, which is the total sum of squares. It is needed to calculate the variation or the percentage on main effects.

3.2 Executing the Experiments

As already mentioned earlier, the results are filled into the y_i column of sign table 3.3. Then the sum, mean sum (in this case $\frac{sum}{8}$), its power of two are calculated per column. The SST (total sum of squares) is the summation of all power two values. SST calculates as 2170.06. The variation results from deviding the power of two per column by SST and multiplying the quotient with 100. This is displayed in table 3.3. A C++-program that has been written to calculate any 2^k design can be found on attached disk. It was compiled on Red-Hat Linus 5.2 using the GNU compiler.

The factors are sorted in order of decreasing importance in table 3.5. So, the highest percentage of variation is explained by factor C “the type of snoopy protocol” with 46.9%. It is followed with 17.9% by the B , the “workload”. Stand three is taken by E “memory access latency” that explains 10.1% of the total variation. “# of processors & workload” and “# of processors & protocol” are fifth and sixth in this list with 6.6% and 5.4%, respectively. Surprisingly, the “# of processors”, factor A , explains the least variation with 1.5%. The results will be discussed in chapter 4.

Table 3.4 summarizes the data for the $2^{(5-2)}$ fractional experimental design exercise. The numbers represent the measured execution time (t_{exe}) as already discussed in the previous section. For instance, the entry 7.9 would be the result that has been achieved by experiment number 7 ($A1B2C2D2E1$) according to experiment list 3.2 and sign table 3.3. Since 24 experiments are missing (to a 2^5 full factorial design) not all effects were determined and the one determined are confounded. This is discussed in the next section.

| | $D1E1$ | $D1E2$ | $D2E1$ | $D2E2$ |
|----------|--------|--------|--------|--------|
| $A1B1C1$ | – | – | 65.4 | – |
| $A1B1C2$ | – | 25.3 | – | – |
| $A1B2C1$ | – | 52.1 | – | – |
| $A1B2C2$ | – | – | 7.9 | – |
| $A2B1C1$ | – | – | – | 154.1 |
| $A2B1C2$ | 7.3 | – | – | – |
| $A2B2C1$ | 29.4 | – | – | – |
| $A2B2C2$ | – | – | – | 5.2 |

Table 3.4: MEASUREMENT RESULTS

Execution time is measured in clock cycles by the simulator. For the evaluation a cycle time of $2\text{ns} = 500\text{ MHz}$ is assumed. The numbers in the table are therefore in ms.

| Symbol | Percentage of Variation |
|-----------------------|-------------------------|
| <i>C</i> (protocol) | 46.9% |
| <i>B</i> (workload) | 17.9% |
| <i>E</i> (latency) | 11.6% |
| <i>D</i> (cache size) | 10.1% |
| <i>AB</i> | 6.6% |
| <i>AC</i> | 5.4% |
| <i>A</i> (CPU) | 1.5% |

Table 3.5: ORDER OF RELEVANCE OF FACTORS

3.3 Confounding

According to the book [2] chapter 19.3 there are 2^p confoundings in a $2^{(k-p)}$ design, so for $p = 2$ there are four confoundings. Table 3.3 is based on a 2^3 design and has been adapted to represent our $2^{(5-2)}$ design by replacing columns BC with D and ABC with E . So, there are the following two confoundings:

$$I = BCD, I = ABCE$$

Considering the subset the complete four confoundings are as follows:

$$I = BCD = ABCE = ADE = ABCDE$$

It is a resolution III design.

Therefore, to be precise, the percentage of variation assigned to the main effects (factor A , B , C , D , and E) and the combined influence of two effects AB , and AC in the previous section 3.5, is actually confounded with the following effects (see table 3.6).

| Confounded Effects | Percentage of Variation |
|------------------------|-------------------------|
| $I = BCD = ADE = ABCE$ | |
| $C = BD = ABE = ACDE$ | 46.9% |
| $B = CD = ACE = ABDE$ | 17.9% |
| $E = AD = ABC = BCDE$ | 11.6% |
| $D = BC = AE = ABCDE$ | 10.1% |
| $AB = CE = ACD = BDE$ | 6.6% |
| $AC = BE = ABD = CDE$ | 5.4% |
| $A = DE = BCE = ABCD$ | 1.5% |

Table 3.6: ORDER OF RELEVANCE OF FACTORS AND CONFOUNDED EFFECTS

It means, taking factor C and its confounded effects for instance, that it is assumed that the second order effect of BD together with the third order ABE , and the fourth order effect $ACDE$ are marginal compared to the main effect C .

3.4 Validity of results

All experiments were conducted two times, and the same or only slightly different results were obtained. Out of curiosity, whether the statistical methods really do work, the full 2^5 design was performed. Partial results of this exercise are shown in table 3.7. Further lists and tables are available upon request. The results confirm pretty good the major effects: factor C , B , E . B was about 6% lower than found in the fractional design, which is off by about 33%. This is rather high, and due to the high impact of the combined influence of CE . But the others are in the range of less than 1% correct. I find this very

fascinating, if one personal note might be allowed. D is off by 8% points, or about 25% of the found variation value. CE turns out to be a big player with 6.3%, as well as BC with 4.2%. In the full factorial design, the three major factors are C , B , and E , the same as in the fractional factorial design.

| | $D1E1$ | $D1E2$ | $D2E1$ | $D2E2$ |
|----------|--------|--------|--------|--------|
| $A1B1C1$ | 66.2 | 156.1 | 65.4 | 151.4 |
| $A1B1C2$ | 24.7 | 25.3 | 24.1 | 24.1 |
| $A1B2C1$ | 21.1 | 52.1 | 21.1 | 52.1 |
| $A1B2C2$ | 8.0 | 8.1 | 7.9 | 8.0 |
| $A2B1C1$ | 63.5 | 154.1 | 63.5 | 154.1 |
| $A2B1C2$ | 7.3 | 7.5 | 7.2 | 7.2 |
| $A2B2C1$ | 29.4 | 72.9 | 29.4 | 72.9 |
| $A2B2C2$ | 4.9 | 5.2 | 4.9 | 5.2 |

Table 3.7: MEASUREMENT RESULTS (COMPLETE)

Execution time is measured in clock cycles by the simulator. For the evaluation a cycle time of $2\text{ns} = 500\text{ MHz}$ is assumed. The numbers in the table are therefore in ms.

Chapter 4

Conclusions

The conclusions that can be drawn are as follows:

- The WTI-protocol (*C1*) is much more sensitive to memory access latency than the Berkeley protocol.
- For the Berkeley protocol, increasing the memory access latency increases execution time at most by 5.8% (Radix application, 16 processors, and 1MB cache size).
- Increasing the number of processors when using the WTI-protocol (*C1*) does not improve performance. The WTI-protocol is already in saturation. This was clear from the beginning and confirms the assumption made in the beginning.
- The WTI-protocol really is a performance bottleneck.
- Using the Berkeley protocol (*C2*), increasing the number of processors does improve performance. For the FFT (*B1*) workload a speedup of 3.4 can be observed, while for the RADIX (*B2*) kernel a speed-up of 1.6 is seen (by comparing lines 2 with 6 for FFT and 4 with 8 for RADIX in table 3.7). This sounds realistic, since RADIX does not scale as well as FFT [5].
- Increasing the cache size from 1MB to 2MB seems to have only very little or none influence for Berkeley protocol cases (*C2*). It seems to have more influence for the FFT workload (*B1*), than for RADIX (*B2*). It seems that the problem size of the workload is not big enough, for the larger cache to take effect. So choosing the Berkeley protocol, 16 processors and a memory latency of 3 clock cycles for the FFT workload with a problem size of 2^{20} complex data points, the following results were obtained: for a cache size of 1MB, t_{exe} is found to be 205.23 [ms], while for a cache size of 2MB, the execution time t_{exe} shrinks down to 177.35 [ms]. Please note that basically for this new scenario, the two cases *A2B1C1D1E1* and *A2B1C1D2E1* as shown in table 3.2 were rerun. This signifies a performance increase of almost 14%.
- From the last item, it becomes apparent that for the selected problem sizes, and workloads the cache size of 1MB is sufficient.

Taking out the WTI-protocol should make the number of processors a major factor. The next steps would be to find out the influence of the memory access latencies (read/write latencies as two separate and configurable entities) to the SPLASH-2 suite. The cache

size would be fixed at 1MB, and the Berkeley protocol would be used. Then, increasing the write latency (but not the read latency) in the RADIX application should show more performance impact than other applications, due to the high portion of writes (as explained in section 2.4).

To sum up, the type of protocol is the major factor in influencing the performance in a bus-based multi-processor system, if there are the two protocols WTI and MESI under consideration. The second factor, is the workload. The memory latency takes third. However, it must be considered, that this is due to its great influence when using the WTI-protocol. The WTI-protocol is not advised to be used in such systems.

Bibliography

- [1] Magdic Davor. Limes: A multiprocessor simulation environment for pc platform. *IEEE TCCA Newsletter*, March 1997.
- [2] Raj Jain. *The Art of Computer Systems Performance Analysis*. Wiley, New York, 1991.
- [3] John L. Hennessy and David A. Patterson. *Computer Architecture A Quantitative Approach*. Morgan Kaufmann, San Francisco, CA, 2 edition, 1996.
- [4] Janak H. Patel Mark S. Papamarcos. A low-overhead coherence solution for multiprocessors with private cache memories. *Proc. of 11th ISCA*, pages 348–354, May 1984.
- [5] S.C. Woo, M. Ohara, E. Torrie, J.P. Singh, and A. Gupta. The SPLASH-2 programs: Characterization and methodological considerations. *22nd Ann. Int. Symp. on Computer Architecture*, pages 24–36, 1995.