RE-EVALUATING MPEG MOTION COMPENSATION SEARCH CRITERIA

Benjamin Bishop, Thomas P. Kelliher, Robert M. Owens, Mary Jane Irwin
Department of Computer Science and Engineering
The Pennsylvania State University
University Park, PA 16802

Abstract

As processors evolve from simple scalar machines to more advanced designs using dynamic instruction scheduling, speed critical algorithms should be re-examined to ensure that the optimal strategy is being used. One important example of this class of algorithms is MPEG encoding. Although the Mean Absolute Distance approach is recognized in the literature as more efficient in terms of speed, our research indicates that the Mean Square Error approach becomes much more attractive as instruction level parallelism increases. In a machine that can exploit a high degree of instruction level parallelism, the Mean Square Error approach yields better performance. In addition, the Mean Square Error approach achieves better compression quality than the Mean Absolute Distance method.

1 Introduction

MPEG is certainly one of the most widely used forms of video compression. The importance of MPEG can be seen by the inclusion of MPEG routines in the Intel Media Benchmark [5]. In addition to its current uses, MPEG compression has applications in the areas of HDTV and Video on Demand.

In MPEG encoding, the compression algorithm relies on the comparison of motion vector errors. These errors are typically measured using one of two techniques. MAD, or Mean Absolute Distance, is the most popular error measure. It is computed by accumulating the absolute luminance difference over a certain range of pixels. MSE, or Mean Square Error, is another technique which is computed by accumulating the square of the luminance difference over a certain range of pixels.

Although there has been a great deal of work targeted at improving MPEG encoding delay, these works accept the conclusion that MAD is a more efficient search criterion than MSE [1, 3, 4]. On typical scalar machines, it is believed that MAD is the more efficient choice since it requires no time-consuming multiplication. However, MSE yields slightly higher quality compression. Our results suggest that increases in instruction level parallelism will cause MSE to become the more efficient method.

2 Background

This section provides basic information about MPEG encoding algorithms and dynamic instruction scheduling.

2.1 MPEG Encoding Algorithms

There are a variety of approaches to performing MPEG encoding. Gong et al. [1] provides a more detailed description of these techniques. MPEG compression makes use of spatial and temporal locality since similarities tend to exist between the same subsections of nearby frames. The algorithm searches for portions of a given frame which are closely repeated in later frames. These repeated portions are then reconstructed in later frames using motion vectors and error terms.

The computational problem of the encoding algorithm involves finding repeated image portions. For each block in a given frame, a window is chosen in order to limit the motion vector search area. The block is then superimposed on a series of blocks from a nearby frame within the search window. The best match can then be chosen by comparing error estimate information.

There are many different methods for selecting the series of blocks to compare with the initial block. Such search techniques include exhaustive, subsample, and hierarchical. Exhaustive search simply compares the initial block to all possible blocks in the search window of the nearby frame. Subsample search is a similar technique which compares a smaller number of uniformly distributed blocks. Hierarchical techniques involve computing the error for a certain subsample of the search window and then using a greedy approach to recursively apply the same subsample strategy at a smaller scale.

2.2 Dynamic Instruction Scheduling

A number of modern superscalar machines use dynamic instruction scheduling (DIS) in the form of a restricted dataflow, including the HP PA 8000, IBM/Motorola PowerPC 604, Intel Pentium Pro and II, and the SGI/MIPS R10000. In a restricted dataflow machine, instructions in a certain pool are eligible for execution as soon as their operands become available. These instructions can then be executed out of order on a number of functional units. To preserve the appearance of sequential execution, the results of these instructions are then committed in-order to user-visible resources, such as the memory and register file.

2.3 Error Evaluation with Dynamic Instruction Scheduling

MSE error evaluation is typically not used for MPEG encoding on general purpose processors. The main disadvantage of MSE is that it requires time consuming multiplications. On a machine with dynamic instruction scheduling (DIS), however, these latencies can be hidden by the dynamic instruction scheduler. Also, the MAD approach relies heavily on simple operations such as addition, as these are very easy to compute on scalar machines. On a machine which uses DIS and special-purpose functional units, this reliance on simple operations overburdens the addition/subtraction functional unit(s) while the multiplication unit(s) is idle. Another problem is that MAD requires the computation of absolute values. This operation can be problematic if we assume there is no absolute value instruction. This assumption appears to be valid since GCC does not generate the abs instruction for a significant number of machines. Abs is usually implemented via a small block of code containing branches. For example, consider the following MIPS assembly fragment:

```
     slt r2,r1,r0
     beq r2,r0,end
     sub r1,r0,r1
end: nop
```

Since the branch required for the multi-instruction absolute value operation is difficult to predict, frequent stalls will occur. On a scalar machine this stall will effect only one functional unit. On a highly parallel DIS machine, however, this stall could halt all functional units.

3 Simulation and Test Results

We had reported previously that the dynamic execution of decoded traces [2] could be applied to particular advantage in the execution of multimedia kernels. In later simulations, we found that an implementation of the MSE algorithm outperformed that of the MAD algorithm for our highly parallel simulated DIS processor. Since MAD is currently considered to be the more efficient algorithm, we conducted a comprehensive study of the impact of the system architecture on the performance of these benchmarks.

Tests on current processor families showed that the gap between MSE and MAD performance narrows as machines become better at exploiting instruction level parallelism (ILP). These results, presented in Table 1, were generated by compressing the Berkeley "flower" and "tennis" movies [6] on three machine families. Typically, the high ILP machines (UltraSPARC, R10000, Cyrix 686 and Pentium II) had a significantly lower performance gap between the MSE and MAD techniques than did the low ILP machines (TurboSPARC, R5000 and Pentium). One interesting exception to this trend is the Pentium II, which showed an increased performance gap over the Pentium. Due to the lack of architectural information about the Pentium II, it is difficult to guess the cause of this decrease in relative MSE performance.

Proc/Benchmark	MSE	MAD	MAD Speedup
TSPARC/Tennis	.561	.687	22.5%
USPARC/Tennis	.690	.804	16.5%
TSPARC/Flower	.485	.588	21.2%
USPARC/Flower	.598	.679	13.5%
R5000/Tennis	.976	1.162	19.1%
R10000/Tennis	2.021	2.347	16.1%
R5000/Flower	.774	.896	15.8%
R10000/Flower	1.551	1.746	12.6%
Pentium/Tennis	.439	.625	42.4%
C686/Tennis	.388	.524	35.0%
PentII/Tennis	2.098	3.251	54.9%
Pentium/Flower	.359	.480	33.7%
C686/Flower	.316	.406	28.5%
PentII/Flower	1.726	2.519	45.9%

Table 1: Actual Run-Time MPEG Compression Performance Measurements (in frames/second)

In order to acquire detailed information about the performance of the MPEG encoding search criteria for higher levels of instruction level parallelism, a simulator was constructed. The simulator was designed to simulate a typical DIS superscalar processor as well as wider issue versions. The simulated machine uses a simple RISC ISA. It operates by fetching instructions, decoding them into Pentium Pro style micro operations, adding them to a dynamically scheduled pool of instructions, issuing the instructions as their operands become available, and committing them to user-visible resources in an in-order fashion. In the issue stage, the machine can use a variable number of functional units for testing purposes.

A series of simulations was performed in order to measure the performance of each architecture with different search criteria. For this purpose, the Berkeley MPEG tool set [6] was adapted for the simulator. Figure 1 shows the results of various search criteria, in terms of instructions completed per cycle, on a 4-way issue machine. Figure 2 shows these results for an 8-way machine. As the

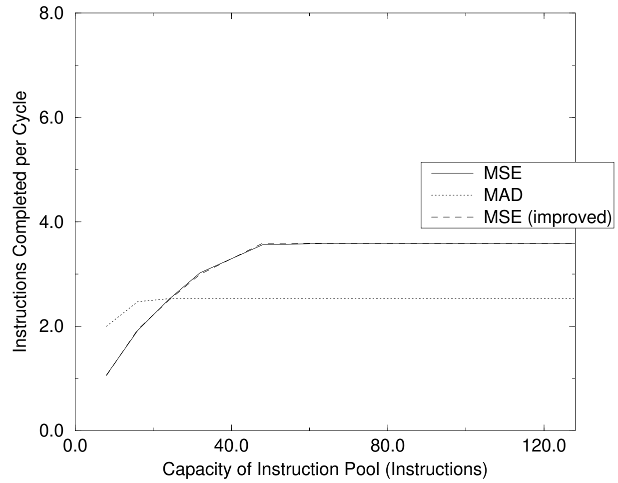

Figure 1: Comparison of Performance Between MPEG Search Criteria in an 4-way issue machine

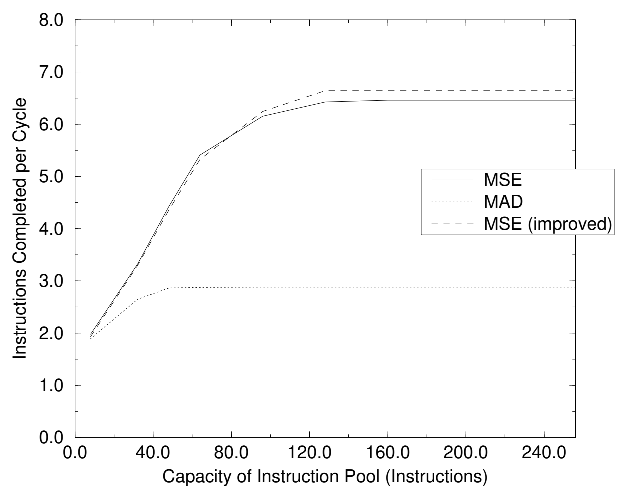

Figure 2: Comparison of Performance Between MPEG Search Criteria in an 8-way issue machine

number of functional units is increased and the instruction pool grows, the MSE search criterion shows significant gains over the MAD criterion, in terms of instructions completed per cycle. These gains can be further improved for highly parallel machines by rewriting the MSE code to expose additional parallelism (as shown in Figures 1 and 2 for MSE (improved)).

4 Conclusion

As processors become better able to exploit parallelism through the use of dynamic instruction scheduling and specialized functional units, algorithms should be re-evaluated to ensure that they are taking full advantage of these optimizations. In this paper, MPEG encoding using MSE and MAD error estimation was considered. Our results show that, for current processors, the performance gap between the MAD and MSE techniques narrows as instruction level parallelism increases. In addition, we show that MSE yields significantly higher performance than MAD when simulated on highly parallel processors which use dynamic instruction scheduling. This performance advantage is due to the less data-dependent nature of MSE, addition/subtraction functional unit saturation in MAD, the ability to hide multiplication latencies by using dynamic instruction scheduling, and the difficulty of predicting absolute value branches in MAD. An added benefit of the MSE search criterion is that it also yields better quality compression. As a whole, these results suggest that future processors will be able to perform MPEG encoding using MSE more efficiently than when using MAD.

References

[1] K.L. Gong and L.A. Rowe, "Parallel MPEG-1 Video Encoding", 1994 Picture Coding Symposium, Sacramento, CA, September 1994.

[2] B. Bishop, R. Owens, M. Irwin, "Aggressive Dynamic Execution of Decoded Traces", 1997 IEEE Workshop on Signal Processing Systems, Leicester, UK, November 1997.

[3] B. Liu and A. Zaccarin, "Fast Algorithms for Estimation of Block Motion Vectors", IEEE Trans. Video Technology. April 1993.

[4] H.G. Musmann, P. Pirsch, H.J. Gralleer, "Advances in Picture Coding", Proc. IEEE, vol. 73, no. 4, pp. 523-548, April 1985.

[5] M. Slater, "The Land Beyond Benchmarks", Comput. Commun. OEM, Mag. 4, 31, pp. 64-77. September 1996.

[6] L. Rowe et al., "Berkeley MPEG Tools" *ftp://mm-ftp.cs.berkeley.edu/pub/ multimedia/mpeg/bmt1r1.tar.gz & ftp://bmrc.berkeley.edu/pub//pub/mpeg/EncodeData/*