Automatic Generation of Matrix-Vector Code Using SPIRAL for the Power10 ISA

James Nguyen*, Sanil Rao*, Jose Moreira†, Franz Franchetti*
*Electrical and Computer Engineering, Carnegie Mellon University
†IBM
jnnguyen, sanilr, franzf@andrew.cmu.edu
jmoreira@us.ibm.com

***Abstract*—We present SPIRAL-based automatic program generation utilizing Power10's novel Matrix Multiply Assist (MMA) instructions. These MMA instructions allow for acceleration for matrix vector multiplication [2]. SPIRAL allows for the generation of linear transform programs that take advantage of these instructions to make it more efficient for developers to update their linear transform libraries [1].**

I. Introduction

Power10 is a novel instruction set architecture (ISA) developed by IBM that implements Matrix Multiply Assist (MMA) instructions that allows for acceleration for matrix vector multiplication operations [2]. Since this is a new instruction set, there are very few applications that are developed in order to utilize these new instructions, and the development cost to update older applications to take advantage of MMA instructions is high. Using SPIRAL, we are able to generate code fragments that are able to take advantage of these instructions for developers to utilize. The goal is to be able to generate large scale optimized code with Power10 operations, targeted towards high performance applications.

II. Matrix Multiply Assist Instructions

The MMA instructions are new outer product instructions for the Power10 ISA. The hardware allocates special 128-bit registers for these instructions. Current development in SPIRAL only targets double precision 64-bit floating point elements, so these vectors would each contain two 64-bit elements. Each register holds one column vector, with the number of elements dependent of the size of the element. The outer product calculations are stored in accumulators, which are software managed shadow of four consecutive registers. In the context of 64-bit double precision floating point numbers, this means that the accumulator is 4x2 sized matrix. Currently, SPIRAL generates programs that utilize the rank-64 update instructions. These instructions add the product of two vector registers into the accumulator. An example of such an operation is shown as:

$$A \leftarrow A + X * Y^T \tag{1}$$

Where X and Y are 2 element vectors containing 64-bit double precision floating point numbers, and A is the 4x2 matrix of 64-bit double precision floating point numbers.

III. Matrix-Vector Algorithm

The algorithm developed into SPIRAL, shown in figure 1, breaks down matrix-vector products into a linear combination. In the figure shown below, the input matrix, which is stored in a accumulator, is multiplied with the input vector. The outer product is computed with the column of the input matrix and a element of the input vector. This step is done using the MMA matrix-vector multiplication instruction. The result is then added to the first column of the accumulator and is repeated for the rest of the columns of the matrix. Once this is completed, then the accumulator is disassembled and the output vector is retrieved. By using the MMA instructions, we can take advantage of the instruction level parallelism of the vector instructions to do the outer products and addition in parallel. Currently, SPIRAL generates basic algorithms that multiply a constant value matrix with a input vector. Specifically, the Walsh Hadamard Transform (WHT) and the discrete Fourier Transform (DFT) are currently implemented for Power10 code generation.

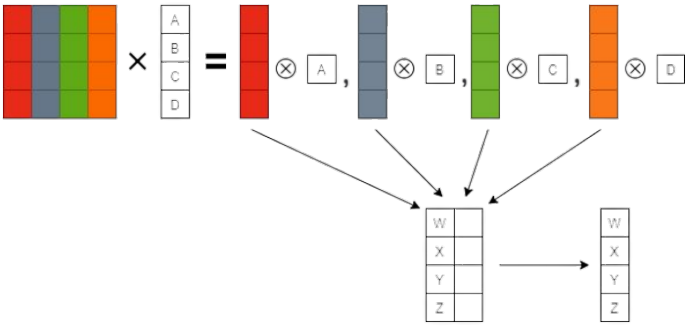

Fig. 1. Matrix-Vector Algorithm

IV. SPIRAL

SPIRAL is a program generation system targeting linear transforms and other mathematical functions, such as discrete Fourier Transforms (DFT) [1]. The objective of SPIRAL is to automate the development of performance libraries, generating architecture optimized code in a high level language such as C. As shown in figure 2, a user of SPIRAL can request code to be generated for certain linear transforms targeting specific architectures. It will then generate compiler optimized code

snippets of the linear transform for these architectures. In the case of Power10, our goal is to generate Power10 optimized code, utilizing the ISA's MMA instructions for certain linear transforms.

V. Conclusion

This work demonstrates the potential of using SPIRAL to generate linear transform programs that take advantage of Power10's novel MMA instructions. While only basic matrix vector multiplication algorithms are implemented, such as the WHT and the DFT, future work will involve more complex algorithms that can recursively break down to 4x4 blocks consisting of two accumulator blocks, such as the Fast Fourier Transform.

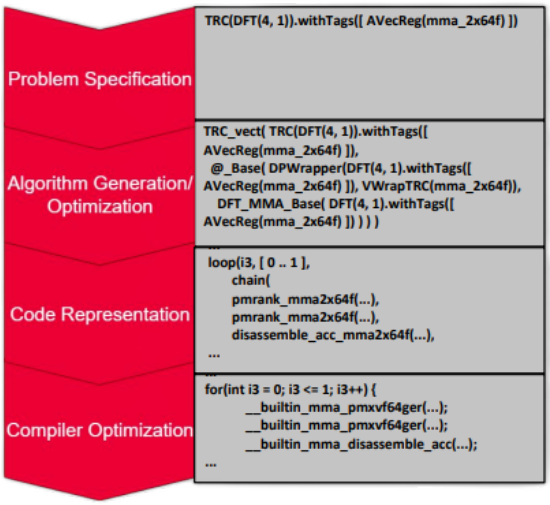

Fig. 2. Breakdown of SPIRAL generated Power10 Code

References

[1] F. Franchetti, M. Low, and M. Franusich, "SPIRAL Automating High Quality Software Production Tutorial at HPEC 2019 Tutorial based on joint work with the Spiral team at CMU, UIUC, and Drexel." [Online]. Available: https://users.ece.cmu.edu/∼franzf/papers/spiral-tutorial2019.pdf

[2] J. Moreira, et al, "A matrix math facility for Power ISA (TM) processors", arXiv preprint arXiv:2104.03142, 2021. [Online]. Available: https://arxiv.org/abs/2104.03142