Effects of High-Level Discrete Signal Transform Formulations on Partitioning for Multi-FPGA Architectures

Rafael Arce-Nazario, Manuel Jimenez and Domingo Rodriguez
Department of Electrical and Computer Engineering, University of Puerto Rico, Mayagüez, PR
{*rafael.arce,mjimenez,domingo*}*@ece.uprm.edu*

1 Introduction

The achievement of effective implementations to multi-FPGA architectures is greatly dependent on the process of partitioning. Although several automated high-level partitioning (HLP) methods have been reported [2], most of them are designed to solve general partitioning problems, and tend to apply generic local optimization techniques that miss out on alternate formulations that become apparent only with knowledge of the algorithm's functionality. The algorithmic formulation of discrete signal transforms (DST), especially that of the DFT, has been extensively studied. Automated computational algebra platforms for the algorithmic manipulation of fast transform algorithms have been proposed, as well as automated methods to optimize DST implementations to general purpose processor platforms [1]. However, these methods have yet to be successfully adapted to automated partitioning methodologies for *dedicated* distributed hardware platforms.

We hypothesize that the integration of algorithmic manipulation of DSTs into the partitioning strategy should result in a more focused exploration of the design space, and consequently, better implementations. To this end, we propose a methodology that incorporates formulation-level transformations into the partitioning optimization loop. Our methodology performs such transformations on formulations of the DST algorithm using a Kronecker Products Algebra (KPA) framework. This framework was chosen because of its representation compactness and its ability to induce reformulations on DSTs at the algorithmic level, which conveniently maps the level of abstraction sought for our methodology. Several of the components of our methodology have been implemented, and we have used them to conduct experiments to study the effect of DST formulation on partitioning results. Besides allowing us to define heuristic rules for our partitioning methodology, our tools and experiments could be used by designers to choose DST formulations targeted to established multi-FPGA topologies or to design cost-effective communication topologies for specific DSTs.

2 Partitioning methodology

Figure 1 shows a conceptual map of our partitioning methodology. The inputs are (1) a DST specified as a KPA formulation and parameterized at least by the resolution of its points, and (2) a high-level specification of the target architecture, which includes the number and logic capacity of the devices and their connection topology.

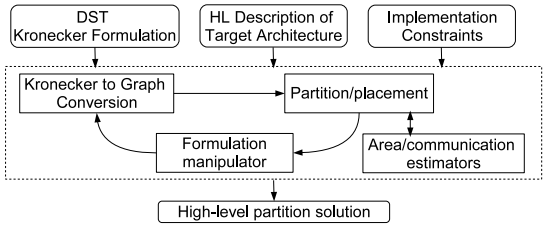

Figure 1: Proposed partitioning methodology.

The core of our methodology is an optimization loop that performs exploration in the space of equivalent formulations and partitions using DST-specific transformations. This is accomplished by the interaction of several processes. First, a component called the Kronecker-to-Graph (KTG) tool converts the algorithmic formulation to a coarse-grained dataflow graph (DFG) whose nodes denote functional primitives. Each DFG node is assigned a weight that depends on the estimated area of its represented primitive. Then, a deterministic partitioning/placement (P/P) algorithm is run on the DFG, assisted by high-level area and communication estimators. Our P/P algorithm is a communication-channel-aware, k-way partitioning adaptation of the Kernighan-Lin heuristic, which utilizes channel cost as the minimization objective. Besides trying to optimize the partitioning scheme for the current DFG, the P/P mechanism outputs indicators that instruct the heuristic formulation manipulator (FM) of what changes to perform on the

14th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM'06)
0-7695-2661-6/06 $20.00 © 2006 IEEE

formulation to promote improved solutions. Both the KTG and P/P components have been implemented. To be able to close our methodology's optimization loop, we need to have an idea of the effect of formulation properties on partitioning quality. We have conducted several experiments to gain insight into these effects and to see whether or not we can envision a strategy to heuristically guide the optimization.

3 Experiments and Results

We conducted experiments to assess the effect of two properties which can be controlled algorithmically on DST formulations: the granularity of its operands and the permutations between them. A range of sizes for five common FFT formulations where partitioned to target architectures consisting of four and eight FPGAs connected in linear array and ring topologies with crossbars. For all the experiments we assumed that communication through the crossbar has a latency cost twice that of the neighbor-to-neighbor channels.

3.1 Effects of Granularity

General-purpose HLP methods rely exclusively on graph properties (e.g. connectedness) or manual assistance for clustering DFG nodes, which helps prune the partitioning solution space. Our focus on a specific class of algorithms allows us to use algorithm-specific properties to supplement such clustering techniques. In our first experiment, we used the Cooley-Tukey factorization formula to study the effect of granularity in the partitioning of FFTs. This formula states that if $n = pm$ then $F_n = (F_p \otimes I_m)\, T_{n,m}\, (I_p \otimes F_m)\, P_{n,p}$, where F_n represents a size-n DFT, I_n is an identity matrix, $T_{n,m}$ is a diagonal matrix of weights, and $P_{n,p}$ is a stride permutation matrix. The recursive application of this formula was used to derive an exponential number of formulations, each exhibiting different levels of granularity at the computational stages of the transform. These formulations where converted to DFG and partitioned to measure the solution quality in terms of communication cost. Table 1 summarizes our results by showing the formulations that achieved minimum cost for each of the FFT sizes. Abbreviations used for the formulations correspond to the functional primitive sizes used at each of the FFT stages. The effect of granularity is evident, albeit for the general case we cannot easily establish a correspondence between granularity scheme and quality of solution. The finest grained formulations do not necessarily obtain the best results, so in many cases it would be wise to avoid these formulations as they also represent an increased exploration time.

Table 1: Granularity experiment results.

	Array 4 Topology		Array 8 Topology	
Size	Cost	Min. Cost Form.	Cost	Min. Cost Form.
32	11	2,2,2,4*	32	2,2,2,2,2*
64	22	2,2,2,2,4*	48	2,2,2,2,4
128	43	2,2,2,2,2,2,2*	92	2,2,2,2,2,4
256	86	4,4,2,2,4	132	4,2,2,2,2,4
512	171	2,2,2,2,2,2,2,2,2*	276	2,2,2,2,2,2,4,2

∗ For cases where multiple formulations achieved minimum cost we show formulation with finest granularity.

3.2 Effects of Permutations

In an effort to reflect algorithm regularity onto its final hardware implementation, the P/P process begins with a balanced horizontal linear partition of the formulation's computational structure. Hence, inter-stage permutations in the DFT formulations determine the initial partition and are expected to affect final solution quality. To test this hypothesis, and possibly detect heuristic strategies to be applied in the partitioning of FFT formulations, we used our tools to partition several sizes of five common FFT formulations. A representative example of this experiment's results is presented in Figure 2. The graph shows the percent difference in solution cost for various formulations on an architecture with four FPGAs connected in a linear array topology. Best results where obtained when starting with formulation-aware initial solutions, rather than randomly generated ones. None of the formulations exhibit a consistent advantage over others, even though some of them (e.g. Pease) begin with higher-cost initial solutions.

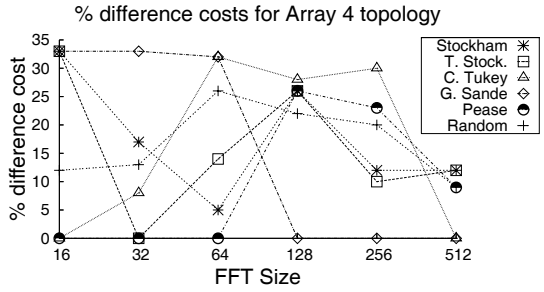

Figure 2: Permutation experiment results.

References

[1] M. Püschel, et al. SPIRAL: Code generation for DSP transforms. *Proceedings of the IEEE*, 93(2):232–275, 2005.

[2] V. Srinivasan, S. Govindarajan, and R. Vemuri. Fine-grained and coarse-grained behavioral partitioning with effective utilization of memory and design space exploration for multi-FPGA architectures. *IEEE Trans. Very Large Scale Integr. Syst.*, 9(1):140–159, 2001.