ILP-Based Scheduling with Time and Resource Constraints in High Level Synthesis[*]

Samit Chaudhuri[†] *Robert A. Walker*[‡†]
Rensselaer Polytechnic Institute
Troy, NY 12180

Abstract

In this paper, we present a formal analysis of the constraints of the scheduling problem, and evaluate the structure of the scheduling polytope described by those constraints. Polyhedral theory and duality theory are used to demonstrate that efficient solutions of the scheduling problem can be expected from a carefully formulated integer linear program (ILP). Furthermore, we present an algorithm to lower bound the resource requirement of the time-constrained scheduling problem that enables us to solve the ILP more efficiently.

1 Introduction

The scheduling problem in high-level synthesis is concerned with sequencing the operators of a control/data flow graph (cdfg) in correct order. This optimization problem, is specified in two ways: (1) *resource-constrained* scheduling (RCS) minimizes the number of control steps when the number of FU's are fixed; (2) *time-constrained* scheduling (TCS) minimizes the number of resources when the number of control steps is fixed. We can also consider a third problem, called *time-and resource-constrained* scheduling (TRCS), which optimizes a given objective function when both the number of functional units and the number of control steps are fixed. The decision problem [2] corresponding to TRCS is known to be NP-complete [8], therefore at present none of the scheduling problems can be solved in polynomial time.

To solve the scheduling problem, both heuristic scheduling algorithms and ILP-based algorithms have been used. Although an ILP formulation always solves the scheduling problem optimally, care has to be taken so that the formulation can be also be solved efficiently (because solving a general ILP is an NP-hard problem [2]). It has been said in [5] that *"formulating a good model is of crucial importance to solving the model"*. Therefore, to efficiently solve the scheduling problem, it is important to use a structured formulation, and a mathematical analysis of the constraints is needed to find a structured formulation.

Well-known ILP-based schedulers are CHARM [7], ALPS [4], and OASIC [3]. Among them, only the OASIC system has identified some structure in the (precedence) constraints. Instead of extending such analysis, recent work on ILP-based scheduling algorithms [9] has concentrated on adding additional design parameters into the formulation in order to solve a more global problem. Because of the robustness of the general ILP model, a correct formulation can always be extended with additional design parameters. However, an in-depth mathematical analysis of all the constraints of the scheduling problem is still lacking, even though it is extremely important for improving the efficiency of the ILP approach.

The motivation for this paper is not to present just another ILP model for the scheduling problem, but to formally analyze the ILP approach to the scheduling problem, which will serve as a theoretical basis for future improvement to this approach. In the next section, we present a general ILP formulation that can be used to solve the three scheduling problems. In the limited space of this paper, we will exclude RCS from our discussion. For efficiency, we add resource constraints to a TCS problem to convert it into a TRCS problem, as discussed in Section 3. The analysis of TRCS and further improvement of the formulation is outlined in Section 4. The experimental results in Section 5 are used to show the validity of the predictions made from the analysis, and the effectiveness of solving TRCS instead of directly solving TCS. This paper will assume single-cycle operators only; however, the formulation can be extended to multicycle operators.

2 ILP Formulation of the Problem

Given a cdfg let I be the index set of all operators, and $o_i \rightarrow o_j$ indicate a precedence relation, which means that operator i must finish execution before operator j can start. Suppose the cdfg is to be scheduled onto a set S of control steps. As-soon-as-possible (ASAP) and as-late-as-possible (ALAP) schedules give a continuous range $\mathcal{S}_i$ of control steps, called the *schedule interval*, over which an operator o_i can be scheduled.

The type of an FU indicates its functionality (eg. multiplication, addition). Let K be the set of types that are available. Let a_k and n_k respectively be the area and number of functional units of type $k \in K$. The type of the operators are determined by the type function $\tau : I \rightarrow K$. $\tau(i) = k$ means operator o_i is executed on a functional unit of type k.

The values of n_k and S can be fixed or variables depending on the scheduling problem being solved. In TCS, the total area $\sum_{k \in K} a_k n_k$ is minimized for a fixed S. In RCS, $|S|$ is minimized for fixed values of n_k. In TRCS, both S and n_k are fixed. In all cases the same preliminary formulation, to be presented in this section, can be used with different objective functions.

Consider the set of nodes $V = \{ (i,s) | i \in I;\ s \in S_i \}$, where a node (i,s) indicates the event that operator o_i is scheduled in control step s. Each operator o_i cor-

[*]This material is based upon work supported by the National Science Foundation under Grant No. MIP-9211323

[†]Dept. of Electrical, Computer, and Systems Engineering.

[‡]Department of Computer Science.

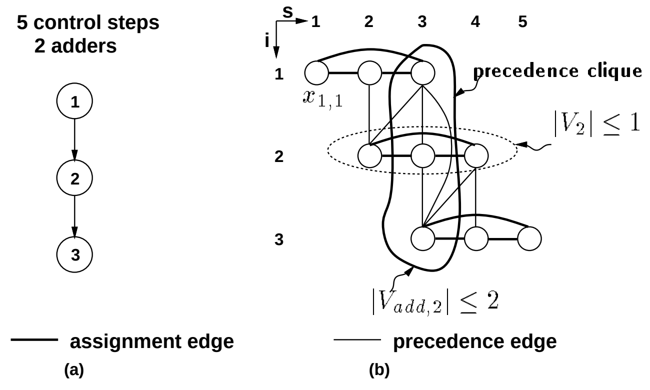

Figure 1: Constraint graph (a) The precedence graph (b) The corresponding constraint graph

responds to a set of nodes $V_i = \{ (i,s) | s \in S_i \}$. Furthermore, each functional unit type k relates, for each control step s, to a set of nodes $V_{k,s} = \{ (i,s) \mid s \in S_i\ ;\ \tau(i) = k \}$. Each feasible schedule $Q \subseteq V$ contains exactly one node from each V_i, satisfies all the precedence constraints between operators, and uses no more than the available number of functional units. The feasible schedules will be described by the following notations:

x^Q A real $|V|$-vector ($x^Q \in \mathbb{R}^{|V|}$), called the *incidence* or *characteristic* vector of Q, where $Q \subseteq V$, defined as follows:

$$x^Q_{i,s} = \begin{cases} 1, & \text{if } (i,s) \in Q \\ 0, & \text{if } (i,s) \notin Q \end{cases}$$

$\mathcal{Q}$ Set of all feasible schedules.

2.1 Constraint Graph

Before we can characterize $\mathcal{Q}$, we need to introduce the *constraint graph* (G_c), which is defined as $G_c = \{V, E_1 \cup E_2\}$. The edges in E_1 are called *assignment edges*, and the edges in E_2 are called *precedence edges.* The assignment edges connect the nodes of V in such a way that each V_i represents a clique. When stated formally, $E_1 = \{ (v_1, v_2) \mid \exists\, i \text{ such that } v_1, v_2 \in V_i \}$. The precedence edges connect two nodes in V that cannot simultaneously belong to a feasible schedule because of a precedence conflict. Formally, $E_2 = \{ (v_1, v_2) \mid v_1 = (i, s_1)\ ;\ v_2 = (j, s_2)\ ;\ s_1 \ge s_2\ ;\ o_i \to o_j \}$.

A *precedence clique* (C_p) is defined as a clique in G_c that contains at least one edge from E_2. All of these definitions are illustrated in Figure 1.

2.2 Description of the Scheduling Polytope

The *scheduling polytope* constitutes the feasible region for our integer linear programming (ILP) formulation of the scheduling problem, and is defined as the convex hull of the incidence vectors of all feasible schedules as described below:

$$P_I(\mathcal{Q}) = conv\{\, x^Q \in \mathbb{R}^{|V|} \mid Q \in \mathcal{Q} \,\}$$

We have defined $P_I(\mathcal{Q})$ in a way such that its extreme points are integral, and thus denote feasible schedules.

It is known that the worst case performance of an ILP is exponential in time. However, if we can describe the feasible region $P_I(\mathcal{Q})$ with only a set of linear equality and inequality constraints (omitting the integrality constraints), then a linear program (LP) will produce the optimal solution to the ILP. An LP can be solved in polynomial time; therefore, we will attempt to describe $P_I(\mathcal{Q})$ as tightly as possible using only equality and inequality constraints, so that the ILP solution can be efficiently found by solving a small number of LP's.

The set $\mathcal{Q}$ of all feasible schedules is described in terms of the incidence vector x of its subsets in the following way:

$$\begin{array}{rclcll} & & x_{i,s} & \ge & 0,\ \forall\,(i,s) \in V & \text{(O)} \\ |x^{V_i}| & = & \sum_{v \in V_i} x_v & = & 1,\ \forall\, i \in I & \text{(A)} \\ |x^{C_p}| & = & \sum_{v \in C_p} x_v & \le & 1,\ \forall\, C_p \in V \text{ (Sec 2.1)} & \text{(P)} \\ |x^{V_{k,s}}| & = & \sum_{v \in V_{k,s}} x_v & \le & n_k,\ \ s \in S,\ \forall\, k & \text{(R)} \\ & & x_{i,s} \text{ integer} & & & \text{(I)} \end{array}$$

The constraints (A), (P), and (R) are called the *assignment*, *precedence*, and *resource* constraints, respectively.

The above constraints can be represented in the form $\{\, x \in \mathbb{R}^{|V|}_+ \mid M_a x = 1\ ;\ M_p x \le 1\ ;\ M_r x \le n\ ;\ x \text{ integer} \,\}$, where M_a is the coefficient matrix due to the assignment constraints, M_p is the coefficient matrix due to the precedence constraints, and M_r is the coefficient matrix due to the resource constraints. If we denote the *fractional scheduling polytope* as:

$$P_F(\mathcal{Q}) = \{\, x \in \mathbb{R}^{|V|}_+ \mid M_a x = 1\ ;\ M_p x \le 1\ ;\ M_r x \le n \,\} \quad (1)$$

then we can write:

$$P_I(\mathcal{Q}) = conv\{\, x \in P_F(\mathcal{Q}) \mid x \text{ integer} \,\} \quad (2)$$

The description of $P_I(\mathcal{Q})$ requires, in addition to equality and inequality constraints, the variables to be integral. Using this preliminary description, we will examine in Section 4 how much it possible to avoid the integrality constraints so that the ILP can be solved quickly by solving a set of LP's.

3 Lower Bounding the Resources for a TCS Problem

The previous section has stated that the TCS (time-constrained scheduling) problem can be solved by minimizing $\sum_{k \in K} a_k n_k$ for a fixed S. However the ILP can be solved more efficiently if we add some resource constraints to TCS, changing the problem into a time- and resource-constrained scheduling (TRCS) problem. In this section we will discuss a method for generating tight resource constraints for TCS.

Instead of treating n_k, $k \in K$ as variables, we want to fix the values of n_k to reduce the search space. However, now we need a method to estimate an accurate lower bound on the amount of resources, so that tight values for n_k can be found. We accomplish this by quickly solving a relaxation of the TCS problem.

We relax TCS by ignoring the precedence relations between operators; thus its ILP formulation is similar to the original formulation without the precedence constraints (P). A *feasible assignment* (FA) assigns each

```
*Step 1 (initialization):*
    $p_{st}^k$ = number of operators of type $k$
        with [ASAP, ALAP] = $[s,t]$
    $P_{s,t}^k = 0$
*Step 2:*
for $s = |S| \ldots 1$ do
    $tmp_t = 0$
    for $t = s \ldots |S|$ do
        $tmp_t = p_{s,t} + tmp_t$
        $P_{s,t}^k = P_{s,t-1}^k + tmp_t$
        $n_k^\star = \max\{n_k^\star, \lceil \frac{P_{s,t}}{t-s+1} \rceil\}$
    end
end
```

Figure 2: Algorithm LBND1

operator to exactly one control step within its schedule interval. Our problem is to find the FA that requires the minimum FU area, $\sum_{k \in K} a_k n_k$. It can be easily seen that each n_k can be minimized independently, because the operators can be executed on only one type of FU and the precedence relation between operators have been relaxed.

The algorithm LBND1, presented above, can be used to compute the minimum values of n_k, $k \in K$. Let p_{st}^k be the number of operators with ASAP time s and ALAP time t. We define another quantity $P_{s,t}^k$ to denote the number of operators whose ASAP and ALAP times are within the closed interval $[s,t]$. The values of p_{st}^k can be computed while finding the ASAP and ALAP schedules. The concentration of operators in interval $[s,t]$ is indicated by $P_{s,t}^k/(t-s+1)$. It will be shown that the minimum number of FU's, $n_k^\star$, is given by the maximum operator concentration of all the intervals. The algorithm to compute the values of $P_{s,t}^k$ and $n_k^\star$ is presented in Figure 2.

To see how the algorithm works, consider the dataflow graph in Figure 3 (a). The schedule interval of each operator for a total schedule length of 4 control steps is shown in Figure 3 (b). The values of p_{st}^k for the data flow graph are given in Figure 3 (c), and the corresponding values of $P_{s,t}^k$ are given in Figure 3 (d). Maximum operator concentration occurs in the shaded box, and the corresponding value of $n_k^\star$ is found as $\lceil \frac{10}{3} \rceil = 4$.

Although the algorithm is intuitively plausible, the correctness proof is somewhat long, and will be omitted in the interest of space. From Figure 2 it can be easily seen that the complexity of LBND1 is independent of the number of operators, and is given by $O(|S|^2)$, where $|S|$ is the number of control steps.

4 Analysis of the Structure of the TRCS Problem

The previous section has shown how we covert an instance of the TCS problem to a TRCS (time- and resource-constrained scheduling) problem by using the algorithm LBND1. In the rest of the paper we will consider the ILP formulation of TRCS, for which both n_k and S have been specified. The efficiency of an

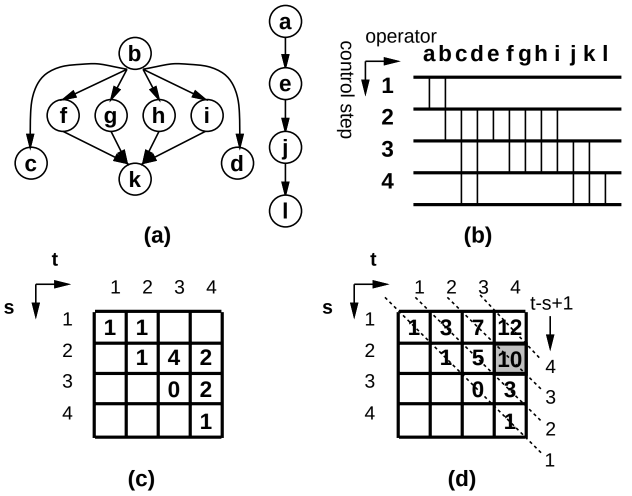

Figure 3: Execution of algorithm LBND1. **(a)** Data flow graph **(b)** Schedule intervals of operators **(c)** Values of p_{st} **(d)** Values of P_{st}. Maximum operator concentration occurs in the interval [2,4] as indicated by the shaded box.

ILP algorithm depends on how tightly we can define $P_I(\mathcal{Q})$ without using the integrality constraints. In Section 2, we first defined $P_F(\mathcal{Q})$ in terms of the assignment, precedence, and resource constraints, and then obtained $P_I(\mathcal{Q})$ by adding the integrality constraints.

The purpose of this section is to examine how close $P_F(\mathcal{Q})$ is to $P_I(\mathcal{Q})$. Although a thorough examination is as hard as solving the scheduling problem itself, we can get some useful information by selectively dropping some of the constraints.

First we drop the precedence constraints, and consider the subset of $P_F(\mathcal{Q})$, called the *resource-assignment* polytope $P_F(\mathcal{R})$, that satisfy the resource and the assignment constraints, and is described as:

$$P_F(\mathcal{R}) = \{x \in \mathbb{R}_+^{|V|} \mid M_a x = 1 \; ; \; M_r x \leq n\}$$

Next we drop the resource constraints and consider the subset of $P_F(\mathcal{Q})$ called the *precedence-assignment* polytope, that satisfy the assignment and precedence constraints, and is described as:

$$P_F(\mathcal{N}) = \{x \in \mathbb{R}_+^{|V|} \mid M_a x = 1 \; ; \; M_p x \leq 1\}$$

We can show that the polytopes $P_F(\mathcal{R})$ and $P_F(\mathcal{N})$ are integral polytopes. The proofs of these properties involve extensive use of polyhedral theory and graph theory, and are given in [1]. The significance of the these results is that, as long as the resource constraints and the precedence constraints are considered independent of each other, the constraints presented in our formulation are the tightest constraints possible.

The original scheduling polytope $P_F(\mathcal{Q})$ is the intersection two integral polytopes $P_F(\mathcal{R})$ and $P_F(\mathcal{I})$. However, this does not necessarily imply $P_F(\mathcal{Q})$ is integral. It can be easily demonstrated with a counterexample [1] that $P_F(\mathcal{Q})$ can have fractional extreme points (i.e. $P_I(\mathcal{Q}) \subset P_F(\mathcal{Q})$), so an LP-relaxation of the problem could lead to fractional solutions, and we will have to use branch-and-bound to find the integral optimal solution. In order for the branch-and-bound approach to be successful, it is important to find a sharp bound on the objective function, so that branches can be pruned efficiently.

The structure of $P_F(\mathcal{Q})$ presented above can be interpreted using duality theory [6] to prove that the bounds produced by the LP-relaxation are as good as the bounds from the Lagrangian relaxation. Lagrangian bounds are tight and have led to the success of other combinatorial optimization problems. Such tight bounds increase the likelihood that the optimum solution can be found in a small number of branches, as will be illustrated through experimental results.

In order to further improve the formulation we have to tighten the description of $P_F(\mathcal{Q})$ so that it approximates $P_I(\mathcal{Q})$ more closely. This can be done by introducing new valid inequalities which take into account the effect of the precedence and resource constraints upon one another. We will present a class of valid inequalities in the following:

Valid Inequality *Let $|x^{V_{k,s}}| \leq n_k$ be a resource constraint of $\mathcal{Q}$. Consider a minimal clique cover $V_{k,s} = \bigcup_{l=1}^{p} V_l$ where each V_l represents a clique made by precedence edges. If, for each $v \in V_{k,s}$, p_v gives the number of cliques that contain v, then the following expression is a valid inequality of $\mathcal{Q}$,*

$$\sum_{v \in V_{k,s}} c_v x_v \leq n_k \qquad (3)$$

where $c_v = \max\{1, n_k + p_v - p\}$

5 Results

The analysis of the ILP formulation presented in the previous section provides us with a theoretical ground to expect optimal solutions in a relatively few number of branches. In this section we will demonstrate the validity of this prediction using two benchmark examples: the 34-operator elliptical wave filter (EWF), and the 48-operator discrete cosine transform (DCT).

It should be noted here that any ILP approach produces optimal results, so we can not expect our schedules to be better than other ILP solutions. Instead, our objective was to offer a theoretical foundation for evaluating the ILP formulation. Thus for our purposes, we will use the number of branches taken by the ILP as the indicator of performance. We will demonstrate that the number of branches are small, as we predicted in the previous section.

The scheduling results are shown in Tables 1 and 2; we used an objective function that tries to minimize the number of registers. First we solved LBND1 to find lower bound on resources and then solved the ILP to construct the schedule. In a couple of cases, the bounds given by LBND1 were too tight for a feasible schedule; in those cases we specified a larger number of FU's until a feasible schedule could be found. The "LV" column indicate the maximum number of live variables that cross a control step boundary.

We also solved the TCS problems for the above benchmarks to observe their performance. These formulations are less structured, and are expected to require greater computation time. For EWF, the TCS problems could be solved to optimality; however, they took a larger number of branches. For DCT, the ILP solver failed to produce the optimal results in some cases even after hundreds of branches. This indicates

No. of csteps		Non-Pipelined Mult				Pipelined Mult			
Total	Loop	ALU	Mul	LV	Branch	ALU	Mul	LV	Branch
17	17	3	3	10	0	3	2	10	0
18	18	2	2	9	0	3	1	10	0
						2	2	9	0
18	16	3	2	10	0	3	1	10	0
19	19	2	2	9	0	2	1	9	0
19	17	2	2	9	0	2	1	9	2
21	21	2	1	9	0	2	1	9	1
21	19	2	1	9	0	2	1	9	0

Table 1: Scheduling Results for the Elliptic Wave Filter

No. of csteps	Non-Pipelined Mult				Pipelined Mult			
	ALU	Mul	LV	Branch	ALU	Mul	LV	Branch
7	6	5	12	1	6	8	11	1
8	5	4	12	1	5	6	13	4
9	4	3	13	2	4	6	13	1
9	4	4	13	1	5	6	13	0
9	5	4	12	1	5	7		0

Table 2: Scheduling Results for the Discrete Cosine Transform Example

that although any ILP formulation theoretically leads to optimal results, a careful choice should be made in order solve it efficiently.

6 Conclusion

In this paper, we have presented an ILP formulation of the scheduling problem, and have formally evaluated the structure of the formulation in the presence of time and resource constraints. Formal analysis has been performed to indicate that the efficiency of the ILP formulation on the benchmark examples is not an arbitrary event – we have given a theoretical basis for expecting efficient solutions from our ILP based scheduling algorithm. To further increase the efficiency of solving a TCS problem, a methodology has been presented to add resource constraints by optimally solving a relaxation of TCS.

References

[1] Samit Chaudhuri, Robert A. Walker, and John Mitchell. The Structure of Assignment, Precedence and Resource Constraints in the ILP Approach to the Scheduling Problem. To appear in *Proc. of ICCD*, 1993.

[2] M. R. Garey and D. S. Johnson, editors. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

[3] C.H. Gebotys and M.I. Elmasry. Simultaneous Scheduling and Allocation for Cost Constrained Optimal Architectural Synthesis. In *Proc. of 28th DAC*, pages 2–7, 1991.

[4] Cheng-Tsung Hwang, Jiahn-Hurng Lee, and Yu-Chin Hsu. A Formal Approach to the Scheduling Problem in High Level Synthesis. *IEEE Trans. on CAD*, 10(4):464–475, 1991.

[5] G. L. Nemhauser and L.A. Wolsey. *Optimization*, volume 1 of *Handbooks in Operations Research and Management Science*, chapter 6. Elsevier Science Publishers B. V., 1989.

[6] G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization*. John Wiley & Sons, 1988.

[7] H. Shin and N. S. Woo. A Cost Function Based Optimization Technique for Scheduling in Data Path Synthesis. In *Proc. of ICCD*, pages 424–427, 1989.

[8] J. D. Ullman. NP-Complete Scheduling Problems. *J. Comput. System Sci*, 10(10):384–393, 1975.

[9] T. C. Wilson, N. Mukherjee, M. K. Garg, and D. K. Banerjee. An Integrated and Accelerated ILP Solution for Scheduling, Module Allocation, and Binding in Datapath Synthesis. In *6th International Conference on VLSI Design*, pages 192–197, Bombay, India, Jan 1993.