The Structure of Assignment, Precedence, and Resource Constraints in the ILP Approach to the Scheduling Problem

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

Abstract

This paper presents a general treatment of the combinatorial approach to the scheduling problem, enhancing previous formulations in the literature. The focus of this paper is a formal analysis of the ILP approach, which we use to evaluate the structure of our formulation. 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 use the theory of valid inequalities to tighten the constraints and make the formulation more efficient.

1 Introduction

In high-level synthesis [10], the scheduling problem involves sequencing a set of operators into different control steps, while not using more than some specified number of functional units. In the past, methods based on heuristics and integer linear programming (ILP) have been employed to solve the problem. ILP formulations guarantee optimal results, and schedulers such as OASIC[4] and ALPS[7] have produced better schedules than heuristic algorithms, in comparable time for medium-sized problems.

However, existing ILP based scheduling algorithms have provided little formalism to indicate the quality of the corresponding formulation, i.e., how tight it is. The worst case performance of an ILP solver is exponential in time. Therefore, in order to use the ILP approach to solve practical size problems, it is important that the formulation be well structured, and formal analysis is needed to determine the tightness of a problem formulation.

The purpose of formal analysis goes far beyond theoretical interest; it is necessary for further improvement of the ILP formulation. For other NP-complete problems, most notably the traveling salesman problem (TSP), ILP approaches have optimally solved large size problems, but only after the problem constraints were carefully analyzed [9]. For the scheduling problem, we also show that new tighter constraints can be found, using the theory of valid inequalities.

Our objective is not just to present another ILP based algorithm, but to introduce a formal structure to the ILP approach of solving the scheduling problem. The analysis presented here can be used to identify the strengths and weaknesses of the other ILP formulations in the literature; but we will refrain from doing this in the interest of space.

In the next section, we describe the scheduling problem polytope. The main work is presented in Sections 3, and 4, where we describe the structure of the constraints, and the solution methodologies. The experimental results in Section 4 are used to show the validity of the predictions made from the analysis in the previous sections.

2 The Scheduling Problem Polytope

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 m_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.

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 corresponds 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. This is a set of subsets of V.

[*] Dept. of Electrical, Computer, and Systems Engineering.
[†] Department of Computer Science.
[‡] Department of Mathematics.

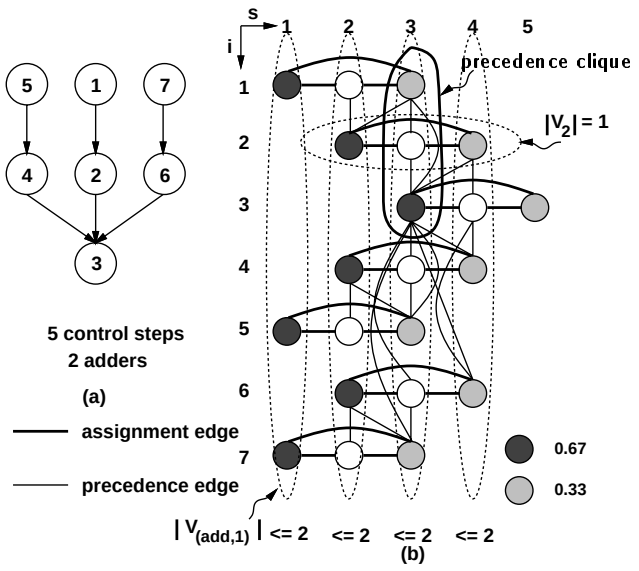

Figure 1: Constraint graph (a) The precedence graph (b) The corresponding constraint graph. The nodes have been shaded to indicate a fractional extreme point of the scheduling polytope

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, $E_1 = \{ (v_1, v_2) \mid \exists\, i \; such \; that \; v_1, v_2 \in V_i \}$. The precedence edges connect two nodes in V that cannot simultaneously belong to a feasible schedule because of a precedence conflict. Formally, $E_2 = \{ (v_1, v_2) \mid v_1 = (i, s_1) \; ; \; v_2 = (j, s_2) \; ; \; s_1 \geq s_2 \; ; \; o_i \rightarrow o_j \}$.

A *precedence clique* (C_p) is defined as a clique in G_c that has at least one edge from E_2 connecting two of its nodes. All of these definitions are illustrated in the constraint graph of Fig 1(b) which corresponds to the precedence graph in Fig 1(a).

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.

In this paper, we assume single cycle operators only, although the formulation can easily be extended to multicycle operators. Under this assumption, the set $\mathcal{Q}$ of all feasible schedules is described in terms of the incidence vector x of its subsets in the following way:

$$\begin{aligned}
x_{i,s} &\geq 0, \;\; \forall\, (i,s) \in V && \text{(O)} \\
|x^{V_i}| = \sum_{v \in V_i} x_v &= 1, \;\; \forall\, i \in I && \text{(A)} \\
|x^{C_p}| = \sum_{v \in C_p} x_v &\leq 1, \;\; \forall\, C_p \in V \text{ (Sec 2.1)} && \text{(P)} \\
|x^{V_{k,s}}| = \sum_{v \in V_{k,s}} x_v &\leq m_k, \;\; s \in S, \;\; \forall\, k && \text{(R)} \\
x_{i,s} &\text{ integer} && \text{(I)}
\end{aligned}$$

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

The above constraints can be represented in the form $\{ x \in \mathbb{R}_+^{|V|} \mid M_a x = 1 \; ; \; M_p x \leq 1 \; ; \; M_r x \leq n \; ; \; x \text{ integer} \}$, 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:

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

then we can write:

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

In this section we have given a description of $P_I(\mathcal{Q})$, that in addition to equality and inequality constraints, also requires the variables to be integral. Using this preliminary description, we will examine in later sections 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 The Problem Structure

The success of a combinatorial scheduling algorithm depends on how tightly we can define $P_I(\mathcal{Q})$ without using the integrality constraints. In the previous section, 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. In Section 3.1 and 3.2 we will drop the precedence and resource constraints respectively, and in Section 3.3 we will consider all the constraints together to see how they interact.

3.1 The Polytope of the Assignment and the Resource Constraints

In this section we will drop the precedence constraints, and consider the family $\mathcal{R}$ of subsets of V that satisfy the resource and the assignment constraints. These subsets are called *feasible resource allocations*. The *resource-assignment* polytope $P_I(\mathcal{R})$ is the convex hull of the incidence vectors of all the feasible resource allocations, and is described as:

$$\begin{aligned} P_F(\mathcal{R}) &= \{x \in \mathbb{R}_+^{|V|} \mid M_a x = 1\ ;\ M_r x \leq n\} \\ P_I(\mathcal{R}) &= \text{conv}\{x \in P_F(\mathcal{R}) \mid x \text{ integer}\} \end{aligned}$$

We want to show that the assignment and the resource constraints completely describe $P_I(\mathcal{R})$, i.e. $P_F(\mathcal{R}) = P_I(\mathcal{R})$.

Lemma 1 *The matrix* $A = \begin{bmatrix} M_a \\ M_r \end{bmatrix}$ *describing the constraints of* $P_F(\mathcal{R})$ *is totally unimodular (TU).*

Proof: The rows of A are nothing but the incidence vectors of V_i and $V_{k,s}$, $i \in I, k \in K, s \in S$. From the definition it is clear that each node $(i,s) \in V$ belongs to exactly one V_i and one $V_{k,s}$. Thus each node has an entry 1 in exactly two rows; in other words, each column of A contains two 1's, one in M_a and one in M_r. This implies that A is TU [11]. □

Proposition 1 *The polytope defined by the assignment and the resource constraints is integral, i.e.* $P_F(\mathcal{R}) = P_I(\mathcal{R})$

Proof: It is a well-known result that if A is TU then $P = \{x \in \mathbb{R}_+^n | Ax \leq b\}$ is integral when the elements of b are integral [11]. When this fact is applied to the description of $P_F(\mathcal{R})$ along with the result of the previous lemma, the proof is obvious. □

We have demonstrated that the resource constraints and the assignment constraints together describe an integral polytope $P_I(\mathcal{R})$. It is an important result that tells us that **so long the resource constraints are considered independent of the precedence constraints,** $M_r x \leq n$ **describes the tightest possible resource constraints.**

3.2 The Polytope of the Precedence and the Assignment Constraints

In this section we will drop the resource constraints and consider the subsets of V, called *feasible precedence allocations*, that satisfy the assignment and precedence constraints. Let $\mathcal{N}$ be the set of all feasible precedence allocations. The convex hull of the incidence vectors of all the feasible precedences constructs the *precedence-assignment* polytope, and can be described as follows:

$$\begin{aligned} P_F(\mathcal{N}) &= \{x \in \mathbb{R}_+^{|V|} \mid M_a x = 1\ ;\ M_p x \leq 1\} \\ P_I(\mathcal{N}) &= \text{conv}\{x \in P_F(\mathcal{N}) \mid x \text{ integer}\} \end{aligned}$$

Instead of considering only the feasible precedences, if we also include all the subsets of each feasible precedence, then we get the *monotone precedence-assignment* polytope; its fractional counterpart is given below:

$$P_F(\tilde{\mathcal{N}}) = \{x \in \mathbb{R}_+^{|V|} \mid M_a x \leq 1\ ;\ M_p x \leq 1\} \qquad (3)$$

Definition 1 A *node packing* on a graph $G = \{V, E\}$ is a set $U \subseteq V$ with the property that no two nodes in U are joined by an edge.

Definition 2 A *clique matrix* of a graph G is a 0-1 matrix K whose columns correspond to the nodes of G and rows correspond to the incidence vectors of all the maximal cliques of G.

Definition 3 The *fractional node packing polytope* of a graph G is $P = \{x \in \mathbb{R}_+^n,\ Kx \leq 1\}$, where K is the clique matrix of G.

The rows of the constraints graph's clique matrix are nothing but the rows of M_a and M_p. Therefore, the contraint graph's fractional node-packing polytope is the same as its monotone fractional precedence-resource polytope, as can be verified from (3).

Definition 4 A graph $G = \{V, E\}$ is called *transitively orientable* if each edge can be assigned a one-way direction in such a way that the resulting oriented graph (V, F) satisfies the following property:

$$(a,b) \in F \text{ and } (b,c) \in F \ \textit{ implies } \ (a,c) \in F$$

In the following proposition we show that the constraint graph is transitively orientable.

Proposition 2 *The constraint graph* G_c *is transitively orientable*

Outline of Proof: To prove the proposition, we have to find a method of assigning directions to each edge in G_c and then show that the resulting orientation is transitive. □

Proposition 3 *The fractional monotone precedence assignment polytope is integral, i.e* $P_F(\tilde{\mathcal{N}}) = P_I(\tilde{\mathcal{N}})$.

Proof: A transitively orientable graph is a *perfect graph* [5], hence the constraint graph G_c is also a perfect graph. By definition, the fractional node-packing polytope of a perfect graph is integral [11], which implies that $P_F(\tilde{\mathcal{N}})$ is an integral polytope. □

The above result immediately leads to the integrality of the fractional precedence assignment polytope, and is stated in the following corollary:

Corollary 4 *The polytope defined by the assignment and the precedence constraints is integral, i.e.* $P_F(\mathcal{N}) = P_I(\mathcal{N})$

The precedence cliques of the constraint graph G_c can also be expressed using precedence constraints similar to those used in OASIC [4], but we have used the precedence clique description here because it is more suitable for the analysis of the polytope structure.

In this section we have proved that the precedence and assignment constraints together describe an integral polytope $P_I(\mathcal{N})$. Thus, **as long as there are no resource constraints, or if the resource constraints are relaxed with penalty terms in the objective function, the resulting ILP can be solved as an LP.**

3.3 The Polytope of the Assignment, Precedence, and the Resource constraints

We have demonstrated that $P_I(\mathcal{R})$ and $P_I(\mathcal{N})$ can be completely described even if we drop the integrality constraints; we want to know if the same is true for $P_I(\mathcal{Q})$. In other words, is $P_F(\mathcal{Q}) = P_I(\mathcal{Q})$, meaning that an ILP on $P_I(\mathcal{Q})$ can be solved as an LP? The answer is "no", as will be shown in Example 1.

However, we can at least try to tighten our description of $P_F(\mathcal{Q})$ so that it approximates $P_I(\mathcal{Q})$ more closely. This can be done by introducing new tighter constraints which take into account the effect of the precedence and resource constraints upon one another. In the following two propositions, we will show two new types of tighter constraints.

We first define two quantities $npred_{i,k}$, and $nsucc_{i,k}$ which give the number of predecessors and successors of operator o_i that are executed on functional units of type k.

Proposition 5 *If $npred_{i,k} > (s-1)m_k$, or $nsucc_{i,k} > (|S| - s)m_k$ for any $k \in K$, $s \in S_i$, then $x_{i,s} = 0$ in any feasible schedule.*

Proof: The number of operators that can be scheduled on functional units of type k before and after control step s are respectively $(s-1)m_k$ and $(|S| - s)m_k$. If these numbers are less than the number of o_i's predecessors and successors of type k, then it is impossible for operator o_i to be scheduled in control step s. □

Example 1 *As an illustration of the constraints introduced by the application of the above proposition, consider the constraint graph of Fig 1. The polytope $P_F(\mathcal{Q})$ constructed from the graph has a fractional extreme point as denoted by the shading of the nodes. When an objective function composed of a weighted sum of the node variables is optimized, the linear program yields the fractional extreme point. After it is found that $npred_{3,adder} = 6 > (3-1)m_{adder} = 4$, we can set $x_{3,3} = 0$, which produces an integral solution.*

The resource constraints can also be tightened by considering the effects of the precedence relations. For example, some nodes in $V_{k,s}$ might have precedence edges among them. We can construct a minimal clique cover $V_{k,s} = \bigcup_{l=1}^{p} V_l$ where each V_l represents a clique made by precedence edges. Suppose, for each $v \in V_{k,s}$, p_v gives the number of cliques that contain v. We can safely assume $p > m_k$, because otherwise the corresponding resource constraint can be dropped.

Proposition 6 *If $|x^{V_{k,s}}| \leq m_k$ is a resource constraint of $\mathcal{Q}$, then $\sum_{v \in V_{k,s}} c_v x_v \leq m_k$ is also a valid inequality of $\mathcal{Q}$; where $c_v = \max\{1, m_k + p_v - p\}$.*

Outline of Proof: For any feasible schedule x, let:

$$\begin{aligned} V' &= \{\, v \in V_{k,s} \mid c_v > 1; x_v = 1 \,\} \\ V'' &= \{\, v \in V_{k,s} \mid c_v = 1; x_v = 1 \,\} \end{aligned}$$

Then we can show $m_k \geq \sum_{v \in V'} p_v + |V''|$. This result is used in the following deduction:

$$\begin{aligned} \sum_{v \in V_{k,s}} c_v x_v &= \sum_{v \in V'} p_v + |V''| + |V'|(m_k - p) \\ &\leq m_k + (|V'|)(m_k - p) \\ &\leq m_k \qquad \text{since } p > m_k \end{aligned}$$

□

Here we want to point out that the clique cover of $V_{k,s}$ may not be unique, and therefore, the same resource constraint can be extended in more than one way. Although each of them are stronger than the original constraint, all of them may be needed to describe $P_I(\mathcal{Q})$.

In this section we have shown that **the formulation given in Sec 2.2 is not enough to ensure $P_I(\mathcal{Q}) = P_F(\mathcal{Q})$, and have presented two methods to further tighten the formulation**.

4 Solution Methodology

In the preceding sections we presented structured constraints that tightly describe the scheduling polytope $P_I(\mathcal{Q})$. However, if we consider all three sets of constraints simultaneously, there can still be cases where the LP-relaxation produces non-integral optimal solutions, and we have to use branch-and-bound to find the integral optimal solution. In this section we will try to quantify the performance of the branch-and-bound by evaluating the quality of the bound obtained.

The bound on the objective function value is obtained by solving a relaxation of the original problem, most commonly an LP-relaxation. Another kind of relaxation, called the *Lagrangian relaxation* produces a tighter bound, and has led to the success of Lagrangian relaxation-based branch-and-bound algorithms to solve the traveling salesman problem [6], and the minimum-tardiness-scheduling problem [2]. Fisher [3] has reported that the bounds produced by Lagrangian relaxation are on the average 95% within the optimum value, and such tight bounds allow efficient pruning of the branches.

Let Z_{AR} and Z_{AP} be the Lagrangian bounds obtained when the precedence and resource constraints are relaxed respectively. Let Z_{LP} be the bound given by the LP-relaxation. The integrality of the resource-assignment and the precedence-assignment polytopes proved in the previous section implies $Z_{AR} = Z_{AP} = Z_{LP}$, which follows from the application of a known result of the duality theory [11]. The significance of this fact is that **the bounds produced by the LP-relaxation are as good as the bounds from the Lagrangian relaxation**. This is probably the reason why a small number of branches were required to optimally solve all the test problems we have run. Such experimental results will be presented in the next section.

5 Results

The analysis of the ILP formulation presented in the previous sections provides us with a theoretical ground to expect optimal solutions in a relatively few number of branches. In this section we will demonstrate the

	Schedule Length	Loop Length	Non-Pipelined Mult			Pipelined Mult			Branches
			Mul	ALU	LV	Mul	ALU	LV	
RPI-ILP	17	17	2	3	10	3	3	10	0
FDLS	17	17	2	3		3	3		
SA	17	17				3	2		
ALPS	17	17	2	3		3	3		
OASIC	17	17	2	3	10				
RPI-ILP	18	18	2	2	9	1	3	10	0
FDLS	18	18	2	2		1	3	12	
ALPS	18	18	2	2		1	3		
OASIC	18	18	2	2	10	1	3	10	
RPI-ILP	18	16	2	3	10	1	3		
ALPS	18	16	2	3		1	3		
RPI-ILP	19	19	2	2	9	1	2	9	2
FDLS	19	19				1	2	12	
ALPS	19	19				1	2		
OASIC	19	19				1	2	9	
RPI-ILP	21	21	1	2	9				0
FDLS	21	21	1	2					
ALPS	21	21	1	2					

Table 1: Scheduling results for the Elliptic Wave Filter

validity of this prediction using a number of benchmark examples: the 34-operator elliptical wave filter (ewf), the 48-operator discrete cosine transform (dct), and the 56-operator Kalman filter.

It should be noted here that any ILP formulation produces optimal results, so we don't expect our schedules to be better than other ILP solutions. Our objective is to offer a theoretical foundation for evaluating the structure of the ILP formulation. For this purpose, it is better to use the number of branches taken by the ILP as the indicator of performance. We will demonstrate the number of branches are small as we predicted in the previous section.

The scheduling results are shown in Tables 1, 2, and 3. We minimized an objective function that approximates the number live values across any control step. We compare our results (RPI-ILP) with force-directed list scheduling (FDLS)[12], simulated annealing (SA)[1], ALPS[7], OASIC[4] and SALSA [8]. For all the benchmarks, we used the standard assumptions: addition requires one control step, and multiplication requires two control steps. For the kalman filter, we used the additional provision that a multiplication and an addition can be chained into two clock cycles. The columns labeled LV gives the maximum number of live variables across any control step, and can be considered as the number of registers. The results reported by RPI-ILP were achieved in less than one second of CPU time for the ewf and the kalman example, and in a few seconds for the dct example. It can be seen that very few branches were required to compute the optimal solution in all the cases. This indicates that the ILP formulation is well behaved, and can be used to generate exact solutions to the scheduling problem in satisfactory time.

6 Conclusion

In this paper, we have presented an ILP formulation of the scheduling problem, and have formally evaluated the structure of the formulation. The analysis presented indicates that the efficiency of a carefully formulated ILP, on the benchmark examples, is not an arbitrary event. For the first time, we have given a theoretical basis for expecting efficient solutions from an ILP based scheduling algorithm. Furthermore, we have used the theory of valid inequalities to introduce tighter constraints to increase the efficiency of the algorithm.

	Schedule Length	Non-Pipelined Mult				Pipelined Mult			
		ALU	Mul	LV	Bran	ALU	Mul	LV	Bran
RPI-ILP	7	6	5	11	1	6	8	11	1
RPI-ILP	8	5	4	12	1	5	6	13	4
RPI-ILP	9	4	3	13	2	4	6	13	1
RPI-ILP	9	4	4	13	1	5	6	13	0
RPI-ILP	9	5	4	12	1	5	7		0
SALSA	7					6	8	12	
SALSA	10					4	4	15	

Table 2: Scheduling results for the Discrete Cosine Transform Example

	Schedule Length	Mul	ALU	LV	Branches
RPI-ILP	18	1	1	6	1
RPI-ILP	20	1	1	6	1

Table 3: Scheduling results for the Kalman Filter

Acknowledgement

This material is based upon work in which the first two authors were supported by the National Science Foundation under Grant No. MIP-9211323, and the third author was supported by the Office of Naval Research under Grant No. N00014–90–J–1714.

References

[1] Srinivas Devadas and A. Richard Newton. Algorithms for Hardware Allocation in Data Path Synthesis. *IEEE Trans. on Computer-Aided Design*, 8(7):768–781, July 1989.

[2] M. Fisher. Optimal Solution of Scheduling Problems Using Lagrange Multipliers: Part I. *Operations Research*, 21:1114–1127, 1973.

[3] M. Fisher. The Lagrangian Relaxation Method for Solving Integer Programming Problems. *Management Science*, 27(1):1–18, 1981.

[4] Catherine H. Gebotys and Mohammed I. Elmasry. Simultaneous Scheduling and Allocation for Cost Constrained Optimal Architectural Synthesis. In *Proc. of 28th ACM/IEEE Design Automation Conf.*, pages 2–7. ACM/IEEE, 1991.

[5] Martin Charles Golumbic. *Algorithmic Graph Theory and Perfect Graphs*. Academic Press, 1980.

[6] M. Held and R. M. Karp. The Travelling Salesman Problem and Minimal Spanning Trees: Part II. *Mathematical Programming*, 1:6–25, 1971.

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

[8] Ganesh Krishnamoorthy and John A. Nestor. Data path allocation using an extended binding model. In *Proc. of 28th ACM/IEEE Design Automation Conf.*, pages 279–284, 1992.

[9] E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, and D. B. Shmoys. *The Travelling Salesman Problem*, chapter Polyhedral Theory, pages 251–305. John Wiley & Sons, 1985.

[10] M. C. McFarland, A. C. Parker, and R. Camposano. The High Level Synthesis of Digital Systems. *Proceedings of the IEEE*, 78(2):301–318, February 1990.

[11] George L. Nemhauser and Laurence A. Wolsey. *Integer and Combinatorial Optimization*. John Wiley & Sons, 1988.

[12] Pierre G. Paulin and John P. Knight. Force Directed Scheduling for the Behavioral Synthesis of ASICs. *IEEE Trans. on Computer-Aided Design*, 8(6):661–679, June 1989.