Application of the Principal Partition and Principal Lattice of Partitions of a graph to the problem of decomposition of a Finite State Machine

Subir Roy, H. Narayanan
Department of Electrical Engineering
Indian Institute of Technology, Bombay 400076, INDIA

Abstract

We relate the Principal Partition of a graph G to the problem of Finite State Machine (FSM) Decomposition by modelling the FSM as a State Transition Graph (STG) and using the underlying graph of its STG. We obtain efficient algorithms to decompose a FSM by relating the Principal Partition to the more general notion of the Principal Lattice of Partitions of an appropriately defined submodular function.

1 Introduction

The Principal Partition (PP) of a graph G [3,4,6] is the collection of all subsets of E(G) (the edge set of G) which maximize $|X| - \lambda r(X)$ for some value of $\lambda > 0$, where $r(X)$ is the rank of the subgraph of G on X and $\lambda$ is a real number. The Principal Partition can also be understood as a collection of special node partitions. Although this aspect has been largely ignored in the literature it has substantial applications. One such application, to Finite State Machine (FSM) decomposition is described in this paper.

A related and more general notion is that of the Principal Lattice of Partitions (PLP) of a Bipartite Graph [5,8]. In this paper we show that application of this technique permits us greater flexibility in the choice of measures of optimality for evaluating the quality of decomposition of a given prototype FSM into several smaller FSMs.

In this paper we use the State Transition Graph (STG) to describe our strategy for FSM decomposition. Several cost criteria have been reported in the literature to evaluate a decomposition. Some of these are 1). Minimum number of states and edges in the interacting FSMs obtained after decomposition, 2). Minimum interaction between the component FSMs, 3). Ease of state assignment in each of the component FSMs as compared to the original FSM, 4). Simpler logic to realize the next state functions and the output functions of each component FSM.

In this paper we concentrate on minimizing the interaction between the component FSMs. We will show that construction of the Principal Partition of the underlying undirected graph G of the STG of an FSM yields a number (usually large) of partitions $\Pi'$ of nodes in our FSM which are optimal in the sense that if $\Pi'$ has k blocks no partition of the vertex set into k blocks has less number of edges with endpoints in different blocks. Since there are fast algorithms available for building the Principal Partition as well as the PLP the value of this technique is enhanced.

2 FSM Decomposition

Below we informally describe a simple strategy for constructing an essentially equivalent decomposed FSM to the prototype FSM given a partition $\Pi_5$ of its states. An analogous edge based decomposition can be given using a specified partition of the inputs. We assume the Moore machine model for our FSM. Consider the STG of a FSM as shown in fig.1(a) with $\Pi_S = \{\{s_0,s_1,s_2,s_3,s_4\},\{s_5,s_6,s_7\},\{s_8,s_9,s_{10},s_{11},s_{12},s_{13}\}\}$. The internal edges corresponding to the transitions between states in the same block are suppressed.

The equivalent decomposed machine would consist of FSM1, FSM2, and FSM3 each with its own pseudo reset state. If there is a state transition between states, in any block $S_i$ in the prototype, there would be a corresponding transition in FSM$i$, whereas FSM2 and FSM3 would remain in their pseudo reset state. If there is a transition say between $s_2$ and $s_3$ in the prototype FSM due to an input $i_j$, the machine FSM1 will move to its pseudo reset state while FSM3 would respond to an input which is derived from the (state, input) pair ($s_2, i_j$) and move from its pseudo reset state to $s_3$. 

0-7803-1254-6/93$03.00 © 1993 IEEE

2564
It can be seen that the pseudo reset states for FSM1, FSM2 and FSM3 have as many edges incident on them as the number of edges in the cut separating blocks S1, S2 and S3 respectively from the rest of the graph in the prototype FSM1. We may, depending on the context assume that the complexity of this decomposition is an increasing function of either the total number of edges or the total number of (input) kinds of edges which lie between the blocks of the specified partition of the states. Other decomposition models satisfying similar criteria as above have also been reported [9]. Below we show that finding partitions of nodes which minimize the number of such inter-block edges is related to the Principal Partition and PLP problems.

3 Principal Partition and Principal Lattice of Partitions of a Graph

The problem of FSM partitioning becomes NP Complete if we impose conditions such as (1) The number of blocks equals a specified size k, (2) That the blocks be of equal size, and (3) The total number of input overlaps is a minimum. The technique of Principal Partition (and more generally, PLP) [4,5] optimizes partitions in the above sense if (1) and (2) are relaxed.

The Principal Partition can also be understood as a special collection of partitions of V(G) as follows. Corresponding to X ⊆ E(G) we can define the partition p(X) of V(G) whose blocks are the vertex sets of the connected component of the subgraph (X, V(G)) of G on the edge set X and the vertex set V(G). We then have the following theorem.

**Theorem.** Let X' maximize |X| - λτ(X) in the graph G. Let p(X') have k blocks. Let Π be any partition of V(G) with k blocks. Then the number of edges with endpoints in different blocks of p(X') is not greater than the number of edges of G with endpoints in different blocks of Π.

The value of the Principal Partition for the FSM decomposition problem lies in the fact that for most graphs the number of sets which maximize |X| - λτ(X) for some λ would be quite large. Each such set X would yield a partition p(X) of vertices optimal in the sense of the above theorem.

We describe the PLP of a graph and show how it is related to the above ideas. Essential to the technique of PLP is a set function which is submodular. We however, restrict our attention to such functions which arise out of Bipartite Graphs. Let B = (V1, V2, E) be a bipartite graph. We use, \( L(.): 2^V \rightarrow \mathbb{R} \), the incidence function of B as defined below.

Let X be a subset of V, \( L(X) = \text{Number of vertices in } V_1 \text{ which are joined by edges to some vertex in } X \). \( L(.) \) can be shown to be a submodular function for \( L(X) + L(Y) \geq L(X \cup Y) + L(X \cap Y), \forall X, Y \subseteq V, \).

Consider a partition \( \Pi \) of \( V \) into blocks \( R_1, ..., R_n \). Define, \( (\Pi - \lambda)(\Pi) = \sum_{i=1}^{n} L(R_i) - k \lambda \). Suppose \( \Pi = \{ R_1, ..., R_n \} \) minimises \( (\Pi - \lambda) \). Let \( \Pi' = \{ R_1', ..., R_n' \} \) be another partition of \( V \) of size \( k \). Then \( (\Pi - \lambda)(\Pi') \leq (\Pi - \lambda)(\Pi) \), i.e. \( \sum_{i=1}^{n} L(R_i) - k \lambda \leq \sum_{i=1}^{n} L(R_i') - k \lambda \), or \( \sum_{i=1}^{n} L(R_i) \leq \sum_{i=1}^{n} L(R_i') \). Thus, we say that among all partitions of size \( k \) blocks, \( \Pi \) results in the minimum value of \( \sum_{i=1}^{n} L(R_i) \). The overlap in \( V_1 \) with respect to \( \Pi \) is given by \( \sum_{i=1}^{n} L(R_i) - |V_1| \). Since \( |V_1| \) is a constant, we have that \( \sum_{i=1}^{n} L(R_i) - |V_1| \leq \sum_{i=1}^{n} L(R_i') - |V_1| \), i.e \( \Pi \) also minimizes the overlap in \( V_1 \).

The PLP of \( L(.) \) is a compact description of all partitions \( \Pi \) over which \( (\Pi - \lambda) \) reaches a minimum for any real \( \lambda \). We use the technique of PLP (for details refer to [8]) to generate the Principal Partition of a graph. The relation between the two is brought out in the following theorem. For a given graph G we can build a bipartite graph \( B_G \) by taking \( V_1 = E(G), V_2 = V(G) \) with \( e \in E(G) \) joined to \( v \in V(G) \) iff \( e \) is incident on \( v \) in the graph. We then have the following theorem.

**Theorem.** Let the partition \( \Pi \) of \( V(G) \) minimize \( L - \lambda \). Let \( e(\Pi) \) be the set of edges with both endpoints in the same block of \( \Pi \). Then \( e(\Pi) \) maximizes the function \( |X| - \lambda \tau(X) \), i.e, \( e(\Pi) \) is a set in the Principal Partition of \( G \). Conversely, if \( X \) is a set in the Principal Partition of \( G \) it can be obtained as \( e(\Pi') \) for some partition \( \Pi' \) which minimises \( L - \lambda \) for some \( \lambda \).

Below we show that PLP offers considerably more flexibility in the choice of measures of optimality.

In our informal decomposition procedure the pseudo reset state had arcs corresponding to different inputs. It is conceivable that the number of such arcs is less important than the number of kinds of inputs they correspond to. [It is possible that the inputs are already available in convenient groups because of physical layout considerations]. Our PLP technique will permit us to find partitions of the nodes which minimize the number of kinds of inputs that the arcs between blocks correspond to.

Let us now describe these ideas in the context of a general graph \( G \). Let \( \Pi_B \) be a specified partition of
the edges of the graph. Let $B(G,v)$ be built as follows. We take $V_i = V(G), V_1 = \Pi_G$. A line joins $v \in V(G)$ to $X \in V_i$ (where $X$ is a block of $\Pi_G$) iff some edge in $X$ is incident on $v$ in the graph $G$. The partitions $\Pi$ of $V(G)$ in the PLP of this bipartite graph have the desired minimum overlap property. It may be noted that $B(G,v)$ is obtained from $B_0$ by fusing the vertices in $V_i ( = E(G))$ belonging to the same block of $\Pi_G$. Good algorithms are available for the construction of PLP also. The Principal Sequence of $B$ can be found in $O(V_1^2V_2^2)$ time.

Every one of the partitions of nodes of $V(G)$ resulting from the Principal Partition or PLP, as the case may be, can be examined keeping in mind other criteria such as whether the sizes of the blocks are nearly uniform. These partitions can also be used as initial seeds for heuristic algorithms which attempt, in addition to nearly minimizing edges between blocks, also to keep the sizes of the blocks balanced.

It may be remarked that the problem of finding a partition of $V(G)$ into prescribed $k$ blocks such that the number of edges between the blocks is a maximum has received considerable attention in the literature[2]. These algorithms are slow $O(\mid V \mid^{k-1})$ even for $k = 3$ and impractical for larger $k$. Our approach rests on the fact that in decomposition problems $k$ is not predetermined but can be chosen in a way that is natural to the instance in question, with the result that we can get a whole host of optimal partitions cheaper than one Min- $k$-cut (where $k$ is fixed).

We have implemented the PLP algorithm in a program called PLOP. We ran PLOP on a few large FSM's from the MCNC 1991 benchmark (eg. sand, planet and scf) and obtained encouraging results (TABLE I). We are presently enhancing PLOP to result in partitions which have balanced block sizes using partitions obtained by the PLP algorithm. For example, sand with 32 states resulted in 7 partitions of the state set besides the obvious partitions consisting of singleton blocks and the full state set being present in the Principal Sequence. For a critical value of 3.00 the corresponding state set partition was $(0,1,2,3,4), (5,6,7,8,9,12,31)$ and remaining blocks as singletons. While a critical value of 1.5 resulted in a partition consisting of a single block viz. the full set. In TABLE I, some of the examples result in only the singleton set and the full set as the Principal Sequence. We can show that, such a situation is characterised by the ease with which, the underlying state set can be partitioned by heuristic methods, without much deviation in the cost of overlap. Recently, in [10] we give two simple approxi-

**References**


TABLE I. FSM instances taken from set of MCNC benchmarks (1991)

<table>
<thead>
<tr>
<th>Name</th>
<th>Number of Inputs</th>
<th>Number of Outputs</th>
<th>Number of Product Terms</th>
<th>Number of States</th>
<th>Number of Natural Partitions</th>
<th>Largest Critical Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>sand</td>
<td>11</td>
<td>9</td>
<td>184</td>
<td>32</td>
<td>9</td>
<td>15.000</td>
</tr>
<tr>
<td>planet</td>
<td>7</td>
<td>19</td>
<td>115</td>
<td>48</td>
<td>10</td>
<td>11.000</td>
</tr>
<tr>
<td>scf</td>
<td>27</td>
<td>56</td>
<td>116</td>
<td>121</td>
<td>2</td>
<td>1.375</td>
</tr>
<tr>
<td>#208</td>
<td>11</td>
<td>2</td>
<td>153</td>
<td>18</td>
<td>2</td>
<td>10.000</td>
</tr>
<tr>
<td>s1488</td>
<td>8</td>
<td>19</td>
<td>251</td>
<td>48</td>
<td>2</td>
<td>16.000</td>
</tr>
<tr>
<td>s298</td>
<td>3</td>
<td>6</td>
<td>1098</td>
<td>218</td>
<td>2</td>
<td>6.046</td>
</tr>
</tbody>
</table>

Fig. 1a. STG of original FSM

Fig. 1b. STG's of partitioned FSMs