Solving Graph Optimization Problems with ZBDDs

Olivier Coudert
Synopsys, Inc., 700 East Middlefield Rd.
Mountain View, CA 94043

Abstract

This paper presents a ZBDD (Zero-Suppressed Binary Decision Diagram) based framework that solves a collection of graph optimization problems. We show how these problems reduce to three primitive problems, and how the later can be solved exactly using ZBDDs. The application of this framework is illustrated on multi-layer planar routing, where it can solve real-life instances that cannot be handled otherwise.

1 Introduction

A number of graph and set optimization problems occur in logic synthesis. For example, clique partitioning (equivalently, graph coloring) is used in microcode optimization [10, pp. 168–169], scheduling [7, pp. 248–252], resource binding and sharing [7, pp. 277–294] [10, pp. 230–233], constrained state encoding [10, pp. 323–327], and planar routing [3]. Set covering is used in logic partitioning [2, 16], state encoding [17, 18, 1, 15], and logic minimization [14]. BDD and ZBDD based methods have been proposed to address some of these problems on a one-to-one basis, e.g., [5, 15, 9, 12].

This paper focuses on a ZBDD based framework that solves a collection of graph and set related optimization problems. In particular, it captures most of the applications mentioned above. Section 2 shows how a number of graph problems can be expressed thanks to three primitive problems. Section 3 shows how the primitive problems are solved exactly with ZBDDs, with a complexity that is not related to the size of the original optimization problem, but to the size of the ZBDD that encodes it. Section 4 illustrates the use of this framework in solving multi-level topological planar routing.

2 Graph and Set Problems

This section presents the primitive problems from which several graph optimization problems can be expressed. An undirected graph G is denoted with (V, E), where V is the set of vertices, and E the set of edges. An undirected edge will be seen as a set of two vertices, e.g., $\{u, v\}$. Two edges are adjacent if they share a common vertex. A clique is a set of vertices that are all linked to each other by edges.

2.1 Primitive Problems

Definition 1 (Maximal cliques) *Find all cliques of a graph that are not a proper subset of another clique.*

Computing a maximum cardinality clique is NP–complete [8], but computing all proper maximal cliques (i.e., maximal w.r.t. set inclusion) is exponential[1]. We let *PropMaxClique*(G) be the set of maximal cliques of a graph G.

Let X and Y be two sets, and R be a relation defined on $X \times Y$. An element y covers x (or x is covered by y) iff $x\ R\ y$. A subset Y' of Y covers a subset X' of X iff every element in X' is covered by some element in Y'.

Definition 2 (Minimum α-covering) *Given the triple $\langle X, Y, R\rangle$, the minimum α-covering problem consists of finding a minimum cardinality subset Y' of Y that covers at least a fraction α of X.*

Definition 3 (Maximum k-cover) *Given the triple $\langle X, Y, R\rangle$, the maximum k-cover problem consists of finding k elements of Y that covers as many elements in X as possible.*

Note that minimum 1-covering is the *minimum set covering* problem. A generalization consists of assigning non negative weights w to elements of X and Y, and replacing the cardinality of a set Z with $\sum_{z \in Z} w(z)$. All these problems are NP–complete [8] (But the unweighted set cover is polynomial if $|\{x \in X \mid x\ R\ y\}| \leq 2$ for all y in Y).

2.2 Related Problems

A number of graph optimization problems can be expressed with the primitive problems, and several reductions can be obtained between all of them. We give here some of these reductions on some well known NP–complete problems [8]. Fig. 1 summarizes these *algorithmic* reductions. Fig. 2 shows the solution of these problems (in gray or with dotted lines) on a simple graph.

[1] Moon & Moser graphs have $\Omega(3^{n/3})$ [13] maximal cliques. For an easier proof of non-polynomiality, let (V, E) be a graph made of n disjoint cliques of n vertices. Thus $|V| = n^2$, and the number of maximal independent sets is n^n, each of them made of n vertices picked in the n cliques.

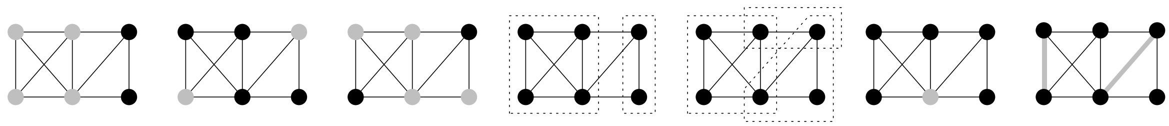

Figure 2: Clique, ind. set, vertex cover, clique partition, clique cover, dom. set, edge dom. set.

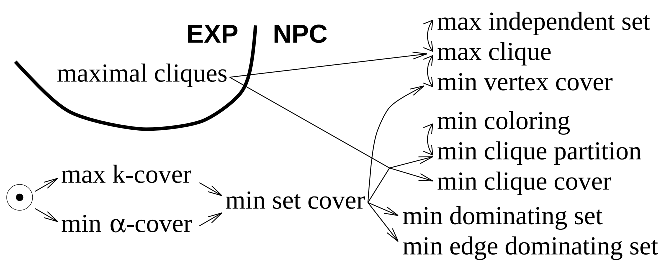

Figure 1: Algorithmic dependencies.

Let V' be a set of vertices. It is an *independent set* iff no two vertices of V' are linked by an edge. It is a *vertex covering* iff for each edge $\{u, v\}$ of E, at least one of u and v belongs to V' (i.e., all edges intersect V'). If C is a clique of G, then C is an independent set of the complementary graph $\overline{G}$, and $V - C$ is a vertex covering of $\overline{G}$. Thus maximum clique, maximum independent set, and minimum vertex covering are all equivalent. Minimum vertex covering[2] is the minimum set covering $\langle E, V, \ni\rangle$.

A set of vertices V' is a *dominating set* iff for each u of $V - V'$, it exists a v of V' such that $\{u, v\} \in E$ (i.e., all vertices are connected to V'). Let us associate the set $y(v) = \{v\} \cup \{u \mid \{u, v\} \in E\}$ with each vertex v. Then V' is the set of vertices associated with the solution of the minimum set covering $\langle V, \{y(v) \mid v \in V\}, \in\rangle$.

A set of edge E' is an *edge dominating set* iff for each e of $E - E'$, it exists a e' of E' that is adjacent to e (i.e., all edges are adjacent to E'). Let us associate the set $y(e) = \{e' \mid e \cap e' \neq \emptyset\}$ with each edge e of E. Then E' is the set of edges associated with the solution of the minimum set covering $\langle E, \{y(e) \mid e \in E\}, \subseteq\rangle$.

A *clique partition* is a set of cliques forming a partition of V. A clique partition is a *coloring* of the complementary graph (i.e., assigning a color to each vertex such that two vertices connected by an edge have different colors). A *clique cover* is a set of cliques such that each edge of E belongs to some clique. Minimum clique partition is solved with $\langle V, \mathit{PropMaxClique}(G), \in\rangle$, and minimum clique cover is solved with $\langle E, \mathit{PropMaxClique}(G), \subseteq\rangle$.

[2] The similar problem that consists of covering the vertices with a minimum number of edges is polynomial, since it is a set covering problem where each covering set has the cardinality 2.

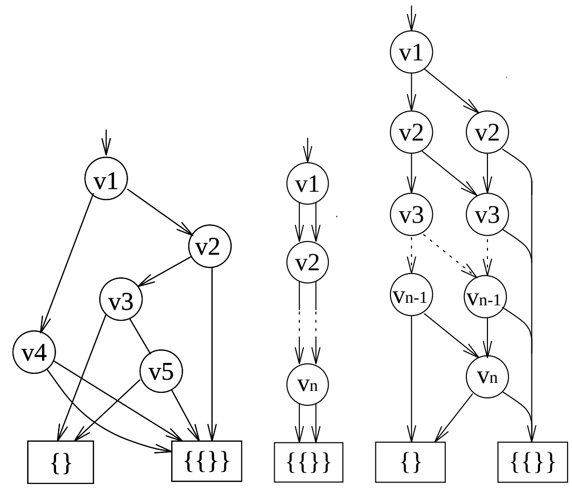

Figure 3: A ZBDD, the ZBDD of 2^V, and of $\mathit{AllEdge}(V)$.

3 Solving the Primitive Problems

This section shows how the primitive problems presented in Section 2.1 are expressed with set operations on 2^V. These set operations are implemented with ZBDDs (Zero-Suppressed Binary Decision Diagrams) [11].

Let $V = \{v_1, \ldots, v_n\}$. Given $X \subseteq 2^V$, we will denote X_{ε_k} the set of elements of X that do not contain v_k, and X_{v_k} the set made of the elements of X that contain v_k, from which v_k has been removed. For instance, with $X = \{\{\}, \{v_1, v_2\}, \{v_1, v_3, v_5\}, \{v_4\}\}$, we have $X_{\varepsilon_1} = \{\{\}, \{v_4\}\}$ and $X_{v_1} = \{\{v_2\}, \{v_3, v_5\}\}$. A ZBDD is a directed acyclic graph that encodes the recursive decomposition of a set X w.r.t. the elements of V in a given order. We will write $\texttt{vertex}(v_k, X_{\varepsilon_k}, X_{v_k})$ to denote the decomposition of X w.r.t. v_k. Fig. 3 shows on the left the ZBDD of the set X given above as example. Left (respectively right) branches denote X_{ε_k} (respectively X_{v_k}).

3.1 Maximal Cliques

Let $G = (V, E)$. The ZBDD of the set of edges $E \subseteq 2^V$ has a size bound by $2|E|$. Let *AllClique* be the set of all cliques, and *PropMaxClique* be the set of all maximal cliques. A clique is a subset of V that does not contains any couple of vertices that are not joined by an edge. We have:

$$\begin{aligned} PropMaxClique &= \max_{\subseteq} AllClique, \qquad \text{where} \\ AllClique &= NotSupSet(2^V, AllEdge(V) - E), \\ AllEdge(V) &= \{\{u,v\} \mid u \in V, v \in V, u \neq v\}, \\ NotSupSet(X,Y) &= \{x \in X \mid \forall y \in Y,\ x \not\supseteq y\}, \\ \max_{\subseteq} X &= \{x \in X \mid \forall x' \in X, x \subseteq x' \Rightarrow x = x'\} \end{aligned}$$

Fig. 3 shows the ZBDDs of 2^V (whose size is $|V|$), and of the set of all possible edges $AllEdge(V)$ (whose size is $2|V|$).

Both operations *NotSupSet* and $\max_{\subseteq}$ are implemented with ZBDDs, as shown by Fig. 4 and 5, using a divide-and-conquer strategy on the decompositions of the arguments w.r.t. their top labels v_k. The evaluation of $\max_{\subseteq}$ makes use of *NotSubSet*, which is defined as:

$$NotSubSet(X,Y) = \{x \in X \mid \forall y \in Y,\ x \not\subseteq y\}.$$

We actually use a more efficient algorithm, which combines both operations *NotSupSet* and $\max_{\subseteq}$ in a single pass. It produces directly the ZBDD of *PropMaxClique* from the ZBDD of E, without building the intermediate ZBDD representing all the cliques.

3.2 Max. k-cover and Min. α-covering

We consider the maximum k-cover and minimum α-covering[3] problems on the structure $\langle V, Y, \in\rangle$, where $Y \subseteq 2^V$. We show that both of these problems are solved with the operation $\odot$ defined on $2^V \times 2^V$ as:

$$X \odot Y \;=\; \max_{\subseteq}\{x \cup y \mid x \in X, y \in Y\}.$$

Fig. 6 shows how $\odot$ is evaluated on ZBDDs. This is a commutative and associative operation, thus we introduce the following notation (e.g., $Y^i \odot Y^j = Y^{i+j}$):

$$Y^k \;=\; \begin{cases} \{\{\}\} & \text{if } k = 0 \\ \underbrace{Y \odot Y \cdots \odot Y}_{k} & \text{if } k > 0 \end{cases}$$

The set Y^k is the set of all the maximal unions of k elements of Y. Let us denote $MaxCard(Y^k)$ all maximum cardinality elements of Y^k. The cost of extracting $MaxCard(Y^k)$ from Y^k is linear w.r.t the size of the ZBDD of Y^k. $MaxCard(Y^k)$ is the set of *all* maximum cardinality covers made of k sets picked in Y: this is exactly the set of all solutions of the maximum k-cover problem.

[3] Minimum 1-cover can be solved with the ZBDD based algorithm developed in [5].

```
function MaxSet(X);
if X = {} or X = {{}} return X;
let T1 = MaxSet(X_{v_k})
    T0 = MaxSet(X_{ε_k}) in
    return vertex(v_k, NotSubSet(T0, T1), T1);
```

Figure 5: Evaluating $\max_{\subseteq}$ on ZBDDs.

```
function MaxDot(X, Y);
if X = {} or Y = {} return {};
if X = {{}} return Y;
if Y = {{}} return X;
let T1 = MaxSet(MaxDot(X_{v_k}, Y_{ε_k} ∪ Y_{v_k}) ∪ MaxDot(X_{ε_k}, Y_{v_k}));
    T0 = MaxDot(X_{ε_k}, Y_{ε_k}) in
    return vertex(v_k, NotSubSet(T0, T1), T1);
```

Figure 6: Evaluating $\odot$ on ZBDDs.

Since $\odot$ is associative, we do not need to iterate $k-1$ evaluations of $\odot$ to compute Y^k and solve the maximum k-cover problem. Indeed, we just need to use a binary decomposition of k:

$$\begin{aligned} k &= \sum_{i=0}^{p} k_i 2^i, \qquad \text{with } k_i \in \{0,1\}, \text{ and} \\ p &= \lfloor \log_2 k \rfloor. \end{aligned}$$

Then Y^k is expressed as:

$$Y^k \;=\; Y^{k_0 2^0} \odot Y^{k_1 2^1} \odot Y^{k_2 2^2} \cdots \odot Y^{k_p 2^p},$$

which requires only $p - 1 + \sum_{i=0}^{p} k_i$ (i.e., $2p$ in the worst case) evaluations of $\odot$. For instance, the following sequence builds Y^{25} with 6 evaluations of $\odot$ ($25 = \overline{11001}^2$, and $\lfloor \log_2 25 \rfloor - 1 + (1+1+1) = 6$):

$$\begin{aligned} Y^2 &= Y \odot Y \\ Y^4 &= Y^2 \odot Y^2 \\ Y^8 &= Y^4 \odot Y^4 \\ Y^{16} &= Y^8 \odot Y^8 \\ Y^{24} &= Y^{16} \odot Y^8 \\ Y^{25} &= Y^{24} \odot Y \end{aligned}$$

One can also compute in a linear time w.r.t the size of the ZBDD of Y^k the greatest fraction $\alpha(Y^k)$ of the elements in V that are covered by k sets in Y. When

$$\alpha(Y^{k-1}) < \alpha \leq \alpha(Y^k),$$

k is the minimum number of sets in Y whose union covers at least a fraction α of V, thus solving the minimum α-covering problem. In the same way as above, we do not need to sequentially compute Y^2, Y^3, ..., Y^k, until at least a fraction α of V is covered by some set in Y^k. One can compute k by dichotomy, which requires at most $2p+1$ evaluations of $\odot$.

```
function NotSupSet(X, Y);
if X = {} or {} ∈ Y or X = Y return {};
if X = {{}} or Y = {} return X;
let T1 = NotSupSet(X_{v_k}, Y_{ε_k}) ∩ NotSupSet(X_{v_k}, Y_{v_k})
    T0 = NotSupSet(X_{ε_k}, Y_{ε_k}) in
    return vertex(v_k, T0, T1);
```

```
function NotSubSet(X, Y);
if Y = {} return X;
if Y = {{}} return X − {{}};
if X = {} or X = {{}} or X = Y return {};
let T1 = NotSubSet(X_{v_k}, Y_{v_k})
    T0 = NotSubSet(X_{ε_k}, Y_{ε_k}) ∩ NotSubSet(X_{ε_k}, Y_{v_k}) in
    return vertex(v_k, T0, T1);
```

Figure 4: Evaluating *NotSupSet* and *NotSubSet* on ZBDDs.

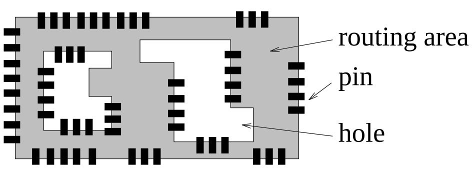

Figure 7: A routing region.

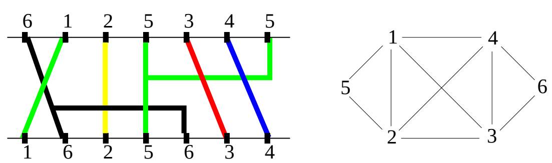

Figure 8: The compatibility graph of a routing channel.

4 Example: Multi-layer Planar Routing

A *routing region* (Fig. 7) is an area enclosed by an external boundary, with some blocks inside the boundary (called holes), and pins on the external and internal boundaries. A *net* is a set of pins to be connected without going through any boundary. A *planar subset* is a set of nets that can be routed in one layer without crossing each other. The maximum k-layer problem consists of routing as many nets as possible with k planar subsets (with the current technology, $k \leq 5$). The minimum α-cover problem consists of routing at least a fraction α of the nets with as few planar subsets as possible.

For submicron technologies, wire delays dominate gate delays, and the routing area exceeds the area used by transistors. Thus multi-layer routing technology is of increasing importance in performance-driven IC routing. Polynomial algorithms have been developed for special cases, but the general problem is NP–complete [3, 4].

Two nets are *compatible* iff they can be routed in a single layer without crossing. The *compatibility graph* (V, E) is made of the set of nets V as the set of vertices, and the set of compatible pairs E as a set of edges. When the routing area is a switchbox (no hole) or a channel (the pins are located on the top and bottom of the boundaries), sets of mutually compatible nets are exactly the planar subsets: multi-layer planar routing becomes a minimum clique partition problem. In Fig. 8, the maximal planar subsets (i.e., the maximal cliques of the compatibility graph) are $\{1, 2, 3, 4\}$, $\{1, 2, 5\}$ and $\{3, 4, 6\}$. The minimum layer routing is $\{\{1, 2, 5\}, \{3, 4, 6\}\}$.

Table 1 shows the results obtained to solve the minimum 1-cover problem for topological planar routing. These examples are benchmarks reported in [4]. Note that the size of the ZBDD (i.e., its number of vertices) remains small, even when the number of maximal planar subsets is large. The cyclic core is implicitly computed using the ZBDD based method presented in [5]. The size of the set covering problems are reduced by a factor > 20. The cyclic core is then explicitly solved using the minimizer Scherzo [6].

Table 2 gives the results obtained to solve the maximum k-cover problem for topological planar routing, i.e., finding a maximum number of nets that can be routed with no more than k layers. It is the first time one can obtain the exact solution of these problems.

A notable interest of the ZBDD based framework in this application is that one can handle practical design constraints thanks to filtering operators on the set of all possible cliques. For instance, the physical capacity constraint limits the number of nets that can be routed in a single layer. This consists of keeping only the cliques whose cardinalities do not exceed this limit.

5 Discussion & Conclusion

This paper has presented a ZBDD based framework that solves a collection of graph optimization problems, e.g., clique partitioning. It can be used to solve several CAD problems, e.g., logic minimization, constrained encoding, or multi-layer topological routing.

This framework can tackle large scale optimization problems that cannot be solved explicitly. Its use of ZBDD makes it able to work with graph and set problems that are too large to be explicitly built (e.g., graphs with more than 10^{15} cliques, set covering involving more than 10^{40} sets).

example			max planar subset		CC			
Name	**#net**	**#uncomp**	**|ZBDD|**	**#plan**	**row**	**col**	**sol**	**CPU**
burs	24	133	62	45	6	8	9	0.13
*ex*1	21	77	71	50	9	14	7	0.11
ex3a	44	176	149	495	16	41	10	0.44
ex3b	47	283	139	7685	31	804	9	1.85
ex3c	54	336	224	3329	32	621	12	1.46
ex4b	54	298	524	6885	31	535	11	2.05
*ex*5	64	405	282	61269	47	6749	9	135.36
ex5b	64	427	5988	44938	46	4773	10	38.05
deut	72	763	196	101427	50	5777	16	18.49
*exam*1	200	17124	734	6711	107	1512	126	51.02
*exam*2	250	26081	1061	19409	146	1897	141	159.53
*exam*3	300	36801	2294	100520	228	13316	162	597.53

For each **example**, we give its number of nets (**#net**), and its number of uncompatible pairs (**#uncomp**). |**ZBDD**| is the size (number of vertices) of the ZBDD representing the maximal planar subsets (their number is **#plan**). The cyclic core **CC** has size **row** × **col** (note that the size of the explicit initial covering problem is **#net** × **#plan**). **sol** is the exact solution for the minimum 1-cover problem, and the **CPU** time is given in seconds on a 60 MHz SuperSparc (85.4 SpecInt).

Table 1: Minimum 1-cover problem.

	k = 1		**k = 2**		**k = 3**		**k = 4**		**k = 5**	
burs	10	41%	14	58%	17	70%	19	79%	20	83%
*ex*1	8	38%	12	57%	15	71%	17	80%	19	90%
ex3a	21	47%	27	61%	31	70%	34	77%	36	81%
ex3b	15	31%	26	55%	33	70%	37	78%	40	85%
ex3c	20	37%	31	57%	36	66%	40	74%	43	79%
ex4b	22	40%	31	57%	38	70%	43	79%	46	85%
*ex*5	20	31%	32	50%	42	65%	49	76%	55	85%
ex5b	20	31%	32	50%	41	64%	48	75%	54	84%
deut	17	23%	29	40%	39	54%	46	63%	50	69%

This table gives the maximum number, and the corresponding percentage, of nets that can be routed with k layers, $1 \leq k \leq 5$.

Table 2: Maximum k-cover problem.

References

[1] R. K. Brayton, G. D. Hachtel, C. T. McMullen, A. L. Sangiovanni-Vincentelli, *Logic Minimization Algorithms for VLSI Synthesis*, Kluwer Academic Publishers, Dordrecht, 1984.

[2] N.-C. Chou, L.-T. Liu, C.-K. Cheng, W.-J. Dai, R. Lindelof, "Local Ratio Cut and Set Covering Partitioning for Huge Logic Emulation Systems", in *IEEE Trans. on CAD*, **14**-9, pp. 1085–1092, Sep. 1995.

[3] J. Cong, C. L. Liu, "On the k-layer Planar Subset and Topological Via Minimization Problems", in *IEEE Trans. on CAD*, **10**-8, pp. 972–981, Aug. 1991.

[4] J. Cong, M. Hossain, N. A. Sherwani, "A Provably Good Multilayer Topological Planar Routing Algorithm in IC Layout Designs," *IEEE Trans. on CAD*, **12**-1, pp. 70–78, Jan. 1993.

[5] O. Coudert, "Two-Level Logic Minimization: An Overview", in *Integration*, **17**-2, pp. 97–140, Oct. 1994.

[6] O. Coudert, "On Solving Covering Problems", in Proc. of *33rd DAC*, pp. 197–202, Las Vegas NV, June 1996.

[7] D. Gajski, N. Dutt, A. Wu, S. Lin, *High-Level Synthesis*, Kluwer Ac. Pub., 1992.

[8] M. R. Garey, D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP–Completeness*, San Francisco, CA, Freeman, 1979.

[9] T. Kam, T. Villa, R. Brayton, A. L. Sangiovanni-Vincentelli, "A Fully Implicit Algorithm for Exact State Minimization", in Proc. of *31st DAC*, pp. 684–690, San Diego CA, June 1994.

[10] G. D. Micheli, *Synthesis and Optimization of Digital Circuits*, McGraw-Hill, 1994.

[11] S. Minato, "Zero-Suppressed BDDs for Set Manipulation in Combinatorial Problems", in Proc. of *30th DAC*, Dallas TX, pp. 272–277, June 1993.

[12] S. Minato, "Fast Factorization Method for Implicit Cube Set Representation", IEEE Trans. on CAD, bf 15-4, pp. 377–384, April 1996.

[13] J.W. Moon and L. Moser, "On Cliques in Graphs", *Israel Journal of Mathematics*, **3**, pp. 23–28, 1965.

[14] R. L. Rudell, A. L. Sangiovanni-Vincentelli, "Multiple-Valued Minimization for PLA Optimization", *IEEE Trans. on CAD*, **6**-5, pp. 727–750, Sept. 1987

[15] A. Saldanha, T. Villa, R. K. Brayton, A. L. Sangiovanni-Vincentelli, "Satisfaction of Input and Output Encoding Constraints", in *IEEE Trans. on CAD*, **13**-5, pp. 589–602, May 1994.

[16] P. Sawkar and D. Thomas, "Multi-way Partitioning for Minimum Delay for Look-up Table based FPGAs", in Proc. of *32rd DAC*, San Francisco CA, pp. 201-205, June 1995.

[17] J. H. Tracey, "Internal State Assignment for Asynchronous Sequential Machines", in *IEEE Trans. on El. Comp.*, pp. 551–560, Aug. 1966.

[18] S. Yang, M. J. Ciesielski, "Optimum and Suboptimum Algorithms for Input Encoding and its Relationship to Logic Minimization", in *IEEE Trans. on CAD*, **10**-1, pp. 4–12, Jan. 1991.