Automata and Binary Decision Diagrams

Jean-Francis Michon and Jean-Marc Champarnaud

LIFAR
Université de Rouen, Faculté des Sciences et des Techniques
76821 Mont-Saint-Aignan Cedex, France
{michon, champarnaud}@dir.univ-rouen.fr

Abstract. We show that the concept of automata minimization leads to a nice interpretation of the famous canonicity of binary decision diagrams discovered by Bryant.

1 Introduction

The aim of this paper is to enlighten the links between automaton minimization and the construction of binary decision diagrams of Boolean functions. We give some direct applications to complexity. Section 2 recalls basic definitions regarding the theories of Boolean functions and automata. Section 3 deals with complexity results. Section 4 asks some questions.

2 Definitions and Conventions

2.1 Boolean Functions

In this field it is convenient to introduce mathematical structure "just in time" so we choose this poor definition:

Definition 1 *A Boolean function on an arbitrary set A is a map from A to the set of integers $B = \{0, 1\}$.*

So a Boolean function can be identified with its graph which is a part of $A \times \{0, 1\}$. This graph is a set of pairs, called the truth table of the Boolean function. In computer science we don't deal with arbitrary sets A but with sets of n-tuples of bits. The set A is $B^n = \{0, 1\}^n$ for some integer $n > 0$. We shall assume that we are dealing with those sets only. We would like to stress that this is not a light assumption: it means that the set A is very special set with plenty of magic properties.

This implies that a Boolean function is now a function of n variables each ranging over B and with values in B:

$$(x_1, ..., x_n) \in B^n \mapsto f(x_1, ..., x_n) \in B$$

We have a canonical order on B induced by that of $N : 0 < 1$. This order on B induces lexicographically a total order on A. This permits one to present the

J.-M. Champarnaud, D. Maurel, D. Ziadi (Eds.): WIA'98, LNCS 1660, pp. 178–182, 1999.

truth table of any Boolean fonction f as the well known ordered table with 2^n rows and 2 columns. The classical other way to present the truth table is that of a binary ordered tree. Its construction is as follows:
a) The root vertex is labeled with x_1.
b) The two sons of any vertex labeled x_i are labeled x_{i+1} if $i < n$.
c) The two sons of the vertices labeled x_n are leaves.
d) The left (resp. right) edge starting from an interior vertex is labeled with 0 (resp. 1).
e) Each of the 2^n leaves is labeled with the value of f on the n-tuple corresponding to the unique path from the root x_1 to this leaf.

We call this tree the canonical tree associated with f and denote it $T(f)$.

2.2 Binary Decision Diagrams

A binary decision diagram is a member of a family of graphs containing binary trees like $T(f)$ and all graphs inductively reduced by the following two rules:
(Reduction 1) Identify two isomorphic subtrees G and H by deleting one and redirecting incoming edges to the root of the other.
(Reduction 2) If a vertex v has exactly one son, suppress v and the two edges starting from it and connect its incoming edges The graphs obtained at each step are always directed, acyclic, single originated; their branching index is at most two and they have one or two terminal vertices respectively labeled 0 and 1. This process is confluent and terminates. Applied to $T(f)$ the result is called the BDD or, more precisely, the ROBDD (reduced ordered BDD) of f. The complexity of a BDD is defined as the number of its vertices. Sometimes the complexity is very low and sometimes exponential in the number of variables. When the complexity of a BDD is low computers can work with and manipulate the Boolean function it refers to. This subject is of great interest in verification and also in reliability theory. We refer the reader to [BR, BR2] for excellent accounts and detailed explanations.

Definition 2 *The quasi-reduced BDD of a Boolean function is the graph obtained by iterating reduction steps of type 1 until this reduction cannot be applied any longer.*

We shall speak of the QRBDD of f.

2.3 Automata

Let X be a finite alphabet. An *automaton* over X is a 4-tuple $\mathcal{A} = (Q, I, F, E)$ where Q is a set of *states*, I is a subset of Q whose elements are the *initial states*, F is a subset of Q whose elements are the *final states*, (or *terminal states*), E is a subset of the cartesian product $Q \times X \times Q$ whose elements are the *edges*. Let $\mathcal{A} = (Q, I, F, E)$ be an automaton. A *path* of $\mathcal{A}$ is a sequence (q_i, a_i, q_{i+1}), $i = 1, \ldots, n$, for some $n \geq 1$, of consecutive edges. Its *label* is the

word $w = a_1 a_2 \dots a_n$. A word $w = a_1 a_2 \dots a_n$ is *recognized* by the automaton $\mathcal{A}$ if there is a path with label w such that $q_1 \in I$ and $q_{n+1} \in F$. The language *recognized* by the automaton $\mathcal{A}$ is the set of words which it recognizes.

We now add some extra structure to $T(f)$ to equip it with the structure of an automaton.

Definition 3 *The truth automaton* $\mathcal{T}(f) = (Q, I, F, E)$ *of* f *over the alphabet* $\{0, 1\}$ *deduces from* $T(f)$ *as follows:*
1) Q *is made of all the vertices of* $T(f)$*.*
2) The unique initial state is the root of $T(f)$*.*
3) F *is the set of all the leaves of* $T(f)$ *labeled* 1*.*
4) The edges in E *are the edges of* $T(f)$*, with the same labels.*

The truth automaton of f can be viewed as the canonical disjunctive normal form of f. This automaton exactly recognizes the n-tuples of B^n where f takes the value 1 (called the support of f). In other terms the support of f is the (regular) language recognized by the automaton $\mathcal{T}(f)$. This automaton is obviously deterministic and accessible, but it is not complete. The completion of $\mathcal{T}(f)$ can be canonically performed by adding a non-final extra state (a sink) and connecting every leaf (and this sink) to the sink by each of the letters 0 and 1. We then loose the tree structure of the underlying graph. We are now in the position to minimize this automaton; we shall denote the minimal automaton by $\mathcal{T}'(f)$.

3 Minimal Automata and BDDs

Theorem 1. *The quasi reduced BDD of* f *is isomorphic to* $\mathcal{T}'(f)$*.*

Proof. This is a consequence of the Nerode equivalence [NE], which can be stated as follows: two vertices are said to be Nerode equivalent if the sub-automata rooted in these two vertices recognize the same language.
Suppose two vertices of $\mathcal{T}(f)$ are Nerode equivalent. This is exactly equivalent with the fact that the subtrees rooted at each of these vertices are isomorphic.

The reduction step of type 2 of ROBDDs is proper to Boolean algebra: it means that if the substitution of x_i by 0 or 1 does not affect the value of a Boolean function then the function is independent of x_i, the variable is "dumb". This is not exactly the same for non Boolean functions. So it's not surprising that the reduction of type 2 cannot be captured by the theory of minimization. The truth automaton $\mathcal{T}(f)$ of f being acyclic, the minimal automaton $\mathcal{T}'(f)$ can be computed in time linear in terms of the number of states of $\mathcal{T}(f)$ [RE]. So the complexity of computing the QRBDD of f starting from $\mathcal{T}(f)$ is $O(2^n)$. Of course, this is quite unrealistic because a Boolean function is never given by its truth table when the number of variables is large.

We now consider the complexity of ROBDDs and QRBDDs. Akers [AK] says that the worst case ROBDDs complexity is $O(2^n/n)$. The result is made more precise in [LL] where this worst case complexity is proved to be less than $(2^n/n)(2+\varepsilon)$ as n grows and ε is any arbitrarily small strictly positive real number. Some further results are given in [W].

An interesting result, coming from automaton theory, is given by Champarnaud and Pin [CP]. Translated into our context, it says that the maximal number of states, denoted by $g(n)$, of any QRBDD is exactly

$$g(n) = \sum_{1 \le i \le n} \min(2^i, 2^{2^{n-i}} - 1)$$

and

$$\liminf_{n \to \infty} ng(n)/2^n = 1,$$

$$\limsup_{n \to \infty} ng(n)/2^n = 2.$$

For a given n, an explicit construction of all the QRBDDs with a maximal number of states is also suggested. The construction implies the result of [LL] for ROBDDs because reduction steps of type 2 can occur only in the tail part of their automaton (where the number of states decays doubly exponentially). The reduction of type 2 is, consequently, negligible.

A side consequence is that we can explicitly give a family of very complex Boolean functions from the point of view of BDDs. Boolean functions under permutations of variables.

4 Conclusion

The conclusion is that automata are one essential part of BDD theory and that this part is quite independent of Boolean algebra. It follows that the same process (minimization) can be applied to more general BDD-like structures for integer valued functions, arithmetic functions ... Conversely the BDD theory suggests good ideas to automata specialists. For example, we don't touch here the crucial problem of BDDs which is the sensitivity of its complexity to the ordering of variables [FS]. This must be reflected in automata theory by the notion of commutative equivalence of languages. In the same type of idea the process of identifying isomorphic subtrees can be viewed as a Fourier-Hadamard transformation (called the spectrum of a Boolean function). This suggests there may exist a sort of "Fourier transform" under the minimization process for automata.

References

[AK] S. B. Akers, Binary Decision Diagrams, *IEEE Transactions on Computers* **C-27**(6) (1978) 509–516.

[BR] R. E. Bryant, Graph Based Algorithms for Boolean Functions Manipulation, *IEEE Transactions on Computers* **C-35**(8) (1986) 677–691.

[BR2] R. E. Bryant, Binary Decision Diagrams and beyond, *Survey Report*, Carnegie Mellon University.

[CP] J.-M. Champarnaud and J.-E. Pin, A Maxmin Problem on Finite Automata, *Discrete Applied Mathematics* **23** (1989) 91–96.

[FS] S. J. Friedman, K. J. Supowit, Finding the Optimal Ordering for Binary Decision Diagram, *IEEE Transactions on Computers* **39**(5) (1990).

[LL] H.-T. Liaw and C.-S. Lin, On the OBDD Representation of General Boolean Functions, *IEEE Transactions on Computers* **C-41**(46) (1992) 661–664.

[NE] A. Nerode, Linear Automata Transformation, Proc. *AMS 9* (1958) 541–544.

[RE] D. Revuz, Minimization of Acyclic Deterministic Automata in Linear Time, *Theoret. Comput. Sci.* **92** (1992) 181–189.

[W] I. Wegener, Efficient Data Structure for Boolean Functions, *Discrete Mathematics* **136** (1994) 347–372.