The Algebraic Mu-Calculus and MTBDDs

Christel Baier
Universität Mannheim, Germany
baier@pi2.informatik.uni-mannheim.de

Edmund M. Clarke
CMU, Pittsburgh, , USA
emc@cs.cmu.edu

May 22, 1998

Abstract

The paper presents a new calculus (called *algebraic mu-calculus*) which generalizes Park's relational mu-calculus by representing arithmetic expressions and real-valued functions rather than formulas and relations. Moreover, we give an algorithm for computing the MTBDD-representation of the semantics for the expressions and terms and show how several problems (such as graph theoretic problems or verification problems) can be embedded into the algebraic mu-calculus (and hence, can be solved using our MTBDD-based method).

1 Introduction

In several disciplines of mathematics and theoretical computer science, fixed point problems play a crucial role. For example, reachability problems in graph theory, solving linear or non-linear equation systems for real or complex numbers, the computation of eigenvalues of matrices, the definition of denotational semantics for recursive programs or various verification problems for parallel or randomized systems can be reduced to certain fixed point problems. In most cases, the solution can be computed by iteration (on the basis of Tarski's or Banach's fixed point).

Many set-based fixed point problems (e.g. graph theoretical problems or verification problems for parallel systems like verification against temporal logic specifications) can be embedded into Park's relational mu-calculus [Par74], and hence, can be solved with the BDD-based method of [BCM+92, CGL93]. The algebraic mu-calculus is a generalization of the relational mu-calculus which replaces formulas of the relational mu-calculus by expressions (that are interpreted by real numbers), relational terms by algebraic terms (that are interpreted by real-valued functions) and the least and greatest fixed point operators by an operator which "computes" the limit of a certain sequence of functions. Under certain conditions, this limit operator specializes to a least or greatest fixed point operator. The algebraic mu-calculus is a very powerful language. For instance, it subsumes the relational mu-calculus (and hence, all logics that can be embedded into the relational mu-calculus, cf. [BCM+92]), the modal mu-calculus with the non-standard interpretations of [Sei96, HK97] and the probabilistic temporal logics of [HJ94, BA95].

We describe an algorithm that computes the representation of the semantics for the expressions and terms by multiterminal binary decision diagrams (MTBDDs) [CFM+93]. This method yields a "language" for MTBDDs which has various applications. For instance, our framework can be applied for solving graph theoretical problems such as computing cheapest paths, for implementing matrix operations such as solving linear equation systems or computing eigenvalues and for the verification of probabilistic programs.

The paper is organized as follows. The syntax and semantics of the algebraic mu-calculus is contained in Section 2. Section 3 presents our MTBDD-based method for computing the semantics of the expressions and terms. In Section 4 we briefly explain some possible applications. Section 5 concludes the paper.

2 The algebraic mu-calculus

The syntax of the algebraic mu-calculus arises from the relational mu-calculus [Par74] by using arbitary arithmetic operators (e.g. summation $+$ or multiplication $*$) instead of the boolean connectives $\vee$ and $\wedge$, replacing the quantifiers $\exists z$ and $\forall z$ by arithmetic ones $\sum_z$, $\min_z$ and $\max_z$ and the least/greatest fixed point operators by a limit operator.

Syntax: We fix finite sets of *individual variables* and *term variables*. The term variables are associated with an arity (a natural number ≥ 1). Let Op be a set of binary arithmetic operators on the reals such as summation $+$, multiplication $*$, the binary minimum and maximum operators op_{min}, op_{max} (where e.g. $q_1\ op_{min}\ q_2 = \min\{q_1, q_2\}$) and the "comparison operators" $op_{\bowtie}$ where $\bowtie \in \{\leq, <, \geq, >, =, \neq\}$ and $q_1\ op_{\bowtie}\ q_2 = 1$ if $q_1 \bowtie q_2$ and $q_1\ op_{\bowtie}\ q_2 = 0$ otherwise. *Algebraic expressions* and *algebraic terms* of arity n are built from the following production systems:

$$\begin{array}{lcl} expr & ::= & q \mid expr_1\ op\ expr_2 \mid -\ expr \mid term(z_1, \ldots, z_n) \mid \sum_z[expr] \mid \min_z[expr] \mid \max_z[expr] \\ term & ::= & Z \mid \lambda z_1, \ldots, z_n[expr] \mid limit\ Z\ [\ term \uparrow term_0\ ] \end{array}$$

where q is a real number, $op \in Op$, z, $z_1, \ldots, z_n$ are individual variables such that $z_1, \ldots, z_n$ are pairwise distinct and Z is an n-ary term variable. For the terms *limit* Z [$term \uparrow term_0$], we require that Z is a term variable and $term$ and $term_0$ are algebraic terms such that Z, $term$ and $term_0$ have the same arity. Intuitively, *limit* $Z[term \uparrow term_0]$ stands for the "limit" of the sequence $(term_n)_{n\geq 0}$ where $term_{n+1} = term\{Z \leftarrow term_n\}$ and where the brackets $\{\ldots\}$ denote syntactic replacement. As usual we define the boundedness and freeness of occurrences of variables in algebraic expressions or algebraic terms. The individual variables can be bounded by the operators $\sum$, min, max and λ-abstraction while the term variables can be bounded by the limit operator. An algebraic term is called *closed* if it does not contain free occurrences of individual or term variables.

The relational mu-calculus à la Park [Par74] can be viewed as a sublanguage of the algebraic mu-calculus. Formally, formulas and terms of the relational mu-calculus are special instances of *boolean expressions* and *boolean terms* that are built from the following production systems.

$$\begin{array}{lcl} bexpr & ::= & 0 \mid 1 \mid bexpr_1 \ \wedge \ bexpr_2 \mid \neg bexpr \mid expr_1 \ op_{\bowtie} \ expr_2 \mid bterm(z_1, \ldots, z_n) \mid \ \forall z \ [bexpr] \\ bterm & ::= & Z \ \mid \ \lambda z_1, \ldots, z_n[bexpr] \ \mid \ lfp Z[bterm] \ \mid \ gfp Z[bterm] \end{array}$$

where Z is an n-ary term variable, $expr_1$, $expr_2$ are (arbitrary) algebraic expressions, z, $z_1, \ldots, z_n$ are individual variables such that $z_1, \ldots, z_n$ are pairwise distinct. Here, $\wedge = op_{min}$, $\neg bexpr = 1 - bexpr$ and $\forall z[\ldots] = \min_z[\ldots]$. The *least* and *greatest fixed point operators* $lfpZ[\ldots]$ and $gfpZ[\ldots]$ are given by: $lfpZ\,[term] = limit\ Z\ [\ term \uparrow \lambda z_1, \ldots, z_n[0]\ ]$ and $gfpZ\,[term] = limit\ Z\ [\ term \uparrow \lambda z_1, \ldots, z_n[1]\ ]$. As usual, disjunction $\vee$, equivalence $\leftrightarrow$ and the existential quatifier $\exists$ can be derived from $\wedge$, $\neg$ and $\forall$. In the boolean subcalculus, we write $expr_1 \bowtie expr_2$ rather than $expr_1\, op_{\bowtie}\, expr_2$.

Semantics: The domain of the *extended reals* denotes the set $\overline{I\!R} = I\!R \cup \{\bot\}$ where $I\!R$ is the set of real numbers and $\bot$ is a special symbol which can be interpreted as "undefined" or "divergence". Expressions are interpreted by an extended real number, n-ary terms by n-ary functions into the extended reals. A *model* for the algebraic mu-calculus is a pair $\mathcal{M} = (D, I)$ consisting of a nonempty finite set D (called the *domain*) and an interpretation I for the individual and term variables, i.e. a function I which assigns to each individual variable z an element $I(z) \in D$ and to each n-ary term variable Z a function $I(Z) : D^n \to \overline{I\!R}$.

Let $\mathcal{M} = (D, I)$ be a model. For each algebraic expression $expr$ and algebraic term $term$ we define $[\![expr]\!]^{\mathcal{M}} \in \overline{I\!R}$ and $[\![term]\!]^{\mathcal{M}} : D^n \to \overline{I\!R}$ as follows. We assume that all operators $op \in Op$ are extended to operators on $\overline{I\!R}$. We define an operator $Limit : \overline{I\!R}^{\omega} \to \overline{I\!R}$ as follows. Here, $\overline{I\!R}^{\omega}$ denotes the set of infinite sequences in $\overline{I\!R}$. Let $q_0, q_1, q_2, \ldots$ be an infinite sequence in $\overline{I\!R}$. If $q_n \in I\!R$ for almost all n, e.g. $q_n \in I\!R \cup \{+\infty, -\infty\}$ for all $n \geq n_0$, then $Limit(q_0, q_1, q_2, \ldots) = \bot$ iff $(q_n)_{n\geq n_0}$ does not converge in $I\!R$, and, if $(q_n)_{n\geq n_0}$ converges in $I\!R$ then $Limit(q_0, q_1, q_2, \ldots) = \lim q_n$, the (usual) limit of $(q_n)_{n\geq n_0}$ in $I\!R$. If $q_n = \bot$ for infinitely many n then we define $Limit(q_0, q_1, q_2, \ldots) = \bot$. If $(F_j)_{j\geq 0}$ is a sequence of functions $F_j : D^n \to \overline{I\!R}$ then $Limit(F_0, F_1, F_2, \ldots)$ denotes the function $D^n \to \overline{I\!R}$, $\overline{d} \mapsto Limit\left(F_0(\overline{d}), F_1(\overline{d}), F_2(\overline{d}), \ldots\right)$.

$$\begin{array}{l} [\![q]\!]^{\mathcal{M}} = q, \quad [\![\ expr_1\ op\ expr_2\ ]\!]^{\mathcal{M}} = [\![expr_1]\!]^{\mathcal{M}}\ op\ [\![expr_2]\!]^{\mathcal{M}}, \quad [\![\ -expr\ ]\!]^{\mathcal{M}} = -[\![expr]\!]^{\mathcal{M}}, \\ [\![term(z_1, \ldots, z_n)]\!]^{\mathcal{M}} = [\![term]\!]^{\mathcal{M}}(I(z_1), \ldots, I(z_n)), \quad [\![\sum_z [expr]]\!]^{\mathcal{M}} = \sum_{d\in D} [\![expr]\!]^{\mathcal{M}[z:=d]}, \\ [\![\min_z[expr]]\!]^{\mathcal{M}} = \min\left\{[\![expr]\!]^{\mathcal{M}[z:=d]} : d \in D\right\}, \quad [\![\max_z[expr]]\!]^{\mathcal{M}} = \max\left\{[\![expr]\!]^{\mathcal{M}[z:=d]} : d \in D\right\}, \\ [\![Z]\!]^{\mathcal{M}} = I(Z), \quad [\![\ \lambda z_1, \ldots, z_n[expr]\ ]\!]^{\mathcal{M}}(d_1, \ldots, d_n) = [\![expr]\!]^{\mathcal{M}[z_1:=d_1, \ldots, z_n:=d_n]}, \\ [\![\ limit\ Z\ [\ term \uparrow term_0\ ]\ ]\!]^{\mathcal{M}} = Limit(F_0, F_1, F_2, \ldots) \text{ where } F_0 = [\![term_0]\!]^{\mathcal{M}},\ F_{n+1} = [\![term]\!]^{\mathcal{M}[Z:=F_n]}. \end{array}$$

Here, if n, $m \geq 0$ and $z_1, \ldots, z_n$ are pairwise distinct individual variables and $Z_1, \ldots, Z_m$ are pairwise distinct term variables where the arity of Z_j is k_j and $d_i \in D$, $F_j : D^{k_j} \to \overline{I\!R}$ then $\mathcal{M}[z_1 := d_1, \ldots, z_n := d_n, Z_1 := F_1, \ldots, Z_m := F_m]$ denotes the model (D, J) where $J(z_i) = d_i$, $J(Z_j) = F_j$ and $J(z) = I(z)$, $J(Z) = I(Z)$ in all other cases.

We present conditions about the arguments $term$ and $term_0$ of the limit operator that turn $limit Z[\ldots]$ into a least or greatest fixed point operator (Theorem 1). Let $\Re$ be a nonempty subset of $I\!R$ that is closed under monotonic limits (i.e. whenever $q_0 < q_1 < \ldots$ is a sequence in $\Re$ then $Limit(q_0, q_1, \ldots) \in \Re$). Moreover, we assume that $a = \min \Re$ and $b = \max \Re$ exist. For instance, $\Re = \{a, b\}$ or $\Re$ is the closed interval $[a, b]$. A model $\mathcal{M} = (D, I)$ is called a $\Re$-model iff, for each term variable Z, the range of $I(Z)$ is contained in $\Re$. $term$ is called $\Re$-*closed* iff, for each $\Re$-model $\mathcal{M}$, the range of $[\![term]\!]^{\mathcal{M}}$ is a subset of $\Re$. A term variable Z is called *formally continuous* in $term$ iff, for each subexpression $expr_1\ op\ expr_2$ of $term$ that contains a free occurrence of Z, the operator op preserves infima and suprema (e.g. $op \in \{+, *, op_{min}, op_{max}\}$). Z is called *formally monotone* in $term$ iff all free occurrences of Z in $term$ fall under an even number of negations. $term$ is called $\Re$-*divergence-free* iff, for each subterm *limit* $Z\ [term' \uparrow term_0]$, Z is formally continuous and formally monotone in $term'$ and $term'$ and $term'_0$ are $\Re$-closed.

Theorem 1 *Let term and $term_0$ be n-ary terms that are $\Re$-closed and $\Re$-divergence-free and Z an n-ary term variable that is formally continuous and formally monotone in term. Then, $limit Z[term \uparrow term_0]$ is $\Re$-closed (and $\Re$-divergence-free). Moreover, if $\mathcal{M} = (D, I)$ a $\Re$-model and $\Omega : (D^n \to [a, b]) \to (D^n \to [a, b])$, $\Omega(F) = [\![term]\!]^{\mathcal{M}[Z:=F]}$, then $[\![limit\ Z[term \uparrow \lambda s[a]]]\!]^{\mathcal{M}}$ is the least fixed point of Ω and $[\![limit\ Z[term \uparrow \lambda s[b]]]\!]^{\mathcal{M}}$ the greatest fixed point of Ω.*

Using Theorem 1, it is easy to see that the semantics for expressions or terms of the relational mu-calculus agrees with the standard semantics à la Park. In the full paper we also show that the positive modal mu-calculus with the non-standard interpretations of [Sei96, HK97] can be embedded into the algebraic mu-calculus. In these subcalculi, the expressions can be viewed as properties for *durational transition systems* where the semantics is a quantitative measure of how long a property holds [Sei96] or for *probabilistic transition systems* where the semantics is a real number between 0 and 1 that can be viewed as the probability with which the property is satisfied.

3 An algorithm for computing the semantics

We describe an algorithm for computing the semantics $[\![\ldots]\!]^{\mathcal{M}}$ which uses multiterminal binary decision diagrams (MTBDDs) as introduced by Clarke et al [CFM+93]. MTBDDs are an extension of Bryant's ordered BDDs [Bry86] that can be viewed as a data structure for real-valued functions and that we use to represent the meaning of algebraic expressions and terms. If the expressions are viewed as properties (or formulas) whose validity is measured by an extended real (rather than the usual truth values 0 or 1) then our method can be viewed as a *model checker* that takes a model $\mathcal{M}$ and an expression (or term) as its input and returns the "truth value" $[\![\ldots]\!]^{\mathcal{M}}$ for the corresponding property. Our algorithm works similar to the one for the relational mu-calculus presented in [BCM+92] that computes the BDD-representations for the formulas and relational terms. In this paper, we only consider the case where the domain D is $\{0,1\}$. The general case where D is an arbitrary finite set can be handled as in [BCM+92] which is based on an encoding of the elements of D by bit vectors. Of course, the correctness of our method is up to the errors that arise from the approximations for the limit operator.

In the same way as the relational mu-calculus (together with the method of [BCM+92]) can be viewed as a language for BDDs the algebraic mu-calculus yields a language for MTBDDs. For this, we introduce a *mixed* calculus that extends the algebraic mu-calculus by considering MTBDDs as term variables with a fixed interpretation. Any closed mixed term can be retransformed into an algebraic term for which our method computes the corresponding MTBDD, and hence, can be considered as a procedure for the computation of a MTBDD.

MTBDDs: We briefly recall the definition of MTBDDs [CFM+93, CFZ96]. Let *Var* be a finite set of variables and $<$ a total order on *Var*. A MTBDD over $\langle Var, < \rangle$ is a rooted, directed graph with vertex set V containing two types of vertices, *nonterminal* and *terminal*. Each nonterminal vertex v is labelled by a variable $var(v) \in Var$ and two sons $left(v)$, $right(v) \in V$. Each terminal vertex v is labelled by an extended real number $value(v)$. For each nonterminal node v, we require that $var(v) < var(left(v))$ if $left(v)$ is nonterminal and $var(v) < var(right(v))$ if $right(v)$ is nonterminal. A BDD is a MTBDD where all terminal vertices are labelled by 0 or 1. If $Var = \{x_1, \ldots, x_n\}$ and $x_1 < x_2 < \ldots < x_n$ then we speak about MTBDDs over $(x_1, \ldots, x_n)$ rather than MTBDDs over $\langle Var, < \rangle$. Each MTBDD Q over $(x_1, \ldots, x_n)$ represents a function $F_{\mathrm{Q}} : \{0,1\}^n \to \overline{I\!R}$ as follows. Given a bit vector $\overline{b} = (b_1, \ldots, b_n)$, we traverse Q starting in the root. If we reach a nonterminal vertex v labelled by x_i then we go to $left(v)$ (resp. $right(v)$) if $b_i = 0$ (resp. $b_i = 1$). If we reach a terminal vertex v then we put $F_{\mathrm{Q}}(\overline{b}) = value(v)$. Vice versa, given a function $F : \{0,1\}^n \to \overline{I\!R}$ then the "Shannon tree" can be viewed as a MTBDD that represents F. Canonical ("minimized") MTBDD-representations can be obtained using Bryant's REDUCE operator [Bry86].

Our algorithm uses the following operators on MTBDDs which are taken from [Bry86, CFM+93]. Let Q, Q_1, Q_2 be MTBDDs over $(x_1, \ldots, x_n)$. If *op* a binary operator on the extended reals then $\textsc{Apply}(\mathrm{Q}_1, \mathrm{Q}_2, op)$ returns the unique reduced MTBDD Q over $(x_1, \ldots, x_n)$ where $F_{\mathrm{Q}} = F_{\mathrm{Q}_1}\ op\ F_{\mathrm{Q}_2}$. $\textsc{Neg}(\mathrm{Q})$ returns the MTBDD for the function $F = -F_{\mathrm{Q}}$. If $y_1, \ldots, y_n$ be pairwise distinct variables then $\mathrm{Q}\{x_1 \leftarrow y_1, \ldots, x_n \leftarrow y_n\}$ denotes those MTBDD over $(y_1, \ldots, y_n)$ that arises from Q by renaming simultaneously the variables x_i by y_i. If $i \in \{1, \ldots, n\}$ and $b \in \{0,1\}$ then $\mathrm{Q}|_{x_i = b}$ denotes the unique reduced MTBDD over $(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$ that represents the function $\{0,1\}^{n-1} \to \{0,1\}$, $(b_1, \ldots, b_i, b_{i+1}, \ldots, b_n) \mapsto F_{\mathrm{Q}}(b_1, \ldots, b_{i-1}, b, b_{i+1}, \ldots, b_n)$.

Computing the semantics for the algebraic mu-calculus using MTBDDs: Let *IndVar* denote the set of individual variables. We fix a total order $<$ on *IndVar* and define an algebraic expression and term (with individual variables belonging to *IndVar*) to be *well-formed* iff, for each subexpression of the form $term(z_1, \ldots, z_n)$, we have $z_1 < \ldots < z_n$. In what follows, we assume a model $\mathcal{M} = (D, I)$ where $D = \{0,1\}$ and, for each n-ary term variable Z, the function $I(Z)$ is represented by a MTBDD (also called $I(Z)$) over $(\vartheta_1, \ldots, \vartheta_n)$ where $\vartheta_1, \vartheta_2, \ldots, \vartheta_n$ are *dummy variables* ordered by $z < \vartheta_1 < \ldots < \vartheta_n$ for all $z \in IndVar$. For *expr* well-formed, we define $\textsc{Mtbdd}^I[\![expr]\!]$ to be a MTBDD over $\langle IndVar, < \rangle$. Each n-ary well-formed algebraic term *term* is associated with a MTBDD $\textsc{Mtbdd}^I[\![term]\!]$ over $\langle IndVar \cup \{\vartheta_1, \ldots, \vartheta_n\}, < \rangle$. We compute $\textsc{Mtbdd}^I[\![\ldots]\!]$ for well-formed algebraic expressions and terms by structural induction as shown in Figure 1. Here, we suppose an extension of $+$ and $*$ to operators on the extended reals where $\bot * q = q * \bot = \bot$ for all $q \in \overline{I\!R} \setminus \{0\}$, $0 * \bot = \bot * 0 = 0$ and $\bot + q = q + \bot = \bot$ for all $q \in \overline{I\!R}$ and $I[\ldots]$ is defined in the obvious way.

$\textsc{Mtbdd}^I[\![q]\!]$ denotes the MTBDD that consists of a terminal vertex labelled by q.

$\textsc{Mtbdd}^I[\![\ expr_1\ op\ expr_2\ ]\!] = \textsc{Apply}\left(\textsc{Mtbdd}^I[\![expr_1]\!], \textsc{Mtbdd}^I[\![expr_2]\!], op\right)$

$\textsc{Mtbdd}^I[\![\ -expr\ ]\!] = \textsc{Neg}\left(\textsc{Mtbdd}^I[\![expr]\!]\right)$

$\textsc{Mtbdd}^I[\![\ term(z_1,\ldots,z_n)\ ]\!] = \textsc{Mtbdd}^I[\![term]\!]\{\vartheta_1 \leftarrow z_1,\ldots,\vartheta_n \leftarrow z_n\}$

$\textsc{Mtbdd}^I[\![\ \sum_z\ [expr]\ ]\!] = \textsc{Apply}\left(\textsc{Mtbdd}^I[\![expr]\!]|_{z=0}, \textsc{Mtbdd}^I[\![expr]\!]|_{z=1}, +\right)$

$\textsc{Mtbdd}^I[\![\ \min_z\ [expr]\ ]\!] = \textsc{Apply}\left(\textsc{Mtbdd}^I[\![expr]\!]|_{z=0}, \textsc{Mtbdd}^I[\![expr]\!]|_{z=1}, op_{min}\right)$

$\textsc{Mtbdd}^I[\![\ \max_z\ [expr]\ ]\!] = \textsc{Apply}\left(\textsc{Mtbdd}^I[\![expr]\!]|_{z=0}, \textsc{Mtbdd}^I[\![expr]\!]|_{z=1}, op_{max}\right)$

$\textsc{Mtbdd}^I[\![Z]\!] = I(Z), \quad \textsc{Mtbdd}^I[\![\ \lambda z_1,\ldots,z_n[expr]\ ]\!] = \textsc{Mtbdd}^I[\![expr]\!]\{z_1 \leftarrow \vartheta_1,\ldots,z_n \leftarrow \vartheta_n\}$

$\textsc{Mtbdd}^I[\![\ limit\ Z[\ term \uparrow term_0\ ]\ ]\!] = \mathrm{Q}$

where Q is computed as follows. We fix some $\varepsilon > 0$ "sufficiently small" and some natural number n_{max} (the maximal number of iterations). Let m be the arity of Z_j and $\overline{z} = (z_1,\ldots,z_m)$ an m-tuple of individual variables with $z_1 < \ldots < z_m$. We choose some "new" m-ary term variables Z_{new}, Z_{old} and Y and compute Q as follows.

$n := 0;\ \mathrm{Q}_0 := \textsc{Mtbdd}^I[\![term_0]\!];$
Repeat
  $n := n+1;\ \mathrm{Q}_n := \textsc{Mtbdd}^{I[Z:=\mathrm{Q}_{n-1}]}[\![term]\!];$
  $\mathrm{B} := \textsc{Mtbdd}^{I[Z_{old}:=\mathrm{Q}_{n-1}, Z_{new}:=\mathrm{Q}_n]}[\![\ \lambda\overline{z}[\ (Z_{old}(\overline{z}) - Z_{new}(\overline{z}) > \varepsilon) \vee (Z_{old}(\overline{z}) - Z_{new}(\overline{z}) < -\varepsilon)\ ]\ ]\!]$
until $n = n_{max}$ **or** $\mathrm{B} = \lambda\overline{z}[0];$
$\mathrm{Q} := \textsc{Mtbdd}^{I[Y:=\mathrm{B}, Z_{new}:=\mathrm{Q}_n]}[\![\ \lambda\overline{z}\ [\ Y(\overline{z}) * \bot\ +\ (1 - Y(\overline{z})) * Z_{new}(\overline{z})\ ]\ ]\!].$

Figure 1: MTBDD-based method for computing the the semantics of the (well-formed mu-calculus

The mixed calculus: The syntax of *mixed* expressions and terms is given by the following production system.

$$\begin{array}{lcl} \mathsf{expr} & ::= & q \mid z \mid \mathsf{expr}_1\ op\ \mathsf{expr}_2 \mid -\mathsf{expr} \mid \mathsf{term}(\kappa_1,\ldots,\kappa_n) \mid \sum_z[\mathsf{expr}] \mid \min_z[\mathsf{expr}] \mid \max_z[\mathsf{expr}] \\ \mathsf{term} & ::= & Z \mid \mathrm{Q} \mid \lambda z_1,\ldots,z_n[\mathsf{expr}] \mid limit\ Z[\ \mathsf{term} \uparrow \mathsf{term}_0\ ] \end{array}$$

where $q \in \mathbb{R}$, $op \in Op$, Q is a MTBDD over $(\vartheta_1,\ldots,\vartheta_n)$, $\kappa_1,\ldots,\kappa_n \in \{0,1\} \cup IndVar$, z, $z_1,\ldots,z_n \in IndVar$ such that $z_1,\ldots,z_n$ are pairwise distinct and Z is an n-ary term variable. For $\overline{z} = (z_1,\ldots,z_n)$, we briefly write $\sum_{\overline{z}}$ rather than $\sum_{z_1}\ldots\sum_{z_n}$. Similarly, $\min_{\overline{z}}$, $\max_{\overline{z}}$ or $\lambda\overline{z}$ have the obvious meaning. Free and bounded occurrences of individual or term variables in mixed expressions or terms, closedness of mixed terms and the boolean subcalculus are defined in the obvious way.

For the closed mixed terms we compute the corresponding MTBDD with the model checker described for the algebraic mu-calculus. For this, we "retransform" any closed mixed term term into a well-formed algebraic term in the following way. We use individual variables of $IndVar' = IndVar \cup \{\zeta_1,\ldots,\zeta_N\}$ where $\zeta_1,\ldots,\zeta_N$ are pairwise distinct "fresh" variables (not contained in $IndVar$) and where N is the maximal number n where term contains a subterm of the form $\mathsf{term}'(\kappa_1,\ldots,\kappa_n)$. We extend the order $<$ on $IndVar$ to an order on $IndVar'$ (also denoted by $<$) where we define $z < \zeta_1 < \ldots < \zeta_N$ for all $z \in IndVar$. For each individual variable $z \in IndVar'$ we choose a "new" 1-ary term variable Z_z. For each MTBDD Q over $(\vartheta_1,\ldots,\vartheta_n)$ that occurs as an n-ary subterm of term we choose a "new" term variable Z_{Q} of arity n. Each subexpression $\mathsf{term}'(\kappa_1,\ldots,\kappa_n)$ of term is replaced by the mixed expression $\mathsf{term}''(y_1,\ldots,y_k)$ where $\mathsf{term}'' = \lambda y_1,\ldots,y_k[\sum_{\zeta_1,\ldots,\zeta_n}[\mathsf{term}'(\zeta_1,\ldots,\zeta_n) * \mathsf{bexpr}]]$, $\{y_1,\ldots,y_k\} = \{\kappa_1,\ldots,\kappa_n\} \cap IndVar$ such that $y_1 < y_2 < \ldots < y_k$ and

$$\mathsf{bexpr} = \bigwedge_{\substack{1\le i\le n\\ \kappa_i = 0}} \neg\zeta_i \ \wedge \bigwedge_{\substack{1\le i\le n\\ \kappa_i = 1}} \zeta_i \ \wedge \bigwedge_{1\le j\le k} \bigwedge_{\substack{1\le i\le n\\ \kappa_i = y_j}} (\zeta_i \leftrightarrow y_j).$$

Each occurrence of a MTBDD Q as a subterm of term is replaced by Z_{Q}, each occurrence of an individual variable $z \in IndVar'$ as a subexpression of term by the algebraic expression $Z_z(z)$. Let $term$ be the resulting algebraic term. Clearly, $term$ is well-formed with respect to the chosen ordering $<$ on $IndVar'$. We define $\textsc{Mtbdd}[\![\mathsf{term}]\!] = \textsc{Mtbdd}^I[\![term]\!]$ where I is an interpretation with $I(Z_{\mathrm{Q}}) = \mathrm{Q}$ and $I(Z_z)$ is the BDD over (ϑ_1) that represents the boolean function $F(b) = b$. In what follows, we identify each closed mixed term term with the corresponding MTBDD $\textsc{Mtbdd}[\![\mathsf{term}]\!]$. For instance, if P is a MTBDD over $(\vartheta_1,\vartheta_2,\vartheta_3)$ then $\lambda y[\mathrm{P}(0,y,y)]$ is identified with the MTBDD for $\lambda y[term(y)]$ where $term = \lambda y[\sum_{\zeta_1,\zeta_2,\zeta_3} Z_{\mathrm{P}}(\zeta_1,\zeta_2,\zeta_3) * bexpr]$ and $bexpr = \neg Z_{\zeta_1}(\zeta_1) \wedge (Z_{\zeta_2}(\zeta_2) \leftrightarrow Z_y(y)) \wedge (Z_{\zeta_3}(\zeta_3) \leftrightarrow Z_y(y))$.

4 Applications

As the relational mu-calculus is contained in the algebraic mu-calculus all problems that can be reduced to the relational mu-calculus (and solved with the model checker of [BCM+92], e.g. the symbolic verification methods for parallel systems) can also be solved with our framework. In this section, we present some possible applications of the algebraic mu-calculus (more precisely, the mixed calculus) where the relational mu-calculus fails as it is not able to deal with arithmetic operations.

4.1 Solving graph theoretical problems with MTBDDs

[BCM+92] shows that several reachability problems in graphs can be embedded into the relational mu-calculus (and hence, can be solved using the BDD-based model checker for the relational mu-calculus). Here, we present a simple example (the problem of cheapest paths in directed graphs) where our MTBDD-based framework can be applied while the approach with BDDs fails as it cannot handle arithmetic operations.

Let $G = (V, E, cost)$ be a finite directed graph with a positive cost function, i.e. V is a finite set of vertices, $E \subseteq V \times V$ a set of directed edges and $cost : E \to \mathbb{R}_{>0}$ a function that assigns to each edge (v, w) the cost $cost(v, w)$ for passing the edge from v to w. For simplicity, we assume that $(v, v) \notin E$ for all vertices v and extend $cost$ to a function $C : V \times V \to \overline{\mathbb{R}}$ where we put $C(v, v) = 0$ and $C(v, w) = \bot$ if $(v, w) \notin E$, $v \neq w$. We encode V in $\{0,1\}^n$ where $n = \lceil \log |V| \rceil$ and represent G by a MTBDD C over $(\vartheta_1, \ldots, \vartheta_{2n})$ such that $F_C(\overline{b}, \overline{c}) = C(v, w)$ if $\overline{b}$ is the encoding of v and $\overline{c}$ the encoding of w (and $F_C(\overline{b}, \overline{c}) = 0$ if $\overline{b}$ or $\overline{c}$ does not represent the encoding of a vertex). The MTBDD Q for the mixed term $limit\ Z\ [\ \lambda v, w[\ \min_u\ [\ Z(v, u) + C(u, w)\ ]\ ]\ \uparrow C\ ]$ represents the function $V \times V \to \overline{\mathbb{R}}$ whichs assigns to each pair (v, w) of vertices the cost for a cheapest path from v to w in G. (Here, we assume an extension of op_{min} on $\overline{\mathbb{R}}$ where $q\ op_{min} \bot = \bot\ op_{min}\ q = q$ and that u, v, w are n-tuples of pairwise distinct individual variables that range over the encodings of the vertices.)

4.2 Matrix operations using MTBDDs

As shown in [CFM+93, HMP+96, BFG+97], MTBDDs yield an efficient representation for matrices. In this section, we briefly explain how the mixed calculus can be used to implement operations on matrices. As described in [CFM+93], using an encoding of the indices for the rows and columns in $\{0,1\}^n$ and $\{0,1\}^m$ respectively, any $2^n \times 2^m$-matrix $\mathbf{A}$ can be represented by a MTBDD over $n + m$ variables. For simplicitity of the expressions and terms, we deal here with a representation of a $2^n \times 2^m$-matrix $\mathbf{A}$ by a MTBDD A over $(\vartheta_1, \ldots, \vartheta_{n+m})$ where $(\vartheta_1, \ldots, \vartheta_n)$ stands for the index for the row while $(\vartheta_{n+1}, \ldots, \vartheta_{n+m})$ ranges over the indices for the columns.

Let A_1 and A_2 be MTBDDs over $(\vartheta_1, \ldots, \vartheta_{n+m})$ and $(\vartheta_1, \ldots, \vartheta_{m+k})$ that represent matrices $\mathbf{A}_1$ and $\mathbf{A}_2$ respectively (where $\mathbf{A}_1$ is a $2^n \times 2^m$-matrix and $\mathbf{A}_2$ a $2^m \times 2^k$-matrix). The MTBDD corresponding to the closed mixed term $\lambda \overline{x}, \overline{z}\ [\ \sum_{\overline{y}}\ A_1(\overline{x}, \overline{y}) * A_2(\overline{y}, \overline{z})\ ]$ represents the product $\mathbf{A}_1 \cdot \mathbf{A}_2$. Here, $\overline{x} = (x_1, \ldots, x_n)$, $\overline{y} = (y_1, \ldots, y_m)$ and $\overline{z} = (z_1, \ldots, z_k)$.

In the literature about numerical analysis a variety of iterative methods for solving linear equation systems are proposed, see e.g. [Var62, YG73]. Most of them can be implemented with MTBDDs using our method to compute the approximate meaning of mixed terms. For simplicity, we only describe how to embed the following "naive" method into the mixed calculus. This simple method can be viewed as the basis of several methods, e.g. the methods by Jacobi or Gauss-Seidel or the relaxation methods. (See [HMP+96] for a more detailed discussion how to use MTBDDs for solving linear equation systems by iterative methods.)

Let $\mathbf{A}$ be a $2^n \times 2^n$-matrix, $\mathbf{I}$ the $2^n \times 2^n$-identity matrix. We assume that $\|\mathbf{I} - \mathbf{A}\| < 1$ for some matrix norm $\|\cdot\|$. Then, $\mathbf{I} - \mathbf{A}$ is regular and, for each 2^n-vector $\mathbf{q}$, the sequence $(\mathbf{z}^k)_{k \geq 0}$ converges to the unique solution of the equation system $(\mathbf{I} - \mathbf{A}) \cdot \mathbf{z} = \mathbf{q}$ where $\mathbf{z}^0$ is an arbitrary real vector with 2^n components and $\mathbf{z}^{k+1} = \mathbf{q} + \mathbf{A} \cdot \mathbf{z}^k$. We represent $\mathbf{A}$ by a MTBDD A over $2n$ dummy variables and $\mathbf{q}$ by a MTBDD Q over n dummy variables. The MTBDD corresponding to the mixed term $limit\ Z\ [\ \lambda \overline{z}[\ Q(\overline{z})\ +\ \sum_{\overline{y}}\ A(\overline{z}, \overline{y}) * Z(\overline{y})\ ]\ \uparrow\ Q_0\ ]$ represents the solution of $(\mathbf{I} - \mathbf{A}) \cdot \mathbf{z} = \mathbf{q}$. (Here, Q_0 is a MTBDD over n dummy variables with terminal values in $\mathbb{R}$.)

Similarly, other iterative methods for solving classical problems of linear algebra (e.g. the iteration methods by Mises or Wielandt for computing eigenvalues) can be embedded into the mixed calculus.

4.3 Symbolic model checking for probabilistic processes

In this section we describe how our algorithm of Section 3 can be used to obtain MTBDD-based verification methods for probabilistic processes. First, we deal with the logic *PCTL* (probabilistic computation tree logic) as introduced by Hansson & Jonsson and show how (a variant of) the model checking algorithm of [HJ94] can be implemented using MTBDDs. (See also [BCH+97] which describes a MTBDD-based model checker for the method presented in [HJ94] without using the algebraic mu-calculus.) In the full paper we show how the algebraic mu-calculus can be used to obtain a MTBDD-based model checker for *PCTL* interpreted over Markov chains with non-determinism [BA95]. Second, we briefly explain how to use the mixed calculus for computing the bisimulation equivalence classes à la Larsen

& Skou [LS91].

Markov chains: A *Markov chain* is a pair $(S, \mathbf{P})$ where S is a finite set of *states* and $\mathbf{P} : S \times S \to [0,1]$ a function such that, for all $s \in S$, $\sum_{t \in S} \mathbf{P}(s,t) = 1$. A *path* in a Markov chain $(S, \mathbf{P})$ is a nonempty (finite or infinite) sequence $\pi = s_0 s_1 s_2 \ldots$ of states $s_i \in S$ such that $\mathbf{P}(s_i, s_{i+1}) > 0$, $i = 0, 1, 2, \ldots$. A *fulpath* is an infinite path. $Path(s)$ denotes the collection of all fulpaths $\pi = s_0 s_1 s_2 \ldots$ that start in s (i.e. $s_0 = s$). Let $\Sigma(s)$ be the smallest σ-field on $Path(s)$ that contains all basic cylinders $Cylinder(s_0 \ldots s_k) = \{\pi \in Path(s) : s_0 \ldots s_k \text{ is a prefix of } \pi\}$ where $s_0 \ldots s_k$ ranges over all finite paths that start in s. *Prob* denotes the unique probability measure on $\Sigma(s)$ such that $Prob(Cylinder(s_0 \ldots s_k)) = \mathbf{P}(s_0, s_1) \cdot \ldots \cdot \mathbf{P}(s_{k-1}, s_k)$.

To represent the transition matrix of a Markov chain by a MTBDD we abstract from the names of states and instead, similarly to [BCM+92, CGL93], use an encoding of the states by bit vectors (cf. [BCH+97]). Let $(S, \mathbf{P})$ be a Markov chain and $n = \lceil \log |S| \rceil$. We fix an encoding of S in $\{0,1\}^n$, i.e. an injective function $code : S \to \{0,1\}^n$, and replace $(S, \mathbf{P})$ by the Markov chain $(\{0,1\}^n, \overline{\mathbf{P}})$ where $\overline{\mathbf{P}} : \{0,1\}^{2n} \to [0,1]$ is given by $\overline{\mathbf{P}}(\overline{b}, \overline{c}) = \mathbf{P}(s,t)$ if $\overline{b} = code(s)$ and $\overline{c} = code(t)$, $\overline{\mathbf{P}}(\overline{b}, \overline{c}) = 1$ if $\overline{b} = \overline{c} \notin code(S)$ and $\overline{\mathbf{P}}(\overline{b}, \overline{c}) = 0$ in all other cases. We represent $\overline{\mathbf{P}}$ by a MTBDD P over $(\vartheta_1, \ldots, \vartheta_{2n})$ where $F_{\mathrm{P}} = \overline{\mathbf{P}}$. In what follows, we use n-tuples s and t of pairwise distinct individual variables to range over the encodings of the states. Similarly, we can deal with *action-labelled* Markov chains. Let $(S, \mathbf{P}, Act)$ be an action-labelled Markov chain (i.e. Act is a finite set of actions and $\mathbf{P} : S \times Act \times S \to [0,1]$ is a function with $\sum_t \mathbf{P}(s, \alpha, t) \in \{0,1\}$ for all $s \in S$ and $\alpha \in Act$.) We use an encoding of the actions in $\{0,1\}^k$ (where $k = \lceil \log |Act| \rceil$) and represent $\mathbf{P}$ by a MTBDD for the function $\overline{\mathbf{P}} : \{0,1\}^{2n+k} \to [0,1]$ which is given by $\overline{\mathbf{P}}(\overline{b}, \overline{a}, \overline{c}) = \mathbf{P}(s, \alpha, t)$ if $\overline{b} = code(s)$, $\overline{a} = code(\alpha)$ and $\overline{c} = code(t)$ and $\overline{\mathbf{P}}(\overline{b}, \overline{a}, \overline{c}) = 0$ in all other cases.

Model checking for the probabilistic temporal logic *PCTL*: *PCTL* [HJ94] is an extension of *CTL* [CES86] which contains propositional logic and the temporal operator $\mathcal{U}$ ("until") which is used in connection with an interval of probabilities. (For simplicity, we omit the bounded until operator and use slightly different notations for the until operator than [HJ94].) The syntax of *PCTL* is as follows:

$$\Phi \quad ::= \quad tt \mid a \mid \Phi_1 \wedge \Phi_2 \mid \neg\Phi \mid \mathbb{P}_{\bowtie p}(\Phi_1 \mathcal{U} \Phi_2)$$

where a is an atomic proposition, $p \in [0,1]$ and $\bowtie \in \{\leq, \geq, <, >\}$. Intuitively, $\mathbb{P}_{\bowtie p}(\Phi_1 \mathcal{U} \Phi_2)$ asserts that the probability of fulpaths starting in the current state fulfilling $\Phi_1 \mathcal{U} \Phi_2$ is $\bowtie p$. Operators for modelling "eventually" or "always" can be derived by $\mathbb{P}_{\bowtie p}(\Diamond\Phi) = \mathbb{P}_{\bowtie p}(tt \mathcal{U} \Phi)$ and $\mathbb{P}_{\bowtie p}(\Box\Phi) = \mathbb{P}_{\overline{\bowtie}(1-p)}(\Diamond\neg\Phi)$ where $\overline{\leq} = \geq$, $\overline{<} = >$, $\overline{\geq} = \leq$ and $\overline{>} = <$. For instance, if *error* is an atomic proposition that characterizes all states where a system error has happened then $\mathbb{P}_{<0.001}(\Diamond error)$ asserts that the probability for a system error is less than 0.001. If $crit_1$ and $crit_2$ are atomic propositions stating that certain processes $\mathcal{P}_1$ or $\mathcal{P}_2$ are in their critical sections then $\mathbb{P}_{\geq p}(\Box(\neg crit_1 \vee \neg crit_2))$ asserts mutual exclusion with probability at least p. *PCTL* formulas can be interpreted over the states of a Markov chain $(S, \mathbf{P})$ which is endowed with a labelling function L that assigns to each state s a set $L(s)$ of atomic propositions. (Intuitively, $L(s)$ is the set of atomic propositions that hold in s.) The satisfaction relation $\models$ is defined as follows. As usual, $s \models tt$ for all $s \in S$, $s \models a$ iff $a \in L(s)$, $s \models \Phi_1 \wedge \Phi_2$ iff $s \models \Phi_i$, $i = 1, 2$ and $s \models \neg\Phi$ iff $s \not\models \Phi$. For formulas involving the probabilistic operator $\mathbb{P}_{\bowtie}$, we have $s \models \mathbb{P}_{\bowtie p}(\Phi_1 \mathcal{U} \Phi_2)$ iff $Prob\{\pi \in Path(s) : \pi \models \Phi_1 \mathcal{U} \Phi_2\} \bowtie p$ where the satisfaction relation for the until operator is defined as in the non-probabilistic case.

We describe a method for computing the BDD representation $\textsc{Sat}(\Phi)$ of the (characteristic function of the) set $Sat(\Phi) = \{s \in S : s \models \Phi\}$. Using the mixed calculus, the BDDs $\textsc{Sat}(\Phi)$ are computed by structural induction. We assume that for each atomic proposition a there is a BDD B_a that represents the (characteristic function of the) set of states where a holds, i.e. the set $\{s \in S : a \in L(s)\}$. The handling of the propositional formulas is clear: We put $\textsc{Sat}(tt) = \lambda s[1]$, $\textsc{Sat}(a) = \mathrm{B}_a$, $\textsc{Sat}(\Phi_1 \wedge \Phi_2) = \lambda s[\textsc{Sat}(\Phi_1)(s) \wedge \textsc{Sat}(\Phi_2)(s)]$ and $\textsc{Sat}(\neg\Phi) = \lambda s[\neg \textsc{Sat}(\Phi)]$. In contrast to [HJ94] we use an iterative method which involves the limit operator. We define $\textsc{Sat}(\mathbb{P}_{\bowtie p}(\Phi_1 \mathcal{U} \Phi_2)) = \lambda s[\mathsf{term}(s) \bowtie p]$ where

$$\mathsf{term} \;=\; \mathit{limit}\ Z\ [\ \lambda s\,[\textsc{Sat}(\Phi_2)(s) \;+\; (1 - \textsc{Sat}(\Phi_2)(s)) * \textsc{Sat}(\Phi_1)(s) * (\textstyle\sum_t\ [\ \mathrm{P}(s,t) * Z(t)\ ])\ ]\ \uparrow\ \lambda s[0]\ ],$$

The correctness of this method is based on Theorem 1 and the observation that the function $F : S \to [0,1]$, $F(s) = Prob\{\pi \in Path(s) : \pi \models \Phi_1 \mathcal{U} \Phi_2\}$ is the least fixed points of the operator $\Omega : (S \to [0,1]) \to (S \to [0,1])$ which is given by: $\Omega(F)(s) = 1$ if $s \in S_2$, $\Omega(F)(s) = 0$ if $s \in S \setminus (S_1 \cup S_2)$ and $\Omega(F)(s) = \sum_{t \in S_1 \cup S_2} \mathbf{P}(s,t) \cdot F(t)$ if $s \in S_1 \setminus S_2$. Here, $S_i = Sat(\Phi_i)$.

Deciding bisimulation equivalence: We show how the bisimulation equivalence classes of an action-labelled Markov chain $(S, \mathbf{P}, Act)$ can be characterized by a mixed term. A *bisimulation* is an equivalence relation on S such that, for all $(s, s') \in R$, $\alpha \in Act$ and all $C \in S/R$, $\sum_{t \in C} \mathbf{P}(s, \alpha, t) = \sum_{t \in C} \mathbf{P}(s', \alpha, t)$. s and s' are called *bisimilar* iff there is a bisimulation that contains (s, s'). (See [LS91].)

As in the non-probabilistic case, bisimulation equivalence can be characterized as the greatest fixed point of the set-valued operator which assigns to each equivalence relation R on the state space S those equivalence relation that

identifies exactly those states s and s' where $(s,s') \in R$ and $\sum_{t \in C} \mathbf{P}(s,\alpha,t) = \sum_{t \in C} \mathbf{P}(s',\alpha,t)$ for all $\alpha \in Act$ and all $C \in S/R$. Thus, the bisimulation equivalence classes are represented by the BDD corresponding to the mixed boolean term $gfp\, Z\, [\, \lambda s, s'\, [\, \forall \alpha\, \forall t\, [\, \mathsf{bexpr}\,]\,]\,]$ where bexpr is $(\sum_{t'}\ Z(t,t') * \mathrm{P}(s,\alpha,t')\ )\ =\ (\sum_{t'}\ Z(t,t') * \mathrm{P}(s',\alpha,t')\ )$.

5 Concluding remarks

We have presented a new general framework by which a variety of fixed point problems involving real numbers can be solved using our MTBDD-based method presented in Section 3. On the basis of Theorem 1, it is possible to formalize a series of syntactic requirements on *term* and the free occurrences of Z in *term* that ensure the existence of a least and greatest fixed point of the operator $F \mapsto [\![term]\!]^{\mathcal{M}[Z:=F]}$ and that allow us to replace the limit operator by fixed points operators $lfp Z[term]$ and $gfp Z[term]$ whose semantics with respect to a $\Re$-model $\mathcal{M}$ is always a real number in $\Re$. (For this, $\Re$-closedness has to be replaced by a stronger but purely syntactic condition. This will be explained in the full paper.) The resulting subcalculi capture both the *declarative* interpretations (i.e. $F = \Omega(F)$) and the *procedural* interpretations (by limits) of the fixed point concepts. For these subcalculi, the method of Section 3 might be modified in such a way that the semantics of the least/greatest fixed point operators is computed by alternative methods, possibly yielding the exact value rather than the approximate meaning that we obtain by our iterative method. For instance, in the case of probabilistic model checking against *PCTL* formulas (Section 4.3), the precise probabilities for the path formulas $\Phi_1 \mathcal{U} \Phi_2$ in a Markov chain with rational transition probabilities can be obtained by solving linear equation systems with the Gauss elimination method (rather than the iterative method that we use).

As future work, we plan to implement our algorithm to get experimental results and to extend the algebraic mu-calculus to a *typed* calculus that allows us to use alternative (possibly more efficient) representations for special functions, e.g. for integer-valued functions by EVBDDs [LPV94], BMDs [BC95] or HDDs [CFZ96].

References

[BFG+97] I. Bahar, E. Frohm, C. Gaona, G. Hachtel, E. Macii, A. Padro, F. Somenzi: Algebraic Decision Diagrams and their Applications, Journal of Formal Methods in Systems Design, Vol. 10, No. 2/3, pp 171-206, 1997.

[BCH+97] C. Baier, E. Clarke, V. Hartonas-Garmhausen, M. Kwiatkowska, M. Ryan: Symbolic Model Checking for Probabilistic Processes, Proc. ICALP'97, Lecture Notes in Computer Science 1256, pp 430-440, 1997.

[BA95] A. Bianco, L. de Alfaro: Model Checking of Probabilistic and Nondeterministic Systems, Proc. Foundations of Software Technology and Theoretical Computer Science, LNCS 1026, pp 499-513, 1995.

[Bry86] R. Bryant: Graph-Based Algorithms for Boolean Function Manipulation, IEEE Transactions on Computers, Vol. C-35, No. 8, pp 677-691, 1986.

[BC95] R. Bryant, Y. Chen: Verification of Arithmetic Functions with Binary Moment Diagrams, Proc. 32nd ACM/IEEE Design Automation Conference, pp 535-541, 1995.

[BCM+92] J. Burch, E. Clarke, K. McMillan, D. Dill, L. Hwang: Symbolic Model Checking: 10^{20} States and Beyond, Information and computation, Vol. 98 (2), pp 142-170, 1992.

[CES86] E. Clarke, A. Emerson, P. Sistla: Automatic Verification of Finite-State Concurrent Systems using Temporal Logic Specifications, ACM Trans. Programming Languages and Systems, 1(2), 1986.

[CFM+93] E. Clarke, M. Fujita, P. McGeer, J. Yang, X. Zhao: Multi-Terminal Binary Decision Diagrams: An Efficient Data Structure for Matrix Representation, In IWLS'93: International Workshop on Logic Synthesis, Tahoe City, 1993.

[CFZ96] E. Clarke, M. Fujita, X. Zhao: Multi-Terminal Binary Decision Diagrams and Hybrid Decision Diagrams, Representations of Discrete Functions. In T. Sasao and M. Fujita, eds, Kluwer Academic Publishers, pp 93-108, 1996.

[CGL93] E. Clarke, O. Grumberg, D. Long: Verfication Tools for Finite-State Concurrent Programs, Proc. REX, LNCS 803, pp 124-175, 1993.

[HMP+96] G. Hachtel, E. Macii, A. Padro, F. Somenzi: Markovian Analysis of Large Finite State Machines, IEEE Transactions on CAD/ICAS, Vol. 15, No. 12, pp 1479-1493, 1996.

[HJ94] H. Hansson, B. Jonsson: A Logic for Reasoning about Time and Probability, Formal Aspects of Computing, Vol. 6, pp 512-535, 1994.

[HK97] M. Huth, M. Kwiatkowska: Quantitative Analysis and Model Checking, Proc. LICS'97, IEEE Computer Society Press, 1997.

[LPV94] Y. Lai, M. Pedram, B. Vrudhula: Edge-Valued Binary Decision Diagrams for Integer Linear Programming, Spectral Transformation, and Function Decomposition, IEEE Transactions on CAD, Vol. 13, No. 8, pp 959-975, 1994.

[LS91] K. Larsen, A. Skou: Bisimulation through Probabilistic Testing, Information and Computation, Vol. 94, pp 1-28, 1991.

[McM93] K. McMillan: Symbolic Model Checking: An Approach to the State Explosion Problem, Kluwer Academic Publishers, 1993.

[Par74] D. Park: Finiteness is Mu-ineffable: Theory of Computation Report No. 3, The University of Warwick, 1974.

[Sei96] H. Seidl: A Modal Mu-Calculus for Durational Transition Systems, Proc. LICS'96, pp 128-137, 1996.

[Var62] R. Varga: Matrix Iterative Analysis, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1962.

[YG73] D. Young, R. Gregory: A Survey of Numerical Mathematics, Vol. II, Addison-Wesley Publishing Company Reading, Massachussetts, 1973.