A complete deductive system for the μ-calculus

Igor Walukiewicz[1,2]

BRICS[3]
Department of Computer Science
University of Aarhus
Ny Munkegade
DK-8000 Aarhus C, Denmark

Abstract

The propositional μ-calculus as introduced by Kozen in [12] is considered. In that paper a finitary axiomatisation of the logic was presented but its completeness remained an open question. Here a different finitary axiomatisation of the logic is proposed and proved to be complete. The two axiomatisations are compared.

1 Introduction

It is now common to view computer programs as state transformers, that is actions that can change one state of computer hardware to another. In contrast with classical logics the notion of *change* is intrinsic in *modal* logics. Within modal logic, one can speak about multiple possible *worlds* and relations between them, as, for example, the changes of an environment during time. This property makes the modal logic a valuable tool for description of program behaviour that is itself observable through the changes of computer states.

The properties of logic we should consider, when we have program verification in mind, are expressiveness, completeness and decidability. The more expressive is the logic, the more properties of the systems we can describe. A complete axiom system allows us to reason about the properties. This facilitates "proof theoretic" approach to specification and verification. Decidability and particularly computational complexity of the decision procedure is important for machine aided verification. Finally, there is a question of model checking, i.e., establishing the truth of a formula in a given state of a (usually very large) structure, which describes a behaviour of a complex program.

We consider propositional μ-calculus in the form introduced by Kozen in [12]. It turned out that this logic is very well suited for specification and verification purposes. First it is very expressive and most of the other logics of programs

[1]This work was partially supported by Polish KBN grant No. 2 1192 91 01
[2]On the leave from: Institute of Informatics, Warsaw University, Banacha 2, 02-097 Warsaw, POLAND
[3]Basic Research in Computer Science, Centre of the Danish National Research Foundation.

can be encoded into the μ-calculus. On binary trees the logic is as expressive as monadic second order logic of two successors [18, 6]. On the other hand the logic is manageable. Satisfiability problem for the logic was shown to be EXPTIME-complete [15, 23, 4] which means that it is of the same complexity as for many much less expressive logics. The best known upper bound for the model checking problem is exponential but it is polynomial if nesting of fixpoints is bounded [3, 1].

One of lacking elements in this picture was finitary complete axiomatisation of the logic (infinitary axiomatisation was given in [13]). There exist finitary axiomatisations for many weaker propositional logics of programs like PDL [8, 19, 14, 11, 16], CTL* [5] or Process Logic [9]. Completeness proofs for these logics use the so called Henkin method which consists of constructing a model for a non refutable formula. The use of this method depends on the ability of syntactic model construction which is provided by collapsed model theorem or a result of similar kind. For several more expressive logics of programs like: PDLΔ, PAL, temporal μ-calculus (see [10, 22]) the completeness problem remains open. One of the reasons for this is that these logics, as well as the μ-calculus, do not enjoy the collapsed model property (see for example [10]), hence known methods do not work in this case.

In [12] Kozen proposed a natural axiom system and showed that it can prove all valid formulas satisfying some syntactic restrictions. In the present paper a different finitary axiomatisation is proposed and shown to be complete for the whole logic. The systems differ only in one, the most important, rule. The original system has Park's rule:

$$\frac{\alpha(\varphi) \Rightarrow \varphi}{\mu X.\alpha(X) \Rightarrow \varphi}$$

which expresses the property that the least fixpoint is the least pre-fixpoint. The rule proposed here is derived from Knaster-Tarski characterisation of the least fixpoint as the limit of a chain of approximations. We show that Park's rule is derivable in our system but our rule is not derivable in Kozen's system. Hence the presented completeness result cannot be used to prove the completeness of the, weaker, Kozen's axiomatisation.

Although our rule may seem to be less natural than Park's rule we think that our proof system has some advantages.

First we think that it is easy to prove the facts in this system mainly due the strength of our rule. Although we have a cut rule in the system, the completeness proof gives an algorithm for constructing a proof of a given valid formula. Hence there is a way of controlling cut's.

We believe that the system can be naturally integrated with a decidability algorithm. This allows construction of a tool which given a formula will construct either a proof of it or a counterexample model. Moreover this tool should also allow interaction with the user who can speed up some proving steps.

For propositional modal logics of programs reduction to ω-automata is a general method of model construction. As our method of proving completeness is closely related to this technique we hope that it can be also generalised to other logics.

The outline of the paper is as follows. We begin by giving basic definitions and recalling a result from [17] which we will use. Next we present our axiomatisation, prove some of its properties and relate it to Kozen's system. Finally we present the completeness proof.

Acknowledgements

I am very grateful to Dexter Kozen for his introduction to the μ-calculus and sharing with me his understanding of the subject. Damian Niwiński was the one from whom I learned automata theory. Long discussions with him made development of this material possible. Results presented here come from my PhD thesis. Jerzy Tiuryn guided development of the thesis and was always ready with help and advice. I would also like to thank my PhD referees: Dexter Kozen, Andrzej W. Mostowski and Paweł Urzyczyn for their valuable comments. I am indebted to my first two supervisors: Leszek Holenderski and Grażyna Mirkowska, without whom I would not have started the subject at all.

2 Preliminary definitions

In this section we will give basic definitions and recall a result from [17] which we will use in the completeness proof.

Let $Prop = \{p, q, \ldots\}$ be a set of propositional letters, $Var = \{X, Y, \ldots\}$ a set of variables and $Act = \{a, b, \ldots\}$ a set of actions. Formulas of the μ-calculus over this three sets can be defined by the following grammar:

$$F := Var \mid Prop \mid \neg F \mid F \vee F \mid F \wedge F \mid \langle Act \rangle F \mid [Act]F \mid \mu Var.F \mid \nu Var.F$$

Additionally we require that in formulas of the form $\mu X.\alpha(X)$ and $\nu X.\alpha(X)$, variable X occurs in $\alpha(X)$ only positively, i.e., under even number of negations. We will use σ to denote μ or ν. We use ff as an abbreviation for a formula $p \wedge \neg p$ for some propositional constant p.
Formulas are interpreted in Kripke models of the form $\mathcal{M} = \langle S, R, \rho \rangle$, where:

- S is a nonempty set of states,
- $R : Act \to \mathcal{P}(S \times S)$ is a function assigning a binary relation on S to each action in Act.
- $\rho : Prop \to \mathcal{P}(S)$ is a function assigning a set of states to each propositional letter in $Prop$.

For a given model $\mathcal{M}$ and a valuation $Val : Var \to \mathcal{P}(S)$, the set of states in which a formula α is true, $\| \alpha \|_{Val}^{\mathcal{M}}$ is defined inductively as follows (we will omit superscript $\mathcal{M}$ when it causes no ambiguity):

$$\begin{aligned} \| X \|_{Val} &= Val(X) \\ \| p \|_{Val} &= \rho(p) \\ \| \neg\alpha \|_{Val} &= S - \| \alpha \|_{Val} \end{aligned}$$

$$
\begin{aligned}
\| \alpha \wedge \beta \|_{Val} &= \| \alpha \|_{Val} \cap \| \beta \|_{Val} \\
\| \alpha \vee \beta \|_{Val} &= \| \alpha \|_{Val} \cup \| \beta \|_{Val} \\
\| \langle a \rangle \alpha \|_{Val} &= \{ s : \exists s'. (s, s') \in R(a) \wedge s' \in \| \alpha \|_{Val} \} \\
\| [a] \alpha \|_{Val} &= \{ s : \forall s'. (s, s') \in R(a) \ \Rightarrow s' \in \| \alpha \|_{Val} \} \\
\| \mu X. \alpha(X) \|_{Val} &= \bigcap \{ S' \subseteq S : \| \alpha \|_{Val[S'/X]} \subseteq S' \} \\
\| \nu X. \alpha(X) \|_{Val} &= \bigcup \{ S' \subseteq S : S' \subseteq \| \alpha \|_{Val[S'/X]} \}
\end{aligned}
$$

When we write $\mathcal{M}, s, Val \models \alpha$ we mean that $s \in \| \alpha \|^{\mathcal{M}}_{Val}$. For a set of formulas Γ and a formula β, we write $\Gamma \models \beta$ to mean that for any structure $\mathcal{M}$, state s and valuation Val: if $\mathcal{M}, s, Val \models \Gamma$ then $\mathcal{M}, s, Val \models \beta$. If $\Gamma = \{\alpha\}$ is one element set we just write $\alpha \models \beta$.

A *sequent* is pair of finite sets of formulas which we write $\Gamma \vdash \Delta$. The meaning of such a sequent is that a conjunction of formulas from Γ implies a disjunction of formulas from Δ, in other words it is equivalent to a formula $\neg(\bigwedge \Gamma) \vee \bigvee \Delta$. We use $\bigwedge \Gamma$ and $\bigvee \Gamma$ for respectively conjunction and disjunction of formulas from Γ. Conjunction of the empty set is true and disjunction of the empty set is false.

Definition 2.1 We call a formula *positive* iff all negations in the formula appear only before propositional constants and free variables.

Variable X in $\mu X.\alpha(X)$ is *guarded* iff every occurrence of X in α is in the scope of some modality operator $\langle\rangle$ or $[]$. We say that a formula is guarded iff every bound variable in the formula is guarded. ■

Proposition 2.2 (Kozen) *Every formula is equivalent to a positive guarded formula.*

Proof

Let φ be arbitrary formula. We first show how to obtain an equivalent guarded formula. The proof proceeds by induction on the structure of the formula with the only difficult case for fixpoint formulas.

Let φ be of the form $\mu X.\alpha(X)$ with $\alpha(X)$ a guarded formula. Suppose X is unguarded in some subformula of $\alpha(X)$ of the form $\sigma Y.\beta(Y, X)$. As $\alpha(X)$ is guarded, Y must be guarded in $\sigma Y.\beta(Y, X)$. Equivalence $\sigma Y.\beta(Y, X) \equiv \beta(\sigma Y.\beta(Y, X), X)$ gives us a formula with all unguarded occurrences of X outside the fixpoint operator. Repeating this process we obtain a formula equivalent to $\alpha(X)$ with all unguarded occurrences of X not in the scope of a fixpoint operator.

Now using the laws of classical propositional logic we can transform this formula to a conjunctive normal form (considering fixpoint formulas and formulas of the form $\langle a \rangle \gamma$ and $[a]\gamma$) as propositional constants. This way we obtain a formula

$$(X \vee \alpha_1(X)) \wedge \ldots \wedge (X \vee \alpha_i(X)) \wedge \beta(X) \tag{1}$$

where all occurrences of X in $\alpha_1(X), \ldots, \alpha_i(X), \beta(X)$ are guarded. Variable X occurs only positively in (1) because it did so in our original formula. Formula (1) is equivalent to

$$(X \vee (\alpha_1(X) \vee \ldots \vee \alpha_i(X))) \wedge \beta(X)$$

We will show that $\mu X.(X \vee \bar{\alpha}(X)) \wedge \beta(X)$ is equivalent to $\mu X.\bar{\alpha}(X) \wedge \beta(X)$. It is obvious that

$$(\mu X.\bar{\alpha}(X) \wedge \beta(X)) \;\Rightarrow\; (\mu X.(X \vee \bar{\alpha}(X)) \wedge \beta(X))$$

Let $\gamma(X)$ stand for $\bar{\alpha}(X) \wedge \beta(X)$. To prove the other implication it is enough to observe that $\mu X.\gamma(X)$ is a prefixpoint of $\mu X.(X \vee \bar{\alpha}(X)) \vee \beta(X)$ as the following calculation shows:

$$\begin{array}{rl} ((\mu X.\gamma(X)) \vee \bar{\alpha}(\mu X.\gamma(X))) \wedge \beta(\mu X.\gamma(X)) & \Rightarrow \\ ((\bar{\alpha}(\mu X.\gamma(X)) \wedge \beta(\mu X.\gamma(X))) \vee \bar{\alpha}(\mu X.\gamma(X))) \wedge \beta(\mu X.\gamma(X)) & \Rightarrow \\ \bar{\alpha}(\mu X.\gamma(X) \wedge \beta(\mu X.\gamma(X)) & \end{array}$$

If φ is a guarded formula then we use dualities of the μ-calculus like $\neg[a]\alpha \equiv \langle a\rangle\neg\alpha$ or $\neg\mu X.\alpha(X) \equiv \nu X.\neg\alpha(\neg X)$ to produce an equivalent positive formula. It is easy to see that it will still be a guarded formula. $\square$

In our completeness proof we will need a result from [17] which gives a characterisation of the validity of the μ-calculus formulas by means of infinite tableaux. We will briefly recall the result here.

First we introduce the concept of a definition list [21] which will name the fixpoint subformulas of a given formula in order of their nesting.

We extend vocabulary of the μ-calculus by a countable set *Dcons* of fresh symbols that will be referred to as *definition constants* and usually denoted $U, V, \ldots$ These new symbols are now allowed to appear positively in formulas, like propositional variables.

A *definition list* is a finite sequence of equations :

$$\mathcal{D} = ((U_1 = \sigma_1 X.\alpha_1(X)), \ldots, (U_n = \sigma_n X.\alpha_n(X)) \tag{2}$$

where $U_1, \ldots, U_n \in DCons$ and $\sigma_i X.\alpha_i(X)$ is a formula such that all definition constants appearing in α_i are among $U_1, \ldots, U_{i-1}$. We assume that $U_i \neq U_j$ and $\alpha_i \neq \alpha_j$, for $i \neq j$. If $i < j$ then U_i is said to be *older* than U_j (U_j *younger* than U_i) with respect to the definition list $\mathcal{D}$.

We construct a definition list for a formula γ by means of the contraction operation $\rrparenthesis\gamma\llparenthesis$ which is defined recursively as follows:

1. $\rrparenthesis p\llparenthesis = \rrparenthesis\neg p\llparenthesis = \rrparenthesis X\llparenthesis = \rrparenthesis U\llparenthesis = \emptyset$;

2. $\rrparenthesis\neg\alpha\llparenthesis = \rrparenthesis\langle a\rangle\alpha\llparenthesis = \rrparenthesis[a]\alpha\llparenthesis = \rrparenthesis\alpha\llparenthesis$;

3. $\rrparenthesis\alpha \wedge \beta\llparenthesis = \rrparenthesis\alpha \vee \beta\llparenthesis = \rrparenthesis\alpha\llparenthesis \circ \rrparenthesis\beta\llparenthesis$, operation $\circ$ is defined below;

4. $\rrparenthesis\mu X.\alpha(X)\llparenthesis = ((U = \mu X.\alpha(X)), \rrparenthesis\alpha(U)\llparenthesis)$ where U is new;

5. $\rrparenthesis\nu X.\alpha(X)\llparenthesis = ((U = \nu X.\alpha(X)), \rrparenthesis\alpha(U)\llparenthesis)$ where U is new.

The operation $\rrparenthesis\alpha\llparenthesis \circ \rrparenthesis\beta\llparenthesis$ is defined as follows. First we make sure that the definition constants used in $\rrparenthesis\alpha\llparenthesis$ are disjoint form those used in $\rrparenthesis\beta\llparenthesis$. Then if it happens that $(U = \gamma) \in \rrparenthesis\alpha\llparenthesis$ and $(V = \gamma) \in \rrparenthesis\beta\llparenthesis$, we delete the definition from list $\rrparenthesis\beta\llparenthesis$ and replace V with U in $\rrparenthesis\beta\llparenthesis$. This may cause other formulas to be doubly defined and we deal with them in the same way.

We will say that U is a μ-constant if $(U = \mu X.\beta(X)) \in \mathcal{D}$, if $(U = \nu X.\beta(X)) \in \mathcal{D}$, constant U will be called a ν-constant. Observe that every constant occurring in $\mathcal{D}$ is either μ or ν-constant.

For a formula α and a definition list $\mathcal{D}$ containing all definition constants occurring in α we define the *expansion operation* $\langle\!\langle\alpha\rangle\!\rangle_{\mathcal{D}}$, which subsequently replaces definition constants appearing in the formula by the right hand sides of the defining equations,

$$\langle\!\langle\alpha\rangle\!\rangle_{\mathcal{D}} = [\sigma_n X.\alpha_n(X)/U_n]\dots[\sigma_1 X.\alpha_1(X)/U_1] \ , \quad \text{where } \mathcal{D} \text{ is as in (2)}$$

A *tableau sequent* is a pair $(\Gamma, \mathcal{D})$, where $\mathcal{D}$ is a definition list and Γ is a set of tableau formulas such that the only constants that occur in them are those from $\mathcal{D}$. We will denote $(\Gamma, \mathcal{D})$ by $\Gamma \vdash_{\mathcal{D}}$. A tableau sequent $\Gamma \vdash_{\mathcal{D}}$ will be called *tableau axiom* iff $p, \neg p \in \Gamma$ for some propositional letter or variable p.

Remark: We have two kinds of sequents. Ordinary ones are two sided sequents of formulas without definition constants. In a tableau sequent $\Gamma \vdash_{\mathcal{D}}$ formulas may contain definition constants from definition list $\mathcal{D}$. Index $\mathcal{D}$ also allows us to distinguish between an ordinary sequent $\Gamma \vdash$ which happens to have empty right hand side and a tableau sequent $\Gamma \vdash_{\mathcal{D}}$.

Definition 2.3 Let $\mathcal{S}$ be the following set of *tableau rules*:

$$(\wedge_t) \qquad \frac{\alpha, \beta, \Gamma \vdash_{\mathcal{D}}}{\alpha \wedge \beta, \Gamma \vdash_{\mathcal{D}}}$$

$$(\vee_t) \qquad \frac{\alpha, \Gamma \vdash_{\mathcal{D}} \qquad \beta, \Gamma \vdash_{\mathcal{D}}}{\alpha \vee \beta, \Gamma \vdash_{\mathcal{D}}}$$

$$(const) \qquad \frac{\alpha(U), \Gamma \vdash_{\mathcal{D}}}{U, \Gamma \vdash_{\mathcal{D}}} \quad (U = \sigma X.\alpha(X)) \in \mathcal{D}$$

$$(\sigma_t) \qquad \frac{U, \Gamma \vdash_{\mathcal{D}}}{\sigma X.\alpha(X), \Gamma \vdash_{\mathcal{D}}} \quad (U = \sigma X.\alpha(X)) \in \mathcal{D}$$

$$(\langle\rangle_t) \qquad \frac{\alpha, \{\beta : [a]\beta \in \Gamma\} \vdash_{\mathcal{D}}}{\langle a\rangle\alpha, \Gamma \vdash_{\mathcal{D}}}$$

Notice that if we assume that a tableau sequent $\Gamma \vdash_{\mathcal{D}}$ denotes ordinary sequent $\{\langle\!\langle\gamma\rangle\!\rangle_{\mathcal{D}} : \gamma \in \Gamma\} \vdash$ then tableau rules become sound logical rules.

Definition 2.4 Given a positive guarded formula γ and definition list $\mathcal{D} = \rangle\!\rangle\gamma\langle\!\langle$, a *tableau* for γ is any labeled tree $\langle K, L\rangle$, where K is a tree and L a labeling function, such that

1. the root of K is labeled with $\gamma \vdash_{\mathcal{D}}$,
2. if $L(n)$ is a tableau axiom then n is a leaf of K,
3. if $L(n)$ is not an axiom then the sons of n in K are created and labeled according to the rules of the system $\mathcal{S}$, i.e. in such a way that $L(n)$ is the conclusion and the labels of the sons assumptions of a rule from S.

Remark: We see applications of rules as a process of reduction. Given a finite set of formulas Γ we want to derive, we look for the rule the conclusion of which matches our set. Then we apply the rule and obtain the assumptions of the instance of the rule in which Γ is the conclusion.

Definition 2.5 Let $\mathcal{T} = \langle K, L\rangle$ be a tableau for a formula γ with $\mathcal{D} = [\![\gamma]\!]$. Let $P = (v_1, v_2, \ldots)$ be an infinite path in the tree K, i.e. each v_{i+1} is a son of v_i. A *trace* on the path P is any sequence of formulas $(\alpha_1, \alpha_2, \ldots)$ such that $\alpha_n \in L(v_n)$ and α_{n+1} is either: (i) α_n if the formula α_n is not reduced by the rule applied in v_n or (ii) if α_n is reduced in v_n then α_{n+1} is one of the resulting formulas.

This last notion should be clear for all the rules other then $(\langle\rangle_t)$. For the rule:

$$(\langle\rangle_t) \qquad \frac{\alpha, \{\beta : [a]\beta \in \Gamma\} \vdash_{\mathcal{D}}}{\langle a\rangle\alpha, \Gamma \vdash_{\mathcal{D}}}$$

we have: $\alpha_{n+1} = \alpha$ if $\alpha_n = \langle a\rangle\alpha$, and $\alpha_{n+1} = \beta$ if $\alpha_n = [a]\beta$ for some β; in all other cases the trace ends on α_n.

Definition 2.6 A constant U *regenerates* on the trace $(\alpha_1, \alpha_2, \ldots)$ if for some i, $\alpha_i = U$ and $\alpha_{i+1} = \alpha(U)$, where $(U = \sigma X.\alpha(X)) \in \mathcal{D}$. A trace is called *$\mu$-trace* iff it is an infinite trace on which the oldest constant regenerated infinitely often is a μ-constant.

Definition 2.7 A tableau $\mathcal{T}$ for γ is called a *refutation* of γ iff every leaf of $\mathcal{T}$ is labeled with a tableau axiom and on every infinite path of $\mathcal{T}$ there exists a μ-trace.

Theorem 2.8 (Characterisation) *For every positive guarded formula γ: γ is not satisfiable iff there is a refutation of γ*

3 Axiomatisation

This section is divided into three parts. In the first subsection we present a finitary axiom system for the μ-calculus. Next we prove some basic properties of the system. In the third subsection we relate the axiomatisation to the one proposed by Kozen in [12].

3.1 The system

We will present the system in a sequent calculus form. We prefer this formalisation because sequent calculus rules closely correspond to the tableau rules.

It will be convenient to introduce one piece of notation. For a finite set of atomic actions $P = \{a_1, \ldots, a_n\} \subseteq Act$ and a μ-calculus formula α we let $\langle P^*\rangle\alpha$ to be an abbreviation of the formula $\mu X.\langle a_1\rangle X \vee \ldots \vee \langle a_n\rangle X \vee \alpha$. Intuitively such a formula says that a state satisfying α is reachable by sequence of actions from P.

Our system consists of two groups of rules and some additional axioms. First we take the rules of the simple propositional modal logic.

$$(\neg) \qquad \frac{\Gamma \vdash \Delta, \alpha}{\Gamma, \neg\alpha \vdash \Delta} \qquad \frac{\Gamma, \alpha \vdash \Delta}{\Gamma \vdash \neg\alpha, \Delta}$$

$$(\wedge) \qquad \frac{\alpha, \beta, \Gamma \vdash \Delta}{\alpha \wedge \beta, \Gamma \vdash \Delta} \qquad \frac{\Gamma \vdash \alpha, \Delta \quad \Gamma \vdash \beta, \Delta}{\Gamma \vdash \alpha \wedge \beta, \Delta}$$

$$(\vee) \qquad \frac{\alpha, \Gamma \vdash \Delta \quad \beta, \Gamma \vdash \Delta}{\alpha \vee \beta, \Gamma \vdash \Delta} \qquad \frac{\Gamma \vdash \alpha, \beta, \Delta}{\Gamma \vdash \alpha \vee \beta, \Delta}$$

$$(\langle\rangle) \qquad \frac{\alpha, \{\beta : [a]\beta \in \Gamma\} \vdash \{\gamma : \langle a\rangle\gamma \in \Delta\}}{\langle a\rangle\alpha, \Gamma \vdash \Delta}$$

$$(cut) \qquad \frac{\Gamma \vdash \Delta, \gamma \quad \Sigma, \gamma \vdash \Omega}{\Gamma, \Sigma \vdash \Delta, \Omega}$$

Then we add the rules concerning fixpoints

$$(\mu) \qquad \frac{\Gamma \vdash \alpha(\mu X.\alpha(X)), \Delta}{\Gamma \vdash \mu X.\alpha(X), \Delta}$$

$$(ind) \qquad \frac{\varphi(\mathit{ff}) \vdash \Delta \quad \varphi(\alpha(\mu X.Z \wedge \alpha(X))) \vdash \Delta, \langle P^*\rangle\varphi(\mu X.Z \wedge \alpha(X))}{\varphi(\mu X.\alpha(X)) \vdash \langle P^*\rangle\Delta}$$

$$Z \notin FV(\varphi(\mu X.\alpha(X)), \Delta)$$

In the last rule $P \subseteq Act$ is a finite set of actions, $\langle P^*\rangle\Delta$ is an abbreviation of $\{\langle P^*\rangle\delta : \delta \in \Delta\}$ and Z is a new propositional variable not occurring free in $\varphi(\mu X.\alpha(X)) \vdash \Delta$.

In the above notation $\varphi(\mu X.\alpha(X))$ stands for the result of the substitution $\varphi[\mu X.\alpha(X)/[]]$ where $[]$ is a distinguished variable. A substitution $\varphi[\alpha/X]$ is legal only if no free variable of α becomes bound in $\varphi[\alpha/X]$.

Finally because we had chosen to include constructions $[a]\alpha$ and $\nu X.\alpha(X)$ into the language we have to define them using other connectives by adding the following sequents as specific axioms:

$$([]L) \qquad [a]\alpha \vdash \neg\langle a\rangle\neg\alpha \qquad ([]R) \qquad \langle a\rangle\alpha \vdash \neg[a]\neg\alpha$$

$$(\nu L) \quad \nu X.\alpha(X) \vdash \neg\mu X.\neg\alpha(\neg X) \quad (\nu R) \quad \mu X.\alpha(X) \vdash \neg\nu X.\neg\alpha(\neg X)$$

Definition 3.1 A finite tree constructed with the use of the above rules will be called *diagram*. A *proof* will be a diagram whose all leaves are labeled with *axioms*, i.e., instances of the axiom sequents above or sequents with the same formula on both sides. ■

Remark: Observe the difference between diagrams and tableaux. The first are always finite while the second not necessary so. They are also constructed using different, although similar, sets of rules. ■

Let us give some intuitions about rule (ind). Especially we would like to hint what well ordered set we are interested in and what kind of induction stands behind the rule. First let us consider the following rule:

$$\frac{\varphi(\mathit{ff}) \vdash \Delta \quad \varphi(\alpha^{n+1}(\mathit{ff})) \vdash \Delta, \langle P^* \rangle \varphi(\alpha^n(\mathit{ff}))}{\varphi(\mu X.\alpha(X)) \vdash \langle P^* \rangle \Delta} \tag{3}$$

where n ranges over the natural numbers and $\alpha^n(\mathit{ff})$ stands for n-unfolding: $\alpha(\ldots\alpha(\mathit{ff})\ldots)$. Of course because n is treated as a variable, this rule is not a finitary rule of the μ-calculus.

Suppose that the assumptions of rule (3) are valid in a finite structure $\mathcal{M}$. We would like to show that the conclusion of the rule is valid. This follows from two observations. First using the assumptions and induction on n we can show that $\varphi(\alpha^n(\mathit{ff})) \vdash \langle P^* \rangle \Delta$ is valid for any n. Then it is enough to observe that in a finite structure $\mathcal{M}$ with, say k, states, formula $\mu X.\alpha(X)$ is equivalent to $\alpha^k(\mathit{ff})$.

Rule (3) would be sufficient for our purposes but it is not a rule of the μ-calculus. We can easily weaken the rule to:

$$\frac{\varphi(\mathit{ff}) \vdash \Delta \quad \varphi(\alpha(Z)) \vdash \Delta, \langle P^* \rangle \varphi(Z)}{\varphi(\mu X.\alpha(X)) \vdash \langle P^* \rangle \Delta} \tag{4}$$

where Z is a fresh variable. This rule is much weaker because the second assumption is much stronger. In (3) we required the property to hold only for the approximations of $\mu X.\alpha(X)$ in present rule we want it to be true for all sets of states. We don't know whether the system with this rule is complete. Our rule (ind) is the stronger version of (4) which is still a finitary rule of the μ-calculus. Instead of taking just free variable Z we take $\mu X.Z \wedge \alpha(X)$. The following fact shows that for Z being a finite approximation of $\mu X.\alpha(X)$ the two formulas are equivalent. Nevertheless in general the assumption of (ind) rule is much weaker than that of rule (4)

Fact 3.2 For any structure $\mathcal{M}$, formula $\mu X.\alpha(X)$ and valuation Val such that $\mathit{Val}(Z) = \| \alpha^k(\mathit{ff}) \|_{Val}$ for some $k \in \mathcal{N}$ and $Z \notin FV(\alpha(\mathit{ff}))$ we have:

$$\| \mu X.Z \wedge \alpha(X) \|_{Val} = \mathit{Val}(Z)$$

Proof: Let $\mathit{Val}(Z) = \| \alpha^k(\mathit{ff}) \|_{Val}$ then

$$\| \mu X.Z \wedge \alpha(X) \|_{Val} = \| \mu X.\alpha^k(\mathit{ff}) \wedge \alpha(X) \|_{Val} = \| \alpha^k(\mathit{ff}) \|_{Val}$$

$\square$

Inclusion of a cut rule into our system also deserves a comment. Since the rule (ind) has the specific shape of reducing connective which can be "hidden" deep into the formula, the addition of the cut rule seems to be a reasonable way to make the system usable. The completeness proof will give us an algorithm for constructing a proof of a valid formula and as we will see this proof will have some kind of subformula property even though cuts will be used in it. The conclusion is that cuts in the proof can be used in a very restricted way and one can substitute them by appropriate rules which may be added to the system. Addition of the cut rule makes the system simpler and since we have an algorithm for automatic construction of a proof of a valid sequent we prefer the presented formulation.

The next proposition states soundness of the system. There are at least three possible notions of soundness of a rule in a modal logic:

- The strongest will be to require that if the assumptions of a rule are satisfied in a state of a structure with a given valuation then the conclusion must be satisfied in this state and valuation. It is easy to see that rules (neg), (and), (or), (cut) and (μ) are sound in this sense.
- The other would be to require that for any structure $\mathcal{M}$, if all assumptions of the rule are valid in the structure (i.e. satisfied in every state and every valuation) then the conclusion must be valid in this structure. The rule $(\langle\rangle)$ is sound in this sense. We will show in the next subsection that all the rules of Kozen's axiomatisation are also sound in this sense.
- The weakest of the three is the condition that if all the assumptions of the rule are valid then the conclusion is valid. The rule (ind) is sound in this sense and as we will show in the next subsection it is not sound in the sense of any of the two preceding notions of soundness. Of course this type of soundness is all that we need for an axiomatisation of the validity relation. In the following by soundness we will always understand this weak notion of soundness.

Proposition 3.3 All the rules of our system are sound and all the axioms are valid

Proof
Validity of the axioms is standard. Also standard is validity of all the rules except (ind) which soundness we prove below.

Let us assume conversely that the rule (ind) is not sound. Then from the finite model property we have a finite structure $\mathcal{M}$ such that the sequents:

$$\varphi(\mathit{ff}) \vdash \Delta \quad \varphi(\alpha(\mu X.Z \wedge \alpha(X))) \vdash \Delta, \langle P^* \rangle \varphi(\mu X.Z \wedge \alpha(X))$$

are valid in $\mathcal{M}$ and $\varphi(\mu X.\alpha(X)) \vdash \langle P^* \rangle \Delta$ is not. We assume that the variable Z does not appear free in $\varphi(\mu X.\alpha(X))$ or Δ. Let k be the smallest integer for which there exists a state s of $\mathcal{M}$ and valuation Val satisfying: $\mathcal{M}, s, Val \models \varphi(\alpha^k(\mathit{ff}))$ and for all $\delta \in \Delta$, $\mathcal{M}, s, Val \not\models \langle P^* \rangle \delta$.

We have two cases depending on whether $k = 0$ or not.

If $k = 0$ then $\mathcal{M}, s, Val \models \varphi(\mathit{ff})$ hence from the assumption that $\varphi(\mathit{ff}) \vdash \Delta$ is valid in $\mathcal{M}$ we have that $\mathcal{M}, s, Val \models \delta$ for some $\delta \in \Delta$, contradiction.

If $k > 0$ then let Val' be identical to Val except that for the variable Z and let $Val'(Z) = \| \alpha^{k-1}(\mathit{ff}) \|_{Val}$. Hence $\mathcal{M}, s, Val' \models \varphi(\alpha(Z))$ and from Fact 3.2 it follows that $\mathcal{M}, s, Val' \models \varphi(\alpha(\mu X.Z \wedge \alpha(X)))$. From the validity of the second premise we have that either: $\mathcal{M}, t, Val' \models \varphi(\mu X.Z \wedge \alpha(X))$ for some state t

reachable from s by P, or $\mathcal{M}, s, Val \models \delta$ for some $\delta \in \Delta$. The second case is impossible by assumption.

If $\mathcal{M}, t, Val' \models \varphi(\mu X.Z \wedge \alpha(X))$ then by Fact 3.2 $\mathcal{M}, t, Val' \models \varphi(Z)$, hence $\mathcal{M}, t, Val \models \varphi(\alpha^{k-1}(ff))$. By the choice of k there must exist $\delta \in \Delta$ s.t. $\mathcal{M}, t, Val \models \langle P^* \rangle \delta$. But then because t is reachable from s by a sequence of actions from P we also have that $\mathcal{M}, s, Val \models \langle P^* \rangle \delta$. Contradiction. $\square$

3.2 Some properties of the system

In this subsection we would like to prove some properties of the presented system. First we show that a substitution of equals for equals is admissible.

Fact 3.4 Let $\psi(X_1, \ldots, X_n)$ be a formula with variables $X_1, \ldots, X_n$ occurring only positively in ψ. If for any $i = 1, \ldots, n$ a sequent $\delta_i \vdash \gamma_i$ is provable then the sequent $\psi(\delta_1, \ldots, \delta_n) \vdash \psi(\gamma_1, \ldots, \gamma_n)$ is provable.

This will follow from

Lemma 3.5 Let $\vec{X}$ and $\vec{Y}$ denote the sequences of variables $X_1, \ldots, X_n$ and $Y_1, \ldots, Y_n$ respectively. Let $\varphi(\vec{X})$ be a formula with free variables among $\vec{X}$ each of them occurring only positively or only negatively in it. Let $P = \{a_1, \ldots, a_i\}$ be the set of all the actions occurring in $\varphi(\vec{X})$. For any finite set of formulas Δ there is a diagram of $\varphi(\vec{X}) \vdash \varphi(\vec{Y}), \langle P^* \rangle \Delta$ in which the only leaves which are not axioms are of the form $X_i \vdash Y_i, \langle P^* \rangle \Delta$, if X_i occurs only positively, and $Y_i \vdash X_i, \langle P^* \rangle \Delta$, if X_i occurs only negatively in $\varphi(X_1, \ldots, X_n)$.

Proof
We proceed by induction on the structure of φ.

— if φ is one of the free variables or some propositional constant then the lemma is obvious.

— if $\varphi = \neg\psi$ then the application of the derivable rule

$$\frac{\psi(\vec{Y}) \vdash \psi(\vec{X}), \langle P^* \rangle \Delta}{\neg\psi(\vec{X}) \vdash \neg\psi(\vec{Y}), \langle P^* \rangle \Delta}$$

gives us the desired conclusion.

— if $\varphi = \psi_1 \wedge \psi_2$ then we use the derivable rule

$$\frac{\psi_1(\vec{X}) \vdash \psi_1(\vec{Y}), \langle P^* \rangle \Delta \quad \psi_2(\vec{X}) \vdash \psi_2(\vec{Y}), \langle P^* \rangle \Delta}{\psi_1(\vec{X}) \wedge \psi_2(\vec{X}) \vdash \psi_1(\vec{Y}) \wedge \psi_2(\vec{Y}), \langle P^* \rangle \Delta}$$

— if $\varphi = \langle a \rangle \psi$ then the rule to use is $(\langle\rangle)$

$$\frac{\psi(\vec{X}) \vdash \psi(\vec{Y}), \langle P^* \rangle \Delta}{\langle a \rangle \psi(\vec{X}) \vdash \langle a \rangle \psi(\vec{Y}), \langle P^* \rangle \Delta}$$

— if $\varphi = \mu V.\beta(V)$ then we use rule (ind) to the sequent

$$\mu V.\beta(V,\vec{X}), \nu V.\neg\beta(\neg V,\vec{Y}) \vdash \langle P^*\rangle\Delta \tag{5}$$

from which the sequent

$$\mu V.\beta(V,\vec{X}) \vdash \mu V.\beta(V,\vec{Y}), \langle P^*\rangle\Delta$$

can be obtained by a (cut) with an axiom. The sequents:

$$ff, \nu V.\neg\beta(\neg V,\vec{Y}) \vdash \langle P^*\rangle\Delta \tag{6}$$

$$\beta(\mu V.Z \wedge \beta(V,\vec{X}),\vec{X}), \nu V.\neg\beta(\neg V,\vec{Y}) \vdash \\ \langle P^*\rangle((\mu V.Z \wedge \beta(V,\vec{X})) \wedge \nu V.\neg\beta(\neg V,\vec{Y})), \langle P^*\rangle\Delta \tag{7}$$

are assumptions of the instance of (ind) rule in which (5) is the conclusion.

Clearly (6) is provable so it has a diagram with all the leaves labeled by axioms. Sequent (7) can be obtained by cut rule from an axiom and the sequent:

$$\beta(\mu V.Z \wedge \beta(V,\vec{X}),\vec{X}) \vdash \\ \beta(\mu V.\beta(V,\vec{Y}),\vec{Y}), \langle P^*\rangle((\mu V.Z \wedge \beta(V,\vec{X})) \wedge \nu V.\neg\beta(\neg V,\vec{Y})), \langle P^*\rangle\Delta \tag{8}$$

By induction hypothesis there is a diagram for (8) which leaves are labeled by axioms or sequents of one of the forms:

$$X_i \vdash Y_i, \langle P^*\rangle((\mu V.Z \wedge \beta(V,\vec{X})) \wedge \nu V.\neg\beta(\neg V,\vec{Y})), \langle P^*\rangle\Delta \tag{9}$$

$$Y_i \vdash X_i, \langle P^*\rangle((\mu V.Z \wedge \beta(V,\vec{X})) \wedge \nu V.\neg\beta(\neg V,\vec{Y})), \langle P^*\rangle\Delta \tag{10}$$

$$\mu V.Z \wedge \beta(V,\vec{X}) \vdash \\ \mu V.\beta(V,\vec{Y}), \langle P^*\rangle((\mu V.Z \wedge \beta(V,\vec{X})) \wedge \nu V.\neg\beta(\neg V,\vec{Y})), \langle P^*\rangle\Delta \tag{11}$$

Sequents of the form (9) and (10) can be put into the desired form after application of weakening to the right side. Sequent (11) is clearly provable hence it has a diagram with all the leaves labeled by axioms.

— in cases when $\varphi = \psi_1 \vee \psi_2$, $\varphi = [a]\psi$ or $\varphi = \nu V.\beta(V)$ we can just use the fact that these connectives are defined by the connectives we have already considered.

$\Box$

The next proposition allows us to restrict attention to positive guarded formulas.

Proposition 3.6 Any formula is provably equivalent to some positive guarded formula

Proof
The proof is a repetition of the arguments from Proposition 2.2. Fact 3.4 allows us to show that all the steps use provable equivalences. One can show that $\mu X.(X \vee \bar{\alpha}(X)) \wedge \beta(X)$ is provably equivalent to $\mu X.\bar{\alpha}(X) \wedge \beta(X)$ using Lemma 3.5. $\Box$

3.3 Comparison with Kozen's axiomatisation

Here we would like to relate presented axiomatisation to Kozen's axiom system from [12]. Let us call this system $K\mu$ here.

The system $K\mu$ uses a different form of judgement. It has the form of equality $\varphi = \psi$ with the meaning that the formulas φ and ψ are semantically equivalent. Inequality $\varphi \leq \psi$ is considered as an abbreviation of $\varphi \vee \psi = \psi$.

Apart from the axioms and rules of equational logic (including substitution of equals by equals) there are the following axioms and rules:

$$
\begin{array}{ll}
(K1) & \text{axioms for Boolean algebra} \\[1em]
(K2) & \langle a\rangle\varphi \vee \langle a\rangle\psi = \langle a\rangle(\varphi \vee \psi) \\[1em]
(K3) & \langle a\rangle\varphi \wedge [a]\psi \leq \langle a\rangle(\varphi \wedge \psi) \\[1em]
(K4) & \langle a\rangle\mathit{ff} = \mathit{ff} \\[1em]
(K5) & \alpha(\mu X.\alpha(X)) \leq \mu X.\alpha(X) \\[1em]
(K6) & \dfrac{\alpha(\varphi) \leq \varphi}{\mu X.\alpha(X) \leq \varphi}
\end{array}
$$

Our sequent $\Gamma \vdash \Delta$ corresponds to inequality $\bigwedge \Gamma \leq \bigvee \Delta$ in this notation. We will call a rule *derivable* in system $K\mu$ iff there is a way to derive the conclusion of the rule, assuming all the premises of the rule.

Proposition 3.7 Every rule derivable in the system $K\mu$ must have the property: for any structure $\mathcal{M}$, if all the premises of the rule are valid in $\mathcal{M}$ (i.e. true in every state of $\mathcal{M}$) then the conclusion is valid in $\mathcal{M}$

The proposition follows directly from the observation that all the rules of $K\mu$ have this property. We will show that the rule (ind) of our system does not have this property hence it cannot be derived in $K\mu$.

Proposition 3.8 There is a structure $\mathcal{M}$ and an instance of (ind) rule such that the premises are true in $\mathcal{M}$ but the conclusion is not.

Proof
Consider a structure $\mathcal{M} = \langle S, R, \rho\rangle$ such that:

- $S = \mathcal{N} \cup \{\infty\}$
- $R(a) = \{(i+1, i) : i \in \mathcal{N}\} \cup \{(\infty, i) : i \in \mathcal{N}\}$
- $\rho(p) = \{\infty\}$

Here a is some action, p is some propositional constant and $\mathcal{N}$ is the set of natural numbers.

Consider a sequent:

$$p \wedge [a]\mu X.[a]X \vdash$$

which is not valid in $\mathcal{M}$ because $\mathcal{M}, \infty \models p \wedge [a]\mu X.[a]X$. If we were to prove this sequent we could use rule (ind) directly and have the premises:

$$p, [a]ff \vdash \tag{12}$$

$$p, [a]([a]\mu X.Z \wedge [a]X) \vdash p \wedge [a]\mu X.Z \wedge [a]X \tag{13}$$

Clearly sequent (12) is true in $\mathcal{M}$. To see why sequent (13) is true in $\mathcal{M}$ suppose that for some valuation Val we have $\mathcal{M}, \infty, Val \models p \wedge [a][a](\mu X.Z \wedge [a]X)$ then clearly $\mathcal{N} \subseteq Val(Z)$ hence also by Fact 3.2, $\mathcal{N} \subseteq \| \mu X.Z \wedge \alpha(X) \|_{Val}$ which means that $\mathcal{M}, \infty, Val \models p \wedge [a](\mu X.Z \wedge [a]X)$.

$\square$

Of course Proposition 3.8 does not imply incompleteness of the system $K\mu$ even though our system is strictly stronger than $K\mu$ as the following proposition shows.

Proposition 3.9 All the axioms of the system $K\mu$ are provable in our system. All the rules of $K\mu$ are derivable in our system

Proof
Provability of the axioms is easy, so is also deriveability of all the rules other then $(K6)$ and substitution of equals by equals. This second rule is derivable by Fact 3.4.

For rule $(K6)$ let us assume $\alpha(\varphi) \vdash \varphi$. We will show how to prove $\neg\varphi \wedge \mu X.\alpha(X) \vdash$. Let $P = \{a_1, \ldots, a_i\}$ be the set of all the actions occurring in $\alpha(\varphi)$.

We can obtain $\neg\varphi \wedge \mu X.\alpha(X) \vdash$ from (ind) rule if we prove

$$ff \wedge \neg\varphi \vdash \quad \text{and} \quad \neg\varphi \wedge \alpha(\mu X.Z \wedge \alpha(X)) \vdash \langle P^* \rangle \neg\varphi \wedge \mu X.Z \wedge \alpha(X)$$

The first sequent is clearly provable. Using assumption that $\alpha(\varphi) \vdash \varphi$ we will have a proof of the second sequent if we only prove:

$$\neg\alpha(\varphi) \wedge \alpha(\mu X.Z \wedge \alpha(X)) \vdash \langle P^* \rangle \neg\varphi \wedge \mu X.Z \wedge \alpha(X)$$

This in turn is equivalent to proving

$$\alpha(\mu X.Z \wedge \alpha(X)) \vdash \alpha(\varphi), \langle P^* \rangle \neg\varphi \wedge \mu X.Z \wedge \alpha(X)$$

By Lemma 3.5 there is a diagram for this sequent in which the only leaves which are not axioms are of the form

$$\mu X.Z \wedge \alpha(X) \vdash \varphi, \langle P^* \rangle \neg\varphi \wedge \mu X.Z \wedge \alpha(X)$$

But such sequent is easily provable.

$\square$

4 Completeness

In this section we will prove completeness of our system. Let us fix an unsatisfiable positive guarded formula φ_0 of the μ-calculus. We also fix $\mathcal{D}_{\varphi_0}$ to denote the definition list $\rrbracket\varphi_0\llbracket = (W_1 = \sigma X.\gamma_1(X))\dots(W_d = \sigma X.\gamma_d(X))$. We will consider only definition constants from $\mathcal{D}_{\varphi_0}$ and when we will say that a definition constant is older (younger) than the other we will mean that it is older (younger) with respect to the definition list $\mathcal{D}_{\varphi_0}$. Our goal in this section will be to construct a proof of the sequent $\varphi_0\vdash$ in our system.

4.1 Overview of the proof method

Let us now give an overview of the proof method we will use. We would like to show that it is an extension of a very natural method of proving completeness for system K of modal logic.

Let us consider only positive formulas of simple modal logic, that is positive formulas of the μ-calculus without fixpoint operators. We consider sequents of such formulas of the form $\Gamma\vdash$. The interpretation of such a sequent is standard, i.e., that the conjunction of formulas in Γ is not satisfiable. For such sequents of this simple logic the system consisting of one side versions of $(\wedge)$, $(\vee)$ and $(\langle\rangle)$ rules is sufficient. This are exactly rules $(\wedge_t)$, $(\vee_t)$ and $(\langle\rangle_t)$ of our tableau system as there are no definition constants in this case.

A *proof* of a sequent $\Gamma\vdash$ in this system is a diagram obtained using the rules mentioned above with the root labeled by $\Gamma\vdash$ and all the leaves labeled by sequents containing some propositional letter and its negation.

For completeness proof we will use tableaux which will be constructed with the above rules except $(\langle\rangle_t)$-rule which is replaced by:

$$(all\langle\rangle) \quad \frac{\{\alpha, \{\beta : [a]\beta \in \Gamma\}\vdash : \langle a\rangle\alpha \in \Gamma\}}{\Gamma\vdash}$$

This rule has as many assumptions as the number of formulas of the form $\langle a\rangle\alpha$ in Γ. When there are no such formulas at all then we cannot apply this rule. Rule $(all\langle\rangle)$ can be seen as a very weak logical rule but it is probably better not to look at it this way. It is rather different type of rule, because it is enough to prove only one of the assumptions of the rule to prove the conclusion, while for other rules we have to prove all the assumptions. This distinction will be elaborated below.

Given a sequent $\Gamma\vdash$ of positive formulas, we construct a tableau for it in this tableau system. We would like tableaux to be maximal in a sense that rule $(all\langle\rangle)$ is used only to sequents $\Sigma\vdash$ such that each formula in Σ is either a propositional constant, its negation or a formula of the form $\langle b\rangle\beta$ or $[b]\beta$ for some action b and a formula β. A tableau constructed in this way is obviously finite. Some of the leaves are axioms and others are unreducible sequents which are not axioms.

The general property of such a tableau for $\Gamma\vdash$ is that one can either find a proof of $\Gamma\vdash$ in the tableau or one can read a finite model for the formula $\bigwedge\Gamma$ from it. To see this let us apply the following simple marking procedure.

We mark all the leaves labeled with axiom sequents with 1 and all other leaves with 0. Then if in a node of the tableau rule $(\wedge_t)$ was used we mark it by the same number as its only son. If $(\vee_t)$ was used in a node and both sons are labeled by 1 then we mark the node by 1 otherwise we mark it by 0. For each node where $(all\langle\rangle)$ was used we mark it with 0 if all the sons of it are marked with 0, otherwise we mark it with 1. It should be obvious that if the root of the tableau is marked with 1 then there is a proof of $\Gamma \vdash$ in the tableau. If the root of the tableau is labeled by 0 then we can find a model for $\bigwedge \Gamma$ in the tableau.

When we try to apply the above method to the full μ-calculus we meet one complication. It seems that the only obvious rules we can add for dealing with fixpoints are unwinding rules of the form:

$$\frac{\alpha(\mu X.\alpha(X)), \Gamma \vdash}{\mu X.\alpha(X), \Gamma \vdash}$$

But tableaux constructed with this rule may not be finite and finiteness of tableaux was crucial for the above completeness proof.

In case of the full μ-calculus Theorem 2.8 takes the place of the marking procedure. In system K for any unsatisfiable formula the marking procedure gave us a finite tree designated by 1's. By Theorem 2.8 any unsatisfiable formula of the μ-calculus has a refutation. Refutation is very similar to the tree obtained from the marking procedure except that it may have infinite paths. While marking procedure gave us a proof right away here we will have to work hard to transform a refutation into a proof which must be a finite tree.

The difficulty we face is as follows. On any infinite path of a refutation for φ_0 there is a μ-trace, i.e., there is an occurrence of a μ-formula which regenerates i.o. and never disappears. It was already shown by Kozen [12] how to use some derivable rules of his system in such a clever way that it is possible to "cut" such a path, i.e., arrive at an axiom sequent after a finite number of steps. We can of course apply such a cutting strategy to each path of a refutation. The problem is that we will obtain a set of finite paths, each of them ending with an axiom, but usually it would be impossible to compose them back to a tree.

There is a similar problem in automata theory, when one wants to construct a tree automaton recognising trees the paths of which are accepted by a nondeterministic Büchi automaton. The solution there is to determinize the automaton first. In our case it means that we need a deterministic strategy of transforming a path of a refutation into a path of a proof. Here deterministic means that the actions we perform in a node depend only on the predecessors of the node and not on the whole path.

First step of the proof is construction of a deterministic Rabin automaton on infinite words $\mathcal{A}_{\varphi_0}$ recognising paths of a tableau for φ_0 with μ-trace on them. Later states of $\mathcal{A}_{\varphi_0}$ will be used to guide our converting procedure hence the simpler automaton we construct the simpler will be our procedure. This is why we do not use any standard determinization constructions but define $\mathcal{A}_{\varphi_0}$ from the scratch. The construction is nevertheless based on Safra's determinization [20].

From $\mathcal{A}_{\varphi_0}$ we construct an appropriate Rabin automaton over one letter alphabet $\mathcal{TA}_{\varphi_0}$ whose runs closely correspond to refutations of φ_0. This allows us to conclude that there is a small graph $\mathcal{G}_{\varphi_0}$, with states of $\mathcal{TA}_{\varphi_0}$ as nodes, which unwinds to an accepting run of $\mathcal{TA}_{\varphi_0}$.

Next step is to examine the structure of $\mathcal{G}_{\varphi_0}$. It turns out that in this graph special nodes, which we call *loop nodes*, can be distinguished. This nodes can be seen as "witnesses" that unwinding of $\mathcal{G}_{\varphi_0}$ is accepted by the automaton. On every cycle there is exactly one loop node which "confirms" that the unwinding of the cycle is accepted by $\mathcal{A}_{\varphi_0}$. Another important thing is that there is a natural partial order on loop nodes. In this way we can put some order of "importance" on different loops we have in $\mathcal{G}_{\varphi_0}$.

We use this ordering to unwind $\mathcal{G}_{\varphi_0}$ into T_{φ_0} which is a finite tree with back edges, i.e., edges from some of the leaves to their ancestors. The last step is construction of a *sound* sequent assignment for T_{φ_0}; that is assign $\varphi_0 \vdash$ to the root, assign an easily provable sequent to every leaf of T_{φ_0}, and for any other node assign a sequent which is provable from the sequents assigned to its sons.

The rest of this section is organised as follows. In the next subsection we describe the construction of the automaton $\mathcal{A}_{\varphi_0}$. Subsection 4.3 is devoted to the construction of T_{φ_0}. Finally we define a sound sequent assignment for T_{φ_0}.

4.2 Automaton

Our goal here is to construct a special Rabin automaton $\mathcal{TA}_{\varphi_0}$ on trees over one letter alphabet, which accepting runs closely correspond to the refutations of φ_0. We do this by constructing a special deterministic Rabin automaton $\mathcal{A}_{\varphi_0}$ on paths, which recognises exactly the paths of a tableau for φ_0 with μ-trace on them.

Let us first observe that the set of formulas which can appear in a refutation for φ_0 is finite. Let us call this set $FL(\varphi_0)$ as it is almost Fisher-Ladner closure [7] of φ_0. It is not exactly the closure because we have definition constants around.

Definition 4.1 For any formula φ, possibly with definition constants, we define the FL-closure of φ, $FL(\varphi)$ as the set of all formulas which can occur in a sequent $\Sigma \vdash_{\mathcal{D}_{\varphi_0}}$ obtainable from $\varphi \vdash_{\mathcal{D}_{\varphi_0}}$ by application of some number of the tableau rules. ∎

Lemma 4.2 For any formula φ the size of $FL(\varphi)$ is linear in the size of $\langle\!\langle \varphi \rangle\!\rangle_{\mathcal{D}_{\varphi_0}}$, which is φ with all definition constants replaced by appropriate formulas.

There is of course simple nondeterministic Büchi automaton recognising the paths with a μ-trace. This automaton goes along a path and picks one formula from each tableau sequent in such a way that the sequence of chosen formulas forms a trace. Acceptance conditions are such that this chosen trace is accepted iff it is a μ-trace. Obviously this automaton is nondeterministic because of the tableau rule (and). If we have chosen formula $\alpha \wedge \beta$ in a sequent $\alpha \wedge \beta, \Gamma \vdash_{\mathcal{D}}$ then after application of rule (and) we obtain a sequent $\alpha, \beta, \Gamma \vdash_{\mathcal{D}}$ and we may choose either α or β as the next formula in the trace being constructed.

Instead of determinizing this Büchi automaton we will construct $\mathcal{A}_{\varphi_0}$ from the scratch. This construction is an adaptation of the Safra's determinization construction to the special case we are to deal with. We have to adapt the construction to avoid some redundancies which we would get if we applied Safra's construction directly.

As in Safra's construction states of our automaton will be labeled trees. Since we will quickly arrive at trees labeled with states which are itself trees, we will always call nodes of states *vertices* and nodes of trees of states just *nodes*.

The idea of Safra's construction was to construct for a given nondeterministic automaton a deterministic automaton which, for an infinite word σ, calculates during the run the set of states S which has the special property that there are positions $k_1, k_2, \ldots$ such that:

- every state from S is reachable on the word $\sigma[1, k_1]$, ($\sigma[i, j]$ stands for the part of σ between positions i and j),
- for any $i = 1, 2, \ldots$ and any state $s \in S$ there is a state $s' \in S$ such that the automaton started in s' will reach s on word $\sigma[k_i, k_{i+1}]$ and go through a green state on the way.

Obviously if there is such a nonempty set of states then there is an accepting run of the original automaton. It turns out that the converse also holds.

We will use the same idea in our case. We will design an automaton which given a path of a tableau ($\Gamma_1 \vdash_{\mathcal{D}_{\varphi_0}}, \Gamma_2 \vdash_{\mathcal{D}_{\varphi_0}}, \ldots$) will calculate a set of formulas Ω and a μ-constant U from $\mathcal{D}_{\varphi_0}$ such that there are integers $k_1, k_2, \ldots$ with the property that:

- $\Omega \subseteq \Gamma_{k_1}$,
- for any $i = 1, 2, \ldots$ and any $\alpha \in \Omega$ there is $\beta \in \Omega$ and a trace from occurrence of β in Γ_{k_i} to the occurrence α in $\Gamma_{k_{i+1}}$ on which U is the oldest regenerated constant.

We are now going to construct such an automaton $\mathcal{A}_{\varphi_0}$ for a formula φ_0. Its states will be trees labeled with formulas. As we have to calculate definition constant we introduce also labeling of the edges with definition constants.

States of the automaton $\mathcal{A}_{\varphi_0}$ will be labeled ordered trees, i.e., tuples $T = \langle N, r, p, \prec, nl, el \rangle$ where:

- N is a set of vertices,
- $r \in N$ is a root of the tree,
- $p : (N \setminus \{r\}) \to N$ is a parenthood function,
- $\prec$ is a sibling ordering, i.e., partial ordering relation which linearly orders nodes with a common father,
- $nl(v)$, for any vertex $v \in N$, is a non-empty subset of $FL(\varphi_0)$ called a *node label* of v.

- $el(v)$, for any vertex $v \in N \setminus \{r\}$, is a definition constant from $\mathcal{D}_{\varphi_0}$ called an *edge label* of v.

Additionally each vertex can be colored white or green and the following conditions hold:

1. For any vertex v and its son w the label of w is contained in the label of v.
2. For any vertex v the union of the node labels of those sons of v which edge labels are equal $el(v)$ is a proper subset of $nl(v)$.
3. Any two sons of the same vertex have disjoint node labels.
4. If w is a son of v then $el(w)$ is not older than $el(v)$ with respect to the definition list $\mathcal{D}_{\varphi_0}$.
5. If $el(v)$ is a ν-constant then v has no ν-sons, i.e., sons with the edge label being a ν-constant.
6. For any two sons, v and w, of the same vertex, if $el(v)$ is older than $el(w)$ with respect to $\mathcal{D}_{\varphi_0}$, then $v \prec w$.

Ordering $\prec$ can be extended to an ordering between not necessarily ancestral vertices. We say that v is to the *left* of w iff some (maybe not proper) ancestor of v is smaller in $\prec$ ordering than some (maybe not proper) ancestor of w. If v is a son of w and $el(v) = W$ then we say that v is a *W-son* of w. Conditions 1 and 3 guarantee that for any formula ψ occurring in a state s there is the lowest vertex v such that $\psi \in nl(v)$. We will call such a vertex the *ψ-vertex of s*.

The initial state of the automaton will be a tree consisting only of the root labeled $\{\varphi_0\}$.

Next we describe the deterministic transition function of the automaton. It will be always the case that after reading a sequent $\Sigma \vdash_{\mathcal{D}_{\varphi_0}}$ the automaton will enter a state with the root labeled by Σ. Suppose the automaton is in a state s after reading $\Sigma \vdash_{\mathcal{D}_{\varphi_0}}$ and the next input is $\Sigma' \vdash_{\mathcal{D}}$. The next state is obtained by applying to the tree s the following sequence of actions:

1. Set the color of all the vertices to white.
2. Look at the next sequent $\Sigma' \vdash_{\mathcal{D}_{\varphi_0}}$ on the path and locally transform the labels of some vertices, depending on the rule applied to obtain this sequent:

   — for all the rules other then $(const)$, in all vertices of s replace the reduced formula with the resulting formulas appearing in Σ'. This means that for all the rules other then $(\langle\rangle_t)$ the reduced formula is replaced by at most two resulting formulas and the other formulas remain intact. In case of the $(\langle\rangle_t)$ rule:

$$\frac{\langle a\rangle\alpha, \Sigma \vdash_{\mathcal{D}}}{\alpha, \{\beta : [a]\beta \in \Sigma\} \vdash_{\mathcal{D}_{\varphi_0}}}$$

we first delete all formulas not of the form $[a]\beta$, for some β, except the formula $\langle a\rangle\alpha$. Then we replace $\langle a\rangle\alpha$ with α and $[a]\beta$ with β in each node label of s

— Suppose we apply regeneration rule (*const*):

$$\frac{U, \Gamma \vdash_{\mathcal{D}_{\varphi_0}}}{\alpha(U), \Gamma \vdash_{\mathcal{D}_{\varphi_0}}}$$

Let v be the lowest vertex of s such that $U \in nl(v)$ and $el(v)$ is not younger (older or equal with respect to $\mathcal{D}_{\varphi_0}$) than U. If there is no such vertex then let v be the root of s. First replace U with $\alpha(U)$ in the node label of v and in the labels of all ancestors of v. Next delete U from node labels of all proper descendants of v. Additionally, if U or $el(v)$ is not a ν-constant, create a son w of v, with labels $el(w) = U$ and $nl(w) = \{\alpha(U)\}$. Finally make w $\prec$-bigger than all its brothers with edge labels not younger than U and smaller from all other brothers.

3. For any vertex v, if a formula φ occurs already in a vertex to the left of v then delete φ from the node label of v.
4. Delete all vertices with empty labels.
5. For any vertex v such that $el(v) = U$ is a μ-constant and $nl(v)$ is equal to the sum of the labels of its U-sons, light v green and delete all descendants of v from the tree.

A run of automaton $\mathcal{A}_{\varphi_0}$ is accepting iff there is a vertex which disappears only finitely many times and lights green infinitely many times on the run.

One word must be said about "vertex management". In clause 2 of the definition of the transition function we are at some point required to add a new vertex to a state, i.e., one not occurring in it already. Clearly we cannot have an infinite supply of vertices because the number of states of the automaton must be finite. The solution is to "reuse" vertices, i.e., when a vertex is deleted we put it into a common pool and when a new vertex is needed just take any vertex from the pool. If we put initially into the pool more vertices then the size of the largest possible state then we would be sure that there is always something in the pool when needed. Hence our first step is to find the bound on the size of the states of $\mathcal{A}_{\varphi_0}$.

Lemma 4.3 The size of a state tree is bounded by $n * m + 1$ where n is the number of formulas in $FL(\varphi_0)$ and m is the number of definition constants in $\mathcal{D}_{\varphi_0}$. The number of states is finite.

Proof
Let s be any state of $\mathcal{G}_{\varphi_0}$. With every vertex v of s, except the root, we can assign a pair $(el(v), \varphi)$, where $\varphi \in nl(v)$ is a formula not occurring in any $el(v)$-son of v. This shows the upper bound on the number of states because the assignment is injective.

The second part of the lemma follows directly from the first if we apply the strategy for "reusing vertices" described above. $\square$

Hence what we have described is a Rabin automaton on strings. Before proving correctness of the construction let us focus on one more specialised property of the $\mathcal{A}_{\varphi_0}$ which we will need in the future.

Lemma 4.4 For any μ-constant $U_i = \mu X.\alpha_i(X)$ in $\mathcal{D}_{\varphi_0}$, state s reachable from the start state of the automaton $\mathcal{A}_{\varphi_0}$ and vertex v of s such that $el(v) = U_i$, the formula $\mu X.\alpha_i(X)$ does not belong to $nl(v)$.

Proof
Let us take a μ-constant U_i as above and define the set of formulas Cl as the smallest set such that:

- $\alpha(U_i)$ belongs to Cl,
- if $\psi \in Cl$ then all subformulas of ψ belong to Cl,
- if $\sigma X.\beta(X) \in Cl$ and there is a definition constant W such that $(W = \sigma X.\beta(X))$ is in $\mathcal{D}_{\varphi_0}$ then $\beta(W)$ belongs to Cl.

Observe that $\mu X.\alpha_i(X) \notin Cl$ because all formulas in Cl are shorter. For any state s and its vertex v, s.t. $el(v) = U_i$ we show by induction on the number of steps needed to reach state s, that $nl_s(v) \subseteq Cl$.

The base step is when the vertex v is created in a state s. It's label is then exactly $\{\alpha(U_i)\}$. The induction step is straightforward.

$\square$

Finally we show correctness of the construction

Proposition 4.5 The automaton accepts a path $\mathcal{P}$ of a tableau for φ_0 iff there exists an infinite μ-trace on $\mathcal{P}$.

The proof of the proposition is very similar to the proof of correctness of the Safra's construction. First let us prove left to right implication.

Lemma 4.6 If the automaton $\mathcal{A}_{\varphi_0}$ accepts a path $\mathcal{P}$ of a tableau for φ_0 then there exists an infinite μ-trace on the path.

Proof
Consider an accepting run of the automaton $s_0, s_1, \ldots$ We will denote by nl_i, el_i the node labeling and the edge labeling of the state s_i respectively. Let v be a vertex which lights green i.o. in this run and disappears only finitely many times. Consider two positions i, j, after v disappears last time, such that v lights green in s_i and next time it lights green in s_j.

First we would like to show that for every formula in $nl_j(v)$ there exists a trace from some formula in $nl_i(v)$ such that the oldest constant regenerated on the trace is $el_i(v) = el_j(v)$. Clearly $el_i(v)$ is a μ-constant.
To do this we show that for any s_k, between s_i and s_j, and any formula $\varphi \in nl_k(v)$:

- if $\varphi \in nl_k(w)$ for some son w of v, then there is a part of a trace to φ from some formula in $nl_i(v)$ such that the highest regeneration on it is that of $el_k(w)$
- otherwise there is a part of a trace to φ from some formula in $nl_i(v)$ without any regeneration at all.

The proof is by induction on the distance from s_i

- Base step when $k = i$ is trivial.
- All steps except of regenerations are rather straightforward.
- If the last step was regeneration of a constant V then we have two cases. The first is when $V \notin nl_k(v)$ or V is older than $el_k(V)$. This case is easy because the only thing which can happen is that some formula can be removed from the labels of all the vertices in the subtree of v.

  In the other case when $V \in nl_k(v)$ and $el_k(v)$ is not younger than V we have two possibilities

  - If there is a son w of v, such that $V \in nl_k(w)$ and $el_k(w)$ is not younger than V then no new son of v is created and V is replaced by an appropriate $\alpha(V)$ in the label of w. This is sound as there is a trace to $\alpha(V)$ with the oldest regeneration being that of $el_k(w)$ because there was one to V by induction hypothesis.
  - If, on the other hand, V is older than $el_k(w)$ or there is no w at all then from the induction hypothesis there is a trace to V on which no constant older than V was regenerated. Hence there is a trace to $\alpha(V)$ where V is the oldest regenerated constant. According to the definition of the transition function a new son w' of v is created with the node label $\{\alpha(V)\}$ and edge label V. Additionally V is deleted from the label of w. These are exactly the steps which must be done to make the induction thesis satisfied.

Observe that vertices with edges labeled with ν-constants were needed in the last step of the above induction.

Now consider a graph with the set of nodes

$$\{(\varphi, k) : \varphi \in nl_k(v), v \text{ lights green in } s_k\}$$

and an edge from (φ_1, k_1) to (φ_2, k_2) whenever there is a trace from φ_1 in s_{k_1} to φ_2 in s_{k_2} on which $el(v)$ is the oldest regenerated constant. Here v is our node which lights green infinitely often on the run. From what we have shown above it follows that at for any (φ, k) there is an edge leading to it from some (φ', k'). This is because v lights green only when the sum of the labels of its $el(v)$-sons is equal $nl(v)$. This means that at least a part of this graph is an infinite connected directed acyclic graph. The degree of every vertex of it is of course finite hence there must exist an infinite path in this graph. This path is a μ-trace we were looking for.

$\square$

Lemma 4.7 If there is a μ-trace on a path $\mathcal{P}$ of a tableau for φ_0 then the automaton $\mathcal{A}_{\varphi_0}$ accepts $\mathcal{P}$.

Proof
Let us consider a path $\mathcal{P}$ with a μ-trace and the run of the automaton, $s_0, s_1, \ldots$ on it. The trace on $\mathcal{P}$ is a sequence of formulas $\{\alpha_i\}_{i\in\mathcal{N}}$, s.t. α_i occurs in the i-th tableau sequent of $\mathcal{P}$. On the other hand the automaton $\mathcal{A}_{\varphi_0}$ has the property that after reading a tableau sequent $\Sigma \vdash_{\mathcal{D}_{\varphi_0}}$, the root of its current state is labeled by Σ. Hence α_i is always present in the root of the state s_i.

From some moment j_0, the oldest constant regenerated on the trace is some μ-constant U. We can assume, without loss of generality, that α_{j_0} occurs in a son of the root with edge label, not younger than U. Indeed, otherwise we just have to wait, in the worst case, for the next regeneration of U. From that moment formulas from the trace can only move to the left according to clause 3 of the description of the transition function. But because they can do it only finitely many times, after some time the trace must settle in some son of the root, say v_0, forever.

If v_0 lights green i.o. then we are done. If not then let j_1 be the last moment v_0 lights green. Not later than the next regeneration of U the formula from the trace will appear in some W_1-son of v_0 where W_1 is not younger than U. After some moves to the left it must settle in some v_1 forever. If v_1 lights green i.o. we are done if not we repeat the reasoning and find a son of v_1, v_2 an so on. Observe that we cannot go this way forever because each state is a tree of bounded size. $\square$

Together Lemmas 4.7 and 4.6 prove Proposition 4.5 thereby showing correctness of the construction.

Automaton $\mathcal{A}_{\varphi_0}$ expands to a nondeterministic automaton on trees over one letter alphabet $\mathcal{TA}_{\varphi_0}$, such that its accepting runs closely correspond to the refutations of φ_0. Automaton $\mathcal{TA}_{\varphi_0}$ nondeterministically constructs a refutation for φ_0 and runs $\mathcal{A}_{\varphi_0}$ on each path. There is no need to remember refutation separately because sequents which are read by $\mathcal{A}_{\varphi_0}$ are remembered in the roots of the corresponding states. Hence states of $\mathcal{TA}_{\varphi_0}$ are exactly the states of $\mathcal{A}_{\varphi_0}$. The roots of the states of an accepting run of $\mathcal{TA}_{\varphi_0}$ give us a refutation of φ_0. Other way around given a refutation of φ_0 we can run the path automaton $\mathcal{A}_{\varphi_0}$ on each path of the refutation separately and because it is deterministic we will obtain a tree which is an accepting run of $\mathcal{TA}_{\varphi_0}$.

In [2] the following theorem is (implicitly) stated

Theorem 4.8 (Emerson) *Suppose $\mathcal{A}$ is a Rabin automaton over a single letter alphabet. If $\mathcal{A}$ accepts some tree then there is a graph $\mathcal{G}$ with states of $\mathcal{A}$ as nodes which unwinds to an accepting run of $\mathcal{A}$.*

Because we know that there is a refutation of φ_0, from this theorem we conclude that there is a graph $\mathcal{G}_{\varphi_0}$, with states of $\mathcal{A}_{\varphi_0}$ as nodes, which unwinds to an accepting run of $\mathcal{TA}_{\varphi_0}$. In the next sections we will show how to construct a proof of $\varphi_0 \vdash$ from this graph.

4.3 Constructions on the graph

In this section we investigate the properties of the graph $\mathcal{G}_{\varphi_0}$. We define several concepts which in turn will allow us to define an unwinding of $\mathcal{G}_{\varphi_0}$ into a finite tree with "back edges" T_{φ_0}.

Let us examine the structure of $\mathcal{G}_{\varphi_0}$. Since the unwinding of $\mathcal{G}_{\varphi_0}$ is accepted by $\mathcal{TA}_{\varphi_0}$, we know that on any infinite path of it there is a *confirming vertex*, i.e., a vertex which lights green i.o. and disappears only finitely many times. The crucial observation is that for any state s of $\mathcal{G}_{\varphi_0}$ and any path $\mathcal{P}$ passing i.o. through s, a confirming vertex of $\mathcal{P}$ belongs to s. Moreover we can linearly order such confirming vertices. Intuitively, if we have this ordering $\mathit{Ord}(s) = (v_1, \ldots, v_k)$ we know that on any cycle going through s, some v_i lights green and none of $v_1, \ldots, v_{i-1}$ disappears.

More formally let us start by fixing some arbitrary linear ordering on the set of all the vertices occurring in $\mathcal{G}_{\varphi_0}$, then $\mathit{Ord}(s)$ can be described as follows:

Definition 4.9 For any node s of $\mathcal{G}_{\varphi_0}$ we define an ordering of some of the vertices from s. Let $\mathit{Ord}(s)$ be the list $(v_1, \ldots, v_k)$ such that for each $i = 1, \ldots, k$, v_i is defined by the following rule:

> Consider a graph obtained from $\mathcal{G}_{\varphi_0}$ by deleting all the nodes where one of the vertices $v_1, \ldots, v_{i-1}$ lights green. Let $\mathcal{C}_i$ be the nontrivial strongly connected component of this graph containing node s. Then v_i is the smallest vertex, in our fixed linear ordering on all vertices, such that v_i does not disappear in any node of $\mathcal{C}_i$ and lights green in some node of $\mathcal{C}_i$. ■

It easy to check that we can always find a vertex v_i required in the above definition. If there were no such vertex then we would take a cycle through all the nodes of the strongly connected component and unwind it to an infinite path which would not be accepted by the automaton $\mathcal{A}_{\varphi_0}$. This would be a contradiction with the assumption that $\mathcal{G}_{\varphi_0}$ unwinds to an accepting run of $\mathcal{TA}_{\varphi_0}$.

The ordering $\mathit{Ord}(s)$ allows us to distinguish some special nodes of $\mathcal{G}_{\varphi_0}$, namely such that a vertex from $\mathit{Ord}(s)$ lights green in s. Observe that such a vertex must be the last vertex in $\mathit{Ord}(s)$.

Definition 4.10 A node s of $\mathcal{G}_{\varphi_0}$ is called a *loop node* if $\mathit{Ord}(s) = (v_1, \ldots, v_i)$ and v_i lights green in s. Vertex v_i will be called the *green vertex* of s. ■

Loop nodes are the nodes where we will apply our induction rule. This is why we duplicate each loop node in the definition of T_{φ_0} below.

Definition 4.11 We will unwind graph $\mathcal{G}_{\varphi_0}$ to a finite tree with *back edges* T_{φ_0}. Nodes of T_{φ_0} will be sequences consisting of nodes from $\mathcal{G}_{\varphi_0}$ and pairs $(\mathit{node}, 0\ \mathit{or}\ 1)$. For any sequence n and a node or a pair x we denote by nx a path obtained by concatenating x to the end of n; we denote the last element of the sequence by $n\downarrow$. Hence $nx\downarrow = x$. It will be also convenient to extend the definition of Ord, we let $\mathit{Ord}((s,1)) = \mathit{Ord}((s,0)) = \mathit{Ord}(s)$. Tree T_{φ_0} is defined as follows:

- The root of T_{φ_0} is the sequence consisting only of the initial node of $\mathcal{G}_{\varphi_0}$.
- A node n of T_{φ_0} is *final* iff one of the conditions holds:
  1. There is a prefix $m(m\downarrow, 0)$ of n such that for the green vertex v of $m\downarrow$ we have: (i) definition constant $el(v)$ appears in the node label of v in $n\downarrow$, and (ii) v appears in $Ord(m'\downarrow)$, for any prefix m' of n longer than m.
  2. There is a prefix $m(n\downarrow, 1)$ of n with the property that for any longer prefix m' of n, $Ord(n\downarrow)$ is a prefix of $Ord(m'\downarrow)$. In this case we will say that there is a *back edge* from n to m.

  Finial nodes will have no prolongation in T_{φ_0}.
- If not final node n belongs to T_{φ_0} and $n\downarrow$ is not a loop node then for any son s in $\mathcal{G}_{\varphi_0}$ of the last state in n we add a node ns.
- If not final node n belongs to T_{φ_0}, and $n\downarrow$ is a loop node then we add two nodes $n(n\downarrow, 0)$ and $n(n\downarrow, 1)$. ■

Fact 4.12 Tree T_{φ_0} constructed above is finite.

Proof
Suppose that there is an infinite path P in T_{φ_0} without taking a back edge. Let us take any node s of $\mathcal{G}_{\varphi_0}$ which occurs infinitely often on this path. Consider $Ord(s) = (v_1, \ldots, v_k)$. By definition of $Ord(s)$ on any path of $\mathcal{G}_{\varphi_0}$ between any two occurrences of s some vertex from $v_1, \ldots, v_k$ must light green. Let us take the smallest i s.t. v_i lights green i.o. on P.

Let s' be the state which occurs i.o. on the path and where v_i lights green. Because from some point $v_1, \ldots, v_{i-1}$ don't light green on P there must be a cycle in $\mathcal{G}_{\varphi_0}$ from s to s' and back to s on which none of $v_1, \ldots, v_{i-1}$ lights green. It follows from the definition of $Ord(s)$ that none of $v_1, \ldots, v_i$ disappears on such a cycle. Hence $Ord(s') = (v_1, \ldots, v_i)$. So s' is a loop node and v_i is its green vertex.

Let ms' and ns' be two nodes of P such that none of $v_1, \ldots, v_{i-1}$ lights green in the nodes in between. It should be clear that for any node o between ms' and ns', $Ord(s')$ is a prefix of $Ord(o\downarrow)$. To light green v_i must have at least one $el(v_i)$-son and this may be created only when $el(v_i) \in nd(v_i)$. If the next node after ms' were $ms'(s', 0)$ then between ms' and ns' we could find a final node of the first type. So the next node after ms' is $ms'(s', 1)$. But in this case ns' is a final node of the second type. □

The next concept is very important although quite straightforward. We will need to assign provable sequents to final nodes. To do this for any node n we need to know all its ancestors to which we may come by taking a back edge. This set will be called the set of *active nodes* of a node. We will use it for final nodes of the second type. For final nodes of the first type it is enough to keep track of all the prefixes which may cause some descendant of a node to become a final node. This gives us the set AN_0.

Definition 4.13 For any node n of T_{φ_0} we define the set of *active nodes* of n, $AN(n)$, as the smallest set such that:

- if there is a back edge from n or some descendant of n to some proper ancestor m of n then $m \in AN(n)$,
- if $m \in AN(n)$ then $AN(m) \subseteq AN(n)$.

Let $AV(n) = \{v : m \in AN(n),\ v \text{ the green vertex of } m\!\downarrow\}$. We call it the set of *active vertices* of n. Observe that only nodes m with $m\!\downarrow$ being a loop node can belong to $AN(n)$.

Let $AN_0(n)$ be the set of nodes m such that $m(m\!\downarrow, 0)$ is a, maybe not proper, prefix of n and the green vertex of $m\!\downarrow$ appears in $\mathit{Ord}(m'\!\downarrow)$ for any prefix m' of n longer than m.

Let $AV_0(n) = \{v : m \in AN_0(n),\ v \text{ the green vertex of } m\!\downarrow\}$. ■

Final lemma of this subsection presents the properties of active sets we will use in the last step of the proof.

Lemma 4.14 For any node n of T_{φ_0} and $v \in AV(n)$, v is a vertex of $n\!\downarrow$. For any node nx of T_{φ_0}:

- if x is not a pair then $AN(n) \subseteq AN(nx)$ and $AN_0(n) \subseteq AN_0(nx)$,
- otherwise

$$\begin{array}{ll} AN(n(n\!\downarrow, 0)) \subseteq AN(n) & AN(n(n\!\downarrow, 1)) \subseteq AN(n) \cup \{n\} \\ AN_0(n(n\!\downarrow, 0)) = AN_0(n) \cup \{n\} & AN_0(n(n\!\downarrow, 1)) = AN_0(n) \end{array}$$

Proof

To prove the first part of the lemma observe that if m is an ancestor of n to which leads a back edge from some descendant of n then $\mathit{Ord}(m\!\downarrow)$ is a prefix of $\mathit{Ord}(n\!\downarrow)$. From this follows that for any $m \in AN(n)$ sequence $\mathit{Ord}(m\!\downarrow)$ is a prefix of $\mathit{Ord}(n\!\downarrow)$. Of course all vertices from $\mathit{Ord}(n\!\downarrow)$ appear in $n\!\downarrow$.

The second part of the lemma follows directly from the definition of T_{φ_0} and sets of active nodes. □

4.4 Overview of the sequent assignment

The last step in the completeness proof is to construct a proof of $\varphi_0 \vdash$ from T_{φ_0}. This will be done by assigning a sequent to every node of T_{φ_0} in such a way that leaves will be assigned provable sequents and a sequent assigned to a father will be provable from the sequents assigned to its sons. Here we would like to give some intuitions about the way it is done. For n a node of T_{φ_0} we will use Γ_n to denote the set of all the formulas appearing in the last state of n. We will use $\langle\!\langle \Gamma \rangle\!\rangle_{\mathcal{D}}$ as an abbreviation of $\{\langle\!\langle \gamma \rangle\!\rangle_{\mathcal{D}} : \gamma \in \Gamma\}$, that is a set of formulas obtained from Γ by replacing definition constants with their definitions form $\mathcal{D}$.

Each node of T_{φ_0} is a sequence consisting of states of $\mathcal{A}_{\varphi_0}$ and pairs of the form $(\mathit{state}, 0 \text{ or } 1)$. There is a very easy way of assigning sequents. To every node n of T_{φ_0} assign $\langle\!\langle \Gamma \rangle\!\rangle_{\mathcal{D}_{\varphi_0}} \vdash$. Expansion operation is used to remove possible occurrences of definition constants. By definition of $\mathcal{A}_{\varphi_0}$ for every internal node n the set of formulas Γ_n is a conclusion in some tableau rule in which sets of formulas occuring in the sons of n are the assumptions. Our simple sequent assignment transforms such a rule into a rule of our system. Hence this assignment is *locally sound*, that is a sequent assigned to some internal node of T_{φ_0} is provable from the sequents assigned to its sons. If all the leaves of T_{φ_0} were labeled by axioms than the result would be a proof of $\varphi_0 \vdash$.

In general in T_{φ_0} we may have also final nodes which are not axioms and after the simple assignment defined above this nodes will not be labeled by axioms. We need more refined assignment to get axioms in finial nodes. Let us describe its idea.

There are two types of final nodes. Let n be a final node of the first type. This means that there is a prefix $m(s,0)$ of n such that for the green vertex v of s definition constant $el(v)$ appears in the node label of v in $n\!\downarrow$ and v appears in $Ord(m'\!\downarrow)$, for any prefix m' of n longer than m.

Let $U = el(v)$ and $(U = \mu X.\alpha(X)) \in \mathcal{D}_{\varphi_0}$. For simplicity assume that $nd(v)$ is one element set, say, $\{\varphi(U)\}$. The simple assignment described above would give us for node m the sequent $\langle\!\langle \Gamma_m \rangle\!\rangle_{\mathcal{D}_{\varphi_0}} \vdash$. Now $\varphi(U) \in \Gamma_m$ and $\langle\!\langle \varphi(U) \rangle\!\rangle_{\mathcal{D}_{\varphi_0}}$ is of the form $\varphi'(\mu X.\beta(X))$. We can apply (ind) rule to this sequent and assign to $m(s,0)$ the assumption

$$\varphi'(\mathit{ff}), \langle\!\langle \Gamma_m \setminus \{\varphi(U)\} \rangle\!\rangle_{\mathcal{D}_{\varphi_0}} \vdash \tag{14}$$

we use $\setminus$ to denote set subtraction. Let $\mathcal{D}^0_{\varphi_0}$ be a definition list obtained from $\mathcal{D}_{\varphi_0}$ by replacing the definition of U by $(U = \mathit{ff})$. Sequent (14) can be presented as $\langle\!\langle \varphi(U) \rangle\!\rangle_{\mathcal{D}^0_{\varphi_0}}, \langle\!\langle \Gamma \rangle\!\rangle_{\mathcal{D}_{\varphi_0}} \vdash$. Let o be a descendant of m and let Γ'_o be the node label of v in o. We assign to o the sequent $\langle\!\langle \Gamma'_o \rangle\!\rangle_{\mathcal{D}^0_{\varphi_0}}, \langle\!\langle \Gamma \setminus \Gamma'_o \rangle\!\rangle_{\mathcal{D}_{\varphi_0}} \vdash$. This gives us another locally sound assignment if only v does not disappear on the way from m to o.

If v disappears on the path to o and reappears once again then there is no way of restoring ff in formulas from the label of v. An example of a situation of this kind is presented below. This is a schematic representation of a path from m to o. The first row contains unexpanded sequents the second their expanded versions; third row gives the names of the nodes on the path.

$\varphi(U), \Gamma \vdash_{\mathcal{D}_{\varphi_0}}$	$\varphi'(\mathit{ff}), \langle\!\langle \Gamma \rangle\!\rangle_{\mathcal{D}_{\varphi_0}} \vdash$	$m(s,0)$
$\Gamma' \vdash_{\mathcal{D}_{\varphi_0}}$	$\langle\!\langle \Gamma' \rangle\!\rangle_{\mathcal{D}_{\varphi_0}} \vdash$	$m(s,0)s'$
$\vdots$	$\vdots$	$\vdots$
$U, \Gamma'' \vdash_{\mathcal{D}_{\varphi_0}}$	$\mu X.\beta(X), \langle\!\langle \Gamma'' \rangle\!\rangle_{\mathcal{D}_{\varphi_0}} \vdash$	
$\alpha(U), \Gamma'' \vdash_{\mathcal{D}_{\varphi_0}}$	$\beta(\mu X.\beta(X)), \langle\!\langle \Gamma'' \rangle\!\rangle_{\mathcal{D}_{\varphi_0}} \vdash$	o

Suppose that v disappears in $m(s,0)s'$. This means that in this node all the formulas are expanded using $\mathcal{D}_{\varphi_0}$. Next when v reappears in o and has $\alpha(U)$ in the label we cannot use $\mathcal{D}'_{\varphi_0}$ for expanding $\alpha(U)$ and have a sound sequent assignment at the same time. This is because the rule

$$\frac{\beta(\mathit{ff}), \Sigma \vdash}{\mu X.\beta(X), \Sigma \vdash}$$

is not sound.

We know that v does not disappear on the path from m to n and that U appears in the node label of v in n. This means that a sequent assigned to n is of the form $\mathit{ff}, \Sigma \vdash$ for some Σ. As ff stands for $p \wedge \neg p$ this sequent is easily provable. This shows how to use induction rule to assign an axiom to a final node of the first type.

Let n be a final node of the other type, i.e., the one with a back edge to some $m(m\!\downarrow, 1)$. In this case $n\!\downarrow$ is a loop node, $m\!\downarrow = n\!\downarrow$ and for every prefix m' of n longer than m, $\mathit{Ord}(n\!\downarrow)$ is a prefix of $\mathit{Ord}(m'\!\downarrow)$.

Let v be the green vertex of $n\!\downarrow$ and let us assume as before that the node label of v is $\{\varphi(U)\}$ and $\langle\!\langle \varphi(U) \rangle\!\rangle_{\mathcal{D}_{\varphi_0}} = \varphi'(\mu X.\beta(X))$. Let $\widetilde{\Gamma}_m$ denote $\Gamma_m \setminus \{\varphi(U)\}$ and let P be the set off all the actions appearing in φ_0. To $m(m\!\downarrow, 0)$ we have assigned one assumption of the rule (ind) to $m(m\!\downarrow, 1)$ we will assign the other one:

$$\varphi'(\beta(\mu X.Z \wedge \beta(X))), \langle\!\langle \widetilde{\Gamma}_m \rangle\!\rangle_{\mathcal{D}_{\varphi_0}} \vdash \langle P^* \rangle (\varphi'(\mu X.Z \wedge \beta(X)) \wedge \bigwedge \langle\!\langle \widetilde{\Gamma}_m \rangle\!\rangle_{\mathcal{D}_{\varphi_0}}) \quad (15)$$

Let $\mathcal{D}^1_{\varphi_0}$ and $\mathcal{D}^2_{\varphi_0}$ be definition lists obtained from $\mathcal{D}_{\varphi_0}$ by replacing the definition of U by $(U = \alpha(\mu X.Z \wedge \alpha(X)))$ and by $(U = \mu X.Z \wedge \alpha(X))$ respectively. With this notation sequent (15) becomes

$$\langle\!\langle \varphi(U) \rangle\!\rangle_{\mathcal{D}^1_{\varphi_0}}, \langle\!\langle \widetilde{\Gamma}_m \rangle\!\rangle_{\mathcal{D}_{\varphi_0}} \vdash \langle P^* \rangle (\langle\!\langle \varphi(U) \rangle\!\rangle_{\mathcal{D}^2_{\varphi_0}} \wedge \bigwedge \langle\!\langle \widetilde{\Gamma}_m \rangle\!\rangle_{\mathcal{D}_{\varphi_0}})$$

Let o be a descendant of m and let Γ^1_o, Γ^2_o be respectively the label of v in o and the sum of the labels of the U-sons of v in o. We will assign to o the sequent:

$$\langle\!\langle \Gamma^2_o \rangle\!\rangle_{\mathcal{D}^2_{\varphi_0}}, \langle\!\langle \Gamma^1_o \setminus \Gamma^2_o \rangle\!\rangle_{\mathcal{D}^1_{\varphi_0}}, \langle\!\langle (\Gamma_o \setminus \Gamma^1_o) \setminus \Gamma^2_o \rangle\!\rangle_{\mathcal{D}_{\varphi_0}} \vdash \langle P^* \rangle (\langle\!\langle \varphi(U) \rangle\!\rangle_{\mathcal{D}^2_{\varphi_0}} \wedge \bigwedge \langle\!\langle \widetilde{\Gamma}_m \rangle\!\rangle_{\mathcal{D}_{\varphi_0}}) \quad (16)$$

Compared to the previous case the new element this time is that we consider also sons of v. A son of v is created when definition constant U appears in the node label of v and rule (const) is applied. The previous case was simpler because we terminated a path as soon as U appeared in the node label of v. To see how sons of v come on the scene let us examine what happens when a new son of v is created.

Let o be a node in which (const) rule is applied to U, and an U-son of v is created in the result. Roughly speaking obtained state s is constructed from $o\!\downarrow$ by adding a son of v with the node label $\{\alpha(U)\}$. The only son of o is os. The sequent assigned to o is of the form (16). Because a new son of v is created we know that $U \in \Gamma^1_o \setminus \Gamma^2_o$. Hence the sequent assigned to os is exactly the same as the sequent assigned to o. In other words this application of (const) becomes an identity under our assignment but a "category" of one of the formulas was changed and this will allow us to prove a sequent assigned to n.

It is quite easy to check that such an assignment is sound for all the nodes o such that on the path from m to o, vertex v does not disappear. This condition on v is important for the same reasons as in the previous case.

We know that v does not disappear on the path from m to n and the last states of m and n are the same. Moreover v lights green in $n\downarrow$. By definition of $\mathcal{A}_{\varphi_0}$ this means that the label of v is equal to the sum of the labels of its U-sons hence to all formulas in the label of v we can use $\mathcal{D}^2_{\varphi_0}$. Hence the sequent assigned to n is:

$$\langle\!\langle \varphi(U)\rangle\!\rangle_{\mathcal{D}^2_{\varphi_0}}, \langle\!\langle \widetilde{\Gamma}_m\rangle\!\rangle_{\mathcal{D}_{\varphi_0}} \vdash \langle P^*\rangle(\langle\!\langle \varphi(U)\rangle\!\rangle_{\mathcal{D}^2_{\varphi_0}} \wedge \bigwedge \langle\!\langle \widetilde{\Gamma}_m\rangle\!\rangle_{\mathcal{D}_{\varphi_0}})$$

which is easily provable.

Important point is that while describing this sequent assignment, we have taken a leaf of T_{φ_0} first and only then constructed a part of a proof. This approach does not guarantee that if we transform every path of T_{φ_0} separately we will obtain a set of paths composing back to a tree. The simplest solution seems to be to apply (ind) rule in every node m of T_{φ_0} for which $m\downarrow$ is a loop node and consequently m has two sons $m(m\downarrow, 0)$ and $m(m\downarrow, 1)$. Such a strategy would be deterministic and this is what we will do but there is one subtle point here.

Schematically our procedure looks like this. Each application of (ind) rule allows us to remember one node ending in a loop node. Remembered information is represented by variables Z added to μ-formulas. When arriving at a final node we can use the information we remembered to obtain an axiom sequent.

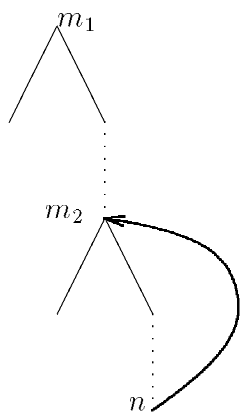

Figure 1: An example

Suppose we have a situation as pictured in Figure 1. Nodes m_1 and m_2 are places where we are to apply (ind) rule and there is a back edge from n to m_2. First we remember node m_1 and, say, use Z_1 for this. Next we remember m_2 and use Z_2. This means that m_2 has assigned a sequent of the form

$$\varphi_2(\mu X.\beta_2(X)), \Gamma \vdash \langle P^*\rangle\Delta$$

and we assign sequents

$$\varphi_2(\mathit{ff}), \Gamma \vdash \langle P^*\rangle\Delta$$
$$\varphi_2(\beta(\mu X.Z_2 \wedge \beta_2(X))), \Gamma \vdash \langle P^*\rangle\Delta, \langle P^*\rangle(\varphi_2(\mu X.Z_2 \wedge \beta(X)) \wedge \bigwedge\Gamma)$$

to $m_2(m_2\!\downarrow, 0)$ and $m_2(m_2\!\downarrow, 1)$ respectively. We have used here without mentioning an equivalence of $\langle P^*\rangle\Delta$ with $\langle P^*\rangle\langle P^*\rangle\Delta$. Observe that Z_1 may appear in Γ as well as in φ_2.

Arriving at n we hope to have something like

$$\varphi_2(\mu X.Z_2 \wedge \beta_2(X)), \Gamma \vdash \langle P^*\rangle(\varphi_2(\mu X.Z_2 \wedge \beta(X)) \wedge \bigwedge\Gamma), \langle P^*\rangle\Delta' \tag{17}$$

with Z_1 occuring in Γ. But if the green vertex of m_1 disappears on the way to n then instead of Γ on the left side we would obtain set Γ with Z_1 erased. Such a sequent is not valid in general hence it is also not provable. In this situation we can save ourself by forgetting about Z_1 before applying rule (ind) in m_2. This can be done using (cut) rule. But what if there is a leaf below m_2 where we need Z_1?

This is the place where sets of active nodes come to rescue. The solution is to remember only active nodes. This means that we start our assignment from the root of T_{φ_0} and at a node m_2 we first use weakening rule to remove all Z's connected with nodes which are not in $AN(m_2)$ and all *ff* connected with nodes not in $AN_0(m_2)$. Only then we apply (ind) rule as described above.

In our example we either forget about Z_1 before reaching m_2 but then we know by definition of T_{φ_0} and AN that Z_1 will not be needed below m_2, or we have Z_1 around but then the green node of m_1 will not disappear on the path to n. By definition we have that $AN(m_2) = AN(n) \cup \{m_2\}$ and $AN_0(m_2) \subseteq AN_0(n)$. This means that all Z's we had in m_2 are present in n and at least the same occurrences of definition constants are replaced by *ff*. Hence in n we get a sequent like (17) but maybe with even stronger left hand side.

4.5 Construction of a proof

In this final subsection we will show that sequent $\varphi_0 \vdash$ is provable thereby proving completeness of our axiomatisation. We will do it in two steps. First we will define a sequent $S(n)$ for very node n of T_{φ_0}. Next we will show that such a sequent assignment is sound. That is we will show that sequents assigned to the leaves are provable and for any other node n, sequent $S(n)$ is provable from the sequents assigned to the sons of n. As we will assign $\varphi_0 \vdash$ to the root of T_{φ_0} this will show that $\varphi_0 \vdash$ is provable.

Recall that $\mathcal{D}_{\varphi_0}$ denotes the definition list $\llbracket\varphi_0\rrbracket$. In what follows we will use the convention that W_i stands for i-th definition constant in $\mathcal{D}_{\varphi_0}$ and γ_i for the formula it defines. Similarly we denote by U_i the i-th μ-constant in $\mathcal{D}_{\varphi_0}$ and let $\alpha_i(X)$ be such that $(U_i = \mu X.\alpha_i(X)) \in \mathcal{D}_{\varphi_0}$. Letters d and d_μ will stand for the number of definition constants in $\mathcal{D}_{\varphi_0}$ and the number of μ-constants in $\mathcal{D}_{\varphi_0}$ respectively. Let P be the set of all the actions appearing in φ_0. We will assume that we have a fresh variable Z_m for every node m of T_{φ_0} such that $m\!\downarrow$ is a loop node. This variables do not appear in φ_0.

Definition of $S(n)$ will be done in a several steps. With any node n of T_{φ_0} is associated a set of formulas Γ_n which is the set of all the formulas in the root of $n\!\downarrow$ ($n\!\downarrow$ being a state of $\mathcal{A}_{\varphi_0}$ is itself a tree). We start with determining a definition list for each vertex of $n\downarrow$, we call it *signature* of a vertex. Next we define definition list $\mathcal{D}(n,\varphi)$ for every formula $\varphi \in \Gamma_n$. For every $\varphi \in \Gamma_n$ there is the lowest vertex in $n\!\downarrow$ containing φ. Let us call it the *φ-vertex* of s. We then define $\mathcal{D}(n,\varphi)$ from the signature of this vertex. The sequent $S(n)$ assigned to a node n will have a form $\{(\!|\varphi|\!)_{\mathcal{D}(n,\varphi)} : \varphi \in \Gamma_n\} \vdash \langle P^*\rangle\Delta$, where Δ will be somehow defined from the structure of $n\!\downarrow$ and $AN(n)$.

Definition 4.15 Let n be a node of T_{φ_0}. For any vertex v of $n\downarrow$ we define its signature, $\| v \|_n$, to be a sequence of formulas $(\delta_1,\ldots,\delta_{d^\mu})$ where for each $i = 1,\ldots,d^\mu$ formula δ_i is defined as follows:

- First find the lowest ancestor u of v with $el(u) = U_i$.
- Let N be the set of nodes $o \in AN(n)$ such that the green vertex of o is an ancestor of u and has U_i as an edge label. Let m be the closest to n, in T_{φ_0}, node from N.
- Finally let $\mathcal{V} = \bigwedge\{Z_o : o \in N\}$ and $\mathcal{V}' = \bigwedge\{Z_o : o \in N, o \neq m\}$. We define δ_i by cases:
  1. if $u \in AV_0(n)$ then $\delta_i = \mathit{ff}$
  2. if $u \in AV(n)$ then $\delta_i = \mathcal{V}' \wedge \alpha_i(\mu X.\mathcal{V} \wedge \alpha_i(X))$
  3. otherwise $\delta_i = \mu X.\mathcal{V} \wedge \alpha_i(X)$

For $n\downarrow$ being a loop node we define $\overline{\| v \|}_n$ and $\|\widehat{v}\|_n$ to be identical to $\| v \|_n$ if v is not the green vertex of $n\downarrow$. In case v is the green vertex of $n\!\downarrow$ then $el(v) = U_i$ for some μ-constant U_i in $\mathcal{D}_{\varphi_0}$ and the difference between $\| v \|_n$ and $\overline{\| v \|}_n$ or $\|\widehat{v}\|_n$ is only at index i, we let $\overline{\delta}_i = \mu X.\mathcal{V} \wedge \alpha_i(X)$ and $\widehat{\delta}_i = \mu X.\mathcal{V} \wedge Z_n \wedge \alpha_i(X)$.

Definition 4.16 Let n be a node of T_{φ_0}, $\varphi \in \Gamma_n$ and let v be the φ-vertex of $m\downarrow$. Let $(\delta_1,\ldots,\delta_{d^\mu}) = \| v \|_n$. We define $\mathcal{D}(n,\varphi)$ to be a definition list obtained from $\mathcal{D}_{\varphi_0}$ by replacing for every $i = 1,\ldots,d^\mu$ a definition of i-th μ-constant U_i by $(U_i = \delta_i)$. Similarly we define $\overline{\mathcal{D}}(n,\varphi)$ and $\widehat{\mathcal{D}}(n,\varphi)$ but using $\overline{\| v \|}_n$ and $\|\widehat{v}\|_n$ respectively.

For any formula $\varphi \in \Gamma_n$ we define $\| \varphi \|_n$, $\overline{\| \varphi \|}_n$ and $\|\widehat{\varphi}\|_n$ to be $(\!|\varphi|\!)_{\mathcal{D}(n,\varphi)}$, $(\!|\varphi|\!)_{\overline{\mathcal{D}}(n,\varphi)}$ and $(\!|\varphi|\!)_{\widehat{\mathcal{D}}(n,\varphi)}$ respectively.
For a node n we define three formulas

$$F(n) = \bigwedge\{\| \varphi \|_n : \varphi \in \Gamma_n\},\quad \overline{F}(n) = \bigwedge\{\overline{\| \varphi \|}_n : \varphi \in \Gamma_n\}$$
$$\widehat{F}(n) = \bigwedge\{\|\widehat{\varphi}\|_n : \varphi \in \Gamma_n\}$$

Finally we associate a sequent with every node n of T_{φ_0}. If $n\downarrow$ is not a loop node then $S(n)$ is

$$F(n) \vdash \{\langle P^*\rangle \widehat{F}(m) : m \in AN(n)\}$$

and if $n\downarrow$ is a loop node we let $S(n)$ to be

$$\overline{F}(n) \vdash \{\langle P^*\rangle \widehat{F}(m) : m \in AN(n)\} \qquad \blacksquare$$

Probably a few more words is needed to justify this definition of the sequent assignment. The right hand side of the sequent is used to keep the labels of the nodes to which we may return. The real work is done on the left hand side using formulas $F(n)$. These are expansions of Γ_n with Z-variables in places designated by n and $AN(n)$. Formula $\overline{F}(n)$ is different from $F(n)$ only when $n\downarrow$ is a loop node and the green vertex v of $n\downarrow$ belongs to $AV(n)$. In this situation $\overline{F}(n)$ is a stronger version of $F(n)$. It is possible to have this stronger formula because we know that v lights green in $n\downarrow$ hence its label was equal to the sum of the labels of its $el(v)$-sons. Formula $\widehat{F}(p)$ is what we would obtain on the right hand side of the longer assumption if we applied (ind) rule to $\overline{F}(m)$. The left hand side of the assumption will be just $F(m(m\downarrow, 1))$ hence we don't need special symbol for this one.

Having defined sequent assignment for T_{φ_0} we are ready to prove its properties. In a sequence of lemmas we will prove that if m is a leaf then $S(m)$ is provable and if m has sons $m_1, \ldots, m_k$ then $S(m)$ is provable form $S(m_1), \ldots, S(m_k)$. Let us first show why all the leaves of T_{φ_0} are easily provable.

Lemma 4.17 Every leaf of T_{φ_0} has assigned a provable sequent.

Proof

There are three kinds of leaves in T_{φ_0}. First there are nodes n such that $n\downarrow$ has no son in $\mathcal{G}_{\varphi_0}$. In this case we know that some propositional constant p and its negation occur in Γ_n. By definition of $F(n)$, we have $p, \neg p \in F(n)$.

Next case is when n is a final node from which there is no back edge. This means that there is a prefix $m(m\downarrow, 0)$ of n and the green vertex v of $m\downarrow$ such that: (i) definition constant $el(v)$ appears in the node label of v in $n\downarrow$, and (ii) for any prefix m' of n longer than m, v appears in $Ord(m'\downarrow)$. Hence $v \in AV_0(n)$.

Now $el(v) = U_i$ is some, say i-th, μ-definition constant form $\mathcal{D}_{\varphi_0}$ and by definition i-th component of $\parallel v \parallel_n$ is $f\!f$. It is easy to see that v cannot have any U_i-sons so v is the U_i-vertex of $n\downarrow$. Hence we have that $\parallel U_i \parallel_n = f\!f$ and $f\!f \in F(n)$.

The last case is when n is a final node from which there is a back edge to some node m. Because $n\downarrow$ is a loop node, $S(n)$ is of the form:

$$\overline{F}(n) \vdash \{\langle P^*\rangle \widehat{F}(m) : m \in AN(n)\}$$

We know that $m\downarrow = n\downarrow$ and for any prefix m' of n longer than m, $Ord(n\downarrow)$ is a prefix of $Ord(m'\downarrow)$. From definition it follows that $AN(n) = AN(m) \cup \{m\}$ and $AN_0(m) \subseteq AN_0(n)$. It is easy to check that for every vertex v of $n\downarrow$, every $j \in \{1, \ldots, d^\mu\}$ and δ_j^m, δ_j^n j-th components of $\widehat{\parallel v \parallel_m}$ and $\overline{\parallel v \parallel_n}$ respectively the sequent $\delta_j^n \vdash \delta_j^m$ is provable. Because all occurrences of definition constants are positive we have by Lemma 3.4 that $\overline{F}(n) \vdash \widehat{F}(m)$ is provable. Hence $\overline{F}(n) \vdash \langle P^*\rangle \widehat{F}(m)$ is provable. $\square$

Next our goal is to prove that the sequent associated with a father is provable from the sequents associated with its sons. Let us start by showing a few technical lemmas.

Lemma 4.18 Let n be a node of T_{φ_0} and let v, w be two vertices of $n\!\downarrow$, such that v is an ancestor of w. Suppose δ_i, δ'_i are i-th components of $\parallel v \parallel_s$ and $\parallel w \parallel_s$ respectively for some $i = 1, \ldots, d^\mu$, then the sequent $\delta'_i \vdash \delta_i$ is provable.

Proof
Let us choose any $i \in \{1, \ldots, d^\mu\}$. If w_i, the lowest ancestor of w s.t. $el(w_i) = U_i$, is also an ancestor of v then $\delta_i = \delta'_i$ and we are done. Otherwise let v_i be either the lowest ancestor of v s.t. $el(v_i) = U_i$ or the root if there is no such ancestor.

- if $w_i \in AN_0(n)$ then $\delta'_i = f\!f$ and of course $\delta'_i \vdash \delta_i$ is provable.
- if $w_i \in AN(n)$ then

  $$\delta'_i = \mathcal{V}' \wedge \alpha(\mu X.\mathcal{V} \wedge \alpha(X))$$
  $$\mathcal{V} = \bigwedge\{Z_o : o \in N\} \quad \mathcal{V}' = \bigwedge\{Z_o : o \in N, o \neq m\}$$

  where N is the set of nodes $o \in AN(n)$ such that some ancestor u' of w_i with $el(u') = U_i$ is the green vertex in $o\!\downarrow$, and m is the closest to n node form N.

  There are two possibilities for δ_i

  $$\delta_i = \mathcal{V}'_1 \wedge \alpha(\mu X.\mathcal{V}_1 \wedge \alpha(X)) \quad \text{or} \quad \delta_i = \mu X.\mathcal{V}_1 \wedge \alpha(X)$$
  where
  $$\mathcal{V}_1 = \bigwedge\{Z_o : o \in N_1\} \quad \mathcal{V}'_1 = \bigwedge\{Z_o : o \in N_1, o \neq m_1\}$$

  and N_1, m_1 are defined as above but with respect to v_i. Because w_i is a proper descendant of v_i and $w_i \in AN(n)$, we have $N_1 \subseteq N \setminus \{m\}$. Hence $\mathcal{V}' \vdash \mathcal{V}_1$ is provable. Because all occurrences of $\mathcal{V}, \mathcal{V}', \mathcal{V}_1, \mathcal{V}'_1$ are positive we have the proof of $\delta'_i \vdash \delta_i$ by Lemma 3.4.
- In case $w_i \notin AN(n)$ the argument is very similar.

$\square$

Lemma 4.19 The notion of a signature of a vertex v from a state $n\!\downarrow$ depends on the sets $AN(n)$ and $AN_0(n)$. It is easy to generalise the notion to arbitrary sets of nodes R, R_0 such that $R \cap R_0 = \emptyset$. So we can have something like $\parallel v \parallel_n^{R,R_0}$. This way we can also define $F^R_{R_0}(n)$. If we do so then $F^R_{R_0}(n) \vdash F(n)$ will be provable for any $R \supseteq AN(n)$, $R_0 \supseteq AN_0(n)$.

Proof
Let v be a vertex of $m\!\downarrow$ and $R \supseteq AN(n)$, $R_0 \supseteq AN_0(n)$. It is enough to show that for any i and δ_r, δ being i-th elements of $\parallel v \parallel_n^{R,R_0}$ and $\parallel v \parallel_n$ respectively, the sequent $\delta_r \vdash \delta$ is provable.

If $\delta_r = \mathit{ff}$ then it is obvious. Suppose $\delta_r = \mathcal{V}'_r \wedge \alpha_i(\mu X.\mathcal{V}_r \wedge \alpha_i(X))$ where $\mathcal{V}_r = \bigwedge\{Z_o : o \in N\}$, $\mathcal{V}'_r = \bigwedge\{Z_o : o \in N, o \neq m\}$ and N is the set of nodes $o \in R$ such that the green vertex of o is an ancestor of v and has U_i as an edge label; m is the closest to n node from N. According to Definition 4.15

$$\delta = \mathcal{V}' \wedge \alpha_i(\mu X.\mathcal{V} \wedge \alpha_i(X)) \text{ or } \delta = \mu X.\mathcal{V} \wedge \alpha_i(X)$$

where $\mathcal{V}$ and $\mathcal{V}'$ are defined similarly but with respect to $AN(n)$. Because $R \supseteq AN(n)$ it should be clear that $\mathcal{V}'_r \vdash \mathcal{V}$ is provable hence also $\delta_r \vdash \delta$ is provable. The remaining case is similar. $\square$

Lemma 4.20 Let n be a node of T_{φ_0}. Suppose a state r is obtained from $n\downarrow$ by applying all but the fifth step of the transition function and t is the state resulting after the application of this last step, i.e., some of the vertices of r might light green in t and have all its sons deleted. For sequence nr we can define $F(nr)$ in a similar way we defined $F(nt)$. The sequent $F(nr) \vdash F(nt)$ is provable and moreover if t is a loop node then $F(nr) \vdash \overline{F}(nt)$ is provable.

Proof
Let Γ be the label of the root of r which is the same as the label of the root of t. By the construction of the states, Γ is also the set of all the formulas occurring in r as well as in t. For any formula $\psi \in \Gamma$, the ψ-vertex in t, call it v_t, is either an ancestor of the ψ-vertex in r, denoted v_r, or vertices v_t and v_r are the same. If they are the same then $\| \psi \|_{nr} = \| \psi \|_{nt}$. Otherwise let $(\delta_1, \ldots, \delta_{d^\mu}) = \| v_t \|_{nr} = \| v_t \|_{nt}$ and $(\delta'_1, \ldots, \delta'_{d^\mu}) = \| v_r \|_{nr}$. Because v_r is a descendant of v_t in r then from Lemma 4.18 it follows that $\delta'_i \vdash \delta_i$ is provable for $i = 1, \ldots, d^\mu$. Hence the sequent $\| \psi \|_{nr} \vdash \| \psi \|_{nt}$ is provable by Lemma 3.4 because all occurrences of definition constants are positive. This shows that $F(nr) \vdash F(nt)$ is provable.

To see why $F(nr) \vdash \overline{F}(nt)$ is provable when t is a loop node, first observe that $\| \psi \|_{nt} = \overline{\| \psi \|}_{nt}$ if the ψ-vertex of t is not the green vertex of t.

Let u be the green vertex of t, $\psi \in nl(u)$ and let $el(u) = U_i$ be i-th μ-definition constant in $\mathcal{D}_{\varphi_0}$. Because u lights green in t there must be a descendant u' of u, in r s.t. $\psi \in nl(u')$ and $el(u') = el(u)$. The only difference between $\overline{\| u \|}_{nt} = (\delta_1, \ldots, \delta_{d^\mu})$ and $\| u' \|_{nr} = (\delta'_1, \ldots, \delta'_{d^\mu})$ may be at position i. But from the definition of signatures it follows that $\delta'_i \vdash \delta_i$ is provable. Then using once again Lemma 3.4 we have that $\| \psi \|_{nr} \vdash \overline{\| \psi \|}_{nt}$ is provable. $\square$

We are now ready to prove that the sequent associated with a father node is provable from the sequents associated with its sons. We do it by cases depending on the rule which was used in the node.

Lemma 4.21 Suppose that the only son m' of m was obtained from m by application of $(\wedge_t)$ rule:

$$\frac{\alpha, \beta, \Gamma \vdash_{\mathcal{D}_{\varphi_0}}}{\alpha \wedge \beta, \Gamma \vdash_{\mathcal{D}_{\varphi_0}}}$$

that is $\Gamma_m = \{\alpha \wedge \beta\} \cup \Gamma$ and automaton $\mathcal{A}_{\varphi_0}$ goes from $m\downarrow$ to $m'\downarrow$ after reading $\alpha, \beta, \Gamma \vdash_{\mathcal{D}_{\varphi_0}}$. In this case sequent $S(m)$ is provable from $S(m')$.

Proof
Let r be a state obtained from $m\!\downarrow$ by applying all but the last fifth step of the transition function on the input $\alpha, \beta, \Gamma \vdash_{\mathcal{D}_{\varphi_0}}$. This means that r was obtained from $m\downarrow$ by first setting the color of all the vertices in $m\downarrow$ to white. Then formula $\alpha \wedge \beta$ was replaced by two formulas, α and β. Next, if a formula α or β occured in a vertex v and in a vertex to the left of it then the formula was deleted from the label of v. Finally the vertices with empty labels were removed. Observe that in r there may still occur vertices which would light green and have all its sons removed if the last step of transition function were applied. We will show that

$$F(m) \vdash F(mr)$$

is provable.

For any formula $\varphi \in \Gamma, \varphi \neq \alpha, \varphi \neq \beta$ we have $\| \varphi \|_m = \| \varphi \|_{mr}$ because φ-vertices in $m\!\downarrow$ and r are the same.

If also $\alpha\wedge\beta$-vertex in $m\!\downarrow$ is the same as α-vertex in r then clearly $\| \alpha \wedge \beta \|_m \vdash \| \alpha \|_{mr}$. If it is not the case then it must be because $\alpha \in \Gamma$ and α occurs to the left of $\alpha \wedge \beta$ in $m\downarrow$. But then α-vertices in $m\downarrow$ and r are the same and $\| \alpha \|_m = \| \alpha \|_{mr}$. We can use similar argument for formula β.

This shows that the sequent $F(m) \vdash F^{AN(m)}_{AN_0(m)}(mr)$ is provable. We know from Lemma 4.14 that $AN(mr) \subseteq AN(m)$ and $AN_0(mr) \subseteq AN_0(m)$ hence by Lemma 4.19 the sequent $F(m) \vdash F(mr)$ is provable. From Lemma 4.20 we obtain that either $F(m) \vdash F(mt)$ or $F(m) \vdash \overline{F}(mt)$ is provable depending on whether t is a loop node or not. This shows that we can prove $S(m)$ from $S(mt)$.

□

Similar arguments also work in case of the other tableau rules. This is summarised in the following lemma.

Lemma 4.22 If $m_1, \ldots, m_k$ are the sons of m and one of the rules: $(\vee_t)$, (μ), (ν), $(const)$ or $(\langle\rangle_t)$ was applied in m then the sequent $S(m)$ is provable from $S(m_1), \ldots, S(m_k)$. (Actually $k = 1$ or $k = 2$.)

Proof
Let us just show one case. This is the case where Lemma 4.4 is needed. Assume that the rule applied was:

$$\frac{U_i, \Gamma \vdash_{\mathcal{D}_{\varphi_0}}}{\mu X.\alpha_i(X), \Gamma \vdash_{\mathcal{D}_{\varphi_0}}}$$

then, as before, let r be a state obtained form $m\!\downarrow$ by applying all but the last fifth step of the transition function on the input $U_i, \Gamma \vdash_{\mathcal{D}_{\varphi_0}}$. We will show that the sequent

$$F(m) \vdash F(mr)$$

is provable.

For any $\varphi \in \Gamma$, $\varphi \neq U_i$, φ-vertices in $m\!\downarrow$ and r are the same hence $\| \varphi \|_m = \| \varphi \|_{mr}$. For formula U_i we have two possibilities. The first possibility is that U_i-vertices in $m\!\downarrow$ and r are the same. In this case also $\| U_i \|_m = \| U_i \|_{mr}$.

Otherwise the U_i-vertex in r is a $\mu X.\alpha(X)$-vertex in $m\!\downarrow$, call it v. By Definition 4.16, $\| \mu X.\alpha_i(X) \|_m = \mu X.\alpha_i(X)[\gamma'_d/W_d]\ldots[\gamma'_1/W_1]$ where each γ'_i is determined using the signature $\| v \|_m$. Now $\| U_i \|_{mr} = U_i[\gamma'_d/W_d]\ldots[\gamma'_1/W_1]$ for the same formulas γ'_j, $j = 1,\ldots,d$, because v is the U_i-vertex in r.

By construction of the definition list, only constants older then U_i can appear in $\mu X.\alpha_i(X)$. From Lemma 4.4 it follows that neither v nor any of its ancestors have U_i as its edge label. This means that in the signature $\| v \|_m = (\delta_1,\ldots,\delta_{d^\mu})$, its i-th coordinate δ_i is $\mu X.\alpha_i(X)$. Hence

$$\begin{aligned} \| \mu X.\alpha_i(X) \|_m &= \mu X.\alpha_i(X)[\gamma_j/W_j]\ldots[\gamma_1/W_1] \\ \| U_i \|_{mr} &= U_i[\mu X.\alpha_i(X)/U_i][\gamma_j/W_j]\ldots[\gamma_1/W_1] \end{aligned}$$

where W_j is the youngest definition constant older than U_i.

Having shown that $F(m) \vdash F(mr)$ is provable it is enough to use Lemma 4.20 to complete this case. □

Finally we have to take care of the situation when $m\!\downarrow$ is a loop node and m has two sons $m(m\!\downarrow,0)$ and $m(m\!\downarrow,1)$. This is the only situation when the set of active nodes can increase and we get new nodes to "remember".

Lemma 4.23 Suppose m ends in a loop node and has two sons $m(m\!\downarrow,0)$ and $m(m\!\downarrow,1)$ then we can prove $S(m)$ from $S(m(m\!\downarrow,0))$ and $S(m(m\!\downarrow,1))$.

Proof

Suppose v is the green node of $m\!\downarrow$ and i, l are such that $U_i = W_l = el(v)$. Let $\mu X.\beta(X)$ be i-th element of $\overline{\| v \|_m}$ and let $\{\psi_1,\ldots,\psi_k\}$ be the node label of v in $m\!\downarrow$. We take definition list $\overline{\mathcal{D}}(m,\psi_1) = (W_1 = \gamma'_1)\ldots(W_d = \gamma'_d)$ as described in Definition 4.16 and set

$$\begin{aligned} \psi'_j &= \psi_j[\gamma'_n/W_n]\ldots[\gamma'_{l+1}/W_{l+1}][\gamma'_{l-1}/W_{l-1}]\ldots[\gamma'_1/W_1]; \quad j = 1,\ldots,k \\ \psi &= \bigwedge\{\psi'_j : j = 1,\ldots,k\} \\ \delta(X) &= \beta(X)[\gamma'_{l-1}/W_{l-1}]\ldots[\gamma'_1/W_1] \\ \gamma &= \bigwedge\{\overline{\| \phi \|_n} : \phi \in \Gamma_m \setminus \{\psi_1,\ldots,\psi_k\}\} \end{aligned}$$

Sequent $S(m)$ is by definition $\overline{F}(m) \vdash \{\langle P^*\rangle \widehat{F}(n) : n \in AN(m)\}$ and with the notation introduced above $\overline{F}(m)$ can be presented as $\psi[\mu X.\delta(X)/U_i] \wedge \gamma$. Hence $S(m)$ is of the form of the conclusion of (ind) rule with the assumptions:

$$\begin{aligned} \psi[ff/U_i] \wedge \gamma &\vdash \{\langle P^*\rangle \widehat{F}(n) : n \in AN(m)\} \\ \psi[\delta(\mu X.Z_m \wedge \delta(X))/U_i] \wedge \gamma &\vdash \{\langle P^*\rangle \widehat{F}(n) : n \in AN(m)\}, \\ &\qquad \langle P^*\rangle(\psi[\mu X.Z_m \wedge \delta(X)/U_i] \wedge \gamma) \end{aligned}$$

Because v is the U_i-vertex on $m\!\downarrow$, the first sequent is just

$$F^{AN(m)}_{AN_0(m)\cup\{m\}}(m(m\!\downarrow,0)) \vdash \{\langle P^*\rangle \widehat{F}(n) : n \in AN(m)\}$$

and we can prove it from $S(m(m\downarrow,0))$ using Lemma 4.19 because by Lemma 4.14 $AN(m(m\downarrow,1)) \subseteq AN(m)$ and $AN_0(m(m\downarrow,1)) = AN_0(m) \cup \{m\}$.
Similarly the second sequent is

$$F_{AN_0(m)}^{AN(m)\cup\{m\}}(m(m\downarrow,1)) \vdash \{\langle P^*\rangle \widehat{F}(n) : n \in AN(m)\} \cup \{\langle P^*\rangle \widehat{F}(m)\}$$

and by Lemmas 4.14 and 4.19 we can prove it from $S(m(m\downarrow,1))$. □

Let us summarise the development of this section in the completeness theorem.

Theorem 4.24 (Completeness) *For any unsatisfiable formula φ_0, the sequent $\varphi_0 \vdash$ is provable.*

Proof
By Proposition 3.6 we can assume that φ_0 is positive and guarded. For this formula we construct an automaton $\mathcal{A}_{\varphi_0}$ as described in Section 4.2. From it we obtain a tree T_{φ_0} defined in Section 4.3. Then we associate a sequent $S(n)$ with every node n of T_{φ_0} in a way described in Definition 4.16. By Lemma 4.17 we know that all the leaves of T_{φ_0} have associated provable sequents. Using induction on the height of a node and Lemmas 4.21, 4.22 and 4.23 we show that the sequent associated with the root of T_{φ_0} is provable. But it is just $\varphi_0 \vdash$. □

5 Conclusions

We have presented a finitary proof system for the propositional μ-calculus and proved its completeness. The system is stronger than original Kozen's system. We think that this is an advantage of this system as it is easier to prove facts in it. Of course original system is so natural that its completenes is still a very interesting question.

The presented system has also this advantage that it is closely connected with a decision procedure for the logic. This shows a possibility of integrating the two in the way we now describe.

Essentially the only know method for checking satisfiability of the μ-calculus formulas comes from the tableau method and the use of automata theory [23]. For a given formula the algorithm constructs an automaton over infinite trees which accepts models of the formula. Then, if formula is satisfiable, a model is constructed from an accepting run of the automaton. In [17] it was shown that the process of checking satisfiability can be viewed as a game between two players. The strategy for one player gives a model for a formula. The strategy for the second gives a refutation. Moreover it was show that for a given formula there is an algorithm which in exponential time constructs a model or a refutation. This gives a complete setting for analysing μ-calculus formulas: given a formula the algorithm either gives an example of a model in which the formula holds or gives "proof" that this formula is unsatisfiable. Our completeness proof gives an algorithm of converting a refutation of a formula into a proof of the negation of the formula. Moreover the rules of the system are closely connected with construction of refutations. This means that it may be possible to interfere with the algorithm to help it in what is essentially task of exponential complexity.

As we have shown our completeness proof is an extension of some completeness proof method for modal system K. This method is really two methods put together as it can be also used for model construction. Conversions to ω-automata are general method of model construction for propositional logics of programs. Such conversions can be seen as an extensions of the model construction technique for system K. Our proof technique seams to be "the other side of the coin" for this conversions. One can expect that it should also work for other logics. It would be particularly interesting to apply the method to the temporal μ-calculus.

References

[1] J.R. Brunch, E.M. Clarke, K.L. McMillan, D.L. Dill, and L.J. Hwang. Symbolic model checking: 10^{20} states and beyond. *Information & Computation*, 98(2):142–170, 1992.

[2] E. Allen Emerson. Automata, tableaux and temporal logic. In *Colledge Conference on Logic of Programs*, volume 193 of *LNCS*. Springer-Verlag, 1985.

[3] E. Allen Emerson and C.L.Lei. Efficient model checking in fragments of propositional mu-calculus. In *First IEEE Symp. on Logic in Computer Science*, pages 267–278, 1986.

[4] E. Allen Emerson and Charanjit S. Jutla. The complexity of tree automata and logics of programs. In *29th IEEE Symp. on Foundations of Computer Science*, 1988.

[5] E.Allen Emerson and J.Y. Halpern. Decision procedures and expressiveness in the temporal logic of branching time. *Journal of Computer and System Sciences*, 30(1):1–24, 1985.

[6] E.Allen Emerson and C.S. Jutla. Tree automata, mu calculus and determinacy. In *Proc. FOCS 91*, 1991.

[7] M.J. Fisher and R.E. Ladner. Propositional modal logic of programs. In *9th ACM Ann. Aymp. on Theory of Computing*, pages 286–294, 1977.

[8] D. Gabbay. Axiomatizations of logics of programs. Unpublished manuscript, Bar-Ilan Univ., 1977.

[9] D. Harel, D Kozen, and R. Parikh. Process logic: Expressiveness, decidability and completeness. *Journal of Computer and System Sciences*, 25:144–201, 1982.

[10] David Harel. Dynamic logic. In *Handbook of Philosophical Logic Vol II*, pages 497–604. D.Reidel Publishing Company, 1984.

[11] P.M.W. Knijnenburg and J. van Leeuwen. On models for propositional dynamic logic. *Theoretical Computer Science*, 91:181–203, 1991.

[12] Dexter Kozen. Results on the propositional mu-calculus. *Theoretical Computer Science*, 27:333–354, 1983.

[13] Dexter Kozen. A finite model theorem for the propositional μ-calculus. *Studa Logica*, 47(3):234–241, 1988.

[14] Dexter Kozen and R. Parikh. An elementary proof of the completeness of the PDL. *Theoretical Computer Science*, 14:113–118, 1981.

[15] Dexter Kozen and R.J.Parikh. A decision procedure for the propositional mu-calculus. In *Second Workshop on Logics of Programs*, 1983.

[16] Dexter Kozen and Jerzy Tiuryn. Logics of programs. In J.van Leeuven, editor, *Handbook of Theoretical Computer Science Vol.B*, pages 789–840. Elsvier, 1990.

[17] D. Niwiński and I. Walukiewicz. Games for μ-calculus. Technical Report TR 94-03(192), Institute of Informatics, Warsaw Univeristy, February 1994.

[18] Damian Niwiński. Fixed points vs. infinite generation. In *Proc. 3rd. IEEE LICS*, pages 402–409, 1988.

[19] R. Parikh. The completeness of propositional dynamic logic. In *Proc 7th Symp on Mathematical Foundations of Computer Science*, volume 64 of *LNCS*, pages 403–415, 1978.

[20] Shmuel Safra. On the complexity of ω-automata. In *29th IEEE Symp. on Foundations of Computer Science*, 1988.

[21] Colin P. Stirling and David J. Walker. Local model checking in the modal mu-calculus. In *International Joint Conference in Theory and Practice of Software Development*, volume 351 of *LNCS*, pages 369–382. Springer-Verlag, 1989.

[22] C.S. Stirling. Modal and temporal logics. In S.Abramsky, D.Gabbay, and T.Maibaum, editors, *Handbook of Logic in Comuter Science*, pages 477–563. Oxford University Press, 1991.

[23] Robert S. Street and E. Allan Emerson. An automata theoretic procedure for the propositional mu-calculus. *Information & Computation*, 81:249–264, 1989.