The Proof Theory of Modal-μ Logics

J. Aldwinckle J.R.B Cockett

August 20, 2001

1 Overview

This paper presents a proof theoretic view for modal logics which are used for reasoning about concurrent systems. We start with a novel presentation of the modal-μ logic and its completeness with respect to regular labeled trees (up to bisimulation). We show the equivalence of this logic to the Kozen system, and thus prove the completeness of the Kozen system confirming a published result [11].

Finally, we discuss a variant of the above logic, modal circuit logic, which allows the economical expression of independent actions which are frequently present in real problems. This logic is also decidable and has a complete semantics in circuits.

2 Modal Logic

The modal logic, also commonly known as Hennessy-Milner Logic [7] was extended by the addition of fixed points by Kozen [8]. The result is often called the modal μ-logic [1, 10] which may be presented as a Gentzen style inference system.

Terms of the logic are defined by:

$$P := \bigvee\{P_i \mid i \in I\} \ \mid\ \bigwedge\{P_i \mid i \in I\} \ \mid\ [\mathcal{A}]\mathcal{P} \ \mid\ \langle\mathcal{A}\rangle\mathcal{P} \ \mid\ \mathcal{X} \ \mid\ \mu\mathcal{X}.\mathcal{P} \ \mid\ \nu\mathcal{X}.\mathcal{P}$$

where $\mathcal{A}$ is a set of actions and $\mathcal{X}$ is a set of variables. The actions $\mathcal{A}$ are used as guards over the proposition P in $\langle\mathcal{A}\rangle\mathcal{P}$ (possibly P) and $[\mathcal{A}]\mathcal{P}$ (necessarily P).

We present this logic using a notion of "circular proofs" inspired by Santocanale [9]. This proof system allows one to establish the basic proof theoretic properties such as cut-elimination in a relatively straightforward manner. From this, the decidability of the logic and generation of counter-examples follow quite easily. Similar rules were introduced in [11] and for linear functors, and whence for linear modal logic in [3].

The construction and destruction rules for the fixed point rules in the table below will look familiar, however the other two rules (called μ-Fold and ν-Fold) need some explanation. The rules create boxes, a fresh type variable Z whose scope is contained inside the proof box, and an axiom $(Z, \Gamma \vdash \Delta)$ which may be freely used inside the box any number of times. The axiom is called the induction assumption. The variable Z may also be used as the subject of further folds, such a usage is called a regeneration, with the axiom also available for use in the proof boxes so created.

$$\frac{}{x \vdash x}\ \text{id}$$

$$\frac{\Gamma \vdash \Delta}{\Gamma, \Gamma_1 \vdash \Delta}\ \text{L-weak} \qquad \frac{\Gamma \vdash \Delta}{\Gamma \vdash \Delta, \Delta_1}\ \text{R-weak}$$

$$\frac{\Gamma_1, \Gamma \vdash \Delta}{\bigwedge \Gamma_1, \Gamma \vdash \Delta}\ \text{L-}\bigwedge \qquad \frac{\Gamma \vdash \Delta_1, \Delta}{\Gamma \vdash \bigvee \Delta_1, \Delta}\ \text{R-}\bigvee$$

$$\frac{\{A, \Gamma \vdash \Delta\}_{A \in \Gamma_1}}{\bigvee \Gamma_1, \Gamma \vdash \Delta}\ \text{L-}\bigvee \qquad \frac{\{\Gamma \vdash A, \Delta\}_{A \in \Delta_1}}{\Gamma \vdash \bigwedge \Delta_1, \Delta}\ \text{R-}\bigwedge$$

$$\frac{P_1, ...P_n, P \vdash Q_1, ...Q_m}{[a]P_1, ...[a]P_n, \langle a \rangle P \vdash \langle a \rangle Q_1, ...\langle a \rangle Q_m}\ \text{Poss} \qquad \frac{P_1, ...P_n \vdash Q, Q_1, ...Q_m}{[a]P_1, ...[a]P_n \vdash [a]Q, \langle a \rangle Q_1, ...\langle a \rangle Q_m}\ \text{Box}$$

$$\frac{\Gamma \vdash \Delta, x \quad \Gamma', x \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\ \text{cut}$$

$$\frac{f(\nu x.f(x)), \Gamma \vdash \Delta}{\nu x.f(x), \Gamma \vdash \Delta}\ \nu - \text{Con} \qquad \frac{\Gamma \vdash f(\mu x.f(x)), \Delta}{\Gamma \vdash \mu x.f(x), \Delta}\ \mu - \text{Con}$$

$$\begin{array}{|c|} \hline Z = \mu x.f(x) \ \mid \ Z, \Gamma \vdash \Delta \\ \hline \\ f(Z), \Gamma \vdash \Delta \\ \hline \end{array} \qquad \begin{array}{|c|} \hline Z = \nu x.f(x) \ \mid \ \Gamma \vdash Z, \Delta \\ \hline \\ \Gamma \vdash f(Z), \Delta \\ \hline \end{array}$$
$$\mu x.f(x), \Gamma \vdash \Delta \qquad\qquad\qquad \Gamma \vdash \nu x.f(x), \Delta$$

Circular Inference Rules

2.0.1 Rolling

Below is a proof of the rolling identity, $f(\mu x.g(f(x))) = \mu x.f(g(x))$, which requires a proof for each direction of the identity:

$$\begin{array}{|c|} \hline Z = \mu x.\langle b \rangle \langle a \rangle x \ \mid \ Z \vdash \langle b \rangle \mu x.\langle a \rangle \langle b \rangle x \\ \hline \\ \dfrac{\dfrac{\dfrac{\dfrac{}{Z \vdash \langle b \rangle \mu x.\langle a \rangle \langle b \rangle x}\ \text{A(Z)}}{\langle a \rangle Z \vdash \langle a \rangle \langle b \rangle \mu x.\langle a \rangle \langle b \rangle x}\ \text{Poss}}{\langle a \rangle Z \vdash \mu x.\langle a \rangle \langle b \rangle x}\ \mu\text{-Con}}{\langle b \rangle \langle a \rangle Z \vdash \langle b \rangle \mu x.\langle a \rangle \langle b \rangle x}\ \text{Poss} \\ \hline \end{array}$$
$$\dfrac{\dfrac{\mu x.\langle b \rangle \langle a \rangle x \vdash \langle b \rangle \mu x.\langle a \rangle \langle b \rangle x}{\langle a \rangle \mu x.\langle b \rangle \langle a \rangle x \vdash \langle a \rangle \langle b \rangle \mu x.\langle a \rangle \langle b \rangle x}\ \text{Poss}}{\langle a \rangle \mu x.\langle b \rangle \langle a \rangle x \vdash \mu x.\langle a \rangle \langle b \rangle x}\ \mu\text{-Con}$$

$$\begin{array}{|c|} \hline Z = \mu x.\langle a \rangle \langle b \rangle x \ \mid \ Z \vdash \langle a \rangle \mu x.\langle b \rangle \langle a \rangle x \\ \hline \\ \dfrac{\dfrac{\dfrac{\dfrac{}{Z \vdash \langle a \rangle \mu x.\langle b \rangle \langle a \rangle x}\ \text{A(Z)}}{\langle b \rangle Z \vdash \langle b \rangle \langle a \rangle \mu x.\langle b \rangle \langle a \rangle x}\ \text{Poss}}{\langle b \rangle Z \vdash \mu x.\langle b \rangle \langle a \rangle x}\ \mu\text{-Con}}{\langle a \rangle \langle b \rangle Z \vdash \langle a \rangle \mu x.\langle b \rangle \langle a \rangle x}\ \text{Poss} \\ \hline \end{array}$$
$$\mu x.\langle a \rangle \langle b \rangle x \vdash \langle a \rangle \mu x.\langle b \rangle \langle a \rangle x$$

2.0.2 Diagonalisation

The Diagonalisation identity, $\mu x.\mu y.f(x,y) = \mu x.f(x,x)$, also requires two proofs; we present one direction below. Although we use a general function symbol the logic does not actually support such and so f stands for an expression such as $f(x,y) = \langle a \rangle x \bigwedge [b] y$ for which monotonicity (which we have as one step) requires a proof. Note the use of regeneration in this proof.

$$
\begin{array}{|c|}
\hline
X = \mu x.f(x,x) \;\mid\; X \vdash \mu x.\mu y.f(x,y) \\
\hline
\dfrac{\dfrac{\dfrac{}{X \vdash \mu x.\mu y.f(x,y)}\,\text{A(X)} \qquad
\begin{array}{|c|}
\hline
X' = X \;\mid\; X' \vdash \mu y.f(\mu x.\mu y.f(x,y),y) \\
\hline
\dfrac{\dfrac{\dfrac{}{X' \vdash \mu x.\mu y.f(x,y)}\,\text{A(X)} \quad \dfrac{}{X' \vdash \mu y.f(\mu x.\mu y.f(x,y),y)}\,\text{A(X')}}{f(X',X') \vdash f(\mu x.\mu y.f(x,y), \mu y.f(\mu x.\mu y.f(x,y),y))}\,\text{Mon}}{f(X',X') \vdash \mu y.f(\mu x.\mu y.f(x,y),y)}\,\mu\text{-Con} \\
\hline
\end{array}
\atop X \vdash \mu y.f(\mu x.\mu y.f(x,y),y)}{\dfrac{f(X,X) \vdash f(\mu x.\mu y.f(x,y), \mu y.f(\mu x.\mu y.f(x,y),y))}{f(X,X) \vdash \mu y.f(\mu x.\mu y.f(x,y),y)}\,\mu\text{-Con}}\,\text{Mon}}{f(X,X) \vdash \mu x.\mu y.f(x,y)}\,\mu\text{-Con} \\
\hline
\end{array}
\atop \mu x.f(x,x) \vdash \mu x.\mu y.f(x,y)
$$

2.1 Comparison with other Fixed Point Rules

These rules may be compared to the Fixed Point rules variously attributed to Park and Kozen [8].

$$
\frac{f(p) \leq p}{\mu x.f(x) \leq p} \qquad \frac{p \leq f(p)}{p \leq \nu x.f(x)}
$$

The circular rules are generally more powerful. Thus, a fixed point proof can easily be transformed to a circular proof. The translation is given below.

$$
\frac{f(\bigvee \Delta) \vdash \Delta}{\mu x.f(x) \vdash \Delta}\,\mu \quad := \quad
\begin{array}{|c|}
\hline
X = \mu x.f(x) \;\mid\; X \vdash \Delta \\
\hline
\dfrac{\dfrac{\dfrac{\dfrac{}{X \vdash \Delta}\,\text{A(X)}}{X \vdash \bigvee \Delta}\,\text{R-}\bigvee}{f(X) \vdash f(\bigvee \Delta)}\,\text{Mon} \qquad f(\bigvee \Delta) \vdash \Delta}{f(X) \vdash \Delta}\,\text{Cut} \\
\hline
\end{array}
\atop \mu x.f(x) \vdash \Delta
$$

As circular proofs allow recursive proofs to be completed in context, to translate in the reverse direction is more difficult. In general, one requires some aspect of division (or negation) to complete the translation. The proof of equivalence consists of a translation which has the effect of replacing complex recursive patterns by simpler patterns which eventually can be expressed by the Fixed Point rules. Below we give an example of how a regeneration can be removed.

$$
\dfrac{\begin{array}{|c}
\hline
X = \mu x.f(x) \;\mid\; X,\Gamma \vdash \Delta \\
\hline
\dfrac{\dfrac{X,\Gamma \vdash \Delta \qquad \dfrac{\begin{array}{|c}
\hline
X' = X \;\mid\; X',\Gamma \vdash \Delta' \\
\hline
\dfrac{\dfrac{X',\Gamma \vdash \Delta \quad X',\Gamma' \vdash \Delta'}{\Pi_2}}{f(X'),\Gamma \vdash \Delta}
\end{array}}{X,\Gamma' \vdash \Delta'}}{\Pi_1}}{f(X),\Gamma \vdash \Delta}
\end{array}}{\mu x.f(x),\Gamma \vdash \Delta}
\qquad \Longrightarrow
$$

$$
\dfrac{\dfrac{\dfrac{\begin{array}{|c}
\hline
X = \mu x.f(x) \;\mid\; X \vdash \Gamma \multimap \Delta \bigwedge \Gamma' \multimap \Delta' \\
\hline
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{X \vdash \Gamma \multimap \Delta \bigwedge \Gamma' \multimap \Delta'}{X \vdash \Gamma \multimap \Delta}}{\Gamma, X \vdash \Delta} \quad \dfrac{\dfrac{X \vdash \Gamma \multimap \Delta \bigwedge \Gamma' \multimap \Delta'}{X \vdash \Gamma' \multimap \Delta'}}{\Gamma', X \vdash \Delta'}}{\Pi_1}}{\Gamma, f(X) \vdash \Delta}}{f(X) \vdash \Gamma \multimap \Delta} \quad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{X \vdash \Gamma \multimap \Delta \bigwedge \Gamma' \multimap \Delta'}{X \vdash \Gamma \multimap \Delta}}{\Gamma, X \vdash \Delta} \quad \dfrac{\dfrac{X \vdash \Gamma \multimap \Delta \bigwedge \Gamma' \multimap \Delta'}{X \vdash \Gamma' \multimap \Delta'}}{\Gamma', X \vdash \Delta'}}{\Pi_2}}{\Gamma', X \vdash \Delta'}}{f(X) \vdash \Gamma' \multimap \Delta'}}{f(X) \vdash \Gamma \multimap \Delta \bigwedge \Gamma' \multimap \Delta'}
\end{array}}{\mu x.f(x) \vdash \Gamma \multimap \Delta \bigwedge \Gamma' \multimap \Delta'}}{\mu x.f(x) \vdash \Gamma \multimap \Delta}}{\Gamma, \mu x.f(x) \vdash \Delta}
$$

2.2 Cut Elimination

Cut elimination procedures work by transforming a Cut rule whose premises consist of a pair of proof rules into another proof where the Cut has been moved up the proof tree. Applied inductively, this results in a proof where all cuts have either been eliminated or have been moved to the leaves of the proof tree. The circular rules introduce a complication since cuts can unwind the circular proofs; see table 1. It is necessary to prove that this unwinding can be terminated. Note that while the proof has increased in size, a new induction assumption has been added, $\Gamma, \Gamma' \vdash Z, \Delta, \Delta'$. We may argue that the size of the proof is finite as only subterms of the original sequent are being used: eventually an induction assumption will recur, and thus, the proof can be terminated.

2.3 Decision Procedure

A decision procedure can be derived by showing that proofs may be put into normal forms. For example, if the L-AND rule is applicable, then it can always be applied first.

Completeness of the Logic is shown by generating a counter example from failed proofs which are in a normal form. These failed proofs are regular trees which are also CCS style agents [7].

The completeness of this logic together with the equivalence of this proof system with Kozen's (with the addition of negation) confirms a result of Walukiewicz [11], that Kozen's system is complete.

$$\dfrac{\dfrac{\Pi}{\Gamma \vdash x, \Delta} \qquad \dfrac{\dfrac{\dfrac{\overline{Z = \nu x.f(x) \;\mid\; x, \Gamma' \vdash Z, \Delta'}}{\Pi'[Z]}}{x, \Gamma' \vdash f(Z), \Delta'}}{x, \Gamma' \vdash \nu x.f(x), \Delta'}}{\Gamma, \Gamma' \vdash \nu x.f(x), \Delta, \Delta'}\;\text{Cut} \quad \Longrightarrow \quad \dfrac{\overline{Z = \nu x.f(x) \;\mid\; \Gamma, \Gamma' \vdash Z, \Delta, \Delta'} \qquad \dfrac{\dfrac{\Pi}{\Gamma \vdash x, \Delta} \qquad \dfrac{\dfrac{\dfrac{\overline{Z' = Z \;\mid\; x, \Gamma' \vdash Z', \Delta'}}{\Pi'[Z']}}{x, \Gamma' \vdash Z', \Delta'}}{\dfrac{\Pi'[Z]}{x, \Gamma' \vdash f(Z), \Delta'}}}{\Gamma, \Gamma' \vdash f(Z), \Delta, \Delta'}\;\text{Cut}}{\Gamma, \Gamma' \vdash \nu x.f(x), \Delta, \Delta'}$$

Table 1: Cut Elimination Example

3 The modal μ-circuit logic

An extension to the logic permits the expression of dependent actions [5]; this is of practical importance as it expresses the dependence of actions exhibited by petri nets. Note that the logic remains essentially a propositional modal logic, however with the ability to express dependencies. This makes the logic more elaborate, but also more expressive.

3.1 Grammar for the Logic

The logic is extended by permitting actions to accept inputs from, and send outputs to, other actions. The actions communicate via wires; each of these wires has a name and a type. Just as actions have inputs and outputs, so must both variables and expressions in the logic. Thus, in an expression such as $\langle a \rangle P$, it is possible to specify whether all of P's inputs come from a's outputs (as in the modal μ-logic) or whether some of its inputs come from the environment.

A circuit logic expression is generated by:

- if V is a variable, and $\{x\}$ is a non repeating wire list which matches the input type of V, and $\{y\}$ is a non repeating wire list whose types match the output type of V, then $\{x\}V\{y\}$ is a circuit logic expression.

- If $\{\{w\}P_i\{z\}\}$ is a set of circuit logic expressions, all with the same input and output wires, then $\bigwedge\{\{w\}P_i\{z\}\}$ and $\bigvee\{\{w\}P_i\{z\}\}$ are circuit logic expressions of the same input and output wires as their constituents.

- if A is an action, and $\{x\}$ is a non repeating wire list which matches the input type of A, and $\{y\}$ is a non repeating wire list whose types match the output type of A, and $\{w\}P\{z\}$ is a circuit logic expression then the compositions $\langle\{x\}A\{y\}\rangle\{w\}P\{z\}$ and $[\{x\}A\{y\}]\{w\}P\{z\}$ which must satisfy the circuit criteria below, are circuit logic expressions.

Compositions are subject to the following circuit criteria:

- types of the composed ports must match;

- the composition must be linear; for each output port of A there must be precisely one matching input port in P. Output wires from A are not permitted to escape P.

- the composition must be sequential; A is not permitted to use P's outputs in its inputs, and P is not permitted to use A's input ports
- the composition must not introduce variable name clash; inputs of P which are not composed with A must not have the same name as A's inputs, and similarly for A's outputs and B's outputs. This can always be avoided with α-renaming.
- the input type of the composition is the union of the input type of the guard with the inputs of the expression which are not captured by the guards's outputs.

Modalities may also be applied on the right of a proposition (output guards) as in: $P\langle\{x\}A\{y\}\rangle$ and $P[\{x\}A\{y\}]$, subject to the the same constraints as above. This abstract will not consider right modalities further.

Thus, the term

$$\langle\{x\}A\{y\}\rangle\bigwedge(\langle\{y\}B\{z\}\rangle\{z\}P\{\},\langle\{y\}C\{w\}\rangle\{w\}Q\{\})$$

satisfies the linearity condition as there is precisely one matching input port which matches A's output port y; this port belongs to the conjunction. The term

$$\langle\{x\}A\{y\}\rangle\langle\{x\}B\{z\}\rangle\{z\}P$$

violates the circuit criteria, as $\langle\{x\}B\{z\}\rangle\{z\}P$ does not use the y wire and the wire x introduces variable name clash.

3.2 Inference Rules

There is a change in the format of sequents from the HML logic; previously sequents were represented as $\Gamma\vdash\Delta$ where both Γ and Δ were lists of formulae. For the circuit logic, a sequent is

$$I \mid \Psi \mid \Gamma\vdash\Delta \mid \Upsilon \mid O$$

where, as before both Γ and Δ are lists of formulae. The input wires and output wires of these formulae are captured in the lists I, and O. As guards are removed from formulae, they are added to the lists Ψ and Υ, where prefix guards are put in Ψ, and postfix guards are put in Υ.

Side conditions apply to the rules to ensure that all of the terms remain well typed.

The L-Poss and R-Box rules have the requirement that the variables y_f is a set of fresh variables; both the output of the guard and input of the proposition in the premise are substituted with these variables. Conversely, in the L-Box and R-Poss rules, the proposition P is bound with the existing set of output variables from the guard $\{x\}A\{y\}$.

The L-Guard rule requires that A must be the only guard which uses x as input, and no term in either Γ or Δ may use inputs in x. This rule turns a circuit with I,y inputs into one with I,x inputs. The L-Guard-W rule requires that there can be no term in either Γ or Δ which uses y as input wires, and no guard in Ψ which uses y as input wires.

The Right modal rules are dual to the Left modal rules; released actions are put into Υ instead of Ψ. Other rules, such as the $\bigwedge$, $\bigvee$, Weak, Id and Cut rules are the same as the Circular Inference Rules.

3.3 Some Sample Proofs

3.3.1 $\langle A\rangle\langle B\rangle P=\langle B\rangle\langle A\rangle P$ for Independent A and B

A simple consequence of the logic is that for independent actions A and B, $\langle A\rangle\langle B\rangle P=\langle B\rangle\langle A\rangle P$. Fresh variables will be subscripted with "f" for clarity.

$$\frac{I \mid \Psi, \{x\}A\{y_f\} \mid \{y_f\}P, \Gamma \vdash \Delta \mid \Upsilon \mid O}{I \mid \Psi \mid \langle \{x\}A\{y\} \rangle \{y\}P, \Gamma \vdash \Delta \mid \Upsilon \mid O}\ \text{L-Poss}$$

$$\frac{I \mid \Psi, \{x\}A\{y\} \mid \{y\}P, \Gamma \vdash \Delta \mid \Upsilon \mid O}{I \mid \Psi, \{x\}A\{y\} \mid [\{x\}A\{y'\}]\{y'\}P, \Gamma \vdash \Delta \mid \Upsilon \mid O}\ \text{L-Box}$$

$$\frac{I \mid \Psi, \{x\}A\{y_f\} \mid \Gamma \vdash \Delta, \{y_f\}P \mid \Upsilon \mid O}{I \mid \Psi \mid \Gamma \vdash \Delta, [\{x\}A\{y\}]\{y\}P \mid \Upsilon \mid O}\ \text{R-Box}$$

$$\frac{I \mid \Psi, \{x\}A\{y\} \mid \Gamma \vdash \Delta, \{y\}P \mid \Upsilon \mid O}{I \mid \Psi, \{x\}A\{y\} \mid \Gamma \vdash \Delta, \langle \{x\}A\{y'\} \rangle \{y'\}P \mid \Upsilon \mid O}\ \text{R-Poss}$$

$$\frac{I, y \mid \Psi \mid \Gamma \vdash \Delta \mid \Upsilon \mid O}{I, x \mid \Psi, \{x\}A\{y\} \mid \Gamma \vdash \Delta \mid \Upsilon \mid O}\ \text{L-Guard}$$

$$\frac{I \mid \Psi \mid \Gamma \vdash \Delta \mid \Upsilon \mid O}{I \mid \Psi, \{x\}A\{y\} \mid \Gamma \vdash \Delta \mid \Upsilon \mid O}\ \text{L-Guard-W}$$

Typed Left Modal Circuit Logic Rules

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{v_f, x_f \mid \mid \{v_f, x_f\}P \vdash \{v_f, x_f\}P \mid \mid}\ \text{ID}}{u, w \mid \{u\}A\{v_f\}, \{w\}B\{x_f\} \mid \{v_f, x_f\}P \vdash \{v_f, x_f\}P \mid \mid}\ \text{L-Guard}}{u, w \mid \{u\}A\{v_f\}, \{w\}B\{x_f\} \mid \{v_f, x_f\}P \vdash \langle \{u\}A\{v\} \rangle \{v, x_f\}P \mid \mid}\ \text{R-Poss}}{u, w \mid \{u\}A\{v_f\}, \{w\}B\{x_f\} \mid \{v_f, x_f\}P \vdash \langle \{w\}B\{x\} \rangle \langle \{u\}A\{v\} \rangle \{v, x\}P \mid \mid}\ \text{R-Poss}}{u, w \mid \{u\}A\{v_f\} \mid \langle \{w\}B\{x\} \rangle \{v_f, x\}P \vdash \langle \{w\}B\{x\} \rangle \langle \{u\}A\{v\} \rangle \{v, x\}P \mid \mid}\ \text{L-Poss}}{u, w \mid \mid \langle \{u\}A\{v\} \rangle \langle \{w\}B\{x\} \rangle \{v, x\}P \vdash \langle \{w\}B\{x\} \rangle \langle \{u\}A\{v\} \rangle \{v, x\}P \mid \mid}\ \text{L-Poss}$$

3.3.2 $\langle A \rangle [B] P \vdash [B] \langle A \rangle P$

A less expected consequence is that for independent actions A and B, $\langle A \rangle [B] P \vdash [B] \langle A \rangle P$. This can be understood as "if there exists an A such that for all B, P holds, then for all B there exists an A such that P holds". The converse does not hold and has a counter example in relations where both A and B are the empty relation.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{v_f, x_f \mid \mid \{v_f, x_f\}P \vdash \{v_f, x_f\}P \mid \mid}\ \text{ID}}{u, w \mid \{u\}A\{v_f\}, \{w\}B\{x_f\} \mid \{v_f, x_f\}P \vdash \{v_f, x_f\}P \mid \mid}\ \text{L-Guard}}{u, w \mid \{u\}A\{v_f\}, \{w\}B\{x_f\} \mid [\{w\}B\{x\}]\{v_f, x\}P \vdash \{v_f, x_f\}P \mid \mid}\ \text{L-Box}}{u, w \mid \{u\}A\{v_f\}, \{w\}B\{x_f\} \mid [\{w\}B\{x\}]\{v_f, x\}P \vdash \langle \{u\}A\{v\} \rangle \{v, x_f\}P \mid \mid}\ \text{R-Poss}}{u, w \mid \{u\}A\{v_f\} \mid [\{w\}B\{x\}]\{v_f, x\}P \vdash [\{w\}B\{x\}] \langle \{u\}A\{v\} \rangle \{v, x\}P \mid \mid}\ \text{R-Box}}{u, w \mid \mid \langle \{u\}A\{v\} \rangle [\{w\}B\{x\}]\{v, x\}P \vdash [\{w\}B\{x\}] \langle \{u\}A\{v\} \rangle \{v, x\}P \mid \mid}\ \text{L-Poss}$$

3.4 Comparisons with other Logics

The algebra of relations [2] gives a model for this logic, as it allows composition on the right and left as well as division (or Box) on the right and left. The Circuit logic permits only single relations for composition and single relations for division to maintain decidability. It could be extended to unions and compositions of single relations but can not express compositions of intersections $((P \bigcap Q); (R \bigcap S))$ or divisions of intersections $(P \bigcap Q) \multimap R$

It is similar to Guarded Logic [6] but the conditions for the guards are simpler, and typing for the propositions is more restricted.

3.5 Decision Procedure and Counter Examples

The logic supports a cut elimination procedure which is actually a standard translation from predicate logic. Thus, the decision procedure need only consider cut free proofs. As with the decision procedure for the modal μ-logic, a decision procedure is derived by showing that proofs may be put into normal forms. Care must be taken with the L-Box (and dually R-Poss) rule as the $[a]P$ term must be contracted and kept in the premise for possible future use in another L-Box rule with another guard. Weakening of terms or guards is avoided until prior to an application of the ID rule.

The models for the typed logic are typed modal circuits [4]; these have an alternative presentation as petri nets. A typed modal circuit is a formula in the logic restricted to $\langle A \rangle P$, $\bigwedge\{P_i\}$, and variables.

As with the modal μ-logic, these models are generated from failed proofs in a normal form.

4 Conclusions

A set of μ-proof rules were described for the modal μ-logic which use "circular rules" following Santocanale [9]. These rules were shown to be decidable, complete and support cut elimination. The authors have been unable to find a previous account of cut-elimination in the μ-logic. These rules were shown to be equivalent to the Park/Kozen rules (with negation) confirming a completeness result previously published by Walukiewicz.

This logic was extended with "wires" to permit the expression of dependence between actions. Again, the logic supports cut elimination and a decision procedure. In the event of a sequent being unprovable a counter example procedure can generate a circuit counter model. Lack of space precluded discussion of the fixed point rules for the Circuit logic.

References

[1] A. Arnold and D. Niwinski. *Rudiments of μ-calculus*. Elsevier, New York, 2001.

[2] Richard Bird and Oege De Moor. *The Algebra of Programming*. Prentice Hall International Series in Computer Science. Prentice Hall, 1996.

[3] R. Blute, R. Cockett, and R. Seely. The logic of linear functors. *http://euclid.math.mcgill.ca/~rags/*, pages 1 – 25, 2001.

[4] R. Blute, R. Cockett, R. Seely, and C. Trimble. Natural deduction and coherence for weakly distributive categories. *Journal of Pure and Applied Algebra*, 113:229 – 296, 1996.

[5] J. C. Bradfield. Independence: Logics and concurrency. In *CSL 2000*, LNCS. Springer-Verlag, 2000.

[6] E. Grädel and I. Walukiewicz. Guarded Fixed Point Logic. In *Proceedings of 14th IEEE Symposium on Logic in Computer Science LICS '99, Trento*, pages 45–54, 1999.

[7] M. Hennessy and R. Milner. Algebraic laws for nondeterminism and concurrency. *Journal of the ACM*, 32(1):137–161, 1985.

[8] D. Kozen. Results on the propositional mu-calculus. *Theoretical Computer Science*, 27:113–118, 1983.

[9] L. Santocanale. *Sur les μ-treillis libre*. PhD thesis, Université du Québec à Montréal, 1999.

[10] Colin Stirling. Modal and temporal logics. In T.S.E. Maibaum S. Abramsky, Dov M. Gabbay, editor, *Handbook of Logic in Computer Science*, volume Volume 2; Background: Computational Structure, pages 477 – 563. Cambridge University Press, 1992.

[11] I. Walukiewicz. Completeness of Kozen's axiomatisation of the propositional μ-calculus. In *Logic in Computer Science*, pages 14–24. IEEE, 1995.