The Madness of the Modal μ-Calculus

Not your granny's syntax with binding!

Sean Watters

University of Strathclyde

28th Oct 2024

Plan

1. Introduction to the μ-calculus.
    1. Model Checking
    2. Kripke Semantics
    3. Strict Positivity
    4. The Closure (enter the madness)
2. Introduction to well-scoped De Bruijn syntax.
    1. Thinnings
    2. Weakening
    3. Parallel substitution
    4. Defining the closure?
3. A peek at *sublimely-scoped* De Bruijn syntax (if time permits). (It won't).

Logic as a Specification Language (1)

A **logic** is a formal system for making statements of fact. eg: propositional logic, first-order logic, etc.

Traditionally, we define the meaning of formulae relative to some mathematical structure called a **model**. (Model-theoretic semantics).

The **model checking problem** for a logic is the problem of deciding whether, given a formula and model of the logic, the formula is validated in that model.

Model Checking as a **formal verification** technique involves representing the system undergoing verification as a model, and the properties being verified as formulas. As long as the model checking problem for the logic is decidable, we can algorithmically verify that the property is true.

Logic as a Specification Language (2)

Two questions:

1. What is the right notion of model for real-world systems?
2. Which logics (best) let us reason about such models?

Propositional Logic

Given some set A of propositional atoms, for all $a \in \mathrm{At}$, let PL be the set of terms generated by:

$$\varphi := a \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \varphi \Rightarrow \varphi$$

Models are valuation functions $\mathrm{At} \to 2$.

Propositional Logic for Model Checking?

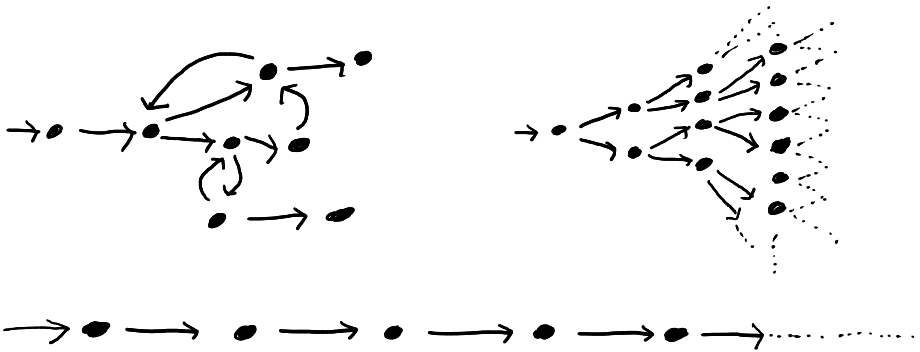
- **The Good**: Model checking problem is decidable.
- **The Bad**: Not very expressive; purely propositional encodings are clunky.

How about first-order logic? Much more expressive, but too expressive! Not decidable!

Kripke Semantics (1)

Key idea: Computers and programs tend to progress in discreet time steps, with their properties potentially changing at each step.

Let's consider models that reflect that!



Kripke Semantics (2)

Definition

A **Kripke model** is a tuple (S, T, V) of a set of states S, a transition relation $T : S \to S \to \text{Prop}$, and a valuation function $V : A \to S \to \text{Prop}$ (given some set A of atomic propositions).

Notice that:

- S and T induce a (possibly infinite) graph-like structure.
- Each state $s \in S$ is equipped with a model of propositional logic, $a \mapsto V\ a\ s$.
- Partial application of T on some state $s \in S$ yields the "is a successor of s" predicate.

Modal Logic (1)

For all propositional atoms $a \in \mathrm{At}$, let ML be the set of terms generated by:

$$\varphi := a \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \varphi \Rightarrow \varphi \mid \Box\varphi \mid \Diamond\varphi$$

Given a Kripke model $\mathcal{M} = (S, T, V)$, let $[\![-]\!] - : ML \to S \to \mathrm{Prop}$, where:

$$\begin{aligned}
[\![a]\!]\ s &:= V\ a\ s \\
[\![\neg\varphi]\!]\ s &:= \neg\ ([\![\varphi]\!]\ s) \\
[\![\varphi \wedge \psi]\!]\ s &:= ([\![\varphi]\!]\ s) \times ([\![\psi]\!]\ s) \\
[\![\varphi \vee \psi]\!]\ s &:= ([\![\varphi]\!]\ s) + ([\![\psi]\!]\ s) \\
[\![\varphi \Rightarrow \psi]\!]\ s &:= ([\![\varphi]\!]\ s) \to ([\![\psi]\!]\ s) \\
[\![\Box\varphi]\!]\ s &:= \forall t \in (T\ s).\ [\![\varphi]\!]\ t \\
[\![\Diamond\varphi]\!]\ s &:= \exists t \in (T\ s).\ [\![\varphi]\!]\ t
\end{aligned}$$

Modal Logic (2)

ML for Model Checking?

- **Good**: Model checking problem is still decidable.
- **Good**: Kripke models are well-suited to representing real systems.
- **Bad**: Expressivity is still limited; can only reason about fixed, finite time steps.

We cannot express, for example:

- φ is always true.
- φ eventually becomes true, in at least one possible future.
- φ remains true at least until ψ becomes true, in all possible futures.

We currently can't reason about infinte or unbounded behaviour.

There's a neat way around that...

The Modal μ-Calculus (1)

For all propositional atoms $a \in \text{At}$ and variable names $x \in \text{Var}$, let μML be the set of terms generated by:

$$\varphi := a \mid \neg a \mid x \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \Box\varphi \mid \Diamond\varphi \mid \mu x.\varphi \mid \nu x.\varphi$$

Given a Kripke model $\mathcal{M}$ and an intepretation of free variables $i : \text{Var} \to S \to \text{Prop}$, let:

$$\begin{aligned} [\![x]\!]_i\ s &:= i\ x\ s \\ [\![\mu x.\ \varphi]\!]_i\ s &:= \mu\ ((U : S \to \text{Prop}) \mapsto [\![\varphi]\!]_{i[x:=U]})\ s \\ [\![\nu x.\ \varphi]\!]_i\ s &:= \nu\ ((U : S \to \text{Prop}) \mapsto [\![\varphi]\!]_{i[x:=U]})\ s \end{aligned}$$

Where $i[x := U]$ denotes updating the mapping of x to U in i.

The Modal μ-Calculus (2)

Syntax:

$$\varphi := a \mid \neg a \mid x \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \Box\varphi \mid \Diamond\varphi \mid \mu x.\varphi \mid \nu x.\varphi$$

Semantics of $\mu x.\ \varphi$:

$$[\![\mu x.\ \varphi]\!]_i := \mu\ (U \mapsto [\![\varphi]\!]_{i[x:=U]})$$

Important notes:

- Not every map has well-defined least/greatest fixpoints. Kleene's Fixpoint Theorem guarantees that this one will, so long as it is monotone.
- $[\![-]\!]$ being strictly positive guarantees that this will be the case.
- This is why the syntax of μML does not allow arbitrary negations or implication.

Fixpoint Unfolding

At the heart of the μ-calculus is the semantic equivalance:

$$\eta x.\ \varphi \ \equiv\ \varphi[\eta x.\varphi\ /\ x]$$

We call $\varphi[\eta x.\varphi\ /\ x]$ the **unfolding** of $\eta x.\ \varphi$.

For example: let $E(p) := \mu x.\ p \vee \Diamond x.$ Then:

$$\begin{aligned} & E(p) \\ \equiv\ & p \vee \Diamond(E(p)) \\ \equiv\ & p \vee \Diamond(p \vee \Diamond(E(p))) \\ \equiv\ & \ldots \end{aligned}$$

This being a *least* fixpoint says that the formula will be validated in *finitely* many unfoldings.

Fixpoint Formula Examples

(corrections in blue pen.)

Formula	Meaning
$\mu x.x$	$\bot$
$\nu x.x$	$\top$
$\nu x.(\Box x \wedge \Diamond \top)$	All paths are infinite.
$\mu x.(\Box x \vee \Box \bot)$	All paths are finite.
$\nu x.(\Diamond x \wedge \Diamond \top)$	There is at least one infinite path.
$\nu x.p \wedge \Box x$	"Always p"
$\mu x.p \vee \Diamond x$	"Possibly p"
$\mu x.p \vee \Box x$	"Eventually p"
$\nu y.\mu z.(\Diamond z \wedge \Box y)$	~~exercise for the reader~~ $\top$ (again)

Could use some propositions to say something more interesting.

The Closure (1)

Definition

The **closure** of a formula φ is the minimal set which contains φ, and is closed under taking unfoldings of fixpoint formulas, and direct subformulas of non-fixpoint formulas.

In other words, it is the minimal set C satisfying:

$$\varphi \in C$$
$$\bigcirc\varphi \in C \Rightarrow \varphi \in C, \text{ where } \bigcirc \in \{\Box, \Diamond\}$$
$$\varphi \star \psi \in C \Rightarrow \varphi \in C \text{ and } \psi \in C, \text{ where } \star \in \{\wedge, \vee\}$$
$$\eta x.\varphi \in C \Rightarrow \varphi[\mu x.\varphi \,/\, x] \in C, \text{ where } \eta \in \{\mu, \nu\}$$

The Closure (2)

The closure encapsulates the semantics of a formula in a syntactic way:

Its graph directly yields an automaton that accepts (possibly infinite) paths through Kripke models that validate the formula.

Some facts about the closure:

1. It is always non-empty (trivial).
2. It is always finite (a bit less trivial).
3. Fixpoint formulae and their unfoldings always have the same closure.
4. If φ and ψ have the same closure, then $\varphi \equiv \psi$. False! Oops.
5. But the other direction does not hold; semantically equivalent formulae may have different closures.
6. Worse: even if $\varphi =_\alpha \psi$, then they may still have different closures (up to α-equivalence). (Wickedness! Heresy!)

Finiteness of the Closure (1)

Theorem

For all φ, the closure of φ is finite.

Intuition: Unfoldings can increase the size of the formula, but the number of new subformulas always decreases.

Finiteness of the Closure (1)

Theorem

For all φ, the closure of φ is finite.

Intuition: Unfoldings can increase the size of the formula, but the number of new subformulas always decreases.

$\alpha := \nu y.\, \mu z.\, (\Diamond z \wedge \Box y)$

Finiteness of the Closure (1)

Theorem

For all φ, the closure of φ is finite.

Intuition: Unfoldings can increase the size of the formula, but the number of new subformulas always decreases.

$\alpha := \nu y.\, \mu z.\, (\Diamond z \wedge \Box y)$

$\beta := \mu z.\, (\Diamond z \wedge \Box \alpha)$

Finiteness of the Closure (1)

Theorem

For all φ, the closure of φ is finite.

Intuition: Unfoldings can increase the size of the formula, but the number of new subformulas always decreases.

$\alpha := \nu y.\, \mu z.\, (\Diamond z \wedge \Box y)$

$\beta := \mu z.\, (\Diamond z \wedge \Box \alpha)$

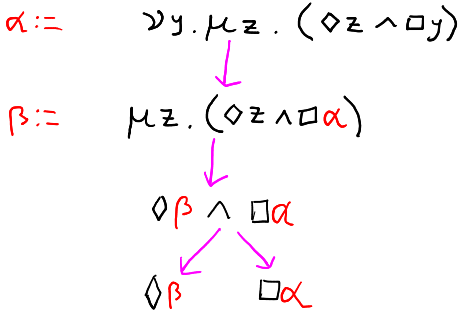
$\Diamond \beta \wedge \Box \alpha$

Finiteness of the Closure (1)

Theorem

For all φ, the closure of φ is finite.

Intuition: Unfoldings can increase the size of the formula, but the number of new subformulas always decreases.

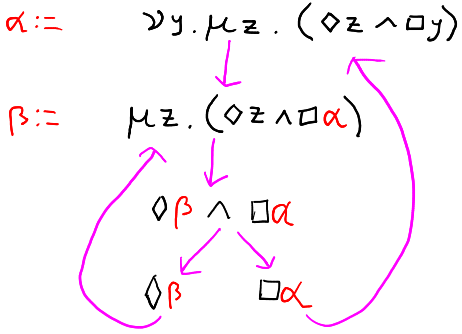


Finiteness of the Closure (1)

Theorem

For all φ, the closure of φ is finite.

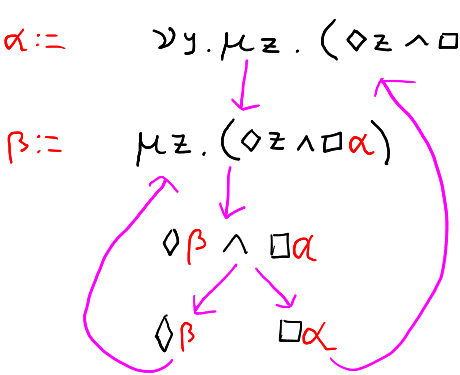
Intuition: Unfoldings can increase the size of the formula, but the number of new subformulas always decreases.



Closures of Unfoldings

Theorem

For any fixpoint formula $\eta x.\varphi$, both it and its unfolding $\varphi[\eta x.\varphi \,/\, x]$ have the same closure.

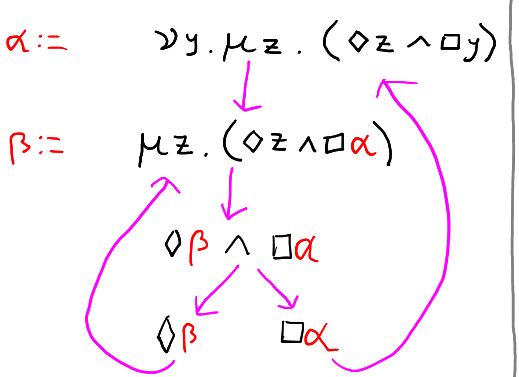


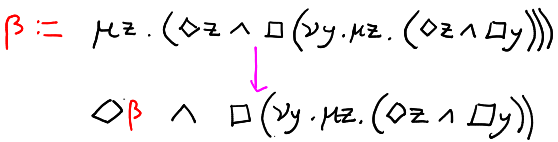
β := $\mu z.(\Diamond z \wedge \Box(\nu y.\mu z.(\Diamond z \wedge \Box y)))$

Closures of Unfoldings

Theorem

For any fixpoint formula $\eta x.\varphi$, both it and its unfolding $\varphi[\eta x.\varphi \,/\, x]$ have the same closure.

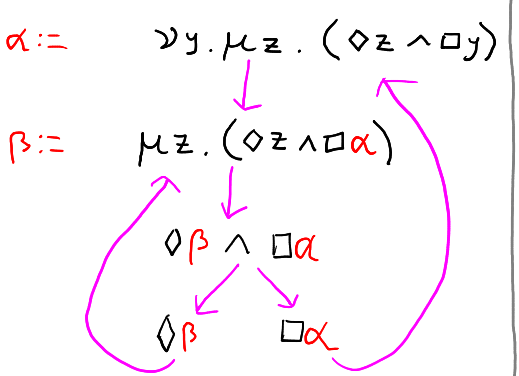


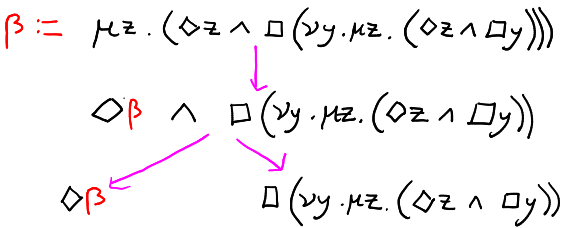


Closures of Unfoldings

Theorem

For any fixpoint formula $\eta x.\varphi$, both it and its unfolding $\varphi[\eta x.\varphi \,/\, x]$ have the same closure.

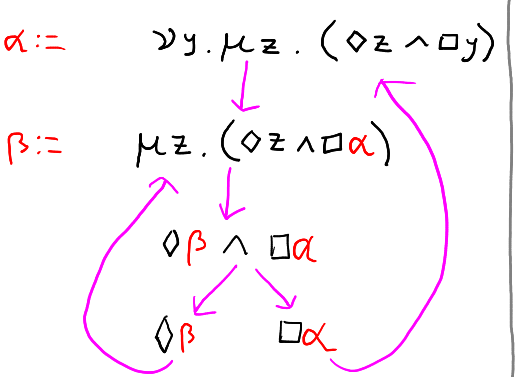


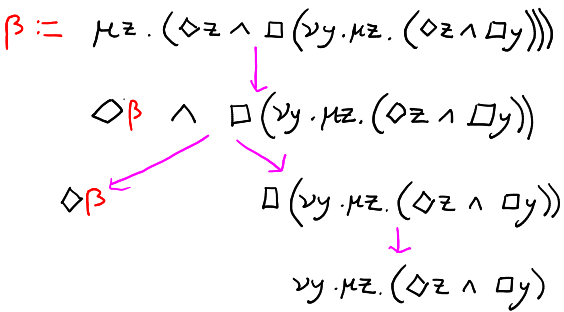


Closures of Unfoldings

Theorem

For any fixpoint formula $\eta x.\varphi$, both it and its unfolding $\varphi[\eta x.\varphi \,/\, x]$ have the same closure.

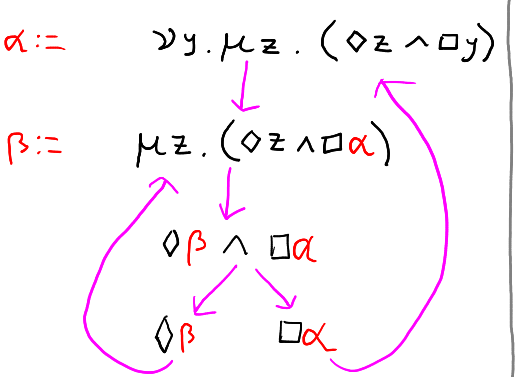


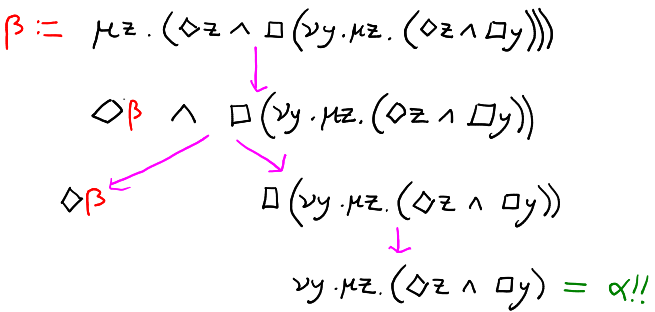


Closures of Unfoldings

Theorem

For any fixpoint formula $\eta x.\varphi$, both it and its unfolding $\varphi[\eta x.\varphi \,/\, x]$ have the same closure.

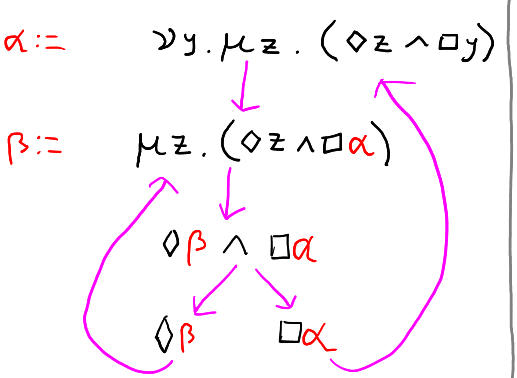


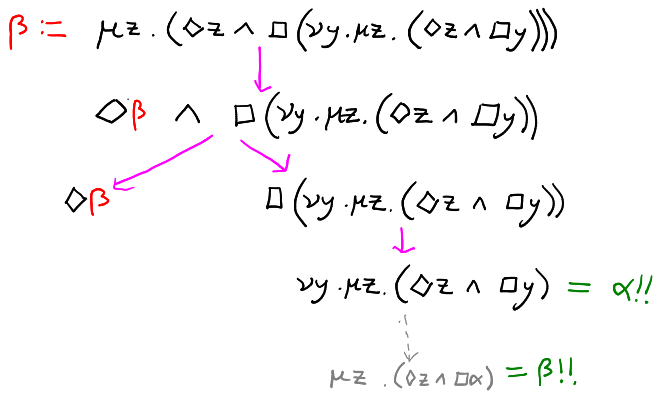


Closures of Unfoldings

Theorem

For any fixpoint formula $\eta x.\varphi$, both it and its unfolding $\varphi[\eta x.\varphi \,/\, x]$ have the same closure.

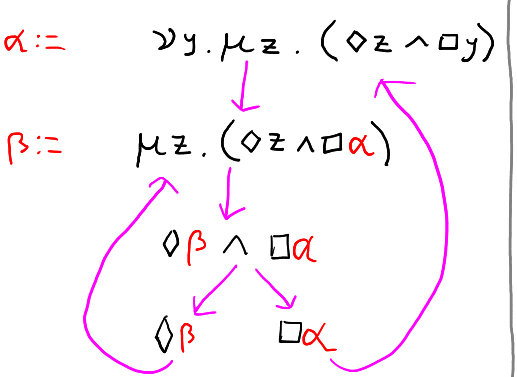


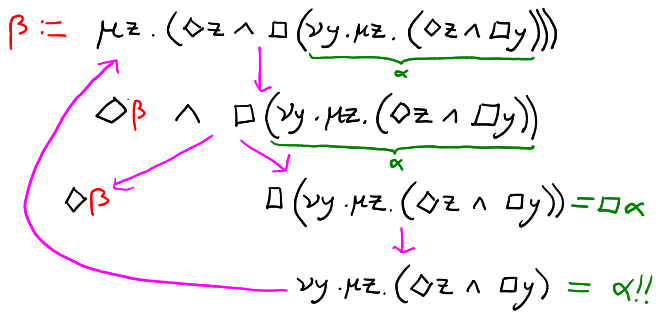


Closures of Unfoldings

Theorem

For any fixpoint formula $\eta x.\varphi$, both it and its unfolding $\varphi[\eta x.\varphi \,/\, x]$ have the same closure.





Instability Under Alpha Equivalence

Theorem

There exist formulas φ and ψ such that $\varphi =_\alpha \psi$, but $C_\varphi \neq C_\psi$.

"That's just not nice!" - Sam Lindley, 2024

$$\beta := \mu z.\,(\Diamond z \wedge \Box(\nu y.\,\mu z.\,(\Diamond z \wedge \Box y)))$$

Instability Under Alpha Equivalence

Theorem

There exist formulas φ and ψ such that $\varphi =_\alpha \psi$, but $C_\varphi \neq C_\psi$.

"That's just not nice!" - Sam Lindley, 2024

$$\beta := \mu z.\,(\Diamond z \wedge \Box(\nu y.\,\mu z.\,(\Diamond z \wedge \Box y)))$$

$$\beta' := \mu x.\,(\Diamond x \wedge \Box(\nu y.\,\mu z.\,(\Diamond z \wedge \Box y)))$$

$$\beta =_\alpha \beta'$$

Instability Under Alpha Equivalence

$$\beta' := \quad \mu x \,.\, (\Diamond x \,.\, \wedge \Box (\nu y \,.\, \mu z \,.\, (\Diamond z \wedge \Box y)))$$

Instability Under Alpha Equivalence

$$\beta' := \quad \mu x \,.\, (\Diamond x \wedge \Box(\nu y \,.\, \mu z \,.\, (\Diamond z \wedge \Box y)))$$

$$\Diamond \beta' \wedge \Box(\nu y \,.\, \mu z \,.\, (\Diamond z \wedge \Box y))$$

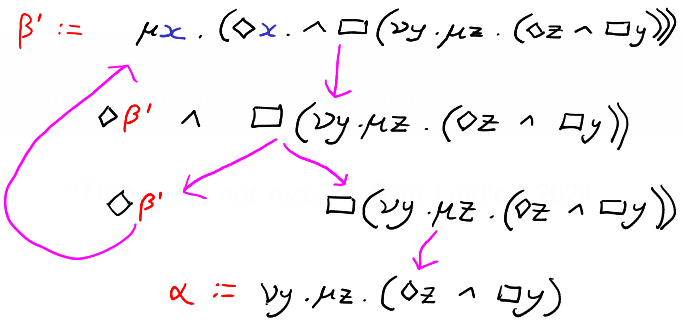
Instability Under Alpha Equivalence

$$\beta' := \mu x.\,(\Diamond x \wedge \Box(\nu y.\,\mu z.\,(\Diamond z \wedge \Box y)))$$

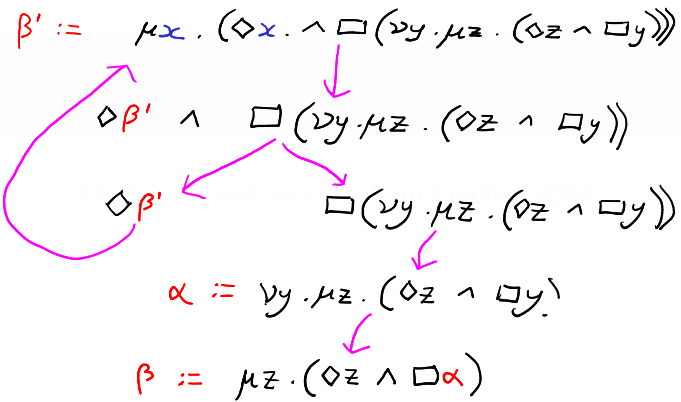
$$\Diamond\beta' \wedge \Box(\nu y.\,\mu z.\,(\Diamond z \wedge \Box y))$$

$$\Diamond\beta' \qquad \Box(\nu y.\,\mu z.\,(\Diamond z \wedge \Box y))$$

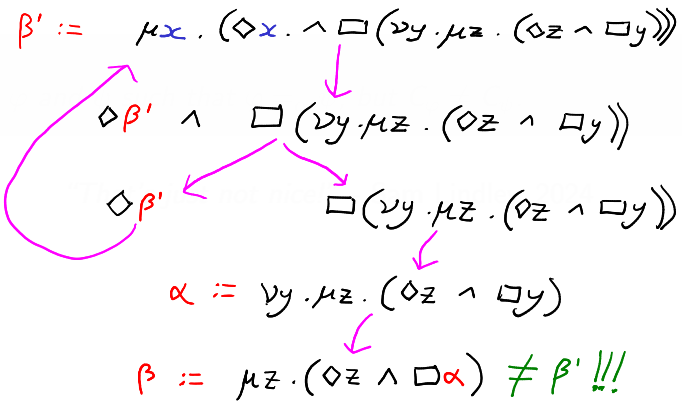
Instability Under Alpha Equivalence



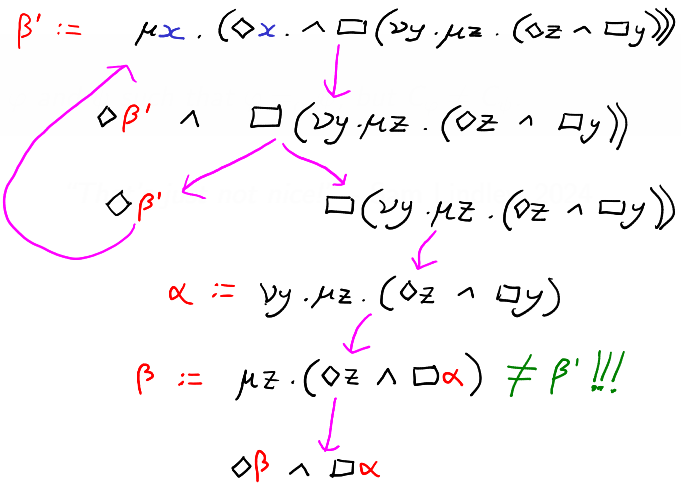
Instability Under Alpha Equivalence



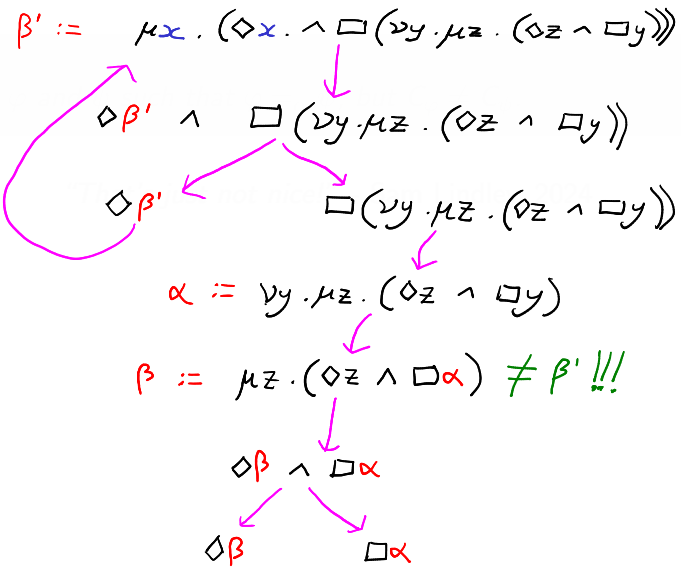
Instability Under Alpha Equivalence



Instability Under Alpha Equivalence



Instability Under Alpha Equivalence



Instability Under Alpha Equivalence