The Expressive Power of the Modal μ-Calculus

Matt Wetmore

November 14, 2014

Introduction

"Started from the bottom now we here"

– Aubrey "Drake" Graham

The setting

We would like to verify the behaviour of our programs

The setting

We would like to verify the behaviour of our programs
Transition systems model the behaviour of programs...

The setting

We would like to verify the behaviour of our programs
Transition systems model the behaviour of programs...
...hence it would be useful to state and verify properties of transition systems

$*$

The setting

Over the latter half of the 20th century, computer scientists devised a number of logics which could be used to express properties of transition systems, such as

The setting

Over the latter half of the 20th century, computer scientists devised a number of logics which could be used to express properties of transition systems, such as

PDL

The setting

Over the latter half of the 20th century, computer scientists devised a number of logics which could be used to express properties of transition systems, such as

- PDL
- CTL

The setting

Over the latter half of the 20th century, computer scientists devised a number of logics which could be used to express properties of transition systems, such as

PDL

CTL

CTL*

The setting

Over the latter half of the 20th century, computer scientists devised a number of logics which could be used to express properties of transition systems, such as

PDL

CTL

CTL*

In 1983, Dexter Kozen introduced the modal μ-calculus $L\mu$, which enhances a simple syntax with powerful fixed-point operators and subsumes the logics above.

Today we will show that $L\mu$ subsumes PDL in particular. The goal is to show that $L\mu$ is **strictly** more expressive than PDL.

$*$

Syntax of $L\mu$

$$\varphi, \psi ::= P$$

Atomic propositions $P \in AP$. Includes $\top$

Syntax of $L\mu$

$$\varphi, \psi ::= P \mid X$$

Atomic propositions $P \in AP$. Includes $\top$
Propositional variables

Syntax of $L\mu$

$$\varphi, \psi ::= P \mid X \mid \varphi \wedge \psi \mid \neg\varphi$$

Atomic propositions $P \in AP$. Includes $\top$
Propositional variables
Boolean stuff

Syntax of $L\mu$

$$\varphi, \psi ::= P \mid X \mid \varphi \wedge \psi \mid \neg\varphi \mid [a]\varphi$$

Atomic propositions $P \in AP$. Includes $\top$
Propositional variables
Boolean stuff
Box modality. a is a label, associated with actions

Syntax of $L\mu$

$$\varphi, \psi ::= P \mid X \mid \varphi \wedge \psi \mid \neg\varphi \mid [a]\varphi \mid \nu X.\varphi(X)$$

Atomic propositions $P \in AP$. Includes $\top$
Propositional variables
Boolean stuff
Box modality. a is a label, associated with actions
Greatest fixed point of φ

$\ast$

Syntax of $L\mu$

We have an additional syntactic constraint on $\varphi(X)$ in $\nu X.\varphi(X)$: X must be free in φ and occur *positively* - in the scope of an even number of negations.

Syntax of $L\mu$

We have an additional syntactic constraint on $\varphi(X)$ in $\nu X.\varphi(X)$: X must be free in φ and occur *positively* - in the scope of an even number of negations.

The other usual operators can be obtained by de Morgan duality:

$$\bot \equiv \neg\top$$

$$\varphi \vee \psi \equiv \neg(\neg\varphi \wedge \neg\psi)$$

$$\langle a\rangle\varphi \equiv \neg[a]\neg\varphi$$

$$\mu X.\varphi(X) \equiv \neg\nu X.\neg\varphi(\neg X)$$

$*$

Semantics of $L\mu$

We can define the semantics of $L\mu$ in terms of states of a transition system TS over a set of states S, where we have a function $D : AP \to 2^S$ mapping atomic propositions to the states at which they hold ($D(\top) = S$). We define $[\![\varphi]\!]$, the set of all states satisfying φ, inductively as follows:

$$[\![P]\!] = D(P)$$
$$[\![\varphi \wedge \psi]\!] = [\![\varphi]\!] \cap [\![\psi]\!]$$
$$[\![\neg\varphi]\!] = S \setminus [\![\varphi]\!]$$
$$[\![[a]\varphi]\!] = \{s \in S \,|\, \forall t \,.\, s \xrightarrow{a} t \implies t \in [\![\varphi]\!]\}$$
$$[\![\langle a\rangle\varphi]\!] = \{s \in S \,|\, \exists t \,.\, s \xrightarrow{a} t \wedge t \in [\![\varphi]\!]\}$$

$*$

Semantics of $L\mu$

If a formula contains a variable X, we interpret $[\![\varphi(X)]\!]$ as a function $T \mapsto [\![\varphi[T/X]]\!]$ mapping sets of states $T \subseteq S$ to an interpretation of φ where all instances of X have been replaced by the states in T. We interpret this mixing of formulas and states like this (for example):

$$s \in [\![\psi \wedge T]\!] \text{ if } s \in [\![\psi]\!] \text{ and } s \in T$$

Semantics of $L\mu$

If a formula contains a variable X, we interpret $[\![\varphi(X)]\!]$ as a function $T \mapsto [\![\varphi[T/X]]\!]$ mapping sets of states $T \subseteq S$ to an interpretation of φ where all instances of X have been replaced by the states in T. We interpret this mixing of formulas and states like this (for example):

$$s \in [\![\psi \wedge T]\!] \text{ if } s \in [\![\psi]\!] \text{ and } s \in T$$

For notational simplicity we will consider formulas of a single variable, and write $[\![\varphi(\psi)]\!]$ to express $[\![\varphi(X)]\!]([\![\psi]\!])$.

$*$

Semantics of $L\mu$

Formulas $\varphi(X)$ that obey the positivity restriction define monotonic functions $[\![\varphi(X)]\!] : 2^S \to 2^S$ on the powerset lattice, which is complete. Hence we can define $[\![\mu X.\varphi(X)]\!]$ and $[\![\nu X.\varphi(X)]\!]$ to be the least and greatest fixed points of $[\![\varphi(X)]\!]$.

$*$

Recursion semantics

Lattice theory tells us that monotone functions f mapping a complete lattice to itself have fixed points, which is how we defined the semantics of the formulas $\nu X.\varphi(X)$ and $\mu X.\varphi(X)$.

Recursion semantics

Lattice theory tells us that monotone functions f mapping a complete lattice to itself have fixed points, which is how we defined the semantics of the formulas $\nu X.\varphi(X)$ and $\mu X.\varphi(X)$.

Furthermore, we may obtain these fixed points by successive iterations of f. For instance, $\mu f = \bigvee_n f^n(\bot)$

Recursion semantics

Lattice theory tells us that monotone functions f mapping a complete lattice to itself have fixed points, which is how we defined the semantics of the formulas $\nu X.\varphi(X)$ and $\mu X.\varphi(X)$.

Furthermore, we may obtain these fixed points by successive iterations of f. For instance, $\mu f = \bigvee_n f^n(\bot)$

Hence the phrase "started from the bottom now we're here"

$$*$$

Recursion semantics

$$\mu f = \bigvee_n f^n(\bot) \quad \rightsquigarrow \quad [\![\mu X.\varphi(X)]\!] = \bigcup_n [\![\varphi^n(\bot)]\!]$$

This iteration will take at most $|S| + 1$ powers of φ to reach the fixed point.

Recursion semantics

$$\mu f = \bigvee_n f^n(\bot) \quad \rightsquigarrow \quad [\![\mu X.\varphi(X)]\!] = \bigcup_n [\![\varphi^n(\bot)]\!]$$

This iteration will take at most $|S| + 1$ powers of φ to reach the fixed point.

$$[\![\bot]\!] \subseteq [\![\varphi(\bot)]\!] \subseteq [\![\varphi(\varphi(\bot))]\!] \subseteq \ldots \subseteq [\![\varphi^n(\bot)]\!] \subseteq \ldots$$

If the fixed point is at some power n, then there is a finite increasing chain of sets of states which satisfy $\mu X.\varphi(X)$.

"μ is finite looping"

$*$

Example

What does this express?

$$\mu X.[a]X$$

Example

What does this express?

$$\mu X.[a]X$$

$$\begin{aligned} [\![[a]\bot]\!] &= \{s \in S \mid \forall t \,.\, s \xrightarrow{a} t \implies t \in [\![\bot]\!] \} \\ &= \text{set of states with no outgoing } a \text{ transitions} \end{aligned}$$

(all a paths are length 0)

Example

What does this express?

$$\mu X.[a]X$$

$$[\![[a]\bot]\!] = \{s \in S \mid \forall t \,.\, s \xrightarrow{a} t \implies t \in [\![\bot]\!]\}$$

$= \text{set of states with no outgoing } a \text{ transitions}$

(all a paths are length 0)

$$[\![[a][a]\bot]\!] = \{s \in S \mid \forall t \,.\, s \xrightarrow{a} t \implies t \in [\![[a]\bot]\!]\}$$

$= \text{set of states whose } a \text{ transitions go to states with no } a \text{ transitions}$

(all a paths are length 1)

Example

What does this express?

$$\mu X.[a]X$$

And so on. If a state s is in $[\![\mu X.[a]X]\!]$, then all a paths starting at s are finite.
We can say $TS \models \varphi$ if every initial state s_0 is in $[\![\varphi]\!]$.
Hence $TS \models \mu X.[a]X$ if TS contains no infinite initial a paths.

$*$

Propositional Dynamic Logic

"I've got a proposition for you..."

– Joseph "Proposition Joe" Stewart

Introduction to PDL

Propositional Dynamic Logic is another modal logic. Labels on modalities like $\langle\alpha\rangle$ and $[\alpha]$ represent (non-deterministic) programs, and we read formulas with these modalities as:

$\langle\alpha\rangle\varphi \quad \mapsto$ "**Some** terminating execution of α ends in a state satisfying φ"

$[\alpha]\varphi \quad \mapsto$ "**Every** execution of α leads to a state satisfying φ"

*

Introduction to PDL

If we give ourselves a set of basic atomic programs $a, b, \ldots$ which go from state to state, we can write more complex programs with the familiar operations in regular expressions.

If α, β are programs, then we can create the following programs:

Introduction to PDL

If we give ourselves a set of basic atomic programs $a, b, \ldots$ which go from state to state, we can write more complex programs with the familiar operations in regular expressions.

If α, β are programs, then we can create the following programs:

$\alpha \cup \beta$: Non-deterministically choose to execute either α or β

Introduction to PDL

If we give ourselves a set of basic atomic programs $a, b, \ldots$ which go from state to state, we can write more complex programs with the familiar operations in regular expressions.

If α, β are programs, then we can create the following programs:

$\alpha \cup \beta$: Non-deterministically choose to execute either α or β

$\alpha;\ \beta$: Sequentially execute α, then β

Introduction to PDL

If we give ourselves a set of basic atomic programs $a, b, \ldots$ which go from state to state, we can write more complex programs with the familiar operations in regular expressions.

If α, β are programs, then we can create the following programs:

$\alpha \cup \beta$: Non-deterministically choose to execute either α or β

$\alpha; \beta$: Sequentially execute α, then β

α^* : Execute α some finite number of times (perhaps 0)

$*$

Syntax of PDL

Formulas in PDL follow the usual syntax

$$\varphi, \psi ::= P \mid \varphi \wedge \psi \mid \neg\varphi \mid [\alpha]\varphi$$

We can obtain $\langle\alpha\rangle\varphi$ and $\varphi \vee \psi$ by taking the de Morgan dual as usual.

Syntax of PDL

Formulas in PDL follow the usual syntax

$$\varphi, \psi ::= P \mid \varphi \land \psi \mid \neg\varphi \mid [\alpha]\varphi$$

We can obtain $\langle\alpha\rangle\varphi$ and $\varphi \lor \psi$ by taking the de Morgan dual as usual.
Formulas express properties of states in transition systems, so we may make judgements such as $s \models \varphi$ for some state s, and extend the satisfaction relation to transition systems, such that $TS \models \varphi$ if every initial state $s_0 \models \varphi$.

$*$

Small Model Property

PDL (like the other logics mentioned earlier) has the **small model property**, which means that if φ is satisfiable, i.e. if there is a transition system TS such that $TS \models \varphi$, then there is a finite transition system TS_{FIN} such that $TS_{FIN} \models \varphi$.

$*$

Small Model Property

The proof of the Small Model Property for PDL uses *filtration*, in which we basically collapse states which are suitably indistinguishable into a single state, giving a new model satisfying the given φ.

Small Model Property

The proof of the Small Model Property for PDL uses *filtration*, in which we basically collapse states which are suitably indistinguishable into a single state, giving a new model satisfying the given φ.

In this way, we get a usable method to transform transition systems satisfying φ into other, finite transition systems.

$*$

Small Model Property

Formally, because this is both cool and crucial to our main result: suppose $TS \models \varphi$.

Small Model Property

Formally, because this is both cool and crucial to our main result: suppose $TS \models \varphi$.

Let Γ be the set of all sub-formulas of φ and their negations; Γ is finite. Define an equivalence relation $\sim$ on the states S in TS such that $s \sim t$ if for all $\psi \in \Gamma$, $s \models \psi \iff t \models \psi$.

Small Model Property

Formally, because this is both cool and crucial to our main result: suppose $TS \models \varphi$.

Let Γ be the set of all sub-formulas of φ and their negations; Γ is finite. Define an equivalence relation $\sim$ on the states S in TS such that $s \sim t$ if for all $\psi \in \Gamma$, $s \models \psi \iff t \models \psi$.

There are at most $2^{|\Gamma|}$ equivalence classes in $S/\sim$ (2 possible truth values for each sub-formula); if we let $[s], [t] \in S/\sim$ represent states in a new TS_{FIN}, with $[s] \xrightarrow{a} [t]$ if for some $s' \in [s]$ and $t' \in [t]$, $s' \xrightarrow{a} t'$, then one can show TS_{FIN} also satisfies φ.

$*$

Expressing PDL in $L\mu$

"I'm expressin' with my full capabilities"

– Dr. Dre

Expressing the modalities

Expressing PDL with the tools available in $L\mu$ is simple - the syntax and semantics are similar, with the exception of the ways in which we may combine programs in PDL. The translations for these are still straightforward:

Expressing the modalities

Expressing PDL with the tools available in $L\mu$ is simple - the syntax and semantics are similar, with the exception of the ways in which we may combine programs in PDL. The translations for these are still straightforward:

$$\langle \alpha_1 \cup \alpha_2 \rangle \varphi \equiv \langle \alpha_1 \rangle \varphi \vee \langle \alpha_2 \rangle \varphi$$

Expressing the modalities

Expressing PDL with the tools available in $L\mu$ is simple - the syntax and semantics are similar, with the exception of the ways in which we may combine programs in PDL. The translations for these are still straightforward:

$$\langle \alpha_1 \cup \alpha_2 \rangle \varphi \equiv \langle \alpha_1 \rangle \varphi \vee \langle \alpha_2 \rangle \varphi$$

$$\langle \alpha_1 \, ; \, \alpha_2 \rangle \varphi \equiv \langle \alpha_1 \rangle \langle \alpha_2 \rangle \varphi$$

Expressing the modalities

Expressing PDL with the tools available in $L\mu$ is simple - the syntax and semantics are similar, with the exception of the ways in which we may combine programs in PDL. The translations for these are still straightforward:

$$\langle \alpha_1 \cup \alpha_2 \rangle \varphi \equiv \langle \alpha_1 \rangle \varphi \vee \langle \alpha_2 \rangle \varphi$$

$$\langle \alpha_1 \; ; \; \alpha_2 \rangle \varphi \equiv \langle \alpha_1 \rangle \langle \alpha_2 \rangle \varphi$$

$$\langle \alpha^* \rangle \varphi \equiv \mu X . \varphi \vee \langle \alpha \rangle X$$

$$*$$

Expressing the modalities

Verifying these formulas are equivalent is an exercise in semantics; let's look at the most interesting case:

$$\langle \alpha^* \rangle \varphi \equiv \mu X. \varphi \vee \langle \alpha \rangle X$$

Using our iteration again, $[\![\varphi \vee \langle \alpha \rangle \bot]\!]$ is the set of all states satisfying φ (no states satisfy $\langle \alpha \rangle \bot$). Then $[\![\varphi \vee \langle \alpha \rangle (\varphi \vee \langle \alpha \rangle \bot)]\!]$ is the set of all states which either satisfy φ, or in which there is a α transition to a state satisfying φ.

Expressing the modalities

Verifying these formulas are equivalent is an exercise in semantics; let's look at the most interesting case:

$$\langle \alpha^* \rangle \varphi \equiv \mu X.\varphi \vee \langle \alpha \rangle X$$

Using our iteration again, $[\![\varphi \vee \langle \alpha \rangle \bot]\!]$ is the set of all states satisfying φ (no states satisfy $\langle \alpha \rangle \bot$). Then $[\![\varphi \vee \langle \alpha \rangle (\varphi \vee \langle \alpha \rangle \bot)]\!]$ is the set of all states which either satisfy φ, or in which there is a α transition to a state satisfying φ.

Iterating this, $s \models \mu X.\varphi \vee \langle \alpha \rangle X$ if and only if there is an α path from s reaching a state satisfying φ. This is precisely the condition defining $\langle \alpha^* \rangle \varphi$.

$*$

Showing $L\mu$ is strictly more expressive

Our final goal is to show that there is a formula in $L\mu$ with no PDL equivalant - two formulas are equivalent if they agree on every TS.

Showing $L\mu$ is strictly more expressive

Our final goal is to show that there is a formula in $L\mu$ with no PDL equivalant - two formulas are equivalent if they agree on every TS.

We will use our old friend $\mu X.[a]X$ – recall $TS \models \mu X.[a]X$ if there are no infinite initial a paths in TS.

Showing $L\mu$ is strictly more expressive

Our final goal is to show that there is a formula in $L\mu$ with no PDL equivalant - two formulas are equivalent if they agree on every TS.

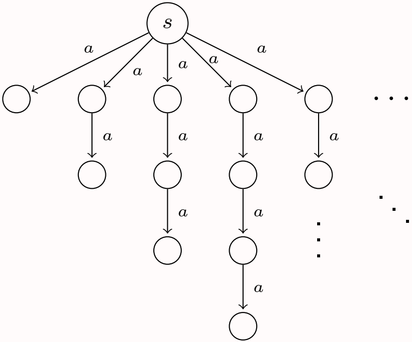
We will use our old friend $\mu X.[a]X$ – recall $TS \models \mu X.[a]X$ if there are no infinite initial a paths in TS.

Suppose φ is a PDL formula which is equivalent to $\mu X.[a]X$. Then if $TS \models \mu X.[a]X$, $TS \models \varphi$ as well.

$$*$$

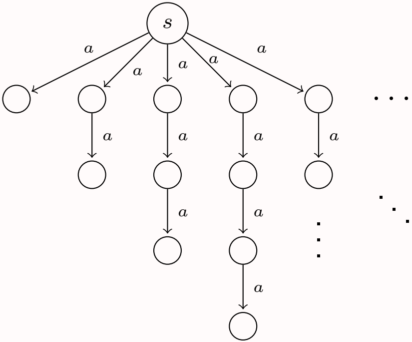
Showing $L\mu$ is strictly more expressive

Consider the following transition system TS with initial state s:



Showing $L\mu$ is strictly more expressive

Consider the following transition system TS with initial state s:



Every path from s is finite length, hence $TS \models \mu X.[a]X$.

$*$

Showing $L\mu$ is strictly more expressive

If φ (the PDL formula) is equivalent to $\mu X.[a]X$, then $TS \models \varphi$ as well.

By the proof of the small model property, we can then collapse TS to a finite TS_{FIN} which also satisfies φ. Since $\varphi \equiv \mu X.[a]X$, it follows that $TS_{FIN} \models \mu X.[a]X$.

Showing $L\mu$ is strictly more expressive

If φ (the PDL formula) is equivalent to $\mu X.[a]X$, then $TS \models \varphi$ as well.

By the proof of the small model property, we can then collapse TS to a finite TS_{FIN} which also satisfies φ. Since $\varphi \equiv \mu X.[a]X$, it follows that $TS_{FIN} \models \mu X.[a]X$.

But TS_{FIN} must contain a loop as a result of the filtration process, so there is an infinite a path. This gives a contradiction.

$*$

Showing $L\mu$ is strictly more expressive

So there is no PDL formula equivalent to $\mu X.[a]X$, and $L\mu$ is strictly more expressive than PDL.

Showing $L\mu$ is strictly more expressive

So there is no PDL formula equivalent to $\mu X.[a]X$, and $L\mu$ is strictly more expressive than PDL.

Thank you for your time! Questions?

$$*$$