Closing a Gap in the Complexity of Refinement Modal Logic

Antonis Achilleos[1] Michael Lampis[2]

1. Graduate Center, City University of New York
aachilleos@gc.cuny.edu

2. KTH Royal Institute of Technology
mlampis@kth.se

July 18, 2013

Outline

Refinement Modal Logic
- Who? When? What? Why?
- Defining RML

The existential fragment
- A tableau procedure

Full RML
- Background
- Closing the Gaps

You (we) are here:

Refinement Modal Logic
- Who? When? What? Why?
- Defining RML

The existential fragment
- A tableau procedure

Full RML
- Background
- Closing the Gaps

Refinement Modal Logic

Who and When?

Defined by Bozzeli, van Ditmarsch and French in 2012.

The complexity of RML satisfiability was studied by Bozzeli, van Ditmarsch and Pinchinat in 2012.

We give a modification of their methods to close the gaps in complexity from BvDP 2012.

Refinement Modal Logic

What?

An extension of the basic normal modal logic, K.

Includes quantifiers $\exists_r$ and $\forall_r$. Intuitively, $\exists_r \phi$ is true in a state of a model if there is a refinement of the original model where ϕ is true.

Think of refinements as submodels until we define them in a few slides.

Refinement Modal Logic

Why?

The goal is to model situations where information is added along the way.
From BvDP 2012:

> *...refinement quantification has applications in many settings: in logics for games ...it may correspond to a player discarding some moves; for program logics ...it may correspond to operational refinement; and for logics for spatial reasoning, it may correspond to subspace projections ...*

Refinement Modal Logic

Syntax

Propositional variables: $p, q, \ldots$

$$\phi ::= p \mid \neg p \mid \phi \wedge \phi \mid \phi \vee \phi \mid \Diamond\phi \mid \Box\phi \mid \exists_r\phi \mid \forall_r\phi$$

If p is a propositional variable, then $p, \neg p$ are literals.
Notice that (for convenience) negations are allowed only at the propositional level.
The existential fragment of RML, $\mathrm{RML}^{\exists_r}$ allows only formulas without $\forall_r$.
$\top, \bot$ as short for a tautology and a contradiction respectively.

Models, Bisimulations, Refinements

Models

We consider the standard Kripke models for modal logic K:
$\mathcal{M} = (W, R, V)$

Models, Bisimulations, Refinements

Models

We consider the standard Kripke models for modal logic K:
$\mathcal{M} = (\mathbf{W}, R, V)$ - (non-empty) Set of worlds/states

Models, Bisimulations, Refinements

Models

We consider the standard Kripke models for modal logic K:
$\mathcal{M} = (W, \mathbf{R}, V)$ - Binary relation on W

Models, Bisimulations, Refinements

Models

We consider the standard Kripke models for modal logic K:
$\mathcal{M} = (W, R, \mathbf{V})$ - Function which assigns to each state in W a set of propositional variables.

Models, Bisimulations, Refinements

Models

We consider the standard Kripke models for modal logic K:
$\mathcal{M} = (W, R, V)$
For p a propositional variable, ϕ, ψ formulas and $s \in W$:
$\mathcal{M}, s \models p$ iff $p \in V(s)$;
$\mathcal{M}, s \models \neg\phi$ iff $\mathcal{M}, s \not\models \phi$;
$\mathcal{M}, s \models \phi \wedge \psi$ iff $\mathcal{M}, s \models \phi$ and $\mathcal{M}, s \models \psi$;
$\mathcal{M}, s \models \phi \vee \psi$ iff $\mathcal{M}, s \models \phi$ or $\mathcal{M}, s \models \psi$;
$\mathcal{M}, s \models \Box\phi$ iff for every $(s, t) \in R$, $\mathcal{M}, t \models \phi$;
$\mathcal{M}, s \models \Diamond\phi$ iff there is some $(s, t) \in R$ such that $\mathcal{M}, t \models \phi$.

Models, Bisimulations, Refinements

Models

$\mathcal{F} = (W, R)$ is called a frame.

Models, Bisimulations, Refinements

Bisimulations and Refinements

For two Kripke models $\mathcal{M} = (W, R, V)$ and $\mathcal{M}' = (W', R', V')$ we say that $\mathcal{M}'$ is *bisimilar* to $\mathcal{M}$ ($\mathcal{M} \approx \mathcal{M}'$) if there exists a relation $\mathcal{R} \subseteq W \times W'$ such that:

- For all $(s, s') \in \mathcal{R}$ we have $V(s) = V'(s')$.
- For all $s \in W$, $s', t' \in W'$ such that $(s, s') \in \mathcal{R}$ and $s'R't'$ there exists $t \in S$ such that $(t, t') \in \mathcal{R}$ and sRt.
- For all $s, t \in W$, $s' \in W'$ such that $(s, s') \in \mathcal{R}$ and sRt there exists $t' \in S$ such that $(t, t') \in \mathcal{R}$ and $s'R't'$.

We call $\mathcal{R}$ a *bisimulation* from $\mathcal{M}$ to $\mathcal{M}'$.

$(\mathcal{M}, a) \approx (\mathcal{M}', b)$ if additionally $a\mathcal{R}b$.

Models, Bisimulations, Refinements

Bisimulations and Refinements

For two Kripke models $\mathcal{M} = (W, R, V)$ and $\mathcal{M}' = (W', R', V')$ we say that $\mathcal{M}'$ is *bisimilar* to $\mathcal{M}$ ($\mathcal{M} \approx \mathcal{M}'$) if there exists a relation $\mathcal{R} \subseteq W \times W'$ such that:

- For all $(s, s') \in \mathcal{R}$ we have $V(s) = V'(s')$.
- For all $s \in W$, $s', t' \in W'$ such that $(s, s') \in \mathcal{R}$ and $s'R't'$ there exists $t \in S$ such that $(t, t') \in \mathcal{R}$ and sRt.
- For all $s, t \in W$, $s' \in W'$ such that $(s, s') \in \mathcal{R}$ and sRt there exists $t' \in S$ such that $(t, t') \in \mathcal{R}$ and $s'R't'$.

We call $\mathcal{R}$ a *bisimulation* from $\mathcal{M}$ to $\mathcal{M}'$.
$(\mathcal{M}, a) \approx (\mathcal{M}', b)$ if additionally $a\mathcal{R}b$.

Models, Bisimulations, Refinements

Bisimulations and Refinements

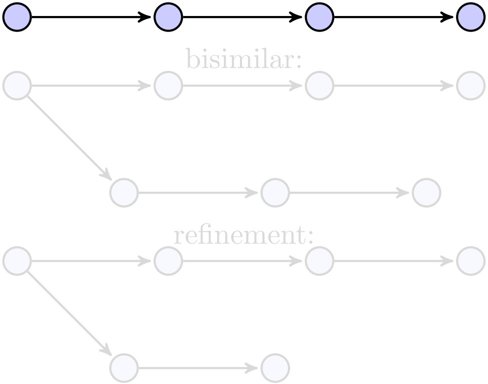
For two Kripke models $\mathcal{M} = (W, R, V)$ and $\mathcal{M}' = (W', R', V')$ we say that $\mathcal{M}'$ is a *refinement* of $\mathcal{M}$ ($\mathcal{M} \succcurlyeq \mathcal{M}'$) if there exists a relation $\mathcal{R} \subseteq W \times W'$ such that:

- For all $(s, s') \in \mathcal{R}$ we have $V(s) = V'(s')$.
- For all $s \in W$, $s', t' \in W'$ such that $(s, s') \in \mathcal{R}$ and $s'R't'$ there exists $t \in S$ such that $(t, t') \in \mathcal{R}$ and sRt.

We call $\mathcal{R}$ a *refinement* mapping from $\mathcal{M}$ to $\mathcal{M}'$.
$(\mathcal{M}, a) \succcurlyeq (\mathcal{M}', b)$ if additionally $a\mathcal{R}b$.

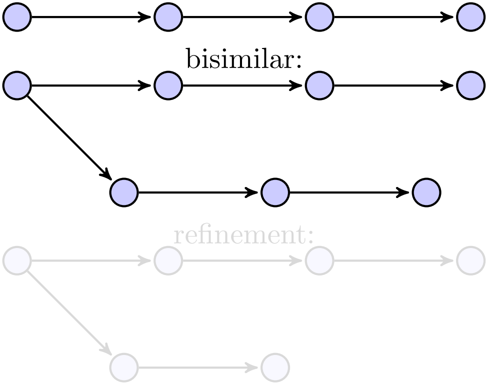
Models, Bisimulations, Refinements

Bisimulations and Refinements



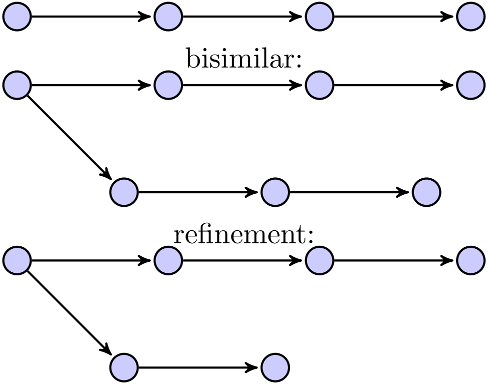
Models, Bisimulations, Refinements

Bisimulations and Refinements



Models, Bisimulations, Refinements

Bisimulations and Refinements



Refinement Modal Logic

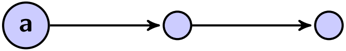
$\mathcal{M}, s \models \exists_r \phi$ iff there is some $(\mathcal{M}', s')$, refinement of (M, s), such that $M', s' \models \phi$;

$\mathcal{M}, s \models \forall_r \phi$ iff for all $(\mathcal{M}', s')$, refinements of (M, s), $M', s' \models \phi$.

Refinement Modal Logic

$$\mathcal{M}, a \models \Box\Diamond\top \land \exists_r(\Diamond\Diamond\top \land \Diamond\Box\bot),$$

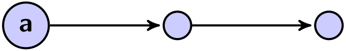
where $\mathcal{M}$ is:

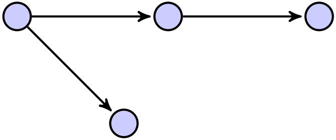


Refinement Modal Logic

$$\mathcal{M}, a \models \Box\Diamond\top \wedge \exists_r(\Diamond\Diamond\top \wedge \Diamond\Box\bot),$$

where $\mathcal{M}$ is:





You (we) are here:

Refinement Modal Logic
- Who? When? What? Why?
- Defining RML

The existential fragment
- A tableau procedure

Full RML
- Background
- Closing the Gaps

Tableau rules for $\text{RML}^{\exists_r}$

- Formulas prefixed by (μ, σ), where $\mu, \sigma \in \mathbb{N}^*$.
- Intuitively, μ represents a model, σ a state.
- $(\mu.i, \sigma)$ is (represents) a refinement of (what is represented by) (μ, σ).
- So is $(\mu.i.j, \sigma)$, because the refinement relation is *transitive*.
- If $(\mu.\nu, \sigma.i), (\mu.\nu, \sigma)$ have appeared, then in the model $\mu.\nu$, $\sigma R \sigma.i$.
- By the definition of refinement and induction on σ, in the model μ, $\sigma R \sigma.i$.
- In general, $\mu, \nu, \sigma \in \mathbb{N}^*$ and $i, j, m, n \in \mathbb{N}$.

Tableau rules for RML$^{\exists_r}$

The rules

$$\frac{(\mu,\sigma)\ \phi\land\psi}{\begin{array}{c}(\mu,\sigma)\ \phi\\(\mu,\sigma)\ \psi\end{array}}\land \qquad \frac{(\mu,\sigma)\ \phi\lor\psi}{(\mu,\sigma)\ \phi\ \mid (\mu,\sigma)\ \psi}\lor \qquad \frac{(\mu.\nu,\sigma)\ l}{(\mu,\sigma)\ l}L$$

$$\frac{(\mu,\sigma)\ \Diamond\phi}{(\mu,\sigma.i)\ \phi}\Diamond$$

where $\sigma.i$ has not appeared

$$\frac{(\mu,\sigma)\ \exists_r\phi}{(\mu.m,\sigma)\ \phi}\exists_r$$

where $\mu.m$ has not appeared

$$\frac{(\mu,\sigma)\ \Box\phi}{(\mu,\sigma.i)\ \phi}\Box$$

where $(\mu.\nu,\sigma.i)$ has appeared

Tableau rules for RML$^{\exists_r}$

Accepting conditions

A tableau branch is propositionally closed when it includes some $(\mu, \sigma)\ p$ and $(\mu, \sigma)\ \neg p$.

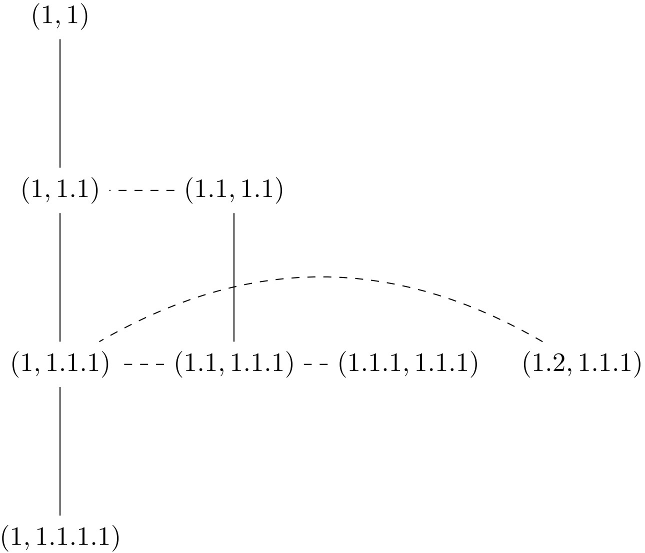
The tableau procedure for ϕ starts from $(1, 1)\ \phi$ and accepts iff we can construct some branch closed under the tableau rules and not propositionally closed.

Tableau Example

$$
\begin{array}{cl}
(1,1)\ \Diamond(\Box((p \vee \Diamond p) \wedge \exists_r \Box \bot) \wedge \exists_r \Diamond(\Box \neg r \wedge \exists_r \neg p)) & \\
\hline
(1,1.1)\ \Box((p \vee \Diamond p) \wedge \exists_r \Box \bot) \wedge \exists_r \Diamond(\Box \neg r \wedge \exists_r \neg p) & \Diamond \\
\hline
(1,1.1)\ \Box((p \vee \Diamond p) \wedge \exists_r \Box \bot) & \wedge \\
(1,1.1)\ \exists_r \Diamond(\Box \neg r \wedge \exists_r \neg p) & \\
\hline
(1.1,1.1) \Diamond(\Box \neg r \wedge \exists_r \neg p) & \exists_r \\
\hline
(1.1,1.1.1)\ \Box \neg r & \Diamond\wedge \\
(1.1,1.1.1)\ \exists_r \neg p & \\
\hline
(1.1.1,1.1.1)\ \neg p & \exists_r \\
\hline
(1,1.1.1)\ \neg p & L \\
(1.1,1.1.1)\ \neg p & \\
\hline
(1,1.1.1)\ p \vee \Diamond p & \Box\wedge \\
(1,1.1.1)\ \exists_r \Box \bot & \\
\hline
(1,1.1.1)\ \Diamond p & \vee \\
\hline
(1,1.1.1.1)\ p & \Diamond \\
\hline
(1.2,1.1.1)\ \Box \bot & \exists_r
\end{array}
$$

Tableau Example

The tree(s) of the prefixes



Tableau

Correctness

Lemma

ϕ is satisfiable if and only if starting from $(1,1)\ \phi$ we can make appropriate non-deterministic choices to end up with a complete accepting tableau branch.

Tableau

Bounding the prefixes

Lemma

In any branch b such that $(\mu, \sigma)\ \psi \in b$, we have $|\mu| \leq d_\exists(\phi)$ and $|\sigma| \leq d_\Diamond(\phi)$.

This observation gives us the key to give an algorithm for $\text{RML}^{\exists_r}$-satisfiability.

$d_\exists(\phi)$ is the nesting depth of $\exists_r$ in ϕ and $d_\Diamond(\phi)$ the modal depth of ϕ.

Tableau

Bounding the prefixes

Lemma

In any branch b *such that* $(\mu, \sigma)\ \psi \in b$, *we have* $|\mu| \leq d_\exists(\phi)$ *and* $|\sigma| \leq d_\Diamond(\phi)$.

This observation gives us the key to give an algorithm for $\mathrm{RML}^{\exists_r}$-satisfiability.

$d_\exists(\phi)$ is the nesting depth of $\exists_r$ in ϕ and $d_\Diamond(\phi)$ the modal depth of ϕ.

An algorithm

That is, exploring a branch using only polynomial space

- A non-deterministic algorithm using polynomial space.
- Keep a set (P) of prefixed formulas in the branch currently under consideration and a subset of this which includes all such formulas that have already been used in a tableau rule (called M).
- For each $(\mu, \sigma)\ \psi \in P$, where ψ a literal, a disjunction or conjunction, apply the appropriate rule(s) and mark the formula as used (put it in M).
- For each $\alpha = (\mu, \sigma)\ \Diamond\psi \in P$, $P_\alpha := \{(\lambda, \sigma.i)\ \chi \mid (\lambda, \sigma)\ \Box\chi \in P \text{ and } \lambda \sqsubseteq \mu\} \cup \{(\mu, \sigma.i)\ \psi\}$ for some new i and explore P_α.
- For each $\alpha = (\mu, \sigma)\ \exists_r\psi \in P$, $P_\alpha := \{(\lambda, \sigma)\ \chi \in P \mid \lambda \sqsubseteq \mu\} \cup \{(\mu.i, \sigma)\ \psi\}$ for some new i and explore P_α. Keep any $(1, \sigma)\ l$ formulas in P.

An algorithm

That is, exploring a branch using only polynomial space

- A non-deterministic algorithm using polynomial space.
- Keep a set (P) of prefixed formulas in the branch currently under consideration and a subset of this which includes all such formulas that have already been used in a tableau rule (called M).
- For each $(\mu, \sigma)\ \psi \in P$, where ψ a literal, a disjunction or conjunction, apply the appropriate rule(s) and mark the formula as used (put it in M).
- For each $\alpha = (\mu, \sigma)\ \Diamond\psi \in P$,
  $P_\alpha := \{(\lambda, \sigma.i)\ \chi \mid (\lambda, \sigma)\ \Box\chi \in P \text{ and } \lambda \sqsubseteq \mu\} \cup \{(\mu, \sigma.i)\ \psi\}$ for some new i and explore P_α.
- For each $\alpha = (\mu, \sigma)\ \exists_r\psi \in P$,
  $P_\alpha := \{(\lambda, \sigma)\ \chi \in P \mid \lambda \sqsubseteq \mu\} \cup \{(\mu.i, \sigma)\ \psi\}$ for some new i and explore P_α. Keep any $(1, \sigma)\ l$ formulas in P.

An algorithm

That is, exploring a branch using only polynomial space

- A non-deterministic algorithm using polynomial space.
- Keep a set (P) of prefixed formulas in the branch currently under consideration and a subset of this which includes all such formulas that have already been used in a tableau rule (called M).
- For each $(\mu, \sigma)\ \psi \in P$, where ψ a literal, a disjunction or conjunction, apply the appropriate rule(s) and mark the formula as used (put it in M).
- For each $\alpha = (\mu, \sigma)\ \Diamond\psi \in P$, $P_\alpha := \{(\lambda, \sigma.i)\ \chi \mid (\lambda, \sigma)\ \Box\chi \in P \text{ and } \lambda \sqsubseteq \mu\} \cup \{(\mu, \sigma.i)\ \psi\}$ for some new i and explore P_α.
- For each $\alpha = (\mu, \sigma)\ \exists_r\psi \in P$, $P_\alpha := \{(\lambda, \sigma)\ \chi \in P \mid \lambda \sqsubseteq \mu\} \cup \{(\mu.i, \sigma)\ \psi\}$ for some new i and explore P_α. Keep any $(1, \sigma)\ l$ formulas in P.

The algorithm is correct

- The algorithm (non-deterministically) explores a tableau branch.
- The union of all the P's that come up is a branch closed under the rules.
- All literals are gathered under prefix $(1, \sigma)$.
- So...

Theorem

The satisfiability problem for $RML^{\exists_r}$ *is in* PSPACE.

The algorithm is correct

- The algorithm (non-deterministically) explores a tableau branch.
- The union of all the P's that come up is a branch closed under the rules.
- All literals are gathered under prefix $(1, \sigma)$.
- So...

Theorem

The satisfiability problem for $RML^{\exists r}$ is in PSPACE*.*

You (we) are here:

Refinement Modal Logic
- Who? When? What? Why?
- Defining RML

The existential fragment
- A tableau procedure

Full RML
- Background
- Closing the Gaps

The Algorithm by Bozzeli, van Ditmarsch and Pinchinat (2012)

Alternation depth and fragments

- The *weak refinement alternation depth* of ϕ ($\mathcal{Y}_w(\phi)$) is the quantifier alternation depth of $\exists_r\phi$.
- $\mathcal{Y}_w(\exists_r\phi) = \mathcal{Y}_w(\phi)$ and $\mathcal{Y}_w(\forall_r\phi) = \mathcal{Y}_w(\neg\forall_r\phi) + 1$.
- RML^k consists of all RML formulas of weak refinement alternation depth at most k.
- $\mathrm{RML}^{\exists_r} = \mathrm{RML}^1$

The Algorithm by Bozzeli, van Ditmarsch and Pinchinat (2012)

The picture

K	PSPACE-complete
$\mathrm{RML}^{\exists} = \mathrm{RML}^1$	$\in$ NEXPTIME PSPACE-hard
RML^2	$\in \Sigma_2^{EXP}$ NEXPTIME-hard
RML^{k+1} $(k \leq 1)$	$\in \Sigma_{k+1}^{EXP}$ Σ_k^{EXP}-hard
RML	AEXP_{pol}-complete

The complexity of satisfiability for fragments of RML

The Algorithm by Bozzeli, van Ditmarsch and Pinchinat (2012)

The algorithm

The algorithm first non-deterministically guesses a tree model of at most an exponential number of states[1] for ϕ and then runs the following to check that ϕ is satisfied:

- Given a tree model and ϕ, non-deterministically spread its subformulas tableau-wise on the tree (do not analyse $\exists_r\psi$ and $\forall_r\psi$).
- Wherever you see an $\exists_r\psi$, non-deterministically guess a tree model of at most an exponential number of states and which is a refinement of the original. Go on to check that ψ is satisfied there.
- Wherever you see a $\forall_r\psi$, use an oracle for $\neg\forall_r\psi = \exists_r\neg\psi$ and the subtree with root the current state. Notice that the weak alternation depth of $\neg\forall_r\psi$ is one less than that of $\forall_r\psi$.

[1] Yes, we can do that.

Our Variation

- We do the same thing.
- Except, when given an $\mathrm{RML}^{\exists_r}$ formula, we do not have to guess a model. We can *deterministically construct* it (all of them, actually) using polynomial space, so exponential time.
- This saves us a step in the exponential hierarchy and closes the complexity gaps.

Our Variation

- We do the same thing.
- Except, when given an $\mathrm{RML}^{\exists_r}$ formula, we do not have to guess a model. We can *deterministically construct* it (all of them, actually) using polynomial space, so exponential time.
- This saves us a step in the exponential hierarchy and closes the complexity gaps.

The resulting picture

K	PSPACE-complete
$\mathrm{RML}^{\exists} = \mathrm{RML}^{1}$	PSPACE-complete
RML^{2}	NEXPTIME-complete
RML^{k+1} $(k \leq 1)$	Σ_k^{EXP}-complete
RML	AEXP_{pol}-complete

The complexity of satisfiability for fragments of RML

Thank you.

Questions?