Model Checking for Compositional Models of General Linear Time

Tim French, **John M^{c}Cabe-Dansted**, Mark Reynolds

University of Western Australia

17th September 2013

Introduction

- Real Temporal Lo ic: Simple, Expressively complete re: First Order Monadic Lo ic of Order.
- French et al. [2012] proposed a lan ua e of model expressions, based on pioneerin work of Läuchli and Leonard [1966], Bur ess and Gurevich [1985], Reynolds [2001, 2010]
  - Complete: Every satisfiable formula has a model in this language.
- Have complete and finite representations of models for Real Temporal Lo ic formulas
  - Obvious question: Can we Model Check?
  - Yes, and generalise to General Linear Time
  - And we have an efficient implementation [McCabe-Dansted, 2012]

Compositional Model Expressions

Model Expressions are an abstract syntax to define models usin the followin set of primitive operators based on the four operations:

$$\mathcal{I} ::= a \mid \lambda \mid \mathcal{I} + \mathcal{J} \mid \overleftarrow{\mathcal{I}} \mid \overrightarrow{\mathcal{I}} \mid \langle \mathcal{I}_0, \ldots, \mathcal{I}_n \rangle$$

where $a \in \Sigma = 2^{\mathbf{L}}$ so the letter indicates the atoms true at a point. We refer to these operators, respectively, as a letter, the empty order, concatenation, lead, trail, and shuffle.

- How does this relate to a model like $(\mathbb{Q}, <, g)$?

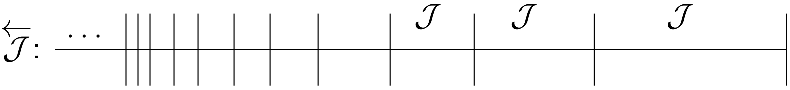
Correspondence

We define whether a model expression corresponds to a model recursively as follows: (or more formally in paper)

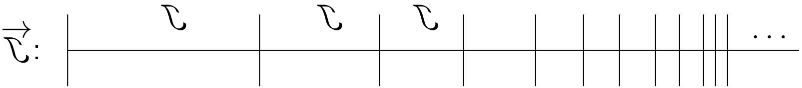
- A letter a corresponds to a sin le point at which the set of atoms satisfied is a.
- The empty order λ corresponds to an empty (psuedo-frame)
- The concatenation $\mathcal{I} + \mathcal{J}$ corresponds to a model where first $\mathcal{I}$, then $\mathcal{J}$.
- The lead $\overleftarrow{\mathcal{I}}$ corresponds to an infinite number of repeats of $\mathcal{I}$, formin a limit point on the left.
- The trail $\overrightarrow{\mathcal{I}}$ corresponds to an infinite number of repeats of $\mathcal{I}$, formin a limit point on the ri ht.
- A shuffle $\langle \mathcal{I}_0, \ldots, \mathcal{I}_n \rangle$ corresponds to a dense mixture of $\mathcal{I}_0, \ldots, \mathcal{I}_n$.

Lead: $\mathcal{I} = \overleftarrow{\mathcal{J}}$

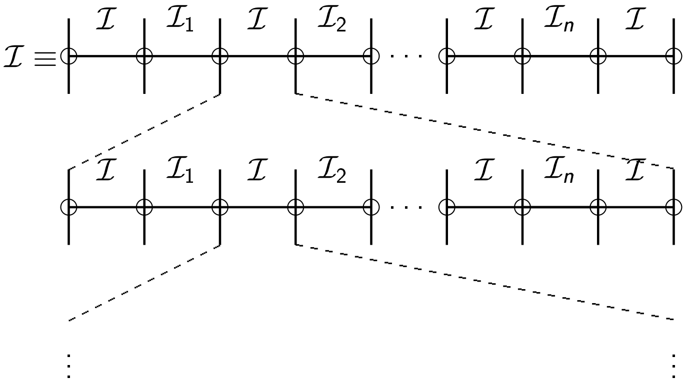
Here is a lead $\overleftarrow{\mathcal{J}}$.



A trail $\overrightarrow{\reflectbox{$\mathcal{J}$}}$ is just the mirror ima e of $\overleftarrow{\mathcal{J}}$.



Shuffle $\mathcal{I} = \langle \mathcal{I}_1, \ldots, \mathcal{I}_n \rangle$

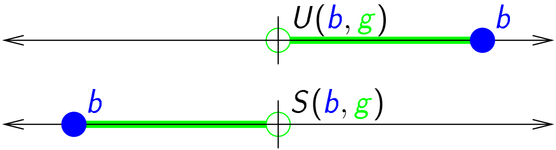


Language of Until and Since

- The Lan ua e of Until and Since

$$\phi ::= \; p \mid \neg\phi \mid \phi \wedge \phi \mid U(p,q) \mid S(p,q)$$

- In RTL:



- The US/L Lo ic like RTL, but models can be over any linear flow of time (not just Reals, e. . Inte ers).

An Example: Zeno

If we walk into a wall: we will halve the distance; whenever we have halved the distance, we will halve the distance a ain, but only after a period of not halvin the distance; once we have reached the wall we will not halve the distance anymore; and finally, we reach the wall. Where h represents "we have halved the distance" and r represents "we have reached the wall". The US/L formula for the intended expression is:

$$Fh \wedge G\,(h \rightarrow U\,(h, \neg h)) \wedge G\,(r \rightarrow G\neg h) \wedge Fr\ .$$

We see that this does not cause a contradiction as this formula is satisfied at the leftmost point x of any structure $\mathbf{T}$ correspondin to $\overrightarrow{\{h\} + \emptyset} + \{r\}$.

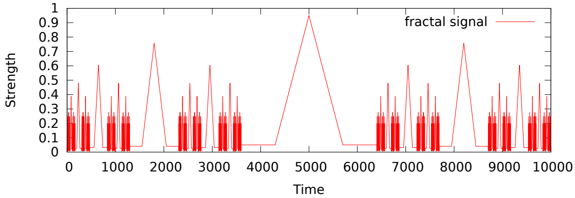
An Example: Detector

At some sportin event, it may be that a player is awarded a point if the ball bounces twice. An automated system may detect such bounces and make a rulin as to whether to award a point. The system may poll a iven sensor, to determine whether a bounce has occurred since the previous pollin event. If the system detects two bounces it awards a point. Where b indicates that a bounce has just occurred, s indicates that system has just checked its sensor, and e indicates the end of the round.

The player deserves a point precisely if $\theta = F(b \wedge F(b \wedge Fe))$ holds while the system awards a point precisely where the formula $\theta_s = F(b \wedge F(s \wedge F(b \wedge F(s \wedge Fe))))$ holds. The result is correct if $\theta \leftrightarrow \theta_s$ holds. We can verify the result is correct for a run of the system a ainst the environment described by the ME:

$$\{s\} + \{b\} + \{s\} + \{s\} + \{b\} + \{s\} + \overrightarrow{\{b\}} + \{e\}$$

An Example: Fractal Signal



Let p represent an increment, q describe a decrement, and r represent a constant si nal, then we can model this si nal usin :

$$\langle\langle r\rangle + \langle p\rangle + \langle q\rangle + \langle r\rangle\rangle$$

Some properties of this si nal can be represented in $L(U, S)$, for example we can reach a re ion of increment directly from a constant re ion so $r \wedge U\,(p, r \vee p)$ is satisfied within the model. However, we cannot reach a re ion of increment directly from a re ion of decrement, so $q \wedge U\,(p, q)$ is satisfied nowhere.

Model Checking

Definition

We define the model checkin problem as follows: iven an ME $\mathcal{I}$ and formula ϕ, determine whether there exists a structure $\mathbf{T} = (T, <, h)$ corresponded to by $\mathcal{I}$ and point $x \in T$ such that $\mathbf{T}, x \vDash \phi$.

A traditional approach to model checkin is to add subformula as atoms

Definition

The model checkin procedure takes as input an ME $\mathcal{I}$ and formula ϕ. We enumerate the subformulas $\phi_1, \ldots, \phi_n$ of ϕ from shortest to lon est (so $\phi_n = \phi$), let $\mathcal{I}_0 = \mathcal{I}$, and $\mathcal{I}_i = \mathrm{add\_atom}_{\ i}(\mathcal{I}_{i-1})$ for each $i \in \{0, \ldots, n\}$. Finally, we return "true" if there is a letter a in $\mathcal{I}_n$ such that $\phi \in a$, and "false" otherwise.

Adding $\wedge$ and $\neg$ as atoms

Addin PC formulas as atoms is trivial consider:

	ϕ	$\mathcal{I}$
1	$(a \wedge b) \wedge \neg c$	$\{a, b\} + \{c\}$
2	$(a \wedge b) \wedge p_{\neg c}$	$\{a, b, p_{\neg c}\} + \{c\}$
3	$p_{a \wedge b} \wedge p_{\neg c}$	$\{a, b, p_{a \wedge b}, p_{\neg c}\} + \{c\}$
4	$p_{(a \wedge b) \wedge \neg c}$	$\{a, b, p_{a \wedge b}, p_{\neg c}, p_{(a \wedge b) \wedge \neg c}\} + \{c\}$

Pre(satisfaction)

"pre $(\mathcal{K}, \dashv)$" is true if $U(p,q)$ is true before $\mathcal{K}$ where

- $\mathcal{K}$ is an ME
- And $\dashv$ means "$U(p,q)$" is true after $\mathcal{K}$

Definition

We define a function "pre" from Booleans and MEs to Booleans such that: for any Boolean $\dashv$ and pair of MEs $\mathcal{I}, \mathcal{J}$

1. $\mathrm{pre}\,(a, \dashv) = p \in a \vee (\dashv \wedge q \in a)$
2. $\mathrm{pre}\,(\mathcal{I} + \mathcal{J}, \dashv) = \mathrm{pre}\,(\mathcal{I}, \mathrm{pre}\,(\mathcal{J}, \dashv))$
3. $\mathrm{pre}\left(\overrightarrow{\mathcal{I}}, \dashv\right) = \mathrm{pre}\,(\mathcal{I}, \dashv) \quad = \mathrm{pre}\,(\mathcal{I}, \mathrm{pre}\,(\mathcal{I}, \dashv))$
4. $\mathrm{pre}\,(\mathcal{J}, \dashv) = (\dashv \vee \exists l \in L\,(\mathcal{J})$ s.t. $p \in l) \wedge \forall l \in L\,(\mathcal{J})\,, q \in l$; where $\mathcal{J}$ is of the form $\overleftarrow{\mathcal{I}}$ or $\langle \ldots \rangle$ and $L\,(\mathcal{J})$ is the set of letters within $\mathcal{J}$.

Adding Until as an Atom

Definition

We define $\text{add\_atom}_{U(p;q)}(\mathcal{I})$ as $t(\mathcal{I}, \bot)$: where t is a function that takes an ME and a Boolean as input, and outputs an ME as follows: for any Boolean $\dashv$, pair of MEs $\mathcal{I}, \mathcal{J}$ and sequence of MEs $\mathcal{I}_0, \ldots, \mathcal{I}_n$

1. $t(a, \dashv) = \begin{cases} a & \text{if } \dashv = \bot \\ a \cup \{U(p, q)\} & \text{if } \dashv = \top \end{cases}$
2. $t(\mathcal{I} + \mathcal{J}, \dashv) = t(\mathcal{I}, \text{pre}(\mathcal{J}, \dashv)) + t(\mathcal{J}, \dashv)$
3. $t\left(\overleftarrow{\mathcal{I}}, \dashv\right) = \overleftarrow{t(\mathcal{I}, \text{pre}(\mathcal{I}, \dashv))} + t(\mathcal{I}, \dashv)$
4. $t\left(\overrightarrow{\mathcal{I}}, \dashv\right) = \overrightarrow{t(\mathcal{I}, \text{pre}(\mathcal{I}, \dashv))}$
5. $t(\mathcal{K}, \dashv) = \langle t(\mathcal{I}_0, \dashv'), \ldots, t(\mathcal{I}_n, \dashv') \rangle$ where $\mathcal{K} = \langle \mathcal{I}_0, \ldots, \mathcal{I}_n \rangle$ and $\dashv' = \text{pre}(\mathcal{K}, \dashv)$

Complexity (1)

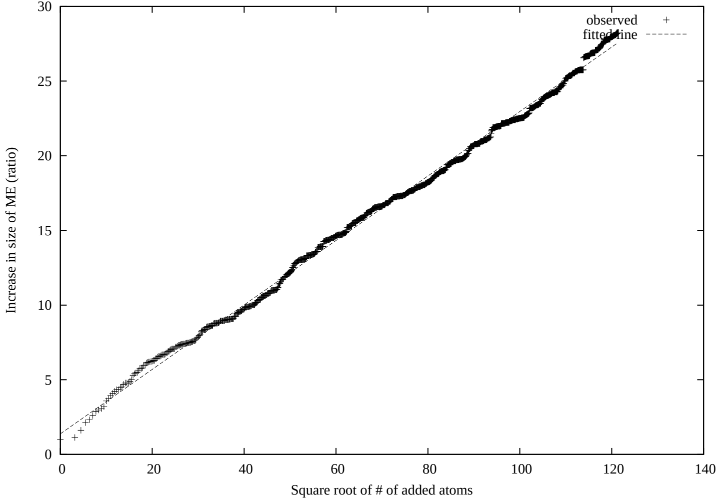
- Addin one Until: $\overleftarrow{\mathcal{I}} \rightsquigarrow \overleftarrow{\mathcal{I}_0} + \mathcal{I}_1$
- Addin n Untils: $\overleftarrow{\mathcal{I}} \rightsquigarrow \overleftarrow{\mathcal{I}_0} + \mathcal{I}_1 + \cdots + \mathcal{I}_n$
- With m nested leads, has $\approx (n+1)^m$ len th
- $\mathcal{O}\left(|\phi|^{|\mathcal{I}|}\right)$
- **Polynomial in Length of the Formula**

Complexity (2)

Traditionally we want model checkin to be polynomial in the Model

- Represent MEs as Directed Acyclic Graphs (1 node per unique subME)
- Each unique $\mathcal{I}$ becomes $t\,(\mathcal{I}, \top)$ or $t\,(\mathcal{I}, \bot)$
- Addin U as atom ~~doubles~~ **triples** # unique nodes
    - E.g. $\overleftarrow{\{p\}}$ becomes $\overleftarrow{\{p, U\,(p, p)\}} + \{p\}$
- $|\mathcal{I}|\,3^{|\ |}$ nodes (or $|\mathcal{I}|\,|\phi|\,2^{|\ |}$)
- **Linear in Length of ME**

Benchmark: Random Square Problem(s)



Plot of rowth of $|\mathcal{I}_n|/|\mathcal{I}_0|$ vs. $\sqrt{n}$ for $2^{15} \times 2^{15}$ problem

Estimated Space: $\approx |\mathcal{I}|\,|\phi|^{3=2}$ and Time $\approx |\mathcal{I}|\,|\phi|^{5=2}$

Ongoing Work

Every RTL model M of an formula ϕ is also a US/L model.

- Can model check Real MEs usin same model checker
- Minor Detail: MEs correspond to <u>countable</u> structures

Expressive completeness re Linear flows requires Stavi Until/Since

- Can translate into RTL usin special atom c for aps.

Can translate Metrics with error into RTL.

Conclusions

- ME provide models for all RTL formulas.
- Found efficient model checker:
  - Polynomial in Length of Formula
  - Linear in length of ME
  - Moderate $\sqrt{n}$ growth on random formulas
  - But PSPACE-Complete in general (Present at TIME 2013)
- Extensions for Metrics, Stavi etc. inpro ress.

References

J. P. Burgess and Y. Gurevich. The decision problem for linear temporal logic. Notre Dame J. Formal Logic, 26(2):115–128, 1985.

Tim French, John Christopher McCabe-Dansted, and Mark Reynolds. Synthesis for temporal logic over the reals. In Thomas Bolander, Torben Braüner, Silvio Ghilardi, and Lawrence S. Moss, editors, Advances in Modal Logic, pages 217–238. College Publications, 2012. ISBN 978-1-84890-068-4.

H. Läuchli and J. Leonard. On the elementary theory of linear order. Fundamenta Mathematicae, 59:109–116, 1966.

John Christopher McCabe-Dansted. Model checker for general linear time (online applet and data), 2012.
`http://www.csse.uwa.edu.au/~mark/research/Online/mechecker.html`.

M. Reynolds. The complexity of the temporal logic over the reals. Annals of Pure and Applied Logic, 161(8):1063–1096, 2010. doi: 10.1016/j.apal.2010.01.002.

Mark Reynolds. Continuous temporal models. In Markus Stumptner, Dan Corbett, and Michael J. Brooks, editors, Australian Joint Conference on Artificial Intelligence, volume 2256 of Lecture Notes in Computer Science, pages 414–425. Springer, 2001. ISBN 3-540-42960-3.

NP-Hardness (PSPACE-Hardness)

We can show NP-Hardness (extend to PSPACE-hardness in Paper)

1. $\mathcal{I}_1 = \overleftarrow{p_0} + q_0$
2. $\mathcal{I}_i = \overleftarrow{p_i + \mathcal{I}_{i-1}} + q_i$

SAT problem: Is $\phi = (r_1 \wedge r_0) \vee \neg r_0$ satisfiable

- replace r_i with $r_i' = U\left(q_i, \neg p_i\right)$
    - i.e. $\phi' = \left(U\left(q_1, \neg p_1\right) \wedge U\left(q_0, \neg p_0\right)\right) \vee \neg U\left(q_0, \neg p_0\right)$
- if n atoms in ϕ: model check $\mathcal{I}_n$ a ainst ϕ'

Example

- Model check a ainst $\mathcal{I}_1$ (or $\mathcal{I}_{n-1}$ if n atoms in ϕ)
- $\mathcal{I}_2 = \overleftarrow{p_1 + \overleftarrow{p_0} + q_0} + q_1$
- $\mathcal{I}_2 \rightsquigarrow \overleftarrow{p_1 + \overleftarrow{\{p_0\}} + \{p_0, U(q_0, \neg p_0)\} + q_0} + q_1$
- $\mathcal{I}_2 \rightsquigarrow \overleftarrow{p_1 + \overleftarrow{\{p_0\}} + \{p_0, U(q_0, \neg p_0)\} + q_0} +$ $\overleftarrow{\{p_0, U(q_1, \neg p_1)\}} + \{p_0, U(q_0, \neg p_0), U(q_1, \neg p_1)\} + q_1$

PSPACE

We use a recursively defined function $\mathfrak{A}$. Informally, $\mathfrak{A}(\mathcal{I}, \Phi, \phi) = (\Theta, \Psi)$ means

- if we have an interval $\mathbf{T}_V$ that corresponds to $\mathcal{I}$
  - Φ is the set of formulas in $\mathbb{U}$ presatisfied immediately after V in $\mathbf{T}$
  - we are only interested in ϕ and subformulas of ϕ,
- Then it must be the case that the set of formulas satisfied within $\mathbf{T}_V$ is Θ,
  - and the set formulas in $\mathbb{U}$ presatisfied immediately before V in $\mathbf{T}$ is Ψ.
  - The formula ϕ indicates that we are only interested in whether $\phi \in \Theta$ and so we can limit ourselves to subformulas of ϕ and their negations. The algorithm works by generating increasing accurate approximations to (Θ, Ψ) that are accurate up to some subformula ϕ_j.

Excerpt of Recursive Function $\mathfrak{A}$

Definition

Let $\mathcal{K}$ be an ME, and Φ be a set of formulae. We define $\mathfrak{A}(\mathcal{K}; \Phi; \ )$ to be the pair of sets of formulas $(\Theta; \Psi)$ as follows.
We consider various possible forms of $\mathcal{K}$. The first case we consider is a letter. In the following construction we build increasingly accurate approximations of $(\Theta; \Psi)$: for each j we have $\Theta_j \approx_{\leq j} \Theta$ and $\Psi_j \approx_{\leq j} \Psi$.
Case 0. $\mathcal{K} = \ $. Since $\mathcal{K}$ corresponds to the empty pseudo frame it cannot satisfy any formula so we let $\Theta = \emptyset$, likewise it cannot affect pre or postsatisfaction so $\Psi = \Phi$.

Case 1. $\mathcal{K}$ is a letter:

- $\Theta_0 = \emptyset$, $\Psi_0 = \emptyset$
- for each i:
  - We let Θ_i be a set such that $\Theta_i \setminus \{ \ _i\} = \Theta_{i-1} \setminus \{ \ _i\}$ and: if $\ _i$ is of the form $\neg \ $ then $\ _i \in \Theta_i$ iff $\ \notin \Theta_{i-1}$; if $\ _i$ is of the form $\ \wedge \ $ then $\ _i \in \Theta_i$ iff $\ \in \Theta_{i-1} \wedge \ \in \Theta_{i-1}$; if $\ _i$ is an atom then $\ _i \in \Theta_i$ iff $\ _i \in \mathcal{K}$; if $\ _i \in \mathbb{U} \cup \mathbb{S}$ then $\ _i \in \Theta_i$ iff $\ \in \Phi$.
  - $\ _i \in \Psi$ iff $\ _i$ is of the form $U(\ ; \ )$ or $S(\ ; \ )$ and either $\ \in \Theta_i$ or $(\ \in \Theta_i) \wedge (\ _i \in \Phi)$.
  - We let Θ be the minimal expansion of Θ_n such that for each $\ _i$, we have $\ _i \in \Theta$ iff $\ _i \in \Theta_n$ and $\neg \ _i \in \Theta$ iff $\ _i \notin \Theta_n$. We add the negations into Θ so that when we have an ME with multiple letters we can express " occurs everywhere" as $\neg \ \notin \Theta$.

. . .