Programming in Linear Temporal Logic

Wolfgang Jeltsch

Brandenburgische Technische Universität Cottbus
Cottbus, Germany

Seminar talk at the Institute of Cybernetics
Tallinn, Estonia

February 10, 2011

Linear Temporal Logic

- trueness of a proposition depends on time
- times are natural numbers
- propositional logic extended with four new constructs:

$\bigcirc\varphi$	φ will hold at the next time
$\Box\varphi$	φ will always hold
$\Diamond\varphi$	φ will eventually hold
$\varphi \rhd \psi$	φ will hold for some time, and then ψ will hold

- for now only $\Box$ and $\Diamond$:
  - restricted LTL
  - continuous time also possible

Embedding into predicate logic

- temporal formula φ can be translated into predicate logic formula $\langle\varphi\rangle$
- $\langle\varphi\rangle$ may contain a single free variable t that denotes the time
- atomic propositions p correspond to predicates $\hat{p}$ that take a time argument
- translation for propositional logic fragment:

$$\begin{aligned} \langle p\rangle &= \hat{p}(t) & \langle\varphi \wedge \psi\rangle &= \langle\varphi\rangle \wedge \langle\psi\rangle \\ \langle\top\rangle &= \top & \langle\varphi \vee \psi\rangle &= \langle\varphi\rangle \vee \langle\psi\rangle \\ \langle\bot\rangle &= \bot & \langle\varphi \to \psi\rangle &= \langle\varphi\rangle \to \langle\psi\rangle \end{aligned}$$

- translation for $\Box$ and $\Diamond$:

$$\begin{aligned} \langle\Box\varphi\rangle &= \forall t' \in [t, \infty) \,.\, \langle\varphi\rangle[t'/t] \\ \langle\Diamond\varphi\rangle &= \exists t' \in [t, \infty) \,.\, \langle\varphi\rangle[t'/t] \end{aligned}$$

Restricted LTL as a type system

- type inhabitation depends on time
- simple type system extended with two new type constructors $\Box$ and $\Diamond$
- temporal type α can be translated into dependent type $\langle\alpha\rangle$
- $\langle\alpha\rangle$ may contain a single-free variable t that denotes the time
- translation for $\Box$ and $\Diamond$:

$$\langle\Box\alpha\rangle = \Pi t' \in [t, \infty) \,.\, \langle\alpha\rangle[t'/t]$$
$$\langle\Diamond\alpha\rangle = \Sigma t' \in [t, \infty) \,.\, \langle\alpha\rangle[t'/t]$$

- concepts from Functional Reactive Programming (FRP):
  - $\Box$ behaviors
  - $\Diamond$ events
- restricted LTL corresponds to a strongly typed form of FRP
- t denotes start times of behaviors and events

Basics

- categorical models should be CCCCs:
  - LTL extends propositional logic
  - FRP extends simply-typed λ-calculus
- components of a categorical model:

objects	propositions/types
morphisms	time-independent proofs/functions:

$$f : \alpha \to \beta \quad \Rightarrow \quad f : \Pi t \, . \, \langle \alpha \rangle \to \langle \beta \rangle$$

- $\Box$ and $\Diamond$ are (endo)functors:

$$\frac{f : \alpha \to \beta}{\Box f : \Box\alpha \to \Box\beta} \qquad \frac{f : \alpha \to \beta}{\Diamond f : \Diamond\alpha \to \Diamond\beta}$$

- start time consistency is ensured:

$$\Box : (\Pi t \, . \, \langle \alpha \rangle \to \langle \beta \rangle) \to (\Pi t \, . \, \langle \Box\alpha \rangle \to \langle \Box\beta \rangle)$$
$$\Diamond : (\Pi t \, . \, \langle \alpha \rangle \to \langle \beta \rangle) \to (\Pi t \, . \, \langle \Diamond\alpha \rangle \to \langle \Diamond\beta \rangle)$$

Operations on behaviors

- $\Box$ is a comonad:

$$\mathsf{head} : \Box\alpha \to \alpha$$
$$\mathsf{tails} : \Box\alpha \to \Box\Box\alpha$$

- $\Box$ is a strong cartesian functor:

$$\mathsf{units} : 1 \to \Box 1$$
$$\mathsf{zip} : \Box\alpha \times \Box\beta \to \Box(\alpha \times \beta)$$

- $\Box$ is not an applicative functor:
  - lifting of pure values would have to be possible:

$$\mathsf{const} : \alpha \to \Box\alpha$$

  - would break start time consistency:

$$\mathsf{const} : \Pi t \,.\, \langle\alpha\rangle \to \Pi t' \in [t, \infty) \,.\, \langle\alpha\rangle[t'/t]$$

- however, this is possible:

$$\frac{f : 1 \to \alpha}{\Box f \circ \mathsf{units} : 1 \to \Box\alpha}$$

Operations on events

- $\Diamond$ is a monad:

$$\text{now} : \alpha \to \Diamond\alpha$$

$$\text{join} : \Diamond\Diamond\alpha \to \Diamond\alpha$$

- $\Diamond$ is not a strong monad:
  - time shifting of values would have to be possible:

$$\text{shift} : \alpha \times \Diamond\beta \to \Diamond(\alpha \times \beta)$$

  - would break start time consistency:

$$\text{shift} : \Pi t \,.\, \langle\alpha\rangle \times \langle\Diamond\beta\rangle \to \Sigma t' \in [t, \infty) \,.\, \langle\alpha\rangle[t'/t] \times \langle\beta\rangle[t'/t]$$

- however, $\Diamond$ is $\Box$-strong:

$$\text{age} : \Box\alpha \times \Diamond\beta \to \Diamond(\Box\alpha \times \beta)$$

- sampling can be derived:

$$\text{sample} : \Box\alpha \times \Diamond\beta \to \Diamond(\alpha \times \beta)$$

$$\text{sample} = \Diamond(\text{head} \times \text{id}) \circ \text{age}$$

From S4 to restricted LTL

- until now, we have categorical models for CS4/IS4
- no big surprise:
  - classically, restricted LTL is a specialization of S4
  - intuitionistically, it is too
- classical S4 and restricted LTL differ in their restrictions on the accessibility relation:

S4	reflexive order
restr. LTL	total reflexive order

- add a further operation that ensures totality of time:

$$\mathrm{race} : \Diamond\alpha \times \Diamond\beta \to \Diamond(\alpha \times \beta + \alpha \times \Diamond\beta + \Diamond\alpha \times \beta)$$

- possible outcomes of time comparison represented by the different alternatives:

$=$	$\alpha \times \beta$
$<$	$\alpha \times \Diamond\beta$
$>$	$\Diamond\alpha \times \beta$

$\triangleright$-LTL and its corresponding FRP dialect

- translation of $\triangleright$-formulas into predicate logic formulas:

$$\langle \varphi \triangleright \psi \rangle = \exists t' \in (t, \infty) \,.\, (\forall t'' \in [t, t') \,.\, \langle \varphi \rangle [t''/t]) \wedge \langle \psi \rangle [t'/t]$$

- $\triangleright$ as a type constructor of FRP:

$$\langle \alpha \triangleright \beta \rangle = \Sigma t' \in (t, \infty) \,.\, (\Pi t'' \in [t, t') \,.\, \langle \alpha \rangle [t''/t]) \times \langle \beta \rangle [t'/t]$$

- components of a value of type $\alpha \triangleright \beta$:
  - a finite behavior with values of type α
  - a terminating event with a value of type β
- introduction of weak variant of $\triangleright$ that does not guarantee termination
- notation:

$\triangleright_\bot$	strong variant ($\triangleright$ as defined above)
$\triangleright_\top$	weak variant

- $\Box$ and $\Diamond$ now derivable:

$$\Box \alpha = \alpha \triangleright_\top 0$$

$$\Diamond \beta = \beta + 1 \triangleright_\bot \beta$$

Applications of $\rhd$-types

- $\rhd$-types are useful as such:
  - temperatures from some sensor that may be detached from the computer:

    $$\mathbb{R} \rhd_{\top} 1$$

  - dialog window:

    $$\mathrm{UI} \rhd_{\top} \alpha$$

  etc.
- $\rhd$-types are useful in combination with (co)induction:
  - audio signal that may switch between stereo and mono:

    $$\nu\sigma\,.\,(\mathbb{R} \times \mathbb{R}) \rhd_{\top} \mathbb{R} \rhd_{\top} \sigma$$

  - positions of a pen that might be taken off from the drawing area:

    $$\nu\sigma\,.\,(\mathbb{R} \times \mathbb{R}) \rhd_{\top} 1 \rhd_{\top} \sigma$$

  etc.

The $\triangleright$-functor

- categorical model C is a CCCC
- derive a category U from C:

$$
\begin{aligned}
&\operatorname{Obj} U \\
&\qquad = \operatorname{Obj} C \times \operatorname{Obj} C \times \{\bot, \top\} \\
&\hom((\alpha_1, \beta_1, w_1), (\alpha_2, \beta_2, w_2)) \\
&\qquad = \begin{cases} \hom(\alpha_1, \alpha_2) \times \hom(\beta_1, \beta_2) & \text{if } w_1 \leqslant w_2 \\ \emptyset & \text{otherwise} \end{cases}
\end{aligned}
$$

- $\triangleright$ is a functor from U to C
- notation:

$$\alpha \triangleright_w \beta = \triangleright(\alpha, \beta, w)$$

- applying $\triangleright$ to morphisms allows for several things:
  - mapping of values of the behavior part
  - mapping of value of the terminating event
  - weakening

Comonadic and monadic structure

- $\_ \rhd_w \beta$ is a comonad:

$$\begin{aligned} \text{head} &: \alpha \rhd_w \beta \to \alpha \\ \text{tails} &: \alpha \rhd_w \beta \to (\alpha \rhd_w \beta) \rhd_w \beta \end{aligned}$$

- $\beta = 0$ and $w = \top$ leads to comonadic structure of $\Box$
- $\alpha \rhd_w \_$ is an ideal monad:

$$\text{optjoin} : \alpha \rhd_w (\beta + \alpha \rhd_w \beta) \to \alpha \rhd_w \beta$$

- monad can be derived:

$$\begin{aligned} \text{now} &: \beta \to (\beta + \alpha \rhd_w \beta) \\ \text{join} &: (\beta + \alpha \rhd_w \beta) + \alpha \rhd_w (\beta + \alpha \rhd_w \beta) \to \beta + \alpha \rhd_w \beta \end{aligned}$$

- $\alpha = 1$ and $w = \bot$ leads to monadic structure of $\Diamond$

Monoidal structure

- make U a symmetric monoidal category:

$$(\alpha_1, \beta_1, w_1) \otimes (\alpha_2, \beta_2, w_2) = (\alpha_1 \times \alpha_2, \rho, w_1 \sqcap w_2)$$
$$I = (1, 0, \top)$$

  where

$$\rho = \beta_1 \times \beta_2 + \beta_1 \times \alpha_2 \rhd_{w_2} \beta_2 + \alpha_1 \rhd_{w_1} \beta_1 \times \beta_2$$

- $\rhd$ is a strong symmetric monoidal functor from U to C:

$$\text{merge} : \alpha_1 \rhd_{w_1} \beta_1 \times \alpha_2 \rhd_{w_2} \beta_2 \to \alpha_1 \times \alpha_2 \rhd_{w_1 \sqcap w_2} \rho$$
$$\text{never} : 1 \rhd_\top 0$$

Specializations

- $\rhd$ is a strong symmetric monoidal functor from U to C:

$$\begin{aligned} \text{merge} &: \alpha_1 \rhd_{w_1} \beta_1 \times \alpha_2 \rhd_{w_2} \beta_2 \to \alpha_1 \times \alpha_2 \rhd_{w_1 \sqcap w_2} \rho \\ \text{never} &: 1 \rhd_\top 0 \end{aligned}$$

  where

$$\rho = \beta_1 \times \beta_2 + \beta_1 \times \alpha_2 \rhd_{w_2} \beta_2 + \alpha_1 \rhd_{w_1} \beta_1 \times \beta_2$$

- strong cartesian functor structure of $\Box$:

$$\beta_1 = \beta_2 = 0 \qquad\qquad w_1 = w_2 = \top$$

- from merge to age:

$$\begin{aligned} \beta_1 &= 0 & \qquad w_1 &= \top \\ \alpha_2 &= 1 & \qquad w_2 &= \bot \end{aligned}$$

- from merge to race:

$$\alpha_1 = \alpha_2 = 1 \qquad\qquad w_1 = w_2 = \bot$$

The inverse of merge

- the type of the terminating event:

$$\rho = \beta_1 \times \beta_2 + \beta_1 \times \alpha_2 \rhd_{w_2} \beta_2 + \alpha_1 \rhd_{w_1} \beta_1 \times \beta_2$$

- drop information from the terminating event:

$$\begin{aligned}\mathrm{restrict}_i &: \rho \to \beta_i + \alpha_i \rhd_{w_i} \beta_i \\ \mathrm{restrict}_i &= [\iota_1 \circ \pi_i, \iota_i \circ \pi_i, \iota_{1-i} \circ \pi_i]\end{aligned}$$

- recover the original $\rhd$-values:

$$\begin{aligned}\mathrm{recover}_i &: \alpha_1 \times \alpha_2 \rhd_{w_1 \sqcap w_2} \rho \to \alpha_i \rhd_{w_i} \beta_i \\ \mathrm{recover}_i &= \mathrm{optjoin} \circ (\pi_i \rhd \mathrm{restrict}_i)\end{aligned}$$

- combine the recovered values:

$$\begin{aligned}\mathrm{merge}^{-1} &: \alpha_1 \times \alpha_2 \rhd_{w_1 \sqcap w_2} \rho \to \alpha_1 \rhd \beta_1 \times \alpha_2 \rhd \beta_2 \\ \mathrm{merge}^{-1} &= \langle \mathrm{recover}_1, \mathrm{recover}_2 \rangle\end{aligned}$$

The Temporal Curry–Howard Correspondence

Categorical Semantics for Restricted LTL and FRP

Hybrid Signals

Functional Reactive Dataflow Programming

$\bigcirc$ in LTL and FRP

- use $\mathbb{N}$ as the set of times
- translation of $\bigcirc$-formulas into predicate logic formulas:

$$\langle \bigcirc\varphi \rangle = \langle \varphi \rangle [t + 1/t]$$

- $\bigcirc$ as a type constructor of FRP:

$$\langle \bigcirc\alpha \rangle = \langle \alpha \rangle [t + 1/t]$$

- value of type $\bigcirc\alpha$ is a value of type α occurring at the next time
- semantically, $\bigcirc$ is just a strong cartesian functor:

$$\frac{f : \alpha \to \beta}{\bigcirc f : \bigcirc\alpha \to \bigcirc\beta} \qquad \begin{aligned} \mathrm{unit} &: 1 \to \bigcirc 1 \\ \mathrm{pair} &: \bigcirc\alpha \times \bigcirc\beta \to \bigcirc(\alpha \times \beta) \end{aligned}$$

Deriving the other constructs

- $\square$, $\diamond$, and $\triangleright$ derivable via induction and coinduction:

$$\square\alpha = \nu\sigma\,.\,\alpha \times \bigcirc\sigma$$
$$\diamond\beta = \mu\sigma\,.\,\beta + \bigcirc\sigma$$
$$\alpha \triangleright_{\bot} \beta = \mu\sigma\,.\,\alpha \times \bigcirc(\beta + \sigma)$$
$$\alpha \triangleright_{\top} \beta = \nu\sigma\,.\,\alpha \times \bigcirc(\beta + \sigma)$$

- interesting exercise:
  - derive all operations of $\triangleright$-FRP from the $\bigcirc$-operations
  - proof that the derived operations fulfill the necessary laws

Advanced dataflow programming

- $\bigcirc$-FRP is a kind of dataflow language:
  - streams over α:

$$\Box\alpha$$

  - partial streams over α:

$$(1 + \alpha) \times \nu\sigma \,.\, 1 \rhd_{\top} (\alpha \times \sigma)$$

- more powerful than traditional dataflow languages:
  - productive partial streams over α:

$$(1 + \alpha) \times \nu\sigma \,.\, 1 \rhd_{\bot} (\alpha \times \sigma)$$

  - streams with values of different type

Shifting

- fby operator appends a stream to an initial value:

$$\text{fby} : \alpha \times \Box\alpha \rightarrow \Box\alpha$$

- needs to shift values to the future
- cannot be done implicitely, since it would break start time consistency
- can be made possible by introducing tensorial strength:

$$\text{shift} : \alpha \times \bigcirc\beta \rightarrow \bigcirc(\alpha \times \beta)$$

- simpler operator is sufficient:

$$\text{later} : \alpha \rightarrow \bigcirc\alpha$$

- $\bigcirc$ is now an applicative functor

Programming in Linear Temporal Logic

Wolfgang Jeltsch

Brandenburgische Technische Universität Cottbus
Cottbus, Germany

Seminar talk at the Institute of Cybernetics
Tallinn, Estonia

February 10, 2011