Timed Pattern Matching

FORMATS'14

Doğan Ulus[1] Thomas Ferrère[1] Eugene Asarin[2] Oded Maler[1]

[1] VERIMAG, CNRS and the University of Grenoble-Alpes
[2] LIAFA, University of Paris Diderot / CNRS

September 10, 2014

Pattern matching

Problem (Pattern matching for regular expressions)

Given a word $w \in \Sigma^*$ and a regular expression φ find subwords v of w that match φ.

Classical solutions

- algorithms: often based on automata
- tools: `grep`, `sed`, programming languages: PERL, PYTHON etc.

Timed pattern matching

Why do it?

- Monitoring of embedded systems
- Hardware specification languages
- Timed texts (music, speech)
- Quantitative pattern matching (texts, DNA sequences)

Why is it not straightforward?

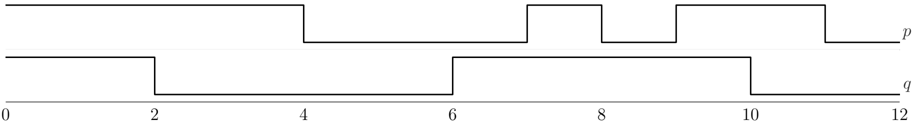
- Classical algorithms based on (implicit) determinization
  $\Rightarrow$ heavy subset construction
- Timed automaton does not explicitely provide all matches

Example

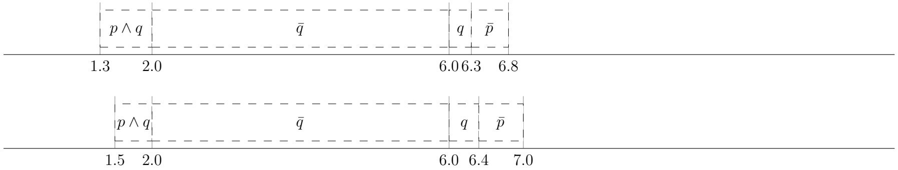
- Expression:

$$\varphi = \langle (p \wedge q) \cdot \bar{q} \cdot q \rangle_{[4,5]} \cdot \bar{p}$$

- Signals:



- Matches:



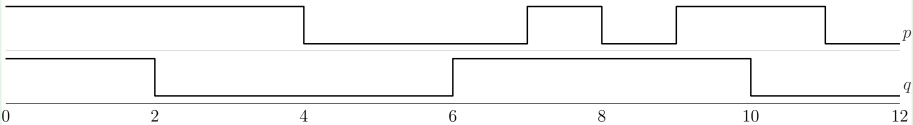
Outline

1. Problem statement
2. The solution
3. Practical algorithmics
4. Experiments

Outline

Timed Boolean signals

Example



Definition (Boolean Signals)

Let $\mathbb{T} = [0, d]$ (time domain).

- a *Boolean signal* is a function $w : \mathbb{T} \to \mathbb{B}^m$.
- can be represented using m Boolean variables $p(t)$.
- assumption: the number of discontinuities is finite.

Timed regular expressions – to specify sets of signals

Definition (Syntax of TRE)

$$\varphi := \epsilon \mid p \mid \overline{p} \mid \varphi \cdot \varphi \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \varphi^* \mid \langle \varphi \rangle_I$$

p propositional variable, I integer bounded interval

Definition (Semantics of TRE)

$$\begin{array}{lll}
(w,t,t') \models \epsilon & \leftrightarrow & t = t' \\
(w,t,t') \models p & \leftrightarrow & t < t' \text{ and } \forall t'' \in (t,t'),\ p[t''] = 1 \\
(w,t,t') \models \overline{p} & \leftrightarrow & \ldots \\
(w,t,t') \models \varphi \cdot \psi & \leftrightarrow & \exists t'' \in (t,t'),\ (w,t,t'') \models \varphi \text{ and } (w,t'',t') \models \psi \\
(w,t,t') \models \varphi \vee \psi & \leftrightarrow & (w,t,t') \models \varphi \text{ or } (w,t,t') \models \psi \\
(w,t,t') \models \varphi \wedge \psi & \leftrightarrow & \ldots \\
(w,t,t') \models \varphi^* & \leftrightarrow & \exists k \geq 0,\ (w,t,t') \models \varphi^k \\
(w,t,t') \models \langle \varphi \rangle_I & \leftrightarrow & t' - t \in I \text{ and } (w,t,t') \models \varphi
\end{array}$$

Timed regular expressions – to specify sets of signals

Definition (Syntax of TRE)

$$\varphi := \epsilon \mid p \mid \overline{p} \mid \varphi \cdot \varphi \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \varphi^* \mid \langle \varphi \rangle_I$$

p propositional variable, I integer bounded interval

Definition (Semantics of TRE)

$$\begin{array}{lll}
(w,t,t') \models \epsilon & \leftrightarrow & t = t' \\
(w,t,t') \models p & \leftrightarrow & t < t' \text{ and } \forall t'' \in (t,t'),\ p[t''] = 1 \\
(w,t,t') \models \overline{p} & \leftrightarrow & \ldots \\
(w,t,t') \models \varphi \cdot \psi & \leftrightarrow & \exists t'' \in (t,t'),\ (w,t,t'') \models \varphi \text{ and } (w,t'',t') \models \psi \\
(w,t,t') \models \varphi \vee \psi & \leftrightarrow & (w,t,t') \models \varphi \text{ or } (w,t,t') \models \psi \\
(w,t,t') \models \varphi \wedge \psi & \leftrightarrow & \ldots \\
(w,t,t') \models \varphi^* & \leftrightarrow & \exists k \geq 0,\ (w,t,t') \models \varphi^k \\
(w,t,t') \models \langle \varphi \rangle_I & \leftrightarrow & t' - t \in I \text{ and } (w,t,t') \models \varphi
\end{array}$$

Problem statement

Definition (Match-set)

For a signal w and an expression φ the match-set is

$$\mathcal{M}(\varphi, w) := \{(t, t') \in \mathbb{T} \times \mathbb{T} : (w, t, t') \models \varphi\}$$

Problem (Timed pattern matching)

Given a signal and an expression compute the match-set.

Problem statement

Definition (Match-set)

For a signal w and an expression φ the match-set is

$$\mathcal{M}(\varphi, w) := \{(t, t') \in \mathbb{T} \times \mathbb{T} : (w, t, t') \models \varphi\}$$

Problem (Timed pattern matching)

Given a signal and an expression compute the match-set.

Outline

Data structure

Definition (2d zone)

A 2d zone is a subset of $\mathbb{R}^2$ described by inequalities

$$a < x < b \qquad c < y < d \qquad e < x - y < f$$

with a, b, c, d, e, f real constants.

About 2d zones

- convex polygons (with up to 6 edges)
- only vertical, horizontal and diagonal edges

Data structure

Definition (2d zone)

A 2d zone is a subset of $\mathbb{R}^2$ described by inequalities

$$a < x < b \qquad c < y < d \qquad e < x - y < f$$

with a, b, c, d, e, f real constants.

About 2d zones

- convex polygons (with up to 6 edges)
- only vertical, horizontal and diagonal edges

Main result

Theorem

The match-set $\mathcal{M}(\varphi, w)$ is computable as a finite union of 2d zones.

Proof principle

Structural induction over φ

$\Rightarrow$ recursive algorithm over the expression syntactic tree

Main result

Theorem

The match-set $\mathcal{M}(\varphi, w)$ is computable as a finite union of 2d zones.

Proof principle

Structural induction over φ
$\Rightarrow$ recursive algorithm over the expression syntactic tree

Structural induction: base cases

Empty word

$\mathcal{M}(\epsilon, w) = \{(t, t') : t = t'\}$ and the diagonal is a zone

A literal

$\mathcal{M}(p, w)$ is a union of triangles over the diagonal

Example

Structural induction: base cases

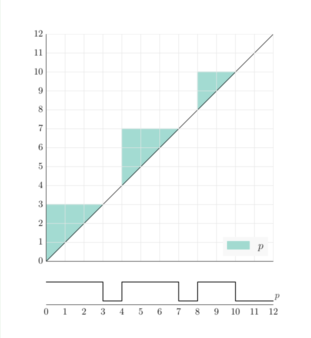
Empty word

$\mathcal{M}(\epsilon, w) = \{(t, t') : t = t'\}$ and the diagonal is a zone

A literal

$\mathcal{M}(p, w)$ is a union of triangles over the diagonal

Example



Structural induction: Boolean closure

Intersection

Zones are closed under intersection

Union

Unions of zones are closed under union

Time restriction

$\mathcal{M}(\langle\varphi\rangle_I, w) = \mathcal{M}(\varphi, w) \cap \{(t, t') : t' - t \in I\}$
is an intersection of zones

Example

Structural induction: Boolean closure

Intersection

Zones are closed under intersection

Union

Unions of zones are closed under union

Time restriction

$\mathcal{M}(\langle\varphi\rangle_I, w) = \mathcal{M}(\varphi, w) \cap \{(t, t') : t' - t \in I\}$
is an intersection of zones

Example

Structural induction: Boolean closure

Intersection

Zones are closed under intersection

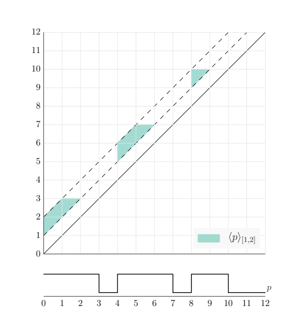
Union

Unions of zones are closed under union

Time restriction

$\mathcal{M}(\langle\varphi\rangle_I, w) = \mathcal{M}(\varphi, w) \cap \{(t,t') : t' - t \in I\}$
is an intersection of zones

Example



Structural induction: concatenation

Lemma (Concatenation = composition of binary relations)

$$\mathcal{M}(\varphi \cdot \psi, w) = \mathcal{M}(\varphi, w) \circ \mathcal{M}(\psi, w)$$

Composition preserves zones

$$(t, t') \in z_1 \circ z_2 \leftrightarrow \exists t'' : (t, t'') \in z_1 \wedge (t'', t') \in z_2.$$

Can be obtained using standard zones operations.

Structural induction: concatenation

Lemma (Concatenation = composition of binary relations)

$$\mathcal{M}(\varphi \cdot \psi, w) = \mathcal{M}(\varphi, w) \circ \mathcal{M}(\psi, w)$$

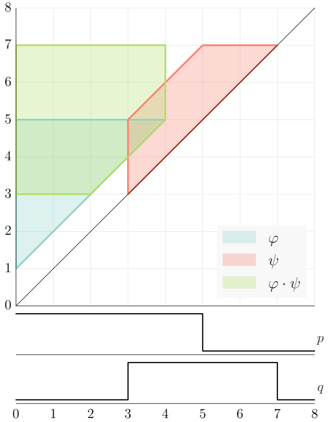
Composition preserves zones

$$(t, t') \in z_1 \circ z_2 \leftrightarrow \exists t'' : (t, t'') \in z_1 \wedge (t'', t') \in z_2.$$

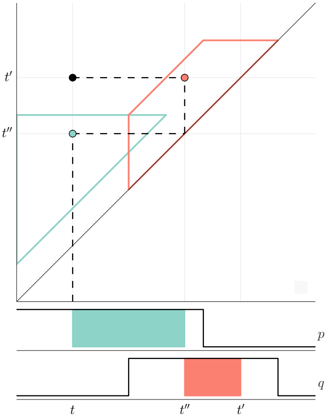
Can be obtained using standard zones operations.

Example

Concatenation of $\varphi = \langle p \rangle_{[1,\infty]}$ with $\psi = \langle q \rangle_{[0,2]}$



zones for φ, ψ and $\varphi \cdot \psi$



a match (t, t') of $\varphi \cdot \psi$

Structural induction: Kleene star

Definition (size of a signal)

$\sigma(w)$ – minimal number of intervals that:

- cover the time domain of w
- are of length < 1
- w is constant in each interval

Lemma (Star is bounded)

If $\sigma(w) = k$ then $\mathcal{M}(\varphi^*, w) = \mathcal{M}(\varphi^{\leq 2k+1}, w)$.

Structural induction: Kleene star

Definition (size of a signal)

$\sigma(w)$ – minimal number of intervals that:

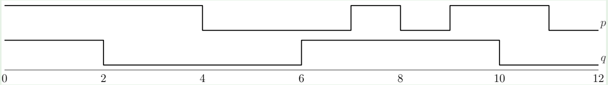
- cover the time domain of w
- are of length < 1
- w is constant in each interval

Lemma (Star is bounded)

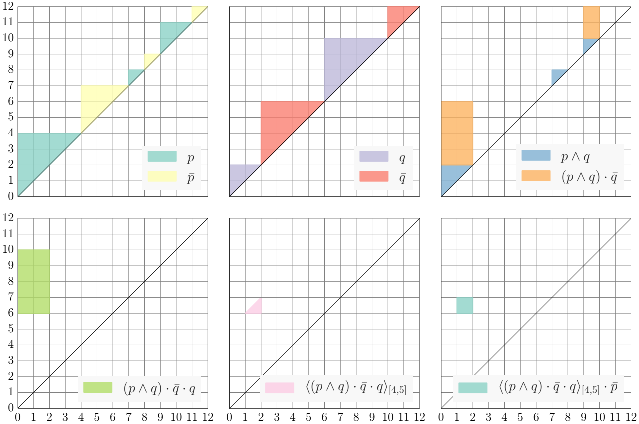
If $\sigma(w) = k$ *then* $\mathcal{M}(\varphi^*, w) = \mathcal{M}(\varphi^{\leq 2k+1}, w)$.

Example (from introduction)

$$\langle (p \wedge q) \cdot \bar{q} \cdot q \rangle_{[4,5]} \cdot \bar{p}$$



Inductive steps:



Outline

Overall program structure

function $\textsc{zones}(\varphi, w)$
select φ
case ϵ, p, $\overline{p}$**:**
  $Z_\varphi := \textsc{atom}(\varphi, w)$
case $\bullet\,\psi$**:**
  $Z_\psi := \textsc{zones}(\psi, w)$
  $Z_\varphi := \textsc{combine}(\bullet, Z_\psi)$
case $\psi_1 \bullet \psi_2$**:**
  $Z_{\psi_1} := \textsc{zones}(\psi_1, w)$
  $Z_{\psi_2} := \textsc{zones}(\psi_2, w)$
  $Z_\varphi := \textsc{combine}(\bullet, Z_{\psi_1}, Z_{\psi_2})$
end select
return Z_φ

$\textsc{combine}$ implemented for all operations $\bullet \in \{\cdot, \vee, \wedge, ^*, \langle\,\rangle_I\}$

An important issue for intersection and concatenation

Does it explode?

$$\bigcup_i z_i \cap \bigcup_j z'_j = \bigcup_{ij} z_i \cap z'_j \qquad \bigcup_i z_i \circ \bigcup_j z'_j = \bigcup_{ij} z_i \circ z'_j$$

Not in practice

- Most resulting zones are empty.
- *Plane-sweep* algorithm: sorting zones by start / end time allows to avoid most empty operations

An important issue for intersection and concatenation

Does it explode?

$$\bigcup_i z_i \cap \bigcup_j z'_j = \bigcup_{ij} z_i \cap z'_j \qquad \bigcup_i z_i \circ \bigcup_j z'_j = \bigcup_{ij} z_i \circ z'_j$$

Not in practice

- Most resulting zones are empty.
- *Plane-sweep* algorithm: sorting zones by start / end time allows to avoid most empty operations

Star: computed by squaring

function $\textsc{combine}({}^*, Z)$
$X := \emptyset$
$Y := Z$
while $\exists z \in Y, \forall z' \in X, z \not\subseteq z'$ **do**
  $X := X \cup Y \cup \ \textsc{combine}(\cdot, X, Y)$
  $Y := \textsc{combine}(\cdot, Y, Y)$
end while
return $X \cup \{\varepsilon\}$

Invariant

After k iterations $\cup X_k = (\cup Z)^{<2^k}$ and $\cup Y_k = (\cup Z)^{2^k}$

Lemma

The algorithm stops after $\log(|Z| + \Delta(Z))$ *iterations*

Star: computed by squaring

function $\textsc{combine}(\,^*, Z)$
$X := \emptyset$
$Y := Z$
while $\exists z \in Y, \forall z' \in X,\ z \not\subseteq z'$ **do**
  $X := X \cup Y \cup\ \textsc{combine}(\cdot, X, Y)$
  $Y := \textsc{combine}(\cdot, Y, Y)$
end while
return $X \cup \{\varepsilon\}$

Invariant

After k iterations $\cup X_k = (\cup Z)^{<2^k}$ and $\cup Y_k = (\cup Z)^{2^k}$

Lemma

The algorithm stops after $\log(|Z| + \Delta(Z))$ *iterations*

Outline

Experimental setting

- Program: Python implementation calling IF zones library.
- Signals: random signals of given
  - variability $\mathcal{V}$ (discontinuities per time unit)
  - length $\mathcal{L}$ (number of time units)
- Expression: $\varphi = \langle (\langle p \cdot \bar{p} \rangle_{[0,10]})^* \wedge (\langle q \cdot \bar{q} \rangle_{[0,10]})^* \rangle_{[80,\infty]}$

A table of results

$\mathcal{V}$	$\mathcal{L}$	$\lvert Z_\varphi \rvert$	Time (s)
0.025	40000	0	0.08
0.025	80000	0	0.17
0.025	160000	0	0.37
0.05	40000	0	0.27
0.05	80000	0	0.60
0.05	160000	0	1.27
0.075	40000	1	0.64
0.075	80000	4	1.40
0.075	160000	5	2.88
0.1	40000	10	1.35
0.1	80000	23	2.73
0.1	160000	47	5.83

Conclusion

Contribution

- The problem of timed pattern matching stated and solved
- A prototype tool developed
- Experiments on synthetic data witness scalability

Perspectives

- Online matching / monitoring
- New operators (mixing expressions and logic, as in hardware specification languages)
- Extending from signals to timed event sequences