On the Reaction Time of Some Synchronous Systems

Ilias Garnier, Christophe Aussagès, Vincent David, Guy Vidal-Naquet

June 9, 2011
Plan

- Context
- Formal model
- Reaction time
- Compositionality of reaction time
- Applications to refinement
Real-time embedded systems: computer systems interacting with the physical environment under strong timing constraints.

- Studied systems: synchronous system
- Study of delay of significant reaction time, called here reaction time
Motivation: introducing reaction time

Analogy: wave through a physical medium

Reaction time \( \approx \) delay between input and functionally dependent output.
Introducing synchronous systems

Our study concentrates on the Moore model of synchronous computation.

**Synchronous round of computation**

- Observable output part of the state
- Next state depends on input and current state

Synchronous computation = (possibly infinite) succession of rounds. When state space is finite, defines a *Moore machine.*
Investigating reaction time

- What does it *mean* to **react** to an input?
  - existence of an “observable effect”
  - observable state is a function of the input
  - \( \Rightarrow \) need a **formal notion of observational equivalence**

- What is a reaction **time**?
  - in our case, number of transitions until observable effect

- Is it **compositional**?
  - in this presentation, study of sequential composition
Moore machines

Moore machine = FSM where states are labelled with output. Let $\text{In}$ set of inputs, $\text{Out}$ set of outputs, $\mathcal{M} = \langle \text{In}, \text{Out}, Q, E, \text{out} \rangle$.

- $Q$ finite set of states,
- $E \subseteq Q \times \text{In} \times Q$ edges, Input-enabled
- $\text{out} : Q \rightarrow \text{Out}$ outputs.

Quick example: outputs 0 on $\text{ff}$, outputs 1 on $\text{tt}$:

```
|   | tt | ff | ff | tt | 0 |
|---|----|----|----|----|---|
| 0 | 0  | 1  | 2  | 3  | 4 |
| 0 | 1  | 0  | 0  | 1  |
```
Bigger example

A synchronous program and its corresponding machine

\[ x_0 := \text{true}; \]
\[ x_1 := \text{true}; \]
\[ \text{next(true)}; \]
\[ \text{while true do} \]
\[ \quad \text{next}(x_0); \]
\[ \quad x_0 := x_1; \]
\[ \quad x_1 := \text{input} \]
\[ \quad \text{done} \]
Observational equivalence

In order to define observational difference, we need to define observational equivalence. We choose bisimilarity (finest-grained). Let $p, q$ be two states.

$$ p \sim q \iff \text{out}(p) = \text{out}(q) \land $$
$$ \forall a, \forall p \xrightarrow{a} p', \exists q \xrightarrow{a} q', p' \sim q' \land $$
$$ \forall a, \forall q \xrightarrow{a} q', \exists p \xrightarrow{a} p', p' \sim q' $$

Intuitive view: $p \sim q$ is simply equality on infinite unfoldings starting from $p$ and $q$. 
Recall that reaction time $\approx$ delay between input and related output. “Related” means functional dependency between input and output.

$f : D \rightarrow E$ a total function. We have:

\[
f \text{ constant } \iff f(D) = \{ e \}
\]

\[
f \text{ non-constant } \iff \exists d_1, d_2, d_1 \neq d_2 \land f(d_1) \neq f(d_2)
\]

$(d_1, d_2)$ allows to prove $f$ non-constant: separating pair.
Reaction time: states of Moore machines

state = function from inputs to set of states

\[
\begin{array}{c}
\text{init} \\
\text{in}_1 \\
\text{in}_2 \\
p_1 \\
p_2 \\
p_3 \\
q_1 \\
q_2 \\
\end{array}
\]

reached sets \( P, Q \) not equivalent when \( \exists \ p_i \in P \) s.t. \( \forall q_j \in Q, \ p_i \sim q_j \) (or the other way around).

Reaction time

Given \( p \sim q \), we want to study how many transitions are needed to have an \textit{observable effect}. 
Observable effects extracted from proofs of non-bisimilarity

Inductive definition of $p \sim q$:

**BASE** \[
\text{out}(p) \neq \text{out}(q) \rightarrow p \sim q
\]

**IND** \[
\exists p \xrightarrow{a} p', \forall q \xrightarrow{a} q', p' \sim q' \lor \exists q \xrightarrow{a} q', \forall p \xrightarrow{a} p', p' \sim q'
\]

\[
p \sim q
\]

$p = p_0 \xrightarrow{a_0} p_1 \xrightarrow{a_1} p_2 \ldots p_n$

$q = q_0 \xrightarrow{a_0} q_1 \xrightarrow{a_1} q_2 \ldots q_n$

s.t. $\text{out}(p_n) \neq \text{out}(q_n)$.

**Observable effect, separator**

The pair $(\text{out}(p_n), \text{out}(q_n))$ is an observable effect. The word $a_0.a_1 \ldots a_{n-1}$ is called a separator.
Observable effects example

Unfolding of state $q_0$ of delay program.

In this case, every word of length 2 is a separator.
An unfolding where \( \texttt{ff}^* \) does not contain separators.
From obs. effects to reaction time

1. Reactivity of a state $q$ (≡ non-constantness) $\iff \exists$ separating pair of inputs $in_1 \neq in_2$ s.t. $q \xrightarrow{in_1} Q_1$, $q \xrightarrow{in_2} Q_2$ and set $Q_1$ not equivalent to set $Q_2$

2. Any proof of $q_i \sim q_j$ yields an observable effect triggered by a particular input word called separator;
   - in a non-deterministic fashion: may-separator
   - for all runs of the separator: must-separator

Reaction time

Exists iff all infinite input words are prefixed by a must-separator. It is the worst-case number of transitions necessary to obtain the first observable effect.
Computing separators and observable effects in order to obtain reaction time is costly.

**Compositionality property**

Given machines $M_1$, $M_2$ and binary composition operation $C$, can we compute the observable effects of the composed machine $C(M_1, M_2)$ without performing the whole state-space exploration?

I.e. how easily can we compute the observable effects of $C(M_1, M_2)$ given those of $M_1$ and $M_2$?

In our case, $C = \circ$, the sequential composition.
Sequential composition of states

Compound state space = cartesian product. Compound transition exists iff receiving transition labelled by output of sending state.

\[
\begin{align*}
q_f \xrightarrow{in_f} q'_f & \quad q_g \xrightarrow{out(q_f)} q'_g \\
(q_f, q_g) \xrightarrow{in_f} (q'_f, q'_g) & \quad \text{out}(q_g \circ q_f) = \text{out}(q_g)
\end{align*}
\]
Necessity of a well-behaved composition

States $q_6$ becomes unreachable in composition, states $q_2$ and $q_3$ no more separable. Sequential composition yields

Moore machines: to compute all compound obs. effects, full state space search necessary.

Possible solution: under-approximate obs. effects until they are compositional.
Approximating obs. effects

Core idea: reduce branching property to a linear one.

**linear time criterion:** \( \text{init}_2 \circ \text{init}_1 \) reactive if \( \bigcup \text{obs.eff.} \subseteq \bigcap \text{SP}(q_i) \)

**Problem:** sets of separating pairs and obs. effects can be complex.
Determinism and separability

**Solution 2:** restriction to sep. pairs and obs. effects present for all input words, i.e. **deterministic sep. pairs and obs. effects.**

\[
\begin{align*}
\text{context tt.bool}^*, \text{ obs. effect } & \emptyset.(0,1).(0,1) \ldots \\
\text{context ff.bool}^*, \text{ obs. effect } & \emptyset.(1,2).(0,1) \ldots \\
\text{deterministic obs. effect: } & \emptyset.\emptyset.(0,1) \ldots
\end{align*}
\]
We aim at developing real-time embedded systems by validated stepwise refinement, s.t. the refinement step i) preserves reaction time and ii) is a congruence.

- Simulation as refinement, i.e. \( p \) refines \( q \) if \( p \) has less possible behaviours than \( q \) (tree inclusion)

- Problem: does simulation preserve observable effects? Is it compositional?
Observable effect preservation

Simulation doesn’t preserve non-deterministic separators.

Corollary: observable effects and reaction time not preserved in general. Solution: consider a subset of observable effects generated by proofs of non-bisimulation which do not rely on non-determinism. I.e. Deterministic obs. effects are preserved by refinement.
Reaction time can be defined in terms of observable effects, which correspond to proofs of non-bisimulation.

In general,
- not preserved by sequential composition
- not preserved by refinement.

Det. obs. effects preserved.