Simple Synthesis of Reactive Systems with Tolerance for Unexpected Environmental Behavior

Shigeki Hagihara\textsuperscript{1} Atsushi Ueno\textsuperscript{2} Takashi Tomita\textsuperscript{3} Masaya Shimakawa\textsuperscript{1} Naoki Yonezaki\textsuperscript{4}

\textsuperscript{1}Tokyo Institute of Technology

\textsuperscript{2}Nintendo Co., Ltd.

\textsuperscript{3}Japan Advanced Institute of Science and Technology

\textsuperscript{4}The Open University of Japan

FormaliSE’16
Timing critical systems in our society

- Ex. systems controlling
  - elevators,
  - vending machines,
  - nuclear power plants, and
  - air traffic.

- Most of these systems are open systems which respond to requests from the environment at an appropriate time.

- Unlike general business software, fatal accidents cause severe damages, and it is important to ensure safety of the system.
  - Such systems became complex. It became unmanageable to develop the systems without flaws.

- One of the solutions is to use tools based on formal methods.
  - Recently, formal methods have become essential tools for developing safety critical open systems.
Formal methods for developing safety critical open systems

- Model checking
  - a method for checking whether models of systems satisfy specifications,
  - popular technique for developing practical systems such as systems controlling car, airplane, and rockets.

- System synthesis from specification
  - a method for checking whether there exists a system that satisfies specifications and synthesizing transition systems representing the system if there exists
  - it is not necessary to design systems manually, and systems which satisfy specifications can be synthesized automatically.

In this work, we focus on synthesis of open systems.
A formalization of open systems.

RS is a triple \( \langle X, Y, r \rangle \), where
- \( X \) is a set of events caused by an environment,
- \( Y \) is a set of events caused by the system and
- \( r : (2^X)^* \rightarrow 2^Y \) is a reaction function.
Realizability

Reactive system specifications must satisfy realizability.

Realizability: there exists a reactive system such that for any environmental events of any timing, the reactive system produces system events such that the specification holds.

Definition (Realizability)

A specification $\varphi$ is realizable if the following holds:

$$\exists RS \forall \tilde{i} (\text{behave}_{RS}(\tilde{i}) \models \varphi),$$

where $\tilde{i}$ is an infinite sequence of sets of environmental events, i.e., $\tilde{i} \in (2^I)^\omega$. $\text{behave}_{RS}(\tilde{i})$ is the infinite behavior by $RS$ for $\tilde{i}$.

Complexity

- Realizability problem is 2EXPTIME-complete [PR89, Ros92].
Several tools have been proposed for checking realizability of specifications written LTL
- Lily
- AcaciaPlus
- Unbeast
- Our implementation (No name currently)

These tools can synthesize reactive system which is an evidence of realizability if the specification is realizable.
Example: Synthesis of a control system for a door

<table>
<thead>
<tr>
<th>Example (A specification of a control system for a door)</th>
</tr>
</thead>
<tbody>
<tr>
<td>• If the open button is pushed, the door opens at the next time unit.</td>
</tr>
<tr>
<td>$\varphi_{sys1} = G(open \rightarrow Xd)$</td>
</tr>
<tr>
<td>• If the close button is pushed, the door eventually closes.</td>
</tr>
<tr>
<td>$\varphi_{sys2} = G(close \rightarrow XF \neg d)$</td>
</tr>
</tbody>
</table>

(environmental events) open, close: ‘the open or close button is pushed,’

(system event) $d$: the door is open.
The specification of this example is not realizable.

- Let us consider the case that the close button is held (i.e., pushed continuously) after the open button is pushed.
  - if the system eventually opens the door, the behavior contradicts $\varphi_{sys2}$,
  - if the system always closes the door, the behavior contradicts $\varphi_{sys1}$,
  $\Rightarrow$ The collision of $\varphi_{sys1}$ and $\varphi_{sys2}$ results in $\varphi$ being unrealizable.

By adding the following assumption to the specification, the specification become realizable, i.e. $\varphi_{env} \rightarrow \varphi_{sys1} \land \varphi_{sys2}$ is realizable.

**Assumption about environmental behavior**

The buttons cannot be pushed simultaneously due to a physical constraint.

$$\varphi_{env} = G(\neg open \lor \neg close)$$
Synthesis from the specification of the door

This transition system is assured to satisfy the specification of the door.
Safraless synthesis method [KV05] (refined version [SF07, FJR09])

(1) We construct a universal coBüchi automaton $A_\varphi$, which only accepts the set of behaviors that satisfies $\varphi$.

(2) $k = 0$ and we iterate the following steps (a) and (b).

(a) We regard $A_\varphi$ as a $k$-coBüchi automaton, and determinize it to obtain a safety game $SG_k$.

(b) Then we compute a winning region $WR_k$ on $SG_k$. If the initial state is included in $WR_k$, we go to step (3), otherwise $k = k + 1$ and we return to (a).

(3) We derive a strategy from $WR_k$ as a resulting reactive system.
Motivation

- From $\varphi_{env} \rightarrow \varphi_{sys}$,
  - we can synthesize a reactive system, whose behavior satisfies $\varphi_{sys}$ as long as the behavior of the environment satisfies $\varphi_{env}$.
  - However, if the behavior of the environment does not satisfy $\varphi_{env}$, there is no restriction on the behavior of the synthesized reactive system, meaning that it may do nothing.
- In a real-world setting, it is often difficult to ensure that the behavior of the environment satisfies the desired constraints for the environment,

It is desirable for reactive systems to behave in such a way as to satisfy the given constraint $\varphi_{sys}$ as much as possible, even if the behavior of the environment does not satisfy $\varphi_{env}$.
Goal

We provide a simple definition of environmental tolerance, and propose a method for synthesizing reactive systems with environmental tolerance based on our definition.

- In our synthesis method, we devise a way to derive a winning strategy from a winning region.
Towards defining Environmental tolerance

Let $RS$ be a realization of $\varphi_{env} \rightarrow \varphi_{sys}$.

### Intention of environmental tolerance

- Even for environmental behavior not satisfying $\varphi_{env}$, $RS$ behaves in such a way as to satisfy $\varphi_{sys}$ as much as possible.

- Several definitions for environmental tolerance were proposed in [BCG$^+$14] and [Ehl11]. In these works, environmental tolerance was defined by using the ratio of the number of violations of $\varphi_{sys}$ to the number of violations of $\varphi_{env}$.

We give a simpler quantitative definition for environmental tolerance, where only the number of violations of $\varphi_{sys}$ is considered. To define this, we use the concept of mean-payoff to represent how often the behavior of $RS$ satisfies $\varphi_{sys}$. 
A mean-payoff expression in LTL$^{mp}$ represents *quantitative temporal properties*, e.g.,

<table>
<thead>
<tr>
<th>Term</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>MP($t$)</td>
<td>The <em>limit inferior of mean-payoffs</em> under a payoff setting $t$.</td>
</tr>
<tr>
<td>$1_\psi$</td>
<td>Payoff 1 is gained for every step satisfying $\psi$.</td>
</tr>
</tbody>
</table>

- $[[\text{MP}(1_{e_1})]]_{\sigma} \geq 1/2$: the long-run frequency of event $e_1$ is bounded below by $1/2$.
- $[[\text{MP}(\neg 2 \cdot 1_{e_2} - 10 \cdot 1_{e_3})]]_{\sigma} \geq -4$: the long-run average of costs is bounded above by 4, when it costs 2 and 10 for each occurrences of events $e_2$ and $e_3$, respectively.
For $\varphi_{sys} = G\psi_1 \land \ldots \land G\psi_n$, we obtain a payoff term $t_{\varphi_{sys}} = \sum_i c_i \cdot 1_{\psi_i}$.

**Definition (Degree of environmental tolerance)**

Let $RS$ be a reactive system, $\varphi_{sys}$ be a system constraint for $RS$, and $t_{\varphi_{sys}}$ be a payoff term. The degree of environmental tolerance of $RS$ for $t_{\varphi_{sys}}$ is defined as follows.

$$Tol(RS, t_{\varphi_{sys}}) = \min_{\tilde{i} \in (2^\omega \omega)} [MP(t_{\varphi_{sys}})]_{\text{behave}_{RS}(\tilde{i})}$$

If $Tol(RS, t_{\varphi_{sys}})$ is a large value, we call $RS$ a reactive system with environmental tolerance.
Let \( \varphi = \varphi_{env} \rightarrow \varphi_{sys} \) be a specification, and \( t \) be a payoff term obtained from \( \varphi_{sys} \).

1. Based on the refined Safraless synthesis method, we construct \( WR_k \) from \( \varphi \), where \( k \) is a number that obtains realization.

2. We construct mean-payoff game \( MG_t \) that can be considered equivalent to \( t \).

3. We compute the synchronized product of \( WR_k \) and \( MG_t \) and obtain mean-payoff game \( MG(t, \varphi) \).

4. We compute the optimal winning strategy of \( MG(t, \varphi) \) and output it.
We restrict any temporal operators occurring in any formula in $t$ to be the $X$ operator.

- 'Until' formula $\implies$ bounded 'Until' formula

If we use the boundary $n = 2$ for the bounded Until operator, we use the following formula with the $X$ operator instead of $fUg$.

$$fUg \approx fU^2g = g \lor (f \land Xg) \lor (f \land Xf \land XXg)$$
Constructing mean-payoff game

- Since $t_\psi$ contains an operator $X$ only, such as $\psi = a \rightarrow (Xb \lor XXb)$, then determination of whether a payoff is earned depends on the behavior occurring within the time interval with length equal to the depth of $X$ (in this case, the depth is 2).

- Instead of seeing a behavior at a time $d$ later, by assigning a payoff according to a time $d$ before, we can easily construct a mean-payoff game $MG_t$, which is equivalent to $t$. 
we might find a reactive system $rs'$, which has a higher degree $\text{deg}' > \text{deg}$ of environmental tolerance.

However, the representation size of $rs'$ is much larger than that of $rs$, and the computational cost for synthesis of $rs'$ is much larger than that of $rs$.

There is a trade-off between size, cost and degree of environmental tolerance of synthesized reactive systems.
Implementation and Evaluation

- We implemented our method and used the LTL3BA tool [BKRS12] to convert the LTL formulae into universal coBüchi automata.
- We describe syntheses of reactive systems from a simple specification and a more practical specification at a non-trivial scale.
  - Using the minimal $k$ that allows realization to be obtained, we compute a winning region $WR_k$, and deduce the optimal strategy on $WR_k$ using an algorithm proposed in [ZP96], where we compute convergence of values on states.
  - For this algorithm, we introduce a heuristic to check the optimality of the tentative value, after the value has almost converged.
The specification for a reactive system controlling a door.

\[ \varphi_{env} \rightarrow \varphi_{sys1} \land \varphi_{sys2}, \text{ where} \]

- \( \varphi_{env} = G(\neg open \lor \neg close) \)
- \( \varphi_{sys1} = G(open \rightarrow Xd), \) and \( \varphi_{sys2} = G(close \rightarrow XF\neg d). \)

Here, we approximate \( F\neg d \) as \( \neg d \lor X\neg d \), and obtain the following payoff term \( t \) from \( \varphi_{sys} \). (we use 1 for each \( c_{\varphi_{sysi}} \).)

- \( t = 1_{open \rightarrow Xd} + 1_{close \rightarrow X(\neg d \lor X\neg d)} \)
Results of synthesis

A reactive system synthesized with our method ($rs_1$, left) and a reactive system that does not consider environmental tolerance ($rs_2$, right).

- Synthesis of both $rs_1$ and $rs_2$ completed within a second.
- $rs_1$ is environmentally more tolerant than $rs_2$.
  - $Tol(rs_1, t) = 3/2$. $Tol(rs_2, t) = 1$. 
The most important difference between $rs_1$ and $rs_2$ is their behavior after the unexpected occurrence of environmental events $\{\text{open, close}\}$. For $\tilde{i}_1 = (\{\text{open, close}\})^\omega$, 

$$\text{behave}_{rs_1}(\tilde{i}_1) = \{\text{open, close, d}\}(\{d\}\{\text{open, close}\})^\omega$$

$$\text{behave}_{rs_2}(\tilde{i}_1) = (\{\text{open, close, d}\}\{d\})^\omega$$

- $[\text{MP}(t)]_{\text{behave}_{rs_1}(\tilde{i}_1)} = 2$.
  - the constraints ($\text{open} \rightarrow Xd$) and ($\text{close} \rightarrow X(\neg d \lor X\neg d)$) are always satisfied.
- $[\text{MP}(t)]_{\text{behave}_{rs_2}(\tilde{i}_1)} = 3/2$.
  - ($\text{open} \rightarrow Xd$) is always satisfied, while ($\text{close} \rightarrow X(\neg d \lor X\neg d)$) is only satisfied with a frequency of $1/2$.

We confirmed that the obtained reactive system has environmental tolerance.
A specification of an $m$-floor elevator system

- $m + 2$ environmental events, $2m + 4$ system events, and $12m + 8$ temporal formulae.
- The specification consists of
  - a specification for each floor $\cdots (a)$ divided into
    - functional requirements for $k$-th floor $\cdots (akf)$, and
    - non-functional requirements for $k$-th floor $\cdots (akn)$
      (a specification for $k$-th floor $(ak) = (akf) \cup (akn)$)
  - a specification for a door of the lift $\cdots (b)$
  - a specification of the physical constraints $\cdots (c)$
A specification of an $m$-floor elevator system

### Environmental events

- $\text{LocBtn}_i (i = 1..m)$, //Request button at $i$th floor is pushed.
- $\text{OpenBtn}$, //Open button in the lift is pushed.
- $\text{CloseBtn}$ //Close button in the lift is pushed.

### System events

- $\text{Loc}_i (i = 1..m)$, //The lift is located at $i$th floor.
- $\text{ReqLoc}_i (i = 1..m)$, //The lift is requested to go to $i$th floor.
- $\text{Open}$, //The door is open.
- $\text{Movable}$, //The lift can move.
- $\text{OpenTimedOut}$, //The time-limit that the door can open //has past.
- $\text{ReqOpen}$ //The door is requested to open.
(a) a specification for each floor

(af) functional requirements

//If a request button is pushed, the lift eventually go there.
\[ \forall \ 1 \leq i \leq m \quad G(LocBtn_i \rightarrow F Loc_i \land ReqLoc_i \land W(Loc_i \land ReqLoc_i)) , \]

//If the lift reaches the requested floor, the door open.
\[ \forall \ 1 \leq i \leq m \quad G(Loc_i \land ReqLoc_i \rightarrow Open \land Loc_iW Movable) , \]

(an) non-functional requirements

//Until request button is pushed, lift is not requested to go there.
\[ \forall \ 1 \leq i \leq m \quad G(Loc_i \land Movable \rightarrow (\neg ReqLoc_i)W LocBtn_i) , \]

//If the lift is not requested at a floor, the door will not open there.
\[ \forall \ 1 \leq i \leq m \quad G(Loc_i \land \neg ReqLoc_i \rightarrow \neg Open) , \]
(b) a specification for a door of the lift

```plaintext
// The time-limit that the door open is set.
G(Open → F OpenTimedOut),

// If open button is pushed, the door is requested to open.
G(OpenBtn ∧ ¬OpenTimedOut → ReqOpen),

// When the time that door can open passed, the door closes.
G(OpenTimedOut → ¬Open),

// If close button is pushed, the door closes.
G(CloseBtn ∧ ¬ReqOpen → ¬Open),

// If the door is requested to open and the lift is not movable, the door opens.
G(ReqOpen ∧ ¬Movable → Open)
```
(c) a specification of the physical constraints

//The lift is located at some floor.
\[ G( \bigvee_{1 \leq i \leq m} \text{Loc}_i) \land G( \bigwedge_{1 \leq i \leq m} (\text{Loc}_i \rightarrow \bigwedge_{j=1..m, i \neq j} \neg \text{Loc}_j)), \]

//The lift must pass floors on the way to the destination. \((m \geq 3)\)
\[ G( \bigwedge_{1 \leq i \leq m-2} (\text{Loc}_i \land \text{ReqLoc}_j \rightarrow \bigwedge_{i+2 \leq k \leq j} (\neg \text{Loc}_k) \cup (\neg \text{Loc}_k \land \text{Loc}_k-1))), \]
\[ 1 \leq i \leq m-2 \]
\[ 3 \leq j \leq m \]
\[ i < j-2 \]

\[ G( \bigwedge_{1 \leq i \leq m-2} (\text{Loc}_j \land \text{ReqLoc}_i \rightarrow \bigwedge_{i+2 \leq k \leq j} (\neg \text{Loc}_k) \cup (\neg \text{Loc}_k \land \text{Loc}_k+1))), \]
\[ 1 \leq i \leq m-2 \]
\[ 3 \leq j \leq m \]
\[ i < j-2 \]

//A relation between the door open/close and the lift movable/unmovable
\[ G(\text{Open} \rightarrow (\neg \text{Movable}) \cup \neg \text{Open}), \]
\[ G(\neg \text{Open} \rightarrow \text{Movable} \cup \text{Open}), \]
Environmental constraint and payoff term

An environmental constraint

//If the floor button of each floor is pushed, then the button is released at the next time unit.
\[ \land_{1 \leq i \leq m} G (LocBtn_i \rightarrow X \neg LocBtn_i) \]

We obtain a payoff term from the four requirements: two functional requirements of (a), (b) and (c), by using each payoff \( c_i = 1 \). In the payoff term we approximate the Until operator with the bounded Until operator (the bound is 2).
Results

For the case where $m = 2$, we successfully synthesized a reactive system in a reasonable period of time.
From these results, we confirmed that with our method we can synthesize a reactive system with environmental tolerance from the more practical specification of a non-trivial scale, although the size of the resulting reactive system is non-trivial and the computation time is larger than that for synthesis without the consideration of environmental tolerance.
Related works

- Tools for synthesizing reactive systems from specifications written in LTL
  - Lily [JB06], AcaciaPlus[BBF+12, BBFR13], and Unbeast[Ehl10].
- Methods for synthesizing reactive systems with environmental tolerance [BCG+14] and [Ehl11].
- A method for synthesizing optimal reactive systems from quantitative specifications [BCHJ09].
- A study on practical reactive system synthesis from LTL specifications considering unpredictable environmental behavior [DBSU15].
- A detailed survey of synthesis from specifications with environmental assumptions [BEJK14]
Conclusion

- We describe a simple quantitative definition for environmental tolerance.
- We also proposed a method for synthesizing a reactive system with environmental tolerance.
- The method was implemented, and reactive systems synthesized using our method compared to reactive systems synthesized without the consideration of environmental tolerance.
  - We confirmed that, for the unexpected behavior of an environment, the behavior of the reactive system synthesized by our method satisfied the system constraints more often than the behavior of the reactive system synthesized without the consideration of environmental tolerance.
Aaron Bohy, Véronique Bruyère, Emmanuel Filiot, Naiyong Jin, and Jean-François Raskin. 
Acacia+, a tool for LTL synthesis. 

Aaron Bohy, Véronique Bruyère, Emmanuel Filiot, and Jean-François Raskin. 
Synthesis from LTL specifications with mean-payoff objectives. 

Synthesizing robust systems. 
Roderick Bloem, Krishnendu Chatterjee, Thomas A. Henzinger, and Barbara Jobstmann.
Better quality in synthesis through quantitative objectives.

Roderick Bloem, Rüdiger Ehlers, Swen Jacobs, and Robert Könighofer.
How to handle assumptions in synthesis.

Tomás Babiak, Mojmír Kretínský, Vojtech Rehák, and Jan Strejček.
LTL to Büchi automata translation: Fast and more deterministic.
Nicolas D’Ippolito, Victor Braberman, Daniel Sykes, and Sebastian Uchitel.
Robust degradation and enhancement of robot mission behaviour in unpredictable environments.

Rüdiger Ehlers.
Symbolic bounded synthesis.

Rüdiger Ehlers.
Generalized Rabin(1) synthesis with applications to robust system synthesis.
Emmanuel Filiot, Naiyong Jin, and Jean-François Raskin.
An antichain algorithm for LTL realizability.

Barbara Jobstmann and Roderick Bloem.
Optimizations for LTL synthesis.

Orna Kupferman and Moshe Y. Vardi.
Safraless decision procedures.

Amir Pnueli and Roni Rosner.
On the synthesis of a reactive module.
Roni Rosner.  
*Modular Synthesis of Reactive Systems.*  

Sven Schewe and Bernd Finkbeiner.  
Bounded synthesis.  

Takashi Tomita, Shin Hiura, Shigeki Hagihara, and Naoki Yonezaki.  
A temporal logic with mean-payoff constraints.  

Uri Zwick and Mike Paterson.  
The complexity of mean payoff games on graphs.