Mimicking Behaviors in Separated Domains

Giuseppe De Giacomo[1] Dror Fried[2] Fabio Patrizi[1] Shufang Zhu[1]

[1]Sapienza University of Rome
[2]The Open University of Israel

GenPlan 2022
Jul 23, 2022

ERC Advanced Grant
WhiteMech:
White-box Self Programming Mechanisms

Behavior Mimicking

Example

Kid mimicking parent making recipe (unknown to kid):

- Parent acts in real kitchen: real eggs, pans, fire, water,...
- Kid in toy kitchen: plastic eggs, toy pots only (no toy pans), no fire, no water,...
- Kid's goal: make recipe in toy kitchen
- Kid maps real kitchen states/actions to toy kitchen:
  - egg in real pan $\Rightarrow$ toy egg in pot
  - pan on stove $\Rightarrow$ pot on toy stove

Behavior Mimicking

Main features:

- Two agents: A (parent), B (kid)
- A operates in $\mathcal{D}_A$ (real kitchen)
- B operates in $\mathcal{D}_B$ (toy kitchen)
- $\mathcal{D}_A$ and $\mathcal{D}_B$ *separated*:
  - Actions in $\mathcal{D}_A$ do not affect $\mathcal{D}_B$, and viceversa
- *Mapping* φ between behaviors of A and B (on $\mathcal{D}_A$ and $\mathcal{D}_B$):
  - egg in real pan $\Rightarrow$ toy egg in pot
  - pan on stove $\Rightarrow$ pot on toy stove
- Goal:
  - find a strategy for B to mimic A
  - *Mimicking* defined by φ

Dynamic Domains

Dynamic Domain over propositions *Prop*

- Domain $\mathcal{D} = (S, s_0, \delta, \lambda)$:
  - S finite set of states
  - $s_0 \in S$ initial state
  - $\delta \subseteq S \times S$ transition relation
  - $\lambda : S \mapsto 2^{Prop}$ state-labeling function
- Finite/infinite *traces* as standard: $\tau = s_0 s_1 \cdots s_\ell$ (possibly $\ell = \infty$)

- Actions correspond to selecting next transition
- In fact, deterministic

Problem Instance

Mimicking Behaviors in Separated Domains (MBSD) Problem Instance

$\mathcal{P} = (\mathcal{D}_A, \mathcal{D}_B, \Phi, Ag_{stop})$, where:

- $\mathcal{D}_A = (S, s_0, \delta^A, \lambda^A)$, dynamic domain over $Prop^A$
- $\mathcal{D}_B = (T, t_0, \delta^B, \lambda^B)$, dynamic domain over $Prop^B$
- $Prop^A \cap Prop^B = \varnothing$
- Φ, mapping specification: LTL_f formula over $Prop^A \cup Prop^B$
- $Ag_{stop} \in \{A, B\}$, designated *stop agent*

Mappings

Intuition: Φ expresses properties of *joint traces* of $\mathcal{D}_A$ and $\mathcal{D}_B$

$$\tau_A = s_0 \; s_1 \cdots s_\ell \qquad \tau_B = t_0 \; t_1 \cdots t_\ell \qquad \tau_A \cup \tau_B = \begin{pmatrix} s_0 \\ t_0 \end{pmatrix} \begin{pmatrix} s_1 \\ t_1 \end{pmatrix} \cdots \begin{pmatrix} s_\ell \\ t_\ell \end{pmatrix}$$

- Φ: Linear-time temporal formulae over finite traces (LTL_f) over $Prop^A$ and $Prop^B$
- Combines:
  - boolean operators, temporal operators: *next* ($\mathbf{X}$), *until* ($\mathbf{U}$), always ($\Box$), eventually ($\Diamond$), ...
- can express:
  - $\Box\phi$ (always ϕ), $\Diamond\phi$ (eventually ϕ), $\Box\phi \rightarrow \Diamond\psi$ (whenever ϕ eventually ψ), $\phi\mathbf{U}\psi$ (ϕ until ψ), ...

Examples:

- $\Phi = \Box(egg\_in\_real\_pan \rightarrow egg\_in\_toy\_pot) \wedge \Box(pan\_on\_stove \rightarrow pot\_on\_toy\_stove)$
- In general, arbitrarily complex mappings: $\Box(a \rightarrow \Diamond(b\mathbf{U}c)) \wedge \Box\Diamond q$

Designated Stop Agent

Designated *Stop Agent* decides when to stop

Examples:

- B (parent) announces when recipe is completed
- A (kid) decides when to leave

Choice of Stop Agent strongly affects solution:

- if B (kid) leaves right after starting the game, it trivially mimics B (parent)
- (In kitchen example, parent stops)

Strategies

Strategy for B:

- function $\sigma : S^+ \to T$
- given sequence of $\mathcal{D}_A$ states, returns move for B (i.e., next $\mathcal{D}_B$ state)

In general, depends on history

As A operates in $\mathcal{D}_A$ and B acts according to a strategy σ, a joint trace is *induced*:

$$\tau_{A,\sigma} = \begin{pmatrix} s_0 \\ \sigma(s_0) \end{pmatrix} \begin{pmatrix} s_1 \\ \sigma(s_0\ s_1) \end{pmatrix} \cdots \begin{pmatrix} s_\ell \\ \sigma(s_0\ s_1 \cdots s_\ell) \end{pmatrix}$$

In general, many joint traces exist:

- Result of all choices available to A and consequent B's responses

Solution

Definition (MBSD Solution)

Solution to MBSD problem instance $\mathcal{P} = (\mathcal{D}_A, \mathcal{D}_B, \Phi, Ag_{stop})$:

- *Executable* strategy σ s.t.:
  - $Ag_{stop} = A$ and for every finite trace τ_A of $\mathcal{D}_A$, $\tau_{A,\sigma} \models \Phi$; or
  - $Ag_{stop} = B$ and for every infinite trace τ_A of $\mathcal{D}_A$ t.e. finite prefix τ'_A s.t. $\tau'_{A,\sigma} \models \Phi$

Intuition:

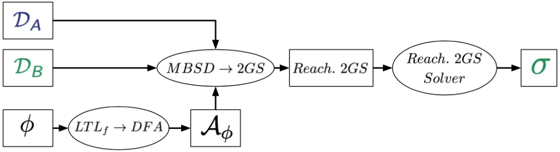
- A Stop Agent: B has a strategy to always keep Φ enforced, no matter how A acts
- B Stop Agent: B has a strategy to enforce Φ at least once, no matter how A acts

Essentially:

- *Synthesis* where environment and system do not affect each other

Solution Approach

- Search for a strategy in a 2-Player game
- Based on constructing DFA $\mathcal{A}_\phi$ for mapping ϕ (2EXPTIME [De Giacomo&Vardi, 2013])
- DFA $\mathcal{A}_\phi$ embedded in 2-Player Reachability Game



Theorem

MBSD with general mappings is in:

- *2EXPTIME in combined complexity and mapping complexity*
- *PTIME in domain complexity*

Mapping Classes

Form of mapping affects solution complexity

Two classes of mappings investigated:

- Point-wise Mappings: $\Phi = \bigwedge_{i=1}^{k} \Box(\varphi_i \rightarrow \psi_i)$
  - $\Phi = \Box(egg\_in\_real\_pan \rightarrow egg\_in\_toy\_pot) \wedge \Box(pan\_on\_stove \rightarrow pot\_on\_toy\_stove)$
- Target Mappings: $\Phi = \bigwedge_{i=1}^{k} (\Diamond\varphi_i) \rightarrow (\Diamond\psi_i)$
  - $\Phi = \Diamond salt\_added \rightarrow \Diamond talco\_added$
- Recall: properties over $\mathcal{D}_A$ and $\mathcal{D}_B$ are separated:
  - φ_i over $Prop^A$, ψ_i over $Prop^B$
  - $Prop^A$ and $Prop^B$ disjoint

Results

Theorem

MBSD with Point-wise mappings is in PTIME for:

- *domain complexity*
- *mapping complexity*
- *combined complexity*

Theorem

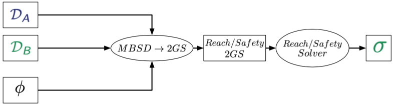
MBSD with Target mappings is

- *PTIME in: domain complexity*
- *PSPACE in:*
  - *mapping complexity*
  - *combined complexity*
- *PSPACE-hard (even with DAG-like* $\mathcal{D}_A$ *and* $\mathcal{D}_B$*)*

Intuition

Each mapping class leads to a different game:

- Point-wise Mappings:
  - Safety Game
  - B's objective: maintain (continuously) the game in a region where A can be mimicked
  - Game Structure polynomial in size of domains and mapping
- Target Mappings:
  - Reachability Game
  - B's objective: reach a state where A is (eventually) mimicked
  - Game Structure polynomial in size of domains, exponential in # of conjuncts in mapping
- Knowing form of mapping saves constructing DFA for ϕ (2EXPTIME)
- Reachability and safety games solvable in PTIME wrt state space of game



Conclusions

Contributions:

- Proposed and formalized MBSD
- General solution approach (2EXPTIME)
- Identified classes of mappings with better computational behavior:
  - Point-wise mappings (PTIME)
  - Target mappings (PSPACE-hard, PTIME wrt domains)
    - Also Tree-like domains (PTIME, not covered in talk)

Open point:

- To what extent separation can yield computational improvements in general, e.g:
- $\Phi = \bigwedge_{i=1}^{k} \Box(\varphi_i \rightarrow \psi_i)$, with: φ_i LTL_f over $Prop^A$, ψ_i LTL_f over $Prop^B$
- Conjunction of Point-wise and Target-mappings

Thank you!

Questions?