To Assume, Or Not To Assume

[Computing Adequately Permissive Assumptions for Synthesis]

May 16, 2023

Ashwani Anand
MPI-SWS

with K. Mallik (ISTA), S.P. Nayak (MPI-SWS), and A.-K. Schmuck (MPI-SWS)
No cleaning strategy may exist if Environment is adversarial.
No cleaning strategy may exist if Environment is adversarial.

Environment rarely acts adversarially.
No cleaning strategy may exist if Environment is adversarial. Humans may know this assumption. Computers don’t.

Environment rarely acts adversarially.
Reactive Synthesis

Task for a system

Clean a house

Implementation

Roomba software
Reactive Synthesis

Task for a system

Clean a house

Implementation

Roomba software

Works well if no assumption is needed.
Reactive Synthesis

Task for a system

Clean a house

Implementation

Roomba software

Works well if no assumption is needed.

Might fail without some assumptions. E.g. the user will not block the path.
What has been done?

Chatterjee et. al. [CONCUR'08] introduce the notion of assumption for games on graphs.
What has been done?

Chatterjee et. al. [CONCUR'08] introduce the notion of assumption for games on graphs.

Their method requires solving NP-hard problem.
What has been done?

Chatterjee et. al. [CONCUR'08] introduce the notion of assumption for games on graphs.

Their method requires solving NP-hard problem.

Fails to give a sufficient assumption, even if it exists.
What has been done?

Chatterjee et. al. [CONCUR'08] introduce the notion of assumption for games on graphs.

Their method requires solving NP-hard problem.

Fails to give a sufficient assumption, even if it exists.
Reactive Synthesis: The Standard
Reactive Synthesis: The Standard

Convert to a game on graph
Reactive Synthesis: The Standard

Convert to a game on graph

Winning strategy acts as the software.
Reactive Synthesis: The Standard

Convert to a game on graph

Winning strategy acts as the software.

Find assumptions on the environment via the game graph.
The Return of Reactive Synthesis

Convert to a game on graph
The Return of Reactive Synthesis

Convert to a game on graph

Winning strategy under assumption acts as software

Compute assumptions

I will never block Roomba
The Return of Reactive Synthesis

Do not block me!

Suggest user

Winning strategy acts as software

Convert to a game on graph

Compute assumptions

I will never block Roomba
The Return of Reactive Synthesis

Do not block me!

Winning strategy under assumption acts as software

Suggest user

I will never block Roomba

Convert to a game on graph

Compute assumptions
Precap

Assumptions computation
Precap

Assumptions computation

Permissive
Precap

Assumptions computation

Novel Templates

Permissive
Complete
Prepap

Novel Templates

Assumptions computation

Permissive

Complete

Faster
Precap

Novel Templates

Assumptions computation

Permissive
Complete
Faster
Games on Graphs
Games on Graphs
Games on Graphs
Games on Graphs

Always eventually visit h
Games on Graphs

Always eventually visit h
Games on Graphs

Always eventually visit h

\[ g \rightarrow i \]
Games on Graphs

Always eventually visit h

g → i → h
Games on Graphs

Always eventually visit h

g → i → h → h
Games on Graphs

Always eventually visit \( h \)

\[
g \rightarrow i \rightarrow h \rightarrow h \rightarrow h
\]
Games on Graphs

Always eventually visit h

\[ g \rightarrow i \rightarrow h \rightarrow h \rightarrow h \rightarrow \ldots \]
Always eventually visit h

\[ g \rightarrow i \rightarrow h \rightarrow h \rightarrow h \rightarrow \ldots \]
Games on Graphs

Always eventually visit h

\[ g \rightarrow i \rightarrow h \rightarrow h \rightarrow h \rightarrow \ldots \]

Assumptions restrict the choices of Environment
Assumptions on Environment

LTL formula $\psi$ on vertices of the game graph
Assumptions on Environment

LTL formula \( \psi \) on vertices of the game graph

Sufficient
If environment satisfies assumption,
  system can finish the task
Assumptions on Environment

LTL formula \( \psi \) on vertices of the game graph

Sufficient
If environment satisfies assumption, system can finish the task
Assumptions on Environment

LTL formula $\psi$ on vertices of the game graph

Sufficient
If environment satisfies assumption, system can finish the task

Implementable
Environment can satisfy the assumption
Assumptions on Environment

LTL formula $\psi$ on vertices of the game graph

Sufficient
If environment satisfies assumption,
system can finish the task

Implementable
Environment can satisfy the assumption

Leave the room
Assumptions on Environment

LTL formula $\psi$ on vertices of the game graph

Sufficient
If environment satisfies assumption, system can finish the task

Implementable
Environment can satisfy the assumption

Permissive
Assumption does not restrict the environment too much
Assumptions on Environment

LTL formula \( \psi \) on vertices of the game graph

Sufficient
   If environment satisfies assumption, system can finish the task

Implementable
   Environment can satisfy the assumption

Permissive
   Assumption does not restrict the environment too much
Assumptions on Environment

LTL formula \( \phi \) on vertices of the game graph

Sufficient
If environment satisfies assumption, system can finish the task

Implementable
Environment can satisfy the assumption

Permissive
Assumption does not restrict the environment too much
Checkpoint

Assumptions computation

Permissive
Complete
Faster
Checkpoint

Novel Templates → Assumptions computation

- Permissive
- Complete
- Faster
Büchi objective: Always eventually visit
Büchi objective: Always eventually visit $\square$. No way of satisfying the objective from $c$. Hence, it must never be visited.
Büchi objective: Always eventually visit $\square$

- Always not $b \rightarrow c$.

No way of satisfying the objective from $c$. Hence, it must never be visited.
Computing Adequately Permissive Assumptions

Büchi objective: Always eventually visit

- Always not $b \rightarrow c$. 
Computing Adequately Permissive Assumptions

Büchi objective: Always eventually visit $\square$

- Always not $b \rightarrow c$.

Enough to compute assumptions to "reach" $h$ from remaining vertices.
Büchi objective: Always eventually visit $\square$

- Always not $b \rightarrow c$. 
Computing Adequately Permissive Assumptions

Büchi objective: Always eventually visit

- Always not $b \rightarrow c$. 
Computing Adequately Permissive Assumptions

Büchi objective: Always eventually visit

- Always not $b \rightarrow c$. 
Computing Adequately Permissive Assumptions

Büchi objective: Always eventually visit

- Always not $b \rightarrow c$.
- Always eventually $\{f, g\} \Rightarrow$ always eventually $f \rightarrow e$ or $g \rightarrow i$
Computing Adequately Permissive Assumptions

Büchi objective: Always eventually visit □

- Always not b → c.
- Always eventually \( f,g \) ⇒ always eventually \( f \rightarrow e \) or \( g \rightarrow i \)
Computing Adequately Permissive Assumptions

Büchi objective: Always eventually visit

- Always not $b \rightarrow c$.
- Always eventually $\exists f, g \Rightarrow$ always eventually $f \rightarrow e$ or $g \rightarrow i$
- Always eventually $\exists b \exists \Rightarrow$ always eventually $b \rightarrow f$
Computing Adequately Permissive Assumptions

Büchi objective: Always eventually visit

- Always not $b \rightarrow c$.
- Always eventually $\exists f, g \Rightarrow$ always eventually $f \rightarrow e$ or $g \rightarrow i$.
- Always eventually $\exists b \Rightarrow$ always eventually $b \rightarrow f$. 
Computing Adequately Permissive Assumptions

Büchi objective: Always eventually visit □

Safety template

- Always not b → c.
- Always eventually \( \exists f, g \) ⇒ always eventually \( f \rightarrow e \) or \( g \rightarrow i \)
- Always eventually \( \exists b \) ⇒ always eventually b → f

Group liveness templates
Computing Adequately Permissive Assumptions

Büchi objective: Always eventually visit □

Runs in time \( O(m+n) \)

Safety template

- Always not \( b \to c \).
- Always eventually \( \exists f,g \Rightarrow \text{always eventually } f \to e \text{ or } g \to i \)
- Always eventually \( \exists b \equiv \Rightarrow \text{always eventually } b \to f \)

Group liveness templates
coBüchi objective: Eventually always visit
Computing Adequately Permissive Assumptions

cobuchi objective: Eventually always visit

- Always not $b \rightarrow c$. 
Computing Adequately Permissive Assumptions

cobüchi objective: Eventually always visit

- Always not b→c.

Need to restrict from going further away from h eventually.
Computing Adequately Permissive Assumptions

cөBüchi objective: Eventually always visit $\square$

Need to restrict from going further away from $h$ eventually.

- Always not $b \to c$. 
Computing Adequately Permissive Assumptions

cOBüchi objective: Eventually always visit

Need to restrict from going further away from h eventually.

- Always not b→c.
- Eventually always not h→f
CoBüchi objective: Eventually always visit $\square$

Need to restrict from going further away from $h$ eventually.

- Always not $b \rightarrow c$.
- Eventually always not $h \rightarrow f$.
Computing Adequately Permissive Assumptions

cOBüchi objective: Eventually always visit

- Always not $b \rightarrow c$.
- Eventually always not $h \rightarrow f$

Need to restrict from going further away from $h$ eventually.
coBüchi objective: Eventually always visit $\square$

Need to restrict from going further away from $\blacksquare$ eventually.

- Always not $b \rightarrow c$.
- Eventually always not $h \rightarrow f$
- Eventually always not $f \rightarrow g$
- Eventually always not $g \rightarrow f$
Computing Adequately Permissive Assumptions

cOBüchi objective: Eventually always visit

- Always not $b \rightarrow c$
- Eventually always not $h \rightarrow f$
- Eventually always not $f \rightarrow g$
- Eventually always not $g \rightarrow f$

Need to restrict from going further away from $h$ eventually.
Computing Adequately Permissive Assumptions

cOBüchi objective: Eventually always visit $\square$

- Always not $b \rightarrow c$.
- Eventually always not $h \rightarrow f$.
- Eventually always not $f \rightarrow g$.
- Eventually always not $g \rightarrow f$.
- Eventually always not $b \rightarrow a$.

Need to restrict from going further away from $h$ eventually.
Computing Adequately Permissive Assumptions

cobuchi objective: Eventually always visit $\square$

- Always not $b \rightarrow c$.
- Eventually always not $h \rightarrow f$
- Eventually always not $f \rightarrow g$
- Eventually always not $g \rightarrow f$
- Eventually always not $b \rightarrow a$

Need to restrict from going further away from $h$ eventually.
Computing Adequately Permissive Assumptions

coBüchi objective: Eventually always visit

Need to restrict from going further away from \( h \) eventually.

- Always not \( b \to c \).
- Eventually always not \( h \to f \).
- Eventually always not \( f \to g \).
- Eventually always not \( g \to f \).
- Eventually always not \( b \to a \).
Computing Adequately Permissive Assumptions

cOBüchi objective: Eventually always visit

Need to restrict from going further away from h eventually.

Runs in time $O(m+n)$

- Always not $b \rightarrow c$.
- Eventually always not $h \rightarrow f$
- Eventually always not $f \rightarrow g$
- Eventually always not $g \rightarrow f$
- Eventually always not $b \rightarrow a$
Computing Adequately Permissive Assumptions

Parity objective: highest priority visited infinitely is even
Computing Adequately Permissive Assumptions

Parity objective: highest priority visited infinitely is even

Needs conditional group liveness templates

$C_3 \Rightarrow \text{live group } (C_4 \cup C_6 \ldots)$
Computing Adequately Permissive Assumptions

Parity objective: highest priority visited infinitely is even

Runs in time $O(n^4)$

Needs conditional group liveness templates

$\square \Diamond C_3 \Rightarrow \text{live group } (C_4 \cup C_6 \ldots)$
Checkpoint

Novel Templates

Assumptions computation

Permissive
Complete
Faster
Experiments

C++ Tool
SImPA
Experiments

C++ Tool
SIMPA

230 SYNTCOMP benchmarks
Experiments

*Chatterjee et al., CAV’10

C++ Tool
SImPA

230 SYNTCOMP benchmarks
Experiments

C++ Tool
SImPA

230 SYNTCOMP benchmarks

<table>
<thead>
<tr>
<th></th>
<th>SImPA</th>
<th>GIST*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean-time</td>
<td>64.8 s</td>
<td>1079.0 s</td>
</tr>
<tr>
<td>Non-timeout mean-time</td>
<td>64.8 s</td>
<td>209.2 s</td>
</tr>
<tr>
<td>Termination(1hr)</td>
<td>100%</td>
<td>74%</td>
</tr>
<tr>
<td>Generated assumptions</td>
<td>100%</td>
<td>91%</td>
</tr>
<tr>
<td>Fastest</td>
<td>100%</td>
<td>0%</td>
</tr>
</tbody>
</table>

* Chatterjee et. al., CAV’10
Experiments

<table>
<thead>
<tr>
<th></th>
<th>SimPA</th>
<th>GIST*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean-time</td>
<td>64.8s</td>
<td>1079.0s</td>
</tr>
<tr>
<td>Non-timeout</td>
<td>64.8s</td>
<td>209.2s</td>
</tr>
<tr>
<td>mean-time</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Termination(1hr)</td>
<td>100%</td>
<td>74%</td>
</tr>
<tr>
<td>Generated</td>
<td>100%</td>
<td>91%</td>
</tr>
<tr>
<td>assumptions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fastest</td>
<td>100%</td>
<td>0%</td>
</tr>
</tbody>
</table>

* Chatterjee et al., CAV’10

C++ Tool
SimPA

230 SYNTCOMP benchmarks

Always gives an assumption
Experiments

<table>
<thead>
<tr>
<th></th>
<th>SImPA</th>
<th>GIST*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean-time</td>
<td>64.8s</td>
<td>1079.0s</td>
</tr>
<tr>
<td>Non-timeout mean-time</td>
<td>64.8s</td>
<td>209.2s</td>
</tr>
<tr>
<td>Termination(1hr)</td>
<td>100%</td>
<td>74%</td>
</tr>
<tr>
<td>Generated assumptions</td>
<td>100%</td>
<td>91%</td>
</tr>
<tr>
<td>Fastest</td>
<td>100%</td>
<td>0%</td>
</tr>
</tbody>
</table>

* Chatterjee et. al., CAV’10

C++ Tool
SImPA

230 SYNTCOMP benchmarks

Always gives an assumption

Orders of magnitude faster
Summary

Assumptions computation

Novel Templates

Permissive
Complete
Faster
Furthermore...

Novel Templates

Assumptions on Environment \cite{TACAS'23}

Negotiation between agents \cite{Best Poster, HSCC'23}*

Composition of specifications \cite{CAV'23}*

* Joint work with S. Nayak and A. K. Schmuck.
Furthermore...

Novel Templates

Assumptions on Environment [TACAS '23]

Negotiation between agents [Best Poster, HSCC '23]*

Composition of specifications [CAV '23]*

*Joint work with S. Nayak and A. K. Schmuck.
May keep blocking each other
Recall that...
Recall that...

unsafe edges
Recall that...

Unsafe edges
Co-live edges
Recall that...

Unsafe edges
Co-live edges
[Group /conditional]
Live edges
Recall that...

Unsafe edges
Co-live edges
[Group / Conditional]
Live edges
Unconstrained edges
Recall that...

Unsafe edges
Co-live edges
[Group /conditional]
Live edges
Unconstrained edges

Only on Environment’s vertices
Recall that...

Unsafe edges
Co-live edges
[Group / Conditional]
Live edges
Unconstrained edges

Only on Environment’s vertices

Can be shared with other agent as added specification
Assumptions to the Rescue
Assumptions to the Rescue

Please satisfy assumption $\Psi_c$

Please satisfy assumption $\Psi_R$
Assumptions to the Rescue

Please satisfy assumption $\psi_c$

Solves $\phi_R \land \psi_R$

Please satisfy assumption $\psi_R$

Solves $\phi_c \land \psi_R$
Assumptions to the Rescue

\[ \Phi_R \land \Psi_R \]

Solves \( \Phi_R \land \Psi_R \)

May not terminate!

\[ \Phi_C \land \Psi_R \]

Solves \( \Phi_C \land \Psi_R \)
Assumptions to the Rescue

Please satisfy assumption $\Psi_{c}$

Solves $\Phi_{R} \land \Psi_{R}$

May not terminate!
Requires solving games in every iteration!
Strategy Templates

- Represent strategies of each agent by templates.
Strategy Templates

- Represent strategies of each agent by templates.
- Can be computed during assumptions computation.
Strategy Templates

- Represent strategies of each agent by templates.
- Can be computed during assumptions computation.
Strategy Templates

- Represent strategies of each agent by templates.
- Can be computed during assumptions computation.
Strategy Templates

- Represent strategies of each agent by templates.
- Can be computed during assumptions computation.

• Always eventually \{i\} \Rightarrow always eventually \ i \rightarrow b
Strategy Templates

- Represent strategies of each agent by templates.
- Can be computed during assumptions computation.

- Always eventually $\{i?\} \Rightarrow$ always eventually $i \rightarrow b$
Strategy Templates

- Represent strategies of each agent by templates.
- Can be computed during assumptions computation.

• Always eventually \( \{i\} \Rightarrow \) always eventually \( i \rightarrow b \)
Strategy Templates

- Represent strategies of each agent by templates.
- Can be computed during assumptions computation.

• Always eventually \{i?\} ⇒ always eventually \(i \rightarrow b\)
• Always eventually \{d?\} ⇒ always eventually \(d \rightarrow g\)
Strategy Templates

- Represent strategies of each agent by templates.
- Can be computed during assumptions computation.

- Always eventually \( \{ i \} \Rightarrow \text{always eventually } i \rightarrow b \)
- Always eventually \( \{ d \} \Rightarrow \text{always eventually } d \rightarrow g \)
Strategy Templates

- Represent strategies of each agent by templates.
- Can be computed during assumptions computation.

• Always eventually {i?} ⇒ always eventually i→b
• Always eventually {d?} ⇒ always eventually d→g
Strategy Templates

- Represent strategies of each agent by templates.
- Can be computed during assumptions computation.

- Always eventually $\{i?\} \Rightarrow$ always eventually $i \rightarrow b$
- Always eventually $\{d?\} \Rightarrow$ always eventually $d \rightarrow g$
- Always eventually $\{a?\} \Rightarrow$ always eventually $a \rightarrow d$
Strategy Templates

- Represent strategies of each agent by templates.
- Can be computed during assumptions computation.

- Always eventually $\{i?\} \Rightarrow$ always eventually $i \rightarrow b$
- Always eventually $\{d?\} \Rightarrow$ always eventually $d \rightarrow g$
- Always eventually $\{a?\} \Rightarrow$ always eventually $a \rightarrow d$
**Strategy Templates**

- Represent strategies of each agent by templates.
- Can be computed during assumptions computation.

- Always eventually \( \{i\} \Rightarrow \) always eventually \( i \rightarrow b \)
- Always eventually \( \{d\} \Rightarrow \) always eventually \( d \rightarrow g \)
- Always eventually \( \{a\} \Rightarrow \) always eventually \( a \rightarrow d \)
Negotiation
Negotiation

Assumption

Strategy

Assumption

Strategy
Negotiation

Strategy

Assumption

Strategy

Assumption
Negotiation

Strategy + Assumption

Assumption + Strategy

Assumption
Negotiation

Strategy + Assumption → Assumption + Strategy
Negotiation

Strategy + Assumption → Assumption + Strategy
Negotiation

Strategy + Assumption → Assumption + Strategy

Robot serving coffee
Negotiation

Strategy + Assumption → Assumption + Strategy
Negotiation

Strategy + Assumption → Assumption + Strategy

No direct conversion between Strategy and Assumption.

Red cross marks the invalid conversion.
Negotiation

Strategy + Assumption

Assumption + Strategy

Recomputation needed
Summary

Templates for negotiation
Summary

- Templates for negotiation
- Terminates after few rounds
Summary

- Templates for negotiation
- Terminates after few rounds
- Sound
Summary

- Templates for negotiation
- Terminates after few rounds
- Sound
- Complete
Summary

- Templates for negotiation
- Terminates after few rounds
- Sound
- Complete
- Fault tolerant
Furthermore...

Novel Templates

- Assumptions on Environment [TACAS'23]
- Negotiation between agents [Best Poster, HSCC'23]*
- Composition of specifications [CAV'23]*

* Joint work with S. Nayak and A. K. Schmuck.
Furthermore...

- Novel Templates
- Assumptions on Environment [TACAS'23]
- Negotiation between agents [Best Poster, HSCC'23]
- Composition of specifications [CAV'23]

* Joint work with S. Nayak and A.K. Schmuck.
Put my cup in dishwasher
Composition of Specifications
Composition of Specifications

Bring coffee
Composition of Specifications

Bring coffee + Cup in dishwasher
Composition of Specifications

Bring coffee + Cup in dishwasher =
Composition of Specifications

- Bring coffee
- Cup in dishwasher

- Might need to resolve conflicts
Composition of Specifications

- Might need to resolve conflicts
- Not complete under adversarial environment
Composition of Specifications

- Might need to resolve conflicts
- Not complete under adversarial environment
- Faster than existing approaches
Composition of Specifications

C++ Tool
PeSTel

~1400 SYNTCOMP benchmarks
Composition of Specifications

- C++ Tool: PeSTel
- ~1400 SYNTCOMP benchmarks
Composition of Specifications

C++ Tool
PeSTel

~1400 SYNTCOMP benchmarks

Complete on all instances
Composition of Specifications

- **C++ Tool PeSTel**
- ~1400 SYNTCOMP benchmarks
- Complete on all instances
- Faster as specifications arrive
Summary

- Novel Templates
- Assumptions on Environment [TACAS'23]
- Negotiation between agents [Best Poster, HSCC'23]
- Composition of specifications [CAV'23]

* Joint work with S. Nayak and A. K. Schmuck.