Synthesize the ABP (Alternating Bit Protocol) using Transit

Stavros Tripakis (UC Berkeley)
Joint work with Antti Halme (Aalto University), Abhishek Udupa, Rajeev Alur, Milo Martin (UPenn)

NSF ExCAPE PI meeting – June 2013
ABP: reliable transmission over an unreliable channel

Channels are lossy but FIFO.
Challenge problem: 
synthesize the ABP automatically!

Why:
- Warm up
- Gauge state of art
- Education
- Instance of generic problem

Channels are lossy but FIFO.
Can be formalized as a decentralized controller synthesis problem

Plant

Controller 1

Controller 2

(locally) observable events

(locally) controllable events
Can be formalized as a decentralized controller synthesis problem.
Challenges (?)

- Decentralized control (and observation) problems are **undecidable** in general [Pnueli-Rosner’90, Lamouchi-Thistle’00, Tripakis’01]
  - Note: all inputs are finite-state
- Getting the **specification** right => getting the right controllers
- Representation of controllers
- Understanding the result
- ...
- Applications to ExCAPE challenge domains:
  - Multicore protocols
  - Robotics
  - Concurrent programs
  - Network protocols
Transit [Alur, Martin et al]
Transit [Alur, Martin et al]

- Synthesis from concolic snippets

- Developed originally for cache coherence protocols

**Our goal:** use it to synthesize the ABP
ABP synthesis from snippets

Listing 1: A pair of exhaustive, trivial snippets

```plaintext
[[] \Rightarrow (SEND, DataNet OutMsg) {
    (MyBit) \Rightarrow {
        OutMsg. Val;
    }
};
}
```

```plaintext
[[] \Rightarrow (SEND, DataNet OutMsg) {
    (!MyBit) \Rightarrow {
        !OutMsg. Val;
    }
};
}
```

Listing 2: Generalized form synthesized by Transit

```plaintext
[(MyBit | (! (MyBit)))] \Rightarrow (SEND, DataNet OutMsg) {
    OutMsg. Val := MyBit;
}
```
ABP synthesis from snippets

• Results:
  – “Successfully” synthesized ABP using Transit.
  – Somewhat cheated:
    • State-space (alternating bit) given as input.
    • Exhaustive set of snippets (cover all cases).
  – Still:
    • Transit managed to synthesize symbolic guards from (small # of) concrete snippets.

• Challenges:
  – Liveness properties (not supported by Murphi).
  – Non-determinism issues.
Potential improvements

• Property specification:
  – Already in Transit, vs. in Murphi.

• Counter-example highlighting.

• From counter-examples to new scenarios/snippets automatically.
### Counter-example highlighting

<table>
<thead>
<tr>
<th>Class</th>
<th>Original Transit</th>
<th>Generated Transit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Implicit</td>
<td>Networks {</td>
<td>( A \rightarrow B ))</td>
</tr>
<tr>
<td></td>
<td>[ ... ]</td>
<td>( A \rightarrow C ) )</td>
</tr>
<tr>
<td></td>
<td>}</td>
<td>( A \rightarrow D ) )</td>
</tr>
<tr>
<td>Original, input-complete</td>
<td>( A?,g,B! )</td>
<td>( A?,g,B! )</td>
</tr>
<tr>
<td></td>
<td>( \overrightarrow{A?,g,C!} )</td>
<td>( A?,g,C! )</td>
</tr>
<tr>
<td>Original, non-input-complete</td>
<td>( A?,g,B! )</td>
<td>( A?,g,B! )</td>
</tr>
<tr>
<td>Synthesized, deductively</td>
<td>( { \text{snippet}_1 ) )</td>
<td>( A?,g,B! )</td>
</tr>
<tr>
<td></td>
<td>( \ldots )</td>
<td>( A?,g,B! )</td>
</tr>
<tr>
<td></td>
<td>( { \text{snippet}_n ) )</td>
<td>( A?,g,B! )</td>
</tr>
<tr>
<td>Synthesized, arbitrarily</td>
<td>( { \text{snippet}_1 ) )</td>
<td>( A?,g,B! )</td>
</tr>
<tr>
<td></td>
<td>( \text{snippet}_3 )</td>
<td>( A?,g,B! )</td>
</tr>
<tr>
<td></td>
<td>( \text{snippet}_8 )</td>
<td>( A?,h,D! )</td>
</tr>
</tbody>
</table>

Table 1: A taxonomy of Transit transitions
Thank you – References

• Alur, Martin et al. “Protocol Design with Concolic Snippets”.
• Antti Halme, 2013, “Synthesizing the ABP with Transit”, tech. report.
• Pnueli, Rosner, 1990, “Distributed Reactive Systems are Hard to Synthesize” http://dl.acm.org/citation.cfm?id=1398817
• Lamouchi, Thistle, 2000, “Effective control synthesis for DES under partial observations” http://dl.acm.org/citation.cfm?id=1654806