Modular Code Generation

- Code generated for a composite block independently from used contexts:
Motivations

• Reusability
  • Incremental compilation
  • IP protection
• Modularity
  • Unit verification and testing
  • Parallelization
  • Scalability
• Reduction of runtime overhead
  • Speeding up simulations
Naïve SDF Code Generation

Finite-buffer schedule: FAABG

No finite-buffer schedule!
Naïve SDF Code Generation

Schedule: AAABB

Initial tokens on the channel

Schedule: AABQAB

Deadlocked!
Modular SDF Code Generation

• Non-monolithic firing function
• DSSF = Deterministic SDF with Shared FIFOs

How do we synthesize?

Can we do (AAB)(Q)(AAB)?

(AAB)(Q)(AB)
Synthesis

- Unfolding (non-homogeneous to homogeneous)

- Clustering
Clustering Algorithm Effectiveness

Valid clustering yet bad!

<table>
<thead>
<tr>
<th>Test cases</th>
<th># ins/outs</th>
<th># actors</th>
<th># nodes in unfolding graph</th>
<th># clusters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test</td>
<td>2/2</td>
<td>3</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>Entropy</td>
<td>1/1</td>
<td>15</td>
<td>1545</td>
<td>1</td>
</tr>
<tr>
<td>CD to DAT</td>
<td>1/1</td>
<td>4</td>
<td>156</td>
<td>15</td>
</tr>
</tbody>
</table>
Conclusions

• Hierarchical SDF models are not compositional
• Introduce DSSF profiles as a compositional representation of composite actors and show how this representation can be used for modular code generation
• Propose a synthesis algorithm that can handle hierarchical models of arbitrary depth
Future Work

• Implement more advanced clustering algorithms to reduce the number of clusters
• Understand the relations between the number of clusters, the size of generated code and performance
• Estimate throughput and delay of a code-generated model
References


Thank you