The 39th IEEE Real-Time Systems Symposium (RTSS'18)

Optimal Implementation of Simulink Models on Multicore Architectures with Partitioned Fixed Priority Scheduling

Shamit Bansal, Yecheng Zhao, Haibo Zeng, and Kehua Yang
Virginia Tech

VirginiaTech
Invent the Future

Outline

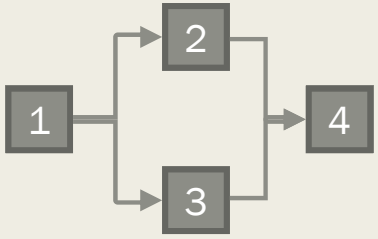
- Problem Statement
  - Software synthesis of Simulink models on multicore architectures w/ partitioned fixed-priority scheduling
- Our Approach
  - Combining offset assignment with Simulink RT blocks
  - Problem Specific Optimization Algorithm (MIXO-guided Optimization)
- Experiment Results
  - 1 to 5 orders of magnitude faster than Integer Linear Programming

Synchronous Reactive Models

- Hardware description languages
  - *VHDL, Verilog, SystemC, …*
- 99% (?) of safety-critical control software
  - *Automotive, Avionics, Rail, Nuclear, …*
- Supported in tools such as Simulink, SCADE, LabVIEW
  - *We focus on Simulink*

System Model

- A Simulink model is a directed graph $G = \langle V, E \rangle$
- Node $N_i \in V$ represents a functional block
  - *A triggering period T_i*
  - *A worst-case execution time C_i*
  - *A worst-case response time R_i*
  - *Should finish before next trigger*



- Edge (N_i, N_j) represents the communication between writer N_i and reader N_j.

- Partitioned fixed-priority scheduling of blocks on multi-core platforms

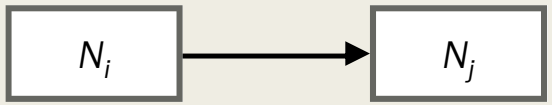
Simulink on Multicore

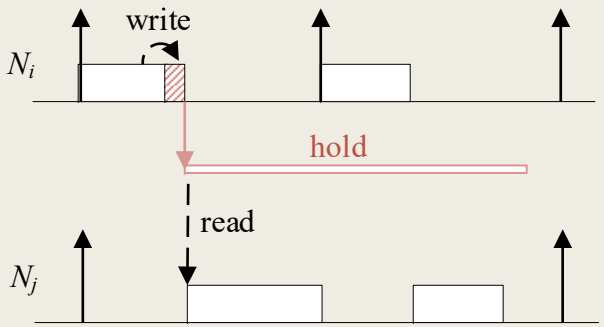
- Simulink uses rate-transition (RT) blocks to ensure data integrity and concurrency determinism

- However, RT blocks on multicore
  - *Need to be manually specified in current tools*
  - *Do not guarantee semantics preservation by themselves*
  - *May degrade control performance*

Semantics Preservation on Single-core

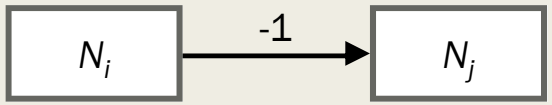
Direct-feedthrough

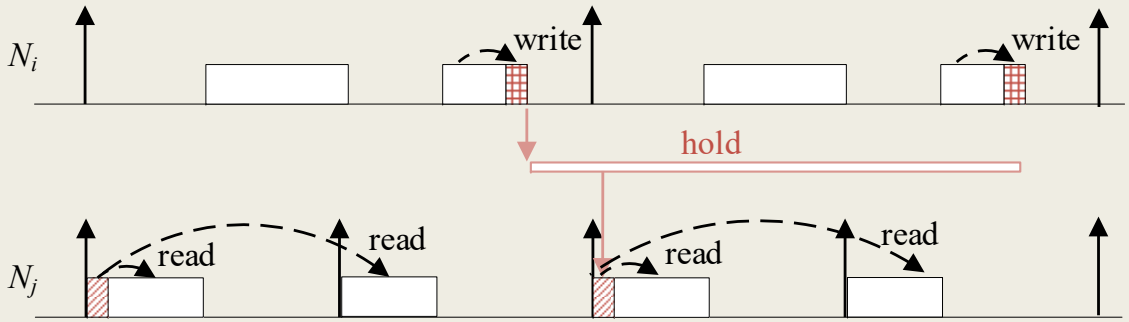




- Reader block reads from most recent instance of writer block
- Writer must finish before reader
- Enforced by assigning writer with higher priority

Unit delay

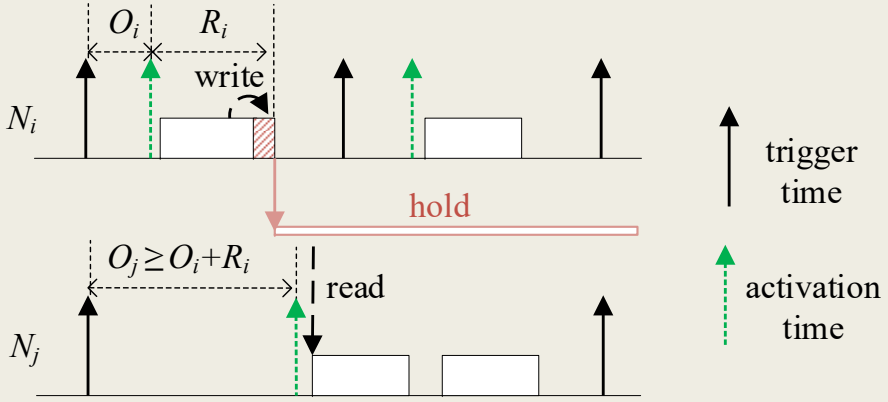




- Reader block reads from previous instance of writer block
- Reader must finish before writer
- Enforced by assigning writer with lower priority

Semantics Preservation on Multicore w/ Partitioned Scheduling

- No notion of priority among blocks assigned to different cores
- To ensure deterministic execution order, we additionally assign each block with an activation offset O_i
- Direct-feedthrough: writer instance must finish before the following reader instance



- The offset of reader block must be no smaller than the offset of writer block plus its WCRT

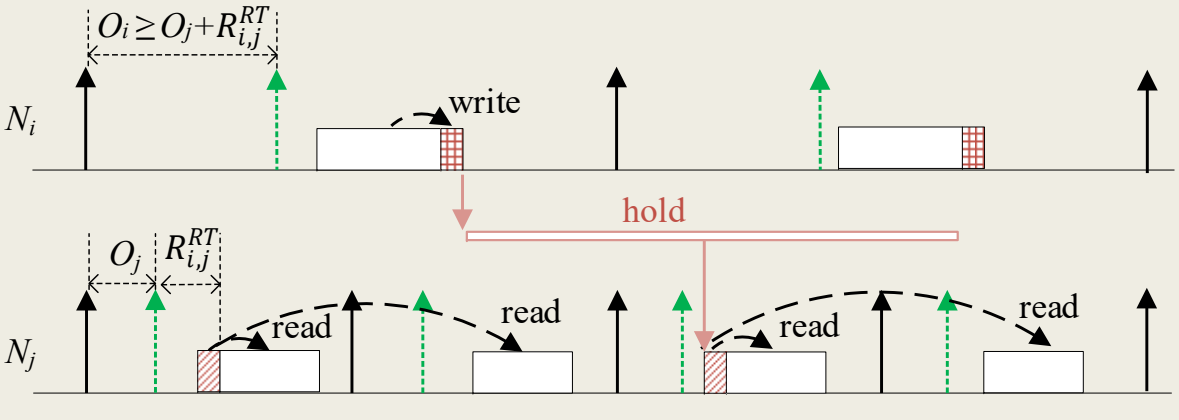
$$O_j \geq O_i + R_i$$

- The reader block must finish before its next triggered instance

$$O_j + R_j \geq T_j$$

Semantics Preservation on Multicore w/ Partitioned Scheduling

- Unit Delay: writer must not start before the RT block complete updating data for reader.
  - *Let $R_{i,j}^{RT}$ denote the completion time of the update*



- The offset of writer block must be no smaller than the offset of reader block plus $R_{i,j}^{RT}$

$$O_i \geq O_j + R_{i,j}^{RT}$$

- The writer block must finish before its next triggered instance

$$O_i + R_i \geq T_i$$

Problem Statement

- Decision variables
  - *Execution orders between each pair of communicating blocks*

$$t_{i,j} = \begin{cases} 1, & \text{direct feedthrough} \\ 0, & \text{unit delay} \end{cases}$$

  - *Priority assignment* $p_{i,j}$
  - *Offset assignment*
- Problem formulation

$$\min \sum_{\forall (N_i, N_j)} w_{i,j} \times t_{j,i}$$

$$s.t. \quad O_i + R_i \geq T_i, \forall\, i$$

$$t_{i,j} = 1 \rightarrow O_j \geq O_i + R_i$$
$$t_{i,j} = 0 \rightarrow O_i \geq O_j + R_{i,j}^{RT}$$

If N_i and N_j on different cores

$$t_{i,j} = 1 \rightarrow O_j \geq O_i \text{ and } p_{i,j} = 1$$
$$t_{i,j} = 0 \rightarrow O_i \geq O_j \text{ and } p_{i,j} = 0$$

If N_i and N_j on the same core

All linear except the calculation of R

Challenges in ILP

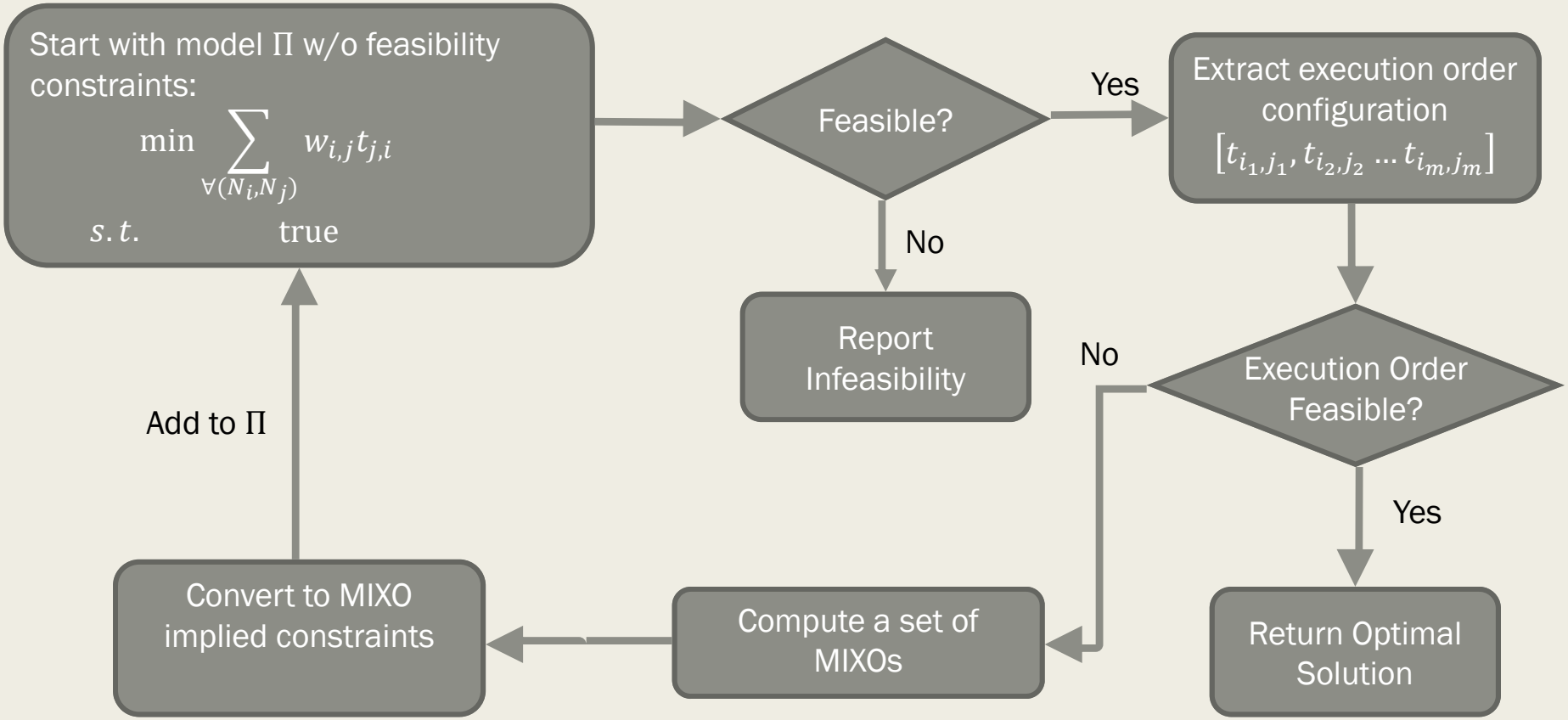
- Response time analysis requires $O(n^2)$ integer variables to formulate in Integer Linear Programming

$$R_i = C_i + \sum_{\forall j \in hp(i)} \left\lceil \frac{R_i}{T_i} \right\rceil C_j$$

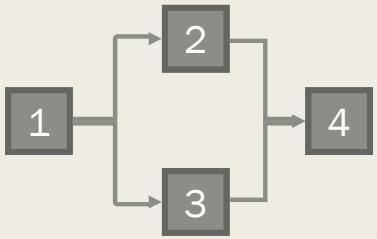
- Hard to scale to real-word problems

- We propose a problem specific framework that abstracts away the detail of timing analysis

MIXO-guided Optimization



Intuition



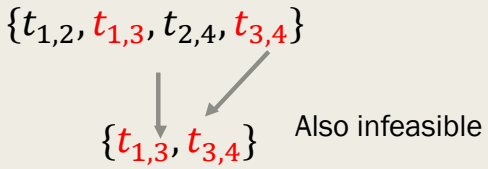
- Consider an infeasible solution $\{t_{1,2}, t_{1,3}, t_{2,4}, t_{3,4}\}$, which configures all communications as direct-feedthrough
- The solution can be removed from decision space using a simple constraint

$$t_{1,2} + t_{1,3} + t_{2,4} + t_{3,4} \leq 4$$

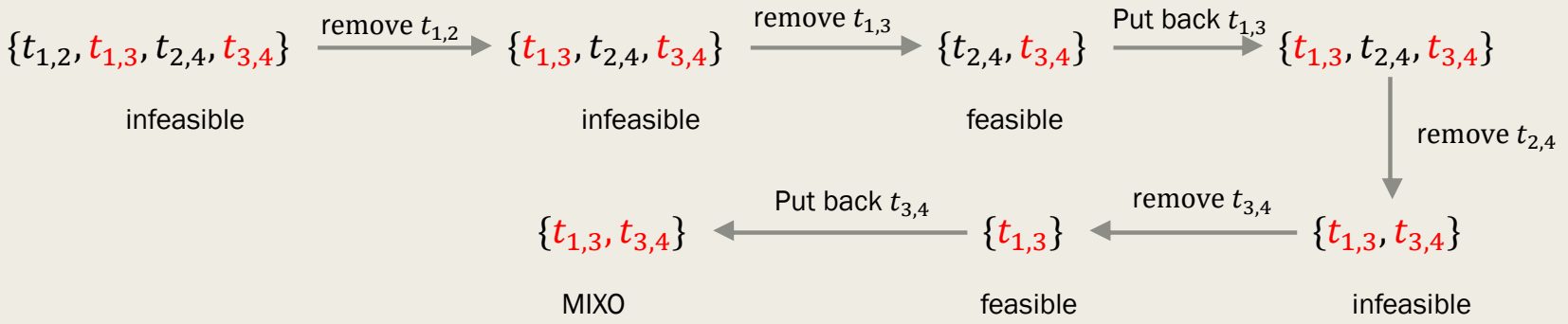
- However, the number of infeasible solutions is exponential.

The concept of MIXO

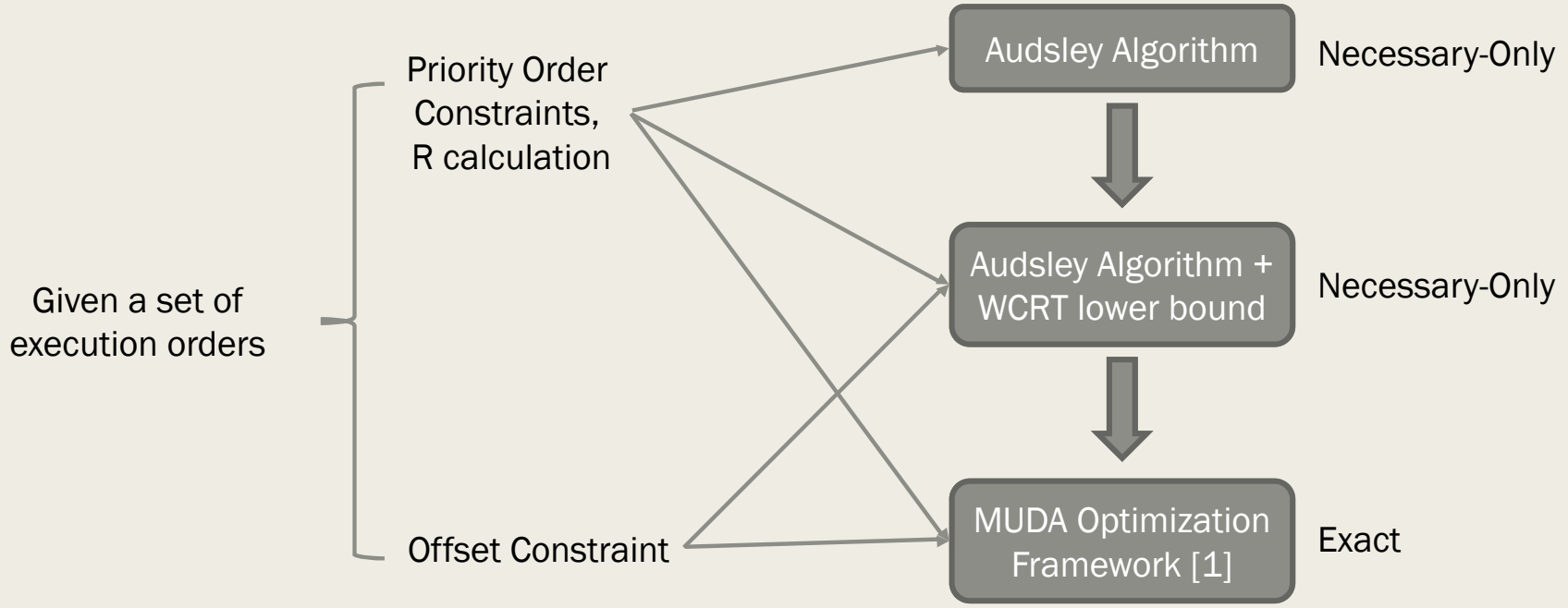
- For each infeasible solution, there are usually small subsets that are sufficient to cause infeasibility



- Find a **minimal** subset of execution orders that is infeasible
  - $\{t_{1,3}, t_{3,4}\}$—*Minimal Infeasible eXecution Orders (MIXO)*
  - $t_{1,3} + t_{3,4} \leq 2$—*MIXO implied constraint*
- Add MIXO implied constraints only when necessary (i.e., feasibility is violated).
- Example to calculate MIXO



Hierarchical Feasibility Analysis



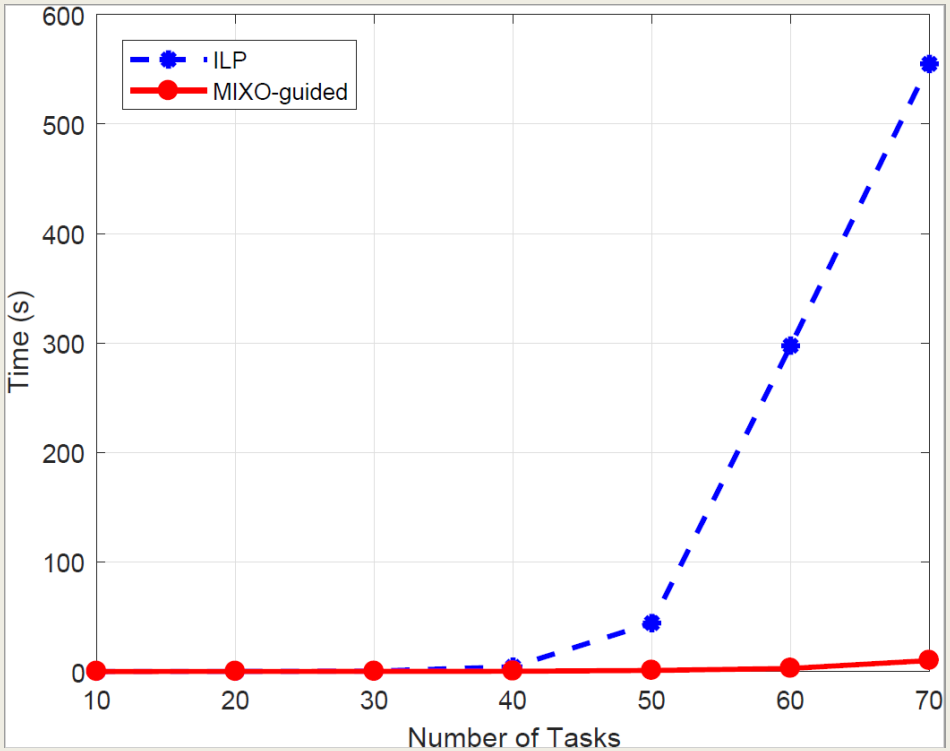
[1] Yecheng Zhao and Haibo Zeng. The Virtual Deadline based Optimization Algorithm for Priority Assignment in Fixed-Priority Scheduling. 38th IEEE Real-Time Systems Symposium (RTSS), 2017

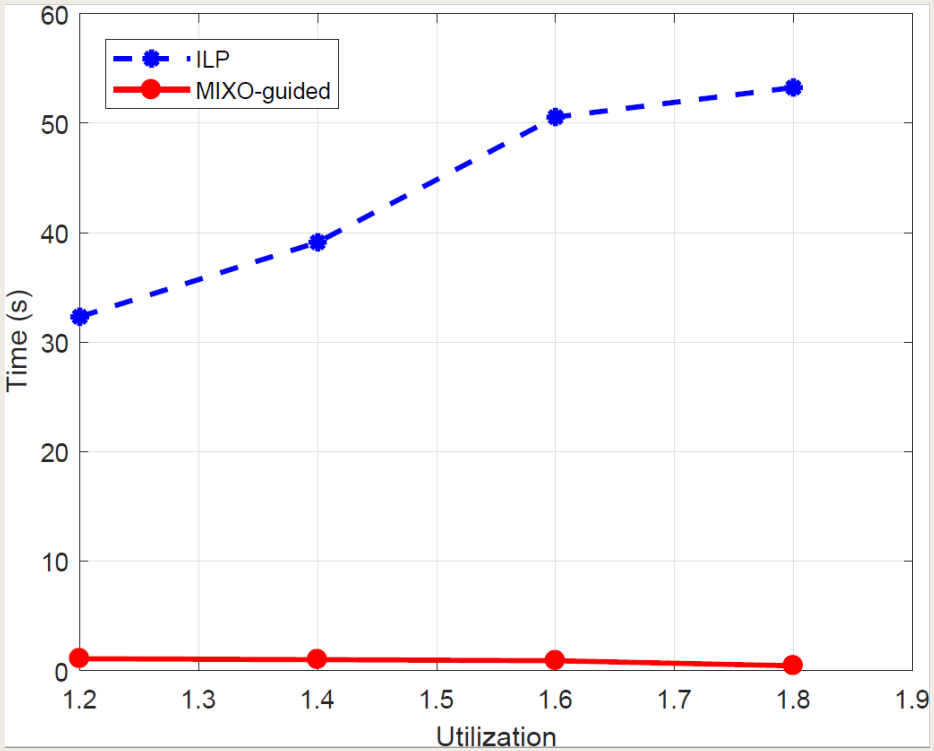
An efficient optimization procedure to optimize priority orders under schedulability constraints and linear constraints on task WCRTs

Experiment Result—Random Systems

- Experiment Settings:
  - *Dual core platforms*
  - *Number of blocks varies from 10 to 70*
  - *System total utilization varies from 1.2 to 1.8*
  - *Periods randomly selected from set {1, 5, 10, 20, 40, 50, 100, 200 400, 500, 1000} ms*
- Compared algorithms
  - *MIXO-guided Optimization*
  - *Integer Linear Programming (ILP)*

Experiment Result—Random Systems





Industrial Case Study

- An automotive fuel injection system
  - *Dual core system*
  - *90 functional blocks*
  - *106 communication links*
  - *7 different periods: 4, 5, 8, 12, 50, 100, and 1000 ms*

# Tasks		Runtime		Objective	Util
Core1	Core2	ILP	MIXO-guided		
90	0	39197 sec	0.33 sec	21	94%
80	10	13967 sec	1.36 sec	21	190%
70	20	8866 sec	10.47 sec	21	190%
60	30	2325 sec	9.22 sec	21	190%
45	45	1219 sec	5.46 sec	21	190%

Conclusion

- Problem Statement
  - Software synthesis of Simulink models on multicore architectures w/ partitioned fixed-priority scheduling
- Our Approach
  - Combining offset assignment with Simulink RT blocks
  - Problem Specific Optimization Algorithm (MIXO-guided Optimization)
- Experiment Results
  - 1 to 5 orders of magnitude faster than Integer Linear Programming

Questions?

Email: hbzeng@vt.edu