Temporal Constraints Analysis for Timing Verification of Systems

A. Azzabi, E.M. Aboulhamid, G. Nicolescu

Outline

- Introduction
- Temporal constraint analysis
- Mixed integer problem formulation
- Cplex and Branch and Cut Algorithm
- Experimentation
- Conclusion
Introduction

Motivation

- Systems more and more complex
  ⇒ Analysis and verification of temporal constraints more difficult
  ⇒ Expensive cost of the verification and validation process
- Check the consistency of a designed system
- Check if the system meets the timing requirements
- Check whether the integration of sysbystems is correct
- Locate at which level the problem occurs

Solutions

- Timing verification using simulation
  Exhaustive simulation impractical for large systems
- Timing verification using analytical algorithms
  Speeds up the verification process
  ⇒ Acceleration of the overall design process and time-to-market.
Given an acyclic set of timing constraints, find the maximum separation between two specific events:

  - Solve the problem with min-max-linear constraints
  - Exponential worst case time
  - Proof of NP-completeness of the min-max problem

- Burks and Sakallah, Min Max Linear programming and the timing analysis of digital circuits, *ICCAD 1993*
  - Uses branch and bound algorithm

---

**Systems with non-repetitive behavior (cont’d)**

  - Other adaptation of branch and bound algorithm to solve the problem with min-max-linear constraints.

  - Using linearization and a branch and bound algorithm
  - Assume – commit constraints
  - Local consistency
Cyclic systems with repetitive behavior

  - Using unfolded graphs (allow only relations between two successive cycles – height = 1)

  - Adaptation of the previous approach. Still many limitations on the number of min constraints to handle, etc.

Approach

- Verification of cyclic systems with repetitive behavior

- Handling all constraint types (Min-Max-Linear). No unfolding.

- Formulation of the problem as a mixed integer program then use a branch and cut algorithm implemented within a solver
Temporal constraint analysis

- Events with timing constraints

- Time to fire a transition is between 0 and ∞
- Weights - a lower and an upper bounds of the delay constraint
- Height: specify after how many cycles the next events occur
- Timing Model based on min-max-linear constraints
- Solve the problem of maximum separation.

Temporal constraints and Mixed I.P. formulation

- \(1 \leq a_{n+1} - b_n \leq 2 \Rightarrow 1 \leq a_n - b_n + 2p \leq 2\)
- \(2 \leq c_n - a_n \leq 2\)
- \(\max(d_{n+1}+1, c_n+1) \leq b_n \leq \max(d_{n+1}-p, c_n+1) \Rightarrow \max(d_{n+1}-p, c_n+1) \leq b_n \leq \max(d_{n+1}-p, c_n+1)\)
- \(2 \leq d_n + b_n \leq 2 \Rightarrow 2 \leq d_n - b_n + p \leq 2\)
- \(1 \leq c_n - d_n \leq 1 \Rightarrow 1 \leq c_n - d_n + p \leq 1\)

Inter-iteration constraints ⇒ No need for unfolding
Mixed integer problem formulation

Maximize
Obj: \( a - b + 2p \)

subject to:
\[
\begin{align*}
  a - b + 2p & \leq 2; & a - b + 2p & \geq 1 \\
  a - b & \leq 1; & a = 1 & \Rightarrow b - d + p \leq 1 \\
  b - c & \geq 1; & b - d + p & \geq 1 \\
  c - a & \leq 2; & c - a & \geq 2 \\
  d - b + p & \leq 2; & d - b + p & \geq 2 \\
  e - d + p & \geq 1; & e - d + p & \leq 1 \\
\end{align*}
\]

\( \alpha = 0 \Rightarrow b - c \leq 1; \) \( \alpha = 1 \Rightarrow b - d + p \leq 1 \)

\( a b c d f \) binaries; \( \alpha \) semi-continuous

\( \alpha \leq max (d_n+1-p, c_n+1) \)

- Can deduce Integer k-periodic schedule
- Complexity comes from binaries and computation of \( p \) otherwise shortest path algorithm

Results: \( p \in [2, 2.5] \)

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>[0,0]</td>
<td>[-3,-3]</td>
<td>[-2,-2]</td>
<td>[-3,-2.5]</td>
<td>[-1,-1]</td>
</tr>
</tbody>
</table>

Temporal constraint analysis

- Maximum separation time computation
- Complexity for graphs with V vertexes and E edges

<table>
<thead>
<tr>
<th>Constraint type</th>
<th>Complexity</th>
<th>Proposed by</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear only</td>
<td>( O(VE) )</td>
<td>Shortest path algorithms</td>
</tr>
<tr>
<td>Max only</td>
<td>( O(E) )</td>
<td>McMillan and Dill, [5]</td>
</tr>
<tr>
<td>Max+Linear</td>
<td>( O(V^2\log V + VE) ), conjecture</td>
<td>T.Y.Yen and al. [3]</td>
</tr>
<tr>
<td>Min+Max</td>
<td>( NP )-complete</td>
<td>McMillan and Dill [1]</td>
</tr>
<tr>
<td>Min+Max+Linear</td>
<td>( NP )-complete</td>
<td>T.M.Burks, K.A.Sakallah [2]</td>
</tr>
<tr>
<td>Min-Max Cyclic</td>
<td>( O((\sum K_{max} V)^2 \cdot p) )</td>
<td>(Approximative) Dill, Chakraborty, Yen [5]</td>
</tr>
</tbody>
</table>

MPSOC 2010
Cplex and Branch and Cut Algorithm

- Solver implements many algorithms used for solving different types of linear problems
- Offers a mechanism to express non-linear functions which helps us to express the min and max constraints.
- Method used by Cplex: branch and cut.
  - Relax the integer constraints and use regular simplex algorithm
  - Use a cutting plane algorithm to improve the relaxation
  - Execute the branch and bound algorithm for the sub-problems and iterate until a solution is found satisfying all the integer constraints

Experimentation

Asynchronous Intel differential equation solver chip used in [5]

(a) Data flow graph and (b) Architecture of differential equation solver[5]
Experimentation

- Correctness of the design depends on the delay of the signal A1M.
  - After several hours of SPICE simulation designers discovered that the chip could malfunction if the delay of the buffer driving A1M was small.
  - We discover this in few minutes by reducing the delay of the buffer driving A1M.
  - Identical results (same time separations) in few minutes to Chakraborty et al.
    - Took them 3 hours to verify the results obtained in few seconds with their approximate algorithm.
- Used randomly generated graphs to test our approach
  - Difficult to find large examples

MPSOC 2010

---

Experimentation

- No related works for the cyclic systems with different heights

<table>
<thead>
<tr>
<th>Nb of events</th>
<th>Nb Min-Max events</th>
<th>Nb computed sep</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>120</td>
<td>17</td>
<td>120</td>
<td>1s</td>
</tr>
<tr>
<td>152</td>
<td>37</td>
<td>152</td>
<td>3.8s</td>
</tr>
<tr>
<td>688 (acyclic unfolded)</td>
<td>688</td>
<td>1376</td>
<td>2.4 min (1.4 with 2 threads)</td>
</tr>
<tr>
<td>176 (folded cyclic)</td>
<td>176</td>
<td>176</td>
<td>4 min 4 s (with precomputed period)</td>
</tr>
</tbody>
</table>

MPSOC2010
Conclusion

- Integer Programming Formulation
  - Efficient execution with exact formulation, no unfolding
  - Cyclic behavior with different heights, no limitation on the type of constraints or bounds
  - Can compute periods
  - Allows k-periodic scheduling
  - Compute maximum separations
- Can be applied to different levels of abstraction

Future work:
- Add assume and commit constraints
- Customize the branch and cut algorithm to fit our problem.

Reference

Temporal constraint analysis

• Problem of timing verification
  \[ s_y = \max_i \left( t(e_i) - t(e_j) \right) \]

• Constraint interpretation

\[
\begin{align*}
\max_i \{t(e_i) + l_{ij}\} & \leq t(e_j) \leq \min_i \{t(e_i) + u_{ij}\} \\
19 \leq t(D) & \leq 20 \\
\max_i \{t(e_i) + l_{ij}\} & \leq t(e_j) \leq \max_i \{t(e_i) + u_{ij}\} \\
19 \leq t(D) & \leq 27 \\
\min_i \{t(e_i) + l_{ij}\} & \leq t(e_j) \leq \min_i \{t(e_i) + u_{ij}\} \\
10 \leq t(D) & \leq 20
\end{align*}
\]