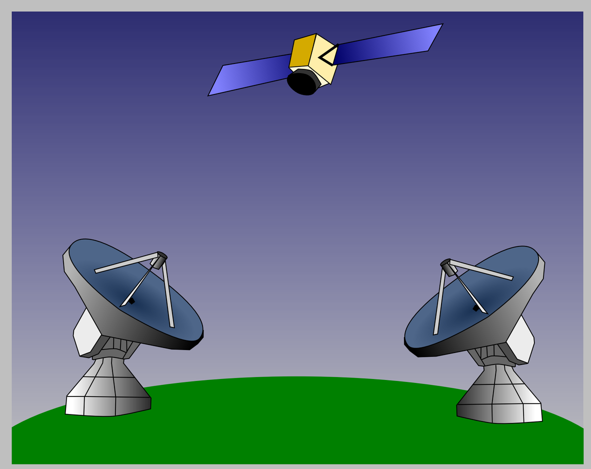
Learning Temporal Properties over Data Streams

Samuel Huang, Rance Cleaveland

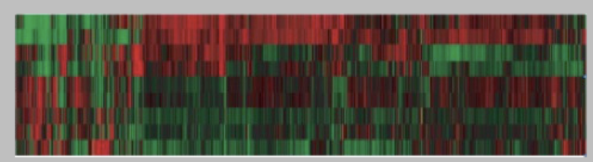
Department of Computer Science

Setting

- Data streams are pervasive!
- Many result from outputs of structured processes



(a) Software engineering

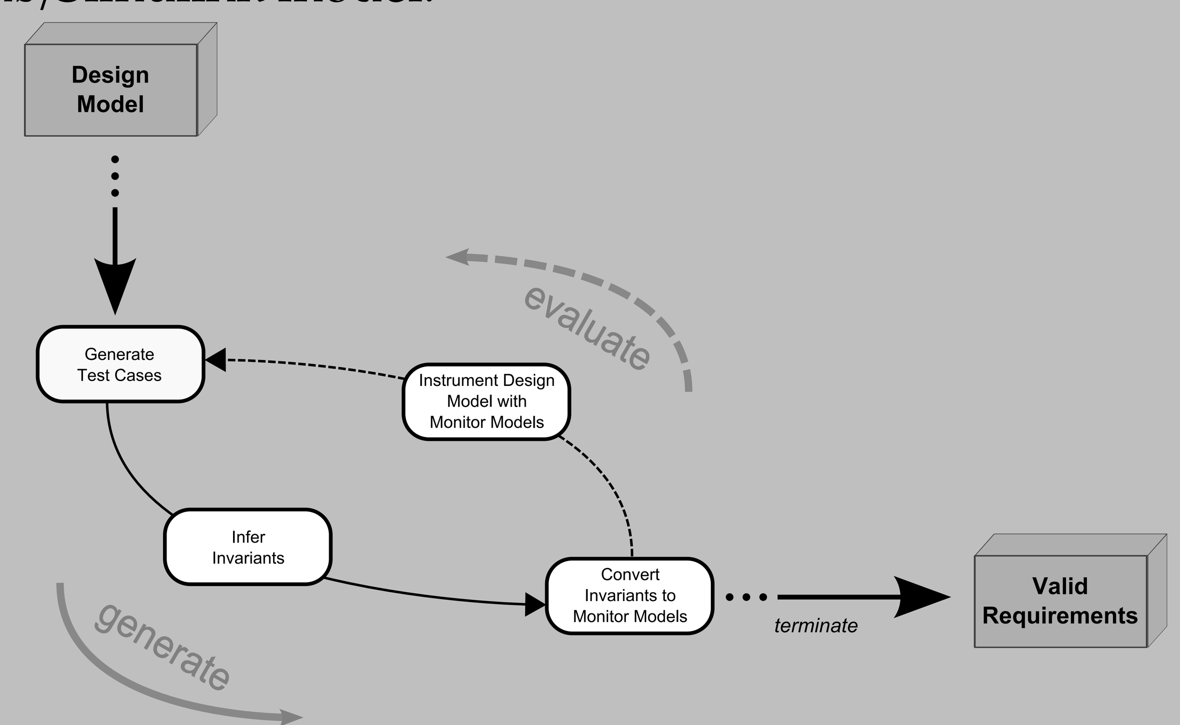


(b) Metabolic pathways

- Question: Can we reason about a process's internal structure via reasoning over these outputs?

Early Efforts

- Pilot study [1]: real-time recovery of invariants from execution traces of known programs in the automotive domain.
- Used combination of data mining-based techniques [2] and Instrument-Based Verification (IBV) for discovering rules based on set of test cases (input output pairs) from a Matlab/Simulink model:



- Allowed for verification of specifications that could be represented as:

$$a \wedge b \wedge c \wedge \ldots \rightarrow \alpha \wedge \beta \wedge \gamma \wedge \ldots$$

- Incorporated notions of a rule's *support* and *confidence* to select significant and accurate rules. The approach was shown to be robust to noise, and allowed for detection of incorrect implementations/specifications.

Expanding Expressiveness

- Previous work towards mining properties, (e.g. [5]), but all recover restricted classes of properties and are not "complete."
- Different domains necessitate different classes of interest:
  - Software engineering:

    $$\mathbf{G}\,(\text{lock} \rightarrow \mathbf{F}\,\text{release\_lock})$$

  - Metabolic pathways:

    $$\mathbf{G}\,(\uparrow \text{protA}\ \mathbf{U}\ \downarrow \text{protB})$$

- Currently would have to use different techniques/approaches to handle per-domain properties. In more complex domains, multiple techniques would be required to cover the span of all "interesting" properties.
- Would like to mine properties from a larger space of more interesting properties, such as all of LTL or CTL.

Model Checking as an Oracle

- Model checking traditionally answers "does model $\mathcal{M} \models \phi$?"
- Can modify this to ask "does data stream d satisfy property ϕ?"
- Equivalent to verifying if one particular execution trace of $\mathcal{M}$ (path through the program's state machine) complies with ϕ.

> **Key point:**
> Performing verification of property ϕ over a set of data streams generated by a model $\mathcal{M}$ gives a good indication if $\mathcal{M} \models \phi$.

- Can use existing model checking algorithms/solvers for data stream verification (e.g. NuSMV [4]). Simplicity of data stream structure aids in efficiency of this model checking.
- Simulators could also be used (e.g. BioNetGen [3]).

Learning from Data Streams

- Given a hypothesis property ϕ, we can assign it a fitness (potential) based upon its success in satisfying each of the data streams.
- This fitness can help guide our search through the hypothesis space (such as space of all LTL formulas).

Sampling/Handling Noise

- Consider the set of all data streams $\mathcal{D}$ capable of being emitted from a model $\mathcal{M}$.
- Practically, we would only have a sample of streams from $\mathcal{D}$ that have been observed.
- Biasing of this sample can lead to interesting situations. How is it biased?
  - Of all possible starting conditions, only some may be observed: draw firm conclusions for only this class of scenarios.
  - What about noise introduced by erroneous/buggy systems? Or an adaptation to the underlying process for a fraction of the streams?

Initial Results

- Currently using a genetic programming approach to search space of possible CTL solutions.
- Success for recovering properties on small examples such as:
  - $a\ \mathbf{U}\ b$
  - $a \rightarrow \mathbf{F}\ b$
- Investigating impact of noise on results, as well as scaling up to larger applications (e.g. software systems, metabolic pathways) and more complex patterns.

References

[1] Christopher Ackermann, Rance Cleaveland, Samuel Huang, Arnab Ray, Charles P. Shelton, and Elizabeth Latronico.
Automatic requirement extraction from test cases.
In *RV*, pages 1–15, 2010.

[2] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami.
Mining association rules between sets of items in large databases.
In *SIGMOD '93: Proceedings of the 1993 ACM SIGMOD international conference on Management of data*, pages 207–216, New York, NY, USA, 1993. ACM.

[3] Michael L. Blinov, Jin Yang, James R. Faeder, and William S. Hlavacek.
Graph theory for rule-based modeling of biochemical networks.
pages 89–106, 2006.

[4] Alessandro Cimatti, Edmund M. Clarke, Enrico Giunchiglia, Fausto Giunchiglia, Marco Pistore, Marco Roveri, Roberto Sebastiani, and Armando Tacchella.
Nusmv 2: An opensource tool for symbolic model checking.
In *CAV*, pages 359–364, 2002.

[5] David Lo, Siau-Cheng Khoo, and Chao Liu.
Mining past-time temporal rules from execution traces.
In *WODA*, pages 50–56, 2008.