Leveraging Parallelization to Synthesize Recomposition Maps for Advancing Model Checking

Eduardo S. Lozano, *Stony Brook University*
eduardo.lozanonaranjo@stonybrook.edu

Mentors: Ian Dardik, Eunsuk Kang
SoDa Lab @ CMU

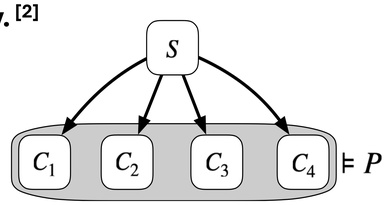
Background

Verification is the process of proving a system (specification) satisfies a property. In the real world it is utilized to ensure the safety of aerospace, healthcare, autonomous vehicle systems, etc.[1] Verification allows for extensive checking of a system.

Specification → $S \models P$ ← Property

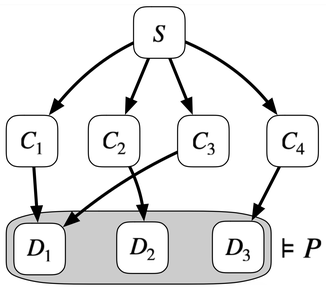
Compositional Verification

Compositional Verification is a divide and conquer technique in which a system is broken down into components with their own properties. If all the components satisfy their property, the specification also satisfies its property. [2]



Recomposition

Recomposition is a technique where components C_i are strategically combined into new components D_i for verification. This approach can make compositional verification significantly more efficient (in verification time and state space) than verifying the original specification directly. [3]



$$C_1 || C_3 = D_1$$

Problem

The usual technique in compositional verification has been to run a singular recomposition map. However, no singular heuristic has been found to be more efficient. [4]

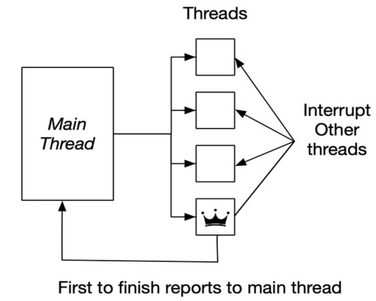
Objectives

Expand the state of the art of model checking by:

- Building a program to run recomposition maps in parallel
- Designing new recomposition map synthesis techniques
- Comparing the performance of our technique against the state of the art

Methods

We run one thread per recomposition map. The first of these threads to finish (marked by crown), then interrupts the other threads and returns the results of verification back to the main thread.



Future Work

As there are many possible recomposition maps, there's a lot of potentials optimizations!

For example:

- Efficiently running all re-composition maps
- Sharing resources between different threads
- Determine which recomposition strategies work well with specific *categories* of specifications
  - This will help reduce the search space of recomposition maps

Works Cited

[1] Rozier, K.Y.: Specification: The biggest bottleneck in formal methods and autonomy. In: Verified Software. Theories, Tools, and Experiments: 8th International Conference, VSTTE 2016, Toronto, ON, Canada, July 17–18, 2016, Revised Selected Papers 8, pp. 8–26. Springer International Publishing (2016).
[3] Cobleigh, J., Giannakopoulou, D., Pasareanu, C.: Learning Assumptions for Compositional Verification. RIACS Technical Report 02.09, Kestrel Technologies, University of Massachusetts, Amherst, November 2002
[3] Dardik, I., Porter, A., Kang, E.: Recomposition: A New Technique for Efficient Compositional Verification. In: *to appear in FMCAD 2024*.
[4] Cobleigh, J.M., Avrunin, G.S., Clarke, L.A.: Breaking up is hard to do: An investigation of decomposition for assume-guarantee reasoning. In: Proceedings of the 2006 International Symposium on Software Testing and Analysis (ISSTA '06), pp. 97–108. Association for Computing Machinery, Portland, Maine, USA (2006).