Boosting Verification Scalability via Structural Grouping and Semantic Partitioning of Properties

Rohit Dureja[*], Jason Baumgartner[†], Alexander Ivrii[†], Robert Kanzelman[†], Kristin Y. Rozier[*]

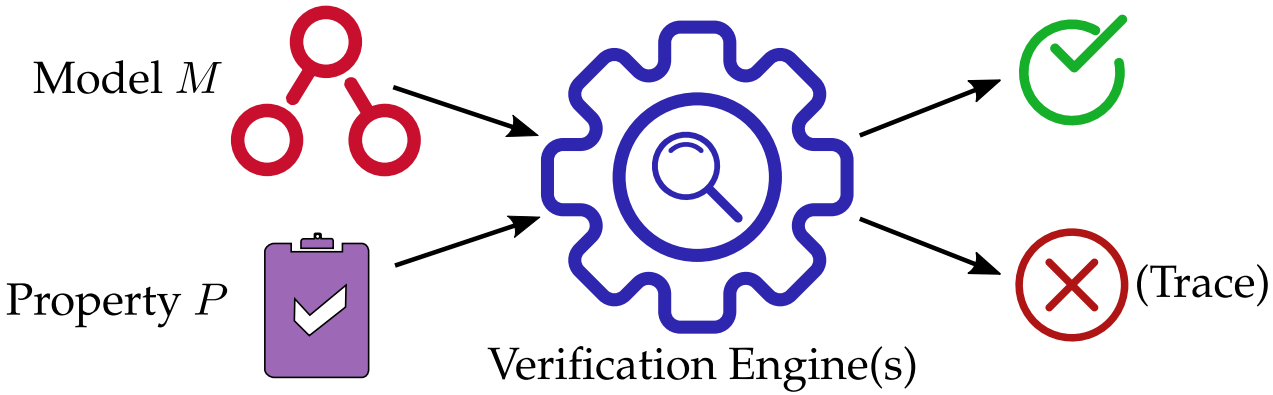
[*] Iowa State University

[†] IBM Corporation

fmcad.19

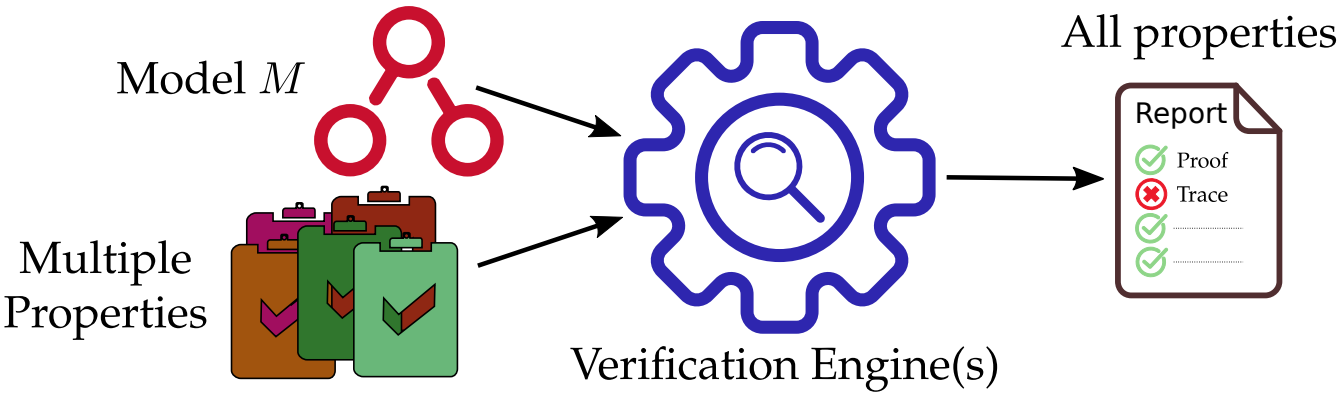
October 23, 2019

Model Checking



Usually multiple properties to be verified

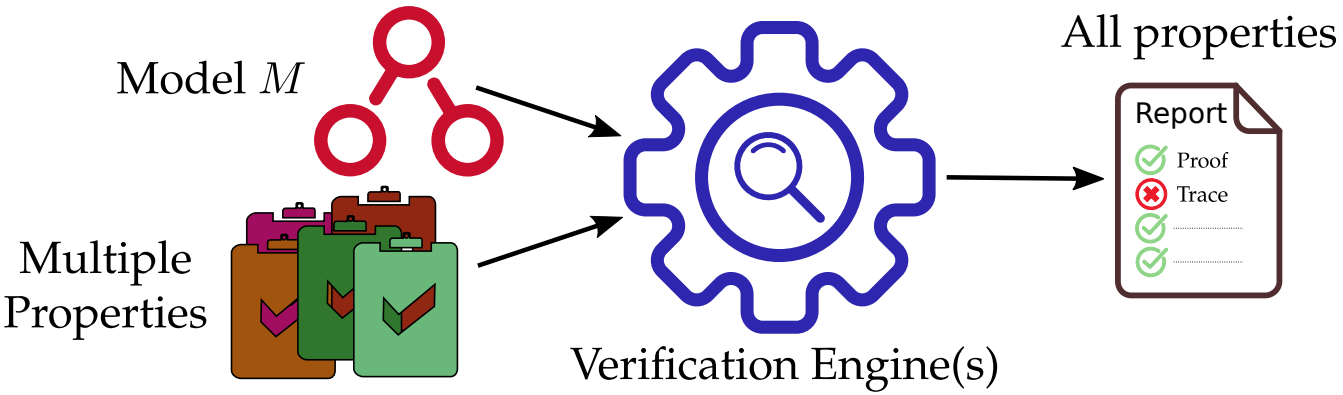
Model Checking



Make multi-property verification scalable

Multi-Property Verification

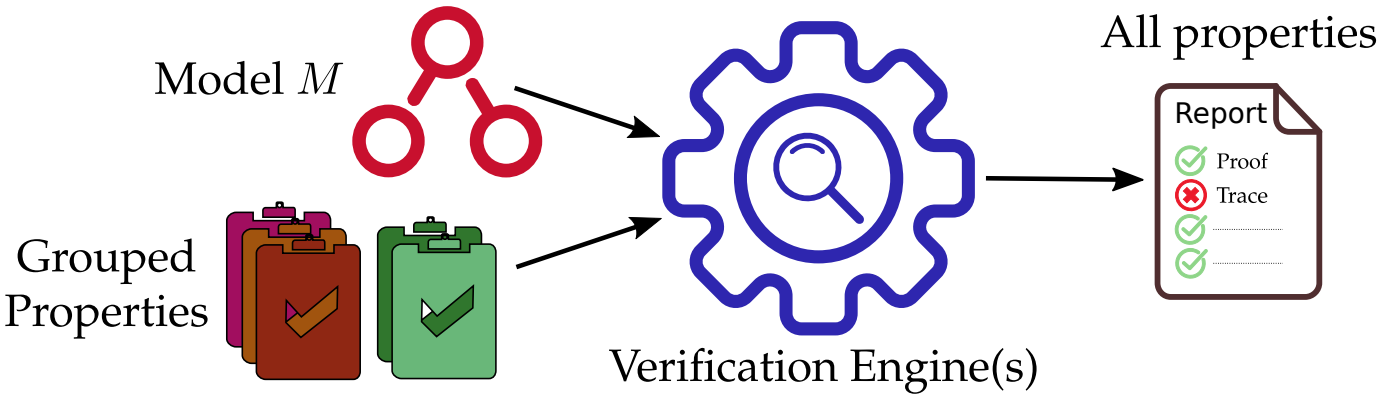
- Properties checked concurrently, or one-at-a-time
  - Doesn't optimally exploit sub-problem sharing



Opportunity to save verification resources!

Improved Multi-Property Verification

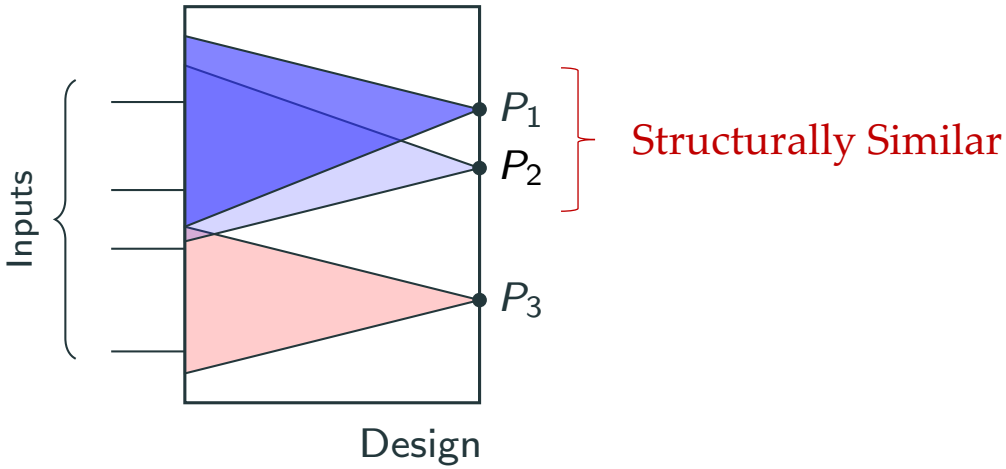
- Group 'high-affinity' properties; similarity metric
  - Properties in a group are concurrently solved; parallel groups
  - Engine effort reused across properties in a group



What similarity metric to use?

Similarity Measure

- Every property has distinct minimal cone-of-influence (COI)
- Multiple properties → exponential complexity w.r.t to collective COI
  - Concurrent verification slower that one-at-a-time
- Nearly identical COI → save verification resource*
  - Experimental demonstrated, offline-grouping



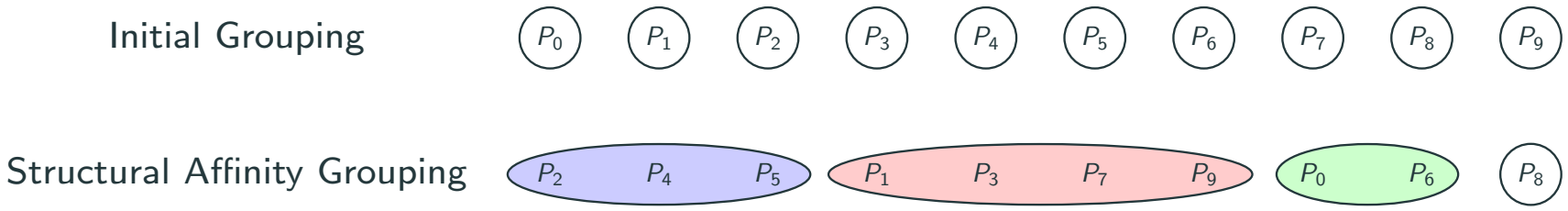
* G. Cabodi, P. E. Camurati, C. Loiacono, M. Palena, P. Pasini, D. Patti, and S. Quer, "To split or to group: from divide-and-conquer to sub- task sharing for verifying multiple properties in model checking," *International Journal on Software Tools for Technology Transfer (STTT)*, vol. 20, no. 3, pp. 313–325, Jun 2018

Our Contributions

- Online procedure to partition properties into high-affinity groups
  - Near-linear runtime and automated; provable affinity bounds

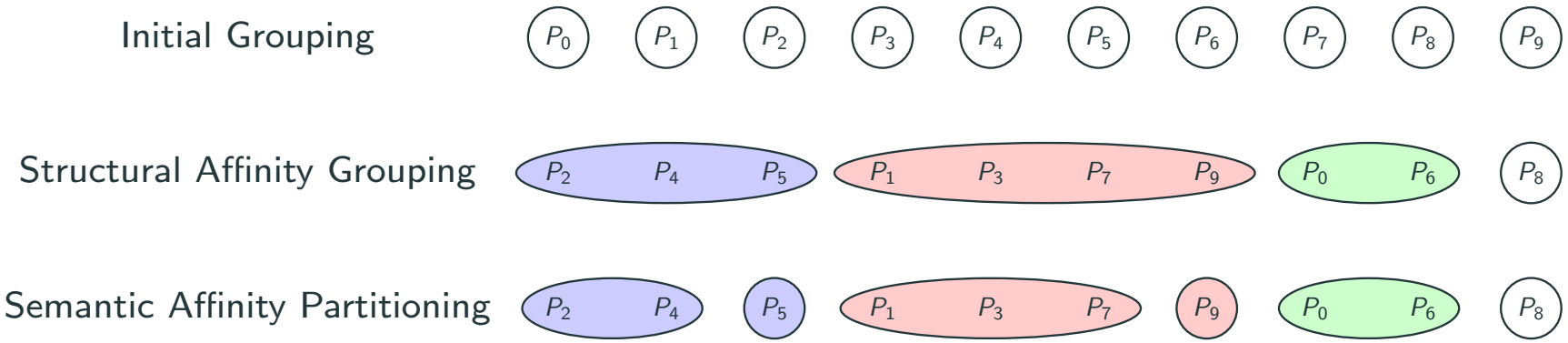
Our Contributions

- Online procedure to partition properties into high-affinity groups
  - Near-linear runtime and automated; provable affinity bounds
- Property grouping based on cone-of-influence
  - Structural information (static)
- Structurally-similar properties may have different semantics
  - Subset of design logic in cone-of-influence



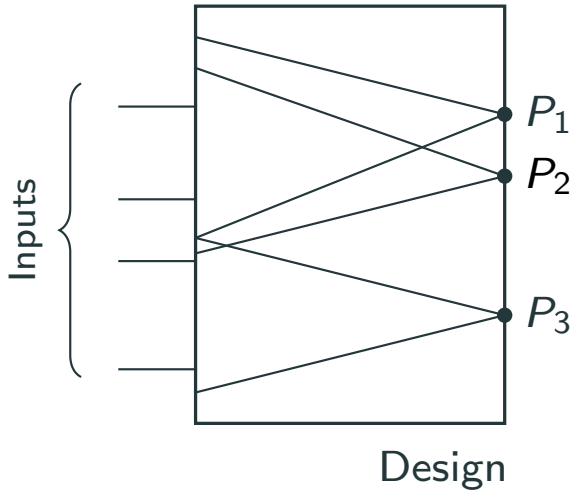
Our Contributions

- Online procedure to partition properties into high-affinity groups
  - Near-linear runtime and automated; provable affinity bounds
- Property grouping based on cone-of-influence
  - Structural information (static)
- Structurally-similar properties may have different semantics
  - Subset of design logic in cone-of-influence
- Property-group refinement using localization abstraction
  - Semantic information (dynamic)



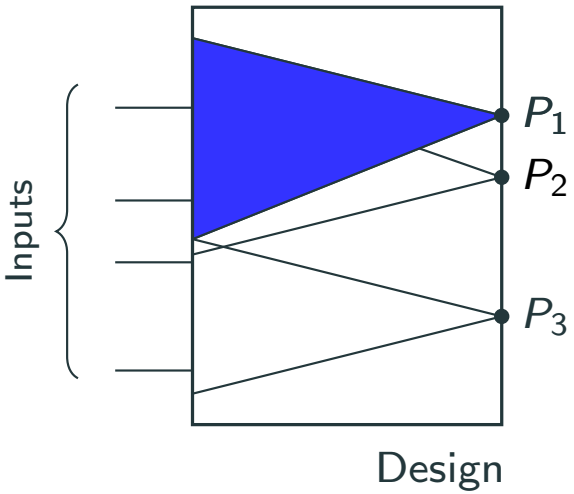
Cone-of-Influence Computation

Iterative



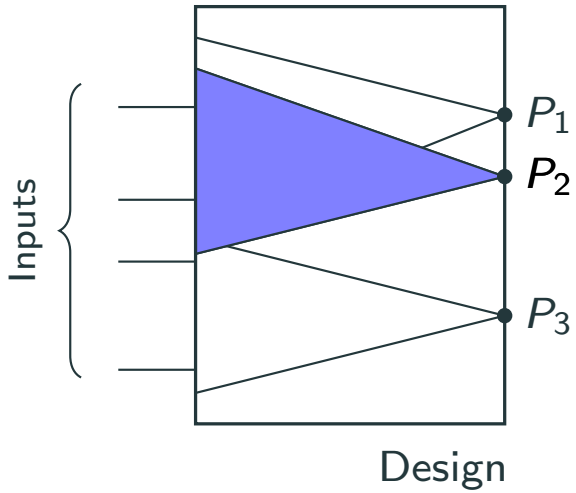
Cone-of-Influence Computation

Iterative



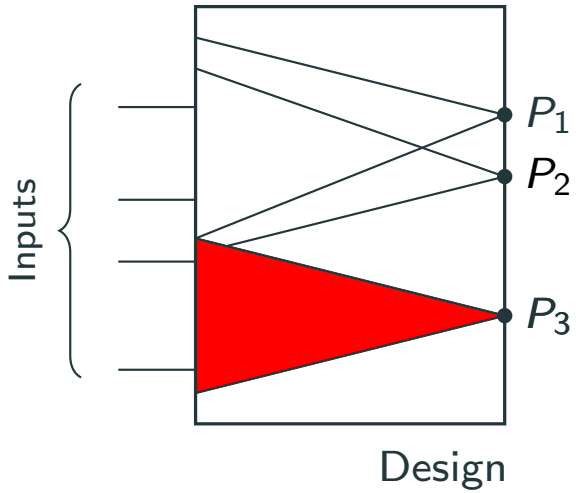
Cone-of-Influence Computation

Iterative



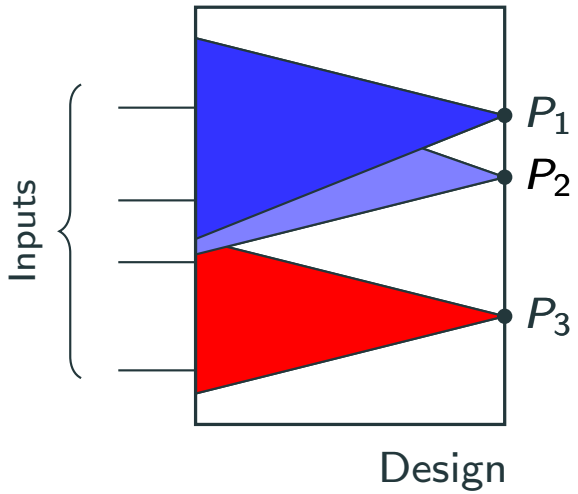
Cone-of-Influence Computation

Iterative

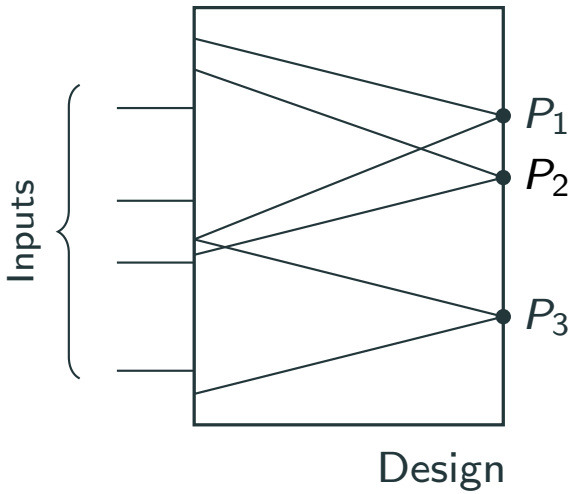


Cone-of-Influence Computation

Iterative



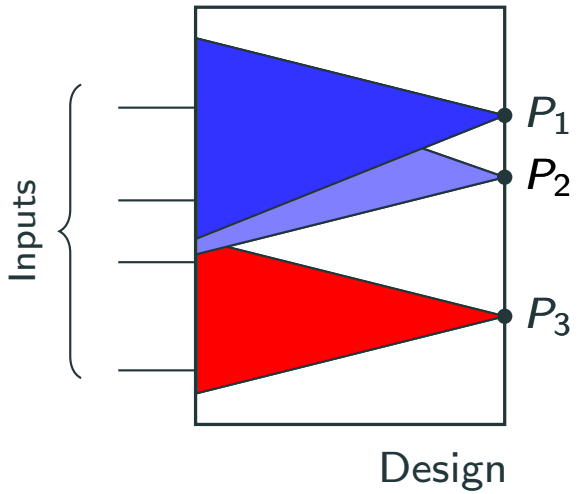
Our Method



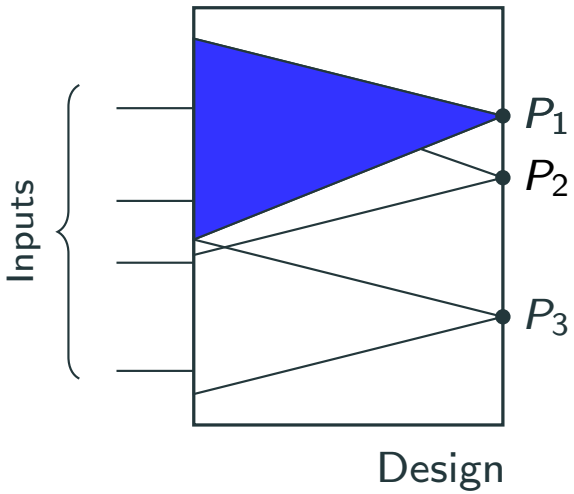
- Repeated traversals
- Does not scale!

Cone-of-Influence Computation

Iterative



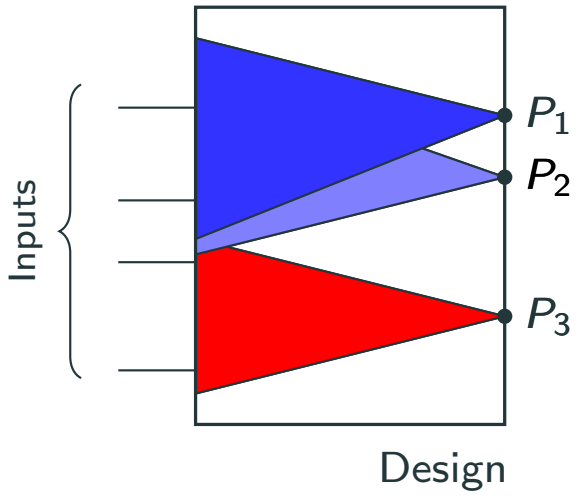
Our Method



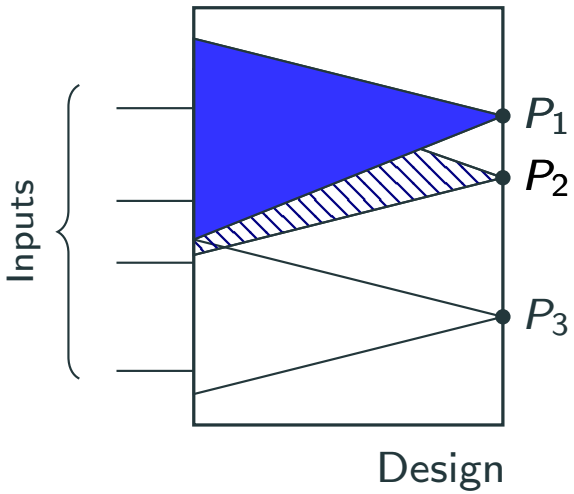
- Repeated traversals
- Does not scale!

Cone-of-Influence Computation

Iterative



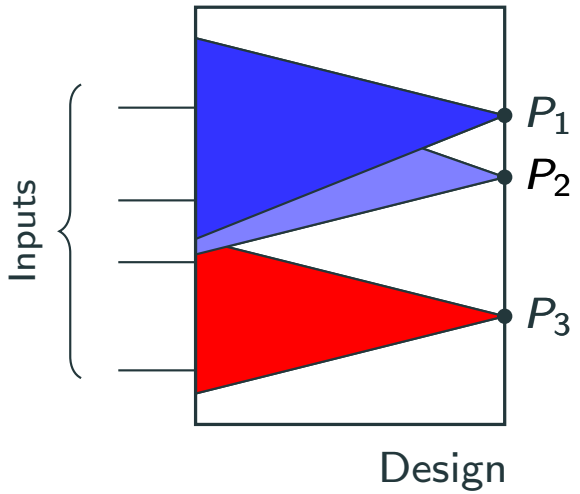
Our Method



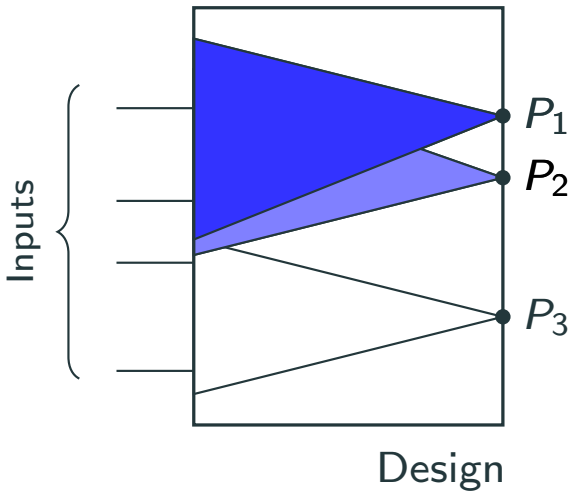
- Repeated traversals
- Does not scale!

Cone-of-Influence Computation

Iterative



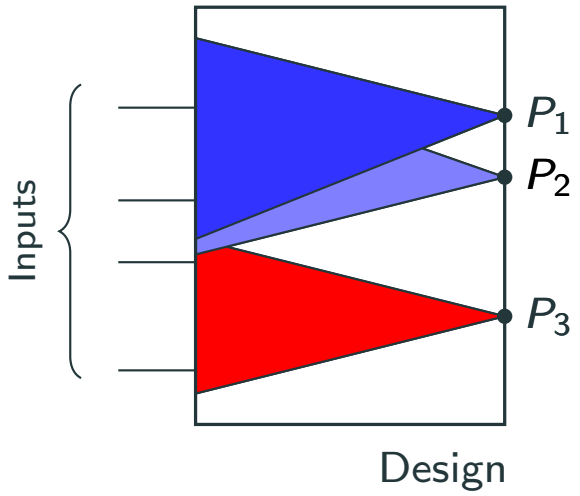
Our Method



- Repeated traversals
- Does not scale!

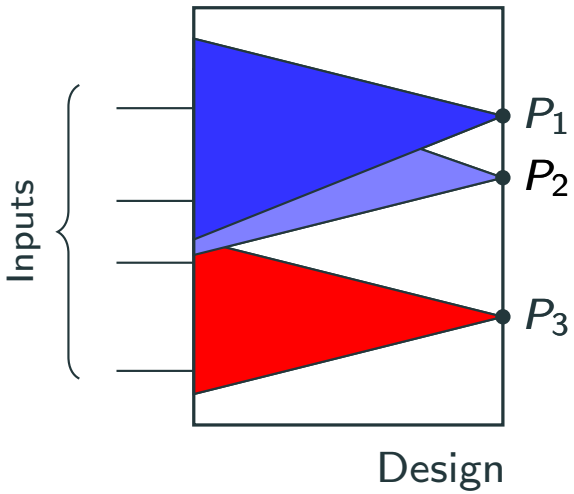
Cone-of-Influence Computation

Iterative



- Repeated traversals
- Does not scale

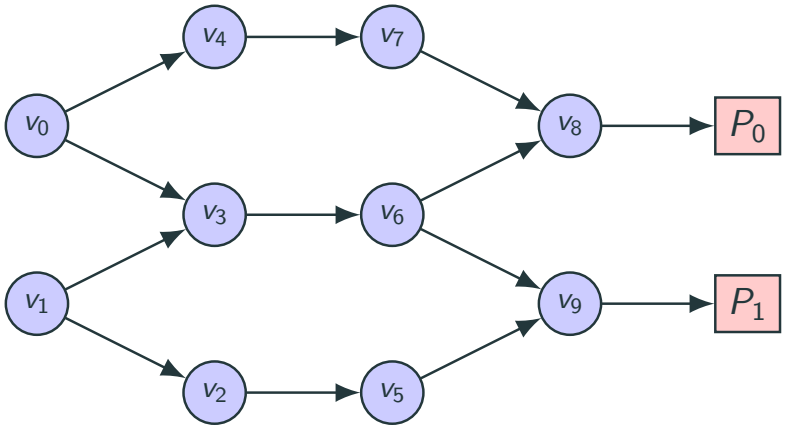
Our Method

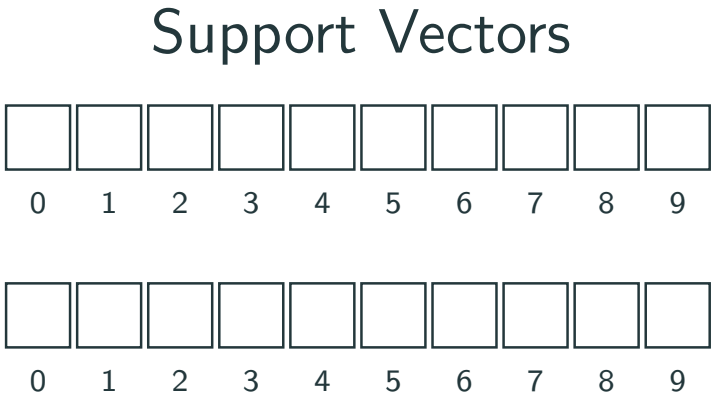


- **One** traversal
- Very scalable

COI Computation via Support Vectors

- *Support variable* – registers and inputs in COI
- Represent every support variable as a bit
  - Bitvector operations to compute support (linear)

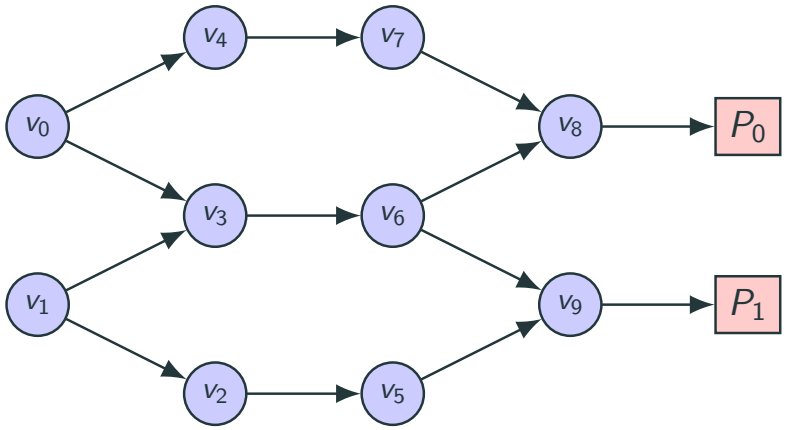


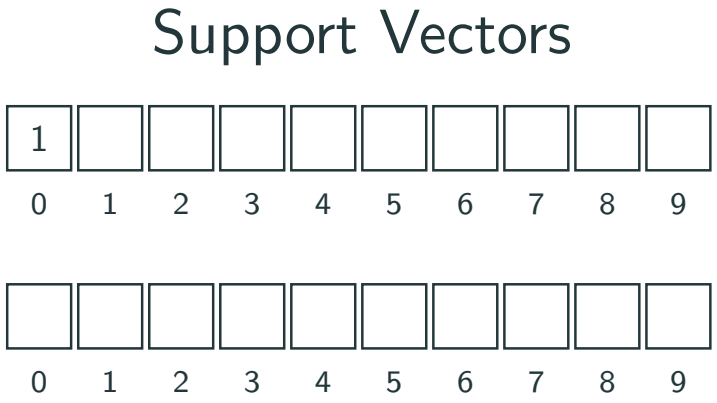


G. Cabodi, P. Camurati, and S. Quer, "A graph-labeling approach for efficient cone-of-influence computation in model-checking problems with multiple properties," *Software: Practice and Experience*, vol. 46, no. 4, pp. 493–511, 2016.

COI Computation via Support Vectors

- *Support variable* – registers and inputs in COI
- Represent every support variable as a bit
  - Bitvector operations to compute support (linear)

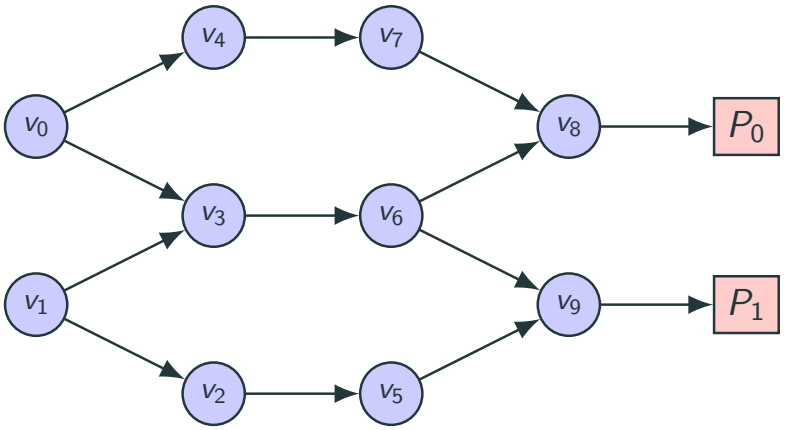


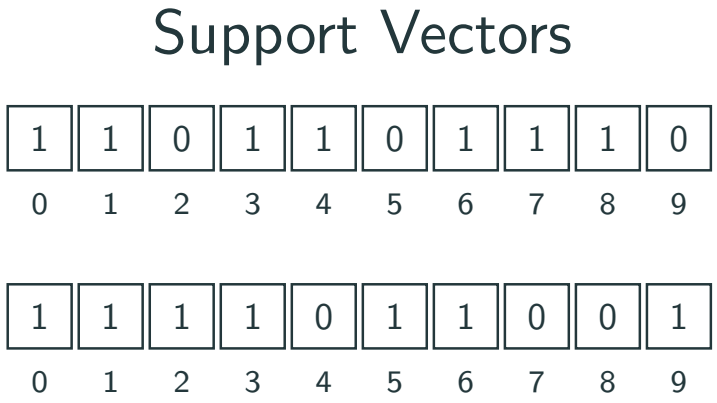


G. Cabodi, P. Camurati, and S. Quer, "A graph-labeling approach for efficient cone-of-influence computation in model-checking problems with multiple properties," *Software: Practice and Experience*, vol. 46, no. 4, pp. 493–511, 2016.

COI Computation via Support Vectors

- *Support variable* – registers and inputs in COI
- Represent every support variable as a bit
  - Bitvector operations to compute support (linear)
  - Constant-time inspection

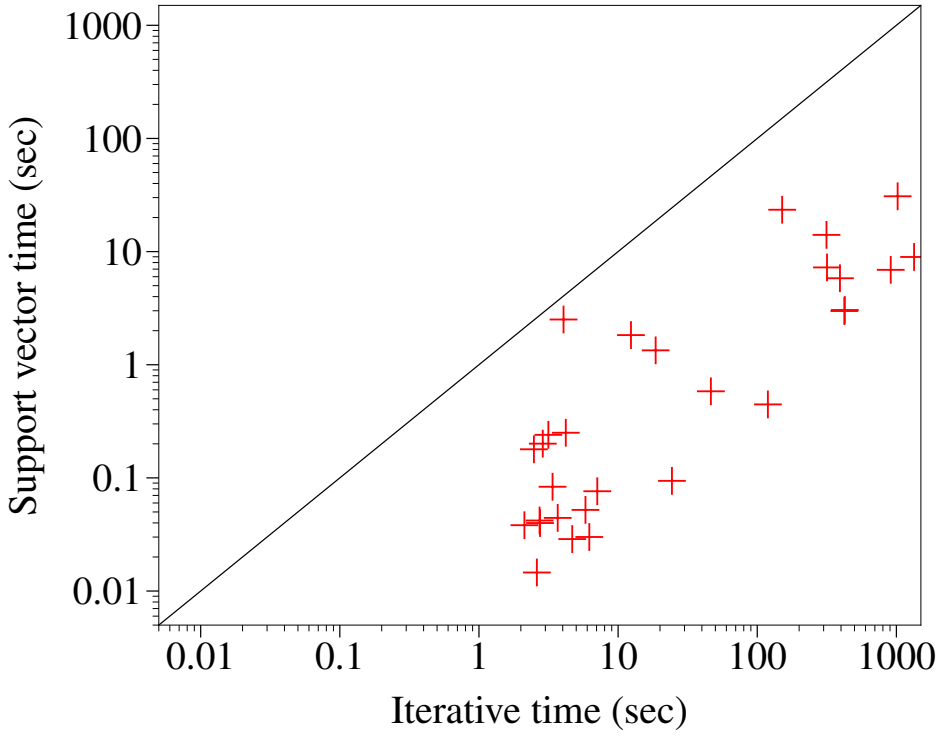




G. Cabodi, P. Camurati, and S. Quer, "A graph-labeling approach for efficient cone-of-influence computation in model-checking problems with multiple properties," *Software: Practice and Experience*, vol. 46, no. 4, pp. 493–511, 2016.

Support Vector Computation

- Several optimizations to improve time/memory
  - Directed acyclic graph – SCCs → shorter bitvectors
  - Garbage collection → peak memory requirement



Several orders of magnitude faster!

G. Cabodi, P. Camurati, and S. Quer, "A graph-labeling approach for efficient cone-of-influence computation in model-checking problems with multiple properties," *Software: Practice and Experience*, vol. 46, no. 4, pp. 493–511, 2016.

Structural Grouping

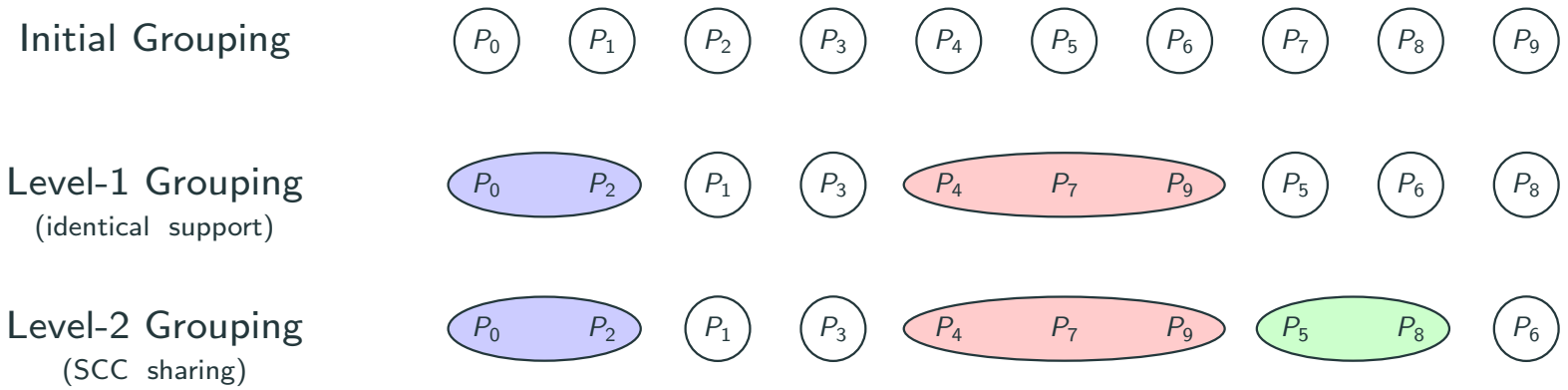
- Properties with 'similar' support bitvectors above threshold t
  - Classical clustering – very slow, at least $O(n^2)$
- Three-level approximate clustering (near-linear runtime)

Structural Grouping

- Properties with 'similar' support bitvectors above threshold t
  - Classical clustering – very slow, at least $O(n^2)$
- Three-level approximate clustering (near-linear runtime)

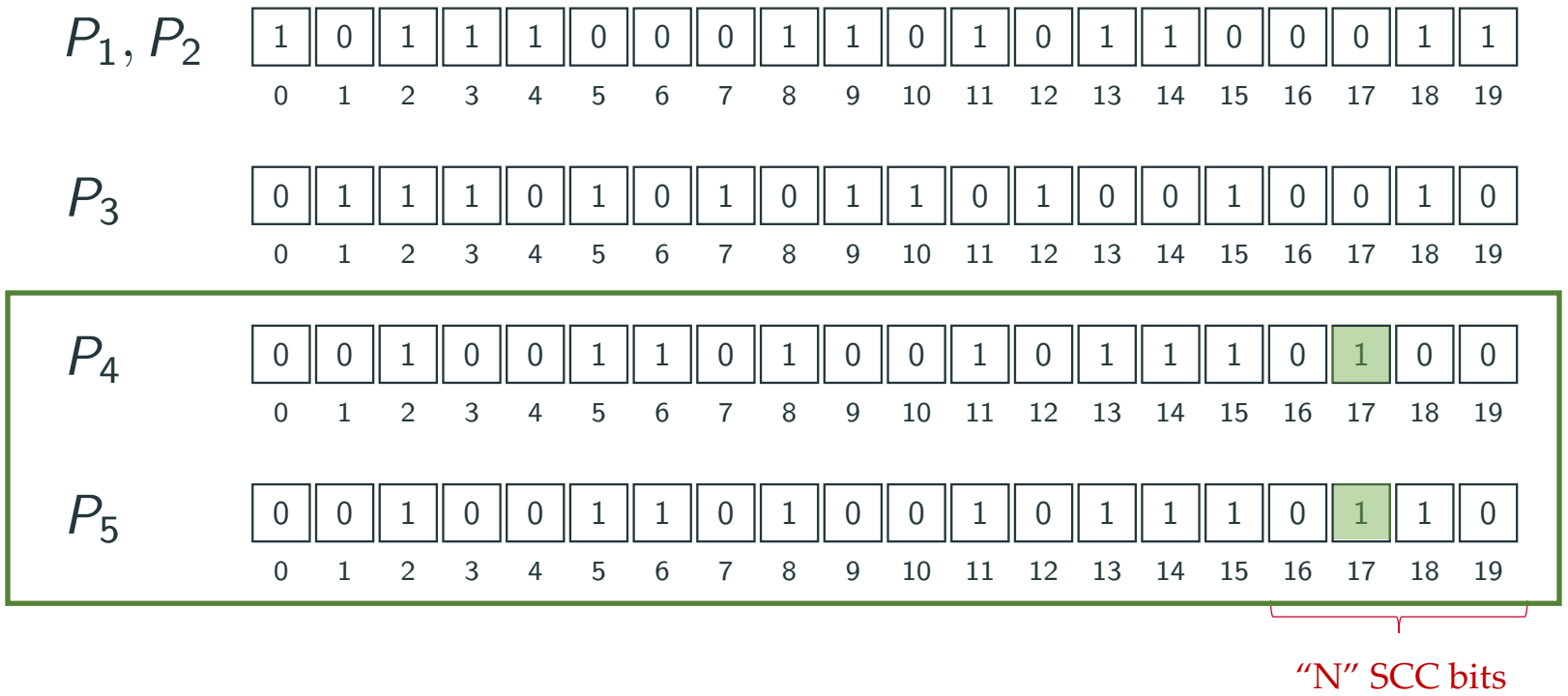
Structural Grouping

- Properties with 'similar' support bitvectors above threshold t
  - Classical clustering – very slow, at least $O(n^2)$
- Three-level approximate clustering (near-linear runtime)



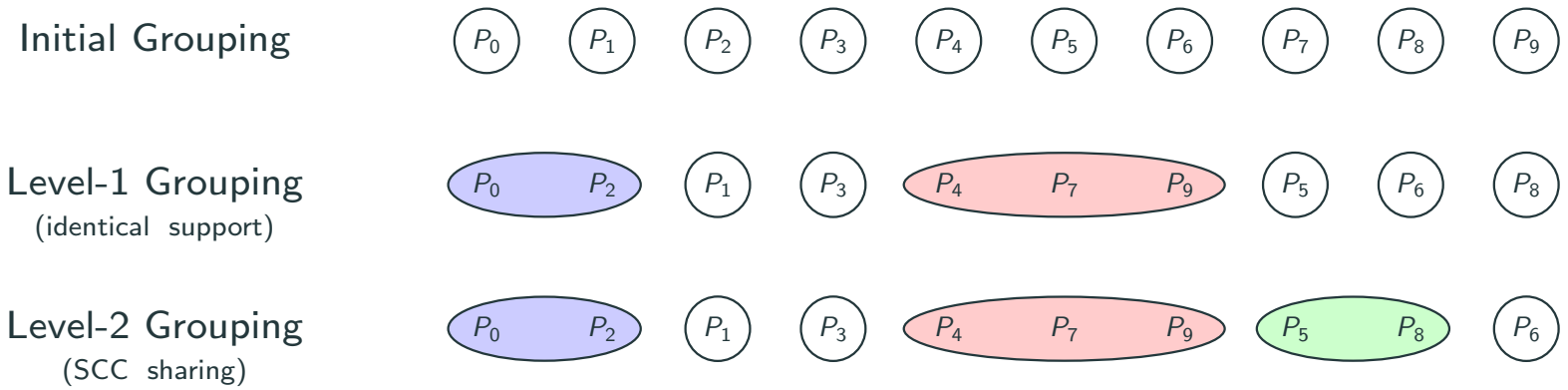
Level 2 – SCC Sharing

- Several designs contain large SCCs in cone-of-influence
- Every SCC has a weight – number of registers in SCC
- Group properties that share large SCCs – at least weight *t*



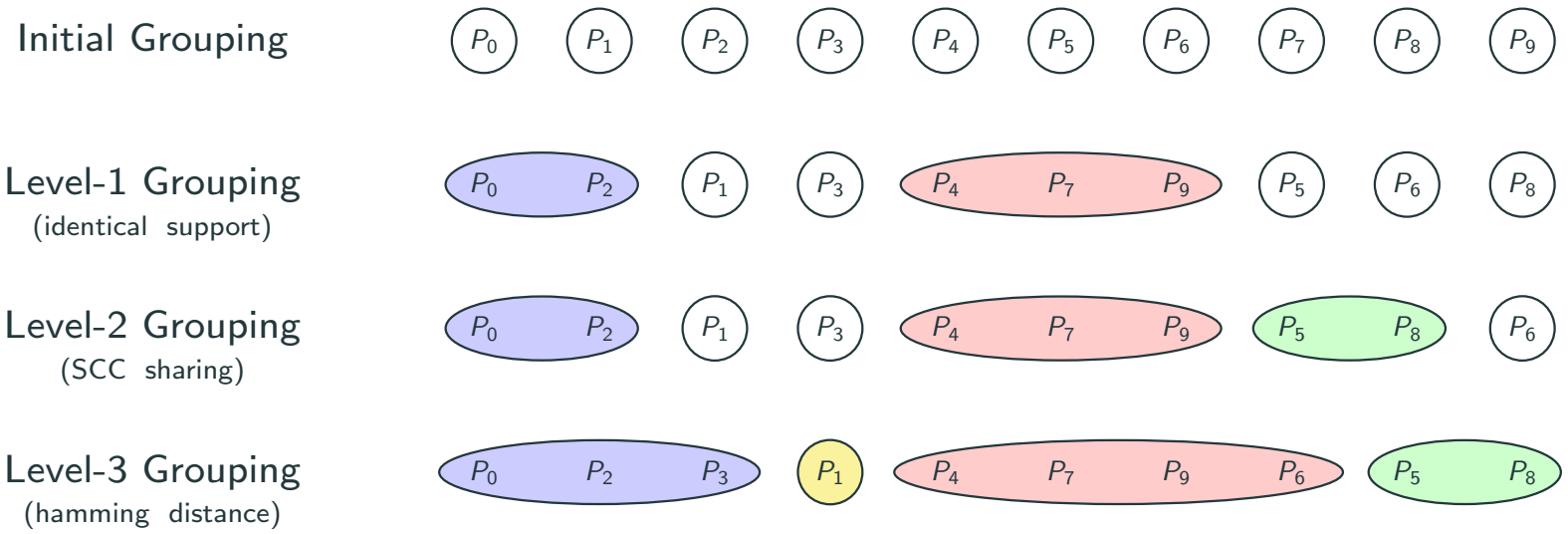
Structural Grouping

- Properties with 'similar' support bitvectors above threshold t
  - Classical clustering – very slow, at least $O(n^2)$
- Three-level approximate clustering (near-linear runtime)



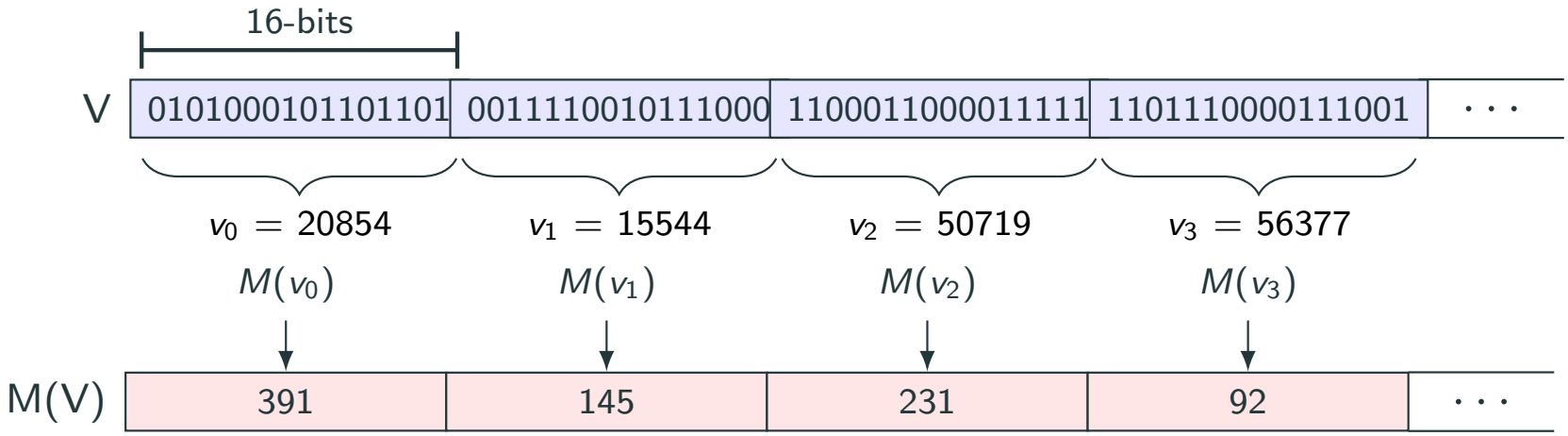
Structural Grouping

- Properties with 'similar' support bitvectors above threshold t
  - Classical clustering – very slow, at least $O(n^2)$
- Three-level approximate clustering (near-linear runtime)



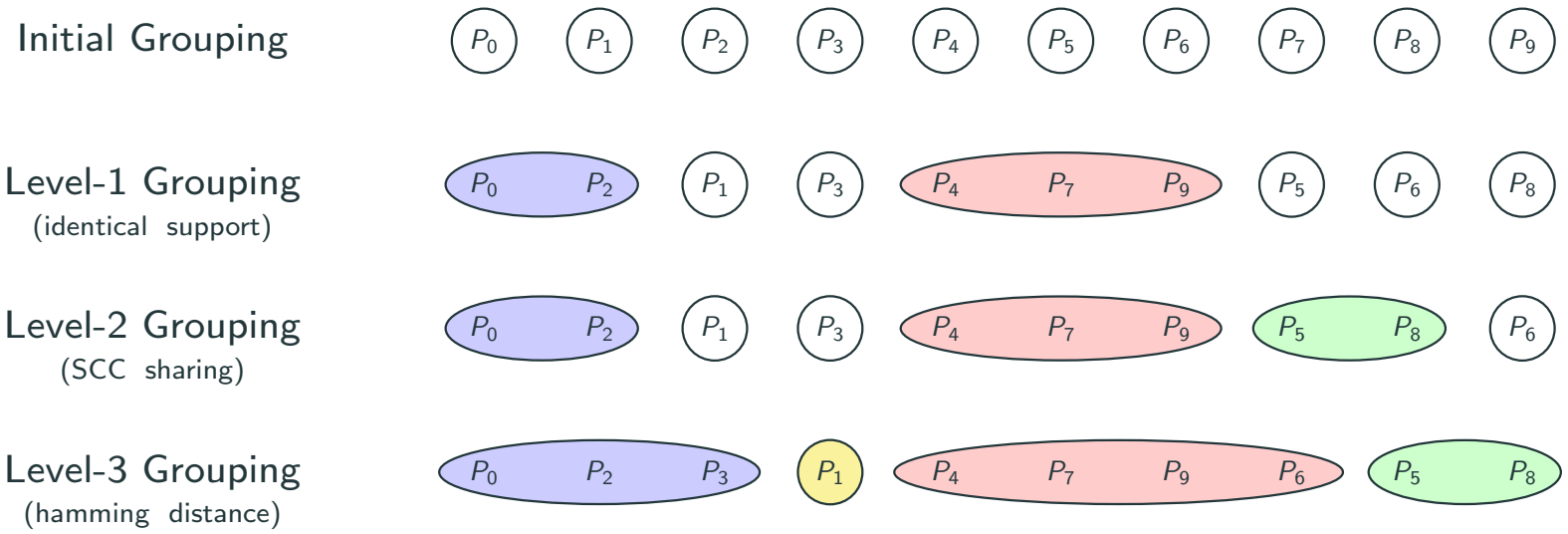
Level 3 – Hamming Distance

- Exact Hamming distance calculation is slow, $O(n^2)$
- Generate *normalized* support bitvectors
  - Map generated offline or on-the-fly, < 1sec
- Group properties with identical mapped bitvectors



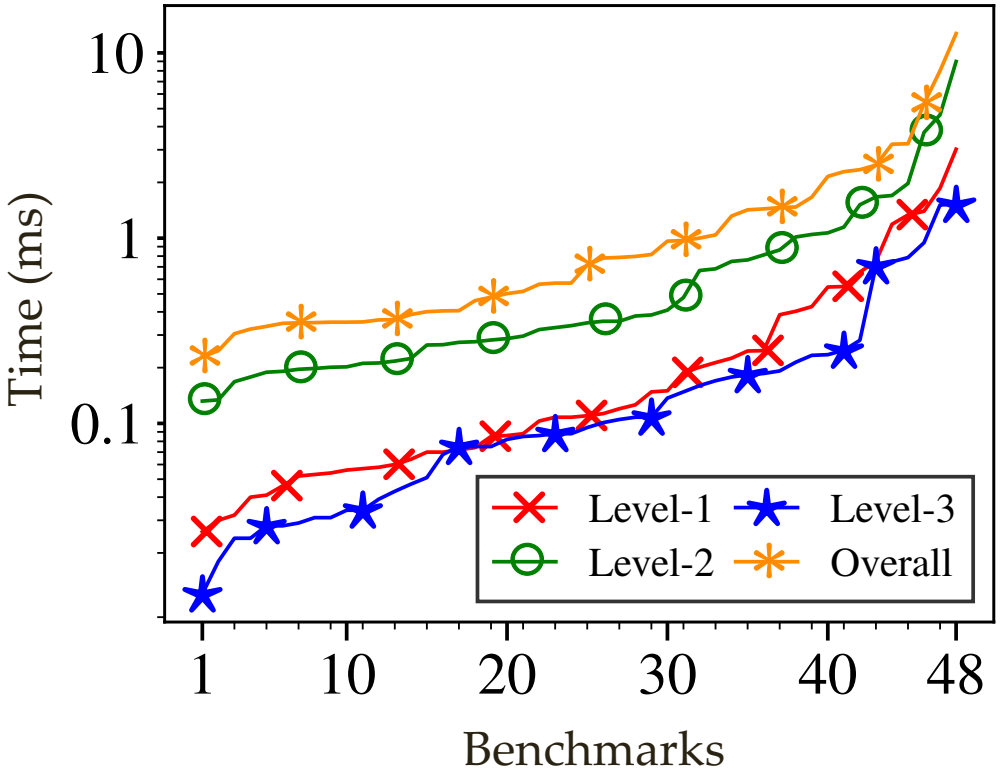
Structural Grouping

- Properties with 'similar' support bitvectors above threshold *t*
  - Classical clustering – very slow, at least $O(n^2)$
- Three-level approximate clustering (near-linear runtime)
- Proof: affinity >= *3*t - 2*
- Properties in a group are checked concurrently; groups in parallel



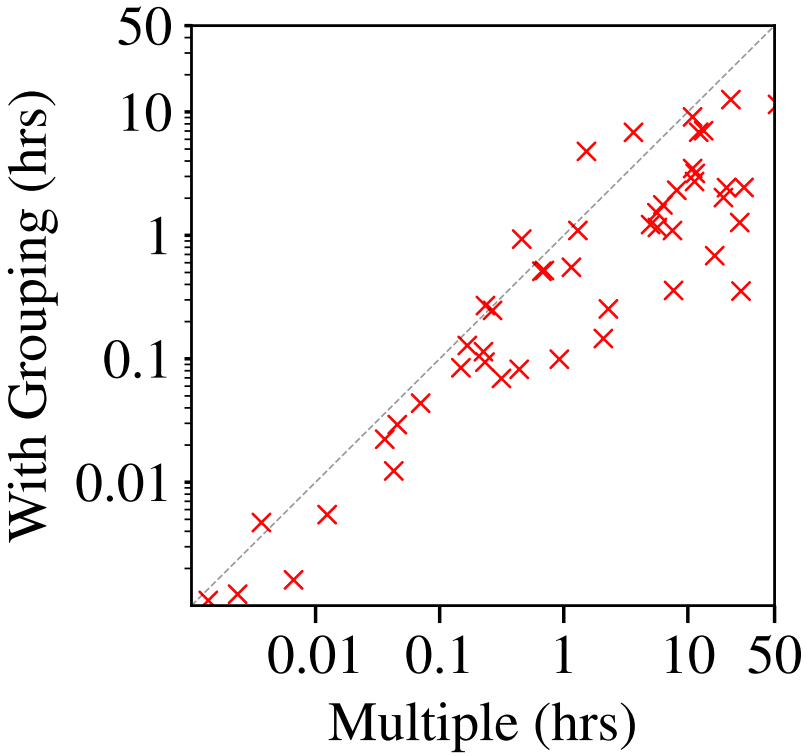
Grouping Time

- Grouping takes <10ms for the largest benchmarks (HWMCC)
  - Simplified by logic synthesis; hard properties only
  - 100 – 2,500 properties in a benchmark



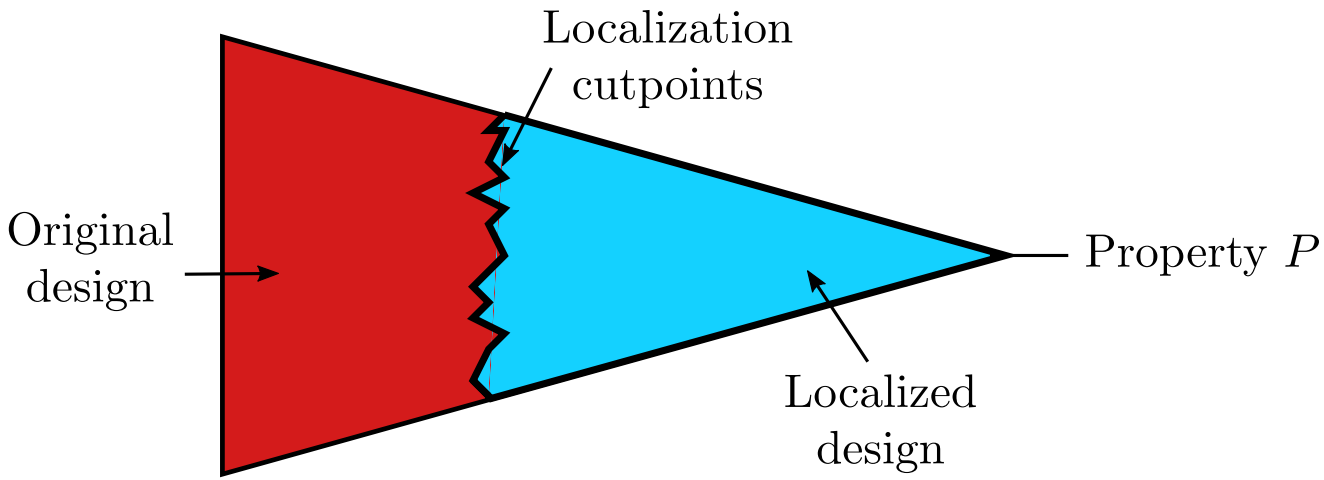
End-to-End Speedup

- Engine portfolio – BMC, IC3, and Localization
  - BMC and IC3 can process multiple properties, LOC concurrently



Median 4.3x speedup

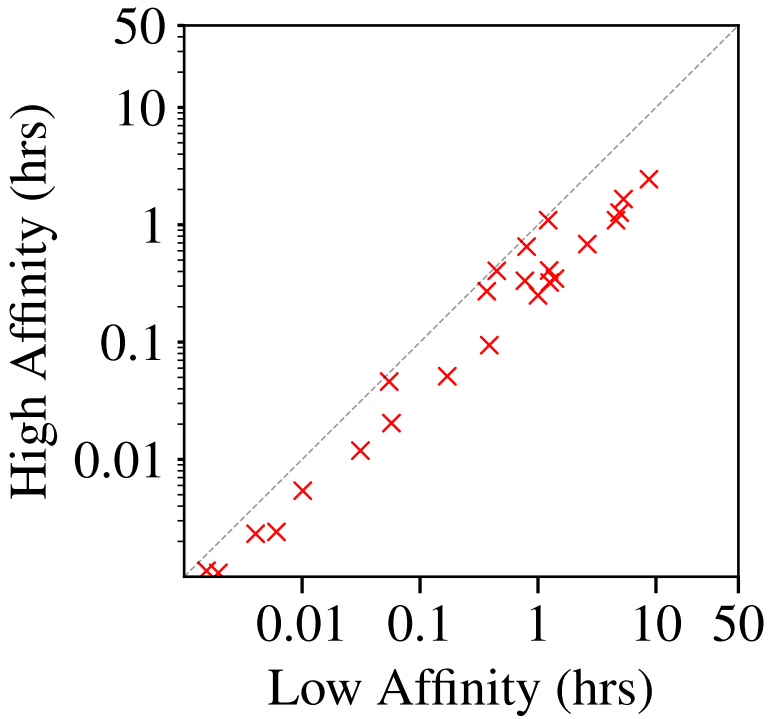
Impact on Localization Abstraction



- Technique to remove irrelevant logic
  - Iterative method, repeated *cutpointing* and *refinement*
- Concurrent localization of low-affinity properties
  - Large localized designs, disjoint logic subsets, slow proofs
- Our procedure ensures high-affinity property localization
  - Small localized designs, faster proofs

Impact on Localization Abstraction

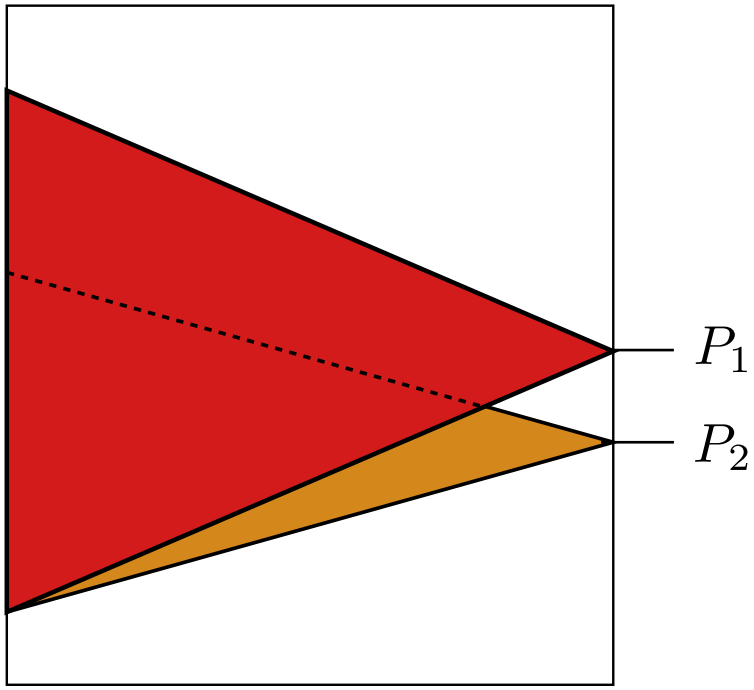
- Low-affinity groups – sort then partition
- First efficient multi-property localization solution!



Median 2.5x speedup

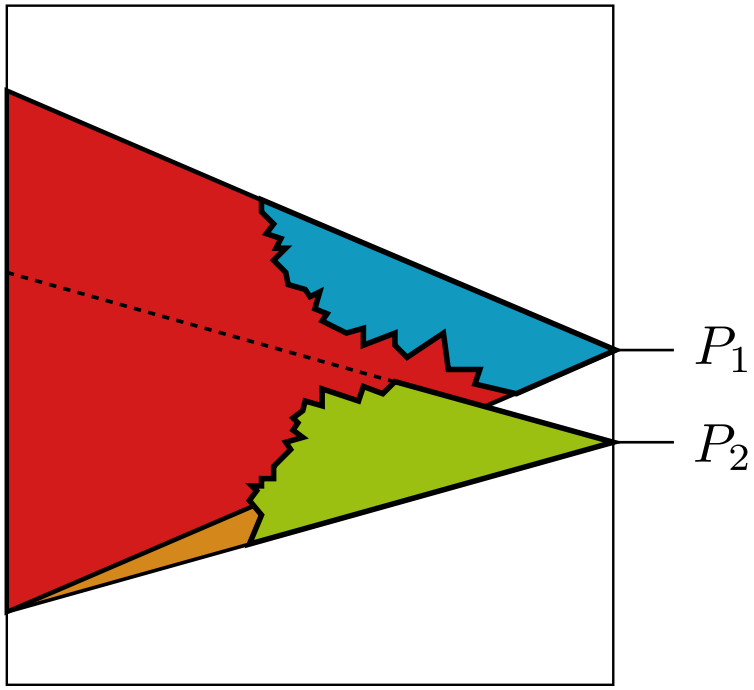
Structural Grouping

- Structurally-similar properties may have different semantics
  - Subset of design logic in cone-of-influence



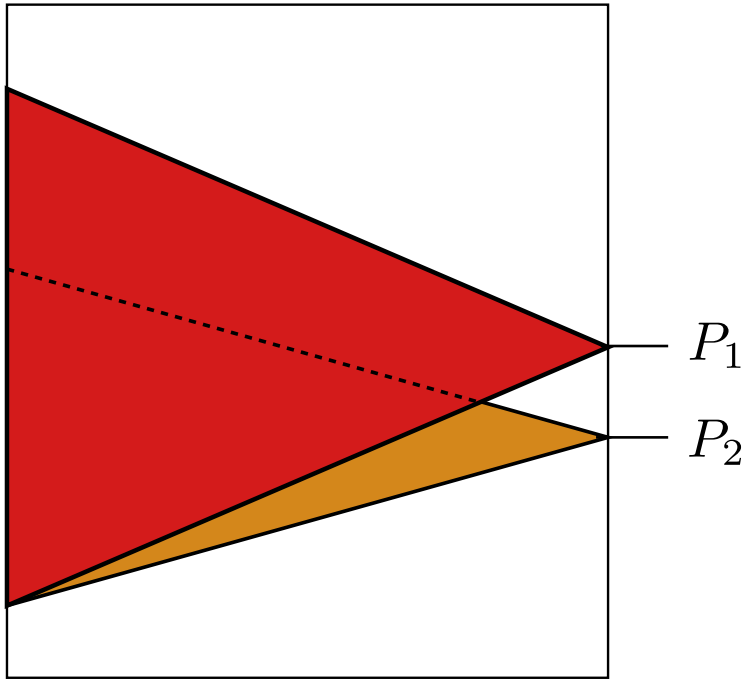
Structural Grouping

- Structurally-similar properties may have different semantics
  - Subset of design logic in cone-of-influence, mix of hittable/unhittable
- Learn semantic information via localization abstraction



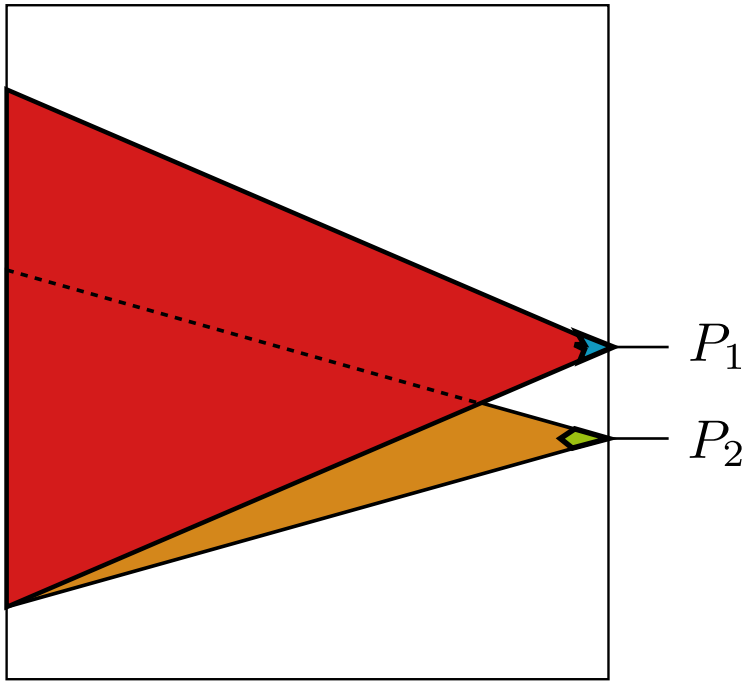
Semantic Partitioning

- Concurrently localize high-affinity property group



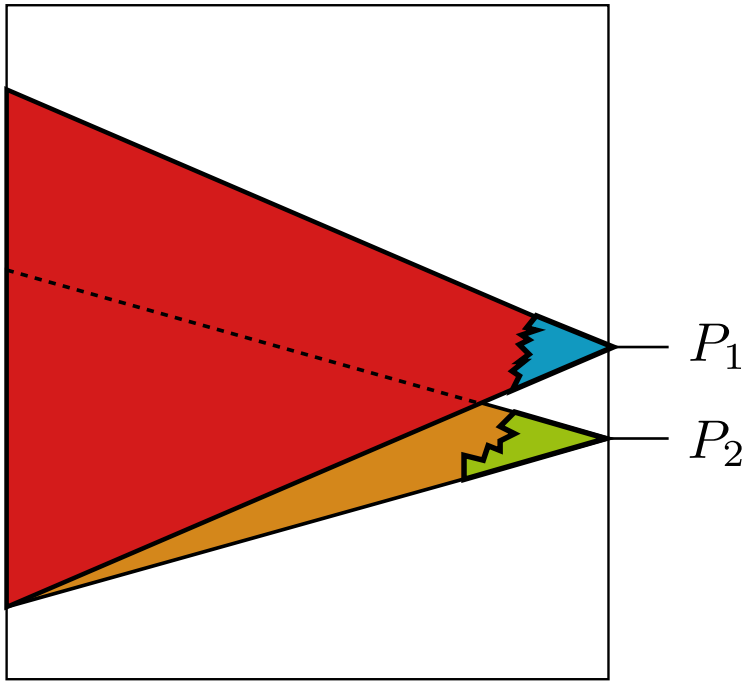
Semantic Partitioning

- Concurrently localize high-affinity property group
- Repeated BMC steps to generate localized design



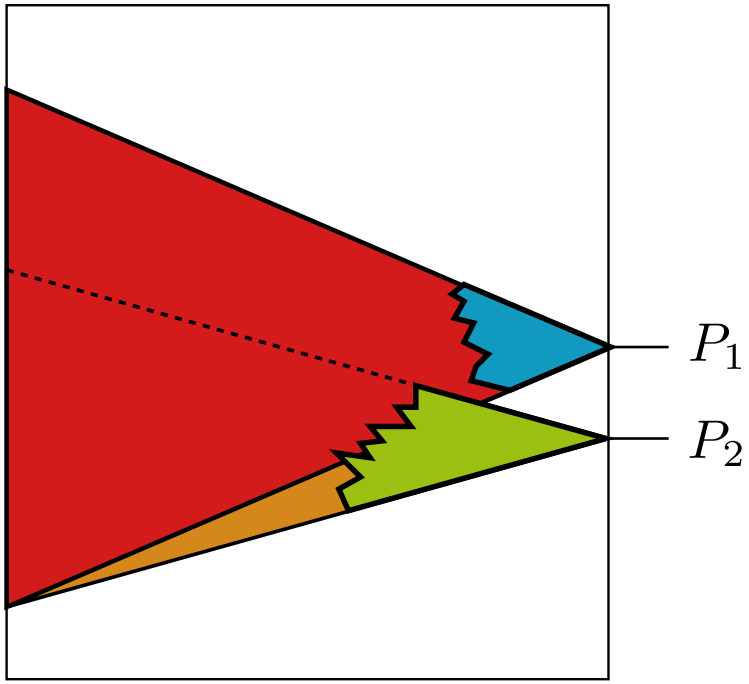
Semantic Partitioning

- Concurrently localize high-affinity property group
- Repeated BMC steps to generate localized design



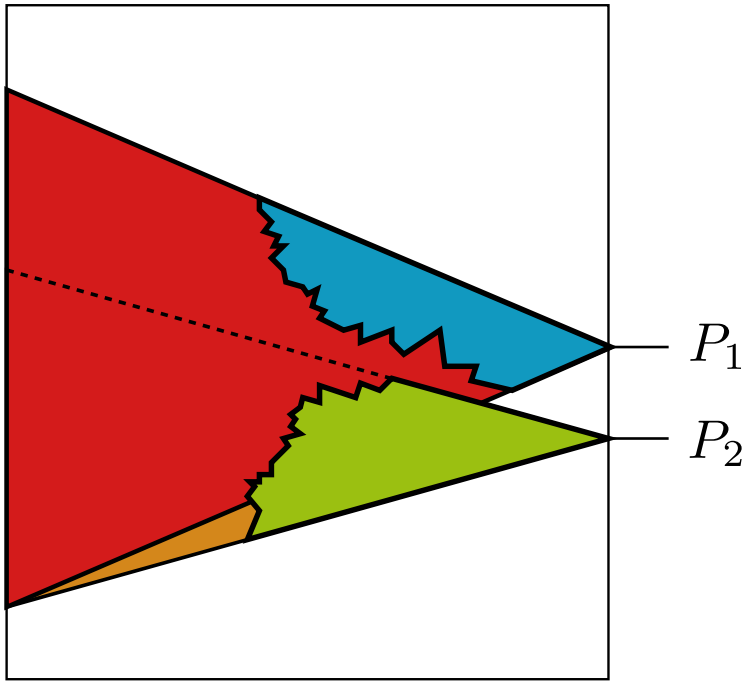
Semantic Partitioning

- Concurrently localize high-affinity property group
- Repeated BMC steps to generate localized design



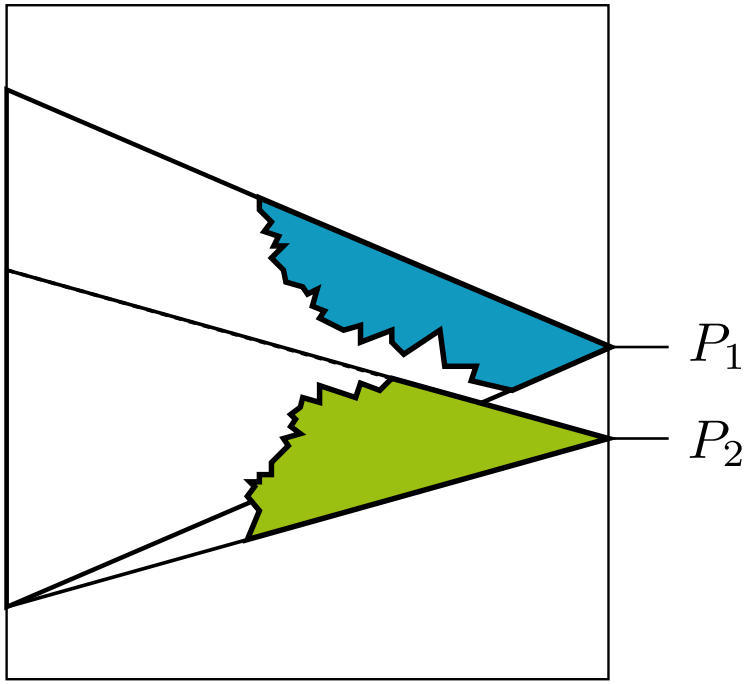
Semantic Partitioning

- Concurrently localize high-affinity property group
- Repeated BMC steps to generate localized design
- Attempt partitioning after N consecutive steps with no refinement



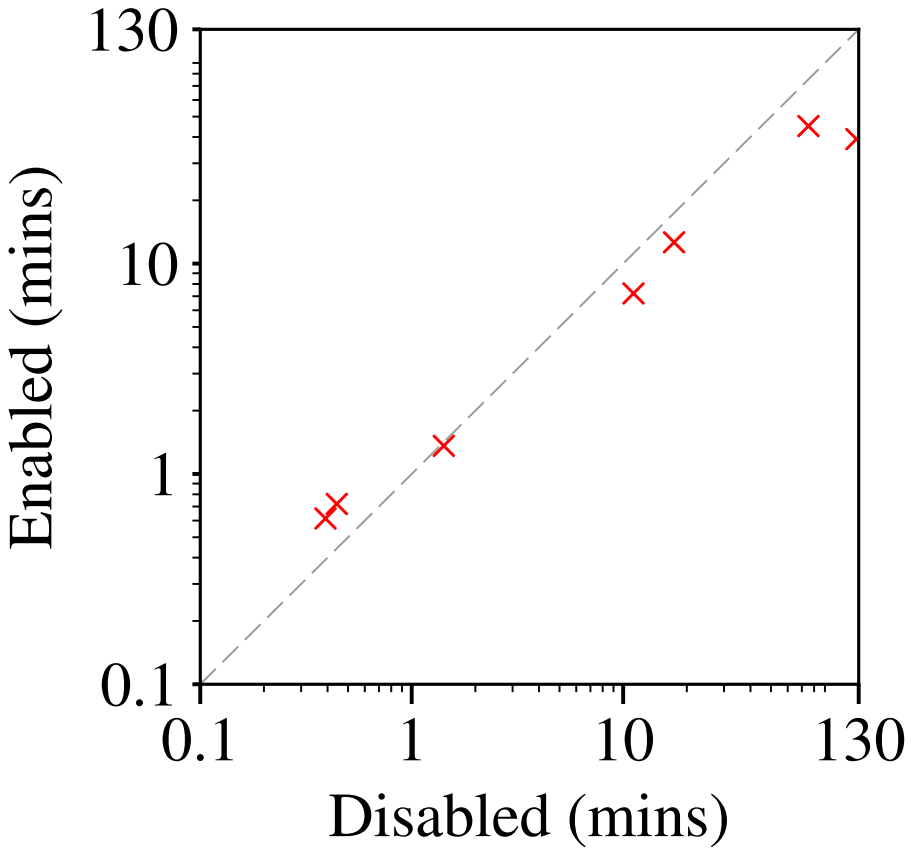
Semantic Partitioning

- Concurrently localize high-affinity property group
- Repeated BMC steps to generate localized design
- Attempt partitioning after N consecutive steps with no refinement
- Structural grouping procedure w.r.t localized design



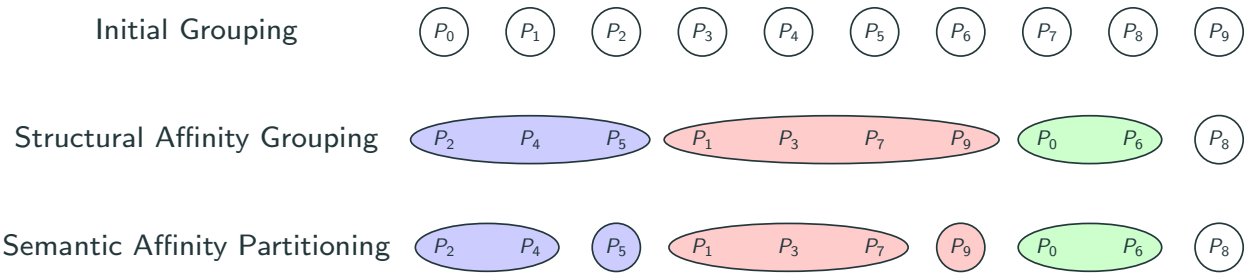
Impact on Localization Abstraction

- Selected benchmarks; some property groups solved by localization
  - Single proof run; no spurious counterexamples



Summary

- **Fast and online** algorithm to group "high-affinity" properties



- Three leveled grouping; identical, SCC sharing, and Hamming distance
- **Substantial speedup, minimal resource** overhead
- Yields groups with **provable affinity bounds**; might err (tradeoff)
- First approach to **optimize multi-property localization**
- Ongoing and future work
  - Sequential equivalence checking (SEC) – each equivalence point is a property
  - Structural vs. semantic – hard to know without consuming verification resource

Thank you!

http://temporallogic.org/research/FMCAD19/