Ultimate TestGen: Test-Case Generation with Automata-based Software Model Checking

Max Barth[1] and Daniel Dietsch[2] and Matthias Heizmann[2] and Marie-Christine Jakobs[1]

LMU Munich, Munich, Germany[1]
University of Freiburg, Freiburg, Germany[2]

Test-Comp 2024

Automata-Based Software Model Check

- At SV-Comp 23 and 24 the highest overall score had the model checker UAutomizer.
- UAutomizer uses automata-based model checking.
- Ultimate TestGen is the first test-case generator with automata based model checking.

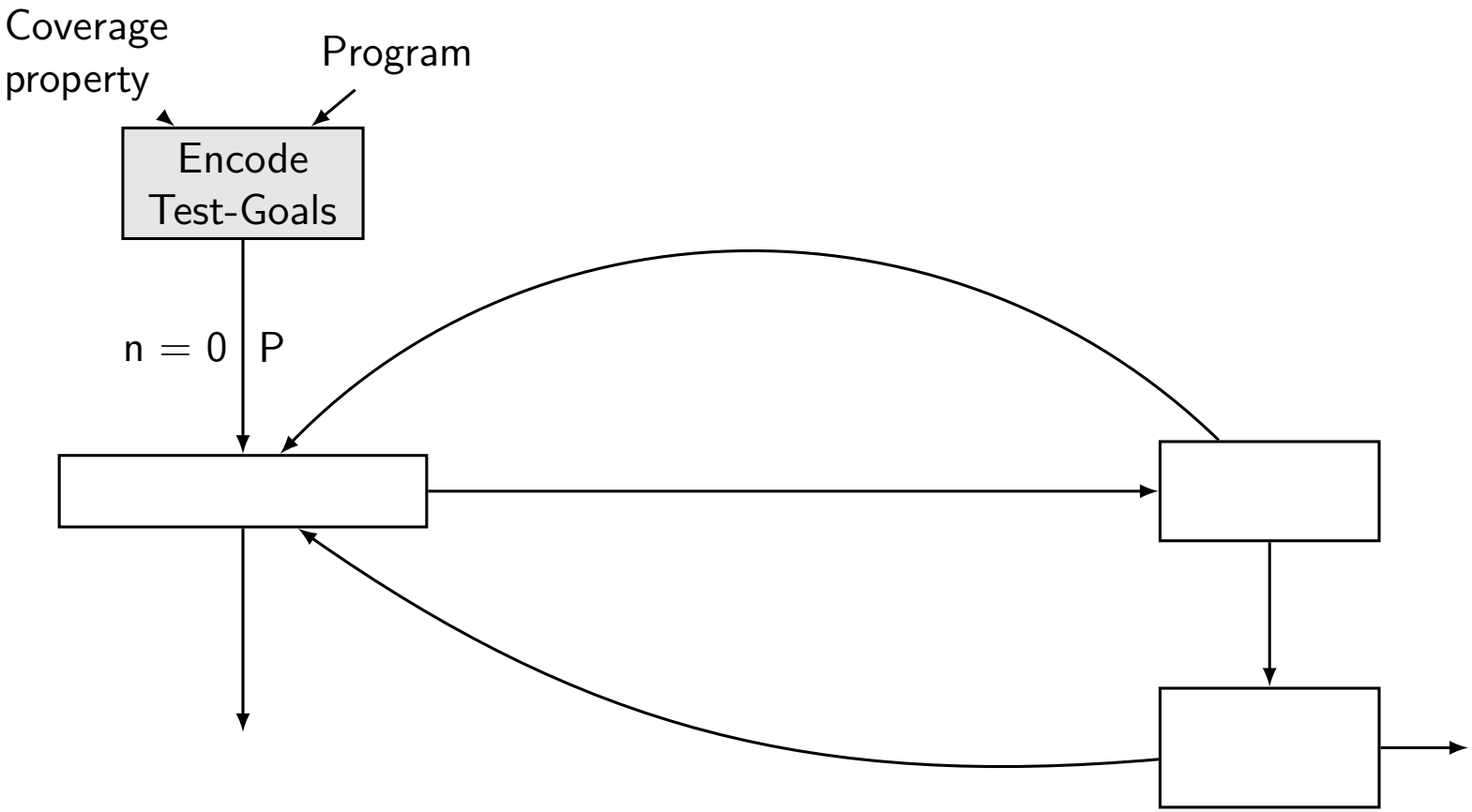
Ultimate TestGen

- Implemented in the Ultimate framework.
  `github.com/ultimate-pa/ultimate`
- A purely model checking based test case generation.

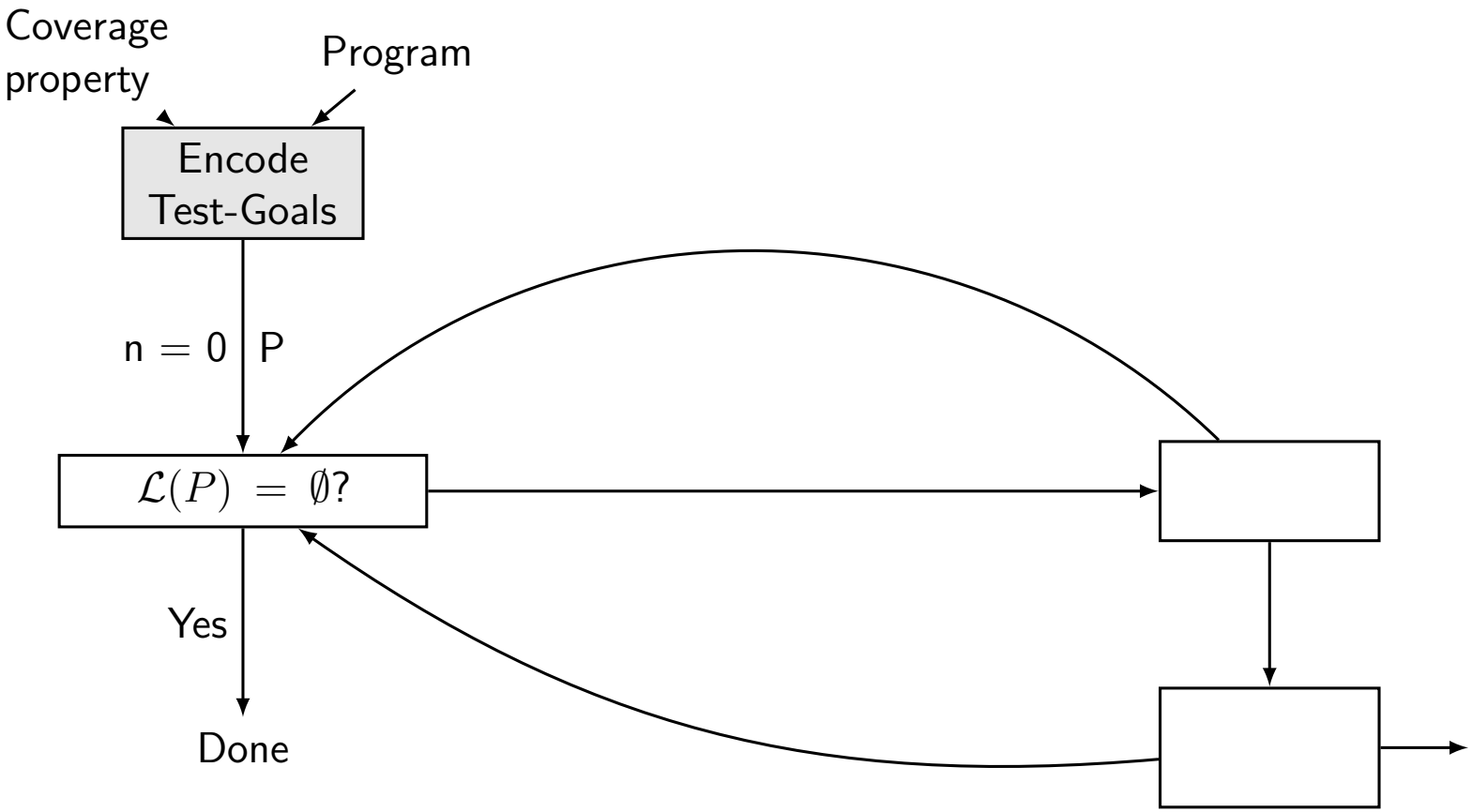
Automata-Based Model Checking

- A program is represented by a Program Automaton.
- Test goals are accepting states.
- The language is the set of all program paths to test goals.
- We call an accepted word error trace.

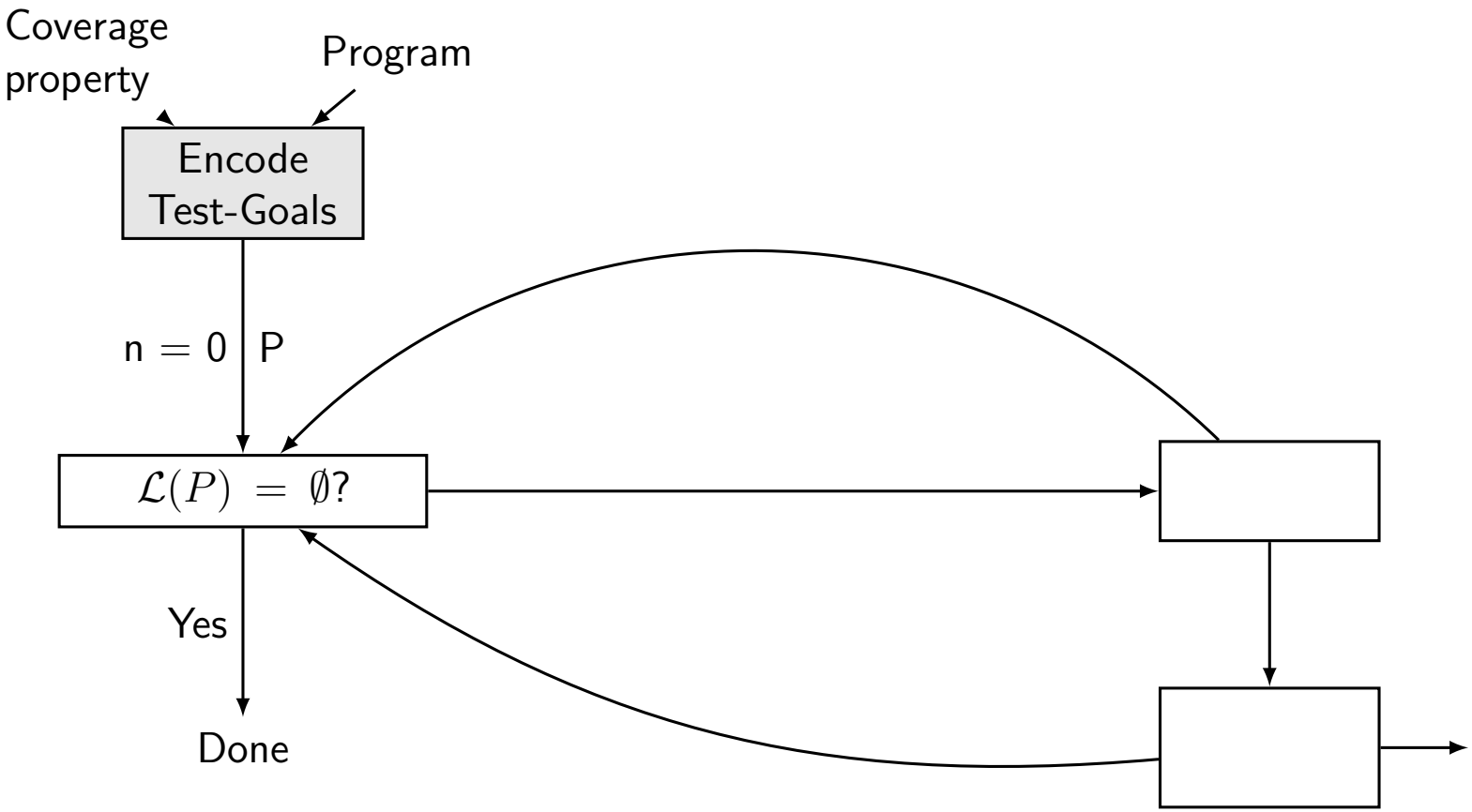
Automata-Based CEGAR for Test-Case Generation



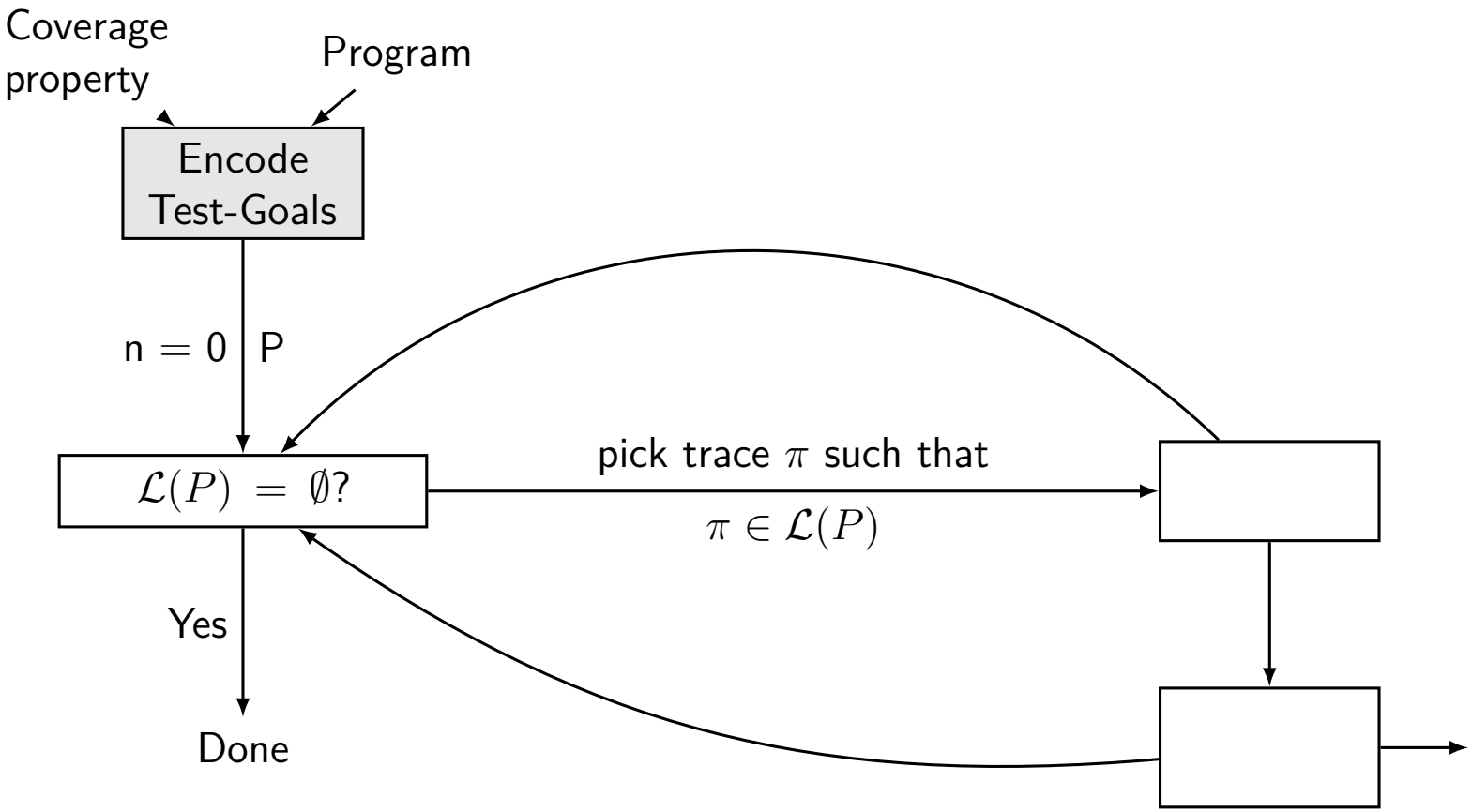
Encode test goals as accepting states in the Program Automaton.



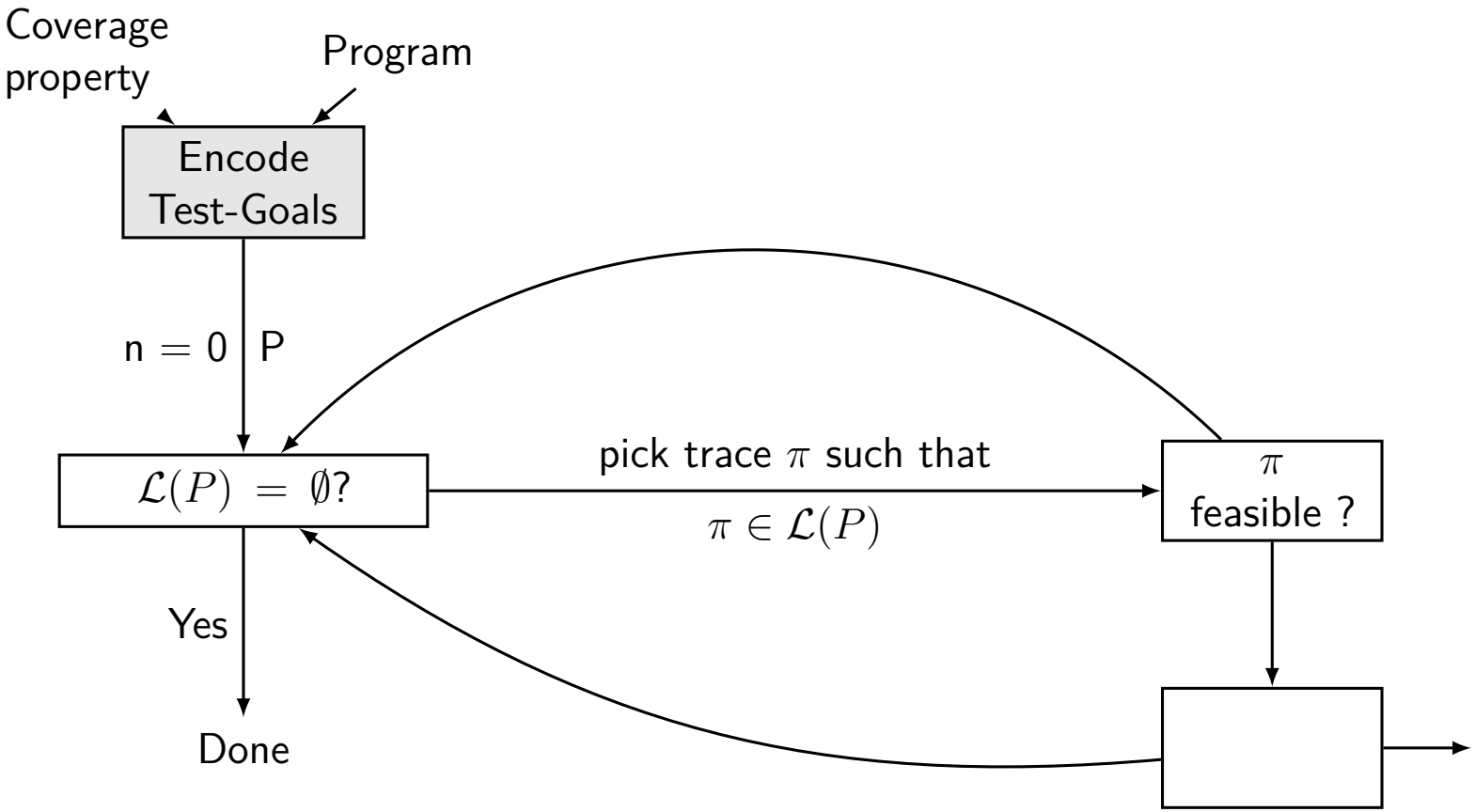
$\mathcal{L}(P)$ contains every error trace to a test goal.



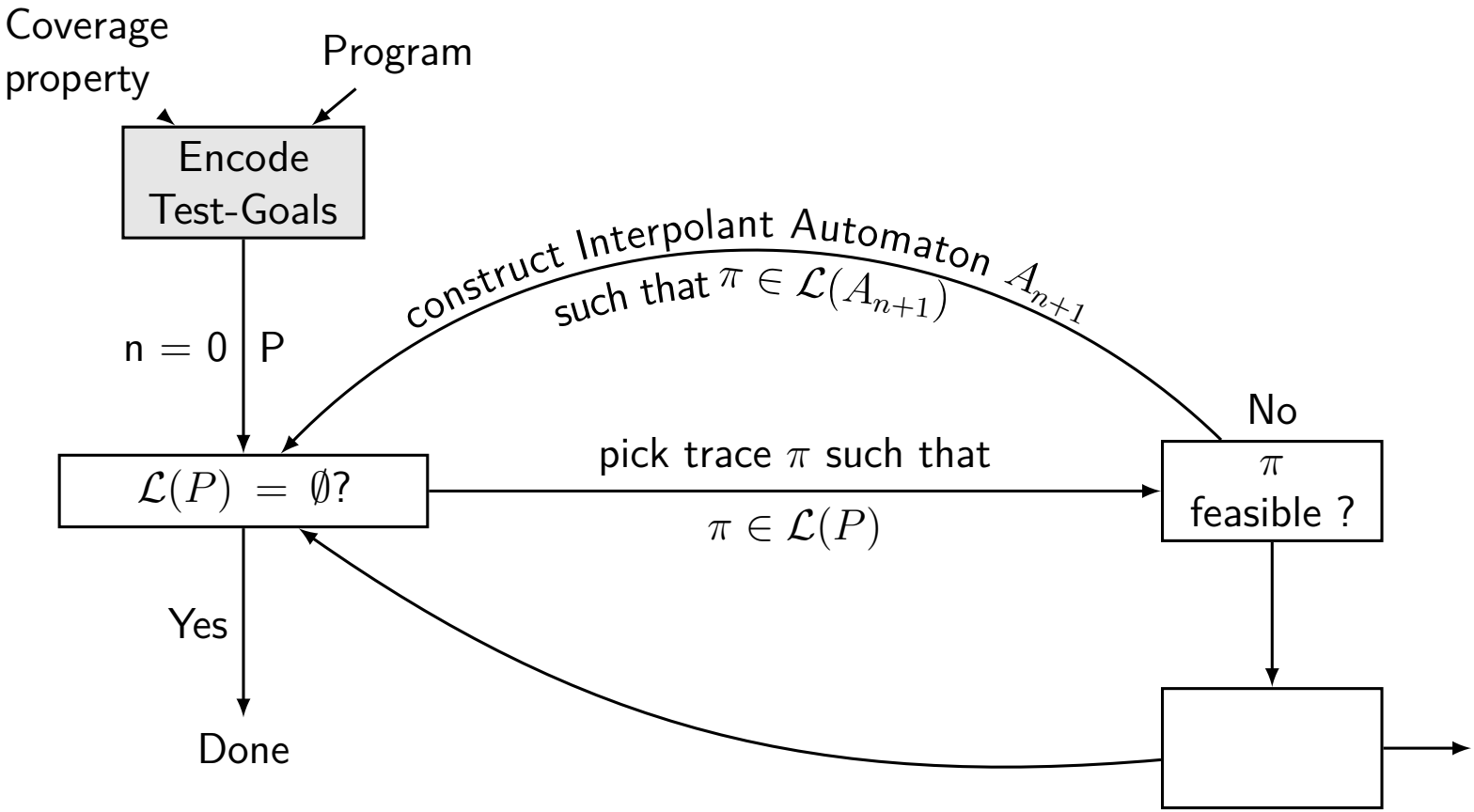
If $\mathcal{L}(P) = \emptyset$ there exists no path to a test goal in P.



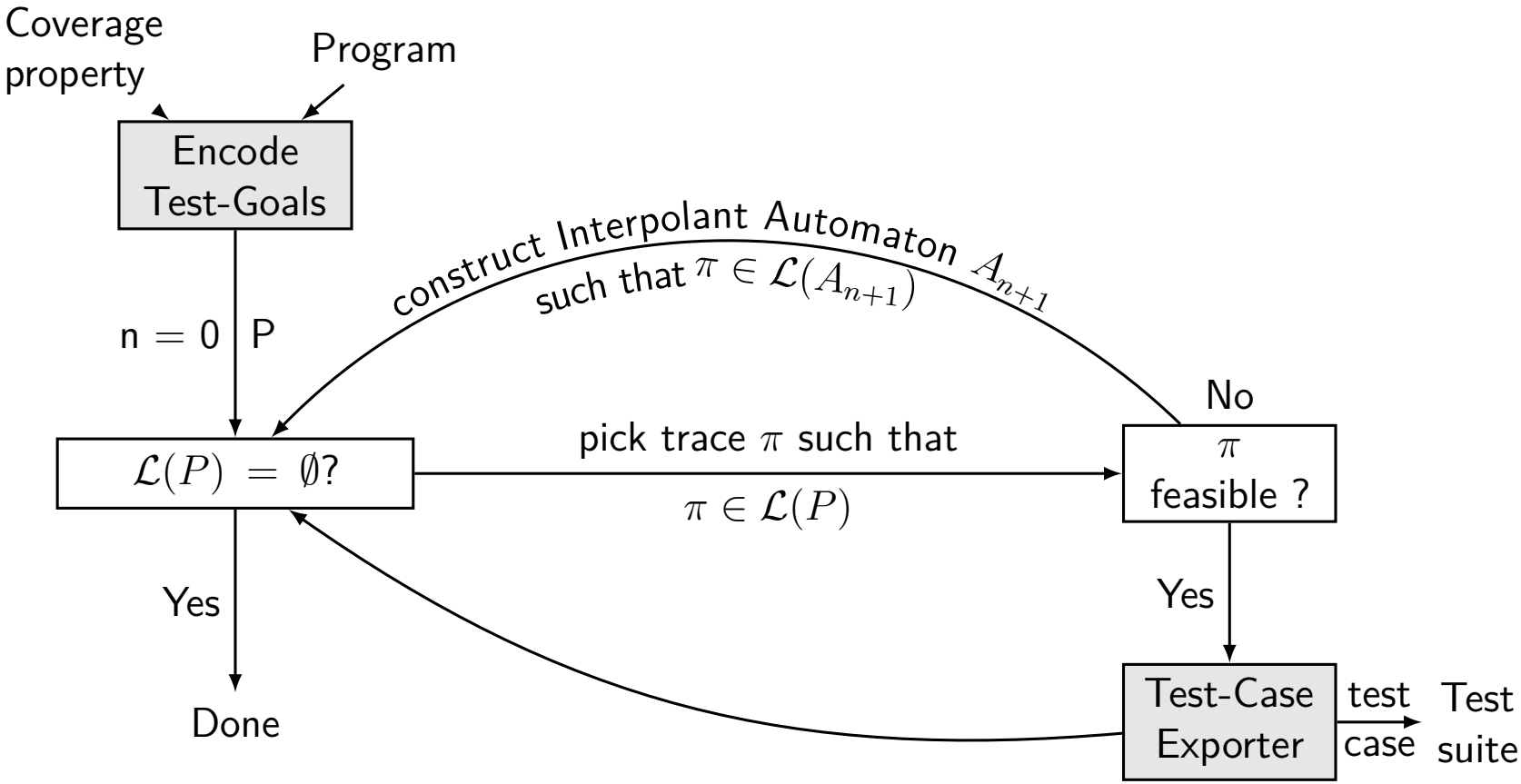
Pick an error trace $\pi \in \mathcal{L}(P)$ to a test goal.



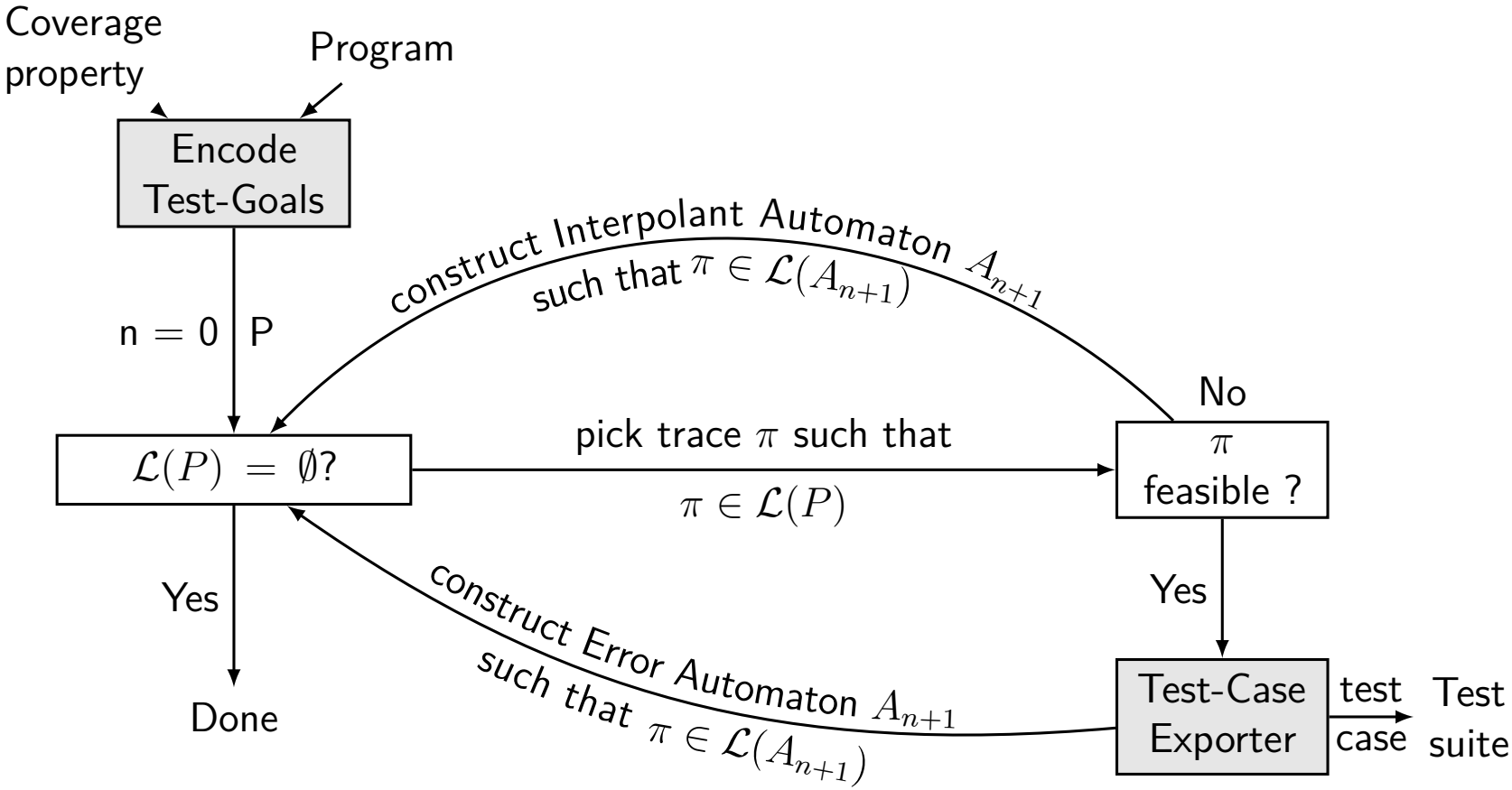
π is feasible iff its SSA-based formula encoding is satisfiable.



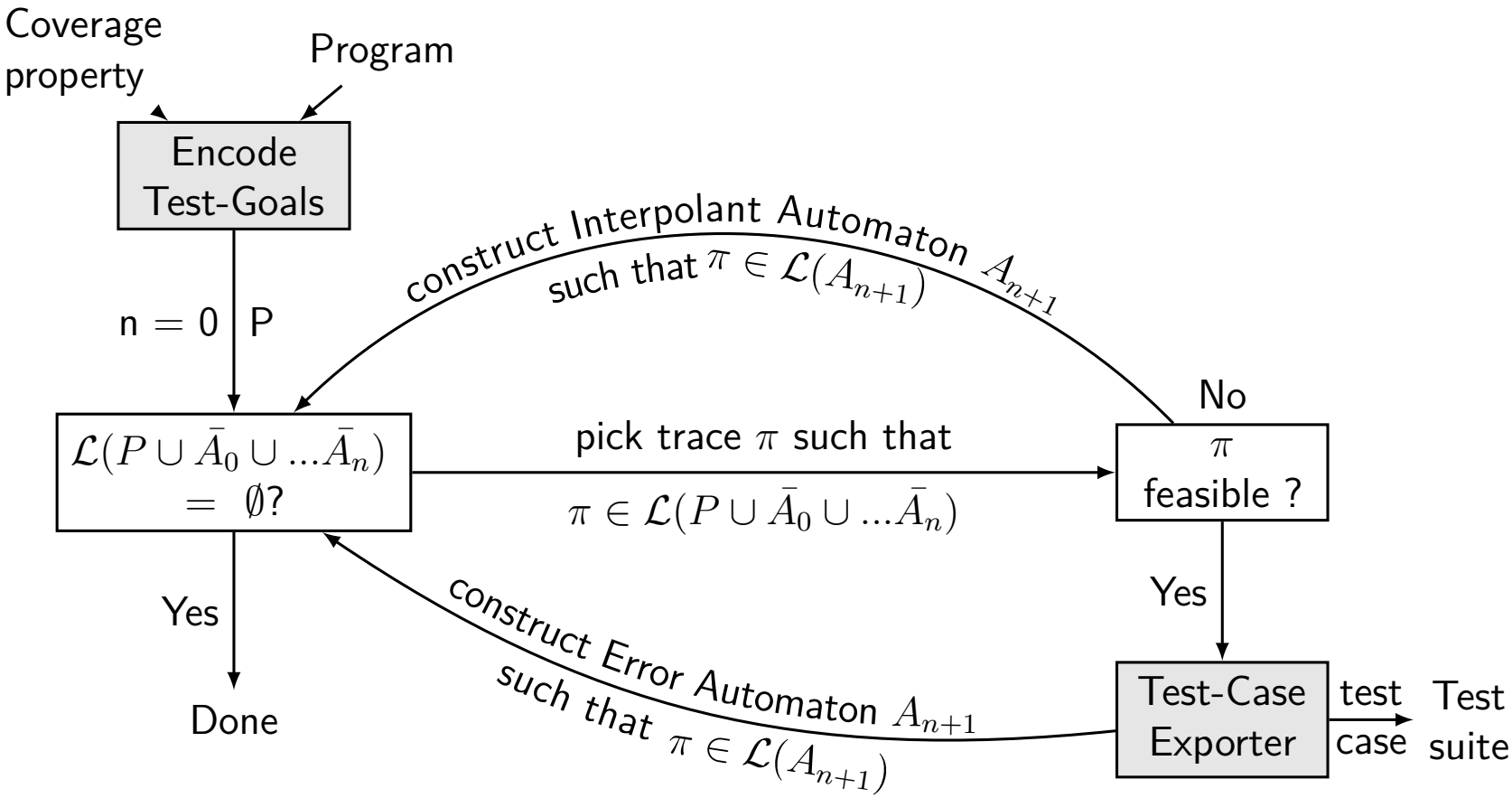
An Interpolant Automaton accepts π and error traces infeasible for similar reason.



Construct a test case from the feasibility proof.



An Error Automaton accepts every error trace to a specific test goal.



Pick a new error trace $\pi \in \mathcal{L}(P)$ and $\pi \notin \mathcal{L}(A_0 \cup ... \cup A_n)$.

Test-Comp24 Results

Ranking of Ultimate TestGen:

- 10th. place in the Cover-Error category
- 8th. place in the Cover-Branches category
- 10th. place overall

Weaknesses

- Checking the feasibility of an error trace can be very expensive.
- Proving a test goal is not reachable is often prioritized over achieving high coverage.

Advantages

- If Ultimate TestGen terminates, every test goal not covered is guaranteed not reachable.
- Ultimate TestGen creates exactly one test case for each test goal.

Test-Comp24 Results

- We had bugs in the test-case exporter, costing us 320 raw coverage.
- TestCov fails to calculate coverage for some of our test-cases.
  - The run allocates to much memory.
  - The run does not terminate.

Conclusion

- Ultimate TestGen is a first time participant.
- We purely do automata-based model checking.
- Automata represent sets of error traces.
- CEGAR loop over the set of all error traces.
- We create test cases from feasibility proofs (counterexamples).