Property Directed Reachability in Ultimate

Bachelor Thesis Presentation

By **Jonas Werner**
Advised by **Dr. Daniel Dietsch**

24.09.2018

Introduction: Motivation

➢ Property Directed Reachability (PDR) was first devised as hardware verification technique in 2010 by Aaron Bradley[1]

1: Aaron R. Bradley. Sat-based model checking without unrolling. In *VMCAI*, volume 6538 of *Lecture Notes in Computer Science*, pages 70–87. Springer, 2011.

Introduction: Motivation

➢ Property Directed Reachability (PDR) was first devised as hardware verification technique in 2010 by Aaron Bradley[1]

➔ Surprisingly won 3rd place at CAV 2010 hardware checking competition[2]

1: Aaron R. Bradley. Sat-based model checking without unrolling. In *VMCAI*, volume 6538 of *Lecture Notes in Computer Science*, pages 70–87. Springer, 2011.

2: Hwmcc10 results. http://fmv.jku.at/hwmcc10/results.html. Accessed: 2018-07-20

23.09.2018

Introduction: Motivation

➢ Property Directed Reachability (PDR) was first devised as hardware verification technique in 2010 by Aaron Bradley[1]

➔ Surprisingly won 3rd place at CAV 2010 hardware checking competition[2]

"This new method appears to be the most important contribution to bit-level formal verification in almost a decade" [3]

1: Aaron R. Bradley. Sat-based model checking without unrolling. In *VMCAI*, volume 6538 of *Lecture Notes in Computer Science*, pages 70–87. Springer, 2011.

2: Hwmcc10 results. http://fmv.jku.at/hwmcc10/results.html. Accessed: 2018-07-20

3: Niklas Een, Alan Mishchenko, and Robert Brayton. 2011. Efficient implementation of property directed reachability. In Proceedings of the International Conference on Formal Methods in Computer-Aided Design (FMCAD '11). FMCAD Inc, Austin, TX, 125-134.

Introduction: Motivation

➤ Property Directed Reachability (PDR) was first devised as hardware verification technique in 2010 by Aaron Bradley[1]

    ➔ Surprisingly won 3rd place at CAV 2010 hardware checking competition[2]

> "This new method appears to be the most important contribution to bit-level formal verification in almost a decade" [3]

➤ Using PDR on software may have similar performance!

1: Aaron R. Bradley. Sat-based model checking without unrolling. In *VMCAI*, volume 6538 of *Lecture Notes in Computer Science*, pages 70–87. Springer, 2011.

2: Hwmcc10 results. http://fmv.jku.at/hwmcc10/results.html. Accessed: 2018-07-20

3: Niklas Een, Alan Mishchenko, and Robert Brayton. 2011. Efficient implementation of property directed reachability. In Proceedings of the International Conference on Formal Methods in Computer-Aided Design (FMCAD '11). FMCAD Inc, Austin, TX, 125-134.

Introduction: Motivation

➢ **Our Goals:**

- Use PDR on software in the verification framework Ultimate[1]
  - ➔ Combining Trace Abstraction and PDR
  - ➔ Comparison to existing techniques

1: Ultimate. https://ultimate.informatik.uni-freiburg.de. Accessed: 2018-07-20.

Overview

➢ **How does our PDR algorithm work?**

- Preliminaries

- Running Example

- Related Work

➢ **How do we use PDR in Ultimate?**

- Combination of Trace Abstraction and our PDR algorithm

- Implemented Improvements

Overview

➢ Evaluation:

- Comparison of Trace Abstraction using PDR and Trace Abstraction using Nested Interpolants

➢ What can be done in the future?

- Implementing more Improvements

PDR Algorithm: Preliminaries

Control flow graph (CFG) $A = (X, L, E, \ell_0, \ell_E)$ is a graph consisting of

- Finite set of first-order variables X

- Finite set of locations L

- Finite set of transitions $\mathsf{E} \subseteq L \times FO \times L$
  - ➔ FO is a quantifier free first-order logic formula over variables in X and $X' = \{x \in X \,|\, x' \in X'\}$

- Initial location $\ell_0 \in L$

- Error location $\ell_E \in L$

PDR Algorithm: Datastructures

Frame $F_{i,\ell}$:

- Represents a first-order formula
- ℓ is the corresponding location
- i is the corresponding iteration
 ➜ Each location has multiple assigned frames

Proof-Obligation (p, ℓ, i) :

- p is a first-order formula
- ℓ is the corresponding location
- i is the corresponding iteration
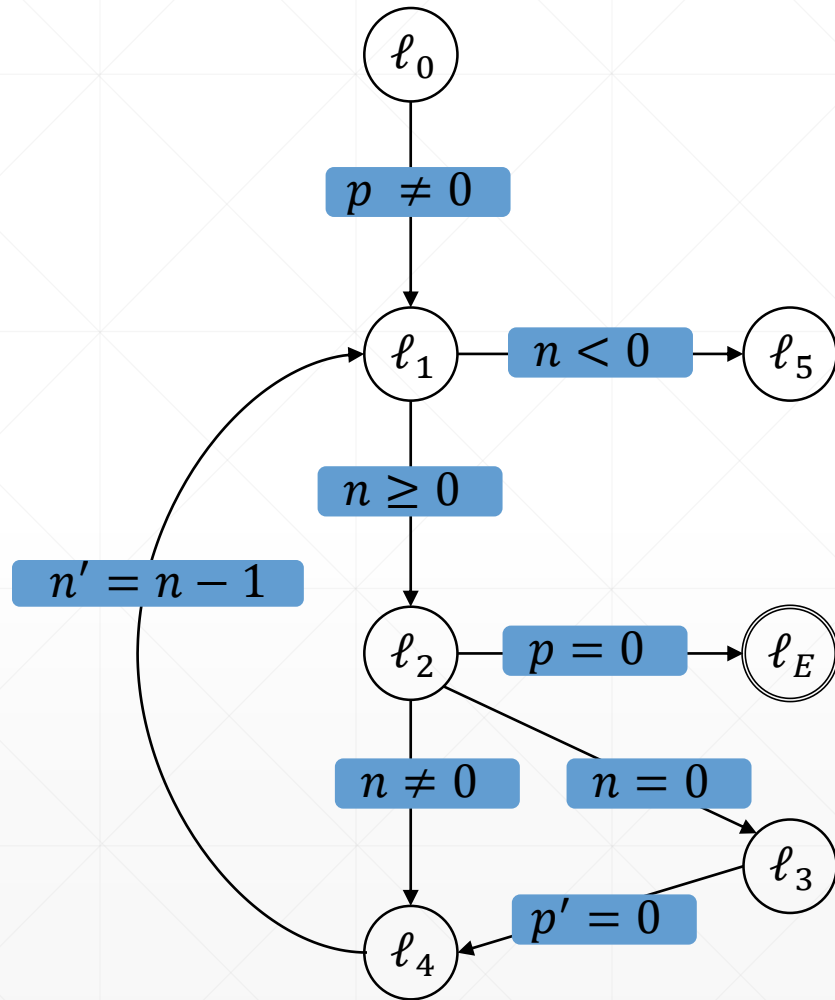➜ Need to be blocked

PDR Algorithm: Description

➤ Starts with checking for a **0-Counter-Example**

➤ **Global Initialization**

➤ Repeats three phases until termination:

1. **Next Iteration Initialization**

2. **Blocking-Phase**

3. **Propagation-Phase**
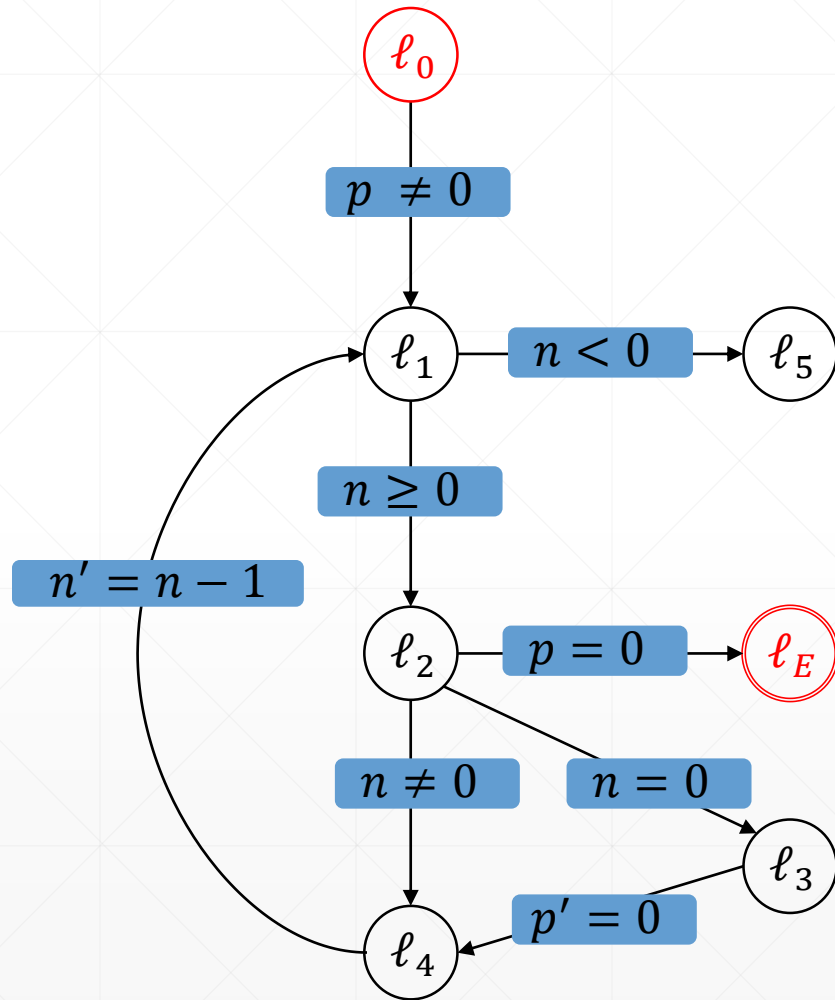
Example: Running Example
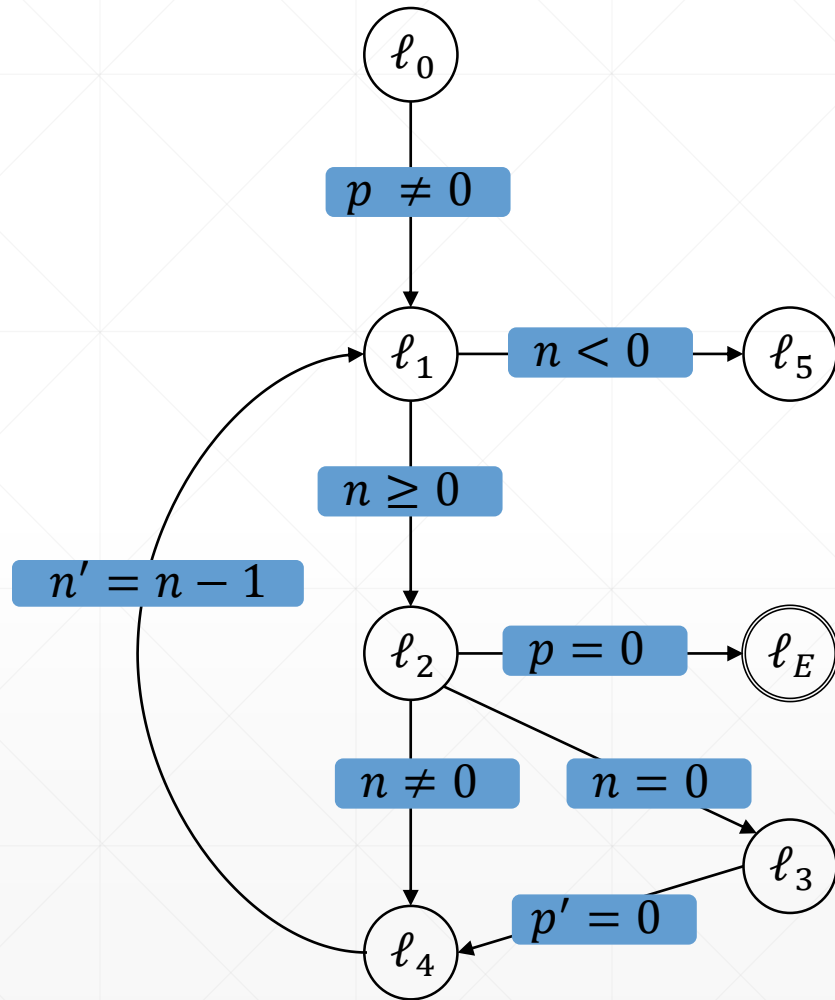
Example:

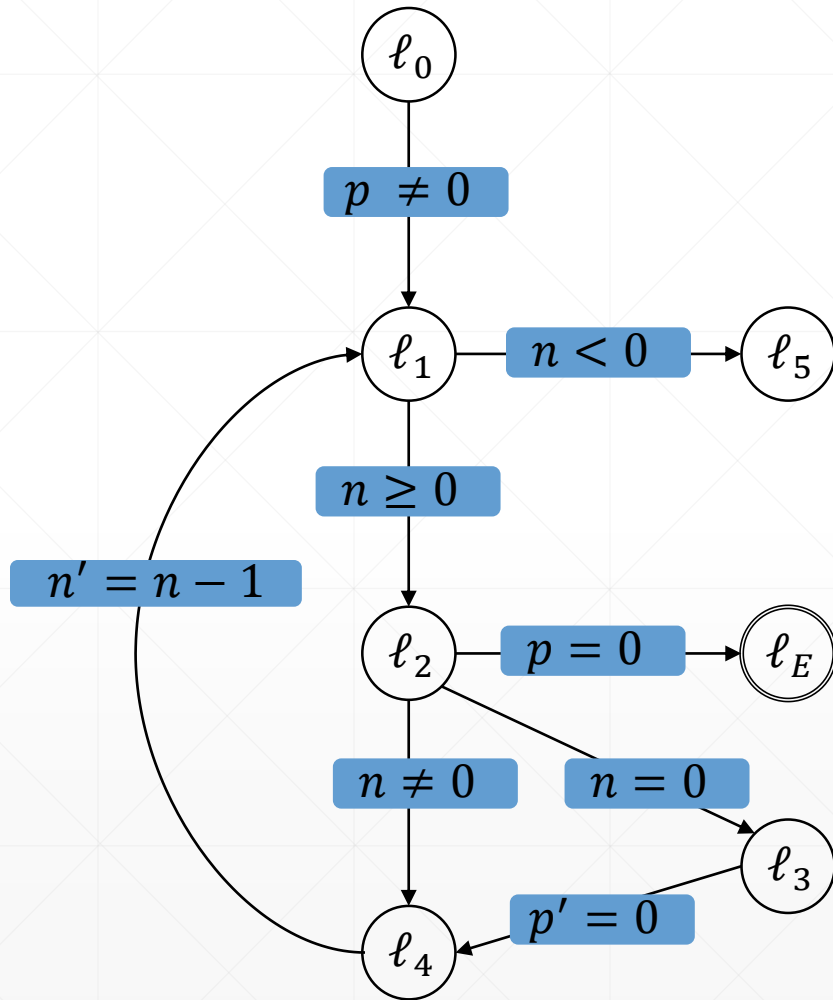

1. Step: Check for 0-Counter-Example

Example:

1. **Step:** Check for 0-Counter-Example

➤ Is $\ell_0 = \ell_E$?
  ➜ No. Continue with initialization

Example:

location	0
ℓ_0	
ℓ_1	
ℓ_2	
ℓ_3	
ℓ_4	

2. Step: Global Initialization

$$\triangleright \quad F_{0,\ell} = \begin{cases} \text{true,} & \ell = \ell_0 \\ \text{false,} & otherwise \end{cases}$$

Example:

location	0
ℓ_0	t
ℓ_1	f
ℓ_2	f
ℓ_3	f
ℓ_4	f

2. Step: Global Initialization

$$\triangleright \quad F_{0,\ell} = \begin{cases} \text{true,} & \ell = \ell_0 \\ \text{false,} & otherwise \end{cases}$$
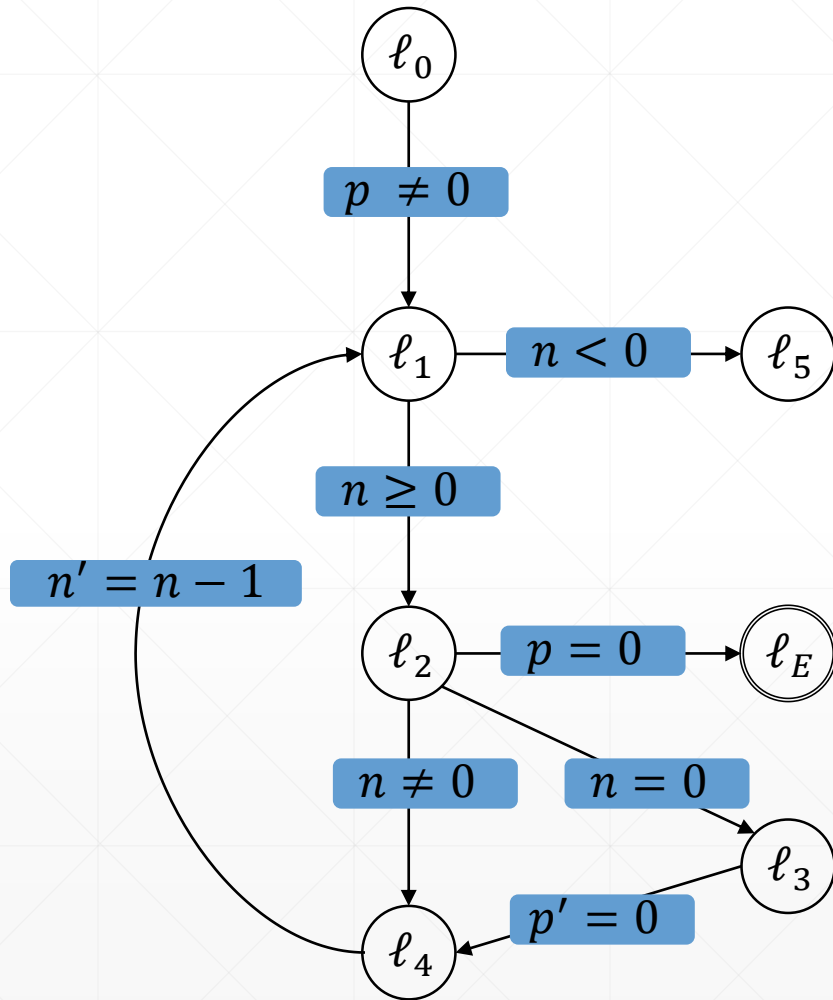
Example:
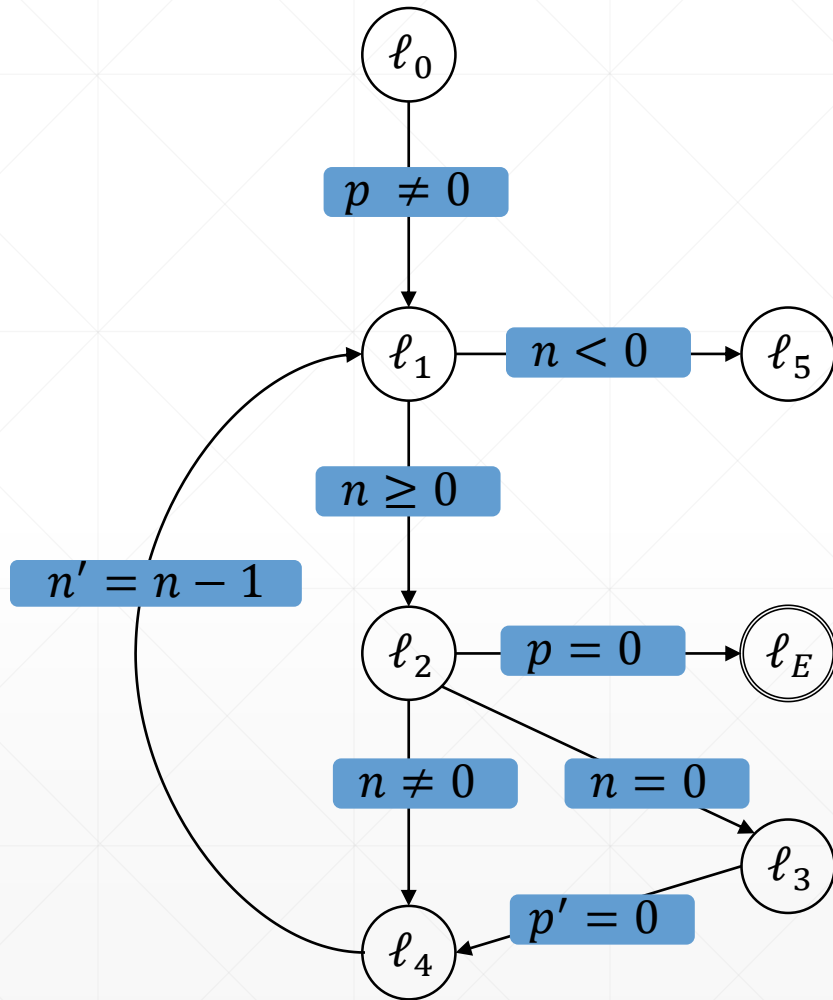

location	0	1
ℓ_0	t	
ℓ_1	f	
ℓ_2	f	
ℓ_3	f	
ℓ_4	f	

3. **Step:** Iteration 1 Initialization

➤ Initialize iteration 1 frames as true

23.09.2018

Example:

location	0	1
ℓ_0	t	t
ℓ_1	f	t
ℓ_2	f	t
ℓ_3	f	t
ℓ_4	f	t

3. Step: Iteration 1 Initialization

➤ Initialize iteration 1 frames as true

Example:
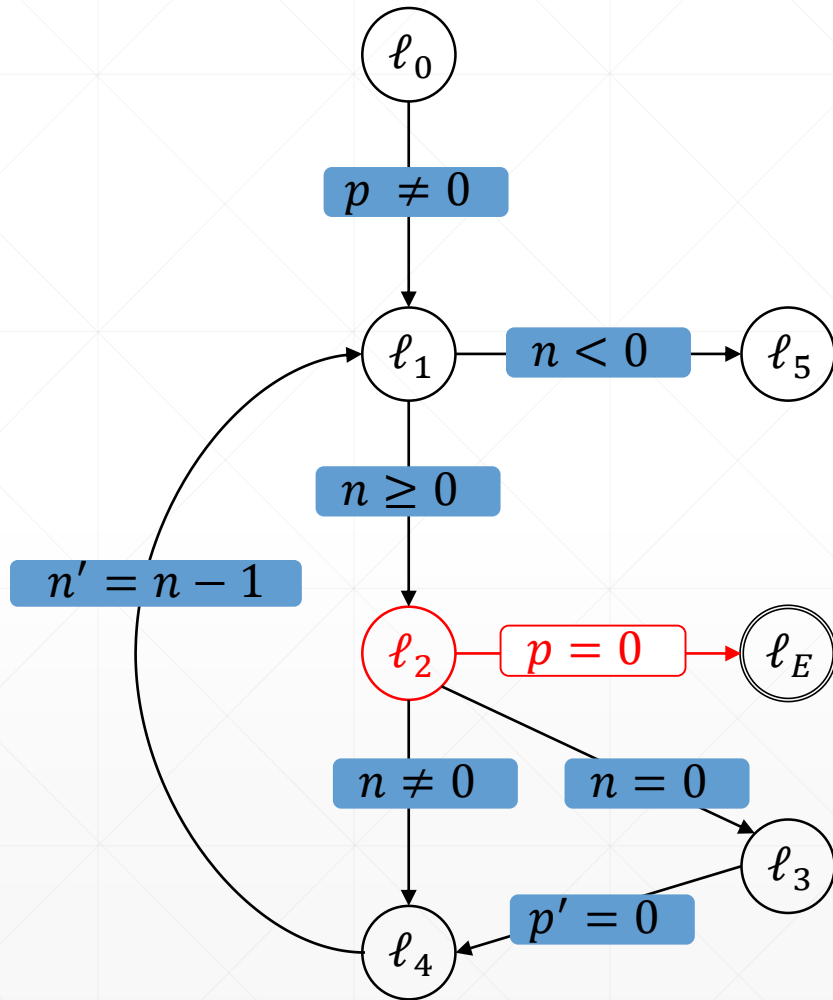
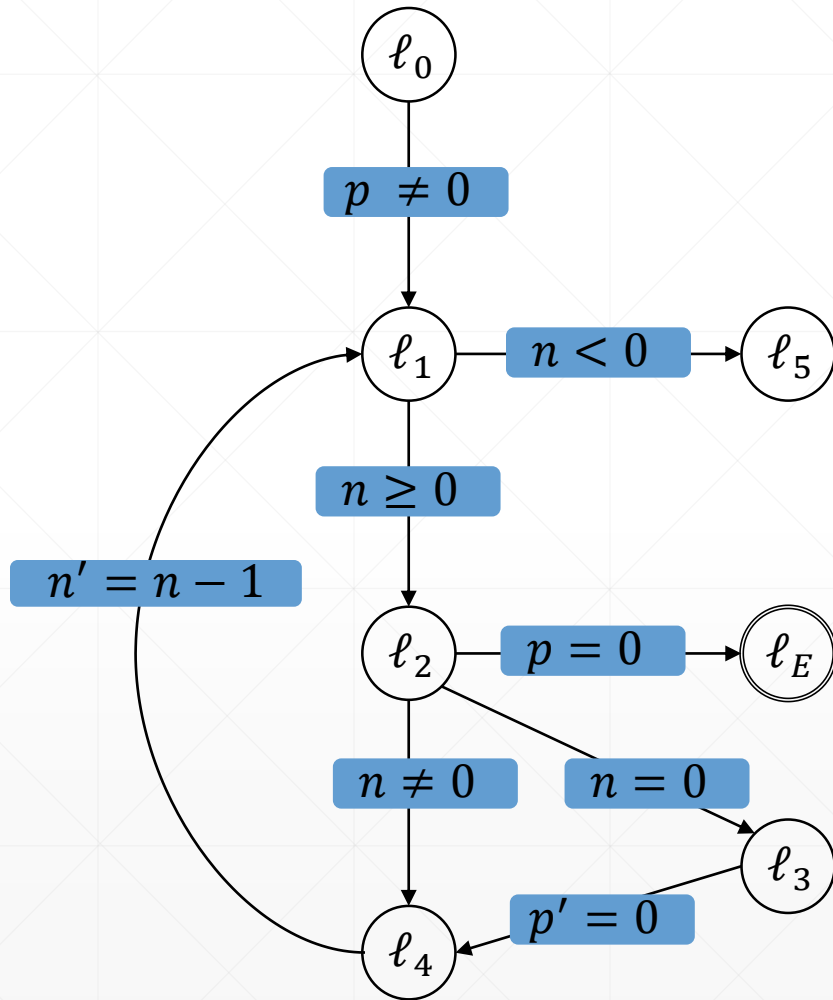| location	0	1
ℓ_0	t	t
ℓ_1	f	t
ℓ_2	f	t
ℓ_3	f	t
ℓ_4	f	t

3. Step: Iteration 1 Initialization

➢ Get initial proof-obligation:
  ➔ $(p = 0, \ell_2, 1)$

Proof-Obligations:
- $(p = 0, \ell_2, 1)$

Example:

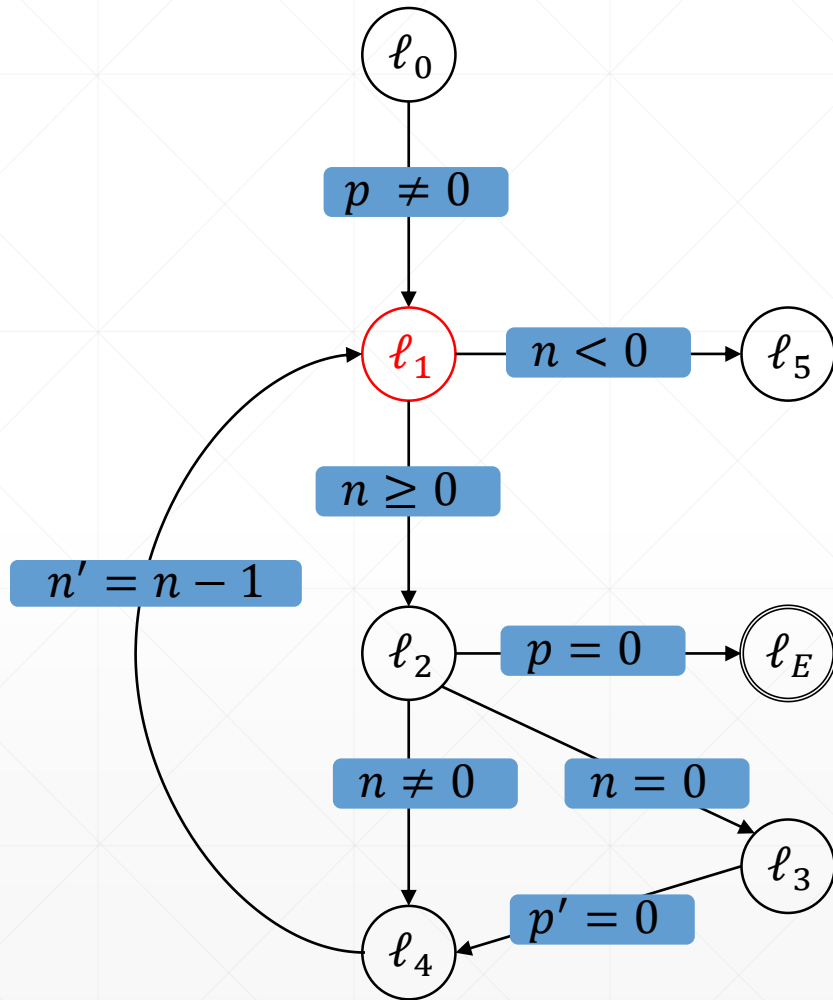| location	0	1
ℓ_0	t	t
ℓ_1	f	t
ℓ_2	f	t
ℓ_3	f	t
ℓ_4	f	t

4. Step: Iteration 1 Blocking-Phase:

➤ Try to block $(p = 0, \ell_2, 1)$

Proof-Obligations:
- $(p = 0, \ell_2, 1)$

Example:

location	0	1
ℓ_0	t	t
ℓ_1	f	t
ℓ_2	f	t
ℓ_3	f	t
ℓ_4	f	t

4. Step: Iteration 1 Blocking-Phase:

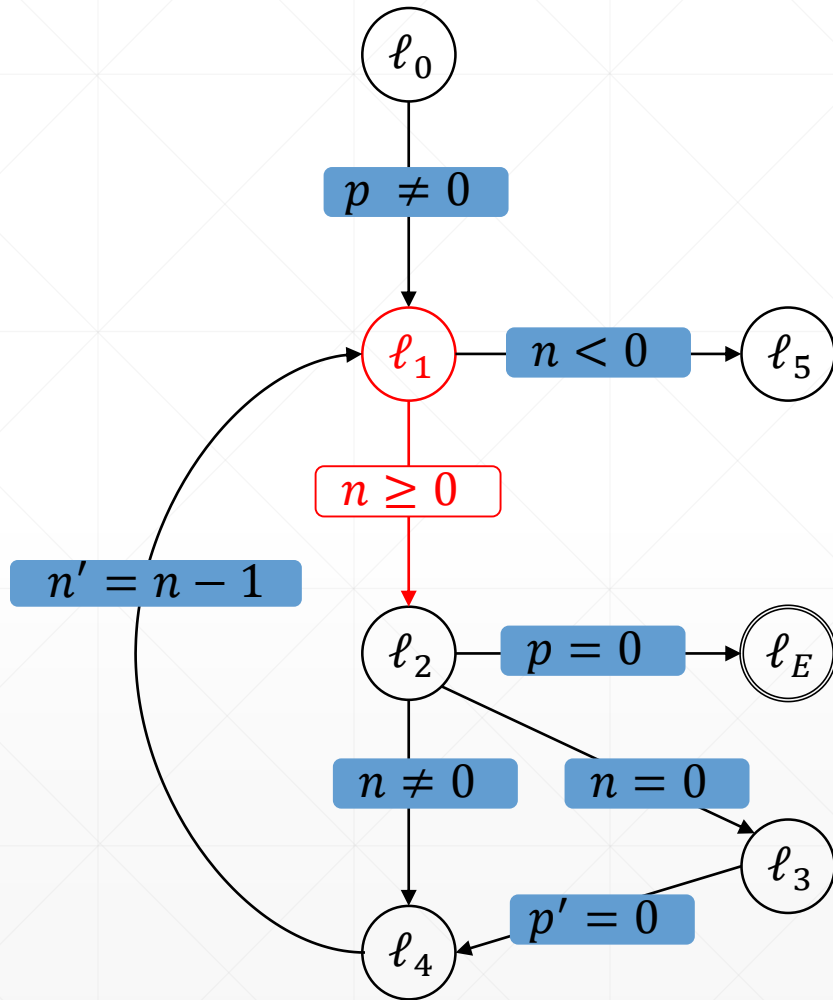➢ Try to block $(p = 0, \ell_2, 1)$

▪ Predecessor ℓ_1:
$$F_{0,\ell_1} \wedge T_{\ell_1 \to \ell_2} \wedge obligation'$$

Proof-Obligations:
- $(p = 0, \ell_2, 1)$

Example:

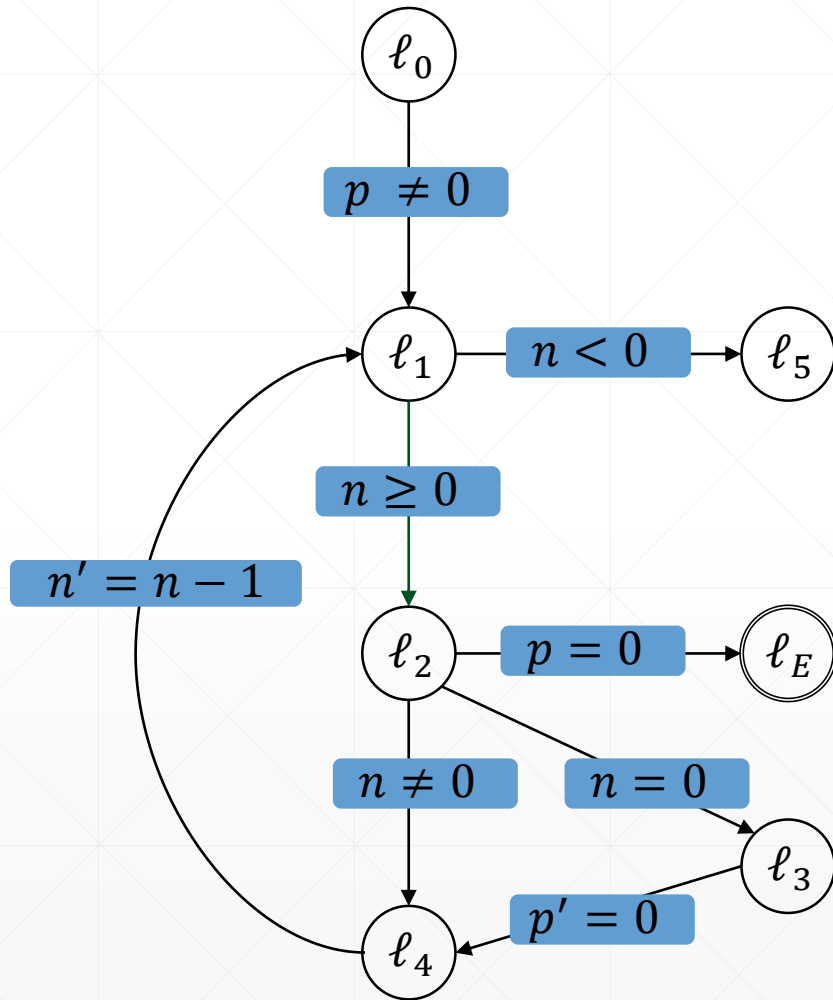| location	0	1
ℓ_0	t	t
ℓ_1	f	t
ℓ_2	f	t
ℓ_3	f	t
ℓ_4	f	t

4. Step: Iteration 1 Blocking-Phase:

➢ Try to block $(p = 0, \ell_2, 1)$

▪ Predecessor ℓ_1:
$$f \wedge n \geq 0 \wedge p' = 0$$

Proof-Obligations:
• $(p = 0, \ell_2, 1)$

23.09.2018

Example:

location	0	1
ℓ_0	t	t
ℓ_1	f	t
ℓ_2	f	t
ℓ_3	f	t
ℓ_4	f	t

4. Step: Iteration 1 Blocking-Phase:

➤ Try to block $(p = 0, \ell_2, 1)$

▪ Predecessor ℓ_1:
  $f \wedge n \geq 0 \wedge p' = 0$
  ➔ Unsatisfiable
  ➔ Strengthen frames $F_{0,\ell_2}, F_{1,\ell_2}$

Proof-Obligations:
• ∅

Example:

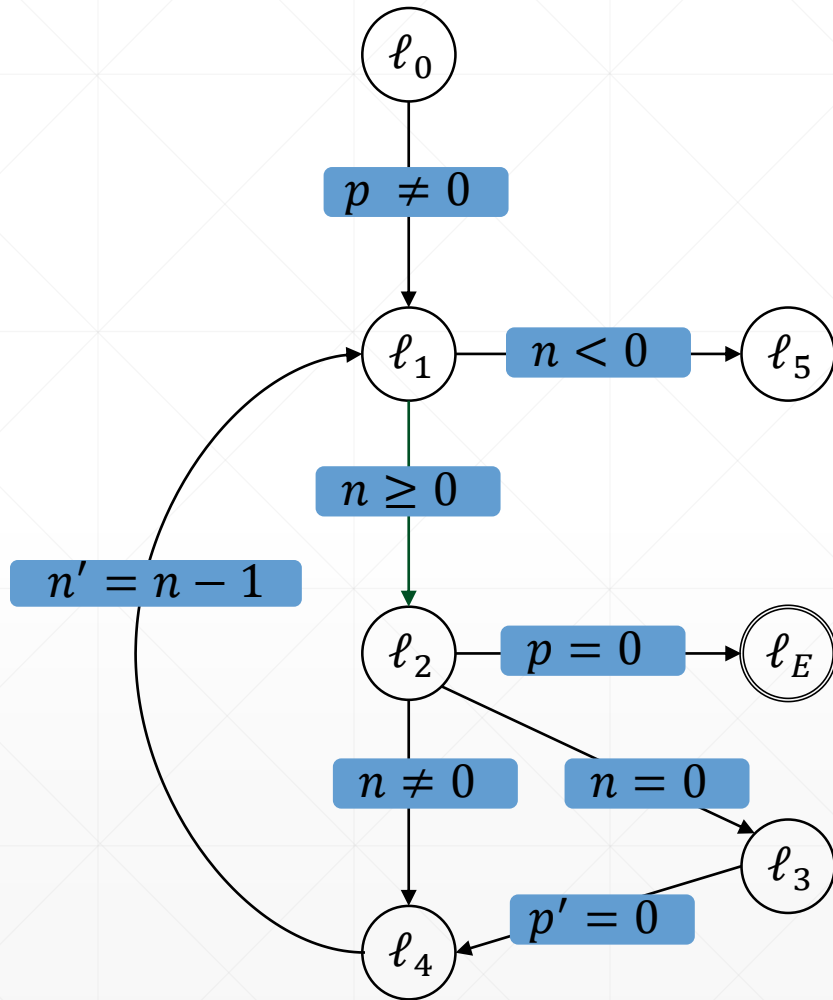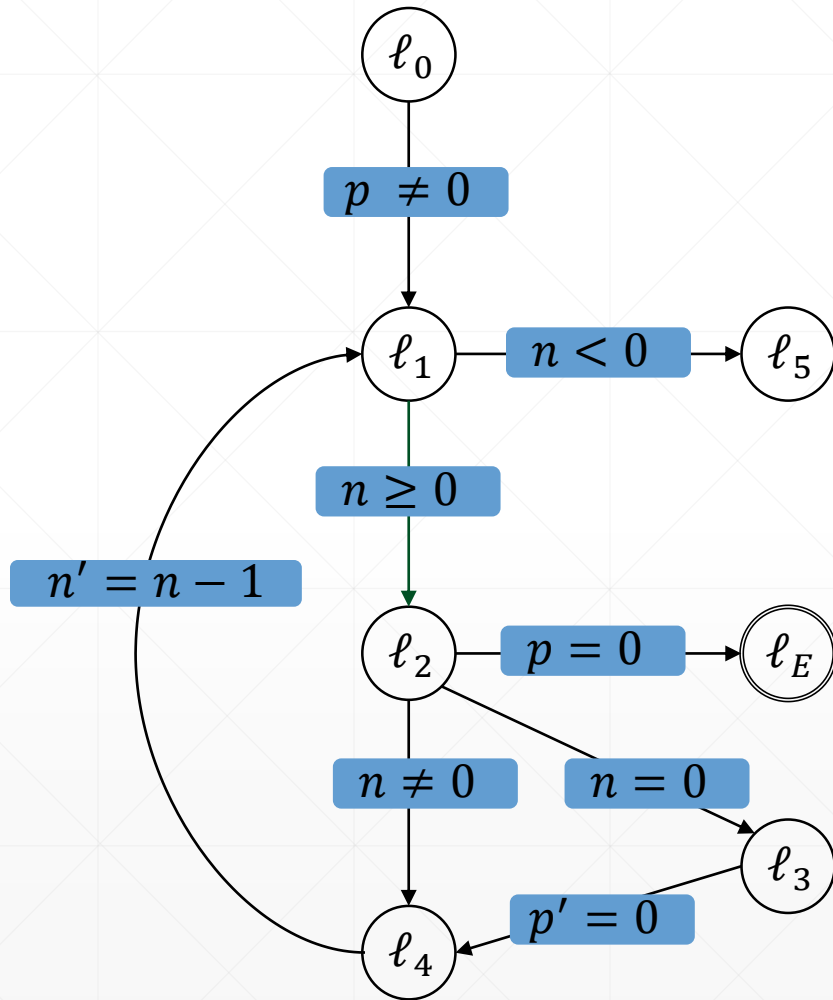| location	0	1
ℓ_0	t	t
ℓ_1	f	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$
ℓ_3	f	t
ℓ_4	f	t

4. Step: Iteration 1 Blocking-Phase:

➢ Try to block $(p = 0, \ell_2, 1)$

▪ Predecessor ℓ_1:
  $$f \wedge n \geq 0 \wedge p' = 0$$
  ➔ Unsatisfiable
  ➔ Strengthen frames $F_{0,\ell_2}, F_{1,\ell_2}$

Proof-Obligations:
- $\emptyset$

Example:

location	0	1
ℓ_0	t	t
ℓ_1	f	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$
ℓ_3	f	t
ℓ_4	f	t

5. Step: Iteration 1 Propagation-Phase

➤ Is there a global fixpoint?

Proof-Obligations:
- $\emptyset$
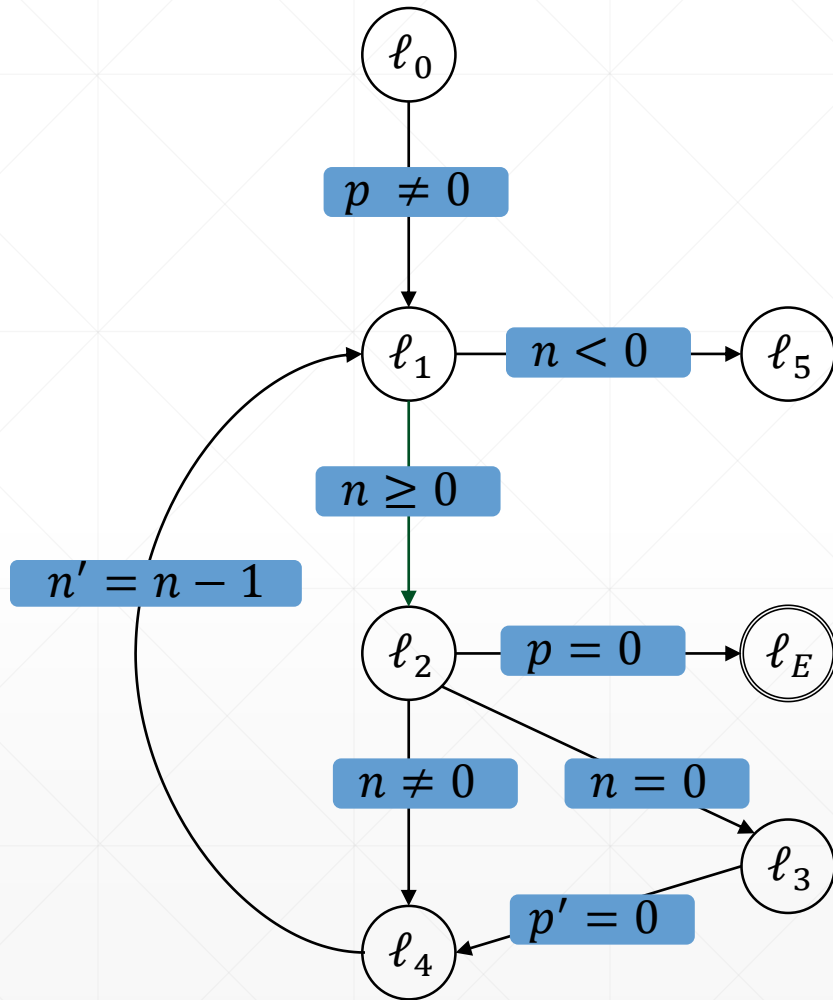
Example:

location	0	1
ℓ_0	t	t
ℓ_1	f	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$
ℓ_3	f	t
ℓ_4	f	t

5. Step: Iteration 1 Propagation-Phase

➤ Is there an i where
$F_{i-1,\ell} = F_{i,\ell}$ for $\ell \in L \setminus \{\ell_E\}$?

Proof-Obligations:
- $\emptyset$

Example:

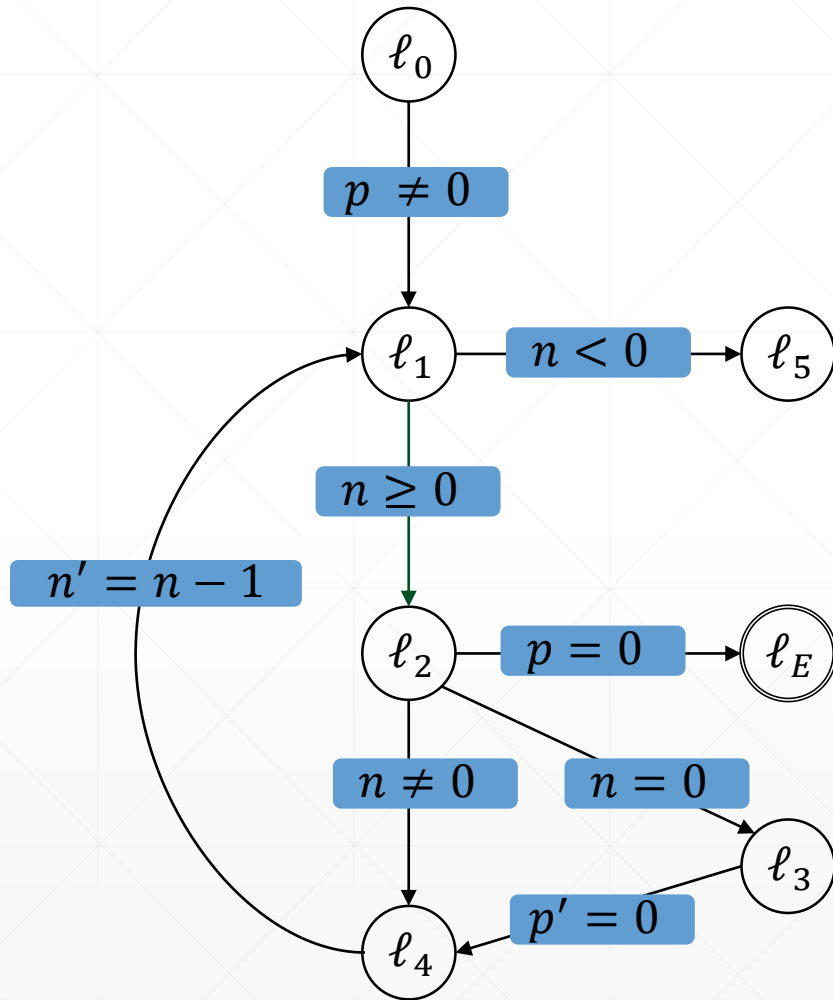| location	0	1
ℓ_0	t	t
ℓ_1	f	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$
ℓ_3	f	t
ℓ_4	f	t

5. Step: Iteration 1 Propagation-Phase

➤ Is there an i where
$F_{i-1,\ell} = F_{i,\ell}$ for $\ell \in L \setminus \{\ell_E\}$?

➔ No. Continue with iteration 2

Proof-Obligations:
- $\emptyset$

23.09.2018

Example:

location	0	1
ℓ_0	t	t
ℓ_1	f	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$
ℓ_3	f	t
ℓ_4	f	t

6. Step: Iteration 2 Initialization

➤ Initialize new frames

➤ Add initial proof-obligation
$(p = 0, \ell_2, 2)$

Proof-Obligations:
- $\emptyset$

Example:

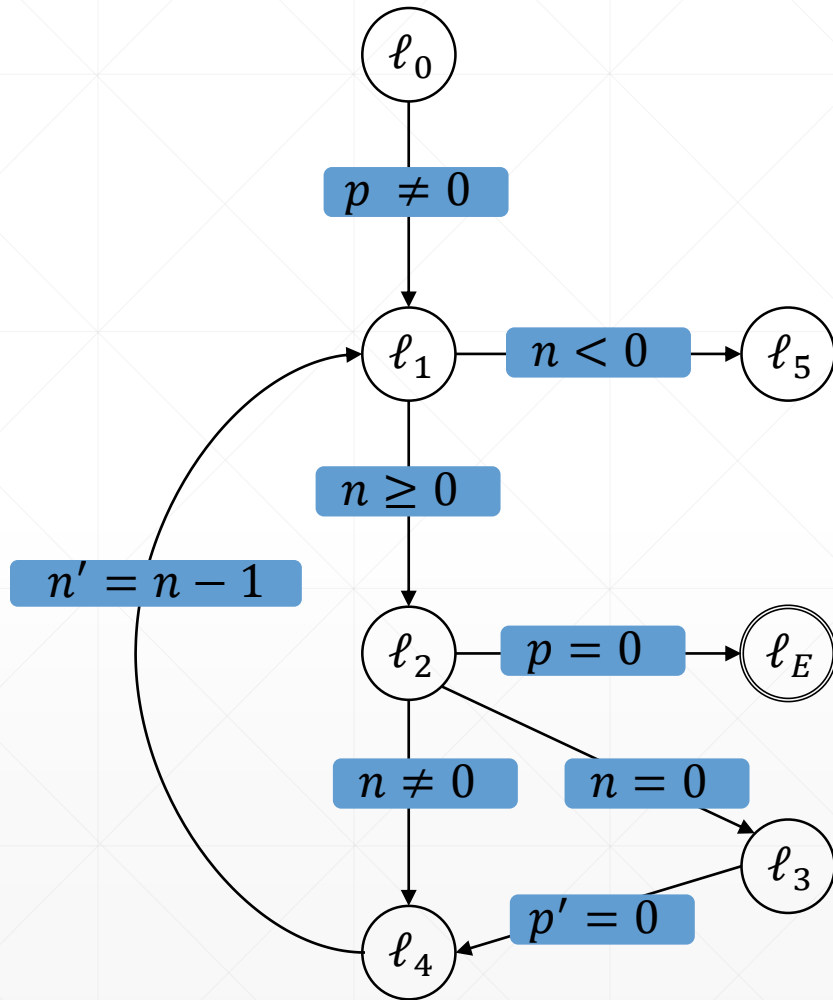| location	0	1	2
ℓ_0	t	t	t
ℓ_1	f	t	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_3	f	t	t
ℓ_4	f	t	t

6. Step: Iteration 2 Initialization

➢ Initialize new frames

➢ Add initial proof-obligation
  $(p = 0, \ell_2, 2)$

Proof-Obligations:
- $(p = 0, \ell_2, 2)$

23.09.2018

Example:

location	0	1	2
ℓ_0	t	t	t
ℓ_1	f	t	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_3	f	t	t
ℓ_4	f	t	t

7. Step: Iteration 2 Blocking-Phase:

➤ Try to block $(p = 0, \ell_2, 2)$

▪ Predecessor ℓ_1:
$$t \ \wedge \ n \geq 0 \ \wedge \ p' = 0$$

Proof-Obligations:
- $(p = 0, \ell_2, 2)$

Example:

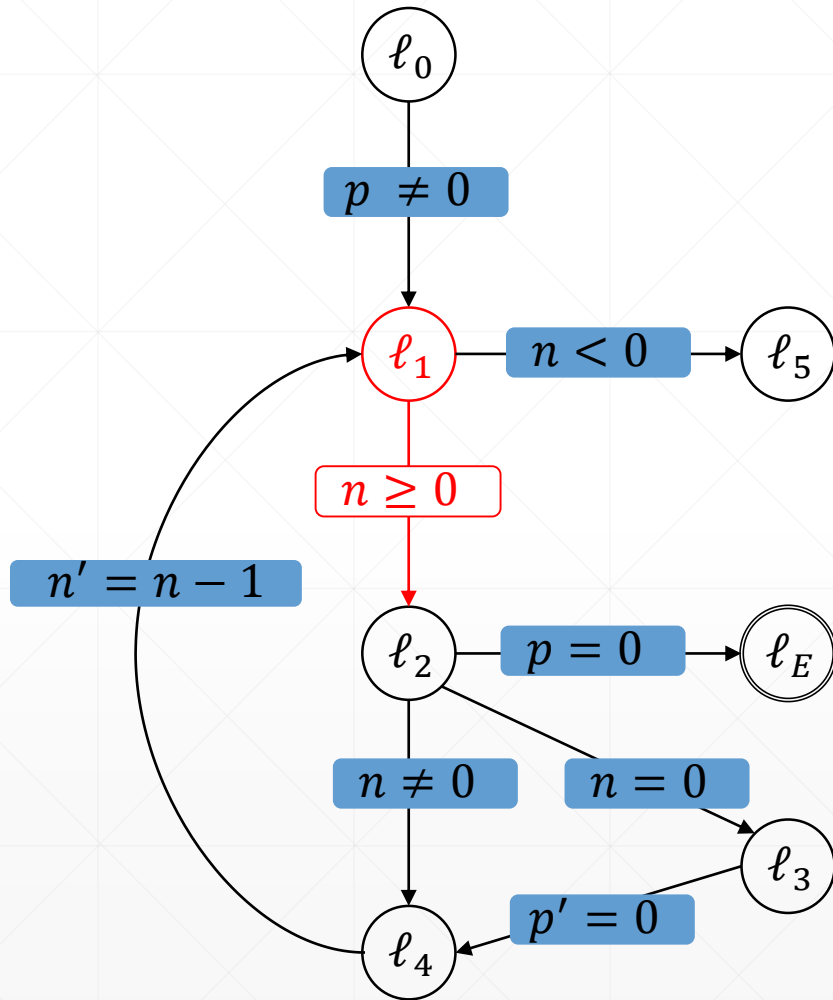| location	0	1	2
ℓ_0	t	t	t
ℓ_1	f	t	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_3	f	t	t
ℓ_4	f	t	t

7. Step: Iteration 2 Blocking-Phase:

➤ Try to block $(p = 0, \ell_2, 2)$

■ Predecessor ℓ_1:
$$t \wedge n \geq 0 \wedge p' = 0$$
➔ Satisfiable!
➔ $wp(n \geq 0, p' = 0) = (p = 0)$
➔ New proof-obligation $(p = 0, \ell_1, 1)$

Proof-Obligations:
- $(p = 0, \ell_2, 2)$

Example:

location	0	1	2
ℓ_0	t	t	t
ℓ_1	f	t	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_3	f	t	t
ℓ_4	f	t	t

7. Step: Iteration 2 Blocking-Phase:

➤ Try to block $(p = 0, \ell_2, 2)$

▪ Predecessor ℓ_1:
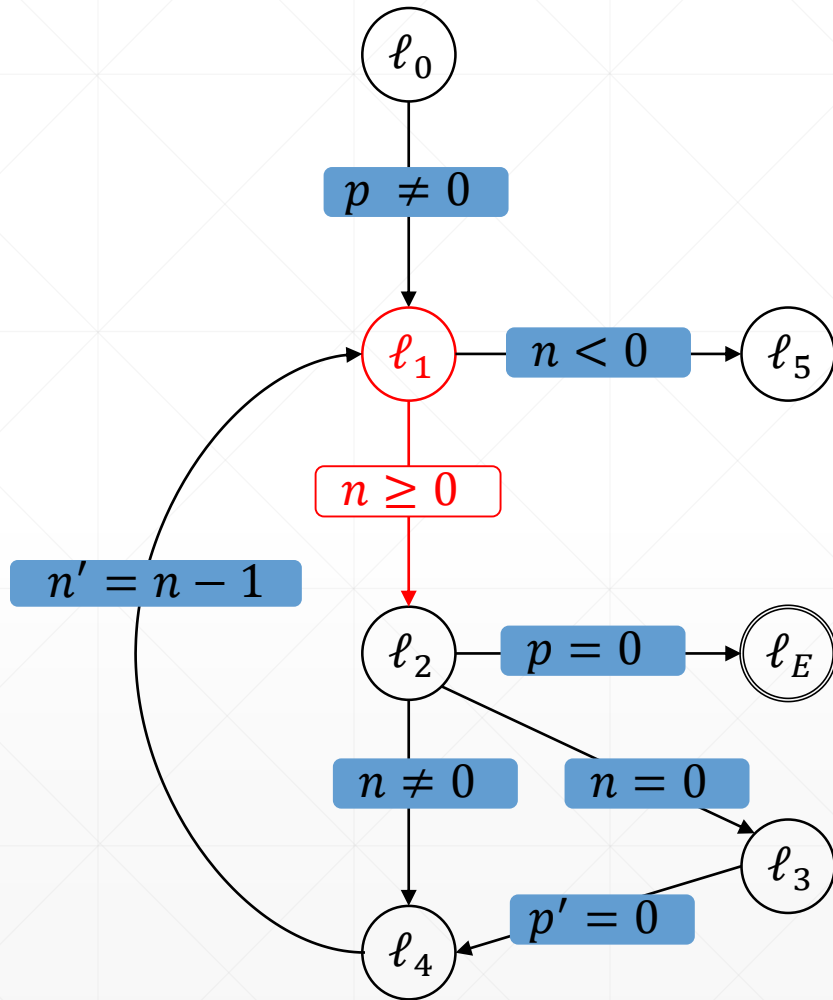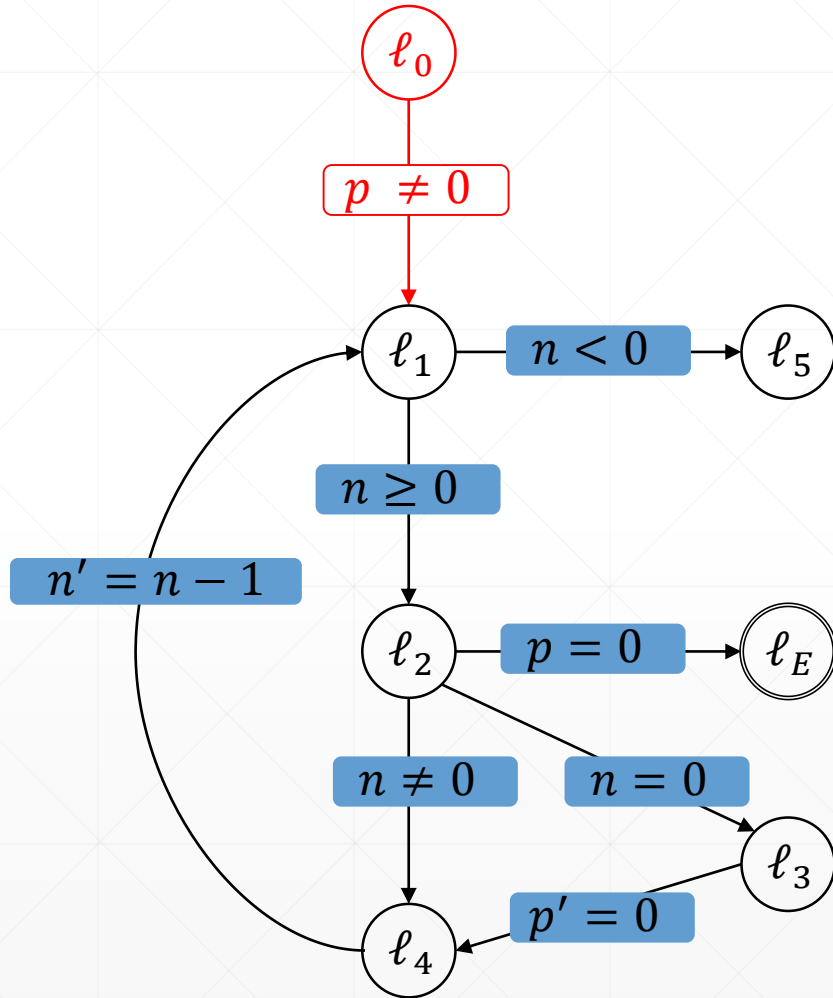$$t \wedge n \geq 0 \wedge p' = 0$$
➔ Satisfiable!
➔ $wp(n \geq 0, p' = 0) = (p = 0)$
➔ New proof-obligation $(p = 0, \ell_1, 1)$

Proof-Obligations:
- $(p = 0, \ell_2, 2)$
- $(p = 0, \ell_1, 1)$

Example:

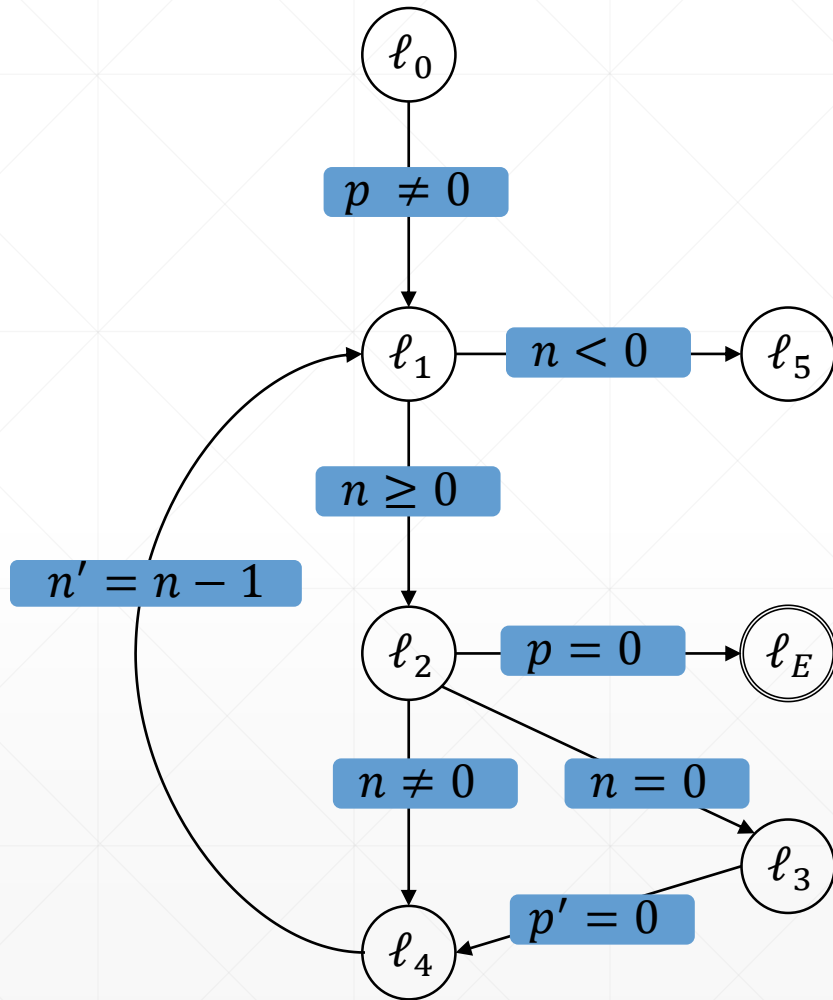| location	0	1	2
ℓ_0	t	t	t
ℓ_1	f	t	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_3	f	t	t
ℓ_4	f	t	t

7. Step: Iteration 2 Blocking-Phase:

➢ Try to block $(p = 0, \ell_1, 1)$

▪ Predecessor ℓ_0:
$$t \wedge p \neq 0 \wedge p' = 0$$

Proof-Obligations:
- $(p = 0, \ell_2, 2)$
- $(p = 0, \ell_1, 1)$

Example:

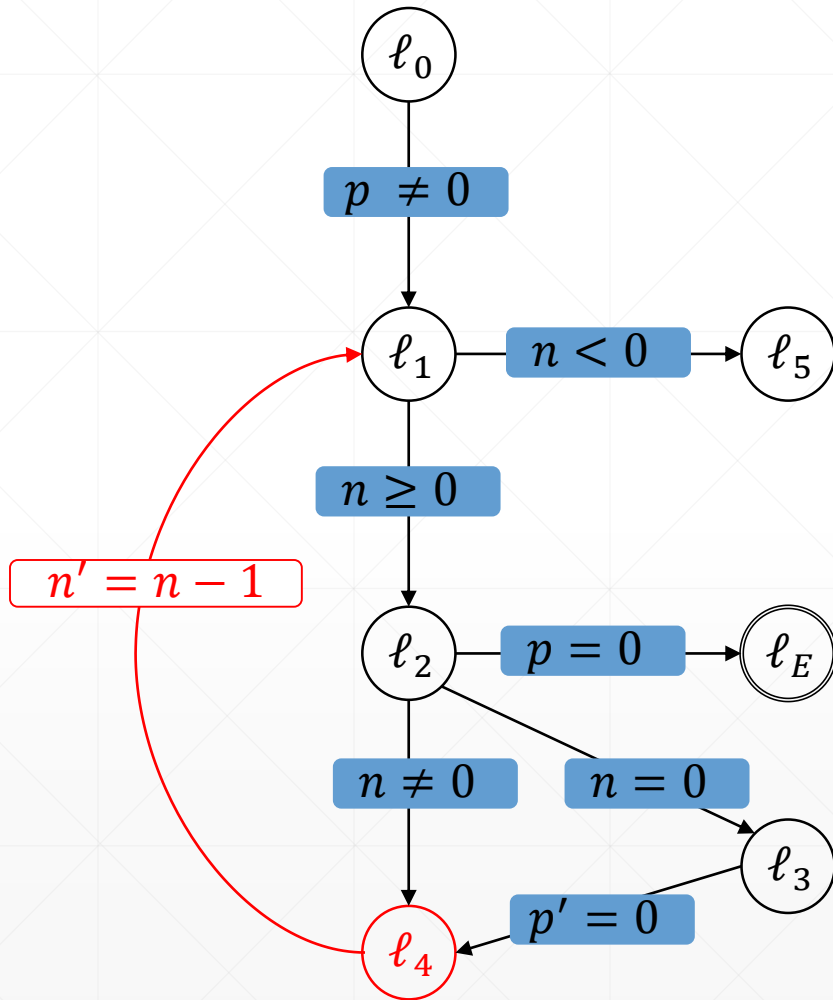| location	0	1	2
ℓ_0	t	t	t
ℓ_1	$f \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_3	f	t	t
ℓ_4	f	t	t

7. Step: Iteration 2 Blocking-Phase:

➢ Try to block $(p = 0, \ell_1, 1)$

▪ Predecessor ℓ_0:
$$t \wedge p \neq 0 \wedge p' = 0$$
➔ Unsatisfiable!
➔ Strengthen frames $F_{0,\ell_1}, F_{1,\ell_1}$

Proof-Obligations:
- $(p = 0, \ell_2, 2)$
- $(p = 0, \ell_1, 1)$

Example:

location	0	1	2
ℓ_0	t	t	t
ℓ_1	$f \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_3	f	t	t
ℓ_4	f	t	t

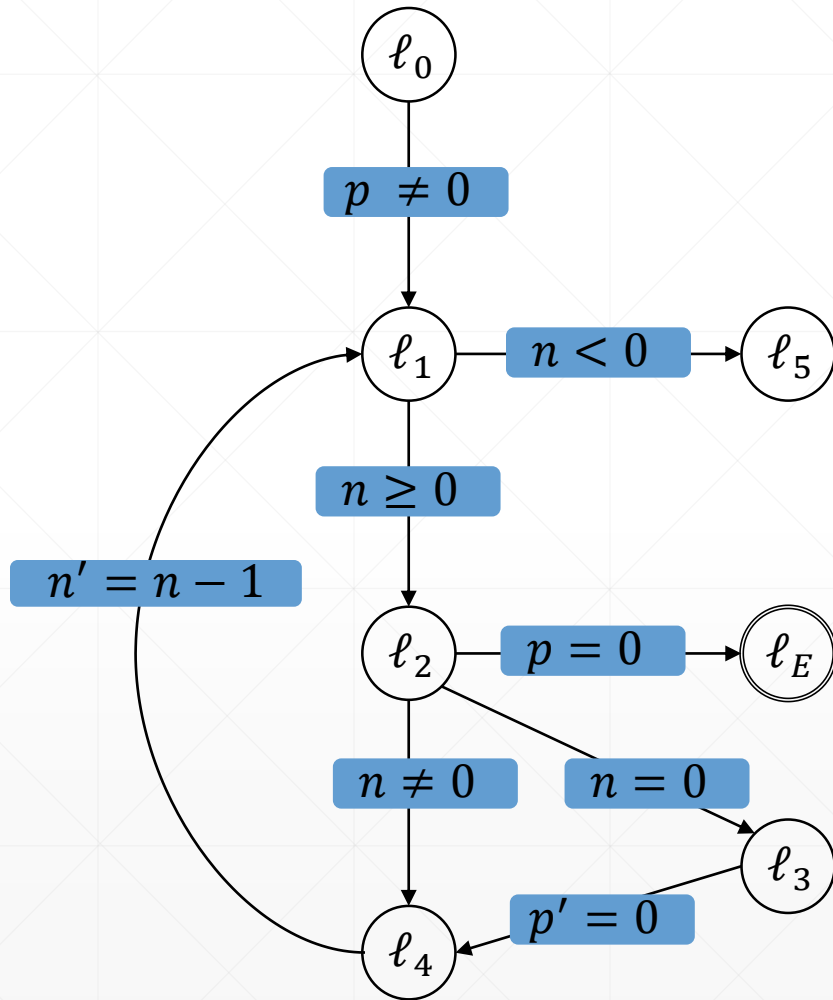**7. Step:** Iteration 2 Blocking-Phase:

➤ Try to block $(p = 0, \ell_1, 1)$

▪ Predecessor ℓ_4:
$$f \wedge n' = n - 1 \wedge p' = 0$$

Proof-Obligations:
- $(p = 0, \ell_2, 2)$
- $(p = 0, \ell_1, 1)$

23.09.2018

Example:

location	0	1	2
ℓ_0	t	t	t
ℓ_1	$f \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_3	f	t	t
ℓ_4	f	t	t

7. Step: Iteration 2 Blocking-Phase:

➢ Try to block $(p = 0, \ell_1, 1)$

▪ Predecessor ℓ_4:
$$f \wedge n' = n - 1 \wedge p' = 0$$
➔ Unsatisfiable!

Proof-Obligations:
- $(p = 0, \ell_2, 2)$
- $(p = 0, \ell_1, 1)$

23.09.2018

Example:

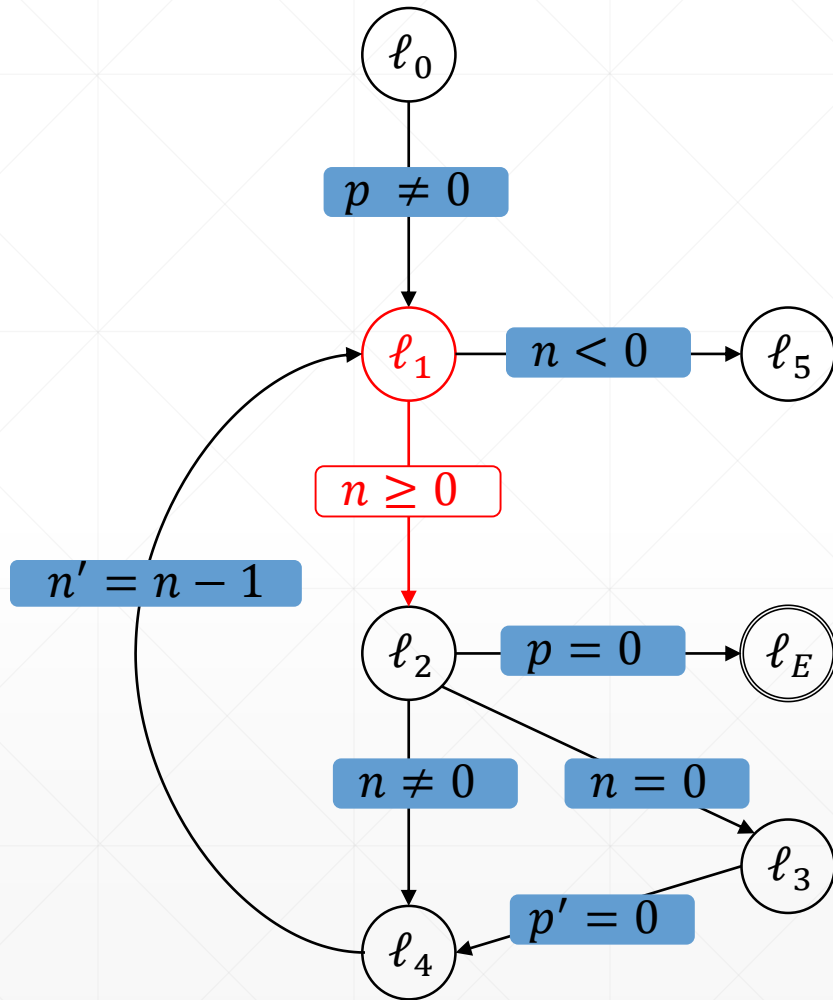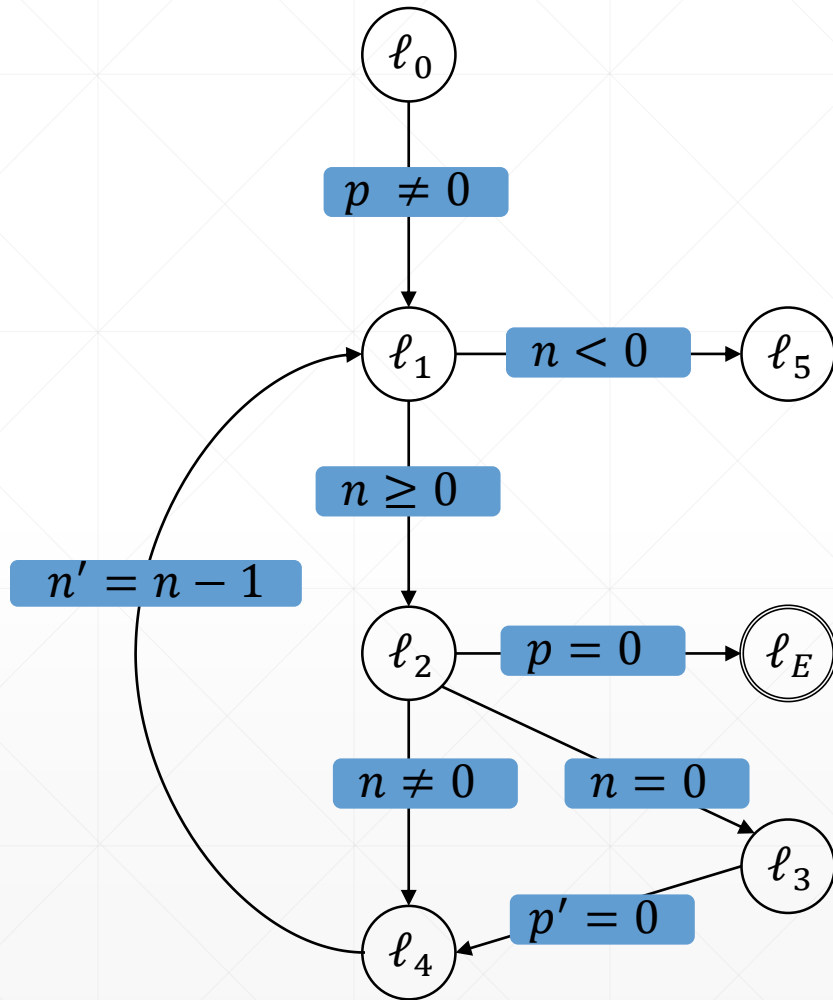| location	0	1	2
ℓ_0	t	t	t
ℓ_1	$f \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_3	f	t	t
ℓ_4	f	t	t

7. Step: Iteration 2 Blocking-Phase:

➢ Try to block $(p = 0, \ell_1, 1)$

▪ Predecessor ℓ_4:
$$f \wedge n' = n - 1 \wedge p' = 0$$
➔ Unsatisfiable!

Proof-Obligations:
- $(p = 0, \ell_2, 2)$

23.09.2018

Example:

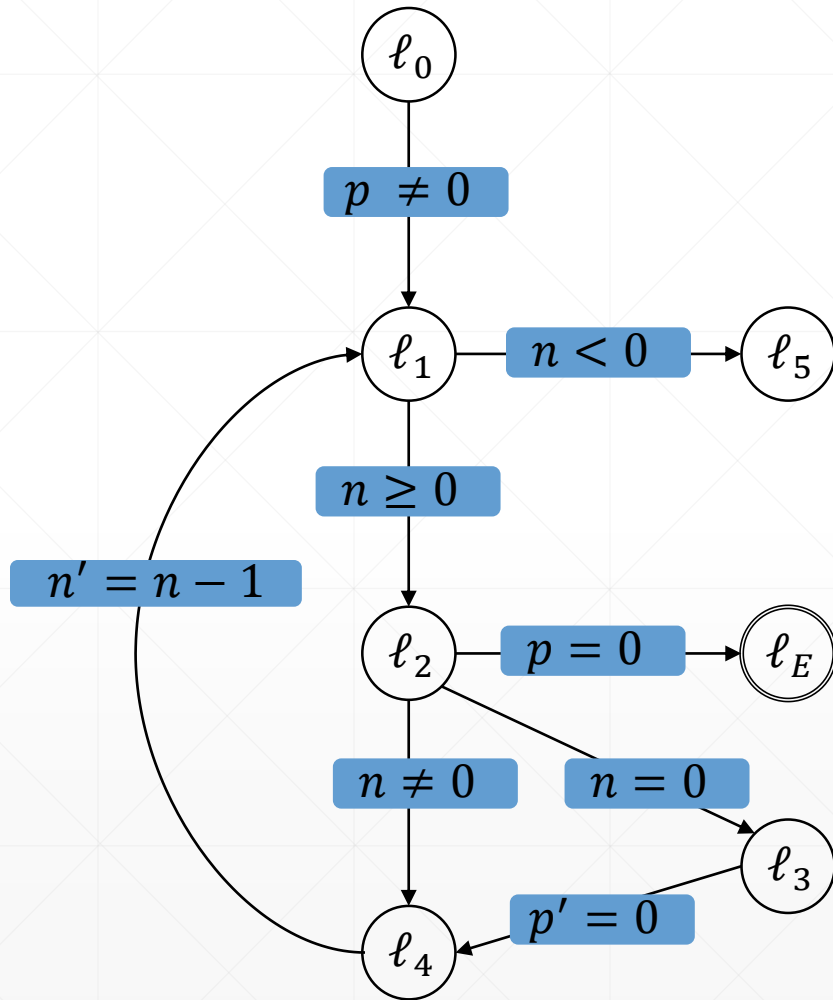| location	0	1	2
ℓ_0	t	t	t
ℓ_1	$f \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_3	f	t	t
ℓ_4	f	t	t

7. Step: Iteration 2 Blocking-Phase:

➢ Try to block $(p = 0, \ell_2, 2)$ again

▪ Predecessor ℓ_1:
$$t \wedge p \neq 0 \wedge n \geq 0 \wedge p' = 0$$

Proof-Obligations:
- $(p = 0, \ell_2, 2)$

23.09.2018

Example:

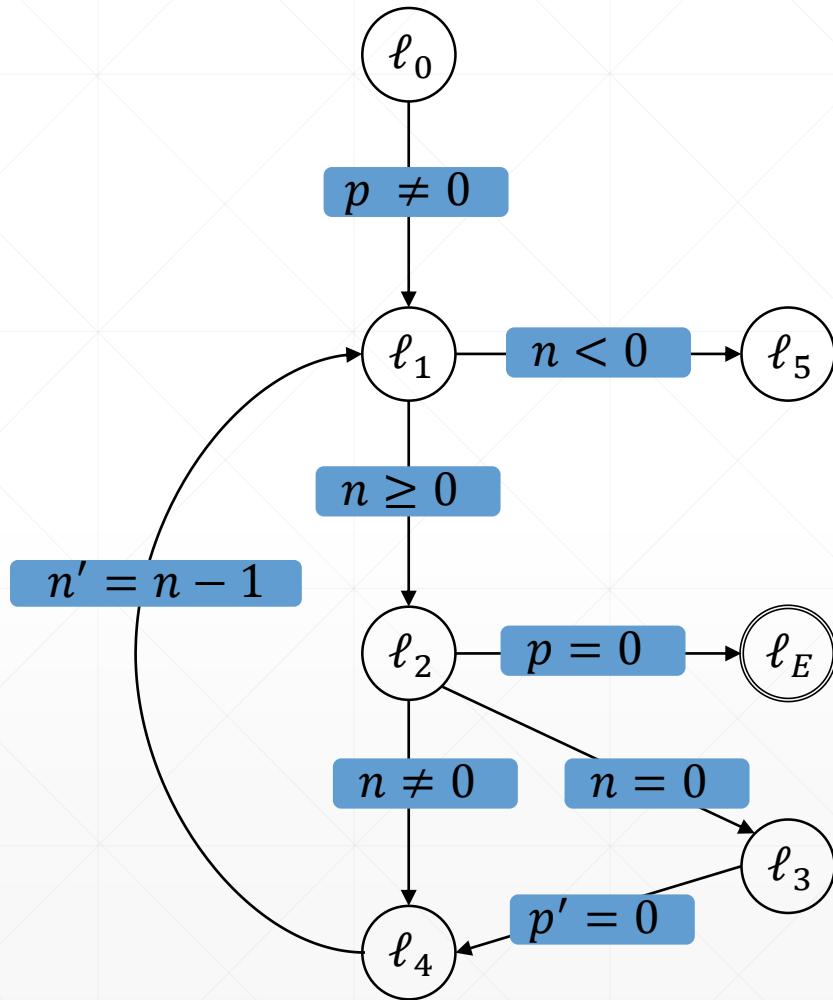| location	0	1	2
ℓ_0	t	t	t
ℓ_1	$f \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$
ℓ_3	f	t	t
ℓ_4	f	t	t

7. Step: Iteration 2 Blocking-Phase:

➢ Try to block $(p = 0, \ell_2, 2)$ again

▪ Predecessor ℓ_1:
  $t \wedge p \neq 0 \wedge n \geq 0 \wedge p' = 0$
  ➔ Unsatisfiable!
  ➔ Strengthen frames F_{2,ℓ_2}

Proof-Obligations:
• ∅

Example:

location	0	1	2
ℓ_0	t	t	t
ℓ_1	$f \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$
ℓ_3	f	t	t
ℓ_4	f	t	t

8. Step: Iteration 2 Propagation-Phase:

➤ Is there a global fixpoint?
➔ No. Continue with Iteration 3

Proof-Obligations:
- $\emptyset$

Example:

location	0	1	2
ℓ_0	t	t	t
ℓ_1	$f \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$
ℓ_3	f	t	t
ℓ_4	f	t	t

9. Step: Iteration 3 Initialization

➢ Initialize new frames

➢ Get initial proof-obligations

Proof-Obligations:
- $\emptyset$

Example:

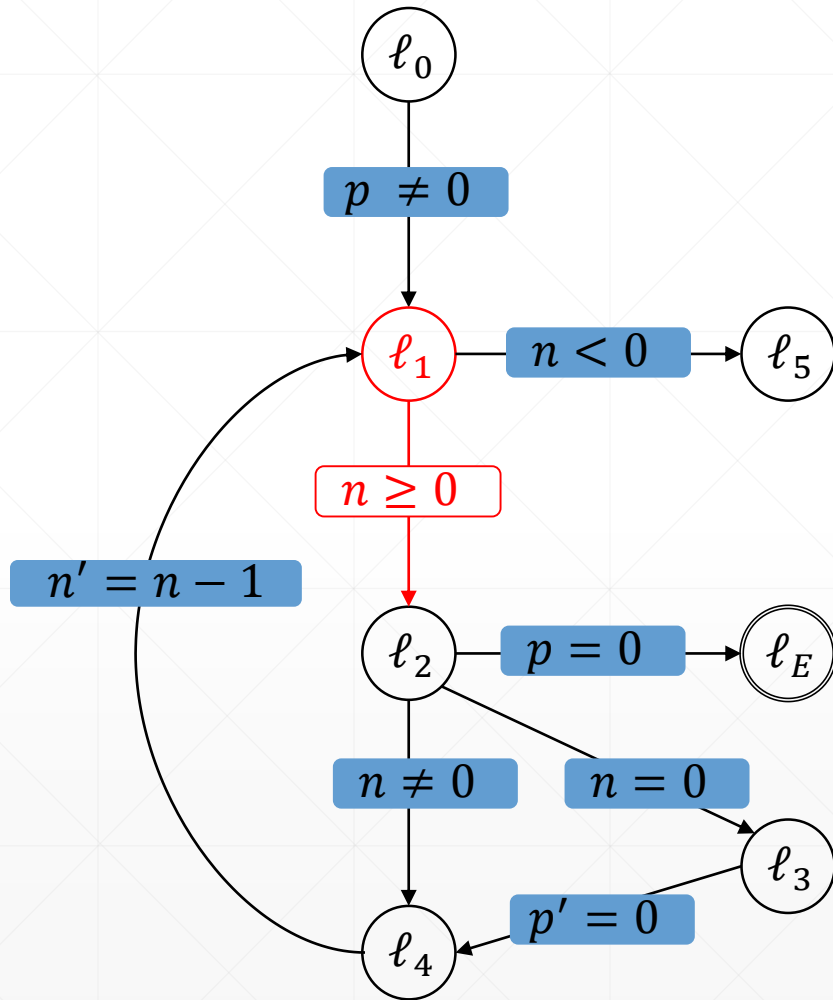| location	0	1	2	3
ℓ_0	t	t	t	t
ℓ_1	$f \wedge p \neq 0$	$t \wedge p \neq 0$	t	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_3	f	t	t	t
ℓ_4	f	t	t	t

9. Step: Iteration 3 Initialization

➤ Initialize new frames

➤ Get initial proof-obligations

Proof-Obligations:
- $(p = 0, \ell_2, 3)$

23.09.2018

Example:

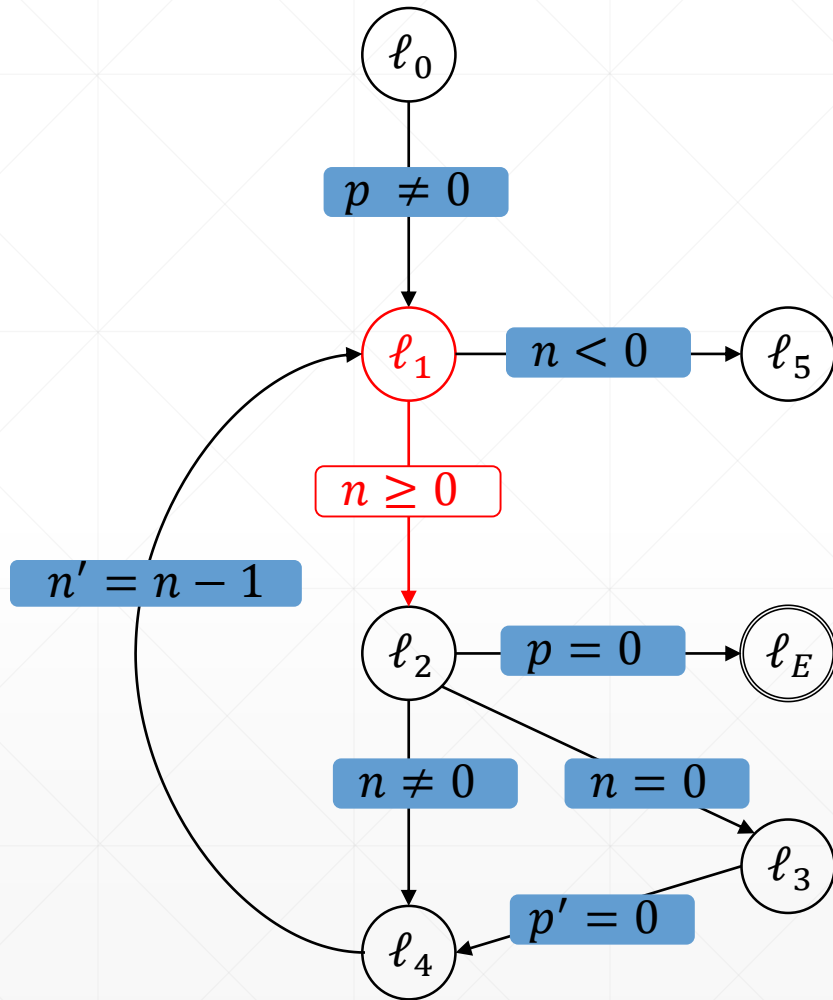| location	0	1	2	3
ℓ_0	t	t	t	t
ℓ_1	$f \wedge p \neq 0$	$t \wedge p \neq 0$	t	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_3	f	t	t	t
ℓ_4	f	t	t	t

10. Step: Iteration 3 Blocking-Phase

➢ Try to block $(p = 0, \ell_2, 3)$

▪ Predecessor ℓ_1:
   $$t \wedge n \geq 0 \wedge p' = 0$$
   ➔ Like the Iteration before this is satisfiable

Proof-Obligations:
- $(p = 0, \ell_2, 3)$

Example:

location	0	1	2	3
ℓ_0	t	t	t	t
ℓ_1	$f \wedge p \neq 0$	$t \wedge p \neq 0$	t	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_3	f	t	t	t
ℓ_4	f	t	t	t

10. Step: Iteration 3 Blocking-Phase

➢ Try to block $(p = 0, \ell_2, 3)$

▪ Predecessor ℓ_1:
  $$t \wedge n \geq 0 \wedge p' = 0$$
  ➔ Get same proof-obligation as before but on Iteration 2
  ➔ $(p = 0, \ell_1, 2)$

Proof-Obligations:
• $(p = 0, \ell_2, 3)$

Example:

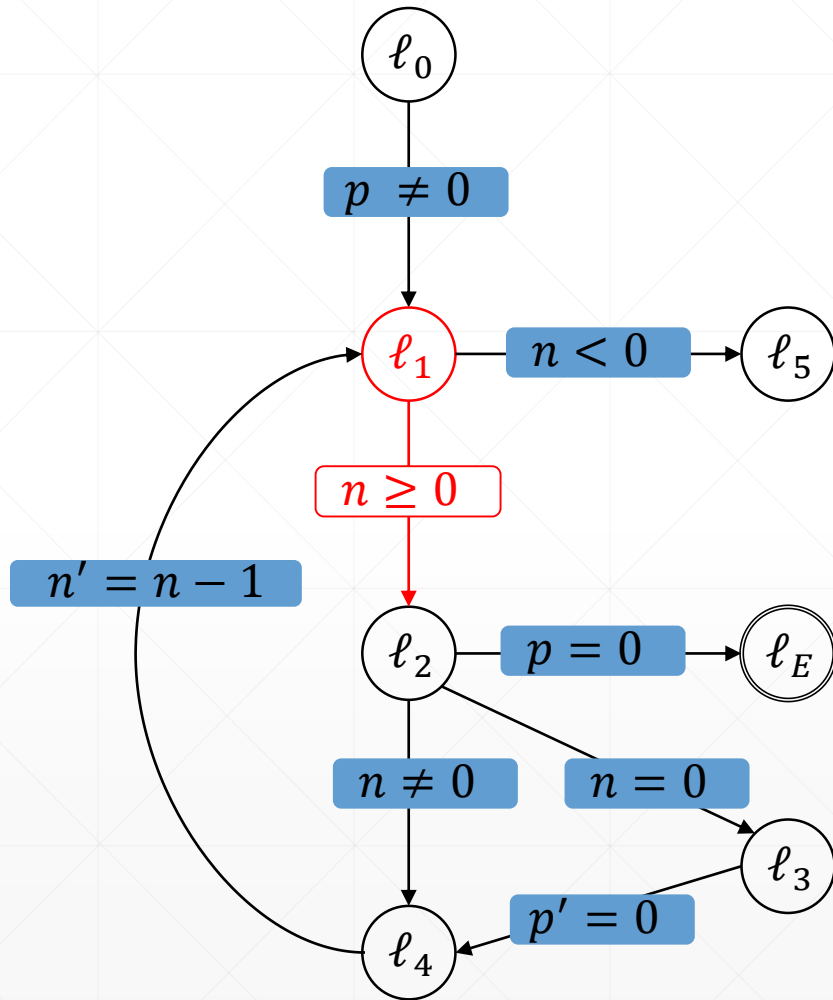| location	0	1	2	3
ℓ_0	t	t	t	t
ℓ_1	$f \wedge p \neq 0$	$t \wedge p \neq 0$	t	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_3	f	t	t	t
ℓ_4	f	t	t	t

10. Step: Iteration 3 Blocking-Phase

➢ Try to block $(p = 0, \ell_2, 3)$

▪ Predecessor ℓ_1:
$$t \wedge n \geq 0 \wedge p' = 0$$
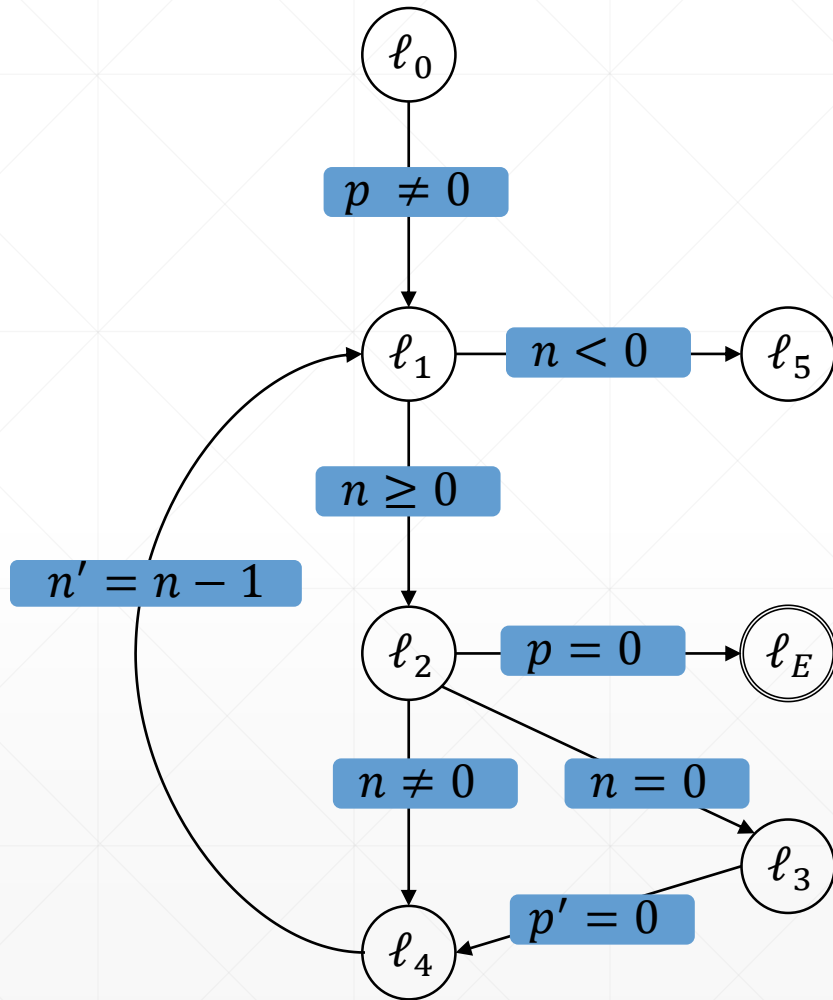➔ Get same proof-obligation as before but on Iteration 2
➔ $(p = 0, \ell_1, 2)$

Proof-Obligations:
- $(p = 0, \ell_2, 3)$
- $(p = 0, \ell_1, 2)$

23.09.2018

Example:

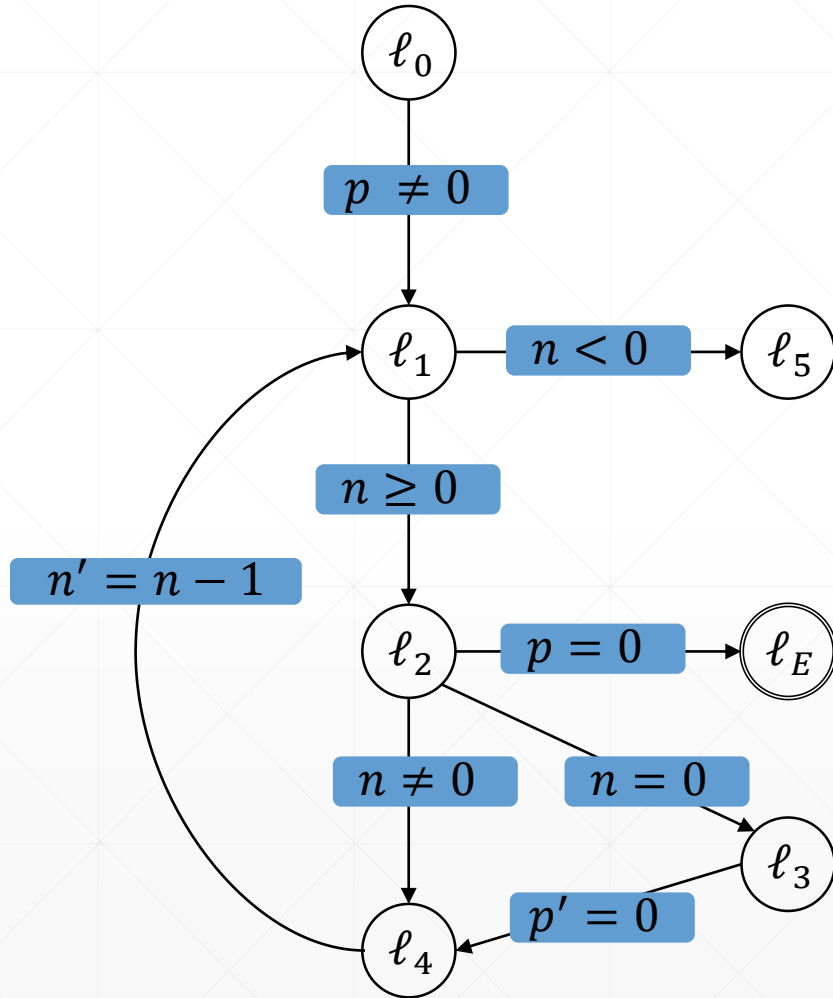| location	0	1	2	3
ℓ_0	t	t	t	t
ℓ_1	$f \wedge p \neq 0$	$t \wedge p \neq 0$	t	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_3	f	t	t	t
ℓ_4	f	t	t	t

10. Step: Iteration 3 Blocking-Phase

➤ There are a lot of repetitions
  ➜ Duplicate proof-obligations

Proof-Obligations:
- $(p = 0, \ \ell_2, \ 3)$
- $(p = 0, \ \ell_1, \ 2)$

Example:

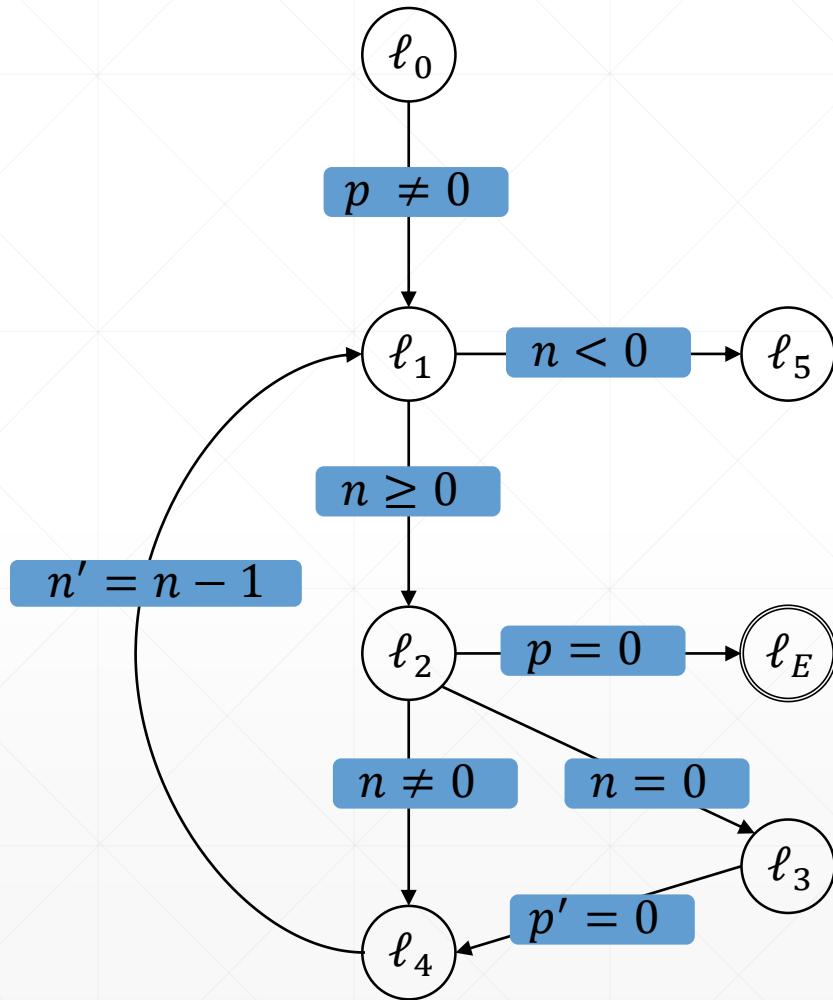| location	0	1	2	3
ℓ_0	t	t	t	t
ℓ_1	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$
ℓ_3	f	t	t	t
ℓ_4	$f \wedge p \neq 0$	$t \wedge p \neq 0$	t	t

10. Step: Iteration 3 Blocking-Phase

➢ There are a lot of repetitions
  → Duplicate proof-obligations

Proof-Obligations:
- $\emptyset$

Example:

location	0	1	2	3
ℓ_0	t	t	t	t
ℓ_1	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$
ℓ_3	f	t	t	t
ℓ_4	$f \wedge p \neq 0$	$t \wedge p \neq 0$	t	t

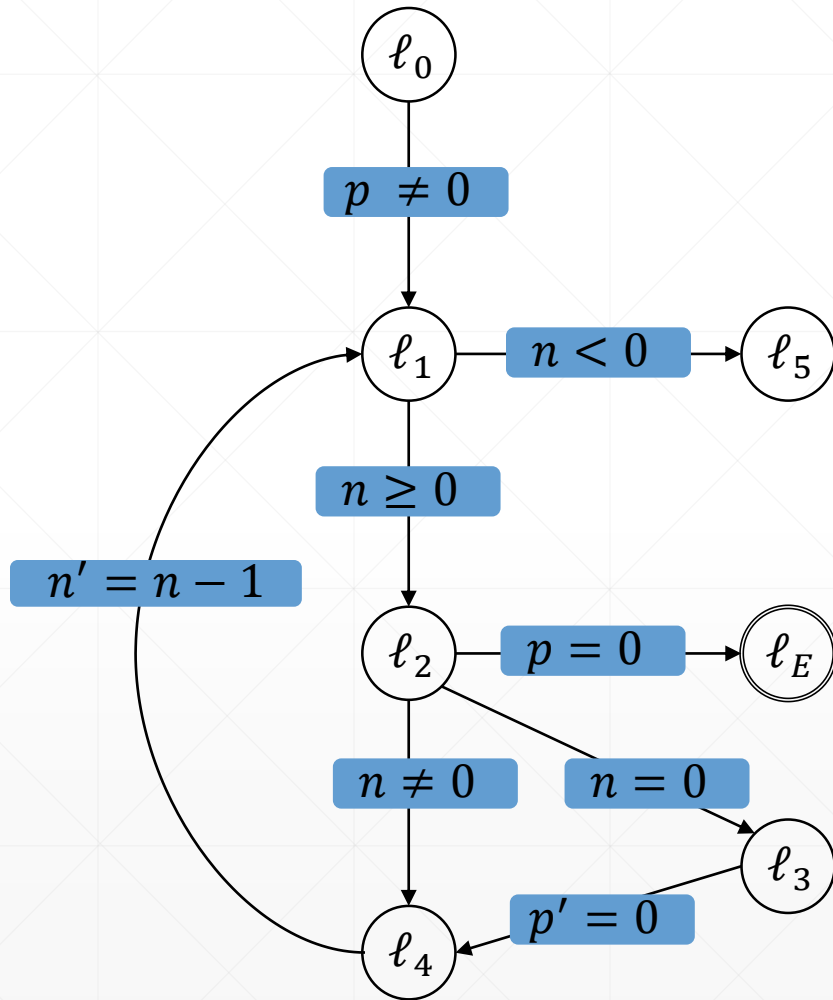**11. Step:** Iteration 3 Propagation-Phase

➢ Is there a global fixpoint?
➔ No. Continue with Iteration 4

Proof-Obligations:
- ∅

Example:

location	0	1	2	3	4
ℓ_0	t	t	t	t	t
ℓ_1	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_3	f	t	t	t	t
ℓ_4	$f \wedge p \neq 0$	$t \wedge p \neq 0$	t	t	t

11. Step: Iteration 4 Initialization

Example:

location	0	1	2	3	4
ℓ_0	t	t	t	t	t
ℓ_1	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_3	f	t	t	t	t
ℓ_4	$f \wedge p \neq 0$	$t \wedge p \neq 0$	t	t	t

12. Step: Iteration 4 Blocking-Phase

Example:

location	0	1	2	3	4
ℓ_0	t	t	t	t	t
ℓ_1	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$
ℓ_3	$f \wedge f$	$t \wedge f$	t	t	t
ℓ_4	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t	t

12. Step: Iteration 4 Blocking-Phase

Example:

location	0	1	2	3	4
ℓ_0	t	t	t	t	t
ℓ_1	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$
ℓ_3	$f \wedge f$	$t \wedge f$	t	t	t
ℓ_4	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t	t

13. Step: Iteration 4 Propagation-Phase

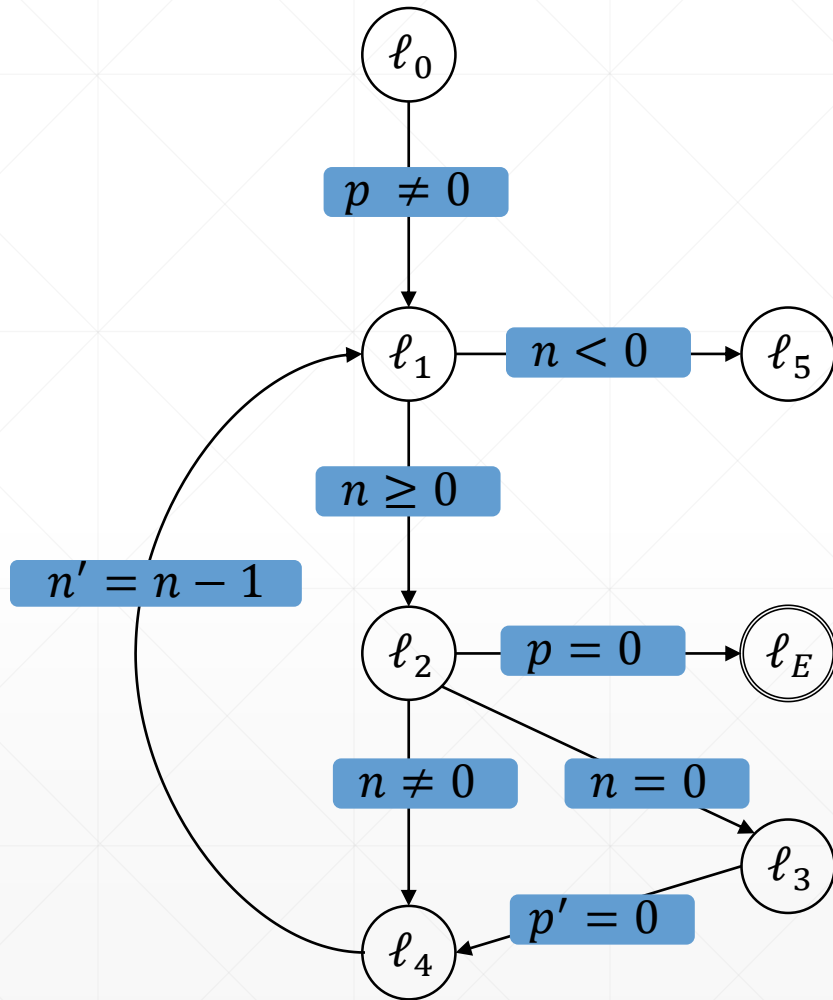➤ Is there a global fixpoint?
   ➔ No. Continue with Iteration 5

Example:

location	0	1	2	3	4	5
ℓ_0	t	t	t	t	t	t
ℓ_1	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_3	$f \wedge f$	$t \wedge f$	t	t	t	t
ℓ_4	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t	t	t

14. Step: Iteration 5 Initialization

Example:

location	0	1	2	3	4	5
ℓ_0	t	t	t	t	t	t
ℓ_1	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_3	$f \wedge f$	$t \wedge f$	t	t	t	t
ℓ_4	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t	t	t

15. Step: Iteration 5 Blocking-Phase

Example:

location	0	1	2	3	4	5
ℓ_0	t	t	t	t	t	t
ℓ_1	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$
ℓ_3	$f \wedge f$	$t \wedge f$	$t \wedge f$	t	t	t
ℓ_4	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t	t

15. Step: Iteration 5 Blocking-Phase

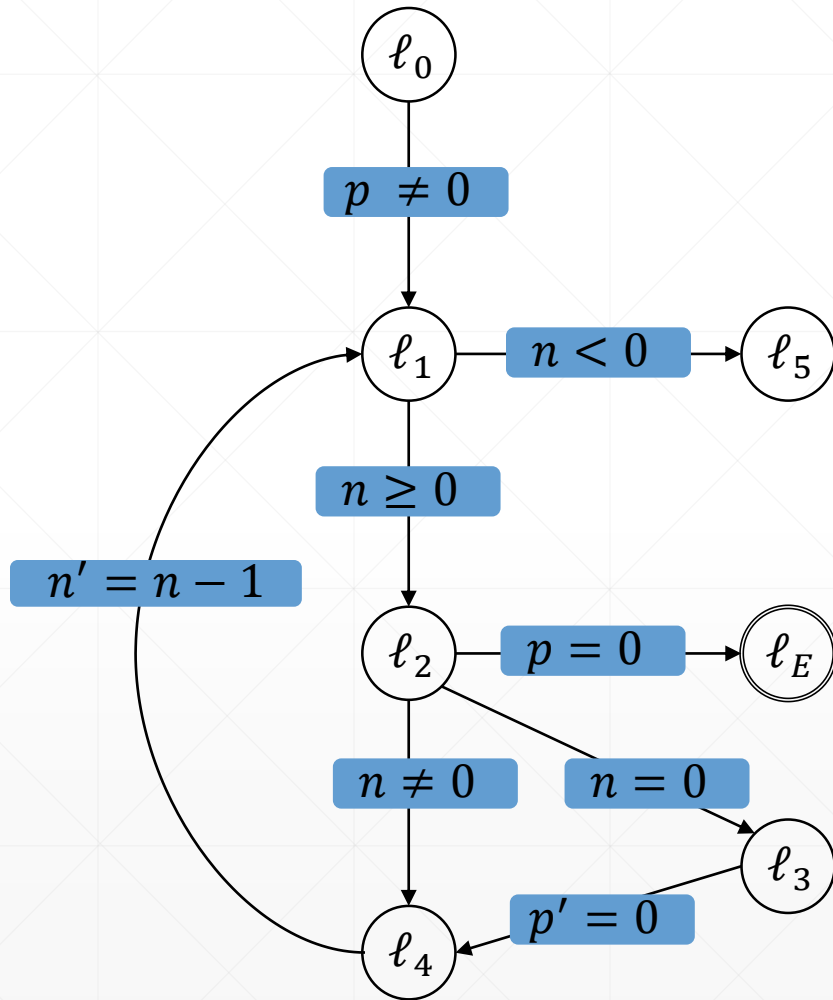# Example:

location	0	1	2	3	4	5
ℓ_0	t	t	t	t	t	t
ℓ_1	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$
ℓ_3	$f \wedge f$	$t \wedge f$	$t \wedge f$	t	t	t
ℓ_4	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t	t

16. Step: Iteration 5 Propagation-Phase

➢ Is there a global fixpoint?

Example:

location	0	1	2	3	4	5
ℓ_0	t	t	t	t	t	t
ℓ_1	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$
ℓ_3	$f \wedge f$	$t \wedge f$	$t \wedge f$	t	t	t
ℓ_4	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t	t

16. Step: Iteration 5 Propagation-Phase

➤ Is there a global fixpoint?

➔ Yes!
   ➔ Algorithm termintes returning that error location is not reachable

PDR Algorithm: Termination

➢ Error location is reachable, if a proof-obligation $(p, \ell, 0)$ is generated

Related Work: Other Approaches

➤ Our Algorithm is based on the approach by Lange et al.[1]

➤ Other possible ways of using PDR on software:

> Bit-Blasting[2]:
> - Encode the variables as bitvectors with new variable pc representing the control-flow
> - Use original bit-level PDR algorithm
> - ➔ Not very competitive because tedious handling of pc variable

1: Tim Lange, Martin R. Neuhäußer, and Thomas Noll. IC3 software model checking on control flow automata. In *FMCAD*, pages 97–104. IEEE, 2015.

2: Tobias Welp and Andreas Kuehlmann. QF BV model checking with property directed reachability. In *DATE*, pages 791–796. EDA Consortium San Jose, CA, USA / ACM DL, 2013.

Related Work: Other Approaches

➢ Our Algorithm is based on the approach by Lange et al.[1]

➢ Other possible ways of using PDR on software:

> ### Abstract Reachability Tree (ART) Unrolling[3]:
> - Transform CFG into an ART
>   - ➔ Attach program-counter variable pc and first-order formula φ to locations
> - Block proof-obligations like in our approach

1: Tim Lange, Martin R. Neuhäußer, and Thomas Noll. IC3 software model checking on control flow automata. In *FMCAD*, pages 97–104. IEEE, 2015.

3:Alessandro Cimatti and Alberto Griggio. Software model checking via IC3. In *CAV*, volume 7358 of *Lecture Notes in Computer Science*, pages 277–293. Springer, 2012.
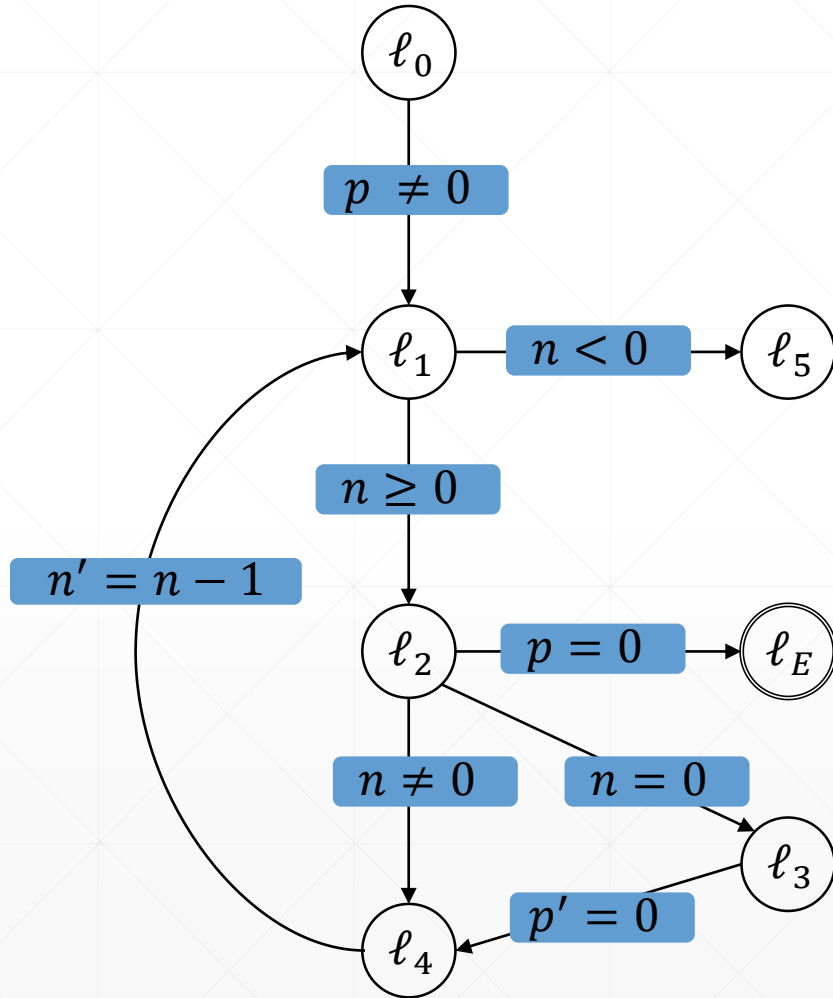
Implementation in Ultimate: Description Trace Abstraction with PDR

1. Calculate sequence of statements from initial location to error location

   ➜ Possible error trace

2. Construct a path program of error trace, by projecting given program to the transitions found in trace

3. Use PDR to show if error is reachable or not

   ➜ If reachable:
   - Error trace is feasible, program is unsafe

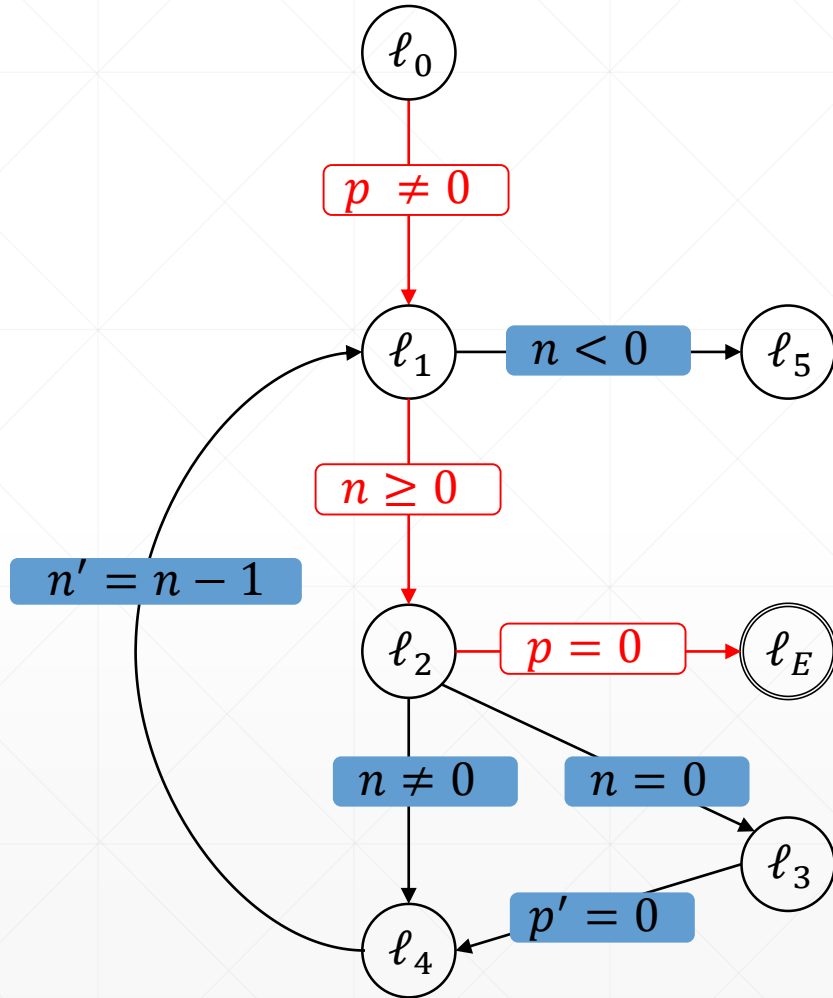# Implementation in Ultimate: Description Trace Abstraction with PDR

1. Calculate sequence of statements from initial location to error location

   ➔ Possible error trace

2. Construct a path program of error trace, by projecting given program to the transitions found in trace

3. Use PDR to show if error is reachable or not

   ➔ If unreachable:
   - Use formulas at the fixpoint as interpolant sequence to refute other error traces

1. Step: Get possible error trace

1. **Step:** Get possible error trace

2. **Step:** Construct Path Program

ℓ_0

$p \neq 0$

ℓ_1

$n \geq 0$

2. Step: Construct Path Program

ℓ_2

$p = 0$

ℓ_E

Implementation in Ultimate: Trace Abstraction with PDR

ℓ_0

$p \neq 0$

ℓ_1

$n \geq 0$

ℓ_2

$p = 0$

ℓ_E

3. Step: Use PDR

location	0	1	2	3
ℓ_0				
ℓ_1				
ℓ_2				

ℓ_0

$p \neq 0$

ℓ_1

$n \geq 0$

ℓ_2

$p = 0$

ℓ_E

3. Step: Use PDR

location	0	1	2	3
ℓ_0	t	t	t	t
ℓ_1	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$
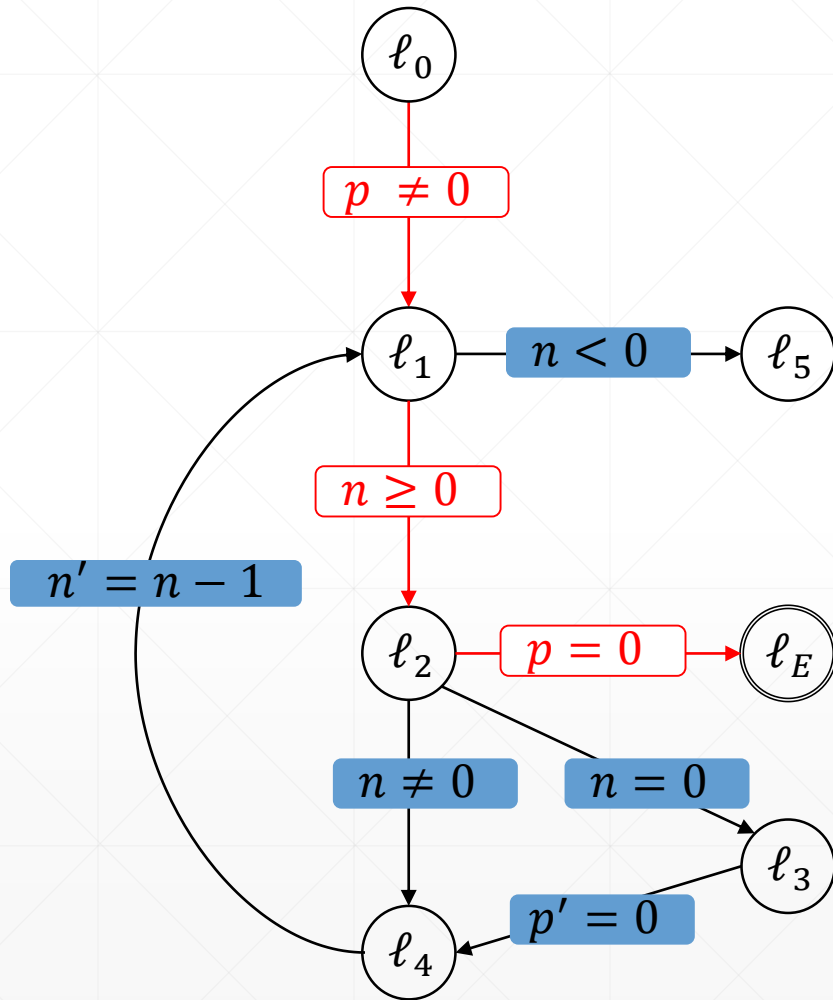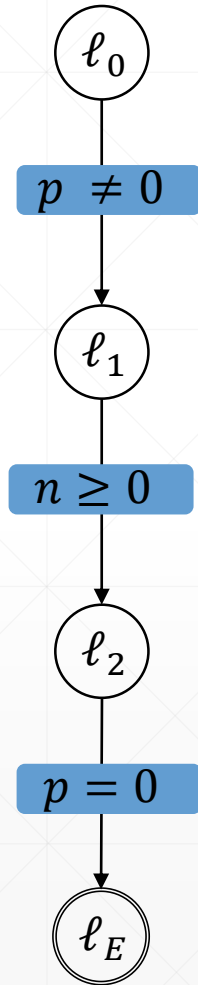
3. Step: Use PDR

Implementation in Ultimate: Trace Abstraction with PDR
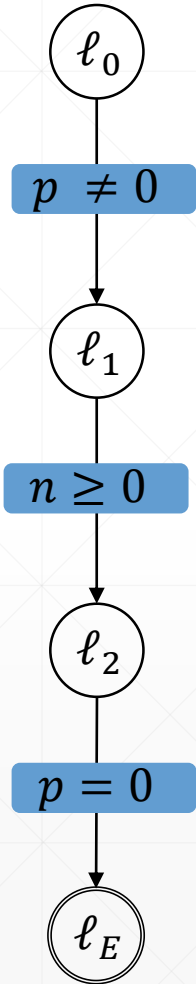
location	0	1	2	3
ℓ_0	t	t	t	t
ℓ_1	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$

ℓ_0

$p \neq 0$

ℓ_1

$n \geq 0$

ℓ_2

$p = 0$

ℓ_E

4. Step: Use fixpoint invariants as
          interpolant sequence

location	0	1	2	3
ℓ_0	t	t	t	t
ℓ_1	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	t
ℓ_2	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$

ℓ_0

$p \neq 0$

ℓ_1

$n \geq 0$

ℓ_2

$p = 0$

ℓ_E

4. Step: Use fixpoint invariants as interpolant sequence

Implementation in Ultimate: Implemented Improvements

Caching proof-obligations:

- Cache the proof-obligation queue

- Start every new Iteration with the latest blocked proof-obligation

➜ Only proof-obligation that differs from Iteration before

This chain of obligations is always the same on each new level

Initial Obligation

1. Obligation: generated by Initial

2. Obligation: generated by 1.

Newest Obligation: Generated by 2.

Initial Obligation: Blocked

1. Obligation: Blocked

2. Obligation: Blocked
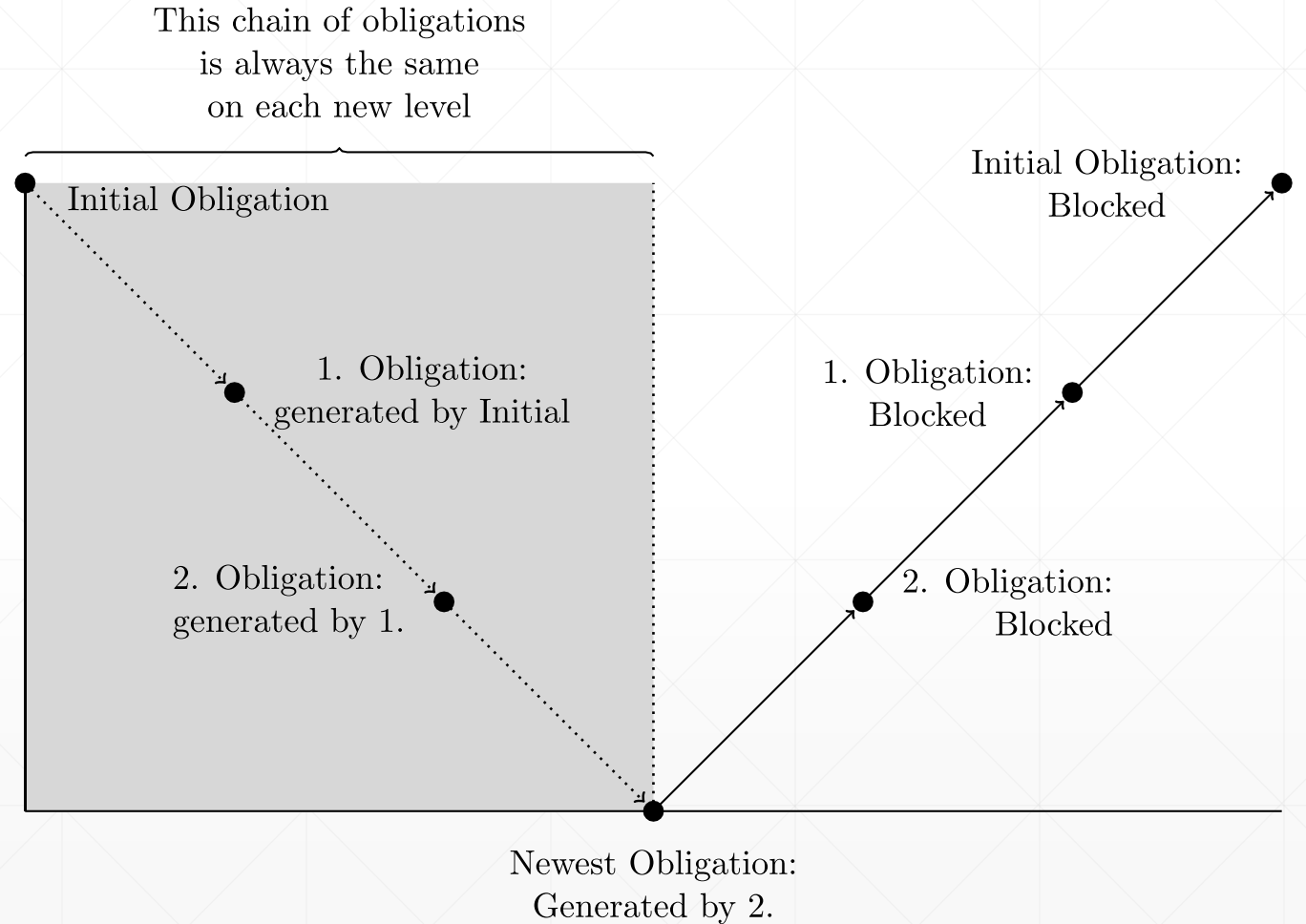
Implementation in Ultimate: Implemented Improvements

Skipping already blocked proof-obligations:

- Cache unsatisfiable queries to SMT-solver
  - ➔ When a query to the SMT-solver is proven unsatisfiable, cache it
  - ➔ If a cached query is seen again, do not call SMT-solver again, strengthen frames right away

Evaluation: Introduction

➢ We compared Trace Abstraction using PDR with Trace Abstraction using Nested Interpolants

➢ Tested on Ultimate version 0.1.23-e6fd87c, time limit: 300s, memory limit: 8000MB

Evaluation: Introduction

➢ We compared Trace Abstraction using PDR with Trace Abstraction using Nested Interpolants

➢ Tested on Ultimate version 0.1.23-e6fd87c, time limit: 300s, memory limit: 8000MB

➢ Benchmarkset contained 250 Boogie[1] Programs

- 31 real-life code

- 40 programs without disjunctions

- 134 difficult programs that could not be solved in three iterations

- 37 programs with difficult loop invariants

- 8 non-linear arithmetic

Evaluation: Data Comparison

CPU Time Consumption in Seconds
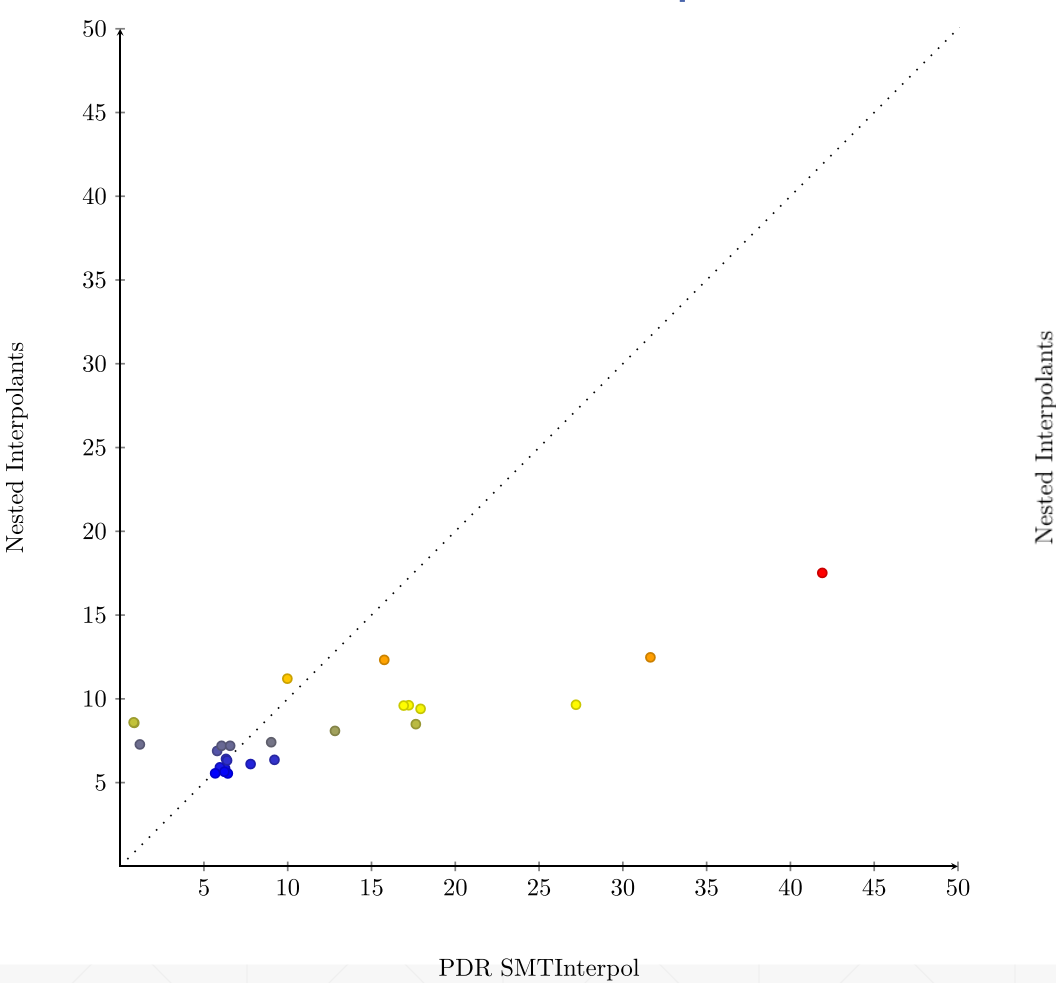
Memory Consumption in Bit

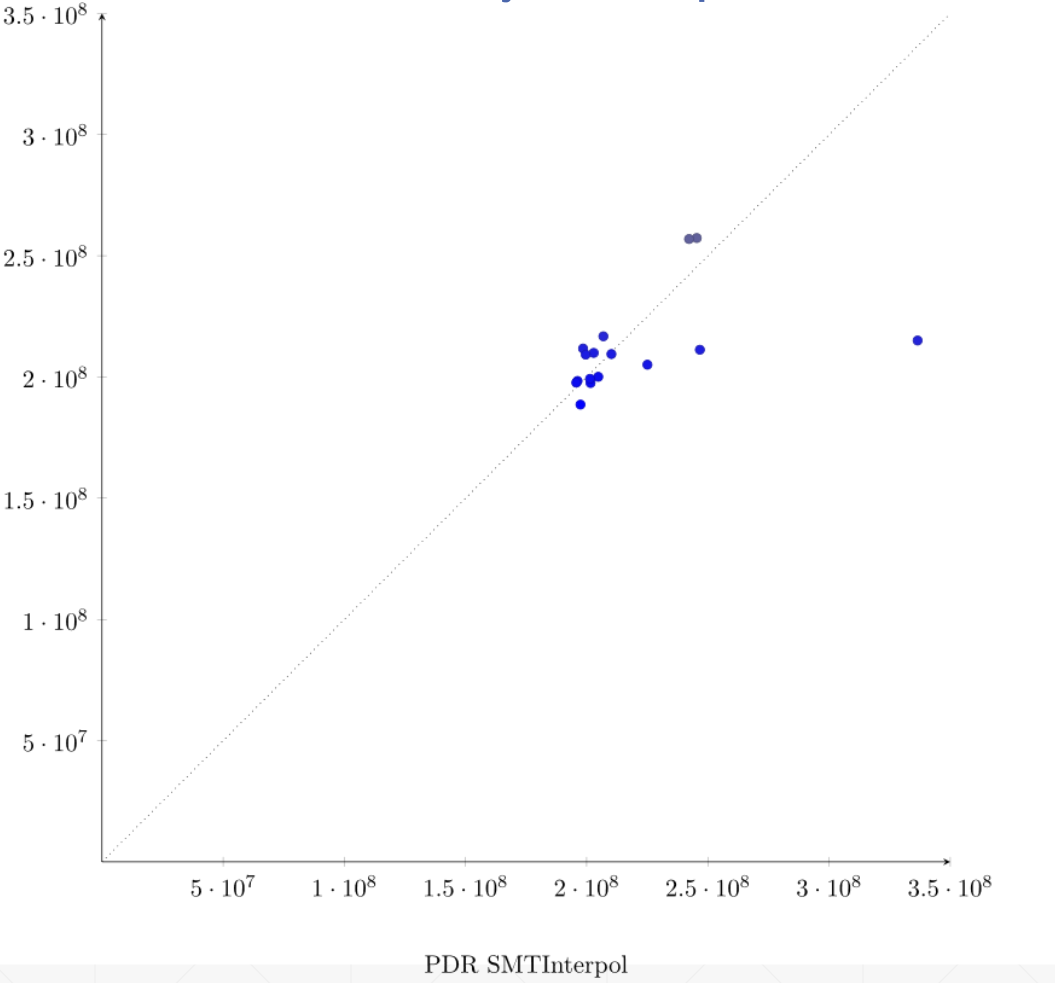|	Nested Interpolants	PDR SMTInterpol	PDR Z3
Tests Solved	179/250	49/250	62/250
Solve Time	3543s	575s	1332s
Timeouts	65	90	133
Exceptions	6	111	55

real-life

	Nested Interpolants	PDR SMTInterpol	PDR Z3
Tests Solved	20/31	3/31	9/31
Solve Time	598s	8s	76s
Timeouts	11	10	14
Exceptions	0	18	8

20170319-ConjunctivePathPrograms

	Nested Interpolants	PDR SMTInterpol	PDR Z3
Tests Solved	29/40	6/40	16/40
Solve Time	531s	35s	191s
Timeouts	11	15	20
Exceptions	0	19	4

20170304-DifficultPathPrograms

	Nested Interpolants	PDR SMTInterpol	PDR Z3
Tests Solved	105/134	24/134	24/134
Solve Time	1435s	449s	975s
Timeouts	24	44	74
Exceptions	5	66	36

tooDifficultLoopInvariants

	Nested Interpolants	PDR SMTInterpol	PDR Z3
Tests Solved	17/37	8/37	8/37
Solve Time	944s	42s	57s
Timeouts	19	21	22
Exceptions	1	8	7

nonlinear

	Nested Interpolants	PDR SMTInterpol	PDR Z3
Tests Solved	8/8	8/8	5/8
Solve Time	35s	41s	33s
Timeouts	0	0	3
Exceptions	0	0	0

Evaluation: Discussion

PDR with SMTInterpol:

- 90 Timeouts, mostly due to loops

- 111 Exceptions:

  ➔ 16 Syntax Exceptions

  ➔ 95 Exceptions due to exist quantifier

Evaluation: Discussion

PDR with z3:

- 131 Timeouts, mostly due to loops

- 55 Exceptions:
  - ➜ 48 Solver returned unknown
  - ➜ 2 overapproximation Exceptions
  - ➜ 2 Unsupported Operation Exception
  - ➜ 1 z3-Internal Exception
  - ➜ 1 Procedure Exception

Evaluation: Discussion

	Nested Interpolants	PDR
Exclusively solved	116	13

Evaluation: Discussion

	Nested Interpolants	PDR
Exclusively solved	116	13

13 programs were exclusively solved by PDR
➔ Timed out with Nested Interpolants
➔ PDR solved them in 2 iterations each

Future Work: Implementing Further Improvements

➢ **Using Interpolation:**

- Our algorithm is inefficient when dealing with loops

Idea:

- Instead of strengthening frames with negated proof-obligation, calculate Interpolant for transition and proof-obligation and add that

Future Work: Implementing Further Improvements

➢ **Dealing with procedures:**

- C programs often contain procedures with which PDR cannot deal

Ideas:

1. Use a non-linear approach of PDR
2. Calculate a procedure summary, add that to the CFG, removing the procedure altogether

Conclusion

➢ We have seen:

- How PDR works on software

- How we combined Trace Abstraction and PDR

- How the combination compared to Trace Abstraction with Nested Interpolants

- What can be done to make it more efficient

Bibliography

➢ Aaron R. Bradley. Sat-based model checking without unrolling. In *VMCAI*, volume 6538 of *Lecture Notes in Computer Science*, pages 70–87. Springer, 2011.

➢ Hwmcc10 results. https://fmv.jku.at/hwmcc10/results.html. Accessed: 2018-07-20

➢ Niklas Een, Alan Mishchenko, and Robert Brayton. 2011. Efficient implementation of property directed reachability. In Proceedings of the International Conference on Formal Methods in Computer-Aided Design (FMCAD '11). FMCAD Inc, Austin, TX, 125-134.

➢ Tim Lange, Martin R. Neuhäußer, and Thomas Noll. IC3 software model checking on control flow automata. In *FMCAD*, pages 97–104. IEEE, 2015.

➢ Tobias Welp and Andreas Kuehlmann. QF BV model checking with property directed reachability. In *DATE*, pages 791–796. EDA Consortium San Jose, CA, USA / ACM DL, 2013.

➢ Alessandro Cimatti and Alberto Griggio. Software model checking via IC3. In *CAV*, volume 7358 of *Lecture Notes in Computer Science*, pages 277–293. Springer, 2012.

➢ Ultimate. https://ultimate.informatik.uni-freiburg.de. Accessed: 2018-07-20.

➢ https://www.microsoft.com/en-us/research/project/boogie-an-intermediate-verification-language/