Introduction: Motivation

- 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

Introduction: Motivation

- 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

- 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

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

- A control flow graph (CFG) $A = (X, L, E, \ell_0, \ell_E)$ is a graph consisting of
  - A finite set of first-order variables X
  - A finite set of locations L
  - A finite set of transitions E $\subseteq L \times FO \times L$

    ➔ FO is a quantifier free first-order logic formula over variables in X and $X' = \{x \in X \mid x' \in X'\}$
  - An initial location $\ell_0 \in L$
  - An 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 level

  ➔ 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 level

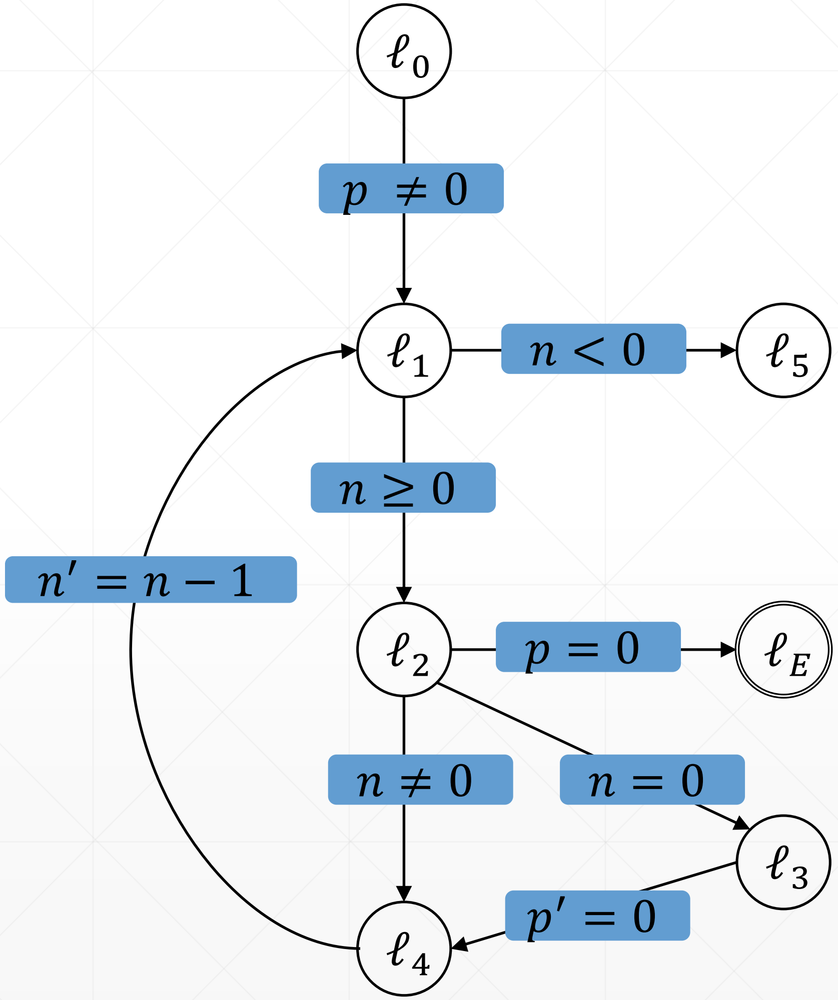
  ➔ Need to be blocked

PDR Algorithm: Description

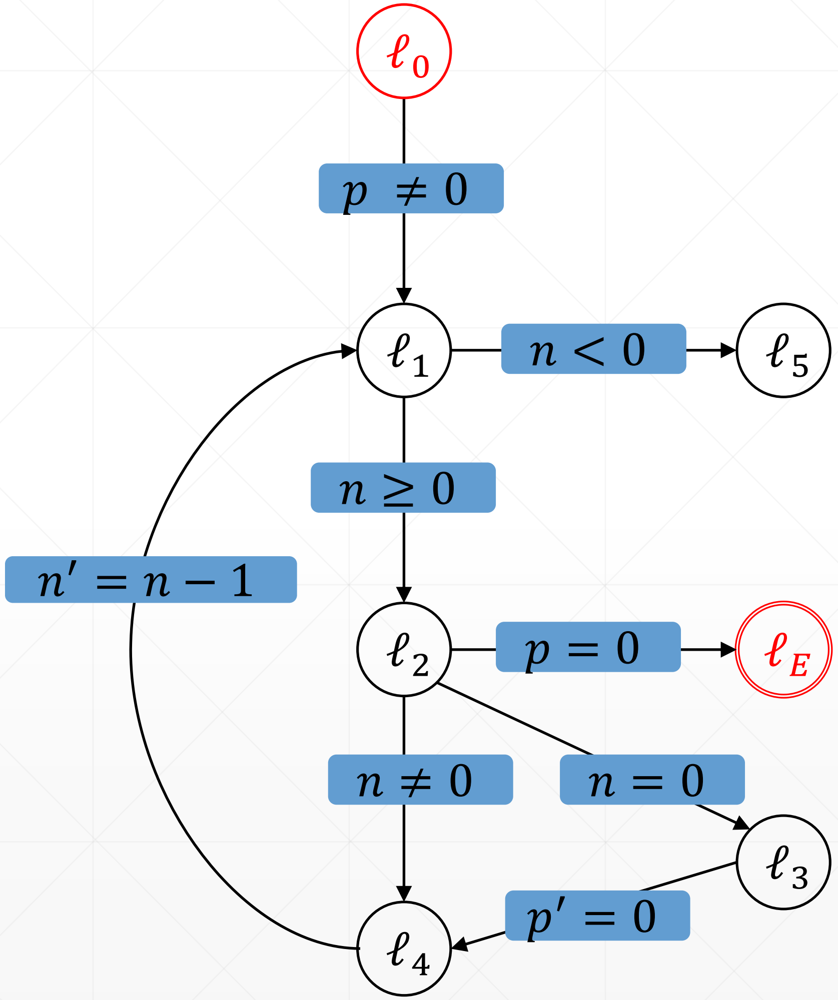
- Starts with checking for a 0-Counter-Example

- Repeats three phases until termination:
  1. Next Level Initialization Phase
  2. Blocking-Phase
  3. Propagation-Phase

Example: Running Example



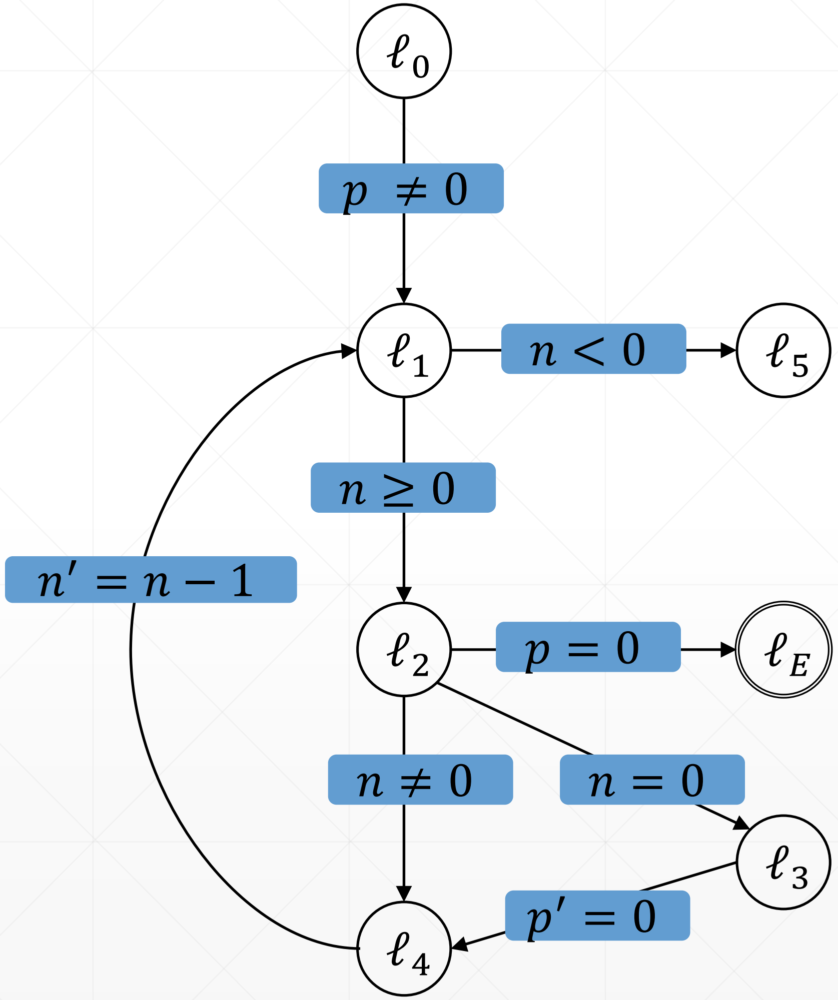
Example:



1. Step: Check for 0-Counter-Example

- Is $\ell_0 = \ell_E$?
  - ➔ No, continue with initialization

Example:

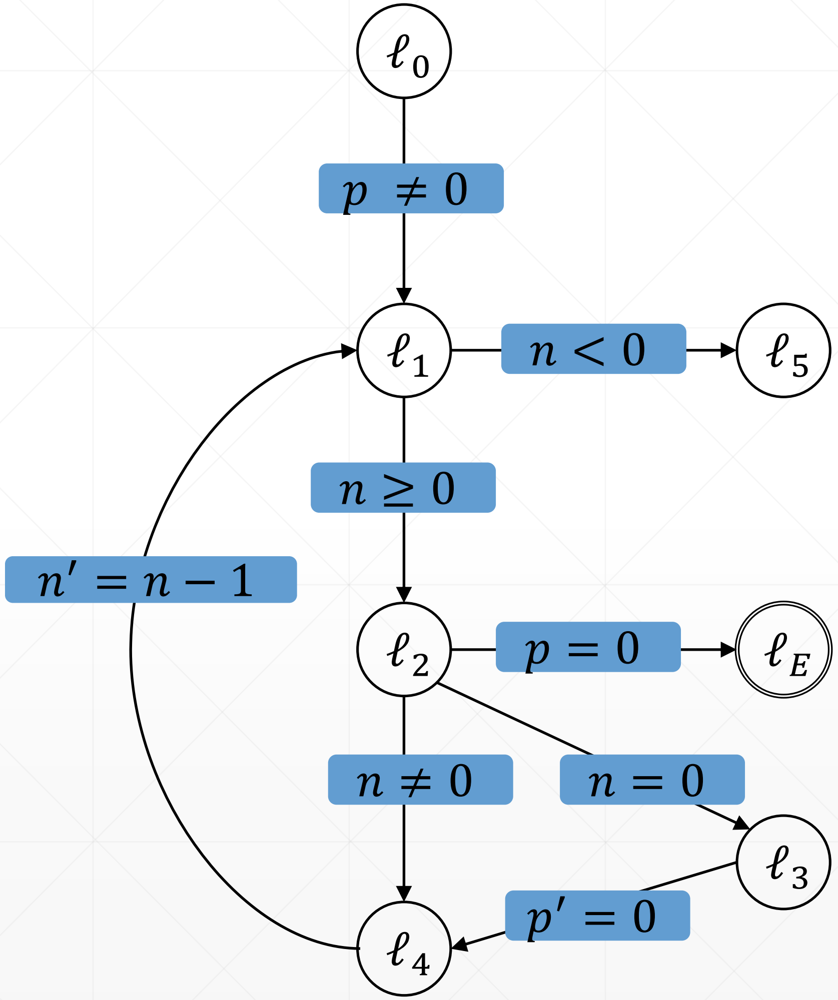


location	0
ℓ_0	
ℓ_1	
ℓ_2	
ℓ_3	
ℓ_4	

2. Step: Initialization of level 0

- $F_{0,\ell} = \begin{cases} \mathrm{T}, & \ell = \ell_0 \\ \mathrm{F}, & otherwise \end{cases}$

Example:

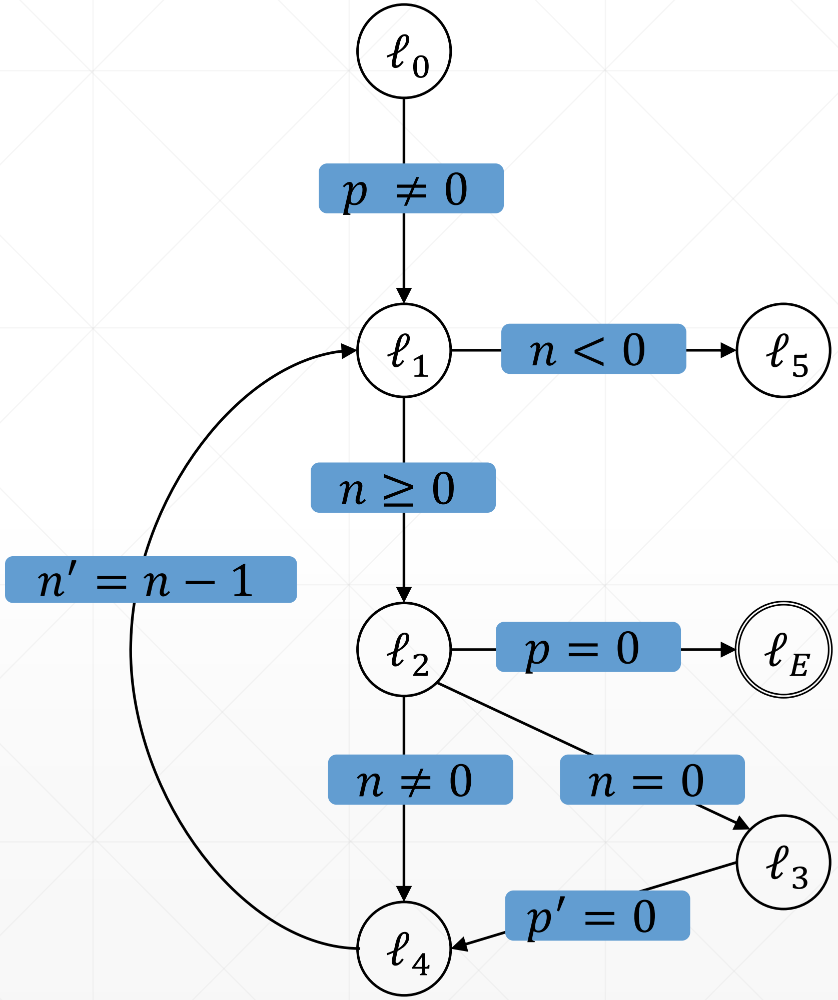


location	0
ℓ_0	t
ℓ_1	f
ℓ_2	f
ℓ_3	f
ℓ_4	f

2. Step: Initialization of level 0

- $F_{0,\ell} = \begin{cases} \mathrm{T}, & \ell = \ell_0 \\ \mathrm{F}, & otherwise \end{cases}$

Example:

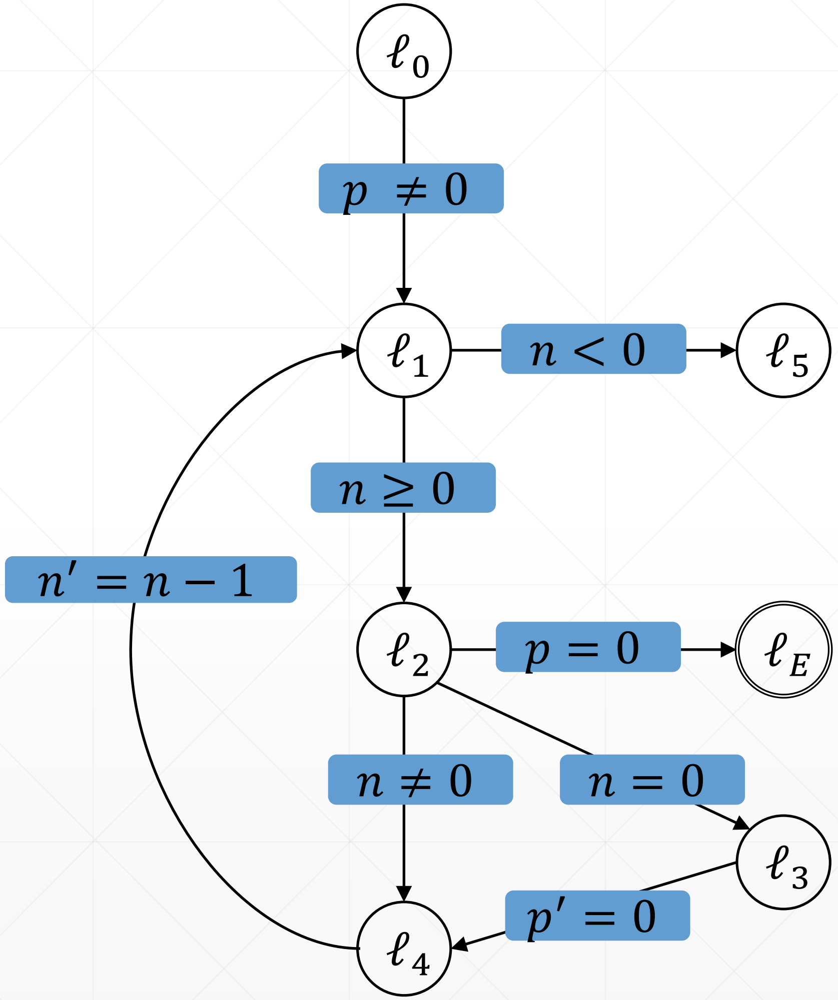


location	0	1
ℓ_0	t	
ℓ_1	f	
ℓ_2	f	
ℓ_3	f	
ℓ_4	f	

3. Step: Level 1

- Initialize level 1 frames as true

Example:

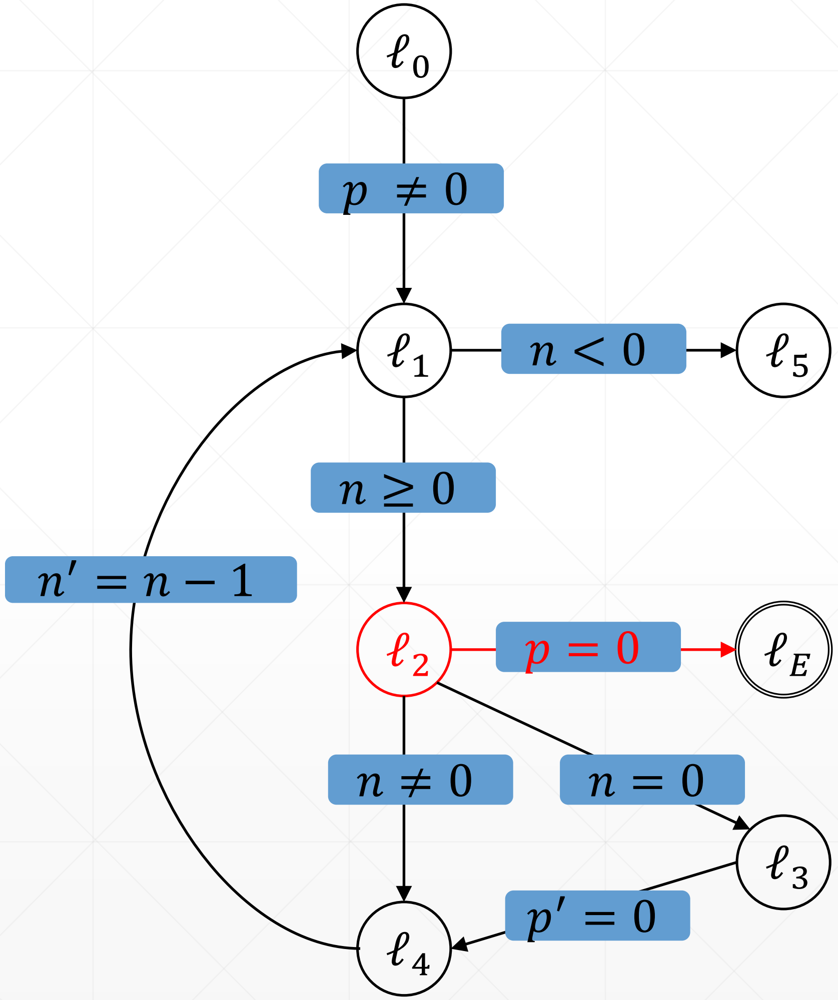


location	0	1
ℓ_0	t	t
ℓ_1	f	t
ℓ_2	f	t
ℓ_3	f	t
ℓ_4	f	t

3. Step: Level 1

- Initialize level 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

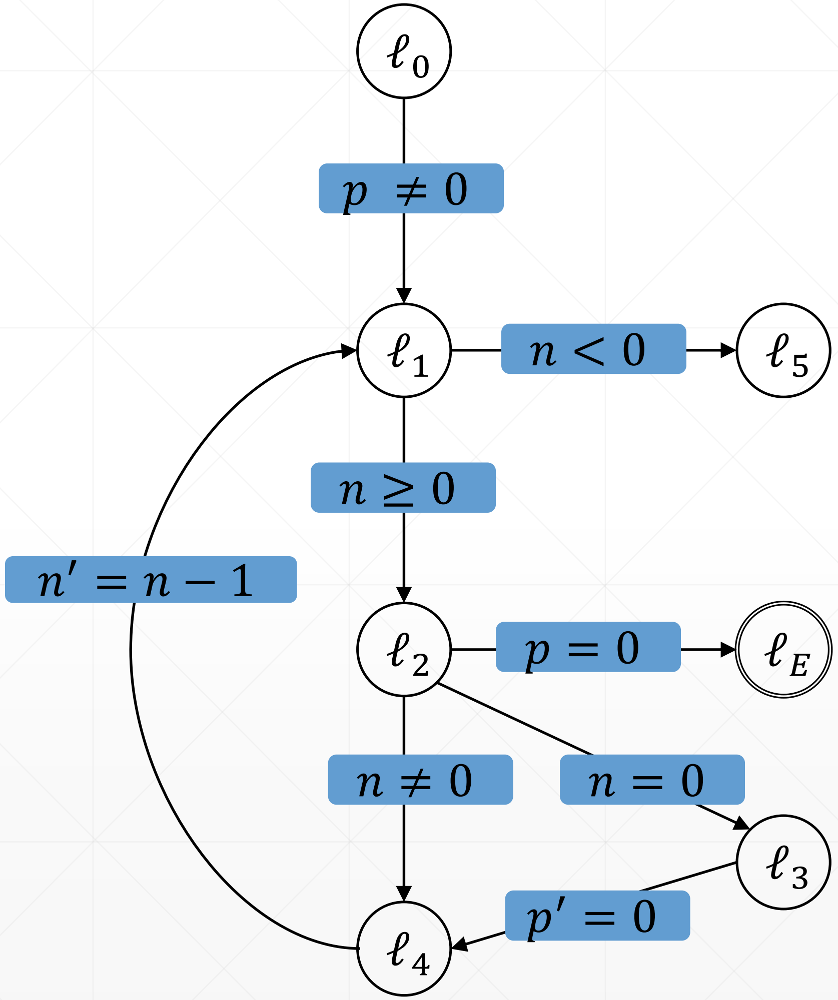
3. Step: Level 1

- Get initial proof-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

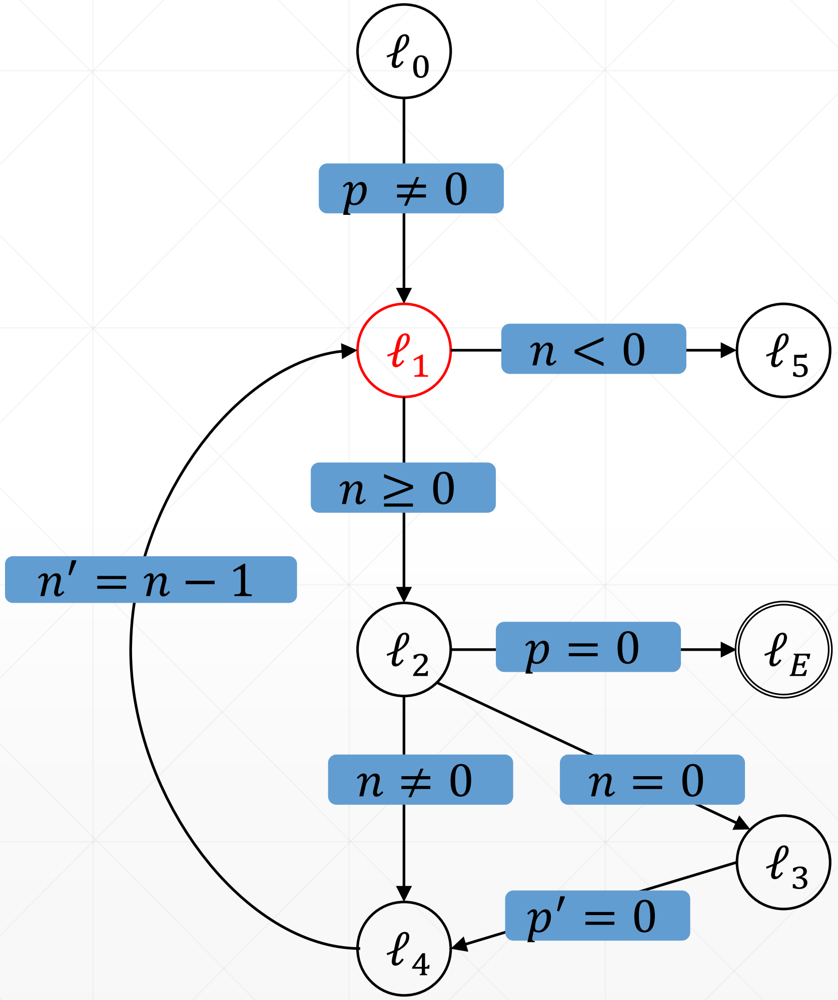
4. Step: Level 1 Blocking-Phase:

- 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: Level 1 Blocking-Phase:

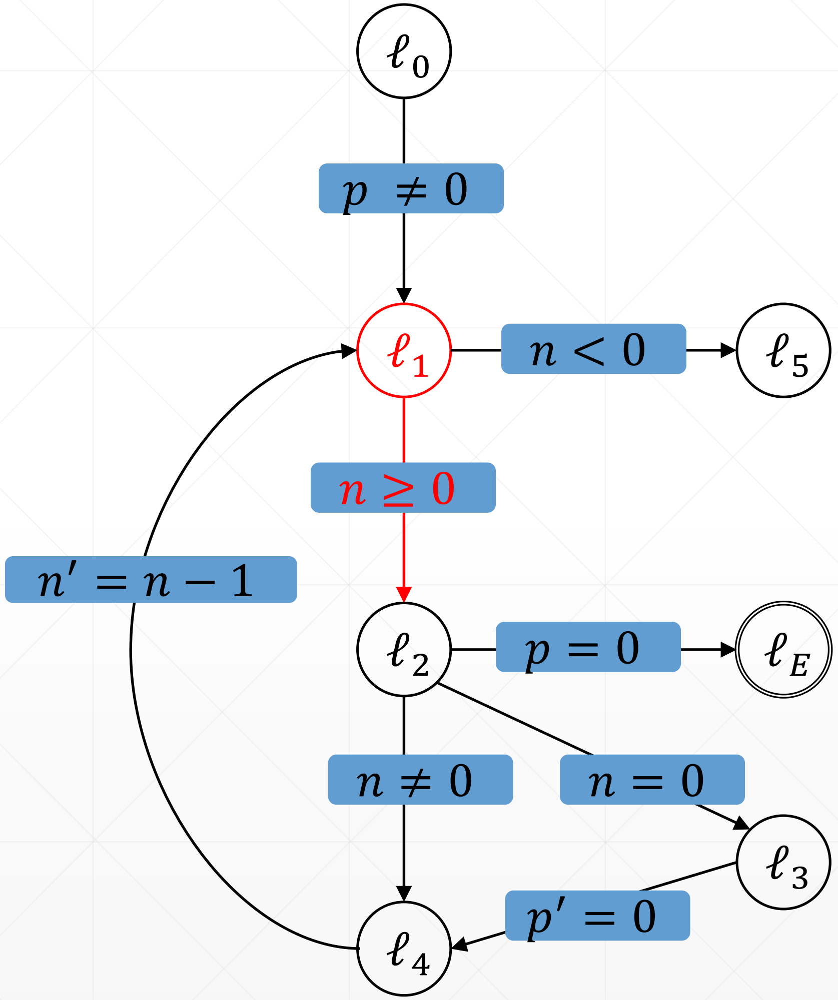
- Try to block $(p = 0, \ell_2, 1)$

- Predecessor ℓ_1:
  - $F_{0,\ell_1} \wedge T_{\ell_1 \to \ell_2} \wedge p' = 0$

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: Level 1 Blocking-Phase:

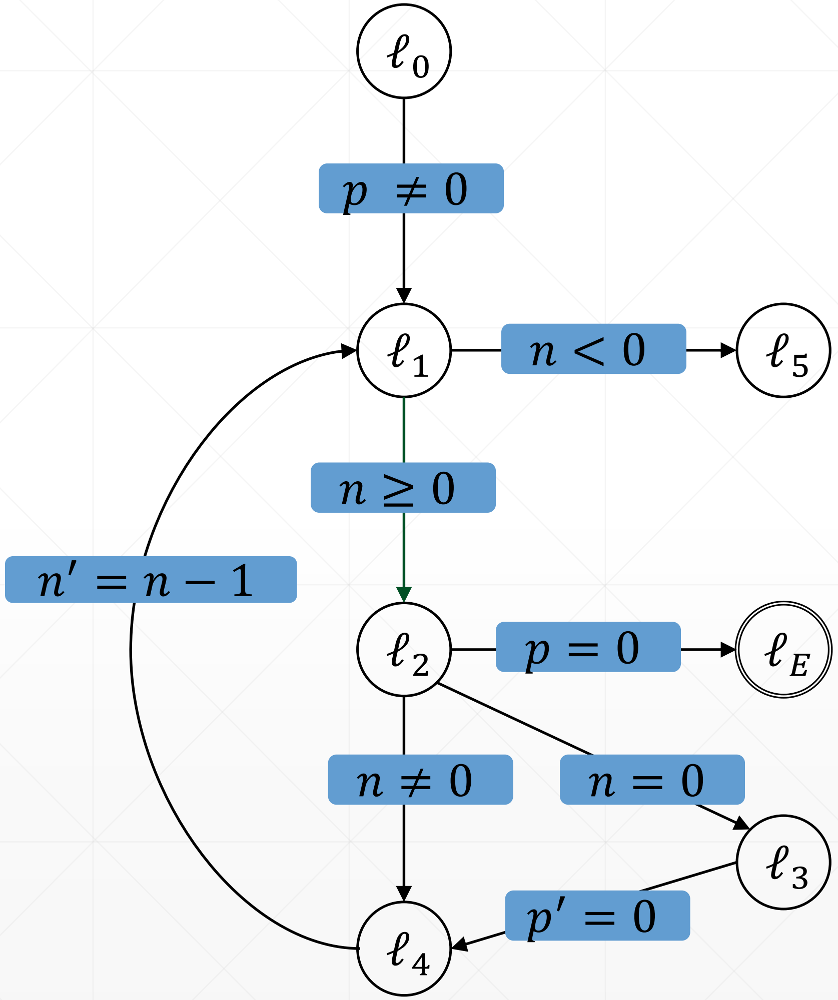
➢ Try to block $(p = 0, \ell_2, 1)$

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

Proof-Obligations:

- $(p = 0, \ell_2, 1)$

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: Level 1 Blocking-Phase:

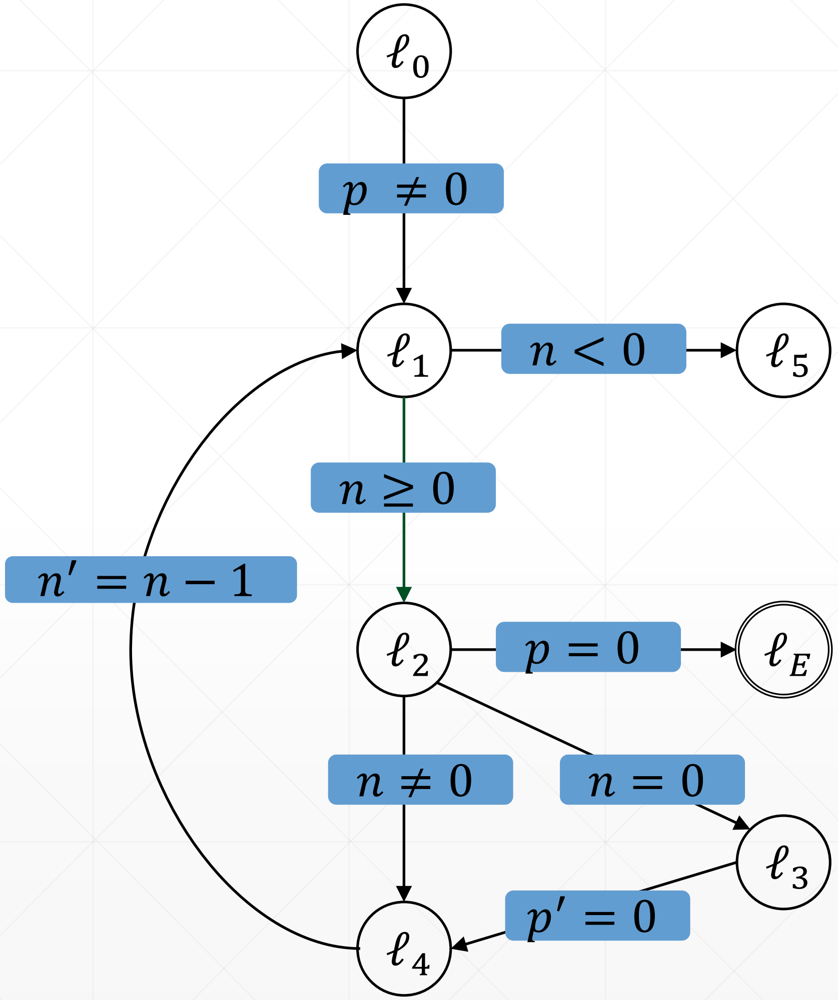
➢ Try to block $(p = 0, \ell_2, 1)$

- 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

4. Step: Level 1 Blocking-Phase:

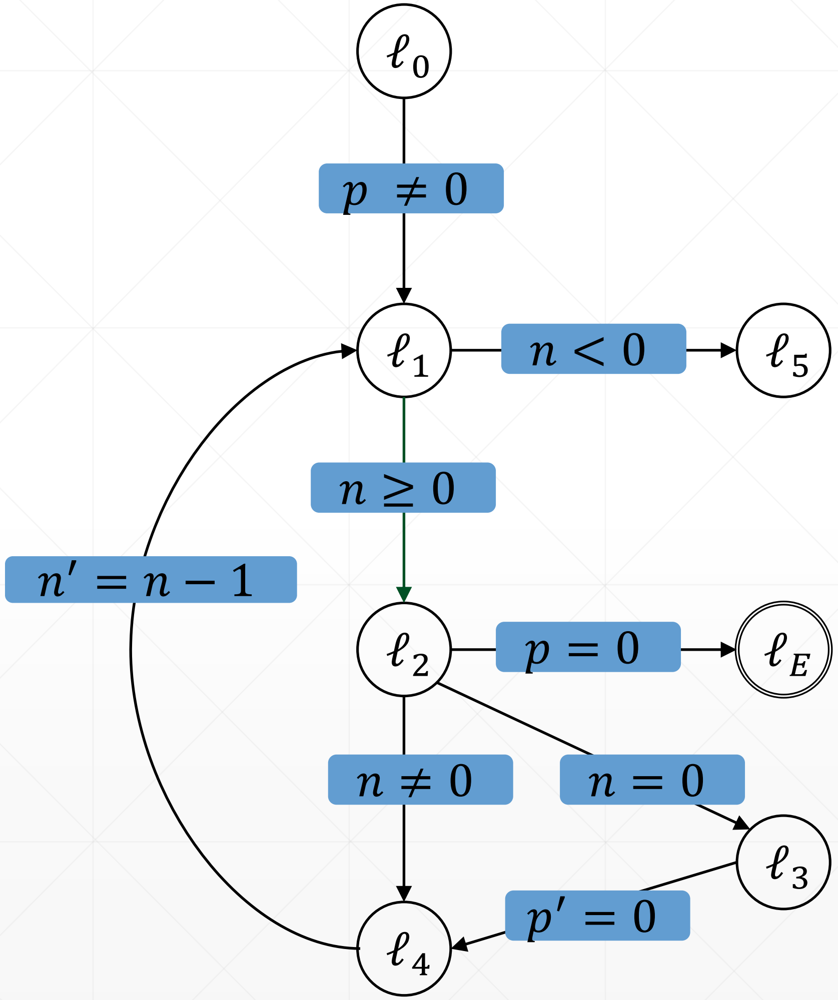
➢ Try to block $(p = 0, \ell_2, 1)$

- 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

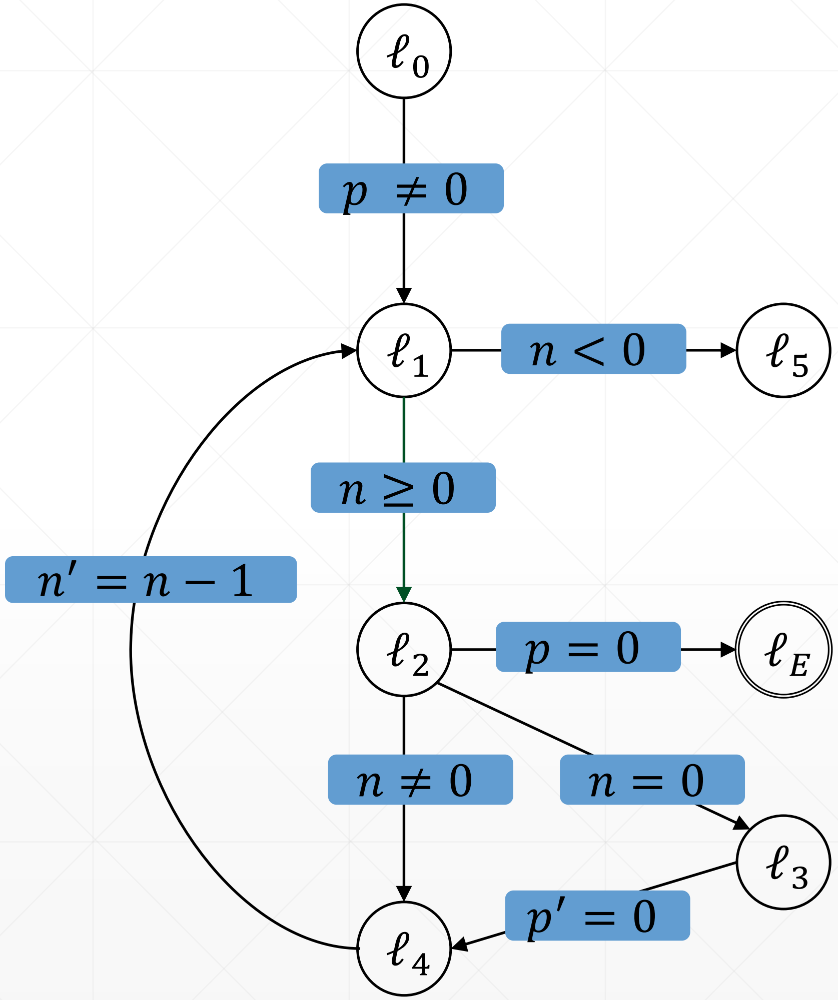
5. Step: Level 1 Propagation-Phase

- 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

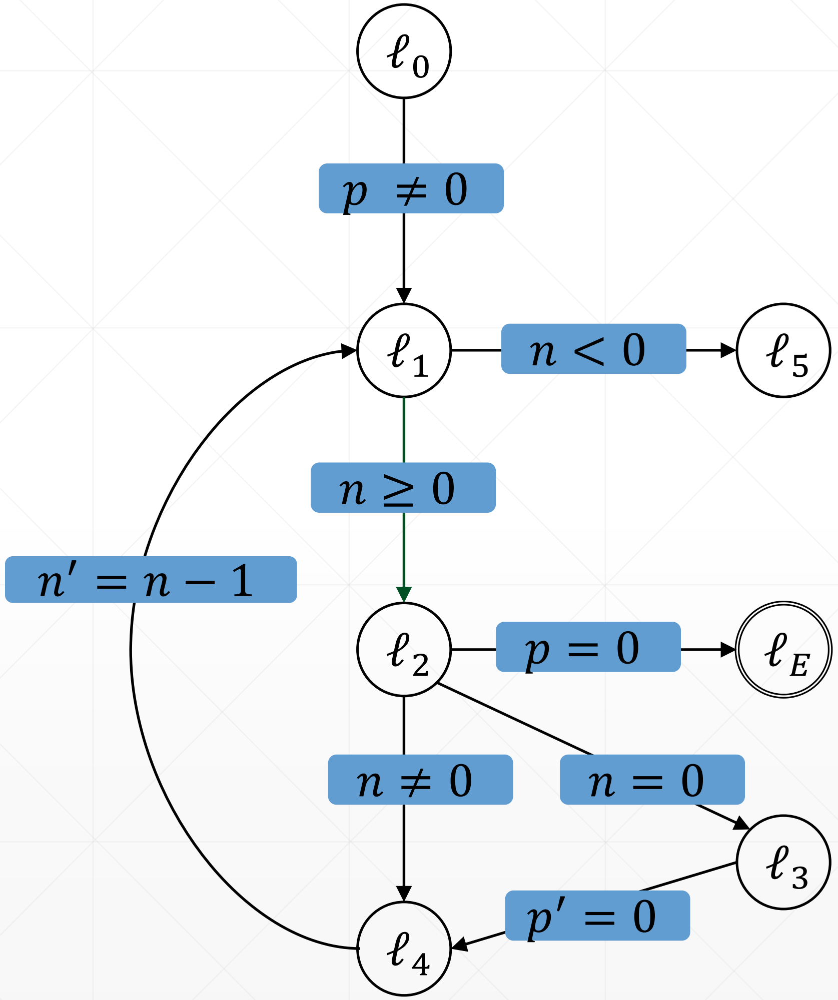
5. Step: Level 1 Propagation-Phase

➢ 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: Level 1 Propagation-Phase

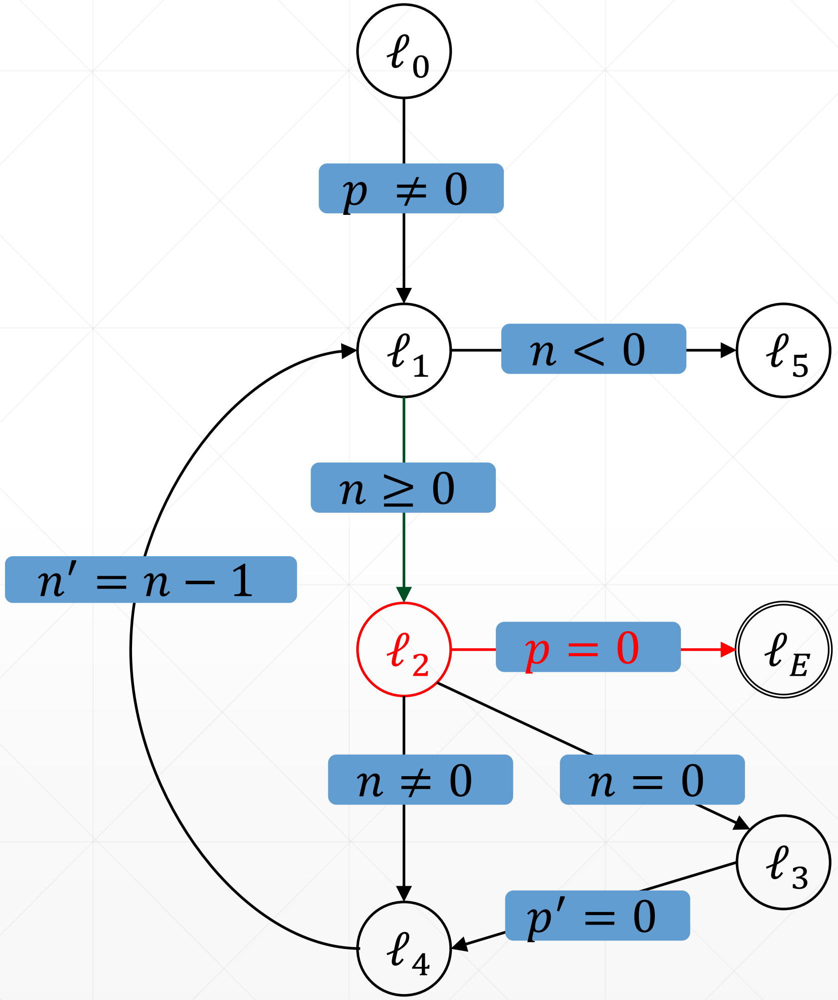
➢ Is there an i where $F_{i-1,\ell} = F_{i,\ell}$ for $\ell \in L \setminus \{\ell_E\}$?

➔ No. Continue with next level.

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

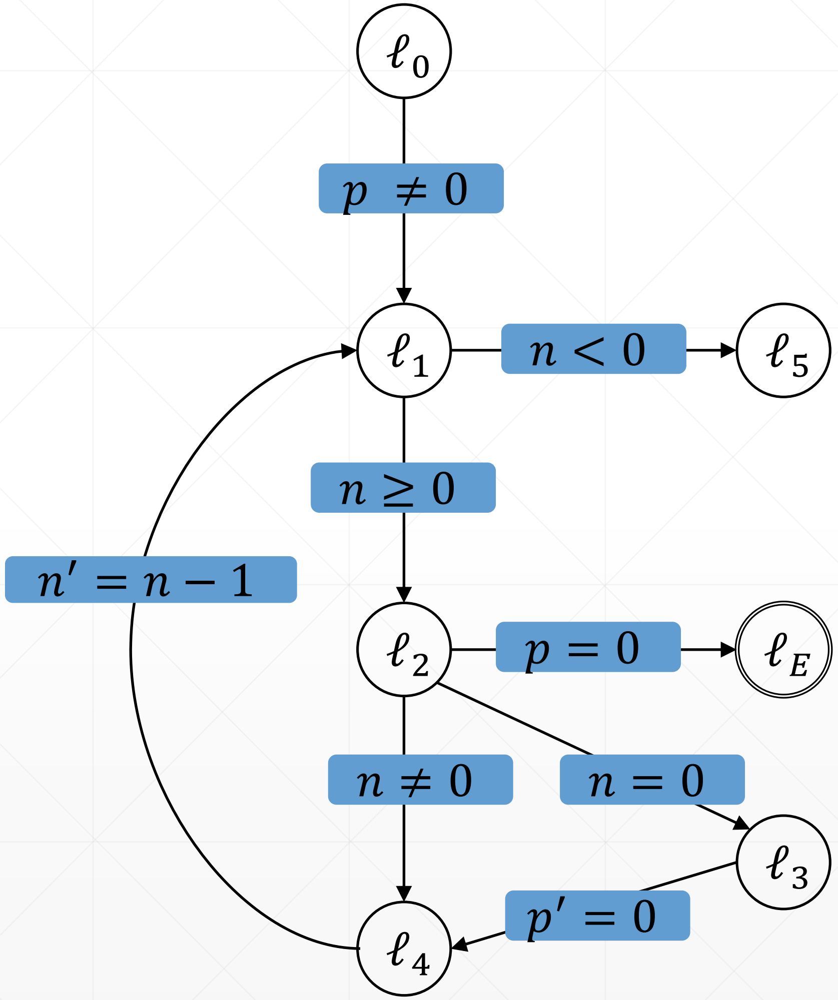
6. Step: Level 2

- Initzialize 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

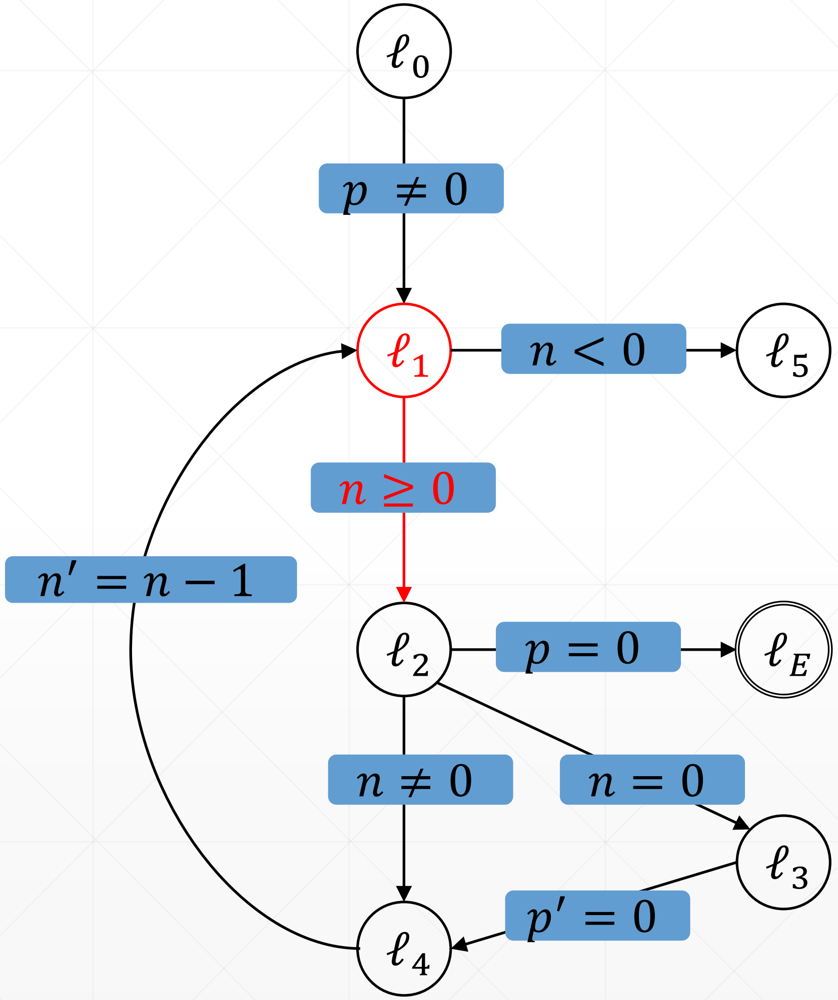
6. Step: Level 2

- Initzialize new frames
- Add initial proof-obligation $(p = 0, \ell_2, 2)$

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: Level 2 Blocking-Phase:

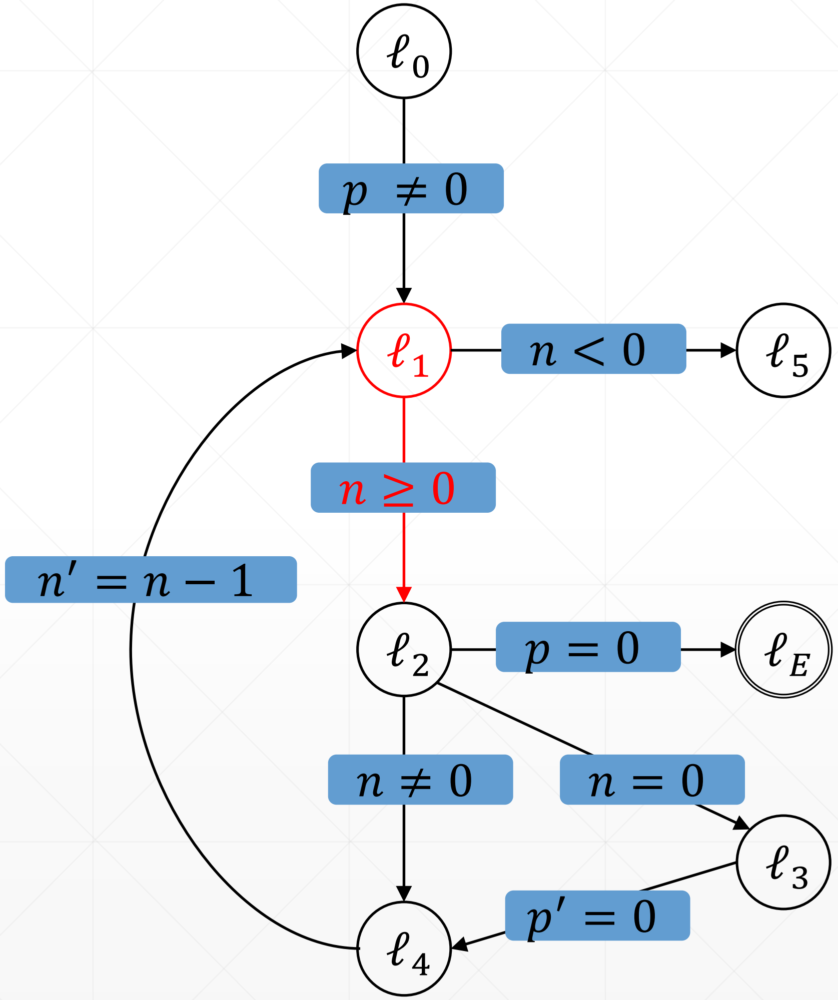
- Try to block $(p = 0, \ell_2, 2)$

- 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: Level 2 Blocking-Phase:

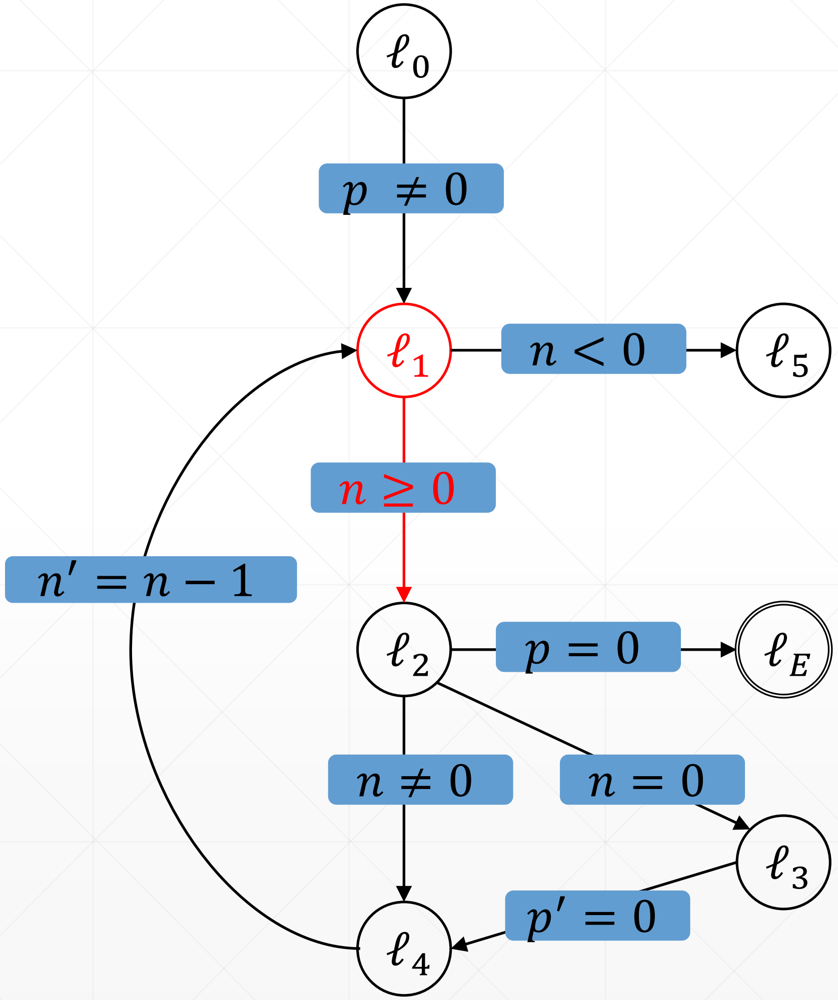
➢ Try to block $(p = 0, \ell_2, 2)$

- 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: Level 2 Blocking-Phase:

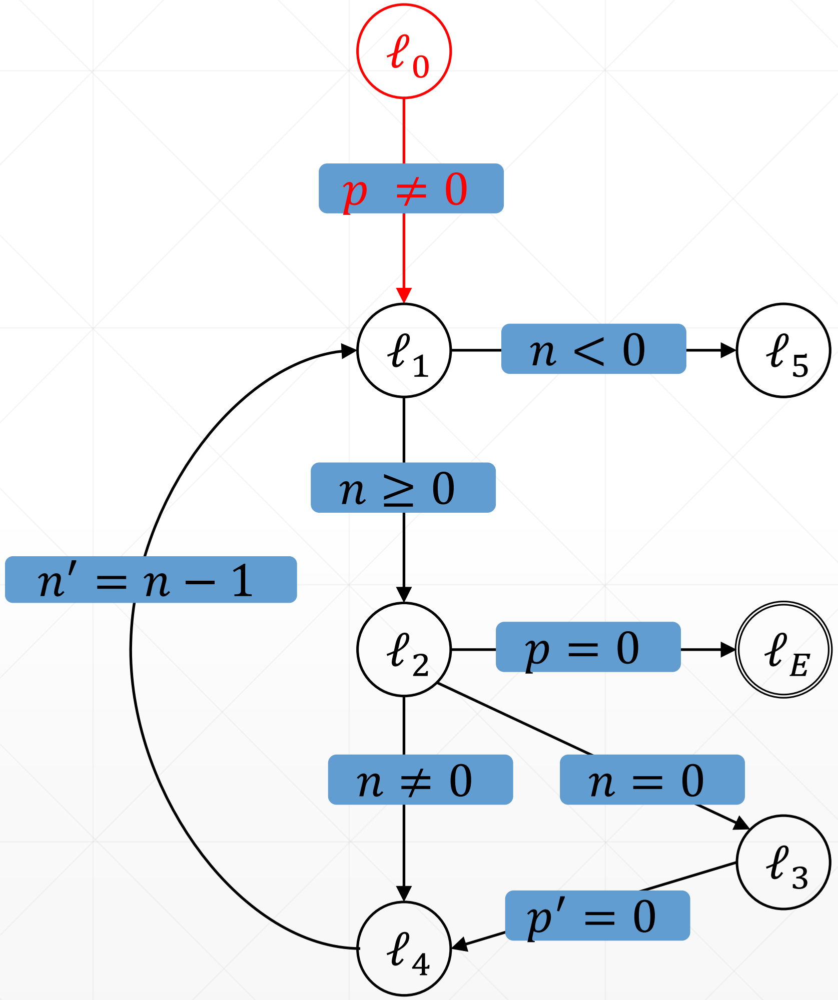
- Try to block $(p = 0, \ell_2, 2)$

- 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: Level 2 Blocking-Phase:

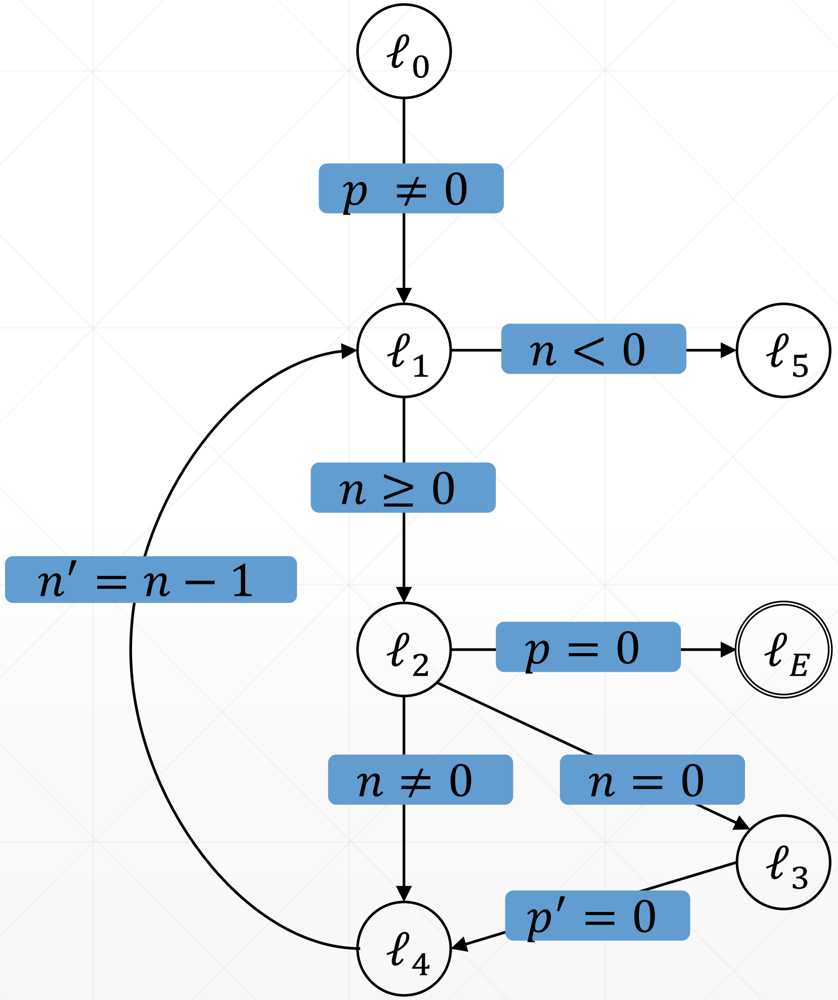
- Try to block $(p = 0, \ell_1, 1)$

- 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: Level 2 Blocking-Phase:

- Try to block $(p = 0, \ell_1, 1)$

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

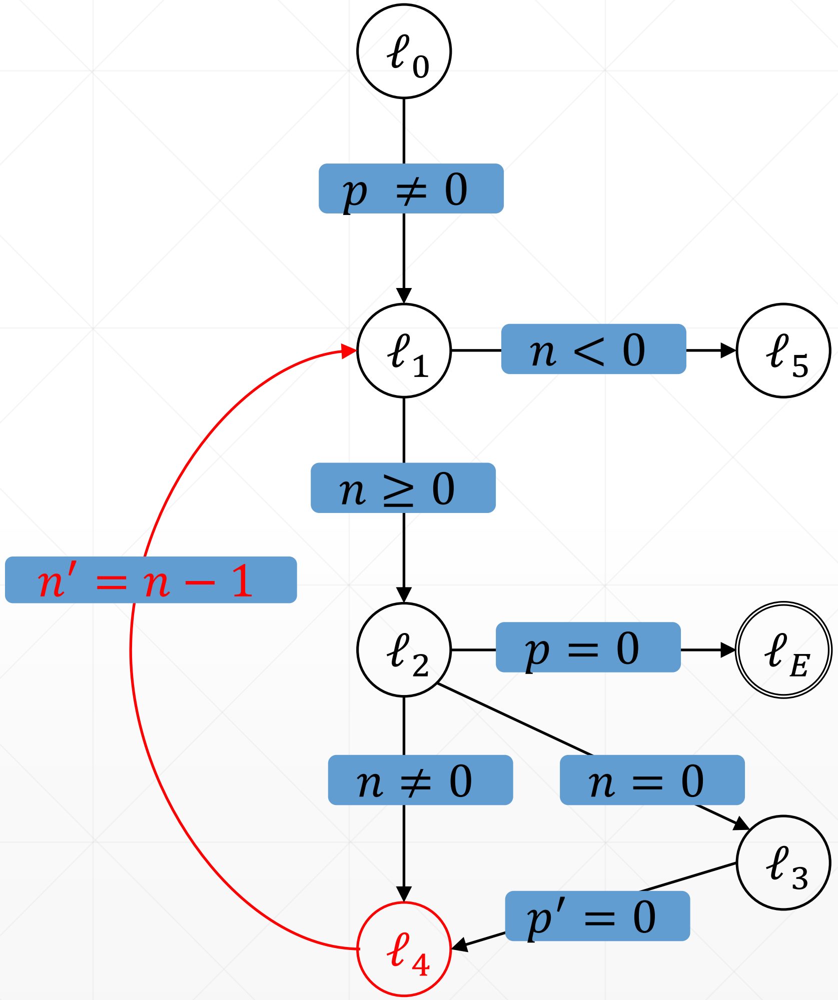
  ➔ Unsatisfiable!

  ➔ 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

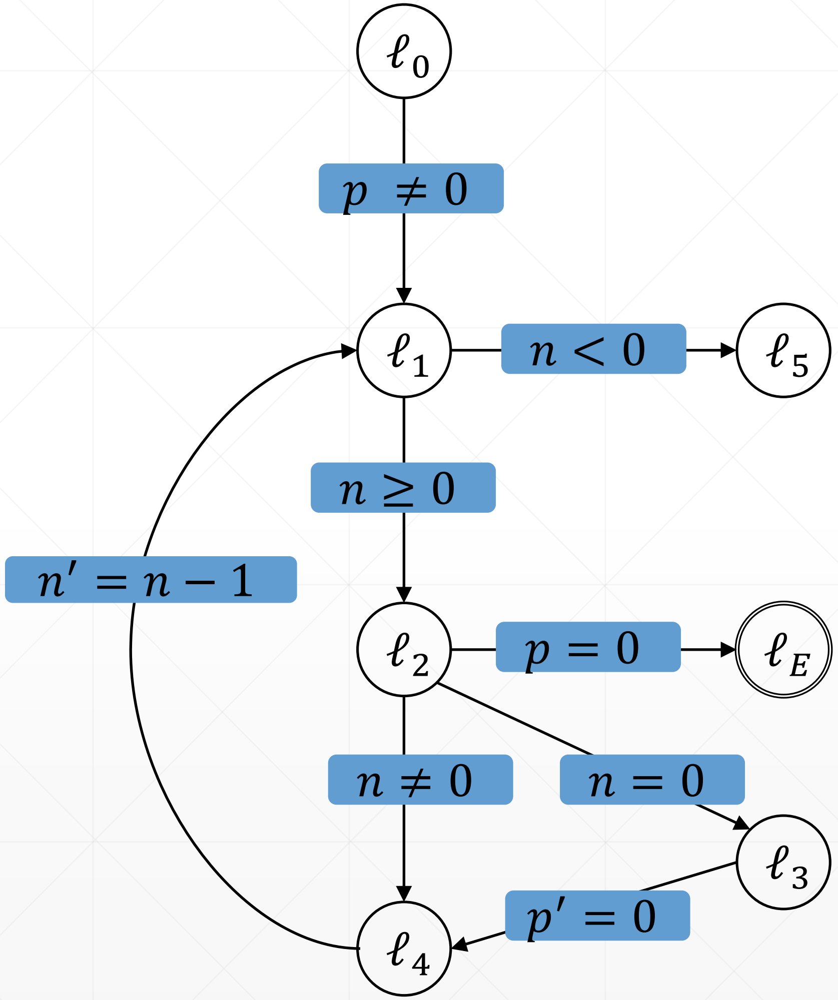
7. Step: Level 2 Blocking-Phase:

- Try to block $(p = 0, \ell_1, 1)$

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

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

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: Level 2 Blocking-Phase:

- Try to block $(p = 0, \ell_1, 1)$

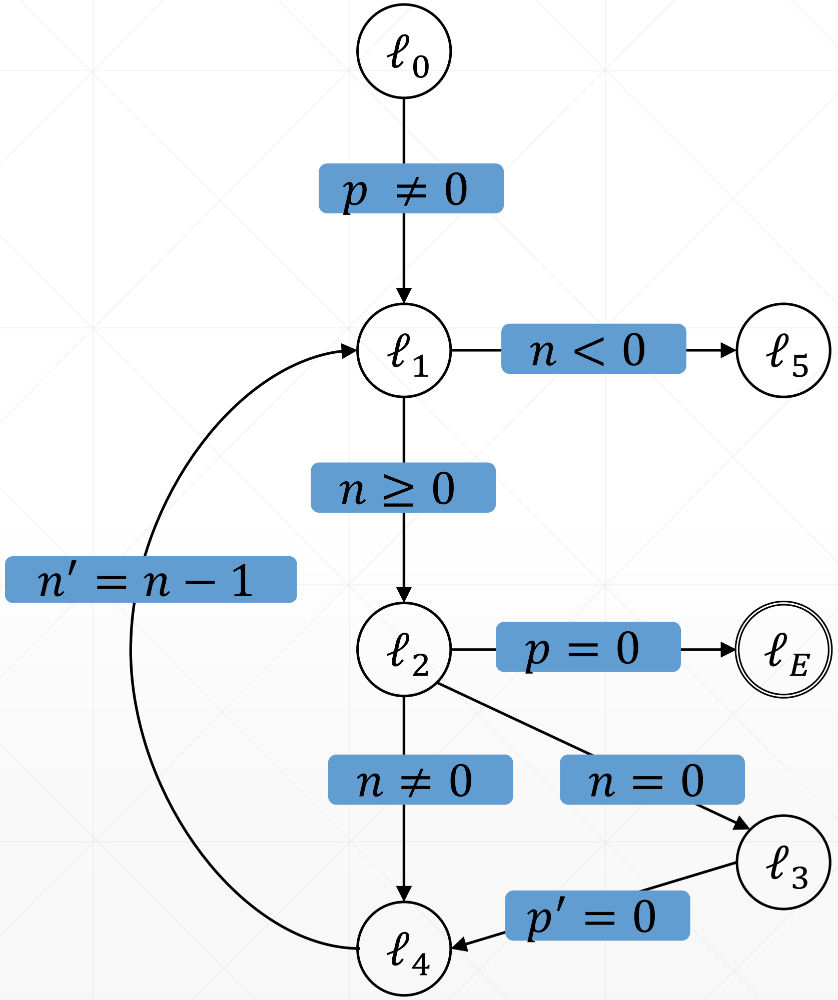
- Predecessor ℓ_4:
  - $f \wedge n' = n - 1 \wedge p' = 0$

  ➔ Unsatisfiable!

Proof-Obligations:

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

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: Level 2 Blocking-Phase:

- Try to block $(p = 0, \ell_1, 1)$

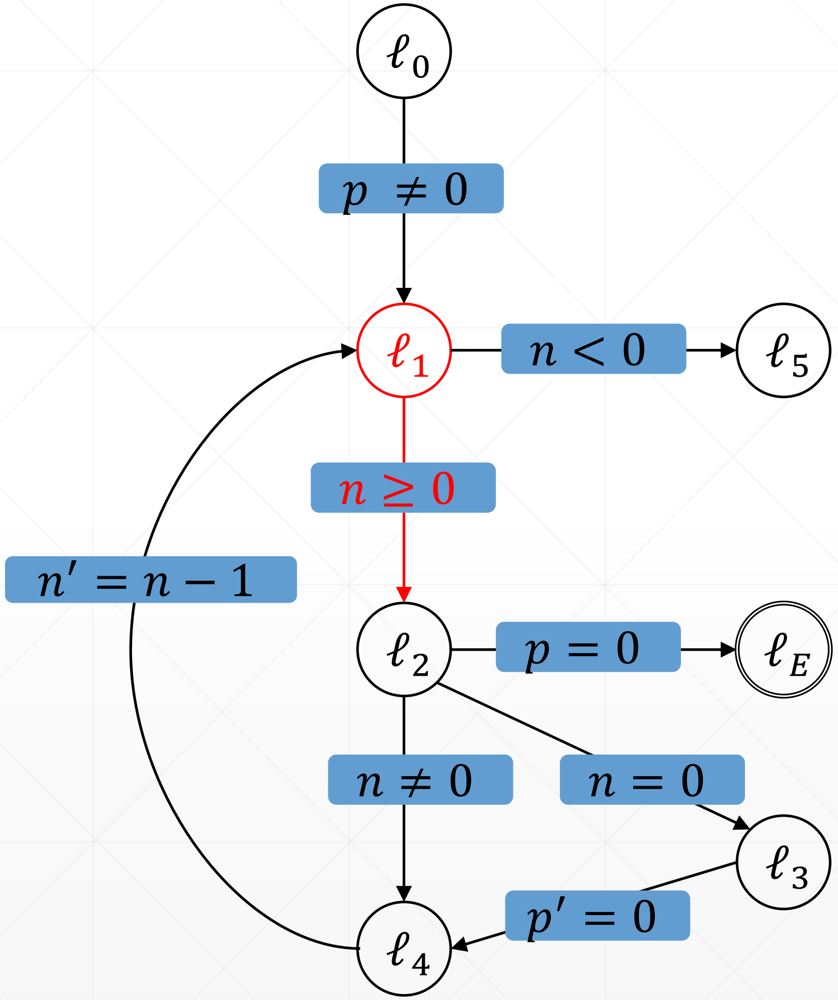
- Predecessor ℓ_4:
  - $f \wedge n' = n - 1 \wedge p' = 0$

  ➔ Unsatisfiable!

Proof-Obligations:

- $(p = 0, \ell_2, 2)$

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: Level 2 Blocking-Phase:

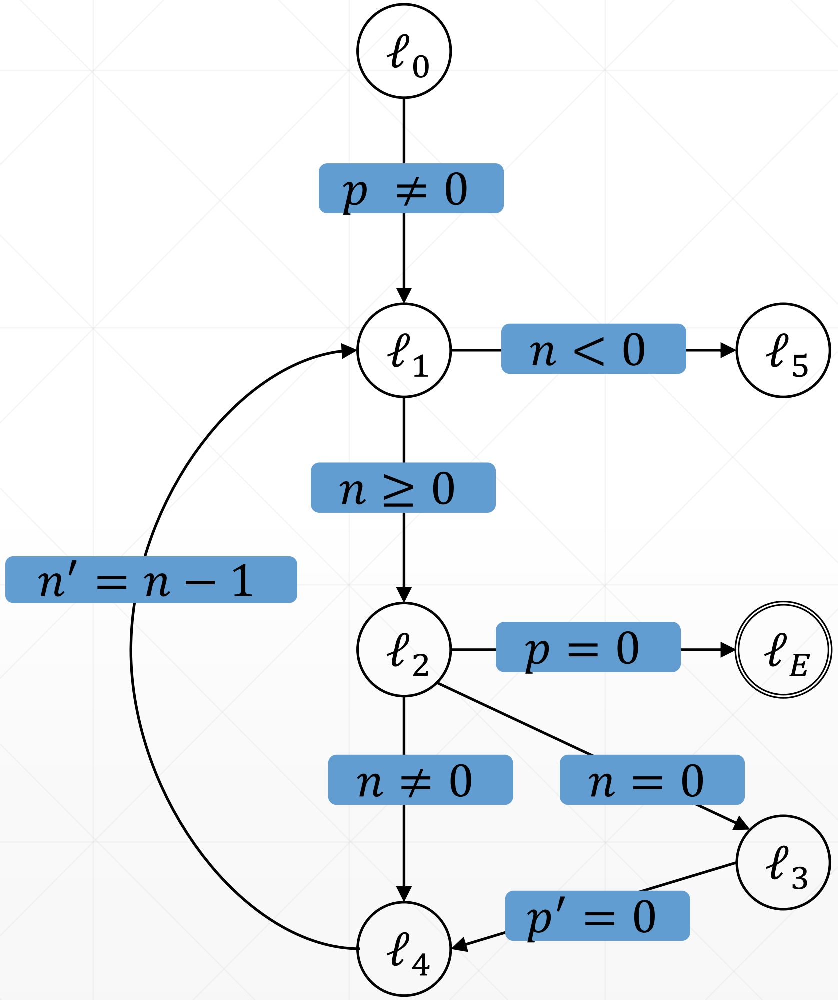
- Try to block $(p = 0, \ell_2, 2)$ again

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

Proof-Obligations:

- $(p = 0, \ell_2, 2)$

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: Level 2 Blocking-Phase:

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

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

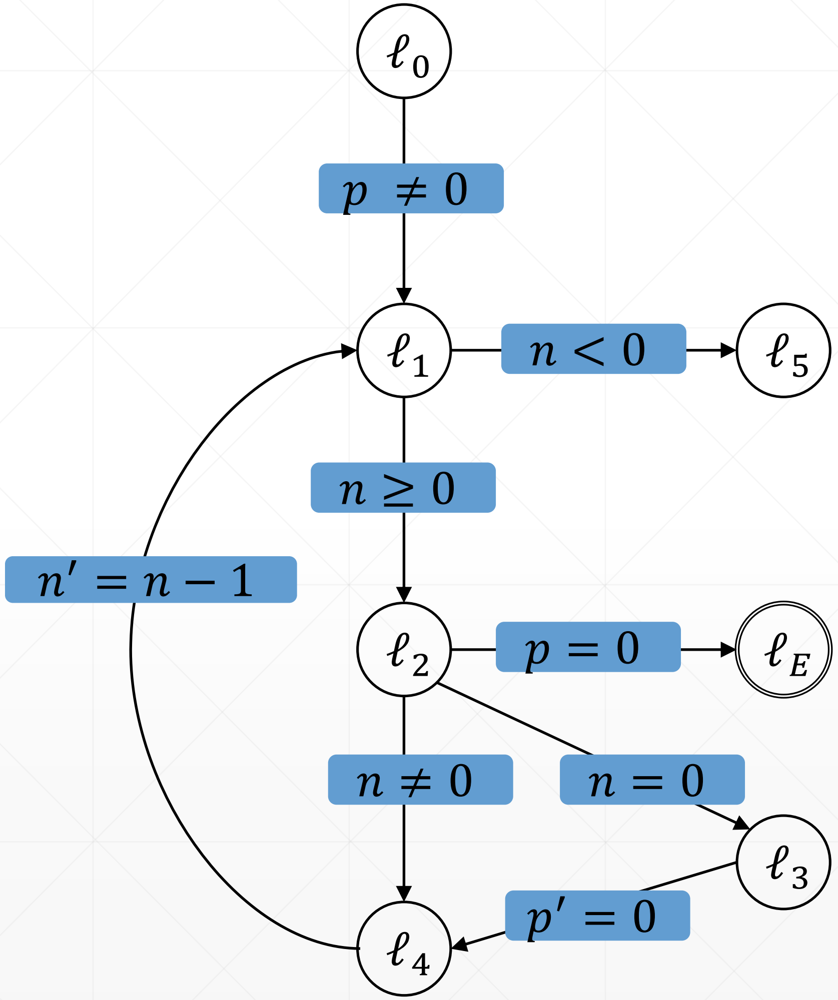
  ➔ Unsatisfiable!

  ➔ Strengthen frames F_{2,ℓ_2}

Proof-Obligations:

- $\emptyset$

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

8. Step: Level 2 Propagation-Phase:

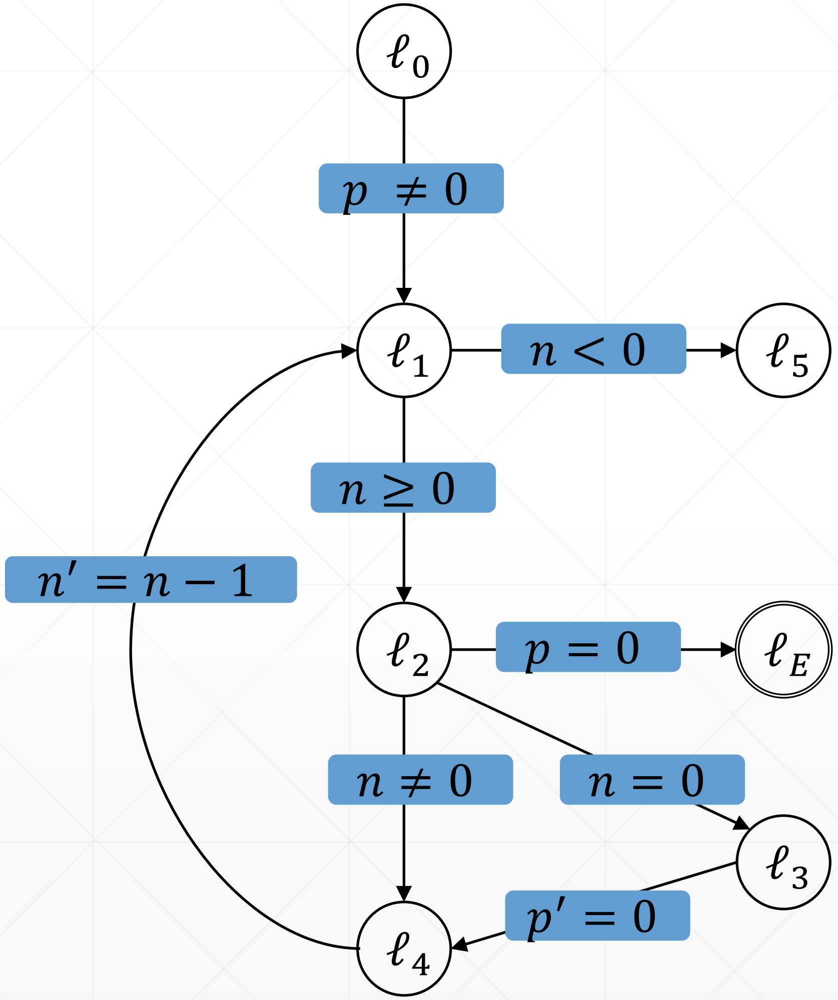
- Is there a global fixpoint?

➔ No, continue with level 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

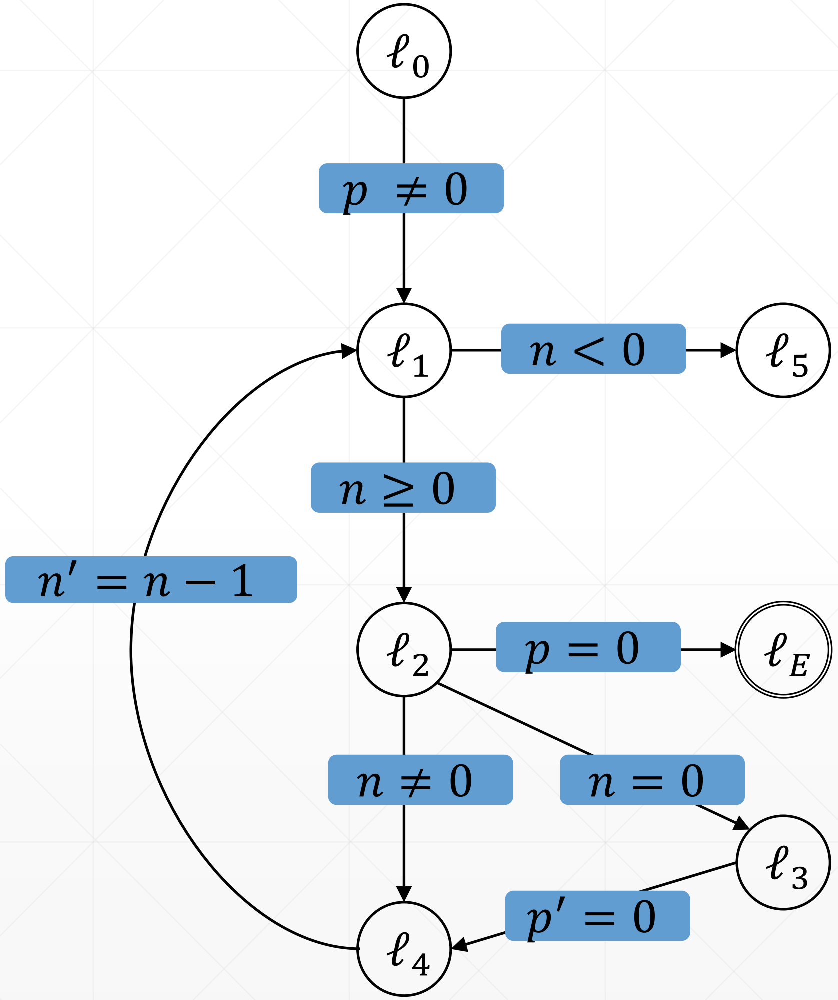
9. Step: Level 3

- Initzialize 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

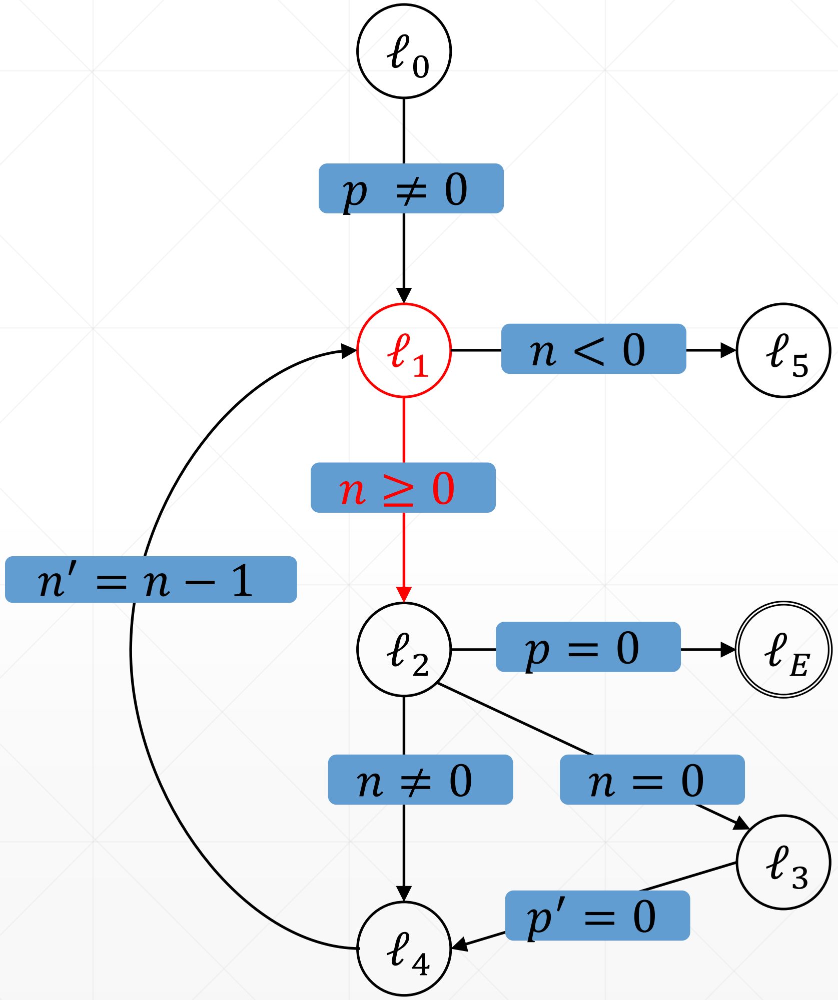
9. Step: Level 3

- Initialize new frames
- Get initial proof-obligations

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: Level 3 Blocking-Phase

- Try to block $(p = 0, \ell_2, 3)$

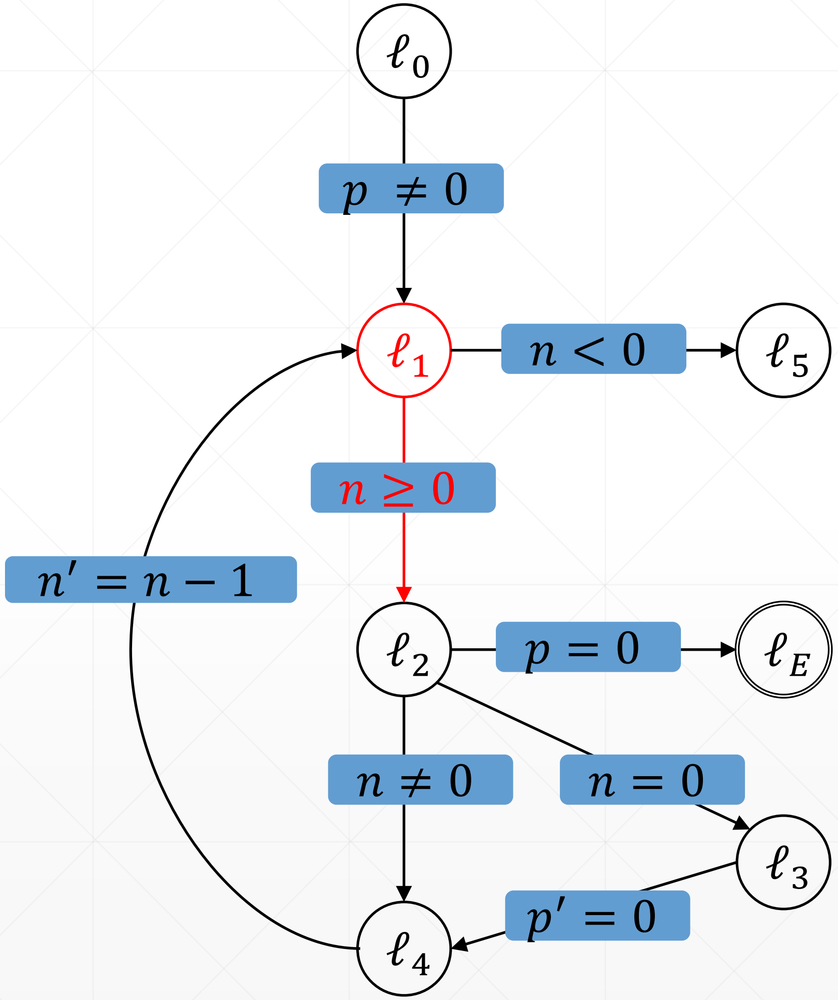
- Predecessor ℓ_1:
  - $t \wedge n \geq 0 \wedge p' = 0$

  ➔ Like the level 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: Level 3 Blocking-Phase

- Try to block $(p = 0, \ell_2, 3)$

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

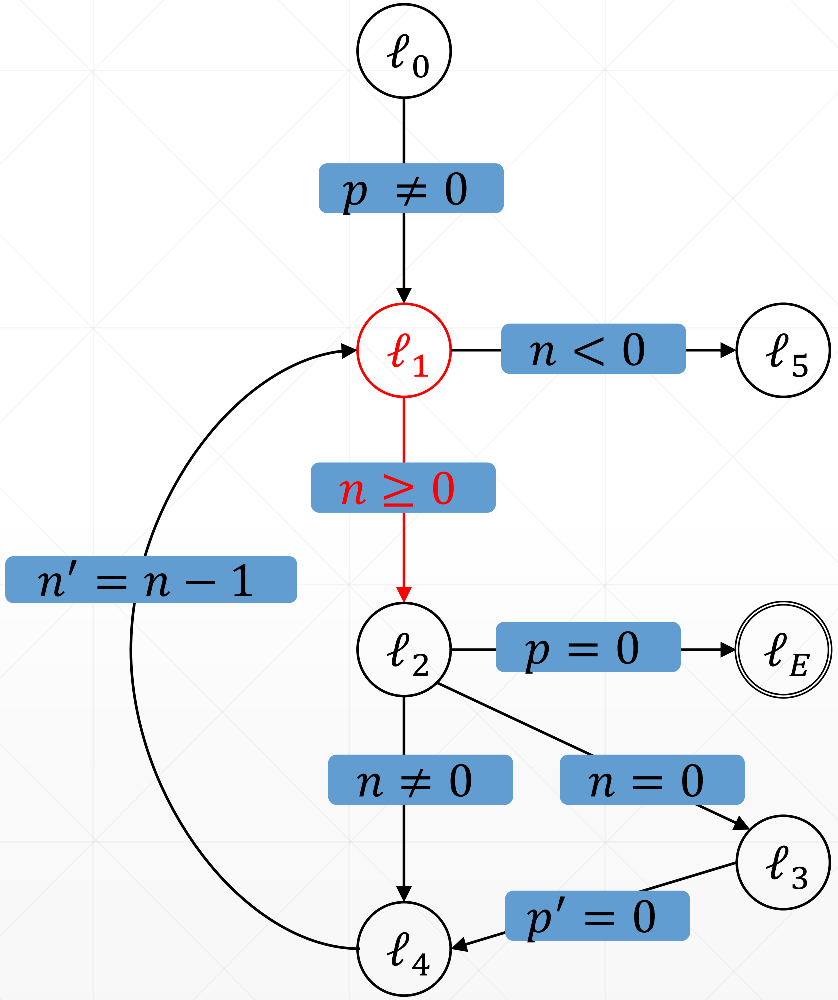
  ➔ Like the level before, get the same new proof-obligation but on level 2

  ➔ $(p = 0, \ell_1, 2)$

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: Level 3 Blocking-Phase

- Try to block $(p = 0, \ell_2, 3)$

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

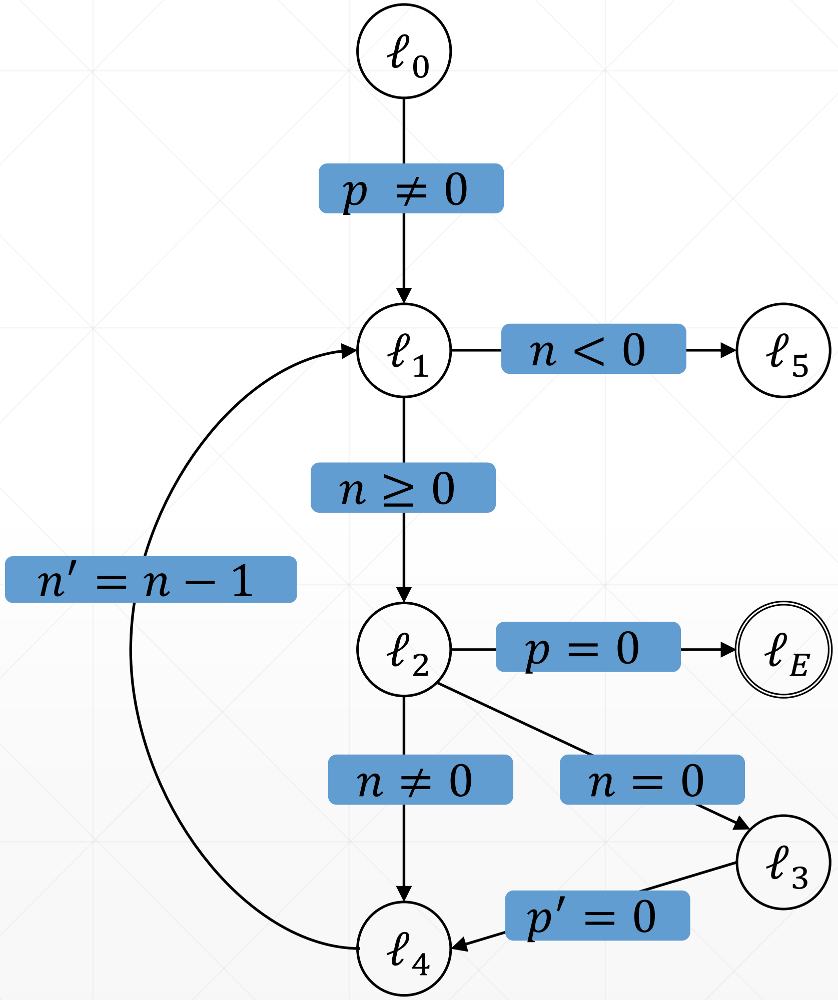
➔ Like the level before, get the same new proof-obligation but on level 2

➔ $(p = 0, \ell_1, 2)$

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	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

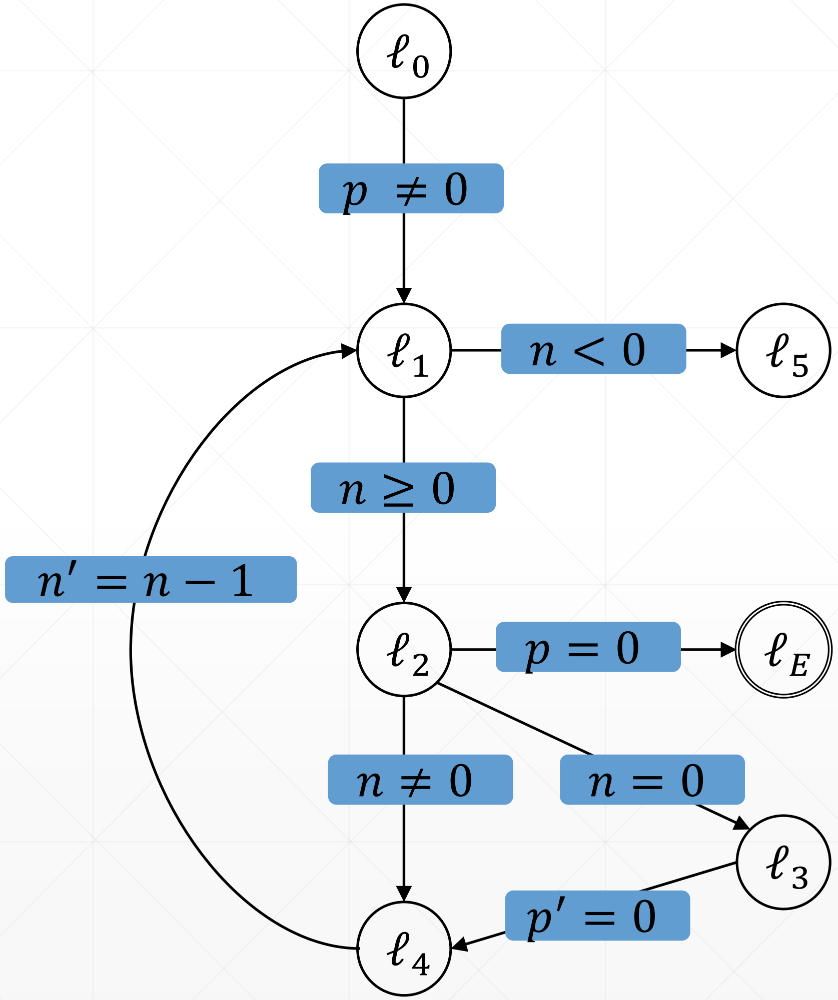
10. Step: Level 3 Blocking-Phase

- There are a lot of repetitions

Proof-Obligations:

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

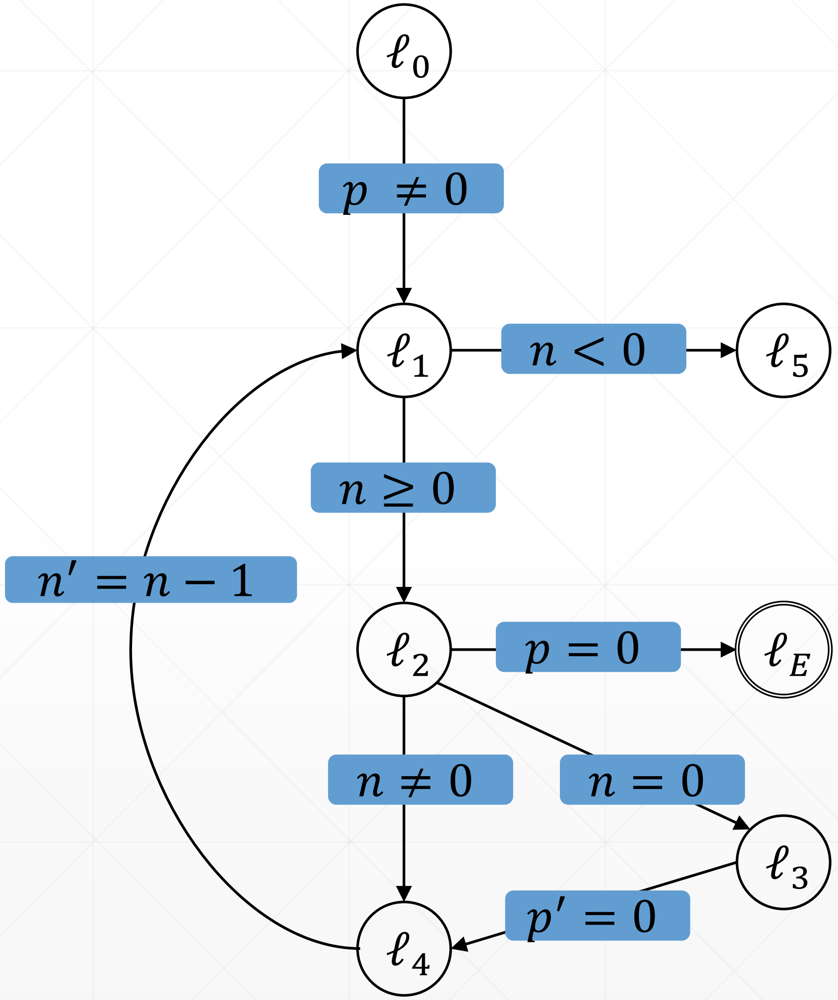
Example:



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

11. Step: Level 3 Done

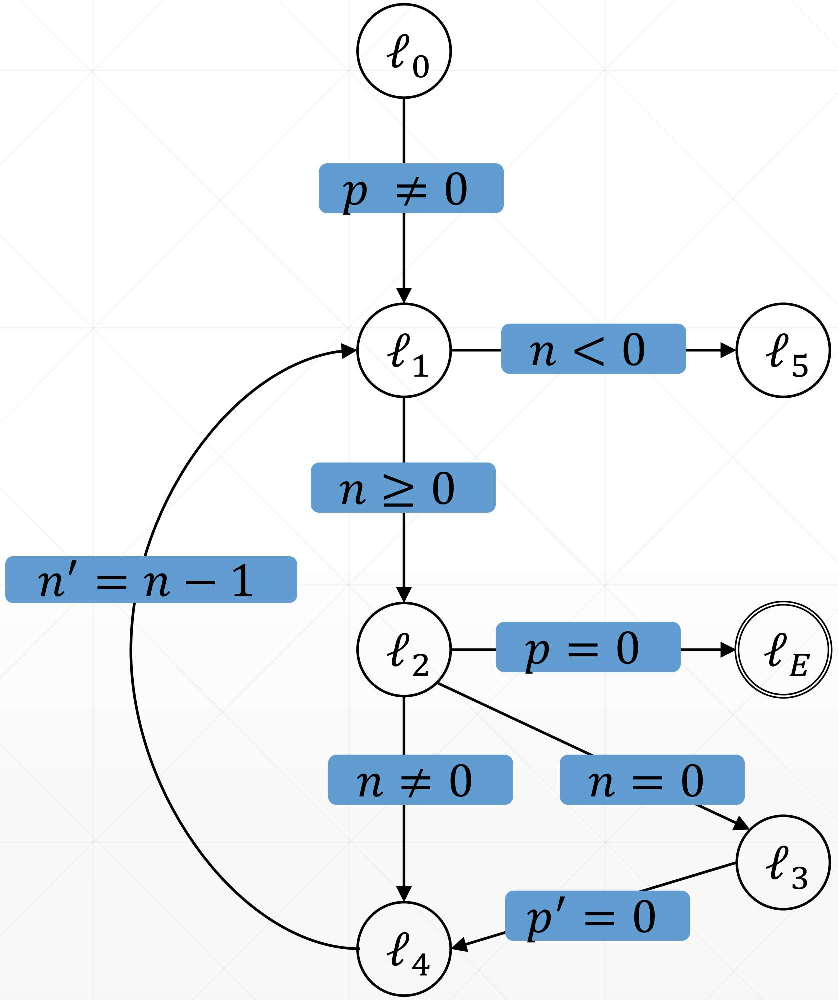
Example:



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: Level 4

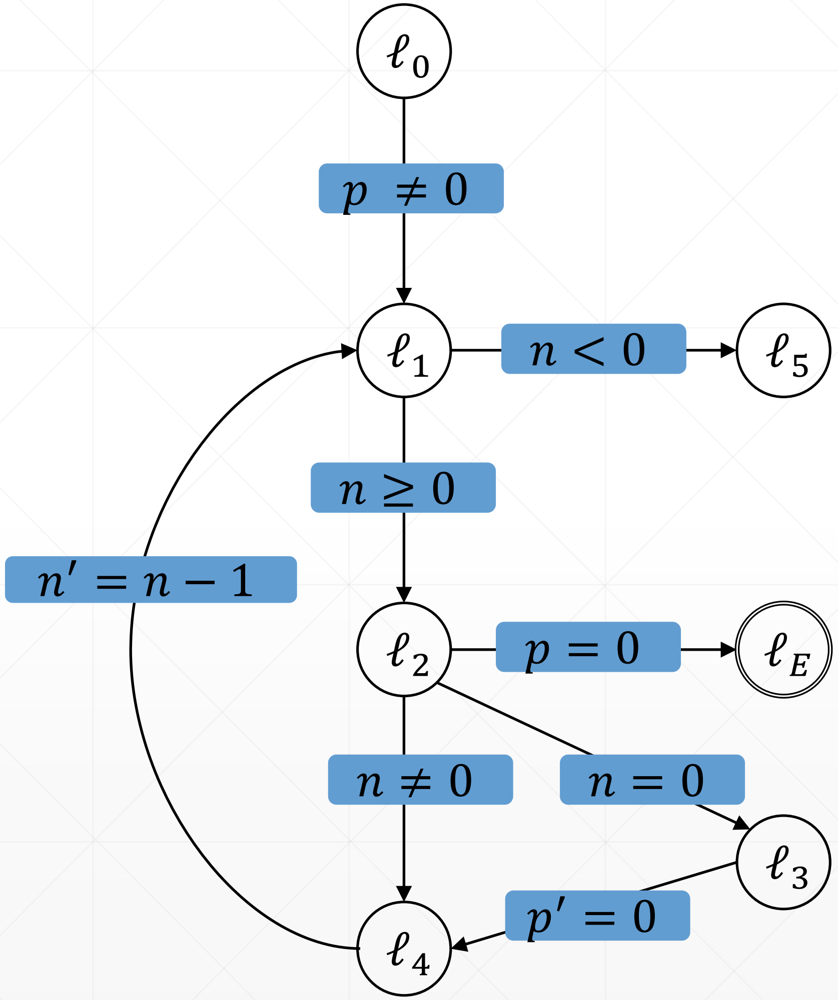
Example:



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: Level 4 Initialization

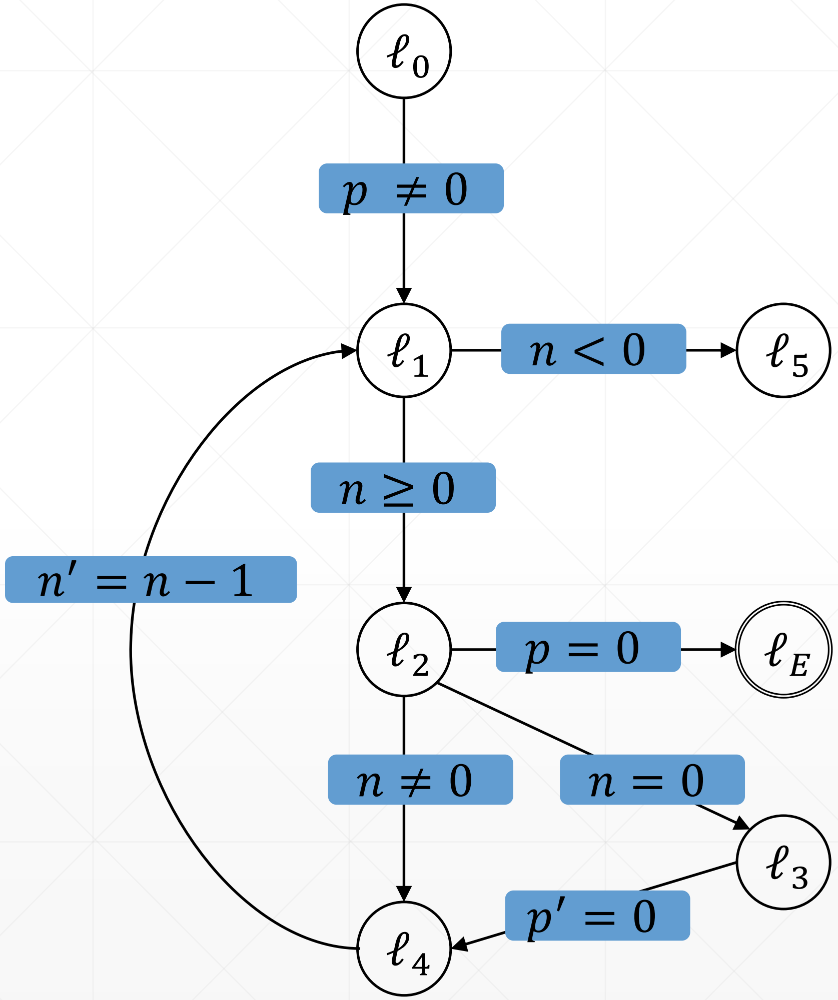
Example:



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

TODO The new interesting proof-obligation!

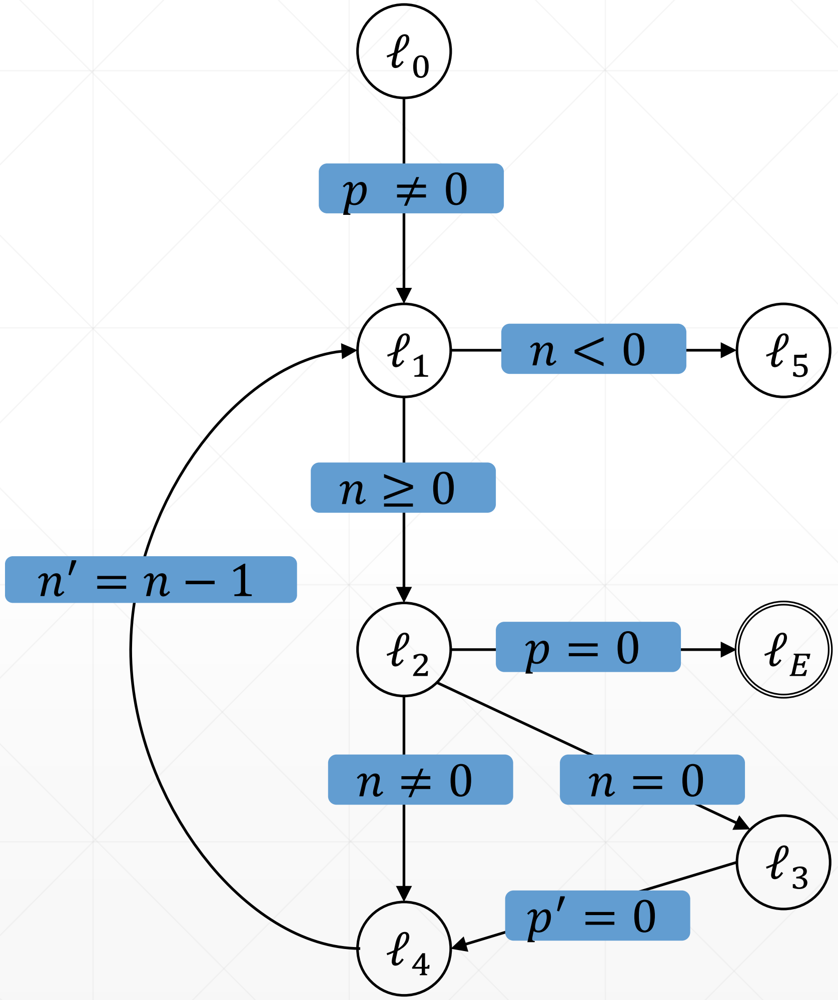
Example:



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

TODO The Last step:
Spoiler: Error is unreachable

Example:



location	0	1	2	3	4	5
ℓ_0						
ℓ_1						
ℓ_2						
ℓ_3						
ℓ_4						

Text

Proof-Obligations:

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 the 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 new CFG of error 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

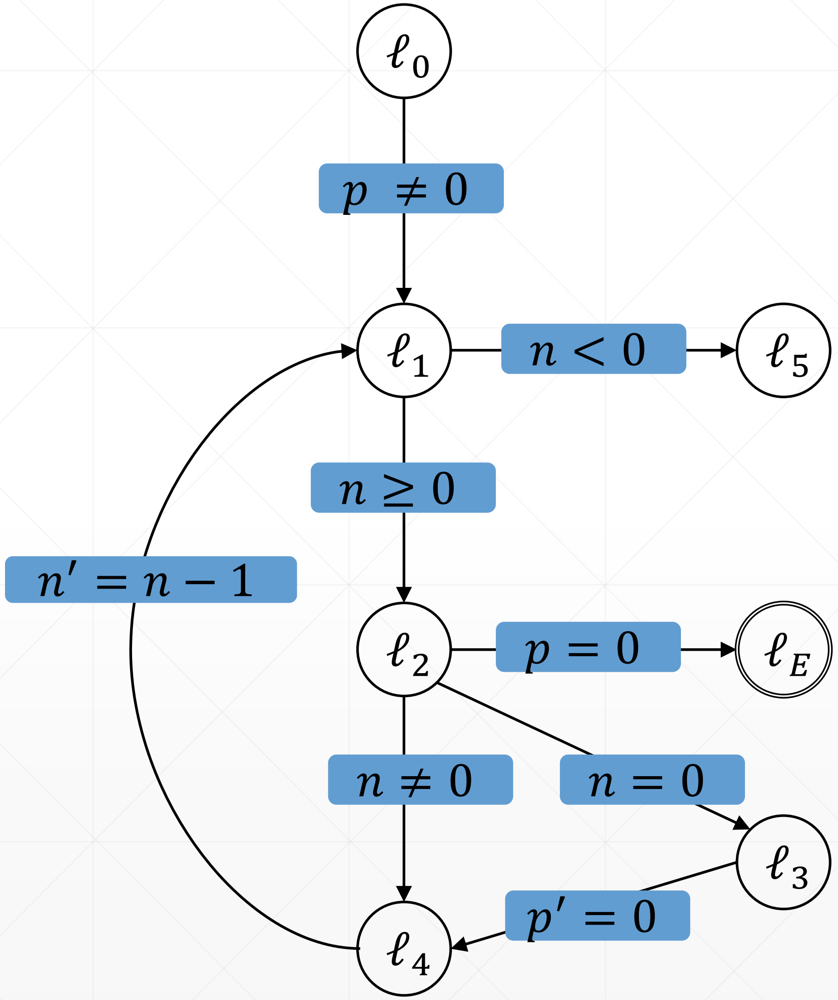
2. Construct a new CFG of error trace

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

   ➔ If unreachable:

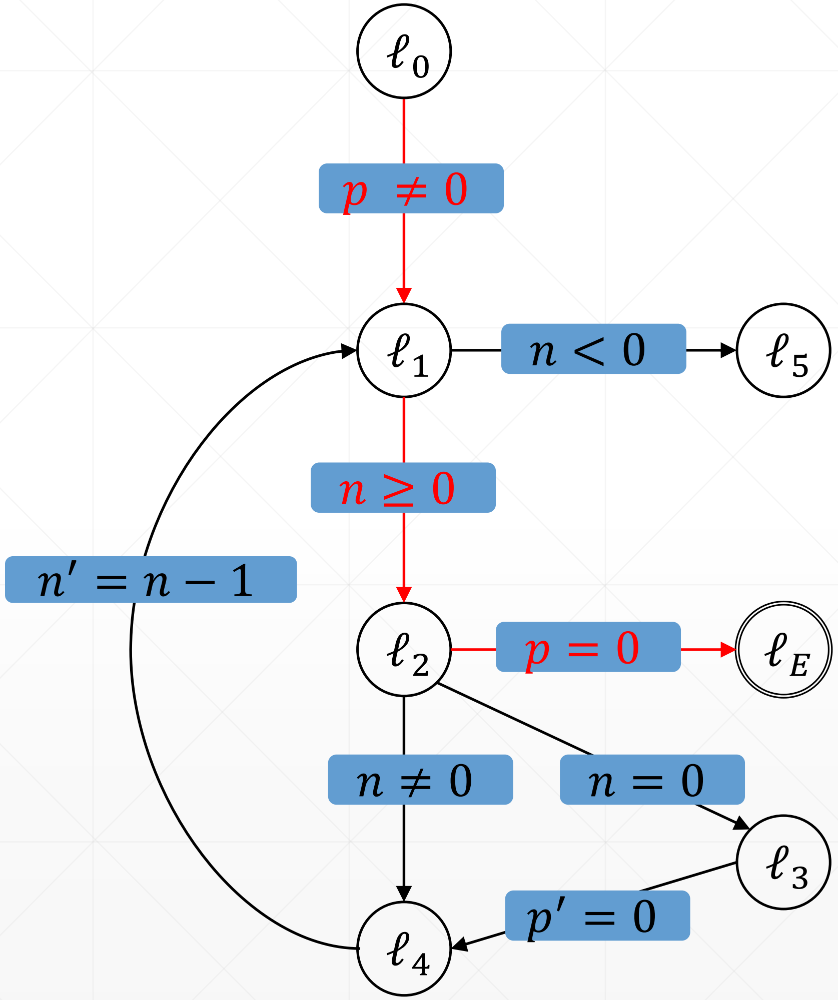
   - Use formulas at the fixpoint as interpolant sequence to refute other error traces

Implementation in Ultimate: Trace Abstraction with PDR



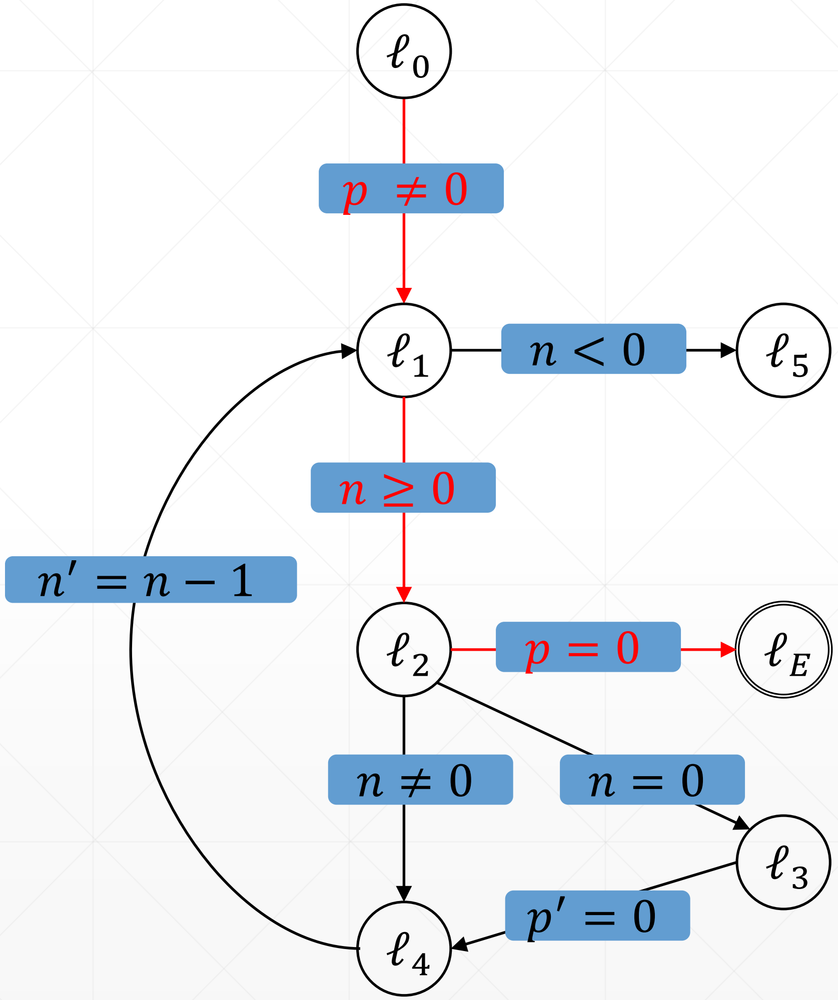
1. Step: Get possible error trace

Implementation in Ultimate: Trace Abstraction with PDR



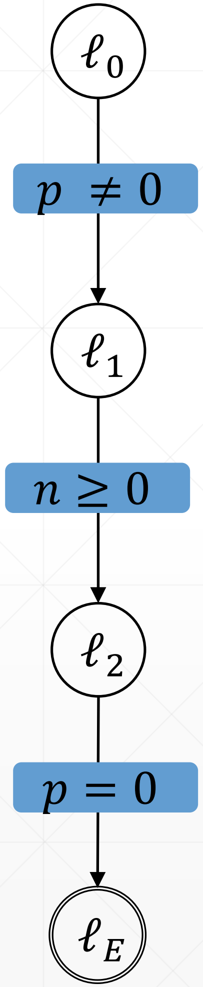
1. Step: Get possible error trace

Implementation in Ultimate: Trace Abstraction with PDR



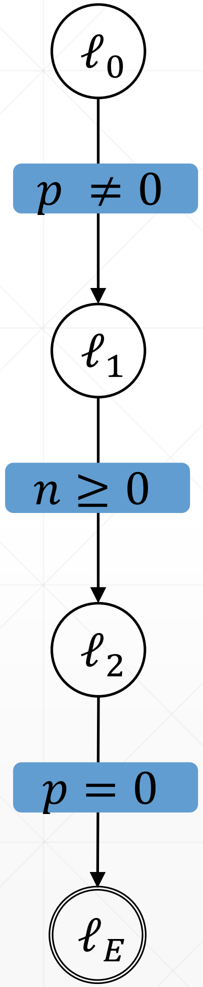
2. Step: Construct new CFG

Implementation in Ultimate: Trace Abstraction with PDR



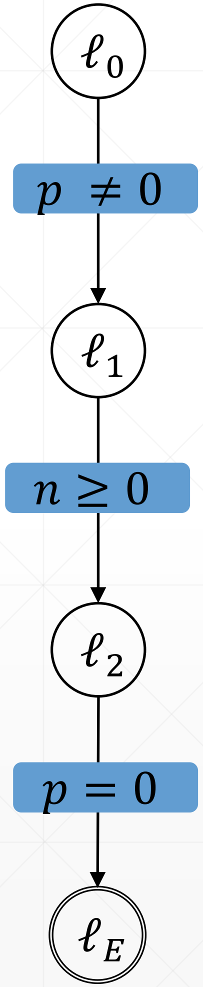
2. Step: Construct new CFG

Implementation in Ultimate: Trace Abstraction with PDR



3. Step: Use PDR

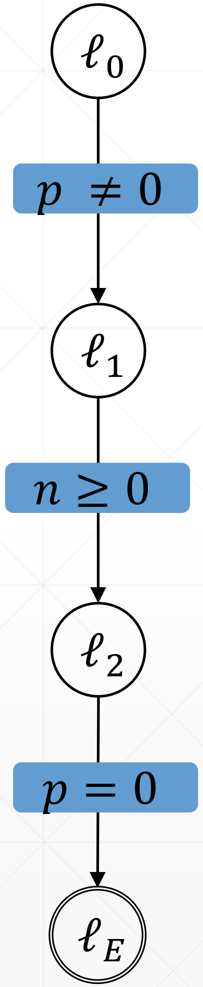
Implementation in Ultimate: Trace Abstraction with PDR



location	0	1	2	3
ℓ_0				
ℓ_1				
ℓ_2				

3. Step: Use PDR

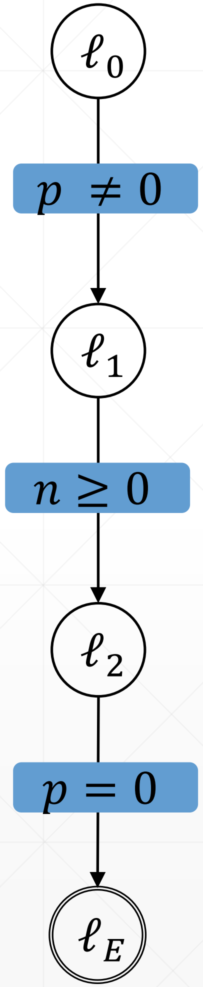
Implementation in Ultimate: Trace Abstraction with PDR



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

3. Step: Use PDR

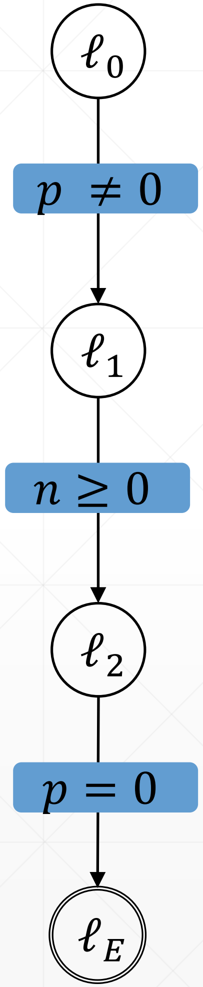
Implementation in Ultimate: Trace Abstraction with PDR



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	$f \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$	$t \wedge p \neq 0$

4. Step: Use fixpoint invariants as interpolants

Implementation in Ultimate: Trace Abstraction with PDR



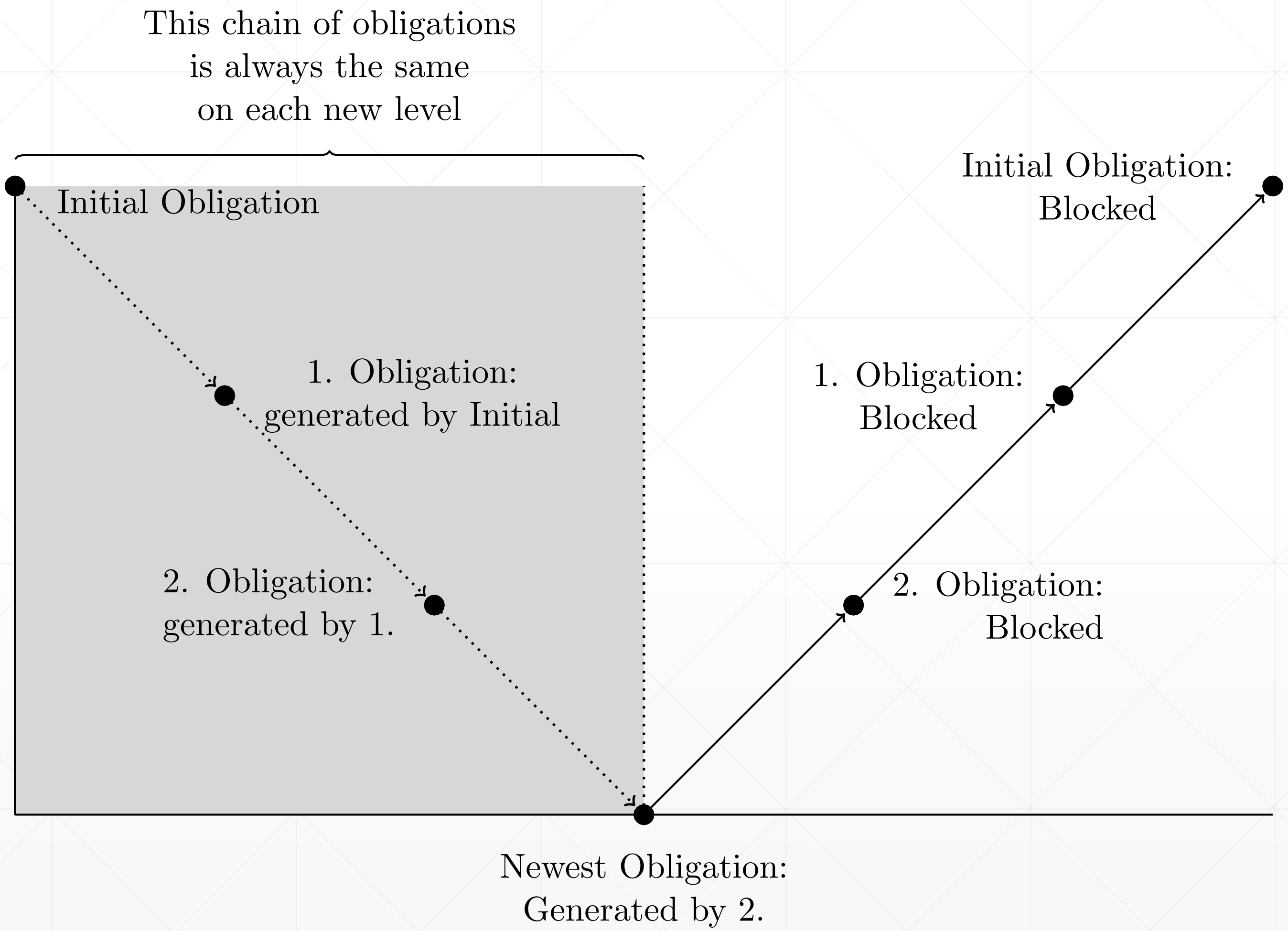
location	0	1	2	3
ℓ_0	t	t	t	t
ℓ_1	$f \land p \neq 0$	$t \land p \neq 0$	$t \land p \neq 0$	t
ℓ_2	$f \land p \neq 0$	$t \land p \neq 0$	$t \land p \neq 0$	$t \land p \neq 0$

4. Step: Use fixpoint invariants as interpolant sequence

Implementation in Ultimate: Implemented Improvements

- Caching proof-obligations:
  - Save the proof-obligation queue
  - Start every new level with the latest blocked proof-obligation

➔ Only proof-obligation that differs from level before



Implementation in Ultimate: Implemented Improvements

- Skipping already blocked proof-obligations:
  - Save unsatisfiable queues to SMT-solver

    ➔ If a saved queue is seen again, do not call SMT-solver again, strengthen frames right away

Evaluation: Data Comparison

- We benchmarked PDR

Evaluation: Discussion

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

- Idea:
  - Use a non-linear approach of PDR
  - Calculate a procedure summary and add that to the CFG, removing the procedure altogether

Conclusion