Bisimulation Minimization and Symbolic Model Checking

Sylvain Julmy

December 18, 2017

The Lee-Yannakakis algorithm

LY - idea

- Stabilize only reachable blocks.
- Reachable block use a representative that has to bee reachable.
- The first state is the representative for the initial block.
- To find new reachable state, we look for transition from representative of reachable state to state from unreachable block.

LY - idea

Two loops :

- Search new reachable blocks
- Stabilize reachable but unstable blocks

LY - termination

With the exception of the initial block, all new blocks created by the algorithm have paths to the bad block.

LY - termination

Therefore, when a second block becomes reachable, the algorithm should raise a violation and terminate.

LY - new algorithm

Basic idea[1] :

- Search new reachable blocks.
- Stabilize reachable but unstable blocks.
- When a second block becomes reachable $\rightarrow$ raise a violation.

[1] Very similar to BR

LY - search

To search for new reachable block, the algorithm is searching from all the successor of the initial state if one of those is in a different block.

The algorithm also determine if the initial block has to be stabilize or not.

LY - search

$D := post(B)$
for all $\langle C, q\rangle \in post(init)$ **do**
    if $B \neq C$ **then**
        raise violation
    end if
    if $B \cap pre(C) \neq B$ **then** ▷ Not all predecessor of B are in B
        B is not stable
    end if
    $D := D - C$
end for
if $D \neq \emptyset$ **then** ▷ $post(init) = \emptyset$
    B is not stable
end if

LY - search

$queue := \emptyset$

$partition = \{B, Bad\}$

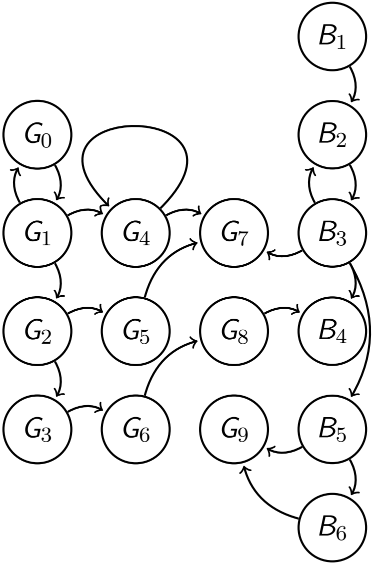
$init = G_0$

$B = \{G_0, \ldots, G_9\}$

$Bad = \{B_1, \ldots, B_6\}$

$block_{init} = \langle B, init \rangle$

$D = post(B) = \{G_0, \ldots, G_8, B_4\}$



LY - search

$post(init) = \{G_1\}$

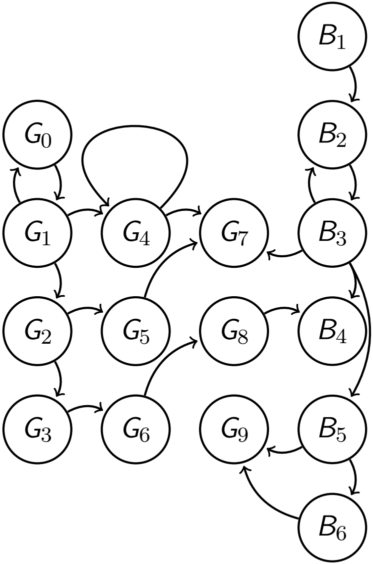
$\langle C, q\rangle = \langle B, init\rangle$

$pre(C) = \{G_0, \ldots, G_6, B_3, B_5, B_6\}$

$B \cap pre(C) = \{G_0, \ldots, G_6\}$

$\rightarrow enqueue(\langle B, init\rangle)$

$D_1 = D - C = \emptyset$



LY - stabilization

while B is not stable **do**
    Mark B as stable
    Compute the frontier of B
    Let B' the state of B that can only reach B
    Let B'' the state of B that can reach a bad block
    if $\emptyset \neq B' \cap \mathit{pre}(B') \neq B'$ or $\emptyset \neq B' \cap \mathit{pre}(B'') \neq B'$ **then**
        Mark B as unstable
    end if
end while

LY - stabilization

Iteration 1

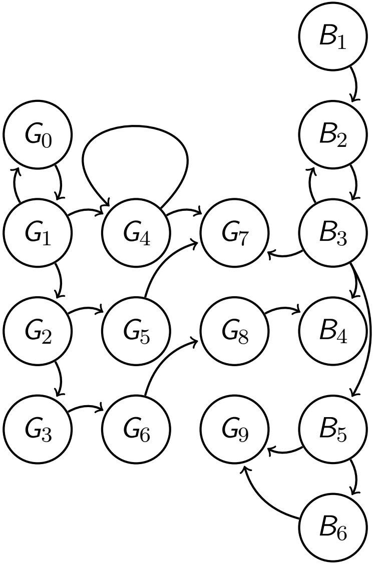
$init = G_0$

$partition = \{\{G0, \ldots, G_9\}, \{B_1, \ldots, B_6\}\}$

$B = \{G0, \ldots, G_9\}$

$pre(B) = \{G_0, \ldots, G_6, B_3, B_5, B_6\}$

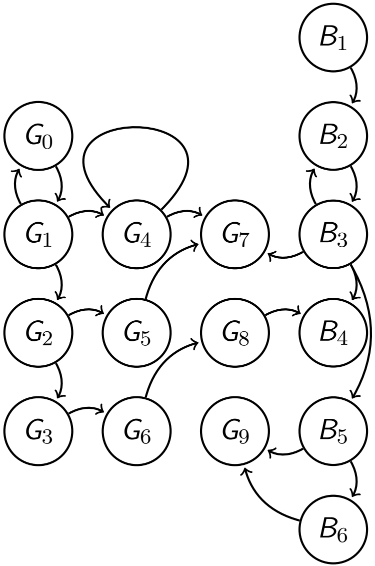
$post(B) = \{G_0, \ldots, G_8, B_4\}$



LY - stabilization

Iteration 1

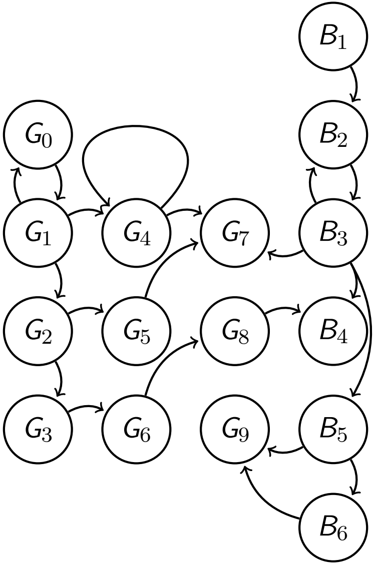
$$B'_1 = B \cap pre(B) = \{G_0, \ldots, G_5\}$$
$$B'_2 = pre(post(B) - B) = pre(\{B_4\}) = \{G_8\}$$
$$B' = B'_1 - B'_2 = \{G_0, \ldots, G_5\}$$
$$B'' = B - B' = \{G_6, \ldots, G_9\}$$
$$partition = \{B, Bad, \{G_6, \ldots, G_9\}\}$$
$$B := B' = \{G_0, \ldots, G_5\}$$



LY - stabilization

Iteration 1

$B := B' = \{G_0, \ldots, G_5\}$
$pre(B) = \{G_0, G_1, G_2, G_4\}$
$B'' = \{G_6, \ldots, G_9\}$
$pre(B'') = \{G_3, G_4, G_5, B_3, G_6, B_5, B_6\}$
$B \cap pre(B) \neq B \neq \emptyset$
$\rightarrow enqueue(\langle B, init \rangle)$
$post(init) \cap B'' = \{G_1\} \cap \{G_6, \ldots, G_9\} = \emptyset$



LY - stabilization

Iteration 2

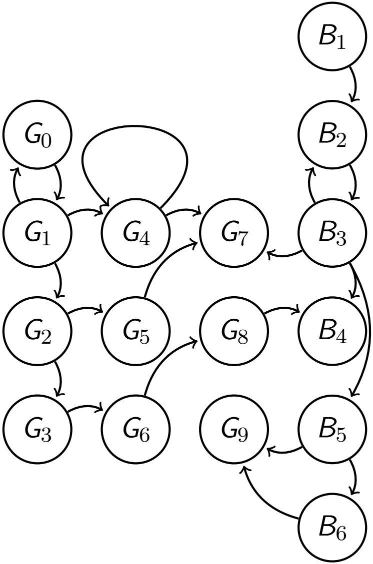
$init = G_0$

$partition = \{B, Bad, \{G_6, \dots, G_9\}\}$

$B = \{G_0, \dots, G_5\}$

$pre(B) = \{G_0, G_1, G_2, G_4\}$

$post(B) = \{G_0, \dots, G_8\}$



LY - stabilization

Iteration 2

$B'_1 = B \cap pre(B) = \{G_0, G_1, G_2, G_4\}$

$B'_2 = pre(post(B) - B) = pre(\{G_6, G_7, G_8\})$

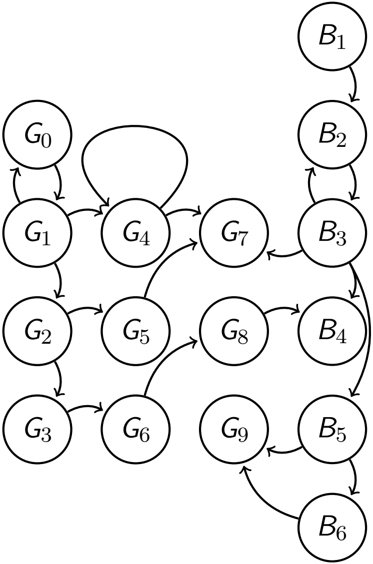
$B'_2 = \{G_3, G_4, G_5, G_6, B_3\}$

$B' = B'_1 - B'_2 = \{G_0, G_1, G_2\}$

$B'' = B - B' = \{G_4\}$

$partition = \{B, Bad, \{G_6, \ldots, G_9\}, \{G_4\}\}$

$B := B' = \{G_0, G_1, G_2\}$



LY - stabilization

Iteration 2

$B := B' = \{G_0, G_1, G_2\}$

$pre(B) = \{G_0, G_1\}$

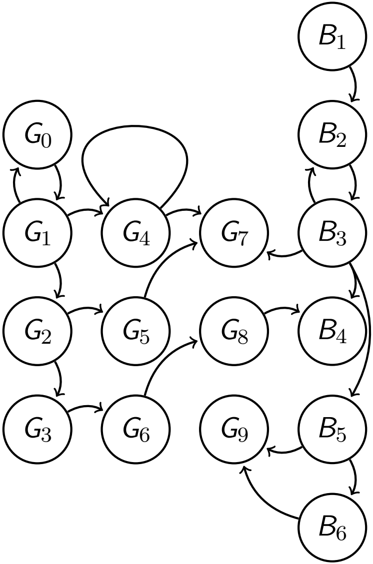
$B'' = \{G_4\}$

$pre(B'') = \{G_1, G_4\}$

$B \cap pre(B) \neq B \neq \emptyset$

$\to enqueue(\langle B, init\rangle)$

$post(init) \cap B'' = \{G_1\} \cap \{G_4\} = \emptyset$



LY - stabilization

Iteration 3

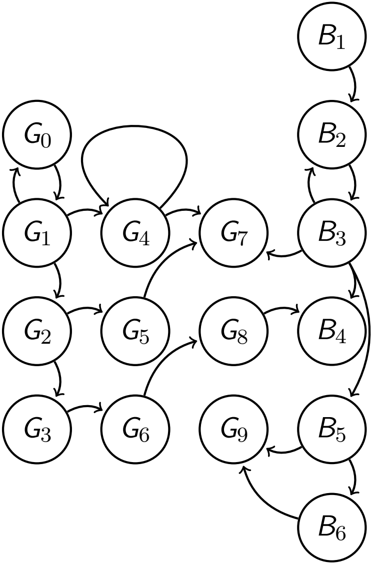
$init = G_0$

$partition = \{B, Bad, \{G_6, \ldots, G_9\}, \{G_4\}\}$

$B := \{G_0, G_1, G_2\}$

$pre(B) = \{G_0, G_1\}$

$post(B) = \{G_0, \ldots, G_5\}$



LY - stabilization

Iteration 3

$B'_1 = B \cap pre(B) = \{G_0, G_1\}$

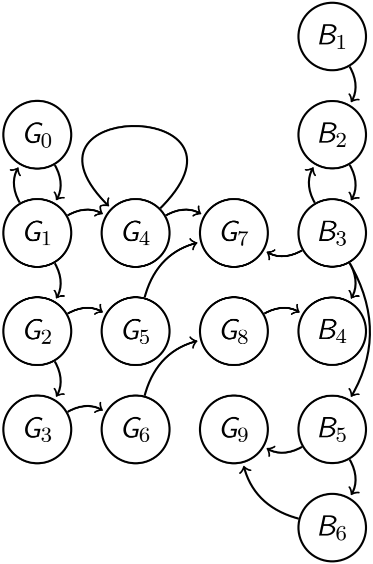
$B'_2 = pre(post(B) - B) = pre(G_3, G_4, G_5)$

$B'_2 = \{G_1, G_2, G_4\}$

$B' = B'_1 - B'_2 = \{G_0\}$

$B'' = B - B' = \{G_1, G_2\}$

$B := B' = \{G_0\}$



LY - stabilization

Iteration 3

$B := B' = \{G_0\}$

$pre(B) = \{G_1\}$

$B'' = \{G_1, G_2\}$

$pre(B'') = \{G_0, G_1\}$

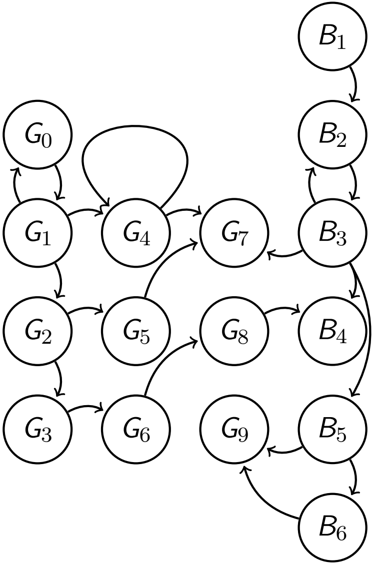
$B \cap pre(B) = \emptyset$

$B \cap pre(B'') = \{G_0\} = B$

$\rightarrow$ don't enqueue !

$post(init) \cap B'' = \{G_1\} \cap \{G_2\} = \{G_1\}$

$\rightarrow$ raise violation !



LY - complexity

$$(n-1) * 5M + 4I + 3D + 4E$$

where

- n : number of BR iterations
- M : number of image iterations
- I : number of intersection operations
- D : number of set difference operations
- E : number of equality check
- U : number of union operations

BFH

The Bouajjani-Fernandez-Halbwachs algorithm

BFH - idea

- BFH, like LY, selects reachable blocks to stabilize but differ in how to stabilize a block.
- BFH stabilize a block w.r.t. all the other blocks (either reachable or unreachable).
- The algorithm become simplier but unnecessary work is done.

BFH - termination

As in LY, BFH could terminate when a second block becomes reachable. The algorithm correctly determine violations of invariants but not as soon as they occur.

BFH - termination

The algorthim may traverse a path from the bad block to the initial state before the initial block becomes stable.
Thus, the algorithm take more iteration to terminate.

BFH - new Algorithm

Mark the bad block
$I = [init]_p$
while I is not marked **do**
    $N := split(I, p)$
    if $N = \{I\}$ **then**
        if $post(I) - I \neq \emptyset \rightarrow$ violation, **else** break
    else
        $p := (p - \{I\}) \cup N$
        $I := [init]_p$
    end if
end while
if I is marked **then**
    Signal safety violation
end if

BFH - new algorthim (split)

```
function SPLIT(X : state, p : partition)
    N = {X}
    for all Y : block ∈ p do
        M := ∅
        for all W : state ∈ N do
            W_1 = W ∩ pre(Y)
            if W_1 = W or W_1 = ∅ then
                a
            else
                b
            end if
        end for
    end for
    return N
end function
```