

Generating fixes for failed proofs: experiments and results

No Author Given

No Institute Given

The experiment ran on a Windows 11 machine with a 2.1 GHz Intel 12-Core and 32 GB of memory. AutoProof or Proof2Fix was the only computationally-intensive process running during the experiments. Version numbers for the underlying technology are: EiffelStudio 22.05; Boogie 2.11.1.0; Z3 4.8.14.

1 Examples with numeric computations

1.1 ACCOUNT

`ACCOUNT`, whose implementation is shown below, is a class that describes the behaviors of bank accounts; it includes a set of features representing basic operations on bank account: `deposit` (line 51), `withdraw` (line 64), and `transfer` (line 77). Fig.1 shows the verification result of this version of `ACCOUNT`, which suggests a complete functional correctness. Different faults are injected into the correct version, which results in 7 faulty variants of the `ACCOUNT` class, which will be discussed as follows.

```
1 class
2   ACCOUNT
3 create
4   make
5 feature {NONE} -- Initialization
6   make
7     -- Initialize empty account.
8   note
9     status: creator
10  do
11    balance := 0
12    credit_limit := 0
13  ensure
14    balance_set: balance = 0
15    credit_limit_set: credit_limit = 0
16  end
17 feature -- Access
18   balance: INTEGER
19     -- Balance of this account.
20   credit_limit: INTEGER
21     -- Credit limit of this account.
```

```

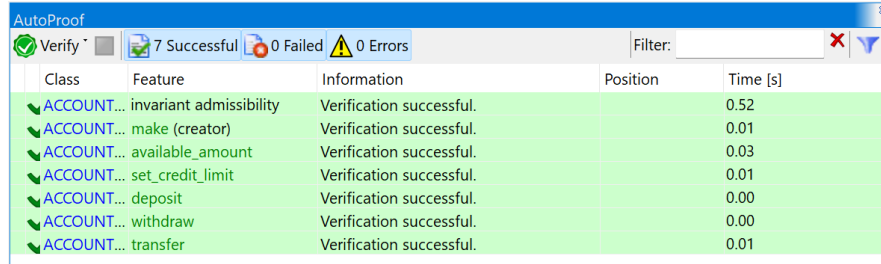
22     available_amount: INTEGER
23     -- Amount available on this account.
24     note
25         status: functional
26     do
27         Result := balance - credit_limit
28     end
29 feature -- Basic operations
30     set_credit_limit (limit: INTEGER)
31         -- Set 'credit_limit' to 'limit'.
32     require
33         limit_not_positive: limit ≤ 0
34         limit_valid: limit ≤ balance
35     do
36         credit_limit := limit
37     ensure
38         modify_field ([“credit_limit”, “closed”], Current)
39         credit_limit_set: credit_limit = limit
40     end
41     deposit (amount: INTEGER)
42         -- Deposit 'amount' in this account.
43     require
44         amount ≥ 0
45     do
46         balance := balance + amount
47     ensure
48         modify_field ([“balance”, “closed”], Current)
49         balance_increased: balance ≥ old balance
50         balance_set: balance = old balance + amount
51     end
52     withdraw (amount: INTEGER)
53         -- Withdraw 'amount' from this account.
54     require
55         amount_not_negative: amount ≥ 0
56         amount_available: amount ≤ available_amount
57     do
58         balance := balance - amount
59     ensure
60         modify_field ([“balance”, “closed”], Current)
61         balance_set: balance = old balance - amount
62         balance_decrease: balance ≤ old balance
63     end
64     transfer (amount: INTEGER; other: ACCOUNT_1)
65         -- Transfer 'amount' from this account to 'other'.
66     note

```

```

67         explicit: wrapping
68     require
69         amount_not_negative: amount  $\geq$  0
70         amount_available: amount  $\leq$  available_amount
71         other  $\neq$  Current
72     do
73         withdraw (amount)
74         other.deposit (amount)
75     ensure
76         modify_field ([“balance”, “closed”], [Current, other])
77         withdrawal_made: balance = old balance - amount
78         deposit_made: other.balance = old other.balance + amount
79     end
80 invariant
81     credit_limit_not_positive: credit_limit  $\leq$  0
82     balance_non_negative: balance - credit_limit  $\geq$  0
83 end

```



Class	Feature	Information	Position	Time [s]
ACCOUNT...	invariant admissibility	Verification successful.		0.52
ACCOUNT...	make (creator)	Verification successful.		0.01
ACCOUNT...	available_amount	Verification successful.		0.03
ACCOUNT...	set_credit_limit	Verification successful.		0.01
ACCOUNT...	deposit	Verification successful.		0.00
ACCOUNT...	withdraw	Verification successful.		0.00
ACCOUNT...	transfer	Verification successful.		0.01

Fig. 1. Verification result of **ACCOUNT** in AutoProof: all routines are verified successfully (highlighted with green), which indicates that implementations of those routines are correct with respect to their specifications.

Variant 1 of **ACCOUNT**

- Fault injection: at line 73, change the postcondition *balance_set* from “balance = old balance - amount” into “balance = old balance + amount”.
- Resulting failure: as shown in Fig. 2(a), the fault results in a violation of postcondition *balance_set* of the **withdraw** procedure.
- Cause of the failure: the implementation of **withdraw** (which *deduces* **balance** by **amount**) and specification (which requires the *increment* of **balance** by **amount**) is inconsistent.
- Proof time: 0.247 sec
- Proof2Fix: No valid fixes are found.

Class	Feature	Information	Position	Time [s]
ACCOUNT_1	invariant a...	Verification successful.		0.40
ACCOUNT_1	make (cre...	Verification successful.		0.01
ACCOUNT_1	available_...	Verification successful.		0.03
ACCOUNT_1	set_credit_...	Verification successful.		0.01
ACCOUNT_1	deposit	Verification successful.		0.00
ACCOUNT_1	withdraw	Postcondition <i>balance_set</i> may be violated.	77	0.03
ACCOUNT_1	transfer	Verification successful.		0.00

(a)

Variant 2 of *ACCOUNT*

- Fault injection: at line 68, remove the precondition *amount_available* of *withdraw*.
- Resulting failure: as shown in Fig. 2(b), the class invariant *balance_non_negative* (line 96), which states that the balance (represented by *balance* – *amount*) should not be negative, is violated. (Note that a class invariant which is supposed to hold at the entry and exit of every routine.)
- Cause of the failure: the precondition of *withdraw* is too weak; there should be a precondition to constrain the amount permitted in a withdrawal operation.
- Proof time: 0.275 sec
- Proof2Fix: 4 valid fixes out of 211 candidate fixes
- Fixing time: 1.58 minutes

Class	Feature	Information	Position	Ti...
ACCOU...	invariant a...	Verification successful.		0.18
ACCOU...	make (cre...	Verification successful.		0.01
ACCOU...	available_...	Verification successful.		0.03
ACCOU...	set_credit_...	Verification successful.		0.02
ACCOU...	deposit	Verification successful.		0.00
ACCOU...	withdraw	Invariant <i>balance_non_negative</i> might not hold on call to (ANY).wrap.	79	0.02
ACCOU...	transfer	Verification successful.		0.00

(b)

```

1  withdraw_ID_188 (amount: INTEGER)
2      -- Withdraw 'amount' from this account.
3  require
4      amount_not_negative: amount  $\geq$  0
5      -- amount_available: amount  $\leq$  available_amount
6      not ((amount) > (0)) -- **Fix
7  do
8      balance := balance - amount
9  ensure
10     modify_field (["balance", "closed"], Current)
11     balance_set: balance = old balance - amount
12     balance_decrease: balance  $\leq$  old balance
13 end

```

Fig. 2. Fix from Proof2Fix

```

1  withdraw_ID_202 (amount: INTEGER)
2      -- Withdraw 'amount' from this account.
3  require
4      amount_not_negative: amount  $\geq$  0
5      -- amount_available: amount  $\leq$  available_amount
6      not (balance < amount) -- **Fix
7  do
8      balance := balance - amount
9  ensure
10     modify_field (["balance", "closed"], Current)
11     balance_set: balance = old balance - amount
12     balance_decrease: balance  $\leq$  old balance
13 end

```

Fig. 3. Test case from failed proof of *withdrawal_made*

Fig. 4. Fix from Proof2Fix

```

1  withdraw_ID_206 (amount: INTEGER)
2      -- Withdraw 'amount' from this account.
3  require
4      amount_not_negative: amount  $\geq$  0
5      -- amount_available: amount  $\leq$  available_amount
6      not (credit_limit < amount) -- **Fix
7  do
8      balance := balance - amount
9  ensure
10     modify_field(["balance", "closed"], Current)
11     balance_set: balance = old balance - amount
12     balance_decrease: balance  $\leq$  old balance
13 end

```

Fig. 5. Fix from Proof2Fix

```

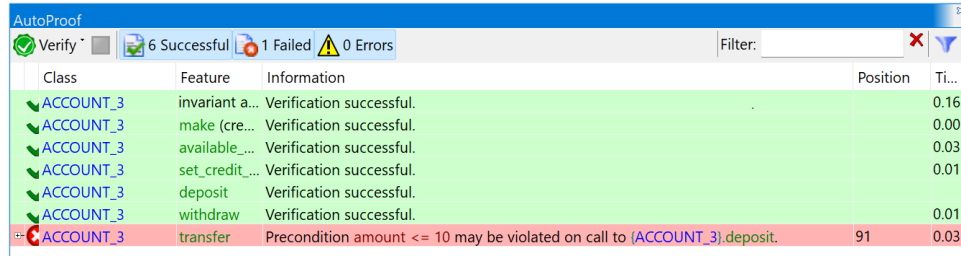
1  withdraw_ID_210 (amount: INTEGER)
2      -- Withdraw 'amount' from this account.
3  require
4      amount_not_negative: amount  $\geq$  0
5      -- amount_available: amount  $\leq$  available_amount
6      not (amount > available_amount) -- **Fix
7  do
8      balance := balance - amount
9  ensure
10     modify_field(["balance", "closed"], Current)
11     balance_set: balance = old balance - amount
12     balance_decrease: balance  $\leq$  old balance
13 end

```

Fig. 6. Fix from Proof2Fix

Variant 3 of ACCOUNT

- Fault injection: after line 54, add a precondition $\text{amount} \leq 10$ for **deposit** to strengthen the precondition.
- Resulting failure: as shown in Fig. 7(a), the injected fault results in a failure of **transfer** — it does not satisfy the new precondition $\text{amount} \leq 10$ when calling **deposit**.
- Cause of the failure: the inconsistency of specification between a supplier routine **deposit** and its client routine **transfer**: when the precondition of a routine is changed, its client routine should be changed accordingly. In this example, the upper limit of **transfer** should be consistent with the upper limit of **deposit**. In other words, the amount of money in a transfer operation should not exceed the maximum amount that is permitted in a deposit operation.
- Proof time: 0.248 sec
- Proof2Fix: 6 valid fixes out of candidate 111 fixes
- Fixing time: 1.38 minutes



Class	Feature	Information	Position	Time
ACCOUNT_3	invariant a...	Verification successful.		0.16
ACCOUNT_3	make (cre...	Verification successful.		0.00
ACCOUNT_3	available_...	Verification successful.		0.03
ACCOUNT_3	set_credit_...	Verification successful.		0.01
ACCOUNT_3	deposit	Verification successful.		
ACCOUNT_3	withdraw	Verification successful.		0.01
ACCOUNT_3	transfer	Precondition amount <= 10 may be violated on call to [ACCOUNT_3].deposit.	91	0.03

(a)

```

1  transfer_ID_85 (amount: INTEGER; other: ACCOUNT_5)
2      -- Transfer 'amount' from this account to 'other'.
3  note
4      explicit: wrapping
5  require
6      other ≠ Void
7      amount_not_negative: amount ≥ 0 and 0 ≥ credit_limit and 0 ≥ other.credit_limit
8      amount_available: amount ≤ available_amount
9      balance_non_negative: balance ≥ credit_limit and other.balance ≥ other.
        credit_limit
10     not ((amount) > (0)) -- **Fix
11 do
12     withdraw (amount)
13     other.deposit (amount)
14 ensure
15     modify_field (["balance", "closed"], [Current, other])
16     withdrawal_made: balance = old balance - amount
17     desposit_made: other.balance = old other.balance + amount
18 end

```

Fig. 7. Fix from Proof2Fix

```

1  transfer_ID_93 (amount: INTEGER; other: ACCOUNT_5)
2      -- Transfer 'amount' from this account to 'other'.
3  note
4      explicit: wrapping
5  require
6      other ≠ Void
7      amount_not_negative: amount ≥ 0 and 0 ≥ credit_limit and 0 ≥ other.credit_limit
8      amount_available: amount ≤ available_amount
9      balance_non_negative: balance ≥ credit_limit and other.balance ≥ other.
        credit_limit
10     not (balance > credit_limit) -- **Fix
11 do
12     withdraw (amount)
13     other.deposit (amount)
14 ensure
15     modify_field (["balance", "closed"], [Current, other])
16     withdrawal_made: balance = old balance - amount
17     desposit_made: other.balance = old other.balance + amount
18 end

```

Fig. 8. Fix from Proof2Fix


```

1  transfer_ID_101 (amount: INTEGER; other: ACCOUNT_5)
2      -- Transfer 'amount' from this account to 'other'.
3  note
4      explicit: wrapping
5  require
6      other ≠ Void
7      amount_not_negative: amount ≥ 0 and 0 ≥ credit_limit and 0 ≥ other.credit_limit
8      amount_available: amount ≤ available_amount
9      balance_non_negative: balance ≥ credit_limit and other.balance ≥ other.
        credit_limit
10     not (credit_limit < amount) -- **Fix
11 do
12     withdraw (amount)
13     other.deposit (amount)
14 ensure
15     modify_field (["balance", "closed"], [Current, other])
16     withdrawal_made: balance = old balance - amount
17     desposit_made: other.balance = old other.balance + amount
18 end

```

Fig. 9. Fix from Proof2Fix

```

1  transfer_ID_103 (amount: INTEGER; other: ACCOUNT_5)
2      -- Transfer 'amount' from this account to 'other'.
3  note
4      explicit: wrapping
5  require
6      other ≠ Void
7      amount_not_negative: amount ≥ 0 and 0 ≥ credit_limit and 0 ≥ other.credit_limit
8      amount_available: amount ≤ available_amount
9      balance_non_negative: balance ≥ credit_limit and other.balance ≥ other.
        credit_limit
10     not (credit_limit < available_amount) -- **Fix
11 do
12     withdraw (amount)
13     other.deposit (amount)
14 ensure
15     modify_field (["balance", "closed"], [Current, other])
16     withdrawal_made: balance = old balance - amount
17     desposit_made: other.balance = old other.balance + amount
18 end

```

Fig. 10. Fix from Proof2Fix

```

1  transfer_ID_109 (amount: INTEGER; other: ACCOUNT_5)
2      -- Transfer 'amount' from this account to 'other'.
3  note
4      explicit: wrapping
5  require
6      other ≠ Void
7      amount_not_negative: amount ≥ 0 and 0 ≥ credit_limit and 0 ≥ other.credit_limit
8      amount_available: amount ≤ available_amount
9      balance_non_negative: balance ≥ credit_limit and other.balance ≥ other.
        credit_limit
10     not (amount > deposit_limit) -- **Fix**
11 do
12     withdraw (amount)
13     other.deposit (amount)
14 ensure
15     modify_field (["balance", "closed"], [Current, other])
16     withdrawal_made: balance = old balance - amount
17     desposit_made: other.balance = old other.balance + amount
18 end

```

Fig. 11. Fix from Proof2Fix

```

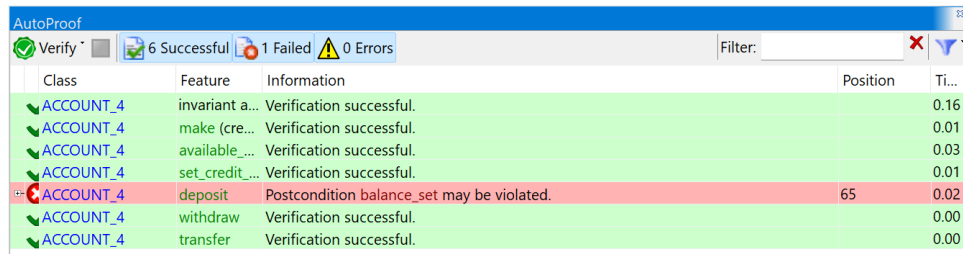
1  transfer_ID_111 (amount: INTEGER; other: ACCOUNT_5)
2      -- Transfer 'amount' from this account to 'other'.
3  note
4      explicit: wrapping
5  require
6      other ≠ Void
7      amount_not_negative: amount ≥ 0 and 0 ≥ credit_limit and 0 ≥ other.credit_limit
8      amount_available: amount ≤ available_amount
9      balance_non_negative: balance ≥ credit_limit and other.balance ≥ other.
        credit_limit
10     not (available_amount > deposit_limit) -- **Fix
11 do
12     withdraw (amount)
13     other.deposit (amount)
14 ensure
15     modify_field (["balance", "closed"], [Current, other])
16     withdrawal_made: balance = old balance - amount
17     desposit_made: other.balance = old other.balance + amount
18 end

```

Fig. 12. Fix from Proof2Fix

Variant 4 of `ACCOUNT`

- Fault injection: at line 56, change the body of `deposit` from “`balance := balance + amount`” into “`balance := balance - amount`”.
- Resulting failure: as shown in Fig. 13(a), the postcondition `balance_set` is violated.
- Cause of the failure: this failure is similar to the failure in Variant 1, which results from the inconsistency between the implementation of `deposit` and its postcondition.
- Proof time: 0.241 sec
- Possible fixes: No valid fixes



The screenshot shows the AutoProof application window. At the top, there is a status bar indicating '6 Successful', '1 Failed', and '0 Errors'. Below this is a table with columns: Class, Feature, Information, Position, and Time. The table lists several features for the ACCOUNT_4 class, most of which are successful. The 'deposit' feature is highlighted in red, indicating a failure, with the message 'Postcondition balance_set may be violated.' and a time of 0.02. The 'withdraw' and 'transfer' features are successful with a time of 0.00.

Class	Feature	Information	Position	Time
ACCOUNT_4	invariant a...	Verification successful.		0.16
ACCOUNT_4	make (cre...	Verification successful.		0.01
ACCOUNT_4	available_...	Verification successful.		0.03
ACCOUNT_4	set_credit_...	Verification successful.		0.01
ACCOUNT_4	deposit	Postcondition <code>balance_set</code> may be violated.	65	0.02
ACCOUNT_4	withdraw	Verification successful.		0.00
ACCOUNT_4	transfer	Verification successful.		0.00

(a)

Variant 5 of ACCOUNT

- Fault injection: at line 87, remove the precondition `other ≠ Current` of `transfer`.
- Resulting failure: as shown in Fig. 13(b), the fault injection leads to the violation of postcondition `withdrawal_made` when verifying `transfer`.
- Cause of the failure: the precondition of `transfer` is too weak; it should exclude the case where an account transfers money to itself.
- Proof time: 0.243 sec
- Fixing time:
- Proof2Fix: 5 valid fixes out of 98 candidate fixes

Class	Feature	Information	Position	Time
ACCOUNT_5	invariant a...	Verification successful.		0.16
ACCOUNT_5	make (cre...	Verification successful.		0.01
ACCOUNT_5	available_...	Verification successful.		0.03
ACCOUNT_5	set_credit_...	Verification successful.		0.01
ACCOUNT_5	deposit	Verification successful.		0.00
ACCOUNT_5	withdraw	Verification successful.		0.01
ACCOUNT_5	transfer	Postcondition <code>withdrawal_made</code> may be violated.	94	0.02

(b)

```

1  transfer_ID_71 (amount: INTEGER; other: ACCOUNT_5)
2      -- Transfer 'amount' from this account to 'other'.
3      note
4          explicit: wrapping
5      require
6          other ≠ Void
7          amount_not_negative: amount ≥ 0 and 0 ≥ credit_limit and 0 ≥ other.credit_limit
8          amount_available: amount ≤ available_amount
9          balance_non_negative: balance ≥ credit_limit and other.balance ≥ other.
              credit_limit
10         not (other = Current) -- **Fix
11     do
12         withdraw (amount)
13         other.deposit (amount)
14     ensure
15         modify_field (["balance", "closed"], [Current, other])
16         withdrawal_made: balance = old balance - amount
17         desposit_made: other.balance = old other.balance + amount
18     end

```

Fig. 13. Fix from Proof2Fix

```

1  transfer_ID_75 (amount: INTEGER; other: ACCOUNT_5)
2      -- Transfer 'amount' from this account to 'other'.
3  note
4      explicit: wrapping
5  require
6      other ≠ Void
7      amount_not_negative: amount ≥ 0 and 0 ≥ credit_limit and 0 ≥ other.credit_limit
8      amount_available: amount ≤ available_amount
9      balance_non_negative: balance ≥ credit_limit and other.balance ≥ other.
        credit_limit
10         not ((other = Void) or else (other = Current))
11  do
12      withdraw (amount)
13      other.deposit (amount)
14  ensure
15      modify_field (["balance", "closed"], [Current, other])
16      withdrawal_made: balance = old balance - amount
17      desposit_made: other.balance = old other.balance + amount
18  end

```

Fig. 14. Fix from Proof2Fix

```

1  transfer_ID_85 (amount: INTEGER; other: ACCOUNT_5)
2      -- Transfer 'amount' from this account to 'other'.
3  note
4      explicit: wrapping
5  require
6      other ≠ Void
7      amount_not_negative: amount ≥ 0 and 0 ≥ credit_limit and 0 ≥ other.credit_limit
8      amount_available: amount ≤ available_amount
9      balance_non_negative: balance ≥ credit_limit and other.balance ≥ other.
        credit_limit
10         not ((amount) > (0))
11  do
12      withdraw (amount)
13      other.deposit (amount)
14  ensure
15      modify_field (["balance", "closed"], [Current, other])
16      withdrawal_made: balance = old balance - amount
17      desposit_made: other.balance = old other.balance + amount
18  end

```

Fig. 15. Fix from Proof2Fix

```

1  transfer_ID_93 (amount: INTEGER; other: ACCOUNT_5)
2      -- Transfer 'amount' from this account to 'other'.
3      note
4          explicit: wrapping
5      require
6          other ≠ Void
7          amount_not_negative: amount ≥ 0 and 0 ≥ credit_limit and 0 ≥ other.credit_limit
8          amount_available: amount ≤ available_amount
9          balance_non_negative: balance ≥ credit_limit and other.balance ≥ other.
              credit_limit
10             not (balance > credit_limit)
11      do
12          withdraw (amount)
13          other.deposit (amount)
14      ensure
15          modify_field (["balance", "closed"], [Current, other])
16          withdrawal_made: balance = old balance - amount
17          desposit_made: other.balance = old other.balance + amount
18      end

```

Fig. 16. Fix from Proof2Fix

```

1  transfer_ID_97 (amount: INTEGER; other: ACCOUNT_5)
2      -- Transfer 'amount' from this account to 'other'.
3      note
4          explicit: wrapping
5      require
6          other ≠ Void
7          amount_not_negative: amount ≥ 0 and 0 ≥ credit_limit and 0 ≥ other.credit_limit
8          amount_available: amount ≤ available_amount
9          balance_non_negative: balance ≥ credit_limit and other.balance ≥ other.
              credit_limit
10             not (credit_limit < amount)
11      do
12
13          withdraw (amount)
14          other.deposit (amount)
15      ensure
16          modify_field (["balance", "closed"], [Current, other])
17          withdrawal_made: balance = old balance - amount
18          desposit_made: other.balance = old other.balance + amount
19      end

```

Fig. 17. Fix from Proof2Fix

Variant 6 of ACCOUNT

- Fault injection: at line 73, remove the postcondition *balance_set* of **withdraw**.
- Resulting failure: as shown in Fig. 18(a), the injected fault results in the violation of postcondition *withdrawal_made* when verifying **transfer**.
- Cause of the failure: the postcondition of **withdraw** is incomplete — not strong enough to represent the functionality of **withdraw**; as a result, when reasoning its client routine **transfer**, the prover is not able to establish the postcondition related to the functionality of **withdraw**.
- Proof time: 0.253 sec
- Possible fixes: No valid fixes

AutoProof				
<div> Verify 6 Successful 1 Failed 0 Errors </div>				
Class	Feature	Information	Position	Ti...
ACCOUNT_6	invariant a...	Verification successful.		0.17
ACCOUNT_6	make (cre...	Verification successful.		0.01
ACCOUNT_6	available_...	Verification successful.		0.03
ACCOUNT_6	set_credit_...	Verification successful.		0.00
ACCOUNT_6	deposit	Verification successful.		0.01
ACCOUNT_6	withdraw	Verification successful.		0.01
ACCOUNT_6	transfer	Postcondition <i>withdrawal_made</i> may be violated.	92	0.02

(a)

Variant 7 of `ACCOUNT`

- Fault injection: at line 60, remove the postcondition `balance_set` of `deposit`.
- Resulting failure: the injected fault, as shown in Fig. 18(b), results in the violation of postcondition `deposit_made` when verifying `transfer`.
- Cause of the failure: similar to the previous failure (in Variant 6), this failure of `transfer` is due to the weakness of the postcondition of its supplier class `deposit` – the postcondition is not strong enough to represent the functionality of `deposit`.
- Proof time: 0.244 sec
- Resulting test case: Fig. ?? shows the test from Proof2Test, which calls `transfer` with input — `Current.balance = 0`, `Current.credit_limit = 0`, `amount = 0`, `other.balance = 0`, and `other.credit_limit = -7720`.
- Possible fixes: No valid fixes

Class	Feature	Information	Position	Ti...
ACCOUNT_7	invariant a...	Verification successful.		0.16
ACCOUNT_7	make (cre...	Verification successful.		0.01
ACCOUNT_7	available_...	Verification successful.		0.03
ACCOUNT_7	set_credit_...	Verification successful.		0.02
ACCOUNT_7	deposit	Verification successful.		0.00
ACCOUNT_7	withdraw	Verification successful.		0.00
ACCOUNT_7	transfer	Postcondition deposit_made may be violated.	93	0.02

(b)

1.2 CLOCK

`CLOCK` class implements a digital clock counting seconds, minutes, and hours. The version of `CLOCK` displaying below is verified successfully. The experiment includes 8 different variants of `CLOCK` class, with different faults injected based on the verified version.

```

1 class
2     CLOCK
3 create
4     make
5 feature {NONE} -- Initialization
6     make
7         note
8             status: creator
9         do
10             hours := 0
11             minutes := 0

```



```

12         seconds := 0
13     ensure
14         modify_model ([“hours”, “minutes”, “seconds”], Current)
15         initialized: hours = 0 and minutes = 0 and seconds = 0
16     end
17 feature -- Access
18     hours: INTEGER
19         -- Hours of clock.
20     minutes: INTEGER
21         -- Minutes of clock.
22     seconds: INTEGER
23         -- Seconds of clock.
24 feature -- Element change
25     set_hours (a_value: INTEGER)
26         -- Set ‘hours’ to ‘a_value’.
27     require
28         valid_hours:  $0 \leq a\_value$  and  $a\_value < 24$ 
29     do
30         hours := a_value
31     ensure
32         hours_set: hours = a_value
33         modify_model (“hours”, Current)
34     end
35     set_minutes (a_value: INTEGER)
36         -- Set ‘minutes’ to ‘a_value’.
37     require
38         valid_minutes:  $0 \leq a\_value$  and  $a\_value < 60$ 
39     do
40         minutes := a_value
41     ensure
42         minutes_set: minutes = a_value
43         modify_model (“minutes”, Current)
44     end
45     set_seconds (a_value: INTEGER)
46         -- Set ‘seconds’ to ‘a_value’.
47     require
48         valid_seconds:  $0 \leq a\_value$  and  $a\_value < 60$ 
49     do
50         seconds := a_value
51     ensure
52         seconds_set: seconds = a_value
53         modify_model (“seconds”, Current)
54     end
55 feature -- Basic operations
56     increase_hours

```

```

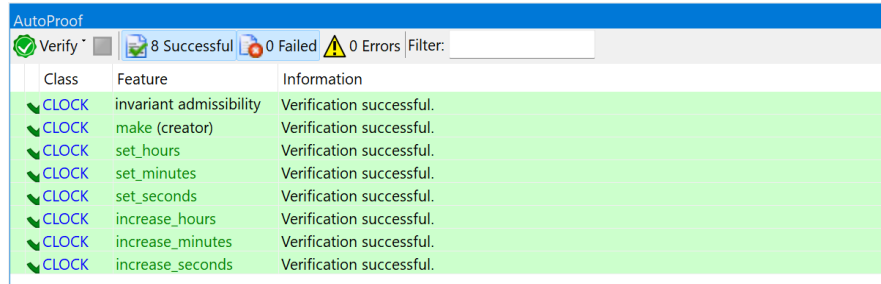
57         -- Increase 'hours' by one.
58     note
59         explicit: wrapping
60     do
61         if hours = 23 then
62             set_hours (0)
63         else
64             set_hours (hours + 1)
65         end
66     ensure
67         hours_increased: hours = (old hours + 1) \ 24
68         modify_model ("hours", Current)
69     end
70 increase_minutes
71     -- Increase 'minutes' by one.
72     note
73         explicit: wrapping
74     do
75         if minutes < 59 then
76             set_minutes (minutes + 1)
77         else
78             set_minutes (0)
79             increase_hours
80         end
81     ensure
82         hours_increased: old minutes = 59 implies
83             hours = (old hours + 1) \ 24
84         hours_unchanged: old minutes < 59 implies
85             hours = old hours
86         minutes_increased: minutes = (old minutes + 1) \ 60
87         modify_model (["minutes", "hours"], Current)
88     end
89 increase_seconds
90     -- Increase 'seconds' by one.
91     note
92         explicit: wrapping
93     do
94         if seconds ≥ 59 then
95             set_seconds (0)
96             increase_minutes
97         else
98             set_seconds (seconds + 1)
99         end
100    ensure
101        hours_increased: old seconds = 59 and old minutes = 59 implies

```

```

102             hours = (old hours + 1) \ 24
103     hours_unchanged: old seconds < 59 or old minutes < 59 implies
104             hours = old hours
105     minutes_increased: old seconds = 59 implies
106             minutes = (old minutes + 1) \ 60
107     minutes_unchanged: old seconds < 59 implies
108             minutes = old minutes
109     seconds_increased: seconds = (old seconds + 1) \ 60
110     modify_model ([“seconds”, “minutes”, “hours”], Current)
111     end
112 invariant
113     hours_valid: 0 ≤ hours and hours ≤ 23
114     minutes_valid: 0 ≤ minutes and minutes ≤ 59
115     seconds_valid: 0 ≤ seconds and seconds ≤ 59
116 end

```



The screenshot shows the AutoProof application window. At the top, there is a status bar with a green checkmark icon, the text 'Verify', and a summary: '8 Successful', '0 Failed', and '0 Errors'. Below this is a table with three columns: 'Class', 'Feature', and 'Information'. The table contains eight rows, all of which are highlighted in green, indicating successful verification.

Class	Feature	Information
✓CLOCK	invariant admissibility	Verification successful.
✓CLOCK	make (creator)	Verification successful.
✓CLOCK	set_hours	Verification successful.
✓CLOCK	set_minutes	Verification successful.
✓CLOCK	set_seconds	Verification successful.
✓CLOCK	increase_hours	Verification successful.
✓CLOCK	increase_minutes	Verification successful.
✓CLOCK	increase_seconds	Verification successful.

Fig. 18. Proof result of **CLOCK** in AutoProof

Variant 1 of **CLOCK**

- Fault injection: at line 73, in the **increase_hours** procedure, change the condition of the then branch from “**hours** = 23” into “**hours** = 24”.
- Resulting failure: as shown in Fig. 19(a), the injected fault results in the violation of precondition **valid_hours** when calling **set_hours** from **increase_hours**.
- Cause of the failure: incorrect implementation of the routine body.
- Proof time: 0.253 sec
- Proof2Fix: 5 valid fixes out of 356 candidate fixes
- Fixing time: 2.72 minutes

AutoProof		
<div> Verify 7 Successful 1 Failed 0 Errors Filter: <input type="text"/> </div>		
Class	Feature	Information
CLOCK_1	invariant admissibility	Verification successful.
CLOCK_1	make (creator)	Verification successful.
CLOCK_1	set_hours	Verification successful.
CLOCK_1	set_minutes	Verification successful.
CLOCK_1	set_seconds	Verification successful.
CLOCK_1	increase_hours	Precondition <code>valid_hours</code> may be violated on call to <code>(CLOCK_1).set_hours</code> .
CLOCK_1	increase_minutes	Verification successful.
CLOCK_1	increase_seconds	Verification successful.

(a)

```

1  increase_hours_ID_185
2      -- Increase 'hours' by one.
3      note
4      explicit: wrapping
5      do
6          if hours = 23 then -- **Fix
7              hours := 0 -- **Fix
8          else
9              if hours = 24 then
10                 set_hours (0)
11             else
12                 set_hours (hours + 1)
13             end
14         end
15     ensure
16         hours_increased: hours = (old hours + 1) \ 24
17         modify_model ("hours", Current)
18     end

```

Fig. 19. Fix from Proof2Fix

```

1  increase_hours_ID_44
2      -- Increase 'hours' by one.
3  note
4      explicit: wrapping
5  do
6      if (hours) > (0) then -- **Fix
7          increase_hours -- **Fix
8      else
9          if hours = 24 then
10             set_hours (0)
11         else
12             set_hours (hours + 1)
13         end
14     end
15 ensure
16     hours_increased: hours = (old hours + 1) \ 24
17     modify_model ("hours", Current)
18 end

```

Fig. 20. Fix from Proof2Fix

```

1  increase_hours_ID_78
2      -- Increase 'hours' by one.
3  note
4      explicit: wrapping
5  do
6      if (hours) ≥ (0) then -- **Fix
7          increase_hours -- **Fix
8      else
9          if hours = 24 then
10             set_hours (0)
11         else
12             set_hours (hours + 1)
13         end
14     end
15 ensure
16     hours_increased: hours = (old hours + 1) \ 24
17     modify_model ("hours", Current)
18 end

```

Fig. 21. Fix from Proof2Fix

```

1  increase_hours_ID_180
2      -- Increase 'hours' by one.
3  note
4      explicit: wrapping
5  do
6      if hours = 23 then -- **Fix
7          increase_hours -- **Fix
8      else
9          if hours = 24 then
10             set_hours (0)
11          else
12             set_hours (hours + 1)
13          end
14      end
15  ensure
16      hours_increased: hours = (old hours + 1) \ 24
17      modify_model ("hours", Current)
18  end

```

Fig. 22. Fix from Proof2Fix

```

1  increase_hours_ID_342
2      -- Increase 'hours' by one.
3  note
4      explicit: wrapping
5      require -- **Fix
6          not ((hours) > (0)) -- **Fix
7  do
8      if hours = 24 then
9          set_hours (0)
10     else
11         set_hours (hours + 1)
12     end
13  ensure
14      hours_increased: hours = (old hours + 1) \ 24
15      modify_model ("hours", Current)
16  end

```

Fig. 23. Fix from Proof2Fix

```

1  increase_hours_ID_350
2      -- Increase 'hours' by one.
3  note
4      explicit: wrapping
5      require
6          not (hours = 23) -- **Fix
7  do
8      if hours = 24 then
9          set_hours (0)
10     else
11         set_hours (hours + 1)
12     end
13 ensure
14     hours_increased: hours = (old hours + 1) \ 24
15     modify_model ("hours", Current)
16 end

```

Fig. 24. Fix from Proof2Fix

Variant 2 of `CLOCK`

- Fault injection: at line 92, remove the call `increase_hours` in the body of `increase_minutes`.
- Resulting failure: as shown in Fig. 25(a), the postcondition `hours_increased` is not satisfied in the proof of `increase_minutes`.
- Cause of the failure: incorrect implementation of the routine body.
- Proof time: 0.251 sec
- Proof2Fix: 11 valid fixes out of 301 candidate fixes
- fixing time: 2.43 minutes

AutoProof				
<div> Verify 7 Successful 1 Failed 0 Errors </div>				
Class	Feature	Information	P...	Ti...
✓ <code>CLOCK_2</code>	invariant a...	Verification successful.		0.16
✓ <code>CLOCK_2</code>	make (cre...	Verification successful.		0.03
✓ <code>CLOCK_2</code>	set_hours	Verification successful.		0.01
✓ <code>CLOCK_2</code>	set_minutes	Verification successful.		
✓ <code>CLOCK_2</code>	set_seconds	Verification successful.		0.00
✓ <code>CLOCK_2</code>	increase_h...	Verification successful.		0.05
✗ <code>CLOCK_2</code>	increase_...	Postcondition <code>hours_increased</code> may be violated.	106	0.00
✓ <code>CLOCK_2</code>	increase_s...	Verification successful.		0.00

(a)

```

1  increase_minutes_ID_50
2      -- Increase 'minutes' by one.
3  note
4      explicit: wrapping
5  do
6      if (minutes) > (0) then -- **Fix
7          increase_minutes -- **Fix
8      else
9          if minutes < 59 then
10             set_minutes (minutes + 1)
11         else
12             set_minutes (0)
13         end
14     end
15 ensure
16     hours_increased: old minutes = 59 implies hours = (old hours + 1) \\\ 24
17     hours_unchanged: old minutes < 59 implies hours = old hours
18     minutes_increased: minutes = (old minutes + 1) \\\ 60
19     modify_model (["minutes", "hours"], Current)
20 end

```

Fig. 25. Fix from Proof2Fix

```

1  increase_minutes_ID_84
2      -- Increase 'minutes' by one.
3  note
4      explicit: wrapping
5  do
6      if (minutes) ≥ (0) then -- **Fix
7          increase_minutes -- **Fix
8      else
9          if minutes < 59 then
10             set_minutes (minutes + 1)
11         else
12             set_minutes (0)
13         end
14     end
15 ensure
16     hours_increased: old minutes = 59 implies hours = (old hours + 1) \\\ 24
17     hours_unchanged: old minutes < 59 implies hours = old hours
18     minutes_increased: minutes = (old minutes + 1) \\\ 60
19     modify_model (["minutes", "hours"], Current)
20 end

```

Fig. 26. Fix from Proof2Fix


```

1  increase_minutes_ID_177
2      -- Increase 'minutes' by one.
3  note
4      explicit: wrapping
5  do
6      if minutes = 59 then -- **Fix**
7          increase_hours -- **Fix**
8      end
9      if minutes < 59 then
10         set_minutes (minutes + 1)
11     else
12         set_minutes (0)
13     end
14 ensure
15     hours_increased: old minutes = 59 implies hours = (old hours + 1) \\\ 24
16     hours_unchanged: old minutes < 59 implies hours = old hours
17     minutes_increased: minutes = (old minutes + 1) \\\ 60
18     modify_model (["minutes", "hours"], Current)
19 end

```

Fig. 27. Fix from Proof2Fix

```

1  increase_minutes_ID_186
2      -- Increase 'minutes' by one.
3  note
4      explicit: wrapping
5  do
6      if minutes = 59 then -- **Fix
7          increase_minutes -- **Fix
8      end
9      if minutes < 59 then
10         set_minutes (minutes + 1)
11     else
12         set_minutes (0)
13     end
14 ensure
15     hours_increased: old minutes = 59 implies hours = (old hours + 1) \\\ 24
16     hours_unchanged: old minutes < 59 implies hours = old hours
17     minutes_increased: minutes = (old minutes + 1) \\\ 60
18     modify_model (["minutes", "hours"], Current)
19 end

```

Fig. 28. Fix from Proof2Fix

```

1  increase_minutes_ID_230
2      -- Increase 'minutes' by one.
3  note
4      explicit: wrapping
5  do
6      if hours < minutes then -- **Fix
7          increase_minutes -- **Fix
8      end
9      if minutes < 59 then
10         set_minutes (minutes + 1)
11     else
12         set_minutes (0)
13     end
14 ensure
15     hours_increased: old minutes = 59 implies hours = (old hours + 1) \\\ 24
16     hours_unchanged: old minutes < 59 implies hours = old hours
17     minutes_increased: minutes = (old minutes + 1) \\\ 60
18     modify_model (["minutes", "hours"], Current)
19 end

```

Fig. 29. Fix from Proof2Fix

```

1  increase_minutes_ID_287
2      -- Increase 'minutes' by one.
3  note
4      explicit: wrapping
5  do
6      if minutes < 59 then
7          set_minutes (minutes + 1)
8      else
9          set_minutes (0)
10     end
11 ensure
12     hours_increased: not ((minutes) = (0)) implies old minutes = 59 implies hours = (
13         old hours + 1) \\\ 24 -- **Fix
14     hours_unchanged: old minutes < 59 implies hours = old hours
15     minutes_increased: minutes = (old minutes + 1) \\\ 60
16     modify_model (["minutes", "hours"], Current)
17 end

```

Fig. 30. Fix from Proof2Fix

```

1  increase_minutes_ID_288
2      -- Increase 'minutes' by one.
3  note
4      explicit: wrapping
5      require
6          not ((minutes) > (0)) -- **Fix
7  do
8      if minutes < 59 then
9          set_minutes (minutes + 1)
10     else
11         set_minutes (0)
12     end
13 ensure
14     hours_increased: old minutes = 59 implies hours = (old hours + 1) \ 24
15     hours_unchanged: old minutes < 59 implies hours = old hours
16     minutes_increased: minutes = (old minutes + 1) \ 60
17     modify_model (["minutes", "hours"], Current)
18 end

```

Fig. 31. Fix from Proof2Fix

```

1  increase_minutes_ID_291
2      -- Increase 'minutes' by one.
3  note
4      explicit: wrapping
5  do
6      if minutes < 59 then
7          set_minutes (minutes + 1)
8      else
9          set_minutes (0)
10     end
11 ensure
12     hours_increased: not ((minutes) ≥ (0)) implies old minutes = 59 implies hours = (
13         old hours + 1) \ 24 -- **Fix
14     hours_unchanged: old minutes < 59 implies hours = old hours
15     minutes_increased: minutes = (old minutes + 1) \ 60
16     modify_model (["minutes", "hours"], Current)
17 end

```

Fig. 32. Fix from Proof2Fix

```

1  increase_minutes_ID_295
2      -- Increase 'minutes' by one.
3  note
4      explicit: wrapping
5  do
6      if minutes < 59 then
7          set_minutes (minutes + 1)
8      else
9          set_minutes (0)
10     end
11 ensure
12     hours_increased: not ((minutes) ≤ (0)) implies old minutes = 59 implies hours =
        (old hours + 1) \ 24 -- **Fix
13     hours_unchanged: old minutes < 59 implies hours = old hours
14     minutes_increased: minutes = (old minutes + 1) \ 60
15     modify_model (["minutes", "hours"], Current)
16 end

```

Fig. 33. Fix from Proof2Fix

```

1  increase_minutes_ID_296
2      -- Increase 'minutes' by one.
3  note
4      explicit: wrapping
5      require
6          not (minutes = 59) -- **Fix
7  do
8      if minutes < 59 then
9          set_minutes (minutes + 1)
10     else
11         set_minutes (0)
12     end
13 ensure
14     hours_increased: old minutes = 59 implies hours = (old hours + 1) \ 24
15     hours_unchanged: old minutes < 59 implies hours = old hours
16     minutes_increased: minutes = (old minutes + 1) \ 60
17     modify_model (["minutes", "hours"], Current)
18 end

```

Fig. 34. Fix from Proof2Fix

```

1  increase_minutes_ID_298
2      -- Increase 'minutes' by one.
3  note
4      explicit: wrapping
5      require
6          not (hours < minutes) -- **Fix
7  do
8      if minutes < 59 then
9          set_minutes (minutes + 1)
10     else
11         set_minutes (0)
12     end
13 ensure
14     hours_increased: old minutes = 59 implies hours = (old hours + 1) \ \ 24
15     hours_unchanged: old minutes < 59 implies hours = old hours
16     minutes_increased: minutes = (old minutes + 1) \ \ 60
17     modify_model (["minutes", "hours"], Current)
18 end

```

Fig. 35. Fix from Proof2Fix

Variant 3 of CLOCK

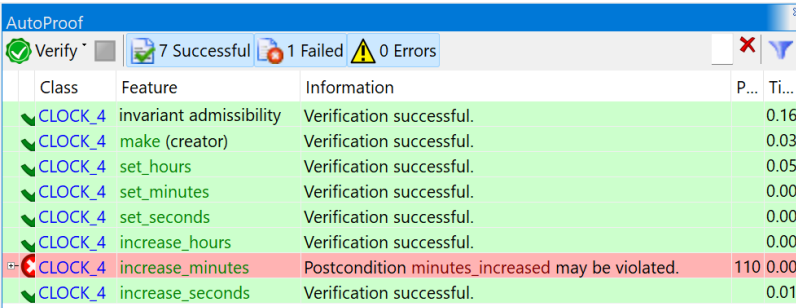
- Fault injection: at line 79, remove the postcondition of `hours_increased` in the `increased_hours` procedure.
- Resulting failure: as shown in Fig. 36(a), the injected fault leads to violation of postcondition `hours_increased` when verifying `increase_minutes`.
- Cause of the failure: the postcondition of `increase_hours` is too weak to express the full functionality of the routine.
- Proof time: 0.263 sec
- Proof2Fix: No valid fixes

AutoProof				
Verify 7 Successful 1 Failed 0 Errors				
Class	Feature	Information	P...	Ti...
CLOCK_3	invariant admissibility	Verification successful.		0.16
CLOCK_3	make (creator)	Verification successful.		0.03
CLOCK_3	set_hours	Verification successful.		0.00
CLOCK_3	set_minutes	Verification successful.		0.00
CLOCK_3	set_seconds	Verification successful.		0.01
CLOCK_3	increase_hours	Verification successful.		0.03
CLOCK_3	increase_minutes	Postcondition hours_increased may be violated.	109	0.01
CLOCK_3	increase_seconds	Verification successful.		0.01

(a)

Variant 4 of `CLOCK`

- Fault injection: at line 51, remove the postcondition of `minutes_set` in the `set_minutes` procedure.
- Resulting failure: as shown in Fig. 36(b), the postcondition `minutes_increased` is not satisfied when verifying the `increase_minutes` procedure.
- Cause of the failure: the postcondition of supplier routine `set_minutes` is too weak to represent its full functionality.
- Proof time: 0.259 sec
- Proof2Fix: No valid fixes

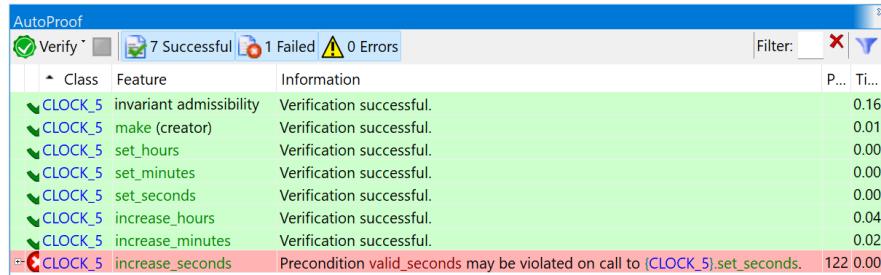


	Class	Feature	Information	P...	Ti...
✓	CLOCK_4	invariant admissibility	Verification successful.		0.16
✓	CLOCK_4	make (creator)	Verification successful.		0.03
✓	CLOCK_4	set_hours	Verification successful.		0.05
✓	CLOCK_4	set_minutes	Verification successful.		0.00
✓	CLOCK_4	set_seconds	Verification successful.		0.00
✓	CLOCK_4	increase_hours	Verification successful.		0.00
✗	CLOCK_4	increase_minutes	Postcondition <code>minutes_increased</code> may be violated.	110	0.00
✓	CLOCK_4	increase_seconds	Verification successful.		0.01

(b)

Variant 5 of `CLOCK`

- Fault injection: at line 108, change the condition of the then branch from “seconds \geq 59” into “seconds>59”.
- Resulting failure: as shown in Fig. 36(c), the injected fault leads to the violation of the precondition `valid_seconds` of `set_seconds` when it is called from the procedure `increase_seconds`.
- Cause of the failure: incorrect implementation of the routine body of `increase_seconds`.
- Proof time: 0.241 sec
- Comment: this test is useful (the values in the test input are meaningful) as its execution shows a specific path that leads to the same contract violation as in the proof.
- Proof2Fix: 8 valid fixes out of 356 candidate fixes.
- fixing time: 3.1 minutes



Class	Feature	Information	P...	Ti...
CLOCK_5	invariant admissibility	Verification successful.		0.16
CLOCK_5	make (creator)	Verification successful.		0.01
CLOCK_5	set_hours	Verification successful.		0.00
CLOCK_5	set_minutes	Verification successful.		0.00
CLOCK_5	set_seconds	Verification successful.		0.00
CLOCK_5	increase_hours	Verification successful.		0.04
CLOCK_5	increase_minutes	Verification successful.		0.02
CLOCK_5	increase_seconds	Precondition <code>valid_seconds</code> may be violated on call to <code>(CLOCK_5).set_seconds</code> .	122	0.00

(c)

```

1  increase_seconds_ID_174
2      -- Increase 'seconds' by one.
3  note
4      explicit: wrapping
5  do
6      if seconds = 59 then -- **Fix**
7          seconds := 0 -- **Fix**
8      end
9      if seconds > 59 then
10         set_seconds (0)
11         increase_minutes
12     else
13         set_seconds (seconds + 1)
14     end
15 ensure
16     hours_increased: old seconds = 59 and old minutes = 59 implies hours = (old hours
17         + 1) \ 24
18     hours_unchanged: old seconds < 59 or old minutes < 59 implies hours = old hours
19     minutes_increased: old seconds = 59 implies minutes = (old minutes + 1) \ 60
20     minutes_unchanged: old seconds < 59 implies minutes = old minutes
21     seconds_increased: seconds = (old seconds + 1) \ 60
22     modify_model (["seconds", "minutes", "hours"], Current)
end

```

Fig. 36. Test case from failed proof of *withdrawal_made*


```

1  increase_seconds_ID_51
2      -- Increase 'seconds' by one.
3      note
4      explicit: wrapping
5      do
6          if (seconds) > (0) then
7              increase_seconds
8          else
9              if seconds > 59 then
10                 set_seconds (0)
11                 increase_minutes
12             else
13                 set_seconds (seconds + 1)
14             end
15         end
16     ensure
17         hours_increased: old seconds = 59 and old minutes = 59 implies hours = (old hours
18             + 1) \ \ 24
19         hours_unchanged: old seconds < 59 or old minutes < 59 implies hours = old hours
20         minutes_increased: old seconds = 59 implies minutes = (old minutes + 1) \ \ 60
21         minutes_unchanged: old seconds < 59 implies minutes = old minutes
22         seconds_increased: seconds = (old seconds + 1) \ \ 60
23         modify_model (["seconds", "minutes", "hours"], Current)
24     end

```

Fig. 37. Test case from failed proof of *withdrawal_made*

```

1   increase_seconds_ID_85
2       -- Increase 'seconds' by one.
3   note
4       explicit: wrapping
5   do
6       if (seconds) ≥ (0) then
7   increase_seconds
8   else
9       if seconds > 59 then
10          set_seconds (0)
11          increase_minutes
12      else
13          set_seconds (seconds + 1)
14      end
15  end
16  ensure
17      hours_increased: old seconds = 59 and old minutes = 59 implies hours = (old hours
18          + 1) \ 24
19      hours_unchanged: old seconds < 59 or old minutes < 59 implies hours = old hours
20      minutes_increased: old seconds = 59 implies minutes = (old minutes + 1) \ 60
21      minutes_unchanged: old seconds < 59 implies minutes = old minutes
22      seconds_increased: seconds = (old seconds + 1) \ 60
23      modify_model (["seconds", "minutes", "hours"], Current)
24  end

```

Fig. 38. Test case from failed proof of *withdrawal_made*

```

1   increase_seconds_ID_187
2       -- Increase 'seconds' by one.
3       note
4           explicit: wrapping
5       do
6           if seconds = 59 then
7 increase_seconds
8 else
9         if seconds > 59 then
10            set_seconds (0)
11            increase_minutes
12        else
13            set_seconds (seconds + 1)
14        end
15    end
16    ensure
17        hours_increased: old seconds = 59 and old minutes = 59 implies hours = (old hours
18            + 1) \ \ 24
19        hours_unchanged: old seconds < 59 or old minutes < 59 implies hours = old hours
20        minutes_increased: old seconds = 59 implies minutes = (old minutes + 1) \ \ 60
21        minutes_unchanged: old seconds < 59 implies minutes = old minutes
22        seconds_increased: seconds = (old seconds + 1) \ \ 60
23        modify_model (["seconds", "minutes", "hours"], Current)
24    end

```

Fig. 39. Test case from failed proof of *withdrawal_made*

```

1   increase_seconds_ID_285
2       -- Increase 'seconds' by one.
3       note
4           explicit: wrapping
5       do
6           if hours < seconds then
7 increase_seconds
8 else
9         if seconds > 59 then
10            set_seconds (0)
11            increase_minutes
12        else
13            set_seconds (seconds + 1)
14        end
15    end
16    ensure
17        hours_increased: old seconds = 59 and old minutes = 59 implies hours = (old hours
18            + 1) \ \ 24
19        hours_unchanged: old seconds < 59 or old minutes < 59 implies hours = old hours
20        minutes_increased: old seconds = 59 implies minutes = (old minutes + 1) \ \ 60
21        minutes_unchanged: old seconds < 59 implies minutes = old minutes
22        seconds_increased: seconds = (old seconds + 1) \ \ 60
23        modify_model (["seconds", "minutes", "hours"], Current)
24    end

```

Fig. 40. Test case from failed proof of *withdrawal_made*

```

1   increase_seconds_ID_342
2       -- Increase 'seconds' by one.
3   note
4       explicit: wrapping
5   require
6   not ((seconds) > (0))
7   do
8       if seconds > 59 then
9           set_seconds (0)
10          increase_minutes
11      else
12          set_seconds (seconds + 1)
13      end
14  ensure
15      hours_increased: old seconds = 59 and old minutes = 59 implies hours = (old hours
16          + 1) \ \ 24
17      hours_unchanged: old seconds < 59 or old minutes < 59 implies hours = old hours
18      minutes_increased: old seconds = 59 implies minutes = (old minutes + 1) \ \ 60
19      minutes_unchanged: old seconds < 59 implies minutes = old minutes
20      seconds_increased: seconds = (old seconds + 1) \ \ 60
21      modify_model (["seconds", "minutes", "hours"], Current)
22  end

```

Fig. 41. Test case from failed proof of *withdrawal_made*

Variant 6 of **CLOCK**

- Fault injection: at line 95, remove the postcondition `hours_increased` of `increase_minutes` procedure.
- Resulting failure: as shown in Fig. 44(a), the injected fault results in the violation of the postcondition `hours_increased` of `increase_seconds`.
- Cause of the failure: the postcondition of the routine `increase_minutes` is too weak to represent its full functionality.
- Proof time: 0.243 sec
- Possible fixes: No valid fixes

```

1   increase_seconds_ID_350
2       -- Increase 'seconds' by one.
3   note
4       explicit: wrapping
5   require
6   not (seconds = 59)
7   do
8       if seconds > 59 then
9           set_seconds (0)
10          increase_minutes
11      else
12          set_seconds (seconds + 1)
13      end
14  ensure
15      hours_increased: old seconds = 59 and old minutes = 59 implies hours = (old hours
16          + 1) \ \ 24
17      hours_unchanged: old seconds < 59 or old minutes < 59 implies hours = old hours
18      minutes_increased: old seconds = 59 implies minutes = (old minutes + 1) \ \ 60
19      minutes_unchanged: old seconds < 59 implies minutes = old minutes
20      seconds_increased: seconds = (old seconds + 1) \ \ 60
21      modify_model (["seconds", "minutes", "hours"], Current)
22  end

```

Fig. 42. Test case from failed proof of *withdrawal_made*

Variant 7 of **CLOCK**

- Fault injection: at line 99, remove the postcondition of `minutes_increased` of `increase_minutes` procedure.
- Resulting failure: as shown in Fig. 44(b), the postcondition `hours_increased` is violated.
- Cause of the failure: the postcondition of the routine `increase_minutes` is too weak to represent its full functionality.
- Proof time: 0.248 sec
- Possible fixes: No valid fixes

```

1  increase_seconds_ID_354
2      -- Increase 'seconds' by one.
3      note
4          explicit: wrapping
5  require
6  not (hours < seconds)
7      do
8          if seconds > 59 then
9              set_seconds (0)
10             increase_minutes
11         else
12             set_seconds (seconds + 1)
13         end
14     ensure
15         hours_increased: old seconds = 59 and old minutes = 59 implies hours = (old hours
16             + 1) \ \ 24
17         hours_unchanged: old seconds < 59 or old minutes < 59 implies hours = old hours
18         minutes_increased: old seconds = 59 implies minutes = (old minutes + 1) \ \ 60
19         minutes_unchanged: old seconds < 59 implies minutes = old minutes
20         seconds_increased: seconds = (old seconds + 1) \ \ 60
21         modify_model ([ "seconds", "minutes", "hours"], Current)
22     end

```

Fig. 43. Test case from failed proof of *withdrawal_made*

AutoProof				
<div> <div>Verify</div> <div>7 Successful</div> <div>1 Failed</div> <div>0 Errors</div> </div>				
Class	Feature	Information	P...	Ti...
CLOCK_6	invariant admissibility	Verification successful.		0.13
CLOCK_6	make (creator)	Verification successful.		0.01
CLOCK_6	set_hours	Verification successful.		
CLOCK_6	set_minutes	Verification successful.		0.00
CLOCK_6	set_seconds	Verification successful.		0.00
CLOCK_6	increase_hours	Verification successful.		0.04
CLOCK_6	increase_minutes	Verification successful.		0.02
CLOCK_6	increase_seconds	Postcondition hours_increased may be violated.		125 0.04

(a)

AutoProof				
<div> <div>Verify</div> <div>7 Successful</div> <div>1 Failed</div> <div>0 Errors</div> </div>				
Class	Feature	Information	P...	Ti...
CLOCK_7	invariant admissibility	Verification successful.		0.13
CLOCK_7	make (creator)	Verification successful.		0.02
CLOCK_7	set_hours	Verification successful.		0.00
CLOCK_7	set_minutes	Verification successful.		
CLOCK_7	set_seconds	Verification successful.		0.05
CLOCK_7	increase_hours	Verification successful.		0.00
CLOCK_7	increase_minutes	Verification successful.		0.00
CLOCK_7	increase_seconds	Postcondition minutes_increased may be violated.		129 0.04

(b)

Variant 8 of `CLOCK`

- Fault injection: at line 88, change the condition of the then branch from “minutes < 59” into “minutes ≤ 59” in the `increase_minutes` procedure.
- Resulting failure: as shown in Fig. 44(c), the injected fault leads to the violation of precondition `valid_minutes` of the routine `set_minutes` when calling it from `increase_minutes`.
- Cause of the failure: incorrect implementation of routine body of `increase_minutes`.
- Proof time: 0.245 sec
- Possible fixes: 8 valid fixes out 356 candidate fixes
- fixing time: 3.18 minutes

Class	Feature	Information	P...	Ti...
CLOCK_8	invariant admissibility	Verification successful.		0.17
CLOCK_8	make (creator)	Verification successful.		0.02
CLOCK_8	set_hours	Verification successful.		0.00
CLOCK_8	set_minutes	Verification successful.		
CLOCK_8	set_seconds	Verification successful.		0.00
CLOCK_8	increase_hours	Verification successful.		0.04
CLOCK_8	increase_minutes	Precondition <code>valid_minutes</code> may be violated on call to <code>(CLOCK_8).set_minutes</code> .	100	0.00
CLOCK_8	increase_seconds	Verification successful.		0.01

(c)

```

1  increase_minutes_ID_50
2      -- Increase 'minutes' by one.
3      note
4          explicit: wrapping
5      do
6
7          if (minutes) > (0) then
8              increase_minutes
9          else
10             if minutes ≤ 59 then
11                 set_minutes (minutes + 1)
12                 -- set_minutes (0)
13                 -- increase_hours
14             else
15                 set_minutes (0)
16                 increase_hours
17             end
18         end
19     ensure
20         hours_increased: old minutes = 59 implies hours = (old hours + 1) \\ 24
21         hours_unchanged: old minutes < 59 implies hours = old hours
22         minutes_increased: minutes = (old minutes + 1) \\ 60

```



```

23     modify_model (["minutes", "hours"], Current)
24 end
25
26
27 increase_minutes_ID_84
28     -- Increase 'minutes' by one.
29     note
30         explicit: wrapping
31     do
32
33         if (minutes) ≥ (0) then
34             increase_minutes
35         else
36             if minutes ≤ 59 then
37                 set_minutes (minutes + 1)
38                 -- set_minutes (0)
39                 -- increase_hours
40             else
41                 set_minutes (0)
42                 increase_hours
43             end
44         end
45     ensure
46         hours_increased: old minutes = 59 implies hours = (old hours + 1) \\ 24
47         hours_unchanged: old minutes < 59 implies hours = old hours
48         minutes_increased: minutes = (old minutes + 1) \\ 60
49     modify_model (["minutes", "hours"], Current)
50 end
51
52 increase_minutes_ID_186
53     -- Increase 'minutes' by one.
54     note
55         explicit: wrapping
56     do
57
58         if minutes = 59 then
59             increase_minutes
60         else
61             if minutes ≤ 59 then
62                 set_minutes (minutes + 1)
63                 -- set_minutes (0)
64                 -- increase_hours
65             else
66                 set_minutes (0)
67                 increase_hours
68             end
69         end
70     ensure
71         hours_increased: old minutes = 59 implies hours = (old hours + 1) \\ 24
72         hours_unchanged: old minutes < 59 implies hours = old hours

```

```

73         minutes_increased: minutes = (old minutes + 1) \\ 60
74     modify_model (["minutes", "hours"], Current)
75 end
76
77 increase_minutes_ID_230
78     -- Increase 'minutes' by one.
79     note
80         explicit: wrapping
81     do
82
83         if hours < minutes then
84             increase_minutes
85         else
86             if minutes ≤ 59 then
87                 set_minutes (minutes + 1)
88                 -- set_minutes (0)
89                 -- increase_hours
90             else
91                 set_minutes (0)
92                 increase_hours
93             end
94         end
95     ensure
96         hours_increased: old minutes = 59 implies hours = (old hours + 1) \\ 24
97         hours_unchanged: old minutes < 59 implies hours = old hours
98         minutes_increased: minutes = (old minutes + 1) \\ 60
99     modify_model (["minutes", "hours"], Current)
100 end
101
102 increase_minutes_ID_342
103     -- Increase 'minutes' by one.
104     note
105         explicit: wrapping
106     require
107     not ((minutes) > (0))
108     do
109
110         if minutes ≤ 59 then
111             set_minutes (minutes + 1)
112             -- set_minutes (0)
113             -- increase_hours
114         else
115             set_minutes (0)
116             increase_hours
117         end
118     ensure
119         hours_increased: old minutes = 59 implies hours = (old hours + 1) \\ 24
120         hours_unchanged: old minutes < 59 implies hours = old hours
121         minutes_increased: minutes = (old minutes + 1) \\ 60
122     modify_model (["minutes", "hours"], Current)

```

```

123     end
124
125
126 increase_minutes_ID_350
127     -- Increase 'minutes' by one.
128     note
129         explicit: wrapping
130     require
131     not (minutes = 59)
132     do
133
134         if minutes ≤ 59 then
135             set_minutes (minutes + 1)
136             -- set_minutes (0)
137             -- increase_hours
138         else
139             set_minutes (0)
140             increase_hours
141         end
142     ensure
143         hours_increased: old minutes = 59 implies hours = (old hours + 1) \ 24
144         hours_unchanged: old minutes < 59 implies hours = old hours
145         minutes_increased: minutes = (old minutes + 1) \ 60
146         modify_model (["minutes", "hours"], Current)
147     end
148
149
150 increase_minutes_ID_352
151     -- Increase 'minutes' by one.
152     note
153         explicit: wrapping
154     require
155         not (hours < minutes)
156     do
157
158         if minutes ≤ 59 then
159             set_minutes (minutes + 1)
160             -- set_minutes (0)
161             -- increase_hours
162         else
163             set_minutes (0)
164             increase_hours
165         end
166     ensure
167         hours_increased: old minutes = 59 implies hours = (old hours + 1) \ 24
168         hours_unchanged: old minutes < 59 implies hours = old hours
169         minutes_increased: minutes = (old minutes + 1) \ 60
170         modify_model (["minutes", "hours"], Current)
171     end

```

1.3 ARITHMETIC

```
1  note
2    description: "Implementation of arithmetic operations based on increment."
3  class
4    ARITHMETIC_1
5  create
6    make
7  feature -- Initialization
8    make
9    do
10   end
11 feature -- Addition
12   add (a, b: INTEGER): INTEGER
13     -- Add two numbers by repeated increment.
14     -- Iterative version.
15   require
16     a_in_range:  $a \geq -100$  and  $a \leq 100$ 
17     b_in_range:  $b \geq -100$  and  $b \leq 100$ 
18   local
19     i: INTEGER
20   do
21     if  $b \geq 0$  then
22       from
23         Result := a
24         i := b
25       invariant
26         Result = a + (b - i)
27          $0 \leq i$  and  $i \leq b$ 
28       until
29         i = 0
30       loop
31         Result := Result + 1
32         i := i - 1
33       variant
34         i
35     end
36   else
37     from
38       Result := a
39       i := b
40     invariant
41       Result = a + (b - i)
42        $b \leq i$  and  $i \leq 0$ 
43     until
44       i = 0
45     loop
46       Result := Result - 1
47       i := i + 1
48     variant
```

```

49         - i
50     end
51 end
52 ensure
53     result_correct: Result = a + b
54 end
55 add_recursive (a, b: INTEGER): INTEGER
56     -- Add two numbers by repeated increment.
57     -- Recursive version.
58     require
59         decreases (if b < 0 then - b else b end)
60         a_in_range: a ≥ -100 and a ≤ 100
61         b_in_range: b ≥ -100 and b ≤ 100
62     do
63         if b = 0 then
64             Result := a
65         elseif b > 0 then
66             Result := add_recursive (a, b - 1) + 1
67         else
68             Result := add_recursive (a, b + 1) - 1
69         end
70     ensure
71         result_correct: Result = a + b
72     end
73 feature -- Multiplication
74 multiply (a, b: INTEGER): INTEGER
75     -- Multiply two numbers by repeated addition.
76     -- Iterative version.
77     require
78         b_not_negative: b ≥ 0
79         a_in_range: a ≥ -10 and a ≤ 10
80         b_in_range: b ≥ -10 and b ≤ 10
81     local
82         i: INTEGER
83     do
84         if a = 0 or b = 0 then
85             Result := 0
86         else
87             from
88                 Result := a
89                 i := b
90             invariant
91                 Result = a * (b - i + 1)
92                 1 ≤ i and i ≤ b
93             until
94                 i = 1
95             loop
96                 Result := add (Result, a)
97                 i := i - 1
98             variant

```

```

99         i
100     end
101 end
102 ensure
103     result_correct: Result = a * b
104 end
105 multiply_recursive (a, b: INTEGER): INTEGER
106     -- Multiply two numbers by repeated addition.
107     -- Recursive version.
108     require
109         b_not_negative: b ≥ 0
110         a_in_range: a ≥ -10 and a ≤ 10
111         b_in_range: b ≥ -10 and b ≤ 10
112     do
113         -- if a = 0 or b = 0 then
114         -- Result := 0
115         -- else
116         if b = 1 then
117             Result := a
118         else
119             Result := add_recursive (a, multiply (a, b - 1))
120         end
121         -- end
122     ensure
123         result_correct: Result = a * b
124     end
125 feature -- Division
126 -- divide (n, m: INTEGER): TUPLE [quotient, remainder: INTEGER]
127 -- -- Integer division of 'n' divided by 'm'.
128 -- -- Iterative version.
129 -- require
130 --     n_not_negative: n ≥ 0
131 --     m_positive: m > 0
132 --     n_in_range: n ≤ 100
133 --     m_in_range: m ≤ 100
134 -- local
135 --     q, r: INTEGER
136 -- do
137 --     from
138 --         r := n
139 --         q := 0
140 --     invariant
141 --         0 ≤ r
142 --         n = m * q + r
143 --     until
144 --         r < m
145 --     loop
146 --         r := add (r, - m)
147 --         q := q + 1
148 --     variant

```

```

149 -- r
150 -- end
151 -- Result := [q, r]
152 -- ensure
153 -- n = m * Result.quotient + Result.remainder
154 -- end
155 divide_recursive (n, m: INTEGER): TUPLE [quotient, remainder: INTEGER]
156     -- Integer division of 'n' divided by 'm'.
157     -- Recursive version.
158     require
159         n_not_negative: n ≥ 0
160         m_positive: m > 0
161         n_in_range: n ≤ 100
162         m_in_range: m ≤ 100
163     local
164         q, r: INTEGER
165         res: TUPLE [quotient, remainder: INTEGER]
166     do
167         if n < m then
168             Result := [0, n]
169         else
170             res := divide_recursive (add_recursive (n, -m), m)
171             Result := [res.quotient + 1, res.remainder]
172         end
173     ensure
174         n = m * Result.quotient + Result.remainder
175     end
176 end

```

ARITHMETIC_1

- 5 fixes out of 22 candidate fixes
- fixing time: 1.1 minutes

```

1 multiply_recursive_ID_1 (a, b: INTEGER): INTEGER
2     -- Multiply two numbers by repeated addition.
3     -- Recursive version.
4     require
5         b_not_negative: b ≥ 0
6         a_in_range: a ≥ -10 and a ≤ 10
7         b_in_range: b ≥ -10 and b ≤ 10
8     do
9
10         -- if a = 0 or b = 0 then
11         -- Result := 0
12         -- else
13         if not((b) = (0)) then
14     if b = 1 then
15         Result := a

```

```

16         else
17             Result := add_recursive (a, multiply (a, b - 1))
18         end
19     end
20     --end
21     ensure
22         result_correct: Result = a * b
23     end
24
25 multiply_recursive_ID_9 (a, b: INTEGER): INTEGER
26     -- Multiply two numbers by repeated addition.
27     -- Recursive version.
28     require
29         b_not_negative: b ≥ 0
30         a_in_range: a ≥ -10 and a ≤ 10
31         b_in_range: b ≥ -10 and b ≤ 10
32     do
33
34         -- if a = 0 or b = 0 then
35         -- Result := 0
36         --else
37         if not((b) ≤ (0)) then
38     if b = 1 then
39         Result := a
40     else
41         Result := add_recursive (a, multiply (a, b - 1))
42     end
43 end
44     --end
45     ensure
46         result_correct: Result = a * b
47     end
48
49
50 multiply_recursive_ID_11 (a, b: INTEGER): INTEGER
51     -- Multiply two numbers by repeated addition.
52     -- Recursive version.
53     require
54         b_not_negative: b ≥ 0
55         a_in_range: a ≥ -10 and a ≤ 10
56         b_in_range: b ≥ -10 and b ≤ 10
57     do
58
59         -- if a = 0 or b = 0 then
60         -- Result := 0
61         --else
62         if not(b = 0) then
63     if b = 1 then
64         Result := a
65     else

```



```

66         Result := add_recursive (a, multiply (a, b - 1))
67     end
68 end
69 --end
70 ensure
71     result_correct: Result = a * b
72 end
73
74
75 multiply_recursive_ID_12 (a, b: INTEGER): INTEGER
76     -- Multiply two numbers by repeated addition.
77     -- Recursive version.
78     require
79         b_not_negative: b ≥ 0
80         a_in_range: a ≥ -10 and a ≤ 10
81         b_in_range: b ≥ -10 and b ≤ 10
82 not ((b) = (0))
83     do
84
85         -- if a = 0 or b = 0 then
86         -- Result := 0
87         --else
88         if b = 1 then
89             Result := a
90         else
91             Result := add_recursive (a, multiply (a, b - 1))
92         end
93         --end
94     ensure
95         result_correct: Result = a * b
96     end
97
98
99 multiply_recursive_ID_20 (a, b: INTEGER): INTEGER
100     -- Multiply two numbers by repeated addition.
101     -- Recursive version.
102     require
103         b_not_negative: b ≥ 0
104         a_in_range: a ≥ -10 and a ≤ 10
105         b_in_range: b ≥ -10 and b ≤ 10
106 not ((b) ≤ (0))
107     do
108
109         -- if a = 0 or b = 0 then
110         -- Result := 0
111         --else
112         if b = 1 then
113             Result := a
114         else
115             Result := add_recursive (a, multiply (a, b - 1))

```

```

116         end
117     --end
118     ensure
119         result_correct: Result = a * b
120     end
121
122
123 multiply_recursive_ID_22 (a, b: INTEGER): INTEGER
124     -- Multiply two numbers by repeated addition.
125     -- Recursive version.
126     require
127         b_not_negative: b ≥ 0
128         a_in_range: a ≥ -10 and a ≤ 10
129         b_in_range: b ≥ -10 and b ≤ 10
130     not (b = 0)
131     do
132
133         -- if a = 0 or b = 0 then
134         -- Result := 0
135         --else
136         if b = 1 then
137             Result := a
138         else
139             Result := add_recursive (a, multiply (a, b - 1))
140         end
141         --end
142     ensure
143         result_correct: Result = a * b
144     end

```

ARITHMETIC_2 No valid fixes

ARITHMETIC_3 Compilation error: ce extraction does not support the tuple type

ARITHMETIC_4

- 5 valid fixes out of 30 candidate fixes
- fixing time: 1.02 minutes

```

1 add_recursive_ID_0 (a, b: INTEGER): INTEGER
2     -- Add two numbers by repeated increment.
3     -- Recursive version.
4     require
5         decreases (if b < 0 then - b else b end)
6         a_in_range: a ≥ -100 and a ≤ 100
7         b_in_range: b ≥ -100 and b ≤ 100

```

```

8      do
9
10         -- if b = 0 then
11         -- Result := a
12         if b > 0 then
13             Result := add_recursive (a, b - 1) + 1
14         else
15             Result := add_recursive (a, b + 1) - 1
16         end
17     ensure
18         result_correct: Result = a + b
19     end
20
21 add_recursive_ID_16 (a, b: INTEGER): INTEGER
22     -- Add two numbers by repeated increment.
23     -- Recursive version.
24     require
25         decreases (if b < 0 then - b else b end)
26         a_in_range: a ≥ -100 and a ≤ 100
27         b_in_range: b ≥ -100 and b ≤ 100
28 not ((a) = (0))
29     do
30
31         -- if b = 0 then
32         -- Result := a
33         if b > 0 then
34             Result := add_recursive (a, b - 1) + 1
35         else
36             Result := add_recursive (a, b + 1) - 1
37         end
38     ensure
39         result_correct: Result = a + b
40     end
41
42 add_recursive_ID_18 (a, b: INTEGER): INTEGER
43     -- Add two numbers by repeated increment.
44     -- Recursive version.
45     require
46         decreases (if b < 0 then - b else b end)
47         a_in_range: a ≥ -100 and a ≤ 100
48         b_in_range: b ≥ -100 and b ≤ 100
49 not ((b) = (0))
50     do
51
52         -- if b = 0 then
53         -- Result := a
54         if b > 0 then
55             Result := add_recursive (a, b - 1) + 1
56         else
57             Result := add_recursive (a, b + 1) - 1

```

```

58     end
59     ensure
60         result_correct: Result = a + b
61     end
62
63 add_recursive_ID_20 (a, b: INTEGER): INTEGER
64     -- Add two numbers by repeated increment.
65     -- Recursive version.
66     require
67         decreases (if b < 0 then - b else b end)
68         a_in_range: a ≥ -100 and a ≤ 100
69         b_in_range: b ≥ -100 and b ≤ 100
70     not ((b) > (0))
71     do
72
73         -- if b = 0 then
74         -- Result := a
75         if b > 0 then
76             Result := add_recursive (a, b - 1) + 1
77         else
78             Result := add_recursive (a, b + 1) - 1
79         end
80     ensure
81         result_correct: Result = a + b
82     end
83
84 add_recursive_ID_22 (a, b: INTEGER): INTEGER
85     -- Add two numbers by repeated increment.
86     -- Recursive version.
87     require
88         decreases (if b < 0 then - b else b end)
89         a_in_range: a ≥ -100 and a ≤ 100
90         b_in_range: b ≥ -100 and b ≤ 100
91     not ((b) ≥ (0))
92     do
93
94         -- if b = 0 then
95         -- Result := a
96         if b > 0 then
97             Result := add_recursive (a, b - 1) + 1
98         else
99             Result := add_recursive (a, b + 1) - 1
100         end
101     ensure
102         result_correct: Result = a + b
103     end
104
105 add_recursive_ID_24 (a, b: INTEGER): INTEGER
106     -- Add two numbers by repeated increment.
107     -- Recursive version.

```

```

108     require
109     decreases (if b < 0 then - b else b end)
110     a_in_range: a ≥ -100 and a ≤ 100
111     b_in_range: b ≥ -100 and b ≤ 100
112 not ((b) < (0))
113 do
114
115     -- if b = 0 then
116     -- Result := a
117     if b > 0 then
118         Result := add_recursive (a, b - 1) + 1
119     else
120         Result := add_recursive (a, b + 1) - 1
121     end
122 ensure
123     result_correct: Result = a + b
124 end
125
126 add_recursive_ID_26 (a, b: INTEGER): INTEGER
127     -- Add two numbers by repeated increment.
128     -- Recursive version.
129     require
130     decreases (if b < 0 then - b else b end)
131     a_in_range: a ≥ -100 and a ≤ 100
132     b_in_range: b ≥ -100 and b ≤ 100
133 not ((b) ≤ (0))
134 do
135
136     -- if b = 0 then
137     -- Result := a
138     if b > 0 then
139         Result := add_recursive (a, b - 1) + 1
140     else
141         Result := add_recursive (a, b + 1) - 1
142     end
143 ensure
144     result_correct: Result = a + b
145 end
146
147 add_recursive_ID_28 (a, b: INTEGER): INTEGER
148     -- Add two numbers by repeated increment.
149     -- Recursive version.
150     require
151     decreases (if b < 0 then - b else b end)
152     a_in_range: a ≥ -100 and a ≤ 100
153     b_in_range: b ≥ -100 and b ≤ 100
154 not (b = 0)
155 do
156
157     -- if b = 0 then

```

```

158     -- Result := a
159     if b > 0 then
160         Result := add_recursive (a, b - 1) + 1
161     else
162         Result := add_recursive (a, b + 1) - 1
163     end
164     ensure
165         result_correct: Result = a + b
166     end
167
168 add_recursive_ID_30 (a, b: INTEGER): INTEGER
169     -- Add two numbers by repeated increment.
170     -- Recursive version.
171     require
172         decreases (if b < 0 then - b else b end)
173         a_in_range: a ≥ -100 and a ≤ 100
174         b_in_range: b ≥ -100 and b ≤ 100
175     not (a ≠ b)
176     do
177
178         -- if b = 0 then
179         -- Result := a
180         if b > 0 then
181             Result := add_recursive (a, b - 1) + 1
182         else
183             Result := add_recursive (a, b + 1) - 1
184         end
185     ensure
186         result_correct: Result = a + b
187     end

```

1.4 HEATER

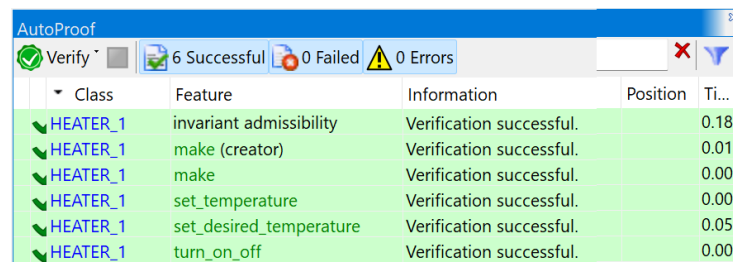
HEATER class, as shown below, implements a heater, which is automatically turned on/off based on the relation between the current temperature and the desired temperature. Fig. 44 displays the verification result of **HEATER**, which indicates that the implementation is correct with respect to its specification. In the experiment, 4 variants of faulty **HEATER** class are derived from this correct version, which will be discussed below.

```
1 class
2     HEATER
3 create
4     make
5 feature
6     make
7         -- By default, desired temperature is 20degree, deviation is 2 and
            heater is off
8     do
9         desired_temp := 20
10        is_on := False
11    ensure
12        default_condition: desired_temp = 20 and is_on = False
13    end
14 feature
15     temperature: INTEGER
16         -- Current temperature
17     desired_temp: INTEGER
18         -- Temperature defined by the user
19     is_on: BOOLEAN
20         -- Is heater turned on?
21     Deviation: INTEGER = 2
22         -- Deviation for turning on/off the heater
23 feature
24     set_temperature (a_value: INTEGER)
25         -- Set the 'temperature' to 'a_value'
26     do
27         temperature := a_value
28     end
29     set_desired_temperature (value: INTEGER)
30         -- Set the 'desired_temp' to 'value'
31     require
32         valid_value: value ≥ 10 and value ≤ 100
33     do
34         desired_temp := value
35     ensure
36         temperature_set: desired_temp = value
37     end
```

```

38     turn_on_off
39         -- Turn on or turn off the heater automatically based on the current
           temperature
40     require
41         desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
42     do
43         if is_on then
44             if temperature > desired_temp + deviation then
45                 is_on := False
46             end
47         else
48             if temperature < desired_temp - deviation then
49                 is_on := True
50             end
51         end
52     ensure
53         heater_is_turned_off: old (is_on and temperature > desired_temp +
           deviation) implies (not is_on)
54         heater_remains_on: old (is_on and temperature ≤ desired_temp +
           deviation) implies is_on
55         heater_is_turned_on: old (not is_on and temperature < desired_temp -
           deviation) implies is_on
56         heater_remains_off: old (not is_on and temperature ≥ desired_temp -
           deviation) implies (not is_on)
57     end
58 invariant
59     desired_temp_in_bound: desired_temp ≥ 10 and desired_temp ≤ 100
60 end

```



The screenshot shows the AutoProof window with a status bar indicating 6 Successful, 0 Failed, and 0 Errors. Below the status bar is a table with 6 columns: Class, Feature, Information, Position, and Ti... (likely Time). The table lists the verification results for the HEATER class, showing that all features were verified successfully.

Class	Feature	Information	Position	Ti...
HEATER_1	invariant admissibility	Verification successful.		0.18
HEATER_1	make (creator)	Verification successful.		0.01
HEATER_1	make	Verification successful.		0.00
HEATER_1	set_temperature	Verification successful.		0.00
HEATER_1	set_desired_temperature	Verification successful.		0.05
HEATER_1	turn_on_off	Verification successful.		0.00

Fig. 44. Proof result of **HEATER** in AutoProof

Variant 1 of **HEATER**

- Fault injection: at line 59, change the condition of the then branch from “temperature < desired_temp - deviation” into “temperature ≤ desired_temp - deviation”.
- Resulting failure: as shown in Fig. 45(a), the injected faults results in the violation of the postcondition heater_remains_off of the procedure turn_on_off.
- Cause of the failure: incorrect implementation of the routine body of turn_on_off.
- Proof time: 0.615 sec
- Comment: the test is useful as its execution shows a specific path along which the same contract violation in the proof would be raised.
- Possible fixes: 24 valid fixes out 437 candidate fixes.
- fixing time: 5.28 minutes

Class	Feature	Information	Position	Time
HEATER_1	invariant admissibility	Verification successful.		0.52
HEATER_1	make (creator)	Verification successful.		0.01
HEATER_1	make	Verification successful.		0.01
HEATER_1	set_temperature	Verification successful.		0.00
HEATER_1	set_desired_temperature	Verification successful.		0.04
HEATER_1	turn_on_off	Postcondition heater_remains_off may be violated.	72	0.03

(a)

```

1
2 turn_on_off_ID_367
3   -- Turn on or turn off the heater automatically based on the current
   temperature
4   require
5     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
6   do
7
8     if is_on then
9       if temperature > desired_temp + deviation then
10        is_on := False
11      end
12    else
13      if temperature ≤ desired_temp - deviation then
14        is_on := True
15      end
16    end
17  ensure
18    heater_is_turned_off: old (is_on and temperature > desired_temp +
   deviation) implies (not is_on)
19    heater_remains_on: old (is_on and temperature ≤ desired_temp +
   deviation) implies is_on
20    heater_is_turned_on: old (not is_on and temperature < desired_temp -
   deviation) implies is_on

```

```

21 heater_remains_off: not (is_on) implies ( old (not is_on and temperature ≥
    desired_temp - deviation) implies (not is_on))
22 end
23
24 turn_on_off_ID_368
25   -- Turn on or turn off the heater automatically based on the current
    temperature
26   require
27     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
28   not (not (is_on))
29   do
30
31     if is_on then
32       if temperature > desired_temp + deviation then
33         is_on := False
34       end
35     else
36       if temperature ≤ desired_temp - deviation then
37         is_on := True
38       end
39     end
40   ensure
41     heater_is_turned_off: old (is_on and temperature > desired_temp +
        deviation) implies (not is_on)
42     heater_remains_on: old (is_on and temperature ≤ desired_temp +
        deviation) implies is_on
43     heater_is_turned_on: old (not is_on and temperature < desired_temp -
        deviation) implies is_on
44     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
        deviation) implies (not is_on)
45   end
46
47 turn_on_off_ID_372
48   -- Turn on or turn off the heater automatically based on the current
    temperature
49   require
50     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
51   not ((temperature) > (0))
52   do
53
54     if is_on then
55       if temperature > desired_temp + deviation then
56         is_on := False
57       end
58     else
59       if temperature ≤ desired_temp - deviation then
60         is_on := True
61       end
62     end
63   ensure

```

```

64     heater_is_turned_off: old (is_on and temperature > desired_temp +
65         deviation) implies (not is_on)
66     heater_remains_on: old (is_on and temperature ≤ desired_temp +
67         deviation) implies is_on
68     heater_is_turned_on: old (not is_on and temperature < desired_temp -
69         deviation) implies is_on
70     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
71         deviation) implies (not is_on)
72 end
73
74 turn_on_off_ID_373
75     -- Turn on or turn off the heater automatically based on the current
76     temperature
77     require
78     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
79     do
80         if is_on then
81             if temperature > desired_temp + deviation then
82                 is_on := False
83             end
84         else
85             if temperature ≤ desired_temp - deviation then
86                 is_on := True
87             end
88         end
89     ensure
90     heater_is_turned_off: old (is_on and temperature > desired_temp +
91         deviation) implies (not is_on)
92     heater_remains_on: old (is_on and temperature ≤ desired_temp +
93         deviation) implies is_on
94     heater_is_turned_on: old (not is_on and temperature < desired_temp -
95         deviation) implies is_on
96     heater_remains_off: not ((temperature) > (0)) implies ( old (not is_on and
97         temperature ≥ desired_temp - deviation) implies (not is_on))
98     end
99
100 turn_on_off_ID_374
101     -- Turn on or turn off the heater automatically based on the current
102     temperature
103     require
104     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
105     not ((temperature) ≥ (0))
106     do
107         if is_on then
108             if temperature > desired_temp + deviation then
109                 is_on := False
110             end
111         else
112             if temperature ≤ desired_temp - deviation then
113                 is_on := True
114             end
115         end
116     ensure
117     heater_is_turned_off: old (is_on and temperature > desired_temp +
118         deviation) implies (not is_on)
119     heater_remains_on: old (is_on and temperature ≤ desired_temp +
120         deviation) implies is_on
121     heater_is_turned_on: old (not is_on and temperature < desired_temp -
122         deviation) implies is_on
123     heater_remains_off: not ((temperature) > (0)) implies ( old (not is_on and
124         temperature ≥ desired_temp - deviation) implies (not is_on))
125     end

```

```

104         if temperature ≤ desired_temp - deviation then
105             is_on := True
106         end
107     end
108 ensure
109     heater_is_turned_off: old (is_on and temperature > desired_temp +
110         deviation) implies (not is_on)
111     heater_remains_on: old (is_on and temperature ≤ desired_temp +
112         deviation) implies is_on
113     heater_is_turned_on: old (not is_on and temperature < desired_temp -
114         deviation) implies is_on
115     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
116         deviation) implies (not is_on)
117 end
118
119 turn_on_off_ID_375
120 -- Turn on or turn off the heater automatically based on the current
121 -- temperature
122 require
123     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
124 do
125     if is_on then
126         if temperature > desired_temp + deviation then
127             is_on := False
128         end
129     else
130         if temperature ≤ desired_temp - deviation then
131             is_on := True
132         end
133     end
134 ensure
135     heater_is_turned_off: old (is_on and temperature > desired_temp +
136         deviation) implies (not is_on)
137     heater_remains_on: old (is_on and temperature ≤ desired_temp +
138         deviation) implies is_on
139     heater_is_turned_on: old (not is_on and temperature < desired_temp -
140         deviation) implies is_on
141     heater_remains_off: not ((temperature) ≥ (0)) implies ( old (not is_on and
142         temperature ≥ desired_temp - deviation) implies (not is_on))
143 end
144
145 turn_on_off_ID_383
146 -- Turn on or turn off the heater automatically based on the current
147 -- temperature
148 require
149     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
150 do
151     if is_on then

```

```

144         if temperature > desired_temp + deviation then
145             is_on := False
146         end
147     else
148         if temperature ≤ desired_temp - deviation then
149             is_on := True
150         end
151     end
152 ensure
153     heater_is_turned_off: old (is_on and temperature > desired_temp +
154         deviation) implies (not is_on)
155     heater_remains_on: old (is_on and temperature ≤ desired_temp +
156         deviation) implies is_on
157     heater_is_turned_on: old (not is_on and temperature < desired_temp -
158         deviation) implies is_on
159 heater_remains_off: not ((desired_temp) > (0)) implies ( old (not is_on and
160     temperature ≥ desired_temp - deviation) implies (not is_on))
161 end
162
163 turn_on_off_ID_385
164     -- Turn on or turn off the heater automatically based on the current
165     temperature
166 require
167     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
168 do
169     if is_on then
170         if temperature > desired_temp + deviation then
171             is_on := False
172         end
173     else
174         if temperature ≤ desired_temp - deviation then
175             is_on := True
176         end
177     end
178 ensure
179     heater_is_turned_off: old (is_on and temperature > desired_temp +
180         deviation) implies (not is_on)
181     heater_remains_on: old (is_on and temperature ≤ desired_temp +
182         deviation) implies is_on
183     heater_is_turned_on: old (not is_on and temperature < desired_temp -
184         deviation) implies is_on
185     heater_remains_off: not ((desired_temp) ≥ (0)) implies ( old (not is_on and
186         temperature ≥ desired_temp - deviation) implies (not is_on))
187 end
188
189 turn_on_off_ID_393
190     -- Turn on or turn off the heater automatically based on the current
191     temperature
192 require

```

```

184     desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
185 do
186
187     if is_on then
188         if temperature > desired_temp + deviation then
189             is_on := False
190         end
191     else
192         if temperature  $\leq$  desired_temp - deviation then
193             is_on := True
194         end
195     end
196 ensure
197     heater_is_turned_off: old (is_on and temperature > desired_temp +
198         deviation) implies (not is_on)
199     heater_remains_on: old (is_on and temperature  $\leq$  desired_temp +
200         deviation) implies is_on
201     heater_is_turned_on: old (not is_on and temperature < desired_temp -
202         deviation) implies is_on
203     heater_remains_off: not ((deviation) > (0)) implies ( old (not is_on and
204         temperature  $\geq$  desired_temp - deviation) implies (not is_on))
205 end
206
207 turn_on_off_ID_395
208
209     -- Turn on or turn off the heater automatically based on the current
210     temperature
211
212 require
213     desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
214 do
215
216     if is_on then
217         if temperature > desired_temp + deviation then
218             is_on := False
219         end
220     else
221         if temperature  $\leq$  desired_temp - deviation then
222             is_on := True
223         end
224     end
225 ensure
226     heater_is_turned_off: old (is_on and temperature > desired_temp +
227         deviation) implies (not is_on)
228     heater_remains_on: old (is_on and temperature  $\leq$  desired_temp +
229         deviation) implies is_on
230     heater_is_turned_on: old (not is_on and temperature < desired_temp -
231         deviation) implies is_on
232     heater_remains_off: not ((deviation)  $\geq$  (0)) implies ( old (not is_on and
233         temperature  $\geq$  desired_temp - deviation) implies (not is_on))
234 end

```

```

225 turn_on_off_ID_406
226     -- Turn on or turn off the heater automatically based on the current
        temperature
227     require
228         desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
229 not ((temperature) < (desired_temp))
230     do
231
232         if is_on then
233             if temperature > desired_temp + deviation then
234                 is_on := False
235             end
236         else
237             if temperature  $\leq$  desired_temp - deviation then
238                 is_on := True
239             end
240         end
241     ensure
242         heater_is_turned_off: old (is_on and temperature > desired_temp +
            deviation) implies (not is_on)
243         heater_remains_on: old (is_on and temperature  $\leq$  desired_temp +
            deviation) implies is_on
244         heater_is_turned_on: old (not is_on and temperature < desired_temp -
            deviation) implies is_on
245         heater_remains_off: old (not is_on and temperature  $\geq$  desired_temp -
            deviation) implies (not is_on)
246     end
247
248 turn_on_off_ID_407
249     -- Turn on or turn off the heater automatically based on the current
        temperature
250     require
251         desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
252     do
253
254         if is_on then
255             if temperature > desired_temp + deviation then
256                 is_on := False
257             end
258         else
259             if temperature  $\leq$  desired_temp - deviation then
260                 is_on := True
261             end
262         end
263     ensure
264         heater_is_turned_off: old (is_on and temperature > desired_temp +
            deviation) implies (not is_on)
265         heater_remains_on: old (is_on and temperature  $\leq$  desired_temp +
            deviation) implies is_on

```

```

266     heater_is_turned_on: old (not is_on and temperature < desired_temp -
        deviation) implies is_on
267 heater_remains_off: not ((temperature) < (desired_temp)) implies ( old (not
        is_on and temperature ≥ desired_temp - deviation) implies (not is_on))
268     end
269
270 turn_on_off_ID_408
271     -- Turn on or turn off the heater automatically based on the current
        temperature
272     require
273         desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
274 not ((temperature) ≤ (desired_temp))
275     do
276
277         if is_on then
278             if temperature > desired_temp + deviation then
279                 is_on := False
280             end
281         else
282             if temperature ≤ desired_temp - deviation then
283                 is_on := True
284             end
285         end
286     ensure
287         heater_is_turned_off: old (is_on and temperature > desired_temp +
            deviation) implies (not is_on)
288         heater_remains_on: old (is_on and temperature ≤ desired_temp +
            deviation) implies is_on
289         heater_is_turned_on: old (not is_on and temperature < desired_temp -
            deviation) implies is_on
290         heater_remains_off: old (not is_on and temperature ≥ desired_temp -
            deviation) implies (not is_on)
291     end
292
293 turn_on_off_ID_409
294     -- Turn on or turn off the heater automatically based on the current
        temperature
295     require
296         desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
297     do
298
299         if is_on then
300             if temperature > desired_temp + deviation then
301                 is_on := False
302             end
303         else
304             if temperature ≤ desired_temp - deviation then
305                 is_on := True
306             end
307         end

```



```

308     ensure
309         heater_is_turned_off: old (is_on and temperature > desired_temp +
            deviation) implies (not is_on)
310         heater_remains_on: old (is_on and temperature ≤ desired_temp +
            deviation) implies is_on
311         heater_is_turned_on: old (not is_on and temperature < desired_temp -
            deviation) implies is_on
312 heater_remains_off: not ((temperature) < (desired_temp)) implies ( old (not
            is_on and temperature ≥ desired_temp - deviation) implies (not is_on))
313     end
314
315 turn_on_off_ID_412
316     -- Turn on or turn off the heater automatically based on the current
            temperature
317     require
318         desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
319 not ((temperature) > (deviation))
320     do
321
322         if is_on then
323             if temperature > desired_temp + deviation then
324                 is_on := False
325             end
326         else
327             if temperature ≤ desired_temp - deviation then
328                 is_on := True
329             end
330         end
331     ensure
332         heater_is_turned_off: old (is_on and temperature > desired_temp +
            deviation) implies (not is_on)
333         heater_remains_on: old (is_on and temperature ≤ desired_temp +
            deviation) implies is_on
334         heater_is_turned_on: old (not is_on and temperature < desired_temp -
            deviation) implies is_on
335         heater_remains_off: old (not is_on and temperature ≥ desired_temp -
            deviation) implies (not is_on)
336     end
337
338 turn_on_off_ID_413
339     -- Turn on or turn off the heater automatically based on the current
            temperature
340     require
341         desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
342     do
343
344         if is_on then
345             if temperature > desired_temp + deviation then
346                 is_on := False
347             end

```

```

348     else
349         if temperature  $\leq$  desired_temp - deviation then
350             is_on := True
351         end
352     end
353 ensure
354     heater_is_turned_off: old (is_on and temperature > desired_temp +
355         deviation) implies (not is_on)
356     heater_remains_on: old (is_on and temperature  $\leq$  desired_temp +
357         deviation) implies is_on
358     heater_is_turned_on: old (not is_on and temperature < desired_temp -
359         deviation) implies is_on
360 heater_remains_off: not ((temperature) > (deviation)) implies ( old (not is_on
361     and temperature  $\geq$  desired_temp - deviation) implies (not is_on))
362 end
363
364 turn_on_off_ID_414
365     -- Turn on or turn off the heater automatically based on the current
366     temperature
367 require
368     desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
369 not ((temperature)  $\geq$  (deviation))
370 do
371     if is_on then
372         if temperature > desired_temp + deviation then
373             is_on := False
374         end
375     else
376         if temperature  $\leq$  desired_temp - deviation then
377             is_on := True
378         end
379     end
380 ensure
381     heater_is_turned_off: old (is_on and temperature > desired_temp +
382         deviation) implies (not is_on)
383     heater_remains_on: old (is_on and temperature  $\leq$  desired_temp +
384         deviation) implies is_on
385     heater_is_turned_on: old (not is_on and temperature < desired_temp -
386         deviation) implies is_on
387     heater_remains_off: old (not is_on and temperature  $\geq$  desired_temp -
388         deviation) implies (not is_on)
389 end
390
391 turn_on_off_ID_415
392     -- Turn on or turn off the heater automatically based on the current
393     temperature
394 require
395     desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
396 do

```

```

388
389     if is_on then
390         if temperature > desired_temp + deviation then
391             is_on := False
392         end
393     else
394         if temperature ≤ desired_temp - deviation then
395             is_on := True
396         end
397     end
398     ensure
399         heater_is_turned_off: old (is_on and temperature > desired_temp +
400             deviation) implies (not is_on)
401         heater_remains_on: old (is_on and temperature ≤ desired_temp +
402             deviation) implies is_on
403         heater_is_turned_on: old (not is_on and temperature < desired_temp -
404             deviation) implies is_on
405     heater_remains_off: not ((temperature) ≥ (deviation)) implies ( old (not is_on
406         and temperature ≥ desired_temp - deviation) implies (not is_on))
407     end
408
409 turn_on_off_ID_423
410     -- Turn on or turn off the heater automatically based on the current
411     temperature
412     require
413         desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
414     do
415         if is_on then
416             if temperature > desired_temp + deviation then
417                 is_on := False
418             end
419         else
420             if temperature ≤ desired_temp - deviation then
421                 is_on := True
422             end
423         end
424     ensure
425         heater_is_turned_off: old (is_on and temperature > desired_temp +
426             deviation) implies (not is_on)
427         heater_remains_on: old (is_on and temperature ≤ desired_temp +
428             deviation) implies is_on
429         heater_is_turned_on: old (not is_on and temperature < desired_temp -
430             deviation) implies is_on
431     heater_remains_off: not ((desired_temp) > (deviation)) implies ( old (not is_on
432         and temperature ≥ desired_temp - deviation) implies (not is_on))
433     end

```

```

429 turn_on_off_ID_425
430     -- Turn on or turn off the heater automatically based on the current
         temperature
431     require
432         desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
433     do
434
435         if is_on then
436             if temperature > desired_temp + deviation then
437                 is_on := False
438             end
439         else
440             if temperature  $\leq$  desired_temp - deviation then
441                 is_on := True
442             end
443         end
444     ensure
445         heater_is_turned_off: old (is_on and temperature > desired_temp +
            deviation) implies (not is_on)
446         heater_remains_on: old (is_on and temperature  $\leq$  desired_temp +
            deviation) implies is_on
447         heater_is_turned_on: old (not is_on and temperature < desired_temp -
            deviation) implies is_on
448         heater_remains_off: not ((desired_temp)  $\geq$  (deviation)) implies ( old (not is_on
            and temperature  $\geq$  desired_temp - deviation) implies (not is_on))
449     end
450
451
452 turn_on_off_ID_431
453     -- Turn on or turn off the heater automatically based on the current
         temperature
454     require
455         desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
456     do
457
458         if is_on then
459             if temperature > desired_temp + deviation then
460                 is_on := False
461             end
462         else
463             if temperature  $\leq$  desired_temp - deviation then
464                 is_on := True
465             end
466         end
467     ensure
468         heater_is_turned_off: old (is_on and temperature > desired_temp +
            deviation) implies (not is_on)
469         heater_remains_on: old (is_on and temperature  $\leq$  desired_temp +
            deviation) implies is_on

```

```

470     heater_is_turned_on: old (not is_on and temperature < desired_temp -
        deviation) implies is_on
471 heater_remains_off: not (deviation = 2) implies ( old (not is_on and
        temperature ≥ desired_temp - deviation) implies (not is_on))
472 end
473
474
475 turn_on_off_ID_432
476     -- Turn on or turn off the heater automatically based on the current
        temperature
477     require
478         desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
479 not (temperature - desired_temp + 2 = 0)
480     do
481
482         if is_on then
483             if temperature > desired_temp + deviation then
484                 is_on := False
485             end
486         else
487             if temperature ≤ desired_temp - deviation then
488                 is_on := True
489             end
490         end
491     ensure
492         heater_is_turned_off: old (is_on and temperature > desired_temp +
            deviation) implies (not is_on)
493         heater_remains_on: old (is_on and temperature ≤ desired_temp +
            deviation) implies is_on
494         heater_is_turned_on: old (not is_on and temperature < desired_temp -
            deviation) implies is_on
495         heater_remains_off: old (not is_on and temperature ≥ desired_temp -
            deviation) implies (not is_on)
496     end
497
498
499 turn_on_off_ID_433
500     -- Turn on or turn off the heater automatically based on the current
        temperature
501     require
502         desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
503     do
504
505         if is_on then
506             if temperature > desired_temp + deviation then
507                 is_on := False
508             end
509         else
510             if temperature ≤ desired_temp - deviation then
511                 is_on := True

```

```

512         end
513     end
514     ensure
515         heater_is_turned_off: old (is_on and temperature > desired_temp +
                    deviation) implies (not is_on)
516         heater_remains_on: old (is_on and temperature ≤ desired_temp +
                    deviation) implies is_on
517         heater_is_turned_on: old (not is_on and temperature < desired_temp -
                    deviation) implies is_on
518         heater_remains_off: not (temperature - desired_temp + 2 = 0) implies ( old (not
                    is_on and temperature ≥ desired_temp - deviation) implies (not is_on))
519     end
520
521
522 turn_on_off_ID_434
523     -- Turn on or turn off the heater automatically based on the current
        temperature
524     require
525         desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
526     not (temperature > deviation)
527     do
528
529         if is_on then
530             if temperature > desired_temp + deviation then
531                 is_on := False
532             end
533         else
534             if temperature ≤ desired_temp - deviation then
535                 is_on := True
536             end
537         end
538     ensure
539         heater_is_turned_off: old (is_on and temperature > desired_temp +
                    deviation) implies (not is_on)
540         heater_remains_on: old (is_on and temperature ≤ desired_temp +
                    deviation) implies is_on
541         heater_is_turned_on: old (not is_on and temperature < desired_temp -
                    deviation) implies is_on
542         heater_remains_off: old (not is_on and temperature ≥ desired_temp -
                    deviation) implies (not is_on)
543     end
544
545
546 turn_on_off_ID_435
547     -- Turn on or turn off the heater automatically based on the current
        temperature
548     require
549         desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
550     do
551

```

```

552     if is_on then
553         if temperature > desired_temp + deviation then
554             is_on := False
555         end
556     else
557         if temperature ≤ desired_temp - deviation then
558             is_on := True
559         end
560     end
561 ensure
562     heater_is_turned_off: old (is_on and temperature > desired_temp +
563         deviation) implies (not is_on)
564     heater_remains_on: old (is_on and temperature ≤ desired_temp +
565         deviation) implies is_on
566     heater_is_turned_on: old (not is_on and temperature < desired_temp -
567         deviation) implies is_on
568 heater_remains_off: not (temperature > deviation) implies ( old (not is_on and
569     temperature ≥ desired_temp - deviation) implies (not is_on))
570 end
571
572 turn_on_off_ID_437
573     -- Turn on or turn off the heater automatically based on the current
574     temperature
575 require
576     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
577 do
578     if is_on then
579         if temperature > desired_temp + deviation then
580             is_on := False
581         end
582     else
583         if temperature ≤ desired_temp - deviation then
584             is_on := True
585         end
586     end
587 ensure
588     heater_is_turned_off: old (is_on and temperature > desired_temp +
589         deviation) implies (not is_on)
590     heater_remains_on: old (is_on and temperature ≤ desired_temp +
591         deviation) implies is_on
592     heater_is_turned_on: old (not is_on and temperature < desired_temp -
593         deviation) implies is_on
594 heater_remains_off: not (desired_temp > deviation) implies ( old (not is_on and
595     temperature ≥ desired_temp - deviation) implies (not is_on))
596 end

```

Variant 2 of HEATER

- Fault injection: at line 55, in the body of `turn_on_off`, change the condition of the then branch from “`temperature > desired_temp+deviation`” into “`temperature ≥ desired_temp+deviation`”.
- Resulting failure: as shown in Fig. 45(b), the injected fault results in the violation of postcondition `heater_remains_on` of `turn_on_off` procedure.
- Cause of the failure: incorrect implementation of the routine body of `turn_on_off`.
- Proof time: 0.642 sec
- Comment: similar to Variant 1 of `HEATER`, the test is useful as its execution shows a specific path along which the same contract violation in the proof would be raised.
- Possible fixes: 25 valid fixes out of 437
- fixing time: 4.05 minutes

Class	Feature	Information	Position	Time
HEATER_2	invariant admissibility	Verification successful.		0.55
HEATER_2	make (creator)	Verification successful.		0.00
HEATER_2	make	Verification successful.		0.01
HEATER_2	set_temperature	Verification successful.		0.00
HEATER_2	set_desired_temperature	Verification successful.		0.05
HEATER_2	turn_on_off	Postcondition heater_remains_on may be violated.	70	0.03

(b)

```

1  turn_on_off_ID_366
2      -- Turn on or turn off the heater automatically based on the current
      temperature
3      require
4          desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
5  not (is_on)
6      do
7
8          if is_on then
9              if temperature ≥ desired_temp + deviation then
10                 is_on := False
11             end
12         else
13             if temperature < desired_temp - deviation then
14                 is_on := True
15             end
16         end
17     ensure
18         heater_is_turned_off: old (is_on and temperature > desired_temp +
            deviation) implies (not is_on)

```



```

19     heater_remains_on: old (is_on and temperature ≤ desired_temp +
20         deviation) implies is_on
21     heater_is_turned_on: old (not is_on and temperature < desired_temp -
22         deviation) implies is_on
23     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
24         deviation) implies (not is_on)
25 end
26
27 turn_on_off_ID_369
28     -- Turn on or turn off the heater automatically based on the current
29     temperature
30     require
31     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
32 do
33     if is_on then
34         if temperature ≥ desired_temp + deviation then
35             is_on := False
36         end
37     else
38         if temperature < desired_temp - deviation then
39             is_on := True
40         end
41     end
42     ensure
43     heater_is_turned_off: old (is_on and temperature > desired_temp +
44         deviation) implies (not is_on)
45     heater_remains_on: not (not (is_on)) implies ( old (is_on and temperature ≤
46         desired_temp + deviation) implies is_on)
47     heater_is_turned_on: old (not is_on and temperature < desired_temp -
48         deviation) implies is_on
49     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
50         deviation) implies (not is_on)
51 end
52
53 turn_on_off_ID_372
54     -- Turn on or turn off the heater automatically based on the current
55     temperature
56     require
57     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
58     not ((temperature) > (0))
59 do
60     if is_on then
61         if temperature ≥ desired_temp + deviation then
62             is_on := False
63         end
64     else
65         if temperature < desired_temp - deviation then

```

```

60         is_on := True
61     end
62 end
63 ensure
64     heater_is_turned_off: old (is_on and temperature > desired_temp +
65         deviation) implies (not is_on)
66     heater_remains_on: old (is_on and temperature ≤ desired_temp +
67         deviation) implies is_on
68     heater_is_turned_on: old (not is_on and temperature < desired_temp -
69         deviation) implies is_on
70     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
71         deviation) implies (not is_on)
72 end
73
74 turn_on_off_ID_373
75     -- Turn on or turn off the heater automatically based on the current
76     temperature
77
78     require
79         desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
80     do
81         if is_on then
82             if temperature ≥ desired_temp + deviation then
83                 is_on := False
84             end
85         else
86             if temperature < desired_temp - deviation then
87                 is_on := True
88             end
89         end
90     ensure
91         heater_is_turned_off: old (is_on and temperature > desired_temp +
92             deviation) implies (not is_on)
93     heater_remains_on: not ((temperature) > (0)) implies ( old (is_on and
94         temperature ≤ desired_temp + deviation) implies is_on)
95     heater_is_turned_on: old (not is_on and temperature < desired_temp -
96         deviation) implies is_on
97     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
98         deviation) implies (not is_on)
99 end
100
101 turn_on_off_ID_374
102     -- Turn on or turn off the heater automatically based on the current
103     temperature
104
105     require
106         desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
107     not ((temperature) ≥ (0))
108     do

```

```

100     if is_on then
101         if temperature  $\geq$  desired_temp + deviation then
102             is_on := False
103         end
104     else
105         if temperature < desired_temp - deviation then
106             is_on := True
107         end
108     end
109 ensure
110     heater_is_turned_off: old (is_on and temperature > desired_temp +
111         deviation) implies (not is_on)
112     heater_remains_on: old (is_on and temperature  $\leq$  desired_temp +
113         deviation) implies is_on
114     heater_is_turned_on: old (not is_on and temperature < desired_temp -
115         deviation) implies is_on
116     heater_remains_off: old (not is_on and temperature  $\geq$  desired_temp -
117         deviation) implies (not is_on)
118 end
119
120 turn_on_off_ID_375
121     -- Turn on or turn off the heater automatically based on the current
122     temperature
123 require
124     desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
125 do
126     if is_on then
127         if temperature  $\geq$  desired_temp + deviation then
128             is_on := False
129         end
130     else
131         if temperature < desired_temp - deviation then
132             is_on := True
133         end
134     end
135 ensure
136     heater_is_turned_off: old (is_on and temperature > desired_temp +
137         deviation) implies (not is_on)
138     heater_remains_on: not ((temperature)  $\geq$  (0)) implies ( old (is_on and
139         temperature  $\leq$  desired_temp + deviation) implies is_on)
140     heater_is_turned_on: old (not is_on and temperature < desired_temp -
141         deviation) implies is_on
142     heater_remains_off: old (not is_on and temperature  $\geq$  desired_temp -
143         deviation) implies (not is_on)
144 end
145
146 turn_on_off_ID_383

```

```

141      -- Turn on or turn off the heater automatically based on the current
      temperature
142  require
143    desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
144  do
145
146    if is_on then
147      if temperature  $\geq$  desired_temp + deviation then
148        is_on := False
149      end
150    else
151      if temperature < desired_temp - deviation then
152        is_on := True
153      end
154    end
155  ensure
156    heater_is_turned_off: old (is_on and temperature > desired_temp +
      deviation) implies (not is_on)
157  heater_remains_on: not ((desired_temp > (0)) implies ( old (is_on and
      temperature  $\leq$  desired_temp + deviation) implies is_on)
158    heater_is_turned_on: old (not is_on and temperature < desired_temp -
      deviation) implies is_on
159    heater_remains_off: old (not is_on and temperature  $\geq$  desired_temp -
      deviation) implies (not is_on)
160  end
161
162
163  turn_on_off_ID_385
164    -- Turn on or turn off the heater automatically based on the current
      temperature
165  require
166    desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
167  do
168
169    if is_on then
170      if temperature  $\geq$  desired_temp + deviation then
171        is_on := False
172      end
173    else
174      if temperature < desired_temp - deviation then
175        is_on := True
176      end
177    end
178  ensure
179    heater_is_turned_off: old (is_on and temperature > desired_temp +
      deviation) implies (not is_on)
180  heater_remains_on: not ((desired_temp  $\geq$  (0)) implies ( old (is_on and
      temperature  $\leq$  desired_temp + deviation) implies is_on)
181    heater_is_turned_on: old (not is_on and temperature < desired_temp -
      deviation) implies is_on

```

```

182     heater_remains_off: old (not is_on and temperature  $\geq$  desired_temp -
183         deviation) implies (not is_on)
184 end
185
186 turn_on_off_ID_393
187     -- Turn on or turn off the heater automatically based on the current
188         temperature
189 require
190     desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
191 do
192     if is_on then
193         if temperature  $\geq$  desired_temp + deviation then
194             is_on := False
195         end
196     else
197         if temperature < desired_temp - deviation then
198             is_on := True
199         end
200     end
201 ensure
202     heater_is_turned_off: old (is_on and temperature > desired_temp +
203         deviation) implies (not is_on)
204 heater_remains_on: not ((deviation) > (0)) implies ( old (is_on and temperature
205      $\leq$  desired_temp + deviation) implies is_on)
206 heater_is_turned_on: old (not is_on and temperature < desired_temp -
207     deviation) implies is_on
208 heater_remains_off: old (not is_on and temperature  $\geq$  desired_temp -
209     deviation) implies (not is_on)
210 end
211
212 turn_on_off_ID_395
213     -- Turn on or turn off the heater automatically based on the current
214         temperature
215 require
216     desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
217 do
218     if is_on then
219         if temperature  $\geq$  desired_temp + deviation then
220             is_on := False
221         end
222     else
223         if temperature < desired_temp - deviation then
224             is_on := True
225         end
226     end
227 ensure

```

```

225     heater_is_turned_off: old (is_on and temperature > desired_temp +
        deviation) implies (not is_on)
226 heater_remains_on: not ((deviation) ≥ 0) implies ( old (is_on and temperature
        ≤ desired_temp + deviation) implies is_on)
227     heater_is_turned_on: old (not is_on and temperature < desired_temp -
        deviation) implies is_on
228     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
        deviation) implies (not is_on)
229 end
230
231
232 turn_on_off_ID_402
233     -- Turn on or turn off the heater automatically based on the current
        temperature
234     require
235         desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
236 not ((temperature) > (desired_temp))
237     do
238
239         if is_on then
240             if temperature ≥ desired_temp + deviation then
241                 is_on := False
242             end
243         else
244             if temperature < desired_temp - deviation then
245                 is_on := True
246             end
247         end
248     ensure
249         heater_is_turned_off: old (is_on and temperature > desired_temp +
            deviation) implies (not is_on)
250         heater_remains_on: old (is_on and temperature ≤ desired_temp +
            deviation) implies is_on
251         heater_is_turned_on: old (not is_on and temperature < desired_temp -
            deviation) implies is_on
252         heater_remains_off: old (not is_on and temperature ≥ desired_temp -
            deviation) implies (not is_on)
253     end
254
255
256 turn_on_off_ID_403
257     -- Turn on or turn off the heater automatically based on the current
        temperature
258     require
259         desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
260     do
261
262         if is_on then
263             if temperature ≥ desired_temp + deviation then
264                 is_on := False

```

```

265         end
266     else
267         if temperature < desired_temp - deviation then
268             is_on := True
269         end
270     end
271 ensure
272     heater_is_turned_off: old (is_on and temperature > desired_temp +
273         deviation) implies (not is_on)
274 heater_remains_on: not ((temperature) > (desired_temp)) implies ( old (is_on
275     and temperature ≤ desired_temp + deviation) implies is_on)
276     heater_is_turned_on: old (not is_on and temperature < desired_temp -
277         deviation) implies is_on
278     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
279         deviation) implies (not is_on)
280 end
281
282 turn_on_off_ID_404
283     -- Turn on or turn off the heater automatically based on the current
284     temperature
285     require
286         desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
287     not ((temperature) ≥ (desired_temp))
288     do
289         if is_on then
290             if temperature ≥ desired_temp + deviation then
291                 is_on := False
292             end
293         else
294             if temperature < desired_temp - deviation then
295                 is_on := True
296             end
297         end
298     ensure
299         heater_is_turned_off: old (is_on and temperature > desired_temp +
300             deviation) implies (not is_on)
301         heater_remains_on: old (is_on and temperature ≤ desired_temp +
302             deviation) implies is_on
303         heater_is_turned_on: old (not is_on and temperature < desired_temp -
304             deviation) implies is_on
305         heater_remains_off: old (not is_on and temperature ≥ desired_temp -
306             deviation) implies (not is_on)
307     end
308
309 turn_on_off_ID_405
310     -- Turn on or turn off the heater automatically based on the current
311     temperature

```

```

305     require
306         desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
307     do
308
309         if is_on then
310             if temperature  $\geq$  desired_temp + deviation then
311                 is_on := False
312             end
313         else
314             if temperature < desired_temp - deviation then
315                 is_on := True
316             end
317         end
318     ensure
319         heater_is_turned_off: old (is_on and temperature > desired_temp +
320             deviation) implies (not is_on)
321         heater_remains_on: not ((temperature)  $\geq$  (desired_temp)) implies ( old (is_on and
322             temperature  $\leq$  desired_temp + deviation) implies is_on)
323         heater_is_turned_on: old (not is_on and temperature < desired_temp -
324             deviation) implies is_on
325         heater_remains_off: old (not is_on and temperature  $\geq$  desired_temp -
326             deviation) implies (not is_on)
327     end
328
329 turn_on_off_ID_412
330     -- Turn on or turn off the heater automatically based on the current
331     temperature
332
333     require
334         desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
335     not ((temperature) > (deviation))
336     do
337
338         if is_on then
339             if temperature  $\geq$  desired_temp + deviation then
340                 is_on := False
341             end
342         else
343             if temperature < desired_temp - deviation then
344                 is_on := True
345             end
346         end
347     ensure
348         heater_is_turned_off: old (is_on and temperature > desired_temp +
349             deviation) implies (not is_on)
350         heater_remains_on: old (is_on and temperature  $\leq$  desired_temp +
351             deviation) implies is_on
352         heater_is_turned_on: old (not is_on and temperature < desired_temp -
353             deviation) implies is_on

```



```

346     heater_remains_off: old (not is_on and temperature  $\geq$  desired_temp -
347         deviation) implies (not is_on)
348 end
349
350 turn_on_off_ID_413
351     -- Turn on or turn off the heater automatically based on the current
352         temperature
353 require
354     desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
355 do
356     if is_on then
357         if temperature  $\geq$  desired_temp + deviation then
358             is_on := False
359         end
360     else
361         if temperature < desired_temp - deviation then
362             is_on := True
363         end
364     end
365 ensure
366     heater_is_turned_off: old (is_on and temperature > desired_temp +
367         deviation) implies (not is_on)
368     heater_remains_on: not ((temperature) > (deviation)) implies ( old (is_on and
369         temperature  $\leq$  desired_temp + deviation) implies is_on)
370     heater_is_turned_on: old (not is_on and temperature < desired_temp -
371         deviation) implies is_on
372     heater_remains_off: old (not is_on and temperature  $\geq$  desired_temp -
373         deviation) implies (not is_on)
374 end
375
376 turn_on_off_ID_414
377     -- Turn on or turn off the heater automatically based on the current
378         temperature
379 require
380     desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
381 not ((temperature)  $\geq$  (deviation))
382 do
383     if is_on then
384         if temperature  $\geq$  desired_temp + deviation then
385             is_on := False
386         end
387     else
388         if temperature < desired_temp - deviation then
389             is_on := True
390         end
391     end
392 end

```

```

389     ensure
390         heater_is_turned_off: old (is_on and temperature > desired_temp +
391             deviation) implies (not is_on)
392         heater_remains_on: old (is_on and temperature ≤ desired_temp +
393             deviation) implies is_on
394         heater_is_turned_on: old (not is_on and temperature < desired_temp -
395             deviation) implies is_on
396         heater_remains_off: old (not is_on and temperature ≥ desired_temp -
397             deviation) implies (not is_on)
398     end
399
400 turn_on_off_ID_415
401     -- Turn on or turn off the heater automatically based on the current
402     temperature
403     require
404         desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
405     do
406         if is_on then
407             if temperature ≥ desired_temp + deviation then
408                 is_on := False
409             end
410         else
411             if temperature < desired_temp - deviation then
412                 is_on := True
413             end
414         end
415     ensure
416         heater_is_turned_off: old (is_on and temperature > desired_temp +
417             deviation) implies (not is_on)
418         heater_remains_on: not ((temperature) ≥ (deviation)) implies ( old (is_on and
419             temperature ≤ desired_temp + deviation) implies is_on)
420         heater_is_turned_on: old (not is_on and temperature < desired_temp -
421             deviation) implies is_on
422         heater_remains_off: old (not is_on and temperature ≥ desired_temp -
423             deviation) implies (not is_on)
424     end
425
426 turn_on_off_ID_423
427     -- Turn on or turn off the heater automatically based on the current
428     temperature
429     require
430         desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
431     do
432         if is_on then
433             if temperature ≥ desired_temp + deviation then
434                 is_on := False

```

```

429         end
430     else
431         if temperature < desired_temp - deviation then
432             is_on := True
433         end
434     end
435 ensure
436     heater_is_turned_off: old (is_on and temperature > desired_temp +
437         deviation) implies (not is_on)
437 heater_remains_on: not ((desired_temp > (deviation)) implies ( old (is_on and
438     temperature ≤ desired_temp + deviation) implies is_on)
438     heater_is_turned_on: old (not is_on and temperature < desired_temp -
439         deviation) implies is_on
439     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
440         deviation) implies (not is_on)
440 end
441
442
443 turn_on_off_ID_425
444     -- Turn on or turn off the heater automatically based on the current
445     temperature
445 require
446     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
447 do
448
449     if is_on then
450         if temperature ≥ desired_temp + deviation then
451             is_on := False
452         end
453     else
454         if temperature < desired_temp - deviation then
455             is_on := True
456         end
457     end
458 ensure
459     heater_is_turned_off: old (is_on and temperature > desired_temp +
460         deviation) implies (not is_on)
460 heater_remains_on: not ((desired_temp ≥ (deviation)) implies ( old (is_on and
461     temperature ≤ desired_temp + deviation) implies is_on)
461     heater_is_turned_on: old (not is_on and temperature < desired_temp -
462         deviation) implies is_on
462     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
463         deviation) implies (not is_on)
463 end
464
465
466 turn_on_off_ID_431
467     -- Turn on or turn off the heater automatically based on the current
468     temperature
468 require

```

```

469     desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
470 do
471     if is_on then
472         if temperature  $\geq$  desired_temp + deviation then
473             is_on := False
474         end
475     else
476         if temperature < desired_temp - deviation then
477             is_on := True
478         end
479     end
480 end
481 ensure
482     heater_is_turned_off: old (is_on and temperature > desired_temp +
483         deviation) implies (not is_on)
484     heater_remains_on: not (deviation = 2) implies ( old (is_on and temperature  $\leq$ 
485         desired_temp + deviation) implies is_on)
486     heater_is_turned_on: old (not is_on and temperature < desired_temp -
487         deviation) implies is_on
488     heater_remains_off: old (not is_on and temperature  $\geq$  desired_temp -
489         deviation) implies (not is_on)
490 end
491
492 turn_on_off_ID_432
493 -- Turn on or turn off the heater automatically based on the current
494 -- temperature
495 require
496     desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
497 not (temperature - desired_temp - 2 = 0)
498 do
499     if is_on then
500         if temperature  $\geq$  desired_temp + deviation then
501             is_on := False
502         end
503     else
504         if temperature < desired_temp - deviation then
505             is_on := True
506         end
507     end
508 end
509 ensure
510     heater_is_turned_off: old (is_on and temperature > desired_temp +
511         deviation) implies (not is_on)
512     heater_remains_on: old (is_on and temperature  $\leq$  desired_temp +
513         deviation) implies is_on
514     heater_is_turned_on: old (not is_on and temperature < desired_temp -
515         deviation) implies is_on
516     heater_remains_off: old (not is_on and temperature  $\geq$  desired_temp -
517         deviation) implies (not is_on)

```

```

510     end
511
512
513 turn_on_off_ID_433
514     -- Turn on or turn off the heater automatically based on the current
        temperature
515     require
516         desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
517     do
518
519         if is_on then
520             if temperature  $\geq$  desired_temp + deviation then
521                 is_on := False
522             end
523         else
524             if temperature < desired_temp - deviation then
525                 is_on := True
526             end
527         end
528     ensure
529         heater_is_turned_off: old (is_on and temperature > desired_temp +
            deviation) implies (not is_on)
530         heater_remains_on: not (temperature - desired_temp - 2 = 0) implies ( old (
            is_on and temperature  $\leq$  desired_temp + deviation) implies is_on)
531         heater_is_turned_on: old (not is_on and temperature < desired_temp -
            deviation) implies is_on
532         heater_remains_off: old (not is_on and temperature  $\geq$  desired_temp -
            deviation) implies (not is_on)
533     end
534
535
536 turn_on_off_ID_434
537     -- Turn on or turn off the heater automatically based on the current
        temperature
538     require
539         desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
540     not (temperature > deviation)
541     do
542
543         if is_on then
544             if temperature  $\geq$  desired_temp + deviation then
545                 is_on := False
546             end
547         else
548             if temperature < desired_temp - deviation then
549                 is_on := True
550             end
551         end
552     ensure

```

```

553     heater_is_turned_off: old (is_on and temperature > desired_temp +
554         deviation) implies (not is_on)
555     heater_remains_on: old (is_on and temperature ≤ desired_temp +
556         deviation) implies is_on
557     heater_is_turned_on: old (not is_on and temperature < desired_temp -
558         deviation) implies is_on
559     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
560         deviation) implies (not is_on)
561 end
562
563 turn_on_off_ID_435
564     -- Turn on or turn off the heater automatically based on the current
565     temperature
566     require
567     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
568     do
569         if is_on then
570             if temperature ≥ desired_temp + deviation then
571                 is_on := False
572             end
573         else
574             if temperature < desired_temp - deviation then
575                 is_on := True
576             end
577         end
578     ensure
579     heater_is_turned_off: old (is_on and temperature > desired_temp +
580         deviation) implies (not is_on)
581     heater_remains_on: not (temperature > deviation) implies ( old (is_on and
582         temperature ≤ desired_temp + deviation) implies is_on)
583     heater_is_turned_on: old (not is_on and temperature < desired_temp -
584         deviation) implies is_on
585     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
586         deviation) implies (not is_on)
587 end
588
589 turn_on_off_ID_437
590     -- Turn on or turn off the heater automatically based on the current
591     temperature
592     require
593     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
594     do
595         if is_on then
596             if temperature ≥ desired_temp + deviation then
597                 is_on := False
598             end
599         end

```

```

593     else
594         if temperature < desired_temp - deviation then
595             is_on := True
596         end
597     end
598     ensure
599         heater_is_turned_off: old (is_on and temperature > desired_temp +
600             deviation) implies (not is_on)
601         heater_remains_on: not (desired_temp > deviation) implies ( old (is_on and
602             temperature ≤ desired_temp + deviation) implies is_on)
603         heater_is_turned_on: old (not is_on and temperature < desired_temp -
604             deviation) implies is_on
605         heater_remains_off: old (not is_on and temperature ≥ desired_temp -
606             deviation) implies (not is_on)
607     end

```

Variant 3 of HEATER

- Fault injection: at line 55, in the body of `turn_on_off`, change the condition of the then branch from “`temperature > desired_temp + deviation`” into “`temperature > desired_temp - deviation`”.
- Resulting failure: as shown in Fig. 45(c), the injected fault results in the violation of postcondition `heater_remains_on` of the procedure `turn_on_off`.
- Cause of the failure: incorrect implementation of the routine body of `turn_on_off`.
- Proof time: 0.250 sec
- Comment: similar to Variant 1 and 2 of `HEATER`, the test is useful as its execution shows a specific path along which the same contract violation in the proof would be raised.
- Possible fixes: 19 valid fixes out of 425 candidate fixes.
- fixing time: 4.17 minutes

Class	Feature	Information	Position	Time
HEATER_3	invariant a...	Verification successful.		0.16
HEATER_3	make (cre...	Verification successful.		0.00
HEATER_3	make	Verification successful.		0.01
HEATER_3	set_tempe...	Verification successful.		0.00
HEATER_3	set_desire...	Verification successful.		0.04
HEATER_3	turn_on_off	Postcondition heater_remains_on may be violated.	70	0.03

(c)

```

1  turn_on_off_ID_366
2      -- Turn on or turn off the heater automatically based on the current
        temperature
3      require
4          desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
5  not (is_on)
6      do
7
8          if is_on then
9              if temperature > desired_temp - deviation then
10                 is_on := False
11             end
12         else
13             if temperature < desired_temp - deviation then
14                 is_on := True
15             end
16         end
17     ensure
18         heater_is_turned_off: old (is_on and temperature > desired_temp +
            deviation) implies (not is_on)

```



```

19     heater_remains_on: old (is_on and temperature ≤ desired_temp +
20         deviation) implies is_on
21     heater_is_turned_on: old (not is_on and temperature < desired_temp -
22         deviation) implies is_on
23     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
24         deviation) implies (not is_on)
25 end
26
27 turn_on_off_ID_369
28     -- Turn on or turn off the heater automatically based on the current
29     temperature
30     require
31     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
32 do
33     if is_on then
34         if temperature > desired_temp - deviation then
35             is_on := False
36         end
37     else
38         if temperature < desired_temp - deviation then
39             is_on := True
40         end
41     end
42     ensure
43     heater_is_turned_off: old (is_on and temperature > desired_temp +
44         deviation) implies (not is_on)
45     heater_remains_on: not (not (is_on)) implies ( old (is_on and temperature ≤
46         desired_temp + deviation) implies is_on)
47     heater_is_turned_on: old (not is_on and temperature < desired_temp -
48         deviation) implies is_on
49     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
50         deviation) implies (not is_on)
51 end
52
53 turn_on_off_ID_372
54     -- Turn on or turn off the heater automatically based on the current
55     temperature
56     require
57     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
58     not ((temperature) > (0))
59 do
60     if is_on then
61         if temperature > desired_temp - deviation then
62             is_on := False
63         end
64     else
65         if temperature < desired_temp - deviation then

```

```

60         is_on := True
61     end
62 end
63 ensure
64     heater_is_turned_off: old (is_on and temperature > desired_temp +
65         deviation) implies (not is_on)
66     heater_remains_on: old (is_on and temperature ≤ desired_temp +
67         deviation) implies is_on
68     heater_is_turned_on: old (not is_on and temperature < desired_temp -
69         deviation) implies is_on
70     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
71         deviation) implies (not is_on)
72 end
73
74 turn_on_off_ID_373
75     -- Turn on or turn off the heater automatically based on the current
76     temperature
77     require
78         desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
79     do
80         if is_on then
81             if temperature > desired_temp - deviation then
82                 is_on := False
83             end
84         else
85             if temperature < desired_temp - deviation then
86                 is_on := True
87             end
88         end
89     ensure
90         heater_is_turned_off: old (is_on and temperature > desired_temp +
91             deviation) implies (not is_on)
92         heater_remains_on: not ((temperature) > (0)) implies ( old (is_on and
93             temperature ≤ desired_temp + deviation) implies is_on)
94         heater_is_turned_on: old (not is_on and temperature < desired_temp -
95             deviation) implies is_on
96         heater_remains_off: old (not is_on and temperature ≥ desired_temp -
97             deviation) implies (not is_on)
98     end
99
100 turn_on_off_ID_374
101     -- Turn on or turn off the heater automatically based on the current
102     temperature
103     require
104         desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
105     not ((temperature) ≥ (0))
106     do

```

```

100
101     if is_on then
102         if temperature > desired_temp - deviation then
103             is_on := False
104         end
105     else
106         if temperature < desired_temp - deviation then
107             is_on := True
108         end
109     end
110 ensure
111     heater_is_turned_off: old (is_on and temperature > desired_temp +
112         deviation) implies (not is_on)
113     heater_remains_on: old (is_on and temperature ≤ desired_temp +
114         deviation) implies is_on
115     heater_is_turned_on: old (not is_on and temperature < desired_temp -
116         deviation) implies is_on
117     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
118         deviation) implies (not is_on)
119 end
120
121 turn_on_off_ID_375
122     -- Turn on or turn off the heater automatically based on the current
123     temperature
124 require
125     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
126 do
127     if is_on then
128         if temperature > desired_temp - deviation then
129             is_on := False
130         end
131     else
132         if temperature < desired_temp - deviation then
133             is_on := True
134         end
135     end
136 ensure
137     heater_is_turned_off: old (is_on and temperature > desired_temp +
138         deviation) implies (not is_on)
139     heater_remains_on: not ((temperature) ≥ (0)) implies ( old (is_on and
140         temperature ≤ desired_temp + deviation) implies is_on)
141     heater_is_turned_on: old (not is_on and temperature < desired_temp -
142         deviation) implies is_on
143     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
144         deviation) implies (not is_on)
145 end

```

```

141 turn_on_off_ID_383
142     -- Turn on or turn off the heater automatically based on the current
        temperature
143     require
144         desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
145     do
146
147         if is_on then
148             if temperature > desired_temp - deviation then
149                 is_on := False
150             end
151         else
152             if temperature < desired_temp - deviation then
153                 is_on := True
154             end
155         end
156     ensure
157         heater_is_turned_off: old (is_on and temperature > desired_temp +
            deviation) implies (not is_on)
158     heater_remains_on: not ((desired_temp > (0)) implies ( old (is_on and
        temperature  $\leq$  desired_temp + deviation) implies is_on)
159     heater_is_turned_on: old (not is_on and temperature < desired_temp -
        deviation) implies is_on
160     heater_remains_off: old (not is_on and temperature  $\geq$  desired_temp -
        deviation) implies (not is_on)
161 end
162
163
164 turn_on_off_ID_385
165     -- Turn on or turn off the heater automatically based on the current
        temperature
166     require
167         desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
168     do
169
170         if is_on then
171             if temperature > desired_temp - deviation then
172                 is_on := False
173             end
174         else
175             if temperature < desired_temp - deviation then
176                 is_on := True
177             end
178         end
179     ensure
180         heater_is_turned_off: old (is_on and temperature > desired_temp +
            deviation) implies (not is_on)
181     heater_remains_on: not ((desired_temp  $\geq$  (0)) implies ( old (is_on and
        temperature  $\leq$  desired_temp + deviation) implies is_on)

```

```

182     heater_is_turned_on: old (not is_on and temperature < desired_temp -
183         deviation) implies is_on
184     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
185         deviation) implies (not is_on)
186 end
187 turn_on_off_ID_393
188     -- Turn on or turn off the heater automatically based on the current
189     temperature
190 require
191     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
192 do
193     if is_on then
194         if temperature > desired_temp - deviation then
195             is_on := False
196         end
197     else
198         if temperature < desired_temp - deviation then
199             is_on := True
200         end
201     end
202 ensure
203     heater_is_turned_off: old (is_on and temperature > desired_temp +
204         deviation) implies (not is_on)
205     heater_remains_on: not ((deviation) > (0)) implies ( old (is_on and temperature
206         ≤ desired_temp + deviation) implies is_on)
207     heater_is_turned_on: old (not is_on and temperature < desired_temp -
208         deviation) implies is_on
209     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
210         deviation) implies (not is_on)
211 end
212
213 turn_on_off_ID_395
214     -- Turn on or turn off the heater automatically based on the current
215     temperature
216 require
217     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
218 do
219     if is_on then
220         if temperature > desired_temp - deviation then
221             is_on := False
222         end
223     else
224         if temperature < desired_temp - deviation then
225             is_on := True
226         end
227     end

```

```

224     end
225     ensure
226         heater_is_turned_off: old (is_on and temperature > desired_temp +
            deviation) implies (not is_on)
227 heater_remains_on: not ((deviation) ≥ 0) implies ( old (is_on and temperature
            ≤ desired_temp + deviation) implies is_on)
228         heater_is_turned_on: old (not is_on and temperature < desired_temp -
            deviation) implies is_on
229         heater_remains_off: old (not is_on and temperature ≥ desired_temp -
            deviation) implies (not is_on)
230     end
231
232
233 turn_on_off_ID_402
234     -- Turn on or turn off the heater automatically based on the current
        temperature
235     require
236         desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
237 not ((temperature) > (deviation))
238     do
239
240         if is_on then
241             if temperature > desired_temp - deviation then
242                 is_on := False
243             end
244         else
245             if temperature < desired_temp - deviation then
246                 is_on := True
247             end
248         end
249     ensure
250         heater_is_turned_off: old (is_on and temperature > desired_temp +
            deviation) implies (not is_on)
251         heater_remains_on: old (is_on and temperature ≤ desired_temp +
            deviation) implies is_on
252         heater_is_turned_on: old (not is_on and temperature < desired_temp -
            deviation) implies is_on
253         heater_remains_off: old (not is_on and temperature ≥ desired_temp -
            deviation) implies (not is_on)
254     end
255
256
257 turn_on_off_ID_403
258     -- Turn on or turn off the heater automatically based on the current
        temperature
259     require
260         desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
261     do
262
263         if is_on then

```

```

264         if temperature > desired_temp - deviation then
265             is_on := False
266         end
267     else
268         if temperature < desired_temp - deviation then
269             is_on := True
270         end
271     end
272 ensure
273     heater_is_turned_off: old (is_on and temperature > desired_temp +
274 heater_remains_on: not ((temperature) > (deviation)) implies ( old (is_on and
275     heater_is_turned_on: old (not is_on and temperature < desired_temp -
276     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
277     deviation) implies (not is_on)
278
279
280 turn_on_off_ID_404
281     -- Turn on or turn off the heater automatically based on the current
282     temperature
283
284 require
285     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
286 not ((temperature) ≥ (deviation))
287 do
288     if is_on then
289         if temperature > desired_temp - deviation then
290             is_on := False
291         end
292     else
293         if temperature < desired_temp - deviation then
294             is_on := True
295         end
296     end
297 ensure
298     heater_is_turned_off: old (is_on and temperature > desired_temp +
299     heater_remains_on: old (is_on and temperature ≤ desired_temp +
300     heater_is_turned_on: old (not is_on and temperature < desired_temp -
301     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
302     deviation) implies (not is_on)
303
304 end
305
306 turn_on_off_ID_405

```

```

305     -- Turn on or turn off the heater automatically based on the current
        temperature
306   require
307     desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
308   do
309
310     if is_on then
311       if temperature > desired_temp - deviation then
312         is_on := False
313       end
314     else
315       if temperature < desired_temp - deviation then
316         is_on := True
317       end
318     end
319   ensure
320     heater_is_turned_off: old (is_on and temperature > desired_temp +
        deviation) implies (not is_on)
321   heater_remains_on: not ((temperature)  $\geq$  (deviation)) implies ( old (is_on and
        temperature  $\leq$  desired_temp + deviation) implies is_on)
322     heater_is_turned_on: old (not is_on and temperature < desired_temp -
        deviation) implies is_on
323     heater_remains_off: old (not is_on and temperature  $\geq$  desired_temp -
        deviation) implies (not is_on)
324   end
325
326
327   turn_on_off_ID_413
328     -- Turn on or turn off the heater automatically based on the current
        temperature
329   require
330     desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
331   do
332
333     if is_on then
334       if temperature > desired_temp - deviation then
335         is_on := False
336       end
337     else
338       if temperature < desired_temp - deviation then
339         is_on := True
340       end
341     end
342   ensure
343     heater_is_turned_off: old (is_on and temperature > desired_temp +
        deviation) implies (not is_on)
344   heater_remains_on: not ((desired_temp) > (deviation)) implies ( old (is_on and
        temperature  $\leq$  desired_temp + deviation) implies is_on)
345     heater_is_turned_on: old (not is_on and temperature < desired_temp -
        deviation) implies is_on

```



```

346     heater_remains_off: old (not is_on and temperature  $\geq$  desired_temp -
347         deviation) implies (not is_on)
348 end
349
350 turn_on_off_ID_415
351     -- Turn on or turn off the heater automatically based on the current
352         temperature
353 require
354     desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
355 do
356     if is_on then
357         if temperature > desired_temp - deviation then
358             is_on := False
359         end
360     else
361         if temperature < desired_temp - deviation then
362             is_on := True
363         end
364     end
365 ensure
366     heater_is_turned_off: old (is_on and temperature > desired_temp +
367         deviation) implies (not is_on)
368     heater_remains_on: not ((desired_temp  $\geq$  (deviation)) implies ( old (is_on and
369         temperature  $\leq$  desired_temp + deviation) implies is_on)
370     heater_is_turned_on: old (not is_on and temperature < desired_temp -
371         deviation) implies is_on
372     heater_remains_off: old (not is_on and temperature  $\geq$  desired_temp -
373         deviation) implies (not is_on)
374 end
375
376 turn_on_off_ID_421
377     -- Turn on or turn off the heater automatically based on the current
378         temperature
379 require
380     desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
381 do
382     if is_on then
383         if temperature > desired_temp - deviation then
384             is_on := False
385         end
386     else
387         if temperature < desired_temp - deviation then
388             is_on := True
389         end
390     end
391 ensure

```

```

389     heater_is_turned_off: old (is_on and temperature > desired_temp +
        deviation) implies (not is_on)
390 heater_remains_on: not (deviation = 2) implies ( old (is_on and temperature ≤
        desired_temp + deviation) implies is_on)
391     heater_is_turned_on: old (not is_on and temperature < desired_temp -
        deviation) implies is_on
392     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
        deviation) implies (not is_on)
393 end
394
395
396 turn_on_off_ID_422
397     -- Turn on or turn off the heater automatically based on the current
        temperature
398     require
399         desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
400 not (temperature > deviation)
401     do
402
403         if is_on then
404             if temperature > desired_temp - deviation then
405                 is_on := False
406             end
407         else
408             if temperature < desired_temp - deviation then
409                 is_on := True
410             end
411         end
412     ensure
413         heater_is_turned_off: old (is_on and temperature > desired_temp +
            deviation) implies (not is_on)
414         heater_remains_on: old (is_on and temperature ≤ desired_temp +
            deviation) implies is_on
415         heater_is_turned_on: old (not is_on and temperature < desired_temp -
            deviation) implies is_on
416         heater_remains_off: old (not is_on and temperature ≥ desired_temp -
            deviation) implies (not is_on)
417     end
418
419
420 turn_on_off_ID_423
421     -- Turn on or turn off the heater automatically based on the current
        temperature
422     require
423         desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
424     do
425
426         if is_on then
427             if temperature > desired_temp - deviation then
428                 is_on := False

```

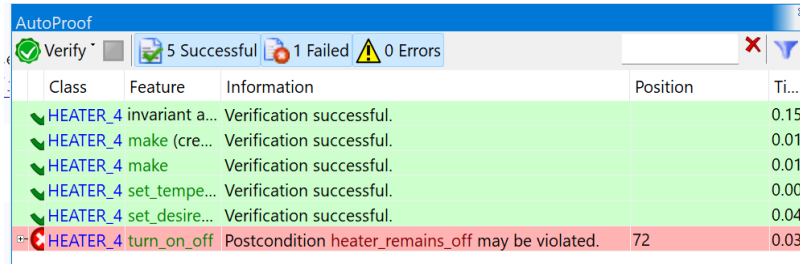
```

429         end
430     else
431         if temperature < desired_temp - deviation then
432             is_on := True
433         end
434     end
435 ensure
436     heater_is_turned_off: old (is_on and temperature > desired_temp +
437         deviation) implies (not is_on)
437 heater_remains_on: not (temperature > deviation) implies ( old (is_on and
438     temperature ≤ desired_temp + deviation) implies is_on)
438     heater_is_turned_on: old (not is_on and temperature < desired_temp -
439     deviation) implies is_on
439     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
440     deviation) implies (not is_on)
440 end
441
442
443 turn_on_off_ID_425
444     -- Turn on or turn off the heater automatically based on the current
445     temperature
445 require
446     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
447 do
448
449     if is_on then
450         if temperature > desired_temp - deviation then
451             is_on := False
452         end
453     else
454         if temperature < desired_temp - deviation then
455             is_on := True
456         end
457     end
458 ensure
459     heater_is_turned_off: old (is_on and temperature > desired_temp +
460         deviation) implies (not is_on)
460 heater_remains_on: not (desired_temp > deviation) implies ( old (is_on and
461     temperature ≤ desired_temp + deviation) implies is_on)
461     heater_is_turned_on: old (not is_on and temperature < desired_temp -
462     deviation) implies is_on
462     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
463     deviation) implies (not is_on)
463 end

```

Variant 4 of HEATER

- Fault injection: at line 59, change the condition of the then branch from “`temperature < desired_temp - deviation`” into “`temperature < desired_temp + deviation`”.
- Resulting failure: as shown in Fig. 45(d), the injected fault leads to the violation of the postcondition `heater_remains_off` in the procedure `turn_on_off`.
- Cause of the failure: incorrect implementation of the routine body of `turn_on_off`.
- Proof time: 0.246 sec
- Comment: similar to the previous variants, the test is useful as its execution shows a specific path along which the same contract violation in the proof would be raised.
- Possible fixes: 17 valid fixes out of 429 candidate fixes
- fixing time: 4.65 minutes



Class	Feature	Information	Position	Ti...
✓ HEATER_4	invariant a...	Verification successful.		0.15
✓ HEATER_4	make (cre...	Verification successful.		0.01
✓ HEATER_4	make	Verification successful.		0.01
✓ HEATER_4	set_tempe...	Verification successful.		0.00
✓ HEATER_4	set_desire...	Verification successful.		0.04
✗ HEATER_4	turn_on_off	Postcondition <code>heater_remains_off</code> may be violated.	72	0.03

(d)

```

1
2 turn_on_off_ID_367
3   -- Turn on or turn off the heater automatically based on the current
   temperature
4   require
5     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
6   do
7
8     if is_on then
9       if temperature > desired_temp + deviation then
10         is_on := False
11       end
12     else
13       if temperature < desired_temp + deviation then
14         is_on := True
15       end
16     end
17   ensure
18     heater_is_turned_off: old (is_on and temperature > desired_temp +
   deviation) implies (not is_on)

```

```

19     heater_remains_on: old (is_on and temperature  $\leq$  desired_temp +
20         deviation) implies is_on
21     heater_is_turned_on: old (not is_on and temperature < desired_temp -
22         deviation) implies is_on
23     heater_remains_off: not (is_on) implies ( old (not is_on and temperature  $\geq$ 
24         desired_temp - deviation) implies (not is_on))
25     end
26 turn_on_off_ID_368
27     -- Turn on or turn off the heater automatically based on the current
28     temperature
29     require
30     desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
31 not (not (is_on))
32 do
33     if is_on then
34         if temperature > desired_temp + deviation then
35             is_on := False
36         end
37     else
38         if temperature < desired_temp + deviation then
39             is_on := True
40         end
41     end
42 ensure
43     heater_is_turned_off: old (is_on and temperature > desired_temp +
44         deviation) implies (not is_on)
45     heater_remains_on: old (is_on and temperature  $\leq$  desired_temp +
46         deviation) implies is_on
47     heater_is_turned_on: old (not is_on and temperature < desired_temp -
48         deviation) implies is_on
49     heater_remains_off: old (not is_on and temperature  $\geq$  desired_temp -
50         deviation) implies (not is_on)
51 end
52 turn_on_off_ID_372
53     -- Turn on or turn off the heater automatically based on the current
54     temperature
55     require
56     desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
57 not ((temperature) > (0))
58 do
59     if is_on then
60         if temperature > desired_temp + deviation then
61             is_on := False
62         end
63     else
64         if temperature < desired_temp + deviation then

```

```

60         is_on := True
61     end
62 end
63 ensure
64     heater_is_turned_off: old (is_on and temperature > desired_temp +
65         deviation) implies (not is_on)
66     heater_remains_on: old (is_on and temperature ≤ desired_temp +
67         deviation) implies is_on
68     heater_is_turned_on: old (not is_on and temperature < desired_temp -
69         deviation) implies is_on
70     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
71         deviation) implies (not is_on)
72 end
73 turn_on_off_ID_373
74     -- Turn on or turn off the heater automatically based on the current
75     temperature
76 require
77     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
78 do
79     if is_on then
80         if temperature > desired_temp + deviation then
81             is_on := False
82         end
83     else
84         if temperature < desired_temp + deviation then
85             is_on := True
86         end
87     end
88 ensure
89     heater_is_turned_off: old (is_on and temperature > desired_temp +
90         deviation) implies (not is_on)
91     heater_remains_on: old (is_on and temperature ≤ desired_temp +
92         deviation) implies is_on
93     heater_is_turned_on: old (not is_on and temperature < desired_temp -
94         deviation) implies is_on
95     heater_remains_off: not ((temperature) > (0)) implies ( old (not is_on and
96         temperature ≥ desired_temp - deviation) implies (not is_on))
97 end
98
99 turn_on_off_ID_374
100     -- Turn on or turn off the heater automatically based on the current
101     temperature
102 require
103     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
104 not ((temperature) ≥ (0))
105 do
106     if is_on then

```

```

100         if temperature > desired_temp + deviation then
101             is_on := False
102         end
103     else
104         if temperature < desired_temp + deviation then
105             is_on := True
106         end
107     end
108 ensure
109     heater_is_turned_off: old (is_on and temperature > desired_temp +
110         deviation) implies (not is_on)
111     heater_remains_on: old (is_on and temperature ≤ desired_temp +
112         deviation) implies is_on
113     heater_is_turned_on: old (not is_on and temperature < desired_temp -
114         deviation) implies is_on
115     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
116         deviation) implies (not is_on)
117 end
118
119 turn_on_off_ID_375
120     -- Turn on or turn off the heater automatically based on the current
121     temperature
122 require
123     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
124 do
125     if is_on then
126         if temperature > desired_temp + deviation then
127             is_on := False
128         end
129     else
130         if temperature < desired_temp + deviation then
131             is_on := True
132         end
133     end
134 ensure
135     heater_is_turned_off: old (is_on and temperature > desired_temp +
136         deviation) implies (not is_on)
137     heater_remains_on: old (is_on and temperature ≤ desired_temp +
138         deviation) implies is_on
139     heater_is_turned_on: old (not is_on and temperature < desired_temp -
140         deviation) implies is_on
141     heater_remains_off: not ((temperature) ≥ (0)) implies ( old (not is_on and
142         temperature ≥ desired_temp - deviation) implies (not is_on))
143 end
144
145 turn_on_off_ID_383
146     -- Turn on or turn off the heater automatically based on the current
147     temperature
148 require

```

```

140     desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
141 do
142
143     if is_on then
144         if temperature > desired_temp + deviation then
145             is_on := False
146         end
147     else
148         if temperature < desired_temp + deviation then
149             is_on := True
150         end
151     end
152 ensure
153     heater_is_turned_off: old (is_on and temperature > desired_temp +
154         deviation) implies (not is_on)
155     heater_remains_on: old (is_on and temperature  $\leq$  desired_temp +
156         deviation) implies is_on
157     heater_is_turned_on: old (not is_on and temperature < desired_temp -
158         deviation) implies is_on
159 heater_remains_off: not ((desired_temp) > (0)) implies ( old (not is_on and
160     temperature  $\geq$  desired_temp - deviation) implies (not is_on))
161 end
162
163 turn_on_off_ID_385
164     -- Turn on or turn off the heater automatically based on the current
165     temperature
166 require
167     desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
168 do
169
170     if is_on then
171         if temperature > desired_temp + deviation then
172             is_on := False
173         end
174     else
175         if temperature < desired_temp + deviation then
176             is_on := True
177         end
178     end
179 ensure
180     heater_is_turned_off: old (is_on and temperature > desired_temp +
181         deviation) implies (not is_on)
182     heater_remains_on: old (is_on and temperature  $\leq$  desired_temp +
183         deviation) implies is_on
184     heater_is_turned_on: old (not is_on and temperature < desired_temp -
185         deviation) implies is_on
186 heater_remains_off: not ((desired_temp)  $\geq$  (0)) implies ( old (not is_on and
187     temperature  $\geq$  desired_temp - deviation) implies (not is_on))
188 end

```



```

181 turn_on_off_ID_393
182     -- Turn on or turn off the heater automatically based on the current
        temperature
183     require
184         desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
185     do
186
187         if is_on then
188             if temperature > desired_temp + deviation then
189                 is_on := False
190             end
191         else
192             if temperature < desired_temp + deviation then
193                 is_on := True
194             end
195         end
196     ensure
197         heater_is_turned_off: old (is_on and temperature > desired_temp +
            deviation) implies (not is_on)
198         heater_remains_on: old (is_on and temperature ≤ desired_temp +
            deviation) implies is_on
199         heater_is_turned_on: old (not is_on and temperature < desired_temp -
            deviation) implies is_on
200     heater_remains_off: not ((deviation) > (0)) implies ( old (not is_on and
        temperature ≥ desired_temp - deviation) implies (not is_on))
201     end
202
203 turn_on_off_ID_395
204     -- Turn on or turn off the heater automatically based on the current
        temperature
205     require
206         desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
207     do
208
209         if is_on then
210             if temperature > desired_temp + deviation then
211                 is_on := False
212             end
213         else
214             if temperature < desired_temp + deviation then
215                 is_on := True
216             end
217         end
218     ensure
219         heater_is_turned_off: old (is_on and temperature > desired_temp +
            deviation) implies (not is_on)
220         heater_remains_on: old (is_on and temperature ≤ desired_temp +
            deviation) implies is_on
221         heater_is_turned_on: old (not is_on and temperature < desired_temp -
            deviation) implies is_on

```

```

222 heater_remains_off: not ((deviation) ≥ (0)) implies ( old (not is_on and
    temperature ≥ desired_temp - deviation) implies (not is_on))
223 end
224
225 turn_on_off_ID_404
226   -- Turn on or turn off the heater automatically based on the current
    temperature
227   require
228     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
229   not ((temperature) > (deviation))
230   do
231
232     if is_on then
233       if temperature > desired_temp + deviation then
234         is_on := False
235       end
236     else
237       if temperature < desired_temp + deviation then
238         is_on := True
239       end
240     end
241   ensure
242     heater_is_turned_off: old (is_on and temperature > desired_temp +
        deviation) implies (not is_on)
243     heater_remains_on: old (is_on and temperature ≤ desired_temp +
        deviation) implies is_on
244     heater_is_turned_on: old (not is_on and temperature < desired_temp -
        deviation) implies is_on
245     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
        deviation) implies (not is_on)
246   end
247
248 turn_on_off_ID_405
249   -- Turn on or turn off the heater automatically based on the current
    temperature
250   require
251     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
252   do
253
254     if is_on then
255       if temperature > desired_temp + deviation then
256         is_on := False
257       end
258     else
259       if temperature < desired_temp + deviation then
260         is_on := True
261       end
262     end
263   ensure

```

```

264     heater_is_turned_off: old (is_on and temperature > desired_temp +
      deviation) implies (not is_on)
265     heater_remains_on: old (is_on and temperature ≤ desired_temp +
      deviation) implies is_on
266     heater_is_turned_on: old (not is_on and temperature < desired_temp -
      deviation) implies is_on
267 heater_remains_off: not ((temperature) > (deviation)) implies ( old (not is_on
      and temperature ≥ desired_temp - deviation) implies (not is_on))
268 end
269
270 turn_on_off_ID_406
271     -- Turn on or turn off the heater automatically based on the current
      temperature
272     require
273         desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
274 not ((temperature) ≥ (deviation))
275     do
276
277         if is_on then
278             if temperature > desired_temp + deviation then
279                 is_on := False
280             end
281         else
282             if temperature < desired_temp + deviation then
283                 is_on := True
284             end
285         end
286     ensure
287         heater_is_turned_off: old (is_on and temperature > desired_temp +
      deviation) implies (not is_on)
288         heater_remains_on: old (is_on and temperature ≤ desired_temp +
      deviation) implies is_on
289         heater_is_turned_on: old (not is_on and temperature < desired_temp -
      deviation) implies is_on
290         heater_remains_off: old (not is_on and temperature ≥ desired_temp -
      deviation) implies (not is_on)
291     end
292
293 turn_on_off_ID_407
294     -- Turn on or turn off the heater automatically based on the current
      temperature
295     require
296         desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
297     do
298
299         if is_on then
300             if temperature > desired_temp + deviation then
301                 is_on := False
302             end
303         else

```

```

304         if temperature < desired_temp + deviation then
305             is_on := True
306         end
307     end
308 ensure
309     heater_is_turned_off: old (is_on and temperature > desired_temp +
310         deviation) implies (not is_on)
311     heater_remains_on: old (is_on and temperature ≤ desired_temp +
312         deviation) implies is_on
313     heater_is_turned_on: old (not is_on and temperature < desired_temp -
314         deviation) implies is_on
315 heater_remains_off: not ((temperature) ≥ (deviation)) implies ( old (not is_on
316     and temperature ≥ desired_temp - deviation) implies (not is_on))
317 end
318
319 turn_on_off_ID_415
320     -- Turn on or turn off the heater automatically based on the current
321     temperature
322 require
323     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
324 do
325     if is_on then
326         if temperature > desired_temp + deviation then
327             is_on := False
328         end
329     else
330         if temperature < desired_temp + deviation then
331             is_on := True
332         end
333     end
334 ensure
335     heater_is_turned_off: old (is_on and temperature > desired_temp +
336         deviation) implies (not is_on)
337     heater_remains_on: old (is_on and temperature ≤ desired_temp +
338         deviation) implies is_on
339     heater_is_turned_on: old (not is_on and temperature < desired_temp -
340         deviation) implies is_on
341 heater_remains_off: not ((desired_temp) > (deviation)) implies ( old (not is_on
342     and temperature ≥ desired_temp - deviation) implies (not is_on))
343 end
344
345 turn_on_off_ID_417
346     -- Turn on or turn off the heater automatically based on the current
347     temperature
348 require
349     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
350 do
351     if is_on then

```

```

344         if temperature > desired_temp + deviation then
345             is_on := False
346         end
347     else
348         if temperature < desired_temp + deviation then
349             is_on := True
350         end
351     end
352 ensure
353     heater_is_turned_off: old (is_on and temperature > desired_temp +
354         deviation) implies (not is_on)
354     heater_remains_on: old (is_on and temperature ≤ desired_temp +
355         deviation) implies is_on
355     heater_is_turned_on: old (not is_on and temperature < desired_temp -
356         deviation) implies is_on
356 heater_remains_off: not ((desired_temp) ≥ (deviation)) implies ( old (not is_on
357     and temperature ≥ desired_temp - deviation) implies (not is_on))
357 end
358
359 turn_on_off_ID_423
360     -- Turn on or turn off the heater automatically based on the current
361     temperature
361 require
362     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
363 do
364
365     if is_on then
366         if temperature > desired_temp + deviation then
367             is_on := False
368         end
369     else
370         if temperature < desired_temp + deviation then
371             is_on := True
372         end
373     end
374 ensure
375     heater_is_turned_off: old (is_on and temperature > desired_temp +
376         deviation) implies (not is_on)
376     heater_remains_on: old (is_on and temperature ≤ desired_temp +
377         deviation) implies is_on
377     heater_is_turned_on: old (not is_on and temperature < desired_temp -
378         deviation) implies is_on
378 heater_remains_off: not (deviation = 2) implies ( old (not is_on and
379     temperature ≥ desired_temp - deviation) implies (not is_on))
379 end
380
381 turn_on_off_ID_426
382     -- Turn on or turn off the heater automatically based on the current
383     temperature
383 require

```

```

384     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
385 not (temperature > deviation)
386 do
387
388     if is_on then
389         if temperature > desired_temp + deviation then
390             is_on := False
391         end
392     else
393         if temperature < desired_temp + deviation then
394             is_on := True
395         end
396     end
397 ensure
398     heater_is_turned_off: old (is_on and temperature > desired_temp +
399         deviation) implies (not is_on)
400     heater_remains_on: old (is_on and temperature ≤ desired_temp +
401         deviation) implies is_on
402     heater_is_turned_on: old (not is_on and temperature < desired_temp -
403         deviation) implies is_on
404     heater_remains_off: old (not is_on and temperature ≥ desired_temp -
405         deviation) implies (not is_on)
406 end
407
408 turn_on_off_ID_427
409 -- Turn on or turn off the heater automatically based on the current
410 -- temperature
411 require
412     desired_temp_valid: desired_temp ≥ 10 and desired_temp ≤ 100
413 do
414
415     if is_on then
416         if temperature > desired_temp + deviation then
417             is_on := False
418         end
419     else
420         if temperature < desired_temp + deviation then
421             is_on := True
422         end
423     end
424 ensure
425     heater_is_turned_off: old (is_on and temperature > desired_temp +
426         deviation) implies (not is_on)
427     heater_remains_on: old (is_on and temperature ≤ desired_temp +
428         deviation) implies is_on
429     heater_is_turned_on: old (not is_on and temperature < desired_temp -
430         deviation) implies is_on
431     heater_remains_off: not (temperature > deviation) implies (old (not is_on and
432         temperature ≥ desired_temp - deviation) implies (not is_on))
433 end

```

```

425
426 turn_on_off_ID_429
427     -- Turn on or turn off the heater automatically based on the current
         temperature
428     require
429         desired_temp_valid: desired_temp  $\geq$  10 and desired_temp  $\leq$  100
430     do
431
432         if is_on then
433             if temperature > desired_temp + deviation then
434                 is_on := False
435             end
436         else
437             if temperature < desired_temp - deviation then
438                 is_on := True
439             end
440         end
441     ensure
442         heater_is_turned_off: old (is_on and temperature > desired_temp +
            deviation) implies (not is_on)
443         heater_remains_on: old (is_on and temperature  $\leq$  desired_temp +
            deviation) implies is_on
444         heater_is_turned_on: old (not is_on and temperature < desired_temp -
            deviation) implies is_on
445     heater_remains_off: not (desired_temp > deviation) implies ( old (not is_on and
        temperature  $\geq$  desired_temp - deviation) implies (not is_on))
446     end

```

1.5 LAMP

The **LAMP** class, as presented below, implements a lamp that has a switch and a dimmer. Its light intensity has three levels: low, medium and high. When the lamp is turned on, its light intensity will be the same as its intensity before it was last turned off. Fig.45 shows the verification result: this version of **LAMP** is correctly verified. Based on the verified version, 4 faulty variants of **LAMP** class are created, which are discussed below.

```
1 class
2     LAMP
3 feature
4     light_intensity: INTEGER
5         -- Light intensity of the lamp
6     is_on: BOOLEAN
7         -- Is the lamp on?
8     previous_light_intensity: INTEGER
9         -- Light intensity of the lamp before it was last turned off
10    High_intensity: INTEGER = 100
11        -- High light intensity
12    Medium_intensity: INTEGER = 75
13        -- Medium light intensity
14    Low_intensity: INTEGER = 25
15        -- Low light intensity
16    Zero_intensity: INTEGER = 0
17        -- Zero light intensity
18 feature
19     turn_on_off
20         -- Turn on the lamp if it is off; turn off the lamp if it is on
21     do
22         if not is_on then
23             is_on := True
24             if previous_light_intensity > 0 then
25                 light_intensity := previous_light_intensity
26             else
27                 light_intensity := Low_intensity
28             end
29         else
30             is_on := False
31             previous_light_intensity := light_intensity
32             light_intensity := Zero_intensity
33         end
34     ensure
35         turn_on_1: old (not is_on and previous_light_intensity > 0) implies (
36             is_on and light_intensity = old previous_light_intensity)
37         turn_on_2: old (not is_on and previous_light_intensity = 0) implies (
38             is_on and light_intensity = Low_intensity)
```



```

37         turn_off: old is_on implies (not is_on and previous_light_intensity =
           old light_intensity and light_intensity = Zero_intensity)
38     end
39     adjust_light
40         -- Adjust the light intensity
41     require
42         lamp_is_on: is_on = True
43     do
44         if light_intensity = Low_intensity then
45             light_intensity := Medium_intensity
46         elseif light_intensity = Medium_intensity then
47             light_intensity := High_intensity
48         elseif light_intensity = High_intensity then
49             light_intensity := Low_intensity
50         end
51     ensure
52         from_low_to_medium: old light_intensity = Low_intensity implies
           light_intensity = Medium_intensity
53         from_medium_to_high: old light_intensity = Medium_intensity implies
           light_intensity = High_intensity
54         from_high_to_low: old light_intensity = High_intensity implies
           light_intensity = Low_intensity
55     end
56 invariant
57     value_of_light_intensity: light_intensity = Zero_intensity or light_intensity
           = Low_intensity or light_intensity = Medium_intensity or light_intensity
           = High_intensity
58     value_of_previous_intensity: previous_light_intensity = Zero_intensity or
           previous_light_intensity = Low_intensity or previous_light_intensity =
           Medium_intensity or previous_light_intensity = High_intensity
59     light_intensity_when_off: is_on = (light_intensity ≠ Zero_intensity)
60 end

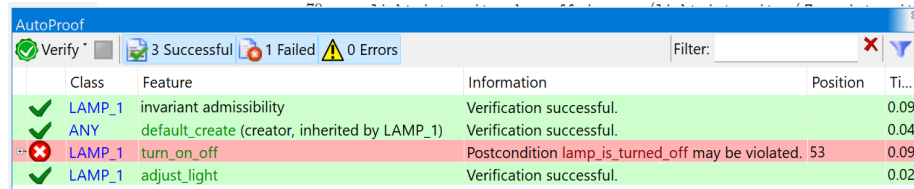
```

AutoProof					
Verify 4 Successful 0 Failed 0 Errors					
	Class	Feature	Information	Position	Ti...
✓	LAMP_1	invariant admissibility	Verification successful.		0.14
✓	ANY	default_create (creator, inherited by LAMP_1)	Verification successful.		0.05
✓	LAMP_1	turn_on_off	Verification successful.		0.04
✓	LAMP_1	adjust_light	Verification successful.		0.02

Fig. 45. Proof result of **LAMP** in AutoProof

Variant 1 of LAMP

- Fault injection: in the body of `turn_on_off`, switch the order of line 40 and line 41.
- Resulting failure: as shown in Fig. 46(a), the injected fault results in the violation of the postcondition `turn_off` of the procedure `turn_on_off`.
- Cause of the failure: incorrect implementation of the routine body of `turn_on_off`; the value of `light_intensity` should be stored into `previous_light_intensity` before being assigned to a new value.
- Proof time: 0.250 sec
- Comment: the test is useful as its execution shows a specific trace illustrating how the program goes to the same contract violation as in the proof.
- Possible fixes: No valid fixes

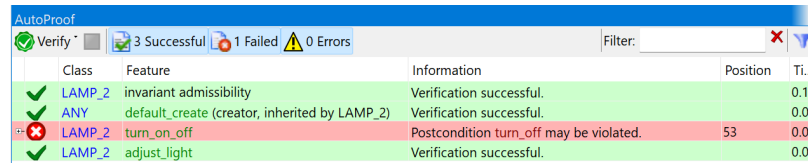


	Class	Feature	Information	Position	Ti...
✓	LAMP_1	invariant admissibility	Verification successful.		0.09
✓	ANY	default_create (creator, inherited by LAMP_1)	Verification successful.		0.04
✗	LAMP_1	turn_on_off	Postcondition lamp_is_turned_off may be violated.	53	0.09
✓	LAMP_1	adjust_light	Verification successful.		0.02

(a)

Variant 2 of LAMP

- Fault injection: at line 40, remove the assignment `previous_light_intensity := light_intensity`.
- Resulting failure: as shown in Fig. 46(b), the injected fault leads to the violation of postcondition `turn_off` in the procedure `turn_on_off`.
- Cause of the failure: incorrect implementation of the body of `turn_on_off`; the postcondition `turn_off` requires that, the `previous_light_intensity` should store, when the light is turned off, the value of `light_intensity`; this is missing in the implementation.
- Proof time: 0.278 sec
- Comment: the test is useful as its execution shows a specific trace illustrating how the program goes to the same contract violation as in the proof.
- Possible fixes: 10 valid fixes out of 534 candidate fixes.
- fixing time: 6.83 minutes



Class	Feature	Information	Position	Time
LAMP_2	invariant admissibility	Verification successful.		0.13
ANY	default_create (creator, inherited by LAMP_2)	Verification successful.		0.05
LAMP_2	turn_on_off	Postcondition turn_off may be violated.	53	0.08
LAMP_2	adjust_light	Verification successful.		0.02

(b)

```

1
2 turn_on_off_ID_362
3     -- Turn on the lamp if it is off; turn off the lamp if it is on
4 require
5 not (is_on)
6 do
7
8     if not is_on then
9         is_on := True
10        if previous_light_intensity > 0 then
11            light_intensity := previous_light_intensity
12        else
13            light_intensity := Low_intensity
14        end
15    else
16        is_on := False
17        -- previous_light_intensity := light_intensity
18        light_intensity := Zero_intensity
19    end
20 ensure
21    turn_on_1: old (not is_on and previous_light_intensity > 0) implies (
        is_on and light_intensity = old previous_light_intensity)

```

```

22     turn_on_2: old (not is_on and previous_light_intensity = 0) implies (
23         is_on and light_intensity = Low_intensity)
24     turn_off: old is_on implies (not is_on and previous_light_intensity =
25         old light_intensity and light_intensity = Zero_intensity)
26 end
27
28 turn_on_off_ID_368
29     -- Turn on the lamp if it is off; turn off the lamp if it is on
30 require
31 not ((is_on) or else (not (not is_on)))
32 do
33     if not is_on then
34         is_on := True
35         if previous_light_intensity > 0 then
36             light_intensity := previous_light_intensity
37         else
38             light_intensity := Low_intensity
39         end
40     else
41         is_on := False
42         -- previous_light_intensity := light_intensity
43         light_intensity := Zero_intensity
44     end
45 ensure
46 turn_on_1: old (not is_on and previous_light_intensity > 0) implies (
47     is_on and light_intensity = old previous_light_intensity)
48 turn_on_2: old (not is_on and previous_light_intensity = 0) implies (
49     is_on and light_intensity = Low_intensity)
50 turn_off: old is_on implies (not is_on and previous_light_intensity =
51     old light_intensity and light_intensity = Zero_intensity)
52 end
53
54 turn_on_off_ID_370
55     -- Turn on the lamp if it is off; turn off the lamp if it is on
56 require
57 not ((is_on) or else (previous_light_intensity > 0))
58 do
59     if not is_on then
60         is_on := True
61         if previous_light_intensity > 0 then
62             light_intensity := previous_light_intensity
63         else
64             light_intensity := Low_intensity
65         end
66     else
67         is_on := False
68         -- previous_light_intensity := light_intensity
69         light_intensity := Zero_intensity

```

```

67     end
68 ensure
69     turn_on_1: old (not is_on and previous_light_intensity > 0) implies (
70         is_on and light_intensity = old previous_light_intensity)
71     turn_on_2: old (not is_on and previous_light_intensity = 0) implies (
72         is_on and light_intensity = Low_intensity)
73     turn_off: old is_on implies (not is_on and previous_light_intensity =
74         old light_intensity and light_intensity = Zero_intensity)
75 end
76
77 turn_on_off_ID_372
78     -- Turn on the lamp if it is off; turn off the lamp if it is on
79 require
80 not ((is_on) or else (not (previous_light_intensity > 0)))
81 do
82     if not is_on then
83         is_on := True
84         if previous_light_intensity > 0 then
85             light_intensity := previous_light_intensity
86         else
87             light_intensity := Low_intensity
88         end
89     else
90         is_on := False
91         -- previous_light_intensity := light_intensity
92         light_intensity := Zero_intensity
93     end
94 ensure
95     turn_on_1: old (not is_on and previous_light_intensity > 0) implies (
96         is_on and light_intensity = old previous_light_intensity)
97     turn_on_2: old (not is_on and previous_light_intensity = 0) implies (
98         is_on and light_intensity = Low_intensity)
99     turn_off: old is_on implies (not is_on and previous_light_intensity =
100         old light_intensity and light_intensity = Zero_intensity)
101 end
102
103 turn_on_off_ID_380
104     -- Turn on the lamp if it is off; turn off the lamp if it is on
105 require
106 not ((light_intensity) > (0))
107 do
108     if not is_on then
109         is_on := True
110         if previous_light_intensity > 0 then
111             light_intensity := previous_light_intensity
112         else
113             light_intensity := Low_intensity
114         end
115     end

```

```

111     else
112         is_on := False
113         -- previous_light_intensity := light_intensity
114         light_intensity := Zero_intensity
115     end
116 ensure
117     turn_on_1: old (not is_on and previous_light_intensity > 0) implies (
118         is_on and light_intensity = old previous_light_intensity)
119     turn_on_2: old (not is_on and previous_light_intensity = 0) implies (
120         is_on and light_intensity = Low_intensity)
121     turn_off: old is_on implies (not is_on and previous_light_intensity =
122         old light_intensity and light_intensity = Zero_intensity)
123 end
124
125 turn_on_off_ID_438
126     -- Turn on the lamp if it is off; turn off the lamp if it is on
127 require
128     not ((light_intensity) ≥ (low_intensity))
129 do
130     if not is_on then
131         is_on := True
132         if previous_light_intensity > 0 then
133             light_intensity := previous_light_intensity
134         else
135             light_intensity := Low_intensity
136         end
137     else
138         is_on := False
139         -- previous_light_intensity := light_intensity
140         light_intensity := Zero_intensity
141     end
142 ensure
143     turn_on_1: old (not is_on and previous_light_intensity > 0) implies (
144         is_on and light_intensity = old previous_light_intensity)
145     turn_on_2: old (not is_on and previous_light_intensity = 0) implies (
146         is_on and light_intensity = Low_intensity)
147     turn_off: old is_on implies (not is_on and previous_light_intensity =
148         old light_intensity and light_intensity = Zero_intensity)
149 end
150
151 turn_on_off_ID_444
152     -- Turn on the lamp if it is off; turn off the lamp if it is on
153 require
154     not ((light_intensity) > (zero_intensity))
155 do
156     if not is_on then
157         is_on := True
158         if previous_light_intensity > 0 then

```

```

155         light_intensity := previous_light_intensity
156     else
157         light_intensity := Low_intensity
158     end
159 else
160     is_on := False
161     -- previous_light_intensity := light_intensity
162     light_intensity := Zero_intensity
163 end
164 ensure
165     turn_on_1: old (not is_on and previous_light_intensity > 0) implies (
166         is_on and light_intensity = old previous_light_intensity)
167     turn_on_2: old (not is_on and previous_light_intensity = 0) implies (
168         is_on and light_intensity = Low_intensity)
169     turn_off: old is_on implies (not is_on and previous_light_intensity =
170         old light_intensity and light_intensity = Zero_intensity)
171 end
172 turn_on_off_ID_524
173     -- Turn on the lamp if it is off; turn off the lamp if it is on
174 require
175 not (light_intensity ≠ previous_light_intensity)
176 do
177     if not is_on then
178         is_on := True
179         if previous_light_intensity > 0 then
180             light_intensity := previous_light_intensity
181         else
182             light_intensity := Low_intensity
183         end
184     else
185         is_on := False
186         -- previous_light_intensity := light_intensity
187         light_intensity := Zero_intensity
188     end
189 ensure
190     turn_on_1: old (not is_on and previous_light_intensity > 0) implies (
191         is_on and light_intensity = old previous_light_intensity)
192     turn_on_2: old (not is_on and previous_light_intensity = 0) implies (
193         is_on and light_intensity = Low_intensity)
194     turn_off: old is_on implies (not is_on and previous_light_intensity =
195         old light_intensity and light_intensity = Zero_intensity)
196 end
197 turn_on_off_ID_528
198     -- Turn on the lamp if it is off; turn off the lamp if it is on
199 require
200 not (light_intensity ≥ low_intensity)
201 do

```

```

199
200     if not is_on then
201         is_on := True
202         if previous_light_intensity > 0 then
203             light_intensity := previous_light_intensity
204         else
205             light_intensity := Low_intensity
206         end
207     else
208         is_on := False
209         -- previous_light_intensity := light_intensity
210         light_intensity := Zero_intensity
211     end
212 ensure
213     turn_on_1: old (not is_on and previous_light_intensity > 0) implies (
214         is_on and light_intensity = old previous_light_intensity)
215     turn_on_2: old (not is_on and previous_light_intensity = 0) implies (
216         is_on and light_intensity = Low_intensity)
217     turn_off: old is_on implies (not is_on and previous_light_intensity =
218         old light_intensity and light_intensity = Zero_intensity)
219 end
220
221 turn_on_off_ID_530
222     -- Turn on the lamp if it is off; turn off the lamp if it is on
223 require
224     not (light_intensity > zero_intensity)
225 do
226
227     if not is_on then
228         is_on := True
229         if previous_light_intensity > 0 then
230             light_intensity := previous_light_intensity
231         else
232             light_intensity := Low_intensity
233         end
234     else
235         is_on := False
236         -- previous_light_intensity := light_intensity
237         light_intensity := Zero_intensity
238     end
239 ensure
240     turn_on_1: old (not is_on and previous_light_intensity > 0) implies (
241         is_on and light_intensity = old previous_light_intensity)
242     turn_on_2: old (not is_on and previous_light_intensity = 0) implies (
243         is_on and light_intensity = Low_intensity)
244     turn_off: old is_on implies (not is_on and previous_light_intensity =
245         old light_intensity and light_intensity = Zero_intensity)
246 end

```


Variant 3 of LAMP

- Fault injection: at line 59, in the body of `adjust_light`, change the right-hand side of the assignment from “Low_intensity” to “Medium_intensity”.
- Resulting failure: as shown in Fig. 46(c), the injected fault causes the violation of postcondition `from_high_to_low` of the `adjust_light` routine.
- Cause of the failure: incorrect implementation of the routine body of `adjust_light`.
- Proof time: 0.265 sec
- Comment: the test is useful as its execution shows a specific trace illustrating how the program goes to the same contract violation as in the proof.
- Possible fixes: 4 valid fixes out of 422 candidate fixes.
- fixing time: 3.3 minutes

AutoProof					
Verify		3 Successful		1 Failed	0 Errors
	Class	Feature	Information	Position	Time
✓	LAMP_3	invariant admissibility	Verification successful.		0.16
✓	ANY	default_create (creator, inherited by LAMP_3)	Verification successful.		0.05
✓	LAMP_3	turn_on_off	Verification successful.		
✗	LAMP_3	adjust_light	Postcondition <code>from_high_to_low</code> may be violated.	72	0.05

(c)

```

1
2 adjust_light_ID_354
3   -- Adjust the light intensity
4   require
5     lamp_is_on: is_on = True
6 not (light_intensity = high_intensity)
7   do
8
9     if light_intensity = Low_intensity then
10      light_intensity := Medium_intensity
11    elseif light_intensity = Medium_intensity then
12      light_intensity := High_intensity
13    elseif light_intensity = High_intensity then
14      light_intensity := Medium_intensity
15    end
16  ensure
17    from_low_to_medium: old light_intensity = Low_intensity implies
18      light_intensity = Medium_intensity
19    from_medium_to_high: old light_intensity = Medium_intensity implies
20      light_intensity = High_intensity
21    from_high_to_low: old light_intensity = High_intensity implies
22      light_intensity = Low_intensity
23  end
24 adjust_light_ID_394

```

```

23     -- Adjust the light intensity
24     require
25         lamp_is_on: is_on = True
26 not ((light_intensity) = (high_intensity))
27     do
28
29         if light_intensity = Low_intensity then
30             light_intensity := Medium_intensity
31         elseif light_intensity = Medium_intensity then
32             light_intensity := High_intensity
33         elseif light_intensity = High_intensity then
34             light_intensity := Medium_intensity
35         end
36     ensure
37         from_low_to_medium: old light_intensity = Low_intensity implies
38             light_intensity = Medium_intensity
39         from_medium_to_high: old light_intensity = Medium_intensity implies
40             light_intensity = High_intensity
41         from_high_to_low: old light_intensity = High_intensity implies
42             light_intensity = Low_intensity
43     end
44
45 adjust_light_ID_398
46     -- Adjust the light intensity
47     require
48         lamp_is_on: is_on = True
49 not ((light_intensity) ≥ (high_intensity))
50     do
51
52         if light_intensity = Low_intensity then
53             light_intensity := Medium_intensity
54         elseif light_intensity = Medium_intensity then
55             light_intensity := High_intensity
56         elseif light_intensity = High_intensity then
57             light_intensity := Medium_intensity
58         end
59     ensure
60         from_low_to_medium: old light_intensity = Low_intensity implies
61             light_intensity = Medium_intensity
62         from_medium_to_high: old light_intensity = Medium_intensity implies
63             light_intensity = High_intensity
64         from_high_to_low: old light_intensity = High_intensity implies
65             light_intensity = Low_intensity
66     end
67
68 adjust_light_ID_406
69     -- Adjust the light intensity
70     require
71         lamp_is_on: is_on = True
72 not ((light_intensity) > (low_intensity))

```

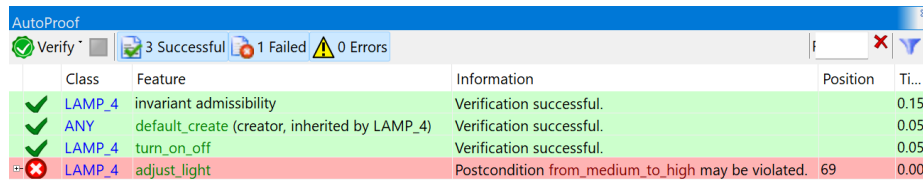
```

67     do
68
69         if light_intensity = Low_intensity then
70             light_intensity := Medium_intensity
71         elseif light_intensity = Medium_intensity then
72             light_intensity := High_intensity
73         elseif light_intensity = High_intensity then
74             light_intensity := Medium_intensity
75         end
76     ensure
77         from_low_to_medium: old light_intensity = Low_intensity implies
78             light_intensity = Medium_intensity
79         from_medium_to_high: old light_intensity = Medium_intensity implies
80             light_intensity = High_intensity
81         from_high_to_low: old light_intensity = High_intensity implies
82             light_intensity = Low_intensity
83     end
84
85 adjust_light_ID_414
86     -- Adjust the light intensity
87     require
88         lamp_is_on: is_on = True
89     not (light_intensity = 100)
90     do
91
92         if light_intensity = Low_intensity then
93             light_intensity := Medium_intensity
94         elseif light_intensity = Medium_intensity then
95             light_intensity := High_intensity
96         elseif light_intensity = High_intensity then
97             light_intensity := Medium_intensity
98         end
99     ensure
100         from_low_to_medium: old light_intensity = Low_intensity implies
101             light_intensity = Medium_intensity
102         from_medium_to_high: old light_intensity = Medium_intensity implies
103             light_intensity = High_intensity
104         from_high_to_low: old light_intensity = High_intensity implies
105             light_intensity = Low_intensity
106     end

```

Variant 4 of LAMP

- Fault injection: at line 57, change the right-hand side of the assignment from “High_intensity” to “Medium_intensity”.
- Resulting failure: as shown in Fig. 46(d), the postcondition `from_high_to_low` is violated.
- Cause of the failure: incorrect implementation.
- Proof time: 0.270 sec
- Possible fixes: 4 valid fixes out of 406 candidate fixes.
- fixing time: 3.62 minutes



Class	Feature	Information	Position	Ti...
✓ LAMP_4	invariant admissibility	Verification successful.		0.15
✓ ANY	default_create (creator, inherited by LAMP_4)	Verification successful.		0.05
✓ LAMP_4	turn_on_off	Verification successful.		0.05
✗ LAMP_4	adjust_light	Postcondition <code>from_medium_to_high</code> may be violated.	69	0.00

(d)

```

1
2 adjust_light_ID_332
3   -- Adjust the light intensity
4   require
5     lamp_is_on: is_on = True
6 not (light_intensity = medium_intensity)
7   do
8
9     if light_intensity = Low_intensity then
10      light_intensity := Medium_intensity
11    elseif light_intensity = Medium_intensity then
12      light_intensity := Medium_intensity
13    elseif light_intensity = High_intensity then
14      light_intensity := Low_intensity
15    end
16  ensure
17    from_low_to_medium: old light_intensity = Low_intensity implies
18      light_intensity = Medium_intensity
19    from_medium_to_high: old light_intensity = Medium_intensity implies
20      light_intensity = High_intensity
21    from_high_to_low: old light_intensity = High_intensity implies
22      light_intensity = Low_intensity
23  end
24
25 adjust_light_ID_378
26   -- Adjust the light intensity

```

```

24     require
25         lamp_is_on: is_on = True
26 not ((light_intensity) < (high_intensity))
27     do
28
29         if light_intensity = Low_intensity then
30             light_intensity := Medium_intensity
31         elseif light_intensity = Medium_intensity then
32             light_intensity := Medium_intensity
33         elseif light_intensity = High_intensity then
34             light_intensity := Low_intensity
35         end
36     ensure
37         from_low_to_medium: old light_intensity = Low_intensity implies
38             light_intensity = Medium_intensity
39         from_medium_to_high: old light_intensity = Medium_intensity implies
40             light_intensity = High_intensity
41         from_high_to_low: old light_intensity = High_intensity implies
42             light_intensity = Low_intensity
43     end
44
45 adjust_light_ID_384
46     -- Adjust the light intensity
47     require
48         lamp_is_on: is_on = True
49 not ((light_intensity) > (low_intensity))
50     do
51
52         if light_intensity = Low_intensity then
53             light_intensity := Medium_intensity
54         elseif light_intensity = Medium_intensity then
55             light_intensity := Medium_intensity
56         elseif light_intensity = High_intensity then
57             light_intensity := Low_intensity
58         end
59     ensure
60         from_low_to_medium: old light_intensity = Low_intensity implies
61             light_intensity = Medium_intensity
62         from_medium_to_high: old light_intensity = Medium_intensity implies
63             light_intensity = High_intensity
64         from_high_to_low: old light_intensity = High_intensity implies
65             light_intensity = Low_intensity
66     end
67
68 adjust_light_ID_392
69     -- Adjust the light intensity
70     require
71         lamp_is_on: is_on = True
72 not (light_intensity = 75)
73     do

```

```
68
69     if light_intensity = Low_intensity then
70         light_intensity := Medium_intensity
71     elseif light_intensity = Medium_intensity then
72         light_intensity := Medium_intensity
73     elseif light_intensity = High_intensity then
74         light_intensity := Low_intensity
75     end
76 ensure
77     from_low_to_medium: old light_intensity = Low_intensity implies
78         light_intensity = Medium_intensity
79     from_medium_to_high: old light_intensity = Medium_intensity implies
80         light_intensity = High_intensity
81     from_high_to_low: old light_intensity = High_intensity implies
82         light_intensity = Low_intensity
83 end
```

2 Examples with loops

2.1 BINARY_SEARCH

The `BINARY_SEARCH` class, as shown below, implements the binary search algorithm, which aims to search a `value` in a sorted integer array by repeatedly dividing the search interval in half. Fig.46 shows the verification result of the class: the implementation is correct with respect to the specification. Based on the correct version, 6 variants of the class are derived and further discussed below.

```
1 class
2   BINARY_SEARCH
3   feature -- Binary search
4     binary_search(a: V_ARRAY [INTEGER]; value: INTEGER): INTEGER
5       -- Index of 'value' in 'a' using binary search. Return 0 if not found
6       .
7       -- https://en.wikipedia.org/wiki/Binary\_search\_algorithm#Iterative
8       note
9         status: impure
10        require
11          no_overflow: a.count < {INTEGER}.max_value
12          a_sorted: across 1|..| a.count as i all
13            across 1|..| a.count as j all
14              i ≤ j implies a.sequence[i] ≤ a.sequence[j] end end
15          a_size_limit: a.count > 0 and a.count ≤ 10
16          a_valid_bound: a.lower < a.upper and a.lower = 1
17        local
18          low, up, middle: INTEGER
19        do
20          from
21            low := a.lower
22            up := a.upper + 1
23            Result := a.lower - 1
24          invariant
25            low_and_up_range: a.lower ≤ low and low ≤ up and up ≤ a.upper +
26              1
27            valid_bound: a.lower < a.upper
28            result_range: Result = a.lower - 1 or a.lower ≤ Result and Result
29              ≤ a.upper
30            not_in_lower_part: across 1|..| (low - a.lower) as i all a.sequence[
31              i] < value end
32            not_in_upper_part: across (up - a.lower + 1) |..| a.sequence.count
33              as i all value < a.sequence[i] end
34            found: (Result ≥ a.lower and Result ≤ a.upper) implies (a.sequence
35              [Result - a.lower + 1] = value)
36          until
```

```

31         low ≥ up or Result ≥ a.lower
32     loop
33         middle := low + ((up - low) // 2)
34         if a[middle] < value then
35             low := middle + 1
36         elseif a[middle] > value then
37             up := middle
38         else
39             Result := middle
40         end
41     variant
42         (a.upper - Result) + (up - low)
43     end
44 ensure
45     present: a.sequence.has (value) = (Result ≥ a.lower and Result ≤ a.
46         upper)
47     not_present: not a.sequence.has (value) = (Result = a.lower - 1)
48     found_if_present: (Result ≥ a.lower and Result ≤ a.upper) implies (a.
49         sequence[Result - a.lower + 1] = value)
end
end

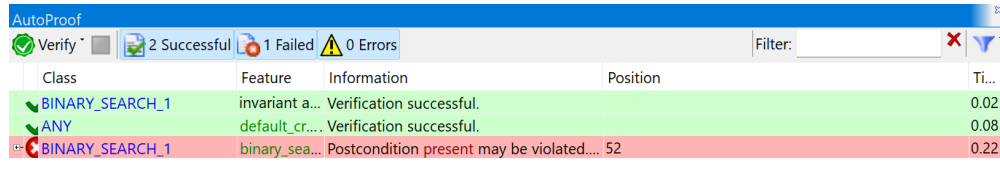
```

AutoProof				
<div> <div>Verify </div> <div>3 Successful </div> <div>0 Failed </div> <div>0 Errors </div> </div>				
	Filter:			
Class	Feature	Information	Position	Ti...
✓ BINARY_SEARCH	invariant admissibility	Verification successful.		0.25
✓ ANY	default_create (creator, inherited by BINARY_SEARCH)	Verification successful.		0.08
✓ BINARY_SEARCH	binary_search	Verification successful.		0.03

Fig. 46. Proof result of `BINARY_SEARCH` in AutoProof

Variant 1 of `BINARY_SEARCH`

- Fault injection: at line 30, remove the loop invariant `found`.
- Resulting failure: as shown in Fig. 47(a), the injected fault leads to the violation of the postcondition `present`.
- Cause of the failure: weakness/incompleteness of loop invariant.
- Proof time: 0.327 sec
- Comment: when trying to verify postcondition `present`, the prover uses the loop invariant, instead of the loop body, to represent the behaviors of the loop; if the loop invariant is not strong enough to express the functionality of the loop, as in this example, the prover is not able to establish the relevant postcondition; in this case, the counterexample (from which the test is extracted from) is not a real “counterexample” — it does not reveal the fault in the implementation and thus running the resulting test will not raise any exception; the passing test, however, indicates the weakness of the loop invariant.
- Possible fixes: No valid fixes



The screenshot shows the AutoProof application window. At the top, there's a status bar with icons for 'Verify', '2 Successful', '1 Failed', and '0 Errors'. Below this is a table with columns: Class, Feature, Information, Position, and Ti... (Time). The table contains three rows: 1. 'BINARY_SEARCH_1' with feature 'invariant a...' and information 'Verification successful.', position 0.02. 2. 'ANY' with feature 'default_cr...' and information 'Verification successful.', position 0.08. 3. 'BINARY_SEARCH_1' with feature 'binary_sea...' and information 'Postcondition present may be violated....', position 0.22. The third row is highlighted in red, indicating a failure.

Class	Feature	Information	Position	Ti...
BINARY_SEARCH_1	invariant a...	Verification successful.		0.02
ANY	default_cr...	Verification successful.		0.08
BINARY_SEARCH_1	binary_sea...	Postcondition present may be violated....	52	0.22

(a)

Variant 2 of `BINARY_SEARCH`

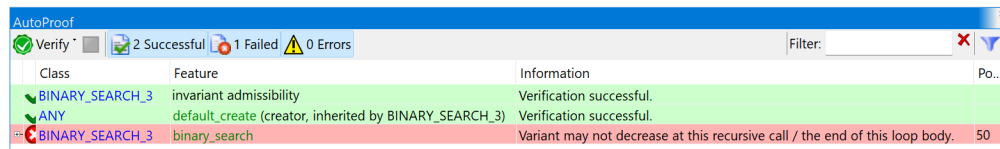
- Fault injection: at line 35, change the condition of the then branch from “`a[middle]<value`” into “`a[middle]≤value`”.
- Resulting failure: as shown in Fig. 47(b), the injected fault results in the violation of loop invariant `not_in_lower_part`.
- Cause of the failure: incorrect implementation of loop body.
- Proof time: 0.395 sec
- Comment: the test is useful as it reveals the fault in the program: the first element of the array has the same value as `value` (the value to search); when running the test, it is supposed that the loop ends at the first iteration — `value` is found at position 1 of the array; but due to the incorrect implementation of the routine, the iteration does not stop (the exit condition remains true after the first iteration) and continues for the second iteration, at which the loop invariant `not_in_lower_part` is evaluated to false (`a.sequence[1] = value`) and thus causes the exception.
- Possible fixes: No valid fixes

AutoProof			
<div> Verify 2 Successful 1 Failed 0 Errors Filter: </div>			
Class	Feature	Information	Position
BINARY_SEARCH_2	invariant admissibility	Verification successful.	
ANY	default_create (creator, inherited by BINARY_SEARCH_2)	Verification successful.	
BINARY_SEARCH_2	binary_search	Loop invariant <code>not_in_lower_part</code> may not be maintained.	35

(b)

Variant 3 of `BINARY_SEARCH`

- Fault injection: at line 36, change the assignment from “`low := middle + 1`” into “`low := middle`”.
- Resulting failure: as shown in Fig. 47(c), the injected fault results in the violation that the variant of the loop does not decrease.
- Cause of the failure: incorrect implementation of loop body.
- Proof time: 0.325 sec
- Comment: the test is useful as it reveals a bug in the program: initially, `low = 1` and `upper = 2`; at the first iteration, the program assigns `middle` with 1 (at line 34); the condition of the then branch at line 35 is evaluated true,
- Possible fixes: No valid fixes



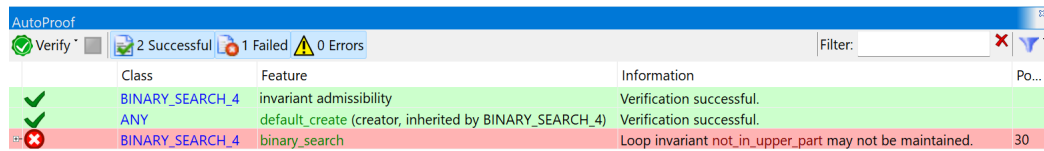
The screenshot shows the AutoProof application window. At the top, there is a status bar indicating '2 Successful' and '1 Failed' with a warning icon. Below this is a table with four columns: Class, Feature, Information, and Po... (likely Proof Obligations). The table contains three rows. The first two rows are green, indicating successful verification. The third row is red, indicating a failure.

Class	Feature	Information	Po...
BINARY_SEARCH_3	invariant admissibility	Verification successful.	
ANY	default_create (creator, inherited by BINARY_SEARCH_3)	Verification successful.	
BINARY_SEARCH_3	binary_search	Variant may not decrease at this recursive call / the end of this loop body.	50

(c)

Variant 4 of `BINARY_SEARCH`

- Fault injection: at line 37, change the condition of the `elseif` branch from “`a[middle]>value`” into “`a[middle]≥value`”.
- Resulting failure: as shown in Fig. 47(d), the loop invariant of `not_in_upper_part` is violated.
- Cause of the failure: incorrect implementation of loop body.
- Proof time: 0.288 sec
- Comment: this variant is similar to Variant 3; the test is useful as it shows a concrete trace that leads to the violation of the same contract in the failed proof.
- Possible fixes: No valid fixes



	Class	Feature	Information	Po...
✓	BINARY_SEARCH_4	invariant admissibility	Verification successful.	
✓	ANY	default_create (creator, inherited by BINARY_SEARCH_4)	Verification successful.	
✗	BINARY_SEARCH_4	binary_search	Loop invariant not_in_upper_part may not be maintained.	30

(d)

Variant 5 of `BINARY_SEARCH`








- Fault injection: at line 28, remove the loop invariant `not_in_lower_part`.
- Resulting failure: as shown in Fig. 47(e), the removal of the loop invariant causes the violation of postcondition of `present`.
- Cause of the failure: weakness/incompleteness of loop invariant.
- Proof time: 0.550 sec
- Comment: this variant is similar to Variant 1; the passing test indicates that the proof failure is caused by the weakness of the auxiliary specification (loop invariant), not the implementation.
- Possible fixes: No valid fixes

AutoProof				
Verify	2 Successful	1 Failed	0 Errors	Filter:
	Class	Feature	Information	Position
✓	BINARY_SEARCH_5	invariant admissibility	Verification successful.	
✓	ANY	default_create (creator, inherited by BINARY_SEARCH_5)	Verification successful.	
✗	BINARY_SEARCH_5	binary_search	Postcondition <code>present</code> may be violated.	53

(e)

Variant 6 of `BINARY_SEARCH`

- Fault injection: at line 21, change the loop initialization “`low := a.lower`” into “`low := a.lower + 1`”.
- Resulting failure: as shown in Fig. 47(f), the injected fault leads to the violation of the loop invariant `not_in_lower_part` at the entry of the loop (after loop initialization).
- Cause of the failure: incorrect implementation of loop initialization.
- Proof time: 0.356 sec
- Comment: the test is useful as its execution demonstrates a specific case where the program goes to a failure state, violating the same contract as in the proof failure; the values in the test input, however, is not that meaningful to this failure, as running the program with any valid input would cause the same contract violation.
- Possible fixes: No valid fixes

AutoProof				
 Verify	 2 Successful	 1 Failed	 0 Errors	Filter: <input type="text"/>
	Class	Feature	Information	Position
	<code>BINARY_SEARCH_6</code>	<code>binary_search</code>	Loop invariant <code>not_in_lower_part</code> may be violated on entry.	35
	<code>ANY</code>	<code>default_create (creator, in...</code>	Verification successful.	
	<code>BINARY_SEARCH_6</code>	<code>invariant admissibility</code>	Verification successful.	

(f)

2.2 LINEAR_SEARCH

The `LINEAR_SEARCH` class, which is displayed below, implements a function that returns the index of a given integer ‘value’ in an integer array ‘a’ using linear search starting from beginning of the array; if the ‘value’ is not found in ‘a’, the function returns the value “a.count + 1” (a.count represents the number of elements in a). Fig.47 shows the verification result of `LINEAR_SEARCH`, which indicates the complete correctness of its functionality. 4 variants of `LINEAR_SEARCH` are produced based on the correct version and are discussed below.

```
1  class
2      LINEAR_SEARCH
3  feature -- Basic operations
4      linear_search (a: SIMPLE_ARRAY [INTEGER]; value: INTEGER): INTEGER
5          require
6              array_not_empty: a.count > 0
7          do
8              from
9                  Result := 1
10             invariant
11                 result_in_bound: 1 ≤ Result and Result ≤ a.count + 1
12                 not_present_so_far: across 1.. (Result - 1) as i all a.sequence [i]
13                     ≠ value end
14             until
15                 Result = a.count + 1 or else a [Result] = value
16             loop
17                 Result := Result + 1
18             variant
19                 a.count - Result + 1
20             end
21         ensure
22             result_in_bound: 1 ≤ Result and Result ≤ a.count + 1
23             present: a.sequence.has (value) = (Result ≤ a.count)
24             found_if_present: (Result ≤ a.count) implies a.sequence [Result] =
25                 value
26             first_from_front: across 1.. (Result - 1) as i all a.sequence [i] ≠
27                 value end
28     end
29 end
```

Class	Feature	Information	Position	Ti...
✓ LINEAR_SEARCH_2	invariant ad...	Verification successful.		0.56
✓ ANY	default_creat...	Verification successful.		0.02
✓ LINEAR_SEARCH_2	linear_search	Verification successful.		0.07

Fig. 47. Proof result of `LINEAR_SEARCH` in AutoProof

Variant 1 of `LINEAR_SEARCH`

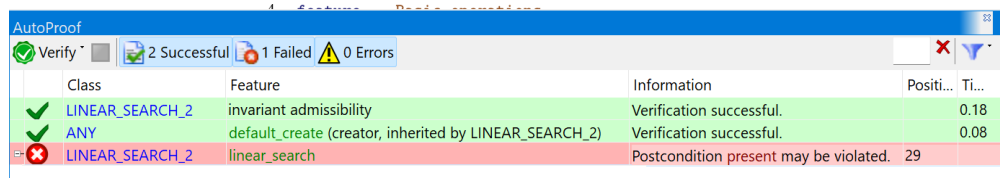
- Fault injection: at line 10, change the loop initialization from “`Result := 1`” into “`Result := 0`”.
- Resulting failure: as shown in Fig. 48(a), the injected fault leads to the violation of the loop invariant `result_in_bound` at the entry of the loop (after loop initialization).
- Cause of the failure: incorrect implementation of loop initialization.
- Proof time: 0.709 sec
- Comment: similar to the Variant 6 of `BINARY_SEARCH`, the test is useful as its execution demonstrates a specific case where the program goes to a failure state, violating the same contract as in the proof failure; the values in the test input, however, is not that meaningful to this failure, as running the program with any valid input would cause the same contract violation.
- Possible fixes: No valid fixes

Class	Feature	Information	P...	Ti...
✓ LINEAR_SEARCH_1	invariant admissibility	Verification successful.		0.61
✓ ANY	default_create (creator, inherited by LINEAR_SEARCH_1)	Verification successful.		0.08
✗ LINEAR_SEARCH_1	linear_search	Loop invariant <code>result_in_bound</code> may be violated on entry.	17	0.01

(a)

Variant 2 of `LINEAR_SEARCH`

- Fault injection: at line 12, change the left part of the exit condition from “`Result = a.count + 1`” into “`Result = a.count`”.
- Resulting failure: as shown in Fig. 48(b), the injected fault results in the violation of the postcondition `present`.
- Cause of the failure: incorrect exit condition (the condition for a loop to terminate).
- Proof time: 0.283 sec
- Comment: during the execution of the test, the program terminates after 1 iteration with `Result = 2`; this leads to the violation of the equality in the postcondition `present` — the left-hand part `a.sequence.has (value)` is false, as `value` does not match to any element of the input array `a`, while the right-hand part `Result ≤ a.count` is true (`Result = 2` and `a.count = 2`).
- Possible fixes: No valid fixes

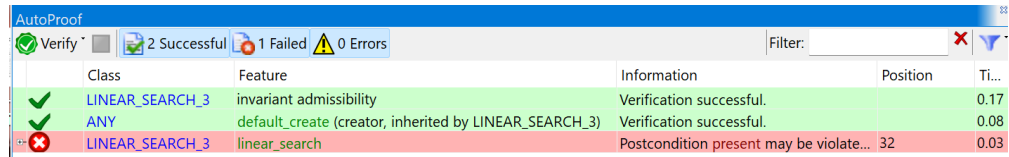


Class	Feature	Information	Positi...	Ti...
✓ LINEAR_SEARCH_2	invariant admissibility	Verification successful.		0.18
✓ ANY	default_create (creator, inherited by LINEAR_SEARCH_2)	Verification successful.		0.08
✗ LINEAR_SEARCH_2	linear_search	Postcondition <code>present</code> may be violated.	29	

(b)

Variant 3 of `LINEAR_SEARCH`

- Fault injection: at line 13, remove the loop invariant `not_present_so_far`.
- Resulting failure: as shown in Fig. 48(c), the injected fault causes the violation of the postcondition `present`.
- Cause of the failure: weakness/incompleteness of loop invariant.
- Proof time: 0.280 sec
- Comment: this variant is similar to Variant 5 of `BINARY_SEARCH`; the passing test indicates that the proof failure is caused by the weakness of the auxiliary specification (loop invariant), not the implementation.
- Possible fixes: No valid fixes

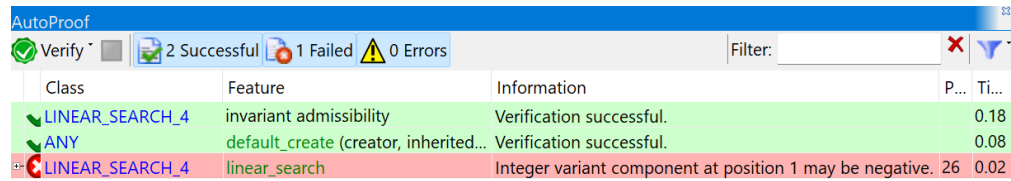


	Class	Feature	Information	Position	Ti...
✓	LINEAR_SEARCH_3	invariant admissibility	Verification successful.		0.17
✓	ANY	default_create (creator, inherited by LINEAR_SEARCH_3)	Verification successful.		0.08
✗	LINEAR_SEARCH_3	linear_search	Postcondition <code>present</code> may be violate...	32	0.03

(c)

Variant 4 of `LINEAR_SEARCH`

- Fault injection: change the loop variant at line 19 from “`a.count - Result + 1`” into “`a.count - Result - 1`”.
- Resulting failure: as shown in Fig. 48(d), the injected faults leads to the violation that “the integer variant component at iteration 1 may be negative”.
- Cause of the failure: incorrect loop variant.
- Proof time: 0.279 sec
- Comment: the test is useful as it is able to show how the value of variant varies at each iteration; the values in the test input, however, is not that meaningful, as any other valid test input will have the same effect.
- Possible fixes: No valid fixes



Class	Feature	Information	P...	Ti...
✓ <code>LINEAR_SEARCH_4</code>	invariant admissibility	Verification successful.		0.18
✓ <code>ANY</code>	default_create (creator, inherited...	Verification successful.		0.08
✗ <code>LINEAR_SEARCH_4</code>	linear_search	Integer variant component at position 1 may be negative.	26	0.02

(d)

2.3 MAX_IN_ARRAY

The `MAX_IN_ARRAY` class, as presented below, implements an algorithm that computes the maximum element of an integer array `a`. Fig.48 shows the verification result of the class, which suggests a complete functional correctness. 6 variants of the class are generated by injecting different faults in the correct version, which will be discussed below.

```
1  class
2      MAX_IN_ARRAY
3  feature -- Basic operations
4      max_in_array (a: SIMPLE_ARRAY [INTEGER]): INTEGER
5          -- Find the maximum element of 'a'.
6          require
7              array_not_empty: a.count > 0
8          local
9              i: INTEGER
10         do
11             Result := a [1]
12             from
13                 i := 2
14             invariant
15                 i_in_bounds: 2 ≤ i and i ≤ a.count + 1
16                 max_so_far: across 1|..| (i - 1) as c all a.sequence [c] ≤ Result end
17                 result_in_array: across 1|..| (i - 1) as c some a.sequence [c] = Result
18                     end
19             until
20                 i = a.count + 1
21             loop
22                 if a [i] > Result then
23                     Result := a [i]
24                 end
25                 i := i + 1
26                 variant
27                     a.count - i
28             end
29         ensure
30             is_maximum: across 1|..| a.count as c all a.sequence [c] ≤ Result end
31             result_in_array: across 1|..| a.count as c some a.sequence [c] = Result end
32     end
```

AutoProof				
Verify	3 Successful	0 Failed	0 Errors	Filter:
	Class	Feature	Information	Position
✓	MAX_IN...	invariant admissibility	Verification successful.	
✓	ANY	default_create (creator, inherited by MAX_IN_ARRAY)	Verification successful.	
✓	MAX_IN...	max_in_array	Verification successful.	

Fig. 48. Proof result of `MAX_IN_ARRAY` in AutoProof

Variant 1 of `MAX_IN_ARRAY`

- Fault injection: at line 22, change the condition of the then branch into “a [i] <Result”.
- Resulting failure: as shown in Fig. 49(a), the loop invariant `max_so_far` is violated during the iteration of the loop.
- Cause of the failure: incorrect implementation of the loop body.
- Proof time: 0.288 sec
- Comment: the test is useful as it is able to show a concrete trace that leads to the contract violation; the values in the test input, however, is not that meaningful, as any other valid test input will have the same effect.
- Possible fixes: No valid fixes

AutoProof				
Verify	2 Successful	1 Failed	0 Errors	Filter:
	Class	Feature	Information	Positi... Time
✓	MAX_IN...	invariant admissibility	Verification successful.	0.14
✓	ANY	default_create (creator, inherited by MAX_IN_ARRA...	Verification successful.	0.08
✗	MAX_IN...	max_in_array	Loop invariant max_so_far may not be maintained.	22 0.06

(a)

Variant 2 of `MAX_IN_ARRAY`

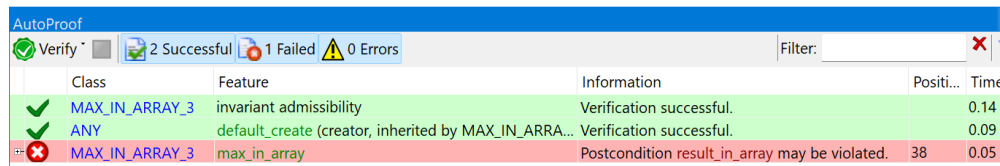
- Fault injection: at line 17, remove the loop invariant `max_so_far`.
- Resulting failure: as shown in Fig. 49(b), the removal of the loop invariant results in the violation of postcondition `is_maximum`.
- Cause of the failure: weakness/incompleteness of loop invariant.
- Proof time: 0.284 sec
- Comment: this variant is similar to Variant 5 of `BINARY_SEARCH`; the passing test indicates that the proof failure is caused by the weakness of the auxiliary specification (loop invariant), not the implementation.
- Possible fixes: No valid fixes

Class	Feature	Information	Positi...	Time [s]
✓ MAX_IN...	invariant admissibility	Verification successful.		0.15
✓ ANY	default_create (creator, inherited by MAX_IN_ARRA...	Verification successful.		0.08
✗ MAX_IN...	max_in_array	Postcondition is_maximum may be violated.	35	0.06

(b)

Variant 3 of MAX_IN_ARRAY

- Fault injection: at line 18, remove the loop invariant *result_in_array*.
- Resulting failure: as shown in Fig. 49(c), the removal of the loop invariant leads to the violation of the postcondition *result_in_array*.
- Cause of the failure: weakness/incompleteness of loop invariant.
- Proof time: 0.278 sec
- Comment: similar to Variant 2, the passing test indicates that the proof failure is caused by the weakness of the loop invariant.
- Possible fixes: No valid fixes



The screenshot shows the AutoProof interface with a table of verification results. The table has columns for Class, Feature, Information, Positi..., and Time. The status bar at the top indicates 2 Successful, 1 Failed, and 0 Errors. The table contains three rows: 'MAX_IN_ARRAY_3' with feature 'invariant admissibility' (successful, 0.14s), 'ANY' with feature 'default_create' (successful, 0.09s), and 'MAX_IN_ARRAY_3' with feature 'max_in_array' (failed, 0.05s, position 38). The failed row is highlighted in red.

Class	Feature	Information	Positi...	Time
MAX_IN_ARRAY_3	invariant admissibility	Verification successful.		0.14
ANY	default_create (creator, inherited by MAX_IN_ARRA...	Verification successful.		0.09
MAX_IN_ARRAY_3	max_in_array	Postcondition result_in_array may be violated.	38	0.05

(c)

Variant 4 of MAX_IN_ARRAY

- Fault injection: at line 20, change the exit condition of the loop from “ $i = \mathbf{a.count} + 1$ ” into “ $i \geq \mathbf{a.count}$ ”.
- Resulting failure: as shown in Fig. 49(d), the injected faults causes the violation of the postcondition *is_maximum*.
- Cause of the failure: incorrect exit condition of the loop.
- Proof time: 0.286 sec
- Comment: this test is useful as it shows a specific scenario from which the program will go to a failing state, violating the same failed contract in the proof; during the execution of the test, after going through 2 iterations, the program terminates with **Result** = 0; this reveals the program fault: the program terminates too early to reach the actual maximum value of **a** (the third element of **a**).
- Possible fixes: No valid fixes

AutoProof					
Verify		2 Successful		1 Failed	0 Errors
		Filter:			
Class	Feature	Information	Positi...	Time	
✓ MAX_IN_ARRAY_4	invariant admissibility	Verification successful.		0.13	
✓ ANY	default_create (creator, inherited by MAX_IN_ARRA...	Verification successful.		0.08	
✗ MAX_IN_ARRAY_4	max_in_array	Postcondition <i>is_maximum</i> may be violated.	38	0.07	

(d)

Variant 5 of MAX_IN_ARRAY

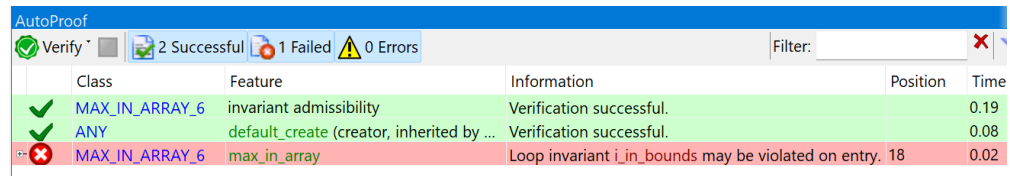
- Fault injection: at line 12, change the statement from “`Result := a [1]`” into “`Result := 0`”.
- Resulting failure: as shown in Fig. 49(e), the injected fault causes the violation of the loop invariant `max_so_far` on the entry.
- Cause of the failure: incorrect implementation in the code snippet before the loop.
- Proof time: 0.290 sec
- Comment: the test is useful as it is able to show a concrete trace that leads to the contract violation; the values in the test input, however, is not that meaningful, as any other valid test input will have the same effect.
- Possible fixes: No valid fixes

AutoProof					
Verify		2 Successful		1 Failed	
				0 Errors	
Filter: <input type="text"/>					
	Class	Feature	Information	Position	Time
✓	MAX_IN_ARRAY_5	invariant admissibility	Verification successful.		0.12
✓	ANY	default_create (creator, inherited by ...	Verification successful.		0.08
✗	MAX_IN_ARRAY_5	max_in_array	Loop invariant max_so_far may be violated on entry...	19	0.10

(e)

Variant 6 of MAX_IN_ARRAY

- Fault injection: at line 14, change the code from “i := 2” into “i := 1”.
- Resulting failure: as shown in Fig. 49(f), the injected fault causes the violation of the loop invariant `i_in_bounds` at the entry of the loop (after loop initialization).
- Cause of the failure: incorrect implementation of the loop initialization.
- Proof time: 0.282 sec
- Comment: similar to Variant 5, the test is useful as it is able to show a concrete trace that leads to the contract violation; the values in the test input, however, is not that meaningful, as any other valid test input will have the same effect.
- Possible fixes: No valid fixes



The screenshot shows the AutoProof interface with a status bar indicating '2 Successful' and '1 Failed' results. Below the status bar is a table with the following data:

	Class	Feature	Information	Position	Time
✓	MAX_IN_ARRAY_6	invariant admissibility	Verification successful.		0.19
✓	ANY	default_create (creator, inherited by ...	Verification successful.		0.08
✗	MAX_IN_ARRAY_6	max_in_array	Loop invariant i_in_bounds may be violated on entry.	18	0.02

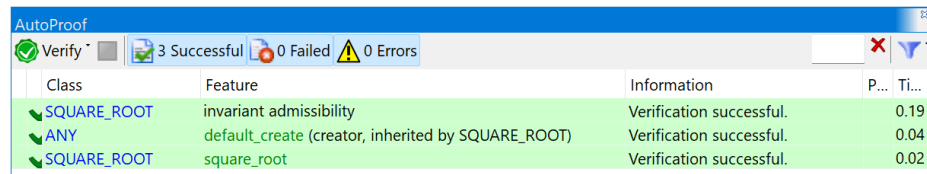
(f)

2.4 SQUARE_ROOT

The `SQUARE_ROOT` class, as shown below, calculates two approximate square roots x and y of a given positive integer n : the value of n falls between x^2 and y^2 ; if n is perfect square, then $y = x$ and $x^2 = n$, otherwise $y = x + 1$. Fig. 49 presents the verification result of the class, which indicates a full functional correctness. By injecting different faults in the correct version, 4 variants of `SQUARE_ROOT` are derived and discussed below.

```
1  class
2      SQUARE_ROOT
3  feature
4      square_root (n: INTEGER): TUPLE [x: INTEGER; y: INTEGER]
5          -- 'x' and 'y' are two approximate square roots of 'n'
6      require
7          valid_n: n ≥ 0
8      local
9          x1, x2, mid: INTEGER
10     do
11         from
12             x1 := 0
13             x2 := n
14         invariant
15             valid_result: (x1 = x2 and x1 * x1 = n) or (x1 < x2 and x1 * x1 < n
16                 and x2 * x2 ≥ n)
17         until
18             x2 - x1 ≤ 1 or x1 = x2
19         loop
20             mid := (x1 + x2) // 2 -- integer division
21             if mid * mid = n then
22                 x1 := mid
23                 x2 := mid
24             else
25                 if mid * mid < n then
26                     x1 := mid
27                 else
28                     x2 := mid
29                 end
30             end
31         variant
32             x2 - x1
33         end
34         Result := [x1, x2]
35     ensure
36         valid_result: (Result.x = Result.y and Result.x * Result.x = n)
37             or (Result.x + 1 = Result.y and Result.x * Result.x < n and Result.y
38                 * Result.y ≥ n)
```

```
37         end
38     end
```



The screenshot shows the AutoProof application window. At the top, there is a status bar with a green checkmark icon, the text 'Verify', and a summary: '3 Successful', '0 Failed', and '0 Errors'. Below this is a table with four columns: 'Class', 'Feature', 'Information', and 'P... Ti...'. The table contains three rows, all with a green checkmark icon in the first column. The first row is for 'SQUARE_ROOT' with feature 'invariant admissibility' and information 'Verification successful.' and a time of '0.19'. The second row is for 'ANY' with feature 'default_create (creator, inherited by SQUARE_ROOT)' and information 'Verification successful.' and a time of '0.04'. The third row is for 'SQUARE_ROOT' with feature 'square_root' and information 'Verification successful.' and a time of '0.02'.

Class	Feature	Information	P... Ti...
✓SQUARE_ROOT	invariant admissibility	Verification successful.	0.19
✓ANY	default_create (creator, inherited by SQUARE_ROOT)	Verification successful.	0.04
✓SQUARE_ROOT	square_root	Verification successful.	0.02

Fig. 49. Proof result of `SQUARE_ROOT` in AutoProof

Variant 1 of SQUARE_ROOT

- Fault injection: at line 17, change the left part of the exit condition from “ $x_2 - x_1 \leq 1$ ” into “ $x_2 - x_1 < 1$ ”.
- Resulting failure: as shown in Fig. 50(a), the injected fault leads to the failure that the loop variant is not decreased during the loop iteration.
- Cause of the failure: incorrect exit condition of the loop.
- Proof time: 0.258 sec
- Comment: the test is useful as it is able to show how the value of variant varies at each iteration.
- Possible fixes: 1 valid fixes out of 9 candidate fixes.
- fixing time: 1.58 minutes

Class	Feature	Information	P...	Ti...
SQUARE_ROOT_1	invariant admissibility	Verification successful.		0.16
ANY	default_create (creator, inherited by SQUARE_ROOT_1)	Verification successful.		0.05
SQUARE_ROOT_1	square_root	Variant may not decrease at this recursive call / the end of this loop body.	37	0.05

(a)

```

1
2 square_root_ID_3 (n: INTEGER): TUPLE [x: INTEGER; y: INTEGER]
3   -- 'x' and 'y' are two approximate square roots of 'n'
4   require
5     valid_n: n ≥ 0
6 not ((n) > (0))
7   local
8     x1, x2, mid: INTEGER
9   do
10
11     from
12       x1 := 0
13       x2 := n
14     invariant
15       valid_result: (x1 = x2 and x1 * x1 = n) or (x1 < x2 and x1 * x1 < n and x2
16         * x2 ≥ n)
17     until
18       x2 - x1 < 1 or x1 = x2
19   loop
20     mid := (x1 + x2) // 2 -- integer division
21     if mid * mid = n then
22       x1 := mid
23     else
24       if mid * mid < n then

```

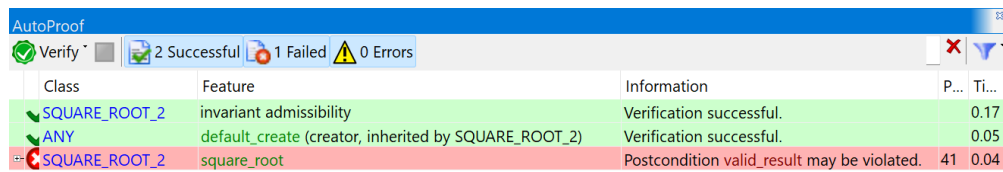
```

25         x1 := mid
26     else
27         x2 := mid
28     end
29 end
30 variant
31     x2 - x1
32 end
33 Result := [x1, x2]
34 ensure
35     valid_result: (Result.x = Result.y and Result.x * Result.x = n)
36     or (Result.x + 1 = Result.y and Result.x * Result.x < n and Result.y *
37         Result.y ≥ n)
end

```

Variant 2 of `SQUARE_ROOT`

- Fault injection: at line 15, remove the loop invariant `valid_result`.
- Resulting failure: as shown in Fig. 50(b), the injected fault leads to the violation of postcondition `valid_result`.
- Cause of the failure: weakness/incompleteness of loop invariant.
- Proof time: 0.257 sec
- Comment: similar to Variant 2 of `MAX_IN_ARRAY`, the passing test indicates that the proof failure is caused by the weakness of the loop invariant.
- Possible fixes: No valid fixes



Class	Feature	Information	P...	Ti...
✓ <code>SQUARE_ROOT_2</code>	invariant admissibility	Verification successful.		0.17
✓ <code>ANY</code>	default_create (creator, inherited by <code>SQUARE_ROOT_2</code>)	Verification successful.		0.05
✗ <code>SQUARE_ROOT_2</code>	square_root	Postcondition <code>valid_result</code> may be violated.	41	0.04

(b)

Variant 3 of SQUARE_ROOT

- Fault injection: change the condition of the then branch at line 20 from “ $\text{mid} * \text{mid} = n$ ” into “ $\text{mid} * \text{mid} \neq n$ ”.
- Resulting failure: as shown in Fig. 50(c), the injected fault results in the violation of the loop invariant `result_so_far`.
- Cause of the failure: incorrect implementation of the loop body.
- Proof time: 0.308 sec
- Comment: the test is useful as it is able to show a concrete trace that leads to the same contract violation as in the proof; during the execution of the test, initially, $x_1=0$ and $x_2=3$; at the first iteration, the program assigns `mid` with 1 (line 19); the condition of the then branch is true ($\text{mid} * \text{mid} \neq n$) and the program assigns both `x1` and `x2` with the value of `mid` (line 21 and 22); at the beginning of the second iteration, the loop invariant `valid_result` is evaluated as false ($x_1=x_2$ is true but $x_1*x_1=n$ is false).
- Possible fixes: 1 valid fixes out of 9 candidate fixes.
- fixing time: 1.68 minutes

AutoProof				
Verify		2 Successful	1 Failed	0 Errors
Class	Feature	Information	P...	Ti...
SQUARE_ROOT_3	invariant admissibility	Verification successful.	0.20	
ANY	default_create (creator, inherited by SQUARE_ROOT_3)	Verification successful.	0.05	
SQUARE_ROOT_3	square_root	Loop invariant <code>valid_result</code> may not be maintained.	16	0.06

(c)

```

1
2 square_root_ID_3 (n: INTEGER): TUPLE [x: INTEGER; y: INTEGER]
3   -- 'x' and 'y' are two approximate square roots of 'n'
4   require
5     valid_n: n ≥ 0
6 not ((n) > (0))
7   local
8     x1, x2, mid: INTEGER
9   do
10
11     from
12       x1 := 0
13       x2 := n
14     invariant
15       valid_result: (x1 = x2 and x1 * x1 = n) or (x1 < x2 and x1 * x1 < n and x2
16         * x2 ≥ n)
17     until
18       x2 - x1 ≤ 1 or x1 = x2
19   loop
20     mid := (x1 + x2) // 2 -- integer division

```



```

20         if mid * mid  $\neq$  n then
21             x1 := mid
22             x2 := mid
23         else
24             if mid * mid < n then
25                 x1 := mid
26             else
27                 x2 := mid
28             end
29         end
30         variant
31             x2 - x1
32         end
33         Result := [x1, x2]
34     ensure
35         valid_result: (Result.x = Result.y and Result.x * Result.x = n)
36         or (Result.x + 1 = Result.y and Result.x * Result.x < n and Result.y *
37             Result.y  $\geq$  n)
38     end

```

Variant 4 of SQUARE_ROOT

- Fault injection: at line 24, change the condition of the then branch from “ $\text{mid} * \text{mid} < n$ ” into “ $\text{mid} * \text{mid} > n$ ”.
- Resulting failure: as shown in Fig. 50(d), the loop invariant `valid_result` is not satisfied during the loop iteration.
- Cause of the failure: incorrect implementation of the loop body.
- Proof time: 0.400 sec
- Resulting test case: Fig. ?? shows the test case from Proof2Test, which calls `square_root` with input argument `n = 11`.
- Comment: the test is useful as it is able to show a concrete trace that leads to the same contract violation as in the proof; during the execution of the test, initially, `x1=0` and `x2=11`; at the first iteration, the program assigns `mid` with 5 (line 19); the condition of the `else` branch is true ($\text{mid} * \text{mid} \neq n$); the condition of the `then` branch at line 24 is true ($\text{mid} * \text{mid} > n$), thus the program assigns `x1` with the value of `mid` (line 25); at the beginning of the second iteration, the loop invariant `valid_result` is evaluated as false (in the right-hand side of the `or` clause, $x1 < x2$ is true but $x1 * x1 < n$ is false), causing an exception of the invariant violation.
- Possible fixes: 1 valid fixes out of 9 candidate fixes
- fixing time: 1.43 minutes

Class	Feature	Information	P...	Ti...
✓ SQUARE_ROOT_4	invariant admissibility	Verification successful.	0.20	
✓ ANY	default_create (creator, inherited by SQUARE_ROOT_4)	Verification successful.	0.05	
✗ SQUARE_ROOT_4	square_root	Loop invariant <code>valid_result</code> may not be maintained.21	0.16	

(d)

```

1
2 square_root_ID_3 (n: INTEGER): TUPLE [x: INTEGER; y: INTEGER]
3   -- 'x' and 'y' are two approximate square roots of 'n'
4   require
5     valid_n: n ≥ 0
6 not ((n) > (0))
7   local
8     x1, x2, mid: INTEGER
9   do
10
11     from
12       x1 := 0
13       x2 := n
14     invariant

```

```

15         valid_result: (x1 = x2 and x1 * x1 = n) or (x1 < x2 and x1 * x1 < n and x2
16             * x2 ≥ n)
17     until
18         x2 - x1 ≤ 1 or x1 = x2
19     loop
20         mid := (x1 + x2) // 2 -- integer division
21         if mid * mid = n then
22             x1 := mid
23             x2 := mid
24         else
25             if mid * mid > n then
26                 x1 := mid
27             else
28                 x2 := mid
29             end
30         end
31     variant
32         x2 - x1
33     end
34     Result := [x1, x2]
35     ensure
36         valid_result: (Result.x = Result.y and Result.x * Result.x = n)
37         or (Result.x + 1 = Result.y and Result.x * Result.x < n and Result.y *
            Result.y ≥ n)
38     end

```

3 Examples of EiffelBase2

This section describes the experiment of porting existing bugs in the “base” library to the fully-verified library “base2”. First, for classes present in “base2” we found the corresponding classes in “base” (Table 1). For each “base” class we ran a 1-hour-long session of AutoTest (running on an Intel i7-7700HQ 3.60GHz processor). This produced 285 failing tests and 79 unresolved tests. We manually analyzed each of 285 failing tests and managed to extract and port 12 faults from “base” to “base2”. The section will continue with the detailed descriptions of each fault.

3.1 `ARRAY.index_set` for empty arrays

Fig. 50 lists the code of a faulty function in the `ARRAY` class. The `ARRAY` class allows setting arbitrary boundaries (`lower` and `upper`) and empty arrays are represented simply by setting `lower = upper + 1`. All of the following (`lower`, `upper`) pairs are valid and represent an empty array: (1, 0), (10, 9), (-2, -3). On the other hand, the creation procedure `INTEGER_INTERVAL.make` sets specific values (1, 0) for the empty interval case (Fig. 51). This implementation is inconsistent with the `same_bounds` postcondition of `ARRAY.index_set`. Calling `ARRAY.index_set` for empty arrays not having `lower = 1` will result in the violation of `same_bounds`.

Table 1. Corresponding classes of “base” library in “base2”

base	base2
<code>ARRAY</code>	<code>V_ARRAY</code>
<code>INDEXABLE_ITERATION_CURSOR</code>	<code>V_ARRAY_ITERATOR</code>
<code>ARRAYED_LIST</code>	<code>V_ARRAYED_LIST</code>
<code>ARRAYED_LIST_CURSOR</code>	<code>V_ARRAYED_LIST_CURSOR</code>
<code>ARRAY2</code>	<code>V_ARRAY2</code>
<code>CELL</code>	<code>V_CELL</code>
<code>LINKABLE</code>	<code>V_LINKABLE</code>
<code>LINKED_LIST</code>	<code>V_LINKED_LIST</code>
<code>LINKED_LIST_ITERATION_CURSOR</code>	<code>V_LINKED_LIST_ITERATOR</code>
<code>LINKED_STACK</code>	<code>V_LINKED_STACK</code>
<code>LINKED_QUEUE</code>	<code>V_LINKED_QUEUE</code>
<code>HASH_TABLE</code>	<code>V_HASH_TABLE</code>
<code>HASH_TABLE_ITERATION_CURSOR</code>	<code>V_HASH_TABLE_ITERATOR</code>
<code>RANDOM</code>	<code>V_RANDOM</code>

```

1  index_set: INTEGER_INTERVAL
2      -- Range of acceptable indexes
3  do
4      create Result.make (lower, upper)
5  ensure then
6      same_count: Result.count = count
7      same_bounds: (Result.lower = lower) and (Result.upper = upper)
8  end

```

Fig. 50. Function `ARRAY.index_set` from “base”

```

1  make (min_index, max_index: INTEGER)
2      -- Set up interval to have bounds 'min_index' and
3      -- 'max_index' (empty if 'min_index' > 'max_index')
4  do
5      lower_defined := True
6      upper_defined := True
7      if min_index ≤ max_index then
8          lower_internal := min_index
9          upper_internal := max_index
10     else
11         lower_internal := 1
12         upper_internal := 0
13     end
14 ensure
15     lower_defined: lower_defined
16     upper_defined: upper_defined
17     set_if_non_empty: (min_index ≤ max_index) implies
18         ((lower = min_index) and (upper = max_index))
19     empty_if_not_in_order: (min_index > max_index) implies is_empty
20 end

```

Fig. 51. Creation procedure `INTEGER_INTERVAL.make` from “base”

The function was ported to “base2” literally by copying `index_set` into the body of `V_ARRAY`. To fix the bug and verify the function we relaxed the postcondition to specify the specific bounds only in the non-empty cases. To fully verify the fix we added missing contracts to the `V_ARRAY.count` function and implemented a verified but simplified version of the `INTEGER_INTERVAL` class (called `V_INTEGER_INTERVAL`). Other fixes can include the strengthening of the contracts to highlight that in

```

1  index_set: V_INTEGER_INTERVAL
2  do
3      create Result.make (lower, upper)
4  ensure
5      not_void: Result ≠ Void
6      same_count_if_not_empty: Result.count = count
7      same_bounds_if_not_empty: not is_empty
8          implies ((Result.lower = lower) and (Result.upper = upper))
9  end

```

Fig. 52. Fixed `index_set` injected into “base2”

case of an empty array the integer interval will have `lower = 1` and `upper = 0`.

This bug could be a result of a simple programmer mistake. Different classes have similar concepts (`lower` and `upper`) which are implemented slightly differently.

- fixing time: 2.42 minutes
- valid fixes out of 267 candidate fixes.

```
1
2  bugged_ID_258: INTEGER_INTERVAL_1
3      -- Range of acceptable indexes
4  note
5      original_name: index_set
6  require
7      -- not is_empty
8      -- not (lower > upper)
9      lower ≥ -10 and lower ≤ 10
10     upper ≥ -10 and upper ≤ 10
11 not ((lower) > (upper))
12 do
13
14     create Result.make (lower, upper)
15 ensure -- from READABLE_INDEXABLE
16 not_void: Result ≠ Void
17 same_count: Result.count = count -- or Result = 0
18 same_bounds: ((Result.lower = lower) and (Result.upper = upper)) -- bug
19     is here
20     -- Empty 'INTEGER_INTERVAL' always has 'lower = 1' and 'upper = 0'
21     -- while this is not necessary so for empty 'ARRAY'
22     -- Examples:
23     -- lower | upper
24     -- =====
25     -- -100 | -101
26 end
27 bugged_ID_259: INTEGER_INTERVAL_1
28     -- Range of acceptable indexes
29 note
30     original_name: index_set
31 require
32     -- not is_empty
33     -- not (lower > upper)
34     lower ≥ -10 and lower ≤ 10
35     upper ≥ -10 and upper ≤ 10
36 do
37
38     create Result.make (lower, upper)
39 ensure -- from READABLE_INDEXABLE
40 not_void: Result ≠ Void
41 same_count: Result.count = count -- or Result = 0
42 same_bounds: not ((lower) > (upper)) implies ( ((Result.lower = lower) and (
43     Result.upper = upper))) -- bug is here
```

```

43         -- Empty 'INTEGER_INTERVAL' always has 'lower = 1' and 'upper = 0'
44         -- while this is not necessary so for empty 'ARRAY'
45         -- Examples:
46         -- lower | upper
47         -- =====
48         -- -100 | -101
49     end
50
51 bugged_ID_260: INTEGER_INTERVAL_1
52     -- Range of acceptable indexes
53     note
54         original_name: index_set
55     require
56         -- not is_empty
57         -- not (lower > upper)
58         lower ≥ -10 and lower ≤ 10
59         upper ≥ -10 and upper ≤ 10
60 not ((lower) ≥ (upper))
61 do
62
63     create Result.make (lower, upper)
64     ensure -- from READABLE_INDEXABLE
65         not_void: Result ≠ Void
66         same_count: Result.count = count -- or Result = 0
67         same_bounds: ((Result.lower = lower) and (Result.upper = upper)) -- bug
68             is here
69         -- Empty 'INTEGER_INTERVAL' always has 'lower = 1' and 'upper = 0'
70         -- while this is not necessary so for empty 'ARRAY'
71         -- Examples:
72         -- lower | upper
73         -- =====
74         -- -100 | -101
75     end
76
77 bugged_ID_261: INTEGER_INTERVAL_1
78     -- Range of acceptable indexes
79     note
80         original_name: index_set
81     require
82         -- not is_empty
83         -- not (lower > upper)
84         lower ≥ -10 and lower ≤ 10
85         upper ≥ -10 and upper ≤ 10
86 do
87
88     create Result.make (lower, upper)
89     ensure -- from READABLE_INDEXABLE
90         not_void: Result ≠ Void
91         same_count: Result.count = count -- or Result = 0

```

```

91  same_bounds: not ((lower) ≥ (upper)) implies ((Result.lower = lower) and (
    Result.upper = upper))) -- bug is here
92      -- Empty 'INTEGER_INTERVAL' always has 'lower = 1' and 'upper = 0'
93      -- while this is not necessary so for empty 'ARRAY'
94      -- Examples:
95      -- lower | upper
96      -- =====
97      -- -100 | -101
98  end
99
100 bugged_ID_266: INTEGER_INTERVAL_1
101     -- Range of acceptable indexes
102     note
103         original_name: index_set
104     require
105         -- not is_empty
106         -- not (lower > upper)
107         lower ≥ -10 and lower ≤ 10
108         upper ≥ -10 and upper ≤ 10
109     not (lower > upper)
110     do
111
112         create Result.make (lower, upper)
113     ensure -- from READABLE_INDEXABLE
114         not_void: Result ≠ Void
115         same_count: Result.count = count -- or Result = 0
116         same_bounds: ((Result.lower = lower) and (Result.upper = upper)) -- bug
            is here
117             -- Empty 'INTEGER_INTERVAL' always has 'lower = 1' and 'upper = 0'
118             -- while this is not necessary so for empty 'ARRAY'
119             -- Examples:
120             -- lower | upper
121             -- =====
122             -- -100 | -101
123     end
124
125 bugged_ID_267: INTEGER_INTERVAL_1
126     -- Range of acceptable indexes
127     note
128         original_name: index_set
129     require
130         -- not is_empty
131         -- not (lower > upper)
132         lower ≥ -10 and lower ≤ 10
133         upper ≥ -10 and upper ≤ 10
134     do
135
136         create Result.make (lower, upper)
137     ensure -- from READABLE_INDEXABLE
138         not_void: Result ≠ Void

```



```

139     same_count: Result.count = count -- or Result = 0
140 same_bounds: not (lower > upper) implies ( ((Result.lower = lower) and (Result.
    upper = upper))) -- bug is here
141     -- Empty 'INTEGER_INTERVAL' always has 'lower = 1' and 'upper = 0'
142     -- while this is not necessary so for empty 'ARRAY'
143     -- Examples:
144     -- lower | upper
145     -- =====
146     -- -100 | -101
147 end
148 --
-----

```

3.2 `ARRAYED_LIST.merge_right` with shared `area_v2` wipes out both lists

Fig. 53 presents a manually composed test which highlights the failure found by AutoTest in `ARRAYED_LIST.merge_right`. The execution ends with the violation of

```

1  test
2    local
3      special: SPECIAL [INTEGER]
4      array: ARRAY [INTEGER]
5      a, b: ARRAYED_LIST [INTEGER]
6    do
7      create special.make_filled (0, 10)
8      special.keep_head (5)
9      -- special now has: count = 5, capacity = 10
10     create array.make_from_special (special)
11     create a.make_from_array (array)
12     create b.make_from_array (array)
13     a.merge_right (b)
14   end

```

Fig. 53. Failing test for `ARRAYED_LIST.merge_right` in “base2”

the `new_count` postcondition in the `merge_right` procedure (Fig. 54). In this procedure, when initially `area_v2 = other.area_v2` (both are references to the same object) and `area`’s capacity is enough to include both lists ($1_new_count \leq area_v2.capacity$), then the command `other.wipe_out` empties the same shared area, so the resulting count is just 0. The provided example test (Fig. 53) satisfies these conditions. This procedure can be ported to a simplified version of `V_ARRAYED_LIST` called `SIMPLE_ARRAYED_LIST` (Fig. 55). Running verification on this procedure reports that postcondition `new_count` may be violated. To explore the code further, notice that removing the `other.wipe_out` line allows the postcondition to verify

```

1  merge_right (other: ARRAYED_LIST [G])
2      -- Merge 'other' into current structure after cursor.
3  require -- from DYNAMIC_CHAIN
4      extendible: extendible
5      not_after: not after
6      other_exists: other ≠ Void
7      not_current: other ≠ Current
8  local
9      l_new_count, l_old_count: INTEGER_32
10 do
11     if not other.is_empty then
12         l_old_count := count
13         l_new_count := l_old_count + other.count
14         if l_new_count > area_v2.capacity then
15             area_v2 := area_v2.aliased_resized_area (l_new_count)
16         end
17         area_v2.insert_data (other.area_v2, 0, index, other.count)
18         other.wipe_out
19     end
20 ensure -- from DYNAMIC_CHAIN
21     new_count: count = old count + old other.count
22     same_index: index = old index
23     other_is_empty: other.is_empty
24 end

```

Fig. 54. Flat view of `ARRAYED_LIST.merge_right` from “base”

```

1  bugged (other: like Current; index: INTEGER)
2      note
3          explicit: wrapping
4      require
5          different_other: Current  $\neq$  other
6          both_wrapped: area.is_wrapped and other.area.is_wrapped
7          not_after:  $0 \leq \text{index}$  and  $\text{index} \leq \text{area.count}$ 
8      local
9          l_new_count: INTEGER
10         count, other_count: INTEGER
11     do
12         count := area.count
13         other_count := other.area.count
14         l_new_count := count + other_count
15     unwrap
16     if l_new_count > count then
17         area := area.aliased_resized_area_with_default2 ({G}.default, l_new_count)
18     end
19     area.move_data (index, index + other_count, count - index)
20     area.copy_data (other.area, 0, index, other_count)
21     sequence := area.sequence
22     wrap
23     other.wipe_out
24     ensure
25         modify (area)
26         modify (Current)
27         modify (other)
28         modify (other.area)
29         new_count: area.count = old area.count + old other.area.count
30         other_is_empty: other.area.count = 0
31     end

```

Fig. 55. Procedure SIMPLE_ARRAYED_LIST.merge_right using classes from “base2”

but breaks the `other_is_empty` postcondition. There can be several fixes. One straightforward fix is to specify explicitly that the `areas` should be different (Fig. 56). Adding the additional `different_areas` precondition allows the fea-

```

1  fixed_1 (other: like Current; index: INTEGER)
2      note
3          explicit: wrapping
4      require
5          different_other: Current  $\neq$  other
6          both_wrapped: area.is_wrapped and other.area.is_wrapped
7          not_after:  $0 \leq \text{index}$  and  $\text{index} \leq \text{area.count}$ 
8          different_areas: area  $\neq$  other.area
9      -- Same lines omitted
10     end

```

Fig. 56. The first fix for `SIMPLE_ARRAYED_LIST.merge_right`

ture to fully verify. Another fix is to properly use the ownership mechanism of AutoProof. For simplicity, now we will specify ownership in the precondition (Fig. 57). Notice that the old precondition `both_wrapped` is now not present.

```

1  fixed_2 (other: like Current; index: INTEGER)
2      note
3          explicit: wrapping
4      require
5          different_other: Current  $\neq$  other
6          not_after:  $0 \leq \text{index}$  and  $\text{index} \leq \text{area.count}$ 
7          current_owns: owns  $\sim$  create {MML_SET[ANY]}.singleton (area)
8          other_owns: other.owns  $\sim$  create {MML_SET[ANY]}.singleton (other.area)
9      -- Same lines omitted
10     end

```

Fig. 57. The second fix for `SIMPLE_ARRAYED_LIST.merge_right`

Since lists (`Current` and `other`) own their `areas`, each of `areas` have an owner. Object cannot be both `wrapped` and owned by a `closed` object. Since objects can have only one `owner`, it can be inferred that `areas` are different, so the two new preconditions `current_owns` and `other_owns` are imply `area \neq other.area` which allows the procedure to fully verify. The `owns` set is constant in this class, so it is more convenient to specify ownership in the class invariant (Fig. 58). In the third fix we move the ownership preconditions to the class invariant `owns_def`. Note that `sequence_equal` invariant is not specific to the fix and was present in all other examples. The bug probably was introduced simple because the pro-

```

1   fixed_3 (other: like Current; index: INTEGER)
2     note
3       explicit: wrapping
4     require
5       different_other: Current  $\neq$  other
6       not_after:  $0 \leq \text{index}$  and  $\text{index} \leq \text{area.count}$ 
7       -- Same lines omitted
8     end
9   invariant
10  owns_def: owns = create {MML_SET[ANY]}.singleton (area)
11  sequence_equal: sequence ~ area.sequence

```

Fig. 58. The third fix for SIMPLE_ARRAYED_LIST.merge_right

grammer simply overlooked this rare case. It is easy to overlook the possible shared usage of `areas` in “base” classes since not all features clearly specify if their contents are copied into new instances or if they are fully reused (as in the `ARRAY.make_from_special`). However, AutoProof’s ownership mechanism forces programmers to make this explicit in contracts.

- fixing time: 9.38 minutes
- fixes

```

1
2   bugged_ID_87 (other: like Current; index: INTEGER)
3     note
4       explicit: wrapping
5     require
6       different_other: Current  $\neq$  other
7       area.is_wrapped and other.area.is_wrapped
8       not_after:  $0 \leq \text{index}$  and  $\text{index} \leq \text{area.count}$ 
9     local
10      l_new_count: INTEGER
11      count, other_count: INTEGER
12    do
13
14      count := area.count
15      other_count := other.area.count
16      l_new_count := count + other_count
17      unwrap
18      if l_new_count > count then
19        area := area.aliased_resized_area_with_default2 ({INTEGER}.default,
20          l_new_count)
21      end
22      area.move_data (index, index + other_count, count - index)
23      area.copy_data (other.area, 0, index, other_count)
24      sequence := area.sequence
25      wrap

```

```

25     other.wipe_out
26   ensure
27     modify (area)
28     modify (Current)
29     modify (other)
30     modify (other.area)
31   new_count: not (other = Void) implies ( area.count = old area.count + old
      other.area.count)
32     other_is_empty: other.area.count = 0
33   end
34
35   bugged_ID_91 (other: like Current; index: INTEGER)
36     note
37       explicit: wrapping
38     require
39       different_other: Current ≠ other
40       area.is_wrapped and other.area.is_wrapped
41       not_after: 0 ≤ index and index ≤ area.count
42     local
43       l_new_count: INTEGER
44       count, other_count: INTEGER
45     do
46
47       count := area.count
48       other_count := other.area.count
49       l_new_count := count + other_count
50     unwrap
51     if l_new_count > count then
52       area := area.aliased_resized_area_with_default2 ({INTEGER}.default,
        l_new_count)
53     end
54     area.move_data (index, index + other_count, count - index)
55     area.copy_data (other.area, 0, index, other_count)
56     sequence := area.sequence
57     wrap
58     other.wipe_out
59   ensure
60     modify (area)
61     modify (Current)
62     modify (other)
63     modify (other.area)
64   new_count: not (other = Current) implies ( area.count = old area.count +
      old other.area.count)
65     other_is_empty: other.area.count = 0
66   end
67
68   bugged_ID_95 (other: like Current; index: INTEGER)
69     note
70       explicit: wrapping
71     require

```

```

72     different_other: Current ≠ other
73     area.is_wrapped and other.area.is_wrapped
74     not_after: 0 ≤ index and index ≤ area.count
75 local
76     l_new_count: INTEGER
77     count, other_count: INTEGER
78 do
79
80     count := area.count
81     other_count := other.area.count
82     l_new_count := count + other_count
83     unwrap
84     if l_new_count > count then
85         area := area.aliased_resized_area_with_default2 ({INTEGER}.default,
86             l_new_count)
87     end
88     area.move_data (index, index + other_count, count - index)
89     area.copy_data (other.area, 0, index, other_count)
90     sequence := area.sequence
91     wrap
92     other.wipe_out
93 ensure
94     modify (area)
95     modify (Current)
96     modify (other)
97     modify (other.area)
98 new_count: not ((other = Void) or else (not (other = Current))) implies ( area.
99     count = old area.count + old other.area.count)
100     other_is_empty: other.area.count = 0
101 end
102
103 bugged_ID_97 (other: like Current; index: INTEGER)
104 note
105     explicit: wrapping
106 require
107     different_other: Current ≠ other
108     area.is_wrapped and other.area.is_wrapped
109     not_after: 0 ≤ index and index ≤ area.count
110 local
111     l_new_count: INTEGER
112     count, other_count: INTEGER
113 do
114
115     count := area.count
116     other_count := other.area.count
117     l_new_count := count + other_count
118     unwrap
119     if l_new_count > count then
120         area := area.aliased_resized_area_with_default2 ({INTEGER}.default,
121             l_new_count)

```

```

119         end
120         area.move_data (index, index + other_count, count - index)
121         area.copy_data (other.area, 0, index, other_count)
122         sequence := area.sequence
123         wrap
124         other.wipe_out
125     ensure
126         modify (area)
127         modify (Current)
128         modify (other)
129         modify (other.area)
130     new_count: not ((not (other = Void)) or else (other = Current)) implies ( area.
        count = old area.count + old other.area.count)
131     other_is_empty: other.area.count = 0
132     end
133
134     bugged_ID_99 (other: like Current; index: INTEGER)
135     note
136         explicit: wrapping
137     require
138         different_other: Current ≠ other
139         area.is_wrapped and other.area.is_wrapped
140         not_after: 0 ≤ index and index ≤ area.count
141     local
142         l_new_count: INTEGER
143         count, other_count: INTEGER
144     do
145
146         count := area.count
147         other_count := other.area.count
148         l_new_count := count + other_count
149         unwrap
150         if l_new_count > count then
151             area := area.aliased_resized_area_with_default2 ({INTEGER}.default,
                l_new_count)
152         end
153         area.move_data (index, index + other_count, count - index)
154         area.copy_data (other.area, 0, index, other_count)
155         sequence := area.sequence
156         wrap
157         other.wipe_out
158     ensure
159         modify (area)
160         modify (Current)
161         modify (other)
162         modify (other.area)
163     new_count: not ((not (other = Void)) or else (not (other = Current))) implies (
        area.count = old area.count + old other.area.count)
164     other_is_empty: other.area.count = 0
165     end

```



```

166
167 bugged_ID_105 (other: like Current; index: INTEGER)
168     note
169         explicit: wrapping
170     require
171         different_other: Current  $\neq$  other
172         area.is_wrapped and other.area.is_wrapped
173         not_after:  $0 \leq \text{index}$  and  $\text{index} \leq \text{area.count}$ 
174     local
175         l_new_count: INTEGER
176         count, other_count: INTEGER
177     do
178
179         count := area.count
180         other_count := other.area.count
181         l_new_count := count + other_count
182     unwrap
183     if l_new_count > count then
184         area := area.aliased_resized_area_with_default2 ({INTEGER}.default,
185             l_new_count)
186     end
187     area.move_data (index, index + other_count, count - index)
188     area.copy_data (other.area, 0, index, other_count)
189     sequence := area.sequence
190     wrap
191     other.wipe_out
192 ensure
193     modify (area)
194     modify (Current)
195     modify (other)
196     modify (other.area)
197 new_count: not ((index)  $\geq$  0) implies ( area.count = old area.count + old other.
198     area.count)
199     other_is_empty: other.area.count = 0
200 end
201
202 bugged_ID_115 (other: like Current; index: INTEGER)
203     note
204         explicit: wrapping
205     require
206         different_other: Current  $\neq$  other
207         area.is_wrapped and other.area.is_wrapped
208         not_after:  $0 \leq \text{index}$  and  $\text{index} \leq \text{area.count}$ 
209     local
210         l_new_count: INTEGER
211         count, other_count: INTEGER
212     do
213
214         count := area.count
215         other_count := other.area.count

```

```

214     l_new_count := count + other_count
215     unwrap
216     if l_new_count > count then
217         area := area.aliased_resized_area_with_default2 ({INTEGER}.default,
                l_new_count)
218     end
219     area.move_data (index, index + other_count, count - index)
220     area.copy_data (other.area, 0, index, other_count)
221     sequence := area.sequence
222     wrap
223     other.wipe_out
224     ensure
225         modify (area)
226         modify (Current)
227         modify (other)
228         modify (other.area)
229     new_count: not (((((((((Current))).area))).sequence))).count = 0 implies ( area.
        count = old area.count + old other.area.count)
230         other_is_empty: other.area.count = 0
231     end
232
233     bugged_ID_117 (other: like Current; index: INTEGER)
234     note
235         explicit: wrapping
236     require
237         different_other: Current ≠ other
238         area.is_wrapped and other.area.is_wrapped
239         not_after: 0 ≤ index and index ≤ area.count
240     local
241         l_new_count: INTEGER
242         count, other_count: INTEGER
243     do
244
245         count := area.count
246         other_count := other.area.count
247         l_new_count := count + other_count
248         unwrap
249         if l_new_count > count then
250             area := area.aliased_resized_area_with_default2 ({INTEGER}.default,
                    l_new_count)
251         end
252         area.move_data (index, index + other_count, count - index)
253         area.copy_data (other.area, 0, index, other_count)
254         sequence := area.sequence
255         wrap
256         other.wipe_out
257     ensure
258         modify (area)
259         modify (Current)
260         modify (other)

```

```

261     modify (other.area)
262 new_count: not (((((((other))).sequence))).count = 0) implies ( area.count = old
    area.count + old other.area.count)
263     other_is_empty: other.area.count = 0
264 end
265
266 bugged_ID_121 (other: like Current; index: INTEGER)
267     note
268         explicit: wrapping
269     require
270         different_other: Current  $\neq$  other
271         area.is_wrapped and other.area.is_wrapped
272         not_after: 0  $\leq$  index and index  $\leq$  area.count
273     local
274         l_new_count: INTEGER
275         count, other_count: INTEGER
276     do
277
278         count := area.count
279         other_count := other.area.count
280         l_new_count := count + other_count
281         unwrap
282         if l_new_count > count then
283             area := area.aliased_resized_area_with_default2 ({INTEGER}.default,
                l_new_count)
284         end
285         area.move_data (index, index + other_count, count - index)
286         area.copy_data (other.area, 0, index, other_count)
287         sequence := area.sequence
288         wrap
289         other.wipe_out
290     ensure
291         modify (area)
292         modify (Current)
293         modify (other)
294         modify (other.area)
295 new_count: not (((((((((((other))).area))).sequence))).count = 0) implies ( area.
    count = old area.count + old other.area.count)
296     other_is_empty: other.area.count = 0
297 end

```

3.3 `ARRAY.force` below empty array

Fig. 59 presents a failing test for procedure `ARRAY.force`. Executing this test results in `ARRAY.force`'s inserted postcondition violation (Fig. 60). The bug is located inside the `if empty_area then` block. When forcing into an array with empty area, area needs to be grown and the new value needs to be written to the area at the specified index. The code grows the area and saves the new

```

1  array_force_below_test
2    local
3      array: ARRAY [INTEGER]
4    do
5      create array.make_filled (0, 10, 9)
6      array.force (1, 5)
7    end

```

Fig. 59. Failing test for `ARRAY.force`

value at the end of the area (`area.extend(v)`) but this is not always correct. If $i < \text{lower}$, the new value should be recorded not as the last element of `area` but as the first element of `area`. To simplify the discussion we extracted the relevant code into a special case procedure for “base2” (Fig. 61). Notice the added `is_empty` precondition which signifies the special case which is covered in this version. Verification fails for the same postcondition `inserted` which was failing in the test. The added broader postcondition `sequence_effect` is also failing to verify. To fix the bug we replace the `if not l_increased_by_one then` instruction to handle the problematic case of forcing the new value below the bounds of the array (Fig. 62). The fixed version fully verifies. The programmer might have simply overlooked the rare case of indices below `lower`.

3.4 `ARRAY.remove_head` postcondition when removing all items

Fig. 63 shows the procedure `ARRAY.remove_head`. Calling this procedure with $n > \text{count}$ leads to the violation of the `same_upper` postcondition. The postcondition states that `upper` should not change, however the `then` part of the `if n > count` construction changes `upper` (`upper := lower - 1`). Fig. 64 presents `remove_head` ported to “base2”. The instruction `upper := lower - 1` had to be changed into `lower := 1; upper := 0` to satisfy `V_ARRAY`’s invariant. Verification fails because the postcondition `same_upper` can be violated. The version from “base” can be fixed by replacing `upper := lower - 1` with `lower := upper + 1` which will satisfy the postcondition. However, this fix is not applicable in the case of “base2” because the class invariant requires `lower = 1` when the array is empty. The second fix is to improve the postcondition (Fig. 65). The replaced precondition `same_upper_if_not_empty` specifies that `upper` is not changed only if the resulting array is not empty. The feature verifies fully with this fix.

3.5 `INDEXABLE_ITERATION_CURSOR.forth` does not always increment `cursor_index`

Fig. 66 shows two connected features `forth` and `cursor_index`. AutoFix has found several counterexamples in which the postcondition `cursor_index_advanced` did not hold (Table 2). Indeed, with these values the postcondition does not hold and `cursor_index` does not advance. The fix (Fig. 67) is based on clearly specifying

```

1  force (v: like item; i: INTEGER_32)
2      -- Assign item 'v' to 'i'-th entry.
3      -- Resize the array if 'i' falls out of currently defined bounds; preserve
        existing items.
4      -- In void-safe mode, if ({G}).has_default does not hold, then you can only
        insert between
5      -- 'lower - 1' or 'upper + 1' position in the ARRAY.
6  require
7      has_default_if_too_low: (i < lower - 1 and lower ≠ {like lower}.min_value)
        implies ({G}).has_default
8      has_default_if_too_high: (i > upper + 1 and upper ≠ {like upper}.max_value)
        implies ({G}).has_default
9  local
10     old_size, new_size: INTEGER_32
11     new_lower, new_upper: INTEGER_32
12     l_count, l_offset: INTEGER_32
13     l_increased_by_one: BOOLEAN
14  do
15     new_lower := lower.min (i)
16     new_upper := upper.max (i)
17     new_size := new_upper - new_lower + 1
18     l_increased_by_one := (i = upper + 1) or (i = lower - 1)
19     if empty_area then
20         make_empty_area (new_size.max (additional_space))
21         if not l_increased_by_one then
22             area.fill_with (({G}).default, 0, new_size - 2)
23         end
24         area.extend (v)
25     else
26         ... -- irrelevant lines omitted
27     end
28     lower := new_lower
29     upper := new_upper
30  ensure
31     inserted: item (i) = v
32     higher_count: count ≥ old count
33     lower_set: lower = (old lower).min (i)
34     upper_set: upper = (old upper).max (i)
35  end

```

Fig. 60. Procedure `ARRAY.force` from “base”

```

1  force (v: like item; i: INTEGER)
2    require
3      is_empty
4    local
5      old_size, new_size: INTEGER_32
6      new_lower, new_upper: INTEGER_32
7      l_count, l_offset: INTEGER_32
8      l_increased_by_one: BOOLEAN
9    do
10     new_lower := lower.min (i)
11     new_upper := upper.max (i)
12     new_size := new_upper - new_lower + 1
13     l_increased_by_one := (i = upper + 1) or (i = lower - 1)
14     area := area.aliased_resized_area (new_size)
15     if not l_increased_by_one then
16       area.fill_with (({G}).default, 0, new_size - 2)
17     end
18     area.extend (v)
19     lower := new_lower
20     upper := new_upper
21   ensure
22     inserted: item (i) = v
23     higher_count: count ≥ old count
24     lower_set: lower = (old lower).min (i)
25     upper_set: upper = (old upper).max (i)
26     modify_model ([ "sequence", "lower_"], Current)
27   end

```

Fig. 61. Special case `ARRAY.force` ported to “base2”

Table 2. Counterexamples for `INDEXABLE_ITERATION_CURSOR.forth`

is_reversed	target_index	step	first_index
True	1	1	0
True	1	2	0
True	1	7	0
True	3	3	0
True	1	100	0

```

1  fixed (v: like item; i: INTEGER)
2    require -- Same contracts omitted
3    local
4      old_size, new_size: INTEGER_32
5      new_lower, new_upper: INTEGER_32
6      l_count, l_offset: INTEGER_32
7      l_increased_by_one: BOOLEAN
8    do
9      new_lower := lower.min (i)
10     new_upper := upper.max (i)
11     new_size := new_upper - new_lower + 1
12     l_increased_by_one := (i = upper + 1) or (i = lower - 1)
13     area := area.aliased_resized_area (new_size)
14     if not l_increased_by_one then
15       if i < lower then
16         area.extend (v)
17         area.fill_with (({G}).default, 1, new_size - 1)
18       else
19         area.fill_with (({G}).default, 0, new_size - 2)
20         area.extend (v)
21       end
22     else
23       area.extend (v)
24     end
25     lower := new_lower
26     upper := new_upper
27   ensure -- Same contracts omitted
28   end

```

Fig. 62. Special case `ARRAY.force` fixed in “base2”

```

1  remove_head (n: INTEGER_32)
2      -- Remove first 'n' items;
3      -- if 'n' > count, remove all.
4  require
5      n_non_negative: n ≥ 0
6  do
7      if n > count then
8          upper := lower - 1
9          area := area.aliased_resized_area (0)
10     else
11         keep_tail (count - n)
12     end
13 ensure
14     new_count: count = (old count - n).max (0)
15     same_upper: upper = old upper
16 end

```

Fig. 63. Original `ARRAY.remove_head` from “base”

```

1  bugged (n: INTEGER_32)
2      note
3      explicit: wrapping
4  require
5      n_non_negative: n ≥ 0
6  do
7      if n > count then
8          unwrap
9          lower := 1
10         upper := 0
11         area := area.aliased_resized_area (0)
12         wrap
13     else
14         keep_tail (count - n)
15     end
16 ensure
17     new_count: count = (old count - n).max (0)
18     same_upper: upper = old upper
19 end

```

Fig. 64. Buggy `ARRAY.remove_head` ported to “base2”


```

1  fixed (n: INTEGER_32)
2    -- Same lines omitted
3  ensure
4    new_count: count = (old count - n).max (0)
5    same_upper_if_not_empty: not is_empty implies upper = old upper
6  end

```

Fig. 65. Fixed `ARRAY.remove_head` ported to “base2”

```

1  forth
2    -- Move to next position.
3  require -- from ITERATION_CURSOR
4    valid_position: not after
5  do
6    if is_reversed then
7      target_index := target_index - step
8    else
9      target_index := target_index + step
10   end
11  ensure then
12    cursor_index_advanced: cursor_index = old cursor_index + 1
13  end
14  cursor_index: INTEGER_32
15    -- Index position of cursor in the iteration.
16  require
17    is_valid: is_valid
18  do
19    Result := ((target_index - first_index).abs + step - 1) // step + 1
20  ensure
21    positive_index: Result ≥ 0
22  end

```

Fig. 66. Original `forth` and `cursor_index` features from “base”
INDEXABLE_ITERATION_CURSOR

the relationship between `is_reversed` and `first_index` with `last_index` in the class invariant. With such invariant in place the other features can be correctly implemented for each case. The fix includes strengthening the class invariant and updating two features. After implementing the fixes both features verify.

3.6 indexable_iteration_cursor_bug_1.log

– fixing time: 2.43 minutes

```

1
2 buggy_ID_264
3     -- Move to next position.
4     require -- from ITERATION_CURSOR
5         valid_position: not (lower ≤ target_index and target_index ≤ upper) and
6             lower ≤ upper
7         -- not before
8         target_index ≥ 0 and target_index ≤ 10
9         lower ≥ 0 and lower ≤ 10
10        upper ≥ 0 and upper ≤ 10
11        first_index ≥ 0 and first_index ≤ 10
12    not ((is_reversed) or else (before))
13    do
14        if is_reversed then
15            target_index := target_index - step
16        else
17            target_index := target_index + step
18        end
19    ensure
20        -- cursor_index_advanced: cursor_index_buggy = old
21        cursor_index_buggy + 1
22        cursor_index_advanced: (cursor_index_fixed = old cursor_index_fixed +
23            1)
24    end
25
26 buggy_ID_265
27     -- Move to next position.
28     require -- from ITERATION_CURSOR
29         valid_position: not (lower ≤ target_index and target_index ≤ upper) and
30             lower ≤ upper
31         -- not before
32         target_index ≥ 0 and target_index ≤ 10
33         lower ≥ 0 and lower ≤ 10
34         upper ≥ 0 and upper ≤ 10
35         first_index ≥ 0 and first_index ≤ 10
36    do
37        if is_reversed then
38            target_index := target_index - step

```

```

1  fixed
2    note
3      explicit: wrapping
4    require
5      valid_position: not after
6    do
7      if before then
8        unwrap; target_index := first_index; wrap
9      else
10       if is_reversed then
11         unwrap; target_index := target_index - step; wrap
12       else
13         unwrap; target_index := target_index + step; wrap
14       end
15     end
16   ensure
17     modify_model ("target_index", Current)
18     cursor_index_advanced: cursor_index_fixed = old cursor_index_fixed + 1
19   end
20 cursor_index_fixed: INTEGER_32
21   note
22     status: pure
23   do
24     if is_reversed then
25       if target_index ≤ first_index then
26         Result := (first_index - target_index + step - 1) // step + 1
27       end
28     else
29       if target_index ≥ first_index then
30         Result := (target_index - first_index + step - 1) // step + 1
31       end
32     end
33   ensure
34     positive_index: Result ≥ 0
35     before_is_zero: before implies Result = 0
36     reversed_in: is_reversed and target_index ≤ first_index implies Result = (
37       first_index - target_index + step - 1) // step + 1
38     not_reversed_in: not is_reversed and target_index ≥ first_index implies Result
39       = (target_index - first_index + step - 1) // step + 1
40   end
41 invariant
42   first_greater_when_reversed: is_reversed implies first_index ≥ last_index
43   last_greater_when_not_reversed: not is_reversed implies last_index ≥ first_index

```

Fig. 67. Fixed forth and cursor_index features using “base2” classes

```

37     else
38         target_index := target_index + step
39     end
40 ensure
41     -- cursor_index_advanced: cursor_index_buggy = old
42     cursor_index_buggy + 1
43 cursor_index_advanced: not ((is_reversed) or else (before)) implies ( (
44     cursor_index_fixed = old cursor_index_fixed + 1))
45 end
46
47 buggy_ID_268
48     -- Move to next position.
49 require -- from ITERATION_CURSOR
50     valid_position: not (lower ≤ target_index and target_index ≤ upper) and
51     lower ≤ upper
52     -- not before
53     target_index ≥ 0 and target_index ≤ 10
54     lower ≥ 0 and lower ≤ 10
55     upper ≥ 0 and upper ≤ 10
56     first_index ≥ 0 and first_index ≤ 10
57 not ((not (is_reversed)) or else (before))
58 do
59
60     if is_reversed then
61         target_index := target_index - step
62     else
63         target_index := target_index + step
64     end
65 ensure
66     -- cursor_index_advanced: cursor_index_buggy = old
67     cursor_index_buggy + 1
68     cursor_index_advanced: (cursor_index_fixed = old cursor_index_fixed +
69     1)
70 end
71
72 buggy_ID_269
73     -- Move to next position.
74 require -- from ITERATION_CURSOR
75     valid_position: not (lower ≤ target_index and target_index ≤ upper) and
76     lower ≤ upper
77     -- not before
78     target_index ≥ 0 and target_index ≤ 10
79     lower ≥ 0 and lower ≤ 10
80     upper ≥ 0 and upper ≤ 10
81     first_index ≥ 0 and first_index ≤ 10
82 do
83
84     if is_reversed then
85         target_index := target_index - step
86     else

```

```

81         target_index := target_index + step
82     end
83     ensure
84         -- cursor_index_advanced: cursor_index_buggy = old
85         cursor_index_buggy + 1
86     cursor_index_advanced: not ((not (is_reversed)) or else (before)) implies ( (
87         cursor_index_fixed = old cursor_index_fixed + 1))
88     end
89     buggy_ID_273
90     -- Move to next position.
91     require -- from ITERATION_CURSOR
92     valid_position: not (lower ≤ target_index and target_index ≤ upper) and
93     lower ≤ upper
94     -- not before
95     target_index ≥ 0 and target_index ≤ 10
96     lower ≥ 0 and lower ≤ 10
97     upper ≥ 0 and upper ≤ 10
98     first_index ≥ 0 and first_index ≤ 10
99     do
100         if is_reversed then
101             target_index := target_index - step
102         else
103             target_index := target_index + step
104         end
105     ensure
106         -- cursor_index_advanced: cursor_index_buggy = old
107         cursor_index_buggy + 1
108     cursor_index_advanced: not (step = 1) implies ( (cursor_index_fixed = old
109     cursor_index_fixed + 1))
110     end
111     buggy_ID_278
112     -- Move to next position.
113     require -- from ITERATION_CURSOR
114     valid_position: not (lower ≤ target_index and target_index ≤ upper) and
115     lower ≤ upper
116     -- not before
117     target_index ≥ 0 and target_index ≤ 10
118     lower ≥ 0 and lower ≤ 10
119     upper ≥ 0 and upper ≤ 10
120     first_index ≥ 0 and first_index ≤ 10
121     not (target_index ≠ first_index)
122     do
123         if is_reversed then
124             target_index := target_index - step
125         else
126             target_index := target_index + step

```

```

125     end
126   ensure
127     -- cursor_index_advanced: cursor_index_buggy = old
128       cursor_index_buggy + 1
129   cursor_index_advanced: (cursor_index_fixed = old cursor_index_fixed +
130     1)
131 end
132
133 buggy_ID_279
134   -- Move to next position.
135   require -- from ITERATION_CURSOR
136     valid_position: not (lower ≤ target_index and target_index ≤ upper) and
137       lower ≤ upper
138     -- not before
139     target_index ≥ 0 and target_index ≤ 10
140     lower ≥ 0 and lower ≤ 10
141     upper ≥ 0 and upper ≤ 10
142     first_index ≥ 0 and first_index ≤ 10
143 do
144   if is_reversed then
145     target_index := target_index - step
146   else
147     target_index := target_index + step
148   end
149   ensure
150     -- cursor_index_advanced: cursor_index_buggy = old
151       cursor_index_buggy + 1
152   cursor_index_advanced: not (target_index ≠ first_index) implies ( (
153     cursor_index_fixed = old cursor_index_fixed + 1))
154 end
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999

```

3.7 linked_list_make_not_before_1.log

– fixing time: 3.13 minutes

```

1
2 buggy_ID_593
3   do
4
5       before := True
6   ensure
7       is_before: before
8 before_constraint: not (before) implies ( before implies (active =
       first_element))
9   end
10
11 buggy_ID_598
12 require
13 not ((not (before)) or else (not (active = first_element)))
14 do
15
16     before := True
17 ensure
18     is_before: before
19     before_constraint: before implies (active = first_element)
20 end
21
22 buggy_ID_599
23 do
24
25     before := True
26 ensure
27     is_before: before
28 before_constraint: not ((not (before)) or else (not (active = first_element)))
       implies ( before implies (active = first_element))
29 end
30
31 buggy_ID_602
32 require
33 not ((not (before)) or else (not (before implies (active = first_element))))
34 do
35
36     before := True
37 ensure
38     is_before: before
39     before_constraint: before implies (active = first_element)
40 end
41
42 buggy_ID_603
43 do
44
45     before := True
46 ensure
47     is_before: before

```

```

48 before_constraint: not ((not (before)) or else (not (before implies (active =
    first_element)))) implies ( before implies (active = first_element))
49 end
50
51 buggy_ID_629
52 do
53
54     before := True
55     ensure
56         is_before: before
57 before_constraint: not ((Current) ≠ Void) implies ( before implies (active =
    first_element))
58 end
59
60 buggy_ID_630
61 require
62 not (active ≠ first_element)
63 do
64
65     before := True
66     ensure
67         is_before: before
68         before_constraint: before implies (active = first_element)
69 end
70
71 buggy_ID_631
72 do
73
74     before := True
75     ensure
76         is_before: before
77 before_constraint: not (active ≠ first_element) implies ( before implies (
    active = first_element))
78 end
79
80 631
81
82 1
83 2 buggy_ID_32
84 3 do
85 4
86 5     if after then
87 6 after := not (after)
88 7 end
89 8 before := True
90 9     ensure
91 10         is_before: before
92 11         not_both: not (after and before)
93 12 end
94 13
95 14 buggy_ID_185

```



```

15     do
16
17         if (after) or else (after and before) then
18 after := not (after)
19 end
20 before := True
21     ensure
22         is_before: before
23         not_both: not (after and before)
24     end
25
26 buggy_ID_221
27     do
28
29         if (after) or else (not (not (after and before))) then
30 after := not (after)
31 end
32 before := True
33     ensure
34         is_before: before
35         not_both: not (after and before)
36     end
37
38 buggy_ID_253
39     do
40
41         before := True
42     ensure
43         is_before: before
44 not_both: not (before) implies ( not (after and before))
45     end
46
47 buggy_ID_256
48 require
49 not (after)
50 do
51
52     before := True
53     ensure
54         is_before: before
55         not_both: not (after and before)
56     end
57
58 buggy_ID_257
59     do
60
61         before := True
62     ensure
63         is_before: before
64 not_both: not (after) implies ( not (after and before))

```

```

65     end
66
67 buggy_ID_260
68 require
69 not ((before) or else (after))
70 do
71
72     before := True
73     ensure
74         is_before: before
75         not_both: not (after and before)
76     end
77
78 buggy_ID_261
79 do
80
81     before := True
82     ensure
83         is_before: before
84 not_both: not ((before) or else (after))implies ( not (after and before))
85     end
86
87 buggy_ID_263
88 do
89
90     before := True
91     ensure
92         is_before: before
93 not_both: not ((before) or else (not (after))))implies ( not (after and before))
94     end
95
96 buggy_ID_264
97 require
98 not ((not (before)) or else (after))
99 do
100
101     before := True
102     ensure
103         is_before: before
104         not_both: not (after and before)
105     end
106
107 buggy_ID_265
108 do
109
110     before := True
111     ensure
112         is_before: before
113 not_both: not ((not (before)) or else (after))implies ( not (after and before))
114     end

```

```

115
116 buggy_ID_268
117 require
118 not ((not (before)) or else (after and before))
119 do
120
121     before := True
122     ensure
123         is_before: before
124         not_both: not (after and before)
125     end
126
127 buggy_ID_269
128 do
129
130     before := True
131     ensure
132         is_before: before
133 not_both: not ((not (before)) or else (after and before))implies ( not (after and
        before))
134     end
135
136 buggy_ID_272
137 require
138 not ((not (before)) or else (not (not (after and before))))
139 do
140
141     before := True
142     ensure
143         is_before: before
144         not_both: not (after and before)
145     end
146
147 buggy_ID_273
148 do
149
150     before := True
151     ensure
152         is_before: before
153 not_both: not ((not (before)) or else (not (not (after and before))))implies (
        not (after and before))
154     end
155
156 buggy_ID_274
157 require
158 not ((after) or else (after and before))
159 do
160
161     before := True
162     ensure

```

```

163         is_before: before
164         not_both: not (after and before)
165     end
166
167     buggy_ID_275
168     do
169
170         before := True
171     ensure
172         is_before: before
173     not_both: not ((after) or else (after and before)) implies ( not (after and
        before))
174     end
175
176     buggy_ID_277
177     do
178
179         before := True
180     ensure
181         is_before: before
182     not_both: not ((after) or else (not (after and before))) implies ( not (after and
        before))
183     end
184
185     buggy_ID_278
186     require
187     not ((after) or else (not (not (after and before))))
188     do
189
190         before := True
191     ensure
192         is_before: before
193         not_both: not (after and before)
194     end
195
196     buggy_ID_279
197     do
198
199         before := True
200     ensure
201         is_before: before
202     not_both: not ((after) or else (not (not (after and before)))) implies ( not (
        after and before))
203     end
204
205     buggy_ID_283
206     do
207
208         before := True
209     ensure

```

```
210         is_before: before
211 not_both: not ((Current) ≠ Void) implies ( not (after and before))
212     end
```

Class	Size _P	Routine	Violated assertion	Failure type	Fixing time	Proof time (s)	Candidate Fixes	Valid Fixes	Proper Fixes
ACCOUNT 7/3	97	withdraw	balance_set	postcondition violation	–	0.247	–	0	–
			balance_non_negative	invariant violation	1.58	0.275	211	4	1
		deposit	balance_set	postcondition violation	1.38	0.241	111	6	1
		transfer	amount ≤ 10	precondition violation	–	0.248	–	0	–
			withdrawal_made	postcondition violation	2.72	0.243	98	5	2
			withdrawal_made	postcondition violation	–	0.253	–	0	–
			deposit_made	postcondition violation	–	0.244	–	0	–
CLOCK 8/3	131	increase_hours	valid_hours	precondition violation	2.72	0.253	356	5	1
		increase_minutes	minutes_increased	postcondition violation	2.43	0.251	301	11	1
			hours_increased	postcondition violation	–	0.263	–	0	–
			minutes_increased	postcondition violation	–	0.259	–	0	–
			minutes_increased	postcondition violation	–	0.259	–	0	–
		increase_seconds	valid_seconds	precondition violation	3.1	0.241	356	8	1
			hours_increased	postcondition violation	–	0.243	–	0	–
			minutes_increased	postcondition violation	–	0.248	–	0	–
			valid_minutes	precondition violation	3.18	0.245	356	8	1
HEATER 4/4 0	73	turn_off	heater_remains_off	postcondition violation	5.28	0.615	437	24	0
			heater_remains_on	postcondition violation	4.05	0.642	437	25	0
			heater_remains_on	postcondition violation	4.17	0.250	425	19	0
			heater_remains_off	postcondition violation	4.65	0.246	429	17	0
LAMP 4/3	71	turn_on_off	turn_off	postcondition violation	–	0.250	–	0	–
			turn_off	postcondition violation	6.83	0.278	534	10	0
		adjust_light	from_high_to_low	postcondition violation	3.3	0.265	422	4	0
			from_medium_to_high	postcondition violation	3.62	0.270	406	4	0
ARITHMETIC 3/2	176	multiply_recursive	result_correct	postcondition violation	1.1	0.301	22	6	3
			is_maximum	postcondition violation	–	0.321	–	0	–
			add_recursive	postcondition violation	1.02	0.307	30	9	0
BINARY_SEARCH 6/0	80	binary_search	present	postcondition violation	–	0.327	–	0	–
			not_in_lower_part	invariant not maintained	–	0.39	–	0	–
			not_in_lower_part	invariant not maintained	–	0.395	–	0	–
			–	variant not decreased	–	0.325	–	0	–
			not_in_upper_part	invariant not maintained	–	0.288	–	0	–
			present	postcondition violation	–	0.550	–	0	–
			not_in_lower_part	invariant violated on entry	–	0.356	–	0	–
MAX_IN_ARRAY 6/0	117	max_in_array	max_so_far	invariant not maintained	–	0.288	–	0	–
			is_maximum	postcondition violation	–	0.284	–	0	–
			result_in_array	postcondition violation	–	0.278	–	0	–
			is_maximum	postcondition violation	–	0.286	–	0	–
			max_so_far	invariant violated on entry	–	0.290	–	0	–
			i.in_bounds	invariant violated on entry	–	0.282	–	0	–
SQUARE_ROOT 4/3	38	square_root	–	variant not decreased	1.58	0.258	9	1	0
			valid_result	postcondition violation	–	0.257	–	0	–
			valid_result	invariant not maintained	1.68	0.308	9	1	0
			result_so_far	invariant not maintained	1.43	0.400	9	1	0
Total	556	14	49		44069	15.968	13.985	37/12	

Class	Size _P	Routine	Violated assertion	Failure type	Fixing time	Proof time (s)	Candidate Fixes	Valid Fixes	Proper Fixes
V_ARRAY 1/1	1756	index_set	same_bounds	postcondition violation	2.42	0.253	267	6	4
V_ARRAYED 1/1	1090	merge_right	new_count	postcondition violation	9.38	0.346	121	9	0
V_INDEXABLE 1/1	1756	ELITE_CURSOR.Cursor_Index_ad	same_bounds	postcondition violation	2.43	0.255	281	7	0
V_LINKED_LIST 2/2	3145	make	before_constraint	class invariant violation	3.13	0.246	631	8	0
			not_both	class invariant violation	1.53	0.310	283	21	2
Total	556	14	49		44069	15.968	13.985	37/12	