Test-Case Generation for Runtime Analysis and Vice-Versa: Verification of Aircraft Separation Assurance

Marko Dimjašević
University of Utah

Dimitra Giannakopoulou
NASA Ames Research Center

ISSTA 2015, Baltimore, Maryland
July 17, 2015
Goals

- Propose verification properties for aircraft separation assurance software
- Verify properties at runtime
AutoResolver

- Part of US federal government’s NextGen project
- Developed at NASA Ames Research Center
- Software system for aircraft separation assurance
- 65K lines of Java code
- Its environment’s core: 450K lines of code
Conflict, Loss of Separation, Separation Assurance
Monitored Requirements

Verification Properties
Monitored Requirements

Verification Properties

\( P_1 \) There should be a resolution for every conflict.
Monitored Requirements

Verification Properties

$P_1$ There should be a resolution for every conflict.

$P_2$ Initial conflicts are resolved in the non-decreasing order of their first time to loss of separation.
Monitored Requirements

Verification Properties

\(P_1\) There should be a resolution for every conflict.

\(P_2\) Initial conflicts are resolved in the non-decreasing order of their first time to loss of separation.

\(P_3\) New conflicts arising as a result of conflict resolution should be inserted into the list of conflicts according to their first loss of separation time.
Monitored Requirements

Verification Properties

$P_1$ There should be a resolution for every conflict.

$P_2$ Initial conflicts are resolved in the non-decreasing order of their first time to loss of separation.

$P_3$ New conflicts arising as a result of conflict resolution should be inserted into the list of conflicts according to their first loss of separation time.

$P_4$ No picked resolution is allowed to cause a more imminent secondary conflict.
Monitored Requirements

Verification Properties

$P_1$ There should be a resolution for every conflict.

$P_2$ Initial conflicts are resolved in the non-decreasing order of their first time to loss of separation.

$P_3$ New conflicts arising as a result of conflict resolution should be inserted into the list of conflicts according to their first loss of separation time.

$P_4$ No picked resolution is allowed to cause a more imminent secondary conflict.

Resolution Monitor

$M_1$ For each conflict, report its resolution type and how it changes over time.
Motivation

- Environment stubbing
- Light-weight testing with different kinds of input than trajectories
  - E.g. airspeed, initial heading, climb rate, heading change, trajectory time, destination coordinates

Purpose

- Test-case generation
- Property verification at runtime
Motivation

- Environment stubbing
- Light-weight testing with different kinds of input than trajectories
  - E.g. airspeed, initial heading, climb rate, heading change, trajectory time, destination coordinates
Wrapper

Motivation

- Environment stubbing
- Light-weight testing with different kinds of input than trajectories
  - E.g. airspeed, initial heading, climb rate, heading change, trajectory time, destination coordinates
- Test-case generation: control conflict creation process
Wrapper

Motivation

- Environment stubbing
- Light-weight testing with different kinds of input than trajectories
  - E.g. airspeed, initial heading, climb rate, heading change, trajectory time, destination coordinates
- Test-case generation: control conflict creation process

Purpose

- Test-case generation
- Property verification at runtime
Wrapper — Aspect-Oriented Programming

- Avoid usual way: instrumentation for verification
- Leave AutoResolver’s source code intact
Wrapper — Aspect-Oriented Programming

- Avoid usual way: instrumentation for verification
- Leave AutoResolver’s source code intact

AspectJ

- Java language extension
- Bytecode weaving (instrumentation)
Wrapper — Aspect-Oriented Programming

- Avoid usual way: instrumentation for verification
- Leave AutoResolver’s source code intact

AspectJ

- Java language extension
- Bytecode weaving (instrumentation)

In-house verification

- No external verification tool used (SMT solvers, MOP tools)
Wrapper — Properties

Properties as AspectJ aspects

- 1 property = 1 aspect
- 1 aspect = multiple pointcuts and advices
Wrapper — Properties

Properties as AspectJ aspects

- 1 property = 1 aspect
- 1 aspect = multiple pointcuts and advices

Pointcuts

- Where are interesting points of execution in AutoResolver?
Wrapper — Properties

Properties as AspectJ aspects

- 1 property = 1 aspect
- 1 aspect = multiple pointcuts and advices

Pointcuts

- Where are interesting points of execution in AutoResolver?
- Points in wrapper itself
Wrapper — Properties

Properties as AspectJ aspects

- 1 property = 1 aspect
- 1 aspect = multiple pointcuts and advices

Pointcuts

- Where are interesting points of execution in AutoResolver?
- Points in wrapper itself

Advices

- Actions to be taken at pointcuts
pointcut callAR(AacTestWrapper wrapper):
    call(public ArrayList conflictDetectResolve()) &&
    target(wrapper) &&
    !cflow(myAspect()) &&
    !cflow(callFlyForMethod(*, *)) &&
    if(isEnabled);

after(AacTestWrapper wrapper): callAR(wrapper) {
    for (t = 60.0; t <= 480.0; t += 60.0) {
        AacTestWrapper w = wrapper.flyFor(t);
        w.conflictDetectResolve();
    }
}
Runtime Verification

- Verification at runtime

“Testing shows the presence, not the absence of bugs.” — Dijkstra
Runtime Verification

- Verification at runtime
- Need for good runtime drivers
  - Test cases
Runtime Verification

- Verification at runtime
- Need for good runtime drivers
  - Test cases

“Testing shows the presence, not the absence of bugs.” — Dijkstra
Test-Case Generation

- Arbitrary many conflicts
Test-Case Generation

- Arbitrary many conflicts
- Secondary conflicts — challenging to create
Test-Case Generation

- Arbitrary many conflicts
- Secondary conflicts — challenging to create
- Time dimension added at runtime
Generating Secondary Conflicts

- Secondary conflicts: created along a resolution trajectory
Generating Secondary Conflicts

- Secondary conflicts: created along a resolution trajectory

Extend black-box test cases through reflection and with runtime verification
Generating Secondary Conflicts

- Secondary conflicts: created along a resolution trajectory

Extend black-box test cases through reflection and with runtime verification
public void test0() throws Throwable {

    AacTestWrapper wrapper = new AacTestWrapper();

    wrapper.setUpCR(CR_parameters1);
    wrapper.setUpCL(CL_parameters2);
    wrapper.setUpCR(CR_parameters3);

    wrapper.conflictDetectResolve();
}

Evaluation

- Test suite of 3.5 million test cases
  - Each test case with about 5 conflicts
- Every test case executed at 9 different time points
  - Fly all aircraft for some time and then call AutoResolver
  - Effectively: \(3.5 \text{ million} \cdot 9 = 31.5\text{ million test cases}\)
- Check if every requirement is exercised
  - Second-level monitors
Results — Property $P_1$

There should be a resolution for every conflict.
Results — Property $P_1$

There should be a resolution for every conflict.

- It does not hold, but this is not a bug
- AutoResolver does not resolve conflicts that:
  - involve aircraft already in violation
  - happen earlier than a predetermined time limit (1 minute)
  - happen later than a predetermined time limit (8 minutes)
  - “Neither plane able to maneuver/neither plane able to be unfrozen” (current resolution round)
Results — Property $P_2$

Initial conflicts are resolved in the non-decreasing order of their first time to loss of separation.
Results — Property $P_2$

Initial conflicts are resolved in the non-decreasing order of their first time to loss of separation.

- No violation found
Results — Property $P_3$

New conflicts arising as a result of conflict resolution should be inserted into the list of conflicts according to their first loss of separation time.
Results — Property $P_3$

New conflicts arising as a result of conflict resolution should be inserted into the list of conflicts according to their first loss of separation time.

- No violation found
- No test case that exercises respective parts of code
  - Second-level monitor
- Need support for weather conflict type
Results — Property $P_4$

No picked resolution is allowed to cause a more imminent secondary conflict.
Results — Property $P_4$

No picked resolution is allowed to cause a more imminent secondary conflict.

- No violation found
  - Several test cases used to indicate violation (bug found in wrapper)
Results — Resolution Monitor $M_1$

For each conflict, report its resolution type and how it changes over time.
Results — Resolution Monitor $M_1$

For each conflict, report its resolution type and how it changes over time.

<table>
<thead>
<tr>
<th>ttlos [s]</th>
<th>Delay time [s]</th>
<th>Res type</th>
</tr>
</thead>
<tbody>
<tr>
<td>430.0</td>
<td>0.0</td>
<td>26</td>
</tr>
<tr>
<td>370.0</td>
<td>60.0</td>
<td>26</td>
</tr>
<tr>
<td>310.0</td>
<td>120.0</td>
<td>26</td>
</tr>
<tr>
<td>250.0</td>
<td>180.0</td>
<td>26</td>
</tr>
<tr>
<td>190.0</td>
<td>240.0</td>
<td>26</td>
</tr>
<tr>
<td>130.0</td>
<td>300.0</td>
<td>13</td>
</tr>
<tr>
<td>70.0</td>
<td>360.0</td>
<td>13</td>
</tr>
<tr>
<td>10.0</td>
<td>420.0</td>
<td>not resolved</td>
</tr>
<tr>
<td>0.0</td>
<td>480.0</td>
<td>not resolved</td>
</tr>
</tbody>
</table>
No-conflict window?

<table>
<thead>
<tr>
<th>ttlos [s]</th>
<th>Delay time [s]</th>
<th>Res type</th>
</tr>
</thead>
<tbody>
<tr>
<td>445.0</td>
<td>0.0</td>
<td>3</td>
</tr>
<tr>
<td>—</td>
<td>60.0</td>
<td>—</td>
</tr>
<tr>
<td>—</td>
<td>120.0</td>
<td>—</td>
</tr>
<tr>
<td>265.0</td>
<td>180.0</td>
<td>3</td>
</tr>
<tr>
<td>205.0</td>
<td>240.0</td>
<td>3</td>
</tr>
<tr>
<td>145.0</td>
<td>300.0</td>
<td>3</td>
</tr>
<tr>
<td>85.0</td>
<td>360.0</td>
<td>3</td>
</tr>
<tr>
<td>25.0</td>
<td>420.0</td>
<td>not resolved</td>
</tr>
<tr>
<td>0.0</td>
<td>480.0</td>
<td>not resolved</td>
</tr>
</tbody>
</table>
Summary

- Light-weight verification of aircraft separation assurance software
- Runtime verification for test-case generation
- Test-case generation for runtime verification
Summary

- Light-weight verification of aircraft separation assurance software
- Runtime verification for test-case generation
- Test-case generation for runtime verification

Credits

- Crossroads — Umberto Nicoletti
- This presentation — Marko Dimjašević, CC-BY-SA 4.0