Analyse statique de programmes avioniques

Presenté par
Jean Souyris (Airbus Opérations SAS)
Outline

• a3 - Timing analysis (WCET)
• a3 – Stack analysis
• Astrée
a3 - Timing Analysis
a3 - Timing analysis (WCET): Determinism

• **Hardware determinism**
  • *It must be possible for AbsInt to model the processor* => constraints when choosing and configuring it
    • Examples of such constraints:
      • No dynamic branch prediction
      • No write-back unless it is caused by program
      • LRU cache is better but pseudo LRU is OK (less precision)
  • *The same hold for the chipset*
    • Constraints:
      • Deterministic access to memory and devices (as few history effects as possible);
      • No asynchronous events
        • No RAM refresh (dynamic RAM)
        • No other bus masters than the processor

If not => statistics to be added to a3 WCET computation
a3 - Timing analysis (WCET): Determinism

- **Software determinism**
  - Periodic tasks that are serialized by design
    - Ex: pattern of 24 code clusters of SCADE nodes executed in a circular way, from 1 to 24 then 1 again, etc...
  - Simple deterministic scheduler (no pre-emption)
  - Unique interrupt: the Real Time Clock that activates the clusters
    - Example of clock rate: 5ms.

- **What is computed by a3**
  - The WCET of each of the 24 clusters

- **What is checked**
  - WCET(cluster) ≤ 5ms.
An automatic workshop based on a3:

- Generation of .apx and .ais files
  - From templates
  - From the assembly listings (.lst files) of SCADE modules
• **COS SCADE component:**
  • COS et SIN interpolent leur résultat à partir des éléments d’un tableau. Ces symboles calculent un modulo entre 0 et 360 en virgule flottante qui est converti en nombre entier pour servir d’index dans le tableau. Or, A3_WCET_PPC ne gère pas les calculs en virgule flottante, l’index d’accès au tableau ne pourra donc pas être déterminé de façon automatique. […] Il est donc possible de forcer la valeur du registre dans l’intervalle [0..360] juste après son chargement. (En registre).

• **COS SCADE component specification of the annotation:**
  • **COS valeur 0x74 0 360**

• **Example of annotation produced automatically:**
  *Instruction "Nm_111610" + 0x178 bytes is entered with r31 = from 0 to 360;*
a3-Timing analysis: automatically generated annotations, other example

- **DELAY SCADE component**
  - SCADE Component Delay handles an array index over successive activations of a cluster (see slide 5) whereas a3 analyses one activation only. Therefore the bounds of the values of the index cannot be computed by the tool: they must be provided by means of annotations.

- Spécification de l’annotation DELAY sur l’unité B :
  - **DELAY valeur 0x08 0 @3-1** (Value of the 3rd parameter of macro DELAY minus 1)

- Example of annotation produced automatically:
  
  `instruction "Nm_121031" + 0xe0 bytes is entered with r31 = from 0 to 124;`
Aeronautics Valley Forum on Formal Methods – Conference cycle

CLOCK EXACTLY 288 MHz;

### N_084028 ###
# PREV_I 1
instruction "Nm_084028" + 0xc4 bytes is entered with r31 = from 0 to 0;
# PREV_I 2
instruction "Nm_084028" + 0x100 bytes is entered with r31 = from 0 to 0;
### SAT.
# PREV_I 3
instruction "Nm_084028" + 0x4b4 bytes is entered with r31 = from 0 to 0;
# PREV_I 4
loop "SA" instruction "Nm_084028" + 0x562 bytes is entered with r31 = from 0 to 0;

# TOUTES POSITIONS

AREA "BUNITA" CONTAINS byte 0x0;
try AREA "BUNITB" CONTAINS byte 0x1;

### AREA "BFALSE" CONTAINS byte 0x0; ###
### AREA "BTRUE" CONTAINS byte 0x1; ###

# TOUTES POSITIONS

AREA "BINIT" CONTAINS byte 0x0;
lo

### CONDITIONS POUR RESTER EN OPERATIONNEL ###

AREA "BFPRIMD" CONTAINS byte 0x0;
 AREA "BSFTSTART" CONTAINS byte 0x0;
 AREA "BGNDTEST" CONTAINS byte 0x0;
 AREA "BUPLOADAUT" CONTAINS byte 0x0;

# Prise en compte COMO 20 sur la gestion des BCLEOK et BCLEOKxy
AREA "BCLEOK" CONTAINS byte 0x0;

AREA 0x80000000 .. 0x800000FF USES ASYNCHRONOUS CLOCK EXACTLY 33 MHz;
a3-Timing analysis: results

Analyses - Overview (1)

<table>
<thead>
<tr>
<th>ID</th>
<th>Type</th>
<th>Result</th>
<th>Expectation Met</th>
</tr>
</thead>
<tbody>
<tr>
<td>WcetAnalysis</td>
<td>aiT</td>
<td>1066042 cycles = 3.702 ms</td>
<td></td>
</tr>
</tbody>
</table>

Messages for analysis WcetAnalysis:

- Control Flow Reconstruction
- Value Analysis
- Cache & Pipeline Analysis
- Prediction File Optimization
- Path Analysis (Prediction File Based)
- Applying Path Analysis Results
- Visualization

Finished after 2 minutes 44 seconds with 0 errors, 0 warnings
### a3-Timing analysis: results

<table>
<thead>
<tr>
<th>Function</th>
<th>Count</th>
<th>Worst-Case Execution Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSEO_SeqOperationnelleC1_1</td>
<td>5761</td>
<td>19,463 us</td>
</tr>
<tr>
<td>SSEO_SeqOperationnelleC2_1</td>
<td>433</td>
<td>1,463 us</td>
</tr>
<tr>
<td>SSEO_SeqOperationnelleC3_1</td>
<td>5018</td>
<td>16,953 us</td>
</tr>
<tr>
<td>SSEO_SeqOperationnelleC4_1</td>
<td>3915</td>
<td>13,227 us</td>
</tr>
<tr>
<td>SSSA_EcrireOctetsInverses32</td>
<td>174</td>
<td>0,588 us</td>
</tr>
<tr>
<td>SSSA_EcrireTBL</td>
<td>100</td>
<td>0,338 us</td>
</tr>
<tr>
<td>SSSA_EcrireTBU</td>
<td>65</td>
<td>0,220 us</td>
</tr>
<tr>
<td>SSSA_LireTBL</td>
<td>912</td>
<td>3,082 us</td>
</tr>
<tr>
<td>SSSA_Log2ValeurSup</td>
<td>825</td>
<td>2,788 us</td>
</tr>
<tr>
<td>SSSA_RetournerOctetsInverses32</td>
<td>14487</td>
<td>48,943 us</td>
</tr>
<tr>
<td>SSSB_GenerSurveillancesMutuelles</td>
<td>1616</td>
<td>5,460 us</td>
</tr>
<tr>
<td>STON_ExecuterScadeOnTache1</td>
<td>599</td>
<td>2,024 us</td>
</tr>
<tr>
<td>SUDE_EnvoieDGO</td>
<td>21827</td>
<td>73,740 us</td>
</tr>
</tbody>
</table>

Computed Worst-Case Execution: 116183 | 3925,281 us
a3-Timing analysis: usage domain at Airbus’s

- **a3 PowerPC MPC 755 + Hurricane Chipset + Dynamic RAM + asynchronous events**
  - Extra statistics must be added to a3 computation
  - Fly-by-wire software (PRIM A380 function)
  - IOM (belonging to IMA A380)

- **a3 PowerPC MPC 7448 + Jekyll chipset + Dynamic RAM + asynchronous events**
  - Extra statistics added to A3 computation
  - Fly-by-wire software (PRIM A350 function)

- **a3 Intel 386 DX and a3 Enhanced Am486 DX**
  - A330 / A340 family Fly-by-wire (PRIM)

- **a3 for TMS320C33**
  - I/O boards for fly-by-wire and IOM
a3 - Stack Analysis
a3 - Stack analysis

• What is computed by a3
  • An upper bound of the memory consumed by the stack of each execution thread of a application

• What is checked
  • The upper bound of each thread’s stack must be lower than the amount of memory allocated to it
a3 - Stack analysis: configuration (Example)

Starting extended feasibility analysis...

- #3077: Loop '__hm_panic.L1' is entered but never left, i.e. it contains a dead end.
- #3077: Loop '__hm_panic.L1' is entered but never left, i.e. it contains a dead end.
- #3077: Loop '__hm_panic.L1' is entered but never left, i.e. it contains a dead end.
- #3077: Loop '__hm_panic.L1' is entered but never left, i.e. it contains a dead end.
- #3077: Loop '__hm_panic.L1' is entered but never left, i.e. it contains a dead end.
- #3077: Already printed 5 messages of this kind, no more messages of this kind will be issued.

Fixed-point iteration processed 853394 nodes * contexts:
Running stack analyzer backend...
Enumerating feasible callstacks starting at 'BAM_Sa_ThreadGeneralProcess'...
Enumerating feasible callstacks...
Writing text-report file...
Writing avi info to report file...
Exporting ctrl2...
Writing xml result...
Last process took 893.431 s and used not more than 1555 MB (RSS 1362 MB) of memory

Reporting
Visualization
Finished after 15 minutes 23 seconds with 0 errors, 2 warnings
a3 - Stack analysis: Analysis (Example)

Aerospace Valley Forum on Formal Methods – Conferences cycle
Astrée
Astrée: industrial use / Configuration

- **Fly-by-wire programs (PRIM functions)**
  - C code automatically generated from SCADE models
  - About 1 million lines of code
  - Astrée is used on major versions

- **Configuration of the first analysis**
  - Emulation of the execution environment of the analysed program
    - Safe and realistic ranges of values assigned to the inputs
    - Software emulation of the analysed program periodic activation ("main()" for the purpose of the analysis)
  - C Stubs for non analysable functions (assembly code)
  - First set of Astrée directives for coping with
    - Small pieces of code known as non analysable precisely (very few cases)
    - In-line assembly blocks
Astrée: workshop / First analysis

• **An Airbus’s workshop for Astrée (developed by ATOS)**
  • *Make the re-use of a configuration as automatic as possible*
    • Quick start of the analysis of a new version of the analysed program
  • *Directives in “patch” files*
    • The patches are applied to the new version
  • *Pre-processing* of the all the C files (after application of the patches)

• **First analysis**
  • Astrée in *parallel mode* (rather specific for Airbus)
  • Duration: *between 10 and 14 hours* (depending on the analysed program)
  • First analysis usually produces plenty of false alarms
Astrée: Analysis iterative process

- **Fine tuning of the first analysis**
  - Reduced example technique (next slide)
  - Use of Astrée directives (examples)
  - `__ASTREE_partition_control / __ASTREE_partition_merge()`
  - `__ASTREE_assert()`

- **New Analysis**
  - Alarms ?
  - Yes:
  - No: the analysis process is finished
    - *New hypotheses to be checked by system designers*
    - *Alarms that cannot be reduced at software level are submitted to system designers*
Astrée: Reduced Example technique

- **Localize the root cause of an alarm**
  - Astrée’s GUI: computed local invariants displayed on demand + many facilities for browsing the results
  - *Backward (abstract) slicing would be interesting...*

- **Reduced example creation and analysis**
  - **C files**:  
    - The one containing the *root cause*
    - The main() for the analysis
  - *Input ranges taken from the analysis results of the whole program*
  - *Quick analysis (tiny program): less than 1 mn*
  - **After Investigation:**
    - Either the root cause and/or hypotheses are recorded for submission to the system designer,
    - Or the alarm is false and directives for making the whole program analysis not to raise it are applied
Astrée: Snapshots (on going alarm reduction process)

 June 28th 2013.

 Aerospace Valley Forum on Formal Methods – Conferences cycle
Thank you for your attention, Questions?