



Advancements in User Interface and Usability of KeY

Wolfram Pfeifer | August 9, 2023



www.kit.edu



KSY Strategy: Applied 64 rules (0.0 sec), closed 1 goal, 1 remaining



KRY Strategy: Applied 64 rules (0.0 sec), closed 1 goal, 1 remaining



KRY Strategy: Applied 64 rules (0.0 sec), closed 1 goal, 1 remaining





KRY Strategy: Applied 64 rules (0.0 sec), closed 1 goal, 1 remaining





Vision

- Open goal should be presented in a view the user is familiar with.
- Interaction should be possible on the input artifact (Java/JML).
- When switching between different representations, KeY should present hints that help to connect them.
- Formulae should be presented in the way the user has written them.

Progress so far



"Integrating Source Code, Specification and Proof State into a Single Interactive View for the Deductive Verification Tool KeY" (Master's Thesis, Mike Schwörer)

Idea: Represent a goal (sequent) of the proof as JML.

Karlsruhe Institute of Technology

Progress so far

"Integrating Source Code, Specification and Proof State into a Single Interactive View for the Deductive Verification Tool KeY" (Master's Thesis, Mike Schwörer)

Idea: Represent a goal (sequent) of the proof as JML.

- Take initial PO and assign origins/categories to the terms
- Transform correctly under rule applications
- Render the new view:

Input: Sequent with origin/category tags, Java/JML

Output: Source code with additional JML assume/assert statements placed

Karlsruhe Institute of Technology

Progress so far

"Integrating Source Code, Specification and Proof State into a Single Interactive View for the Deductive Verification Tool KeY" (Master's Thesis, Mike Schwörer)

Idea: Represent a goal (sequent) of the proof as JML.

- Take initial PO and assign origins/categories to the terms
- Transform correctly under rule applications
- Render the new view:

Input: Sequent with origin/category tags, Java/JML

Output: Source code with additional JML assume/assert statements placed

Assumptions

- Symbolic execution has finished (no modalities).
- All updates are applied.
- Restrictions to allowed programs (e.g., no for loops, only return + variable, ...).

8 * KeY 2.11.0 [schwoerer/ma-ext-source-view] \vee \wedge \times File View Proof Options Origin Tracking About Run Z3. CVC5. Princess 👻 📧 💥 🖿 🕐 😰 🗃 😨 🛫 🖉 🎹 Layouts: Default 🗸 Load Layout: Save Layout Reset Layout 🛛 Exploration Mode 🗌 Hide justification 0 _ 8 0 C Loaded Proofs Sequent Source _ 8 0 C Loaded Proofs Cannot transform formula with modalities. - Finish symbolic execution to continue Proofs with model Part 1@3:07:05 PM CaesarChiffre.java G CaesarChiffreICaesarChiffretcalcChi 19 @ ensures (\forall int i; 0 <= i && i < valuesOutput.length; (valuesInput[i] + offset <= 'Z') ==> (valuesOutput[i] == (valuesInput[i] + offset 20 @ ensures (\forall int i; 0 <= i && i < valuesOutput.length; (valuesInput[i] + offset > 'Z') ==> (valuesOutput[i] == (valuesInput[i] + offset - 26))); 21 • Þ 22 @ ensures (\forall int i: 0 <= i && i < valuesOutput.length: 'A' <= valuesOutput[i] && valuesOutput[i] <= 'Z'); 23 ExtSourceView {{DEBUG}} 24 @ ensures \result == valuesInput.length; Goals 25 1 Info SProof Search Strategy 26 @ assignable valuesInput[*], valuesOutput[*]; Exploration Steps - Proof 27 0*/ Proof _ 8 0 int calcChiffre(int offset) { 28 29 Proof Tree 30 int loopidx = 0; ନ୍ନ 0:OPEN GOAL 31 32 convertToUpper(); 33 34 /*0 35 @ loop invariant @ <= loopidx;</pre> 36 @ loop invariant loopidx <= valuesInput.length;</pre> 37 38 @ loop_invariant (\forall int i; 0 <= i && i < loopidx; (valuesInput[i] + offset <= 'Z') ==> (valuesOutput[i] == (valuesInput[i] + offset - 26))); 39 @ loop_invariant (\forall int i; 0 <= i && i < loopidx; (valuesInput[i] + offset > '2') ==> (valuesOutput[i] == (valuesInput[i] + offset))); 40 41 @ loop invariant (\forall int i: 0 <= i && i < loopidx: 'A' <= valuesOutput[i] && valuesOutput[i] <= 'Z');</pre> 42 43 Ø decreases valuesInput.length - loopidx: 44 45 @ assignable valuesOutput[*]; 46 Symbolic Execution and Simplification Symbolic Execution, Simplification, and Close Provable Goals 47 while (loopidx < valuesInput.length){</pre> 48 **Close If Provable** 49 if (valuesInput[loopidx] <= 'Z' - offset) {</pre> 50 int tmp1 = valuesInput[loopidx] + offset; 51 valuesOutput[loopidx] = (char)tmp1; 52 Split and Close Provable Goals 53 } else { 54 55 int tmp2 = valuesInput[loopidx] + offset - 26; 56 valuesOutput[loopidx] = (char)tmp2; 57 58 59 60 61 loopidx++: 62 Show Postcondition/Assignable Show log 📀 KRY Replaying proof

8 * KeY 2.11.0 [schwoerer/ma-ext-source-view] \vee \wedge \times File View Proof Options Origin Tracking About ►C 🛛 🖻 🖺 🗄 🎜 🗸 **a** II 0 Run Z3. CVC5. Princess 👻 🕿 💥 Layouts: Default 🔻 Load Layout Save Layout Reset Layout Exploration Mode Hide justification _ 8 🗆 C Loaded Proofs Sequent Source _ 8 0 C Loaded Proofs 30 int loopidx = 0; Proofs 31 with model Part 1@3:07:05 PM G CaesarChiffreICaesarChiffretcalcChi 32 convertToUpper(): 33 34 /*0 35 • @ loop invariant 0 <= loopidx; 36 @ loop invariant loopidx <= valuesInput.length;</p> ExtSourceView {{DEBUG}} 37 Goals 38 @ loop_invariant (\forall int i; 0 <= i && i < loopidx; (valuesInput[i] + offset <= 'Z') ==> (valuesOutput[i] == (valuesInput[i] + offset - 26))); 1 Info Proof Search Strategy 39 @ loop invariant (\forall int i; 0 <= i && i < loopidx; (valuesInput[i] + offset > 'Z') ==> (valuesOutput[i] == (valuesInput[i] + offset))); Exploration Steps - Proof 40 41 @ loop invariant (\forall int i: 0 <= i && i < loopidx: 'A' <= valuesOutput[i] && valuesOutput[i] <= 'Z');</pre> Proof _ 8 0 42 43 @ decreases valuesInput.length - loopidx; vertToUpper) 44 ant Initially Valid @ assignable valuesOutput[*]; Preserves Invariant 46 @*/ rmal Execution (x arr != null) //@ assume \old(valuesInput)floopidxl <= (90 + (offset * -1));</pre> Normal Execution (x_arr_1 != null) //@ assume \forall int i; ((i < (\old(valuesInput).length)) && (0 <= i)) ==> ('A' <= valuesInput[i] && valuesInput[i] <= 'Z');</pre> 📑 if x_2 true while (loopidx < valuesInput.length){</pre> + 🗂 Normal Execution (x arr 2 != 48 + 📑 if x 5 true 49 if (valuesInput[loopidx] <= 'Z' - offset) +</pre> + C Normal Execution (x ar //@ assume (\forall int i; 0 <= i && i < loopidx; (valuesInput[i] + offset <= 'Z') ==> (valuesOutput[i] == (valuesInput[i] + offset - 26))); + 📑 Normal Execution (x //@ assume (\forall int i; @ <= i && i < loopidx; (valuesInput[i] + offset > 'Z') ==> (valuesOutput[i] == (valuesInput[i] + offset))); € 705:OPEN GOAL //@ assume (\forall int i; 0 <= i && i < loopidx; 'A' <= valuesOutput[i] && valuesOutput[i] <= 'Z');</pre> Null Reference (x ar int tmp1 = valuesInput(loopidx] + offset; 50 Index Out of Bounds 51 valuesOutput[loopidx] = (char)tmp1: 826:Closed goal 52 Null Reference (x arr 3) 53 } else { Index Out of Bounds (x) 5.4 + 📑 if x 5 false int tmp2 = valuesInput(loopidx) + offset - 26; 55 + 🗂 Normal Execution (x_ar 56 valuesOutput[loopidx] = (char)tmp2: + 📑 Normal Execution (x 57 G 639:OPEN GOAL 58 Null Reference (x ar 59 Index Out of Bounds 60 Null Reference (x arr 3 //@ assume offset >= 0: Index Out of Bounds (x) //@ assume offset < 26;</pre> Null Reference (x_arr_2 = null //@ assume @ <= loopidx;</pre> Index Out of Bounds (x arr 2 61 if x 2 false //@ assert 0 <= loopidx; Null Reference (x arr 1 = null) //@ assert loopidx <= valuesInput.length;</pre> II Reference (x_arr = null) //@ assert (\forall int i; 0 <= i && i < loopidx; (valuesInput[i] + offset <= 'Z') ==> (valuesOutput[i] == (valuesInput[i] + offset - 26))); ase //@ assert (\forall int i; 0 <= i && i < loopidx; (valuesInput[i] + offset > 'Z') ==> (valuesOutput[i] == (valuesInput[i] + offset))); hal Post (convertToUpper) //@ assert (\forall int i: 0 <= i && i < loopidx: 'A' <= valuesOutput[i] && valuesOutput[i] <= 'Z');</pre> ertToUpper) //@ assert @ <= (valuesInput.length - loopidx) < \old<51>(valuesInput.length - loopidx); 62 4 Normal Execution (x arr 5 != null) Show log Ø KGY Strategy: Applied 3000 rules (3.5 sec), closed 11 goals, 192 remaining

8 * KeY 2.11.0 [schwoerer/ma-ext-source-view] \vee \wedge \times File View Proof Options Origin Tracking About ► C Z B 🛯 🗄 🎜 - 🖉 🞹 0 Run Z3. CVC5. Princess 👻 🕿 💥 Layouts: Default 🔻 Load Layout Save Layout Reset Layout Exploration Mode Hide justification _ 8 🗆 C Loaded Proofs Sequent Source _ 8 0 C Loaded Proofs 30 int loopidx = 0; Proofs 31 with model Part 1@3:07:05 PM G CaesarChiffreICaesarChiffretcalcChi 32 convertToUpper(): 33 34 /*0 35 • @ loop invariant 0 <= loopidx;</pre> 36 @ loop invariant loopidx <= valuesInput.leng</pre> ExtSourceView {{DEBUG}} 37 Goals 38 @ loop_invariant (\forall int i; 0 <= i && i < loopidx; (valuesInput[i] + offset <= 'Z') ==> (valuesOutput[i] == (valuesInput[i] + offset - 26))); 1 Info Proof Search Strategy 39 @ loop invariant (\forall int i; 0 <= i && i < loopidx; (valuesInput[i] + offset > 'Z') ==> (valuesOutput[i] == (valuesInput[i] + offset))); Exploration Steps - Proof 40 41 @ loop invariant (\forall int i: 0 <= i && i < loopidx: 'A' <= valuesOutput[i] && valuesOutput[i] <= 'Z');</pre> _ 8 0 Proof 42 43 @ decreases valuesInput.length - loopidx; vertToUpper) 44 ant Initially Valid @ assignable valuesOutput[*]; Preserves Invariant 46 @*/ rmal Execution (x arr != null) //@ assume \old(valuesInput)floopidxl <= (90 + (offset * -1));</pre> Normal Execution (x arr 1 != null) //@ assume \forall int i; ((i < (\old(valuesInput).length)) && (0 <= i)) ==> ('A' <= valuesInput[i] && valuesInput[i] <= 'Z');</pre> 📑 if x_2 true while (loopidx < valuesInput.length){</pre> + 🗂 Normal Execution (x arr 2 != 48 + 📑 if x 5 true 49 if (valuesInput[loopidx] <= 'Z' - offset) +</pre> + C Normal Execution (x ar //@ assume (\forall int i; 0 <= i && i < loopidx; (valuesInput[i] + offset <= 'Z') ==> (valuesOutput[i] == (valuesInput[i] + offset - 26))); + 📑 Normal Execution (x //@ assume (\forall int i; @ <= i && i < loopidx; (valuesInput[i] + offset > 'Z') ==> (valuesOutput[i] == (valuesInput[i] + offset))); € 705:OPEN GOAL //@ assume (\forall int i; 0 <= i && i < loopidx; 'A' <= valuesOutput[i] && valuesOutput[i] <= 'Z');</pre> Null Reference (x ar 50 int tmp1 = valuesInput(loopidx] + offset; Index Out of Bounds 51 valuesOutput[loopidx] = (char)tmp1: 826:Closed goal 52 Null Reference (x arr 3) 53 } else { Index Out of Bounds (x) 5.4 + 📑 if x 5 false int tmp2 = valuesInput(loopidx) + offset - 26; 55 + 🗂 Normal Execution (x_ar 56 valuesOutput[loopidx] = (char)tmp2: + 📑 Normal Execution (x 57 G 639:OPEN GOAL 58 Null Reference (x ar 59 Index Out of Bounds 60 Null Reference (x arr 3 //@ assume offset >= 0: Index Out of Bounds (x) //@ assume offset < 26;</pre> Null Reference (x_arr_2 = null //@ assume @ <= loopidx;</pre> Index Out of Bounds (x arr 2 61 if x 2 false //@ assert 0 <= loopidx, Null Reference (x arr 1 = null) //@ assert loopidx <= valuesInput.length II Reference (x_arr = null) //@ assert (\forall int i; 0 <= i && i loopidx; (valuesInput[i] + offset <= 'Z') ==> (valuesOutput[i] == (valuesInput[i] + offset - 26))); ase //@ assert (\forall int i; 0 <= i && i < loopidx; (valuesInput[i] + offset > 'Z') ==> (valuesOutput[i] == (valuesInput[i] + offset))); hal Post (convertToUpper) //@ assert (\forall int i: 0 <= i && i < loopidx: 'A' <= valuesOutput[i] && valuesOutput[i] <= 'Z');</pre> ertToUpper) //@ assert @ <= (valuesInput.length - loopidx) < \old<51>(valuesInput.length - loopidx); 62 4 Normal Execution (x arr 5 != null) Show log Ø KGY Strategy: Applied 3000 rules (3.5 sec), closed 11 goals, 192 remaining

8 *	KeY 2.11.0 [schwoerer/ma-ext-source-view]	~ ^ ×
File View Proof Options Origin Trac	king	About
Run Z3, CVC5, Princess	🗸 🚭 🛠 " 🖻 🕑 🚝 🕲 🚝 🖉 📮 🕲 🗮 🗳 🖬 Layout: Default 🗸 Load Layout Save Layout Reset Layout 🖉 Exploration N	Mode 🔲 Hide justification
Loaded Proofs Sequent	C Source	_ 8 🗆
🗍 Loaded Proofs 💦 🗕 🗖 🗖	CaesarChiffre java	
Proofs	and the second sec	-
. with model Part_1@3:07:05 PM	43 @ decreases valuesInput.length - looplax;	
CaesarChiffre[CaesarChiffre::calcChi	44 e assignable values0utout[*]	
	46 0*/	
	<pre>//@ assume \old(valuesInput)[loopidx] <= (90 + (offset * -1));</pre>	
	<pre>//@ assume \forall int i; ((i < (\old(valuesInput).length)) && (0 <= i)) ==> ('A' <= valuesInput[i] && valuesIn</pre>	<pre>nput[i] <= 'Z');</pre>
Enter annual Views (VDEBUG))	47 while (loopidx < valuesInput.length) (
Excourceview {{DEBOG}}	48	
6 Info	49 if (valuesInput[loopidx] <= 'Z' - offset) {	
Exploration Steps	<pre>//@ assume (\forall int 1; 0 <= 1 && 1 < loopidx; (valuesInput[i] + offset <= '2') ==> (valuesOutput[</pre>	1] == (valuesInput[1] + offset - 26)));
	//@ assume (\totall int 1; 0 <= 1 &&i < looplox; (valuesinput[i] + ottset > '2') ==> (valuesoutput[:	<pre>ij == (ValuesInput[i] + offset)));</pre>
••• Proor _ 🖬 🗆	50 int the valueSourput[1] <= 2);	
n	51 valueSoutout[loopidx] = (char)tmp1:	
valid	52	
variant	53) else {	
ion (x_arr != null)	54	
cution (x_arr_1 != null)	<pre>55 int tmp2 = valuesInput[loopidx] + offset - 26;</pre>	
e	56 valuesOutput[loopidx] = (char)tmp2;	
al Execution (x_arr_2 != null)	57	
_5 true	58) 50	
Normal Execution (x_arr_3 != null)	55	
Case 1	//@ assume offset >= 0:	
Q 10029:OPEN GOAL	<pre>//@ assume offset < 26;</pre>	-
Case 2	//@ assume 0 <= loopidx;	
Null Reference (x_arr_5 = null)	61 loopidx++;	
Index Out of Bounds (x_arr_5 != i	//@ assert 0 <= loopidx;	
Null Reference (x_arr_3 = null)	<pre>//@ assert loopidx <= valuesInput.length;</pre>	1
Index Out of Bounds (x_arr_3 != nul	//@ assert (\forail int 1; 0 <= 1 && 1 < looptax; (Valuesinput[i] + orrset <= '2') ==> (Valuesuurput[i] = ((A = 1)) ==> ((A =	= (valuesInput[1] + offset - 20)));
_5 faise	//e assert () for all int i, $b \le i$ as i < hopping, (values input[i] - $b = i \ge j = -b$ (values durput[i] = - //b = - /b =	= (valuesinpuc[i] + onsec ()))),
Out of Bounds (x arr 2 is null, but	62 }	Symbolic Execution and Simplification
e	63	symbolic execution, simplification, and close provable doals
hce (x arr 1 = null)	64 return values0utput.length;	Close If Provable
(x_arr = null)	//@ assignable [valuesOutput[*]]; //TODO	Cut on this term (cut_airect)
	65 }	cut c-lia
vertToUpper)		Split and Close Prevable Coals
	0/ //@INDERBALDOLE	Vide this term
	69 0 requires valuestonut length > 8:	Insert hidden (0 items)
	70 0	Testentiste Questifier
	71 e ensures (\forall int i; 0 <= i && i < valuesInput.length; 'A' <= valuesInput[i] && valuesInput[i] <= 'Z');	Instantiate Quantinei
	72 0	
	73 @ assignable valuesInput[*];	
	74 @*/	-
Show tarlet info (inner noder only)	Normal Execution (x arr 5 to null)	
- show water mo (nimer notes only)		
Kg Proof has been pruned: 2 open go	Js remain.	😫 Show log 🥥

8 *	KeY 2.11.0 [schwoerer/ma-ext-source-view] V ^
File View Proof Options Origin Tra	ing Al
Due 13 CHEL Bringer	
Run 23, CVC3, Princess	
Loaded Proofs Sequent	Source _ 5
🗋 Loaded Proofs 🛛 🗖 🗖	CaesarChiffrejava
Proofs	22 Contractionplex (1)
. with model Part_1@3:07:05 PM	34 /**
요esarChiffre[CaesarChiffre::calcChi	25 0 loop invariant 0 <= loopidy:
	36 0 loop invariant loopids / valuesInput length:
	37
	38 Olop_invariant (\forall int i; 0 <= i && i < loopidx; (valuesInput[i] + offset <= 'Z') ==> (valuesOutput[i] == (valuesInput[i] + offset - 26));
FestSevers)/Gene (/DEBUIC3)	39 @ loop_invariant (\forall int i; 0 <= i && i < loopidx; (valuesInput[i] + offset > 'Z') ==> (valuesOutput[i] == (valuesInput[i] + offset));
Excourceview {{DEB0d}}	40 0
B Info	41 @ loop_invariant (\forall int i; 0 <= i && i < loopidx; 'A' <= valuesOutput[i] && valuesOutput[i] <= 'Z');
Exploration Steps	42 0
Exploration steps	43 @ decreases valuesInput.length - loopidx;
A Proof _ 🖬 🗖	44 @
	45 @ assignable valuesOutput[*];
r)	40 er/
Valid	//@ docume (i)didi int ; ((1 < ())((valuesinput).eng(i)) aw (0 < 1))> (A <- valuesinput[i] aw valuesinput[i] <- 2);
ion (x arr l= null)	A7 while (hondy saturation) (hondy) = (50 + (bisec -1))),
rution (x arr 1 != null)	ds
P	<pre>49 if (valuesInput[loopidx] <= 'Z' - offset) {</pre>
al Execution (x arr 2 != null)	<pre>50 int tmp1 = valuesInput(loopidx] + offset;</pre>
_5 true	51 valuesOutput[loopidx] = (char)tmp1;
_5 false	52
Normal Execution (x_arr_3 != null)	53 } else {
C Normal Execution (x_arr_5 != nul	54
← □ Case 1	<pre>//@ assume (\forall int 1; 0 <= 1 && 1 < loopidx; (valuesInput[1] + offset <= 'Z') ==> (valuesOutput[1] == (valuesInput[1] + offset - 26)));</pre>
• Case 1	//@ assume (Yordall Int 1; 0 <= 1 && 1 < looplax; (Valuesinput[1] * 0158t > 2) ==> (Valuesinput[1] == (Valuesinput[1] * 0158t)));
Case 1	//@ dsbume (\Uldati int i, v <- 1 ok i < 100 µuk, v <- Valuesburbur[i] ok Valuesburbur[i] <- 2),
Case 7	56 valueSutructIonnidu = chartmo?
Q 18711:0PEN GO/	57
Case 2	58 }
Case 2	59
🕨 🖿 Case 2	60
Null Reference (x_arr_5 = null)	//@ assume offset >= 0;
Index Out of Bounds (x_arr_5 !=)	//@ assume offset < 26;
Null Reference (x_arr_3 = null)	//@ assume 0 <= loopidx;
Index Out of Bounds (x_arr_3 l= nul	
ererence (x_arr_2 = null)	<pre>//@ assert (\forall int 1; 0 <= 1 && 1 < looplax; (Valuesinput[1] + offset > 2) ==> (Valuesinput[1] == (Valuesinput[1] + offset)));</pre>
Out of Bounds (x_arr_2 != huil, but	
re bre (x arr 1 = pull)	64 return values/lutrut length
(x arr = null)	65
(66
vertToUpper)	67 /*@ normal_behaviour
	68 @
	69 @ requires valuesInput.length > 0;
Show taclet info (inner nodes only)	Normal Execution (x arr 5 != null)
uncerne (internetes only)	
KgY Strategy: Applied 123 rules (0.0 se	closed 1 goal, 2 remaining

8 *	KeY 2.11.0 [schwoerer/ma-ext-source-v	ew] ~ ^ ×
File View Proof Options Origin Tra	king	About
Run 73 CVC5 Princess	- 43 X E C R B B 🚝 🎜 - 🖉 II Lavouts: Default y Load Lavou	t Save Lavout Reset Lavout Fxnloration Mode Hide justification
🗋 Loaded Proofs 📃 🖬 🗖	Sequent _ 2	🗋 Source 💶 🗖 🗖
Proofs	+ (self valuesInnut@hean)[i]@hean&fter convertTolinner[anon/self valuesOutn	CaesarChiffre java
with model Part_1@3:07:05 PM GassarChiffree(CassarChiffre	anon heap LOOP)	22
H caesarchinetcaesarchine.calcelini	\forall int i;	34 /*0
	(i < loopidx 0 & i >= 0	25 A loop invariant A <= loopidy:
	-> offset	36 0 loop invariant loopidx <= valuesInput length:
	+ (self.valuesInput@heap)[i]@heapAfter_convertToUpper[anon(self.valuesOutpu	37 0
	anon_heap_LOOP)]	38 @ loop invariant (\forall int i: 0 <= i && i < loopidx: (valuesInput[i] + off
(> 90	39 @ loop invariant (\forall int i; 0 <= 1 && i < loopidx; (valuesInput[i] + off
ExtSourceview {{DEBUG}}	-> (self.valuesOutput@heap)[i]@heapAfter_convertToUpper[anon(self.valuesOutpu	40 0
Goals	anon_heap_LOOP)]	41 @ loop_invariant (\forall int i; 0 <= i && i < loopidx; 'A' <= valuesOutput[i]
Fundamentarian States	= offset	42 0
Exploration Steps	+ (self.valuesInput@heap)[1]@heapAfter_convertToUpper[anon(self.valuesOutp	43 @ decreases valuesInput.length - loopidx;
Proof _ 🖬 🗖	anon_neap_LOOP)	44 0
	(Torall Int 1;	45 @ assignable valuesOutput[*];
n	(1 < 100p10x_0 @ 1 /= 0 (calf valueCutout@haan)[i]@haan&ftar_convertTolinner[anon(calf valueCutout@haan)[i]@haan&ftar_convertTolinner[anon(calf valueCutout@haan)]	46 @*/
Valid	anon bean 100	<pre>//@ assume \forall int 1; ((1 < (\old(valuesinput).length)) && (0 <= 1)) ==> ()</pre>
variant	>= 65	<pre>//@ assume !(\old(Valuesinput)[loopldx] <= (90 + (ottset * -1)));</pre>
nution (x_arr != null)	& (self.valuesOutput@heap)(i]@heapAfter convertToUpper(anon(self.valuesOu	47 WILLE (Tooptox < ValuesTiput.Tength){
cution (x_arr_r = null)	anon heap LOO	40 49 if (valuesInnutfloonidy) <= '7' = offset) (
al Execution (x arr. 2 l= null)	<= 90)	50 int tmp1 = valuesInput(loopidx] + offset:
5 true		51 valuesOutputfloopidx] = (char)tmo1:
5 false	<pre>(self.valuesInput@heap)[loopidx_0]@anon_heap_convertToUpper <= 90 + offset * -1,</pre>	52
Normal Execution (x_arr_3 != null)	self.valuesOutput = null,	53 } else {
Sormal Execution (x_arr_5 != nul	<pre>self.valuesInput = null,</pre>	54
🔹 🚍 Case 1	self.valuesOutput = self.valuesInput,	<pre>//@ assume (\forall int i; 0 <= i && i < loopidx; (valuesInput[i] + off</pre>
🕈 📑 Case 1	self = null,	<pre>//@ assume (\forall int i; 0 <= i && i < loopidx; (valuesInput[i] + off</pre>
🕈 🚍 Case 1	(forall int 1;	<pre>//@ assume (\forall int i; 0 <= i && i < loopidx; 'A' <= valuesOutput[i]</pre>
- 🖿 Case 1	(1 >= 0 & 1 < 1 + IOOPIOX_0	<pre>55 int tmp2 = valuesInput[loopidx] + offset - 26;</pre>
← Case 2	 offset colf valuerInput(i)@baapAfter_convertTelloper[appn/colf_valuerOutput \$ 	<pre>56 valuesOutput[loopidx] = (char)tmp2;</pre>
- 9 18/11:OPEN GO/	anon bean LOOP)]	57
Case 2	[self.valuesOutput[]oopidx 0]	58 }
Case 2	(59
Null Reference (y arr 5 = null)		//@ assume offset >= 0:
Index Out of Bounds (x arr 5 !=)	> 90	//@ assume offset < 26:
Null Reference (x arr 3 = null)	self.valuesOutput[i]@heapAfter_convertToUpper[anon(self.valuesOutput.*,	<pre>//@ assume @ <= loopidx;</pre>
Index Out of Bounds (x arr 3 != nul	anon_heap_LOOP)]	61 loopidx++;
eference (x_arr_2 = null)	[self.valuesOutput[loopidx_0]	<pre>//@ assert (\forall int i; 0 <= i && i < loopidx; (valuesInput[i] + offset</pre>
Out of Bounds (x_arr_2 != null, but		62 }
e	- offert	63
hce (x_arr_1 = null)	 Uliset solf valuesInnut(i)@heanAfter_convertTelener[anen/colf_valuesOuteut \$ 	64 return valuesOutput.length;
(x_arr = null)	anon bean LOOP)]	65
Teller est	[self.valuesOutput[loopidx_0	00 67 mm2 hebridaya
vertroopper)	(59 59
		Committee unlustanut leasth > 0.
4 F		
Show taclet info (inner nodes only)		Normal Execution (x_arr_5 != null)
KOX stantage Applied 122 miles (0.0 m	at alread 4 and 3 armaining	E thurston @



Evaluation

- Evaluated with 6 KeY experts
- The work shows:
 - Identifying bugs can be faster, even for KeY experts.
 - Origin/category tracking of formulas is really important.
 - For making more interactions available, we need a better parser.

Ongoing work



"Embedding Proof Scripts into Java/JML Source Code" (Master's thesis) Idea: Write the interactions in form of a script into the source code.



Ongoing work

"Embedding Proof Scripts into Java/JML Source Code" (Master's thesis) Idea: Write the interactions in form of a script into the source code.

```
//@ requires reql: (\exists int x; (\forall int y; p(x,y)));
    //@ ensures ens1: (\forall int v; (\exists int u; p(u,v)));
 2
 3
    void m(int param) {
 4
        if (param > 7) {
 5
            //@ pragma [StrategyProperty]QUERYAXIOM_OPTIONS_KEY=QUERYAXIOM_ON;
 6
            /*@ assert phi \by {
 7
                 var sk1 = req1.skolemize();
              6
 8
              6
                var cutTerm = (\forall int z; pred(z));
 9
              @ assert cutTerm \by {
10
              6
                     . . .
11
              6
                  }:
12
              @ ens1.instantiate(v=sk1.x);
13
              @ ...
14
              @ }:
                                                                           @*/
15
        }
16
                       W Pfeifer: Advancements in User Interface and
```

Institute of Information Security and Dependability (KASTEL)

7/20 August 9, 2023



- Proof Slicing
- Navigation History
- Undoing Interactions
- Automatically run JavaC first (Alexander Weigl)
- Background SMT (ongoing)
- Proof Caching (ongoing)



- Proof Slicing
- Navigation History
- Undoing Interactions
- Automatically run JavaC first (Alexander Weigl)
- Background SMT (ongoing)
- Proof Caching (ongoing)



The year bad (pices. Only) Finding Image: City, Finding, Finding, City, Finding, Findi	8 *		KeY 2.11.0 [main]		
Increase of the second seco	File View Proof Options Origin Tracking				Abo
Image: Control intervention Image: Control interventinterventintervention Image: Control inte				8 m -	
Laster for here Ford	Run CVC5, Princess, Z3 👻 🐿 🕏	- C B	🖬 🛐 := 🗧 🕈 💣 🔹 🕼 🛄 Layouts: Default 🔽 Load Layout Save Layout Reset Layout	Exp	loration Mode Hide Justification
Prod Inter Node Inter Node Inter N	Loaded Proofs	_ 8 0	C Sequent	_ 8 0	🗋 Source 📃 🖬 🕻
<pre>fix with model fix wit</pre>	Proofs		Inner Node		No source loaded
<pre>production productions pr</pre>	Env. with no model				
• Verifities Prof section • Verifities • Verifities	V project key				
Wastes S.D? United S.D? (Live(2D) & All(leg(2D, a)ght(B)) * Verification • Verification • Verification # Proof Sticing * Verification • Verification # Verification • Verific	· Programmy		==>		
<pre>b 1 live (sgitts)</pre>			<pre>\exists S Z/; (liveS(Z/) & killed(Z/, agatha))</pre>		
<pre>l liver(builer) l liver(</pre>			& lives(agatha)		
<pre>l live(christs) l Two d Sining Taylowsins targs l live(christs) l Two d Sining Taylowsins targs l live(christs) l Vorall 5 #; (Vorall 5 #; (Vo</pre>			& lives(butler)		
Prod atis is production target if read is ing Pred staing Pred staing Image: Set (staing (spin), staing) Image: Set (st			& lives(charles)		
Pod sking Proof Sking Image: Second Sking Image: Second Sking Proof Sking Image: Second Sking Image: Second Sking Image: Second Sking Proof Sking Image: Second Sking Image: Second Sking Image: Second Sking Proof Sking Image: Second Sking Image: Second Sking Image: Second Sking Proof Sking Image: Second Sking Image: Second Sking Image: Second Sking Proof Sking Image: Second Sking Image: Second Sking Image: Second Sking Proof Sking Image: Second Sking Image: Second Sking Image: Second Sking Proof Sking Image: Second Sking Image: Second Sking Image: Second Sking Proof Sking Image: Second Sking Image: Second Sking Image: Second Sking Proof Sking Image: Second Sking Image: Second Sking Image: Second Sking Proof Sking Image: Second Sking Image: Second Sking Image: Second Sking Image: Second Sking Proof Sking Image: Second Sking Sing Proof Sking Image: Second Sking Image: Second Sking Image:			& \forall S z8; (lives(z8) -> z8 = agatha (z8 = butler z8 = charles))		
ih prof info if prod Starting if tread is twi; (tread is wi; (tread	Goals Proof Slicing Exploration Steps		& \forall S z9; \forall S z0; (killed(z9, z0) -> hates(z9, z0))		
Proof Sticing 4 Moral 3. Strip (Morales, w3) > Mates (contextes, w3)) Compandency graph 4 Moral 3. Strip (Morals, w3) > Mates (contextes, w3)) Compandency graph 4 Moral 3. Strip (Morals, strip) > Mates (contextes, w3)) Compandency graph 4 Moral 3. Strip (Morals, strip) > Mates (contextes, w3)) Compandency graph 4 Moral 3. Strip (Morals, strip) > Mates (contextes, w3)) Compandency graph 4 Moral 3. Strip (Morals, strip) > Mates (contextes, w3)) Compandency Graph 4 Moral 3. Strip (Morals, strip) > Mates (contextes, w3)) Compandency Graph 4 Moral 3. Strip (Morals, strip) > Mates (contextes, w3)) Compandency Graph 4 Moral 3. Strip (Morals, strip) > Mates (contextes, w3)) Compandency Graph 4 Moral 3. Strip (Morals, strip) > Mates (contextes, w3)) Compandency Graph + Moral 3. Strip (Morals, strip) > Mates (contextes, w3)) Compandency Morals + Moral 3. Strip (Morals, strip) > Mates (contextes, w3)) Moral A Strip (Morals, strip) > Mates (strip) > Mates (stri	Proof Info Proof Search S	trategy	& \forall S w1; \forall S w2; (killed(w1, w2) -> !richer(w1, w2))		
A toreal s wij: (twee builter > hates (gatha, wi)) Dependency graph Graph ndes: 126 Graph edge: 244 A Absreviat Formulas Expert ab DDT Show making of graph Poor analysis 2: Dependency: 25 Show making of graph Poor analysis 2: Dependency: 25 Show making of graph Phoor analysis 2: Dependency: 25 Show making of graph Proof analysis 2: Dependency: 25 Show making of graph Phoor analysis 2: Dependency: 25 Show making of graph Phoor analysis 2: Dependency: 26 Show making of graph Phoor analysis 2: Dependency: 26 Show maker 26 2: Dependency: 26 Show maker 26 3: Dependency: 26 Show maker 26 3: Dependency: 26 Show maker 26 3: Dependency: 26 Show maker 26 Site proof Site proof Site proof Site proof Site proof	Proof Slicing	0_00	& \forall S w3; (hates(agatha, w3) -> lhates(charles, w3))		
A toral 5 w; (tracher w, gatha) -> hates(butler, w0)) Dependency graph Graph nodes: 126 Graph nod		•	<pre>& \forall S w4; (!w4 = butler -> hates(aqatha, w4))</pre>		
A toreal S work (these (agatha, woll) > hates (buckler, woll) Dependency graph Graph des 15.02 Graph des 15.02 Show readering of graph Proof analysis Show readering of graph Proof analysis Dependency matrix Show readering of graph Proof allows Proof allows Dependency matrix Dependency matrix Show readering of spant Proof matrix Dependency matrix Show readering of spant Proof matrix Show readering of spant Show readering of spant Show readering of spant Dependency matrix <td></td> <td></td> <td>& \forall S w5; (!richer(w5, agatha) -> hates(butler, w5))</td> <td></td> <td></td>			& \forall S w5; (!richer(w5, agatha) -> hates(butler, w5))		
Dependency graph Graph nodes: 102 Graph nodes: 102 A torcall is xy: tweiting formula Export a formula Export a formula Show rankering of graph Prof adapta Defendency ranges Defendency ranges Show rankering of graph Prof adapta Defendency ranges Defendency ranges Show rankering of graph Prof adapta Defendency ranges Show rankering of graph Prof adapta Show rankering of graph Show rankering Defendency ranges Show rankering Show r			& \forall S w6: (hates(agatha, w6) -> hates(butler, w6))		
appendency spaph- engin ndex 10 Graph degs 238 A lagatha = buller Abbruik formulas >> killed(agtha, agatha) Epert as 307 >> killed(agtha, agatha) Shorn ndex for a fagatha >> killed(agtha, agatha) Verify tags; 7 >> killed(agtha, agatha) Yend analysis >> De bedjucto na kaptications >> Nor ndex faster: >> Proof sloing >> sloe proof to fixed paint >> No source toaled >>			& \forall S w7: \exists S w8: Ihates(w7, w8)		
Implementary prime >> killed(agstha, agstha) >> killed(agstha, agstha) >> Kore anderig > Shore anderig > Shore anderig > Develoption to splottions > Develoption to splottions > Sice proof Sice	Prove des example		& Japatha = butler		
Implements the formulas Implements the formulas Implements the formulas Implements the formulas Implements the formulas Implements the formulas Show rankets: Implements the formulas Implements: Implements the formulas Implements: Implements the formulas Implements: Implements: Impleme	Create and an ACO		<pre>> killed(agatha agatha)</pre>		
The control is trained i	Graph edges: 102				
Proof and/yis Yourd ranky is Yourd ranky is Yourd ranky is Decloyed ranky is Show rule ranky is No source tasked	Abbraviata formular				
Export an 607 Show makering of graph Poor analysis Teal staps? Teal staps? Operadency: ? Usful brancks: ? Dependency: analysis © Dependency: analysis © Dependency: analysis Silice proof Silice proof <t< td=""><td>Abbreviate formulas</td><td></td><td></td><td></td><td></td></t<>	Abbreviate formulas				
Show rankering of graph Poor analysis Textal staps: 0 Comparison Vietal staps: 0 Comparison Discrete staps Comparison Discrete staps Comparison Discrete staps Comparison Silce proof Silce proof Silce proof Silce proof Silce proof Silce proof Silce proof Silce proof	Export as DOT				
Tod analysis Tod areys, ? Varial steps, ?	Show rendering of graph				
Proof analysis Total steps: 7 Unit of steps: 7 Steps: 7 Of developed to resploytations Steps: 7 Step proof Step proof Silice proof					
Total seps: " Vordent	Proof analysis				
Useful starthets? Total starthets? © Dependency starthets © Dependency star	Total steps: ?				
Tad Jandka? Work Jandka? Dedugitation als applications Read markstations Proof slicing Slice proof to fixed point Slice proof to fixed point Slice proof to fixed point Slice proof to fixed point No source toafed No source toafed	Useful steps: ?				
Under landnes: 7 2 Dependency randysis 2 Dependency randysis 3 De	Total branches: ?				
L: Dependency margins De Dedupication we splicitons Rean analysis Show rule statefficia Site proof to fixed point Site proof to fixed point No source toafed No source toafed	Userul branches: ?				
C Declarations Rean analysis Show role statistics Proof statistics Sile proof Sile proof to fixed paint	Cependency analysis				
Readysis Show rule statefics Pool sliding Silce prodi Silce prodi Silce prodi No source loaded No source loaded	C De-duplicate rule applications				
Show rule staffics Proof staffics Silce proof Silce proof Silce proof Silce proof Silce proof Silce proof No source toafted	Run analysis				
Isher risk talkfittes Food Sling Slike proof to fixed paint Slike proof to fixed paint No source tashed No source tashed					
Prod sliding	Show rule statistics				
No source loaded	Proof sliring				
Site prof to fixed point Site prof to fixed point	all a filler				
Silce prod to fixed paint	Silce proor				
	Slice proof to fixed point				
No source loaded					
No source loaded					
No source loaded					
No source loaded					
No source loaded					
No source loaded					
No source loaded					
No source loaded					
No source loaded					
					No source loaded
					1

<u>i</u> *	KeY 2.11.0 [main]		
ile <u>V</u> iew <u>Proof</u> Options Origin Tracking			Abou
🔘 Run CVC5, Princess, Z3 👻 🗳 🗶 🖻 🕻 🖟	P 🛾 🕅 🚝 ← → 🖉 🕶 🖉 🛄 Layouts: Default 💌 Load Layout Save Layout Reset Layout	් - Exp	loration Mode 🗌 Hide justification
Loaded Proofs _ 🛛 🗖	G Sequent	_ 8 0	Source _ B
Proofs	Inner Node		No source loaded
nv. with no model			
✓ project.key	115		
	<pre>\exists S z7; (lives(z7) & killed(z7, agatha))</pre>		
	& lives(agatha)		
	& lives(butler)		
	& lives(charles)		
Goals Proof Slicing Exploration Steps	& \forall S 20; \forall S 20; (killed(29, 20) -> hates(29, 20)]		
🚓 Proof 🕕 Info 🌣 Proof Search Strategy	& \forall S w1; \forall S w2; (killed(w1, w2) -> !richer(w1, w2))		
Proof Slicing	& \forall S w3; (hates(agatha, w3) -> !hates(charles, w3))		
	& \forall S w4; (1w4 = butler -> hates(agatha, w4))		
Dependency graph	& \forall S w5; (!richer(w5, agatha) -> hates(butler, w5))		
Graph edges: 204	<pre>& \Torall S w0; (hates(agatha, w6) -> hates(butler, w6)) </pre>		
Abbreviate formulas	& langtha = butler		
Export as DOT	-> killed(aqatha, aqatha)		
Show rendering of graph			
show rendering of graph			
Proof analysis			
Total steps: 141			
Useful steps: 75 Total branches: 3			
Useful branches: 3			
C Dependency analysis			
☑ De-duplicate rule applications			
Run analysis			
Show rule statistics			
Proof slicing			
Slice proof			
Slice proof to fixed point			
Execution timings			
Algorithm Time			
0 (total time) 6 ms			
1 Dependency Analysis 3 mg			
1a Danandancy Analysis 2 Ills			
The Dependency Analysis, scarch starting or closed goals 2 lifs			
10 Dependency Analysis, analyse Dranching nodes 0 his			
to bependency Analysis: final mark of useless steps 0 ms			
2 Duplicate Analysis 2 ms			
~ Statistical data gathering 1 ms			
			No source loaded

8 *	KeY 2.11.0 [main]			~ ^ >
File View Proof Options Origin Tracking				Abou
C Run CVC5, Princess, Z3 👻 🐿 🗶 🐸 C 🖬	T 🖬 🔄 := 🗧 👻 🖉 🎹 Layouts: Default 💌 Load Layout Save Layout Reset Layout		ploration Mode Hide justification	
🗋 Loaded Proofs 📃 🖉	C Sequent	_ 8 0	Source	_ 8 0
Proofs	Inner Node		No source loaded	
Env. with no model	lives(z7_0),			
project.key	killed(z7_0, agatha),			
	lives(agatha),			
	lives(butler),			
	lives(charles),			
	z7_0 = charles,			
	\forall S z8; (llives(z8) z8 = agatha z8 = butler z8 = charles),			
Goals Evploration Steps	hates(z7_0, agatha),			
Proof D Info Proof Search Strategy	\forall S z0; (hates(z7_0, z0) Ikilled(charles, z0)),			
🕂 Proof 🗳 _ 🖬 🕻	\forall S z0; (hates(butler, z0) !killed(butler, z0)),			
- 99:cut direct	\forall S z0; (hates(agatha, z0) Ikilled(agatha, z0)),			
CUT: 77.0 = charles TRUE	\forall S z0; (hates(charles, z0) !killed(charles, z0)),			
► ¥ 100:One Step Simplification: 1 rule	\forall S z9; \forall S z0; (hates(z9, z0) !killed(z9, z0)),			
-X 122 true left	\forall S w2; (!killed(z7_0, w2) !richer(z7_0, w2)),			
X 123:applyEq	\forall S w2; (!killed(butler, w2) !richer(butler, w2)),			
- X 124:applyEq	\forall S w1; \forall S w2; (Ikilled(w1, w2) Iricher(w1, w2)),			
× 125 applyEq	\forall S w3; (Ihates(agatha, w3) Ihates(charles, w3)),			
X 126:applyEp	w8_1 = butler,			
- X 127 applyEq	w8_0 = butler,			
- ¥ 128 applyEq	\forall S w4; (w4 = butler hates(agatha, w4)),			
- X 129:applyEq	richer(butler, agatha),			
X 130:applyEq	hates(butler, z7_0),			
- X 131:applyEq	\forall S w5; (hates(butler, w5) richer(w5, agatha)),			
- 132:applyEg	\forall S w6; (Ihates(agatha, w6) hates(butler, w6)),			
X 133:applyEq	\forall S w7; \exists S w8; Ihates(w7, w8)			
- 134:allLeft	==>			
- 135:replace known left	killed(butler, agatha),			
 136:One Step Simplification: 2 rules 	killed(butler, butler),			
137:notLeft	killed(agatha, butler),			
– 138:allLeft	killed(charles, w8_2),			
- 139:eqSymm	richer(charles, agatha),			
- 140:replace_known_right	killed(z7_0, w8_3),			
 141:One Step Simplification: 2 rules 	nates(Z/_0, WS_3),			
- 142:closeFalse	nates(charles, wa_2),			
143:Closed goal	nates(agatha, butler),			
CUT: z7_0 = charles FALSE	nates(butter, butter),			
 101:One Step Simplification: 1 rule 	butler = agatha,			
- 102:cut_direct	kiileo(againa, againa)			
CUT: z7_0 = butler TRUE				
X 103:One Step Simplification: 1 rule				
X 112:true_left				
 X 113:applyEq 				
X 114:applyEq				
 × 115:applyEq 				
🗕 🗙 116:applyEq				
 X 117:applyEq 				
- 🗙 118:eqSymm				
- 119:applyEq			No source loaded	

KgY Strategy: Applied 141 rules (0.3 sec), closed 3 goals, 0 remaining

8 *		KeY 2.11.0 [main]			~ ^ ×
<u>File View Proof Options Origin Tracking</u>					About
🔘 Run CVC5, Princess, Z3 👻 🛃 🛠	- C B B	🛐 🖅 🗲 🔿 🎜 🗸 🖉 🎹 Layouts: Default 🖵 Load Layout Save Layout Reset Layout	ຽ 🗸 📄 Exp'	oration Mode 🔲 Hide justification	
Loaded Proofs	_ = = = n	Sequent	_ 8 0	Source	_ 8 0
Proofs	1	ner Node		No source loaded	
Env. with no model		1jues(27-0)			
project.key		kiled(Z, 0, agatha)			
		lives (agatha).			
		lives(butler).			
		lives(charles),			
		z7_0 = charles,			
		<pre>\forall S z8; (llives(z8) z8 = agatha z8 = butler z8 = charles),</pre>			
Goals Proof Slicing Exploration Steps		hates(z7_0, agatha),			
Proof Unfo DProof Search Stra	ategy	<pre>\forall S z0; (hates(z7_0, z0) !killed(charles, z0)),</pre>			
Proof Slicing	9 _ 8 🗖	<pre>\forall S z0; (hates(butler, z0) !killed(butler, z0)),</pre>			
		<pre>\forall S z0; (hates(agatha, z0) !killed(agatha, z0)),</pre>			
Dependency graph		N forall S z0; (hates(charles, z0) Ikilled(charles, z0)),			
Graph nodes: 162		<pre>\forall S z9; \forall S z0; (hates(z9, z0) [killed(z9, z0)],</pre>			
Abbreviate formular		\forall S w2; ([killed(z7_0, w2)] Iricher(z7_0, w2)),			
Austeviace formulas		(forall S W2; (Ikilied(Dutler, W2) Tricher(Dutler, W2)),			
Export as DOT		(forall 5 w1, (forall 5 w2, (formed w1, w2) finite(w1, w2)), (forall 5 w2, (forate(sortha w2) forac(charles w2))			
Show rendering of graph		w8 1 = butler.			
		W6 0 = butler.			
Proof analysis		<pre>\forall S w4: (w4 = butler hates(agatha, w4)),</pre>			
Useful steps: 75		richer(butler, agatha),			
Total branches: 3		hates(butler, 27_0),			
Useful branches: 3		<pre>\forall S w5; (hates(butler, w5) richer(w5, agatha)),</pre>			
C Dependency analysis		\forall S w6; (Ihates(agatha, w6) hates(butler, w6)),			
C De-duplicate rule applications		<pre>\forall S w7; \exists S w8; Ihates(w7, w8)</pre>			
Run analysis					
Show rule statistics		killed(butler, agatha),			
		kile(buter, buter),			
Proof slicing		killed(charles, w8 2).			
Slice proof		richer(charles, agatha),			
Slice proof to fixed point		killed(z7_0, w8_3),			
		hates(z7_0, w8_3),			
Execution timings		hates(charles, w8_2),			
Algorithm	Time	hates(agatha, butler),			
0 (total time)	6 ms	hates(butler, butler),			
		outrer = agatha,			
1 Dependency Analysis	∠ ms	naaawayaana, ugusha)			
1a Dependency Analysis: search starting @ closed goals	s 2 ms				
1b Dependency Analysis: analyze branching nodes	0 ms				
1c Dependency Analysis: final mark of useless steps	0 ms				
2 Duplicate Analysis	2 ms				
~ Statistical data gathering	1 ms				
				No source loaded	
				0	-1

jie jere for digina Gright Auge and Au	8 *	KeY 2.11.0 [main]	~ ^ X
I code for the form is a constrained of the constrained of	File View Proof Options Origin Tracking		About
□ Loaded Parks - 0 ○ Joure - 0 Park Inter Net Inter Net Comments Inter Net Inter Net Inter Net Inter Net Inter Net Inter Net Inter	0 Run CVC5, Princess, Z3 👻 🗳 🗶 🖆 C 📝	■ 10 III Layouts: Default ▼ Load Layout Save Layout Reset Layout "> ▼	Exploration Mode Hide justification
Production Productin Productin Productin	🗋 Loaded Proofs 📃 🗖 🗖	_ Sequent _ t	Source _ Source
Circuit model Inter(r2.9), spatch, like(r2.2), spatch, r2.2), like(r2.2), like(r2.2), spatch, r2.2), like(r2.2), spatch, r2.2), like(r2.2), lik	Proofs	Inner Node	No source loaded
▶ Prodisor Filed(27,8, agsth), Illes(1971b), Illes(19	Env. with no model	lives(z7_0),	
W Goats Freed Stating Freed Stating Freed Stating W Goats Freed Stating Freed Stating Producting synthe Freed Stating Freed Stating State stating Producti	project.key	killed(z7_0, agatha),	
Internet termine Internet termine Internet termine Internet termine Proof statistic Internet termine Internet termine		lives(agatha),	
#2.# = cduality: #0 cash import Proof ling: Proof stands transport Proof ling: Proof ling: Proof stands transport Proof ling: Proof stands transport Proof ling: Proof ling: Proof stands transport Proof ling: Proof stands transport Proof ling: Proof ling: Proof ling: Proof ling: Proof ling: Proof		lives(butter),	
Viewall s zi; (11/ws(20) zi = agthi zi = butler zi = charles), http://www.statistics//wwww.statistics//www.statistics//www.statistics//www.statistics//www.statistics//www.statistics//www.statistics//www.statistics//www.statistics//www.statistics//www.statistics//www.statistics//www.statistics//www.statistics//www.statistics//www.statistics///www.statistics//www.statistics///www.statistics///www.statistics///www.statistics///www.statistics///www.statistics///www.statistics///www.statistics///www.statistics///www.statistics///www.statistics///www.statistics///www.statistics///www.statistics///wwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwww		z7.0 = charles.	
Per det interge Depondention Steps Provide Stationg Provide Statistics Per det for graph description Provide Statistics Per det for graph description Provide Statistics Provide Statistics Provide Statistics Statistics Provide Statistics Provide Statistics Provide Statistics Statistics Provide Statistics Statistics Provide Statistics Statistatis		\forall S z8; (llives(z8) z8 = agatha z8 = butler z8 = charles),	
in root in root in root in root Proof sings in root in root	Pa Goals	hates(z7_0, agatha),	
Proof sloring Varial 3 26: (Medes (Meller, 26) / Melle (Meller, 26)), Proprioring right- reach hode: 126 Varial 3 26: (Medes (Meller, 26) / Melle (Meller, 26)), Proof sole: 126 Varial 3 26: (Medes (Meller, 26) / Melle (Meller, 26)), Proof sole: 126 Varial 3 26: (Medes (Meller, 26) / Melle (Meller, 26)), Proof sole: 126 Varial 3 26: (Melle (Meller, 26) / Melle (Meller, 26)), Proof sole: 126 Varial 3 26: (Melle (Meller, 26) / Melle (Meller, 26)), Proof sole: 126 Varial 3 26: (Melle (Meller, 26) / Melle (Meller, 26)), Proof sole: 126 Varial 3 26: (Melle (Meller, 26) / Melle (Meller, 26)), Varial 3 26: (Melle (Meller, 26) / Melle (Meller, 26)), Varial 3 26: (Melle (Meller, 26) / Melle (Meller, 26)), Varial 3 26: (Melle (Meller, 26) / Melle (Meller, 26)), Varial 3 26: (Meller (Meller, 26)), Varial 3 26: (Meller (Meller, 26) / Meller, 26), Varial 3 26: (Meller (Meller, 26)), Varial 3 26: (Meller (Meller, 26) / Meller, 26), Varial 3 26: (Meller (Meller, 26)), Varial 3 26: (Meller (Meller, 26) / Meller, 26), Varial 3 26: (Meller (Meller, 26)), Varial 3 26: (Meller (Meller, 26) / Meller, 26), Varial 3 26: (Meller (Meller, 26)), Varial 3 26: (Meller (Meller, 26) / Meller, 26), Varial 3 26: (Meller (Meller, 26)), Varial 4 26: Meller, 26: Meller, 26: Meller, 26: Meller, 2	Proor Vinto V Proor Search Strategy	\forall S z0; (hates(z7_0, z0) Ikilled(charles, z0)),	
Personnersy graph (Marci 1 - Siz: (Marci (Calles: , 20) Milled(Calles: , 20), (Marci 1 - Siz: (Marci 2 - Siz) Marci 1 - Siz: (Marci 2 - Siz), (Proof Slicing Proof Slicing	(Torall S 20; (hates(putler, 20) Killed(putler, 20)),	
impedies: Note complete: Solution in the solution in the solution in the solution is solution is solution in the solution is solution is solution is solution in the solution is solution is solution is solution is solution is solution in the solution is solutis solutis is solution is solution is solutis solution is	Dependency graph	\forall S z0; (hates(charles, z0) !killed(charles, z0)),	
Graph degise 204 \fmail 1 sv2; (killed(z7, 0, w2)), Import an OCT \fmail 1 sv2; (killed(z7, 0, w2)), Bower and statistic \fmail 1 sv2; (killed(z7, 0, w2)), Proof analysis \fmail 1 sv2; (killed(z7, 0, w2)), Tool analysis \fmail sv2; (killed(z7, 0, w2), Tool analysis \fmail sv2; (killed(z7, 0, w2), Tool analysis \fmail sv2; (killed(z7, 0, w2), Stor profe finde paint \fmail sv2; (killed(z7, 0, w2), Tool analysis Tool analysis Stor profe finde paint Theorem on analyze transfing of cool gas 20 Tool analysis find analysis franzing of cool gas 20 Tool analysis find analysis find ananalyze trananalyse find analysis <t< td=""><td>Graph nodes: 162</td><td>\forall S z9; \forall S z0; (hates(z9, z0) !killed(z9, z0)),</td><td></td></t<>	Graph nodes: 162	\forall S z9; \forall S z0; (hates(z9, z0) !killed(z9, z0)),	
I Address and the second s	Graph edges: 204	\forall S w2; (Ikilled(z7_0, w2) Iricher(z7_0, w2)),	
Experience Numerical Suc; (Nelling Suc; (N	Abbreviate formulas	\forall S w2; (Ikilled(butler, w2) Iricher(butler, w2)),	
Bow mole statistic w8_1 = buller, w8_2 = buller, w6_2 = buller, w6	Export as DOT	\forall S w3: (Ibates(agatha, w3) Ibates(charles, w3)).	
Prod malysis Test isses: 10 Test isses: 10 Test isses: 10 View 11 S wit; (with = buller (hetes (sgatha, wil)), View 11 S wit; (with = buller (hetes (sgatha, wil)), View 11 S wit; (with the sgatha) View 11 S with the sgatha) View 11 S w	Show rendering of graph	w8_1 = butler,	
Ticket size: 14 Ticket (sublet, r. agatha). Ticket (sublet, r. agatha). Ticket (sublet, r. agatha). Subserved. Ticket (sublet, r. agatha). Subserved. Ticket (sublet, r. balt). Subserved. Tic	Proof analysis	<pre>wo_0 = Duller, \forall S wd: (wd = butler bates(anatha wd))</pre>	
Total branches: 3 uster blanches: 3 uster blanches: 3 uster blanches: 4 2 Deduglaten dan apjatians hates (butler, x ⁰) rickr(wS, agatha)), Vforall 5 wS; (htates (gatha, wO) hates (butler, wS), Vforall 5 wS; (htates (gatha, butler), killed (butler, gatha), killed (butler, gatha), killed (butler, w2,), hilted (butler, w2,), hates (gatha, butler), hates (gatha, butler), butler = agatha, killed (agatha, agatha) 1 Dependency Atalysis search starting @ closed gats 2 ms 1 Dependency Atalysis man/ps branching node 0 ms 2 Dapikate Analysis 0 ms 2 Dapikate Analysis 0 ms 2 Statistical data gathering 1 ms 2 ms - Statistical data gathering 1 ms 2 ms - Statistical data gathering 1 ms 5 ms	Useful steps: 75	richer(butler, agatha),	
Used bandles 3 Yerall 5 w5; (hates (buller, w5)] richer(x5, agatha)), Z begendeny analysis Z begendeny Analysis Site prod	Total branches: 3	hates(butler, z7_0),	
U departement analysis Show rule statistics Poder diaries messis Show rule statistics Poder diaries messis Site provide finds point Statistics Algorithm 0 (that intelligence in the statistics Algorithm 1 Dependency Analysis 1 Dependency Analysis 1 Dependency Analysis 2 Depundency Analysis 2 Depundency Analysis 2 Depundency Analysis 2 Depundency Analysis 3 State provide find a gathering 1 Dependency Analysis 2 Dupulcate Analysis 2 Dupulcate Analysis 1 Dependency Analysis 1 Dependency Analysis 2 Dupulcate Analysis 2 Dupulcate Analysis 2 Dupulcate Analysis 2 Dupulcate Anal	Useful branches: 3	<pre>\forall S w5; (hates(butler, w5) richer(w5, agatha)),</pre>	
Description Construction Bine model Since model Since model Since model Algorithm Time Diapendency Analysis: search starting @ closed gasis Time Since model Since model Since model Since model Dependency Analysis: search starting @ closed gasis Time Since model Time <td>Dependency analysis</td> <td>\forall S wb; (Inates(agatha, wb) nates(butler, wb)),</td> <td></td>	Dependency analysis	\forall S wb; (Inates(agatha, wb) nates(butler, wb)),	
Ref Ref Hilled(buller, agatha), Biser prod Hilled(buller, agatha), Sile prod Hilled(buller, buller), Sile for diagonal Hilled(buller, buller), Sile prod Hilled(buller, buller),	De-duplicate rule applications	=>	
Shor rod Killed (buller, buller), Shor rod Killed (saptha, buller), Killed (saptha, buller), Killed (saptha, buller), Killed (saptha, buller), Killed (saptha, buller), Killed (saptha, buller), Killed (saptha, buller), Agorthm The 0 (total time) 0 ms 1 Dependency Analysis: search starting @ closed gast ; total (saptha, agatha) 1 Dependency Analysis: nanzhring nodes 0 ms 1 Dependency Analysis: nanzhring nodes 0 ms 2 Dapizota Analysis 2 ms - Statistical data gathering 1 ms	Run analysis	killed(butler, agatha),	
Fired stains till de groß Siles prod to find pain. till de (charles, sq.2), Siles prod to find pain. till de (charles, sq.2), Agorthm Time 0 dotal time) 0 ms 1 Dependency Analysis carch starting @ closed gasis 2 ms 1 Dependency Analysis starting @ closed gasis 2 ms 1 Dependency Analysis starting @ closed gasis 2 ms 2 Dupitack Analysis 2 Dupitack Analysis 3 Dupitack Analysis mass - Statistical data gathering 1 ms	Show rule statistics	killed(butler, butler), killed(apatha, butler).	
Site prod Ticher (charles, agatha), Site prod Ticher (charles, agatha), Site prod Nates (27, 9, w3, 3), Agorithm Time 0 (total time) 0 ms 1 Dependency Analysis 2 ms 1 Dependency Analysis search starting @ closed goals 2 ms 2 Dupiticate Analysis 2 ms - Statistical data gathering 1 ms	Proof slicing	killed(charles, w8_2),	
Site provide fixed point It life(2(2,9, w3,), hotes(charles, w3	Slice proof	richer(charles, agatha),	
interest minings hates (chulles, id_2), hates (chulles, id_agatha, butler), hates (chulles, id_agatha, id_agatha) 1 Dependency Analysis: mailyze branching nodes 0 ms ic Dependency Analysis: mailyze branching nodes 0 ms ic Dependency Analysis: mailyze branching nodes 0 ms is chulles in the intervention of the in	Slice proof to fixed point	klied(z7_0, w8_3), hates(z7_0, w8_3),	
Agorithm Time hates (agatha, butler), butler = agatha, butler = a	Execution timings	hates(charles, w8_2),	
0 float tamb 6 ma tutiler = sagetas, 1 Dependency Analysis 2 ma 1 Dependency Analysis search starting @ closed gas 1 1 ma 1 Dependency Analysis manyze branching nodes 0 ma 1 Dependency Analysis manyze branching nodes 0 ma 1 Dependency Analysis manyze branching nodes 0 ma 2 Dupitzer Kanalysis 1 ma	Algorithm Time	hates(agatha, butler),	
1 Dependency Analysis 2 ms 1 a Dependency Analysis canalyze branching odosed galas 2 ms 1 Dependency Analysis canalyze branching nodes 0 ms 2 Dependency Analysis malyze branching nodes 0 ms 2 Duplicate Analysis 0 ms - Statistical data gathering 1 ms	0 (total time) 6 ms	butler = agatha.	
la Dependency Analysis exacti starting @ Closed golas 2 ms lb Dependency Analysis: analyze branching node 0 ms 1 Dependency Analysis: final mark of useless steps 0 ms 2 Duplicate Analysis - Statistical data gathering 1 ms - United Statistical data gathering Not source Isaded	1 Dependency Analysis 2 ms	killed(agatha, agatha)	
Ib Dependency Analysis enangize branching nos 0 ms Ic Dependency Analysis: final mark of useless steps 0 ms 2 Duplicate Analysis (and anges steps 0 ms - Statistical data gathering 1 ms	1a Dependency Analysis: search starting @ closed goals 2 ms		
1 C Pependency Analysis final mark of useless steps 0 ms 2 Duplicate Analysis 2 ms - Statistical data gathering 1 ms - University of the steps of	1b Dependency Analysis: analyze branching nodes 0 ms		
2 Duplicate Analysis 2 ms - Statistical data gathering 1 ms - University of the source Isaded	1c Dependency Analysis: final mark of useless steps 0 ms		
- Statistical data gathering 1 ms	2 Duplicate Analysis 2 ms		
No source loaded	~ Statistical data gathering 1 ms		
No source loaded			
			No source loaded

8 *	KeY 2.11.0 [main]	~ ^ X
File View Proof Options Origin Tracking		About
Run CVC5, Princess, Z3 👻 🕰 🔍	C 🖬 🗟 := 🗧 🖻 🖬 🐨 🐨 🐨 🛄 Layouts: Default 💌 Load Layout Save	ayout Reset Layout Stration
Loaded Proofs	Sequent	_ 🛛 🗖 📋 Source _ 🖾 🗖
Proofs	Closed Goal	No source loaded
Env. with no model	lives(77.0).	
✓ project.key	killed(z7 0. agatha).	
project slice1.proof	lives(agatha).	
	lives(butler).	
	lives(charles),	
	z7 0 = charles.	
	z7_0 = agatha z7_0 = butler true,	
Goals Proof Slicing Exploration Steps	\forall S z8; (llives(z8) z8 = agatha z8 = butler z8 = charles),	
Proof O Info Proof Search Strategy	hates(charles, agatha),	
ch Proof 🗴 _	Image: Solution of the state	
44 replace known left	Intersection (1998)	
45:One Sten Simplification: 2 rules	<pre>\forall S w2; (lkilled(z7_0, w2) !richer(z7_0, w2)),</pre>	
- 46:notLeft	\forall S w1; \forall S w2; (Ikilled(w1, w2) Iricher(w1, w2)),	
- 47:alli eft	<pre>\forall S w3; (Ihates(agatha, w3) Ihates(charles, w3)),</pre>	
- 48 replace known right	w8_0 = butler,	
49:One Step Simplification: 1 rule	false,	
- 50:applyEg	<pre>\forall S w4; (w4 = butler hates(agatha, w4)),</pre>	
- 51;allLeft	hates(butler, z7_0),	
- 52:replace known right	<pre>\forall S w5; (hates(butler, w5) richer(w5, agatha)),</pre>	
 53:One Step Simplification: 1 rule 	<pre>\forall S w6; (Ihates(agatha, w6) hates(butler, w6)),</pre>	
- 54:allLeft	\forall S w7; \exists S w8; Ihates(w7, w8)	
- 55:replace_known_left	==>	
56:One Step Simplification: 2 rules	hates(agatha, agatha),	
- 57:cut_direct	richer(z/_0, agatha),	
CUT: z7_0 = charles TRUE	hates(agatha, w8_0),	
- 58:applyEq	nates(butler, butler),	
- 60:allLeft	butler = agatha,	
 61:replace_known_left 	Kiiieu(againa, againa)	
 62:One Step Simplification: 2 rules 		
63:notLeft		
64:allLeft		
- 65:eqSymm		
66:replace_known_right		
 67:One Step Simplification: 2 rules 		
68:closeFalse		
- ✓ 69:Closed goal		
CUT: Z7_0 = charles FALSE		
 Secone step simplification: 1 rule 20 sub direct 		
 De CUT: 27.0 = butler TRUE 	=	
- 71 applyEq		
71:dppiycq 72:doso		
73.close		
CUT: 27.0 = butter FALSE		
72:One Sten Simplification: 1 rule		
- 75:applyEq		
- 76:close		
77:Closed goal		No source loaded



Evaluation



- Often, very large parts of proofs could be removed.
- Trend: The larger the proof, the larger the percentage.
- Most of the removed steps are normalizations of formulas which are never used later on.

Further Applications of the Dependency Analysis



- Rule de-duplication (implemented):
 - If the same rule is applied to the same formula(s) in two branches, it possibly can be moved in front of the branching rule.

Further Applications of the Dependency Analysis



- Rule de-duplication (implemented):
 - If the same rule is applied to the same formula(s) in two branches, it possibly can be moved in front of the branching rule.
- Slice w.r.t. a selected formula (implemented):
 - "Which steps produced this formula?"
 - "Which input formulas are needed to derive this formula?"

Further Applications of the Dependency Analysis



- Rule de-duplication (implemented):
 - If the same rule is applied to the same formula(s) in two branches, it possibly can be moved in front of the branching rule.
- Slice w.r.t. a selected formula (implemented):
 - "Which steps produced this formula?"
 - "Which input formulas are needed to derive this formula?"
- Proof Reordering (ongoing work):
 - Group certain rule applications (similar to One-Step-Simplification)
 - Possible categories: NNF, Polynomial simplification, ...



Proof Slicing	\checkmark
Navigation History	◄
Undoing Interactions	◄
Automatically run JavaC first (Alexander Weigl)	
Background SMT (ongoing)	
Proof Caching (ongoing)	

8 *	KeY 2.11.0 [main]	× ^
File View Proof Options Origin Tracking		A
🔘 Run CVC5, Princess, Z3 👻 🕊 🗶	C C B B E E + I C II Layouts: Default - Load Layout Save Layout Reset Layout	D - Exploration Mode Hide justification
Loaded Proofs	_ 🛛 🗖 🖸 Sequent	_ 🖬 🗖 🗋 Source _ 🗖
Proofs	Closed Goal	No source loaded
Env. with no model	lives(z7 0).	
✓ project.key	killed(z7 0, agatha).	
project_slice1.proof	lives(agatha),	
	lives(butler),	
	lives(charles),	
	z7_θ = charles,	
	z7_0 = agatha z7_0 = butler true,	
Goals Proof Slicing Exploration Steps	\forall S z8; (llives(z8) z8 = agatha z8 = butler z8 = charles),	
Proof O Info Proof Search Strates	W hates(charles, agatha),	
💠 Proof 🏟 .	_ ■ \forall S z0; (hates(z7_0, z0) Ikilled(z7_0, z0)),	
44:replace known left	\forall S z9; \forall S z0; (hates(z9, z0) Ikilled(z9, z0)),	
+ 45:One Step Simplification: 2 rules	\forall S w2; (Ikilled(z7_0, w2) Iricher(z7_0, w2)),	
- 46:notLeft	<pre>\forall S w1; \forall S w2; (!killed(w1, w2) !richer(w1, w2)),</pre>	
– 47:allLeft	<pre>\forall S w3; (Ihates(agatha, w3) Ihates(charles, w3)),</pre>	
 48:replace_known_right 	w8_0 = butler,	
49:One Step Simplification: 1 rule	Talse,	
- 50:applyEq	(hotails s W4; (W4 = buller hates(agatha, W4)),	
– 51:allLeft	hates(buller, 2/_0),	
 52:replace_known_right 	(forall S w); (hates(bullet, w)) itchef(w), dgd(a)), (forall S w); (hates(bullet, w) bates(bullet, w6))	
 53:One Step Simplification: 1 rule 	\forall S w7: Vexists S w8: Hotes(w2)	
- 54:allLeft	==>	
55:replace_known_left	hates(agatha, agatha).	
 Source step simplification: 2 rules Source step simplification: 2 rules 	richer(z7_0, agatha),	
Cuttor and a charles TRUE	hates(agatha, w8_0),	
58 applyEq	hates(butler, butler),	
- 60 all eft	butler = agatha,	
- 61:replace known left	killed(agatha, agatha)	
 62:One Step Simplification: 2 rules 		
- 63:notLeft		
- 64:allLeft		
- 65:eqSymm		
 66:replace_known_right 		
67:One Step Simplification: 2 rules		
68:closeFalse		
- V 69:Closed goal		
CUT: z7_0 = charles FALSE		
 59:One Step Simplification: 1 rule 		
- /u:cut_direct	-	
Therefore a contract of the co		
- 73:close		
✓ 74:Closed goal		
CUT: z7 0 = butler FALSE		
72:One Step Simplification: 1 rule		
- 75:applyEq		
- 76:close		
✓ 77:Closed goal	v	No source loaded

8 *	Key 2.11.0 [main]	~ ^ X
File View Proof Options Origin Tracking		About
0 Pun CV/C5 Princers 72 - 47 %		0.0
Loaded Proofs	_ 🖸 🗋 Sequent _ 💆 🗋 Source	_ 2 0
Proofs	Closed Goal No source loaded	
Env. with no model	lives(z7_0),	
✓ project.key	killed(z7_0, agatha),	
 project_slice1.proof 	lives(agatha),	
	lives(butler),	
	lives(charles),	
	z7_0 = charles,	
(but each Y = prostellation / functionalise stress)	z7_0 = agatha z7_0 = butler true,	
Boolis Proof Slicing Exploration Steps	\forall S z8; (llives(z8) z8 = agatha z8 = butler z8 = charles),	
Proof Unfo UProof Search Stra	99 hates(charles, agatha),	
+ Proof	_ S □ \\forall S z0; (hates(z7_0, z0) lkilled(z7_0, z0)),	
- 44:replace_known_left	Vorali 5 z9; Vorali 5 z0; (hates(z9, z0) Killed(z9, z0)),	
45:One Step Simplification: 2 rules	\forall S w2; (lkilled(z7_0, w2) lricher(z7_0, w2)),	
- 46:notLeft	<pre>\forall S w1; \forall S w2; (lkiled(w1, w2) fricher(w1, w2)),</pre>	
– 47:allLeft	(forall S w3; (Ihates(agatha, w3) Ihates(charles, w3)),	
 48:replace_known_right 	w8_0 = butler,	
49:One Step Simplification: 1 rule	false,	
- 50:applyEq	<pre>\torail S w4; (w4 = butler nates(agatha, w4)), below to be a state of the st</pre>	
- 51:allLeft	nates (burler, z_0),	
 52:replace_known_right 	(torall S w5; (hates(buller, w5) richer(w5, agatha)),	
 53:One Step Simplification: 1 rule 	(Torall S Wo; (Indes(agata, wo) nates(burler, wo)),	
– 54:allLeft	(TOTALL S W/; VEXISTS S WG; THALES(W/, WG)	
 55:replace_known_left 		
► 56:One Step Simplification: 2 rules	nates (agarna), agarna),	
- 57:cut_direct	Inter(1/2, algoria),	
CUT: z7_0 = charles TRUE	hates(algatha, wo_0),	
- 58:applyEq	hates (bite), bites,	
- 60:allLeft	billed(astha astha)	
 61:replace_known_left 	naacu uguchu, uguchu,	
 62:One Step Simplification: 2 rules 		
- 63:notLeft		
- 64:allLeft		
- 65:eqSymm		
- 66:replace_known_right		
67:One step simplification: 2 rules		
- oscioseraise		
CUT: 77.0 = charles EALSE		
50 One Step Simplification: 1 rule		
- 70 cut direct		
CUT: 77.0 = butler TRUE		
- 71 applyEq		
- 73:close		
✓ 74:Closed goal		
CUT: z7 0 = butler FALSE		
72:One Step Simplification: 1 rule		
- 75:applyEq		
- 76:close		
✓ 77:Closed goal	V No source loaded	



Proof Slicing	\checkmark
Navigation History	\checkmark
Undoing Interactions	\checkmark
Automatically run JavaC first (Alexander Weigl)	•
Background SMT (ongoing)	
Proof Caching (ongoing)	







Proof Slicing	\checkmark
Navigation History	\checkmark
Undoing Interactions	\checkmark
Automatically run JavaC first (Alexander Weigl)	\checkmark
Background SMT (ongoing)	•
Proof Caching (ongoing)	

in ywe for degines digen degin Image: the set of the	8 *	KeY 2.11.0 [appelhagen/backgroundSMT]	~ ^ ×
<pre> two control is a cont</pre>	Eile View Proof Options Origin Tracking		About
<pre>lighted room</pre>	O Run CVC5, Z3 → 🗳 🛠 📂 C 🗭 🖻 🖾	Æ ← → S ▼ Ø III Layouts: Default ▼ Load Layout Save Layout Reset Layout D ▼ Exploration	Mode 🔲 Hide justification
<pre>tred: moduling(=12) // moduling(=12) // modul</pre>	Loaded Proofs _ 2 C	G Sequent _ 🛛 🗆	🗋 Source 📃 🖬 🗖
<pre>min model (geld J) Ministreme (geld J) Mi</pre>	Proofs	Inner Node	SumAndMax.java
<pre> A Candid The Acceleration Experiment of Control and Control Acceleration State Acceleration State and Control Acceleration State and Control Acceleration State and Control Acceleration State Acceleration Stat</pre>	.with model src@c44:23 PM @ SumAndMax(SumAndMax::sumAndMax([0),[ML normal_behavior operation /	wellFormed(heap) A self = mull A self = mull	1 class SumAndMax { 2 3 int sum; 4 int max; 5
Bis Guest Provide • (a = null * a < created: = TNUE)		SumAndMax::exactInstance(self) = TRUE	6 /*@ normal_behaviour
<pre> the second se</pre>	Goals Proof Slicing Exploration Steps	<pre>(a = null v a.<created> = TRUE)</created></pre>	7 @ requires (\forall int
A reg dir A (Y int 1; (0 ≤ 1 ∧ 1 ≤ a length ∧ infint(1) = 0 ≤ a[1] ∧ (self-sime ∧ s = null)) 9 0 <td>The Proof Info Proof Search Strategy</td> <td>A measuredByEmpty</td> <td>8 @ assignable sum, max;</td>	The Proof Info Proof Search Strategy	A measuredByEmpty	8 @ assignable sum, max;
<pre>converting converting conver</pre>	📩 Proof 🌼 📮 🖬 🗖	A (♥ int i; (0 ≤ i A i < a.length A inInt(i) → 0 ≤ a[i]) A (self. <inv> A ¬a = null))</inv>	10 0 ensures (a length > 0
<pre>class :</pre>	C1 Proof Tree	- {heapAtPre:=heap _a:=a}	11 0 ==> (\exists
<pre> Case 1</pre>	+ CI Invariant Initially Valid	ave = null:	12 @ ensures sum == (\sum :
<pre> Case 1</pre>	+ 🗂 Case 1		13 @ ensures sum <= a.leng
<pre>vidi dundhäkkint[]) { vidi Cast vidi Cas</pre>	🕈 📺 Case 1	self.sumAndMax(a)@SumAndMax;	14 @*/
<pre>sec = s;</pre>		<pre>} catch (java.lang.Throwable e) {</pre>	<pre>15 void sumAndMax(int[] a) {</pre>
<pre></pre>		exc = e;	16 sum = 0;
<pre>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>></pre>	• Case 1	3	17 max = 0;
<pre></pre>	Case I	<pre>}\> (\Vec int i; (0 \le i \Lambda i < a.length \Lambda inInt(i) - a[i] \le self.max)</pre>	10 Inc K - 0;
<pre></pre>	+ Cale 2	A ((a.length > 0 - ± int 1; (0 ≤ 1 A 1 < a.length A inint(1) A seit.max = a[1])) A ((calf run = house(int i) (0 = locath atil) + (calf run = a locath + calf run = house(int i) (a.locath atil) + (calf run = a)	20 /*P loop invariant
• • • Case 2 • • • Case 2 • • • • • • • • • • • • • • •	Apply background SMT results.	<pre>A (sell.sum = bsum{int 1;}(0, d.tength, d[1]) A (sell.sum ≤ d.tength * sell.max A sell.sum>))) A ever = null</pre>	21 @ 0 <= k && k <= a.
<pre></pre>		A Y Field f	22 @ && (\forall int i
<pre></pre>	R 1210:OPEN GOAL	¥ java.lang.Object o:	23 @ && (k == 0 ==> ma
<pre> Case 2</pre>	Case 2	<pre>((o, f) e {(self, SumAndMax::\$sum)} ∪ {(self, SumAndMax::\$max)}</pre>	24 @ && (k > 0 ==> (\e
	★ Case 2	v no = null	25 @ && sum == (\sum i
<pre></pre>	P G 1194:OPEN GOAL	A ¬o. <created>@heapAtPre = TRUE</created>	26 @ && sum <= k * max
C = C = 2 C = 2	Case 2 O 1187/OPEN GOAL	v o.f = o.f@heapAtPre))	27 e
Image: Spin Section QA 30 0*/***********************************	T Case 2		20 e assignable son, na 29 e decreases a length
Climatin the served and Used () Normal Secution (a = null) (Q 1180:OPEN GOAL		30 0*/
Commal Execution (a le null)	 Invariant Preserved and Used 		31 while(k < a.length) {
• if b_1 true • if b_1 true 33 max = a[k]; • if b_2 true 33 sum = a[k]; • if b_2 true 34 sum = a[k]; • if b_2 true 34 sum = a[k]; • if b_2 true 34 sum = a[k]; • if b_2 true 10 sum = a[k]; • if b_2 true <td> In Normal Execution (_a != null) </td> <td></td> <td>32 if(max < a[k]) {</td>	 In Normal Execution (_a != null) 		32 if(max < a[k]) {
• • Mormal Execution (a = null) 34) • • • Mormal Execution (a = null) 35 sum == a[k]; • • • Mormal Execution (a = null) 36 k++; • • • Mormal Execution (a = null) 36 k++; • • • Mormal Execution (a = null) 38 33 • • • Mormal Execution (a = null) 38 33 • • • Mormal Execution (a = null) 38 33 • • • Mormal Execution (a = null) 38 33 • • • Mormal Execution (a = null) 38 33 • • • Mult Reference (a = null) 40 40 • • • • • • • • • • • • • • • • • • •			33 max = a[k];
• (* If b, 2 true 33 s.un *= a[k]; • (* If b, 2 true 33 s.un *= a[k]; • (* If b, 2 true 33 s.un *= a[k]; • (* If b, 2 true 33 s.un *= a[k]; • (* If b, 2 true 33 s.un *= a[k]; • (* If b, 2 true 33 s.un *= a[k]; • (* If b, 2 true 33 s.un *= a[k]; • (* If b, 2 true 33 s.un *= a[k]; • (* If b, 2 true 33 s.un *= a[k]; • (* If b, 2 true 33 s.un *= a[k]; • (* If b, 2 true 33 s.un *= a[k]; • (* If b, 2 true 33 s.un *= a[k]; • (* If b, 2 true 34 35 • (* If b, 2 true 34 34 • (* If b, 2 true 35 • (* If b, 2 true 34 • (* If b, 2 true 35 • (* If b, 2 true 36 • (* If b, 2 true 36 • (* If b, 2 true 37 <td< td=""><td> Image: mail of the security of th</td><td></td><td>34 }</td></td<>	 Image: mail of the security of th		34 }
• @ vortine Execution (a) is mult) 30 x** • @ vortine Execution (a) is mult) • @ vortine Execution (a) is mult) 33 • Worth Execution (a) is mult) 33 • Worth Execution (a) is mult) 33 • @ vortine Execution (a) is mult) 34 • @ vortine Execution (a) is mult) 44 <	+ □ if b_2 true		35 sum += a[k];
Image: Section (a) = null) 33 Image: Section (b) = 1 = null) 34 Image: Section (b) = 1 = null) 34 Image: Section (b) = 1 = null) 34 Image: Section (b) = null) 34 Image: Section (b) = null) 35 Image: Section (b) = null) 36 Image: Section (b) = null) 37 Image: Section (b) = null) 36 Image: Section (b) = null) 37 Image: Section (b) = null) 38 Image: Section (b) = null) 37 Image: Section (b) = null) 37 Image: Section (b) = null) 37 Image: Section (b) = null) 38 Image: Section (b)	 Normal Execution (a = null) 		27 L
• • • • • • • • • • • • • • • • • • •	Normal Execution (s = null)		38 3
Image: State	In Normal Execution (s 1 != null)		39 }
	G 1172:OPEN GOAL		40
	Null Reference (s_1 = null)		
Constraint of the state of	Null Reference (_a = null)		
Vige (Viscovin Volut) Viscovin Volut)	Index Out of Bounds (_a != null, but k)		
	P & 1048:OPEN GOAL		
the investment of the first sector of Bounds (a) = null, but K Out of a The sector of Bounds (a) = null, but K Out of a The sector of Bounds (a) = null, but K Out of a	Null Reference (s = null)		
Show Patternition/Asignable	 Gindex Out of Bounds (a la pull, but k Out of - 		•
The set of			Show Postcondition/Assignable
R Z Javac (0) bSMT			
	K ^B Y		📕 🖬 Javac (0) bSMT 🤇

8 *	KeY 2.11.0 [appelhagen/backgroundSMT]	~ ^ ×
Eile View Proof Options Origin Tracking		About
O Run CVC5, Z3 → 🛃 🛠 🖕 C 🗭 🖻 🖻	Æ ← →	Exploration Mode Hide justification
🗋 Loaded Proofs 📃 🗖 🗖	Sequent	_ 🖬 🗖 📋 Source _ 🛤 🗖
Proofs	Closed Goal	SumAndMax java
Loaded Proof Loaded Loaded Proof Loaded Loaded Proof Loaded Loadedd Lo	<pre>is Sequent Come Legon _ L</pre>	■ Cloud and mode
 if b_1 true if b_1 true if b_2 t		1 1 (max = a(k)) [3 max = a(k); 35 sm ≠ = a(k); 36 k++; 37) 38) 39] 40

8 *	KeY 2.11.0 [appelhagen/backgroundSMT]	~ ^ ×
Eile View Proof Options Origin Tracking		About
🔹 Run CVC5, Z3 🗸 🖼 🛠 🖆 🖸 🔂	E C > C - C - C - C - C - C - C - C - C -	Mode 🔲 Hide justification
🗋 Loaded Proofs 🛛 🖉 🗖	🛾 Sequent _ 🛛 🗖	🗋 Source _ 🗳 🗖
Proofs	Inner Node	SumAndMax java
. with model src@6:48:23 PM		1 alass Suntration (
SumAndMax[SumAndMax:sumAndMax([])]JML normal_behavior operation co	-	2
	anlifermed(hose)	3 int sum;
	A self = mull	4 int max;
	A self.screated> = TRUE	5
	<pre>A SumAndMax::exactInstance(self) = TRUE</pre>	6 /*@ normal_behaviour
Goals Proof Slicing Exploration Steps	<pre>A (a = null v a.<created> = TRUE)</created></pre>	7 @ requires (\forall int
📅 Proof 🚯 Info 🍄 Proof Search Strategy	∧ measuredByEmpty	8 @ assignable sum, max;
da Proof 🗢 🖉 🗆	$ (\forall int i; (0 \le i \land i < a.length \land inInt(i) \rightarrow 0 \le a[i]) \land (self. < inv> \land \neg a = null) $	9 e ensures (\torall int)
C ¹ Proof Tree	<pre>- {heapAtPre:=heap _a:=a}</pre>	11 0 ==> (\exists i
←	eve = mill:	12 @ ensures sun == (\sum 1
+ Case 1	try /	13 @ ensures sum <= a.lengt
🕶 📺 Case 1	self.sumAndMax(_a)@SumAndMax;	14 @*/
👻 🚍 Case 1	} catch (java.lang.Throwable e) {	<pre>15 void sumAndMax(int[] a) {</pre>
←	exc = e;	16 sum = 0;
• Case 1	3	17 max = 0;
Case I	}\> (♥ int i; (0 ≤ i ∧ i < a.length ∧ inInt(i) → a[i] ≤ self.max)	10 Inc K = 0;
Case 2	A ((a.length > 0 → ∃ int 1; (0 ≤ 1 ∧ 1 < a.length ∧ inInt(1) ∧ self.max = a[1]))	20 /*@ loop invariant
	<pre>A (set.sum = osum(int i;)(0, a.tength, a[i]) A (set.sum s a.tength - set.max A set.<inv>))) A eve = oull</inv></pre>	21 @ 0 <= k && k <= a.]
+ Case 2	A W Field f	22 @ && (\forall int i
- 🕨 🔓 1210:OPEN GOAL	¥ java.lang.Object o;	23 @ && (k == 0 ==> max
Case 2	<pre>((o, f) e {(self, SumAndMax::\$sum)} ∪ {(self, SumAndMax::\$max)}</pre>	24 @ && (k > 0 ==> (\e)
★ Case 2	v ¬o = null	25 @ && sun == (\sun ir
Y 1194:OPEN GOAL	A -o. <created>@heapAtPre = TRUE</created>	20 @ && SUM <= K * max;
G 1187 OPEN GOAL	v o.f = o.f@heapAtPre))	22 e accimable cum may
T Case 2		29 0 decreases a length
R 1180:OPEN GOAL		30 @*/
 Invariant Preserved and Used 		31 while(k < a.length) {
 Image: mail and the second of t		32 if(max < a[k]) {
		33 max = a[k];
 — Mormal Execution (_a != null) 		34 }
 If D_2 true Normal Execution (a l= null) 		35 Sum T= a[K];
Indimini Execution (s = null)		37 }
 Normal Execution (a != null) 		38 }
In the security of the secu		39 }
유 1172:OPEN GOAL		40
Null Reference (s_1 = null)		
Null Reference (_a = null)		
 Index Out of Bounds (_a != null, but k (
► WILl Reference (s = pull)		
Null Reference (a = null)		
 Index Out of Bounds (_a != null, but k Out of) 		
		Show Postcondition/Assignable

📕 🖬 Javac (0) bSM 🏠 Ø

KBX

£ *	KeY 2.11.0 [appelhagen/backgroundSMT]	
Eile View Proof Options Origin Tracking		Abou
	8 🖅 🖌 之 🦉 – 🧉 🎹 Langute Default y Lond Langut Save Langut Beret Langu	
🗋 Loaded Proofs 📃 🗖	Gequent	_ 🛛 🗖 🖸 Source _ 🗠 🗖
Proofs	Inner Node	SumAndMax.java
with model strop oxes 23 PM SumandMay/SumandMay/SUMay/SUMay/SUMandMay/SUMandMay/SUMandMay/SUMandMay/SUMay/SUMa	10	1 class SumAndMax {
R contraction and an analysis and a fully the second of the second		2
	wellFormed(heap)	3 int sum;
	<pre>^ self = null</pre>	4 int max;
	<pre>> A self.<created> = TRUE</created></pre>	5 /*0 normal behaviour
	A SUMANDMAX::exactinstance(self) = TRUE	7 @ requires ()forall int
Broof States	A measuredByEmpty	8 @ assignable sum, max;
Proor Search Strategy	▲ (♥ int i; (0 ≤ i ∧ i < a.length ∧ inInt(i) → 0 ≤ a(i)) ∧ (self. <inv> ∧ ¬a = null</inv>)) 9 @ ensures (\forall int
Proof 📮 🖬 🕻	- {heapAtPre:=heap _a:=a}	10 @ ensures (a.length > 0
Proof Tree G Invariant Initially Valid	1<	11 0 ==> (\exists
Invariant Initiality Valid Invariant Initiality Valid	exc = null;	12 @ ensures sum == (\sum 13 @ ensures sum <= a land
+ Case 1	🖼 🔓 🖈 Background SMT Status: SumAndMax(SumAnL normal_behavior operation contract.0	v ^ × 14 0*/
+ Case 1	3 Solved Node CVC5 73	15 void sumAndMax(int[] a) {
👻 🖾 Case 1	Goal 1212 Valid (0.105s) Info Valid (0.051s)	Info + 16 sum = 0;
		17 max = 0;
Case 1	} \> Goal 1211 Valid (0.091s) Info Valid (0.043s)	Info 18 int k = 0;
Case 2	Goal 1210 Valid (0.093s) Info Valid (0.042s)	Info = // /*@ loop invariant
	Cost 1104 Valid (0.105r) Info Valid (0.042r)	Info 0 <= k && k <= a.
+ Case 2		22 @ && (\forall int i
R 1210:OPEN GOAL	Goal 1180 Valid (0.084s) Info Valid (0.039s)	Info 23 @ && (k == 0 ==> ma
Case 2	Goal 1048 Valid (0.105s) Info Valid (0.042s)	Info 24 @ && (k > 0 ==> (\e
Case 2 D Q 1104:OPEN COAL		25 @ && Sun == (\Sun 1
Case 2	Goal 1003 Valid (0.1045) Unio Valid (0.0545)	
R 1187:OPEN GOAL	I A A A A A A A A A A A A A A A A A A A	28 @ assignable sum, ma
	Apply Goals Deactivate Backgroun	nd SMT 29 @ decreases a.length
R 1180:OPEN GOAL		30 0*/
Invariant Preserved and Used		31 while(k < a.length) (
 Glifb 1 true 		33 may = a[k] -
Image: A secution (a != null)		34
+ ☐ if b_2 true		35 sum += a[k];
 Image: mail of the security of th		36 k++;
 Investigation (single investigation in the second se		37 }
 Normal Execution (_a != null) 		38 }
 Normal Execution (s_1 i= hull) 		39 }
Null Reference (s. 1 = pull)		40
Null Reference (a = null)		
 Index Out of Bounds (_a != null, but k 0 		
- 🕨 😭 1048:OPEN GOAL		
Null Reference (s = null)		
 Null Reference (_a = null) Calladar Out of Records (_a la null but h Out		
Index Out of Bounds (_a i= hull, but k Out of		
	<u> </u>	Show Postcondition/Assignable
	<u> </u>	Show Postcondition/Assignable



Proof Slicing	\checkmark
Navigation History	\checkmark
Undoing Interactions	\checkmark
Automatically run JavaC first (Alexander Weigl)	\checkmark
Background SMT (ongoing)	\checkmark
Proof Caching (ongoing)	<



Motivation: Finding the correct and provable specification is often an iterative process.



Motivation: Finding the correct and provable specification is often an iterative process.

Observation: If $\Gamma \vdash \Delta$ is valid, then $\Gamma, E \vdash \Delta, Z$ is also valid (*).



Motivation: Finding the correct and provable specification is often an iterative process.

```
Observation: If \Gamma \vdash \Delta is valid, then \Gamma, E \vdash \Delta, Z is also valid (*).
```

(*) Under some restrictions:

- The extended sequent must not have modalities or queries (Java code could differ).
- Both must use the same taclet options.
- The same added rules must be present.



Motivation: Finding the correct and provable specification is often an iterative process.

```
Observation: If \Gamma \vdash \Delta is valid, then \Gamma, E \vdash \Delta, Z is also valid (*).
```

(*) Under some restrictions:

- The extended sequent must not have modalities or queries (Java code could differ).
- Both must use the same taclet options.
- The same added rules must be present.

Ongoing work:

- Which sequents should be in the cache?
- Extend the caching beyond a single run of KeY (ongoing).
- Relax the above conditions.

8 *	KeY 2.11.0 [appelhagen/backgroundSMT]	~ ^ X
Eile View Proof Options Origin Tracking		About
Run CVC5 - 43 5 C C 1 1	E 🗲 > 🔹 🔹 🚳 🖉 🔢 Layouts: Default 🔻 Load Layout Save Layo	ut Reset Layout D - Exploration Mode Hide justification
🗌 Loaded Proofs 📃 🗖 🗖	🗋 Sequent 💶 🖬 🗖	Source _ 2 🗆
Proofs	Inner Node	SumAndMax.java
Loade Proofs C C C C C C C C C C C C C C C C C	<pre></pre>	$ \begin{bmatrix} 1 \text{ same} & - 2 \\ \hline 3 \\ \hline 3 \\ \hline 3 \\ \hline 1 \\ \mathbf{class Sumhodbax (w) \\ \hline 1 \\ \mathbf{class Sumhodbax (} \\ \hline 3 \\ 4 \\ 3 \\ 4 \\ 4 \\ 4 \\ 4 \\ 4 \\ 4 \\ 4 \\ 4 \\ 4 \\ 4 \\ 5 \\ $
		Show Postcondition/Assignable

f 🖈 KeY 2.11.0 [appelhagen/backgroundSMT] 🗸 🗸 x				
Eile View Proof Options Origin T	racking			About
🖆 Load Example	Y ECZER #		I avout : Default w Load I avout Save Lavout	Parat I would be Evaluation Made Wide justification
E Load Ort-O			Layouts. Default - Load Layout Jave Layout	
C Reload Ctrl-R	_ 8 🗆	C Sequent	_ 🛛 🗖	🗋 Source _ 🗖 🗖
🗹 Edit Last Opened File	and the	Inner Node		SumAndMax.java
Save	copened me.			1 class SumAndMax {
Save Proof as Bundle	ax[[1]].jmc.normal_benavior operation ci	⇒		2
Quicksave 15		wellFormed(heap)		3 int sum;
Quickload F6		<pre>^ ¬self = null</pre>		4 int max;
😤 Proof Management 🛛 🗠 Ctrl-M	•	<pre>A self.<created> = TRUE</created></pre>		5 /*0 normal hebaviour
Load User-Defined Taclets	ation Stone	A Sumaridmax::exactinstance(sell) = 1 (a = null y a coreated) = TRUE)	KUE	7 @ requires (\forall int i: 0 <= i && i < a.length: 0 <= a
Prove	Proof Search Strategy	A measuredByEmpty		8 @ assignable sum, max;
U Recent Files	A 85	▲ (♥ int i; (0 ≤ i ▲ i < a.length	<pre>∧ inInt(i) → 0 ≤ a[i])</pre>	9 @ ensures (\forall int i; 0 <= i && i < a.length; a[i] <=
Exit Ctrl-Q	÷	<pre>(self.<inv> A ¬a = null))</inv></pre>		10 0 ensures (a.length > 0
Invariant Initially Valid		<pre>- {heapAtPre:=heap _a:=a}</pre>		11 0 (vexists int i, 0 <= 1 & i < a.length, max 12 0 //ensures sum == (\sum int i: 0 <= 1 && i < a.length; max
Invariant Preserved and Use	d	exc = null:		<pre>13 @ ensures sum <= a.length * max;</pre>
		try (14 e*/
		<pre>self.sumAndMax(_a)@SumAndMax;</pre>		<pre>15 void sumAndMax(int[] a) {</pre>
		<pre>} catch (java.lang.Throwable e)</pre>	{	10 sum = 0;
		exc = e;		17 max - 0, 18 int k = 0:
))> (Winti: (0 < i A i < a ler	onth a inInt(i) , a[i] < self max)	19
		<pre>/ (a.length > 0</pre>	igen in initie(1) + u[1] + serrinux)	20 /*@ loop_invariant
		- 3 int i;		21 @ 0 <= k && k <= a.length
		(0 ≤ i ∧ i < a.leng	<pre>gth A inInt(i) A self.max = a[i]))</pre>	22 @ && (\forall int 1; 0 <= 1 && 1 < k; a[1] <= max)
		∧ (self.sum ≤ a.length * s	<pre>self.max A self.<inv>))</inv></pre>	23 θ δά (k = 0 ==> (levists int i: 0 <= i 88 i < k: may a
		A exc = null		25 Ø //&& sum == (\sum int i; Ø <= i && i < k; a[i])
		V java.lang.Object o:		26 🕘 && sum <= k * max;
		((o, f) < {(self, s	SumAndMax::\$sum)}	27 @
		U {(self, s	SumAndMax::\$max)}	28 e assignable sun, max;
		v ¬o = null		20 et decreases allenger - k,
		x =0. <created>eneapAtPre</created>	re = TRUE	<pre>31 while(k < a.length) {</pre>
		• 0.1 = 0.1encapacite,		32 if(max < a[k]) {
				33 max = a[k];
				34 }
				35 Sum += a[K]; 36 k++
				37 }
				38 }
				39 }
				40
				Show Postcondition/Assignable
1				snow rosconstituti/Assignable

8 *	KeY 2.11.0 [appelhagen/backgroundSMT]			× ^ 3
Eile View Proof Options Origin Tracking				Abou
	Æ ← →	we Layout	t Reset Layout 🛛 🗸 🗌 Explorat	ion Mode 🗌 Hide justification
C Loaded Proofs 2	G Sequent		() Source	- 9 0
Proofs	Current Goal			
Env. with model src@7:05:05 PM	(hearts Dress - hear		SumAnumaxJava	
SumAndMax(SumAndMax:sumAndMax(II)).IMI_normal_behavior_operat	(heapktrieheap		1 class SumAndMax {	
Free with model src@7:10:58 PM			2	
SumAndMax/SumAndMax:sumAndMax/IIIUML normal behavior operation	I the heat to be a set of the beauty of the		3 int sum;	
	(n.=heap[seri.sum .= 0][seri.max .= 0]		4 int max;	
	hh_0 pk 0 * -1 + p length)		5	
	<pre>// ii bean:=bean[self sum := 0]</pre>		6 /*@ normal_behaviour	
Goals Proof Slicing Exploration Steps	[self max := 0]		7 @ requires (\forall in	<pre>it i; 0 <= 1 && 1 < a.length; 0 <= a</pre>
Proof Info Proof Search Strategy	[anon(//self SumåndMax::Smax)]		8 @ assignable sum, max;	6
- Duni 🔶 🖪 🗖	U ((self, SumAndMax::\$sum)).		9 @ ensures (\forall int	<pre>c i; 0 <= i && i < a.length; a[i] <=</pre>
	anon heap LOOP @)]		10 0 ensures (a.length >	0
Proof Tree	[self.max := a[k_0]]		11 @ ==> (\exist:	; int 1; 0 <= 1 && 1 < a.length; max
- Invariant Initially Valid	s_1:=self		e ensures sum == (\sur	iinci; 0 <= 1 && 1 < a.iength; a[:
- IT 42:OPEN GOAL	<pre> 1_5:=a[k_0] + self.sum@anon_heap_LOOP_0}</pre>		13 e ensures sum <= a.ler	igen - max;
Invariant Preserved and Used	(s 1 = null),		14 87	
 Normal Execution (_a := null) 	<pre>self.max@anon_heap_LOOP_0 < a[k_0],</pre>		15 Void sumanomax(int[] a) {	
- CR Normal Computing (a la pull)	k_0 < a.length,		17 500 - 0,	
 Chifb 2 true 	k_0 ≥ 0,		12 int k = 0;	
 C² Normal Execution (a to pull) 	a.length ≥ k_0,		10	
Call Normal Execution (s l= null)	V int i;		70 /*@ loop invariant	
Call Normal Execution (a.1= null)	(i < k_0 ∧ i ≥ 0		21 0 0 <= k && k <= 2	a length
Normal Execution (s 1 != null)	a[i]@heap[self.sum := 0]		22 @ && (\forall int	1: 0 <= 1 && 1 < k: a[i] <= max)
Q 352:OPEN GOAL	[self.max := 0]		23 @ && (k == 0 ==> (nax == 0)
 Image: Null Reference (s 1 = null) 	[anon({(self, SumAndMax::\$max)}		24 @ && (k > 0 ==> ()	exists int i: 0 <= i && i < k; max
- ← 337:Cached goz	<pre>∪ {(self, SumAndMax::\$sum)},</pre>		25 @ && sum == (\sum	<pre>int i; 0 <= i && i < k; a[i])</pre>
 Mull Reference (_a = null) 	anon_heap_LOOP_0) J		26 @ && sum <= k * ma	ax;
➡ 316:Cached goal	<pre>≤ self.max@anon_heap_LOOP_0),</pre>		27 @	
+ 📑 Index Out of Bounds (_a != null, but k (<pre>k_0 = 0 - self.max@anon_heap_LOOP_0 = 0,</pre>		28 @ assignable sum, r	nax;
- 📢 317:Cached goal	K_0 > 0		29 @ decreases a.lengt	th - k;
 Image: Image: Second Sec	- s int 1;		30 @*/	
- ◀ 302:Cached goal	(1 < K_0		<pre>31 while(k < a.length) {</pre>	
 Image: Second sec	A 1 2 0		<pre>32 if(max < a[k]) {</pre>	
4 274:Cached goal	<pre>x a[i]encap[self.sum := 0]</pre>		33 max = a[k];	
 Index Out of Bounds (_a != null, but k Out of 	[acci.mdx.= 0] [acci.lidx= 0]		34 }	
4 275:Cached goal	U ((self_SumAndMax::SumA))		35 sum += a[k];	
	apon heap LOOP 011		36 k++;	
 Image: Image: Normal Execution (s != null) 	= self.max@anon_heap_LOOP_0).		37 }	
 Normal Execution (_a != null) 	bsum(int i:)(0.		38 }	
 Normal Execution (s_1 != null) 	k 0.		39 }	
- ¥ 267:OPEN GOAL	a[i]@heap[self.sum := 0]		40	
 Invuir Reference (s_1 = null) A 353:Cached goal 	[self.max := 0]			
- T 252:cacried goar	[anon({(self, SumAndMax::\$max)}			
All 234/Cached goal	U {(self, SumAndMax::\$sum)},			
- Tindex Out of Rounds (a I- null but k Out	anon_heap_LOOP_0)])			
4 235:Cached goal	= self.sum@anon_heap_LOOP_0,			
Concerned goes The second second goes The second sec	self.max@anon_heap_LOOP_0 * k_0 ≥ self.sum@anon_heap_LOOP_0,			
	wellFormed(anon_heap_LOOP_0),		Null Refere	ence (s_1 = null)
	wellFormed(bean)			

📕 🖬 Javac (0) Ø

KeY 2.11.0 [appelhagen/backgroundSMT] V ^ X					
Eile View Proof Options Origin Tracking					About
	= 1	The second secon			Contemporter Made Catile institution
	· · · ·	Cayouts: Derault V Load Layout Save Layout	ut Re	Iset Layout	Exploration Mode Hide Justification
🗋 Loaded Proofs 📃 🖬 🗖	Sequent	_ 2 🗆	05	ource	_ 2 🗆
Proofs	Current Goal		Su	mAndMax.iava	
Env. with model src@7:05:05 PM	{heapAtPre	:=heap		-1	<i>t</i>
SumAndMax{SumAndMax:sumAndMax[[]].JML normal_behavior operat	_a:=a		1	CLASS SUMANOME	ax {
Env. with model src@7:10:58 PM	exc:=r	ull	2	4	
SumAndMax[SumAndMax::sumAndMax[[]]]ML normal_behavior operation	(h:=he	ap[self.sum := 0][self.max := 0]	5	int max:	
	k:=k_@		5	and hox,	
] a:=k_@	* -1 + a.length)	6	/*0 normal	1 hebaviour
	heap:=	heap[self.sum := 0]	7	0 reg	wires (\forall int i: 0 <= 1 && 1 < a length: 0 <= a
Proof Sticing Exploration Steps	1	[self.max := 0]	8	P assi	ionable sum, max:
Proof V Info V Proof Search Strategy		[anon({(self, SumAndMax::\$max)}	9	@ ensu	ures (\forall int i: 0 <= i && i < a.length: a[i] <=
📫 Proof 🌼 🗖 🗖 🗖		U {(self, SumAndMax::\$sum)},	10	@ ensu	ures (a.length > 0
Proof Tree	1	anon_heap_LOOP_0)]	11	0	==> (\exists int i; 0 <= i && i < a.length; max
🔸 🔚 Invariant Initially Valid		[selt.max := a[K_0]]	12	@ ensu	ures sum == (\sum int i; 0 <= i && i < a.length; a[i]
- R 42:OPEN GOAL	11 5_1:=5	(k a) + colf summanan haan 1000 a)	13	@ ensu	ures sum <= a.length * max;
 Invariant Preserved and Used 	1 1 30	<pre>[K_0] + Set1.Suleanon_neap_coor_of</pre>	14	e*/	
 Investigation (_a != null) 	colf maxe	non hean 1008 8 < s[k 8]	15	void sumAr	ndMax(int[] a) {
	k 0 < a le	noth	16	sum =	0;
 Mormal Execution (_a != null) 	k 0 > 0.		17	max =	0;
• G If b_2 true	a.length a	: k Ø.	18	INT K	= 0;
 Wormal Execution (a = null) 	¥ int i;		19	(10.1	and designed and
 Normal Execution (a le pull) 	(i <)	_0 A 1 ≥ 0	20	/ • 1	A car k 22 k car a longth
C Normal Execution (s 1 I= null)	- a[i]@heap[self.sum := 0]	22		<pre>RE ()forall int i: 0 <= i RE i < k: a[i] <= max)</pre>
Q 352:OPEN GOAL		[self.max := 0]	23		88 (k == 0 ==> may == 0)
 Image: A state of the state of		<pre>[anon({(self, SumAndMax::\$max)}</pre>	24		$\delta\delta$ (k > 0 ==> (\exists int i: 0 <= i $\delta\delta$ i < k: max =
		U {(self, SumAndMax::\$sum)},	25		&& sum == (\sum int i: 0 <= i && i < k; a[i])
Null Reference (_a =)		anon_heap_LOOP_0)]	26		&& sum <= k * max;
→ 4 316:Cached goal		t.max@anon_heap_LOOP_0),	27		
 Index Out of Bounds Edit Notes 		selt.max@anon_neap_LOOP_0 = 0,	28		assignable sum, max;
- 📢 317:Cached goal 🔂 Expand All Below	,		29		decreases a.length - k;
 Mull Reference (s = null) Expand Goals On 	ly Below	k a	30	e*/	
Gollapse Below		0	31	while	(k < a.length) {
 Collapse Other E 	ranches	fil@heapfself.sum := 0]	32	11	f(max < a[k]) {
• • 2/4:Cached goal		[self.max := 0]	33		max = a[K];
Index Out of Bounds (a): C Previous sibility		[anon({(self, SumAndMax::\$max)}	34	4	m to afkly
Tifh 2 false		∪ {(self, SumAndMax::\$sum)},	36		tt.
Set All Goals Bel	w to Interactive	anon_heap_L00P_0)]	37	3	,
Call Secution (a != Set All Goals Bel	w to Automatic	elf.max@anon_heap_LOOP_0),	38	3	
 C Normal Execution (s Show Subtree St 	atistics	i;}(0,	39	3	
ନ 267:OPEN GOAL Open Node in Se	parate Buffer	k_0,	40		
 Image: Null Reference (s_1 = Proof Caching 	÷	Close by reference to other proof			
4 252:Cached goal		Copy referenced proof steps here			
 Image: mail of the second secon		[mnon({(Self SumAndMax::Smdx)}			
← ◀ 234:Cached goal		apon bean LOOP 0111			
 Index Out of Bounds (_a != null, but k Out 	= self.su	Manon heap LOOP 0.			
	self.max@a	non heap LOOP 0 * k 0 ≥ self.sum@anon heap LOOP 0.		1	
	wellFormed	(anon_heap_LOOP_0),			Null Reference (r. 1 = pull)
	wellFormer	(/hean)			run vererence (a_r - null)

📕 🗹 Javac (0) Ø

Kgy

🕅 🖈 KeY 2.11.0 [appelhagen/background5MT] 🗸 🗸 🗡				
Eile View Proof Options Origin Tracking		About		
	= 🗧 🕈 🚽 📲 🔮 🋄 Layouts: Default 👻 Load Layout Save Lay	out Reset Layout		
🗌 Loaded Proofs 📃 🖬 🗖	_ Sequent _ 2 t	🗆 🗋 Source 📃 🗖 🗖		
Proofs	Closed Goal	SumAndMax java		
Env. with model src@7:05:05 PM	false.			
SumAndMax[SumAndMax::sumAndMax[[]].JML normal_behavior operation	self.max@anon heap LOOP 0 < a[k 0].	I CLASS SUMANOMAX {		
Env. with model src@7:10:58 PM	k_0 < a.length,			
SumAndMax[SumAndMax:sumAndMax[[]]]ML normal_behavior operation	k_0 ≥ 0,	A int sur,		
	a.length ≥ k_0,	5		
	¥ int i;	6 /*0 normal behaviour		
(bu and) = and all the V and ender the state	(i < k_0 ∧ i ≥ 0	7 P requires (\forall int i: 0 <= 1 && i < a.length: 0 <= a		
po Goals = Proor Silcing Exploration Steps	a[i]@heap[self.sum := 0]	8 @ assignable sum, max;		
Proof Unito UProof Search Strategy	[self.max := 0]	9 @ ensures (\forall int i: 0 <= i && i < a.length: a[i] <=		
📥 Proof 🌼 📮 🖬 🗖	[anon({(self, SumAndMax::\$max)}	10 0 ensures (a.length > 0		
Proof Tree *	<pre>U {(selt, SumAndMax::\$sum)},</pre>	11 0 ==> (\exists int i; 0 <= i && i < a.length; max		
+ 🗂 Invariant Initially Valid	anon_neap_LOOP_0) J	12 @ ensures sum == (\sum int i; 0 <= i && i < a.length; a[i]		
R 42:OPEN GOAL	≤ selt.maxganon_neap_LOOP_0),	13 @ ensures sum <= a.length * max;		
- Invariant Preserved and Used	k_0 = 0 - setr.maxeanon_neap_LOOP_0 = 0,	14 @*/		
+ 🖆 Normal Execution (_a != null)	A int in	<pre>15 void sumAndMax(int[] a) {</pre>		
	- 3 10 1, (i c k 0	16 sum = 0;		
 Image: Image: Security of the sec	A 1 > 0	17 max = 0;		
	▲ afil@hean[self.sum := 0]	18 int k = 0;		
 Image: Security of the security o	[self max := 0]	19		
 Normal Execution (s != null) 	<pre>[anon({(self, SumAndMax::Smax)}</pre>	20 /*@ loop_invariant		
Normal Execution (_a != null)	<pre>U {(self, SumAndMax::\$sum)},</pre>	21 @ 0 <= K && K <= a.length		
Normal Execution (s_1 != null)	anon heap LOOP 011	22 @ && (\forall int 1; 0 <= 1 && 1 < k; a[1] <= max)		
- Y 352:OPEN GOAL	= self.max@anon_heap_LOOP_0),	23 Θ & (K == 0 ==> max == 0) 24 Θ = Θ (k = 0 ==> () anists int is Θ (s = i Θ i s is a set		
 Null Reference (s_1 = hull) 	<pre>bsum(int i;)(0,</pre>	24 @ 00 (K > 0> (Verists Int 1, 0 <- 1 00 1 < K, mox - 25 @ 28 cm -= (Veristist 1, 0 <- 1 00 1 < K, mox -		
CR Null Reference (a = pulk)	k_0,	25 e aa sun (tsun int 1, 0 1 aa 1 - K, a[1])		
4 316 Cached goal	a[i]@heap[self.sum := 0]	20 e da Suli - K lidA,		
G Index Out of Bounds (a != null, but k ([self.max := 0]	28 @ assignable sum max:		
4 317:Carbed goal	<pre>[anon({(self, SumAndMax::\$max)}</pre>	29 decreases a length - k:		
 Image: Null Reference (s = null) 	U {(self, SumAndMax::\$sum)},	30 0*/		
4 302:Cached goal	anon_heap_L00P_0)])	31 $while(k < a, length)$ (
Image: Second	= self.sum@anon_heap_LOOP_0,	32 if (max < a[k]) (
4 274:Cached goal	<pre>self.max@anon_heap_LOOP_0 * k_0 ≥ self.sum@anon_heap_LOOP_0,</pre>	33 max = a[k];		
 Index Out of Bounds (_a != null, but k Out of 	wellFormed(anon_heap_LOOP_0),	34 }		
4 275:Cached goal	wellFormed(heap),	35 sum += a[k];		
 if b_2 false 	selt. <created> = TRUE,</created>	36 k++;		
 Image: Image: Security of the sec	SUMANOMAX::exactinstance(selt) = IRUE,	37 }		
 Image: Security of the security o	a.screated/ = 1kde,	38 }		
 Image: Second Sec	a loogth > 0	39 }		
- ♀ 267:OPEN GOAL	Winti: (i c a length a i a 0 , alil a 0)	40		
 Image: Second sec	• Inc 1, (1 - 0.100gen H 1 = 0 - 0[1] = 0)			
← ◀ 252:Cached goal	→			
 Image: Null Reference (_a = null) 	self = null,			
- ← 234:Cached goal	a = null,			
 Index Out of Bounds (_a != null, but k Out 	{heapAtPre:=heap			
The second secon	_a:=a			
Imuli kererence (s = null)	exc:=null	Null Pafarance (r. 1 = null)		
	<pre> (h:=heap[seif.sum := 0][seif.max := 0]</pre>	×		

KgY 1 goal closed, 16 remaining

📕 🗹 Javac (0) Ø



Conclusion

What we have seen

- A novel way to represent (certain) proof goals as JML.
- Multiple new UI features (some already in the new 2.12 release).
- Proof Slicing
- Navigation History
- Undoing Interactions
- Automatically run JavaC first (Alexander Weigl)
- Background SMT (ongoing)
- Proof Caching (ongoing)