POSTECH PL박종현

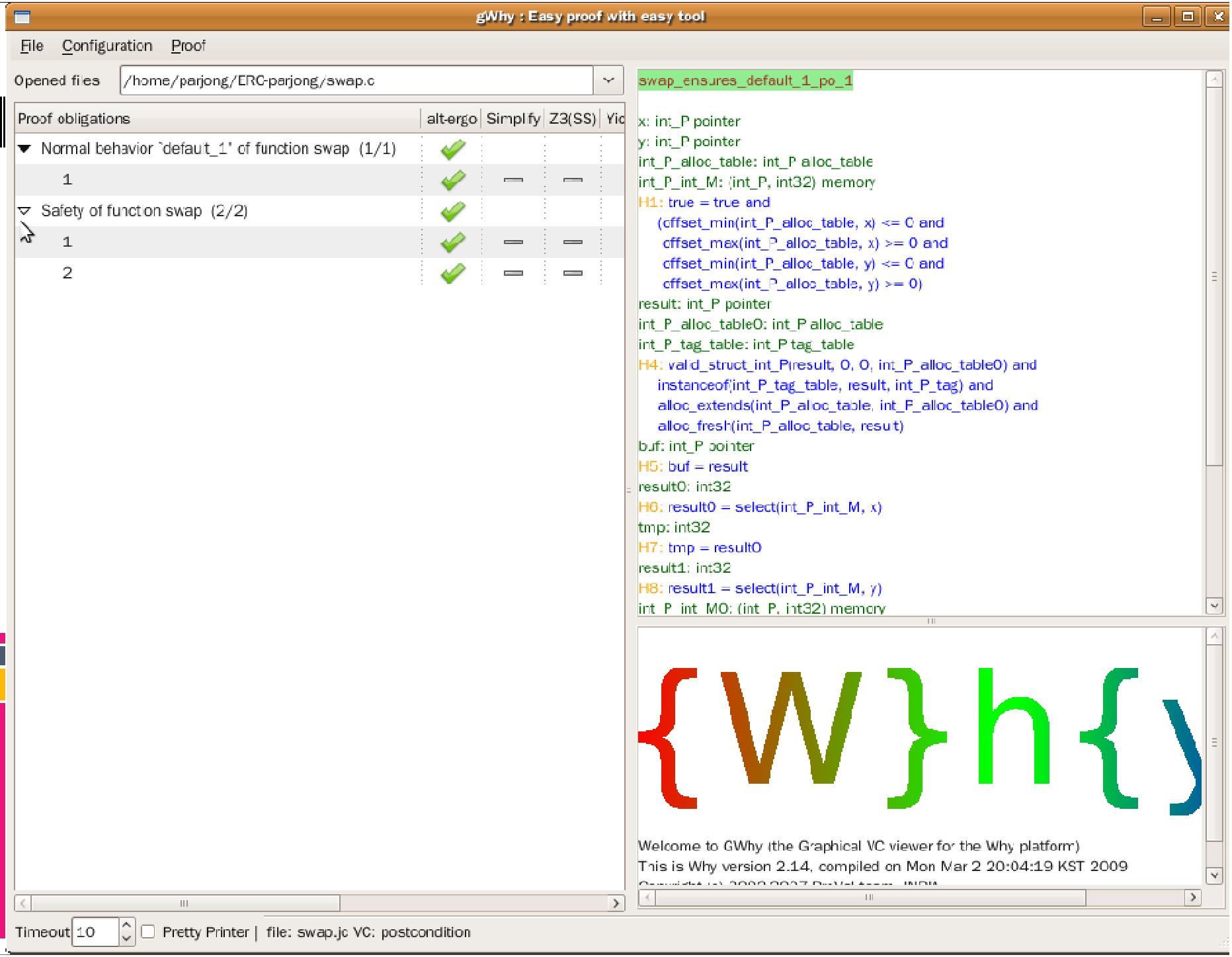
FRAMA-C 소개

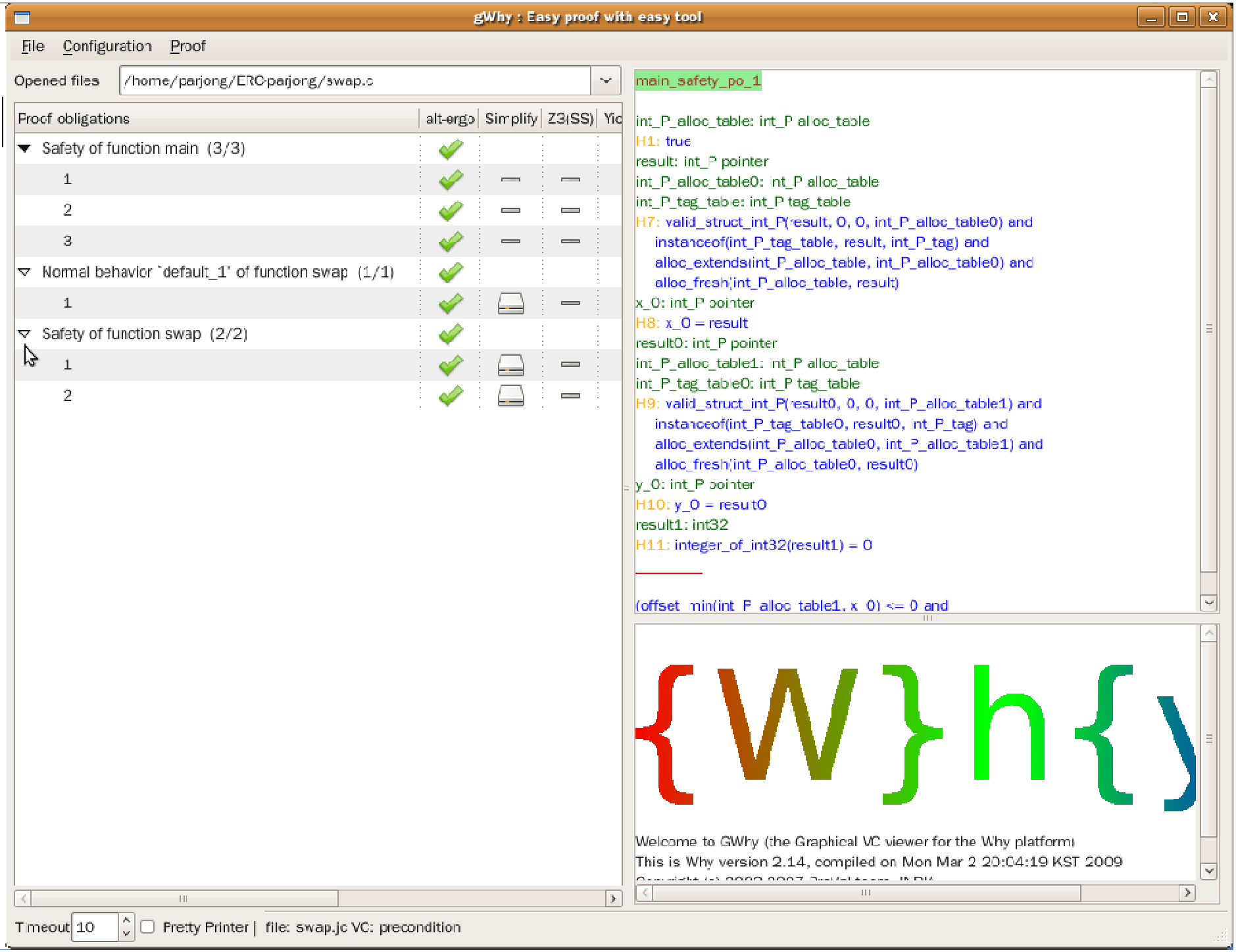
FRAMA-C

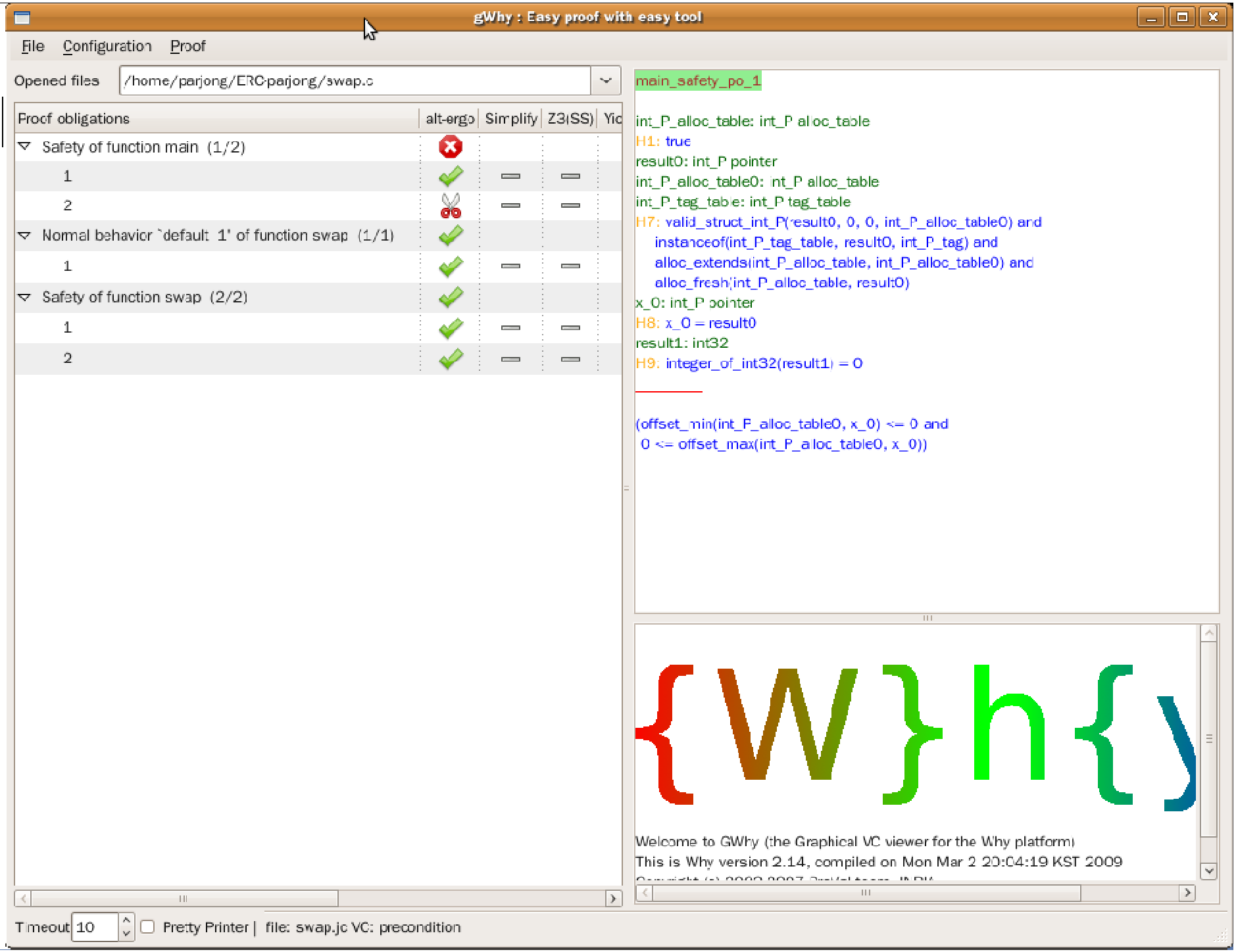
- An extensible platform dedicated to source-code analysis of C programs.
- Available plug-ins
  - Aorai
  - …
  - …
  - Jessie

Jessie

- A plug-in for deductive verification of C progr ams annotated with ANSI/ISO C Specification Language (ACSL)
- Generate verification conditions for external theorem provers
  - Ex) Coq, Simplify, ……


gWhy : Easy proof with easy tool
File
Configuration
Proof
Opened files
/home/parjong/ERC-parjong/swap.c
Proof obligations
alt-ergo
Simplify
Z3(SS)
Normal behavior `default_1' of function swap (1/1)
1
Safety of function swap (2/2)
1
2
swap_ensures_default_1_po_1
x: int_P pointer
y: int_P pointer
int_P_alloc_table: int_P alloc_table
int_P_int_M: (int_P, int32) memory
H1: true = true and
(offset_min(int_P_alloc_table, x) <= 0 and
offset_max(int_P_alloc_table, x) >= 0 and
offset_min(int_P_alloc_table, y) <= 0 and
offset_max(int_P_alloc_table, y) >= 0)
result: int_P pointer
int_P_alloc_table0: int_P alloc_table
int_P_tag_table: int_P tag_table
H4: valid_struct_int_P(result, 0, 0, int_P_alloc_table0) and
instanceof(int_P_tag_table, result, int_P_tag) and
alloc_extends(int_P_alloc_table, int_P_alloc_table0) and
alloc_fresh(int_P_alloc_table, result)
buf: int_P pointer
H5: buf = result
result0: int32
H6: result0 = select(int_P_int_M, x)
tmp: int32
H7: tmp = result0
result1: int32
H8: result1 = select(int_P_int_M, y)
int_P_int_M0: (int_P, int32) memory
Welcome to GWhy (the Graphical VC viewer for the Why platform)
This is Why version 2.14, compiled on Mon Mar 2 20:04:19 KST 2009
Timeout
10
Pretty Printer | file: swap.jc VC: postcondition


gWhy : Easy proof with easy tool
File
Configuration
Proof
Opened files
/home/parjong/ERC-parjong/swap.c
Proof obligations
alt-ergo
Simplify
Z3(SS)
Safety of function main (3/3)
1
2
3
Normal behavior `default_1' of function swap (1/1)
1
Safety of function swap (2/2)
1
2
main_safety_po_1
int_P_alloc_table: int_P alloc_table
H1: true
result: int_P pointer
int_P_alloc_table0: int_P alloc_table
int_P_tag_table: int_P tag_table
H7: valid_struct_int_P(result, 0, 0, int_P_alloc_table0) and
instanceof(int_P_tag_table, result, int_P_tag) and
alloc_extends(int_P_alloc_table, int_P_alloc_table0) and
alloc_fresh(int_P_alloc_table, result)
x_0: int_P pointer
H8: x_0 = result
result0: int_P pointer
int_P_alloc_table1: int_P alloc_table
int_P_tag_table0: int_P tag_table
H9: valid_struct_int_P(result0, 0, 0, int_P_alloc_table1) and
instanceof(int_P_tag_table0, result0, int_P_tag) and
alloc_extends(int_P_alloc_table0, int_P_alloc_table1) and
alloc_fresh(int_P_alloc_table0, result0)
y_0: int_P pointer
H10: y_0 = result0
result1: int32
H11: integer_of_int32(result1) = 0
(offset_min(int_P_alloc_table1, x_0) <= 0 and
Welcome to GWhy (the Graphical VC viewer for the Why platform)
This is Why version 2.14, compiled on Mon Mar 2 20:04:19 KST 2009
Timeout
10
Pretty Printer | file: swap.jc VC: precondition


gWhy : Easy proof with easy tool
File
Configuration
Proof
Opened files
/home/parjong/ERC-parjorg/swap.c
Proof obligations
alt-ergo
Simplify
Z3(SS)
Safety of function main (1/2)
1
2
Normal behavior `default: 1' of function swap (1/1)
1
Safety of function swap (2/2)
1
2
main_safety_po_1
int_P_alloc_table: int_P alloc_table
H1: true
result0: int_P pointer
int_P_alloc_table0: int_P alloc_table
int_P_tag_table: int_P tag_table
H7: valid_struct_int_P(result0, 0, 0, int_P_alloc_table0) and
instanceof(int_P_tag_table, result0, int_P_tag) and
alloc_extends(int_P_alloc_table, int_P_alloc_table0) and
alloc_fresh(int_P_alloc_table, result0)
x_0: int_P pointer
H8: x_0 = result0
result1: int32
H9: integer_of_int32(result1) = 0
(offset_min(int_P_alloc_table0, x_0) <= 0 and
0 <= offset_max(int_P_alloc_table0, x_0))
Welcome to GWhy (the Graphical VC viewer for the Why platform)
This is Why version 2.14, compiled on Mon Mar 2 20:04:19 KST 2009
Timeout
10
Pretty Printer
file: swap.jc VC: precondition

연구방향

- Memory leak detection for C programs
  - Based on deductive verification
  - Proof => Memory leak
  - Annotation inference
- Debugging tool for C programs
  - Reasoning = Debugging
  - Common proof = Common leak pattern

Question?

THANK YOU