MO601 – Project 3
LISC
Learning Instruction Semantics from Code Generators

093311 – Alberto Arruda de Oliveira

Overview

- Binary analysis and instrumentation
  - Program monitoring (Valgrind, Pin)
  - Virtualization
  - Malware Analysis
  - Etc.
- Modeling of instruction Semantics
  - Translation of assembly to intermediate language representation
- Requires previously existing models, created manually
  - Limited Support of new architectures

LISC Objective

- Automated modeling of instruction semantics
- Compilers: Intermediate Language -> Assembly Code
  - GCC, LLVM, etc.
- LISC:
  - Learn from code generators
  - Automatically extract semantics
  - Test extracted semantics

LISC Benefits

- Automated instruction semantics modeling
  - Machine Learning Approach
  - Reduce manual efforts
  - Broaden Architecture Support
- Architecture Neutrality
  - Code Generators
- Well tested compiler code

Project 3 Objective

- Test LISC in a already used architecture of choice
  - x86, ARM, AVR

- Reproduce Table 4 of reference [1]
  - Completeness result

[1] Niranjan Hasabnis and R. Sekar. 2016. Lifting Assembly to Intermediate Representation: A Novel Approach Leveraging Compilers. In *Proceedings of the Twenty-First International Conference on Architectural Support for Programming Languages and Operating Systems* (ASPLOS '16). ACM, New York, NY, USA, 311-324.

LISC Evaluation Completeness

- Are all the instructions lifted correctly?
- 99.5% of Ubuntu/x86 and 99.8% of Debian/ARM binaries
  - Missed Instructions are mostly NOPS
- Evaluation:
  - Testing programs P_{test} and training programs $P_{train} \subset P_{test}$
  - Build a transducer using P_{train}
  - Translate the assembly of P_{test}

LISC Evaluation Completeness

- How was P_{test} chosen?
  - All x86 binaries, including kernel modules, found on Ubuntu-14.04 desktop;
  - Around 38M unique instructions;

- How was P_{train} chosen?
  - Started as *openssl-1.0.1f* + *binutils-2.22* binaries;
  - Each round added a new binary, in order: *ffmpeg-2.3.3* (With no optmizations), *glibc-2.21*, *ffmpeg-2-3-3(With optmizations), gstreamer-1.4.5, qt-5.4.1, linux-kernel-3.19, Manual instructions*.

LISC Evaluation
Completeness Metrics

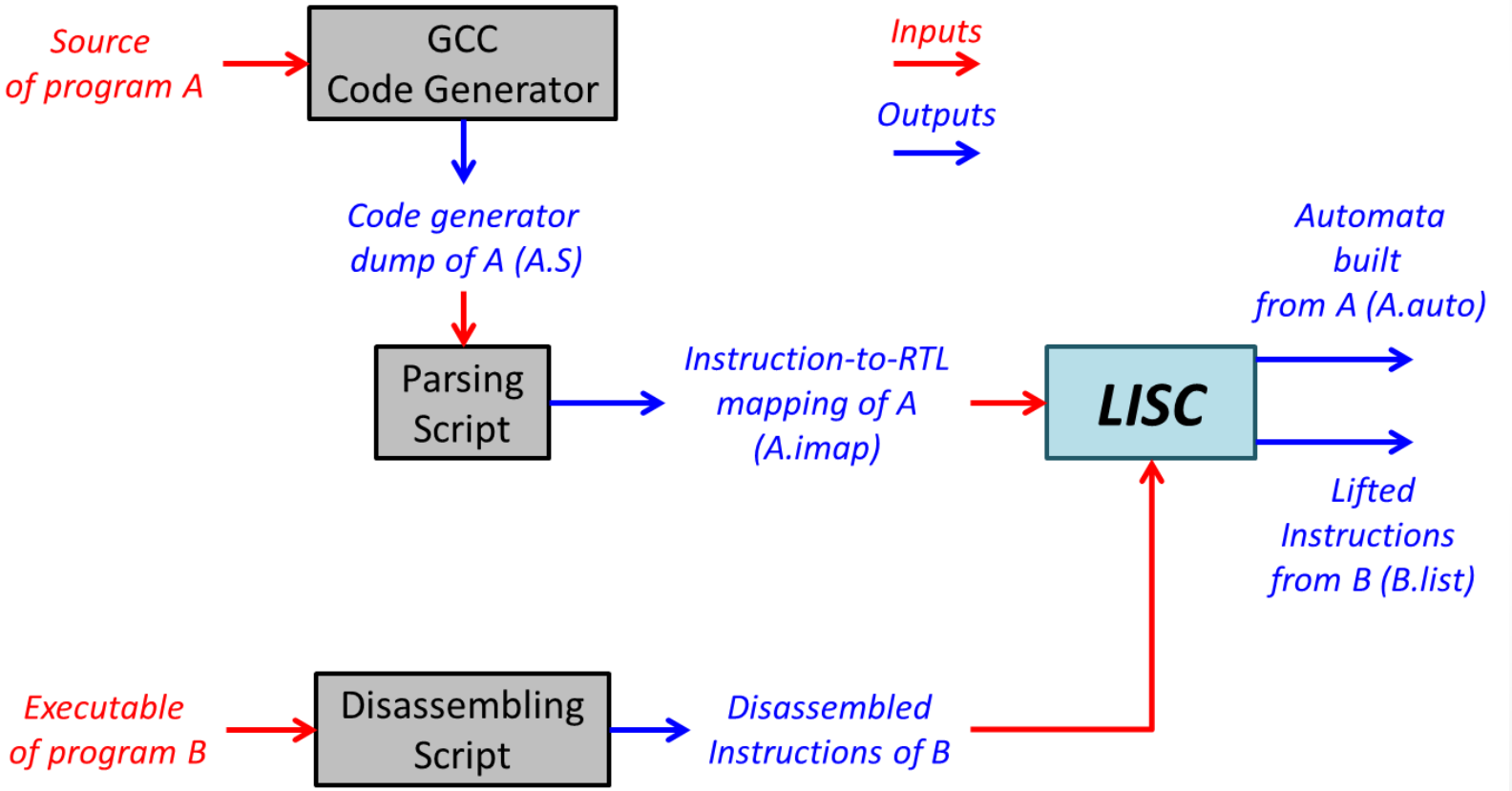
- Comparison between:
  - **Exact Recall:** an instruction from P_{test} is lifted only if it also belongs to P_{train};
  - **LISC %** of instructions lifted

- From the LISC %:
  - Number of Mnemonics missing, from the total of 1187 found in x86:
    - Percentage;
    - Absolute Number;
  - Percentage of operands missing

LISC Evaluation Completeness

P_train	% Instructions Lifted		LISC (%)		Missing Mnemonics (Absolute)
	Exact Recall	LISC	Missing Mnemonics	Missing Operands	
openssl-1.0.1f + binutils-2.22	63.72	98.46	1.05	0.49	464
+ffmpeg-2.3.3 (Non Opt)	68.21	98.74	1.03	0.23	377
+glibc-2.21	68.74	98.8	1.01	0.19	346
+ffmpeg-2.3.3 (Opt)	69.07	98.89	0.88	0.23	303
+gstreamer-1.4.5	71.07	99.1	0.79	0.11	221
+qt-5.4.1	72.45	99.21	0.69	0.09	161
+linuxkern-3.19	73.97	99.49	0.44	0.07	49
+Manual	74.04	100	0	0	0

LISC
Flow of execution



LISC Evaluation Issues

- It is very difficult to reproduce an experiment without the original files (.imap or .auto, and test .bin)
- The set of .imap files provided by the authors was incomplete, and most files were outdated and did not work
- Generating the code generator dumps for some specifica programs was very difficult
- The generation of the P_{test} set was not clear

LISC Evaluation Experiment

- Solution: perform a new experiment based on the completeness experiment described

- P_{test} used: all executable files found in Ubunt-14.04 server paths
  - Around 1349 executables, but not all of them could be disassembled
  - Total number of instructions: 51.857.724
  - Number of Unique instructions: 4.183.060

LISC Evaluation Experiment

- P_{train}: started with *openssl-1.0.0.f* + *binutils-2.24*
  - Add *ffmpeg-3.2* with no optimizations
  - Add *glibc-2.19*
  - Add *ffmpeg-3.2* with optimizations
  - Add *gstreamer-1.2.4*
  - Add Manually defined instructions

- Total Recall was not computed
  - Script in the package lacked a dependency

LISC Evaluation Conclusions

- The authors propose a novel approach for automatically translating assembly to intermediate language
- The approach is architecture neutral and based on learning from well teste compiler code
- The software provided by the authors is simple to use, but unreliable
- The lack of proper documentation makes it hard to reproduce results

THANK YOU!