Абстрактная интерпретация бинарного кода как универсальная платформа анализа

Соловьев М.А., Бакулин М.Г., Горбачев М.С., Манушин Д.В., Падарян В.А., Панасенко С.С.

{icee,bakulinm,sadbear,dman95,vartan,spanasenko}@ispras.ru

РусКрипто'2019

21.03.2019

ИСП РАН

ИСП РАН

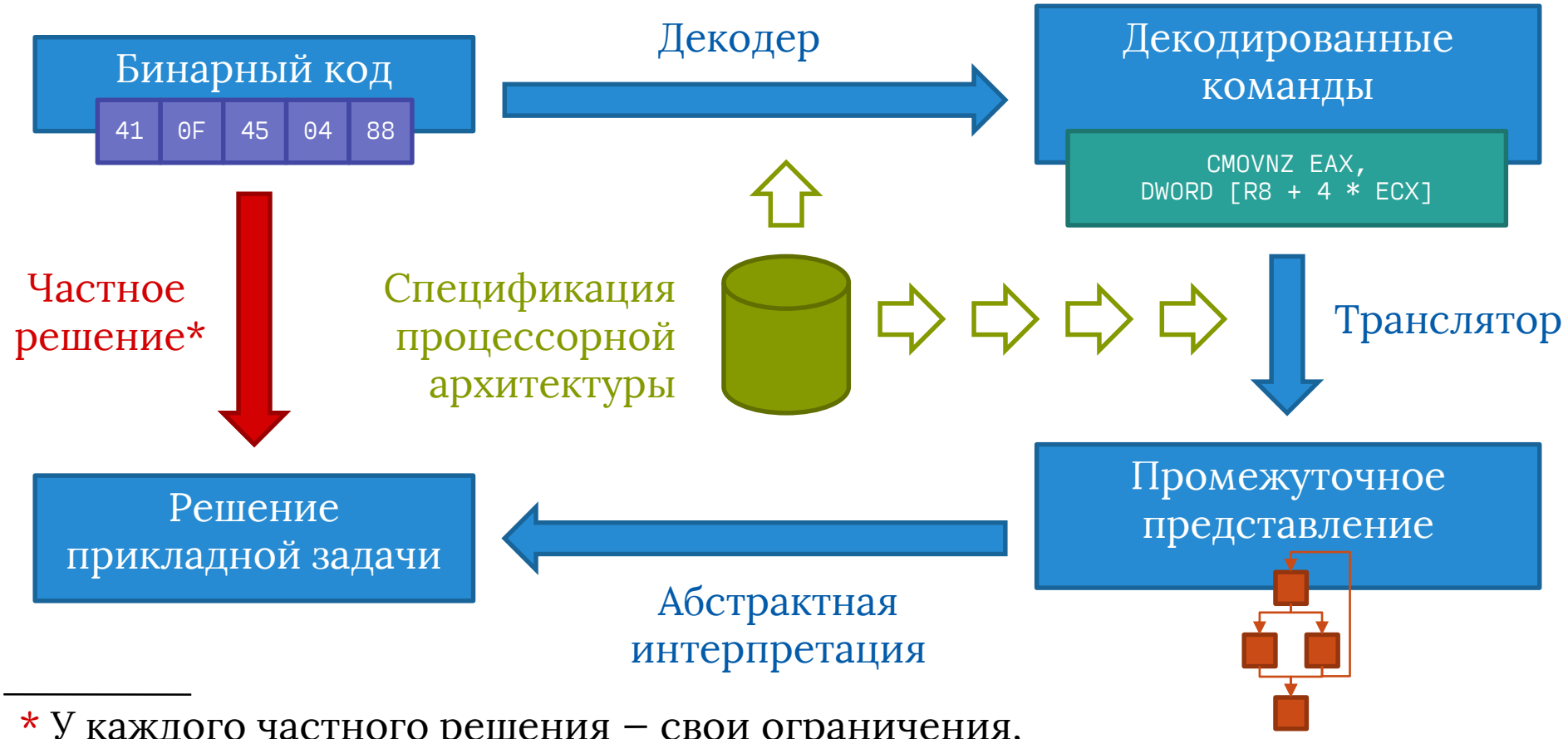
Анализ бинарного кода

- Цели:
  - разработка безопасного ПО;
  - сертификация и тематические исследования;
  - проблемно-ориентированные НИР.
- Возможные условия анализа:
  - отсутствие исходного кода, отладочной информации, символов;
  - при наличии исходного кода – совместный анализ, в т.ч. на предмет оценки критичности ошибок;
  - полносистемный анализ с учётом особенностей аппаратуры – отображение страниц, особенности выборки команд, обработки исключений и т.п.

Автоматизация анализа

- **ТРАЛ** – среда анализа бинарного кода по динамическим профилям работы целевой системы:
  - полносистемный анализ потоков данных и управления;
  - прикладные задачи: восстановление форматов сетевых сообщений, интерактивное построение схем работы алгоритмов, определение условий срабатывания переходов…
- **Тенденция:** применение автоматических цепочек анализа, включающих различные инструменты анализа.
- **Трудности:** различные неточности, ошибки «на стыке» работы разных инструментов в цепочках, дублирование кода в разных инструментах.
- **Решение:** единая платформа анализа бинарного кода – набор библиотек и баз данных для повторного использования разными инструментами.

Универсальная платформа анализа бинарного кода (Glassfrog)



* У каждого частного решения – свои ограничения, неточности, ошибки.

ИСП РАН

Компоненты платформы

- Декодер машинных команд:
  - быстрая адаптация к новым процессорным архитектурам, расширениям наборов команд;
  - универсальное представление результатов декодирования.
- Промежуточное представление:
  - перевод исследуемого кода в представление, не привязанное к конкретной целевой процессорной архитектуре;
  - «терминологическая база» для разработки алгоритмов анализа бинарного кода.
- Абстрактная интерпретация для формализации постановки прикладных задач.
- Реализация базовых алгоритмов анализа кода.

Существующие открытые решения

- angr (VEX);
- B2R2 (LowUIR);
- BAP (BIL);
- BINSEC (DBA);
- BinNavi (REIL);
- BitBlaze (Vine);
- Insight (Microcode);
- Jakstab (SSL);
- Miasm (Miasm IR);
- QEMU (TCG);
- radare2 (ESIL);
- rev.ng (LLVM).

Часть этих решений рассмотрена в статье:

Соловьев М.А., Бакулин М.Г., Горбачев М.С., Манушин Д.В., Падарян В.А., Панасенко С.С. О новом поколении промежуточных представлений, применяемом для анализа бинарного кода // Труды ИСП РАН, том 30, вып. 6, 2018 г., стр. 39-68. DOI: 10.15514/ISPRAS-2018-30(6)-3.

См. также:

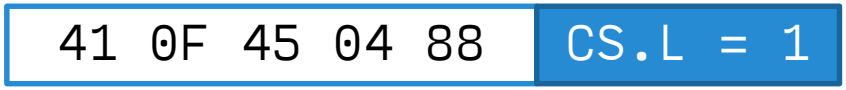
S. Kim, M. Faerevaag, M. Jung, S. J. D. Oh, J. Lee, and S. K. Cha, “Testing intermediate representations for binary analysis,” in Proceedings of the International Conference on Automated Software Engineering, 2017, pp. 353–364.

Особенности современных процессорных архитектур

Особенности	ARM	AVR	MIPS	PowerPC	RISC-V	TMS320	X86	Xtensa
Кодировка команд								
Переменная длина кодировки команд	•	•			•	•	•	•
Декодирование зависит от режима процессора	•					•	•	
Явный параллелизм на уровне команд						•		
Различные версии и наборы расширений команд	•		•		•	•	•	
Особенности выборки команд								
Слоты задержки			•			•		
Аппаратная поддержка циклов								•
«Не-фон-Неймановская» память		•						

Декодирование команды

Или «почему не подходит Capstone?»



Вход: код команды и контекст
Контекст – режим декодирования, поддерживаемые наборы команд

Выход: морфемы и операнды
Морфема – составные части мнемоники, а также префиксы и суффиксы
Операнд – режим адресации и конкретные значения атрибутов

Структура команды

- Структурное описание декодированной команды не зависит от целевой процессорной архитектуры.
- Морфема – мнемоника или её часть, а также префикс, суффикс и т.п.
- Режим адресации – тип операнда, определяет набор атрибутов.
  - Режим `reg`: атрибуты `operand_size` (размер регистрового операнда) и `reg` – номер регистра.


```
{CMOV, NZ}(reg(operand_size = 32, reg = 0 /* EAX */),
    mem(operand_size = 32, address_size = 64,
        sreg = 3 /* DS */, breg = 8 /* R8 */,
        ireg = 1 /* RCX */, scale = 4, disp = null))
```

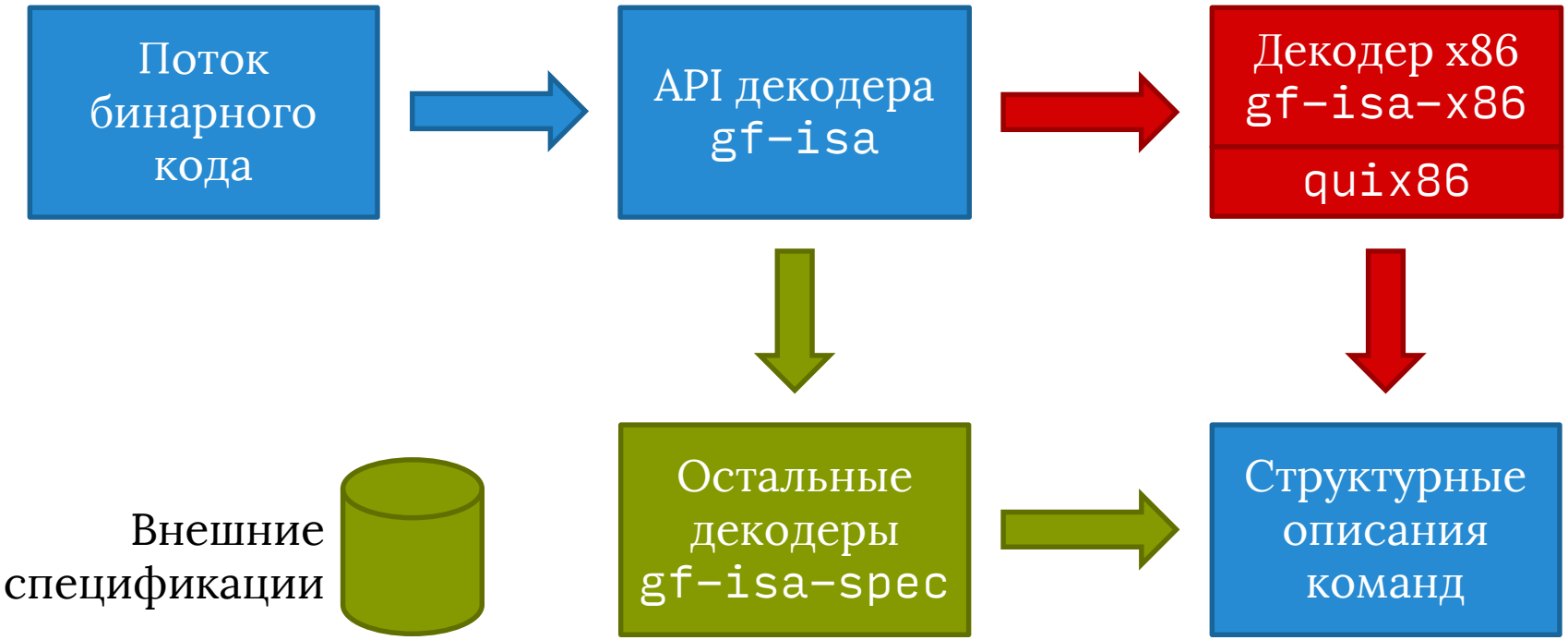
ИСП РАН

Селекторы команд

- Селектор – шаблон команды:
  a) селектор морфем – предикат над множеством морфем;
  b) селектор операндов – фиксирует число операндов, их режимы адресации и накладывает ограничения на значения атрибутов.
- Пример:
  - `{ADD | SUB}(reg(operand_size = 32), mem(sreg = 4 /* FS */))`
  - выбирает все команды, в которых происходит сложение или вычитание 32-разрядного регистра с ячейкой памяти, адресуемой относительно сегмента FS.
- Структура селектора не зависит от целевой процессорной архитектуры.

ИСП РАН

Реализация декодера



ИСП РАН

Представление Pivot 2

- Развитие представления Pivot:
  - В.А. Падарян, М.А. Соловьев, А.И. Кононов. Моделирование операционной семантики машинных инструкций // Программирование, 2011, т. 37, № 3, с. 50-64.
- Не зависит от целевой процессорной архитектуры.
- Все побочные эффекты – явные.
- Переменные в SSA-форме, вместо φ-функций – подстановки на рёбрах.
- Расширяемый словарь элементарных операций, через которые выражается семантика машинных команд.
- Поддержка нескольких адресных пространств, «не-фон-Неймановских» архитектур.

Трансляция в Pivot 2

- Внешние спецификации, задающие операционную семантику отдельных команд целевой процессорной архитектуры.
- Каждая команда представляется как фрагмент представления Pivot 2.
- Сопоставление команд целевой процессорной архитектуры с фрагментами при помощи селекторов команд.
- Отдельный фрагмент описывает поведение процессора «между» командами:
  - особенности выборки команд, обработка исключений…
- Два «вида» представления:
  - высокоуровневый язык для спецификаций – Pivot TR – читает и пишет разработчик спецификации;
  - низкоуровневое представление для проведения анализа – Pivot EX – анализируется алгоритмами.

Pivot EX

- Общий вид представления – направленный мультиграф с петлями.
- Компонента связности графа – фрагмент – всегда имеет один входной и один выходной узел:
  - линейная последовательность фрагментов может быть скомпонована в более крупный фрагмент, это соответствует «склейке» результатов трансляции отдельных команд;
  - при полносистемном анализе между фрагментами-командами подставляются копии фрагмента, описывающего поведение «между» командами.
- Вершина графа – базовый блок, рёбра – переходы.
- Базовый блок содержит последовательность операторов.
- В рамках фрагмента используются временные переменные в SSA-форме, которые могут хранить битовые вектора.

ИСП РАН

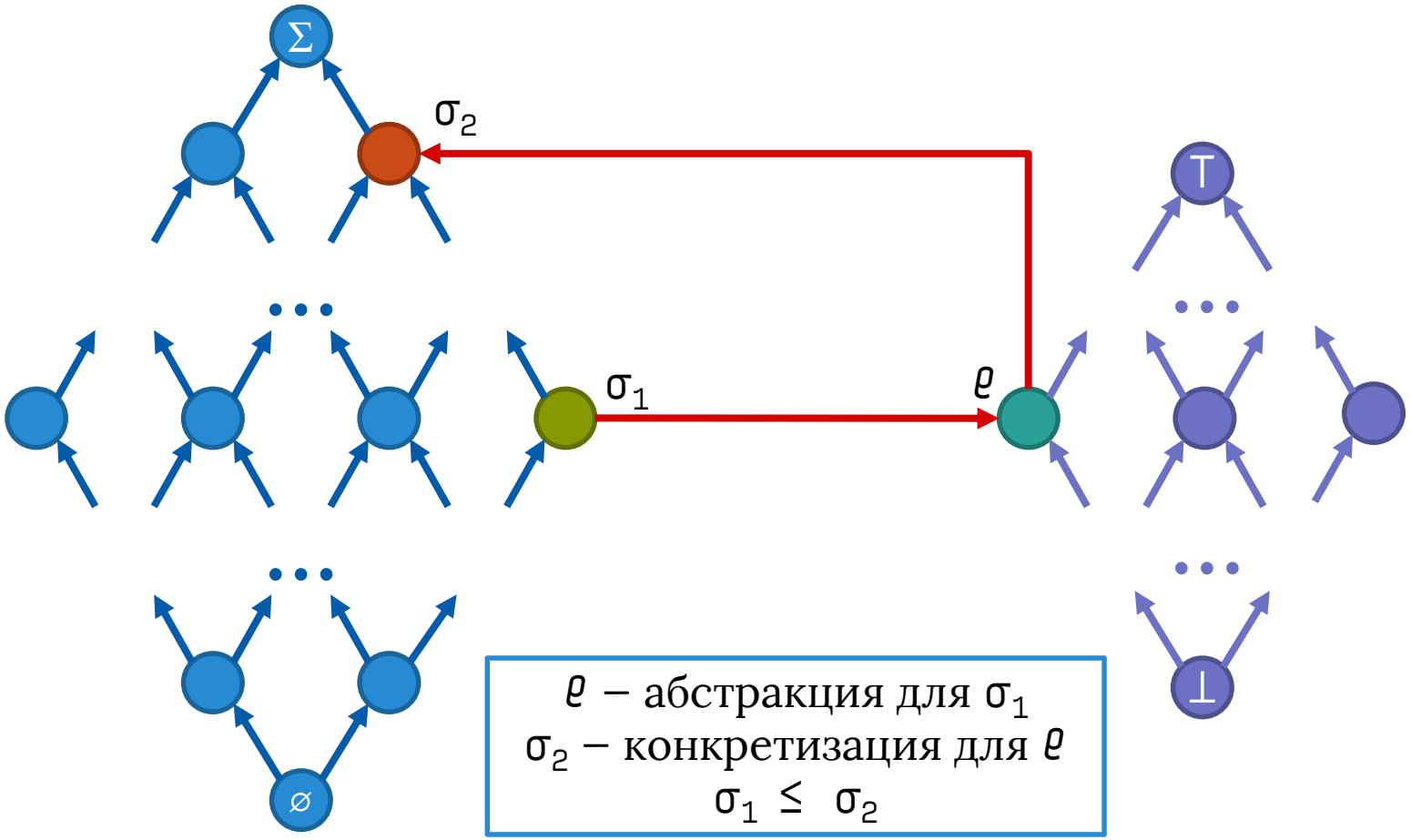
Операторы Pivot EX

1. MIX – копирование временных переменных.
2. EXTRACT – извлечение битового поля из переменной.
3. CONCAT – конкатенация нескольких переменных.
4. INIT – занесение константы в переменную.
5. INVOKE – применение операции.
6. CALL – вызов другого фрагмента кода.
7. LOAD.L – чтение значения из адресного пространства.
8. STORE.L – запись значения в адресное пространство.
9. LOAD.R – чтение значения из адресного пространства с обработкой исключения.
10. STORE.R – запись значения в адресное пространство с обработкой исключения.

Абстрактная интерпретация

- Базовая работа:
  - P. Cousot, R. Cousot. Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. Fourth ACM Symposium on Principles of Programming Languages, 1977, pp. 238-252.
- Основная цель – формализация большого числа разных задач анализа программ при помощи общих математических абстракций.
- Наша задача – адаптация абстрактной интерпретации к решению задач анализа бинарного кода:
  - платформа для максимально быстрого «выхода» на постановку задачи абстрактной интерпретации;
  - готовые компоненты для различных вариантов абстрактной интерпретации.

Абстрактная интерпретация



ϱ – абстракция для σ_1
σ_2 – конкретизация для ϱ
$\sigma_1 \leq \sigma_2$

Конкретные состояния

Абстрактные состояния

Абстрактная интерпретация в Glassfrog

1. Выбор (полу-)решётки абстрактных состояний:
   - как правило, состоит из абстрактного состояния переменных и ячеек адресных пространств.
2. Задание передаточных функций:
   - изменение абстрактного состояния при выполнении оператора;
   - изменение абстрактного состояния при проходе по ребру графа потока управления.
3. Задание начального абстрактного состояния.
4. Применение библиотечного метода решения задачи в варианте:
   - статического анализа;
   - символьного выполнения;
   - динамического анализа, в т.ч. конкретного выполнения.

ИСП РАН

Варианты интерпретации

- **Статический анализ:**
  - вычисление неподвижной точки – MFP-решения задачи анализа потока данных;
  - оптимизационные преобразования;
  - «классический» поиск дефектов.
- **Символьное выполнение:**
  - применение передаточных функций вдоль набора путей, в идеальном случае – MOP-решение задачи анализа потока данных;
  - построение предиката пути;
  - «углублённый» поиск дефектов, построение значений входных переменных.
- **Динамический анализ:**
  - применение передаточных функций вдоль одного пути;
  - частный случай – конкретное выполнение программы.

ИСП РАН

Состояние работы

- Проект поддержан грантом РФФИ 18-07-01256.
- Публикация:
  - Соловьев М.А., Бакулин М.Г., Горбачев М.С., Манушин Д.В., Падарян В.А., Панасенко С.С. О новом поколении промежуточных представлений, применяемом для анализа бинарного кода // Труды ИСП РАН, том 30, вып. 6, 2018 г., стр. 39-68. DOI: 10.15514/ISPRAS-2018-30(6)-3.
- Предварительная версия набора библиотек с открытым исходным кодом планируется к концу текущего года.
- На момент доклада реализованы:
  - общий API декодера и декодирование команд x86;
  - базовые компоненты представления Pivot EX;
  - алгоритм построения неподвижной точки для задач статического анализа;
  - вспомогательные алгоритмы: доминаторы, DU-цепочки…
- Планируется создание интернет-ресурса для совместной работы над базой спецификаций.
- E-mail для обсуждения проекта: icee@ispras.ru.

ИСП РАН

www.ispras.ru