Министерство образования и науки Российской Федерации

МОСКОВСКИЙ ФИЗИКО-ТЕХНИЧЕСКИЙ ИНСТИТУТ
(государственный университет)

ФАКУЛЬТЕТ РАДИОТЕХНИКИ И КИБЕРНЕТИКИ

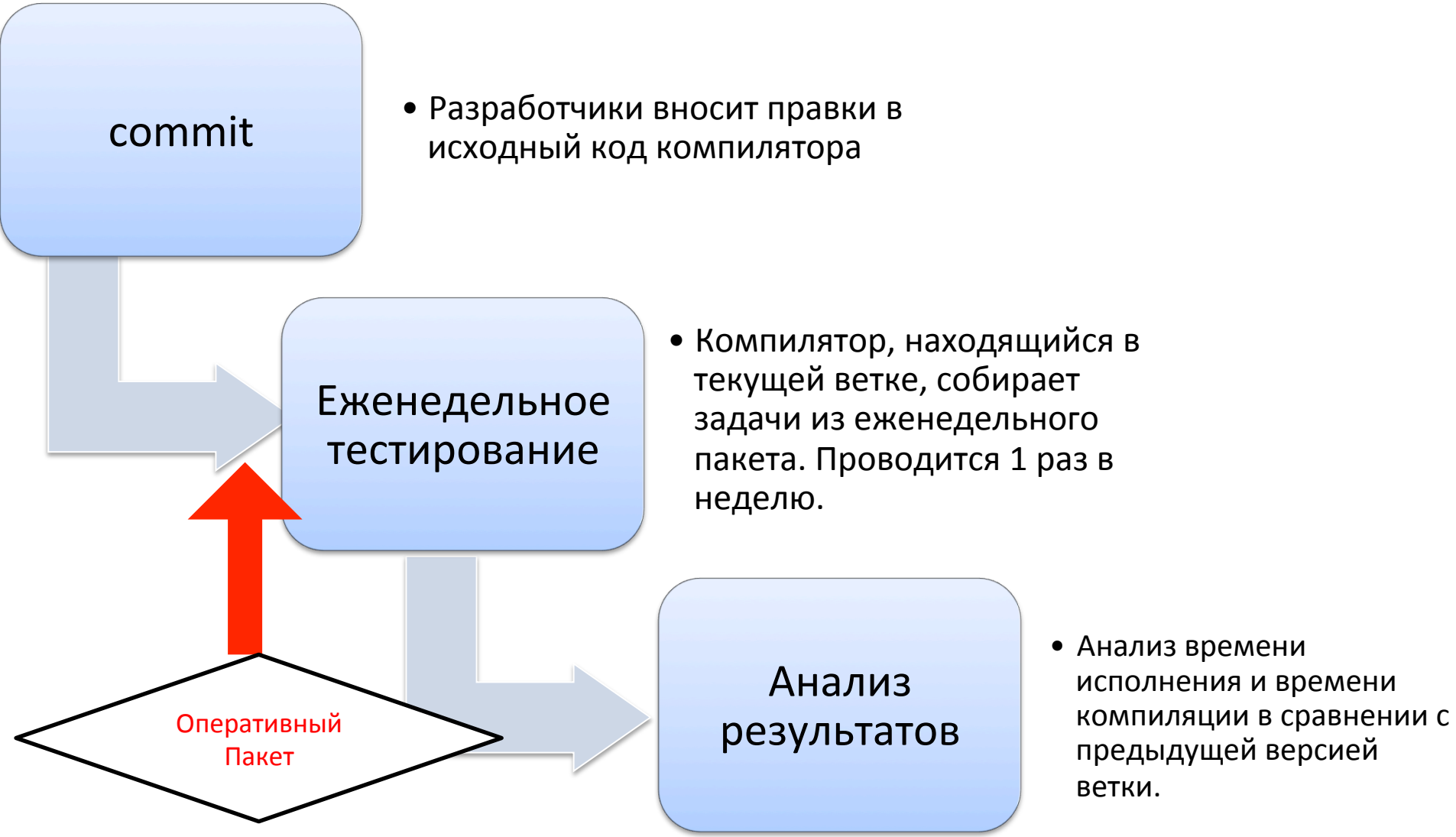
КАФЕДРА ИНФОРМАТИКИ И ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

РАЗРАБОТКА ПАКЕТА КОРОТКИХ ТЕСТОВ ДЛЯ ТЕСТИРОВАНИЯ ПРОИЗВОДИТЕЛЬНОСТИ ЯЗЫКОВОГО КОМПИЛЯТОРА НА БОЛЬШИХ ЗАДАЧАХ

Семёнов Никита

МФТИ, июнь 2014

Система тестирования языкового компилятора в компании МЦСТ



- Разработчики вносит правки в исходный код компилятора
- Компилятор, находящийся в текущей ветке, собирает задачи из еженедельного пакета. Проводится 1 раз в неделю.
- Анализ времени исполнения и времени компиляции в сравнении с предыдущей версией ветки.

Задачи еженедельного тестирования

Пакет SPEC :

- Версии : SPEC 95, SPEC 2000, SPEC 2004, SPEC 2006
- Целочисленные и плавающие задачи
- Большое количество задач
  16 задач на версию
- Большое время исполнения каждой задачи

  10 минут для SPEC 95, 00, 04.
  1-2 часа для SPEC-2006
- Проблема с покрытием исходного кода

Задачи еженедельного тестирования

- EML– мультимедийная библиотека.
  Большое количество файлов и подключаемых библиотек – проблема при поиске и разборе деградации компилятора
- USERTASK– задачи пользователей.
  Поступают из проектных институтов.
  Большое количество файлов в задачах, ошибки в исходном коде (задача ALMAZ) – трудности в анализе и разборе деградаций

Состав оперативного пакета (ОП)

- SPECPERF : выделенные из задач SPEC тесты
- EML : отдельный пакет тестов, выделенных из задач мультимедийной библиотеки EML
- USERTASK : тесты из задач пользователей и сами задачи пользователей

Все задачи пакета исполняются 5000 – 1000000 тактов

Основные характеристики ОП

- После каждого commit’а разработчика запускается тестирование. Если на каком-либо тесте деградация производительности больше 20%, commit блокируется
- Портативен, запускается на симуляторе
- Позволяет оперативно реагировать на проблемы в развитии компилятора и, в силу небольших размеров тестов, быстро выявлять причину деградации – в отличие от еженедельного тестирования

Проблемы ОП

- Тесты исполняются слишком мало тактов – результаты запусков таких тестов заметно «плавают»
- Многие тесты не репрезентативны – запускаются отдельные участки кода процедур задач из пакета SPEC, покрытие не совпадает, контекст нарушен
- РЕЗУЛЬТАТ : деградации, выявленные в процессе еженедельного тестирования, не проявляются в ОП, некорректные commit’ы.

Цель работы

Необходимо разработать методику выделения тестов из больших задач, при этом тесты должны обладать следующими характеристиками:

- Тесты должен быть выделен для «горячих» процедур
- Покрытие теста должно максимально соответствовать покрытию исходной задачи
- Время исполнения теста 5000 – 1000000 тактов
- Контекст исполнения должен быть сохранён
- Тестируемая процедура должна исполняться большую часть времени

Методика формирования коротких тестов

- Горячая процедура

  С помощью утилиты dprof выбирается самая горячая процедура.

- Формирование исходного кода

  1. В файл теста включается исходный код процедуры и все вызванные и подставленные процедуры.

  2. Тело вызванных, но не подставленных процедур заменяется эквивалентом, исполняющимся минимальное время.

  3. Тестируемая процедура вызывается необходимое количество раз, чтобы тест исполнялся 5000 – 1000000 тактов.

Методика формирования коротких тестов

- Формирование входных данных

  1.Сбор входных данных из задачи. Формирование глобальных массивов входных данных (по возможности).
  2. При сложной организации входных данных используется dump памяти исходной задачи и переписывается malloc() для выделения памяти из дампа.

- Стабилизация и корректировка профиля исполнения

  Покрытие исходного кода со счётчиками теста должно соответствовать покрытию на исходной задаче. Этот показатель регулируется входными данными.

Методика проверки корректности тестов

- Работоспособность в режимах –mptr128, -mptr64, -mptr32 в любой комбинации с –O3, -O2, -O0
- Работоспособность на платформах SPARC и Elbrus
- Проверка корректности исполнения
- Совпадение покрытия теста и покрытия исходного кода задачи
- Присутствие тела inline процедур
- Проверка репрезентативности теста

Проверка репрезентативности теста

Собираем тесты и задачу с опциями, существенно изменяющими контекст исполнения:

- Отключение части цикловых оптимизаций. Изменение работы компилятора по слиянию кода. Сильное изменение контекста исполнения.
  -fgos-Line=O2Line
- Линейка дополнительного набора фаз оптимимзаций для работы с памятью в циклах – prefetch, cashopt.
  -fgos-Line=floopLine
- Отключение основных цикловых оптимизаций – softpipeling, nesting, unroll
  --exclude_opt_list=LPO_SOFTPIPE
  --exclude_opt_list=MCO_NESTING
  --exclude_opt_list=LPO_UROLL

Проверка репрезентативности теста

- Задача под утилитой dprof профилируется, анализируется время исполнения «горячих» процедур до и после применения оптимизаций
- Вычисляется коэффициент ускорения

$$k_1 = \frac{t_0}{t}$$

- Аналогичная последовательность действий с выделенным тестом

$$K_2 = \frac{T_0}{T}$$

- Чем ближе отношение коэффициентов $\frac{k_1}{K_2}$ к 1, тем более репрезентативен и корректен тест

Результаты

- Покрытие репрезентативными тестами пакета SPEC увеличилось С 50% до 90%
- Отношение коэффициентов ускорения задач и тестов для непокрытой части пакета на приведённых опциях :

задача	Отношение коэффициентов
175.vpr	i-175_1.c нерепрезентативен – выделен не для горячей процедуры
132.ijpeg	1.42
164.gzip	1.28

Результаты

Для покрытой части пакета :

300.twolf	1.12
255.vortex	1.03
252.eon	1.064
183.equake	1.038
Usertask - LU	1.004
Usertask - Linlst	1.013

Результаты

- Разработана и внедрена в систему тестирования методика выделения тестов из больших задач для повышения репрезентативности Оперативного Пакета.
- Разработана и внедрена в систему тестирования методика проверки репрезентативности тестов на соответствие исходной задаче
- Тесты, выделенные для EML , USERTASK и SPECPERF пакетов позволили ускорить анализ причин ошибок в языковом компиляторе и оперативность реагирования между сеансами еженедельного тестирования