1 слайд -

2 слайд

В результате решения задач, поставленных в прошлой курсовой работе, был определен метод программного анализа сложности алгоритмов с использованием линейной регрессии, организовано построение графика, разработан язык Algolite для описания алгоритмов и транслятор (точнее, транскомпилятор) программного кода с данного языка в язык C++. Программа на C++ подсчитывала число выполнимых элементарных операций на интервале, задающимся пользователем, и определяла асимптотический класс сложности алгоритма с помощью метода линейной регрессии.

3 слайд

Актуальность работы обусловлена наличием излишних затрат сил разработчиков на ручное определение сложности алгоритмов, применяемых ими при конструировании программных систем. Основной целью данной работы является расширение и оптимизация функционала программы для автоматизированного определение класса сложности.

4 слайд

Научная новизна и уникальность работы заключается в применении  
оригинального способа анализа сложности с помощью программных средств, а значимость выражается в пользе, что принесёт разрабатываемый программный комплекс при автоматизации процесса определения сложности алгоритмов.

5 слайд

- Расширение функционала языка описания алгоритмов Algolite, ;  
- Усовершенствование имеющихся методов определения класса сложности;  
- Оптимизировать процесс вставки счётчиков для определения зависимости числа элементарных операций от некоторого значения задаваемого параметра;

6 слайд

Для решение первой поставленной задачи было решено внедрить в язык описания алгоритмов новые конструкции, что описываются следующим образом с помощью РБНФ. Данные механизмы позволят создавать более гибкие конструкции при описании некоторого алгоритма (например, благодаря данному механизму можно избежать подсчёта лишнего числа операций и возможного получения ошибочного класса сложности).

7 слайд

Имеющийся алгоритм, подразумевающий применение метода линейной регрессии, можно применять постепенно на всём интервале вычисленных точек, представляющих собой пары значений параметра и числа элементарных операций, поделённом на некоторое число делений. На каждом из интервалов, отсчитываемых от начала, значение шага будет вычисляться как общее число точек, делённое на число интервалов целочисленным образом. При этом к первому интервалу будет добавлено значение равное общему числу точек минус шаг, умноженный на число интервалов. Данное значение необходимо для того, чтобы захватить весь интервал точек. В результате будет получено число увеличений значения каждой из отслеживаемых ошибок, соответствующих предположительным классам сложности, число изменений степени для каждого из таких классов и значения последних ошибки, которые отображают вычисленные в результате применения метода линейной регрессии значения средних ошибок на всём интервале рассматриваемых точек. В зависимости от того, какая из последних ошибок имеет наименьшее значение, выбирается предположительный класс сложности. Для окончательного определения результата были приняты следующие ограничения: если ошибка предположительного класса сложности увеличивалась менее половины раз на протяжении всего анализа или имеет конечное значение меньшее 1e-04 (0.0001) и при этом число изменений степени старшего слагаемого в конечной функции сложности происходило менее 5 раз, то предположенный класс сложности является верным, после чего может быть выдан асимптотический результат. В противном случае класс сложности будет определён как «Unknown», так как для определения класса требуются дополнительные исследования, которые программа не способна отобразить. Данное решение привносит эвристический элемент в общий алгоритм, но в общем случае такой ответ получается довольно редко (например, при выборе пользователем некорректных параметров для проведения анализа сложности).

8 слайд

К имеющимся классам сложности, которые могут быть определены программой, был добавлен класс экспоненциальной сложности. Так как в программе применяется логарифм с основанием 10, а … так можно применять линейную регрессию на различных интервалах согласно предыдущему нововведению, тем самым переходя к следующей стадии определения класса сложности.

9 слайд

Для того чтобы повысить информативность выдаваемого результата, можно дополнительно определять точный вид функции сложности рассматриваемого алгоритма. В случае принадлежности такой функции к полиномиальному классу можно применить метод полиномиальной регрессии, представляющую собой в статистике форму регрессионного анализа, в которой связь между некоторыми переменными определяется как полином k-й степени… можно составить точный вид функции сложности рассматриваемого алгоритма.

10 слайд

Ввиду того, что подсчет числа элементарных операций, которые выполняются при некотором значении параметра, обеспечивается буквальным запуском программы, образованной транскомпиляцией текста рассматриваемого алгоритма, возникает необходимость ускорения данного процесса. Заметный прирост скорости можно обрести путём внедрения средств параллельных вычислений.

При параллельном подсчёте числа операций каждый из запущенных потоков будет иметь собственную переменную counter, в которую будет записывать конечный результат. В итоге все полученные пары значений параметра и числа элементарных операций будут записаны в общий набор.

При этом стоит учесть, что от общего количества потоков и номера конкретного поток зависит начальное значение параметра для потока и то, с каким шагом ему необходимо увеличить значение рассматриваемого параметра. Начальное значение вычисляется как par = par\_min + par\_step \* thread\_number, а изменение как par += par\_step \* thread\_amount (здесь par – имя параметра, задаваемое пользователем, par\_min – минимальное значение параметра, par\_step – шаг, задаваемый пользователем, thread\_number – номер потока, thread\_amount – общее число потоков). Значение параметра для каждого из потоков будет увеличиваться при условии, что оно не превосходит par\_max (где par\_max – максимальное значение параметра).

11 слайд

…В данном программном комплексе используется компилятор g++ из набора инструментов MinGW. При его установке была выбрана thread-модель Win32, что не поддерживает конструкцию std::thread в естественном виде. Для реализации работы с потоками была подключена библиотека mingw-std-threads, решающая данную проблему. Во время установки указанного инструментария можно выбрать thread-модель POSIX-threads, представляющую реализацию POSIX-совместимых потоков (нитей), но в данном случае пришлось бы использовать конструкции вида pthread из заголовочного файла pthread.h. В основном они выбираются для организации портирования многопоточных программ из UNIX-подобных систем…

12 слайд -

13 слайд -

14 слайд -

15 слайд -

16 слайд

Расширение функционала языка описания алгоритмов позволяет создавать более гибкие конструкции, а также выделять те области алгоритма, которые пользователь не хочет учитывать при анализе. Данный подход реализует оптимизацию на уровне пользовательского контроля.

17 слайд -

18 слайд

В результате расширения этапа анализа алгоритма с применением линейной регрессии увеличилась точность определения асимптотического класса сложности. На основании величины средней ошибки и выбранных ограничений выбирается верный класс сложности или ответ представляется как «Unknown». Данный этап позволяет точнее определять асимптотический класс сложности (в том случае, если функция принимает вид одного из классов, которые оно может определить).

19 слайд

Если функция сложности рассматриваемого алгоритма относится к полиномиальному классу, программа выдаст более подробный ответ

20 слайд -

21 слайд

Чтобы убедиться в оптимизации процесса подсчёта числа элементарных операций на некоторых значениях параметра путём введения средств параллельного вычисления достаточно запустить анализ вышеуказанного алгоритма на достаточно большом интервале параметров. На слайде преимущества использования параллельного подхода (результаты приведены в количестве миллисекунд, затраченных на подсчёт числа операций на каждой из 10 итераций).

22 слайд

В результате работы в приложение, организующее транскомпиляцию кода на языке описания алгоритмов Algolite в код на языке C++ с дальнейшим определением класса сложности рассматриваемого алгоритма и построением графика сложности, были введены новые функциональные элементы языка описания алгоритмов, методы определения асимптотического и точного вида функции сложности и механизмы параллельного вычисления.

С помощью усовершенствованной программы были проведены тестирования на некотором числе существующих алгоритмов, повсеместно применяемых разработчиками, отображающие высокую точность определения классов сложности, что позволяет ускорить процесс внедрения или отсеивания алгоритма при конструировании системы. Скорость работы программы была заметно увеличена благодаря перечисленным нововведениям.

23 слайд -