г) Объекты основаны на бизнес-концепциях, а не на структуре базы данных;
д) Управление транзакциями и автоматическое создание ключей;
е) Позволяет быстрее разрабатывать приложения.

Для платформы Java существует несколько ORM библиотек. Целью данной работы является исследование эффективности известных решений, а также разработка и улучшение текущей архитектуры.

Список использованных источников:

1. Bauer, G. King, G. Gregory, Java Persistence with Hibernate. Second Edition. // Manning Publication Co, Shelter Island, NY 2016

ДЕКОМПОЗИЦИЯ СИГНАЛА НА ВНУТРЕННИЕ МОДОВЫЕ ФУНКЦИИ С ИСПОЛЬЗОВАНИЕМ ГЕТЕРОГЕННЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Мелещеня Д.В.
Брацевич П.Ю. – канд. техн. наук,доцент

Преобразование Гильберта-Хуанга используется для исследования нестационарных и нелинейных систем. Ключевой частью этого метода анализа данных является декомпозиция исходного сигнала на внутренние эмпирические моды. В классическом методе, предложенном Хуангом для построения мод, используется интерполяция кубическими сплайнами кривых, полученных выделением максимумов и минимумов исходного сигнала. Данная процедура является достаточно трудоемкой, поэтому в данной работе предлагается использовать неоднородную вычислительную систему для сокращения времени расчета.

Анализ данных нашел свое применение как в теоретических исследованиях, так и на практике и помогает выявить ключевые характеристики сигналов, полученных в результате измерения физических величин или численного моделирования. Однако зачастую этим данным присущи следующие недостатки: данные нестационарные и представляют нелинейный процесс. Как следствие выбор метода анализа сильно ограничен. Один из подходов к изучению нелинейных нестационарных данных – преобразование Гильберта-Хуанга.

Преобразование Гильберта-Хуанга состоит из двух этапов: эмпирическая модовая декомпозиция (EMD), преобразование Гильберта полученной декомпозиции для вычисления мгновенных фазы и частоты (HSA).

Декомпозиция на модовые функции – итеративная процедура, ставящая в соответствие исходному сигналу набор эмпирических мод (IMF). В классическом определении данном Хуангом IMF собой представляет функцию, удовлетворяющую следующим двум условиям: на всем наборе данных количество пересечений оси абсцисс и экстремумов должно быть равно или отличаться на единицу; в любой точке среднее значение огибающих определяемых максимумами и минимумами должно быть равным нулю. Этим обеспечивается корректность, с физической точки зрения, результатов преобразования Гильберта.

Фактически алгоритм модовой декомпозиции, или просеивания, сводится к следующему. Строятся верхняя и нижняя огибающие, для нахождения которых выделяются локальные максимумы (для верхней) и минимумы (для нижней) на которых строятся интерполяционные сплайны, представляющие собой искомую функцию. Следующим этапом является нахождение разности между средним значением огибающих и исходным сигналом. Далее, если остаток удовлетворяет критерию остановки, он считается очередной модовой функцией, в противном случае разность принимают за исходный сигнал и алгоритм повторяется. После нахождения IMF, ее вычитают из исходного сигнала и, если разность не является монотонной функцией, то алгоритм просеивания повторяется.

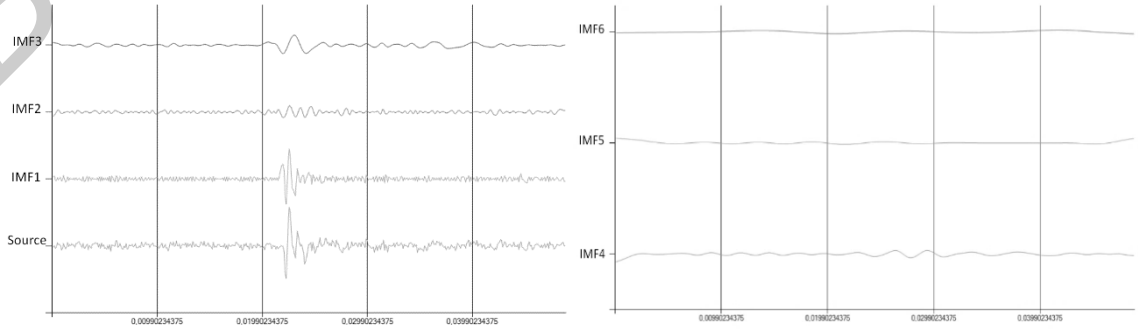

Рис. 1 – Пример разложения исходного сигнала на эмпирические модовые функции (IMF)

В приведенном алгоритме наиболее затратным по времени является построение огибающих. Поэтому ускорение этого этапа обеспечивает сокращение времени выполнения модовой декомпозиции в целом. Следует отметить, что сигнал в общем случае представляет собой достаточно длинную последовательность точек, так что устранение зависимости между ними в процессе расчета и их параллельная обработка позволяет добиться прироста производительности. Для достижения высокой степени параллелизма было предложено использовать гетерогенную вычислительную систему, состоящую из процессора общего назначения (CPU) и графического процессора (GPU). Такой выбор обусловлен тем, что в силу своих архитектурных особенностей, GPU обеспечивает одновременный запуск большого числа потоков, выполняющих вычисления. CPU в данной связке служит для координации работы системы в целом. В качестве платформы реализации был выбран фреймворк OpenCL, предназначенный для написания и запуска вычислений на различных аппаратных платформах.

Для построения огибающей используется сплайн Катмула-Рома. Суть метода заключается в том, что для каждого интервала исходной функции находится свой полином, описывающий кривую на данном участке. При этом значение в некоторой точке произвольного интервала определяется по следующей формуле:

$$p(x) = h_{00}(t)p_k + h_{01}(t)p(x_{k+1} - x_k)m_k + h_{01}(t)p_{k+1} + h_{11}(t)(x_{k+1} - x_k)m_{k+1},$$

где

$$\mathrm{t} = (\mathrm{x} - x_k)/(x_{k+1} - x_k),$$

$$m_n = (p_{n+1} - p_{n-1})/(t_{n+1} - t_{n-1}),$$

$h_{00}, h_{01}, h_{01}, h_{11}$ – базисные функции Эрмита.

Поскольку вычисление значений на каждом интервале не зависит от значений на других интервалах, расчет для каждого отрезка выполняется в отдельном потоке на отдельном вычислительном ядре GPU. Таким образом, с учетом программной модели OpenCL, итерация алгоритма модовой состоит из следующих этапов:

1. Определение максимумов и минимумов исходного сигнала;
2. Копирование исходного сигнала и массивов экстремумов в память устройства (GPU);
3. Построение сплайнов огибающих;
4. Вычисление среднего и разности между средним и исходным сигналом;
5. Вычисление критерия остановки (в простейшем случае инкрементирование счетчика);
6. Если критерий достигнут – эмпирическая мода найдена – скопировать моду в память хоста;
7. Если критерий не достигнут – скопировать разность в участок памяти с исходным сигналом и повторить этапы 3-5.

В результате было разработано программное средство, позволявшее производить декомпозицию сигнала на внутренние модовые функции с использованием графического процессора. Для построение огибающих были использованы сплайны Катмула-Рома. Эксперименты на графическом адаптере AMD Radeon HD 6470M показали, что данная реализация эффективна для анализа последовательностей размер которых превышает $8*10^4$ точек. В противном случае затраты на копирование данных из памяти хоста в память вычислительного устройства составляют существенную часть времени, что делает неэффективным применение GPU.

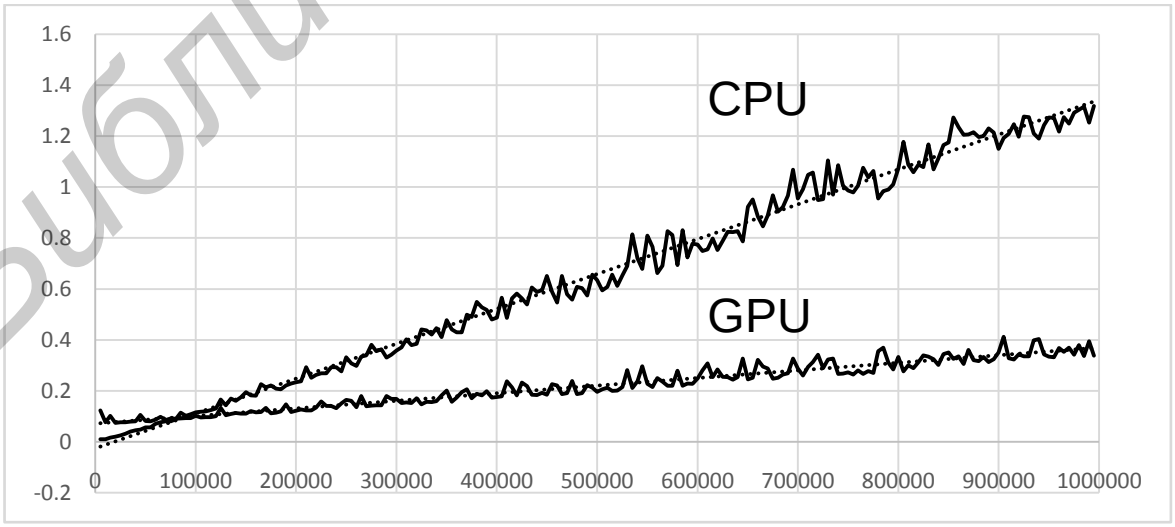

Рис. 2 – Сравнение зависимости времени декомпозиции сигнала от его длины при запуске на гарфическом и центральном процессорах

Список использованных источников:

1. Munshi, A. The OpenCL Specification / A. Munshi // Khronos OpenCL Working Group. – March 2014.
2. Роджерс, Д. Математические основы машинной графики / Д. Роджерс, Дж. Адамс. – М.: Мир. – 2001. – 604 с.
3. N. E. Huang The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis / Huang N. E., Shen Z., Long S. R., Wu M. C., Shih H. H. and other. – Proc. R. Soc. Lond. A. – 1998. – T.454. – c.903 – 995.

ПРОГРАММНОЕ СРЕДСТВО «АТТЕСТАЦИЯ СОТРУДНИКОВ IT-КОМПАНИИ»

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Метельский И.О.

Мельникова Е.В. – ассистент каф. ПОИТ

Рано или поздно, каждый человек работающий в IT сфере сталкивается с тем, что ему необходимо пройти аттестацию для проверки его текущих знаний. Однако для многих сотрудников это является целым испытанием несмотря на то, что в большинстве случаев они готовы пройти этот тест на отлично. Дело заключается в психологическом состоянии каждого человека. Многим людям некомфортно общаться с незнакомыми для них людьми для выяснения их уровня знания. Помимо этого, аттестация проводится часто всего ни у одного человека, а у группы лиц, которые поочередно проходят аттестацию и подтверждают свою квалификацию. На это уходит очень много времени и поэтому очень кстати пришлось бы программное средство, которое бы автоматизировало прохождение аттестации сотрудниками на базе тестирования.

История тестирования, как инструмента проверки знаний, началась в начале 20-го века. Соединенные штаты Америки стали родиной тестов. Затем они очень широко распространились в школах и ВУЗах Великобритании, а после охватили все оставшиеся страны.

Дискуссии о том, какая форма проверки знаний более правильная не утихают до сих пор. Но в последнее время тестирование все-таки стало вытеснять традиционные экзамены и контрольные работы. В какой-то степени, за это надо сказать спасибо глобальной компьютеризации, в том числе и учебного процесса. А помимо этого ученые доказали, что тестирование является более легкой формой проверки знаний с точки зрения эмоционального состояния. Традиционный экзамен для большинства людей – это всегда сильный стресс.

Реализации подобных проектов уже можно встретить на просторах сети Интернет, но данный проект будет предназначен именно для сотрудников IT-компании. Основной задачей проекта является проведение тестирований знаний сотрудников, которые могли бы максимально точно показать уровень владения знаниями в определенной сфере. Данное программное обеспечение необходимо для повышения качества тестирования, а также для экономии времени как руководства компании, так и для её сотрудников. Необходимо продумать архитектуру базы данных, в которой можно было бы хранить личные данные сотрудников компании, сами тесты, а также информацию о том, как сотрудники проходят данные тесты. Готовый программный продукт должен иметь простой интерфейс для того, чтоб каждый пользователь мог бы быстро в нем разобраться.

Тестирование позволяет проверить большое количество материала на знание за определенно короткий срок. Что же касаемо устного экзамена, как правило, обсуждается одна тема и не более, а в случае тестов можно затронуть весь материал различными короткими вопросами. Но минусом тестов является то, что часто этот метод проверки знаний подразумевает некие подсказки, которыми если пользоваться, можно ответить на вопросы и без особо серьезной подготовки. Однако есть возможность в тестировании и это обойти. Если выдаваемый результат будет формироваться не только в зависимости от правильных данных ответов, а также и от неправильных, то проходящий человек тест многократно раз задумается, а стоит ли рисковать своим конечным результатом ради возможности угадать ответ.

Программное средство реализовано в виде web-сайта. Предусмотрено, что использовать программу могут актеры со следующими ролями: гость, авторизированный пользователь, сотрудник, модератор, администратор.

Если рассмотреть функции, которые присуще данным ролям, то можно отметить следующее основные:

− Гость. Неавторизированный пользователь, который имеет право просматривать информации о организации, а также для данной роли доступны функции регистрации и авторизации;

− Пользователь. Это авторизированный гость, который имеет право не только просматривать информацию об организации, но и проходить тесты, общедоступные для всех, просматривать результаты данных тестов, анализировать полученные данные с общими результатами. Также для данной роли предусмотрена возможность просмотра статей на разные тематики и их комментирование;

− Сотрудник. Роль, которую может выдать администратор пользователю, который работает в данной компании. Человек, обладающий данной ролью, сможет проходить аттестацию, которую высылает администратор (помимо функций, разрешенный для роли пользователь);

− Администратор. Сотрудник компании, обладающий данной ролью, администрирует полностью все программное средство. Он может распределять роли между пользователями системы, редактировать личные