УДК 681.325.518.5

СИНХРОНИЗАЦИЯ И ВЕКТОРИЗАЦИЯ ПРОЦЕССОВ ПАРАЛЛЕЛЬНОЙ ОБРАБОТКИ СЕЙСМИЧЕСКИХ СИГНАЛОВ С ИСПОЛЬЗОВАНИЕМ ВОЗМОЖНОСТЕЙ JAVA

Х.Н. Зайнидинов
Доктор наук, Ташкентского Университета Информационных Технологий имени Мухаммада Аль-Хоразмий , Заведующий кафедрой Информационные технологии

О.С. Маллаев
Старший преподаватель кафедры "Основы Информатика" Ташкентского Университета Информационных Технологий имени Мухаммада Аль-Хоразмий

С.П. Халилов
Ассистент кафедры "Информационные технологии" Ташкентского Университета Информационных Технологий имени Мухаммада Аль-Хоразмий.

Ташкентский университет информационных технологий имени Мухаммада Аль-Хоразмий, Республика Узбекистан
E-mail: suraj_24@mail.ru

Х.Н. Зайнидинов
Получил степень магистра в области компьютерных наук в Ташкентском государственном техническом университете, Узбекистан, в 1984 году. Он получил степень доктора наук в Санкт-Петербургском электротехническом университете, Россия, в 1993 году. После этого он также получил степень доктора наук в Ташкентском университете. Информационные технологии, Узбекистан, в 2005 году. Его научные интересы включают информатику, моделирование, цифровую обработку сигналов, параллельные алгоритмы и программы.

О.У. Маллаев
Получил степень магистра в области информационная безопасность в Ташкентском университете информационных технологий, Узбекистан, в 2009 году. Сфера его научных интересов - информатика, моделирование, цифровая обработка сигналов, параллельные алгоритмы и программы. Участник проекта А5-021 «Разработка специализированных систем для вейвлет-анализа и обработки сейсмических сигналов. Его научные интересы включают информатику, параллельные алгоритмы и программы.

С.П. Халилов
Получил степень магистра в области разработки прикладного программного обеспечения в Ташкентском университете информационных технологий, Узбекистан, в 2015 году. Сфера его научных интересов - информатика, моделирование, цифровая обработка сигналов, параллельные алгоритмы и программы. Участник проекта А5-021 «Разработка специализированных систем для вейвлет-анализа и обработки сейсмических сигналов.

Аннотация. Широкая популярность сплайн-методов в задачах анализа и обработки сейсмических и геофизических сигналов объясняется тем, что они служат универсальным инструментом приближения и по сравнению с другими математическими методами при равных с ними информационных и аппаратных затратах обеспечивают большую точность. С другой стороны, применяемые в таких системах аппаратные средства

также должны отвечать требованиям высокой скорости обработки. Для достижения высокой скорости обработки необходимо разработать параллельные алгоритмы с использованием возможностей языка программирования Java и реализовать их на многоядерных архитектурах процессоров. Статья предлагает программную реализацию сплайн-методов цифровой обработки сигналов с использованием возможностей языка программирования Java на основе многоядерной архитектуры процессоров. Это позволяет в целом повысить эффективность функционирования систем за счет увеличения скорости обработки данных при установленных показателях точности.

Ключовые словы: Параллельный алгоритм, поток, многоядерный процессор, сплайн, B-сплайн, сейсмический сигнал, векторизация.

Введение. В настоящее время разработчикам приложений необходимы мощные и удобные инструменты для адаптации существующих или написания новых приложений, максимально использующих производительность многоядерных процессоров. Понятно, что часто графический интерфейс приложений написан с использованием Java или .Net языков. Тем не менее, те части приложений, которые требовательны к производительности (фильтры, кодеки, и т.д.), в большинстве случаев реализованы именно на C/C++, и именно в них важно добиться задействования всех возможностей процессора[7-10]. Сейчас уже нет сомнений, что дальнейшее увеличение производительности приложений будет достигаться за счет того, насколько хорошо эти приложения распараллелены и как хорошо они масштабируются с ростом процессоров в системе. Очевидно, что фактическим стандартным набором C/C++ разработчика является Microsoft Visual Studio. Intel Parallel Studio – это набор из нескольких инструментов, который является расширением Microsoft Visual Studio, и позволяющий за счет удобства использования и понятного интерфейса добиваться хорошей эффективности параллельных программ на многоядерных системах. [5,6]

Широко используются два варианта синхронизации: взаимное исключение и условная синхронизация. В случае взаимного исключения один поток блокирует критическую секцию (область кода, которая содержит общие данные), в результате один или более потоков ждут своей очереди зайти в эту область. Это полезно в тех случаях, когда два или более потоков совместно используют одно и то же пространство памяти и выполняются одновременно.

Несмотря на то что существует довольно много методов синхронизации, только несколько из них регулярно используются разработчиками. Применяемые методы в некоторой степени определяются также средой программирования.

Параллельное программирование, связанное с использованием легковесных процессов, или подпроцессов (multithreading, light-weight processes)- концептуальная парадигма, в которой вы разделяете свою программу на два или несколько процессов, которые могут исполняться одновременно.

Материалы и методы. В технических приложениях наиболее употребительными являются сплайны невысокой степени, в частности параболические и кубические [1–4]. Процесс построения таких сплайнов значительно проще, чем процесс построения сплайнов более высокой степени. Значительно упрощаются вычислительные проблемы при обращении к локальной сплайн - аппроксимации, в которых значения приближающей сплайн - функции на каждом отрезке зависят только от значений аппроксимируемой функции из некоторой окрестности этого отрезка. Другой особенностью таких методов является то, что они не требуют решения систем уравнений при нахождении параметров сплайна.

Любой сплайн Sm(x) степени m дефекта 1, интерполирующий заданную функцию f(x) может быть единственным образом представлен B - сплайнами в виде суммы [2, 5, 10]:

$$f(x) \cong S_m(x) = \sum_{i=-1}^{m+1} b_i \cdot B_i(x), \quad a \le x \le b \,, \tag{1}$$

где bi – коэффициенты, $B_i(x)$ - базисный сплайн

В случае, когда используется кубический базисный сплайн, то его значения вычисляются по формуле:

$$B_3 = \begin{cases} 0, x \geq 2, \\ (2-x)^3 / 6, 1 \leq x < 2, \\ 1/6(1+3(1-x)+3(1-x)^2 - 3(1-x)^3), 0 \leq x < 1, \end{cases} \quad (2)$$

а коэффициенты можно определять по следующей формуле:

$$b_i = (1/6)(-f_{i-1} + 8f_i - f_{i+1}); \quad (3)$$

Согласно формуле (1) значение интерполируемой функции в произвольной точке заданного интервала определяется значениями лишь m+1 слагаемых – парных произведений базисных функций на постоянные коэффициенты. Например, кубические B-сплайны требуют четырех базисных слагаемых. Значение функции вычисляется по формуле:

$$f(x) \cong S_3(x) = b_{-1}B_{-1}(x) + b_0B_0(x) + b_1B_1(x) + b_2B_2(x) \text{ при } x \in [0,1] \quad (4)$$

Остальные базисные сплайны на этом под интервале равны нулю и, следовательно, в образовании суммы не участвуют.

Если использовать один основной базисный сплайн и с помощью переменной j задать адреса разных участков основного сплайна, то уравнение (4) принимает вид:

$$S_3[i] = (b[i-1]\,B[j+30]) + b[i]\,B[j+20] + b[i+1]\,B[j+10]) + b[i+2]\,B[j]) \quad (5)$$

Для разработки параллельного алгоритма выделим на потоки [**4, 5, 6**,]. Для выполнения четырех параллельных умножений выделим m=4 потока. В результате получим формулу (6):

$$S_j(L_j : K_j : P_j : T_j) = \begin{cases} L_j = \sum_{i=-1}^{m+1} b_{i-1} B_{i+30}(x); \\ K_j = \sum_{i=-1}^{m+1} b_i B_{i+20}(x); \\ P_j = \sum_{i=-1}^{m+1} b_{i+1} B_{i+10}(x); \\ T_j = \sum_{i=-1}^{m+1} b_{i+2} B_{i+2}(x); \end{cases} \quad (6)$$

где Lj, Kj, Pj и Tj - потоки умножения матрицы на вектор.

Блок-схема алгоритма параллельного вычисления с выделением потоков с применением пакета OpenMP приведена на Рис. 2.

Векторизация – вид распараллеливания программы, при котором однопоточные приложения, выполняющие одну операцию в каждый момент времени, модифицируются для выполнения нескольких однотипных операций одновременно, векторных операций.

Методы векторизации выбираются на основе типа микропроцессора.

Например, Векторный процессор **(Vector processor)** –э то процессор, поддерживающий на уровне системы команд операции для работы с одномерными массивами – векторами (vector).

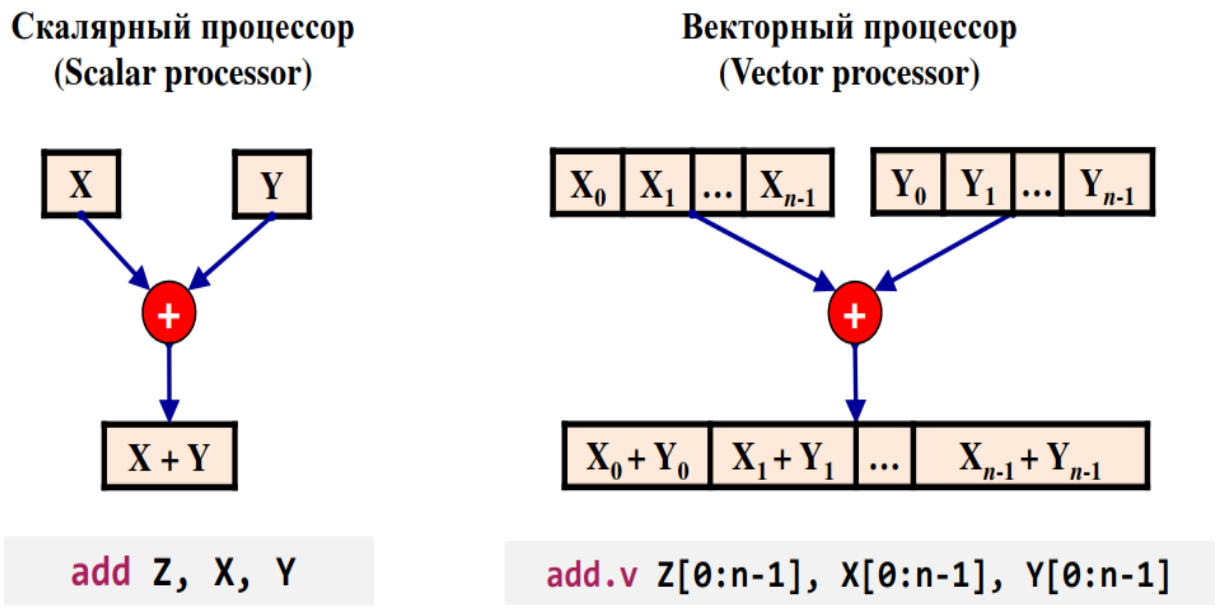

Рисунок 1. Режим работы скалярных и векторных процессоров

Во многих вычислительных программах одна и та же операция применяется к целому набору данных: к массиву, ну либо к его какому-то кусочку. В этом случае наблюдается повторяющаяся модель доступа к данным, а также повторяющаяся операция, которую можно выполнять параллельно. Например, в следующем фрагменте кода показан пример такой повторяющейся модели. В данном примере происходит сложение двух векторов.

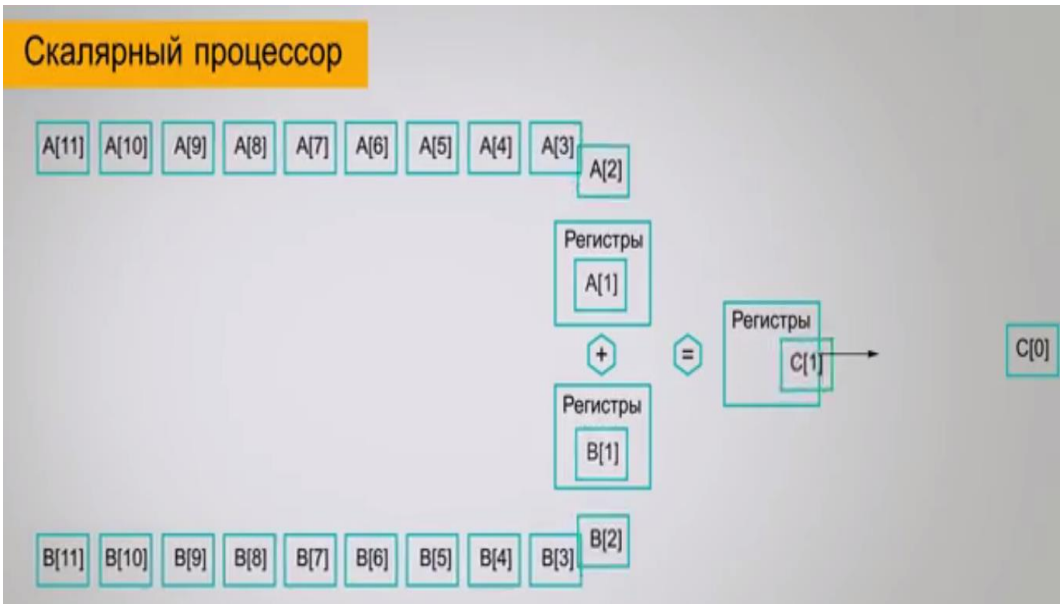

Рисунок 2. Режим программирования обычный скалярных процессоров

Обычный скалярный процессор будет выполнять данную программу следующим образом: сначала загружаются в регистр числа A(0) и B(0), затем они складываются и результат записывается в C(0), далее A увеличивается на 1, в регистры загружаются элементы A1 и B1, и выполняется операция сложения и так далее.

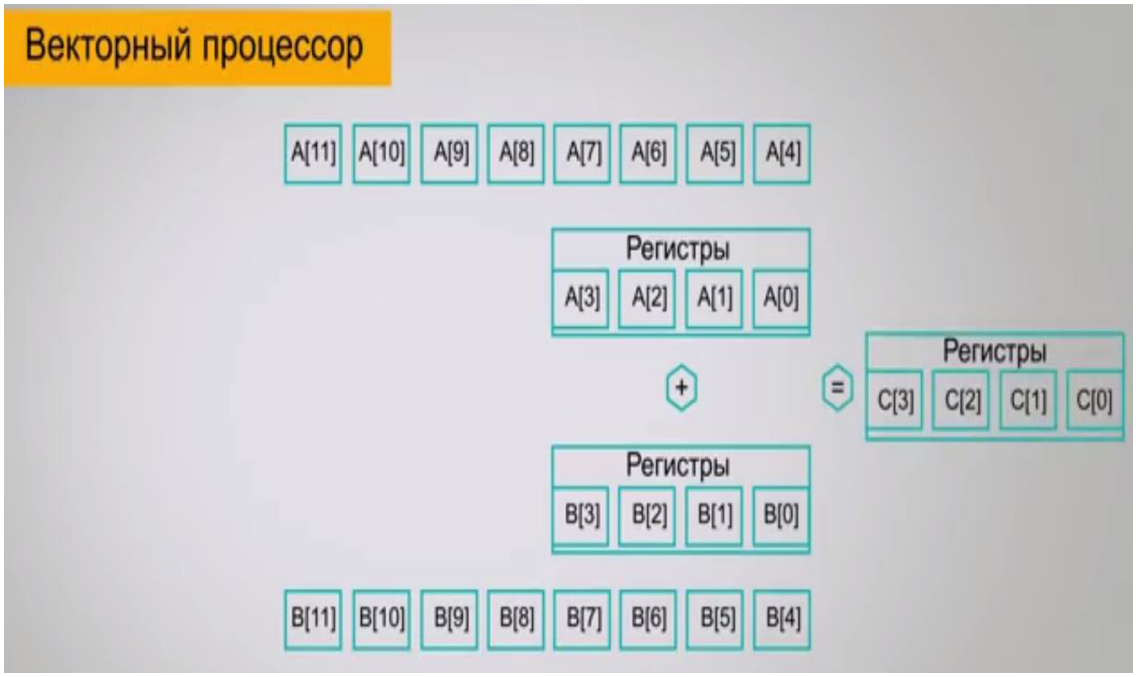

Рисунок 3. Режим программирования векторных процессоров

Если же процессор векторный, то тогда в нём есть векторные регистры, которые позволяют хранить вектор значений, и в процессоре реализованы векторные инструкции. Если воспользоваться векторными инструкциями, то процессор в регистры загрузит сразу несколько значений вектора A и B. Предположим 4 элемента. Тогда в регистрах будут элементы: A(0), A(1), A(2), A(3); и соответственно для вектора B: B(0), B(1), B(2) и B(3). И для всех 4 пар одновременно выполнится операция сложения, а результат запишется в C(0), C(1), C(2) и C(3). Преимущество векторизации в том, что время выполнения векторной операции такое же, как и скалярной, но полезной работы выполняется больше. Одна векторная команда распознаётся, декодируется и выполняется быстрее нескольких скалярных, выполняющих тоже действий.

На самом деле, все современные процессоры содержат в себе векторные инструкции, поэтому векторную обработку данных нужно применять как на суперкомпьютерах, так и на обычных компьютерах. С самого начала развития компьютерной техники велись попытки архитектурно решить задачу ускорения вычислений, то есть создания параллельной обработки данных. Ответственность в этом случае, естественно, перекладывается на плечи и голову разработчика. Если вдруг у вас будет зависимость по данным или еще какие-то условия не будут выполнены, не позволяющие векторизовать код, то программа будет работать неправильно, так что действовать нужно аккуратно.

Результаты. Для проверки разработанного алгоритма были обработаны сейсмические сигналы разной длины N. Алгоритм был реализован в двухядерном процессоре Intel(R) Core(TM) i5-2410M CPU @ 2.30 GHz. Полученные результаты приведены в Таблице 1.

Таблица 1.

Сравнительные данные, полученные в результате обработки сейсмических сигналов в двухядерном процессоре

Количество отсчетов входного сигнала N	**Параллельно с векторизацией (сек.)**	**Параллель без векторизации (сек.)**	**Коэффициент ускорения**
1024	0,00001	0,00002	2
2048	0,000011	0,00003	2,7
4096	0,000073	0,000191	2,6
16000	0,000023	0,000078	2,8

Заключение. В соответствии с данными таблицы 1 для векторизация процессов параллельной обработки сейсмического сигнала при количестве отсчетов N=1024 требовалось $0{,}02x10^{-4}$ секунд в обычном (**Параллель без векторизации**) компиляторе, $0{,}01x10^{-4}$ секунд (после векторизации) в Java. С увеличением количества входных отсчетов N параллельная часть обработки увеличивается и растёт коэффициент ускорения. Так при N=16000 для векторизация процессов параллельной обработки сейсмического сигнала требовалось $0{,}078x10^{-4}$ секунд в обычном (**Параллель без векторизации**) компилятор, $0{,}023x10^{-4}$ секунд в Java.

Таким образом, требования высокой производительности вычислительных систем, применяемых в задачах обработки и восстановления сейсмических данных, могут быть удовлетворены как за счет разработки новых методов и параллельных алгоритмов и программ обработки, так и с помощью многоядерных средств параллельных вычислений. Практическая реализация сплайн-методов в виде векторизации процессов параллельной обработки сейсмических данных с использованием возможностей языка программирования Java и многоядерной архитектуры процессоров в процедурах прогнозирования, интерполирования, сглаживания и идентификации, восстановления и сокращения избыточности позволяет в целом повысить эффективность систем за счет увеличения скорости обработки данных при установленных показателях точности.

Литература

[1] Завьялов Ю.С. Леус В.А., Скороспелов В.А. Сплайны в инженерной геометрии. - М.: Машиностр., 1985.- 224 с.

[2] Воеводин В.В., Воеводин Вл.В. Параллельные вычисления. – СПб: «БХВ-Петербург», 2012. – 608 с.

[3] Богачев К.Ю. Основы параллельного программирования. – М.: «БИНОМ. Лаборатория знаний», 2013. – 342 с.

[4] Х.Н. Зайнидинов, Г.О. Тожибоев, О.У. Маллаев. Параллельные алгоритмы обработки сейсмических сигналов на многоядерных процессорах. Автоматика и программная инженерия. 2018. № 1 (23). С. 89-95.

[5] Герберт Р., БикА., Смит К., ТианК. Оптимизация ПО. Сборник рецептов. - СПб: Питер, 2010. -352 с.

[6] Randal E. Bryant, David R. O'Hallaron. Computer Systems: A Programmer's Perspective.-Addison-Wesley, 2010

[7] AgnerFog. The microarchitecture of Intel, AMD and VIA CPUs (an optimization guide for assembly programmers and compiler makers) // http://www.agner.org/optimize/microarchitecture.pdf

[8] Michael E. Thomadakis. The Architecture of the Nehalem Processor and Nehalem-EP SMP Platforms// http://sc.tamu.edu/systems/eos/nehalem.pdf

[9] Intel® 64 and IA-32 Architectures Optimization Reference Manual// http://www.intel.com/content/dam/doc/manual/64-ia-32-architectures-optimization-manual.pdf

[10] Intel Parallel Studio home page http://software.intel.com/en-us/forums/intel-parallel-studio

SYNCHRONIZATION AND VECTORIZATION OF PARALLEL PROCESSING PROCESSES OF SEISMIC SIGNALS USING THE POSSIBILITIES OF JAVA

H.N. ZAYNIDINOV
Doctor of Science, Tashkent University of Information Technologies named after Muhammad Al-Khorazmi, Head of the Department of Information Technologies

O.S. MALLAYEV
Senior Lecturer, Department of Informatics, Tashkent University of Information Technologies named after Muhammad Al-Khorazmi

S.P. KHALILOV
Assistant of the Department of Information Technology of the Tashkent University of Information Technologies named after Muhammad Al-Kharezmi

Abstract. The wide popularity of spline methods in the tasks of analysis and processing of seismic and geophysical signals is explained by the fact that they serve as a universal tool for approximation and, compared to other mathematical methods, with equal information and hardware costs, provide greater accuracy. On the other hand, the hardware used in such systems must also meet the requirements of high processing speed. To achieve high processing speed, it is necessary to develop parallel algorithms using the capabilities of the Java programming language and implement them on multi-core processor architectures. The article proposes a software implementation of digital signal processing spline methods using the capabilities of the Java programming language based on the multicore processor architecture. This allows, in general, to improve the efficiency of the systems by increasing the speed of data processing at the established accuracy rates.

Keywords: Parallel algorithm, flow, multi-core processor, spline, B-spline, seismic signal, vectorization