УДК 004.042

В. П. Потапов, **С. Е. Попов**, **М. А. Костылев**

Институт вычислительных технологий СО РАН
пр. Академика Лаврентьева, 6, Новосибирск, 630090, Россия

5999ft@gmail.com

ПРИМЕНЕНИЕ МАССОВО-ПАРАЛЛЕЛЬНЫХ ТЕХНОЛОГИЙ ДЛЯ ОРГАНИЗАЦИИ ПОТОКОВОЙ ОБРАБОТКИ РАДАРНЫХ ДАННЫХ

Представлен современный подход к созданию распределенного программного комплекса на базе массово-параллельной технологии Apache Spark для потоковой пре- и постобработки радарных снимков. Отличительной особенностью системы является возможность работы в режиме реального времени с большими объемами потоковых данных, а также возможность применения существующих алгоритмов не предназначенных для распределенной обработки на множестве узлов без изменения реализации алгоритма. В работе приводится сравнение технологий распределенных вычислений, представлено общее описание кластера и механизма выполнения задач пре- и постпроцессинга радарных данных, также приведены особенности имплементации конкретных задач в рамках предложенного подхода. В заключении приведены результаты тестирования разработанных алгоритмов на демонстрационном кластере.

Ключевые слова: Apache Spark, Apache Hadoop, распределенные информационные системы, радарная интерферометрия, алгоритмы обработки.

Введение

Данные дистанционного зондирования Земли получили широкое распространение практически во всех отраслях, как науки, так и промышленности, и используются для решения самых разнообразных задач, число которых постоянно растет. Не исключением стали и радарные данные, получаемые с космических аппаратов радиолокационной съемки.

Основным аппаратом анализа радарных данных является метод радарной дифференциальной интерферометрии (DInSAR), который занимается построением цифровых моделей рельефа (ЦМР) на основе расчета разности фаз нескольких радарных снимков одной территории [1, 2].

В то же время пре- и постобработка радарных данных обусловлена повышенными требованиями к вычислительным ресурсам, даже на современном аппаратном уровне. Тем не менее математические модели и алгоритмизация, заложенные в процедуры пре- и постобработки радарных данных, корректно могут быть перенесены на технологию параллельных вычислений. К основным процедурам обработки можно отнести, например, корегистрацию снимков [3], расчет когерентности и формирования интерферограммы [4], а также, развертку фазы [4–7] и последующий расчет [5], которые можно отнести к наиболее ресурсоемким этапам построения ЦМР.

Оптимизации и ускорению работы, в том числе и за счет распараллеливания расчета математических алгоритмов в процедурах обработки радарных данных посвящено большое количество научных работ [8].

Потапов В. П., *Попов С. Е.*, *Костылев М. А.* Применение массово-параллельных технологий для организации потоковой обработки радарных данных // Вестн. Новосиб. гос. ун-та. Серия: Информационные технологии. 2016. Т. 14, № 3. С. 69–80.

ISSN 1818-7900. Вестник НГУ. Серия: Информационные технологии. 2016. Том 14, № 3

В ряде работ [6, 7, 9–12] авторы работы используют технологию CUDA для улучшения производительности процедуры развертки фазы, на базе различных математических моделей. В частности, рассматривается уравнение Пуассона с весовыми коэффициентами, метод сопряженных градиентов [6, 7], метод роста регионов и отсечения ветвей [9, 10], компенсации фазового набега [10] и совмещения (корегистрации) радарных снимков [12]. В работе [13] представлен алгоритм генерации нативных (синтетических) радарных данных, симулирующих различные ошибки / шумы принимающей / передающей части космических аппаратов в зависимости от наблюдаемой земной поверхности, с целью выработки программных методов коррекции.

Наряду с GPU имплементацией, существуют программные реализации алгоритмов обработки радарных данных на базе параллельных вычислений на CPU. В работе [14] предлагается интеграция технологии параллельных вычислений MPI [1] с системой комплексной обработки радарных данных Doris (Delft object-oriented radar interferometric software) [2]. Рассмотрены основные этапы обработки InSAR/PS-InSAR данных, включая наиболее ресурсоемкие (корегистрация, формирование интерферограммы и развертка фазы). Предложена стратегия распараллеливания декомпозиции главного / подчиненного изображений, с большим количеством сегментов и частичным перекрытием границ, обеспечивающих минимальное межпроцессорное взаимодействие.

Преимуществом подходов, показанных в [6, 7, 9–14], является алгоритмизация решении задач пре- и постобработки, которые легко переносятся на параллельные вычисления, с применением уже широко известных процедур, реализованных в библиотеках для GPU и CPU. Так, в упомянутых выше работах [6, 9–14] за основу взяты элементы пакетов CuFFT, CuBLASS, CuSPARSE, PBLASS, FFTW, ScaLAPACK [15] [3].

В последнее время для стандартного алгоритмического аппарата DInSAR, получил широкое распространение метод интерферометрии малых базовых линий (SBaS), основанный на совместном использовании длинных временных серий радарных изображений одной территории, полученных в повторяющейся геометрии съемки и хронологически упорядоченных моментов времени [16]. SbaS относят к одной из наиболее ресурсоемких технологий обработки радарных данных [14]. Так, например, только на начальном этапе, производится формирование интерферограмм на базе комплексного перемножения всех возможных пар главного/подчиненного изображений. Затем, для каждой парной интерферограммы, проводится развертка фазы с целью получения полное смещение в направлении на спутник. При проведении развертки чаще всего сначала применяют алгоритмы «минимальной стоимости», а затем алгоритм «роста регионов». Учитывая тот факт, что для выявления динамики и средней скорости изменения вертикальных смещений земной поверхности с погрешностью разности высот ЦМР не более чем ±3 м/пиксел необходимо как минимум 30 изображений, на отдельных стадиях расчетов, возникает резкое уменьшение производительности (экспериментальныерасчеты показывают время от 3 до 5 часов для 12 пар снимков, небольшого разрешения в 3000 × 1000 пикселей).

Учитывая, что на сегодняшний день существует, огромное количество программно реализованных высокопроизводительных алгоритмов технологических этапов обработки радарных данных, целесообразно применять их совместно в облачной инфраструктуре. При этом сама облачная инфраструктура выступает как интегратор распределенного исполнения программного кода на данных, получаемых в потоковом режиме. Технологии распределенных вычислений в облачной среде получили широкое распространение в различных областях науки, однако на сегодняшний день авторам не удалось найти применение последних в сфере тематической обработки именно радарных снимков. Поэтому задача разработки интегрального программного комплекса для пре- и постпроцессинга радарных изображений в распределен-

[1] A High Performance Message Passing Library // Open MPI: Open Source High Performance Computing. URL: https://www.open-mpi.org/ (дата обращения 30.06.2016).

[2] Introduction to Doris // The Delft Institute of Earth Observation and Space Systems of Delft University of Technology. URL: http://doris.tudelft.nl/ (дата обращения 30.06.2016).

[3] См. также: GPU-Accelerated Libraries // NVIDIA Developer. URL: https://developer.nvidia.com/gpu-accelerated-libraries/; Introduction to ScaLAPAC // ScaLAPACK – Scalable Linear Algebra PACKage. URL: http://www.netlib.org/scalapack/ (дата обращения 30.06.2016).

ной среде с массово- параллельными механизмами (на примере Apache Spark) является, на наш взгляд, весьма актуальной задачей современной радарной интерферометрии.

Сравнение технологий распределенных вычислений

При выборе технологии распределенных вычислений были учтены особенности пре- и постпроцессинга радарных данных, так популярные технологии параллельного программирования, построенные по стандарту MPI (Message Passing Interface), например, OpenMP, не предлагают распределенного файлового хранилища, что вносит ограничение на возможные объемы обрабатываемой информации из-за необходимости применения дорогостоящих систем хранения данных и отсутствия возможности их линейного масштабирования [17].

На настоящий момент все большую популярность приобретают облачные системы распределенной обработки данных на базе технологии MapReduce. Такие системы включают в себя распределенную файловую систему, обеспечивают отказоустойчивость при выходе из строя отдельных узлов кластера, повышающих надежность хранения данных и выполнения их обработки, а также возможность использования оборудование широкого назначения с низкой стоимостью.

Для сравнения были взяты различные реализации технологии MapReduce и рассмотрены применительно к задаче обработки радарных данных (табл. 1). Наиболее важным критерием при выборе вычислительной платформы являлась возможность хранения промежуточных результатов в оперативной памяти, что позволяет получить существенно большую производительность при обработки радарных данных в сравнение со стандартных подходом, например, Hadoop MapReduce [4], что позволяет сделать Apache Spark.

Таблица 1

Сравнение технологий распределенных вычислений

	Apache Spark	Tez	Hadoop MapReduce
Интерактивный режим	Да	Да	Нет
Язык программирования	Java, Scala, Python, R	Java	Java
Работа в потоковом режиме	Да	Нет	Нет
Хранение промежуточных результатов	RAM, ROM	ROM	ROM

Важным преимуществом системы Apache Spark является возможность потоковой обработки данных, которая упрощает разработку автоматической системы пре- и постпроцессинга радарных данных из-за отсутствия необходимости создавать собственную программную реализацию для работы с потоком радарных данных [18].

На основании результатов сравнения нами была выбрана технология массово-параллельных вычислений Apache Spark. Преимуществами этой технологии являются:

- высокая масштабируемость, достигаемая за счет добавления новых узлов в вычислительный кластер, без необходимости внесения изменений в применяемые алгоритмы;
- встроенная возможность работы в режиме реального времени, позволяющая создавать алгоритмы потоковой обработки радарных данных;
- большое количество вспомогательных технологий необходимых для организации системы, которая будет поддерживать полный цикл задач пре- и постпроцессинга радарных данных.

[4] Apache Spark Streaming // Apache Spark Documentation. URL: http://spark.apache.org/streaming/ (дата обращения 02.08.2016).

Постановка задачи

Создать распределенный программный комплекс на базе массово- параллельной технологии Apache Spark для потоковой пре- и постобработки радарных снимков, со следующими функциональными особенностями:

1) возможность интеграции в программный код существующих высокопроизводительных решений для отдельных этапов (например, расчет когерентности, формирование интерферограмм, развертка фазы) обработки радарных ДДЗ;

2) возможность автоматического выбора высокопроизводительной параллельной технологии (например, CUDA, MPI и пр.) в расчетном ядре реализуемых алгоритмов.

3) использование общего распределенного пространства для хранения промежуточных результатов расчетов в виде бинарных данных стандартов BSQ, с возможностью доступа к ним отдельных алгоритмов последующих этапов обработки с различных узлов экосистемы Apache Spark.

Программная реализация

Для решения задач пре- и постобработки радарных снимков был разработан распределенный программный комплекс на базе технологий обработки больших данных Apache. На рис. 1 представлена архитектура программного комплекса.

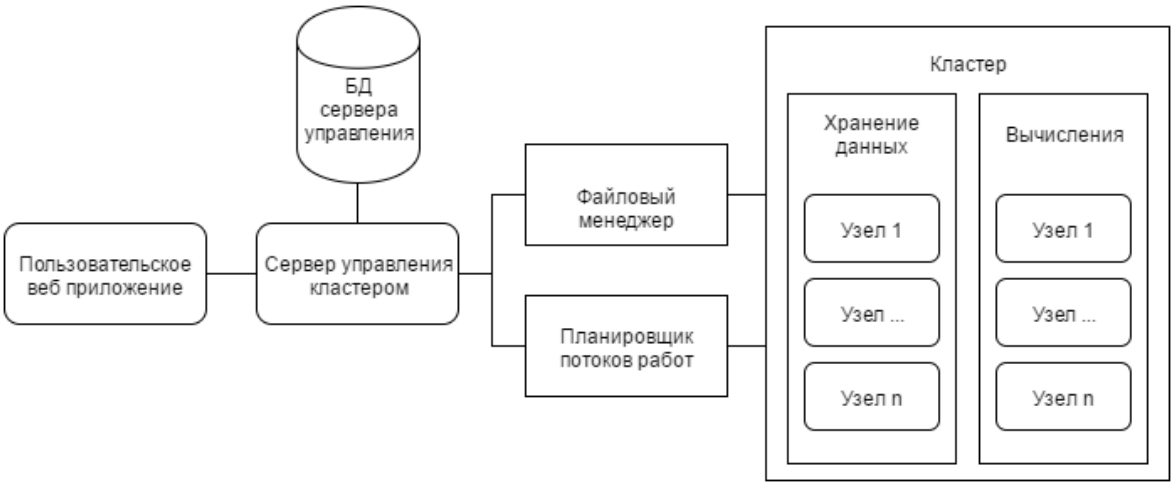

Рис. 1. Архитектура распределенного программного комплекса

Разработанное решение обладает следующими функциональными особенностями.

- Распределенная отказоустойчивая файловая система на базе технологии HDFS позволяет хранить данные и их производные на всех этапах обработки и предоставляет доступ к данным с различных вычислительных узлов кластера. Данный подход позволяет отказаться от применения дорогостоящих систем хранения и предоставляет возможность увеличения доступного пространство за счет простого добавления новых узлов кластера.
- Распределенная вычислительная платформа на базе технологии Apache Spark позволяет производить параллельную обработку потоковых радарных данных на множестве узлов с максимально возможным объемом обрабатываемых данных, ограничивающимся только количеством узлов кластера. Масштабирование программного комплекса осуществляется добавлением новых узлов, без необходимости изменения программной реализации и используемых алгоритмов.

- Система планирования потоков работ на базе технологии Apache Oozie [5] позволяет организовать процесс обработки потока радарных данных в виде последовательности действий, каждое из которых выполняется после успешного завершения предыдущего. Данные последовательности могут выполняться как автоматически при поступлении новых данных, так и по запросу пользователя, либо по расписанию.
- Пользовательское веб приложение на базе технологии Apache Hue [6] предоставляет возможность взаимодействия с программным комплексом. Приложение состоит из файлового менеджера, позволяющего загружать новые файлы, а также просматривать и редактировать уже существующие файлы в распределенной файловой системе. Вторым важным компонентом системы, является интегрированная среда управления потоками работ с возможность их создания и редактирования, а также запуска и мониторинга.

На рис. 2 представлена обобщенная диаграмма обработки радарных данных для разработанного программного комплекса. Исходные данные загружаются в распределенную файловую систему, затем производится обработка отдельных элементов этих данных на различных узлах кластера, полученные результаты сохраняется обратно в распределенную файловую систему.

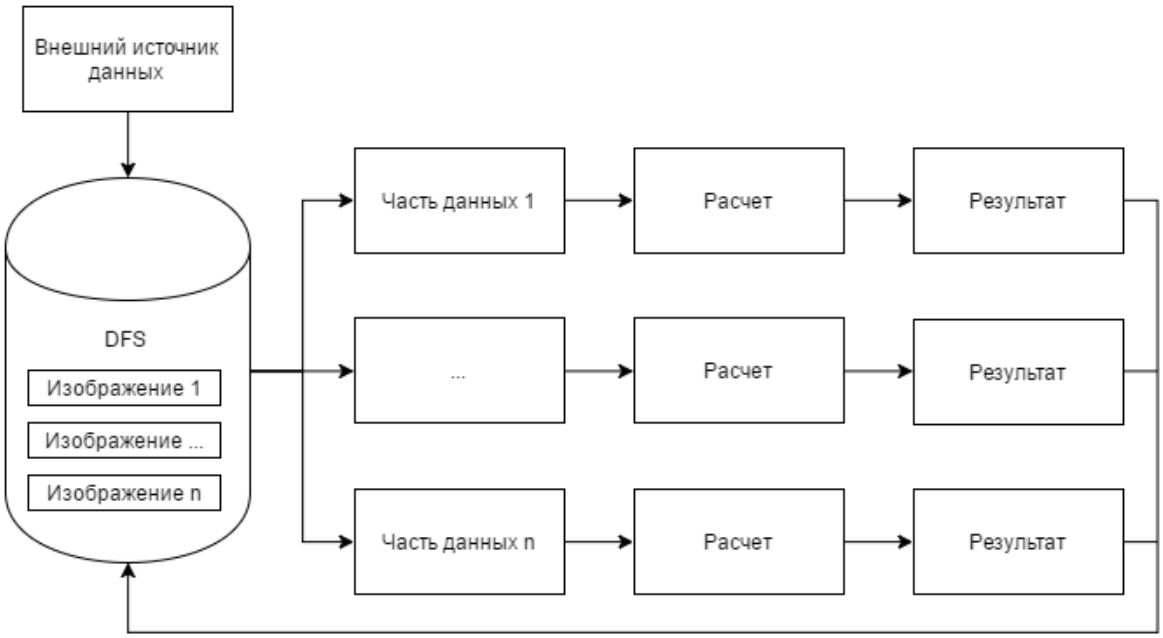

Рис. 2. Диаграмма потоков данных

В зависимости от реализации конкретного алгоритма деление входных данных производится следующими способами.

1. Отдельные файлы радарных данных, предназначенные для обработки, делятся на области, вычисление результата для каждой области производится на отдельном узле, затем полученные данные объединяются и сохраняется готовое изображение.
2. Отдельное изображение обрабатывается целиком на вычислительном узле без деления на области, параллелизация достигается за счет обработки большого количества изображений на множестве узлов.

[5] Apache Oozie // Apache Oozie Documentation. URL: https://oozie.apache.org/docs/4.2.0/ (дата обращения 02.06.2016).

[6] Apache Hue // Apache Hue Documentation. URL: http://gethue.com/spark/ (дата обращения 02.06.2016).

Рассмотрим реализацию представленной выше обобщенной диаграммы потоков данных на примере алгоритма расчета фазы. На рис. 3 представлена блок схема предложенного алгоритма.

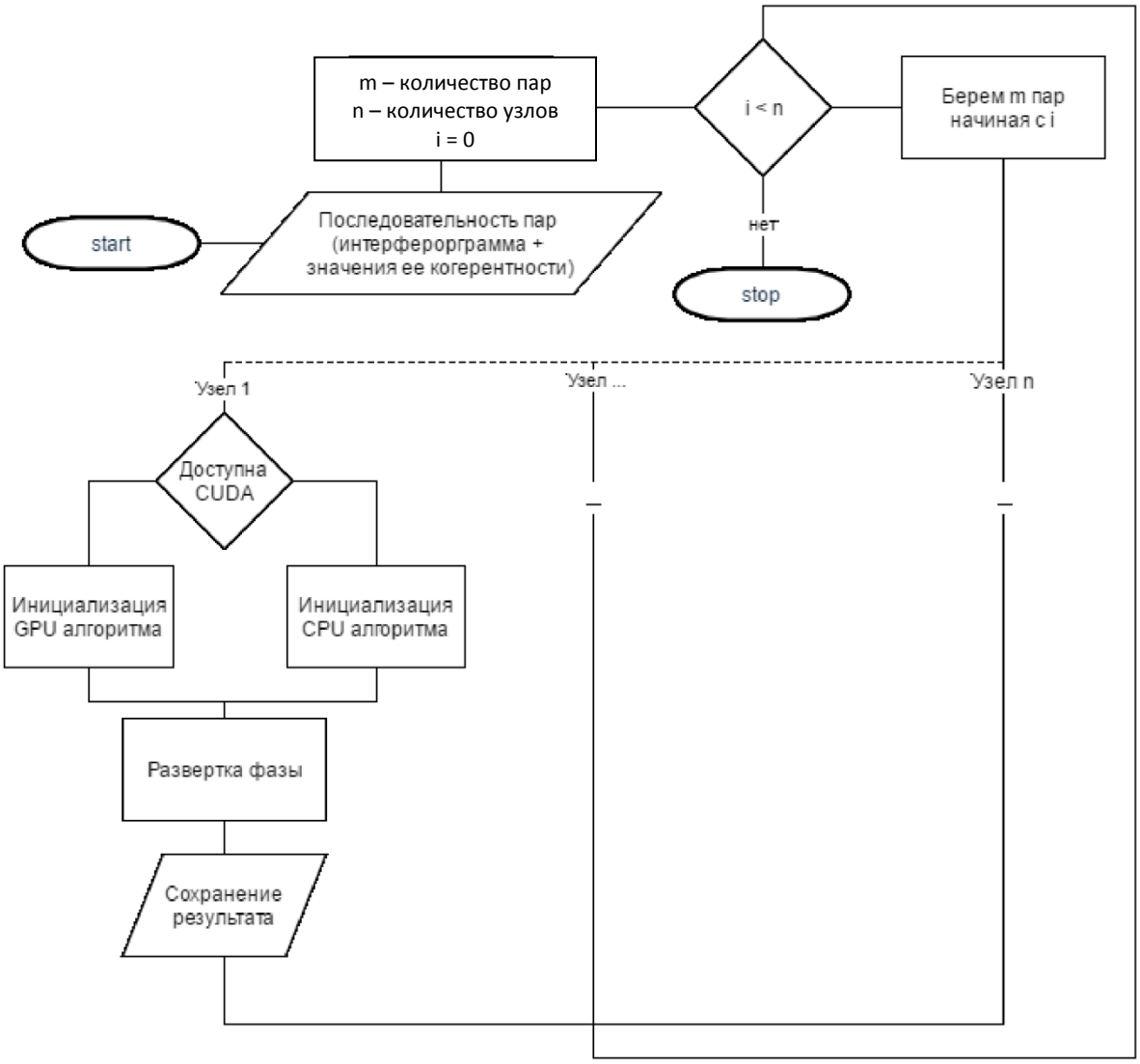

Рис. 3. Распределенный алгоритм развертки фазы

Как показано на диаграмме выше, в процессе работы алгоритма производится параллельная развертка интерферометрической фазы, а количество одновременно выполняемых расчетов определяется количеством узлов в кластере. Входными данными являются массив значений интерферограммы в каждой точке изображения и массив значений когерентности каждого значения интерферограммы. В общем виде этап развертки фазы состоит из следующих шагов.

1. Выбираются начальные точки с наибольшей когерентностью, они объявляются развернутыми, и образуют регионы. Региону с наибольшей когерентностью начальной точки присваивается номер 1, и так далее, при уменьшении значения когерентности увеличивается номер региона.

2. Для каждой развернутой точки ищутся соседние неразвернутые (целевые) точки, образующие кольца роста регионов. Назовем этот процесс итерацией роста.

3. Для каждой целевой точки кольца роста вычисляется предсказываемое значение

$$\phi^p = \frac{\sum_{k=1}^{N_u} \omega^k \phi_k^p}{\sum_{k=1}^{N_u} \omega^k},$$

где $N_u > 1$ (за исключением области начальных точек), N_u – количество соседних пикселей с развернутой фазой; $\phi_k^p = 2\phi[k] - \phi[k']$, $\omega^k = 1.0$, если на пути предсказания (стрелка на рис. 2) к точки лежат два развернутых пиксела, если один развернутый пиксел $\phi_k^p = \phi[k]$, $\omega^k = 0.5$; $\phi[k]$ и $\phi[k']$ – значения развернутых фаз на пути предсказания, соответственно.

4. Рассчитывается число неоднозначности m

$$m = nint\left(\frac{\phi^p - \varphi_\omega}{2\pi}\right),$$

где φ_ω – значение свернутой фазы, целевого пиксела, $nint$ – оператор взятия ближайшего целого.

5. Рассчитывается значение развернутой фазы $\phi_u = \varphi_\omega + 2\pi m$.

6. Значение развернутой фазы в целевой точке принимается, если выполняются тесты надежности

$d^p < T_p$, $d_u < T_u$ и значение когерентность в целевой точке больше T_c,

где $d^p = \frac{\sum_{k=1}^{N_u} \omega^k |\phi_k^p - \phi^p|}{\sum_{k=1}^{N_u} \omega^k}$, $d_u = |\phi_u - \phi^p|$, $T_p = T_u = \frac{\pi}{2}$, $T_c = 0.9 \ldots 0.2$, T_c – называется параметром релаксации.

Если два региона имеют точки пересечения, то запускается процедура их объединения.

7. Определяем количество общих точек (N_{ov}).

8. Для каждой общей точки вычисляется разность их чисел неоднозначности ($\mathrm{D_{md}} = \mathrm{m_{R1}} - \mathrm{m_{R2}}$), $\mathrm{m_{R1}}, \mathrm{m_{R2}}$ – числа неоднозначности общей точки в регионах с номерами R1 и R2, соответственно.

9. Находим моду значений разностей D_{md} и количество точек образующих ее (N_c).

10. Если $\frac{N_c}{N_{ov}} \geq T_{rr}$ и $N_{ov} \geq T_{rn}$, где $T_{rr} = 0.75$, $T_{rn} = 3$, то регионы объединяются. В точках склейки значения развернутой фазы берется для региона с наименьшим номеров, остальные точки исключаются из дальнейших действий алгоритма. Если условия не выполнены, все общие точки исключаются из алгоритма.

В табл. 2 дано описание объектов (массивов) используемых в программной реализации усовершенствованного алгоритма роста регионов.

Ниже представлена блок-схема этапа развертки интерферометрической фазы (рис. 4) и алгоритмы обработки радарных данных (рис. 3).

Таблица 2

Основные объекты (массивы) алгоритма

Название объекта, массива	Описание
U_cpu	Массив, содержащий значения развернутых фаз для соответствующих точек (размерность MxN, Float)
Ubit_pin; Ubit_gpu	«bit»-массив, содержащий 64-битные элементы, в которых закодированы состояния точек: развернута или не развернута, биты 1 или 0, соответственно (размерность MxN/64, Integer64); аналог массива Ubit_pin на стороне GPU
U_idx_cpu; U_idx_gpu	Массив, содержащий только те индексы элементов массива Ubit_pin которые изменились на текущей итерации роста алгоритма (размерность M*N, Integer32); аналог массива U_idx_cpu на стороне GPU
C_idx_cpu; C_idx_gpu	Массив, содержащий индексы точек, чьи значения когерентности больше значения параметра $\mathrm{T_c}$ (размерность $\mathrm{D_{Tc}}$ = количеству таких точек для текущей итерации релаксации, Integer32); аналог массива C_idx_cpu на стороне GPU

Окончание табл. 2

Название объекта, массива	Описание
Fbit_pin; Fbit_gpu	«bit»-массив, аналогичный по структуре Ubit_pin, только содержащий состояния точек: неудачная или удачная, биты 1 или 0, соответственно (размерность MxN/64, Integer64); аналог массива Fbit_pin на стороне GPU
F_idx_cpu; F_idx_gpu	Массив, содержащий только те индексы элементов массива Fbit_pin, которые изменились на текущей итерации роста (размерность M*N, Integer32); аналог массива F_idx_cpu на стороне GPU
GRbit_pin	«bit»-массив, аналогичный по структуре Ubit_gpu, только содержащий состояние точки, определяющее принадлежит ли точки кольцу роста или нет, биты 1 или 0, соответственно
RM_pin	Массив, содержащий номер региона, для каждой развернутой точки, в элементах с индексами, соответствующими индексам элементов массива U_cpu со значениями развернутой фазы (размерность M*N, Short)
S_cpu	Объект-контейнер, содержащий параметры типа «ключ-значение», соответствующие паре номеров («ключ»), объединяемых регионов, и индексов общих точек со значениями их чисел неоднозначности, полученных при развертке фазы в этих двух регионах («значение»)
D	Массив констант {-N,-N+1,1,N+1,N,N-1,-1,-N-1}, прибавляемых к значениям элементов массива C_idx_gpu для определения 8-ми соседних точек

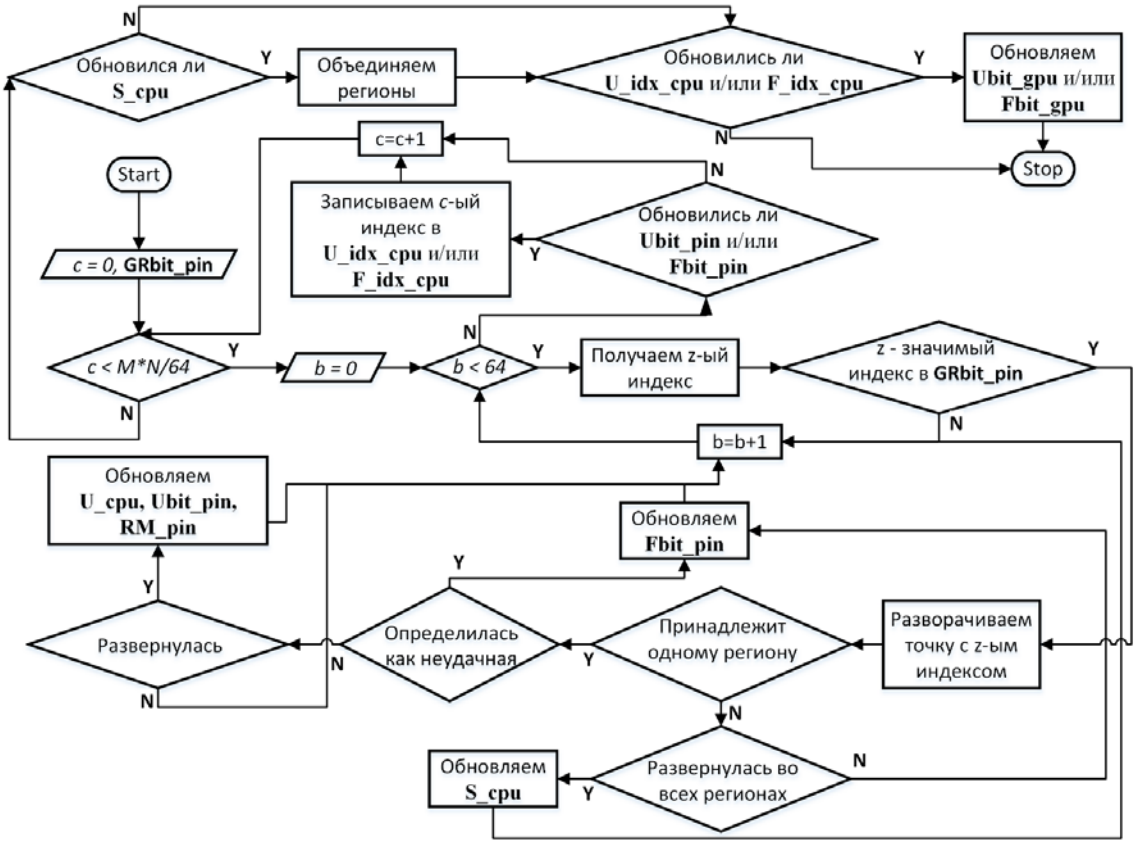

Рис. 4. Распределенный алгоритм развертки фазы

Таблица 3

Алгоритмы обработки радарных данных
с указанием особенностей их распараллеливания

Алгоритм	Входные данные	Особенности реализации
Расчет фазы	Бинарные представления радарных данных аппарата Cosmo-SkyMed уровня L1Ав формате (.hdr + SingleLookComplexbinary) на примере Exelis	Изображение делится на отдельные значения (точки) последовательно, каждое изображение рассчитывается на множестве узлов по частям
Формирование интерферограммы		Изображение делится на регионы заданного размера (7 х 7 точек), каждое изображение рассчитывается на множестве узлов по частям
Расчет когерентности		Изображение делится на регионы заданного размера (7 х 7 точек), каждое изображение рассчитывается на множестве узлов по частям
Развертка фазы		Изображение обрабатывается целиком на отдельном вычислительном узел, производится параллельная обработка множества изображений

Расчетная часть представленных алгоритмов реализована в соответствии с аналогичными схемами в основных системах обработки радарных данных. Так, расчет фазы и формирование интерферограммы производится согласно выражениям (3) и (4), а расчет когерентности выражению (5). Для развертки интерферометрической фазы применяется алгоритм роста регионов.

$$Phase_{i,j} = arctan\left(\frac{(float)I_{i,j}}{(float)(I_{i,j} \gg 32)}\right) \quad (3)$$

$$\widehat{I_{i,j}} = (float)(SLC_{i,j}^{M} * SLC_{i,j}^{S}) + (float)(SLC_{i,j}^{M} \gg 8) * (SLC_{i,j}^{S} \gg 8)$$

$$\widehat{I_{i,j}} \ll 32 = (float)(-(SLC_{i,j}^{M} * (SLC_{i,j}^{S} \gg 8)) + (float)((SLC_{i,j}^{M} \gg 8) * SLC_{i,j}^{S}))$$

$$I_{i,j} = \widehat{I_{i,j}} | (\widehat{I_{i,j}} \ll 32) \quad (4)$$

$$C_{i,j} = \frac{\frac{1}{49}\sum_{i=0}^{7}\sum_{j=0}^{7}|I_{i,j}|}{\left(\frac{1}{49}\sum_{i=0}^{7}\sum_{j=0}^{7}\left|SLC_{i,j}^{M}\right|^{2}\right) * \left(\frac{1}{49}\sum_{i=0}^{7}\sum_{j=0}^{7}\left|SLC_{i,j}^{S}\right|^{2}\right)} \quad (5)$$

Особенностью представленного способа обработки радарных данных является использование распределенных технологий (HDFS и Apache Spark), что позволяет применять данные алгоритмы для потоков радарных данных. Далее представлен графический интерфейс разработанного алгоритма в виде workflow-задания развертки интерферометрической фазы (рис. 5).

Результаты тестирования

Разработанные алгоритмы были протестированы на кластере из четырех и восьми узлов созданном с использованием CDH5 – дистрибутива среды Hadoop, также вычисления были произведены на отдельном компьютере без применения распределенного подхода (Ubuntu 16.04; Oracle JDK 1.8; Intel Xeon i5-3230M 2.6GHz; 8GB RAM). При создании кластера была использована версия среды CDH 5.7.0, которая включает следующие компоненты: Apache HDFS 2.6, Apache Spark 1.6.0, Apache Hue 3.9, Apache Oozie 1.7. Результаты тестирования представлены на рис. 6.

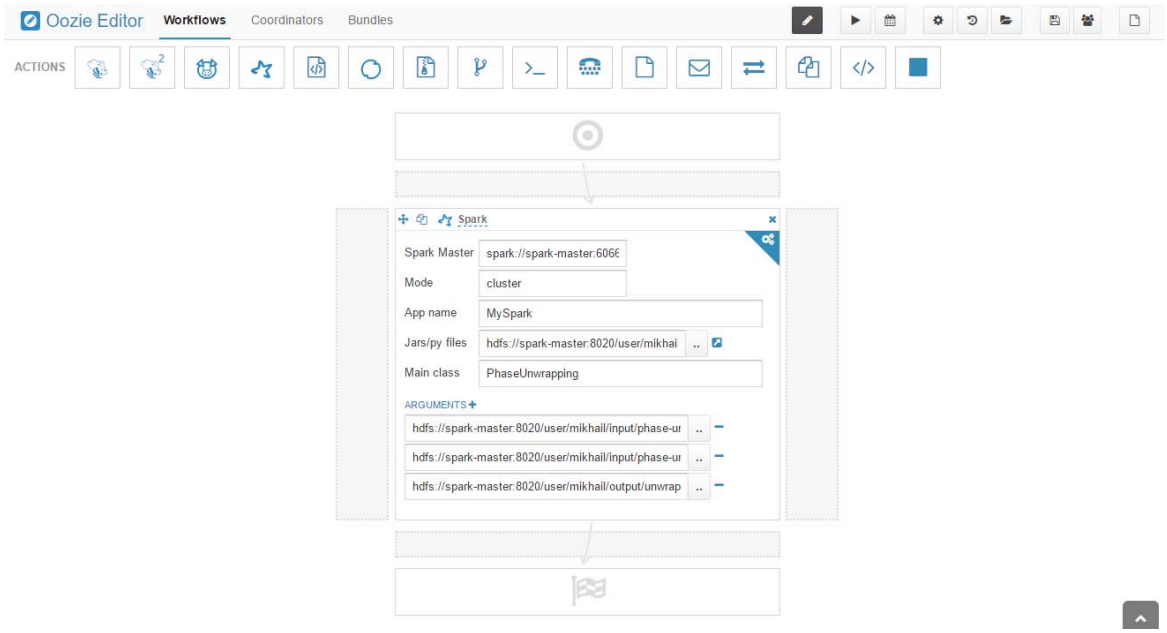

Рис. 5. Графический интерфейс workflow-задания развертки интерферометрической фазы

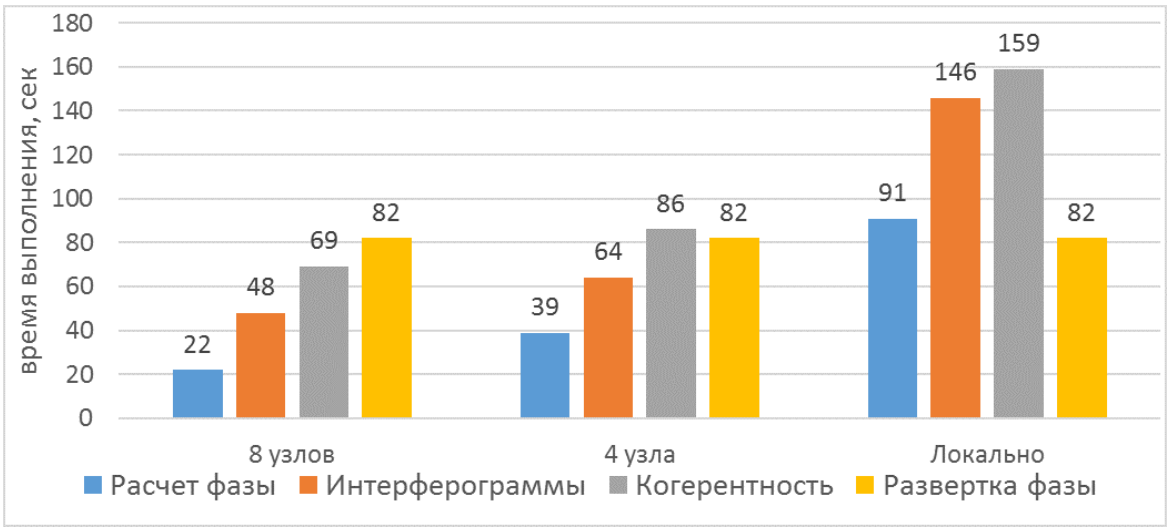

Рис. 6. Результаты тестирования

Каждый узел кластера соответствует следующей конфигурации: Ubuntu 16.04; Oracle JDK 1.8; Intel Xeon E5-2620 v2 @ 2.10GHz; 6GB RAM.

Выводы

В результате анализа различных подходов применяемых при обработке радарных данных, а также обзора технологий распределенных вычислений было предложен и реализован распределенный программный комплекс на базе массово-параллельной технологии Apache Spark для потоковой пре- и постобработки снимков.

По сравнению с традиционными подходами к обработке радарных данных, в которых параллельные вычисления либо не применяются, или применяются только для повышения

производительности расчетов, предложенное решение ориентировано на обработку большого количества потоковых данных.

Программная реализация содержит веб-интерфейс, позволяющий пользователю взаимодействовать с кластером, получая доступ к распределенной файловой системе, а также создавать и исполнять существующие потоки работ, существенно уменьшая вычислительные затраты.

Список литературы

1. *Елизаветин И. В., Шувалов Р. И., Буш В. А.* Принципы и методы радиолокационной съемки для целей формирования цифровой модели местности // Геодезия и картография. 2009. № 1. С. 39–45.

2. *Ferretti A., Monti-Guarnieri A., Prati C., Rocca F., Massonnet D.* InSAR Principles: Guidelines for SAR Interferometry Processing and Interpretation. ESA Publications. 2007. TM-19.

3. *Zhengxiao Li, James Bethel.* Image coregistration in sar interferometry // The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences. Vol. 37. Part B1. Beijing, 2008. P. 433–438.

4. *Massonnet D., Feigl K. L.* Radar interferometry and its application to changes in the earth's surface // Reviews of Geophysics. 1998. Vol. 36 (4). P. 441–500.

5. *Costantini M., Farina A., Zirilli F.* A fast phase unwrapping algorithm for SAR interferometry // IEEE Trans. GARS. 1999.Vol. 37. No. 1. P. 452–460.

6. *Mistry P., Braganza S., Kaeli D., Leeser M.* Accelerating phase unwrapping and affine transformations for optical quadrature microscopy using CUDA // Proc. of 2nd Workshop on General Purpose Processing on Graphics Processing Units, GPGPU 2009. Washington, DC, USA, 2009.

7. *Karasev P. A., Campbell D. P., Richards M. A.* Obtaining a 35x Speedup in 2D Phase Unwrapping Using Commodity Graphics Processors // Radar Conference. 2007 IEEE. P. 574–578.

8. *Верба В. С., Неронский Л. Б., Осипов И. Г., Турук В. Э.* Радиолокационные системы землеобзора космического базирования. М.: Радиотехника, 2010. 675 с.

9. *Zhenhua Wu, Wenjing Ma, Guoping Long, Yucheng Li, Yucheng Li, Yucheng Li.* High Performance Two-Dimensional Phase Unwrapping on GPUs // Proc. of the 11th ACM Conference on Computing Frontiers – CF '14. 2014.

10. *Shi Xin-Liang, Xie Xiao-Chun.* GPU acceleration of range alignment based on minimum entropy criterion // Radar Conference 2013, IET International. 14-16 April 2013. P. 1–4.

11. *Guerriero A., Anelli1 V. W., Pagliara1 A., Nutricato R., Nitti D. O.* High performance GPU implementation of InSAR time-consuming algorithm kernels // Proc. of the 1st Workshop on the State of the art and Challenges Of Research Efforts at POLIBA. 2014. p. 383

12. *Fan Zhang, Bing-nan Wang, Mao-sheng Xiang.* Accelerating InSAR raw data simulation on GPU using CUDA. Geoscience and Remote Sensing Symposium (IGARSS), 2010 IEEE International. 25–30 July 2010. P. 2932–2935.

13. *Marinkovic, P. S., Hanssen, R. F., Kampes, B. M.* Utilization of Parallelization Algorithms in InSAR/PS-InSAR Processing // Proceedings of the 2004 Envisat ERS Symposium (ESA SP-572). 6–10 September 2004. P. 1–7

14. *Gao Sheng, Zeng Qi-ming, Jiao Jian, Liang Cun-ren, Tong Qing-xi.* Parallel processing of InSAR interferogram filtering with CUDA programming // Science of Surveying and Mapping. 2015. No. 1. P. 54–68.

15. *Феоктистов А. А., Захаров А. И., Гусев М. А., Денисов П. В.* Исследование возможностей метода малых базовых линий на примере модуля SBaS программного пакета SARScape и данных PCA ASAR/ENVISat и PALSAR/ALOS. Часть 1. Ключевые моменты метода // Журнал радиоэлектроники. 2015. № 9. С. 1–26.

16. *Reyes-Ortiz J. L., Oneto L., Anguita D.* Big Data Analytics in the Cloud: Spark on Hadoop vs MPI/OpenMP on Beowulf // INNS Conference on Big Data 2015 Program San Francisco. 8–10 August 2015. P. 121–130

17. *Prakasam Kannan*. Beyond Hadoop MapReduce Apache Tez and Apache Spark. San Jose State University. URL: http://www.sjsu.edu/people/robert.chun/courses/CS259Fall2013/s3/F.pdf (дата обращения 02.08.2016).

18. *Потапов В. П., Попов С. Е.* Высокопроизводительный алгоритм роста регионов для развертки интерферометрической фазы на базе технологии CUDA // Программная инженерия. 2016. № 2. С. 61–74. DOI: 10.17587/prin.7.61-74

Материал поступил в редколлегию 15.07.2016

V. P. Potapov, S**. E. Popov**, **M. A. Kostylev**

Institute of Computational Technologies SB RAS
6 Academician Lavrentiev Str., *Novosibirsk*, *630090*, *Russian Federation*

5999ft@gmail.com

APPLICATION OF MASSIVELY PARALLEL SYSTEMS TO ORGANIZE STREAMING PROCESSING OF SAR DATA

This article presents a modern approach of creating of distributed program complex based on mass-parallel technology Apache Spark for pre- and postprocessing of sar images. The unique feature of system is ability to work in real time mode with a huge amounts of streaming data and also ability to apply existed algorithms that are not used for distributed processing on multiple nodes without changing of algorithms implementation. There is a comparison of distributed processing technologies, the common description of cluster and mechanism of executing task of pre- and postprocessing sar images, also the features of exact tasks implementation in proposed approach are shown. In the conclusion there are the results of testing of developed algorithms on demonstration cluster.

Keywords: Apache Spark, Apache Hadoop, distributed information systems, sar interfometry, processing algorithms.