DISTRIBUTED FAULT-TOLERANT COMPUTING WITH SBN-PYTHON ON A REAL COMPANY CASE

D. Tereshchenko, I. Gankevich

Saint Petersburg State University, 13B Universitetskaya Emb., Saint Petersburg, 199034, Russia

E-mail: i.gankevich@spbu.ru

Distributed computing today is in demand in batch data processing tasks, but current solutions that allow to use them in Python either are too specific, or do not guarantee full fault tolerance. As a part of the final qualification work, a high-level Python interface (hereinafter SBN-Python) was developed for the new C++ framework called Subordination, in which the last problem was solved. The interface was implemented as an extension of the Python interpreter to achieve low-level compatibility and adaptation of all functionality. The purpose of this work was to test the possibility of using the new interface on a real case of LLC «Gazpromneft – Digital Solution» and to demonstrate the its core principles. To achieve this goal the current solution was analysed, a new architecture using SBN-Python was thought out and implemented, and eventually, the resulting solution was deployed at the company's facilities. As a result of the work, it turned out that using SBN-Python on a real case also scales performance with the number of nodes in the cluster, gives the ability to process various scenarios of node failure in a limited time, as well as some architectural advantages in organizing calculations. In the future, it is planned to expand the boundaries of the new interface, implementing the possibility of building distributed web services on its basis.

Keywords: distributed computing, fault tolerance, C++, python, frameworks, subordination, interpreter extension, batch processing, case of company, gazpromneft, seismic exploration

Dmitrii Tereshchenko, Ivan Gankevich

РАСПРЕДЕЛЕННЫЕ ОТКАЗОУСТОЙЧИВЫЕ ВЫЧИСЛЕНИЯ С SBN-PYTHON НА РЕАЛЬНОМ КЕЙСЕ КОМПАНИИ

Д. Терещенк, И. Ганкевич

Санкт-Петербургский государственный университет, Россия, 199034, Санкт-Петербург, Университетская наб., д. 7–9

E-mail: i.gankevich@spbu.ru

Распределённые вычисления сегодня востребованы в задачах пакетной обработки данных, но текущие решения, которые позволяют в Python их использовать, либо слишком узкоспециализированные, либо не дают полной отказоустойчивости. В рамках выпускной квалификационной работы был разработан высокоуровневый интерфейс на Python (далее SBN-Python) к новому C++ фреймворку Subordination, в котором последняя проблема была решена. Для достижения низкоуровневой совместимости и адаптации всех сценариев функционирования интерфейс был реализован как расширение интерпретатора Python. Целью данной работы было проверить возможность применения нового интерфейса на реальном кейсе компании ООО «Газпромнефть-ЦР» и продемонстрировать принципы его использования. Для достижения этой цели было проанализировано текущее решение, продумана и реализована новая архитектура с использованием SBN-Python, и в конечном счёте развёрнуто получившееся решение на мощностях компании. По итогу работы вышло, что использование SBN-Python на реальном кейсе всё также даёт рост производительности с увеличением количества узлов в кластере, возможность обработки различных сценариев сбоя узлов за ограниченное время, а также архитектурные преимущества при организации вычислений. В дальнейшем планируется расширить границы применения нового интерфейса, реализовав на его базе возможность построения распределенных веб-сервисов.

Ключевые слова: распределённые вычисления, отказоустойчивость, Си++, пайтон, фреймворки, субординация, SBN-Python, расширение интерпретатора, пакетная обработка, кейс компании, газпромнефть, сейсморазведка

Терещенко Дмитрий, Ганкевич Иван

1. Введение

Экосистема языка программирования Python включает в себя множество инструментов для обработки и анализа данных, за счёт чего он востребован в области научных вычислений. С помощью этого языка решают задачи пакетной обработки данных, для выполнения которых вычислительных ресурсов одного компьютера не хватает. В таком случае прибегают к распределенным вычислениям, т.е. вычислениям, производимым на системе, состоящей из нескольких компьютеров, для работы с которой необходимо использовать специализированный программный интерфейс.

Выход из строя одного из узлов в данном случае является нормальной ситуацией, поэтому данный интерфейс в таком случае должен предоставлять механизмы для сохранения работоспособности системы в целом.

Проблема заключается в том, что текущие программные интерфейсы, как в Python, так и в других языках программирования, либо слишком узкоспециализированы, либо не дают полной отказоустойчивости.

В рамках выпускной квалификационной работы был разработан высокоуровневый интерфейс на Python (далее SBN-Python) к новому C++ фреймворку Subordination [1], в котором последняя проблема была решена. Для достижения низкоуровневой совместимости и адаптации всех сценариев функционирования интерфейс был выполнен как расширение интепретатора Python.

Компания ООО «Газпромнефть – Цифровые решения» является одним из драйверов Цифровой трансформации «Газпром нефти» и сейчас активно занимается задачами по обработке и анализу данных. В связи с этим, проблема наличия универсального отказоустойчивого интерфейса на Python для распределенных вычислений для неё также актуальна. Внедрение такого интерфейса может помочь решать задачи бизнеса быстрее и эффективнее.

В данной работе новый интерфейс применяется на реальном кейсе компании, а именно в задаче шумоподавления в данных сейсморазведки [2], которая является обязательной, но самой длительной операцией в процессе обработки данных. Для этой задачи была продумана и реализована новая архитектура с использованием SBN-Python, а также замерена производительность и проверена работоспособность.

2. Анализ текущего решения

В рамках проекта НИОКР командой компании ООО «Газпромнефть – Цифровые решения» были разработаны программные модули и выстроен рабочий процесс, позволяющий производить необходимые трасформации над сейсмическим кубом для подавления имеющихся в нём шумов, возникших при сейсмосъёмке.

В результате разбора кодовой базы была выявлена следующая схема работы текущего решения (см. рис. 1):

1) Загрузка SGY файла [3] с данными сейсмического куба в двоичном и текстовом формате, включающих в себя координаты отраженных сейсмических волн;
2) Параллельная декомпозиция с помощью пакета multiprocessing [4] SGY файла на NPY файлы [5], каждый из которых содержит данные по различным источникам взрыва, и сохранение в файловую систему;
3) Чтение NPY файлов, разбиение на блоки заданного размера с последующей параллельной обработкой на наборе трансформаций с помощью multiprocessing и сохранение результатов в файловую систему;
4) Чтение обработанных NPY файлов и последовательное составление итогового SGY файла.

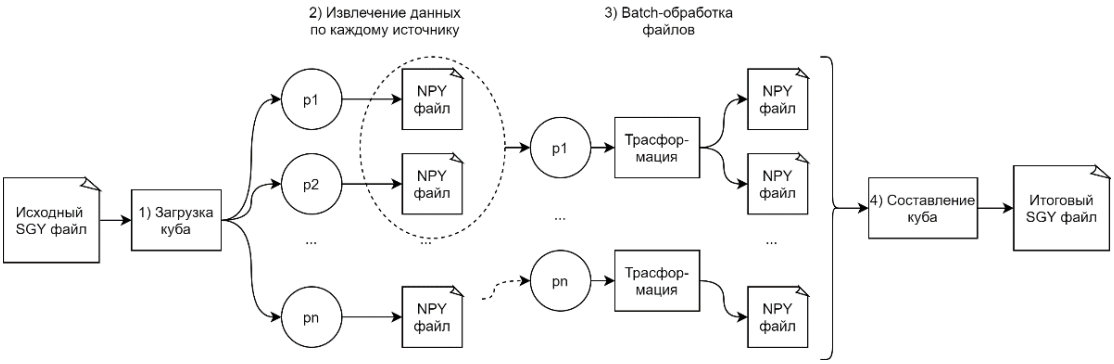

Рисунок 1. Схема работы текущего решения

Из-за того, что данные задачи большого объёма, весь процесс идет долго. Применение пакета multiprocessing хоть и даёт ускорение, но не гарантирует отказоустойчивость, из-за чего в текущем решении имеются следующие особенности, снижающие эффективность.

1. Интенсивная работа с файловой системой для сохранения промежуточных результатов.
2. Синхронизация процессов на каждом шаге (модель Bulk Synchronous Parallel [6]).

3. Построение новой архитектуры решения

Новый интерфейс, как и исходная система Subordination, основан на так называемых управляющих объектах (*kernel*) – объектах в языке программирования, которые содержат данные, которые необходимо обработать, и кода для их обработки. В методе *act* некоторая задача либо последовательно вычисляется, либо разбивается на подзадачи, представленные другим набором управляющих объектов. В методе *react* подчиненные объекты, завершившие работу, обрабатываются их родителем (см. рис. 2).

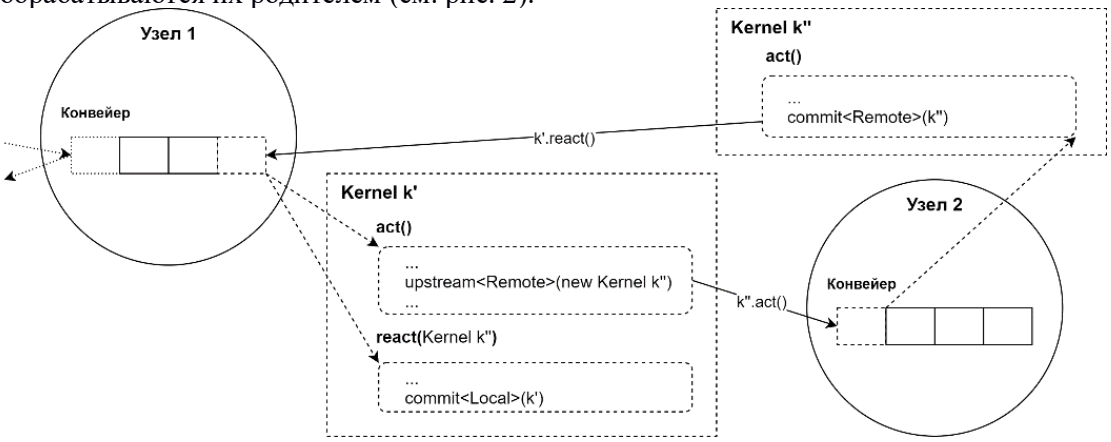

Рисунок 2. Упрощенная схема выполнения

Процедура взаимодействия объектов представляет собой аналог стека вызова функций, но для распределенных систем. Для того чтобы содержащийся в объектах код был исполнен, они отправляются на конвейер - очередь из управляющих объектов.

Исходя из выше изложенного принципа работы интерфейса была построена новая архитектура решения (см. рис. 3):

1) Программа начинает свою работу с *Main Kernel*. В методе *act* инициируется объект *SgyProcess Kernel* и ему передаётся путь к SGY файлу.
2) В методе act объекта *SgyProcess Kernel* SGY файл загружается и инициируются объекты *BatchProcess Kernel* с номерами источников взрыва;
3) В методе act каждого объекта *BatchProcess Kernel* извлекаются данные по соответствующим источникам и далее обрабатываются набором трасформеров;

4) В метод react объекта *SgyProcess Kernel* приходит результат работы объекта *BatchProcess Kernel* и сохраняется в итоговый SGY файл.
5) В методе react объекта *Main Kernel* программа завершается.

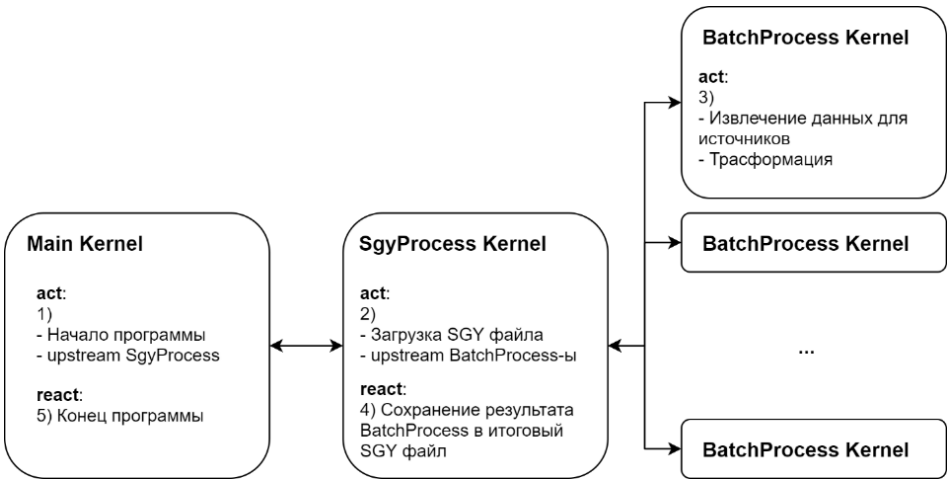

Рисунок 3. Новая архитектура решения

Тем самым новая архитектура позволяет произвести отказоустойчивую распределенную обработку сейсмического куба полностью в оперативной памяти [7], используя модель реактивного программирования [8].

4. Замер производительности

Сначала проводилось тестирование на производительность без сбоев. Количество задействованных узлов в кластере изменялось с 1 до 6. Тест повторялся три раза, затем полученные результаты усреднялись. Далее с целью проверить сценарии отказоустойчивости из статьи [1] тем же образом были произведены замеры производительности с симулированием сбоя узла, на котором находится главный управляющий объект (*superior-сбой*), его копия (*copy-superior-сбой*) или подчинённый объект (*subordinate-сбой*), а также выход из строя всех узлов (*all-сбой*). Результаты представлены на Рисунке 4.

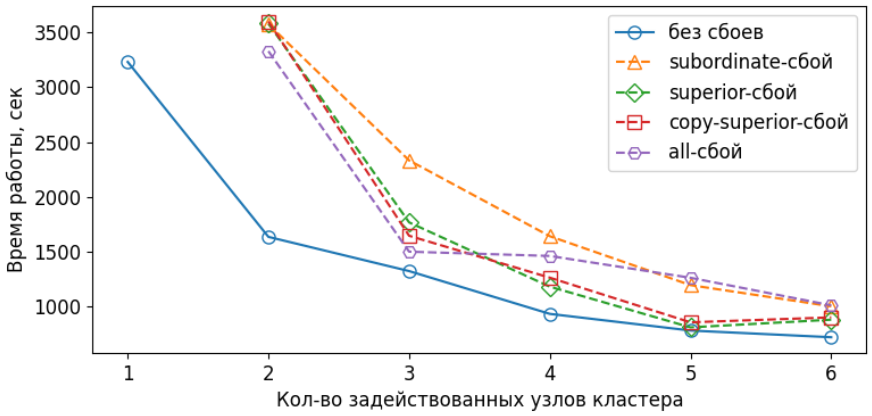

Рисунок 4. Замер производительности решения с SBN-Python

С увеличением количества задействованных узлов наблюдается прирост производительности. Случай со сбоями узлов с главным управляющим объектом или его копией показал, что это приводит к снижению производительности до производительности кластера без отказавшего узла. Сбой узлов с подчинёнными объектами даёт тот же эффект, но с большими накладными расходами из-за их количества. При выходе из строя всех узлов добавляется некоторое время восстановления.

5. Заключение

В рамках работы были получены следующие результаты.

- Во-первых, новый интерфейс для отказоустойчивых распределенных вычислений SBN-Python, был успешно применён на кейсе компании. Во-вторых, этот интерфейс смог показать свои архитектурные преимущества по сравнению с текущим решением.
- Наблюдается прирост производительности с ростом задействованных узлов.
- Сценарии обработки отказа узлов отрабатывают корректно и за ограниченное время.

В будущем планируется:

- Опробовать SBN-Python на других задачах пакетной обработки.
- Проработать возможность построения распределенных WEB-сервисов на основе нового интерфейса.

6. Благодарности

Работа выполнена при поддержке Совета по грантам Президента Российской Федерации (грант № МК-383.2020.9).

Литература

[1] Gankevich I., Tipikin Y., Gaiduchok V. Subordination: Cluster management without distributed consensus // In International Conference on High Performance Computing Simulation (HPCS). 2015. P. 639-642.

[2] Геофизические методы исследований // Сейсморазведка [Электронный ресурс] URL: https://www.geokniga.org/sites/geokniga/files/inbox/1209/chapter1.pdf (дата обращения: 10.03.2021)

[3] SEG-Y [Электронный ресурс] URL: https://ru.qaz.wiki/wiki/SEG-Y (дата обращения: 20.03.2021)

[4] NPY format [Электронный ресурс] URL: https://numpy.org/devdocs/reference/generated/numpy.lib.format.html (дата обращения: 01.04.2021)

[5] Bulk Synchronous Parallel [Электронный ресурс] URL: https://ieeexplore.ieee.org/document/552669 (дата обращения: 10.04.2021)

[6] Технология In-Memory Computing [Электронный ресурс] URL: https://www.crn.ru/news/detail.php?ID=124497 (дата обращения: 13.04.2021)

[7] Введение в реактивное программирование [Электронный ресурс] URL: https://habr.com/ru/company/arcadia/blog/432004/ (дата обращения: 17.04.2021).