электронный журнал
МОЛОДЕЖНЫЙ НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК
Издатель ФГБОУ ВПО "МГТУ им. Н.Э. Баумана". Эл No. ФС77-51038.

**# 10, октябрь 2015**

**УДК [004.051+004.052.2]:004.75**

# Анализ политики репликации в распределённой ФС HDFS

***Гужов А.А.**, студент*
*Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана,*
*кафедра «Системы обработки информации и управления»*

*Научный руководитель: Терехов В.И., к.т.н., доцент*
*Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана,*
*кафедра «Системы обработки информации и управления»*
*terekchov@bmstu.ru*

## Введение

В современном мире количество вовлечённых в информационную среду участников растёт экспоненциально (согласно статистике портала «*Internet Live Stats*» на 01.07.2014, *http://www.internetlivestats.com/internet-users/#trend*), а наступающая эпоха Интернета вещей (*Internet of Things, IoT*) только поддержит данную тенденцию. В связи с этим возникает задача обработки т.н. «больших данных» (*Big Data*), решение которой состоит из множества общих подходов и конкретных методов обработки, которые применяются в зависимости от контекста решаемой проблемы и зачастую слабо связаны друг с другом [1]. Объединяющим фактором, который связывает указанные инструменты на основе общей от него зависимости, становится поддержка со стороны операционной среды. А именно, способность последней обслужить потребности в ресурсах систем, выполняющих задачи обработки «больших данных».

Ключевым элементом при обработке крупных массивов информации становится подсистема памяти. Особенно это проявляется в части постоянной памяти, которая становится узким местом в подавляющем большинстве компьютерных систем всех масштабов. Особенная специфика функционирования постоянного хранилища данных распределённых систем определяется большими объёмами, слабой структурированностью и широким семантическим разнообразием обрабатываемой информации, что означает возможность её обработки с высокой степенью параллелизма.

**Анализ организации распределённой ФС**

Принципиально поддержка сильно распараллеленных вычислений может быть достигнута двумя путями: использование специализированных аппаратно-программных комплексов (АПК), построенных на основе высокопроизводительных компонентов; использование в составе АПК относительно дешёвых массовых компонентов, объединённых в вычислительные кластеры сходной производительности. Второй путь позволяет снизить первоначальные расходы на развёртывание центров обработки данных (ЦОД), однако ведёт к снижению надёжности построенных решений, что напрямую связано с происхождением аппаратного обеспечения. Данное обстоятельство накладывает на применяемое программное обеспечение (ПО) требование поддержания правильного функционирования системы в условиях возможных отказов.

Реализация данного требования в рамках разработки ПО, предназначенного для работы на вычислительных кластерах, вылилась в проекты распределённых файловых систем (ФС) *Google File System* (*GFS*) корпорации *Google* и *Hadoop Distributed File System* (*HDFS*) сообщества *Apache Software Foundation*. К общим чертам этих ФС следует отнести: обеспечение доступа с высокой скоростью чтения/записи, обеспечение целостности хранимой информации, автоматическое восстановление ФС при сбоях в части узлов хранилища. Также их сходство заключается в механизмах обеспечения сохранения файлов и доступа к ним. Как *GFS*, так и *HDFS* разбивают файлы на блоки больших размеров. Характерные размеры блока составляют 64 мегабайта и более. Для сравнения, типичный блок локальной ФС занимает от 512 байт до 128 килобайт. Доступ на запись к файлу в один момент времени разрешён только одному клиенту, в то время как чтение происходит параллельно, причём учитывается расположение клиента. Это делается в связи с тем, что топология вычислительной системы может быть сильно разветвлённой, а сокращение пути между клиентом и узлом позволяет минимизировать задержки, вызванные наличием промежуточной сетевой инфраструктуры (как правило, коммутаторов и маршрутизаторов) между ними.

**Анализ политики репликации**

Различия между *GFS* и *HDFS* заключаются в используемой политике репликации по умолчанию. А именно, какие факторы учитываются контроллером кластера при выборе узла, на который будут записаны реплики блока файла. В частности, *GFS* учитывает следующие параметры узлов хранилища, помимо рассмотренных выше [2]:

- свободное дисковое пространство;

- количество операций ввода/вывода на момент записи;
- местоположение (зал, стойка).

Кроме того, при добавлении новых узлов в кластер производится автоматическая перебалансировка, целью которой является приведение уровня заполнения дискового пространства всех узлов к среднему значению. Для эффективной работы хранилища данных следует поддерживать данный уровень в коридоре +/-10% для каждого узла [6]. Схожий механизм перебалансировки имеется и в *HDFS*, однако он запускается в качестве внешнего модуля и не позволяет предотвращать разбалансировку кластера [3].

Такие серьёзные различия в механизме распределения реплик по узлам определяется происхождением проектов: *GFS* – закрытый для широкого применения проект, развиваемый *Google* собственными силами и для собственных нужд, тогда как *HDFS* – открытый проект, развиваемый сообществом независимых разработчиков и свободный для использования и модификации. Последнее позволяет рассматривать *HDFS* в качестве платформы для развития и исследований, что и будет выполнено далее.

**Описание задачи**

Анализ политики репликации как основной составляющей *HDFS* показывает, что должны быть пройдены следующие этапы, которые позволят ей достигнуть уровня эффективности *GFS* и превзойти эти показатели в дальнейшем:

- развитие механизма распределения реплик по узлам с тем, чтобы не допускалась разбалансировка кластера;
- автоматическая перебалансировка кластера при изменении конфигурации последнего;
- перераспределение реплик по узлам в соответствии с частотой запросов от клиентов и топологией кластера.

Важность первого этапа определяется тем, что решением данной задачи достигается изначальная сбалансированность хранилища и, как следствие, эффективность обслуживания клиентских запросов. Важно отметить, что понятие эффективности в данном контексте определяет не только и не столько скорость обслуживания клиентов, хотя это значащий и интуитивно понятный показатель работы хранилища, но и энергетическую и, в конечном счёте, экономическую эффективность работы ЦОДа. Это является немаловажным в условиях растущей вычислительной нагрузки, характеризующейся экстенсивной природой, когда подавляющая часть затрат владельцев ЦОДов уходит на обеспечение энергоснабжения вычислительных систем [4].

Прохождение второго обозначенного этапа в развитии политики репликации *HDFS* позволит охарактеризовать политику распределения реплик в качестве интеллектуального элемента вычислительной системы, хоть и с ограниченной интеллектуальной составляющей. Тем не менее, даже такая базовая возможность системы отслеживать своё состояние, способность к рефлексии, поднимает её на уровень выше в развитии и позволяет использовать основанные на ней решения в гораздо более масштабных проектах, в которых становится невозможным управлять распределёнными вычислительными системами исключительно вручную ввиду большого объёма контролируемых параметров.

Третий этап, а точнее его достижение проектом станет логическим развитием предыдущего, связанным с усилением интеллектуальной составляющей в функционировании распределённой ФС. Как уже говорилось выше, обеспечение эффективности является одной из ключевых задач, стоящих перед разработчиками масштабных проектов, к каковым относятся центры обработки данных. Снижение накладных расходов на функционирование межузловых связей кластера, достигаемое путём перераспределения файловых блоков между узлами, может быть невелико в процентном выражении, однако на больших масштабах и продолжительных временных промежутках оно выльется в существенное снижение расходов в абсолютном исчислении.

Таким образом, задачей первой очереди совершенствования политики репликации *HDFS* является развитие рефлексивного диагностирования состояния файловой системы, учитывающее значимые для производительности и надёжности параметры.

**Метод решения**

Переформулирование задачи репликации в задачу оптимизации, где целевая функция задаётся соотношением скорости обслуживания клиентов и устойчивости распределённой ФС к сбоям составляющих её узлов, а также некоторым множеством иных параметров, учёт которых позволяет наилучшим образом приблизиться к глобальному оптимуму, делает возможным применение формальных методов поиска решения. При этом трудность заключается в сложности формализации целевой функции. Для этого представляется возможным на первом этапе провести имитационное моделирование работы распределённой вычислительной системы с тем, чтобы определить баланс между скоростью отклика и чтения/записи, отказоустойчивостью ФС, а также энергоэффективностью системы в целом в зависимости от топологии, фактора репликации и размещения реплик.

Обучение искусственной нейронной сети (ИНС) на основе полученных модельных данных позволит использовать её в качестве целевой функции при выработке политики репликации. Фактически это означает отказ от жёстко прописанного алгоритма распределения реплик в пользу гибкого механизма выработки политики, действующего по-разному, в зависимости от условий функционирования ФС.

Следует отметить, что одним из параметров решаемой задачи оптимизации является размещение реплик. Его учёт существенно повышает сложность выработки политики репликации в связи с пространством решений большой размерности. Если на первых порах это можно игнорировать, применяя моделирование системы с заданным способом размещения блоков, то в дальнейшем подобное допущение приведёт к снижению точности найденных решений вплоть до того, что выработанные политики репликации будут давать худшие результаты, нежели применяемая по умолчанию в *HDFS* на данный момент.

Данное ограничение возможно обойти путём использования эволюционных методов, позволяющих в приемлемое время находить близкое к оптимальному распределение реплик по узлам, что в связке с ИНС наделяет модуль выработки политики репликации свойствами гибридной интеллектуальной системы. Это позволяет существенно упростить наращивание его функциональности в дальнейшем и даёт дополнительный толчок развитию методов интеллектуальной оптимизации.

Таким образом, суть предложенного решения состоит в следующих шагах:

- построение имитационной модели распределённой вычислительной системы, учитывающей значимые параметры;
- формирование на основе моделирования ИНС;
- выработка эволюционного алгоритма, описывающего состояние системы и имеющего обученную на предыдущем этапе ИНС в качестве целевой функции.

**Заключение**

Проведённый анализ распределённых ФС и политики репликации на примерах *GFS* и *HDFS* показал основные принципы организации размещения блоков файлов по узлам вычислительных систем, а также выявил существенные недоработки в политике, применяемой в *HDFS*. При дальнейшем исследовании была описана и сформулирована задача её совершенствования, заключающаяся в повышении числа учитываемых внешних и внутренних параметров распределённых вычислительных систем. В качестве метода решения предложено использование механизма выработки политики репликации,

построенного по принципу гибридной интеллектуальной системы, о подходе к реализации которой будет сказано в следующей статье.

**Список литературы**


1. Paganoni A.M., Secchi P. Advances in Complex Data Modeling and Computational Methods in Statistics (Contributions to Statistics). Cham: Springer International Publishing Switzerland, 2015. 209 p.
2. Ghemawat S., Gobioff H., Leung S. The Google File System. // Symposium on Operating Systems Principles (New York, October 19-22, 2003): New York, 2003. Pages 29-43. DOI: 10.1145/945445.945450.
3. White T. Hadoop: The Definitive Guide. 3rd ed. Sebastopol, O'Reilly Media / Yahoo Press, 2012. 688 p.
4. Huusko J., de Meer H., Klingert S., Somov A. Energy Efficient Data Centers. Springer-Verlag Berlin Heidelberg, 2012. 152 p. DOI: 10.1007/978-3-642-33645-4.