ISSN 1512-1232

УДК 004.627

АЛГОРИТМ СЖАТИЯ ДАННЫХ ТЕХНОЛОГИЧЕСКОГО ПРОЦЕССА, ОСНОВАННЫЙ НА ВЕЙВЛЕТЕ ХААРА

Слабинога Марьян Остапович
Ивано-Франковский национальный технический университет нефти и газа,
Украина, г. Ивано-Франковск, ул. Карпатская, 15,
e-mail: slabinoha@i.ua

Аннотация

В статье рассмотрена проблема передачи и хранения данных о технологических параметрах промышленных объектов в условиях постоянного увеличения точности измерения. Проанализирована специфика таких данных, исходя из структуры файлов записей технологических параметров газоперекачивающих агрегатов, выбрана соответствующая концепция алгоритма сжатия на базе вейвлета Хаара. Усовершенствован подход к восстановлению данных, сжатых этим алгоритмом и проанализированы результаты его работы.

Ключевые слова: *сжатие с потерями, адаптивный алгоритм, вейвлет Хаара, технологические параметры.*

1. Введение

В современных условиях функционирования промышленных объектов диагностика и мониторинг их технического состояния является одним из самых важных направлений обеспечения их бесперебойного функционирования. Поэтому точность измерений параметров технологического процесса постоянно возрастает. Это служит причиной значительное увеличения объема информации о технологическом состоянии объекта, который, в свою очередь, замедляет передачу и обработку этой информации. Особенно это заметно на примере больших предприятий, банковских учреждений, которые в последнее время проявляют четкую тенденцию к перенесению своих данных из локальных серверов в специализированные централизованные дата-центры, а обработку данных проводят, используя концепцию облачных вычислений [1]. Поэтому проблема хранения и передачи больших объемов данных является актуальной и нуждается в решении.

2. Анализ существующих решений и постановка проблемы

Одним из решений для хранения и обработки значительных объемов данных, является хранения их в дата-центрах. Это может быть аренда отдельных вычислительных машин, или загрузка данных для хранения их на серверах “облачных” сервисов типа Dropbox, Google Drive, Box. Первое решение подходит для банков и других учреждений, которые выполняют периодическую отчетность и ведут архивный учет. Недостатком данного сервиса для хранения данных промышленных объектов является то, что оперативная работа с этими данными зависит от качества соединения с сетью Internet. Второе решение имеет еще один недостаток — сомнительность обеспечения конфиденциальности информации, которая может быть неприемлемым фактором для некоторых предприятий. Поэтому, несмотря на распространение концепции распределенного хранения информации, для информации о технологическом процессе и техническом состоянии объекта оптимальным решением является организация ее хранения ресурсами локальной вычислительной сети предприятия. Проблему роста объемов данных, вызванного повышением точности и частоты измерений,

ISSN 1512-1232

можно решить, наращивая массивы памяти. Но более экономной с точки зрения использования ресурсов будет разработка методов сжатия информации, которая базируется на специфике данных, которые хранятся.

3. Цель и задачи исследования

В качестве данных для исследования были выбраны данные о технологических параметрах функционирования газоперекачивающего агрегата. Данные представляют собой двумерный массив, строка которого представляет некоторый момент времени, а столбцы — 64 показателя функционирования технологического объекта. Данные записываются с периодом 5 минут. График одного из показателей относительно количества записей показан на рис. 1.

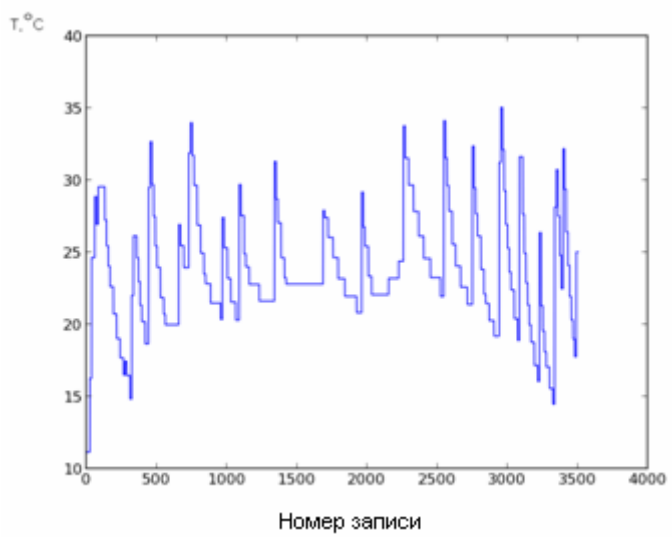

Рис. 1. Пример данных, которые исследовались

Рассмотрим характер изменения этих данных. В таблице 1 приведен фрагмент файла с записью технологических параметров объекта.

Таблица 1

Фрагмент записи технологических параметров объекта

11,16270447	10,38107204	9,28800869	13,53774071	10,53373528
11,16270447	10,38107204	9,28800869	13,53774071	10,53373528
11,16270447	10,38107204	9,28800869	13,53774071	10,53373528
11,16270447	10,38107204	9,28800869	13,53774071	10,53373528
11,16270447	10,38107204	9,28800869	13,53774071	10,53373528
11,16270447	10,38107204	9,28800869	13,53774071	10,53373528
16,26773834	10,94897652	13,78239346	8,719843864	15,51663876
16,26773834	10,94897652	13,78239346	8,719843864	15,51663876
16,26773834	10,94897652	13,78239346	8,719843864	15,51663876
16,26773834	10,94897652	13,78239346	8,719843864	15,51663876
16,26773834	10,94897652	13,78239346	8,719843864	15,51663876
16,26773834	10,94897652	13,78239346	8,719843864	15,51663876
16,26773834	10,94897652	13,78239346	8,719843864	15,51663876

ISSN 1512-1232

Изменения в этих данных происходят с довольно большим интервалом. Поэтому очень часто в таблице последовательно хранятся одинаковые числа. Исходя из этого, целесообразным будет применение к этим данным алгоритма сжатия, который дает хорошие результаты с массивами данных, соседние элементы которых не отличаются или мало отличаются. Задачей исследования является разработка алгоритма и его реализация в виде программного модуля для сжатия таких данных с достижением, по возможности, меньших потерь.

4. Выбор концепции для проектирования алгоритма

Специфика сжатия данных с малым различием между соседними элементами является основательно исследованной темой с большим количеством решений. Весомое место среди большого количества концепций сжатия занимает сжатие на основе вейвлетов [2], которое, к примеру, применяется при сжатии фотографий, текстовых документов (формат DJVU) и т.п. Одним из популярных вейвлетов, которые применяются при решении задач сжатия данных, является вейвлет Хаара [3].

При сжатии данных, которое реализуется на базе данного вейвлета, в массив вместо пары соседних значений записывается их полусумма и полуразница. Если соседние данные мало отличаются, то значение их полуразницы будет малым, и, при определенных условиях, им можно пренебречь. Фактически, при отбрасывании всех полуразниц мы получаем массив полусумм, который по объему будет в два раза меньше исходного. Такой метод сжатия будет нецелесообразным в применении к данным с большим различием между соседними значениями, однако, в случае исследуемых данных, он может быть эффективным.

5. Реализация ядра алгоритма

В качестве программирования для программной реализации концепции, которая базируется на вышеприведенных теоретических основаниях, был выбран Python версии 2.7. Данный язык программирования отличается от других своей гибкостью, простотой работы со списками, матрицами и другими типами массивов[4], кросплатформенностью и большим набором бесплатных стандартных библиотек для проведения научных вычислений, которые в большинстве случаев позволяют равноценно заменить математическое обеспечение таких коммерческих продуктов, как Matlab и Mathcad. Для отображения графиков и сохранения их в графические файлы с расширением .png использован пакет расширения matplotlib.

Ядро разработанного автором программного модуля содержит следующие функции:

1) функция load(),которая загружает исходные данные из файла;
2) функция compress(), которая создает массив полусумм исходных данных;
3) функция decompress(), которая дублирует полусуммы и возвращает массив начальной размерности для сравнения с начальными данными.

Кроме того, для проверки эффективности сжатия и точности восстановления была реализована функция compare(), которая сравнивала исходный и полученный массивы и выводила отношение значений, которые совпали, к общему количеству элементов и коэффициент линейной корреляции между данными массивами значений. Кроме сжатия в 2 раза путем многократного обращения к функции compress() были исследованы сжатия в 4 и 8 раз. Результаты сравнения сжатого и восстановленного массивов данных 5 разных технологических параметров показаны на рис. 2 (идентичность соответствующих значений) и рис. 3 (коэффициент линейной корреляции).

ISSN 1512-1232

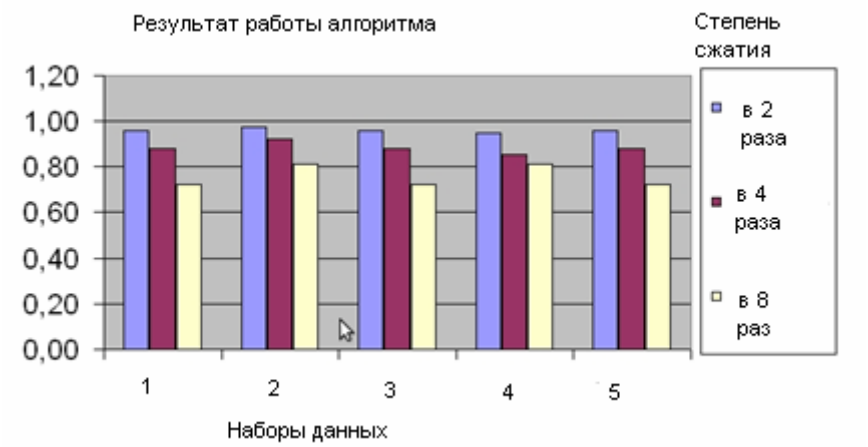

Рис. 2. Отношение значений, которые восстановились корректно к общему количеству значений массива

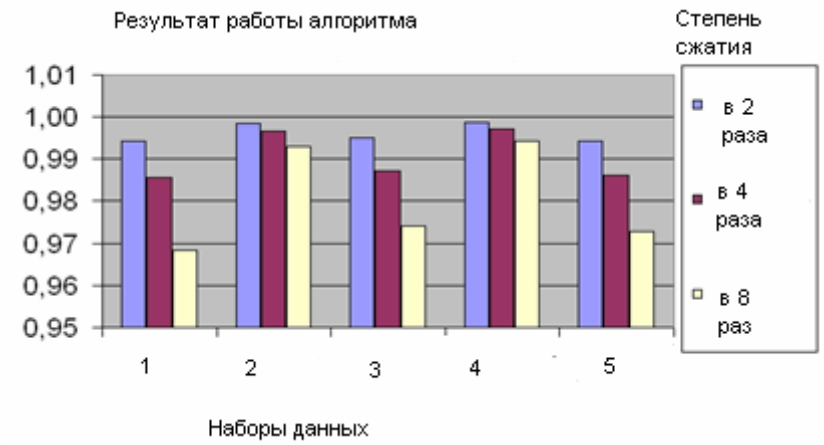

Рис. 3. Коэффициент линейной корреляции между значениями исходного и восстановленного массивов

6. Усовершенствование алгоритма и внесение в него концепции адаптивных свойств

Рассмотрим данные, восстановленные после сжатия в 2, 8 и 32 раза. Сравнительные графики таких данных с исходным массивом представлены соответственно на рис. 4, 5 и 6.

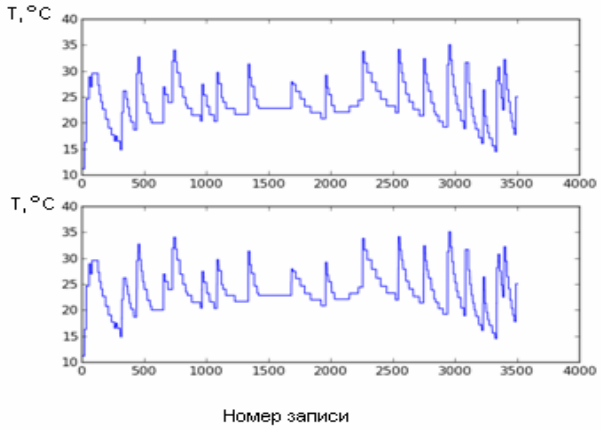

Рис. 4. Исходные данные и данные, восстановленные после сжатия в 2 раза

ISSN 1512-1232

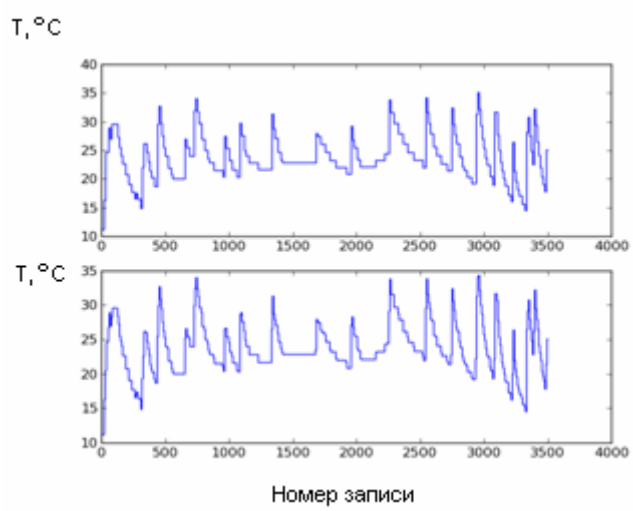

Рис. 5. Исходные данные и данные, восстановленные после сжатия в 8 раз

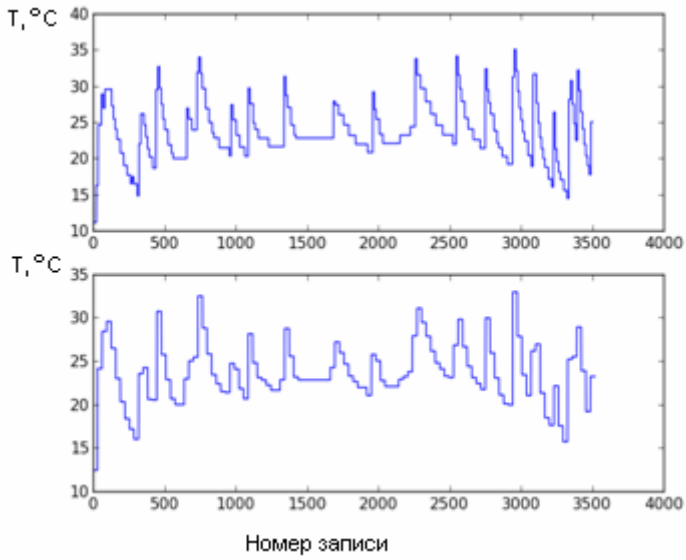

Рис. 6. Исходные данные и данные, восстановленные после сжатия в 32 раза

Как видно из графиков, а также исходя из самой концепции данного алгоритма сжатия, с ростом степени сжатия резкие перепады в исходных значениях сглаживаются. При этом общий характер изменений во времени сохраняется, но высокочастотная составляющая таких временных рядов данных теряется полностью. Избежать этого позволяет учет полуразниц при сжатии данных. Степень сжатия при этом в некоторой степени снижается, однако это позволяет учесть значительные различия между соседними значениями. При этом необязательно учитывать все полуразницы, выделив только те, которые превышают некоторое предельное значение. Для реализации такого подхода, кроме сжатого массива, было принято решение ввести 2 вспомогательные массива, один из которых содержит полуразницы, которые превысили некоторое предельное значение, а второй — позицию элементов, для которых вычислена соответствующая полуразница. В программную реализацию были включены следующие функции:

1) acompress() - функция, которая принимает исходный массив и возвращает сжатый массив и два вспомогательных массива;

2) adecompress() - функция, которая принимает сжатый массив и вспомогательные массивы и восстанавливает данные с учетом записанных полуразниц.

ISSN 1512-1232

Проведем сжатие данных исследуемого параметра с использованием усовершенствованного алгоритма с разным предельным значением для полуразниц.

Результаты эксперимента представлены в Таблице 2.

Как видно из представленной таблицы, усовершенствование алгоритма позволило повысить степень восстановления данных за счет некоторой их избыточности. В частности, это видно в экспериментах 4 и 5, где базовая степень сжатия была 0.125 и 0.0625, однако избыточность вспомогательных массивов приблизила это число к 0.29. Таким образом, при сжатии данных более чем в 3 раза мы достигли восстановления без потерь.

Дополнительная ценность такой реализации сжатия заключается в том, что проведением нескольких экспериментов мы можем найти оптимальное решение для сжатия каждого конкретного массива данных. В частности, эксперименты 1, 2, 4 и 7 позволяют судить, что оптимальным с точки зрения ресурсных затрат на сжатие (количество обращений к функции) и с точки зрения результата (коэффициент сжатия) для заданной точности восстановления (в данном случае, 100%) будет сжатие в 8 раз с предельным значением полуразницы 0.01. Регулируя предельное значение полуразницы и желаемую степень сжатия, можно достичь оптимального результата работы алгоритма, который будет зависеть от значений в каждом конкретном массиве, то есть владеть некоторой степенью адаптивности.

Таблица 2

Результаты сжатия данных усовершенствованным алгоритмом

№ п.п.	Количество обращений к функции сжатия (базовая степень сжатия)	Предельное значение полуразницы	Процент восстановленных значений	Коэффициент линейной корреляции	Фактическая степень сжатия относительно начальных данных
1	1	0.01	100%	1.0	0.54
2	2	0.01	100%	1.0	0.35
3	2	1	97.6%	0.99	0.32
4	3	0.01	100%	1.0	0.29
5	3	1	92,8%	0.99	0.23
6	3	10	85,7%	0.94	0.12
7	4	0.01	100%	1.0	0.29
8	4	1	87,5%	0.99	0.21
9	4	10	74,6%	0.94	0.067

7. Выводы

Разработанный алгоритм владеет хорошей степенью сжатия массивов данных, в которых соседние значения являются близкими или идентичными. В частности, как было показано в ходе экспериментов, описанных в статье, алгоритм эффективно применялся для сжатия массивов данных технологических параметров функционирования газоперекачивающего агрегата. Усовершенствование данного алгоритма с учетом полуразниц в восстановлении данных позволило сделать его адаптивным относительно массива, который сжимается, степеней сжатия и восстановления и улучшить показатели его работы. Таким образом, применение данного алгоритма сжатия позволяет в некоторой мере

ISSN 1512-1232

решить проблему передачи и хранения больших объемов данных технологических параметров промышленных объектов.

Список использованной литературы

1. R. Buyya, J. Broberg, A.Goscinski. Cloud Computing: Principles and Paradigms. New York, USA: Wiley Press, 2011, pp. 1–44.
2. Akansu, Ali N.; Haddad, Richard A. Multiresolution Signal Decomposition: Transforms, Subbands, Wavelets. San Diego: Academic Press, 1992.
3. Дьяконов В. П. Вейвлеты. От теории к практике. — Москва: СОЛОН-Пресс, 2004. — 440 с.
4. Wes McKinney. Python for data analysis. - O'Reilly Media, 2012. – 470 p.
5. Allen B. Downey. Think Python: How to think like computer scientist. – O'Reilly Media, 2012. – 300 p.

Article received: 2013-11-11