Московский авиационный институт
(национальный исследовательский университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

Курсовая работа по курсу «Дискретный анализ»: Методы сжатия данных

Студент: О. Р. Лисовский
Преподаватель: Н. А. Зацепин
Группа: М8О-408Б
Дата:
Оценка:
Подпись:

Москва, 2020

Условие

Необходимо реализовать два известных метода сжатия данных для сжатия одного файла.

Формат запуска должен быть аналогичен формату запуска программы gzip, должны быть поддержаны следующие ключи: -c, -d, -k, -l, -r, -t -1 -9. Должно поддерживаться указание символа дефиса в качестве стандартного ввода.

Метод решения

Как и требуется в условии запуск программы аналогичен запуску утилиты gzip: ./main <ключи> <файлы> <ключи> <файлы> ...

Обработка входных данных

Программа начинается с обработки строки стандартного ввода. Строка обрабатывается по словам. Если слово начинается с символа «-», то предполагается что это набор ключей и ключи передаются в специальную функцию, чтобы исключить противоречия работы ключей. Возможные сочетания и глвенство одних ключей над другими описано в таблице ниже.

	d	k	l	r	t	1	9
c	нет блока	c блокирует k	l блокирует c	нет блока	t блокирует c	нет блока	нет блока
d	-	нет блока	l блокирует d	нет блока	t блокирует d	d блокирует 1	d блокирует 9
k	-	-	l блокирует k	нет блока	t блокирует k	нет блока	нет блока
l	-	-	-	нет блока	l блокирует t	l блокирует 1	l блокирует 9
r	-	-	-	-	нет блока	нет блока	нет блока
t	-	-	-	-	-	t блокирует 1	t блокирует 9
1	-	-	-	-	-	-	последний полученный блокирует прошлые

В случае если полученное слово начинается с другого символа, то программа предполагает что это имя файла или директории и добавляет его в список для дальнейшей обработки.

Препроцессинг

После установления активных ключей и заполнения списка объектов компрессии/-декомпрессии, программа начинает работу этим самым списком. В случае отсутствия ключа -r все директории не рассматриваются. При активации ключа всё содержимое директории рекурсивно обрабатывается программой. При работе с файлами проверяется наличие/отсутствие (в зависимости от ключа -d) файла с расширением .gz и в случае необходимости программа спрашивает у пользователя право на перезапись соответствующего файла.

При подготовке непосредственно компрессии проверяется наличие ключа -1 или -9 для определения необходимого алгоритма. В случае их отсутствия используются оба алгоритма и выбирается лучший результат.

При подготовке непосредственно декомпрессии читается первый байт файла для установления алгоритма декодирования.

Постобработка

При окончании работы компрессии/декомпрессии программа получает сигнал об их завершении. Если этот сигнал соответствует ошибке то работа с конкретным файлом аварийно прекращается и обрабатывается следующий файл. Дальнейшие действия обусловлены введёнными ключами.

Далее незакодированный файл будет упоминаться как файл, а закодированный файл как архив.

Арифметическая компрессия

Арифметическая декомпрессия

LZ77 компрессия

LZ77 декомпрессия

По окончании чтения архива, количество байт, которое было в изначальном файле, сверяется с тем, сколько было записано в его новую версию. При несовпадении выводится соответствующее сообщение, и декомпрессия завершается неудачно.

Описание файлов программы

Код программы разбит на 11 файлов:

1. Arithmetic.h - Содержит перечисление методов и описание класса TArithmetic, необходимого для работы арифметической компрессии и декомпрессии.

2. Arithmetic.cpp - Содержит реализацию всех методов класса TArithmetic.

3. BFile.h - Содержит перечисление методов и описание класса TOutBinary и класса TInBinary, необходимых для записи в файл и чтения из файла соответственно.
4. BFile.cpp - Содержит реализацию всех методов классов TOutBinary и TInBinary.
5. Globals.h - Содержит в себе все необходимые глобальные переменные и библиотеки используемые несколькими файлами.
6. LZ77.h - Содержит перечисление методов и описание класса TLZ77, необходимого для работы алгоритма LZ77.
7. LZ77.cpp - Содержит реализацию всех методов класса TLZ77.
8. main_help.h - Содержит в себе перечисление и описание всех функций необходимых для препроцессинга перед началом работы алгоритмов компрессии и декомпрессии.
9. main_help.cpp - Содержит реализацию всех функций, необходимых для препроцессинга, описанных в файле main_help.h.
10. main.cpp - Содержит в себе алгоритм чтения файлов и ключей.
11. Makefile - Файл для сборки программы.

Основные типы данных

1. TArithmetic - класс, описывающий работу арифметического алгоритма компрессии и декомпрессии.
2. TOutBinary - класс обеспечивающий запись необходимого количества байт в файл.
3. TInBinary - класс обеспечивающий считывание необходимого количества байт из файла.
4. TLZ77 - класс, описывающий работу алгоритма LZ77.
5.

Описание методов и функций программы

Основные свойства и методы класса TArithmetic

public:

1.
2.

3.

4.

5.

6.

7.

8.

private:

1.

2.

3.

4.

5.

6.

7.

8.

Основные свойства и методы класса TOutBinary

public:

1.

2.

3.

4.

5.

6.

7.

8.

private:

1.
2.
3.
4.
5.
6.
7.
8.

Основные свойства и методы класса TInBinary

public:

1.
2.
3.
4.
5.
6.
7.
8.

private:

1.
2.
3.
4.
5.
6.
7.
8.

Основные свойства и методы класса TLZ77

public:

1.
2.
3.
4.
5.
6.
7.
8.

private:

1.
2.
3.
4.
5.
6.
7.
8.

Прочие функции

1. bool KeyManager(std::string) - Обрабатывает полученные ключи. В случае получения неизвестного ключа возвращает false, иначе true.
2. bool DifferensOfSizes(TInBinary*, std::string) - вывод для каждого файла размера сжатого, оригинального, коэффициента сжатия(%) и имя оригинального файла(ключ l). В случае повреждения. архива возвращает false, иначе true.
3. void WorkWithDirectory(std::string) - работает с директорией (ключ r).

4. void WorkWithFile(std::string) - работает с файлом (определяет наличие файла, принимает решение о компрессии или декомпрессии, выполняет прочие ключи).
5. bool IsDirectory(std::string, bool) - Проверяет, является ли файл директорией. Если файл является директорией, возвращает true, иначе false.
6. void PrintDirectoryErrors(std::string) - Уведомляет об ошибках.
7. bool IsArchive(std::string) - Проверяет, является ли файл архивом. Если файл является архивом, возвращает true, иначе false.
8. void Rename(std::string, std::string) - Изменяет название файла после успешной компрессии или декомпрессии.
9. void Delete(std::string) - Удаляет временный файл.
10. void MainDecompress(TInBinary*, std::string) - Отвечает за подготовку декомпрессинга.
11. void MainCompress(TInBinary*, std::string) - Отвечает за подготовку компрессинга.
12. unsigned long long int LZWCompress(TInBinary*, std::string, TOutBinary*) - Подготавливает LZW компрессию. Возвращает размер нового файла.
13. unsigned long long int LZ77Compress(TInBinary*, std::string, TOutBinary*) - Подготавливает LZ77 компрессию. Возвращает размер нового файла.
14. unsigned long long int ArithmeticCompress(TInBinary*, std::string) - Подготавливает арифметический компрессию. Возвращает размер нового файла.
15. void KeepSmall(unsigned long long int, unsigned long long int, unsigned long long int, std::string) - Сохраняет архив самого малого размера.
16. int main(int, char*) - Осуществляет чтение входных данных.

Исходный код

Тест производительности

Тестирование проводилось на предложенном файле enwik8, размером 95 Мб.

	Результат сжатия (б)	Коэф. сжатия	Максимальное потребление памяти (Мб)	Компрессия (с)	Декомпрессия (с)
gzip					
LZ77					
Арифметика					
Последовательная работа					

Выводы

В процессе выполнения данной работы я освоил 2 вида кодирования: арифметическое и LZ77. Было обнаружены как сходства, так и различия. К примеру

Благодаря освоению двух алгоритмов сразу у меня появились представления о рабоче прочих алгоритмов кодирования и стали очевидны различные требования к их работе и результату. Были существенно улучшены навыки работы с файлами: проверка наличия, запись, чтение, перепись.

Список литературы

1. Алгоритм LZ77 [Электронный ресурс]: mf.grsu.by URL: http://mf.grsu.by/UchProc/livak/po/comprsite/theory_lz77.html (дата обращения 10.08.2020)

2. Алгоритмы LZW, LZ77 и LZ78 [Электронный ресурс]: habr.com URL: https://habr.com/ru/post/132683/ (дата обращения 23.08.2020)

3. Арифметическое кодирование [Электронный ресурс]: mf.grsu.by URL: http://mf.grsu.by/UchProc/livak/po/comprsite/theory_arithmetic.html (дата обращения 30.08.2020)

4. Идея арифметического кодирования [Электронный ресурс]: algolist.ru URL: http://algolist.ru/compress/standard/arithm.php (дата обращения 02.09.2020)

5. Arithmetic coding - integer implementation [Электронный ресурс]: stringology.org URL: http://www.stringology.org/DataCompression/ak-int/index_en.html (дата обращения 26.09.2020)