ADAMS

Advanced **D**ata mining **A**nd **M**achine learning **S**ystem

Module: adams-compress

Peter Reutemann

December 24, 2014

Contents

Chapter 1

Introduction

Compressing and decompressing files, archiving multiple files, extracting files from archives. These are all actions that are very common and most of the time manually performed. Using ADAMS, these steps can be automated thanks to the various transformers available that handle archives.

In general, there are two different kinds of actors:

- multi-file archives (compress/decompress)
- single-file algorithms (compression/decompression)

The latter only work on a single file, like for instance gzip. ZIP, on the other hand, works with multiple files.

The following chapters cover the actors for compression and decompression in more detail.

Chapter 2

Compression

In order to compress one or more files, you can use the following transformers:

- **single files**
    - Bzip2 (see [2])
    - GZIP (see [3])
    - Lzf (see [4])
    - Lzma (see [5])
- **multiple files**
    - Tar (see [6])
    - ZIP (see [7])

Single file
These transformers take a single file as input, which gets compressed.[1] You have the choice of selecting a target file. If not, then the generated archive gets placed in the same directory as the input file. Optionally, you can also remove the original file. This is useful if you simply want to save disk-space and compress all files in a directory, but don't need the original files anymore.

Multiple files
Transformers that manage archives instead of only compressing single files, you need to supply them with an array of file names that should get added to the archive (no incremental adding possible at the moment). Using the *stripPath* regular expression, you can exert control over what of the (most likely absolute) path of the file names should end up in the archive. In order to strip the complete path from the names, simply use ".*" as expression.[2]

---

[1] adams-compress-gzip_single_file.flow
[2] adams-compress-generate_zip.flow

Chapter 3

Decompression

In order to decompress a compressed file or extract one more files from a multi-file archive, you have the following transformers available:

- **single file archives**
  - UnBzip2 (see [2])
  - UnGZIP (see [3])
  - UnLzf (see [4])
  - UnLzma (see [5])
- **multi-file archives**
  - UnTar (see [6])
  - UnZIP (see [7])

Single file archives
When using transformers for decompressing the content of a single file archive, you can choose whether you would like to extract the content to a different directory and even whether to use a different file name (by default, the file name is the one with the compression's suffix).[1]

Multi-file archives
By default, all files from a multi-file archive get extracted. If you only require a subset of the files, you can use the *regExp* option to limit the files extracted to the ones that match the regular expression (you can also invert the matching sense).[2] Optionally, you can choose whether to re-create the directory structure stored in the archive.

---

[1] adams-compress-decompress_gzipped_file.flow
[2] adams-compress-extract_from_zip.flow

Bibliography

[1] *ADAMS* – Advanced Data mining and Machine learning System
`https://adams.cms.waikato.ac.nz/`

[2] *bzip2* – Burrows-Wheeler algorithm
`http://en.wikipedia.org/wiki/Bzip2`

[3] *Gzip* – GNU zip
`http://en.wikipedia.org/wiki/Gzip`

[4] *LZF*
`http://en.wikibooks.org/wiki/Data_Compression/Dictionary_`
`compression#LZF`

[5] *LZMA* – Lempel-Ziv-Markov chain algorithm
`http://en.wikipedia.org/wiki/LempelZivMarkov_chain_algorithm`

[6] *Tar* – tape archive file format
`http://en.wikipedia.org/wiki/Tar_%28file_format%29`

[7] *ZIP* – ZIP file format
`http://en.wikipedia.org/wiki/Zip_%28file_format%29`