pymfa: A Tool for Performing Material Flow Analyses in Python 3

Carol Alexandru, 07-926-744
Project Report, Informatics and Sustainable Development, Fall Semester 2013

December 29, 2013

Contents

1 Introduction **2**
1.1 Related Work . . . 2
1.2 Proposing a Web-Based Approach . . . 3
1.3 Report Outline . . . 3

2 pymfa Analysis DSL **3**
2.1 Source File Structure . . . 3
2.2 Reflection . . . 4

3 Implementation **5**
3.1 Core . . . 5
3.2 Importer . . . 6
3.3 Exporters . . . 6
3.4 Web Server . . . 6
3.5 Command Line Script . . . 7
3.6 Testing and Code Base . . . 7
3.7 Visualization . . . 8

4 Limitations and Future Work **9**

5 Conclusion **9**

6 Appendix **9**

1 Introduction

The Material flow analysis (MFA) is an analytic method for determining the flows of material through a well-specified system. Materials may originate in nature, be processed and used in the production of goods, which are used by society and eventually discarded. The discarded products may be recycled, but it is rarely the case that all of its raw materials can be recovered. In this fashion, materials may exist as stock, for example while they are in use, or when they are being deposited in a land fill.

The MFA is used both in research and practice, either to gain a better understanding of existing flows of materials in a given system or to plan and design new processes in such a way that the processes can be implemented in real-world situations. Systems which are analyzed using MFAs can be isolated and specific, as could be the case with a waste management system, or broad, involving all of society, as could be the case with the life cycle of an electronics product[1, 3].

In all cases, the knowledge available when starting the analysis usually comprises the amounts of material being fed into the system, as well as the transfer rates between places in the system. For example, it might be known that a certainty amount of silicon, weighted in tons, goes into the production of photo-voltaic cells, of which a certain percentage is lost during production in the form of saw dust. A part of this saw dust may be recovered, while the other part ends up as waste. The silicon which makes it into the PV cells stays there for the life time of the PV array, likely several decades, after which the cells are disassembled and again, partially recycled. Some percentage may be irrecoverable, ending up as emissions during incineration of the remaining scraps.

In this example, the original amount of silicon is known, as well as the percentages that are lost in production and recycling. An MFA can reveal how much material in tons is actually wasted during production, how much is stocked in PV cells being used at the moment, and how much material is recovered and emitted by the end of the process. In this fashion, the MFA provides insights which are both of economical value, as it reveals how much material is wasted or miss-directed in the system, as well as of importance with regard to ecological questions, as it reveals how much material escapes into nature during the entire process. MFAs can become arbitrarily complex, with many materials traveling through many places.

1.1 Related Work

Material flow analyses can performed using Microsoft Excel, where it is possible to model all required components of an analysis, and even graph them to some degree. When using VBA (Visual Basic for Applications), it is also possible to implement dynamic behavior[1]. Because Microsoft Excel is a very popular software, available both for Microsoft Windows and Mac OS, analyses created in Excel exhibit relatively high portability, since Excel documents can simply be emailed or otherwise transferred for others to view and adjust. Unfortunately, free alternatives to Excel such as LibreOffice Calc do not support all the same features and depending on what Excel functionality is used for an analysis, analyses may not run in LibreOffice Calc or other free alternatives.

STAN[2], a software developed at the Vienna University of Technology, has been written for the specific purpose of performing material flow analyses. It is freeware, but the source code is unavailable to the public. STAN is a stand-alone software for Windows.

GaBi[5], a commercial software developed by PE International, can be used to perform Life Cycle Assessments for products or services. The assessments include material flow analyses. There is a free 30-day trial version available as well as a 'GaBi Education' license which is free for specific educational purposes. GaBi only runs on Windows.

Another commercial solution for Life Cycle Assessments is Umberto NXT LCA[4], which offers similar features to GaBi. There exists a 14-day trial version. It, too, is only available for windows.

All of these existing solutions have in common, that they need to be installed on the computer where the analyses should be performed, edited or viewed. While Microsoft Excel is available for Mac OS as well, the other solutions are available for Windows only.

1.2 Proposing a Web-Based Approach

While the existing software products offer the necessary functionality to perform MFAs, they exhibit certain drawbacks: Most of them are only available for Windows and they all act as stand-alone applications to be installed on a client computer. All participants in a research project, as well as people interested only in the results, need to own the software and install it on their machines. In this project, the attempt is made to use a different approach and provide the functionality necessary to perform analyses as a web application. However, instead of providing a monolithic tool that clones the features of existing stand-alone applications, a modular, more flexible approach is chosen: The web application wraps around a core which only provides functionality to run analyses. The formulation of analyses in CSV format is left to the user. The core of the web application is portable and can be used even without the web application, making it possible to perform batch analyses or to incorporate the simulation code into other applications.

As the analyses are available through a web application, researchers can simply send the link to an analysis to a colleague or other people who are interested in viewing and browsing the results of an analysis. It is also easy to allow others to create new analyses through a minimal user management system.

As a basis for choosing an appropriate language and framework for the development of the web application, the following considerations have been made: A common language should be used, and Python offers itself as a typical beginners programming language. It may be the case that a functional language would be equally, or even better suited to an analytic task such as the MFA, but knowledge of functional programming languages is limited among non-experts. The language used for developing the application should also offer libraries for visualization and data base access and there exist many such libraries for python.

1.3 Report Outline

The following sections are organized as follows: In section 2, the domain specific language used to formulate analyses is presented. Section 3 contains a detailed description of the software implementation and section 4 contains a reflection on the limitations and possible future development of the solution. A conclusion is given in section 5.

2 pymfa Analysis DSL

The stand-alone applications mentioned in section 1.1 (except for Excel) provide an interface that allow users to create material flow systems using graphical tools. It is possible to implement such an analysis designer for the web in Javascript, but such an implementation would exceed the scope of this project. For this reason, it has been decided to design a simple domain-specific language which uses the CSV format for serialization. It is intended to be used by researchers who do not posses programming knowledge. Section 2.1 specifies the DSL and in section 2.2, the advantages and disadvantages of the DSL are discussed. Table 1 in the Appendix shows an example analysis as it could be represented in a spreadsheet calculator.

2.1 Source File Structure

The first row of the source should define the row headers and time indices. Subsequent rows define links in the system, one link per row. Nodes are never defined explicitly; they are created implicitly the first time they are mentioned by a link definition. The same is true for stocks: they do not need to be specified and are created implicitly wherever not all material is forwarded from a given node. Links should be in order of their location in the system, meaning that links that occur later in the material flow should come later in the input file. More specifically, any link that transfers materials into a specific node must come before any link that transfers material out from that node.

Every link is involved with two nodes, a source node and a target node. There are currently five types of links:

- Inflow: This kind of link does not specify any source information, which is why cells 2-4 remain empty. Inflow links are used to provide the system with raw materials. Hence, the value stored for each time index represents a specific amount of material with a particular unit.
- Rate: A rate link simply takes the amount of material present at the source node for the given fraction and multiplies it with the transfer coefficient provided as data for each time index, forwarding the resulting amount of material to the target node. This means, that the sum of transfer coefficients of links leaving a given node for a single material should be 1 if all material should be forwarded from the node. If the sum is smaller than 1, a stock is created. If the sum is greater than 1, a negative stock is created, which probably represents an error in the input data.
- Fraction: Fraction links behave exactly like rate links, but they require different source and destination materials. These links are used to split a fraction from a source node into multiple fractions leaving the node. An example could be that mixed materials from recycling may be split into different fractions such as plastic and different metals.
- Conversion: Conversion links also behave similarly to rate links, but they do not produce any stock because the factors provided as data are simply applied to convert units. This makes it possible to for example convert 'pieces' to 'Kg'.
- Delay: a delay link forwards the materials according to a Weibull Distribution over time. This means that materials which flow into a node in a particular year are distributed over several years when leaving the node. The value stored in each time index represents the alpha and beta parameters of the Weibull distribution.

Rate, fraction and delay links do not need to specify target units because they should be equal to the corresponding source units. Likewise, rate and delay links do not need to specify target materials either, because they should be equal to the source materials as well. All links may specify free text in the description column.

2.2 Reflection

Using the CSV format to formulate analyses has the great advantage that it is possible to create and view analyses in Microsoft Excel or any other spreadsheet calculator application, such as LibreOffice Calc or even online in a Google Drive Spreadsheet. Analyses could also be created programmatically from other sources, such as databases. Drawbacks include the fact that CSV is hard to read in plain text and that there exist different CSV dialects. Different formats, such as XML and JSON have been considered, but none of them can easily be created without having specific programming knowledge.

The DSL attempts to require minimal information. Because only the links in a system need to be specified, the cognitive load on the person creating an analysis remains low. Nodes and stocks do not need to be specified separately, neither is it necessary to connect the nodes and links in a particular way. One particular drawback of the DSL is that there is only one row available for all information on a given link. For the delay links, this means that for example alpha and beta parameters for the Weibull distribution have to be specified in a single cell, separated by the '|' character. Should the DSL be extended with even more complex link types, the definition of such links may become increasingly cumbersome.

In general, CSV is easy to work with for both developers and users who do not posses programming knowledge. Should a graphical editor be developed for pymfa, it would both be possible to keep the DSL and create source files programmatically or to discard the DSL and work with a more structured format such as JSON or XML.

3 Implementation

In this section, the different components of pymfa are presented. The core component needed to perform analyses is written in pure Python and depends only on the SciPy library[1]. A script is provided for running ad-hoc analyses without using the server, and the core components can be used as a library as well. The web server utilizes the light-weight CherryPy web framework[2] and makes use of Jinja2[3] for templating. For reading and writing CSV and JSON files, Python standard library components are used. The client-side Javascript implementation depends only on the d3.js visualization library[4].

3.1 Core

The core of the implementation contains the necessary code to construct and run analyses. It is defined in `lib/core.py` and contains the following classes:

- `System(object)`: A container class which stores a list of `timeIndices`, a dictionary of `nodes` (using node names as keys) and a list of `links` of a simulation system. Once these properties are set, one can call `run()` on the system to start the simulation.
- `Node(object)`: The base class for all nodes. Every node has a `name`, a dictionary of `fractions`, a reference to the systems `timeIndices`, and a `type`. This class provides a method `sumStock`, which, depending on the type of the node and depending on the links connected the node, calculates the stock for each fraction of the node. This function is called only once at the end of the simulation. `Node` also provides a function `data()`, which returns the node properties in a dictionary for further use. The return value of `data()` should never be modified, as it contains references to actual node properties and does not constitute a deep copy of said properties.
- `Inflow(Node)`: A special kind of node which can be initialized with existing values. It is used to supply the system with material.
- `Link(Object)`: The base class for all links. A link has numerous properties which correspond to the cells of a single row in an analysis input file: It stores the optional free-text `description` provided by the user, references to the `src` (source) and `dst` (destination) nodes and it stores the source and destination materials and units as `srcMaterial`, `srcUnit`, `dstMaterial` and `dstUnit`. The amounts of material transferred are stored in an ordered dictionary `values`, using time indices as keys. Each node also holds a copy of the amount of material present at the source in an ordered dictionary `srcValues`. Furthermore, each link has a `type` and a reference to the system's `timeIndices`. Similarly to the base class for nodes, the link class also provides a function `data()`, which returns the link properties in a dictionary. There is also a helper method `initDst()`, which is called for each link at the beginning of the simulation, and which creates the necessary fraction containers inside the target node of each link. This is necessary because node definitions are implicit, and the user does not need to specify which fractions a node contains, as this is handled by `initDst()`. Finally, the base class for links specifies two unimplemented methods, which may be implemented by its subclasses: `propagate()` implements how the material is forwarded by this link and `calculateStock()` is a function similar to a node's `sumStock()` in purpose and use.
- `Rate(Link)`: A kind of `Link` which simply forwards a given fraction of material from the source node to the destination node. Besides the parent class arguments it accepts an additional argument `rates`, which contains an ordered dictionary of time indices to transfer rates. When propagating values for a given time index, this kind of links simply takes the amount of material

[1] The SciPy library is part of the SciPy library stack for scientific computing in Python: `http://www.scipy.org/scipylib/index.html`

[2] `http://www.cherrypy.org/`

[3] `http://jinja.pocoo.org/docs/`

[4] `http://d3js.org/`

available at the source and multiplies it by the given rate for that time index, storing the result at the target node. It implements the `propagate()` and `calculateStock()` functions accordingly and extends `data()` to include the `rates` property.

- `Conversion(Rate)`: A special kind of `Rate` which only differs in its `type` property and in that it does not create any stock, because conversion links are used to convert materials and units, and not forward actual material.
- `Fraction(Rate)`: Another subclass of `Rate` which exhibits exactly the same behavior and only differs in its `type` property.
- `Weibull(Link)`: A kind of `Link` which, similarly to `Rate`, takes an additional argument `parameters`, which should contain a dictionary of time indices to Weibull (alpha, beta) parameter tuples. It overrides the `propagate()` method so that materials are forwarded with a delay. The amount of material forwarded for each time index consists of fractions of materials from several past time indices according to a Weibull distribution. The `data()` function is extended so that it includes the Weibull parameter dictionary.

The core only depends on SciPy for handling Weibull distributions and it is feasible to use the core implementation as a library for other ventures. Through its object-oriented design, it allows for the creation of new kinds of nodes and links. For example, one could implement a `Sink` node, which discards incoming materials with just a few lines. Another plausible addition would be the creating of new delay links, which use a different algorithm to determine the delay with which materials are forwarded.

3.2 Importer

The file `lib/importer.py` defines a class `CSVImporter(object)` which is responsible for reading and parsing CSV analysis source files and constructing a `System` instance via its `load()` method. The importer uses Python's `CSV.Sniffer` implementation to attempt to determine the CSV dialect of the source file, which means that it is able to understand a variety of different quoting and delimiter characters. The importer performs many sanity checks against the values contained in the CSV source file and when errors occur, it throws a `CSVParserException` including the row and column where the specified error occurred. This allows users to debug their source files more easily.

Like it has been mentioned earlier, analyses are defined through their links. Nodes are not specified by the user and are created implicitly whenever a link defines a particular source or target node for the first time.

3.3 Exporters

The current implementation provides two exporters for serializing analyses, contained in `lib/exporter.py`. The `CSVExporter` stores the results of an analysis in CSV format, while the `JSONExporter` stores the entire state of the system as a JSON file. The JSON exporter is primarily used to transfer data from the web application to the client, but it could be used for other purposes as well.

3.4 Web Server

The file `pymfa-server.py` implements a simple web server using the light-weight CherryPy web framework for Python. It uses Jinja2 to render HTML templates contained in the `template` folder and serves static files from the `static` directory. The server configuration is contained in `cfg/cherrypy.ini`, which also contains the user configuration. The web server offers basic functionality that enables users to upload, view and explore analyses as well as download analysis results. The server exposes the following URL scheme:

- `/index`: Shows the list of existing analyses. Users who are logged in are able to delete their own existing analyses from this page. The admin user can delete any analysis.

- `/analysis/<name>`: Serves HTML, CSS and Javascript code that allows users to view the analysis with the given name. The visualizations utilized are discussed in section 3.7. If the analysis with the given name does not exist, a different page is served, providing an upload form for the user to create a new analysis. The upload page also provides instructions on how to create a valid source file.
- `/analysisJSON/<name>`: Serves the JSON representation of the simulation system and its results. This URL is used by the visualization components as well.
- `/source/<name>`: Serves the original, unmodified source analysis file uploaded by the user.
- `/results/<name>`: Serves the results of an analysis in CSV format.
- `/upload`: Upload handle which is used by the upload form to submit a new source file to the server.
- `/login`: Handle used by the login form to authenticate users and create a session for them. The handle is protected by HTTP Basic Authentication, which serves as a simple login mechanism. Note that HTTP Basic Authentication is not secure over plain HTTP. If security is a concern, HTTPS must of course be used.
- `/logout`: Handle used by the logout button to invalidate user sessions. Returns a '401' HTTP status code which causes the browser to discard the HTTP Basic Authentication credentials.

The web server stores uploaded CSV analysis source files in the `analyses` folder, and prefixes the original file name with the user name of the uploader, followed by a '§'. This way, the ownership of an analysis is stored as part of the file name, avoiding the need for storing additional state, separately from the analysis sources. Analysis results are not persisted, and if a user downloads or views an analysis, the results are calculated from the source, instead. However, a simple caching mechanism is employed: When an analysis is viewed for the first time, the `System` instance is stored by the server, so that subsequent views do not require the analysis to be re-run.

There are four Jinja2 templates used by the server. `template/main.html` is always rendered, as contains the necessary `<script>` elements as well as the navigation bar at the top. The other templates, `analysis.html`, `existing.html` and `new.html` are rendered nested inside the main template.

3.5 Command Line Script

A small script is provided for performing ad-hoc analyses without using the server or any of its components: `pymfa-cli.py` can be run from the command line and takes two arguments: The first argument should point to a CSV source analysis file and the second argument specifies the desired output file location. Which exporter is used to write the output is automatically determined from the suffix of the second argument, either '.csv' or '.json'.

The command line script only depends on the files contained in `lib` and their dependencies, which means that this small subset of files can be used to run analyses in a stand-alone environment, for example for the purpose of performing batch analyses or periodical analyses governed by a scheduler such as cron.

3.6 Testing and Code Base

A unit test suite is contained in the `test` folder, with test data contained in `test/testdata`. The tests ensure the correct behavior of the `CSVImporter`. For this purpose, 9 tests are performed against invalid source analysis files, checking whether the importer throws a parser exception containing the correct error message. Another 4 tests check for the correct parsing of 6 different CSV dialects and different file encodings. The source analysis files for these checks implement all possible features (such as multiple inflows or conversions and delays). The script `pymfa-runtests.py` can be used to run the tests.

To give a rough impression of the size of the project and its sub-components, here are the lines of code (not counting empty lines and comment lines): The core implementation for performing analyses has only 148 lines of code and the command line script adds another 35 lines. The importer has 149 lines and the exporter has 80 lines. The Javascript visualization is by far the largest code component, comprising 580 lines of code with 36 lines of helper code for the navigational components of the web site. There exist an additional 356 lines of HTML template code and 178 lines of CSS. The test suite has 72 lines and there exist 20 test files of varying length.

3.7 Visualization

The pymfa web application allows users to view analyses online. On the analysis page, a Sankey chart is drawn to give an overview on the nodes and links of the system. The material flows during each time index are summed up and the Sankey chart hence represents the total flows of materials over all time indices. The user can now click on a link or node in the Sankey chart to drill-down and reveal more detailed information in form of a bar chart. For nodes, the amounts of different materials flowing into the node are visualized, together with the stock of each material. For links, the amounts of material transferred are drawn.

Figure 1 shows an extract of the analysis page for a particular analysis. In this example, the user has clicked on the 'emissions' link. The bar chart shows two columns for each time index: the left column represents the inflow of different materials, while the right column represents stock. The colors used to differentiate materials are always the same, no matter which node or link is visualized. This makes it easier for viewers to associate a color with a given material across visualizations.

The user can hover over nodes and links in the Sankey chart and over time indices and bars in the bar chart to reveal more information. In Figure 1, the user is currently hovering over a stock column at time index 2015. The tooltip shows the exact amounts of different materials as well as the total for the given time index. Hovering over a time index label at the bottom reveals information on both inflows and stock, making it possible to view values even where the columns are too small to be targeted, as is the case for the first few years in this example.

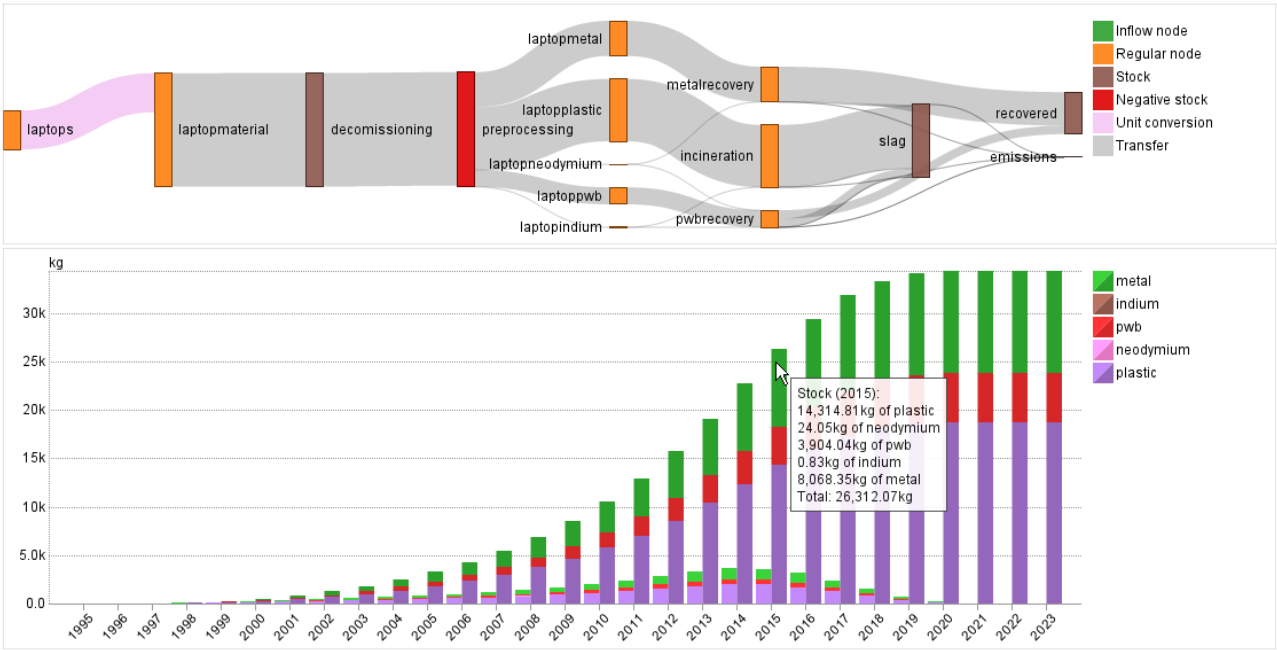

Figure 1: Viewing analysis results using the web application.

4 Limitations and Future Work

The biggest limitation of the current implementation is that it has limited support for circular material flows. The core library is able to process such flows and produces results but the Sankey chart visualization does not support circular topologies properly. Viewing a system with circular flows is difficult for this reason. It would also require a real-world scenario and appropriate data to test the implementation more thoroughly. Furthermore, the Sankey chart visualization is not drawn for every year, but only once over all time frames.

The core implementation of pymfa is intended to be both minimal and extensible. All code concerned with (de-)serialization, visualization and other tasks is separate. This should allow for future extension of the core analysis library and the corresponding DSL. For example, it is easily possible to add new types of links or nodes in order to for example model more complex delay and stock mechanisms.

One particular idea for future extension would be the implementation of an importer that grabs data from Google Drive Spreadsheets to perform analyses and visualize the data. This would offer a work-flow which does not require the user to formulate analyses in a client-side software such as Microsoft Excel as it would occur completely online. The user could simply modify the data on a spreadsheet and the web application would automatically use the most recent data from the online source.

Results of analyses in pymfa are not currently stored anywhere except for a temporary cache. There exist many easy to use data base libraries for Python, so it is very feasible to add database support to pymfa in the future.

5 Conclusion

This project has investigated the feasibility and level of difficulty for performing material flow analyses programmatically in Python. The result is a toolkit which can be used to perform one-off analyses from the command line or which can run a server providing an easy-to-use web application for individuals to perform analyses. No client-side software is required, except for some means of creating CSV files. The source file for an analysis can be created in a spreadsheet calculator, a software which is commonly installed on most machines and is even available online.

In conclusion, it can be said that material flow analyses are conceptually simple. The core implementation has barely 150 lines of code. Simply importing and exporting data from the core system needs just as much code. Furthermore, the visualization using Sankey diagrams and bar charts represents the largest effort. Compared to the effort required to construct MFAs in Microsoft Excel, the simple core implementation is probably easier to use, however it does require some programming knowledge in Python. The web application on the other hand offers non-programmers the opportunity to use the implementation.

6 Appendix

Table 1 contains a table representation of a source analysis formulated using the pymfa DSL. Table 2 contains the results of the same analysis, represented as a table. The first part contains information on how much material has flown along all the links in the system. This is followed by a rundown of the inflows into the different fractions of each node, followed by an aggregation of all fractions with the same unit for each node, followed by the stocks of each node.

Table 1: An example CSV analysis source file represented as a table.

TimeIndex	Source Node	Source Material	Source Unit	Target Node	Target Material	Target Unit	Description	1989	1990	1991	1992	1993	1994
Inflow				DirectSale	Laptops	Pieces	Laptops sold in stores	90000	95000	125000	150000	155000	140000
Inflow				OnlineSale	Laptops	Pieces	Laptops sold via online store	30000	60000	100000	120000	170000	200000
Rate	DirectSale	Laptops	Pieces	Business Use			Devices being sold to business customers	0.4	0.35	0.4	0.4	0.35	0.4
Rate	DirectSale	Laptops	Pieces	Private Use			Devices being sold to private customers	0.5	0.5	0.5	0.5	0.5	0.5
Rate	DirectSale	Laptops	Pieces	Public Use			Devices being sold to public institutions	0.1	0.15	0.1	0.1	0.15	0.1
Rate	OnlineSale	Laptops	Pieces	Business Use			Devices being sold to business customers	0.5	0.5	0.5	0.5	0.5	0.5
Rate	OnlineSale	Laptops	Pieces	Private Use			Devices being sold to private customers	0.4	0.35	0.4	0.4	0.35	0.4
Rate	OnlineSale	Laptops	Pieces	Public Use			Devices being sold to public institutions	0.1	0.15	0.1	0.1	0.15	0.1
Delay	Business Use	Laptops	Pieces	Business Decommissioning			Delayed decomissioning, because devices remain in use for several years	1|3	1|3	1|3	1|3	1|3	1|3
Delay	Private Use	Laptops	Pieces	Private Decommissioning			Delayed decomissioning, because devices remain in use for several years	1|3	1|3	1|3	1|3	1|3	1|3
Delay	Public Use	Laptops	Pieces	Public Decommissioning			Delayed decomissioning, because devices remain in use for several years	1|3	1|3	1|3	1|3	1|3	1|3
Rate	Business Decommissioning	Laptops	Pieces	Decommissioning			Combine all the devices from different user groups into a combined pool of decomissioned devices	1	1	1	1	1	1
Rate	Private Decommissioning	Laptops	Pieces	Decommissioning			Combine all the devices from different user groups into a combined pool of decomissioned devices	1	1	1	1	1	1
Rate	Public Decommissioning	Laptops	Pieces	Decommissioning			Combine all the devices from different user groups into a combined pool of decomissioned devices	1	1	1	1	1	1
Conversion	Decommissioning	Laptops	Pieces	Preprocessing	Mixed Materials	Kg	Convert number of laptops to mass	2.9	2.8	2.8	2.6	2.7	2.6
Fraction	Preprocessing	Mixed Materials	Kg	Plastic Processing	Plastic		Split the mixed materials into different fractions	0.8	0.8	0.8	0.8	0.8	0.8
Fraction	Preprocessing	Mixed Materials	Kg	Meta Processing	Metal		Split the mixed materials into different fractions	0.15	0.15	0.15	0.1	0.1	0.1
Fraction	Preprocessing	Mixed Materials	Kg	RecyclingParts	Parts		Split the mixed materials into different fractions	0.05	0.05	0.05	0.1	0.1	0.1
Rate	Plastic Processing	Plastic	Kg	Incineration			Plastic being incinerated	0.95	0.95	0.95	0.95	0.95	0.95
Rate	Plastic Processing	Plastic	Kg	Dumping			Plastic being dumped	0.05	0.05	0.05	0.05	0.05	0.05
Rate	Meta Processing	Metal	Kg	Incineration			Metal being incinerated	0.3	0.3	0.3	0.2	0.2	0.2
Rate	Meta Processing	Metal	Kg	Dumping			Metal being dumped	0.7	0.7	0.7	0.8	0.8	0.8

Table 2: An example CSV analysis results file represented as a table.

Type	Source Node	Source Material	Source Unit	Destination Node	Destination Material	Destination Unit	Description	1989.00	1990.00	1991.00	1992.00	1993.00	1994.00
rate	directsale	laptops	units	business use	laptops	units	Devices being sold to business customers	36000.00	33250.00	50000.00	60000.00	54250.00	56000.00
rate	directsale	laptops	units	private use	laptops	units	Devices being sold to private customers	45000.00	47500.00	62500.00	75000.00	77500.00	70000.00
rate	directsale	laptops	units	public use	laptops	units	Devices being sold to public institutions	9000.00	14250.00	12500.00	15000.00	23250.00	14000.00
rate	onlinesale	laptops	units	business use	laptops	units	Devices being sold to business customers	15000.00	30000.00	50000.00	60000.00	85000.00	100000.00
rate	onlinesale	laptops	units	private use	laptops	units	Devices being sold to private customers	12000.00	21000.00	40000.00	48000.00	59500.00	80000.00
rate	onlinesale	laptops	units	public use	laptops	units	Devices being sold to public institutions	3000.00	9000.00	10000.00	12000.00	25500.00	20000.00
weibull	business use	laptops	units	business decomissioning	laptops	units	Delayed decomissioning, because devices remain in use for several years	12181.03	23834.96	40962.87	58012.43	74826.72	90875.32
weibull	private use	laptops	units	private decomissioning	laptops	units	Delayed decomissioning, because devices remain in use for several years	13614.09	26115.72	43194.22	60327.79	75948.35	90245.94
weibull	public use	laptops	units	public decomissioning	laptops	units	Delayed decomissioning, because devices remain in use for several years	2866.13	7606.79	10824.49	14204.86	21821.86	23756.74
rate	business decomissioning	laptops	units	decomissioning	laptops	units	Combine all the devices from different user groups into a combined pool of decomissioned devices	12181.03	23834.96	40962.87	58012.43	74826.72	90875.32
rate	private decomissioning	laptops	units	decomissioning	laptops	units	Combine all the devices from different user groups into a combined pool of decomissioned devices	13614.09	26115.72	43194.22	60327.79	75948.35	90245.94
rate	public decomissioning	laptops	units	decomissioning	laptops	units	Combine all the devices from different user groups into a combined pool of decomissioned devices	2866.13	7606.79	10824.49	14204.86	21821.86	23756.74
conversion	decomissioning	laptops	units	preprocessing	mixed materials	kg	Convert number of laptops to mass	83117.63	161160.91	265948.42	344617.24	466011.72	532682.77
fraction	preprocessing	mixed materials	kg	plasticprocessing	plastic	kg	Split the mixed materials into different fractions	66494.11	128928.73	212758.73	275693.79	372809.38	426146.22
fraction	preprocessing	mixed materials	kg	metalprocessing	metal	kg	Split the mixed materials into different fractions	12467.64	24174.14	39892.26	34461.72	46601.17	53268.28
fraction	preprocessing	mixed materials	kg	recyclingparts	parts	kg	Split the mixed materials into different fractions	4155.88	8058.05	13297.42	34461.72	46601.17	53268.28
rate	plasticprocessing	plastic	kg	incineration	plastic	kg	Plastic being incinerated	63169.40	122482.29	202120.80	261909.10	354168.91	404838.91
rate	plasticprocessing	plastic	kg	dumping	plastic	kg	Plastic being dumped	3324.71	6446.44	10637.94	13784.69	18640.47	21307.31
rate	metalprocessing	metal	kg	incineration	metal	kg	Metal being incinerated	3740.29	7252.24	11967.68	6892.34	9320.23	10653.66
rate	metalprocessing	metal	kg	dumping	metal	kg	Metal being dumped	8727.35	16921.90	27924.58	27569.38	37280.94	42614.62
Type	Node Name	Material	Unit					1989.00	1990.00	1991.00	1992.00	1993.00	1994.00
fraction	directsale	laptops	units					90000.00	95000.00	125000.00	150000.00	155000.00	140000.00
fraction	onlinesale	laptops	units					30000.00	60000.00	100000.00	120000.00	170000.00	200000.00
fraction	business use	laptops	units					51000.00	63250.00	100000.00	120000.00	139250.00	156000.00

fraction	private use	laptops	units					57000.00	68500.00	102500.00	123000.00	137000.00	150000.00
fraction	public use	laptops	units					12000.00	23250.00	22500.00	27000.00	48750.00	34000.00
fraction	business decomissioning	laptops	units					12181.03	23834.96	40962.87	58012.43	74826.72	90875.32
fraction	private decomissioning	laptops	units					13614.09	26115.72	43194.22	60327.79	75948.35	90245.94
fraction	public decomissioning	laptops	units					2866.13	7606.79	10824.49	14204.86	21821.86	23756.74
fraction	decomissioning	laptops	units					28661.25	57557.47	94981.58	132545.09	172596.93	204877.99
fraction	preprocessing	mixed materials	kg					83117.63	161160.91	265948.42	344617.24	466011.72	532682.77
fraction	plasticprocessing	plastic	kg					66494.11	128928.73	212758.73	275693.79	372809.38	426146.22
fraction	metalprocessing	metal	kg					12467.64	24174.14	39892.26	34461.72	46601.17	53268.28
fraction	recyclingparts	parts	kg					4155.88	8058.05	13297.42	34461.72	46601.17	53268.28
fraction	incineration	plastic	kg					63169.40	122482.29	202120.80	261909.10	354168.91	404838.91
fraction	incineration	metal	kg					3740.29	7252.24	11967.68	6892.34	9320.23	10653.66
fraction	dumping	plastic	kg					3324.71	6446.44	10637.94	13784.69	18640.47	21307.31
fraction	dumping	metal	kg					8727.35	16921.90	27924.58	27569.38	37280.94	42614.62
fractions w/ same unit	directsale	laptops	units					90000.00	95000.00	125000.00	150000.00	155000.00	140000.00
fractions w/ same unit	onlinesale	laptops	units					30000.00	60000.00	100000.00	120000.00	170000.00	200000.00
fractions w/ same unit	business use	laptops	units					51000.00	63250.00	100000.00	120000.00	139250.00	156000.00
fractions w/ same unit	private use	laptops	units					57000.00	68500.00	102500.00	123000.00	137000.00	150000.00
fractions w/ same unit	public use	laptops	units					12000.00	23250.00	22500.00	27000.00	48750.00	34000.00
fractions w/ same unit	business decomissioning	laptops	units					12181.03	23834.96	40962.87	58012.43	74826.72	90875.32
fractions w/ same unit	private decomissioning	laptops	units					13614.09	26115.72	43194.22	60327.79	75948.35	90245.94
fractions w/ same unit	public decomissioning	laptops	units					2866.13	7606.79	10824.49	14204.86	21821.86	23756.74
fractions w/ same unit	decomissioning	laptops	units					28661.25	57557.47	94981.58	132545.09	172596.93	204877.99
fractions w/ same unit	preprocessing	mixed materials	kg					83117.63	161160.91	265948.42	344617.24	466011.72	532682.77
fractions w/ same unit	plasticprocessing	plastic	kg					66494.11	128928.73	212758.73	275693.79	372809.38	426146.22
fractions w/ same unit	metalprocessing	metal	kg					12467.64	24174.14	39892.26	34461.72	46601.17	53268.28
fractions w/ same unit	recyclingparts	parts	kg					4155.88	8058.05	13297.42	34461.72	46601.17	53268.28
fractions w/ same unit	incineration	plastic	kg					66909.69	129734.54	214088.47	268801.45	363489.14	415492.56
fractions w/ same unit	dumping	plastic	kg					12052.06	23368.33	38562.52	41354.07	55921.41	63921.93
stock	directsale	laptops	units					0.00	0.00	0.00	0.00	0.00	0.00
stock	onlinesale	laptops	units					0.00	0.00	0.00	0.00	0.00	0.00
stock	business use	laptops	units					38818.97	78234.01	137271.14	199258.70	263681.98	328806.67
stock	private use	laptops	units					43385.91	85770.18	145075.96	207748.17	268799.82	328553.88

stock	public use	laptops	units					9133.87	24777.09	36452.60	49247.74	76175.88	86419.14
stock	business decomissioning	laptops	units					0.00	0.00	0.00	0.00	0.00	0.00
stock	private decomissioning	laptops	units					0.00	0.00	0.00	0.00	0.00	0.00
stock	public decomissioning	laptops	units					0.00	0.00	0.00	0.00	0.00	0.00
stock	preprocessing	mixed materials	kg					0.00	0.00	0.00	0.00	0.00	0.00
stock	plasticprocessing	plastic	kg					0.00	0.00	0.00	0.00	0.00	0.00
stock	metalprocessing	metal	kg					0.00	0.00	0.00	0.00	0.00	0.00
stock	recyclingparts	parts	kg					4155.88	12213.93	25511.35	59973.07	106574.24	159842.52
stock	incineration	plastic	kg					63169.40	185651.69	387772.49	649681.59	1003850.50	1408689.41
stock	incineration	metal	kg					3740.29	10992.53	22960.21	29852.56	39172.79	49826.45
stock	dumping	plastic	kg					3324.71	9771.14	20409.08	34193.77	52834.24	74141.55
stock	dumping	metal	kg					8727.35	25649.25	53573.83	81143.21	118424.15	161038.77

References

[1] P. H. Brunner, H. Rechberger: Practical Handbook of Material Flow Analysis (Google eBook); CRC Press, ISBN: 978-1-56670-604-9 (2003)

[2] O. Cencic, H. Rechberger: Material Flow Analysis with Software STAN; Journal of Environmental Engineering and Management (2008)

[3] F. Hinterberger, S. Giljum, M. Hammer: Material Flow Accounting and Analysis (MFA): A Valuable Tool for Analyses of Society-Nature Interrelationships; Sustainable Europe Research Institute (SERI), Background Paper Nr. 2, ISSN: 1729-3545 (2003)

[4] ifu Hamburg GmbH: Umberto NXT LCA; http://www.umberto.de/en/umberto-nxt-lca/ retrieved 20131227

[5] PE INTERNATIONAL AG: GaBi; http://www.gabi-software.com/solutions/life-cycle-assessment/ retrieved 20131227