MEXPLORER 1.0.0 – A mechanism explorer for analysis and visualization of chemical reaction pathways based on graph theory

Rolf Sander

Air Chemistry Department, Max-Planck Institute of Chemistry, P.O. Box 3060, 55020 Mainz, Germany

Correspondence: R. Sander (rolf.sander@mpic.de)

Abstract. The open-source software MEXPLORER 1.0.0[1] is presented here. The program can be used to analyze, reduce, and visualize complex chemical reaction mechanisms. The mathematics behind the tool is based on graph theory: chemical species are represented as vertices, and ~~reactions as~~ each reaction is described as a set of edges. MEXPLORER is a community ~~model~~ tool published under the GNU General Public License.

1 Introduction

When research in atmospheric chemistry started about a century ago, only a couple of reactions were known. For example, Chapman (1930) identified the most important reactions explaining the stratospheric ozone layer:

$$O_3 + O \rightarrow 2\,O_2 \qquad (R1)$$

$$O_2 + O \rightarrow O_3 \qquad (R2)$$

$$O_2 \xrightarrow{h\nu} 2\,O \qquad (R3)$$

$$O_3 \xrightarrow{h\nu} O_2 + O \qquad (R4)$$

Since then, numerous additional chemicals have been discovered in the atmosphere, and our knowledge about their reactions has grown immensely. Today, many models are available to describe the atmospheric degradation of organics in the gas and the aqueous phase, e.g., the Mainz Organic Mechanism (MOM, Sander et al., 2019) and the Jülich Aqueous-phase Mechanism of Organic Chemistry (JAMOC, Rosanka et al., 2021). The most comprehensive set of reactions for tropospheric organic compounds is the Master Chemical Mechanism (MCM), which contains more than 15 000 reactions[1].

Several methods have been developed to reduce the complexity of such large mechanisms. The approach to combine chemical species manually into families has been used for a long time (Crutzen and Schmailzl, 1983). Newer approaches include the Common Representative Intermediates (CRI) by Watson et al. (2008) and the skeletal mechanism reduction (Tomlin and Turányi, 2013). Such automatized methods produce smaller mechanisms which can increase the speed of numerical simulations. However, when used as a black box, they do not contribute to our understanding of the system. Here, the MEXPLORER (mechanism explorer) software is presented, which contains several tools to identify, analyze and visualize important parts of complex reaction mechanisms. Its main features are described in the following sections. Additional information is provided in the user manual, which contains detailed instructions how to install and use the code.

Although this work focuses on atmospheric chemistry, the software can be used in other fields as well, e.g., ~~chemical engineering or marine chemistry~~ combustion chemistry (Westbrook et al., 2009) or the automated generation of complex mechanisms (e.g., Martínez-Núñez et al., 2021; Garay-Ruiz et al., 2022; Sumiya et al., 2022; Maeda et al., 2023).

2 ~~Description~~

From a mathematical point of view, a set of chemical reactions can be seen as a directed graph (digraph). A detailed description how graph theory can be used to analyse

[1] ~~The name of this version indicates that it is a release candidate used for the interactive discussion in GMDD. If necessary, bug fixes can still be made. We plan to release the final version MEXPLORER-1.0.0 together with the final paper in GMD.~~

[1] http://mcm.york.ac.uk

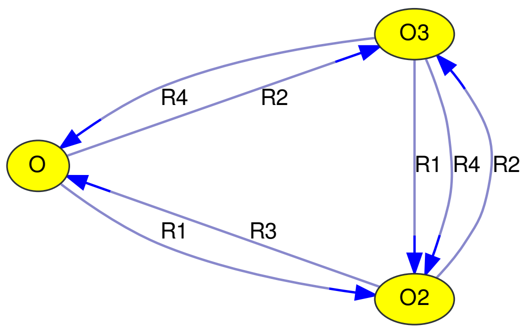


Figure 1. Graph of the Chapman mechanism, created with MEXPLORER.

kinetic reaction mechanisms is presented in the book by Turányi and Tomlin (2014).

2 Description

There are several ways to define ~~such a digraph~~ a digraph of a chemical mechanism, for example with a bipartite ~~graph~~ “species-reaction graph” where both, species and reactions, are represented by vertices (e.g., Silva et al., 2020). ~~For MEXPLORER, however, reactions are represented as edges pointing directly from the reagents to the products.~~ An alternative is the “kinetics graph” where each reaction is represented as a set of edges pointing from all reagents to all products. MEXPLORER uses the latter approach. Especially for the visualization of complex mechanisms (see below), this produces plots with much less visual clutter. As long as the reaction corresponding to each edge is stored as an edge property, both types contain the same information. To convert a kinetics graph into a species-reaction graph, it is only necessary to insert nodes representing reactions into the edges between reagents and products.

Taking the above-mentioned Chapman mechanism as an example, a graph with 3 vertices ($V_1 \ldots V_3$) and 7 edges ($E_1 \ldots E_7$) can be constructed:

V_1:	O			
V_2:	O_2			
V_3:	O_3			
E_1:	O	$\rightarrow$	O_2	(R1)
E_2:	O	$\rightarrow$	O_3	(R2)
E_3:	O_2	$\rightarrow$	O	(R3)
E_4:	O_2	$\rightarrow$	O_3	(R2)
E_5:	O_3	$\rightarrow$	O	(R4)
E_6:	O_3	$\rightarrow$	O_2	(R1)
E_7:	O_3	$\rightarrow$	O_2	(R4)

As input, MEXPLORER needs a text file defining the chemical species and the mechanism in KPP format (Sandu and Sander, 2006). For the Chapman mechanism, the species and their elemental composition are:

```
1: ~~O3  =  3O  ;~~
2: O   =   O ;
3: O2  =  2O ;
4: O3  =  3O ;
```

The reactions are defined as:

```
<R1> O3 + O  = 2 O2   : k1;
<R2> O2 + O  = O3     : k2;
<R3> O2 + hv = 2 O    : k3;
<R4> O3 + hv = O + O2 : k4;
```

Many atmospheric-chemistry models are already using KPP, thus, importing their mechanisms into MEXPLORER is straightforward. Once the mechanism has been stored in the graph-tool-specific xml format, several tools and algorithms from graph theory can be applied in order to analyze, reduce and visualize (Fig. 1) the chemical mechanism, as discussed in the following sections.

The installation requires version 3.6 of Python[2] and graph-tool[3]. Drawing the graphs is based on Graphviz[4]. Execution of the code is controlled via configuration (“config”) files, using the Python ConfigParser.

2.1 Visualization

MEXPLORER can create a plot of the whole mechanism or a selected subset and save it as a vector graphic in pdf format. According to “a picture is worth a thousand words”, such plots can provide an informative overview for presentations and publications. The user can adjust several features of the plot.

2.1.1 Vertices (species)

Chemical species are represented as vertices of the graph. To increase the amount of information contained in the plot, MEXPLORER offers several options to tune the background color and labels of the vertices:

– For plots showing the pathways from a source species to a target species, this can be emphasized with a red background color for the source and a green background color for the target. Intermediates in the path are shown with a yellow background color (see Fig. **??**).

– To illustrate the degradation of large organic molecules in a chain of reactions, the number of carbon atoms can be used to determine the background color, as in Fig. **??**.

[2]https://www.python.org/
[3]https://graph-tool.skewed.de
[4]https://graphviz.org

- In a multiphase chemical mechanism, different colors can be used for gas- and aqueous-phase species, respectively. This coloring option is used in Fig. 4, where the aqueous-phase species contain the suffix "`_a01`".
- Another area of application is the skeletal reduction of a complex mechanism ~~. Here, an~~ to a simpler mechanism. A detailed description of the methodology has been presented by Niemeyer and Sung (2011). Briefly, all reaction sequences from a source to a target species in the mechanism are analyzed. An overall interaction coefficient (OIC) is calculated for every species in the mechanism~~. A~~ , where a high OIC value indicates an important species. Species with an OIC below a certain threshold are removed from the mechanism. MEXPLORER can read such OIC values from a file and use them to define the vertex background color. In addition, the OIC values can be shown inside the vertices below the names of the species. This produces a clear representation of the reduced mechanism (see for example Fig. 4 in Sander et al. (2019)).

~~Alternative~~ All of the options listed above can be controlled via Python config files. For alternative representations of the vertex properties~~can be created with additionalPython functions, if needed~~, the code provides an entry point for additional, user-defined Python functions.

2.1.2 Edges (reactions)

~~Chemical reactions are represented as directed edges of~~ A chemical reaction is represented as a set of directed edges in the graph. To increase the amount of information contained in the plot, MEXPLORER offers several options to tune the thickness, color and labels of the edges:

- When reaction rates are available, they can be used to define the line thickness of the edge arrows. This is especially useful to highlight important reactions found in maximum flow calculations, as in Figs. **??** and 4.
- Edge labels are used to identify reactions. In Fig. 1, the reaction numbers (R1,...,R4) are shown. Although this is a unique and consise way to describe edges, it requires the look-up of the reactions that correspond to these reaction numbers. An alternative is to label edges with the other reactants of the reaction (as in Figs. **??**, **??**, 4, and 5). For photochemical reactions, the label "$h\nu$" is used, and first-order reactions are labeled with "1st". Unfortunately, the graphs quickly become unreadable with increasing complexity of the chemical mechanism. To alleviate this problem, MEXPLORER can merge parallel edges and their labels. For example, the Chapman mechanism in Fig. 1 contains two reactions that convert O_3 to O_2: R1 and R4. After merging the corresponding edges, a new label with a comma-separated list is created: "R1,R4". When the edge labels become too long, they are abbreviated to "n rxns", indicating that there are n reactions contributing to this edge. Examples of labels generated this way can be seen in Figs. **??**, **??**, 4 and 5.

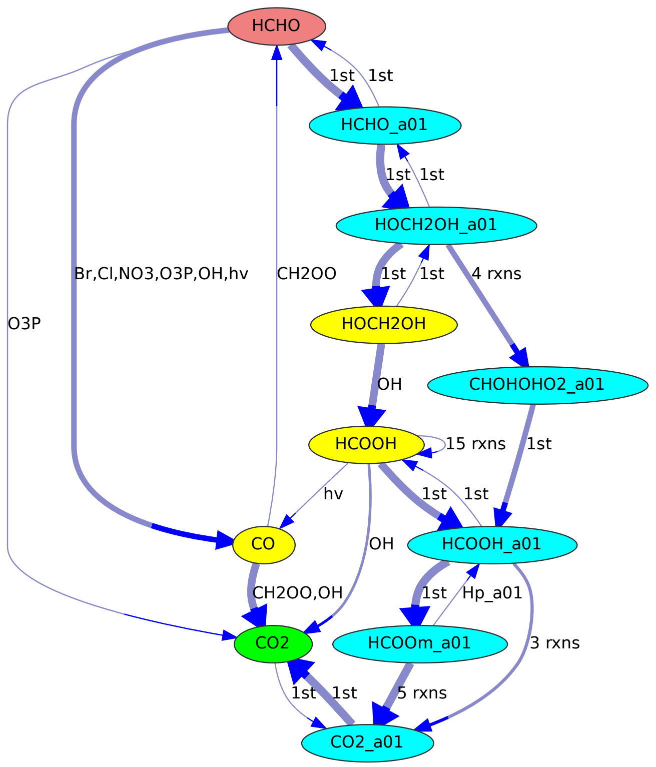


Figure 4. Reaction pathways from HCHO (red) to CO_2 (green) in JAMOC. The oxidation takes place in both the gas phase (yellow species) and also in the aqueous phase (blue species with the suffix "`_a01`").

~~Alternative representations of the edge properties can be created with additional Python functions, if needed.~~

2.1.3 Filters (reduced mechanism)

Plotting the complete graph of a complex mechanism will be unreasonable in many cases, making it necessary to extract a subset in order to obtain a clearly arranged plot. Available filters are:

- It is possible to select only species containing a specified element. For example, organic chemistry is selected with "`element=C`".

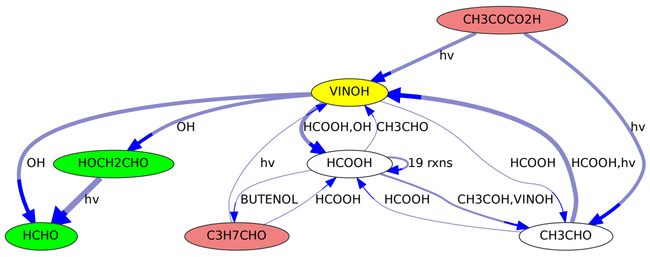


Figure 5. Sources and sinks of ethenol (VINOH, yellow) in MOM. Species that can produce VINOH are shown in red, and species that are produced from VINOH are shown in green. Species for which both applies are shown on a white background.

– From a multiphase mechanism, a specific phase can be selected, e.g., "`phase=gas`" or "`phase=aqueous`".

– Species that should not appear in a plot can be put into a blacklist, e.g., "`blacklist=CO`". Likewise, species that should always appear in the plot, even though they have already been removed by another filter, can be put into a whitelist, e.g., "`whitelist=CO2`".

– It is possible to remove slow reactions, i.e., all edges below a specified reaction rate, e.g., "`minrxnrate=1E-19`".

– When studying the degradation of organic molecules, it is often helpful to remove all edges where the number of C atoms increases, using "`l_filter_out_C_increase=True`". See Sect. 2.2 for details.

– The max-flow algorithm (Sect. 2.3.1) filters out all reactions that do not contribute to the maximum flow from a source species to a target.

– Filters for special cases can be created by adding individual functions to the Python code. For example, MEXPLORER includes a function for the JAMOC mechanism (Rosanka et al., 2021), which excludes all gas-phase species, all inorganic species, and all species that don't have any chemical sources or sinks in the aqueous phase (only transfer from the gas phase). The option is activated with "`vfilter=jamoc`". This filter can serve as a template when a new, user-defined function needs to be written for a specific task.

2.1.4 Interactive visualization

There is also an option to explore a mechanism interactively in a GTK+ window. It is possible to pan, zoom or rotate the graph. A group of vertices can be selected to drag or rotate them together. For reordering the vertices, the dynamic spring-block layout can be activated. Sources and sinks of selected species can be shown. Additional Python functions can be written to activate more keystrokes in the interactive window.

2.2 Mechanism development

MEXPLORER can be used as a tool during mechanism development. Several sanity checks can be performed quickly while a new mechanism is under construction.

– MEXPLORER can show precursors and successors of selected species. As an example, the graph in Fig. 5 was created with the command "`neighbors=VINOH,1,1`". It shows all species in MOM that are directly connected to ethenol (vinyl alcohol, VINOH). Such a visualization can help to find missing sources or sinks while a mechanism is under construction. Using larger numbers, e.g., "`neighbors=VINOH,2,3`" creates a more complex graph with all species that can produce VINOH within 2 steps and all species that are produced from VINOH within 3 steps.

– MEXPLORER can find all reaction pathways from a source to a target species. Fig. **??** was produced with "`vfilter=src_tgt`", "`src=C5H8`", and "`tgt=MACR`". It shows the reaction chain transforming isoprene into methacrolein.

Often, a mechanism is created to study the degradation of large organic molecules with a monotonic decrease of the number of carbon atoms with each reaction step ($\Delta C \le 0$). To remove all edges in which the number of carbon atoms increases ($\Delta C > 0$), "`l_filter_out_C_increase=True`" can be defined in the config file. Without such filtering, edges like ~~for example~~ CO $\rightarrow$ RO· from the reactions of peroxy radicals with carbon monoxide ($RO_2\cdot + CO \rightarrow RO\cdot + CO_2$) would remain in the graph, adding even more reactions and unwanted complexity to the plot.

– While creating a graph from the KPP input files, the mechanism can be checked for unintentional mass balance violations. Specific elements are selected with e.g., "`mass_balance=C,N,Cl`". Of course, this feature requires that the elemental composition of the chemicals has been defined in the input file.

– For most species in a mechanism, both chemical sources and sinks exist. A few have no sources or no sinks. This may be intentional (for example, terpenes are destroyed but not produced in atmospheric chemistry mechanisms) but it may also point to missing reactions in the mechanism under construction. With MEXPLORER, it is easy to scan the whole mechanism for such species (called "leaves" in graph theory) with "`l_show_primary=True`" and "`l_show_final=True`".

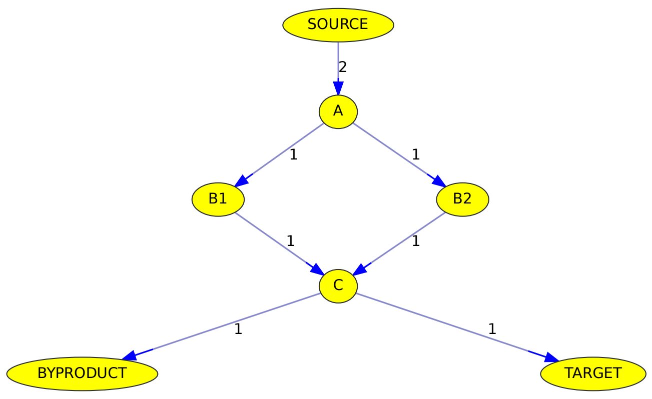


Figure 6. Simplified example of the max-flow problem from the source to a target. Edge labels denote reactions rates in arbitrary units. See text for details.

– Graph theory provides many algorithms that can assign key values, either to the whole graph or to individual vertices. These tools can be applied to chemical mechanisms, when they are available as graphs. For example, Silva et al. (2020) used the graph properties modularity and reciprocity to compare mechanisms of different complexity. MEXPLORER provides access to all algorithms which are implemented in graph-tool.

2.3 Mechanism evaluation

If turnover rates of individual reactions are available, they can be imported into MEXPLORER and used to evaluate the results of a model simulation.

2.3.1 The max-flow problem (finding important reaction pathways)

Often, a key question is not only ~~how much of a certain species has been produced but also which reaction pathways are the most important ones,~~ the magnitude of individual rates but also the overall rate going from a source to a target species. In terms of graph theory, this corresponds to a maximum flow problem. MEXPLORER ~~offers three algorithms~~ allows to choose between three algorithms which are provided by graph-tool:

– Edmonds–Karp algorithm (Edmonds and Karp, 1972)

– Push-relabel algorithm (Goldberg and Tarjan, 1986)

– Boykov-Kolmogorov algorithm (Boykov and Kolmogorov, 2004)

A review by Goldberg and Tarjan (2014) and the graph-tool documentation (https://graph-tool.skewed.de/static/doc/flow.html) provide additional information about these algorithms, including their time complexity (big O notation).

As an example, Fig. **??** shows the reactions leading from C_5H_8 (isoprene) to CO. Using the Edmonds–Karp algorithm, the thickness of the arrows indicates the importance of individual pathways.

Another example is shown in Fig. 4 where the reaction sequence from HCHO to CO_2 in JAMOC is analyzed. This mechanism contains gas-phase as well as aqueous-phase reactions. The Edmonds–Karp algorithm yields important pathways in both phases.

Although all algorithms determine the maximum flow, the calculated pathways are not necessarily identical. Figure 6 illustrates why the path is not unambiguous when parallel pathways exist: Although the source reacts at a rate of 2 (in arbitrary units), the overall flow from the source to the target is only 1, due to the bottleneck (called minimum cut in graph theory) from C to the target. The maximum flow can either be achieved by going from A to C via B1, or via B2. Thus it is impossible here to judge if B1 or B2 is the more important intermediate.

2.3.2 Strongly connected components (chemical “families”)

Mechanisms often contain very fast reactions which form catalytic cycles. They may also contain null cycles which neither produce or remove any species, i.e., have a net effect of zero. In atmospheric chemistry, species are often grouped into so-called families (Crutzen and Schmailzl, 1983). They are defined such that reactions inside the null cycles transform species only within the family but don’t change the sum of the family members. Taking the Chapman mechanism as an example again, an “odd oxygen” family can be defined as the sum of O and O_3. Neither reaction (R2) nor (R4), which together form a null cycle, change the amount of odd oxygen.

Trying to find important reactions in a complex mechanism can be hampered by fast reactions inside null cycles. Therefore, it is important to detect ~~them~~ those, and define suitable families. MEXPLORER can be used to locate fast cycles above a specified threshold in a complex mechanism. Using the command “`family=1E-14`”, MEXPLORER will list all families, in which the members can be interconverted at a rate of $\geq 10^{-14}$. The unit of this value is adopted from the input file with the reaction rates, here it is $\mathrm{mol\,mol^{-1}\,s^{-1}}$. When plotting the graph of the mechanism, all reactions inside the families are shown as red arrows. In terms of graph theory, MEXPLORER first creates a subgraph where the edge property “reaction rate” is above the specified threshold. Next, it finds all strongly connected components in the subgraph.

3 Summary and outlook

The MEXPLORER software was presented which can be used to analyze, reduce, and visualize complex chemical reaction mechanisms.

Currently, MEXPLORER can only import reaction rates at one selected time step. A planned extension is to read reaction rates at several time steps, e.g., for a whole day. This will allow the creation of movies with changing arrow widths, visualizing how important pathways may change throughout the diurnal cycle.

Code availability. MEXPLORER is available as a community tool published under the GNU General Public License[5]. The code and the user manual are available at https://doi.org/10.5281/zenodo.8132472. MEXPLORER will also be included in the next version of the CAABA/MECCA atmospheric chemistry box model at https://gitlab.com/RolfSander/caaba-mecca.

Competing interests. The author is a member of the editorial board of GMD. The peer-review process was guided by an independent editor, and the author has no other competing interests to declare.

Acknowledgements. The code uses the open source software graph-tool (Peixoto, 2014), Python and Graphviz.

References

Boykov, Y. and Kolmogorov, V.: An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision, IEEE Trans. Pattern Anal. Machine Intell., 26, 1124–1137, https://doi.org/10.1109/TPAMI.2004.60, 2004.

Chapman, S.: A theory of upper-atmospheric ozone, Mem. R. Meteorol. Soc., 3, 103–125, 1930.

Crutzen, P. J. and Schmailzl, U.: Chemical budgets of the stratosphere, Planet. Space Sci., 31, 1009–1032, https://doi.org/10.1016/0032-0633(83)90092-2, 1983.

Edmonds, J. and Karp, R. M.: Theoretical improvements in algorithmic efficiency for network flow problems, J. Assoc. Comput. Machinery, 19, 248–264, https://doi.org/10.1145/321694.321699, 1972.

Garay-Ruiz, D., Álvarez-Moreno, M., Bo, C., and Martínez-Núñez, E.: New tools for taming complex reaction networks: The unimolecular decomposition of indole revisited, ACS Phys. Chem. Au, 2, 225–236, https://doi.org/10.1021/ACSPHYSCHEMAU.1C00051, 2022.

Goldberg, A. V. and Tarjan, R. E.: A new approach to the maximum flow problem, in: STOC '86: Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing, pp. 136–146, Association for Computing Machinery, https://doi.org/10.1145/12130.12144, 1986.

Goldberg, A. V. and Tarjan, R. E.: Efficient maximum flow algorithms, Commun. Assoc. Comput. Machinery, 57, 82–89, https://doi.org/10.1145/2628036, 2014.

Maeda, S., Harabuchi, Y., Hayashi, H., and Mita, T.: Toward ab initio reaction discovery using the artificial force induced reaction method, Annu. Rev. Phys. Chem., 74, 287–311, https://doi.org/10.1146/ANNUREV-PHYSCHEM-102822-101025, 2023.

Martínez-Núñez, E., Barnes, G. L., Glowacki, D. R., Kopec, S., Peláez, D., Rodríguez, A., Rodríguez-Fernández, R., Shannon, R. J., Stewart, J. J. P., Tahoces, P. G., and Vazquez, S. A.: AutoMeKin2021: An open-source program for automated reaction discovery, J. Comput. Chem., 42, 2036–2048, https://doi.org/10.1002/JCC.26734, 2021.

Niemeyer, K. E. and Sung, C.-J.: On the importance of graph search algorithms for DRGEP-based mechanism reduction methods, Combust. Flame, 158, 1439–1443, https://doi.org/10.1016/J.COMBUSTFLAME.2010.12.010, 2011.

Peixoto, T. P.: The graph-tool python library, Figshare, https://doi.org/10.6084/M9.FIGSHARE.1164194, 2014.

Rosanka, S., Sander, R., Wahner, A., and Taraborrelli, D.: Oxidation of low-molecular-weight organic compounds in cloud droplets: development of the Jülich Aqueous-phase Mechanism of Organic Chemistry (JAMOC) in CAABA/MECCA (version 4.5.0), Geosci. Model Dev., 14, 4103–4115, https://doi.org/10.5194/GMD-14-4103-2021, 2021.

Sander, R., Baumgaertner, A., Cabrera-Perez, D., Frank, F., Gromov, S., Grooß, J.-U., Harder, H., Huijnen, V., Jöckel, P., Karydis, V. A., Niemeyer, K. E., Pozzer, A., Riede, H., Schultz, M. G., Taraborrelli, D., and Tauer, S.: The community atmospheric chemistry box model CAABA/MECCA-4.0, Geosci. Model Dev., 12, 1365–1385, https://doi.org/10.5194/GMD-12-1365-2019, 2019.

Sandu, A. and Sander, R.: Technical note: Simulating chemical systems in Fortran90 and Matlab with the Kinetic PreProcessor KPP-2.1, Atmos. Chem. Phys., 6, 187–195, https://doi.org/10.5194/ACP-6-187-2006, 2006.

Silva, S. J., Burrows, S. M., Evans, M. J., and Halappanavar, M.: A graph theoretical intercomparison of atmospheric chemical mechanisms, Geophys. Res. Lett., 48, https://doi.org/10.1029/2020GL090481, 2020.

Sumiya, Y., Harabuchi, Y., Nagata, Y., and Maeda, S.: Quantum chemical calculations to trace back reaction paths for the prediction of reactants, J. Am. Chem. Soc. Au, 2, 1181–1188, https://doi.org/10.1021/JACSAU.2C00157, 2022.

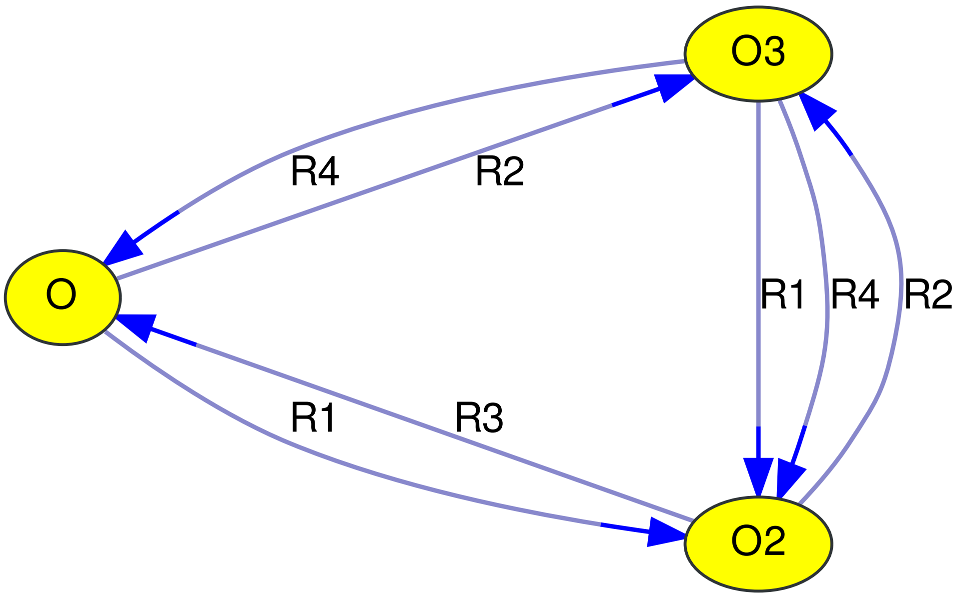
Tomlin, A. S. and Turányi, T.: Mechanism reduction to skeletal form and species lumping, in: Cleaner Combustion, edited by Battin-Leclerc, F., Simmie, J. M., and Blurock, E., pp. 447–466, Springer Verlag, Berlin, https://doi.org/10.1007/978-1-4471-5307-8_17, 2013.

Turányi, T. and Tomlin, A. S.: Analysis of Kinetic Reaction Mechanisms, Springer Verlag, Heidelberg, https://doi.org/10.1007/978-3-662-44562-4, 2014.

Watson, L. A., Shallcross, D. E., Utembe, S. R., and Jenkin, M. E.: A Common Representative Intermediates (CRI) mechanism for VOC degradation. Part 2: Gas phase mechanism reduction, Atmos. Environ., 42, 7196–7204, https://doi.org/10.1016/J.ATMOSENV.2008.07.034, 2008.

[5]http://www.gnu.org/copyleft/gpl.html

Westbrook, C. K., Pitz, W. J., Herbinet, O., Curran, H. J., and Silke, E. J.: A comprehensive detailed chemical kinetic reaction mechanism for combustion of n-alkane hydrocarbons from n-octane to n-hexadecane, Combust. Flame, 156, 181–199, https://doi.org/10.1016/J.COMBUSTFLAME.2008.07.014, 2009.



MEXPLORER-1.0.0 User Manual

The <u>ME</u>chanism e<u>XPLORER</u>

Rolf Sander

Air Chemistry Department
Max-Planck Institute of Chemistry
PO Box 3060, 55020 Mainz, Germany
`rolf.sander@mpic.de`

This manual is part of the electronic supplement of the article "MEXPLORER 1.0.0 – A mechanism explorer for analysis and visualization of chemical reaction pathways based on graph-theory" in Geosci. Model Dev. (2023), available at: `http://www.geosci-model-dev.net`

Contents

1 Installation

~~To install MEXPLORER, simply copy the complete `mexplorer` directory onto your computer. Ensure~~ First, ensure that these prerequisites are fulfilled:

Python: The python version must be at least 3.6. Newer versions should work as well, but are not necessary. ~~Several python packages are also needed. Missing packages can be installed with pip, e.g.:~~

Graph-tool: Graph-tool is a very fast Python module for the manipulation and analysis of graphs (`https://graph-tool.skewed.de`). ~~It is needed to visualize chemical reaction schemes as graphs.~~ The graph-tool software is very fast because it is a C++ library wrapped in Python. The downside of this architecture is that it cannot be installed with a simple pip command because of the C++ dependencies. Detailed installation instructions can be found at `https://git.skewed.de/count0/graph-tool/-/wikis/installation-instructions`. For questions about compilation, installation and usage, there is a web forum at `https://forum.skewed.de/c/graph-tool`. To report bugs and for feature requests, the issue tracker at `https://git.skewed.de/count0/graph-tool/-/issues` can be used.

Additional Python packages: Several additional python packages are needed (matplotlib, netCDF4, numpy, tabulate), as listed in the file `requirements.txt`. They can be installed with: `pip3 install -r requirements.txt`

CAABA/MECCA (optional): MEXPLORER can use reaction rates from the atmospheric chemistry boxmodel CAABA/MECCA (`http://www.rolf-sander.net/messy/mecca`) to identify important reaction pathways. The MEXPLORER code will be included in the `caaba/mecca/mexplorer` directory of CAABA/MECCA version 4.6.0.

To install MEXPLORER, you can now copy the complete `mexplorer` directory onto your computer. For MEXPLORER-related installation questions, an issue can be opened on my CAABA/MECCA gitlab page at `https://gitlab.com/RolfSander/caaba-mecca/-/issues`.

2 Usage

MEXPLORER is started from the command line, supplying a configuration (`*.ini`) file from the python configparser (`https://docs.python.org/3/library/configparser.html`) as a parameter:

`./mexplorer.py` *myconfig*`.ini`

or

`python3 mexplorer.py` *myconfig*`.ini`

The distribution includes several examples that can be used out-of-the-box to get acquainted with the code, e.g.:

- List and plot the species and reactions of the Chapman mechanism:
  `./mexplorer.py chapman.ini`
- Create a file with methane chemistry in graph-tool xml format and perform mass-balance checks:
  `./mexplorer.py CH4_def.ini`
- Plot the most important reaction pathways in the degradation from isoprene to CO:
  `./mexplorer.py mom_C5H8_to_CO_maxflow.ini`
- Plot the complete sulfur chemistry contained in the Master Chemical Mechanism (MCM):
  `./mexplorer.py mcm_S.ini`
- List and plot all species that can produce pyruvic acid within 2 steps or are produced from pyruvic acid within 2 steps:
  `./mexplorer.py mom_pyruvic_acid.ini`
- Show all “primary” species (those that have no precursors) and all “final” species (those that have no sinks) in the MCM:
  `./mexplorer.py mcm_sources_sinks.ini`

3 Code structure

The config (`*.ini`) files are in the `mexplorer/ini` directory. They contain four sections: `[define]`, `[analyze]`, `[info]`, and `[plot]`. A fifth section called `[DEFAULT]` may be added to define settings that apply to all sections.

The code works in four steps: First, a chemical mechanism is converted into a digraph and saved. Next, the mechanism is analyzed and can be modified. This may result in the creation of a smaller subgraph of manageable size. Finally, selected information about the graph can be presented in text format or visualized, either creating a pdf or interactively.

3.1 Step 1: Defining the graph

The script `define_graph.py` expects several input files with information about the chemical mechanism in the input directory:

- Chemical species (including their elemental composition) are defined in KPP format (Sandu and Sander, 2006) in the species file ~~`mecca.spc`~~`(*.spc)`.
- Chemical ~~reactions~~ equations (including their equation tags) are defined in KPP format in the equation file ~~`mecca.eqn`~~`(*.eqn)`.
- Reaction rates (optional) can be defined in netCDF format (`https://www.unidata.ucar.edu/software/netcdf`) in `caaba_mecca_rr.nc`.
- Overall Interaction Coefficients (OIC) data from skeletal mechanism reduction (Sander et al., 2019) can be defined in `OIC.dat` (optional).

The resulting graph is directly transferred to step 2, and it can also be saved to a graph-tool compatible `*.xml` file. The following properties are defined for the graph:

- Graph properties:
  - timestamp = time stamp from mecca.eqn
  - batchfile = batch file from mecca.eqn
  - wanted = wanted string from mecca.eqn
  - mechanism = mechanism
  - rxnrate = reaction rate
- Vertex properties:
  - name = name of the chemical species
  - atoms = string with elemental composition
  - fillcolor = fill color (for plotting)
- Edge properties:
  - color = color (for plotting)
  - eqntag = equation tag
  - prodstoic = stoichiometric factor of product
  - reactant = reactant
  - rxnrate = reaction rate

To define a graph, the following properties can be set in the `[define]` section of the config (`*.ini`) file:

- `inputdir = `*dirname*
  Define the directory containing the input files. *dirname* must be a subdirectory of `input/`. Example:
  `inputdir = mcm`
- `spcfilename = `*filename*
  (default = `mecca.spc`)
  The name of the species file that defines the chemical species. Example:
  `spcfilename = myfile.spc`
- `eqnfilename = `*filename*
  (default = `mecca.eqn`)
  The name of the equation file that defines the chemical equations. Example:
  `eqnfilename = myfile.eqn`
- `mass_balance = `*elements*
  Define the elements for which the mass balance is checked. *elements* must be a comma-separated list of elements. Example:
  `mass balance = C,H,N,O,Br,Cl,I,S,Hg`
- `outputfile = `*filename*
  (default = `inputdir+'.xml.gz'`)
  Define the name of the output file to which the graph is written in graph-tool xml format. Example:
  `outputfile = mom.xml.gz`
- `rxnfilename = `*filename*
  (default = `caaba_mecca_rr.nc`)
  Define the name of the input file from which the reaction rates are read. Example:
  `rxnfilename = rates.nc`
- `timestep = `*number*
  (default = `-36`)
  Define which time step from the reaction rate file is used. Negative values can be used to count from end of model run. Example:
  `timestep = -36`
- `verbose = [True|False]`
  Activate verbose output.

Even though MEXPLORER can handle most of the KPP syntax, there are two limitations that need to be considered:

- KPP allows equations with negative products. This trick is sometimes used to indicate that a species is destroyed in a reaction but does not affect its reaction rate. For example, GEOS-Chem contains the reaction:
  `HMS + OH = 2 SO4 + CH2O - SO2`
  In order to use such a reaction in MEXPLORER, it is necessary to transfer the negative product into a positive reagent, resulting in:
  `HMS + OH + SO2 = 2 SO4 + CH2O`
- KPP allows multiline equations that are split over several lines. Since MEXPLORER reads the equation file line by line, it cannot parse such equations. To fix this problem, newline characters inside equations must be converted to spaces before the file is used as input for MEXPLORER.

3.2 Step 2: Analysing the reaction mechanism

To analyze a graph, the following properties can be set in the `[analyze]` section of the config (`*.ini`) file:

- `blacklist = `*species*
  Define the blacklisted species, i.e., those that are removed from the graph. *species* must be a comma-separated list of species. Example:
  `blacklist = CO,CO2`
- `element = `*element*
  If set, only compounds that contain the specified element are kept in the graph. Example:
  `element = S`
- `family = `*value*
  Find chemical families (strongly connected components) with members that can be interconverted at the specified rate or faster. Example:
  `family = 1E-14`

- `inputfile = ` *filename*
  Define the name of the graph-tool xml input file from which the graph is read. Example:
  `inputfile = mom.xml.gz`
- `l_filter_out_C_increase = [True|False]` Remove edges where the number of carbon atoms increases.
- `maxflow = ` *value*
  (default = `-1`)
  If `maxflow` > 0, show only species that contribute more than a fraction of maxflow to the total flow from source to target. Example:
  `maxflow = 1E-3`
- `maxflow_algorithm = [edmonds_karp|`
  `push_relabel|boykov_kolmogorov]`
  (default = `edmonds_karp`)
  Algorithm for the maximum flow calculation.
- `minrxnrate = ` *value*
  Remove slow reactions from the graph where the rate is less than `minrxnrate`. Example:
  `minrxnrate = 1E-19`
- `neighbors = ` *species,value1,value2*
  Show only near neighbors, i.e., precursors that are up to *value1* steps away and successors that are up to *value2* steps away. Example:
  `neighbors = CH3COCO2H,1,2`
- `outputfile = ` *filename*
  Define the name of an output file to which the filtered graph is written in graph-tool xml format. Example:
  `outputfile = mom.xml.gz`
- `phase = [gas|aqueous]`
  Keep only species in the specified phase. It is assumed here that the names of all aqueous-phase species end with `_a`.
- `src = ` *species*
  Define the name of a chemical source species for maxflow calculations or for `vfilter==src_tgt`. Example:
  `src = C5H8`
- `tgt = ` *species*
  Define the name of a chemical target species for maxflow calculations or for `vfilter==src_tgt`. Example:
  `tgt = MACR`
- `vcolor = [nC|Fe|oic]`
  Assign a vertex property that can be used as fill color when the graph is plotted. Current possibilities are: Number of carbon atoms (`nC`), iron contents (`Fe`), and overall interaction coefficients (`oic`). Additional Python functions can be written in order to apply other criteria.
- `verbose = [True|False]`
  Activate verbose output.
- `vfilter = [src_tgt|jamoc]`
  Select a subset of the species. Current possibilities are the selection of all species involved in the reaction sequence from a source to a target (`src_tgt`) and the selection of all species from the JAMOC mechanism (`jamoc`). Additional Python functions can be written in order to apply other criteria.
- `vlabel = oic`
  Assign a vertex property that can be used as a vertex label when the graph is plotted. Setting `vlabel = oic` will add the OIC values. Additional Python functions can be written in order to apply other criteria.
- `whitelist = ` *species*
  Define the whitelisted species, i.e., those that are included in the graph even if they have been blacklisted before. *species* must be a comma-separated list of species. Example:
  `whitelist = CO`

The resulting graph is directly transferred to step 3, and it can also be saved to a `*.xml` file.

3.3 Step 3: Graph information

To obtain information about a graph, the following properties can be set in the `[info]` section of the config (`*.ini`) file:

- `inputfile = ` *filename*
  Define the name of the graph-tool xml input file from which the graph is read. Example:
  `inputfile = mom.xml.gz`
- `l_show_components = [True|False]`
  List all strongly connected components (“families”).
- `l_show_edges = [True|False]`
  List all edges.
- `l_show_edges_delta_C = [True|False]`
  List all edges, including the change in carbon number.
- `l_show_final = [True|False]`
  List all sinks, i.e., species that do not react further in the mechanism.
- `l_show_graph = [True|False]`
  Show general information about the graph.
- `l_show_primary = [True|False]`
  List all sources, i.e., species that are not created in the mechanism.
- `l_show_reactions = [True|False]`
  List all reactions.
- `l_show_species = [True|False]`
  List all species and their elemental composition.
- `l_sort_rxnrate = [True|False]`
  In the list all reactions, sort reactions by reaction rate, not equation tag.
- `show_sinks = ` *species*
  Define *species* for which a list of sinks (i.e., species that are produced from it) is generated. Example:
  `show_sinks = DMS`
- `show_sources = ` *species*
  Define *species* for which a list of sources (i.e., species that produce from it) is generated. Example:
  `show_sources = SO2`
- `verbose = [True|False]`
  Activate verbose output.

3.4 Step 4: Graph visualization

To visualize a graph, the following properties can be set in the `[plot]` section of the config (`*.ini`) file:

- `backend = [graphviz|cairo]`
  Define the backend for plotting (default = `graphviz`).
- `e_plus =`
  Define additional edge properties. Several examples are shown in `example.ini`.
- `edge_label = [reactant|eqntag]`
  Use the name of the reactant or the equation tag as edge labels (default = `reactant`).
- `g_plus =`
  Define additional graph properties. Several examples are shown in `example.ini`.
- `inputfile =` *filename*
  Define the name of the graph-tool xml input file from which the graph is read. Example:
  `inputfile = mom.xml.gz`
- `l_interactive = [True|False]`
  Create an interactive graph on the screen that can be controlled with keystrokes and the mouse. Press `k` to see the complete keymap and all mouse actions.
- `outputfile =` *filename*
  Define the name of a pdf file to which the plot is written. Example:
  `outputfile = mom.pdf`
- `v_plus =`
  Define additional vertex properties. Several examples are shown in `example.ini`.
- `verbose = [True|False]`
  Activate verbose output.

References

Sander, R., Baumgaertner, A., Cabrera-Perez, D., Frank, F., Gromov, S., Grooß, J.-U., Harder, H., Huijnen, V., Jöckel, P., Karydis, V. A., Niemeyer, K. E., Pozzer, A., Riede, H., Schultz, M. G., Taraborrelli, D., and Tauer, S.: The community atmospheric chemistry box model CAABA/MECCA-4.0, Geosci. Model Dev., 12, 1365–1385, https://doi.org/10.5194/GMD-12-1365-2019, 2019.

Sandu, A. and Sander, R.: Technical note: Simulating chemical systems in Fortran90 and Matlab with the Kinetic PreProcessor KPP-2.1, Atmos. Chem. Phys., 6, 187–195, https://doi.org/10.5194/ACP-6-187-2006, 2006.