MOLGEN

STRUCTURE ELUCIDATION

Reference Guide / Version 5.0

March 9, 2021

Markus Meringer - Christoph Rücker - Alfred Wassermann

PREFACE

The program system **MOLGEN** is devoted to generating all structures (connectivity isomers, constitutions) that correspond to a given molecular formula, with optional further restrictions, e.g. presence or absence of particular substructures. **MOLGEN** arose from the idea to provide an efficient and portable tool for molecular structure elucidation in chemical industry, research, and education.

Historically, up to version **MOLGEN** 3.5, the main intention was to generate structures as fast as possible. The result is one of the fastest generators for molecular structures. However, applications showed that generator efficiency is not the only important topic for molecular structure elucidation.

Thus in the development of series **MOLGEN** 4.x the interface was organized in a much more flexible way. Now advanced restrictions can be passed to the generator that are obtained from spectroscopy. **MOLGEN–MS** and **MOLGEN–QSPR** are special versions that arose from these efforts.

In generating huge libraries without advanced restrictions, the performance of **MOLGEN** 4.x is not comparable to that of **MOLGEN** 3.5. Series **MOLGEN** 5.x is now intended to combine the advantages of both approaches, i.e. the efficiency of **MOLGEN** 3.5 and the flexibility of **MOLGEN** 4.x. To achieve this, the software was reimplemented based on a totally new concept.

The first version **MOLGEN** 5.0 of this new series was released in 2007. We herewith provide a minor revision 5.04.

Contents

Chapter 1

An introduction to MOLGEN

The program system **MOLGEN** allows to compute the complete set of structures (connectivity isomers, structural formulas, constitutions) that correspond to a given molecular formula or a set of molecular formulas. Often the molecular formula is sufficient as input, the generator will then use default values for the valences of all atoms included. Of course, it is possible to override defaults, by e.g. specifying particular atom valences.

The generation is free of redundance, i.e. no structure is generated twice within a single run. Moreover, the construction is complete, which means that the full set of all possible structures is obtained that correspond to a given molecular formula and, optionally, further restrictions. For example, given the input

$$C_8H_{16}O_2$$

MOLGEN will construct exactly 13 190 pairwise different structures. This example already shows that, in general, the number of structures corresponding to a given molecular formula is very large.

Therefore it is often desirable to reduce the output by imposing additional restrictions. For this purpose, together with a molecular formula, substructures may be specified that must be contained in each isomer constructed, or that on the contrary are not allowed. For example, if together with molecular formula $C_8H_{16}O_2$ a carboxyl group is prescribed, **MOLGEN** will generate exactly 39 structures. If additionally the isopropyl group is excluded, then out of the 39 structures just 27 will remain.

Chapter 2

Installation

2.1. Requirements

MOLGEN 5.0 runs under Windows XP / 7 / 10 and Linux.

Access to the internet is most convenient for downloading the software. The licensing procedure necessitates a network card. About 5 MB harddisk space is occupied by the program. The space required to save generated molecular libraries depends on the particular problem.

With respect to editing substructures, any standard molecule editor supporting MDL molfile will do. You may also enter substructures in form of SMILES strings. For interpreting SMILES, **MOLGEN** cooperates with the software OpenBabel, available from `http://openbabel.sourceforge.net`. Add the installation directory of OpenBabel to your PATH in order to help **MOLGEN** to find OpenBabel. Please refer to the manual of your operating system.

By default, generated structures are written in an MDL sd file (`.sdf`). An sd file may be viewed e.g. using freely available software such as jmol[1], ACD Chemsketch[2], or the Accelrys DS Visualizer.[3] Alternatively, structures generated may be written by MOLGEN in mb4 format (`.mb4`) and then viewed in MOLGEN-QSPR.

2.2. Installing MOLGEN under Windows

The installation on an MS Windows system works as follows:

1. Get file 'mgen50.zip' from our ftp-server, as you were instructed by e-mail, and save the file on the harddisk of the licensed computer. Recall that **MOLGEN** is licensed to a specific computer.
2. Decompress the .zip file. For this purpose you can use almost any decompression software availabe for Windows, such as WinZip or PkZip. If these are not available, use the built-in decompression tool in Windows as follows:
   a) Open the Windows Explorer, select the directory where you saved 'mgen50.zip' and double-click that file with the left mouse button. Another folder named 'molgen50' opens.
   b) Move this folder to your destination location (e.g. C:\Program Files) using Drag&Drop. The content of the ZIP file is decompressed and copied to the destination directory (e.g. C:\Program Files\molgen50).

[1] `http://sourceforge.net/projects/jmol`
[2] `http://www.acdlabs.com/chemsketch`
[3] `http://www.accelrys.com/products/downloads/ds_visualizer`

3. In this command line version, **MOLGEN** may be accessed by command prompt only:
   a) In order to obtain a command prompt in Windows, select 'Run' from the Start menu, enter '`cmd`' in the appearing dialog box, and confirm with OK.
   b) Change to the directory where you installed **MOLGEN**, e.g.

      ```
      cd C:\Program Files\molgen50
      ```

Proceed with Section 2.4.

2.3. Installing MOLGEN under LINUX

To install **MOLGEN** under LINUX, carry out the following steps:

1. Get file 'mgen50.tgz' from our ftp server, as you were instructed by e-mail, and save the file on the harddisk of the licensed computer. Recall that **MOLGEN** is licensed to a specific computer.
2. From the directory containing the saved file, type

   ```
   tar -xzf molgen50.tgz
   ```

   A subdirectory 'molgen50' will be created containing all necessary files.
3. Change to this directory

   ```
   cd molgen50
   ```

Proceed with Section 2.4.

2.4. Testing the installation

Once installed as described above, **MOLGEN** is independent of the operating system.

For a first test, enter:

```
mgen C6H6 -v
```

The output should be similar to the following:

```
*** MOLGEN 5.04 generator.  (Mar 9 2021)
*** (C) Dr.  Markus Meringer, PD. Dr.  Christoph Rücker, Prof.
Dr.  A. Wassermann
*** Licensed for:  Test Release

Generate all structures from given fuzzy molecular formulas.

verbose mode 1
Skip aromatic Kekule structures.


No.  structures:  217
```

If the last line contains the number 217 as shown above, then the generator is working correctly. (Note: In this test 217 structures are just generated, not saved on disk.)

Chapter 3

User Guide

We are now going to explain in more detail **MOLGEN**'s main procedures.

Originally, **MOLGEN** was designed to generate all molecular graphs (structures) to a given molecular formula. As time went on, with respect to the input we realized that sometimes compounds of interest are not described by a single molecular formula but by a range of molecular formulas. The present version **MOLGEN** 5.0 was developed to solve such problems.

In describing molecular structure we distinguish several levels of detail:

1. **Fuzzy molecular formula**
   Instead of prescribing exact occurrence numbers for each chemical element (or more exactly for each atom type, cf. Subsection 3.2.1.1), for broader coverage numerical intervals are allowed here. On the other hand, for each atom its state may be partially prescribed (valence, charge, hybridization, etc., see Subsection 3.2.1.2) in a fuzzy as in an exact molecular formula.
2. **(Exact) Molecular formula**
   For each element symbol with optionally restricted state, its exact occurrence number is given.
3. **Atom state pattern**
   For each atom in the molecular formula, its state is fully defined, including the numbers of bonds of various types and the number of hydrogens attached to it.
4. **Molecular graph**
   The connections between atoms are described as covalent bonds. In mathematical terms, a molecular structure can be understood as a graph, possibly with double, triple or aromatic edges.

Given a description of molecular structure on any level, **MOLGEN** is able to generate all corresponding descriptions on more detailed levels.

3.1. Input and output of the generation process

Entering appropriate information, the user may start on any of the first three levels. When starting with atom state patterns, you have to indicate this with the option

`-sp` ***state_pattern*** **[...]** Start generation with one or more atom state pattern(s) as input.

The input, one or more fuzzy or exact formulas or atom state patterns (separated by blanks), is entered on the command line or is read from a file using the option

`-f` ***filename*** Read input from the specified file (or from standard input, if *filename* is '`-`').

By default, all molecular graphs corresponding to the input are generated. However, by excluding candidates generated on intermediate levels, the number of generated molecular graphs may be kept manageable. Therefore, options are available to generate intermediate level information:

`-gf` Generate exact formulas only (for given fuzzy formulas).
`-gp` Generate atom state patterns only (for given fuzzy or exact formulas).

Without any further option, the generator will produce no output at all. If you are interested just in the number of corresponding structures, enter option `-v`. Options `-v2`, `-v3`, `-v4`, `-v+`, `-v2+`, `-v3+`, `-v4+` request more detailed information on the generation process:

`-v` Display the number of generated structures.
`-v2` Display additionally each fuzzy molecular formula treated.
`-v3` Display additionally each exact molecular formula treated.
`-v4` Display additionally each atom state pattern treated.

The corresponding `+` options (`-v+` through `-v4+`) display additionally a `+` sign for each generated structure.

As a rule, you will be interested in the generated structures themselves. To obtain these, store them into a file (or alternatively to standard output) with the following option:

`-o` ***filename*** Write the generated structures to the specified file (or to standard output, if *filename* is '`-`'). Without this option, the generated structures are immediately lost.

Generated molecular formulas or atom state patterns are written into the file in simple ASCII format. Structures are written as MDL SDF by default. The following three options affect the output format:

`-explH` Write explicit hydrogen atoms into the output file (for MDL SDF).
`-l2d` Calculate 2D coordinates for all atoms in all structures in the output file (for MDL SDF).

To use **MOLGEN** 4.1 binary format MB4 instead, enter a filename with extension '.mb4' or use the following option.

`-mb4` Use MB4 format for storing molecular graphs (instead of MDL SDF).

Sometimes many more structures are generated than you are willing to inspect. In such a situation the store option is useful

`-store` ***n*`-`*m*** Store structures $\#n$ to $\#m$ only.

While the store option does not influence the generation process itself, the stop option allows you to limit the generation:

`-stop` ***n*** Stop the generation process after n structures are generated.

A **MOLGEN** run may be aborted at any time using Ctrl-C. Structures generated before abortion will be saved in the output file.

In the following the various generator input specifications are described. Note that in this manual for the examples no output options are explicitly given. Always add options `-v`*n* or `-o` *filename* to obtain visible results.

3.2. Fuzzy and exact molecular formulas

The difference between a fuzzy and an exact molecular formula is the exact prescription of all occurrence numbers in the latter, while intervals of occurrence numbers appear in the former.

3.2.1. Entering a fuzzy or exact molecular formula

An exact molecular formula such as $C_5H_{10}SO_2$ is entered as a string,

`C5H10S[val=2]O2`,

or simply `C5H10SO2`, since for S valence is 2 by default. The string contains the following information:

— **Atom types**, which are chemical element symbols;
— optional **atom states**, describing the environment of an atom within the molecular structure (e.g. its valence).
— **atom occurences**, i.e. the number of atoms of given type and state occuring in a structure.

For a fuzzy molecular formula, each atom occurrence number may be replaced by an interval of numbers, e.g. `C5H10S[val=2]O0-2`, interpreted as $C_5H_{10}SO_{0-2}$.

Note that an element symbol may occur more than once as input for a formula, i.e. in different atom states, e.g. `C2H4N[val=3]0-1N[val=5]0-1`.

3.2.1.1. Atom types (element symbols)

An element symbol is one or two letters, the second of which must be lowercase. The case of the first letter is uncritical as long as the formula is unambiguous. However, if element symbols without explicit state and multiplicity occur in the formula, ambiguity may arise from lowercase letters, such as in `cs2` (interpreted as Cs_2) and `CS2` (C_1S_2). So we advise to uppercase the first letter of each element symbol.

Usually an atom type is an element symbol from the Periodic Table of Elements. However, you may define atom types not yet known to the system. These are called *user-defined atom types*. Initially, **MOLGEN** does not know anything about a user-defined atom type, therefore you have to specify at least its valence as an atom state (see below). As an example, `C4H8Qs[val=2]3O` will produce structures of formula $C_4H_8Qs_3O$, where the user-defined atom type `Qs` has valence 2.

It is a good habit to use a special letter such as `Q` for user-defined atom types. This will minimize ambiguities otherwise possible, such as `CL[val=2]2` (interpreted as C_1L_2) versus `cl[val=2]2` (interpreted as Cl_2, i.e. two chlorine atoms of extraordinary valence 2).

3.2.1.2. Atom states

Atom states describe the environment of an atom within the molecular structure, they are specified immediately after the element symbol and are enclosed in square brackets. Within the brackets, one or more of the following terms may be listed, separated by commas.

`val=`***n*** The valence of an atom in the structure. This is the total number of covalent bonds that connect the atom to its neighbors (including bonds to hydrogen; a double bond is counted twice). Default valences are as follows: C 4, H 1, O 2, N 3, F 1, Cl 1, Br 1, I 1, S 2, Si 4; several other elements according to the octet rule. For an element to occur with valence other than default, the valence has to be specified explicitly, for example `C2H6S[val=6]O2`.

`chg=`***n*** The charge of an atom in the structure. If valence is not specified explicitly, this will adjust the default valence appropriately according to the octet rule.

`rad=`***n*** Specification of an atom as a radical center. rad=1 means one unpaired electron. If valence is not specified explicitly, this will adjust the default valence appropriately according to the octet rule.

`iso=`***n*** Isotope specification. The number n denotes the (integer) difference to the (integer) standard atom mass.

Using the following terms an atom state may be specified even more precisely.

`sp3`, `sp2_n`, `sp2_a`, `sp_st`, `sp_dd` Hybridization, where `sp2_n` and `sp2_a` denote sp^2 in nonaromatic and aromatic systems, respectively; `sp_st` and `sp_dd` denote an sp-hybridized atom bearing a single and a triple, or two double bonds, respectively.

`h=`***n*** Number of hydrogen atoms adjacent to an atom.

`s=`***n*** Number of single bonds (to non-hydrogen atoms) adjacent to an atom.

`d=`***n*** Number of double bonds adjacent to an atom.

`t=`***n*** Number of triple bonds adjacent to an atom.

`a=`***n*** Number of aromatic bonds adjacent to an atom.

Examples:

— `C[d=0]2H6S[val=6,d=2]O2`
— `C[d=0]2H6S[val=6,d=2,h=0]O2`
— `C2H6S[val=6,d=2,h=0]O[d=1]2`

Note that **MOLGEN** 5.0 has a special bond type 'aromatic' and a corresponding hybridization 'sp2_a'. Requesting simply double bonds or sp2_n hybridization will exclude aromatic systems. For example

— `C[d=1]6H6` or `C[sp2_n]6H6`
will each produce 6 structures, benzene not among them.
— `C[a=2]6H6` or `C[sp2_a]6H6`
will each produce one structure, i.e. benzene.
— `C[a=2]8H8` or `C[a=2]4H4`
produce 0 structures each, since cyclooctatetraene and cyclobutadiene are recognized as nonaromatic.

To generate structures with both kinds of sp^2 atoms, use atom sums, see Subsection 3.2.2.2, for example

— `C[sp2_n]0-6C[sp2_a]0-6H6 -sum C=6`
will produce 7 structures: benzene and 6 non-aromatic isomers.

3.2.1.3. Atom occurences

Specify an occurrence number immediately after the corresponding element symbol and its optional state description. As usual, omission of an occurrence number defaults to 1. For example, input `C2H6O` is interpreted as $C_2H_6O_1$.

For fuzzy molecular formulas, specify occurrences as intervals, e.g. `C4-6H6-8`. This feature is particularly useful in combination with the following restrictions.

3.2.2. Restrictions on molecular formulas

3.2.2.1. Global properties

The following restrictions may be imposed on molecular formulas to be generated from a fuzzy molecular formula. For all of these, enter an exact value (integer) or an interval.

`-atoms` ***n*[-*m*]** Specify the number of atoms in a molecular structure (including hydrogens). For example,

```
mgen C2H0-6F0-6Cl0-6Br0-6I0-6 -atoms 8
```

generates ethane and all halogenated ethanes.

`-valence` ***n*[-*m*]** The sum of valences over all atoms.
This is double the number of bonds (bonds to H included, double and triple bonds counted as two and three bonds, respectively). We call this restriction 'valence', in contrast to the number of bonds without considering bond multiplicities, which is called 'bonds', see Subsection 3.3.2.

`-mass` ***n*[-*m*]** The mass of the molecular structure, i.e. the sum over atom masses. For example,

```
mgen C4-6H4-10 -mass 78
```

generates the 217 benzene isomers.

`-charge` ***n*[-*m*]** Charge of the molecular structure, i.e. the sum over all atom charges.

`-isotope` ***n*[-*m*]** Sum over all isotopic mass differences.

`-radicals` ***n*[-*m*]** Total number of unpaired electrons in the molecular structure.

Restrictions on charge, isotopic mass differences, and unpaired electrons make sense only if you explicitly allow charged, isotopic or radical atom states in a molecular formula. By default, charges, isotopes and unpaired electrons are not considered.

3.2.2.2. Atom sums

Atom sums are sums of occurrence numbers of atom types/states with a restriction (an exact value or an interval) on the calculated value.

`-sum` ***atom_ sum*=*n*[-*m*]** Specify a sum of occurrence numbers.

Examples:

— `mgen C6H0-6Cl0-6 -sum H+Cl=6`
generates all C_6H_6 hydrocarbons and their chlorinated analogs;
— `mgen C1-10H4-22 -sum H-2C=2`
generates all alkanes up to the decanes;
— `mgen C1-10H4-22 -sum H-2C=0-2`
generates all alkanes plus monounsaturated alkenes plus saturated monocyclic hydrocarbons of up to ten carbon atoms;
— `mgen C1-10H4-22`
(without any further option) generates all kinds of polyunsaturated and polycyclic C_1-C_{10} hydrocarbons, as well.

This feature can be used to allow alternative atom states for an element. In the following example generation is restricted to structures containing at most two nitrogen atoms of valence 3 or 5:

```
mgen C2H4N[val=3]0-2N[val=5]0-2 -sum N[val=3]+N[val=5]=0-2
```

In an atom sum expression element symbols match all atom states of an element that are mentioned in the molecular formula. Thus the following is a synonymous formulation:

```
mgen C2H4N[val=3]0-2N[val=5]0-2 -sum N=0-2
```

3.3. Atom state patterns

A state pattern describes a molecular structure by listing the fully defined state of each atom as described in Subsection 3.2.1.2, including the number of attached hydrogens.

3.3.1. Syntax

Each atom is listed separately. For coding atom states the following symbols are used.

`H`***n*** the number of attached hydrogens
`=`***n*** the number of adjacent double bonds
`#`***n*** the number of adjacent triple bonds
`~`***n*** the number of adjacent aromatic bonds

If n=0, the symbol `H`, `=`, `#`, or `~` is omitted; if n=1, the numeral 1 is omitted. This information together with an atom's valence defines the number of adjacent single bonds.

Example:

`CH#C#CH=CH=CH2CH`

is the state pattern corresponding to 3-ethynylcyclobutene, where

`CH#` codes a C atom bearing one H and a triple bond,
`C#` is a C atom bearing a triple bond and a single bond to a non-H atom,
`CH=` is a C atom bearing one H, one double bond and one single bond to a non-H atom,
`CH2` is a C atom bearing two H and two single bonds to non-H atoms,
`CH` is a C atom bearing one H and three single bonds to non-H atoms.

Recall that atom state patterns can be prescribed using the following option:

`-sp` ***state_pattern*** **[...]** Start generation with one or more atom state pattern(s) as input (separated by blanks).

Here '-sp' stands for 'start with pattern'. The atom expressions within an atom state pattern may be input in any order.

Examples:

— `mgen -sp CH#C#CH=CH=CH2CH`
generates two structures, 3-ethynylcyclobutene and 3-(2-propynyl)cyclopropene, while
— `mgen -sp CH#C#C=CH=CH2CH2`
leads to three structures, 1-ethynylcyclobutene, 1-(2-propynyl)cyclopropene, and (2-propyn-1-ylene)cyclopropane.

3.3.2. Restrictions

The following restrictions influence the number and type of generated atom state patterns. Again exact numbers or intervals are allowed.

`-maxbond` ***n*** Maximal allowed bond multiplicity (i.e. 1, 2, or 3).
`-bonds1` ***n*****[-*****m*****]** Total number of single bonds between atoms in the molecular structure (including bonds to hydrogens).
`-bonds2` ***n*****[-*****m*****]** Total number of double bonds.
`-bonds3` ***n*****[-*****m*****]** Total number of triple bonds.
`-bondsa` ***n*****[-*****m*****]** Total number of aromatic bonds.
`-bonds` ***n*****[-*****m*****]** Number of bonds between atoms without counting bond multiplicity (including bonds to hydrogens).
`-cycles` ***n*****[-*****m*****]** Number of cycles in the molecular structure. This is the number of bonds that have to be broken in order to obtain an acyclic structure, e.g. naphthalene has two, not three cycles, cubane has 5 cycles. Example:

```
mgen C6H6 -cycles 4
```

generates prismane and another 13 tetracyclic (and saturated) benzene isomers.
`-conn` ***n*****[-*****m*****]** Number of connected components of the molecular graph.

By default connected graphs only are generated, i.e. conn=1. On the other hand, bond and cycle restrictions are inactive by default.

The following formula holds for any molecular graph.

$$\text{atoms} + \text{cycles} = \text{bonds} + \text{connected components}$$

Thus, if two of the three quantities number of atoms, of bonds, and of cycles are prescribed for constant conn (e.g. for a connected molecular graph, conn=1), there is no choice for the third.

3.4. Molecular graphs

Molecular graphs describe the molecular structure in terms of covalent bonds. In order to reduce the number of isomers generated, the following restriction is useful:

`-ringsize` ***n*[-*m*]** Specify the allowed ring sizes. This command does not prescribe a ring of that size (range).

Any closed path in the molecular graph is considered a ring. For example, naphthalene contains rings of sizes 6 and 10, cubane has 4-, 6- and 8-membered rings. If you allow 4-membered rings only, you will miss cubane.

Both power and limitations of the options described hitherto are easily seen in the following example, where we try to restrict the molecular formula $C_6H_5NO_2$ to nitrobenzene.

Example:

— `mgen C6H5NO2`
results in 444 199 structures, nitrobenzene not among them.
— `mgen C6H5N[val=5]O2`
gives 1 038 793 structures, among them nitrobenzene.
— `mgen C6H5N[val=5,d=2]O2`
renders 122 699 structures.
— `mgen C6H5N[val=5,d=2,h=0]O2`
results in 98 687 structures.
— `mgen C6H5N[val=5,d=2,h=0]O[d=1]2`
results in 3 893 structures.
— `mgen C6H5N[val=5,d=2,h=0]O[d=1]2 -cycles 1`
renders 1 436 structures.
— `mgen C6H5N[val=5,d=2,h=0]O[d=1]2 -ringsize 6-9`
gives 452 structures.
— `mgen C6H5N[val=5,d=2,h=0]O[d=1]2 -cycles 1 -ringsize 6-9`
results in 140 structures.
— `mgen C6H5N[val=5,d=2,h=0]O[d=1]2 -cycles 1 -ringsize 6`
produces still 110 structures.
— `mgen C[sp2_n]6H5N[val=5,d=2,h=0]O[d=1]2 -cycles 1 -ringsize 6`
results in 10 structures, nitrobenzene not among them.
— `mgen C[sp2_n]0-6C[sp2_a]0-6H5N[val=5,d=2,h=0]O[d=1]2 -cycles 1 -ringsize 6 -sum C=6`
results in 11 structures.
— `mgen C[sp2_a]6H5N[val=5,d=2]O2`
produces exactly one structure: nitrobenzene.

The example demonstrates the demand for more powerful restrictions, i.e. for substructure restrictions.

3.4.1. Substructures

Thousands or even millions of structures are certainly too many for any further manual processing. However, spectroscopy usually can identify substructures that with certainty either are present in the compound under examination or are missing. **MOLGEN** offers the option to take such substructures into consideration.

Therefore, first provide substructures in a format understandable to **MOLGEN**. Besides the historical **MOLGEN** file formats MBF and MB4, **MOLGEN** supports files in MDL MOL format directly. File format is recognized by the extension (.mbf, .mb4, .mol, or .sdf). For editing substructures, any standard molecule editor supporting MOL files is suitable, for example MDL ISIS Draw or ACD Chemsketch.

When using ISIS Draw, edit a substructure (intended for data base queries), select it and export it as molfile into the **MOLGEN** 5.0 directory. For **ANY atoms** (matching any element symbol except hydrogen), use the symbol 'A'. Advanced bond types are available by right-clicking on a bond. This is especially necessary for aromatic bonds, as **MOLGEN** distinguishes single and double bonds strictly from aromatic bonds. Besides the 'aromatic' bond type, **MOLGEN** supports the **extended bond types** 'single or aromatic', 'double or aromatic', 'single or double' and 'any bond' offered by ISIS Draw. Note that e.g. valence 5 for nitrogen has to be explicitly specified. This is done in ISIS Draw by right-clicking on an atom and choosing 'Edit atom'. Here you can also specify atom charges, isotopes and radical centers.

Substructures thus created are used in **MOLGEN** as follows. The option `-substr` allows a range of occurrence numbers of a substructure.

`-substr open` ***n*[-*m*] *filename***
`-substr induced` ***n*[-*m*] *filename***
: Thereby structures are generated that contain between n and m copies of the substructure contained in *filename*.

Thus, expressions

`-substr open 0-2` *substructure1*`.mol`
`-substr induced 0-2` *substructure1*`.mol`

allow but do not prescribe at most 2 copies of substructure *substructure1*.

In particular, omission of the optional '-*m*' part prescribes exactly n copies, and n=0 (without an m specified) excludes a substructure. To prescribe at least n copies of the substructure, use a sufficiently large number for m, e.g. m=99.

Keywords **open** or **induced** in the above expressions determine which substructures in a molecular graph are recognized to match a given substructure: In the **induced** case, if free valences on different atoms in a given substructure get connected to each other, this is considered a non-match. Thus, additional zero-length bridges within a substructure, or higher bond multiplicities, will cause a non-match. In the **open** case, however, such variations are recognized as a match. In mathematical terms, an induced substructure is an *induced subgraph* of the molecular graph, while an open substructure is a subgraph in general.

Examples:

To exclude all six-membered rings, edit a cyclohexane ring (without hydrogens, specifying all heavy atoms as 'A', not 'C'), save it as molfile as e.g. `general_cyclohexane.mol` and enter:

`-substr open 0 general_cyclohexane.mol`
: Thereby, e.g. cyclohexane, cyclohexene, cyclohexa-1,3-diene, cyclohexa-1,4-diene, benzene, benzyne, piperidine, pyridine, bicyclo[2.2.0]hexane substructures, etc., will be considered matches of the forbidden substructure, and consequently all structures generated will not contain any of these substructures.

`-substr induced 0 general_cyclohexane.mol`
: causes e.g. cyclohexene, cyclohexa-1,3-diene, cyclohexa-1,4-diene, benzene, benzyne, pyridine, bicyclo[2.2.0]hexane substructures to be considered non-matches of the forbidden substructure, and consequently these substructures may occur in the structures generated. Piperidine and of course cyclohexane are recognized as matches and therefore will not occur in any structure generated.

Thus,

— `mgen C8H11N -cycles 1-4 -ringsize 5-9`
: results in 11586 compounds, among them being substituted pyridines, dihydro- and tetrahydropyridines, piperidines, benzenes, cyclohexadienes, cyclohexenes, and cyclohexanes;

— `mgen C8H11N -cycles 1-4 -ringsize 5-9`
`-substr open 0 general_cyclohexane.mol`
: generates 6290 compounds, none of which contains any 6-membered ring;

— `mgen C8H11N -cycles 1-4 -ringsize 5-9`
`-substr induced 0 general_cyclohexane.mol`
: leads to 10857 compounds, among them pyridines, dihydro- and tetrahydropyridines, benzenes, cyclohexadienes and cyclohexenes, but no piperidines or cyclohexanes. So the piperidines and cyclohexanes filtered out amount to 729. In fact,

— `mgen C8H11N -cycles 1-4 -ringsize 5-9`
`-substr induced 1-4 general_cyclohexane.mol`
: produces exactly 729 substituted piperidines and cyclohexanes, and this set is identical to the set filtered out above.

To prescribe the presence of a benzene ring, edit a literal benzene ring (without hydrogens, all heavy atoms specified as carbon, all bonds specified as aromatic), name it e.g. `benzene.mol`, and enter

`-substr induced 1-99 benzene.mol`
: Thereby, dehydrobenzene (benzyne) or a zero-bridged benzene ring will not be considered a match, and consequently structures containing a benzyne but not a benzene will not be generated. Of course, structures containing both a benzene and a benzyne may occur.

`-substr open 1-99 benzene.mol`
: will cause benzyne or a zero-bridged benzene ring to be considered a match, and consequently structures containing a benzyne but no benzene substructure will be generated.

Let us revisit the example of nitrobenzene to illustrate the power of substructure restrictions.

Example:

— `mgen C6H5N[val=5]O2 -substr induced 1 benzene.mol`
results in 143 structures, each containing a benzene substructure, and nitrobenzene being among them;
— `mgen C6H5N[val=5]O2 -substr open 1 benzene.mol`
results in 312 structures, many of which contain a (presumably undesired) zero-bridged benzene ring;
— `mgen C6H5N[val=5,h=0]O2 -substr induced 1 benzene.mol`
renders 7 structures;
— `mgen C6H5N[val=5,d=2]O2 -substr induced 1 benzene.mol`
generates nitrobenzene as the only structure.
— `mgen C6H5N[val=5]O2 -substr induced 1 nitro.mol`
results in 685 structures, among them nitrobenzene;
— `mgen C6H5N[val=5]O2 -substr induced 1 nitro.mol -cycles 1`
gives 197 structures;
— `mgen C6H5N[val=5]O2 -substr induced 1 nitro.mol -cycles 1 -ringsize 6`
renders 14 structures;
— `mgen C6H5N[val=5]O2 -substr induced 1 nitro.mol -substr induced 1 benzene.mol`
of course delivers nitrobenzene as the only structure.

Recall that for the examples to work appropriately it is important that the bonds in benzene.mol are of type 'aromatic' and that the nitrogen in nitro.mol has valence 5.

If the open source program OpenBabel is installed on your computer (see http://openbabel.sourceforge.net), you may specify (small) substructures conveniently as SMILES strings directly in the command line

`-substr open` ***n*[-*m*]** `-smi` ***SMILESstring*** or
`-substr induced` ***n*[-*m*]** `-smi` ***SMILESstring***
since OpenBabel will translate the SMILES string into MOL format.

In SMILES, ANY atoms (matching any element symbol except hydrogens) are encoded by the character '*'. However, there is no possibility to specify aromatic bonds, so far. Thus, this feature is quite limited.

You may have a list of substructures to be excluded from any generated library. Collect such substructures in a MDL SDF file and use the option

`-badlist` ***filename*** All substructures listed in *filename* are excluded from generation.

Two sdf files of 'bad' substructures are shipped together with **MOLGEN**, named `badlist.sdf` and `badlist2.sdf`. The former contains 39 highly strained saturated and unsaturated small mono-, bi-, and polycyclic structures that are considered 'not viable' by many chemists (Figure 3.1). The latter is a collection of 14 'not viable' bridged aromatic structures, shown in Figure 3.2. Though such lists are, of course, somewhat arbitrary, they are useful for removing obviously unwanted structures, as demonstrated in the following examples.

Example:

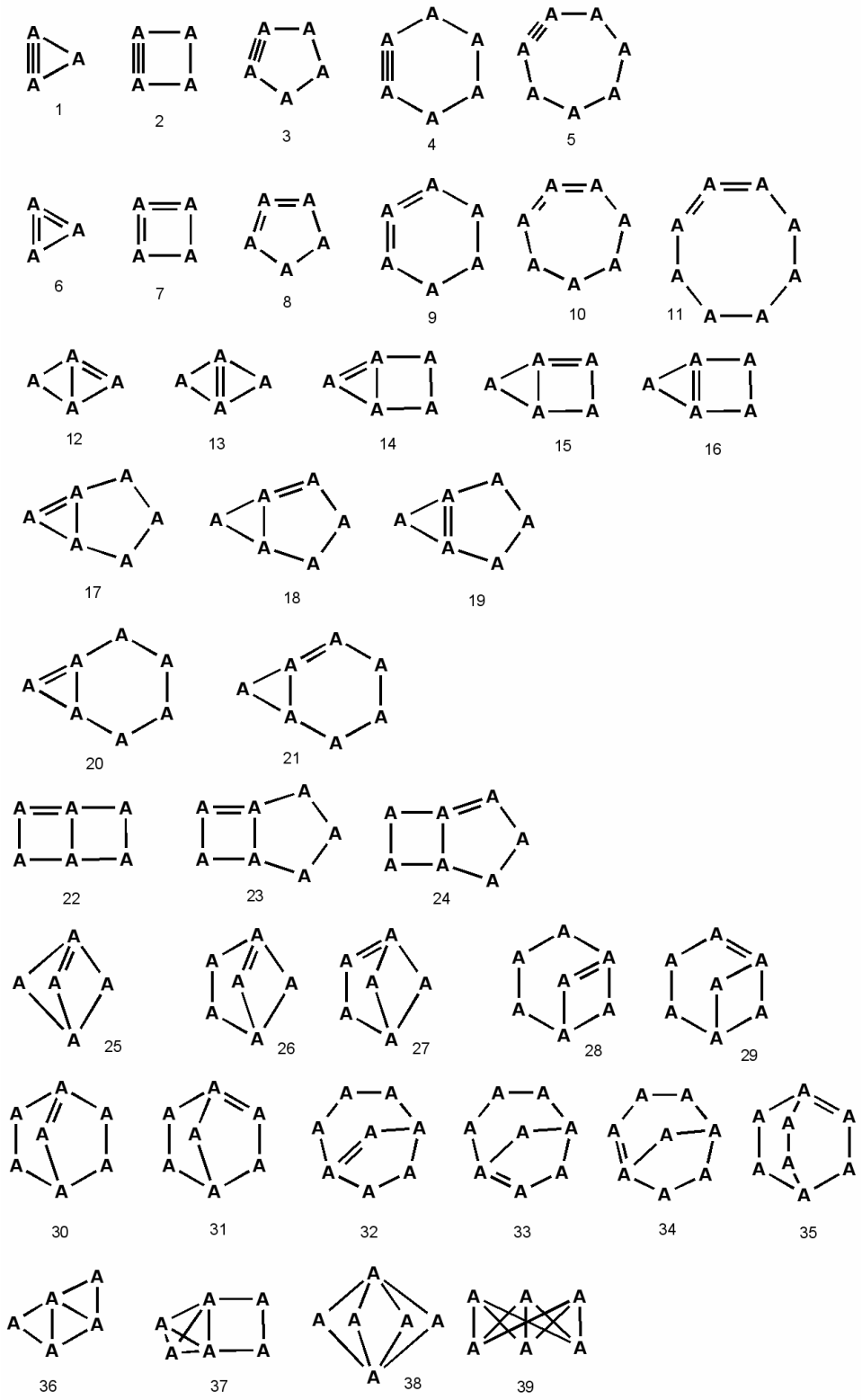

Figure 3.1. 'Bad' cyclic and unsaturated substructures contained in `badlist.sdf`.

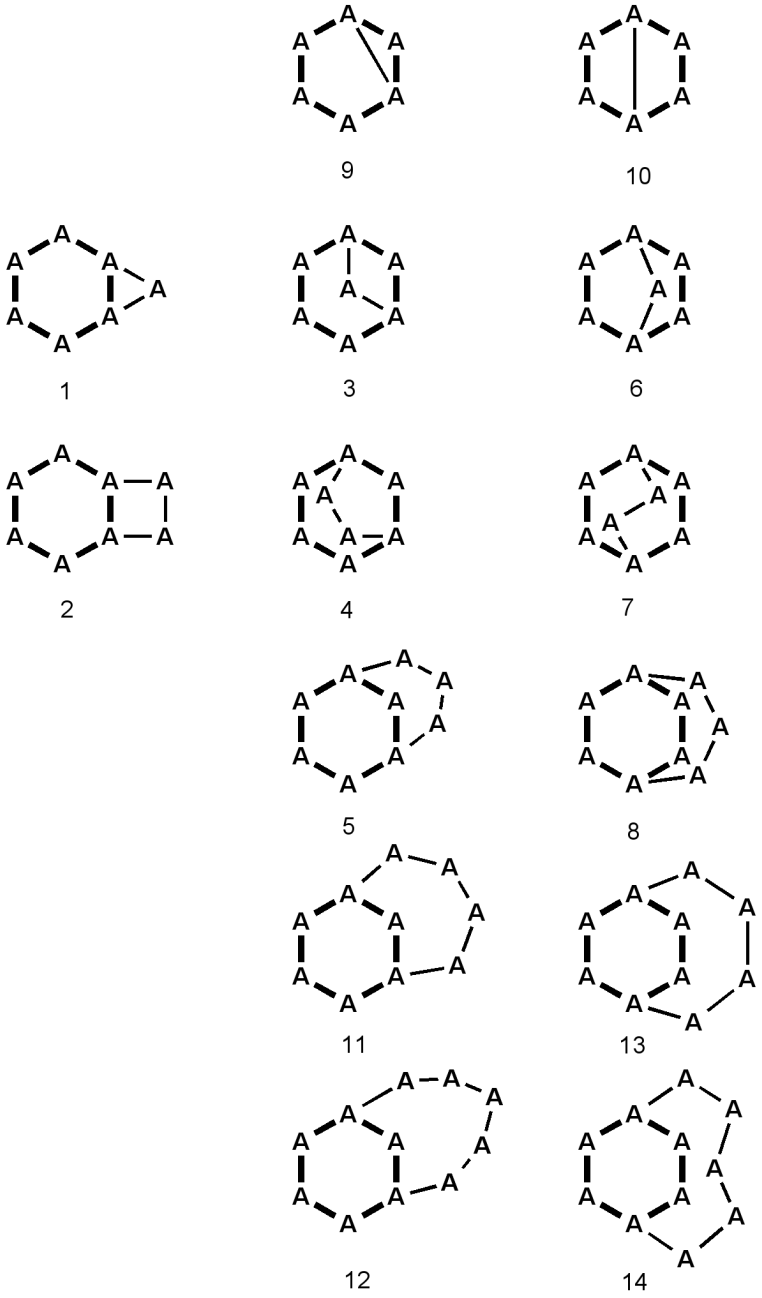

Figure 3.2. 'Bad' bridged aromatic substructures contained in `badlist2.sdf`. Aromatic bonds are symbolized here by thick lines.

— `mgen C6H6`
generates all 217 mathematically possible benzene isomers;
— `mgen C6H6 -badlist badlist.sdf`
results in no more than 66 isomers.

Though 151 isomers are removed thereby, the remaining set still contains those isomers that are known compounds either themselves or as more or less substituted derivatives, such as prismane, Dewar benzene, benzvalene, fulvene, bi-cyclopropenyl, etc.

— `mgen C6H5N[val=5]O2 -substr open 1 benzene.mol`
generates 312 structures (see above);

— `mgen C6H5N[val=5]O2 -substr open 1 benzene.mol -badlist badlist2.sdf`
results in nitrobenzene as the only product.

The `-badlist filename.sdf` option is synonymous to `-substr open 0` with respect to each structure contained in file `filename.sdf`. Because of the 'open', not only all structures contained in that list, but their more highly unsaturated or additionally bridged analogs, as well, will be excluded from generation.

Obviously, the user may edit these badlists or create ones himself.

3.4.2. Aromaticity

MOLGEN 5.0 has a special bond type 'aromatic' for aromatic bonds. Consequently, cyclically conjugated double bonds forming an aromatic system are not generated. Rather, the corresponding structure is generated with the aromatic ring made of aromatic bonds.

Therefore **MOLGEN** has a built-in aromaticity detector plus filter that is based on the famous 4n+2 π-electrons rule (Hückel rule). In the current version cyclically conjugated rings of 6, 10, 14, etc. members are considered aromatic. In a future version, additional rings such as pyrrol, furan, thiophen, tropylium, cyclopentadienide etc. will be recognized as aromatic.

Note that aromaticity handling takes some time and resources. If desired, it may be deactivted by entering the option

`-noaromaticity` Deactivate aromaticity filter.

Thereby, benzene is generated with single and double bonds instead of aromatic bonds. Thus, 1,2-dimethylbenzene (o-xylene) will be generated twice, having either a single or a double bond connecting the substituted ring atoms.

For example,

```
mgen C[sp2_n]10H8 -ringsize 6-10
```

results in 6 molecular graphs, none of which corresponds to naphthalene, whereas

```
mgen C[sp2_n]10H8 -ringsize 6-10 -noaromaticity
```

results in 16 molecular graphs, two of which correspond to naphthalene;

```
mgen C[sp2_a]10H8
```

produces 4 structures, among them naphthalene and azulene.

Chapter 4

Quick Reference

Syntax:

`mgen` *`formula(s)`* `[`*`options`*`]`
`mgen -f <`*`file`*`> [`*`options`*`]`

Examples:

```
mgen C6H6
mgen C6H0-6Cl0-6 -sum H+Cl=6 -substr induced 1 benzene.mol
mgen C[sp2_n]2-4C[sp3]2-4H6 -sum C=6 -ringsize 4
mgen C4H8O[d=1]S[val=2]0-1S[val=4]0-1 -sum S=1
```

General options

`-h` Print this help page.

Options controlling the type of input and output

`-sp` Start generation with atom state pattern(s) as input. Default is to start with fuzzy or exact molecular formulas.

`-f` ***filename*** Read (additional) input (molecular formulas or atom state patterns) from specified file (or from standard input, if the given *filename* is '`-`')

`-gf` Generate exact formulas as output instead of molecular graphs.

`-gp` Generate atom state patterns as output instead of molecular graphs.

`-v`**`[`***`lvl`***`][+]`** (i.e. `-v`, `-v2`, `-v3`, `-v4` or `-v+`, `-v2+`, `-v3+`, `-v4+`)
Verbosity; display some information during generation process. The amount of information is controlled by the optional level *lvl*. The + versions display in addition a `+` sign as visual feedback for each generated structure.

`-o` ***filename*** Write the generated output to the specified file (or to standard output, if *filename* is '`-`') Without this option, the generated structures get lost immediately.

`-explH` Write explicit hydrogen atoms into the output file (for MDL SDF).

`-l2d` Calculate 2D coordinates for all atoms in all structures in the output file (for MDL SDF).

`-mb4` Use **MOLGEN** 4.1 binary format MB4 for storing molecular graphs (instead of MDL SDF).

`-store` ***n-m*** Store only structures of the specified range.

`-stop` ***n*** Stop generation process after generating n structures.

Restrictions for generating exact formulas

`-atoms` ***n*[-*m*]** Specify the number of atoms (including hydrogens).
`-valence` ***n*[-*m*]** The sum of valences over all atoms.
`-mass` ***n*[-*m*]** The integral mass of generated structures.
`-charge` ***n*[-*m*]** The charge of generated structures.
`-isotope` ***n*[-*m*]** The sum over all isotopic mass differences.
`-radicals` ***n*[-*m*]** The total number of unpaired electrons.
`-sum` ***atom_sum*=*n*[-*m*]** Specify a sum of occurence numbers, e.g. `Cl+H=6`.

Restrictions for generating atom state patterns:

`-maxbond` ***n*** Maximal allowed bond multiplicity (<=3).
`-bonds1` ***n*[-*m*]** Restrict number of single bonds.
`-bonds2` ***n*[-*m*]** Restrict number of double bonds.
`-bonds3` ***n*[-*m*]** Restrict number of triple bonds.
`-bondsa` ***n*[-*m*]** Restrict number of aromatic bonds.
`-bonds` ***n*[-*m*]** Restrict number of bonds, counted without multiplicity.
`-cycles` ***n*[-*m*]** Restrict number of cycles.
`-conn` ***n*[-*m*]** Specify number of connected components. By default, only connected structures are allowed.

Restrictions for generating molecular graphs:

`-ringsize` ***n*[-*m*]** Restrict ring sizes.
`-substr` **[**`open`|`induced`**]** ***n*[-*m*]** ***filename***
`-substr` **[**`open`|`induced`**]** ***n*[-*m*]** **-smi** ***SMILES***
A substructure is entered as a file in MDL MOL format. Alternatively, if OpenBabel is available, you can specify a substructure directly on the command line as SMILES code.
The given substructures may occur n to m times in the structure.
If the keyword `induced` is given, a structure having additional bonds between the atoms comprising the substructure (this includes higher bond multiplicities than in the substructure) is considered a non-match. Otherwise, in the `open` case, a structure is considered a match even if it has additional bonds between its atoms corresponding to the substructure.
`-badlist` ***filename*** All substructures listed in *filename* are excluded from generation. Such a file should be in MDL SDF format.
`-noaromaticity` Deactivate aromaticity filter.
I.e. consider aromatic doublettes (written with single and double bonds) as different structures. By default, aromatic systems are recognized and stored as such, containing aromatic bonds.