MIDAS-G v1.2

1. Copyright

2. Quick Start

MIDAS-G is a multi-threading tool implemented in C++. It has two modes. The internal mode is used for investigating fragmentation rules, while the official mode works as an advanced version of MIDAS [1] with the additional ability of using graph grammars.

2.1 Binary Package

Users can use the pre-built standalone executable files that have been built for 64-bit versions of operating systems. The Linux version is available at bin/linux/, while the Windows version is available at bin/win. Both versions do not require additional environment setting. Apple does not encourage static linking (please refer to https://developer.apple.com/library/content/qa/qa1118/_index.html), so we provide two versions at bin/mac/. MIDAS-G_static_mac is a single-threading version, which can run on most Mac machines, while MIDAS-G_static_mac_mt is a multiple-threading version, which requires libiomp5.dylib. Installing clang-omp by homebrew will also install libiomp5.dylib. Please make sure the path to the folder containing libiomp5.dylib is included in DYLD_LIBRARY_PATH. Or you can add it manually by the following command.

```
export DYLD_LIBRARY_PATH=${your_libiomp5_pyth}:$DYLD_LIBRARY_PATH
```

In the rest of this document, we assume the binary package name is MIDAS-G

2.2 Using MIDAS-G

To get command information, please run the following command.

```
./MIDAS-G –h
```

The above command will print all available command options. On windows, please remove "./" in the above command and commands given in the rest of this section.

The complete list of options of MIDAS-G is given as follows.

```
-s spectra_file_path -w working_directory -c configure_file_path
-o output_directory -m mode (internal or official) -a annotation_file_name(optional)
-r results_file_name
-r and -a are just for official mode.
If -r is not specified, the default full output file name is spectra_file_name.meb
If -a is not specified, the default full annotation file name is spectra_file_name.AFT2
If working director is not specified, the current directory will be used as working directory.
If output directory is not specified, the working directory is the output directory.
The current version doesn't search subdirectory
```

Here are some examples. Please refer to the following sections for configuration file setting and input file format.

If MIDAS-G is used for investigating fragmentation rules, an example command is given as follows.

```
./MIDAS-G -s ../../sample/sample.ms2  -c ../../sample/sample.cfg  -o output/  -m internal
```

where sample.ms2 is the annotated MS/MS data file with internal format, sample.cfg is the configuration file, and "internal" indicates the internal mode.

If MIDAS-G is used as an advanced version of MIDAS [1] for identifying metabolites from non-annotated MS/MS data, an example command is given as follows.

```
./MIDAS-G -s ../../sample/sample.FT2  -c ../../sample/sample.cfg  -o output/  -m official
```

where sample.FT2 is the FT2 file, while "official" indicates official mode.

3. Configuration File Setting

The configuration file format of MIDAS-G is similar to that of MIDAS [1]. An example configure file is available at sample/sample.cfg. A line starts with "#" is for comments. The usage of each option is given in the comments of the example configure file. There are some key options users need to specify.

```
Metabolite_Database = path_of_the_database_file
```

It is better to use full path in the above option.

```
Default_Polarity = positive # or negative
```

Please use "positive" and "negative" for $[M+H]^+$ and $[M-H]^-$, respectively.

```
Graph_Grammar = true # or false
```

If a graph grammar is used for recognizing bonds, "Graph_Grammar" should be "true", otherwise, it should be "false". If "Graph_Grammar" is false, MIDAS-G will ignore all specified grammars.

```
Graph_Grammar_Path{no_4} = path_of_grammar_file
Graph_Grammar_Weight{no_4} = 2.0
```

Within "{}", "no_4" is the grammar id. Please avoid using space in grammar id. "2.0" is the grammar weight. It is better to use full path for "path_of_grammar_file". Please note that MIDAS-G can work with one or more graph grammars, if each grammar has its unique id.

4. Grammar File Format

An EGG grammar file should follow GML format [2]. Four example files are available at folder "sample" with extension name gml. A grammar is composed of one or more production rules. Each production rule is represented using "production [ ]". Within "[]", there are "productionID", "directionStatus", "left[subgraph [ ]]", and "right[subgraph [ ]]". "productionID" should be a unique string. "directionStatus" should be "Undirected". "left[subgraph [ ]]", and "right[subgraph [ ]]" represents the left hand side and right hand side, respectively. Each part is composed of "node [ ]" and "edge []". The former represents each vertex, while the latter represents each edge. "node [ ]" has two required fields. One is a unique node id, which should be a positive integer. The other is label, which should be a string. If label is "Initial" or ""Initial"", it is initial status. If label is "*ANY*", it can match any string. Usually, label represents the atom, e.g., "C" and "O" for carbon and oxygen, respectively. If the node is "flexible" (this only happens in the right hand side), it requires "flexible true" and specifying the corresponding node on the left hand side like "left 1". "edge [ ]" contains several fields. "id" requires an unique id that is a positive integer. "source" and "target" are ids of two connected nodes. Please note that "source" and "target" are interchangeable, because all edges are undirected. "label" uses format "field_1|field_2|field_3|field_4". Each field can be "*ANY*" for matching all possible cases. If the edge represents a linear bond, field_1 is "linear", otherwise "ring". If the edge represents an aromatic bond, field_2 is "t", otherwise "f". If the edge represents a conjugated bond, field_3 is "t", otherwise "f". The right most field, field_4, represents bond type classified by RDKit [3]. Bond type includes "UNSPECIFED", "SINGLE", "DOUBLE", "TRIPLE", "QUADRUPLE", "QUINTUPLE", "HEXTUPLE", "ONEANDAHALF", "TWOANDAHALF", "THREEANDAHALF", "FOURANDAHALF", "FIVEANDAHALF", "AROMATIC", "IONIC", "HYDROGEN", "THREECENTER", "DATIVEONE", "DATIVE", "DATIVEL", "DATIVER", "OTHER", and "ZERO".

Please note that if a bond is successfully parsed by two or more grammars, it will be assigned the highest weight.

5. Input file format

MIDAS-G supports two file formats for MS/MS data. If MS/MS data is not annotated, i.e., the measured metabolite is unknown, all spectra should be stored in FT2 format [1], [4], [5]. An example file with FT2 format is available at sample/sample.FT2. Otherwise, The following internal format is suitable.

As for each spectrum, the first line follows the following format.

```
*	Identifier	Scan_Number Precursor_Neutral_Mass	InChI
```

Then each line contains each peak.

```
+m/z	Intentisy
```

The last line should be "//". The following example contains two spectra.
An example file with this format is available at sample/sample.ms2. Please note that the official mode accepts FT2 format, while the internal mode accepts the internal MS/MS data formats.

MIDAS-G also supports two compound database formats. One is the official format supported by MIDAS [1]. There are four columns in a database file, which are "Identifier", "Name", "InChI", and "Links". Column names are given in the first line. Except the first line, each line represents a compound. An example file with this format is available at sample/sample.mdb. The other format does not have the title line and contains five columns, which are "Identifier", "InChI", "precursor neutral mass", "name", and "Links". The latest version of MIDAS-G uses information of "Identifier", "Name", and "InChI". An example file with this format is available at sample/sample.mdb.old. Please note that both database file formats are supported under internal mode and official mode.

Reference

[1] Y. Wang, G. Kora, B. P. Bowen, and C. Pan, "MIDAS: A database-searching algorithm for metabolite identification in metabolomics," *Anal. Chem.*, vol. 86, no. 19, pp. 9496–9503, 2014.

[2] M. Himsolt, "GML : A portable graph file format," Passau, Germany, 1999.

[3] G. Landrum, "RDKit: Open-source cheminformatics," 2013. [Online]. Available: http://www.rdkit.org.

[4] C. Pan, B. H. Park, W. H. McDonald, P. A. Carey, J. F. Banfield, N. C. VerBerkmoes, R. L. Hettich, and N. F. Samatova, "A high-throughput de novo sequencing approach for shotgun proteomics using high-resolution tandem mass spectrometry.," *BMC Bioinformatics*, vol. 11, p. 118, Jan. 2010.

[5] Y. Wang, T.-H. Ahn, Z. Li, and C. Pan, "Sipros/ProRata: A versatile informatics

system for quantitative community proteomics," *Bioinformatics*, vol. 29, no. 16, pp. 2064–2065, 2013.

Appendix

Building from Source Code

1. Linux

Install RDKit 2015_03_1 or above by following the instruction at link given as follows.
http://www.rdkit.org/docs/Install.html
Using anaconda is the easiest way, but you may also try other ways given in that instruction. Please note that Boost is required for installing RDKit.

Set environment variables of RDBASE and BOOSTBASE

```
export RDBASE=/the_path_of_RDKit
export BOOSTBASE=/the_path_of_boost
export
LD_LIBRARY_PATH=$BOOSTBASE/lib:$RDBASE/lib:$LD_LIBRARY_PATH
```

For example,

```
export RDBASE=/Users/yingfeng/software/RDKit/rdkit-Release_2015_03_1
export BOOSTBASE=/usr/local/Cellar/boost/1.60.0_1
export
LD_LIBRARY_PATH=$BOOSTBASE/lib:$RDBASE/lib:$LD_LIBRARY_PATH
```

We suggest to add these three commands at ~/.bashrc.

Enter folder src/MIDAS_G/ and run

```
export EGGBASE=../EGG
```

Make sure g++ refers to g++ 4.8 or above.

```
make
```

The new generated MIDAS-G is the executable binary file.

2. Mac

Install homebrew by following the instruction at the link given as follows.
http://brew.sh/

Install clang-omp using homebrew.

```
brew install clang-omp
```

Install RDKit 2015_03_1 by following the instruction at the link given as follows.
http://www.rdkit.org/docs/Install.html

Using anaconda is easiest way, but you may also try other ways given in that instruction. Please note that Boost is required for installing RDKit.

Set environment variables of RDBASE and BOOSTBASE

```
export RDBASE=/the_path_of_RDKit
export BOOSTBASE=/the_path_of_boost
export
DYLD_LIBRARY_PATH=$BOOSTBASE/lib:$RDBASE/lib:$DYLD_LIBRARY_PAT
H
```

For example,

```
export RDBASE=/Users/yingfeng/software/RDKit/rdkit-Release_2015_03_1
export BOOSTBASE=/usr/local/Cellar/boost/1.60.0_1
export
DYLD_LIBRARY_PATH=$BOOSTBASE/lib:$RDBASE/lib:$DYLD_LIBRARY_PAT
H
```

Enter folder src/MIDAS_G/ and run

```
export EGGBASE=../EGG
make –f Makefile_mac
```

The new generated MIDAS-G is the executable binary file.

3. Windows

Download RDKit from https://github.com/rdkit/rdkit/archive/Release_2016_09_2.zip

Download MSYS2 from https://msys2.github.io/.

Install MSYS2 by following https://msys2.github.io/. The rest of this section assumes you install it at C:\msys64 and your account is abc.

Run following three commands for installing MinGW, Boost, and cmake.

```
pacman -S mingw-w64-x86_64-toolchain
pacman -S mingw64/mingw-w64-x86_64-boost
pacman -S mingw64/mingw-w64-x86_64-cmake
```

Decompress RDKit package at C:\msys64\home\abc\software\RDKit\rdkit-Release_2016_09_2

In MSYS2 shell, run

```
cd ~/software/RDKit/rdkit-Release_2016_09_2
```

Use command "ls", you should many files like

appveyor.yml Code Data External license.txt rdkit rdkit-config-version.cmake.in README.md ReleaseNotes.md setup.cfg CMakeLists.txt Contrib Docs INSTALL Projects rdkit-config.cmake.in README Regress Scripts Web

Run following commands.

```
mkdir build
cd build
export PATH=$PATH:/mingw64/bin
```

Run the following command twice.

```
cmake -DCMAKE_C_COMPILER=gcc -DCMAKE_CXX_COMPILER=g++ -DBOOST_ROOT=/mingw64 -DBOOST_LIBRARYDIR=/mingw64/lib -DMSVC_RUNTIME_DLL=C:/Windows/System32/msvcr120.dll -DRDK_USE_BOOST_SERIALIZATION=OFF -DRDK_BUILD_INCHI_SUPPORT=ON -DRDK_BUILD_PYTHON_WRAPPERS=OFF -G "MinGW Makefiles" ..
```

Please note that DMSVC_RUNTIME_DLL=C:/Windows/System32/msvcr120.dll is for windows 7.

Run the following commands.

```
mingw32-make
mingw32-make install
export RDBASE=~/software/RDKit/rdkit-Release_2016_09_2
export PATH=$PATH:$RDBASE/lib
mingw32-make test
```

If all tests are passed, we can build MIDAS-G now.

Decompress the package of MIDAS-G, and enter src/MIDAS_G.
Run the following command.

```
g++ -std=c++11 -O2 -fopenmp -I../EGG -I/mingw64/include -I${RDBASE}/External -I${RDBASE}/Code -L/mingw64/lib -L${RDBASE}/lib main.cpp Database.cpp MidasCore.cpp ParseConfigure.cpp SettingInfo.cpp SpectrumUnit.cpp ParseSpectraFile.cpp ../EGG/EggCore.cpp ../EGG/FilesOperation.cpp ../EGG/GML.cpp ../EGG/GraphBasics.cpp ../EGG/GraphGrammar.cpp ../EGG/ParseRegularGraph.cpp ../EGG/Production.cpp ../EGG/ReadGML.cpp ../EGG/ReadGrammar.cpp ../EGG/ReadInputGraphs.cpp ../EGG/RegularGraph.cpp -o MIDAS-G_win.exe -lAlignment -lCatalogs -lChemReactions -lChemTransforms -lChemicalFeatures -lDataStructs -lDepictor -lDescriptors -lDistGeomHelpers -lDistGeometry -lEigenSolvers -lFMCS -lFileParsers -lFingerprints -lForceField -lForceFieldHelpers -lFragCatalog -lGraphMol -lhc -lInchi -lInfoTheory -lMolAlign -lMolCatalog -lMolChemicalFeatures -lMolDraw2D -lMolHash -lMolTransforms -lOptimizer -lPartialCharges -lRDGeneral -lRDGeometryLib -lRDInchiLib -lSLNParse -lShapeHelpers -lSimDivPickers -lSmilesParse -lSubgraphs -
```

```
lSubstructMatch -lAlignment -lCatalogs -lChemReactions -lChemTransforms -
lChemicalFeatures -lDataStructs -lDepictor -lDescriptors -lDistGeomHelpers -
lDistGeometry -lEigenSolvers -lFMCS -lFileParsers -lFingerprints -lForceField -
lForceFieldHelpers -lFragCatalog -lGraphMol -lhc -lInchi -lInfoTheory -lMolAlign -
lMolCatalog -lMolChemicalFeatures -lMolDraw2D -lMolHash -lMolTransforms -
lOptimizer -lPartialCharges -lRDGeneral -lRDGeometryLib -lRDInchiLib -lSLNParse -
lShapeHelpers -lSimDivPickers -lSmilesParse -lSubgraphs -lboost_filesystem-mt -
lboost_system-mt
```

If you want to build a static version, please use the following command.

```
g++ -static -std=c++11 -O2 -fopenmp -I../EGG -I/mingw64/include -
I${RDBASE}/External -I${RDBASE}/Code -L/mingw64/lib -L${RDBASE}/lib
main.cpp Database.cpp MidasCore.cpp ParseConfigure.cpp SettingInfo.cpp
SpectrumUnit.cpp
ParseSpectraFile.cpp ../EGG/EggCore.cpp ../EGG/FilesOperation.cpp ../EGG/GML.cpp .
./EGG/GraphBasics.cpp ../EGG/GraphGrammar.cpp ../EGG/ParseRegularGraph.cpp ../E
GG/Production.cpp ../EGG/ReadGML.cpp ../EGG/ReadGrammar.cpp ../EGG/ReadInput
Graphs.cpp ../EGG/RegularGraph.cpp -o MIDAS_static_win.exe -lAlignment_static -
lCatalogs_static -lChemReactions_static -lChemTransforms_static -
lChemicalFeatures_static -lDataStructs_static -lDepictor_static -lDescriptors_static -
lDistGeomHelpers_static -lDistGeometry_static -lEigenSolvers_static -lFMCS_static -
lFileParsers_static -lFingerprints_static -lForceField_static -lForceFieldHelpers_static -
lFragCatalog_static -lGraphMol_static -lhc_static -lInchi_static -lInfoTheory_static -
lMolAlign_static -lMolCatalog_static -lMolChemicalFeatures_static -
lMolDraw2D_static -lMolHash_static -lMolTransforms_static -lOptimizer_static -
lPartialCharges_static -lRDGeneral_static -lRDGeometryLib_static -lRDInchiLib_static -
lSLNParse_static -lShapeHelpers_static -lSimDivPickers_static -lSmilesParse_static -
lSubgraphs_static -lSubstructMatch_static -lAlignment_static -lCatalogs_static -
lChemReactions_static -lChemTransforms_static -lChemicalFeatures_static -
lDataStructs_static -lDepictor_static -lDescriptors_static -lDistGeomHelpers_static -
lDistGeometry_static -lEigenSolvers_static -lFMCS_static -lFileParsers_static -
lFingerprints_static -lForceField_static -lForceFieldHelpers_static -lFragCatalog_static -
lGraphMol_static -lhc_static -lInchi_static -lInfoTheory_static -lMolAlign_static -
lMolCatalog_static -lMolChemicalFeatures_static -lMolDraw2D_static -lMolHash_static
-lMolTransforms_static -lOptimizer_static -lPartialCharges_static -lRDGeneral_static -
lRDGeometryLib_static -lRDInchiLib_static -lSLNParse_static -lShapeHelpers_static -
lSimDivPickers_static -lSmilesParse_static -lSubgraphs_static -lSubstructMatch_static
/mingw64/lib/libboost_filesystem-mt.a /mingw64/lib/libboost_system-mt.a
/mingw64/lib/libboost_thread-mt.a
```