NETWORK MOTIF DETECTION TOOL

mfinder Tool Guide

Table of Contents

CHAPTER 1

CHAPTER 2

CHAPTER 3

CHAPTER 4

Chapter

1

1 General information

Description: mfinder is a software tool for network motifs detection.

Copyrights: Nadav Kashtan, Shalev Itzkovitz, Ron Milo, Uri Alon 2002-2005.

Version info: version: 1.2
(Date: May 2005)
Availability: Application and source code are available at:
http://www.weizmann.ac.il/mcb/UriAlon/

Platforms: A PC running Windows 2000, Windows XP. Version 1.1 is available also for linux OS. For large and dense networks (more than 10000 nodes) it is recommended to run mfinder on a computer with at least 512 Mbyte RAM**.**

1.1 Current version major updates

Version 1.2 (Date :May 2005)

1. **Full compatibility with mDraw – our new network visualization tool. Free download at:**
   http://www.weizmann.ac.il/mcb/UriAlon/
   mDraw has an easy to use interface to mfinder. The results of mfinder can be visualized by mDraw. In addition mDraw employs dedicated algorithms for network layout to display the motifs embedded in the network.

2. Additional network randomization methods such as the stubs method, "go with the winners method", conserve clustering series and more.
3. Additional flags to manipulate mfinder modes of operation and output files.
4. Members file – the subgraphs members are sorted according to their order in the adjacency matrix that represent the subgraph ID.

Version 1.1 (Date Aug 2003)

1. Detection of larger motifs (more than 4 nodes).
2. Output of all subgraphs members
   (that is the names/indexes participating in each subgraph)
3. Sampling algorithm implementation which is useful
   for very large networks or large subgraph size.
4. Output the top 7 motifs in cases there are many motifs found
   (Complete motif list can be found in the output file)

Chapter

2

2 How to use mfinder?

2.1 Download and installation

Follow these instructions:

a) Download

Download the following files: mfinder.exe, network_exmp.txt, output_exmp.txt. (All packed in **mfinder.zip**)

b) Perform a test run

1. Open a DOS window
   (This can be done by :
   Start->Program->accessories->Command Prompt)
2. Change to the directory which you downloaded the mfinder.exe file and the other two files.
3. Run the following command:
   mfinder network_exmp.txt
4. Now you should see an output on the screen that the software started running.
5. A successful run should end with the following line:
   "Output file network_exmp__OUT.txt was generated".
6. Check that you have an output file
   'network_exmp_OUT.txt' (in the same directory).

2.2 Command line options

Once you managed to execute a simple mfinder run, you may want to use the command line options. If you don't use the command line options then the parameters are set to their default values as mentioned below.

The general format is:

```
mfinder  <input network file name> [-s <motif size>]
         [-r <num of random networks>] [-f <output file name>]
         [more flags]
```

Note : < > - means should be replaced by the relevant input.
[ ] - means optional input.

Simple command line example:

```
mfinder network_exmp.txt -s 4 -r 100 -f exmp_mtf_sz_4
```

2.3 Basic Command line options

```
-s <subgraph size> : Motif size to detect (currently can be 2 to 6).
                    Default: 3

-r <no. of random networks> : Number of random networks to generate.
                    Default: 100

-f <Output file name> : Output file name. A suffix "_OUT.txt" is added.
                    Default: <input network file name>_OUT.txt

-nd : Input network is non-directed.
                            (See input file format for this case)
                    Default: Directed.
-p <no. of samples> : Use sampling method to detect motifs.

-omem : Output a file which contains a list of all members of all the
          subgraphs. An output file name with a suffix "_MEMBERS.txt" is
          created.

-h :  help.
```

2.4 Input file format

Input network file format should be a simple ".txt" format. Nodes in the network should be represented by integers. Each edge in the network should be represented by an equivalent line of the following format:

<source node> <target node> <edge weight>.

Example:

```
1 2 1
3 1 1
```

represents a network of 3 nodes with two edges:
(1->2) and (3->1)

- In the current version <edge weight> is ignored and should be 1 for all edges.

- The number of nodes in the network is considered as the highest integer (that represents a node) to appear in the file.

  You may look in the example file :network_exmp.txt
  to see how your network input file should look like.

Non-directed networks:

If the network is non-directed, then every edge should appear only once (means represented by only one line and not two). The order of target and source has no meaning in this case. (Pay attention: DO NOT represent a non-directed edge by the two equivalent directed edges). Remember to use the "-nd" flag when running mfinder with non-directed networks.

2.5 Additional command line options

Motif criteria flags:

-z <value> : Z-score threshold to use when calculating motifs (default z=2).
-u : Uniqueness threshold (default u=4).
-nu : Don't count uniqueness and ignore uniqueness threshold.
-m <value> : mfactor threshold to use when calculating motifs (default m=1.1)

Random networks randomization flags:

-rs : use stubs method for generating random networks
-rclust : Preserve clustering sequence in random networks
-met :Use Metropolis algorithm to conserve triad-census
    in random networks
    (for s>3; Default : Do not use Metropolis)
-t0<temp>:Initial temperature (-met option, default 0.001)
-iter <num> :controls how many steps to perform (-met option, default 2)
-eth <th> : energy threshold (-met option, default 0.005)
-rgrass <colony size>: use grassberger algorithm ("go with the winners") for generating random networks
-rgrass_max_sz <max ratio>: Limit maximal colony size ratio
-rdm: don't conserve mutual edges in random networks.
-rcl <layers num><size1 size2 ..sizem>: conserve layers in random
    networks
-nsr <value>: Global Switches number when generating Random networks. That means the range of number of switches to perform x100,x200 etc (default x100).

Output files flags:

-oi : output intermediate output files (Statistics results are calculated and dumped after each random network generation). Default :No:
-ospmem <subgraph id>: output members list of a specific subgraphs only
-maxmem <list length>: limit length of members list to 'list length'.
Default: 1000. The actual number of members in the' __MEMBERS.txt' file may be larger than this value as effect of isomorphism between subgraphs.
-omat : output matrix file ('__MAT.txt') with statistics of all subgraphs. This is a file in text format, columns are identical to the "__OUT.txt" file. This file can be easily loaded to MATLAB as a matrix.
-omet : output metropolis log
-olog : output general log file
-orall : output matrix file ('__MAT_R.txt') results with subgraph appearances in every one of the random networks.
-ornet : output random networks files in 'mfinder input network format'. This is very useful if you want to further analyze the random networks.
-oclust : output clustering sequences of real and random nets
-otop <no. of top motifs> : No. of top motifs to show
-onodangl: output a list of all non-dangling detected motifs ('non-dangling' means that there are no nodes with only a single incoming or outgoing edge)

Other flags

-ts : <target,source,weight> Old format of input network file
-q : Quiet mode - No output to the screen
-dd : Don't die mode. Wait to user action before terminating the program
-pold <num of samples>: run sampling method old version
-nor : Dont consider Real network. This mode is useful if you only want to generate random networks. Default :No:
-cr : calculate roles statistics. Implemented only for 3-node subgraphs. Roles are defined as in (Kashtan, N., et al., *Topological generalizations of network motifs.* Phys Rev E, 2004. **70**(3 Pt 1): p. 031909.)

Chapter

3

3 Algorithms and comments

3.1 Network motifs detection Algorithms

In order to detect network motifs we mfinder implements two methods:

a) Full enumeration of subgraphs. The algorithm is described in the SOM of Milo, R., et al., *Network Motifs: Simple Building Blocks of Complex Networks.* . Science, 2002. **298**: p. 824-827.

In order to reduce the algorithm run time mfinder implements a semi-dynamic programming algorithm. To count all n-node we begin with an edge e1 and search for all the n-node subgraphs it participates in. We store in a hash table (actually array of hash tables) all the sets of nodes that were already visited. This saves the time of searching, as we stop the searching tree if we already visited a set (or subset) of nodes. When we finish this process for edge e1 we clear the hash tables, and proceed with the next edge in the network, e2. This process is repeated for all network edges, and we accumulate the counts for each subgraph type. At the end we need to divide each subgraph counts by the number of edges the subgraph contains (remember that we cleared the hash tables every search from a different start edge).

b) A Sampling of subgraphs for estimation of subgraph concentrations. This algorithm is described at Kashtan, N., et al., *Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs*. Bioinformatics, 2004. 20(11): p. 1746-58.

3.2 Network randomization methods

The mfinder implements several methods to generate random networks.

a) The switching method –we switch between edges while keeping the number of incoming edges, outgoing edges and mutual edges of each node of the input network. This is the default method mfinder employs. The number of switches is a random number in the range of 100-200 times the number of edges in the network.

b) The stubs method.

c) "Go with the winners" algorithm.

For a review of the three methods above see R. Milo et al, *Uniform generation of random graphs with arbitrary degree sequences,* cond-mat/0312028 (2003).

Additional randomization options (using the switching method):

- Preserve the number of all Triads types (flag '-met')
- Preserve clustering coefficient of all nodes (flag '–rclust')
- Preserve node layers in a feed-forward network (flag '-rcl', see section 2.)
- Do not preserve mutual edges (flag '-rdm').

3.3 Comments

1) If the number of random networks to generate is smaller than 1000, then the criteria for motifs is Zscore>2 and Pvalue is ignored (The number of random networks is not large enough to consider Pvalue).

2) The uniqueness value is the number of times a subgraph appears in the network with completely disjoint groups of nodes. The value in the output is not the maximum number of disjoint groups but is a lower bound for the real number. The default requirement for motifs is U>=4 unless the flags '-u' or '-nu' are used.

3) In the default mode random networks preserve the degree distribution of the nodes (in-degree, out-degree and mutual degree are preserved for each node). When detecting motifs of size >=4 it is possible to generate random networks ensemble that preserves the triad census in the real network (the numbers of all 13 connected 3-node triads) by using the '-met' flag.
4) Sampling mode is recommended when:
    i. The input network is very large (e.g. #edges > 100000) or when analyzing subgraphs of size >=5.
    ii. You want a fast approximate motif analysis of the network.

   In this mode we sample subgraphs in the network and calculate their concentrations (instead of appearances) and compare these numbers with random network results.

   (Concentration of a subgraph is its number of appearances divided by the total number of subgraphs of the same size in the network.)

You will need to insert the number of samples you wish to perform. This number is hard to estimate before trying to run the mfinder. As a rule of thumb the numbers listed in the following table should be OK:
(Of course the sufficient number of samples for good estimation depends on the network and on the minimal concentration of subgraphs you are interested in).

subgraph size	**No. of samples**
3	5000+
4	10000+
5	50000+
6	100000+

A good reference for the quality of the results of a specific subgraph, is the number of hits (more than 10 hits are usually enough).

5) Automatic ranking of top motifs list is shown when there are more than 10 motifs found. (This is very useful with motifs of size>=5).The motifs are sorted by a score which is a function of:
    a. the concentration.
    b. the Zscore.
    c. the existence of dangling edges
       (an edge touching an isolated node in the subgraph)

   You may have interest in motifs which are not in the top list, all the motifs are listed in the output file.
   Note that motifs with dangling edges such as single input modules and chains will not be shown on the top motifs list.

6) Motif id represents the adjacency matrix of the subgraph as a long binary integer extracted by concatenation of the rows of the matrix. For example: feedforward-loop adjacency matrix is:

   0 1 1
   0 0 1
   0 0 0

   Hence the feedforward-loop id is 38 (binary 011001000, with least significant bit on the left)

7) When running mfinder for searching motifs of size >=5, the full subgraph list contains only the subgraphs that were found in the real network. (In order to avoid the long list of all possible subgraphs).

8) Members output file: in case of many occurrences of a subgraph, only a partial list is displayed).

Known Bugs/To be fixed

a. No known memory leaks.

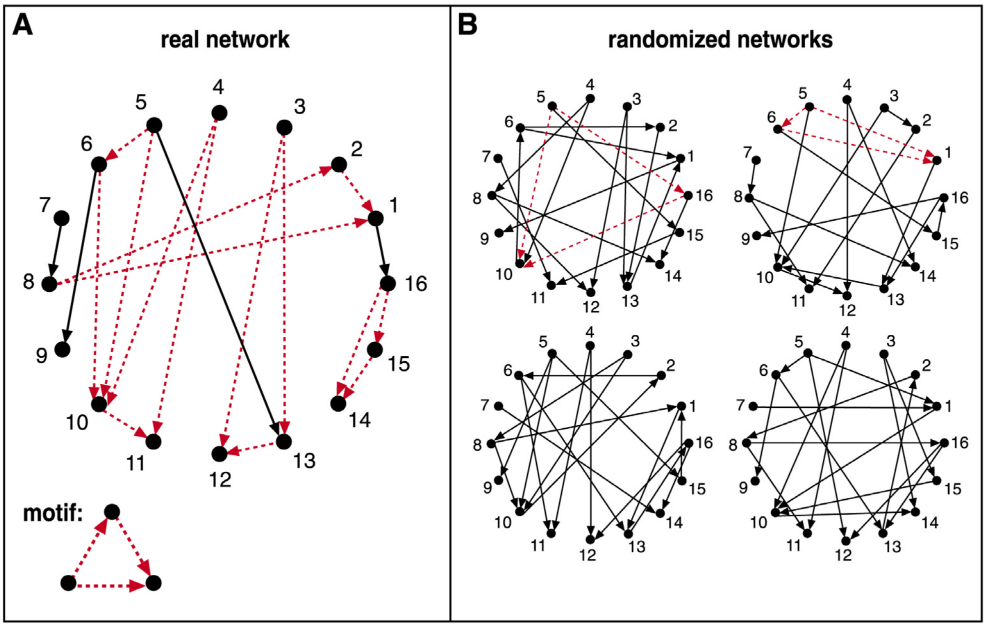
b. There are rare cases of overflows that causes problems with hash function when the mfinder is run with subgraph size=6 (This bug still not fixed - TBD).

Chapter

4

4 Example network and outputs

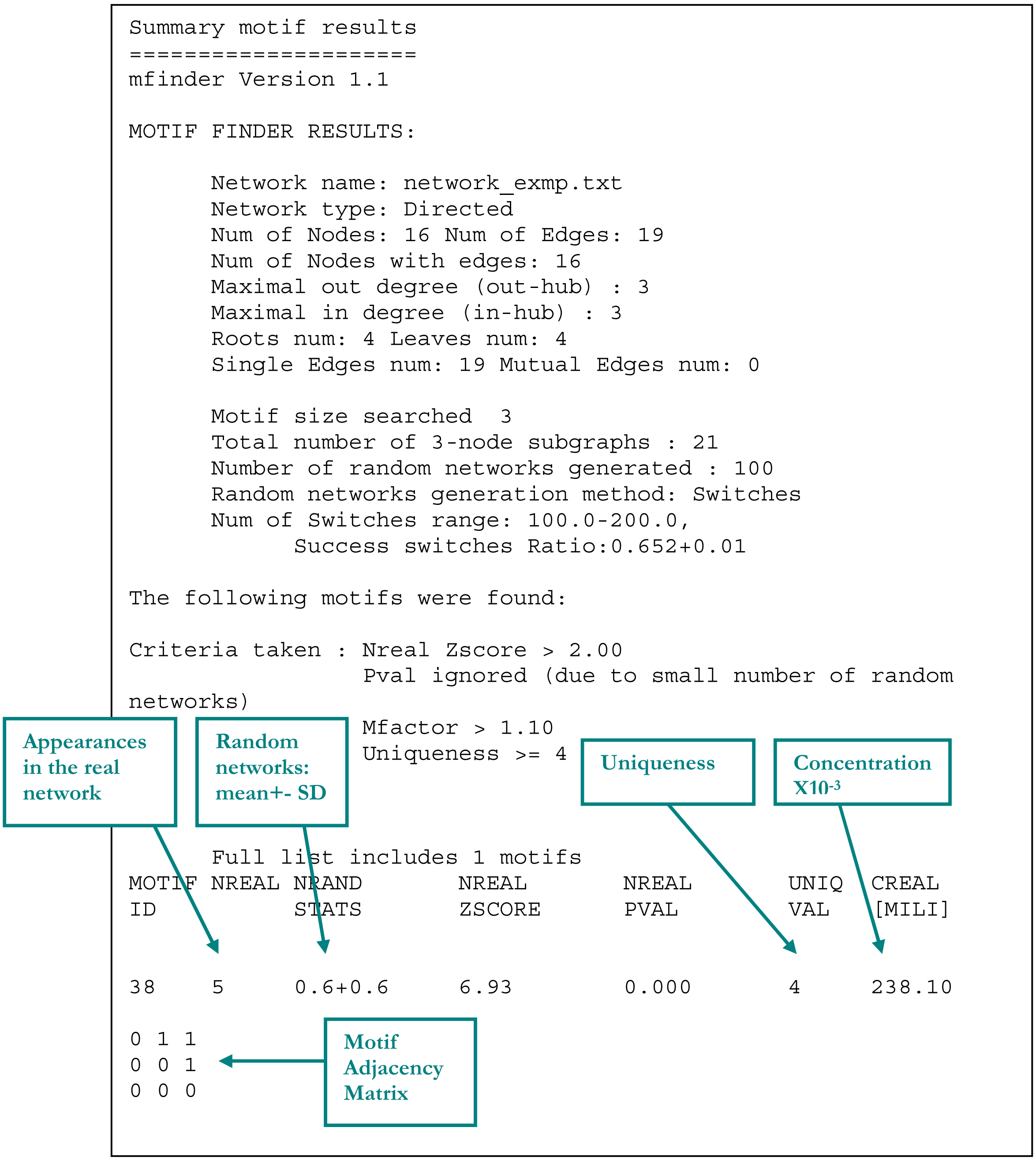
Consider the following directed network of 16 nodes and 19 edges.



The input file for mfinder has a row for each edge in the format <source, target, weight>
(Weight should be ignored and equals one for all edges):

```
1   16  1
2   1   1
3   12  1
3   13  1
4   10  1
4   11  1
5   6   1
5   10  1
5   13  1
6   9   1
6   10  1
7   8   1
8   1   1
8   2   1
10  11  1
13  12  1
15  14  1
16  14  1
16  15  1
```

Running the mfinder for detection of 3-node motifs reveals only a single 3-node motif which is the feedforward loop. The statistical significance of all 13 3-node connected subgraphs is presented as well.
We used the following command line :
 'mfinder network_exampl.txt –s 3 –r 100'
 to yield the following output file:



```
Summary motif results
=====================
mfinder Version 1.1

MOTIF FINDER RESULTS:

      Network name: network_exmp.txt
      Network type: Directed
      Num of Nodes: 16 Num of Edges: 19
      Num of Nodes with edges: 16
      Maximal out degree (out-hub) : 3
      Maximal in degree (in-hub) : 3
      Roots num: 4 Leaves num: 4
      Single Edges num: 19 Mutual Edges num: 0

      Motif size searched  3
      Total number of 3-node subgraphs : 21
      Number of random networks generated : 100
      Random networks generation method: Switches
      Num of Switches range: 100.0-200.0,
            Success switches Ratio:0.652+0.01

The following motifs were found:

Criteria taken : Nreal Zscore > 2.00
                 Pval ignored (due to small number of random
networks)
                 Mfactor > 1.10
                 Uniqueness >= 4

      Full list includes 1 motifs
MOTIF NREAL NRAND       NREAL       NREAL       UNIQ  CREAL
ID          STATS       ZSCORE      PVAL        VAL   [MILI]


38    5     0.6+0.6     6.93        0.000       4     238.10

0 1 1
0 0 1
0 0 0
```

```
Full list of subgraphs size 3 ids:

      ( Total num of different subgraphs size 3 is : 13 )

MOTIF NREAL NRAND        NREAL        NREAL        CREAL    UNIQ
ID          STATS        ZSCORE       PVAL         [MILI]
6     3     7.4+0.6      -6.93        1.000        142.86      2

12    10    13.9+1.1     -3.44        1.000        476.19      2

14    0     0.0+0.0      888888       0.000        0.00        0

36    3     7.4+0.6      -6.93        1.000        142.86      1

38    5     0.6+0.6      6.93         0.000        238.10      4

46    0     0.0+0.0      888888       0.000        0.00        0

74    0     0.0+0.0      888888       0.000        0.00        0

78    0     0.0+0.0      888888       0.000        0.00        0

98    0     0.2+0.4      -0.43        0.160        0.00        0

102   0     0.0+0.0      888888       0.000        0.00        0

108   0     0.0+0.0      888888       0.000        0.00        0

110   0     0.0+0.0      888888       0.000        0.00        0

238   0     0.0+0.0      888888       0.000        0.00        0


 (Application total runtime was:    1.0 seconds.)

 (Real network processing runtime was:    0.0 seconds.)

 (Single Random network processing runtime was:    0.0 seconds.)
```