Clique and Coloring Problems
Graph Format

Last revision: May 08, 1993

This paper outlines a suggested graph format. If you have comments on this or other formats or you have information you think should be included, please send a note to `challenge@dimacs.rutgers.edu`.

1 Introduction

One purpose of the DIMACS Challenge is to ease the effort required to test and compare algorithms and heuristics by providing a common testbed of instances and analysis tools. To facilitate this effort, a standard format must be chosen for the problems addressed. This document outlines a format for graphs that is suitable for those looking at graph coloring and finding cliques in graphs. This format is a flexible format suitable for many types of graph and network problems. This format was also the format chosen for the First Computational Challenge on network flows and matchings.

This document describes three problems: unweighted clique, weighted clique, and graph coloring. A separate format is used for satisfiability.

2 File Formats for Graph Problems

This section describes a standard file format for graph inputs and outputs. There is no requirement that participants follow these specifications; however, compatible implementations will be able to make full use of DIMACS support tools. (Some tools assume that output is appended to input in a single file.)

Participants are welcome to develop translation programs to convert instances to and from more convenient, or more compact, representations; the Unix **awk** facility is recommended as especially suitable for this task.

All files contain ASCII characters. Input and output files contain several types of *lines*, described below. A line is terminated with an end-of-line character. Fields in each line are separated by at least one blank space. Each line begins with a one-character designator to identify the line type.

2.1 Input Files

An input file contains all the information about a graph needed to define either a clique problem or a coloring problem. Some information may be included that is not relevant to one problem (for instance, node weights are not needed for coloring problem) so that information may be ignored.

In this format, nodes are numbered from 1 up to n. There are m edges in the graph.

Files are assumed to be well-formed and internally consistent: node identifier values are valid, nodes are defined uniquely, exactly m edges are defined, and so forth. A input checker will be made available to ensure compatibility with this standard.

- **Comments.** Comment lines give human-readable information about the file and are ignored by programs. Comment lines can appear anywhere in the file. Each comment line begins with a lower-case character **c**.

  ```
  c This is an example of a comment line.
  ```

- **Problem line.** There is one problem line per input file. The problem line must appear before any node or arc descriptor lines. For network instances, the problem line has the following format.

  ```
  p FORMAT NODES EDGES
  ```

  The lower-case character `p` signifies that this is the problem line. The `FORMAT` field is for consistency with the previous Challenge, and should contain the word "edge". The `NODES` field contains an integer value specifying n, the number of nodes in the graph. The `EDGES` field contains an integer value specifying m, the number of edges in the graph.

- **Node Descriptors.** For this Challenge, a node descriptor is required only for the weighted clique problem. These lines will give the weight assigned to a node in the clique. There is one node descriptor line for each node, with the following format. Nodes without a descriptor will take on a default value of 1.

  ```
  n ID VALUE
  ```

  The lower-case character `n` signifies that this is a node descriptor line. The `ID` field gives a node identification number, an integer between 1 and n. The `VALUE` gives the objective value for having this node in the clique. This value is assumed to be integer and can be either positive or negative (or zero).

- **Edge Descriptors.** There is one edge descriptor line for each edge the graph, each with the following format. Each edge (v, w) appears exactly once in the input file and is not repeated as (w, v).

  ```
  e W  V
  ```

  The lower-case character `e` signifies that this is an edge descriptor line. For an edge (w, v) the fields `W` and `V` specify its endpoints.

- **Optional Descriptors.** In addition to the required information, there can be additional pieces of information about a graph. This will typically define the parameters used to generate the graph or otherwise define generator–specific information. The following list may be added to as interesting problem generators are decided on:

  - Geometric Descriptors. One common method to generate or display graphs is to have the nodes be embedded in some space and to have the edges be included according to some function of the distance between nodes according to some metric. The node information can be defined by a dimension descriptor and a vertex embedding descriptor.

    ```
    d DIM METRIC
    ```

is the dimension descriptor. DIM is an integer giving the number of dimensions of the space, while METRIC is a string representing the metric for the space. METRIC is a string that can take a number of forms. Lp (i.e. L1, L2, L122, and so on) denotes the ℓ_p norm where the distance between two nodes embedded at $(x_1, x_2, \ldots, x_d)$ and $(y_1, y_2, \ldots y_d)$ is $(\sum_{i=1}^{d} |x_i - y_i|^p)^{1/p}$. The string LINF is used to denote the ℓ_∞ norm. L2S denotes the squared euclidean norm (which can be less susceptible to computer–differences in round–off and accuracy issues).

```
v  X1  X2  X3  . .. XD
```

The lower-case character `v` signifies that this is a vertex embedding descriptor line. The fields `X1, X2 . . .XD` give the d coordinate values for the vertex. Note that these lines must appear after the `d` descriptor.

– Parameter Descriptors. The parameter descriptors are used to give other information about how the graph was generated. The lines are generator–specific, and as such it is not expected that most codes will use most (or any) of them. They are included only to aid those codes specifically designed to attack specially structured problems. The general form of the parameter descriptor is:

```
x PARAM VALUE
```

The lower-case character `x` signifies that this is a parameter descriptor line. The PARAM field is a string that gives the name of the parameter, while the VALUE field is a numeric value that gives the corresponding value. The following PARAM values have been defined:

PARAM	Description
MINLENGTH	(Geometric Graphs) Edge included only if length greater than or equal to VALUE
MAXLENGTH	(Geometric Graphs) Edge included only if length less than or equal to VALUE

Note that this information is in addition to the required edge descriptors.

2.2 Output Files

Every algorithm or heuristic should create an output file. This output file should consist of one or more of the following lines, depending on the type of algorithm and problem being solved.

- **Solution Line**

  ```
  s TYPE SOLUTION
  ```

  The lower–case character `s` signifies that this is a solution line. The `TYPE` field denotes the type of solution contained in the file. This should be one of the following strings: "col" denotes a graph coloring, "clq" denotes a maximum weighted clique, and "cqu" denotes a maximum unweighted clique (one that has ignored the `n` descriptor lines).

  The `SOLUTION` field contains an integer corresponding to the solution value. This is the clique size for unweighted clique, clique value for weighted clique, or number of colors used for graph coloring.

- **Bound Line**

  ```
  b BOUND
  ```

  The lower–case character `b` signifies that this is a bound on the the solution. The `BOUND` field contains an integer value that gives a bound on the solution value. This bound is an upper bound on the maximum clique value for cliques and weighted clique and a lower bound on the number of colors needed for coloring the graph.

- **Clique Line**

  ```
  v V
  ```

  The lower-case character `v` signifies that this is a clique vertex line. The `V` field gives the node number for the node in the clique. There will be one clique line for each node in the clique.

- **Label Line**

```
l V N
```

The lower-case character `l` signifies that this is a label line, generally used for graph coloring. The `V` field gives the node number for the node in the clique while the `N` field gives the corresponding label. There will be one label line for each node in the graph.