neg2adj

Copyright (c) 1999 William D. Richards
Simon Fraser University
Burnaby, British Columbia, Canada

A utility program for converting NEGOPY-style files, in which there is one link per line of data, into adjacency matrices.

Sometimes it is necessary to convert data that has been organized as NEGOPY-style files, in which there is one link per line of data, into adjacency matrices. The neg2adj program was written to make this task easy for you.

Here are two examples of adjacency matrix formats you might desire:

A. You might want a pure adjacency matrix -- no ID numbers, no format specification. This is the simplest form of adjacency matrix format, since it contains only the matrix itself. The first line of data in the file contains the first row of the matrix. The second line contains the second row of the matrix, etc.

```
0 1 1 0 1 0 0 0 1 0 0 0 0 1 1
0 0 1 0 1 0 1 1 1 0 0 1 0 0 0
1 0 0 0 1 0 0 0 0 0 1 0 0 0 1
1 1 0 0 0 1 1 1 0 0 0 0 0 1 0
0 1 0 1 0 1 0 1 1 0 1 0 0 1 0
           ⋮
           ⋮
```

B. You may wish to prefix each row of the adjacency matrix with an ID number. Once again, the first line of data in the file contains the first row of the matrix, but the first number in each row must be treated differently from all the other numbers.

```
1 0 1 1 0 1 0 0 0 1 0 0 0 0 1 1
2 0 0 1 0 1 0 1 1 1 0 0 1 0 0 0
3 1 0 0 0 1 0 0 0 0 0 1 0 0 0 1
4 1 1 0 0 0 1 1 1 0 0 0 0 0 1 0
5 0 1 0 1 0 1 0 1 1 0 1 0 0 1 0
             :
             :
```

There may be different formats of the data you wish to convert:

A. Some data files contain only the data.

Here is an example:

```
1   1 1
1   2 0
1   3 1
1   5 1
1   6 0
    :
    :
```

B. Some data files begin with a format specification that describes the data. This file is similar to the one described above, but it has a line containing a FORTRAN-style format specification for the data preceding the actual matrix itself.

This example begins with a format specification:

```
(2i4,i2)
   1   1 1
   1   2 0
   1   3 1
   1   5 1
   1   6 0
       :
       :
```

C. Some files contain additional information before the format specification or before the data.

This example has the network size before the format specification:

```
20
(2i4,i2)
   1   1 1
   1   2 0
   1   3 1
   1   5 1
   1   6 0
       :
       :
```

D. Some files separate numbers with commas or tabs instead of blank spaces.

This example shows some numbers separated by spaces, some by commas, and some by tabs:

```
1    2    1
1    4    1
1,8 1
2 1 1
3,4,1
   :
   :
```

The program can handle all of the above situations, as long as there are spaces, commas, or tabs separating the numbers in each line of the data file from one another.

When the program begins, it asks for the name of the data file you want to convert and the name you want to use for the new file.

```
Enter name of data file:
```

If it cannot find that file, you will see this message:

```
File not found: ___________
```

The program then asks:

```
Enter name of output file:
```

After you enter a name for this file, you will see this message:

```
File will be created: ___________
```

If there is already a file with this name, the program ill ask:

```
The present contents of ___________ will be lost. Okay? (Y or N):
```

The program then draws a ruler on the screen, followed immediately by the first four lines of data in the file. This allows you to see what is in the file and helps in answering the following questions.

```
         1         2         3         4
1234567890123456789012345678901234567890
----------------------------------------
(2i4,i2)                                 <- First four lines
   1   1 1                               <- of your data file
   1   2 0                               <-
   1   4 1                               <-
       :
       :
```

1. The program then asks how many nodes are there in the network and how they are numbered:

```
How many nodes in the matrix? (5-2000)

Do ID numbers go from 1 to ____? (y/n)
```

If your anser is ‘n’, you will see this message:

```
Sorry. This program requires ID numbers go from 1 to n.
```

2. Next, the program asks if you want it to skip some lines at the beginning of the file. Answer "y" if there is anything on the lines before the actual data, including a format specification.

```
Should lines at the beginning of your file be skipped? (y/n)
```

If you answer ‘y’ to this question, the program will ask how many lines you want to skip

```
How many lines do you want to skip?  (0-200)
```

3. The program asks if you want rows in the output matrix to begin with ID numbers.

```
Do you want rows in the output matrix to begin with ID numbers?
(y/n)
```

4. Now the program attempts to determine where on the lines of data is the ID number of the sender. It asks:

```
Is the 1st number on each line an ID number of a sender? (y/n)
```

If you answer ‘n’, the program asks:

```
Is the 2nd number on each line an ID number of a sender? (y/n)
```

If the sender’s ID number is not one of the first four numbers on each line, you will see this message:

```
The ID number of the sender must be one of the first four numbers
on each line.
```

5. Now the program attempts to determine where on the lines of data is the ID number of the receiver. If the first number is the ID number of the sender, the program asks:

```
Is the 2nd number on each line an ID number of a receiver? (y/n)
```

If you answer ‘n’, the program asks:

```
Is the 3rd number on each line an ID number of a receiver? (y/n)
```

If the receiver’s ID number is not one of the first four numbers after the sender's ID number, you will see this message:

```
The ID number of the receiver must be one of the first four
numbers after the sender’s ID.
```

6. Now the program has some questions about the type of data you have:

```
Do you have a binary network? (y/n)
```

If you answer is ‘y’, the program asks:

```
Do you have a number on each line of data that tells whether the
link is present or absent? (y/n)
```

If your answer to this question is ‘n’, the program displays this message:

```
Then all links that exist appear in your data and any links that
do not appear in your data do not exist.
```

If your answer to the question that asks “Do you have a binary network?” is ‘n’, the program displays this message, followed by a few questions:

```
Then you must have a scalar network, and there must be a number
on each line of data that tells how strong the link is.
```

If the first two numbers on each line of data are the ID numbers of the sender and receiver, the program asks:

```
Is that number the 3rd number on each line? (y/n)
```

If you answer ‘n’, the program asks:

```
Is that number the 4th number on each line? (y/n)
```

If that number is not one of the first six numbers after the receiver's ID, the program displays this message:

```
The number that describes the link must be one of the first six
numbers after the receiver’s ID.
```

At this point the program will have all the information it needs. It will convert the data for you.

The program can handle networks with up to 2,000 nodes and 60,000 links. If there are more than 60,000 links in your data, the program displays this message:

```
Your network has too many links for this program. The maximum is
60,000.
```