Approximation Algorithms – COMP 670J
Fall Semester 2001 – Class Project I/O Info

This file describes the IO specification for the COMP670J projects. To start with, every project program should work in two modes. In the first mode it should return only *the cost of the solution*. For example, in the MAXSAT project it should report how many clauses are satisfied. In the second mode it should report the cost of the solution *followed by the problem specification and the actual solution*, e.g. for MAXSAT it should report all of the clauses followed by the truth assignment.

You may implement this two mode method in one of two ways. The way that I prefer is that you simply allow calling your program with a parameter -c. that is, if I run "MAXSATRR -c" it will return just the cost while if I run "MAXSATRR" it will return everything. (MAXSATRR being MAXSAT randomized rounding). In the second method you have two different programs. One program returns just the cost the other, everything.

The project specific specifications are given below. If you have any questions concerning them please send me email.

PROJECT I/O SPECIFICATIONS

1. *MaxSat*
   Each of the three algorithms should read data of the following form

```
m
n
L1
L2
.
.
.
Ln
```

   where m is the number of variables, n is the number of clauses and L_i will be the ith clause. The clauses should be written as the weight of the clause (a nonnegative real), a list of the variable indices that are non-negated followed by a 0 and then the list of the variable indices that are negated. For example. $x_3 \vee x_5 \vee \overline{x_2} \vee \overline{x_4}$ with weight 1, $\overline{x_1} \vee \overline{x_3}$ with weight 2.5 and x_3 with weight 2, would be input as

```
5
3
```

```
1.0  3 5 0 2 4
2.5  0 1 3
2.0  3 0
```

2. *Travelling Salesman*
   As mentioned you may assume that the input vertices will be points in the plane and that the weight of the edge connecting two points will be the physical distance between them. The I/O should look like

   ```
   n
   x1 y1
   x2 y2
   .
   .
   .
   xn yn
   ```

   where n is the number of points and $p_n = (x_n, y_n)$ is the nth input point.

3. *Minimum Multicut*
   Please come and talk with me to discuss the I/O specifications.

4. *Set-Cover*
   All of your set cover algorithms should take inputs of the form

   ```
   n
   m
   S1
   S2
   .
   .
   .
   SN
   ```

   where n is the number of elements, m the number of sets and S_i set i. A set will be represented by a list of its indices, i.e, $S_1 = \{1, 3, 5\}$, $S_2 = \{3\}$, $S_3 = \{2, 4\}$ will be represented by

   ```
   5
   3
   1 3 5
   3
   2 4
   ```

5. *Minimum Makespan*
   The input should be

```
m
J1
J2
J3
.
.
.
JN
```

   where m is the number of machines and J_i is the length of job i. For example

```
3
5
3
2
3
```

   Where 3 is the number of machines. You need to have two programs. The first, the exact algorithm takes this input and outputs the solution (in the two ways described at the beginning of this file). The second, the PTAS, needs one more parameter, the approximation ratio. So, besides reading input, your program should also read a parameter when it starts, specifying what the ratio is, i,e, it should be run as "MMPTAS 3" to give an approximation ratio of 3.

6. *Shortest Superstring*
   The input to this is a file with the strings

```
S1
S2
.
.
.
SN
```

   where S_i is string i, e.g.,

```
hello
longer
gerand
```

7. *FPTAS for Knapsack*
   The input should be of the form

```
B
I1
I2
.
.
.
In
```

   where B is the *knapsack capacity* and I_i is the ith item. Each item will be of the form $s_i \quad p_i$ where B, s_i, p_i are integers, s_i is the size of item i and p_i its profit, e.g.,

```
10
2 10
3 15
3 20
8 17
6 9
```

8. *Uncapacitated Facility Location*
   Please come and talk with me to discuss the I/O specifications.

9. *Sonet Rings*
   The input should be of the form

```
n
D1
D2
D3
.
.
.
DM
```

   where n is the number of nodes and Dk is the kth demand. A demand is written in the form

```
i j dij
```

   where i and j are nodes and $d_{i,j}$ is the demand between those two nodes.

10. *The List Accessing Problem*
    Please come and talk with me to discuss the I/O specifications.

11. *Approximation Algorithms for Clustering:*
    After speaking with Professor Mount please come and talk with me to discuss the I/O specifications.

12. *FPTAS for the Restricted Shortest path problem:*
    Please come and talk with me to discuss the I/O specifications.