Computational Intractability 2010/5/27

Lecture 7

Professor: David Avis *Scribe:Yosuke Suzuki*

1 Crew Scheduling Problem

We define a crew scheduling problem as follows.
Input: a directed graph, each node represents an airport, each edge represents a flight, and is labelled with the departure and arrival time.
Objective: assign crews to the flights and minimize a given cost function.

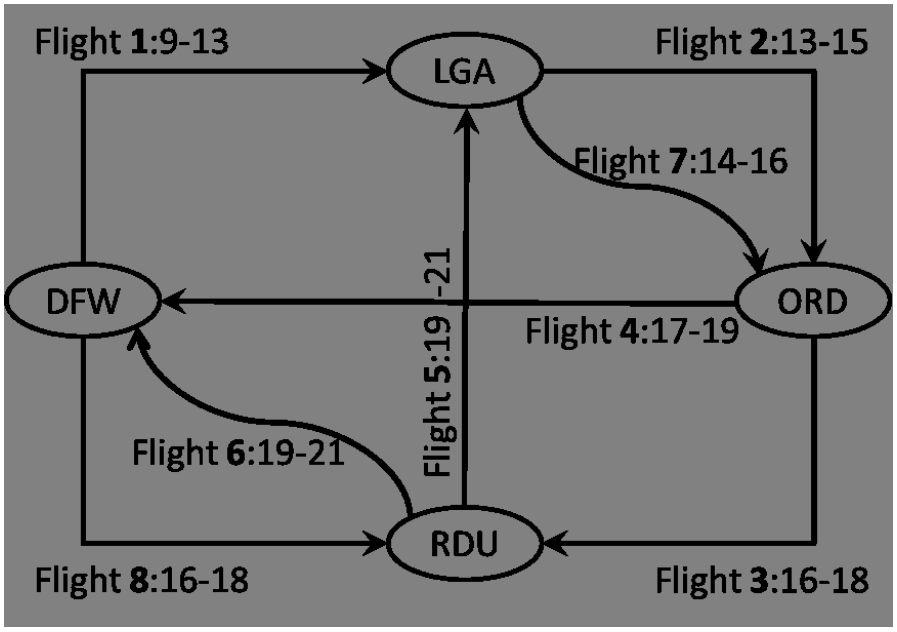

Figure 1: A flight graph

We define a *pairing* of flights as a series of flights which could be serviced by a single crew. This means that the departure time of a flight in the series must be no earlier than the arrival time of the previous flight in the series. In Figure 1 a possible pairing is

given by flights 1,7,3,5. The flights 3,5,2 are not a pairing as flight 5 arrives after flight 2 departs. We will assume that each crew must service at least two flights, so that each pairing contains at least two flights. We can assign multiple crews to one flight if necessary, in order to transport a crew to another airport.

The cost of an pairing is expressed as time interval between the first departure time and the last arrival time +5 hours.

2 Formulation of Flight Scheduling Problem as IP

We have n flights and assign m crews.

One possibility is to define decision variables $y_{ij}, 1 \leq i \leq m, 1 \leq j \leq n$, where:

$$y_{ij} = \begin{cases} 1 & \text{flight } j \text{ has a crew } i \\ 0 & \text{otherwise} \end{cases}$$

To cover flight j we introduce a constraint of the form:

$$\sum_{i=1}^{n} y_{ij} \geq 1$$

for each flight j. However it is not at all obvious how to link the flights together for any given crew.

An alternative method is to construct all pairings by preprocessing . To do this it is convenient to construct a flight connection graph showing which flights may follow which others in a pairing. The flight connection graph for Figure 1 is given in Figure 2.

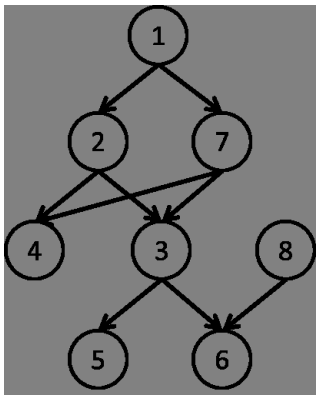

Figure 2: The flight connection graph for Figure 1

A flight connection graph has no cycles, and is constructed in time $O(n)$. Each pairing is a path in the graph of length at least two, and these paths can be enumerated in a straight forward way. For the example they are listed in Table 1.

A valid crew assignment is given by a set of pairings that cover all the flights. This can be formulated as a set covering problem. Define variables x_i for each pairing i that

No.	Pairing	Time
1	12	6
2	123	9
3	124	10
4	173	9
5	174	10
6	1735	12
7	1736	12
8	1235	12
9	1236	12
10	23	5
11	24	6

No.	Pairing	Time
12	73	4
13	74	6
14	735	7
15	736	7
16	235	8
17	236	8
18	86	5
19	35	5
20	36	5
21	17	7

Table 1: Pairings obtained from Figure 2

will satisfy:

$$x_i = \begin{cases} 1 & \text{pairing } i \text{ is used} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

For each flight we get a set cover inequality. In the example, we get the following inequalities:

$$\begin{aligned} x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 + x_9 &\geq 1 \text{ (for flight 1)} \\ x_1 + x_2 + x_3 + x_8 + x_9 + x_{10} + x_{11} + x_{16} + x_{17} &\geq 1 \text{ (for flight 2)} \\ &\ldots \\ x_{18} &\geq 1 \text{ (for flight 8)} \end{aligned} \tag{2}$$

Recall that the cost coefficients in the objective function are obtained by adding 5 to each of the times in Table 1.
The integer programming formulation is therefore:

$$minimize \quad z = 11x_1 + 14x_2 + 15x_3 + ... + 12x_{21} \tag{3}$$

subject to (1) and (2).

3 Solving an integer program

1. First we try solving the integer program as a linear program with the objective of minimizing the number of crews required. For this we replace (3) by

   $$minimize \quad z = x_1 + x_2 + x_3 + ... + x_{21}$$

   Using lp-solve for the linear program, we obtain $x_3 = x_6 = x_{18} = 1, z = 3$, and otherwise $x_j = 0$. Note this solution has a double-covering for flight 1.

   When we introduce the objective function (3), we obtain the solution $x_3 = x_{14} = x_{18} = 1, z = 37$. Since in each case the LP solution was integral, we were done.

2. Next we check if all vertices are integer valued with lrs. This is done by running the command:
   % lrs sched.ine sched.ext
   (A link to a directory containing all files is given on the course home page.) There are 80 vertices, of which 12 are fractional, and 55 extreme rays. Since some vertices are fractional, we will need to use integer programming for some objective functions. For example with objective:

$$\begin{aligned} minimize \;\; z &= x_1 + x_2 + x_3 + x_4 + 3x_5 + 3x_6 + 3x_7 + 2x_8 + 2x_9 + x_{10} + x_{11} \\ &+ x_{12} + x_{13} + 2x_{14} + 2x_{15} + x_{16} + x_{17} + x_{18} + x_{19} + x_{20} + x_{21} \end{aligned}$$

   we get the fractional LP solution $x_3 = x_4 = x_{13} = 1/2, x_{16} = x_{18} = 1, z = 3.5$. Using lp-solve with the variables defined to be integers, we get the integer optimal solution $x_3 = x_4 = x_{16} = x_{18} = 1, z = 4$.

3. We may also compute an ideal formulation by deleting fractional vertices, again by using lrs. The command is:
   % lrs isched.ext isched.ine
   We find that there are 251 inequalities in the ideal formulation, compared to 30 in the original formulation (including 21 non-negativity constraints).

3.1 Cutting Planes

Getting the ideal formulation is useful because we get new inequalities that have these properties.

- These are satisfied by every integer solution.
- Each new inequality always removes some fractional vertex or vertices of the original LP.

These inequalities are called cutting planes.

3.2 Solving any ILP by using LP + cutting plane

1. First solve the LP getting a fraction solution x^* (or if integer stop)
2. Find a cutting plane that removes x^* (i.e. violated by x^*)
3. Repeat from step 1 with new constraint added to LP

References

[1] Algorithms in the Real World, Carnegie Mellon University, Computer Science 15-853, http://www.cs.cmu.edu/afs/cs/project/pscico-guyb/294/class-notes/all/11.ps