Algorithm Miscellany

Jiyue Wang (jiyue@stanford.edu)

March 14, 2015

1 Job Scheduling / Makespan problem

Given m machines and n jobs with workload $p_1, ..., p_n$, give a schedule that

$$\min_{m \text{ machines}} \max\{\text{workload for a single machine}\}$$

This problem is NP hard. We can use a Greedy algorithm to achieve 4/3 approximation. If the number of distinct workload is restricted to k , there is a DP solution of $O(n^{2k})$ which gives the exact solution to the corresponding decision problem : Can the m machines finish the job within T times. (Suppose the workload is the time it takes to complete the job for one machine.)

Suppose there are b_i jobs for workload p_i, and we have $(b_1, ..., b_k)$ jobs in total. Let $M(c_1, ..., c_k)$ denote the minimum number of machines needed to complete $(c_1, ..., c_k)$ jobs within time T. Then it's easy to check whether $M(c_1, ..., c_k) > 1$ and quitely clearly, $M(0, ..., 0) = 0$

```
for c_1 in range(1, b_1 + 1):
  for c_2 in range(1, b_2 + 1):
    ...
    for c_k in range(1, b_k + 1):
      if M(c_1, ..., c_k) > 1:
        S = {(j_1, ..., j_k)| j_i < c_i for all i, M(j_1, ..., j_k) = 1}
        M(c_1, ..., c_k) = 1 + min(M(c_1 - j_1, ..., c_k - j_k) over S)
if M(b_1, ..., b_k) > m:
  return False
else:
  return True
```

2 (Minimum Weight) Perfect Matching/ Minimum Weight Cycle Cover

A **perfect matching** is a matching which matches all vertices of the graph. That is, every vertex of the graph is incident to exactly one edge of the matching. [1] The mini-weight perfect matching problem can be solved in polynomial time.

A **minimum weight cycle cover** of a graph is stated as follows. Let $H = (V, E)$ be a directed graph with non-negative arc weights given by $w : E \rightarrow R^+$. We wish to find a minimum weight collection of vertex-disjoint directed cycles in H such that every vertex is in exactly one of those cycles. [2]. We need this to give an approximation algo to the Asymmetric Traveling Salesman Problem (ATSP).

Algorithm: For each node $v \in V$, split it into v^+ and v^-, where v^+ is the 'in-node' and v^- is the 'out-node'. Solve the minimum perfect matching problem on the new graph. Done.

[1] Matching (graph theory), wikipediea

[2] HW0/6, Spring 2011, CS 598CSC, University of Illinois