USACO April 2006 Problem 'mqueue' Analysis

by Bruce merry

The problem statement is a bit misleading: a long queue of cows waiting to enter the second barn is actually a good thing, because it reduces the chance that Farmer Rob will run out of cows and sit idle. Intuitively, we should start with cows that it is quicker for Farmer John to milk, then move on to those that it is quicker for Farmer Rob to milk.

Before continuing, it will be helpful to have some formula for determining the total time to milk the cows in a particular order. Suppose the cows have been ordered, such that FJ takes times A1, A2, ... An and FR takes times B1, B2, ..., Bn. For any cow i, first FJ has to milk cows 1..i before FR can begin milking cows i..n, so the total time is at least A1 + ... + Ai + Bi + ... + Bn. It is not difficult to show that the largest of these values (over all i) is in fact the total milking time.

Our observations in the first paragraph suggest that there is some kind of sorting going on, but it isn't clear what the sort key should be. Let's imagine that we are bubble-sorting the cows, and decide under what conditions we might wish to swap two adjacent cows Carol and Daisy (currently in that order). Let the times for FJ and FR to milk them be AC, BC, AD and BD. The only lower bounds that change in the formula above are those where Carol or Daisy is i; thus there cannot be any benefit to an exchange unless

```
    max(AD + BD + BC, AD + AC + BC) <=
           max(AC + BC + BD, AC + AD + BD)
    <=>
    min(BD, AC) > min(AD, BC)
```

A bit of work reveals that this is a bona fide comparison operator (i.e. it is what the STL calls a strict weak ordering): it is equivalent to sorting by (Ai < Bi) ? Ai : BIGVALUE - Bi. The implication is that any optimal sequence can be bubble-sorted with this operator without ever lengthening the milking; of course, the bubble-sort is just for theoretical use and a decent O(N log N) sort can be used in the implementation.

```
#include <fstream>
#include <algorithm>
#include <numeric>

using namespace std;

#define MAXN 25000

struct cow {
    int A, B;
    int metric;
    bool operator <(const cow &b) const { return metric < b.metric; }
};

int main() {
    int N;
    cow cows[MAXN];
    ifstream in("mqueue.in");
    ofstream out("mqueue.out");

    in >> N;
    for (int i = 0; i < N; i++) {
        in >> cows[i].A >> cows[i].B;
        if (cows[i].A < cows[i].B) cows[i].metric = cows[i].A;
        else cows[i].metric = INT_MAX - cows[i].B;
    }
    sort(cows, cows + N);

    int sA = 0;
    int sB = 0;
    int ans = -1;
    for (int i = 0; i < N; i++) sB += cows[i].B;
    for (int i = 0; i < N; i++)
    {
        sA += cows[i].A;
        ans >?= sA + sB;
        sB -= cows[i].B;
    }
    out << ans << "\n";
    return 0;
}
```