Grace Wang
Due 2/16/2012
CIS-521

HW 3: Linear Programming, Adversarial Search

**LP/IP Question 1**

1. Maximize daily revenue, $5\ x_1 + 1.5\ x_2 + x_3 + 2\ x_4 + 7\ x_5$, such that:
    a. $0 \leq x_1 \leq 24$
    b. $10 \leq x_2 \leq 24$
    c. $0 \leq x_3 \leq 24$
    d. $15 \leq x_4 \leq 24$
    e. $0 \leq x_5 \leq 24$
    f. $8\ x_1 + 5\ x_2 + 3\ x_3 + 6\ x_4 + 10\ x_5 \leq 400$

    The first 5 constraints come from the minimum daily amount of production, as well as the fact that each machine cannot run more than 24 hours a day (an initial assumption is that each machine produces 1 kg of flour per hour that it is running). The last constraint (e) is due to maximum electricity usage.

    In order to solve this LP problem, we look at the power to price ratios of each product. $x_2$ and $x_4$ both have very high ratios, so we only want to produce the minimum amounts: $x_2 = 10$ and $x_2 = 15$.

    Product $x_5$ has the lowest ratio, so we set that to its maximum quantity, $x_5 = 24$. At this point, of the remaining products $x_1$ has the lowest ratio, so we use our remaining energy and set $x_1 = 2.5$. With values of $x_1 = 2.5$, $x_2 = 10$, $x_3 = 0$, $x_4 = 15$ and $x_5 = 24$, Revenue = \$225.50, and energy used = 400 kWh.

2. No, the solution is no longer valid because not all the quantities were integer values (in particular, $x_1 = 2.5$). In order to find a solution, we need to adjust the values of $x_1$ and $x_5$ (the products with the lowest power-to-price ratios) so that they become integers. Revenue in this revised problem is maximized at \$224 when $x_1 = 5$, $x_2 = 10$, $x_3 = 0$, $x_4 = 15$ and $x_5 = 22$. Once again, we use up all the available energy.

3. Rounding the solution to problem 1 give values of $x_1 = 3$, $x_2 = 10$, $x_3 = 0$, $x_4 = 15$ and $x_5 = 24$, which overreaches our energy constraint by 4 kWh. This is the amount of "slack" that we need to add to constraint (e) to make the rounded solution valid.

4. The linear programming problem posed in question 1 is much easier to solve than that in question 2. Solving for continuous problems involves identifying vertices of the bounded feasible region, while integer programming problems involve search algorithms that can quickly become exponential in time complexity.

**LP/IP Time Slot Scheduling**

Represent the data for each person $p_i$ using 3 variables: $x_i$, the number of meetings I have with that person; $s_i$, the start time of the meeting, and $e_i$, the end time of the meeting.

The objective function is to maximize the sum $x_1 + x_2 + \ldots + x_n$. Enforcing the constraint of separate time slots involves checking that for any person $p_i$, $s_i$ cannot be between $s_j$ and $e_j$ for another person $p_j$. Also, the tuple $(s_i, e_i)$ must be an element of $p_i$'s set of available time slots $S_i^{(av)}$.

**LP/IP Sudoku**

The objective is to have each of the 27 constraints (rows, columns, or 3x3 squares) contain the numbers 1-9. Let us denote *i* as the variable we use to iterate through this set of 27 constraints.

We then define variables to indicate the presence of one of the numbers 1-9 for each of the 27 constraints. These variables are represented by $x_{ij}$, where *i* refers to one of the constraints, and *j* is one of the numbers 1-9. $x_{ij}$ represents a Boolean condition and is either 0 (number *j* is present in constraint *i*) or 0 (not present). Because there are 27 constraints, and 9 numbers in each constraint, this IP model of Sudoku has 243 variables.

The objective is to maximize the overall sum, $\sum_i \sum_j x_{ij}$ at 243.

**Adversarial Search**

1. Nodes J (value 2), L (9) and M (9) will be pruned.

| Node | α-values | β-values |
|---|---|---|
| **A** | 4 | +∞ |
| **B** | -∞ | 4 |
| **C** | 4 | 3 |
| **D** | 4 | 1 |
| **E** | 4 | 4 |
| **F** | 8 | 8 |
| **G** | 6 | 6 |
| **H** | 7 | 7 |
| **I** | 3 | 3 |
| **J** | X | X |
| **K** | 1 | 1 |
| **L** | X | X |
| **M** | X | X |

2. Pruning in Games with Chance Nodes
    a. The best move at the root is to move towards node B: the expected value of taking the left path is (0.8)(2) + (0.2)(1) = 1.8, while the expected value taking the right path towards node C is only (0.6)(-2) + (0.4)(2) = -0.4.
    b. Given the values of the first six leaves, yes, we do still need to evaluate the 7$^{th}$ and 8$^{th}$ nodes because we know nothing about the range of their possible values. We already know that the expected value of the left path is 1.8. The expected value of the right path is (0.6)(-2)+(0.4)*x, where x is the smaller of the 7$^{th}$ and 8$^{th}$ nodes. If x is greater than 7.5, then the expected value of the right path will be greater than that of the left, and it will be more advantageous to travel towards node C.
    c. The range for the expected value of the left-hand chance node is 1.6 +/- .4, or 1.2 to 2.0.
    d. With the additional information that the leaf node values lie in the range [-2,2], we can prune the evaluation of nodes 6, 7 and 8. After reaching node 5, we know that the expected value of the right path is *at most* (0.6)(-2) + (0.4)*x. The value of this expression can never be greater than 1.8, which is the expected value of the left path.