Comprehensive Examination Spring 2005 (Analysis of Algorithms)

1a. [10 pts] Explain why an $O(N^3)$ algorithm would be preferable over an $O(2^N)$ algorithm when one does not have any idea about the expected input problem instance-size N.

1b. [10 pts] How will the decision be affected when there is some idea on how large the problem size (N) would be as input to the chosen algorithm?

2. [20 pts] Solve the following recurrence equation for the general solution.

$$T_n = 3T_{n-1} - 2T_{n-2}$$

3. [20 pts] The following is a recurrence formula (for aligning sequences with gaps, you need not be concerned about the problem that the formula models).

$$\begin{aligned} a[i,0] &= -2i \\ a[0,j] &= -2j \\ a[i,j] &= \max_{i,j>0}\{a[i-1,j]-2,\ a[i,j-1]-2,\ a[i-1,j-1]+p(i,j)\} \end{aligned}$$

where $p[i,\ j]$ is a given matrix of integers, and i and j are integers between 0 and a constant, say n.

a. Write a dynamic programming algorithm for computing `a[i,j]` for given `i` and `j`.

b. Analyze the complexity of your algorithm.

4. [20 pts] A sparse directed binary graph $G = (V,\ E)$ is represented as an adjacency list, where V is the set of n nodes, and E is the set of e edges, each of which is an ordered pair of nodes. Analyze the time-complexity of the following algorithm fragment.

```
For each node N1 in V do {
  Print N1;
  For each adjacent node N2 to N1 such that (N1, N2) is in E do {
    Print N2;
  }
}
```

5. [20 pts] Write a recursive divide-and-conquer algorithm for finding the maximum value over a sequence of numbers. Analyze your algorithm's time-complexity.