Algorithm Mid-term 2003/4/29 (二) 10:10~12:00

1. (10%) By definition:
   - $\Theta(\, g(n)\,) = \{\, f(n)$: there exist positive constants $c_1, c_2,$ and n_0 such that $0 \leqq c_1 g(n) \leqq f(n) \leqq c_2 g(n)$ for all $n \geqq n_0\}$.
   - $O(\, g(n)\,) = \{\, f(n)$: there exist positive constants c and n_0 such that $0 \leqq f(n) \leqq cg(n)$ for all $n \geqq n_0\}$.
   - $\Omega(\, g(n)\,) = \{\, f(n)$: there exist positive constants c and n_0 such that $0 \leqq cg(n) \leqq f(n)$ for all $n \geqq n_0\}$.

   $f(n) = 100n^2 - 101$, $g(n) = n^3$, please find out a small constant c and n_0 to show that $\Theta(\, g(n)\,) = f(n)$ or $O(\, g(n)\,) = f(n)$ or $\Omega(\, g(n)\,) = f(n)$.

2. (10%) If $T(n) = 3 * T(\, n / 3\,) + n$, then for $n > 3$, try to bound $T(n)$ with Θ–notation, and justify your answer..

3. (10%) Show that the lower bound of the comparative sort is O(n log n).

   (hint: $n! \cong \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$)

4. (10%) Determine the satisfiability of the following sets of clauses. If they are satisfiable, show the answer, if unsatisfiable, prove it..

   $(X_1 \text{ v } X_2 \text{ v } X_3\,)$
   $(-X_1 \text{ v } -X_2 \text{ v } X_4\,)$
   $(X_2 \text{ v } -X_3)$
   $(-X_2 \text{ v } -X_4)$

5. (10%) Write a non-deterministic algorithm to solve TSP problem.

6. (10%) Prove that 4-SAT problem is NP-Complete. (Using reduction method)

7. (10%) Now we have the eight messages ($M_1, M_2, \ldots, M_8$) with access frequencies (11, 3, 7, 2, 5, 6, 8, 14) , please design a greedy method to get the binary code such that $\sum_{i=1}^{8} l_i * \text{frequncey}(M_i)$ is minimum. (l_i is the length of code M_i)

8. (12%) In the 2-dimensional space, we say that a point (x_1, x_2) dominates (y_1, y_2) if $x_1 > x_2$ and $y_1 > y_2$. Design a algorithm with Divide-and-Conquer strategy to output the maximal points among these n points which are not dominated by any others.By the way, show the time complexity of your algorithm as well.

9. (15%) Questions : True or False (Explain your answer if it is false.)

   (1) If a problem is NP-complete, then it must be in NP and in NP-hard.
   (2) If we can find a polynomial algorithm to solve one NP-complete problem in average case, then NP-complete problem = P problem.
   (3) 2-SAT is a NP problem.
   (4) TSP is a NP-hard problem.
   (5) If a problem A can be reduced to Merge-Sort in linear time, then we can say that the lower bound of the time complexity of A is Ω***(n*log(n)).***

10. (3%) 請寫出本課程的課堂建議（包括內容、進度、作業、老師、助教…等）