CSE 417: Algorithms and Computational Complexity **Winter 2012**

Final Review Guide

The final will be cumulative. This review sheet only contains material from after the midterm. *You should also use the midterm review sheet to guide your studies for the earlier material.*

List of topics

You should be able to simulate and write pseudocode for the following algorithms:

Divid and conquer:

- QuickSelect for median finding;
- Closest pair algorithm;
- Karatsuba's algorithm;

Dynamic programing:

- Weighted interval scheduling;
- Segmented least squares;
- Subset sum;

You should also be able to

- analyze the time complexity of a divide and conquer algorithm;
- understand how to transform a recursive algorithm to a dynamic programming algorithm;
- go from a recurrence to a dynamic programming algorithm;
- analyze the time complexity of a dynamic programming algorithm;
- write a traceback algorithm to construct a solution as part of a dynamic programming algorithm;
- know the definitions of P, N, NP-hard, and NP-complete;
- understand the significance of the P = NP problem and how the notion of NP-completeness relates to this question;
- be familiar with some decision problems we discussed:

  - SAT
  - 3-SAT
  - Vertex Cover
  - Set Cover
  - Independent Set
  - Clique
  - Hamiltonian Path
  - TSP

- know the definition of a polynomial time reduction;
- know how to show a problem is NP-complete, using reductions;
- be familiar with the result of the Cook-Levin Theorem (but not its proof);
- know the reductions we showed in class
  - 3-SAT to Independent Set
  - Independent Set to Vertex Cover
  - Vertex Cover to Set Cover

Practice Problems

In addition to going through the material presented in lecture, one of the best ways to prepare for the final is to go through the homeworks and make sure you understand the solutions. Here we're giving you some additional problems that we think will help you study for the final. We'll go through some solutions during the review session.

1. The Bipartiteness problem takes as input a graph $G = (V, E)$ and asks whether the graph is bipartite.
   (a) Show that this problem is in NP.
   (b) Do you think it's likely that this problem is NP-complete? Why or why not?

2. Analyze the time complexity of the following algorithm. (Don't try to make sense of what it is doing because it doesn't do anything that makes sense.) Assume `A` is a one-dimensional array of length n.

```
NonSensical(A):
  if len(A) <= 4:
    return a random element of A
```

```
else:
  let A_1 be the subarray of A indexed from 1 to n / 2
  let A_2 be the subarray of A indexed from n / 4 to 3n / 4
  let A_3 be the subarray of A indexed from n / 2 to n
sum = 0
for i = 1 to n:
  sum += A[i]
return sum + NonSensical(A_1) + NonSensical(A_2) + NonSensical(A_3)
```

3. Chapter 6, Exercise 1.