**COT 6410 Spring 2014 Final Exam E2 Sample Name:**

* The related notion of polynomial reducibility and equivalence require that the reducing function, **f** above, be computable in polynomial time in the size of the instance of the element being checked. The notation just replaces the **m** with a **p**, as in **A ≤p B** and **A ≡p B**.
* A decision problem **P** is in **P** if it can be solved by a deterministic Turing machine in polynomial time.
* A function problem **F** is in **FP** if it can be solved by a deterministic Turing machine in polynomial time.
* A decision problem **P** is in **NP** if it can be solved by a non-deterministic Turing machine in polynomial time. Alternatively, **P** is in **NP** if a proposed proof of any instance having answer yes can be verified by a deterministic Turing machine in polynomial time.
* A function problem **F** is in **FNP** if a proposed solution to it can be verified by a deterministic Turing machine in polynomial time. The proposed solution must be at most polynomial larger than the input.
* A decision problem **P** is **NP-complete** if and only if it is in **NP** and, for any problem **Q** in **NP**, it is the case that **Q ≤p P**.
* A function problem **P** is **NP-hard** if and only if there is an **NP-complete** problem **Q** that is polynomial time Turing-reducible to **P**. We often limit our domain of consideration to decision problems when talking of **NP-hard**, but the concept also applies to function problems.
* A function problem **P** is **NP-easy** if and only if it is polynomial time Turing-reducible to some **NP** problem **Q**.
* A function problem **P** is **NP-equivalent** if and only if it is both **NP-hard** and **NP-easy**.

**COT 6410 Spring2014 Final Exam Sample E2 Questions**

 **1.** We described the proof that **3SAT** is polynomial reducible to Subset-Sum.

**a.)** Describe **Subset-Sum**

**b.)** Show that **Subset-Sum** is in **NP**

**c.)** Assuming a **3SAT** expression **(a + ~b + c) (~a + b + ~c)**, fill in the upper right part of the reduction from **3SAT** to **Subset-Sum**.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **a** | **b** | **c** | **a + ~b + c** | **~a + b + ~c** |
| **a** |  |  |  |  |  |
| **~a** |  |  |  |  |  |
| **b** |  |  |  |  |  |
| **~b** |  |  |  |  |  |
| **c** |  |  |  |  |  |
| **~c** |  |  |  |  |  |
| **C1** |  |  |  |  |  |
| **C1’** |  |  |  |  |  |
| **C2** |  |  |  |  |  |
| **C2’** |  |  |  |  |  |
|  |  |  |  |  |  |

 **d.)** List some subset of the numbers above (each associated with a row) that sums to **1 1 1 3 3**. Indicate what the related truth values are for **a**, **b** and **c**.

 **2**. **Partition** refers to the decision problem as to whether some set of positive integers **S** can be partitioned into two disjoint subsets whose elements have equal sums. **Subset-Sum** refers to the decision problem as to whether there is a subset of some set of positive integers **S** that precisely sums to some goal number **G**.

 **a.)** Show that **Partition** **≤p Subset-Sum**.

 **b.)** Show that **Subset-Sum ≤p Partition**.

 **3.** Consider the decision problem asking if there is a coloring of a graph with at most k colors, and the optimization version that asks what is the minimum coloring number of a graph. You can reduce in both directions. So, do that. Make sure you carefully explain for each direction just what it is that you are proving.

 **4. QSAT** is the decision problem to determine if an arbitrary fully quantified Boolean expression is true. Note: **SAT** only uses existential, whereas **QSAT** can have universal qualifiers as well so it includes checking for Tautologies as well as testing Satisfiability. What can you say about the complexity of **QSAT** (is it in **P**, **NP**, **NP-Complete**, **NP-Hard**)? Justify your conclusion.

 **5.** Consider the following set of independent tasks with associated task times:
**(T1,7), (T2,6), (T3,2), (T4,5), (T5,6), (T7,1), (T8,2)**Fill in the schedules for these tasks under the associated strategies below.

Greedy using the list order above:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Greedy using a reordering of the list so that longest running tasks appear earliest in the list:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Greedy using a reordering of the list so that shortest running tasks appear earliest in the list:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

 **6.** Present a gadget used in the reduction of **3-SAT** to some graph theoretic problem where the gadget guarantees that each variable is assigned either **True** or **False**, but not both. Of course, you must tell me what graph theoretic problem is being shown **NP-Complete** and you must explain why the gadget works.