Assignment 6

Problem 13.7

1. We can reduce the problem of Concave-Multiset-Multicover(CMCMC) to regular multiset multicover (MCMC) by doing the following: For each set S_i, construct sets S_i^j, having a cost $f_i(j)$, which represents using S_i j times. To show that this is a proper reduction, we must prove two things:

   Lemma 1:

   Every feasible solution in CMCMC has a corresponding feasible solution of equal cost in the MCMC problem we have created.

   Proof: For each value i, the feasible solution in CMCMC will have picked each S_i j_i times, at a cost of $f_i(j_i)$. For any feasible solution in CMCMC we can easily construct a solution in MCMC which for each value of i, we select $S_i^{j_i}$, which has an equal cost of $f_i(j_i)$, and covers each element an equal number of times, therefore being a feasible solution of equal cost.

   Lemma 2:

   Every feasible solution in MCMC has a corresponding feasible solution of equal or lesser cost in the CMCMC problem.

   Proof: For each value i, we have chosen $S_i^{j_1}, S_i^{j_2}, ...S_i^{j_k}$ at a cost of $\Sigma_{l=1}^{k} f(j_l)$ However, from this, we can obtain a solution that is at least as cheap by replacing all of these sets for a single set S_i^{JTotal}, where $JTotal = \Sigma_{l=1}^{k} j_l$ having a cost of $f_i(JTotal)$. This new solution will cover each element equally, but will have cost at most the combined cost of the other sets, as f is a concave function. From this alternate solution which is at least as cheap in MCMC, we can easily go to an equal cost solution in CMCMC by simply by picking each S_i $JTotal_i$ times, which will give us the same exact coverage and will have the same exact cost.

   Therefore, $Cost(OPT_{MCMC}) = Cost(OPT_{CMCMC})$
   $Cost(CMCMCSolution) \leq Cost(MCMCSolution)$
   $\leq H_n * Cost(OPT_{MCMC}) = H_n * Cost(OPT_{CMCMC})$,

   Which shows that there is an approximation factor preserving reduction from CMCMC to MCMC.

2. To show that there is no need to explicitly construct all sets, we must show that for each S_i in the original problem, only certain S_i^j have a chance of having optimal cost-effectiveness. The cost effectiveness for each i is:

   $E(j) = \frac{ElementsCovered(j)}{f(j)} = \frac{\Sigma_{e \in S} min(r'_e, j)}{f(j)}$

   where r'_e is the number of additional times an element e must be covered.

   If $j \neq r'_e$ for any $e \in S$, then
   $E'(j) = \frac{ElementsCovered'(j)}{f(j)} - \frac{ElementsCovered(j) * f'(j)}{f(j)^2}$, which must be equal to zero if we are to have a local maximum.

   Therefore, multiplying both sides by $f(j)^2$, we have $ElementsCovered'(j) * f(j) - ElementsCovered(j) * f'(j) = 0$

   In order to have a local maximum, $E''(j) < 0$ In the neighboorhood about this local maximum, $E'(j) = 0$, $E''(j) < 0$
   $d(E'(j) * f(j)^2) = E'' * f(j)^2 + E'(j) * f'(j) * f(j)^2 = E'' * f(j)^2 < 0$, as $f(j)^2 \geq 0$

   $E'(j) * f(j)^2 = ElementsCovered'(j) * f(j) - ElementsCovered(j) * f'(j)$

   $d(E'(j) * f(j)^2) = ElementsCovered''(j) * f(j) + ElementsCovered'(j) * f'(j) - ElementsCovered'(j) * f'(j) - ElementsCovered(j) * f''(j)$

   $= -ElementsCovered(j) * f''(j) < 0$

   However, $ElementsCovered(j) \geq 0$, $f''(j) \leq 0$, so we have a contradiction.

   Therefore there can be no local maximum of efficiency outside a neighborhood where $j = r'_e$, which means there can be no aboslute maximum outside one of those neighborhoods where a certain element is "Maxed out". Therefore, the greedy algorithm only has to look at the values of j where $j = r'_e$ for some $e \in S$, which means the most cost effective set can be computed in polynomial time.