1 Book 8.4: Resource Allocation

Mathematical Formulation

Given an input of n processes, $P = \{p_1, p_2, ..., p_n\}$, and a set of m resources, $R = \{r_1, r_2, ..., r_m\}$, each process requires a set of resources $R^*, R^* \subseteq R$. Each process in required active iff every resource $r \in R^*$ is allocated to it, however each resource can only be used once. Given a number $k>0$, determine if resources $\in R$ can be allocated so that at least k processes $\in P$ will be active. For the following cases, give a polynomial algorithm or proove it is NP-Complete.

(a) General Case i.e. $k>0$

(b) $k = 2$

(c) Have 2 types of resources, if either one is fufilled for a process, then the process is considered active

(d) Each resource can be allocated a maximum of 2 times

1.1 Part (a)

Proposition 1. *This is an NP-Complete problem.*

Proof.
We show first that this problem is NP. Given a set of k processes, check to see if there are resources shared between them all which take $k \cdot m^* = totalResources$ time where m^* is the average number of resources all of the processes contain. More specifically we can create a HashSet which we will add each resource as we see them in each set. Therefore we will check to see if the resource is already present in the set, if it is then there is a repetition; if we can get to the end, then there are no more.

Now we say that this is an NP-Hard problem by doing a reduction from the Independent Set problem i.e. $IndependentSet \leq_P Problem$. Assume that we have an independent set of graph G and the number k. We take the verticies, $V \in G$ to represent each process and all edges $E \in G$ to be resources. Therefore verticies are only considered to be adjacent if they have a common resource. So if there exists an independent set, $G^* \in G$ s.t. $|G^*| \geq k$, we say that this implies that verticies within G^* have no common edges, i.e. there are no shared resources $\implies$ solved the problem. Now if there exist k processes which resources from a disjoint set, then the corresponding graph would have no edges in common for these processes $\implies$ independent set.

Since Independent Set is NP-hard $\implies$ this problem is at least NP-Hard

$$\therefore \text{this process is both NP and NP-Hard} \implies \text{NP-Complete.}$$

$\square$

1.2 Part (b)

This can be solved in polynomial time by brute force since there only exist n^2 process combinations. Therefore doing the above mentioned algorithm for checking if sets are indeed valid, we can go through all pairs; if at any pair they have a disjoint set of resources, then we accept, if we go through every possible combination and never get into an accepting state, we will say it is impossible.

1.3 Part (c) and Part (d)

These are seen as more specialized versions of problem (a), therefore we can can generalize and say that these will also be NP-Complete.