CS3231
Tutorial 12

1. n^3 is fully space constructible (nice) as it can be computed within space n^3 as witnessed by the following (students are not required to show this, as I didn't spend much time on constructibility).

   Copy input to three different worktapes.

   For $i = 1$ to n

   For $j = 1$ to n

   For $k = 1$ to n

   Write # on fourth worktape and move right on this worktape.

   EndFor

   EndFor

   EndFor

   The i, j, k can be tracked by moving through the three worktapes with copied input.

   As $\lim_{n\to\infty} \frac{n^2}{n^3} = 0$, using space hierarchy theorem, we have that $DSPACE(n^3) - DSPACE(n^2) \neq \emptyset$.

   As trivially, $DSPACE(n^2) \subseteq DSPACE(n^3)$, we have the result.

2. Done in T11.

3. It is easy to check that Travelling Salesman problem is in NP: Given a weighted graph $G = (V, E)$ and a bound B, guess a permutation $v_1, v_2, \ldots, v_n$ of V, and verify that $v_1 v_2 \ldots v_n v_1$ is a simple circuit which goes through all the vertices exactly once and the weight of the circuit is at most B. If the verification is succesful, then accept.

   To show that Travelling Salesman problem is NP-hard, we reduce from Hamiltonian circuit problem to TSP.

   Suppose $G = (V, E)$ is a Hamiltononian circuit problem, where $|V| = n$. Then construct a TSP problem $G' = (V', E')$, with $B = n$ as follows:

   - $V' = V$
   - $E' = \{(u, v) : u, v \in V, u \neq v\}$, where the weights of the edges are given by
     - If $(u, v) \in E$, then $wt(u, v) = 1$.
     - If $(u, v) \notin E$, then $wt(u, v) = n + 2$.

   It is easy to verify that the reduction can be done in polynomial time.

   Now, G has a Hamiltonian Circuit iff G' has a Hamiltonian circuit of weight $\leq B = n$. To see this, note that

   - Any Hamiltonian circuit in G has weight n in G' (with the same, but weighted, edges).

- There is no Hamiltonian circuit in G' of weight less than n, as each edge has weight at least 1.
- If there is a Hamiltonian circuit C in G' of weight n, then C can only contain the edges which were in G, as otherwise weight of C would have been at least $n+2$. Thus, C (without weights) is also a Hamiltonian circuit in G.

4. It is easy to see that the problem is in NP: Guess a path $v_0 v_1 v_2 \ldots v_k$ and verify that it is indeed a simple path (no repetition of vertices, and (v_i, v_{i+1}) is an edge). If verification is successful, then accept.

   To show that the problem is NP-hard, we show a reduction from Hamiltonion Circuit problem. Suppose $G = (V, E)$ is a given HC problem. Construct $G' = (V', E')$ as follows:

   Suppose $V = \{v_1, \ldots, v_n\}$. Suppose X is the set of vertices adjacent to v_1.

   Then, $V' = V \cup \{v_0, v_{n+1}, v_{n+2}\}$.

   $E' = E \cup \{(v_0, v_1), (v_{n+1}, v_{n+2})\} \cup \{(v, v_{n+1}) : v \in X\}$.

   k (the number of edges needed in the simple path) is $n+2$.

   Now, if G has a Hamiltonian circuit in which there is an edge (v_1, v), then G' has a simple path of size $n+2$, by dropping the edge (v_1, v) from HC and adding the edges (v_0, v_1), (v, v_{n+1}), (v_{n+1}, v_{n+2}).

   If the graph G' has a simple path containing $n+2$ edges, then it must start from (v_0, v_1), and end in (v_{n+1}, v_{n+2}). Suppose the last edge before (v_{n+1}, v_{n+2}) in this path is (v, v_{n+1}). Then, by dropping the edges (v_0, v_1), (v_{n+1}, v_{n+2}) and (v, v_{n+1}) from the path and adding (v_1, v) we get a HC in G.

5. It is easy to see that the processor scheduling problem is in NP: Just guess a schedule (i.e., assignment of jobs to processors), and verify that each job is assigned to some processor and each processor's load is at most D (i.e., sum of the time taken by the jobs assigned to each of the processor is at most D).

   To show NP-hardness, we reduce the partition problem to processor scheduling problem.

   Suppose we are given a partition problem $A = \{a_1, a_2, \ldots, a_n\}$ where $s(a)$ denotes the size of $a \in A$.

   Then, construct the processor scheduling problem as follows. There are $k = 2$ processors. $J = A = \{a_1, a_2, \ldots, a_n\}$, and the time T_i taken for job $a_i \in J$ is $s(a_i)$. The deadline is $\lfloor \sum_{a \in A} s(a)/2 \rfloor$.

   Now, it is easy to verify that there is a partition of A into equal weighted partitions iff these two parts can be scheduled into the two processors using total time at most/exactly $\lfloor \sum_{a \in A} s(a)/2 \rfloor$.

6. Note that for any polynomial $p(n)$, there exists a natural number k such that $p(n) \leq kn^k$, for all n. Using the technique of space compression, we get that $NSPACE(p(n)) \subseteq NSPACE(n^k)$. Thus, using Savitch's theorem, we get:

   $NPSPACE = \bigcup_{k \in N} NSPACE(n^k) \subseteq \bigcup_{k \in N} DSPACE(n^{2k}) \subseteq PSPACE \subseteq NPSPACE$.

   Thus, $NPSPACE = PSPACE$.