1. Scheduling jobs in parallel [25 points]

Suppose that there are \( m \) identical machines in a cluster. There are \( n \) jobs to be run, where the \( i \)th job requires processing time \( p_i \), on any of the machines. Each job must be run on a single machine, and no machine can run two jobs at once. A schedule is an allocation of jobs to machines. The time for each machine is the sum of the times of the jobs allocated to it; the time for the schedule is the maximum time of any machine. (This is called the makespan of the schedule.)

(a) [5 points] For a given set of \( n \) jobs with times \( p_i \) (\( 1 \leq i \leq n \)), let \( T^*_\text{max} \) be the time required by an optimal schedule, with \( m \) machines. Explain why the inequalities

\[
T^*_\text{max} \geq \max_{1 \leq i \leq n} p_i
\]

and

\[
T^*_\text{max} \geq \frac{1}{m} \sum_{i=1}^{n} p_i
\]

must hold.

(b) [15 points] Consider a schedule in which, whenever a machine becomes available, an arbitrary job is assigned to it, from those not already assigned. Let \( T_{\text{max}} \) be its maximum time used by any machine to run its jobs, under this schedule.

Show that the cost \( T_{\text{max}} \) of the resulting schedule satisfies

\[
T_{\text{max}} \leq \frac{1}{m} \sum_{i=1}^{n} p_i + \max_{1 \leq i \leq n} p_i.
\]

Conclude that the scheduling strategy has approximation ratio at most 2.

(c) [5 points] Show that 2 is the correct ratio for the algorithm above. That is, give a class of inputs to the algorithm of the previous part, such that it can choose a schedule with ratio arbitrarily close to 2.

(Problem adapted from CLRS, 35-5, pp 1051–2.)
2. **A meeting of sets** [20 points]

Say that one set *meets* another if their intersection is non-empty. Consider the following parameterized problem.

**Instance:** a collection of sets $B_i$ ($1 \leq i \leq m$), where each $B_i$ has size at most $c$.

**Parameter:** A pair $(c, k)$ of positive integers.

**Question:** Is there a set $M$, of size at most $k$, such that $M$ meets every set $B_i$; i.e., for each $i$, $M \cap B_i \neq \emptyset$? (If some $B_i$ has size greater than the parameter $c$, the answer is simply “no”.)

Show that this problem is fixed-parameter tractable. That is, give an algorithm that solves it in time $f(c, k) \cdot n^d$, for some constant $d$ and an arbitrary function $f$, where $n$ is the total size of all sets $B_i$.

3. **Coloring graphs with small treewidth** [25 points]

Recall that a tree decomposition of a graph $G$ on $n$ nodes consists of a tree $T$ and a set of connected components $C_1, \ldots, C_n$ of $T$, with the property that if $(v_i, v_j) \in E$, then $C_i$ and $C_j$ have at least one node in common. Such a decomposition has width $w$ iff for each node $x$ of $T$, the set $\{ C_i \mid x \in C_i \}$ has cardinality at most $w + 1$.

(a) [10 points] Show that there is a constant $c_2$ such that every graph with treewidth 2 has a colouring that uses at most $c_2$ colours.

Give an algorithm that, on input consisting of a graph $G$ and a tree decomposition $T$ of $G$ with width 2, produces a colouring using $c_2$ colours. Your algorithm should run in polynomial time.

Note that one way to prove that $c_2$ exists is to provide the algorithm and a proof that it always finds a colouring.

For full marks, find the best possible $c_2$, but any constant will achieve most of the marks.

(b) [15 points] Extend the previous part, to show that for each $k$ there is a constant $c_k$ such that any graph of treewidth $k$ can be colored with at most $c_k$ colours.

Describe a fixed-parameter algorithm that, given a graph and a tree decomposition with width $k$, finds an optimal colouring in time $f(k) \cdot n^{O(1)}$ (where $n$ is the size of the graph and the decomposition).

Again, you may justify the existence of $c_k$ by giving an algorithm that achieves it.