Problems for the course
"Implementation of algorithms in software"

A *partition* (or an assignment) of (the elements of) a set (of jobs, or items, or vertices) Y into sets $J_1, J_2, \dots, J_m$ must satisfy $\bigcup_{1 \le i \le m} J_i = Y$, and for any $1 \le i_1 < i_2 \le m$, $J_{i_1} \cap J_{i_2} = \emptyset$ (every element of Y belongs to exactly one subset). For some problems (such as scheduling), the value m is fixed in advance, and for some problems it is not fixed (for example, bin packing). We will use a partition to describe a schedule (without specifying the exact times allocated to the jobs), where J_i is the set of jobs of machine i. A different way to define a schedule or assignment for the set of jobs J is a function $A : J \to \{1, 2, \dots, m\}$. In this case we let $J_i = \{j \in J | A(j) = i\}$. The definitions are similar for bin packing etc.

1. Scheduling problems

1. **Scheduling with few conflicts on identical machines to minimize makespan.**

   Input: A set of n jobs $J = \{1, 2, \dots, n\}$, where job j has an integer processing time $p_j > 0$, and a integer number of (identical) machines $m \ge 2$. A bipartite undirected graph $G = (V_1, V_2, E)$ where $V_1 \cup V_2 = J$ (there are no edges between pairs of vertices of V_ℓ for $\ell = 1, 2$), and $0 \le |E| \le 2|V|$.

   Goal: Find an assignment of the jobs to the m machines, $J_1, J_2, \dots, J_m$, such that if $j_1, j_2 \in J_i$, then $(j_1, j_2) \notin E$.

   Objective:: Minimize $\max_{1 \le i \le m} \sum_{j \in J_i} p_j$.

2. **Scheduling on favorite machines to minimize makespan.**

   Input: A set of n jobs $J = \{1, 2, \dots, n\}$, where job j has an integer size $p_j > 0$, an integer number of (identical) machines $m \ge 2$, and an integer $s \ge 2$. Job j also has a machine index i_j ($1 \le i \le m$) such that its processing time on machine i_j is p_j, and its processing time on any machine $i \ne i_j$ is $s \cdot p_j$ (that is, larger by a factor of s).

   Goal: Find an assignment σ of the jobs to the m machines, $J_1, J_2, \dots, J_m$. Let $C_i(\sigma)$ be defined as follows: $C_i(\sigma) = s \cdot \sum_{j \in J, \sigma(j) = i, i_j \ne i} p_j + \sum_{j \in J, \sigma(j) = i, i_j = i} p_j$ (this is the completion time of machine i).

   Objective:: Minimize $\max_{1 \le i \le m} C_i(\sigma)$.

3. **Scheduling with favorite machines to maximize the minimum completion time.**

   Input: A set of n jobs $J = \{1, 2, \dots, n\}$, where job j has an integer size $p_j > 0$, an integer number of (identical) machines $m \geq 2$, and an integer $s \geq 2$. Job j also has a machine index i_j ($1 \leq i \leq m$) such that its processing time on machine i_j is $s \cdot p_j$, and its processing time on any machine $i \neq i_j$ is p_j.

   Goal: Find an assignment σ of the jobs to the m machines, $J_1, J_2, \dots, J_m$. Let $C_i(\sigma)$ be defined as follows: $\sum_{j \in J, \sigma(j)=i, i_j \neq i} p_j + s \cdot \sum_{j \in J, \sigma(j)=i, i_j = i} p_j$.

   Objective:: Maximize $\max_{1 \leq i \leq m} C_i(\sigma)$.

4. **Scheduling with five types to minimize the makespan.**

   Input: An integer number of (identical) machines $m \geq 2$. A set of n jobs $J = \{1, 2, \dots, n\}$, where every job j has an integer processing time $p_j > 0$ and a type $t_j \in \{1, 2, 3, 4, 5\}$.

   Goal: Find a partition of the jobs of to the machines, $J_1, J_2, \dots, J_m$, such that every subset has jobs of at most three types (for any i there are two values $k_{1i}, k_{2i} \in \{1, 2, 3, 4, 5\}$, such that if $j \in J_i$, then $t_j \neq k_{1i}, k_{2i}$).

   Objective:: Minimize $\max_{1 \leq i \leq m} \sum_{j \in J_i} p_j$.

5. **Scheduling with five types to to maximize the minimum completion time.**

   Input: An integer number of (identical) machines $m \geq 2$. A set of n jobs $J = \{1, 2, \dots, n\}$, where every job j has an integer processing time $p_j \geq 0$ and a type $t_j \in \{1, 2, 3, 4, 5\}$.

   Goal: Find a partition of the jobs of to the machines, $J_1, J_2, \dots, J_m$, such that every subset has jobs of at least four types (for any i, $|\bigcup_{j \in J_i} \{t_j\}| \geq 4$).

   Objective:: Minimize $\max_{1 \leq i \leq m} \sum_{j \in J_i} p_j$.

6. **Scheduling with special jobs on identical machines to minimize the makespan.**

   Input: A set of n jobs $J = \{1, 2, \dots, n\}$, where job j has an integer processing time $p_j > 0$. An integer number of (identical) machines $m \geq 2$ (where $n \geq 2m$).

   Jobs $1, 2, \dots, m$ are called special, jobs $m + 1, \dots, 2m$ are called unique, all other jobs are called regular.

   Goal: Find a partition (assignment) of the jobs to the (m) machines, $I_1, I_2, \dots, I_m$, such that for any i ($1 \leq i \leq m$), $J_\ell \cap \{1, \dots, m\} \neq \emptyset$ and $J_\ell \cap \{m + 1, \dots, 2m\} \neq \emptyset$ (every machine has at least one special job and one unique job).

   Objective:: Minimize $\max_{1 \leq i \leq m} \sum_{j \in J_i} p_j$.

7. **Tradeoff scheduling.**

   Input: A set of n jobs $J = \{1, 2, \dots, n\}$, where job j has an integer processing time $p_j > 0$.

   Goal: Find a partition (assignment) of the jobs into non-empty subsets, $I_1, I_2, \dots, I_k$ (for any $1 \leq i \leq k$, $|I_i| \neq 0$).

   Objective:: Minimize $k + \max_{1 \leq i \leq k} \sum_{j \in J_i} p_j$. (There is a cost to every machine that receives at least one job.)

8. **Scheduling with even cardinality sets on identical machines to minimize the makespan.**

   Input: A set of $2n$ jobs $J = \{1, 2, \ldots, 2n\}$, where job j has an integer processing time $p_j > 0$. An integer number of (identical) machines $m \geq 2$.

   Goal: Find a partition (assignment) of the jobs to the (m) machines, $I_1, I_2, \ldots, I_m$, such that for any i ($1 \leq i \leq m$), $|J_i|$ is even (for each machine, the number of jobs of received by it is divisible by 2).

   Objective:: Minimize $\min_{1 \leq i \leq m} \sum_{j \in J_i} p_j$.

9. **Scheduling with three types on identical machines to minimize makespan.**

   Input: Three sets of jobs, each consisting of n jobs $J_1 = \{1, 2, \ldots, n\}$, $J_2 = \{n+1, n+2, \ldots, 2n\}$, $J_3 = \{2n+1, 2n+2, \ldots, 3n\}$,, where job j has an integer processing time $p_j > 0$. An integer number of (identical) machines $m \geq 2$.

   Let the type of j be denoted by $t_j \in \{1, 2, 3\}$, where $t_j = 1$ if $1 \leq j \leq n$, $t_j = 2$ if $n + 1 \leq j \leq 2n$, and $t_j = 3$ if $2n + 1 \leq j \leq 3n$.

   Goal: Find a partition (assignment) of the jobs to the (m) machines, $I_1, I_2, \ldots, I_m$, such that for any pair i,ℓ ($1 \leq i \leq m$, $\ell = 1, 2, 3$), $I_i \cap J_\ell \neq \emptyset$ (every machine has at least one job of each type).

   Objective:: Minimize $\max_{1 \leq i \leq m} \sum_{j \in J_i} p_j$.

10. **Fair scheduling with two types on identical machines to minimize makespan.**

    Input: Two sets of jobs, each consisting of n jobs $J_1 = \{1, 2, \ldots, n\}$, $J_2 = \{n+1, n+2, \ldots, 3n\}$, where job j has an integer processing time $p_j > 0$. An integer number of (identical) machines $m \geq 2$.

    Let the type of j be denoted by $t_j \in \{1, 2\}$, where $t_j = 1$ if $1 \leq j \leq n$, and $t_j = 2$ if $1 \leq j \leq 3n$.

    Goal: Find a partition (assignment) of the jobs to the (m) machines, $I_1, I_2, \ldots, I_m$, such that for any i ($1 \leq i \leq m$), $|I_i \cap J_1| = 2 \cdot |I_i \cap J_2|$ (every machine has twice as many jobs of the second types as its has of the first type).

    Objective:: Minimize $\max_{1 \leq i \leq m} \sum_{j \in J_i} p_j$.

11. **Almost equitable scheduling.**

    Input: An integer number of (identical) machines $m \geq 2$. A set of $m \cdot n$ jobs $J = \{1, 2, \ldots, mn\}$, where job j has an integer processing time $p_j > 0$.

    Goal: Find a partition (assignment) of the jobs into non-empty subsets, $I_1, I_2, \ldots, I_m$ (where $|I_i| \in \{n-1, n, n+1, n+2\}$, and for any $1 \leq i_1 < i_2 \leq m$, $I_{i_1} \cap I_{i_2} = \emptyset$).

    Objective:: Minimize $\max_{1 \leq i \leq k} \sum_{j \in J_i} p_j$.

12. **Almost equitable covering.**

    Input: An integer number of (identical) machines $m \geq 2$. A set of $m{\cdot}n$ jobs $J = \{1, 2, \ldots, mn\}$, where job j has an integer processing time $p_j > 0$.

    Goal: Find a partition (assignment) of the jobs into non-empty subsets, $I_1, I_2, \ldots, I_m$, where for any $1 \leq i \leq k$, $|I_i| \in \{n-1, n, n+2\}$).

    Objective:: Maximize $\min_{1 \leq i \leq k} \sum_{j \in J_i} p_j$.

13. **Equitable pair scheduling.**

    Input: An even integer number of (identical) machines $m \geq 2$. A set of $m \cdot n$ jobs $J = \{1, 2, \ldots, mn\}$, where job j has an integer processing time $p_j > 0$.

    Goal: Find a partition (assignment) of the jobs into non-empty subsets, $I_1, I_2, \ldots, I_m$ (where $\bigcup_{1 \leq i \leq m} I_i = J$, for any $1 \leq i \leq m/2$, $|I_{2i-1} \cup I_{2i}| = 2n\}$).

    Objective:: Minimize $\max_{1 \leq i \leq k} \sum_{j \in J_i} p_j$.

14. **Scheduling with cardinality constraints.**

    Input: An integer number of (identical) machines $m \geq 2$. A set of n jobs $J = \{1, 2, \ldots, n\}$, where job j has an integer processing time $p_j > 0$. An integer parameter k (where $k \geq \lceil \frac{n}{m} \rceil$).

    Goal: Find a partition (assignment) of the jobs into non-empty subsets, $I_1, I_2, \ldots, I_m$ (where for any $1 \leq i \leq m$, $|I_i| \leq k$).

    Objective:: Minimize $\max_{1 \leq i \leq k} \sum_{j \in J_i} p_j$.

2. Graph problems

1. **Maximum partition into four parts.**

   Input: An undirected graph $G = (V, E)$ where $V = \{1, 2, \ldots, n\}$, such that $|V|$ is divisible by 4, and a positive integral weight function $w : E \to \mathbb{N}$ (on edges).

   Goal: Find a partition of the vertices into four disjoint sets U_1, U_2, U_3, and U_4 (where $U_i \cap U_j = \emptyset$ for $i \neq j$, and $U_1 \cup U_2 \cup U_3 \cup U_4 = V$), such that $U_i = \frac{|V|}{4}$ for $i = 1, 2, 3, 4$.

   Objective:: Maximize $\sum_{i=1,2,3,4} \sum_{u,v \in U_i, (u,v) \in E} w((u, v))$.

2. **Minimum partition into three parts.**

   Input: An undirected graph $G = (V, E)$ where $V = \{1, 2, \ldots, n\}$, such that $|V|$ is divisible by 4, and a positive integral weight function $w : E \to \mathbb{N}$ (on edges).

   Goal: Find a partition of the vertices into three disjoint sets U_1, U_2, and U_3 (where $U_i \cap U_j = \emptyset$ for $i \neq j$, and $U_1 \cup U_2 \cup U_3 = V$), such that $|U_1| = |U_2| = \frac{|V|}{4}$ and $|U_3| = \frac{|V|}{2}$.

   Objective:: Minimize $\sum_{i=1,2,3} \sum_{u,v \in U_i, (u,v) \in E} w((u, v))$.

3. **Double vertex cover.**

   Input: An connected undirected graph $G = (V, E)$ where $V = \{1, 2, \ldots, n\}$, and a positive integral weight function $w : V \to \mathbb{N}$ (on vertices).

   Goal: A vertex cover is a subset $X \subseteq V$, such that for any $e = (u, v) \in E$, $u \in X$ or $v \in X$ (or both). Find two distinct vertex covers X and X' (such that $X \neq X'$).

   Objective:: Minimize $\sum_{v \in X} w(v) + \sum_{u \in X'} w(u)$.

3. Packing problems

1. **Bin packing with special items.**

   Input: An integer bin capacity $C > 0$. A set of n items $I = \{1, 2, \ldots, n\}$, where item i has an integer size s_i such that $0 < s_i \leq C$. A subset $I' \subseteq I$ of special items.

   Goal: Find a partition of the items to bins, $B_1, B_2, \ldots, B_k$, where $\sum_{i \in B_\ell} s_i \leq C$ and $|B_i \cap I'| \leq 2$ (any bin has at most two special items) hold for any $1 \leq \ell \leq k$.

   Objective:: Minimize k.

2. **Bin covering with special items.**

   Input: An integer bin capacity $C > 0$. A set of n items $I = \{1, 2, \ldots, n\}$, where item i has an integer size s_i such that $0 < s_i \leq C$. A subset $I' \subseteq I$ of special items.

   Goal: Find a partition of the items to bins, $B_1, B_2, \ldots, B_k$, where $\sum_{i \in B_\ell} s_i \geq C$ and $|J_i \cap I'| \leq 3$ (no bin has more than three special items) hold for any $1 \leq \ell \leq k$.

   Objective:: Maximize k.

4. Two stage problems (could be harder than other problems)

A vertex cover is a subset $X \subseteq V$, such that for any $e = (u, v) \in E$, $u \in X$ or $v \in X$ (or both). [I am sure that you already know the definition of a simple path.]

1. **Path bin packing (PBP).**

   Input: An connected undirected graph $G = (V, E)$ where $V = \{1, 2, \ldots, n\}$. An integer bin size $C > 0$. A set of n items $I = V$, where item i has an integer size $0 < s_i \leq C$. Two designated vertices $x, y \in V$.

   Goal: Find a simple path P between x and y, and let $V_P \subseteq V$ be its vertices (a subset of V). Find a partition of the set of items V_P to bins, $B_1, B_2, \ldots, B_k$, where $\sum_{i \in B_\ell} s_i \leq C$ for any $1 \leq \ell \leq k$.

   Objective:: Minimize k.

2. **Vertex cover bin packing (VCBP).**

   Input: An undirected graph $G = (V, E)$ where $V = \{1, 2, \dots, n\}$. An integer bin size $C > 0$. A set of n items $I = V$, where item i has an integer size $0 < s_i \le C$.

   Goal: Find a vertex cover X, and let $V_{VC} \subseteq V$ be its vertices (a subset of V). Find a partition of the set of items V_{VC} to bins, $B_1, B_2, \dots, B_k$, where $\sum_{i \in B_\ell} s_i \le C$ for any $1 \le \ell \le k$.

   Objective:: Minimize k.

3. **Path scheduling (PS).**

   Input: An connected undirected graph $G = (V, E)$ where $V = \{1, 2, \dots, n\}$. Two designated vertices $x, y \in V$. A set of n jobs $J = V$, where job j has an integer processing time $p_j > 0$, and a integer number of (identical) machines $m \ge 2$.

   Goal: Find a simple path P between x and y, and let $V_P \subseteq V$ be its vertices (a subset of V). Find an assignment of the set of jobs V_P to the m machines, $J_1, J_2, \dots, J_m$.

   Objective:: Minimize $\max_{1 \le i \le m} \sum_{j \in J_i} p_j$.

4. **Vertex cover scheduling (VCS).**

   Input: An undirected graph $G = (V, E)$ where $V = \{1, 2, \dots, n\}$. A set of n jobs $J = V$, where job j has an integer processing time $p_j > 0$, and a integer number of (identical) machines $m \ge 2$.

   Goal: Find a vertex cover X, and let $V_{VC} \subseteq V$ be its vertices (a subset of V). Find a an assignment of the set of jobs V_{VC} to the m machines, $J_1, J_2, \dots, J_m$.

   Objective:: Minimize $\max_{1 \le i \le m} \sum_{j \in J_i} p_j$.