3SAT is **NP**-complete

3SAT denotes the following restriction of satisfiability.

> INSTANCE: A set of variables, U, and a set of clauses, C, such that each clause contains exactly 3 literals.
>
> QUESTION: Is C satisfiable? i.e. is there a truth assignment to the variables such that all the clauses are satisfied?

In **NP**: guess a satisfying assignment and verify that it indeed satisfies the clauses.

NP-hard:
We show $SAT \leq_m^p 3SAT$.
Suppose (U, C) is an instance of satisfiability.
We construct an instance (U', C') of 3SAT such that, C is satisfiable iff C' is satisfiable (and the reduction can be done in poly time).

Suppose $C = \{c_1, c_2, \dots c_m\}$.
For c_i, we will define C_i' and U_i' below.

$$U' = U \cup \bigcup_{1 \le i \le m} U_i'$$

$$C' = \bigcup_{1 \le i \le m} C_i'$$

If $c_i = (l_1)$, then

$U_i' = \{y_i^1, y_i^2\}$, and
$C_i' = \{(l_1 \vee y_i^1 \vee y_i^2), (l_1 \vee \neg y_i^1 \vee y_i^2), (l_1 \vee y_i^1 \vee \neg y_i^2), (l_1 \vee \neg y_i^1 \vee \neg y_i^2)\}$,
where y_i^1 and y_i^2 are NEW variables (which are not in U, and not used in any other part of the construction).

If $c_i = (l_1 \vee l_2)$, then

$U_i' = \{y_i^1\}$, and
$C_i' = \{(l_1 \vee l_2 \vee y_i^1), (l_1 \vee l_2 \vee \neg y_i^1)\}$,
where y_i^1 is NEW variable (which is not in U, and not used in any other part of the construction).

If $c_i = (l_1 \vee l_2 \vee l_3)$, then

$U_i' = \emptyset$, and
$C_i' = \{c_i\}$.

If $c_i = (l_1 \vee l_2 \vee \cdots \vee l_r)$, where $r \geq 4$, then

$U_i' = \{y_i^1, \cdots, y_i^{r-3}\}$, and
$C_i' = \{(l_1 \vee l_2 \vee y_i^1), (\neg y_i^1 \vee l_3 \vee y_i^2), \cdots, (\neg y_i^{r-4} \vee l_{r-2} \vee y_i^{r-3}), (\neg y_i^{r-3} \vee l_{r-1} \vee l_r)\}$,
where $y_i^1, \cdots, y_i^{r-3}$, are NEW variables (which are not in U, and not used in any other part of the construction).

Clearly the transformation can be done in polynomial time.
We claim that C is satisfiable iff C' is satisfiable.

Suppose C is satisfiable. Fix a satisfying assignment of C. We give a corresponding satisfying assignment of C'.
Variables from U: same truth value as in the satisfying assignment of C.
Other variables are given truth values as follows.
(a) $|c_i| \leq 3$: variables in U'_i are assigned arbitrary truth value (clauses in C'_i are already satisfied).
(b) $|c_i| > 3$:
Suppose $c_i = (l_1, l_2, \cdots, l_r)$. Let l_j be such that l_j is true in the satisfying assignment of C fixed above.
Then let y_i^k be true for $1 \leq k \leq j-2$, and y_i^k be false for $j-2 < k \leq r-3$. It is easy to verify that all the clauses in C'_i are satisfied.

Now suppose C' is satisfiable. Fix a satisfying assignment of C'. Then we claim that the truth assignment of U' restricted to U must be a satisfying assignment for C.
To see this suppose $c_i = (l_1 \vee \ldots \vee l_r)$.
(a) $r \leq 3$: then c_i is clearly true due to construction.
(b) $r > 3$:
If y_i^{r-3} is true, then one of l_{r-1}, l_r must be true.
If y_i^1 is false, then one of l_1, l_2 must be true.
Otherwise pick a k such that y_i^k is true but y_i^{k+1} is false. (Note that there must exists such a k). Then l_{k+2} must be true.

Hence C is satisfiable iff C' is satisfiable.
This completes the proof of 3SAT being **NP**-complete.

3 Dimensional Matching is **NP**-complete

3DM is in **NP**:

To see that 3DM is in **NP** consider the following machine M. Suppose three disjoint sets, X, Y, Z, each of size n, and $S \subseteq X \times Y \times Z$ are given as input to M.

M first "guesses" a subset S' of S of size n. Then M accepts iff S' is a matching.

Clearly M witnesses that 3DM is in **NP**.

3DM is **NP**-hard:

We show that 3SAT $\leq_m^p$ 3DM.

Let $U = \{u_1, \ldots, u_n\}$ be the set of variables and $C = \{c_1, \ldots, c_m\}$ be the set of clauses of an instance of 3SAT.

We construct an instance X, Y, Z, S of 3DM such that C is satisfiable iff S contains a matching.

Construction can be done in polynomial time.

Let $X = \{t_i[j] : 1 \leq i \leq n \text{ and } 1 \leq j \leq m\} \cup$
$\{f_i[j] : 1 \leq i \leq n \text{ and } 1 \leq j \leq m\}$.

Let $Y = A \cup S_1 \cup G_1$, where

$$A = \{a_i[j] : 1 \leq i \leq n \text{ and } 1 \leq j \leq m\},$$
$$S_1 = \{s_1[j] : 1 \leq j \leq m\}, \text{ and}$$
$$G_1 = \{g_1[j] : 1 \leq j \leq m(n-1)\}.$$

Let $Z = B \cup S_2 \cup G_2$, where

$$B = \{b_i[j] : 1 \leq i \leq n \text{ and } 1 \leq j \leq m\},$$
$$S_2 = \{s_2[j] : 1 \leq j \leq m\}, \text{ and}$$
$$G_2 = \{g_2[j] : 1 \leq j \leq m(n-1)\}.$$

Let $S = G \cup (\cup_{1 \le j \le m} E_j) \cup (\cup_{1 \le i \le n} V_i^1) \cup (\cup_{1 \le i \le n} V_i^2)$, where

$V_i^1 = \{(f_i[j], a_i[j], b_i[j]) : 1 \le j \le m\}$.
$V_i^2 = \{(t_i[j], a_i[j{+}1], b_i[j]) : 1 \le j < m\} \cup \{(t_i[m], a_i[1], b_i[m])\}$.
$E_j = \{(t_i[j], s_1[j], s_2[j]) : u_i \text{ appears in } c_j\} \cup \{(f_i[j], s_1[j], s_2[j]) : \neg u_i \text{ appears in } c_j\}$.
$G = \{(t_i[j], g_1[k], g_2[k]), (f_i[j], g_1[k], g_2[k]) : 1 \le i \le n \text{ and } 1 \le j \le m \text{ and } 1 \le k \le m * (n-1)\}$.

We will show later that S has a matching iff C is satisfiable.
Intuition: The set S contains three portions,
(1) $(\cup_{1 \le i \le n} V_i^1) \cup (\cup_{1 \le i \le n} V_i^2)$: truth assignment portion,
(2) $(\cup_{1 \le j \le m} E_j)$: satisfaction testing portion, and
(3) G: garbage collection portion.

Truth assignment Portion

Fix i. Note that if $S' \subset S$ "covers" all of $a_i[j], b_i[j]$, $1 \leq j \leq m$, exactly once then either
(a) S' contains all of V_i^1 and none of V_i^2 OR
(a) S' contains all of V_i^2 and none of V_i^1.
This can be considered as assigning a "truth" value to the variable u_i.
We used $t_i[1], \ldots, t_i[m]$ (and correspondingly $f_i[1], \ldots, f_i[m]$) instead of just using t_i, f_i to give "fan out" of m for the variable u_i, so that one can use different copies in different clauses (see below).

Satisfaction Testing Portion

Note that if $S' \subseteq S$ "covers" $s_1[j], s_2[j]$ exactly once then S' contains exactly one element from E_j. Intuitively, this gives us the literal in c_j which must be "TRUE".

Garbage Collection Portion

The elements of G are essentially for garbage collection.
Note that we had a fan out of m for each variable (giving us a total of $m * n$ "truth items").
However only m instances of these are used in the Satisfaction testing component.
Thus we need to do a garbage collection for remaining $m * (n - 1)$ elements.
This is what G is used for.

We now show that C is satisfiable iff S contains a matching.
Suppose S has a matching S'. Then we claim that an assignment of u_i being true iff $V_i^1 \subseteq S'$ shows that C is satisfiable.

Suppose C is satisfiable. Fix a satisfying assignment $t : U \to \{T, F\}$.
For each j, suppose C_j is true due to u_i being true (false).
Let $w_j[j]$ denote $t_i[j]$ ($f_i[j]$ respectively).
The matching is formed by taking the following three subsets of S.
(1) $\cup_{t(i)=T} V_i^1 \cup \cup_{t(i)=F} V_i^2$.
(2) $\{(w_j[j], s_1[j], s_2[j]) : 1 \leq j \leq m\}$.
(3) G',
where G' is an appropriate subset of G, such that all the elements of $\{t_i[j], f_i[j] : 1 \leq i \leq n \text{ and } 1 \leq j \leq m\} - \{w_j[j] : 1 \leq j \leq m\}$ are covered (using each of $g_1[k], g_2[k]$, $1 \leq k \leq m(n-1)$ exactly once).

Partition is **NP**-complete

In **NP**: Suppose a set A, and corresponding sizes $s(a)$ is given. To see that Partition is in **NP**, one just needs to guess a subset A' of A and verify that $\Sigma_{a \in A'} s(a) = \Sigma_{a \in A - A'} s(a)$.

NP-hard: We show $3DM \leq_m^p Partition$.
Suppose three disjoint sets X, Y, Z of size n each, and $S \subseteq X \times Y \times Z$ is an instance of 3DM.
We construct (in polynomial time) an instance of Partition by giving set A, and $s(a), a \in A$, such that S has a matching iff there exists a subset A' of A such that $\Sigma_{a \in A'} s(a) = \Sigma_{a \in A - A'} s(a)$.

Suppose $X = \{x_1, x_2, \ldots, x_n\}$, $Y = \{y_1, y_2, \ldots, y_n\}$, and $Z = \{z_1, z_2, \ldots, z_n\}$.
Suppose S has k elements $m_1, m_2, \ldots, m_k$.
Then A will have $k+2$ elements, $a_1, \ldots, a_{k+2}$.
The elements $a_1, \ldots, a_k$ will correspond to $m_1, \ldots, m_k$ and a_{k+1}, a_{k+2} will be special elements.
If $m_i = (x_{f(i)}, y_{g(i)}, z_{h(i)})$, then
$s(a_i) = 2^{3pn-pf(i)} + 2^{2pn-pg(i)} + 2^{pn-ph(i)}$, where p is such that $2^p > k$. Intuitively one can consider the number $s(a_i)$ as being divided into $3n$ zones, each of p bits as follows.

The number $s(a_i)$ is formed by placing 1 at the rightmost bit corresponding to zones $x_{f(i)}, y_{g(i)}, z_{h(i)}$, and other bits being 0.

Important characteristic: on adding the sizes corresponding to any subset of $\{a_1, \ldots, a_k\}$, there is no "carry over" from one zone to another as long as $2^p > k$.

Thus if we let $B = \Sigma_{0 \leq j \leq 3n-1} 2^{pj}$, (which is the number formed by placing 1 in the rightmost bit of each zone),
Then any subset $A' \subseteq \{a_1, \ldots, a_k\}$ will satisfy
$\Sigma_{a \in A'} s(a) = B$, iff $\{m_i : a_i \in A'\}$ is a matching of S.

Let $s(a_{k+1}) = [2 * \Sigma_{1 \leq i \leq k} s(a_i)] - B$.
Let $s(a_{k+2}) = \Sigma_{1 \leq i \leq k} s(a_i) + B$.
Note that the number of bits needed to specify $s(a_{k+1})$ and $s(a_{k+2})$ is a polynomial in k, n.

Claim: there exists a subset $A' \subseteq A$ such that $\Sigma_{a\in A'} s(a) = \Sigma_{a\in A-A'} s(a)$ iff S has a matching.
Note that $\Sigma_{a\in A} s(a) = 4\Sigma_{1\leq i\leq k} s(a_i)$.
Suppose S has a matching S'.
Then clearly, $A' = \{a_i : m_i \in S'\} \cup \{a_{k+1}\}$, gives
$\Sigma_{a\in A'} s(a) = ([2\Sigma_{1\leq i\leq k} s(a_i)]-B)+B = 2\Sigma_{1\leq i\leq k} s(a_i) = \Sigma_{a\in A-A'} s(a)$

If there exists $A' \subseteq A$ such that
$\Sigma_{a\in A'} s(a) = 2\Sigma_{1\leq i\leq k} s(a_i)$,
then exactly one of a_{k+1} and a_{k+2} must be in A' (otherwise the sum will be $\geq 3\Sigma_{1\leq i\leq k} s(a_i)$).
Without loss of generality suppose that $a_{k+1} \in A'$.
Then $[\Sigma_{i\in A'} s(a_i)] - s(a_{k+1}) = B$.
Hence $\{m_i : a_i \in A' - \{a_{k+1}\}\}$ is a matching of S.
This shows that partition is **NP**-complete.

Multi Processor Scheduling is **NP**-complete

The multiprocessor scheduling problem is as follows:

> INSTANCE: A finite set A of *tasks*, a *length* $l(a)$ for each $a \in A$, a number m of *processors*, and a deadline D.
>
> A schedule $S = (A_1, A_2, \ldots, A_m)$ is a partition of A into pairwise disjoint sets $A_1, A_2, \ldots, A_m$. Time taken by a schedule S, denoted $Time(S)$, is $\max\{\Sigma_{a \in A_i} l(a) : 1 \leq i \leq m\}$.
>
> QUESTION: Is there a schedule S such that $Time(S) \leq D$?

Clearly, Multiprocessor scheduling problem is in **NP**(one just needs to guess a schedule, S, and verify that $Time(S) \leq D$).

We reduce Partition to Multiprocessor schedule.
Suppose a set A and size $s(a)$, for $a \in A$ is an instance of Partition problem.
Generate the instance of Multiprocessor scheduling as follows.

Let $B = \Sigma_{a \in A} s(a)$.

If B is odd then let $m = 1$, $A = \{a_1\}$, $l(a_1) = 5$, and $D = 2$.

If B is even, then generate an instance of Multiprocessor scheduling as follows:
$m = 2$.
A (of multiprocessor scheduling problem) $=$ A (of Partition problem).
$l(a) = s(a)$.
$D = B/2$.
It is easy to verify that there exists a subset $A' \subseteq A$ such that $\Sigma_{a \in A} s(a) = \Sigma_{a \in A - A'} s(a) = B/2$ iff there exists a schedule such that $Time(S) \leq D = B/2$.