P1 Reduce Subset-Sum to Scheduling.

We denote input of Subset-Sum as $\varphi = (A, t)$
(where $a_1, a_2 \cdots, a_n \in A$, and t is the target)

We denote input of Scheduling as (T, D, P, t)
where $t_1 \sim t_n \in T$ reprents processing time
$d_1 \sim d_n \in D$ reprents deadline
$p_1 \sim p_n \in P$ reprents profit
t is integer (originally denoted P) target

We give an algorithm $R(\varphi) \to (T_\varphi, D_\varphi, P_\varphi, t_\varphi)$
s.t. $\varphi \in$ Subset-Sum $\Leftarrow (T_\varphi, D_\varphi, P_\varphi, t_\varphi) \in$ Scheduling
(proof part 1)
and R runs in polynomial time (part 2)

$R(A, t)$:

for a_i in A:
$t_i \leftarrow a_i$
$d_i \leftarrow t$
$p_i \leftarrow a_i$

$T \leftarrow (t_1 \sim t_n)$
$D \leftarrow (d_1 \sim d_n)$
$P \leftarrow (p_1 \sim p_n)$
$t \leftarrow t$

return (T, D, P, t)

Proof part 1: prove that Scheduling$(R(\varphi))$ = SubsetSum(φ)

which is to prove Scheduling$(T_\varphi, D_\varphi, P_\varphi, t_\varphi)$
= Subset-Sum(A, t)

Suppose $\text{Scheduling}(R(\varphi)) \in \text{SCHEDULING}$

there exists $A' \subset A$ satisfies following:

$$\sum T'_\varphi \leq t_\varphi \text{ AND } \pi(\sigma) = \sum P'_\varphi \geq t_\varphi.$$

(by definition of π)

$$\sum T'_\varphi \leq t_\varphi \text{ AND } \sum T'_\varphi \geq t_\varphi$$

(because $T_\varphi \Leftrightarrow A \Leftrightarrow P_\varphi$)

$$\sum A' \leq t \text{ AND } \sum A' \geq t$$

$$\sum A' = t$$

Suppose $\text{Scheduling}(R(\varphi)) \notin \text{SCHEDULING}$

there does not exist such $A' \subset A$ s.t. $\sum A' = t$

$\square$ prove by definition.

Proof part 2

(prove that R run in poly. time)

$$\text{runtime}(R) = O(n) \quad \square$$

P2 Reduce 3SAT → SYSTEM

We denote input for 3SAT as $\varphi = (3CNF)$
where clauses $c_1, c_2 \cdots c_n \in 3CNF$
where $\{x_{i1}, x_{i2}, x_{i3}\} = c_i$

We denote input for SYSTEM as E
where linear inequalities $e_1, e_2, e_3 \cdots e_n \in E$

We give an algorithm $R(\varphi) \to E_\varphi$
s.t. $\varphi \in 3SAT \Leftarrow E_\varphi \in SYSTEM$ (1)
and R runs in poly. time. (2)

$R(3CNF)$:
for $(x_{i1}, x_{i2}, x_{i3}) \in 3CNF$:
$e_i \leftarrow$ "$L(x_{i1}) + L(x_{i2}) + L(x_{i3}) \geq 1$"
return $\{e_1, e_2, \ldots, e_n\} + V$

where $L(x_{ij}) = (1 - x_{ij})$ if negated(x_{ij}) else x_{ij}
where V is set of inequalities that
for x in all variables, $0 \leq x \leq 1$
(which means x can be 0 or 1)

Proof (1)

for all e_i, if e_i is satisfied, at least
one out of three $L(x_{ij})$ has to be 1
(because $0 \leq x_{ij} \leq 1 \Rightarrow 0 \leq L(x_{ij}) \leq 1$)
which means c_i is true
if all e_i is satisfied, all c_i is true
reverse also holds. $\square$

Proof (2): runtime$(R) = O(n + |X|) \Rightarrow R \in P$

P3 Reduce 3-Color → 4-Color.

We denote input for 3Color as $\varphi = (V, E)$

We denote input for 4Color as $G = (V_\varphi, E_\varphi)$

We give an algorithm $R(\varphi) \to G$
s.t. $4color(R(\varphi)) \Rightarrow 3color(\varphi)$ (1)
and R runs in poly. time (2)

$R(V, E)$:

$n \leftarrow newNode()$
$V_\varphi \leftarrow V + n$
$E_\varphi \leftarrow E + ((v, n) \text{ for } v \in V)$

return (V_φ, E_φ)

Proof (1):

Suppose $G \in 4color$,

because n is connected to all V,
all V do not have same color as n
$\Rightarrow V$ is colored with 3 colors without violation

Suppose $G \notin 4color$:
which means if we color n first and then if we try to 3-color V, we will fail.
$\Rightarrow V$ cannot be 3colored without violation

$\square$

Proof (2):
$runtime(R) = O(V) \Rightarrow R \in P$ $\square$