**Knapsack**

**KNAPSACK** is a problem which generalises many natural scheduling and optimisation problems, and through reductions has been used to show many such problems **NP-complete**.

In the problem, we are given $n$ items, each with a positive integer value $v_i$ and weight $w_i$.

We are also given a maximum total weight $W$, and a minimum total value $V$.

Can we select a subset of the items whose total weight does not exceed $W$, and whose total value exceeds $V$?
**Reduction**

The proof that KNAPSACK is NP-complete is by a reduction from the problem of Exact Cover by 3-Sets.

Given a set $U = \{1, \ldots, 3n\}$ and a collection of 3-element subsets of $U$, $S = \{S_1, \ldots, S_m\}$.

We map this to an instance of KNAPSACK with $m$ elements each corresponding to one of the $S_i$, and having weight and value

$$\sum_{j \in S_i} (m + 1)^{j-1}$$

and set the target weight and value both to

$$\sum_{j=0}^{3n-1} (m + 1)^{j}$$
Scheduling

Some examples of the kinds of scheduling tasks that have been proved \textbf{NP}-complete include:

\textbf{Timetable Design}

Given a set $H$ of \textit{work periods}, a set $W$ of \textit{workers} each with an associated subset of $H$ (available periods), a set $T$ of \textit{tasks} and an assignment $r : W \times T \rightarrow \mathbb{N}$ of \textit{required work}, is there a mapping $f : W \times T \times H \rightarrow \{0, 1\}$ which completes all tasks?
Scheduling

Sequencing with Deadlines

Given a set $T$ of tasks and for each task a length $l \in \mathbb{N}$, a release time $r \in \mathbb{N}$ and a deadline $d \in \mathbb{N}$, is there a work schedule which completes each task between its release time and its deadline?

Job Scheduling

Given a set $T$ of tasks, a number $m \in \mathbb{N}$ of processors a length $l \in \mathbb{N}$ for each task, and an overall deadline $D \in \mathbb{N}$, is there a multi-processor schedule which completes all tasks by the deadline?
Responses to NP-Completeness

Confronted by an NP-complete problem, say constructing a timetable, what can one do?

- It’s a single instance, does asymptotic complexity matter?
- What’s the critical size? Is scalability important?
- Are there guaranteed restrictions on the input? Will a special purpose algorithm suffice?
- Will an approximate solution suffice? Are performance guarantees required?
- Are there useful heuristics that can constrain a search? Ways of ordering choices to control backtracking?
Validity

We define \( \text{VAL} \)—the set of valid Boolean expressions—to be those Boolean expressions for which every assignment of truth values to variables yields an expression equivalent to \text{true}.

\[
\phi \in \text{VAL} \iff \neg \phi \notin \text{SAT}
\]

By an exhaustive search algorithm similar to the one for \text{SAT}, \text{VAL} is in \text{TIME}(n^2 2^n).

Is \( \text{VAL} \in \text{NP} \)?
Validity

\[ \overline{\text{VAL}} = \{ \phi \mid \phi \notin \text{VAL} \} \]—the complement of \( \text{VAL} \) is in \( \text{NP} \).

Guess a a falsifying truth assignment and verify it.

Such an algorithm does not work for \( \text{VAL} \).

In this case, we have to determine whether every truth assignment results in true—a requirement that does not sit as well with the definition of acceptance by a nondeterministic machine.
Complementation

If we interchange accepting and rejecting states in a deterministic machine that accepts the language $L$, we get one that accepts $\overline{L}$.

If a language $L \in \mathbf{P}$, then also $\overline{L} \in \mathbf{P}$.

Complexity classes defined in terms of nondeterministic machine models are not necessarily closed under complementation of languages.

Define,

co-$\mathbf{NP}$ – the languages whose complements are in $\mathbf{NP}$.
Succinct Certificates

The complexity class \textbf{NP} can be characterised as the collection of languages of the form:

\[ L = \{ x \mid \exists y R(x, y) \} \]

Where \( R \) is a relation on strings satisfying two key conditions

1. \( R \) is decidable in polynomial time.

2. \( R \) is \textit{polynomially balanced}. That is, there is a polynomial \( p \) such that if \( R(x, y) \) and the length of \( x \) is \( n \), then the length of \( y \) is no more than \( p(n) \).
Succinct Certificates

$y$ is a *certificate* for the membership of $x$ in $L$.

**Example:** If $L$ is **SAT**, then for a satisfiable expression $x$, a certificate would be a satisfying truth assignment.
co-NP

As co-NP is the collection of complements of languages in NP, and P is closed under complementation, co-NP can also be characterised as the collection of languages of the form:

\[ L = \{ x \mid \forall y \; |y| < p(|x|) \rightarrow R'(x, y) \} \]

NP – the collection of languages with succinct certificates of membership.

c-co-NP – the collection of languages with succinct certificates of disqualification.
Any of the situations is consistent with our present state of knowledge:

- $P = NP = co-NP$
- $P = NP \cap co-NP \neq NP \neq co-NP$
- $P \neq NP \cap co-NP = NP = co-NP$
- $P \neq NP \cap co-NP \neq NP \neq co-NP$