NP-Vollständige Probleme

Satisfiability (SAT)

geg. Ein boolescher Ausdruck in konjunktiver Normalform
ges. Werte für die Variablen des Ausdrucks, damit der Ausdruck wahr wird.

3-Satisfiability (3-SAT)

Entspricht SAT mit höchstens drei Literalen pro Klausel.

Clique

geg. Graph $G = \{V, E\}, k \in \mathbb{N}$
ges. Ein vollständiger Untergraph mit Größe k

Subgraph Isomorphism

geg. Graph $G_1 = \{V_1, E_1\}, G_2 = \{V_2, E_2\}$
ges. Die Antwort auf die Frage: Ist G_1 isomorph zu G_2?

Hamiltonian Circuit

geg. Graph $G = \{V, E\}$
ges. Kreis in G, der alle Knoten in G enthält

Travelling Salesman

geg. Graph $G = \{V, E\}, k \in \mathbb{N}$
ges. Hamiltonkreis mit "Gesamtlänge" k

Vertex Cover

geg. Graph $G = \{V, E\}, k \in \mathbb{N}$
ges. $V' \subset V$ mit $|V'| \leq k$, sodass für jede Kante $(u, v) \in E$ wenigstens einer der beiden Knoten u, v zu V' gehört.

Knapsack

geg. $a_1, a_2, ...a_n \in \mathbb{N}$ und $b \in \mathbb{N}$
Werte von n Elementen und eine Behältergröße b

ges. $I \subset \{1, 2, ..., n\}$ mit $\sum_{i \in I} a_i = b$
Die Nummern der Elemente die zusammen in ihrer Summe den Behälter "füllen".

Partition

geg. $a_1, a_2, ...a_n \in \mathbb{N}$
Werte von n Elementen

ges. $I \subset \{1, 2, ..., n\}$ mit $\sum\limits_{i \in I} a_i = \sum\limits_{i \notin I} a_i$
Die Nummern der Elemente, die zusammen in ihrer Summe genau die Hälfte der Gesamtsumme sind.

Bin Packing

geg. $a_1, a_2, ..., a_n$ $(\forall a : a_i \leq b)$ und $b, k \in \mathbb{N}$
Werte von n Elementen sowie die Anzahl und Größe der Behälter

ges. Verteilung der a_1 auf die k Behälter mit jeweils $\sum \leq b$

Polynomielle Reduktion

Die polynomielle Reduktion zweier Probleme $A \leq_p B, A, B \subset \Sigma^*$ beschreibt die Möglichkeit ein Problem A dadurch zu lösen, dass man die Vorraussetzungen des Problems, als Vorraussetzungen eines anderen Problems B interpretiert und entsprechend die Lösung von B interpretiert als Lösung des Problems A. Dies muss in polynomieller Zeit geschehen, ist jedoch nur in wenigen Fällen von tatsächlicher Relevanz, da es nur selten in nicht polynomieller Zeit geschieht.
A ist also in dem Fall eine speziellere (einfachere) oder gleiche Version des allgemeineren (schwereren) oder gleichen Problems B. Alle NP-Vollständigen Probleme lassen sich ineinander reduzieren und alle Probleme in NP lassen sich in jedes NP-Vollständige Problem reduzieren, da NP-Vollständige Probleme jedes NP-Problem lösen können, jedoch nicht umgekehrt.

Clique $\leq_p$ Subgraph Isomorphism (Clique durch SI lösen)

Unseren Graphen G des Clique-Problems setzen wir als G_1 unseres SI-Problems. G_2 setzen wir daraufhin auf einen vollständigen Graphen der Größe k.

Knapsack $\leq_p$ Partition (Knapsack durch Partition lösen)

Das Partition-Problem wird als $\{a_1, a_2, ..., a_n, M - b + 1, b + 1\}$ gegeben ($M - b + 1$ und $b + 1$ sind dabei zwei neue Elemente, die an die Liste der Elemente, die bereits vorhanden war drangehängt werden). Alles was in Knapsack gegeben ist, befindet sich damit in der Form des Partition-Problems. M ist dabei die Summe aller Elemente $\sum\limits_{i=1}^{n} a_i$.
Wird das Partition-Problem jetzt gelöst, erhalten wir eine Menge I in der die Indizes aller Elemente vorhanden sind, die mit ihrer Summe genau die Hälfte der Gesamtsumme ergeben. Die letzten beiden Elemente verraten uns, wie wir die Menge I für unser Knapsack-Problem interpretieren müssen. Partition wird alle Elemente in I enthalten haben, die zusammen die

Hälfte der Gesamtsumme $2M+2$ ergeben, also $\frac{2M+2}{2}$ ergeben. Es wird jedoch entweder das letzte oder vorletzte Element vorhanden sein, weil diese zusammen $M+2$ ergeben und das damit für Partition schon zu viel ist.
Also gibt es genau zwei Möglichkeiten:

- Der Index von $M-b+1$ ist in I: Das bedeutet, das die sonstigen Elemente in I zusammen genau b ergeben, weil $\frac{2M+2}{2} - M - b + 1 = b$ ergibt. Entfernen wir nun dieses Element also aus I haben wir damit die Lösungsmenge I, die unser Knapsack-Problem löst.

- Befindet sich der Index von $b+1$ in I dann benötigen wir eine neue Menge $I' := \{a|a \notin I\}$, also das Komplement und gehen damit genau so vor wie im vorherigen Punkt.

Beispiel

Du willst in den Urlaub fliegen und hast nur einen Koffer, in den 8kg rein passen. Du willst aber so viel wie möglich mitnehmen. Du hast vier Sachen, die 2,5,1 und 4 Kilo schwer sind. Du erinnerst dich an die Informatikvorlesung und stellst dir dein Problem als Knapsack-Problem auf: Dieses ist in deinem Fall gegeben als $\{2,5,1,4\}, b = 8$ dann suchst du also alle Elemente die zusammen 8 ergeben. Dir ist aber eingefallen, das du gar kein Algorithmus für Knapsack geschrieben hast. Nach einer Weile fällt dir aber ein, dass du das Problem auch mit deinen Partition-Algorithmus lösen kannst, den du bereits geschrieben hattest und fängst bereits an nachzudenken, wie du das anstellen kannst.
Dafür brauchst du nur noch die Summe aller Elemente $M = 2+5+1+4 = 12$ und dann stellst du dir das Partition-Problem nach dem oben beschriebenen Muster auf: $\{2, 5, 1, 4, 12-8+1, 8+1\} = \{2_1, 5_2, 1_3, 4_4, 5_5, 9_6\}$ (Entsprechend der Indizes).
Nun lässt du es durch deinen Partition-Algorithmus laufen. Dieser sucht Elemente, die in ihrer Summe die Hälfte der Gesamtsumme ergeben. Das wäre in deinem Fall die Hälfte von $2+5+1+4+5+9 = 26$ also 13.
Der Algorithmus hat in deinem Fall diese Menge I berechnet: $\{1,2,3,5\}$. Dabei handelt es sich um die **Indizes** deiner Elemente, die zusammen die Hälfte der Gesamtmenge ergeben. Diese Elemente sind also entsprechend $\{2_1, 5_2, 1_3, 5_5\}$, die zusammen auch 13 ergeben.
Das heißt, das das fünfte Element also $M-b+1 = 5$ in der Menge ist. Die Lösung deines Knapsack-Problems ist also Menge jedes Elements außer des letzten Elements, also in deinem Fall $\{2,5,1\}$, was auch gleich unserem $b = 8$ ist.
So konntest du deinen Koffer noch rechtzeitig packen und fuhrst beruhigt zum Flughafen.

Partition $\leq_p$ Knapsack (Partition durch Knapsack lösen)

Man setzt die Behältergröße des Knapsack-Problems auf die Hälfte der Summe aller Elemente $b = \frac{1}{2}\sum_{i=1}^{a_i}$. Die Lösung die man dann erhält, wenn man die Elemente entsprechend als Elemente des Knapsack-Problems interpretiert, entspricht genau der Lösung des Partition-Problems.

Partition $\leq_p$ Bin-Packing (Partition durch Bin-Packing lösen)

Man hat Partition als $\{a_1, a_2, ..., a_n\}$ vorliegen und versucht dieses Problem auf Bin-Packing zu reduzieren.
Bei Bin-Packing benötigt man eine Behältergröße, die man auf $b = \frac{1}{2}\sum\limits_{i=1}^{n} a_1$ setzt. Die Anzahl der Behälter setzt man auf $k = 2$.
Dadurch löst man das Partition-Problem, da bei zwei Behältern, die so groß sind, wie die Hälfte der Summe auch das Partition-Problem gelöst ist.