The Complete Proof of Knapsack Problem is NP-Completeness

Xinyang Wang, Yuncheng Jia, Simin Li

Computer Science, Bishop's University

I. Knapsack Problem Is in NP Class

The Knapsack problem is that given a knapsack with a weight limit W, a constant $P > 0$ and a set of n objects with weights w_i and values p_i, $1 \leq i \leq n$, such that is there a subset $N' \subseteq \{1, \ldots, n\}$ to satisfy $\sum_{i \in N'} p_i > P$ and also subject to $\sum_{i \in N'} w_i \leq W$.

To prove a problem is NP-complete, first we need to prove the problem is a member of NP (nondeterministic polynomial time) class.

NP is the class of decision problems which the solutions can be checked or verified with a deterministic algorithm, and the time complexity of the algorithm is polynomial.[1]

For the Knapsack problem, we only need to verify two values of the solutions. One is whether the weight satisfies the limit, it is not hard to know that it takes O(n) time. The other is the value, the same as the weight, it takes O(n) time as well. Thus, it takes polynomial time to verify all the possible solutions. We finish our proof that the Knapsack problem is a member of NP Class.

II. Knapsack Problem Is NP-Completeness

A language L is NP-complete[2], then L needs to be NP and at least as "hard" as all other languages L' belong to NP. We can use polynomial reduction ($\propto$) to show this relation,

$$L \in NP \wedge \forall L' \in NP : L' \propto L \Longrightarrow L \in NPC$$

By the definition and transitivity of polynomial reduction[3], if we have two languages $L_1, L_2 \in NP$, $L_1 \in NPC$, then if $L_1 \propto L_2$, L_2 is NP-complete. Therefore, we just need to find a NP-complete problem which can be reduced to the Knapsack problem. In this paper, we used the Exact Cover problem.

The following proof mostly refers to the work of Dr. Lagoudakis[4]. We define an exact cover problem with a set $S = \{s_1, s_2, \dots, s_n\}$ that contains n items and $F = \{f_1, f_2, \dots, f_m\}$ which is the collection of m subsets of S. For the Knapsack, we use W as knapsack capacity and U is a set of n objects. Then, there is an exact cover of S in F if and only if the total weight of some objects in U equal to the capacity W (i.e. $\sum_{u \in U'} w(u) = W$, where $U' \subseteq U$).

To avoid the problem caused by the carry, we choose the base as $\beta = m + 1$ instead of 2. Also, $i = 1, \dots, n$, $j = 1, \dots, m$ and

$$x_{ij} = \begin{cases} 1, & s_i \in f_j \\ 0, & s_i \notin f_j \end{cases},$$

which means if the ith item is in the subset i, we assign it with value 1, otherwise assign it with value 0. Then we define the weight of objects in U as,

$$w(u_j) = \sum_{i=1}^{n} x_{ij}\, \beta^{i-1}$$

The capacity of knapsack W is defined as,

$$W = \frac{\beta^n - 1}{\beta - 1} = \beta^0 + \beta^1 + \cdots + \beta^{n-1}$$

Clearly, we have a one-to-one correspondence from F to U and from $\{\beta^{i-1}: i = 1, \dots n\}$ to S. And the transition takes polynomial time, which is O(mn^2).

Consider the following exact cover problem, $S = \{s_1, s_2, s_3, s_4, s_5\}$, and F is given by $f_1 = \{s_1\}, f_2 = \{s_3, s_4\}, f_3 = \{s_1, s_2, s_4\}, f_4 = \{s_1, s_2\}\ and f_5 = \{s_2, s_3, s_4\}$. Then the corresponding instance of Knapsack is $U = \{u_1, u_2, u_3, u_{4,} u_5\}$, $\beta = 5$ and the weight of subsets is calculated to

$$w(u_1) = 6^0 = 1$$

$$w(u_2) = 6^2 + 6^3 = 252$$

$$w(u_3) = 6^0 + 6^1 + 6^6 = 223$$

$$w(u_4) = 6^0 + 6^1 = 7$$

$$w(u_5) = 6^1 + 6^2 + 6^3 = 258$$

$$W = \frac{6^4 - 1}{6 - 1} = 6^0 + 6^1 + 6^2 + 6^3 = 259$$

In this case, subset $u_1$and u_5 is the exact cover of S, at the same time the summation of the weight is equal to the limit W.

Suppose that the total weight of some objects in U equal to the capacity W, then each power $\{\beta^{i-1}: i = 1, \dots n\}$ must appears exactly once in $w(u)$. Because of the one-to-one correspondence from F to U and from β^{i-1} to S, there is an exact cover of S in F. Similarly, if some subsets $\mathrm{F}' \subseteq \mathrm{F}$ is an exact cover for S, then each item in S appears exactly once in F'. Due to the one-to-one correspondence relation, we have that $\sum_{u \in U'} w(u) = W$, where $U' \subseteq U$.

We finish our proof that Exact Cover is polynomial reducible to Knapsack problem and clearly Knapsack problem can reduce to 0-1 Knapsack Problem. Thus, 0-1 Knapsack Problem is NP-complete.

III. The Optimization Problem

The optimization problem of the Knapsack problem does not only satisfy the limit of the volumes of knapsack but also maximizes the total price of the objects in the knapsack.

Formally, we define a knapsack with weight limit W and n objects with weights w_i and values p_i where $1 \leq i \leq n$. Then, we want to find a subset of $N' \subseteq \{1, \ldots, n\}$ to maximize $\sum_{i \in N'} p_i$ and also subject to $\sum_{i \in N'} w_i \leq W$ at the same time.

Reference

[1] Kleinberg, Jon; Tardos, Éva (2006). *Algorithm Design* (2nd ed.). Addison-Wesley. p. 464

[2] Michael R. Garey, David S. Johnson (1979). *Computers and Intractability A Guide to the Theory of NP-Completeness* (1st ed.). W. H. Freeman. p. 37

[3] Michael R. Garey, David S. Johnson (1979). *Computers and Intractability A Guide to the Theory of NP-Completeness* (1st ed.). W. H. Freeman. p. 38

[4] Michail G. Lagoudakis (1996). *The 0–1 Knapsack Problem An Introductory Survey*. p. 4