State Aggregation and Deep Reinforcement Learning for Knapsack Problem

Reza Refaei Afshar Yingqian Zhang, Murat Firat, and Uzay Kaymak

Eindhoven University of Technology, Eindhoven, Netherlands

1 Introduction

In [1], we develop a state aggregation method for solving knapsack problems (KP) with deep reinforcement learning (DRL). Although handcrafted heuristics work well in many COPs, they mostly rely on the nature of problems and they need to be revised for different problem statements. In this paper, we aim to learn and improve the handcrafted heuristics to improve the quality of the solutions. We study *knapsack problem (KP)*, and we propose a state aggregation method to shrink state space in order to solve larger KP instances. A tabular RL method is used to learn the best aggregation strategy for each item. This aggregated features reduces the state space by reducing the number of unique values. Then, *Advantage Actor Critic (A2C)* algorithm as a powerful method of Deep Reinforcement Learning (DRL) is employed to learn the policy of selecting items. The proposed method solves KP by successive item selections and placing them in the knapsack, each is done by following a greedy or softmax algorithm on the output of the policy network. The experimental results show that the method obtains close to optimal solutions for three different types of instances with up to 500 items

2 Proposed method

Figure 1 shows the overview of our method. It consists of two components. Algorithm 1 includes a formulation of KP to MDP, which is solved using a DRL approach. Algorithm 2 is a state aggregation method, which learns an aggregation policy to discretize states that serve as inputs to DRL.

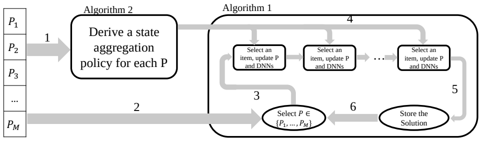
P_1
P_2
P_3
...
P_M
1
2
Algorithm 2
Derive a state aggregation policy for each P
Algorithm 1
4
Select an item, update P and DNNs
Select an item, update P and DNNs
...
Select an item, update P and DNNs
3
5
6
Select $P \in \{P_1, ..., P_M\}$
Store the Solution

Fig. 1: The overview of the KP solver.

DRL knapsack Solver: In order to solve the 0-1 KP, DRL is used to derive a policy through that the items are sequentially added to the solution. The states, actions and rewards of DRL modeling are as follows. *States $s(P)$:* A complete set of information of an instance containing the values and the weights of items and capacity of knapsack. *Actions:* There are N

Table 1: Results of different algorithms and datasets of $M = 1000$ instances.

Dataset	Method	N	$\overline{Val}$	$\#_{opt}$	$\overline{Val}_{opt}$
FI	Greedy	500	111.68	204	111.73
	DRL w/o aggregation	500	111.63	64	
	DRL w/ aggregation	500	**111.70**	**261**	
HI	Greedy	500	80779.23	25	81103.99
	DRL w/o aggregation	500	81022.60	71	
	DRL w/ aggregation	500	**81064.99**	**136**	

actions, each corresponding to select one item. *Reward Function:* The reward function contains three terms: a positive reward for successfully selecting an item; A large negative reward when the item does not exist (in case when the number of items is lower than the selected item id); and a small negative number when an item is heavier than the remaining capacity of knapsack. Employing these definitions of states, actions and rewards, the A2C algorithm is used for training policy and value DNNs.

State Aggregation: As the number of items increases, the state space grows up exponentially and this affects the performance of function approximation with DNN. In order to shrink the state space and boost the method to have the capability of solving large problem instances, a new state embedding is derived by state aggregation. Specifically, the problem is to find a certain number of split points on the values of items and transform the values into integers using these split points. We opt for reinforcement learning to tackle this problem and Q-Learning is used to find the optimal number of split points for each item value.

3 Experiments

The proposed DRL with aggregation algorithm is compared with (1) greedy algorithm, (2) DRL without aggregation (3) DRL approach with pointer network (4) Pointer Network and Supervised learning method. We use three different types of instances in the experiments: *Random Instances (RI)*, *Fixed capacity Instances (FI)* and *Hard Instances (HI)*. The DNNs consist of two layers of 64 nodes. We evaluate the performance based on several metrics. The Average Value of Solutions ($\overline{Val}$) and Number of optimally solved instances ($\#_{opt}$) for FI and HI are shown in table 1. The results show that the proposed methods, with or without aggregation, outperform the greedy algorithm. As shown in table 1, the state aggregation strategy improves the solutions of greedy algorithm for large instances. For FI instances, our method finds close to optimal solutions for instances up to 500 items. In the literature, the authors did not test instances with more than 200 items.

References

1. Refaei Afshar, Reza, Yingqian Zhang, Murat Firat, and Uzay Kaymak. 2020. “A State Aggregation Approach for Solving Knapsack Problem with Deep Reinforcement Learning.” In Proceedings of the 12th Asian Conference on Machine Learning.