A Relative Value Function Based Learning Beam Search for Longest Common Subsequence Problems*

M. Huber and G. R. Raidl

Algorithms and Complexity Group,
Institute of Logic and Computation, TU Wien, Austria
{mhuber, raidl}@ac.tuwien.ac.at

Beam search (BS) is a well-known graph search algorithm frequently used to heuristically find good or near-optimal solutions to combinatorial optimization problems in reasonable time. Starting from an initial state r, BS traverses a state graph in a breadth-first-search manner seeking a best path from r to a goal state t. To keep the computational effort within limits, BS selects at each level only up to β most promising states to pursue further and discards the others. The selected subset of states is called the *beam* and parameter β the *beam width*. To do this selection, each state v obtained at a level is usually evaluated by a function $f(v) = g(v) + h(v)$, where function $g(v)$ represents the length of the path from the root state to state v and $h(v)$, called the heuristic guidance, is an estimate of the best achievable length-to-go from node v to the goal state. The states with the best values according to this evaluation then form the beam.

Clearly, the quality of the solution BS obtains in general depends fundamentally on the heuristic guidance function h. This function needs to be developed carefully for a problem at hand, which is typically a manual and quite time-consuming process requiring problem-specific knowledge and involving many computational experiments and comparisons of different options. We propose to automate this task at least partially by using a machine learning (ML) model as guidance function h, which is trained on a large number of representative randomly generated problem instances in a reinforcement learning manner.

The overall approach is inspired by the way Silver et al. [2] mastered chess, shogi, and Go with AlphaZero. In their approach a neural network is trained to predict the outcome of a game state, called *value*, as well as to provide a *policy* for the next action to perform, and this network is used within a Monte Carlo Tree Search as guidance. Training samples are obtained by means of self-play.

In a previous work [1] we already described a *Learning Beam Search* (LBS) framework following the above concept. In it, an ML model is trained to approximate the expected length-to-go from a state to the target state, which essentially corresponds to a *value*-prediction in the terminology of reinforcement learning. Training samples are obtained from a subset of randomly chosen states, for which a *Nested Beam Search* (NBS) is performed to get target values. Training is interleaved with sample generation. This approach was investigated on the

* This project is partially funded by the "Vienna Graduate School on Computational Optimization", Austrian Science Foundation (FWF), grant W1260-N35.

well-known Longest Common Subsequence (LCS) problem and a constrained variant thereof and yielded new state-of-the-art results for some benchmark instances. In the LCS problem, a set of m input strings on an alphabet Σ is given. A subsequence is obtained from a string by removing zero or more letters from the string. The LCS problems seeks a solution string that is a longest possible subsequence of all input strings. In the Constrained LCS, a pattern string is additionally given that needs to appear in the solution string as a subsequence.

Despite the success of this former LBS, we also recognized weaknesses:

(a) Individual ML models need to be trained for different numbers of input strings m, original input string lengths, alphabet sizes, and letter distributions in the strings. Thus, the approach is not flexible to handle instances differing in any of these aspects with a common model.

(b) Within the BS, the *absolute* values of the lengths-to-go of the candidate states at a current level are actually not that relevant. More important are the *differences* among them as they already determine an ordering and therefore the beam selection. Note that the absolute values may be relatively large in comparison to the differences, and a ML model trained to predict the absolute values may therefore make stronger errors in respect to small differences.

We are addressing these weaknesses in the current work by the so-called *Relative Value Function Based LBS* (RV-LBS). In it, an ML model is trained to predict a relative value indicating the difference of the expected solution length from a given state represented by a feature vector to the expected average solution length from all nodes at the current level. Thus, by evaluating solutions in relation to other states at a current BS level we expect to obtain a more precise ranking of the states. To support the training of such a model, we also provide now different features as input, among which is a compact fixed-size encoding of the distribution of differences of remaining string lengths to the average string length at the current level. For obtaining training samples, we now select BS levels randomly and again apply NBS to obtain absolute values, from which are then the new target values derived.

Detailed experiments are still in work progress, but early results clearly indicate the benefits of RV-LBS. In particular, a model trained for one specific number of input strings and with only rather small input string lengths now generalizes well to problem instances with less or more strings which may even be substantially longer.

References

1. Huber, M., Raidl, G.R.: Learning beam search: Utilizing machine learning to guide beam search for solving combinatorial optimization problems. In: Machine Learning, Optimization, and Data Science – 7th International Conference, LOD 2021. LNCS, vol. 11943. Springer (2021), to appear
2. Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., Lillicrap, T., Simonyan, K., Hassabis, D.: A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. Science **362**(6419), 1140–1144 (2018)