DLS-PBO: a sequential local search solver in the PBO Competition 2024

Zhihan Chen[1,2], Peng Lin[1,2], Yi Chu[3], Shaowei Cai[1,2]
[1]Key Laboratory of System Software (Chinese Academy of Sciences) and State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing, China
[2]School of Computer Science and Technology, University of Chinese Academy of Sciences, Beijing, China
[3]Institute of Software, Chinese Academy of Sciences, China
{chenzh, linpeng, chuyi, caisw}@ios.ac.cn

***Abstract*—This document describes a sequential Pseudo-Boolean Optimization (PBO) solver *DLS-PBO*, submitted to the DEC-LIN and OPT-LIN tracks of the PBO competition 2024[1].**

A representative local search solver for PBO is LS-PBO [1], and it serves as the basis of other local search solvers. LS-PBO mainly incorporates two key ideas *Constraint Weighting Scheme* and *Scoring Functions*.

Firstly, LS-PBO transforms the objective function $min : \sum_i c_i \cdot l_i$ into soft objective constraint $\sum_i c_i \cdot l_i < obj^*$ (with other constraints defined as hard constraints). For each constraint c, LS-PBO assigns an integer weight $w(c)$ (which is initialized to 1). Whenever the local search process is stuck in a local optimum, the weights of unsatisfied constraints are increased, guiding the search to find better solutions (increasing the weight of hard clauses helps find feasible solutions, while increasing the weight of soft clauses helps find solutions with better objective values).

Besides, LS-PBO assigns a score to each variable to guide the search process. The scoring function usually measures the benefits of flipping a Boolean variable. In each step of the search, the variable with the largest score is picked to flip. Specifically, the score of flipping a variable v (denoted as $score(v)$) in LS-PBO was defined as follow:

$$score(v) = hscore(v) + oscore(v) \tag{1}$$

where $hscore(v)$ indicates the decrease of the total penalty of falsified hard constraints caused by flipping v, and $oscore(v)$ indicates the decrease of the penalty of the objective constraint caused by flipping v.

However, the ratio of the soft and hard constraints are the same and are not adjusted during the search process in LS-PBO, which is not reasonable. Suppose a feasible solution cannot be found within a certain period of time, in that case, the search mechanism should adaptively prioritize finding feasible solutions, thereby increasing the ratio attributed to the hard constraints. Conversely, if feasible solutions have been frequently found recently, then it would be beneficial to increase the ratio of the soft constraints to guide the search toward 0 as0 ddiscovering better solutions with higher-quality objective values.

Therefore we introduce a new *dynamic scoring function*, denoted as $score^*(v)$, which is defined as follows:

$$score^*(v) = hscore(v) + p \cdot oscore(v) \tag{2}$$

where p is a dynamic ratio initially set as 0 (guild the search to find feasible solutions at first). It would be decreased as p/inc (where $inc > 1$) if no feasible solution is found during the recent R steps, to guide the search towards feasible solutions. Otherwise, it would be increased as $p \cdot inc$ when a feasible solution is found within the recent R steps, to guide the search process for better solutions.

References

[1] Z. Lei, S. Cai, C. Luo, and H. H. Hoos. Efficient local search for pseudo boolean optimization. In C. Li and F. Manyà, editors, *Theory and Applications of Satisfiability Testing - SAT 2021 - 24th International Conference, Barcelona, Spain, July 5-9, 2021, Proceedings*, volume 12831 of *Lecture Notes in Computer Science*, pages 332–348. Springer, 2021.

[1]Relevant work has been accepted by the 30th International Conference on Principles and Practice of Constraint Programming (CP 2024).