Efficient Local Search for Pseudo Boolean Optimization

Zhendong Lei[1,2], Shaowei Cai[1,2], Chuan Luo[3], and Holger Hoos[4]

1 State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, China

2 School of Computer Science and Technology, University of Chinese Academy of Sciences, China

3 Microsoft Research, China

4 Leiden University

Outline

- Pseudo Boolean Optimization (PBO)
- Related Work
- Local Search Algorithm ---- *LS-PBO*
- Experiment Results

Preliminaries

- **Linear pseudo Boolean (LPB) constraint:**
  - $a_1 l_1 + a_2 l_2 + \cdots + a_n l_n \geq k$, $\boldsymbol{a_i}$, $k \in N^+$, $\boldsymbol{l_i} \in \{\boldsymbol{x_i}, \neg \boldsymbol{x_i}\}$, $\boldsymbol{x_i} \in \{\boldsymbol{0}, \boldsymbol{1}\}$
- **Cardinality constraints:**
  - $l_1 + l_2 + \cdots + l_n \geq k$, $k \in \mathbb{Z}$, $\boldsymbol{l_i} \in \{\boldsymbol{x_i}, \neg \boldsymbol{x_i}\}$, $\boldsymbol{x_i} \in \{\boldsymbol{0}, \boldsymbol{1}\}$
- **CNF clause:**
  - $\{l_1, l_2, \cdots, l_n\}$ $\leftarrow\rightarrow$ $\sum_{i=1}^{i=n} l_i \geq 1$

Pseudo Boolean Optimization (PBO)

- Linear pseudo Boolean Constraints:
  - $a_1 l_1 + a_2 l_2 + \cdots + a_n l_n \geq k$, $\boldsymbol{a_i}$, k ∈ $\mathbb{Z}$, $\boldsymbol{l_i} \in \{\boldsymbol{x_i}, \neg \boldsymbol{x_i}\}$, $\boldsymbol{x_i} \in \{\boldsymbol{0}, \boldsymbol{1}\}$

- Objective Function:
  - Minimize : Z = $c_1 l_1 + c_2 l_2 + \cdots + c_n l_n$, $c_{\boldsymbol{i}} \in \mathbb{Z}$

- Complete assignment: var(F) → {0, 1}
- Feasible assignment: satisfies all constraints
- Value of the objective function of a feasible solution α: obj(α)

Pseudo Boolean Optimization (PBO)

- Expressive Power > Cardinality constraint and CNF clause

- Can be used to model a large range of real-world problems:
  - Operations Research, Economics, Manufacturing

Related Work

- Based on ideas from conflict-driven clause learning (CDCL) SAT solvers
  - OpenWBO [Martins etc. 2014], RoundingSAT [Elffers etc. 2018], HYBRID [Devriendt etc. 2021]
- Branch and Bound Methods
  - Maximum Independent Set [Coudert etc. 1995], Maximum Independent Set [Liao etc. 1997]
- Translated into SAT
  - quite efficient [Sakai etc. 2015]
- These complete methods may fail for very large instances !!!

Local Search (LS) Algorithm

- Incomplete method
- A popular approach to NP-hard combinatorial problems
- Literature on LS algorithms for handling PBO is very sparse!!!

Local Search Algorithm -- *LS-PBO*

- *LS-PBO* contains two main ideas:
  - Constraint Weighting
  - Scoring Function

Main Ideas -- Constraint Weighting

- A PBO instance：
  - Goal：

    Min Z = $c_1 l_1 + c_2 l_2 + \cdots + c_n l_n$

    → **Objective** constraints:

    $c_1 l_1 + c_2 l_2 + \cdots + c_n l_n < obj^*$

    (the objective value of the best solution found)

  - LPB constraints:

    $a_{11} l_1 + a_{12} l_2 + \cdots + a_{1n} l_n \geq k$

    ……

    → **Hard (original)** constraints

Main Ideas -- Constraint Weighting

- Constraint Weighting works as follows:
  1. For each constraint (hard & objective constraints) ***c*** :
     associate ***w(c)*** as its weight, which is initialized to 1

  2. Whenever a "stuck" situation is observed (local optimal), then clause weights are updated as follows:
     - For each falsified hard constraint **c**, ***w(c) := w(c)+1***
     - If the objective constraint ***oc*** is unsatisfied, and $\boldsymbol{w(oc) \leq \xi}$, ***w(oc) := w(oc)+1***

Hard constraint weighting helps to identify those difficult hard Constraints that are usually falsified in local optimal.

Objective constraint weighting help guide the search towards solutions with better objective values.

To find a feasible solution, the weight of the objective constraint should not be too large.

Main Ideas -- Scoring Function

- If a hard constraint ***c*** ($\sum_{i=1}^{i=n} a_i\, l_i \geq \mathrm{k}$) is unsatisfied ($\sum_{i=1}^{i=n} a_i\, l_i < \mathrm{k}$)
  - Incur a penalty of $w(c) * (k - \sum_{i=1}^{i=n} a_i\, l_i)$
- For objective constraint oc, no matter weather it is satisfied or not,
  - Incur a penalty of $w(oc) * \sum_{i=1}^{i=n} c_i\, l_i$

- Hard score of a variable ***x*** (***hscore(x)***)
  - the decrease of the total penalty of unsatisfied hard constraints caused by ***flipping x***
- Objective score of a variable ***x*** (***oscore(x)***)
  - the decrease of the penalty of the objective constraint caused by ***flipping x***

- The score of a variable x is defined as *score(x) := hscore(x) + oscore(x)*

Local Search Algorithm -- *LS-PBO*

Algorithm 1: LS-PBO

```
Input: PBO instance F, cutoff time cutoff
Output: A solution α of F and its objective value
begin
    α* := ∅,   obj* := +∞;
    α := all variables are set to 0;
    while elapsed time < cutoff do
        if α is feasible and obj(α) < obj* then  α* := α; obj* := obj(α) ;
        if D := {x|score(x) > 0} ≠ ∅ then
            x := a variable in D with the highest score;
        else
            update constraint weights using Weighting-PBO;
            if ∃ unsatisfied hard constraints then
                c := a randomly chosen unsatisfied hard constraint;
                x :=the variable with highest score in c;
            else
                x :=a randomly chosen variable with oscore(x) > 0;
        α := α with x flipped;
    return (α*, obj*)
```

Experiments Evaluation

- Competitors:
  - **PBO solvers:** Open-WBO, HYBRID
  - **MaxSAT solvers:** Loandra, SATLike-c
  - **ECNF solvers:** LS-ECNF
  - **ILP solvers:** Gurobi

Benchmarks

- Three real-world application benchmarks
  - Minimum-Width Confidence Band Problem
  - Wireless Sensor Network Optimization Problem
  - Seating Arrangements Problem

- Pseudo-Boolean Competition Benchmark

Empirical results- Minimum-Width Confidence Band

Table 1. Empirical results on MWCB, using a 300s time limit.

Instance	*LS-PBO*	*LS-ECNF*	*Loandra*	*HYBRID*	*Gurobi*	
n m k	min[median,max]	min[median,max]			comp	heur
1000_200_90	**110877**[+1137, +2678]	115437[+688, +1797]	145826	168706	178806	178806
1000_250_90	**148419**[+1728, +3434]	154520[+810, +1773]	212839	229951	225930	225930
1200_200_90	**112315**[+1755, +39538]	116215[+1299, +40078]	181602	223161	220532	220532
1200_250_90	**152635**[+1292, +3085]	156652[+2378, +3361]	258986	294630	292139	292139
1400_200_90	**112449**[+1697, +43271]	116437[+880, +43576]	162754	224998	221419	221419
1400_250_90	**152348**[+2055, +3372]	157077[+1432, +2976]	224473	286857	290957	290957
1600_200_90	**138877**[+3330, +17492]	150257[+2862, +11811]	N/A	353560	353637	353637
1600_250_90	**190110**[+10081, +21720]	200335[+4720, +11411]	N/A	449511	444099	444099
1800_200_90	**226605**[+5755, +12123]	237681[+5843, +12136]	325357	378119	371792	371792
1800_250_90	**286398**[+6610, +14552]	296513[+5038, +13063]	N/A	472753	466396	466396
2000_200_90	**251293**[+4628, +49657]	260974[+6095, +48602]	N/A	393500	386950	386950
2000_250_90	**319214**[+4682, +8080]	324478[+8252, +14350]	N/A	483632	484738	484738
1000_200_95	**117375**[+935, +2624]	124137[+693, +1842]	149161	154645	175815	131435
1000_250_95	**157216**[+1628, +3041]	165082[+1079, +1427]	208022	204125	226035	226035
1200_200_95	**118988**[+1030, +41269]	126289[+1327, +40220]	171594	189875	222200	153473
1200_250_95	**160248**[+1535, +2384]	169527[+1326, +2544]	202194	270289	210573	292950
1400_200_95	**119772**[+459, +42860]	126961[+750, +45110]	169947	208118	223483	223483
1400_250_95	**162509**[+960, +2317]	170748[+1427, +2105]	199947	276115	291315	291315
1600_200_95	**185417**[+7263, +20133]	196546[+3490, +8452]	276634	336499	349746	349746
1600_250_95	**239321**[+3937, +16837]	254685[+2948, +8332]	388998	442173	446997	446997
1800_200_95	**253976**[+3498, +7565]	260176[+2906, +5323]	329055	368134	371603	371603
1800_250_95	**318906**[+2578, +8154]	325120[+2296, +6345]	420992	460488	465933	465933
2000_200_95	**277757**[+3111, +49303]	278487[+2383, +52978]	N/A	375494	387405	387405
2000_250_95	**343670**[+5656, +11008]	349308[+3499, +6921]	N/A	491377	484636	484636

Empirical results- Wireless Sensor Network Optimization

Table 3. Empirical results on WSNO, using a 300s time limit.

Instance	*LS-PBO*	*LS-ECNF*	*SATLike-c*	*HYBRID*	*Gurobi*	
n m k	min[median,max]	min[median,max]	min[median,max]		comp	heur
100_40_4	**210**[**+0**, +4]	**210**[+2, +6]	741[+15, +44]	**210**	**210**	**210**
150_60_4	**602**[**+0**, **+0**]	605[N/A, N/A]	1063[+71, +93]	**602**	1180	1180
200_80_4	**715**[**+0**, +10]	726[N/A, N/A]	N/A[N/A, N/A]	1767	1911	1911
250_100_4	**1305**[**+0**, +433]	2200[N/A, N/A]	N/A[N/A, N/A]	2123	2200	2200
300_120_4	**1257**[+32, +1315]	2572[N/A, N/A]	N/A[N/A, N/A]	2510	2572	2572
350_140_4	**1737**[+206, +1426]	3163[N/A, N/A]	N/A[N/A, N/A]	3137	3163	3163
400_160_4	**2240**[+644, +1296]	N/A[N/A, N/A]	N/A[N/A, N/A]	3509	N/A	N/A
450_180_4	**1869**[+931, +2172]	N/A[N/A, N/A]	N/A[N/A, N/A]	4026	N/A	N/A
500_200_4	**3727**[+886, +886]	N/A[N/A, N/A]	N/A[N/A, N/A]	4613	N/A	N/A
100_40_6	**140**[**+0**, +4]	**140**[+4, +9]	363[+39, +119]	**140**	**140**	**140**
150_60_6	**402**[**+0**, +1]	787[+0, N/A]	727[+30, +53]	**402**	709	709
200_80_6	**477**[**+0**, +8]	504[N/A, N/A]	N/A[N/A, N/A]	911	1274	1274
250_100_6	**870**[**+0**, +89]	1467[+0, +0]	N/A[N/A, N/A]	1299	1467	1467
300_120_6	**839**[**+0**, +876]	1715[+0, +0]	N/A[N/A, N/A]	1580	1715	1715
350_140_6	**1158**[+114, +951]	2109[+0, +0]	N/A[N/A, N/A]	2075	2109	2109
400_160_6	**1493**[**+0**, +864]	2357[+0, +0]	N/A[N/A, N/A]	2340	2357	2357
450_180_6	**1246**[+543, +1448]	2694[+0, N/A]	N/A[N/A, N/A]	2670	N/A	N/A
500_200_6	**1784**[+1291, +1291]	3075[N/A, N/A]	N/A[N/A, N/A]	3075	N/A	N/A

Empirical results - Seating Arrangements

Table 5. Empirical results on SAP, with 300s and 3600s time limits.

Instance	*LS-PBO*	*LS-ECNF*	*SATLike-c*	*HYBRID*	*Gurobi*	
n	min[median,max]	min[median,max]	min[median,max]		comp	heur
TimeLimit=300s						
100	**582**[+4, +9]	606[+14, +30]	N/A[N/A, N/A]	N/A	688	759
110	**623**[+8, +12]	668[+14, N/A]	N/A[N/A, N/A]	N/A	841	841
120	**680**[+10, +13]	698[+8, +12]	N/A[N/A, N/A]	N/A	N/A	N/A
130	**745**[+5, +9]	761[+10, +14]	N/A[N/A, N/A]	N/A	N/A	N/A
140	**762**[+8, +13]	791[+8, +15]	N/A[N/A, N/A]	N/A	N/A	N/A
150	**829**[+5, +10]	845[+10, +16]	N/A[N/A, N/A]	N/A	N/A	N/A
160	**873**[+6, +13]	882[+18, +25]	N/A[N/A, N/A]	N/A	N/A	N/A
170	**907**[+7, +14]	932[+8, +16]	N/A[N/A, N/A]	N/A	N/A	N/A
180	**975**[+10, +14]	994[+20, +28]	N/A[N/A, N/A]	N/A	N/A	N/A
190	**1005**[+10, +17]	1028[+14, +20]	N/A[N/A, N/A]	N/A	N/A	N/A
200	**1066**[+16, +21]	1096[+17, +26]	N/A[N/A, N/A]	N/A	N/A	N/A
210	**1110**[+11, +16]	1145[+10, +15]	N/A[N/A, N/A]	N/A	N/A	N/A
220	**1157**[+17, +26]	1195[+6, +14]	N/A[N/A, N/A]	N/A	N/A	N/A
230	**1202**[+11, +17]	1232[+11, +20]	N/A[N/A, N/A]	N/A	N/A	N/A
240	**1236**[+8, +14]	1262[+20, +28]	N/A[N/A, N/A]	N/A	N/A	N/A
250	**1289**[+12, +24]	1328[+11, +18]	N/A[N/A, N/A]	N/A	N/A	N/A
260	**1333**[+14, +22]	1358[+15, +24]	N/A[N/A, N/A]	N/A	N/A	N/A
270	**1396**[+19, +30]	1432[+19, +30]	N/A[N/A, N/A]	N/A	N/A	N/A
280	**1422**[+13, +21]	1458[+19, +29]	N/A[N/A, N/A]	N/A	N/A	N/A
290	**1473**[+12, +21]	1512[+16, +29]	N/A[N/A, N/A]	N/A	N/A	N/A
300	**1538**[+23, +31]	1582[+18, +31]	N/A[N/A, N/A]	N/A	N/A	N/A

Empirical results -Pseudo-Boolean Competition Benchmark

Table 6. Empirical results on benchmarks from the 2016 PB Competition

Benchmark	#inst.	Timelimit	*LS-OPB* score(avg)	*HYBRID* score(avg)	*Gurobi(comp)* score(avg)	*Gurobi(heur)* score(avg)
PB16	1600	300s	0.6683	0.8018	0.6762	0.6562
PB16	1600	3600s	0.7283	0.8130	0.6990	0.6859

Conclusions and Future Work

- ***LS-PBO*** is highly effective
- Can solve many real-world problems
- **Future work:**
  - more efficient local search solvers for PBO
  - additional real-world combinatorial problems.

Thanks!

Main Ideas -- Scoring Function

- Example:
  - Min Z = $100x_1 + 200x_2 + 300x_3$ w(oc) = 1
  - S.t. $2x_1 + 3x_2 + 4x_3 \geq 5$ w(c) = 2
- Given the assignment (x_1, x_2, x_3) = (1, 0, 0), ($2x_1 + 3x_2 + 4x_3$ = 2 < 5)
  - $hsore(x_1) = -2 * 2, hsore(x_2) = 2 * 3,\ hsore(x_3) = 2 * 3$
  - $osore(x_1) = 1 * 100, osore(x_2) = -1 * 200,\ osore(x_3) = -3 * 200$
- The score of a variable $\boldsymbol{x}$ is defined as $score(x) := hscore(x) + oscore(x)$