中国科学: 信息科学 2019 年 第 49 卷 第 6 期: 685–697
SCIENTIA SINICA Informationis
论文

一种结合结构特征求解诊断问题的 PMS 方法

周慧思[1,2], 欧阳丹彤[1,2], 刘梦[1,2], 田乃予[1,2], 张立明[1,2*]

1. 吉林大学计算机科学与技术学院, 长春 130012
2. 符号计算与知识工程教育部重点实验室 (吉林大学), 长春 130012
* 通信作者. E-mail: limingzhang@jlu.edu.cn

收稿日期: 2018–07–23; 接受日期: 2018–11–01; 网络出版日期: 2019–06–06

国家自然科学基金项目 (批准号: 61872159, 61672261, 61502199) 资助

摘要 部分最大可满足性问题 (partial maximum satisfiability problem, PMS) 是最大可满足性问题 (maximum satisfiability problem, MaxSAT) 的泛化问题, 在很多领域中得到广泛应用. 目前, 在工业诊断实例方面 PMS 求解仍有待改进, 在对基于随机搜索的 PMS 算法深入研究基础上, 本文首次提出一种结合结构特征的随机搜索方法 (structure characteristics partial MaxSAT, SCPMS). 首先, 依据单元传播规则结合问题结构特征逐步将 PMS 问题中硬单元子句分成两部分, 从而构造出因缺乏部分硬单元子句使得问题可满足的子问题; 提出结合结构特征的随机搜索指导策略, 对新的子问题再次利用单元传播找出原问题中的硬单元子句中硬阻塞变量, 再结合子句特征翻转相应软阻塞变量, 从而提高随机搜索的求解效率. 实验结果表明, 提出的 SCPMS 与最新的两个算法 DeciDist 和 DistUp 相比, 在基于模型诊断问题 (model-based diagnosis, MBD) 的工业实例上, SCPMS 求得的不满足软子句数有较大程度的减少.

关键词 PMS, 单元传播, 最大可满足性问题, 随机搜索, 基于模型的诊断

1 引言

SAT (satisfiability problem, SAT) 问题是著名的 NP 完全问题, MaxSAT (maximum satisfiability problem, MaxSAT) 问题是 SAT 问题的优化设计, PMS (partial maximum satisfiability problem, PMS) 是 SAT 和 MaxSAT 问题的泛化版本. 在现实实例中许多问题可以编码为 PMS 问题. 在 PMS 问题中, 子句被分为硬子句和软子句, 问题的目标是要找到一组变量的分配, 使得所有的硬子句全部满足, 软子句的满足个数达到最大. 在现实实例中, 将子句分为硬子句和软子句能够有效解决包含硬约束和软约束的约束问题. 目前, PMS 问题在网络路由调度问题、制定时间表问题、组合拍卖、PGA 路由和模型诊断等问题已经得到很好的应用.

引用格式: 周慧思, 欧阳丹彤, 刘梦, 等. 一种结合结构特征求解诊断问题的 PMS 方法. 中国科学: 信息科学, 2019, 49: 685–697, doi: 10.1360/N112017-00248
Zhou H S, Ouyang D T, Liu M, et al. Partial maximum satisfiability problem method combined with structure characteristics for diagnostic problems (in Chinese). Sci Sin Inform, 2019, 49: 685–697, doi: 10.1360/N112017-00248

PMS 算法主要分为完备算法和不完备算法. 完备算法能够产生一个满足所有子句的解, 但目前的完备算法在面对大规模的诊断实例时不能在有效合理的时间内得到一个最优解. 不完备 PMS 算法不能保证得到一个最优解, 但它能够在合理的时间内发现一个接近最优的解, 因此在实际的求解过程中被广泛应用. PMS 完备算法的一部分采用分支界限算法策略 [1~5], 还有一部分算法是基于迭代调用 SAT 求解器 [6~11]. 不完备的 PMS 算法主要采用局部搜索的方法. 2014 年之前, 大部分基于局部搜索的 PMS 算法是把 PMS 转码为带权重的最大可满足问题 (weighted maximum satisfiability problem, WMS), 从而用普通的 MaxSAT 局部搜索算法求解, CCLS 算法是基于局部搜索的前沿代表算法. 之后的 Dist 算法提出了一条新的研究路线, 即利用软硬子句的区别, 设计专门的 PMS 局部搜索算法. 之后, Cai 等 [12] 通过将单元子句传播及考虑硬子句和软子句的传播顺序, 提出了 Dist 算法的改进算法 DistUP 和 DeciDist, 较大程度地改进了 Partial SAT 求解器在一些随机和人工实例中的求解效果. 但在工业诊断实例上, 这两个算法依然有待改进 [12~15].

本文在充分分析 DeciDist 算法的基础之上提出一种结构特征探索的求解方法 (structure characteristics partial MaxSAT, SCPMS). 在 DeciDist 算法中搜索翻转变量的对象是全局的, 这样随机搜索的起点低, 找到最优解的可能性小. 本文给出的 SCPMS 提出了两个优化策略. 第一, 探索工业诊断问题的结构特征, 优先在硬子句中出现的变量中找出一组阻塞变量, 这些硬阻塞变量消除之后 PMS 将会变成可满足的. 第二, 为了提高随机搜索的起点, 我们结合子句特征找到一组软子句阻塞变量, 采用优先翻转软子句阻塞变量的随机搜索预处理策略, 提高了随机搜索的起点, 进而提高了 Partial SAT 求解器的求解效果.

2 预备知识

本节主要介绍本文所用到的基本知识.

2.1 SAT 问题

可满足性问题的一些基本定义如下.

定义1 (文字 [16]) 对于变量集合 $\{x_1, x_2, x_3, \ldots, x_n\}$, 文字 Q_j 是变量 x_j 或者变量的否定 $\neg x_j$.

定义2 (子句 [16]) 子句 C_i 是文字 Q_j 的析取: $Q_1 \bigvee Q_2 \bigvee \cdots \bigvee Q_n$.

定义3 (单元子句 [16]) 子句 C_i 中只包含一个文字, 这个文字可能是一个变量 x_j 或者变量的否定 $\neg x_j$.

定义4 (合取范式 (CNF) [16]) CNF 是子句的合取: $C_1 \bigwedge C_2 \bigwedge \cdots \bigwedge C_r$.

定义5 (命题可满足性问题 [17]) 对于含有 M 个布尔变量的集合, 若能找到一组变量的赋值, 使得一个给定的命题公式 Φ 的值为真, 则称该问题是可满足的 (SAT), 如果不存在这样的赋值, 则称该 SAT 问题不可满足 (UNSAT).

2.2 PMS 问题

MaxSAT 问题是 SAT 问题的扩展问题, 其目标是找到一组真值指派使得合取范式中满足的子句数最多. PMS 问题的目标是, 找到一组赋值, 使得所有的硬子句被满足, 并且满足最多数目的软子句. 下面介绍 PMS 中的相关概念.

定义6 (硬子句 [12]) 在 CNF 中必须为真的子句.

定义7 (软子句[12]) 在 CNF 中可以不为真的子句.

定义8 (单元传播[16]) 若当前子句含有某单元子句 C, 则将当前子句集中包含该单元子句的子句删除, 被删除的子句是已满足的子句, 对于含有单元子句 $\neg C$ 的子句进行单元归结, 在归结过程中若出现新的单元子句, 则将新的单元子句加入单元子句队列, 称此过程为单元传播. 当没有新的单元子句可以传播时, 单元传播过程结束.

定义9 (硬单元子句[12]) 只包含一个文字的硬子句.

定义10 (软单元子句[12]) 只包含一个文字的软子句.

在 PMS 问题中, 一些子句被定义为硬子句, 另一部分子句被定义为软子句, 当所有的硬子句全部满足, 软子句的不满足数目达到最小化时, 就称 PMS 找到了一个最优解.

2.3 PMS 算法求解 MBD 问题

定义11 (基于模型诊断 (MBD)[18]) 基于模型的诊断问题可以用一个三元组 ⟨SD, COMPS, OBS⟩ 形式来表示, 其中, 系统描述用 SD 表示, 组件集用 COMPS 来表示, 系统的观测集用 OBS 来代表, 即诊断系统的输入与输出所组成的集合. 系统中组件 c 有故障可用一个一元谓词 AB 定义于组件 c 上来表示, 即 AB(c), 真值为 1 时代表组件 c 是故障组件.

定义12 (诊断解[18]) 若存在组件集合 $\Delta \subseteq$ COMPS, 使得 SD $\cup$ OBS $\cup$ $\{$ $\neg$abnormal(c) $\mid$ $c \subseteq$ COMPS $-$ Δ $\}$ 是可满足的, 则该组件集合为一个诊断解.

定义13 (极小诊断解[18]) 若存在一个关于系统 ⟨SD, COMPS, OBS ⟩ 的一个诊断解 Δ, 且 Δ 的任何真子集都不是关于系统 ⟨SD, COMPS, OBS ⟩ 的一个诊断解, 则称该诊断解为极小诊断解.

下面以图 1 中的电路为例介绍 MBD 问题向 PMS 问题的编码, 该逻辑电路中有 6 个组件, 均为与非门, 本文图中用 $\{N_1, N_2, N_3, \ldots, N_6\}$ 来表示 6 个逻辑与非门, “7”, “8”, “9”, “10”, “11” 分别表示系统相对应的输入变量, “12”, “13” 分别表示系统的输出变量, “14”, “15”, “16”, “17” 分别表示组件内部的连接节点.

wcnf 文件中的子句是包含电路逻辑描述子句和电路观测描述子句的子句集, MBD 问题的组件变量和外部输入输出变量在 wcnf 文件中都用单元子句描述, 组件变量用软单元子句描述, 外部输入输出变量用硬单元子句描述. 子句中的变量 “1”, “2”, “3”, “4”, “5”, “6” 分别表示组件 $\{N_1, N_2, N_3, \ldots, N_6\}$. wcnf 文件中每行格式为: 一个对应系统相关描述的子句, 每行第一个数字代表子句权重, 子句的最后以 0 结尾. 该电路逻辑对应的相关 wcnf 文件如下所示.

在 wcnf 形式的文件中的系统描述用多个 wcnf 子句集合表示如下:

7 −7 −9 −14 1 0; 7 7 14 1 0; 7 9 14 1 0;

7 −9 −10 −16 2 0; 7 9 16 2 0; 7 10 16 2 0;

7 −8 −16 −15 3 0; 7 8 15 3 0; 7 16 15 3 0;

7 −14 −15 −12 4 0; 7 14 12 4 0; 7 15 12 4 0;

7 −15 −17 −13 5 0; 7 15 13 5 0; 7 17 13 5 0;

7 −16 −11 −17 6 0; 7 16 17 6 0; 7 11 17 6 0.

若当前逻辑电路输入观测点 “7”, “8”, “9”, “10”, “11” 的观测值分别为低电平、高电平、高电平、低电平、高电平, 诊断系统无故障的情况下输出观测点 “12”, “13” 的观测值应为低电平、高电

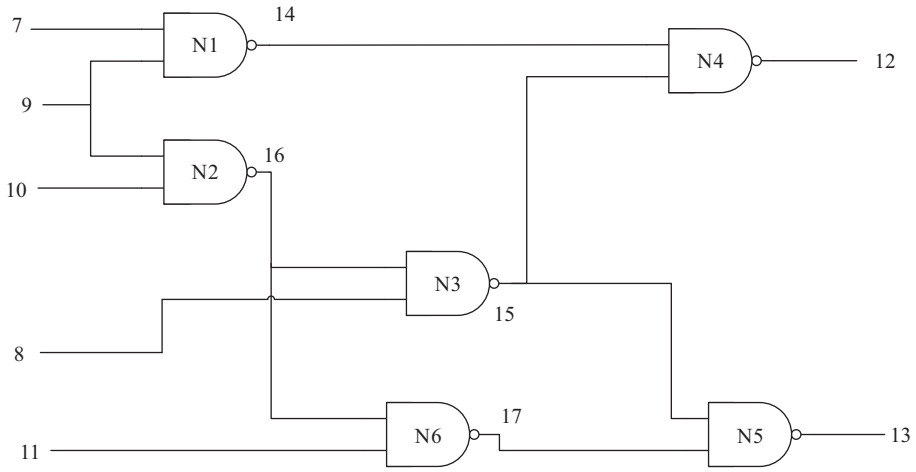

图 1　布尔电路图

Figure 1　Boolean circuit

平. 而 wcnf 文件中给出的输出观测点 “12”, “13” 的观测值为高电平、低电平, 表明该系统是故障系统. 若组件故障, 则变量值为正数, 否则为负数. 在 wcnf 文件中的观测描述和组件描述分别表示如下:

$$\begin{aligned}&7\ {-7}\ 0;\quad 7\ 8\ 0;\quad 7\ 9\ 0;\quad 7\ {-10}\ 0;\\&7\ 11\ 0;\quad 7\ 12\ 0;\quad 7\ {-13}\ 0;\quad 1\ {-1}\ 0;\\&1\ {-2}\ 0;\quad 1\ {-3}\ 0;\quad 1\ {-4}\ 0;\quad 1\ {-5}\ 0;\quad 1\ {-6}\ 0.\end{aligned}\tag{1}$$

对该 wcnf 文件求得的一个最优解即为 MBD 问题的一个极小诊断解.

3　基于随机搜索求解 PMS 问题的算法

由于 PMS 求解问题的结构特殊, 2016 和 2017 年 Cai 等 [12,15] 提出的 DistUp 和 DeciDist 算法针对一些随机和人工实例的求解效率有明显的提升, 对于工业实例特别是 MBD 问题而言, 求解效率依然有待提升. 本节主要介绍 DeciDist 的算法思想及详细描述.

3.1　DeciDist 算法的主要思想

本小节主要介绍将消元法 (decimation algorithm) [19] 和局部搜索算法 (local search algorithm) [20] 应用于求解 PMS 问题的算法: DeciDist 算法. 该算法对 Dist 算法进行了改进, 对于 MSEs 的 PMS 实例表现出较高的求解效率.

首先给出该算法涉及的一些相关概念便于对算法进行描述.

定义14 (硬分数 (hscore) [12])　翻转变量 x 后, 被满足的硬子句的数目 (或总权值). 数值等于由不满足变为满足的硬子句的数目 (权值) 减去由满足变为不满足的的数目 (权值). 若一个变量的硬分数是大于 0 的, 说明翻转该变量会使硬子句集合中不满足子句的数目 (或总权值) 变少.

定义15 (软分数 (sscore) [12]) 翻转变量 x 后, 被满足软子句的数目 (或总权值). 数值等于由不满足变为满足的软子句的数目 (权值) 减去由满足变为不满足的的数目 (权值). 若一个变量的软分数是大于 0 的, 说明翻转该变量会使软子句集合中不满足子句的数目 (或总权值) 变少.

定义16 (子权重策略 (HPAWS) [15]) 使用一个概率参数来决定是否增加权重或者减少权重值. 在一定概率值下, 对于硬子句权重值大于 1 的可满足子句, 其权重值减 1, 对于硬子句权重值大于 1 的不可满足子句, 其子句权重值加 1.

对于一些在单元子句中存在的变量, 消元法优先选择对该变量进行赋值, 然后将该变量的值进行传播. 在进行单元传播过程中硬子句和软子句分开处理, 逐一地进行顺序传播, 若存在一个硬单元子句, 则该硬单元子句相关的变量被选择用来单元传播, 直到没有新的硬单元子句产生. 在硬单元子句进行全部传播之后开始进行软单元子句的顺序传播, 对于在单元子句传播中没有被赋值的变量将对其进行随机的赋值 (0 或者 1).

局部搜索算法在局部搜索过程中综合考虑一个变量的硬分数和软分数, 从而选择合适的变量进行翻转. 该算法在局部搜索过程中选择变量的启发式方法如下:

- 若存在硬分数大于 0 的变量, 意味着翻转该变量会导致不满足硬子句权重减小, 引导搜索找到一个可满足的解, 因此, 该硬分数最大的变量优先翻转.
- 若不存在硬分数大于 0 的变量, 则优先考虑软分数大于 0 的变量进行翻转.
- 若既不存在硬分数大于 0 的变量, 又不存在软分数大于 0 的变量, 意味着搜索陷入一种困境, 此时, 若硬子句集合中还有不满足的子句, 则优先随机选择一个不满足硬子句中的变量进行翻转, 否则选择一个不满足的软子句中的变量进行翻转.

3.2 DeciDist 算法

为了更清楚地描述 DeciDist 算法, 我们首先给出 DeciDist 算法中的消元法 (如算法 1 所示).

算法 1 Decimation(F)

```
Input: A CNF formula F;
Output: an assignment of variables in F;
  while ∃ unassigned variables do
    if ∃ unit clause then
      if ∃ unit clause then
        Hard_unit_clause_Propagation();
      else
        if ∃ soft unit clause then
          Soft_unit_clause_Propagation();
        end if
      end if
    else {∄ unit clause}
      v ← a randomly unassigned vars;
      Assign(v);
    end if
  end while
```

算法 1 中 Hard_unit_clause_Propagation 函数是用硬单元子句进行单元传播, 对于新产生的硬单元子句要优先于软单元子句的传播. 消元法有效地避免了一些新产生的硬单元子句未进行传播, 而软单

元子句已经传播产生低质量的单元子句, 同时避免了一些重要变量因为在单元传播中空子句的产生导致的随机赋值. 结合消元法和局部搜索算法思想的 DeciDist 算法描述如算法 2 所示.

算法 2 DeciDist: a local search algorithm

Input: PMS instance F, cutoff, wp, t, and sp;
Output: A feasible assignment α of F, or "no feasible assignment found";

```
 1: α* ← Decimation(F);
 2: cost* ← +∞;
 3: while elapsed time < cutoff do
 4:    if ∄ falsified hard clauses & cost(α) < cost* then
 5:       α* ← α;
 6:       cost* ← cost(α);
 7:    end if
 8:    if H ← { x || hscore(x) > 0 } ≠ ∅ then
 9:       α* ← α;
10:       cost* ← cost(α);
11:    else
12:       if S ← { x || hscore(x) = 0 & sscore(x)> 0 } ≠ ∅ then
13:          V ← a variable in S with the greatest sscore, breaking ties randomly;
14:       end if
15:    else
16:       Update weights of hard clauses according to HPAWS;
17:    end if
18:    if ∃ falsified hard clauses then
19:       c ← a random falsified hard clause;
20:    else
21:       c ← a random falsified soft clause;
22:    end if
23:    if with probability wp then
24:       v ← a random variable in c;
25:    else
26:       v ← a variable in c with the greatest sscore;
27:       α ← α with v flipped;
28:    end if
29:    if cost* ⩽ SUMWeight(soft clause) then
30:       return (cost*, α*);
31:    else
32:       return "no feasible assignment found";
33:    end if
34: end while
```

算法 2 的第 1 和 2 行得到一组已赋制的变量, cost* 记录当前得到变量的赋值后, 不满足子句的权重和. 算法 2 的第 8~25 行表示 DeciDist 算法在选择变量进行翻转的启发式方法.

4 结合结构特征的 PMS 方法

第 3 节描述了最新的 PMS 求解算法 DeciDist, 与 Dist 算法相比, 该算法通过优先传播硬单元子句, 在 PMS 工业诊断实例的求解上有较大改进. 然而, 工业诊断实例中的子句长度基本相差不大, 硬

单元子句之间及软单元子句和硬单元子句之间联系紧密, 若能够应用子句结构的某些特征, 从中得到有用信息去指导求解 PMS 问题, 则可较大程度地提高求解效率, 使得不满足的软子句个数进一步减少. 下面介绍 SCPMS 算法的相关概念及该算法的伪代码描述.

4.1 相关概念

电路中的变量有输入和输出的区别, SCPMS 算法受此启发提出 PMS 的工业诊断实例应该具有某种结构特征, 不同的变量在某种结构特征下拥有不同的特性, 为了更好地解释 SCPMS 算法, 我们给出如下几个定义.

定义17 (结构特征明显子句集) 若能将子句集 U 的硬单元子句分为两部分 U_1 和 U_2, 使得移除其中一部分硬单元子句 U_2 时, 子句集 $\{U-U_2\}$ 变成可满足的, 则称该 U 为结构特征明显子句集.

定义18 (硬阻塞变量) 当一个子句集 U 结构特征明显, 将单元传播规则应用到子句集 $\{U-U_2\}$ 中得到 U_2 中相同变量的赋值, 若该赋值与 U_2 中该变量的值 (正负) 不同, 则称该变量为硬阻塞变量.

定义19 (邻居变量) 在子句集 U 中, 变量 X 可能出现在子句 C_i 中, 对于在子句 C_i 中出现的不是变量 X 的所有变量均称为变量 X 的邻居变量.

定义20 (软阻塞变量) 若一个变量满足既是某个硬阻塞变量的邻居变量, 又在某个软子句中出现, 则称这个变量是一个软阻塞变量.

4.2 CSV 算法

在给出 SCPMS 算法之前, 先介绍寻找软阻塞变量的 CSV (choose soft-blocking vars, CSV) 算法. 该算法的目标是将选出的软阻塞变量加入到软阻塞变量队列中去, 并返回该队列. 算法 3 是该算法的详细描述.

算法 3 CSV(F)

Input: a CNF formula F;
Output: A choosedSVQueue;

```
1: while DPLL(F) has a conflict do
2:     New CNF formula F ← chooseHClausedel(CNF, DelHClause);
3: end while
4: Vars ← DPLL(F);
5: choosedHVQueue ← Compare(DelHClause, Vars);
6: while choosedHVQueue is not NULL do
7:     V ← choosedHVQueue.pop();
8:     sv ← ChooseSClauseVar(v);
9:     choosedSVQueue(sv);
10: end while
11: Return a choosedSVQueue;
```

从算法 3 的第 1~3 行选择出一组硬单元子句 (DelHClause), 将该组硬单元子句删除后, 子句集能找到一组变量的赋值, 使得剩下的所有子句 (New CNF formula F) 是可满足的. 此时, SCPMS 算法判定子句集结构特征明显.

算法 3 第 4 行得到一组可满足的解, 变量赋值为 Vars. 每一个硬单元子句只涉及一个变量, 第 5 行找出 DelHClause 中的变量与可满足的解 Vars 中不同的变量, 即一组硬阻塞变量, 将其加入

choosedHVQueue 队列中. 第 6~10 行对于每一个硬阻塞变量, 根据硬阻塞变量所在的子句集, 从其邻居变量中找到在软单元子句中存在的变量, 即软阻塞变量, 将得到的一组软阻塞变量压栈.

4.3 SCPMS 算法

在探索子句结构的过程中, 新提出的 SCPMS 算法将结构特征明显的子句集 U 分成两部分: U_1 和 U_2. 对 U_1 调用 DPLL 算法, 可以得到可满足的一组变量的解 α, 该解中一定会涉及到 U_2 中存在的变量, 比较 U_2 中的变量和 α 中变量的赋值, 若相同, 则说明该变量并不会导致 U 中非单元硬子句的不满足性, 若不相同, 则说明 U 中一定有一个相关的软子句中的变量需要翻转, 算法根据子句的特点通过硬阻塞变量选出软阻塞变量进行翻转. 将一组软阻塞变量翻转之后, 该算法依据硬单元子句优先于软单元子句的顺序进行单元传播, 依据启发式的局部搜索过程对未被赋值的变量进行赋值, 从而找到一组解. 该算法的相关描述如算法 4 所示.

算法 4 SCPMS(F)

Input: a CNF formula F;
Output: An assignment of variables;

```
1: Q ← CSV(F);
2: while Q is not NULL do
3:     var ← Q.pop();
4:     Filp(var);
5: end while
6: while unpropagated Hvar do
7:     UnitPropagate(Hvar);
8: end while
9: while unpropagated Svar do
10:    UnitPropagate(Svar);
11: end while
12: while ∃ a var unassigned  do
13:    Assign var with a random number from 0,1;
14: end while
15: LocalSearch();
```

算法 4 的第 2~5 行将找到的软阻塞变量依次进行翻转, 第 6~14 行中变量将按照硬单元子句优先于软单元子句的顺序依次进行变量赋值传播, 对于单元传播中未进行分配的变量进行随机赋值. 第 15 行将赋值之后的所有变量传入局部搜索函数, 在停止机制 (时间上限或者尝试次数达上限) 到达之前得到一组解.

5 实验结果

本节将提出的算法 SCPMS 与 Cai 等提出的 DistUP 和 DeciDist 算法以及在 MBD 工业实例上表现较好的不完备求解器 WPM3 进行比较. 所选的测试用例来自 MSE2016_PMS_Industrail 中的 258 个基于模型诊断相关的实例, 根据实例的起名规则进行分组比较. 实验采用的测试环境是 Ubuntu 14.04LTS, Intel i5-3470 3.20 GHz × 4 处理器, 3.8 G 内存, 我们的算法用 C++ 实现, 编译方式采用 g++ 及 "-O3" 参数.

表 1 SCPMS 算法的求解情况

Table 1 Solution of SCPMS algorithm

Benchmark	# Inst	DistUP #Win.	DeciDist #Win.	SCPMS #Win.	WPM3 #Win.
b14	26	0	0	18	12
b15	6	0	0	1	6
b17	26	0	0	8	26
b20	64	0	0	18	58
b21	120	0	0	21	115
b22	55	0	0	8	55

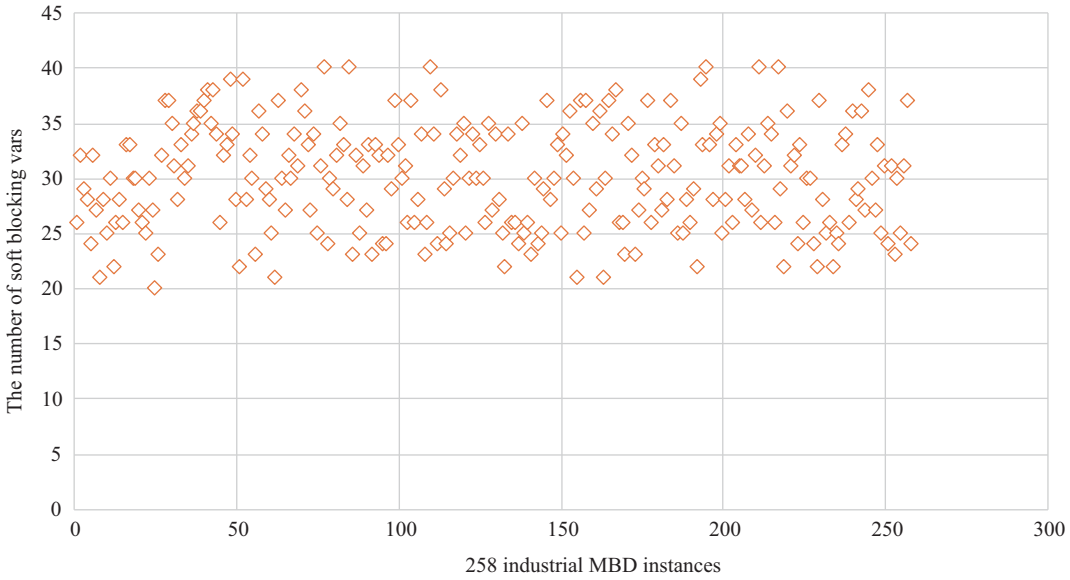

图 2 (网络版彩图) 258 个测试实例的软阻塞变量个数

Figure 2 (Color online) Number of soft blocking variables for 258 test instances

DistUP 和 DeciDist 分别为 Cai 等在 2016 和 2017 年提出的在 PMS 随机和人工实例上表现较好的求解器, WPM3 在随机实例上不及 DeciDist, 但在工业实例上优于 DeciDist. SCPMS 算法在 DeciDist 算法的基础上加入了 CSV 过程, 在不减弱其对于 PMS 随机和人工实例的求解效果的同时较大程度地改进了 MDB 工业实例的求解效果. 实验结果如表 1 所示, 实验对参数的设置为: 程序停止机制的时间是 300 s, 变量最大翻转次数是 1000000. 实验结果是对每个实验用例进行多次实验然后取平均值得到的. 表格中的第 1 列是基于模型诊断的测试用例的分组名称, 第 2 列为该分组的实例总数, 第 3~6 列分别表示该算法对应于其他算法而言找到的最优解的个数. 可以看到, SCPMS 算法较大程度地提高了 2016 年的 DistUP 算法和 2017 年提出的 DeciDist 算法在 MBD 工业实例上的求解效率, 并且有部分实例的求解优于 WPM3.

通过图 2 和表 2 中 SCPMS 算法求得的软阻塞变量的情况分析表 1 中的实验数据. 图 2 中的横坐标表示 258 个工业实例, 纵坐标表示与不满足的软子句个数成正相关的软阻塞变量个数. 通过实验数据可以看出, 将 SCPMS 算法作用于结构特征明显的子句集, 能够选择出的软阻塞变量的数目均值在 20~40 之间. 在求解工业实例问题上, 由于 DeciDist 算法求得的解不满足的软子句个数较多, 因此

表 2　软阻塞变量的求解情况

Table 2　Solution of soft blocking vars

Benchmark	# Inst.	Maximum	Minimum	Mean
b14	26	33	20	25
b15	6	37	28	30
b17	26	40	21	38
b20	64	39	22	32
b21	120	39	21	27
b22	55	40	22	33

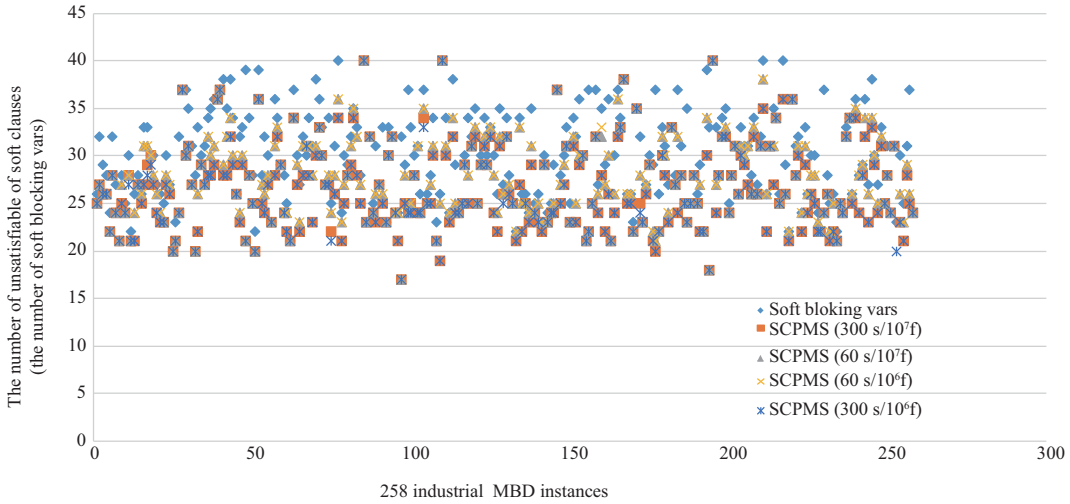

图 3　(网络版彩图) 不同停止机制、不同翻转次数下的 SCPMS 算法求解情况与软阻塞变量求解情况的比较

Figure 3　(Color online) Comparison of SCPMS algorithms with different stop mechanisms and different flip variables and solutions of soft blocking vars

SCPMS 算法优先寻找软阻塞变量的方法能够较大程度地提升随机搜索求解的起点, 使得不满足软子句个数在算法的开始阶段有较大程度的减少. 随后 SCPMS 算法采用随机搜索的方法继续探索更优解, 从而提高求解效率. 对于随机和人工实例而言, 由于实例的子句集不是结构特征明显的子句集, 因此 CSV 算法得到的软阻塞变量个数为 0, SCPMS 算法直接进入随机搜索阶段. 由于 SCPMS 算法的搜索过程和 DeciDist 算法相同, 因而相比于 DeciDist 算法, SCPMS 算法在随机和人工实例的求解效率的高效性不受影响.

在图 3 中, 纵坐标表示软阻塞变量的个数或者 SCPMS 算法求得的不满足软子句个数, 横坐标表示 258 个 MDB 工业实例. 通过图 3 可以看出, 在找到软阻塞变量的情况下, 在设置变量最大翻转次数相同时, 设置停止机制为 300 s 的求解结果要优于设置停止机制为 60 s 时的求解结果. 这证明了 SCPMS 算法在局部搜索过程中的有效性. 当程序停止机制时间为 300 s 时, 设置变量最大翻转次数为 10^6 和 10^7 时的求解效果几乎相同, 此时, 最大翻转次数已经不是影响求解结果的主要因素, 所以我们用最大翻转次数为 10^6、停止机制为 60 s 时 SCPMS 算法的求解结果分别与最大翻转次数为 10^6 并且停止机制为 300 s, 最大翻转次数为 10^6 并且停止机制为 60 s, 最大翻转次数为 10^7 并且停止机制为 300 s, 最大翻转次数为 10^7 并且停止机制为 60 s 的 DeciDist 算法的求解结果进行比较. 实验结果如

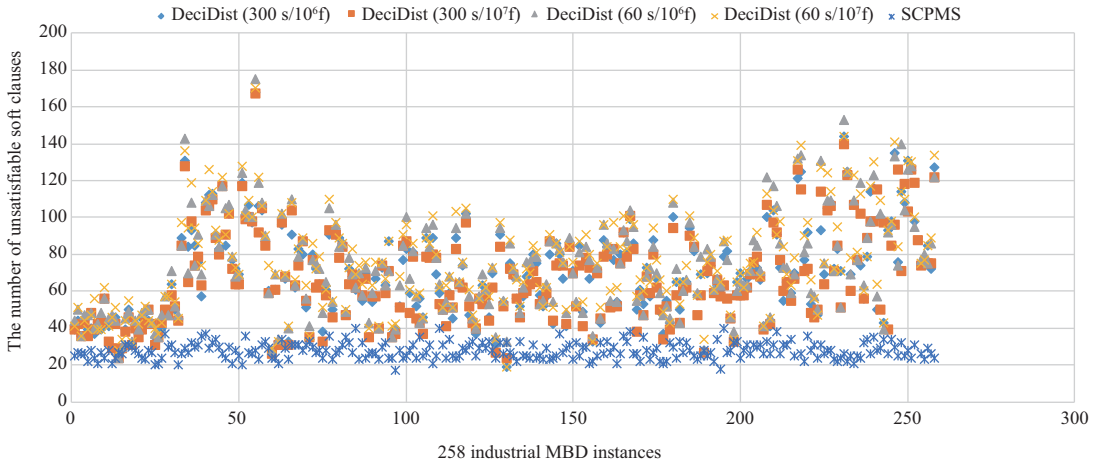

图 4 (网络版彩图) 比较不同停止机制、不同翻转变量下的 DeciDist 算法与 SCPMS 算法的求解情况

Figure 4 (Color online) Comparison of DeciDist and SCPMS algorithms with different stop mechanisms and different flip times

图 4 所示, SCPMS 算法的求解结果用蓝色星号表示, 主要分布在散点图下方. 可以看出, 相比 DeciDist 算法, SCPMS 算法求得的不满足软子句个数有较大程度的减少.

6 结束语

DeciDist 算法在 PMS 随机实例和人工实例的求解效率已经取得了很大进步, 但针对工业实例特别是工业诊断实例的求解效果依然有待改进. 考虑到在工业诊断实例中, 子句集的硬单元子句中的变量之间及软单元子句变量和硬单元子句变量之间可能存在某种紧密联系. 本文提出的 SCPMS 算法结合结构特征探索工业诊断实例问题的子句集结构特征, 将子句集中的硬单元子句结合结构特征进行分组, 由一些可以提高算法搜索起点的变量来引导算法的求解过程, 进而明显减小了不满足的软子句数目. 实验表明, 在不减弱其随机实例和人工实例的求解效果上, SCPMS 算法针对工业诊断实例较大程度地改进了 DistUP 算法和 DeciDist 算法.

参考文献

1 Li C M, Manya F, Planes J. New inference rules for Max-SAT. J Artif Intell Res, 2007, 30: 321–359

2 Lin H, Su K, Li C M. Within-problem learning for efficient lower bound computation in Max-SAT solving. In: Proceedings of the 23rd AAAI Conference of Artificial Intelligence, Chicago, 2008. 351–356

3 Heras F, Larrosa J, Oliveras A. MiniMaxSAT: an efficient weighted Max-SAT solver. J Artif Intell Res, 2008, 31: 1–32

4 Davies J, Cho J, Bacchus F. Using learnt clauses in MAXSAT. In: Proceedings of the 16th International Conference of Principles and Practice of Constraint Programming (CP 2010), St. Andrews, 2010. 176–190

5 Li C M, Manyà F, Mohamedou N, et al. Exploiting cycle structures in Max-SAT. In: Proceedings of the 12th International Conference on Theory and Applications of Satisfiability Testing, Swansea, 2009. 467–480

6 Ansótegui C, Bonet M L, Levy J. SAT-based MaxSAT algorithms. Artif Intell, 2013, 196: 77–105

7 Martins R, Manquinho V, Lynce I. Community-based partitioning for MaxSAT solving. In: Proceedings of the 16th International Conference of Theory and Applications of Satisfiability Testing (SAT 2013), Helsinki, 2013. 182–191

8 Narodytska N, Bacchus F. Maximum satisfiability using core-guided MAXSAT resolution. In: Proceedings of the 28th AAAI Conference on Artificial Intelligence, Québec City, 2014. 2717–2723

9 Fu Z, Malik S. On solving the partial MAX-SAT problem. In: Proceedings of the 9th International Conference of Theory and Applications of Satisfiability Testing, Seattle, 2006. 252–265
10 Davies J, Bacchus F. Solving MAXSAT by solving a sequence of simpler SAT instances. In: Proceedings of the 17th International Conference of Principles and Practice of Constraint Programming (CP 2011), Perugia, 2011. 225–239
11 Koshimura M, Zhang T, Fujita H, et al. QMaxSAT: a partial Max-SAT solver. J Sat Boolean Model, 2012, 8: 95–100
12 Cai S W, Luo C, Zhang H. From decimation to local search and back: A new approach to MaxSAT. In: Proceedings of the 26th International Joint Conference on Atificial Intelligence, Melbourne, 2017. 571–577
13 Morgado A, Heras F, Liffiton M, et al. Iterative and core-guided MaxSAT solving: a survey and assessment. Constraints, 2013, 18: 478–534
14 Luo C, Cai S W, Wu W, et al. CCLS: an efficient local search algorithm for weighted maximum satisfiability. IEEE Trans Comput, 2015, 64: 1830–1843
15 Cai S W, Luo C, Lin J K, et al. New local search methods for partial MaxSAT. Artif Intell, 2016, 240: 1–18
16 Robinson J A, Davis M, Logemann G, et al. A machine program for theorem-proving. J Symb Log, 1967, 32: 118
17 Biere A, Heule M, Maaren H V, et al. Handbook of Satisfiability. Netherlands: IOS Press, 2009. 99–130
18 Reiter R. A theory of diagnosis from first principles. Artif Intell, 1987, 32: 57–95
19 Chieu H L, Lee W S. Relaxed survey propagation for the weighted maximum satisfiability problem. J Artif Intell Res, 2009, 36: 229–266
20 Wang Y, Cai S, Yin M. Two efficient local search algorithms for maximum weight clique problem. In: Proceedings of the 30th AAAI Conference on Atificial Intelligence, Phoenix, 2016. 805–811

Partial maximum satisfiability problem method combined with structure characteristics for diagnostic problems

Huisi ZHOU[1,2], Dantong OUYANG[1,2], Meng LIU[1,2], Naiyu TIAN[1,2] & Liming ZHANG[1,2*]

1. *College of Computer Science and Technology, Jilin University, Changchun* 130012*, China*;
2. *Key Laboratory of Symbolic Computation and Knowledge Engineering* (*Jilin University*)*, Changchun* 130012*, China*
* Corresponding author. E-mail: limingzhang@jlu.edu.cn

Abstract Partial MaxSAT (PMS) is a generalized version of the maximum satisfiability problem (MaxSAT), which has important applications in many fields. At present, the PMS algorithm requires improvements in solving the industrial problems. Based on deep research on the PMS algorithm using random search, this paper presents a structure characteristics partial MaxSAT (SCPMS) method that combines structural feature exploration. First, according to the unit propagation rules and structural features of the problem, we gradually divide the PMS clause into two parts to construct a subproblem that can satisfy the problem due to the absence of the hard unit clause. In this paper, a random search guidance strategy is proposed. The new subproblem is reused to identify the hard-blocking variable in the hard unit clause of the original problem using unit propagation. Then, the corresponding soft-blocking variable is reversed with the feature of the clause to improve the efficiency of the random search. The experimental results show that in comparison with the two latest algorithms, DeciDist and DistUp, the proposed SCPMS greatly reduces the number of unsatisfied soft clauses in solving model-based diagnosis instances.

Keywords PMS, unit propagation, SAT, stochastic search, model-based diagnosis

Huisi ZHOU was born in 1995. She is a master student from Jilin University. Her research interests include model-based diagnosis and the PMS problem.

Liming ZHANG was born in 1980. He received a Ph.D. degree from Jinlin University. His main research interests include model-based diagnosis and the SAT problem.

Dantong OUYANG was born in 1968. She is a professor at Jilin University. Her research interests include model-based diagnosis, the SAT problem, and model checking.