基于 MACR 和 CAL 启发式的求差知识编译算法

牛当当[1,3],吕 帅[2,5],王金艳[6],刘 斌[1,3,4]

(1. 西北农林科技大学信息工程学院,陕西杨凌 712100;2. 吉林大学计算机科学与技术学院,吉林长春 130012;
3. 陕西省农业信息感知与智能服务重点实验室(西北农林科技大学),陕西杨凌 712100;
4. 农业农村部农业物联网重点实验室(西北农林科技大学),陕西杨凌 712100;
5. 符号计算与知识工程教育部重点实验室(吉林大学),吉林长春 130012;
6. 广西师范大学计算机科学与信息工程学院,广西桂林 541004)

摘 要: DKCHER 算法是基于超扩展规则的求差知识编译算法. 本文首先研究了 DKCHER 算法的执行流程,并定义了互补量的概念,然后设计了启发式策略 MACR(maximum complementary amount of clauses with middle result),用于动态选择与中间结果互补量最大的子句. 针对互补展开过程,设计了动态启发式策略 CAL(optimal sequence sorted by complementary amount of literals),将互补展开中的文字按照与输入公式互补量的大小进行排序并展开. 将上述两种启发式策略与 DKCHER 算法相结合,分别设计了 MACR_DKCHER 算法、CAL_DKCHER 算法和 MACR_CAL_DKCHER 算法. 实验结果表明,MACR 启发式策略能够提升 DKCHER 算法的编译效率和编译质量,编译效率最高可提升 9 倍,编译质量最高可提升 1.9 倍;CAL 启发式策略在子句数和变量数比值较大的实例上,能够提高 DKCHER 算法的编译效率,但会降低 DKCHER 算法的编译质量;MACR_CAL 启发式最高可将 DKCHER 算法的编译效率提高 12 倍,但会导致 DKCHER 算法的编译质量有所降低.

关键词: 知识编译; 扩展规则; 超扩展规则; EPCCL 理论; 启发式策略

中图分类号: TP301, TP181 **文献标识码:** A **文章编号:** 0372-2112 (2020)02-0285-06

电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2020.02.009

Knowledge Compilation Algorithm of Computing Difference Based on MACR Heuristics and CAL Heuristics

NIU Dang-dang[1,3], LÜ Shuai[2,5], WANG Jin-yan[6], LIU Bin[1,3,4]

(1. *College of Information Engineering, Northwest A&F University, Yangling, Shaanxi* 712100, *China*;
2. *College of Computer Science and Technology, Jilin University, Changchun, Jilin* 130012, *China*;
3. *Shaanxi Key Laboratory of Agricultural Information Perception and Intelligent Service (Northwest A&F University), Yangling, Shaanxi* 712100, *China*;
4. *Key Laboratory of Agricultural Internet of Things, Ministry of Agriculture and Rural Affairs (Northwest A&F University), Yangling, Shaanxi* 712100, *China*;
5. *Key Laboratory of Symbolic Computation and Knowledge Engineering (Jilin University), Ministry of Education, Changchun, Jilin* 130012, *China*;
6. *School of Computer Science and Information Engineering, Guangxi Normal University, Guilin, Guangxi* 541004, *China*)

Abstract: DKCHER is a knowledge compilation algorithm of computing difference based on hyper extension rule. We study on the executing process of DKCHER algorithm in this paper, and define the concept of complementary amount. We design MACR (maximum complementary amount of clauses with middle result) heuristics based on complementary amount, which is used to dynamically select the clause of maximum complementary amount with middle result. For complementary unfolding in DKCHER, we design dynamic heuristics CAL (optimal sequence sorted by complementary amount of literals), which sort the literals in complementary unfolding based on their complementary amounts. Combining the above

收稿日期:2019-04-26;修回日期:2019-09-09,责任编辑:马兰英
基金项目:国家自然科学基金(No. 61602388, No. 61763003);吉林省科技发展计划资助项目(No. 20180101053JC);陕西省自然科学基础研究计划项目(No. 2017JM6059);中央高校基本科研业务费专项资金(No. 2452019064);中国博士后科学基金(No. 2017M613216);陕西省博士后基金(No. 2016BSHEDZZ121);陕西省重点研发计划项目(No. 2019ZDLNY07-06-01);广西省自然科学基金(No. 2016GXNSFAA380192);西北农林科技大学博士科研启动基金(No. Z109021813);广西多源信息挖掘与安全重点实验室开放基金(No. MIMS19-05)

two heuristic methods with DKCHER, MACR_DKCHER algorithm, CAL_DKCHER algorithm and MACR_CAL_DKCHER algorithm are designed. Experimentally, MACR heuristics improves the compilation efficiency and compilation quality of DKCHER. MACR heuristics can improve the efficiency of DKCHER by 9 times in the best case. CAL heuristics can significantly improve the compilation efficiency of DKCHER on the instances with big ratio of clause number with variable number. MACR_CAL heuristics can improve the efficiency of DKCHER by 12 times in the best case. But MACR_CAL heuristics reduces the compilation quality of DKCHER.

Key words: knowledge compilation; extension rule; hyper extension rule; EPCCL theory; heuristic strategy

1 引言

知识编译是命题逻辑推理方法的重要组成部分,主要通过结合离线编译和在线推理求解重复性推理问题. Newton 等人结合理论分析和实例验证给出了 ROBDDs 最坏情况下的编译规模[1]. Jabbour 等人基于本原蕴含这一重要知识编译目标语言量化了命题逻辑中的冲突[2]. Salhi 提出了几种用于枚举 CNF 公式中所有必要本原蕴含式的方法[3]. Uña 等人通过使用子问题预编译技术,提出了将 CP 子问题编译为 MDDs 和 d-DNNFs 的方法[4]. Darwiche 等人设计了多种比较知识编译目标语言的方法,并提出了知识编译图谱[5]. 基于蕴含文字的概念,Lai 等人设计了带蕴含文字的有序二元决策图[6],同时还基于合取的可分解性设计了几种新的知识编译目标语言[7]. 上述研究提高了相关知识编译目标语言的应用范围、表示能力以及编译算法的性能.

通过提取归结方法的逆过程,Lin 等人于 2003 年提出了扩展规则推理方法(extension rule, ER)[8],受到了国内外专家学者的一致认可. 基于扩展规则,Lin 等人提出了知识编译算法 KCER,可以将子句集编译为 EPCCL 理论[9]. 针对 KCER 算法,谷文祥等人设计了 MCN 和 MO 两种启发式策略,提高了其编译效率和编译质量[10]. 刘大有等人提出了新的 EPCCL 理论编译器 C2E,该编译器具有较高的编译效率和编译质量[11]. 刘磊等人基于超扩展规则提出了求差知识编译算法 DKCHER,具有最优的 EPCCL 理论编译性能[12]. EPCCL 理论是一种高效的知识编译目标语言,提高其编译算法的编译效率和编译质量具有重要意义.

启发式策略可用于提升多个领域中算法的性能[13,14]. 有效的启发式策略也能够提高 EPCCL 理论编译算法的效率和质量,如谷文祥等人针对 KCER 算法设计的高效启发式策略 MCN 和 MO[10]. 然而不同算法的执行流程差距较大,其中的选择过程和计算过程也有较大差别,因此将 MCN 和 MO 等现有启发式策略直接应用到 DKCHER 算法是不可行的.

本文通过深入研究 DKCHER 算法的执行流程,设计并实现两种启发式策略 MACR 和 CAL. 结合 DKCHER 算法与上述启发式策略,设计并实现 MACR_DKCHER 算法、CAL_DKCHER 算法和 MACR_CAL_DKCHER 算法. 实验结果表明,MACR 启发式策略始终能够提高 DKCHER 算法的编译效率,而 CAL 启发式和 MACR_CAL 启发式在子句数与变量数的比值较大的实例上能够显著提高 DKCHER 算法的编译效率.

2 超扩展规则

为了表示方便,本文用变量集 $V(F)$ 和变量集 $V(C)$ 分别存储子句集 F 和子句 C 中出现的所有变量,用极大项集 $J_N(C)$ 和极大项集 $J_N(F)$ 分别存储子句 C 和子句 F 在变量集 N 上所能扩展出的所有极大项. 本文所研究的子句集中不包含重言式. 语义等价用‘≡’符号表示,对于单个子句 C,$C \equiv J_N(C)$.

定义 1(超扩展规则,Hyper extension rule)[12]. 给定两个子句 C 和 A,$D = \{C \vee A, C \vee \neg A\}$,其中 $V(C) \cap V(A) = \varnothing$,将 C 到 D 的推导过程称为超扩展规则,D 中的元素为 C 应用超扩展规则的结果.

超扩展规则具有以下特殊性质:基于超扩展规则能够利用 EPCCL 理论保存两个子句所能扩展出极大项集的差集[12].

定义 2(EPCCL 理论)[9]. 子句集 F 是一个 EPCCL 理论,则 F 中任意两个子句间均含有互补文字对.

令 N 表示任意子句集 $F = \{C_1, \cdots, C_m\}$ 中出现的变量集. DKCHER 算法利用超扩展规则计算 $J_N(\square) - (J_N(\square) - J_N(C_1) - \cdots - J_N(C_m))$,进而得到与 F 等价的 EPCCL 理论[12]. DKCHER 算法分两步执行:第一步计算 $J_N(\square) - J_N(C_1) - \cdots - J_N(C_m)$,得到 EPCCL 理论 $\{E_1, \cdots, E_h\}$. 第二步计算 $J_N(\square) - J_N(E_1) - \cdots - J_N(E_h)$,得到与 F 等价的 EPCCL 理论. 在单阶段计算过程中,DKCHER 算法按照子句编号顺序求解. 然而,求解过程中子句的选择顺序会极大地影响其编译效率和编译质量.

3 MACR 启发式策略

DKCHER 是一种分两阶段执行的编译算法,且两个阶段的编译过程相同. 因此在考虑子句选择策略时,可以仅考虑 DKCHER 算法的单阶段编译过程. 假设 DKCHER 算法单阶段编译过程的输入子句集为 $F = \{C_1, \cdots, C_m\}$,F 中出现的变量集为 N,当前编译过程进行到第 $i(1 \leqslant i \leqslant n)$ 次求差操作,当前 EPCCL 理论为 E

$=\{D_1,\cdots,D_h\}$,若 $i>1$ 则 $J_N(E)=J_N(\square)-J_N(C_1)-\cdots-J_N(C_{i-1})$,否则 $J_N(E)=J_N(\square)$. 假设当前需要与 E 进行求差操作的子句为 C_i,则 $J_N(E)-J_N(C_i)$ 的计算过程可以分解为 E 中每个子句 D_j 与 C_i 的求差运算. 根据文献[12],当 $C_i|=D_j$ 时,两者之间的求差运算会降低 E 的规模,然而通常满足 $C_i|=D_j$ 的条件会较少,且该条件判断起来需要的时间复杂度为 $O(|C_i|)\times O(|D_j|)$;当 D_j 与 C_i 互补时,两者之间的求差运算不会增加 E 的规模. 因此,本文将重点考虑选择与 E 中元素互补较多的子句.

定义 3(互补量). 给定一个子句集 F 和一个子句 C,将 C 中所有文字的反文字在 F 中的总出现次数称为 C 与 F 的互补量.

显然,互补量可以在一定程度上反映一个子句和一个子句集的互补情况.

本文重点考虑选择与中间结果 EPCCL 理论互补率尽可能高的子句. 互补量计算的时间复杂度为 $O(|C|)$. 然而,若不增加空间开销,计算子句与子句集的互补率的时间复杂度为 $O(|C|)\times O(|F|)$. 为了快速选择出较好的子句,本文通过选择与中间结果 EPCCL 理论互补量最高的子句来替代互补率计算.

给定一个子句集 F 和一个文字 l,本文用 $S_F(l)$ 表示 l 在 F 中出现的次数. 假设 DKCHER 算法单阶段编译过程中的输入子句集为 $F=\{C_1,\cdots,C_m\}$,当前编译过程进行到第 i 次求差操作,当前已得到的 EPCCL 理论为 E. 当 $i=1$ 时,$E=\{\square\}$,则本文选择 F 中与其它子句互补量较高的子句,对于任意子句 $C_x=l_1\vee\cdots\vee l_k$,则 C_x 与 F 的互补量计算方法为 $\sum_{1\leqslant j\leqslant k}S_F(\neg l_j)$. 当 $1<i\leqslant n$ 时,选择 F 剩余子句中与 E 互补量最高的子句,假设 F 中任意剩余子句 $R_x=l_1\vee\cdots\vee l_k$,则 R_x 与 E 的互补量计算方法为 $\sum_{1\leqslant j\leqslant k}S_E(\neg l_j)$. 本文将上述选择过程称为 MACR 启发式策略,该策略的执行流程如算法 1 所示.

算法 1　MACR(maximum complementary amount of clauses with middle result)

输入:EPCCL 理论 E,子句集 $F=\{C_1,\cdots,C_m\}$,当前进行到第 i 次求差操作
输出:F 中的某个子句 C

```
1  Max = 0
2  IF i = 1
3    WHILE h≤m
4      k = Σ_{l_j∈C_h} S_F(¬ l_j)
5      IF k > Max
6        C = C_i
7        Max = k
8        h + +
9  ELSE
10   WHILE h≤m - i + 1
11     k = Σ_{l_j∈C_h} S_E(l_j)
12     IF k > Max
13       C = C_i
14       Max = k
15     h + +
16 RETURN C
```

算法 1 首先根据 i 的值判断当前 DKCHER 算法的单阶段编译过程执行到第几步求差操作. 第 3 ~ 8 行用于选择 F 的所有子句中反文字在 F 中出现次数最多的子句;第 10 ~ 15 行用于选择 F 中剩余所有子句中反文字在 E 中出现次数最多的子句. 由于每次求差操作都会从 F 中删除一个子句,因此第 10 行中用 $m-i+1$ 表示当前输入子句集 F 的模.

本文将 MACR 启发式策略与 DKCHER 算法结合,设计了 MACR_DKCHER 算法,该算法的执行流程如算法 2 所示.

算法 2　MACR_DKCHER

输入:令子句集 $F_1=\{C_1,\cdots,C_n\}$,M 包含了 F 中出现的所有变量
输出:与 F_1 等价的 EPCCL 理论

```
1  初始化:F2 = {□},h = i = j = 1
2  WHILE i≤|F1|
3    C = MACR(F2,F1,i)
4    F1 = F1 - {C}
5    WHILE j≤|F2|
6      IF C 与 Cj 互补 THEN SKIP
7      ELSE IF C| = Cj THEN F2 = F2 - {Cj}
8      ELSE Cj = {Cj ∨ ¬ (C - Cj)}
9      j + +
10   i + +
11   j = 1
12 IF h = 1 THEN
13   F1 = F2
14   h - -
15   i = 1;GOTO 2
16 ELSE
17   RETURN F2
```

显然,MACR 启发式仅作用于 DKCHER 算法单阶段编译过程求差操作中子句的选择,因此 MACR_DKCHER 算法的编译结果与 DKCHER 算法的编译结果是语义等价的,即 MACR_DKCHER(F_1) ≡ DKCHER(F_1).

4　CAL 启发式策略

在求差操作中,当 C_i 既不蕴含 D_j 也不与 D_j 互补时,

DKCHER 算法会利用扩展规则计算 $D_j = \{D_j \vee \neg (C_i - D_j)\}$,利用互补展开可得结果 $\{D_j \vee \neg (C_i - D_j)\} = \{D_j \vee \neg l_1, D_j \vee l_1 \vee \neg l_2, \cdots, D_j \vee l_1 \vee \cdots \vee l_{n-1} \vee \neg l_n\}$. 上述展开按照文字编号顺序展开,而事实上采取任何展开顺序都不影响展开结果的等价性,而展开结果会对之后的编译过程产生影响,因此选择一种好的展开方式会对 DKCHER 算法产生积极的影响.

本文针对互补展开设计一种启发式策略 CAL. 令 $A = \{l_1, \cdots, l_n\}$ 是 $C_i - D_j$ 的计算结果,CAL 启发式策略则按照 A 中文字与 F 的互补量的大小进行排序,并返回 $\neg (C_i - D_j)$ 的互补展开结果. CAL 启发式策略的执行过程如算法 3 所示.

算法 3 CAL(optimal sequence sorted by complementary amount of literals)

```
输入:子句集 F = {C1,…,Cm},A = {l1,…,ln}
输出:A 的互补展开子句集
1  Max = 0
2  WHILE j≤n
3    i = j
4    WHILE i≤n
5      k = Σ_{l_i∈A} S_F(¬ l_j)
6      IF k > Max
7        t = i
8        Max = k
9      i + +
10   Swap(l_i,l_j)
11   j + +
12 RETURN ¬ A
```

在算法 3 中,本文使用选择排序按照 A 中文字与 F 的互补量对 A 进行排序,并按照排序结果展开 $\neg A$ 并返回.

本文将 CAL 启发式策略与 DKCHER 算法结合,设计了 CAL_DKCHER 算法,执行流程如算法 4 所示.

算法 4 CAL_DKCHER

```
输入:令子句集 F1 = {C1,…,Cn},M 包含了 F 中出现的所有变量
输出:与 F1 等价的 EPCCL 理论
1  初始化:F2 = {□},F3 = ,h = i = j = 1
2  WHILE i≤|F1|
3    F1 = F1 - {Ci}
4    WHILE j≤|F2|
5      IF Ci与 Cj互补 THEN SKIP
6      ELSE IF Ci| = Cj THEN F2 = F2 - {Cj}
7      ELSE
8        A = Ci - Cj
9        F3 = CAL(F1,A)
10       Cj = Cj ∨ F3
11       j + +
12     i + +
13     j = 1
14 IF h = 1 THEN
15   F1 = F2
16   h - -
17   i = 1;GOTO 2
18 ELSE
19   RETURN F2
```

CAL_DKCEHR 算法与 DKCHER 算法的区别之处在于第 9 行使用 CAL 启发式的输出作为互补展开的结果. 显然,CAL 启发式仅作用于 DKCHER 互补展开的变量顺序选择,因此 CAL_DKCHER 算法的编译结果与 DKCHER 算法的编译结果是等价的.

需要注意的是,由于 DKCHER 算法会动态删除输入子句集中的子句或扩展中间结果 EPCCL 理论,所有文字的出现次数也在动态改变. 因此本文设计的两种启发式策略并不能在算法开始之前就确定好选择顺序,而是随着算法的执行动态选择,即 MACR 启发式策略和 CAL 启发式策略是两种动态启发式策略.

本文还组合 MACR 启发式策略和 CAL 启发式策略,并结合 DKCHER 算法设计了新的知识编译方法 MACR_CAL_DKCHER,该算法使用 MACR 启发式作为求差过程中的子句选择,使用 CAL 启发式作为互补展开的结果. 为节约篇幅,本文在此省略 MACR_CAL_DKCHER 算法的描述.

5 实验结果与分析

本文用 C 语言实现了 MACR_DKCHER 算法、CAL_DKCHER 算法和 MACR_CAL_DKCHER 算法,并将 DKCHER 算法[12]作为对比算法进行测试. 本文参考文献[12],选择随机 3-SAT 问题作为实验测试用例对比测试上述算法的编译效率和编译质量(使用编译结果中子句数量进行衡量). 本文实验平台如下:CPU:Intel(R) Core(TM) i5-6600 CPU @ 3.30 GHZ 3.31 GHZ,内存:8GB,操作系统:Windows 10.

表 1 给出了 $<20, m>$、$<25, m>$ 和 $<30, m>$ 三种不同 3-SAT 子句集实例的样例上的测试. 表中时间单位为毫秒,并用加方框的形式突显最优编译质量,用加粗和下划线的形式突显最优编译效率,测试结果为 50 次实验的平均值.

由表 1 可知,在子句数和变量数的比值较小时,MACR 启发式策略能够提高 DKCHER 算法的编译质量;在子句数和变量数的比值较大时,MACR 启发式策略依旧能够显著提高 DKCHER 算法的编译效率,而对编译质量的提升不够明显,甚至在某些类型的实例上略有降低. CAL 启发式策略能够提高 DKCHER 算法的

编译效率,但同时降低了其编译质量. 原因在于 CAL 启发式策略尽管提高了互补展开结果中长子句与输入公式之间的互补量,短子句会获得更多的扩展机会,进而降低了编译结果保存极大项的能力. MACR_CAL 启发式继承了 MACR 启发式和 CAL 启发式的优点,对 DKCHER 算法在编译效率上有所提高,然而同时继承了 CAL 启发式策略的缺点,降低了 DKCHER 算法的编译质量.

表 1　随机 3-SAT 实例上的实验结果

Instances	MACR_DKCHER		CAL_DKCHER		MACR_CAL_DKCHER		DKCHER	
	size	time	size	time	size	time	size	time
<20,75>	127	7	194	15	251	**6**	133	19
<20,85>	38	5	64	16	71	**4**	44	15
<20,95>	36	5	70	13	69	**4**	48	16
<20,105>	7	4	11	13	9	**2**	8	13
<25,87>	811	41	1753	128	1854	**39**	1111	157
<25,97>	290	45	583	124	568	**24**	367	151
<25,107>	121	24	202	127	214	**20**	118	164
<25,117>	34	21	54	90	46	**19**	35	120
<30,98>	3059	**269**	7237	832	8168	332	3498	838
<30,108>	807	**133**	2061	711	2562	142	1551	1084
<30,118>	779	127	1506	623	1612	**109**	723	725
<30,128>	164	104	259	540	324	**84**	161	751

整体来看,当变量数固定时,随着子句数的增加,表 1 中四种算法的编译结果规模均越来越小,编译过程所需时间也越来越短. 变量数固定时,随着子句数的增加,会导致子句集的可满足性解越来越少,甚至直接导致子句集不可满足. 当子句集不可满足时,四种算法会得到相同的编译结果{□},其规模为 1. 因此,实验部分并未选择子句数较大的测试用例.

本文还用随机产生器生成了相变点附近的 3-SAT 测试样例,即测试样例中 $n/m \approx 4.26$,并在这些测试用例对比测试了表 1 中四种算法的编译效率和编译质量,测试结果如表 2 所示.

表 2　$m/n \approx 4.26$ 的随机 3-SAT 实例上的实验结果

Instances	MACR_DKCHER		CAL_DKCHER		MACR_CAL_DKCHER		DKCHER	
	size	time(ms)	size	time(ms)	size	time(ms)	size	time(ms)
<30,128>	39	116	63	629	64	**85**	75	1009
<31,133>	109	159	282	1019	254	**127**	151	1450
<32,137>	140	201	274	1527	274	**2**	142	1635
<33,141>	317	306	792	1961	873	**219**	445	2273
<34,146>	145	489	285	2629	313	**309**	211	3371
<35,150>	230	464	526	3329	376	**341**	280	4052

表 2 进一步验证了表 1 中的部分结论. MACR_DKCHER 算法显著提高了 DKCHER 算法的编译效率,平均可提升 6 ~ 9 倍,而最好情况下编译质量可提升 1.9 倍;CAL 启发式策略依旧能够显著提高 DKCHER 算法的编译效率;而 MACR_CAL_DKCHER 算法在所有实例上均取得了最优编译效率,并且相对于 DKCHER 算法提升较为显著,平均可提升 8 ~ 12 倍.

6　结论与展望

本文提出了两种新的动态启发式策略 MACR 和 CAL. 其中,MACR 启发式策略主要作用于 DKCHER 算法中 EPCCL 理论与子句单次求差操作过程中子句的选择,而 CAL 启发式策略则作用于两个子句求差操作过程中互补展开的变量顺序选择. 实验结果表明,MACR 启发式策略能够始终提升 DKCHER 算法的编译效率和编译质量;CAL 启发式策略和 MACR_CAL 启发式策略

在子句数和变量数的比值较大的实例上能够显著提高DKCHER算法的编译效率.

未来,将研究如何挖掘更多的启发式信息,进一步提高本文提出的启发式策略的启发效果.

参考文献

[1] Newton J, Verna D. A theoretical and numerical analysis of the worst-case size of reduced ordered binary decision diagrams[J]. ACM Transactions on Computational Logic, 2019, 20(1): 6:1 – 6:36.

[2] Jabbour S, Ma Y, Raddaoui B, Sais L. Quantifying conflicts in propositional logic through prime implicates[J]. International Journal of Approximate Reasoning, 2017, 89: 27 – 40.

[3] Salhi Y. Approaches for enumerating all the essential prime implicants[A]. In Proceedings of Artificial Intelligence: Methodology, Systems, and Applications – 18th International Conference (AIMSA)[C]. Varna, Bulgaria, 2018. 228 – 239.

[4] Uña D, Gange G, Schachte P, Stuckey PJ. Compiling CP subproblems to MDDs and d-DNNFs[J]. Constraints, 2019, 24(1): 56 – 93.

[5] Darwiche A, Marquis P. A knowledge compilation map[J]. Journal of Artificial Intelligence Research, 2002, 17: 229 – 264.

[6] Lai Y, Liu DY, Wang SS. Reduced ordered binary decision diagram with implied literals: A new knowledge compilation approach[J]. Knowledge and Information Systems, 2013, 35(3): 665 – 712.

[7] Lai Y, Liu DY, Yin MH. New canonical representations by augmenting OBDDs with conjunctive decomposition[J]. Journal of Artificial Intelligence Research, 2017, 58: 453 – 521.

[8] Lin H, Sun JG, Zhang YM. Theorem proving based on the extension rule[J]. Journal of Automated Reasoning, 2003, 31(1): 11 – 21.

[9] Lin H, Sun JG. Knowledge compilation using the extension rule[J]. Journal of Automated Reasoning, 2004, 32(2): 93 – 102.

[10] 谷文祥,王金艳,殷明浩. 基于MCN和MO启发式策略的扩展规则知识编译方法[J]. 计算机研究与发展, 2011, 48(11): 2064 – 2073.
Gu WX, Wang JY, Yin MH. Knowledge compilation using extension rule based on MCN and MO heuristic strategies[J]. Journal of Computer Research and Development, 2011, 48(11): 2064 – 2073. (in Chinese)

[11] 刘大有,赖永,林海. C2E:一个高性能的EPCCL理论编译器[J]. 计算机学报, 2013, 36(6): 1254 – 1260.
Liu DY, Lai Y, Lin H. C2E: An EPCCL compiler with good performance[J]. Chinese Journal of Computer, 2013, 36(6): 1254 – 1260. (in Chinese)

[12] 刘磊,牛当当,吕帅. 基于超扩展规则的知识编译方法[J]. 计算机学报, 2016, 39(8): 1681 – 1696.
Liu Lei, Niu Dang-Dang, Lü Shuai. Knowledge compilation methods based on the hyper extension rule[J]. Chinese Journal of Computer, 2016, 39(8): 1681 – 1696. (in Chinese)

[13] Niu DD, Liu L, Lü S. Knowledge compilation methods based on the clausal relevance and extension rule[J]. Chinese Journal of Electronics, 2018, 27(5): 1037 – 1042.

[14] Li HB, Liang YC, Zhang N, Guo JS, Xu D, Li ZS. Improving degree-based variable ordering heuristics for solving constraint satisfaction problems[J]. Journal of Heuristics, 2016, 22(2): 25 – 145.

作者简介

牛当当(通信作者) 男,1990年2月出生,陕西周至人. 现为西北农林科技大学信息工程学院讲师,主要研究方向为自动推理和抽象论辩.
E-mail: niudd@nwafu.edu.cn

吕 帅 男,1981年7月出生,吉林公主岭人. 现为吉林大学计算机科学与技术学院副教授,博士生导师,主要研究方向为人工智能、智能规划与自动推理.
E-mail: lus@jlu.edu.cn