第 38 卷 第 21 期
Vol.38 No.21

计 算 机 工 程
Computer Engineering

2012 年 11 月
November 2012

·人工智能及识别技术· 文章编号：1000—3428(2012)21—0185—04 文献标识码：A 中图分类号：TP18

一种基于图分割的动态回溯算法

王 萌

(吉林大学软件学院，长春 130012)

摘 要：动态回溯算法在进行回溯时保留所有已赋值变量的值，从而可能与后面赋值的变量产生冲突，其在解决不具有明显子问题结构的约束满足问题时效率较低。为此，将图分割技术应用于动态回溯，通过图分割将变量分为若干集合，当发生回溯时，不保留全部变量的值，舍弃那些与引起冲突的变量在同一集合变量中的值。实验结果表明，该算法在求解没有明显子问题结构的约束满足问题时具有较高的效率。

关键词：人工智能；约束满足问题；动态回溯；图分割；约束网络

A Dynamic Backtracking Algorithm Based on Graph Partitioning

WANG Meng

(College of Software, Jilin University, Changchun 130012, China)

【Abstract】 The dynamic backtracking algorithm keeps its assigned value for each variable when it backtracks. However, it becomes inefficient when it comes to the constraint satisfaction problems that do not consist of sub problems. To solve the problem, this paper applies graph partitioning to dynamic backtracking. The main idea is to divide all the variables into several sets using graph partitioning. When it back tracks, it gives up the values of the variables that are in the same set with the culprit variable, instead of keeping all the values. Experimental results show that the new method has higher efficiency on the random satisfaction problems.

【Key words】 artificial intelligence; Constraint Satisfaction Problem(CSP); dynamic backtracking; graph partitioning; constraint network

DOI: 10.3969/j.issn.1000-3428.2012.21.050

1 概述

约束满足问题(Constraint Satisfaction Problem, CSP)是人工智能领域一类重要的问题。现实世界中有许多问题可以建模为约束满足问题并通过约束求解技术解决，如规划问题、作业调度、资源分配、决策支持。约束满足问题的求解过程是 NP-hard 的。在众多的求解算法中，回溯算法(Backtracking algorithm, BT)是一个完备的核心算法。该算法在求解过程中采用深度优先策略实例化变量，若遇到死结点，则进行回溯。

为提高回溯算法的效率，许多优化策略已被提出[1]。其中比较重要的有分支策略、约束传播、冲突记录、回跳、对变量和值排序的启发式策略[2]等。同时，一些智能回溯算法也被提了出来。文献[3]提出的动态回溯算法就是其中的一种。动态回溯算法在回溯时，保留当前所有已赋值变量的值，然后在 nogoods 中找到那些引起冲突的变量(culprit variable)，从中按一定规则选取一个进行回溯，以保证算法的完备性。

对于一些具有明显子问题特征的问题，动态回溯可以比较高效地解决。然而，文献[4]通过对随机测试用例的分析，指出动态回溯对于某些问题的执行效率反而比普通回溯算法执行效率更低，甚至相差指数倍。

已经有许多关于动态回溯算法的改进工作，大多是将传统优化策略应用于动态回溯算法，如带有冲突有向回跳的动态回溯、在动态回溯中维持弧相容、对变量重排序的动态回溯等。

本文不局限于传统的优化策略，而是将图分割应用于动态回溯。由于动态回溯算法在回溯时保留当前所有已赋值变量的值，这些值可能与随后赋值的变量产生冲突，因此增加了回溯的次数，严重影响了效率。本文提出在回溯时不保留全部的值，只保留部分值。将约束网络作为一个图，应用图分割算法将变量分为若干个集合。在回溯时只保留那些与 culprit variable 不在同一集合的变量的值，舍弃那些与 culprit variable 在同一集合的变量的值。

2 动态回溯算法

动态回溯算法[3]的特点是在发生回溯时保留当前已赋值变量的值，从而避免对一些子问题的重复赋值，然后回溯到引起冲突的那个变量。下面将给出与动态回溯相关的概念。这里仅考虑二元约束网络，即每一个约束至多涉及 2 个变量。

定义 1 约束满足问题是一个三元组(X, D, C)，其中，$X=\{x1,x2,\cdots,xn\}$是 n 个变量的有限集合，$D=\{dom(x1), dom(x2), \cdots, dom(xn)\}$是所有变量的一个定义域集合，

作者简介：王 萌(1989－)，男，本科生，主研方向：人工智能，约束网络

收稿日期：2011-12-27 **修回日期**：2012-02-26 **E-mail**：wangm0855@gmail.com

$dom(xi)(i=1,2,\cdots,n)$是变量 xi 的可以取值的一个有限域；$C=\{C1,C2,\cdots,Ce\}$是一个约束集合。其中每一个约束 $Ck(k=1,2,\cdots,e)$是关于 2 个 变量的一个二元约束。xi 和 xj 中的一个约束通常记为 Cij。$(a,b)\in Cij$ 意味着当 $xi=a$ 且 $xj=b$ 时，约束 Cij 成立。

CSP 的一个解是对所有变量的一组有效赋值，使得这些赋值满足所有的约束。

定义 2 CSP 的一个部分解 P 是由二元组 (x,v) 构成的集合。其中，(x,v) 表示变量 x 当前取值为 v，并且这些变量的取值满足所有约束条件；P 中所有变量 x 构成的集合记为 $\overline{P}$。

约束满足问题的求解实际是通过对不在当前部分解 P 中的变量赋值而扩展当前解，直到 $\overline{P}=X$。

动态回溯算法是在依赖有向回跳[5]的基础上发展而来的。通过在算法执行的过程中记录一些有用的信息 eliminating explanation 来指导回溯。eliminating explanation 记录了一个变量为某个值时会与其发生冲突的那些变量，即发生冲突的原因，这样可以在回溯时直接跳到这些变量中的某个。

定义 3(eliminating explanation) 对于一个部分解 P，变量 x 的 eliminating explanation 是一个二元组 (v,S)，其中，$v\in dom(x)$且 $S\in\overline{P}$。其表示 x 不能取 v，原因是 S 中的变量在 P 中被指派的值。给定一个部分解 P 和一个变量 $x\notin\overline{P}$，elimination 函数 $e(P,x)$ 可以给出此时的 eliminating explanation。一个变量 x 的 eliminating explanation 集 E 是有关 x 的所有 (v,S) 构成的集合。其中，所有 v 构成的集合记为 $\overline{E}$，表示变量 x 在部分解 P 下不能取的值的集合。同理，函数 $\overline{e}(P,x)$ 可以给出此时 x 不能取的值的集合。

有了上面的定义，下面给出动态回溯算法的描述：

(1)设当前部分解，对于每个变量 x，设 $E_x=\phi$。

(2)如果 $\overline{P}=X$，则 P 是一个解，算法结束。否则，选择一个变量 $x\in X-\overline{P}$，设 $E_x=E_x\cup e(P,x)$。

(3)令 $S=dom(x)-\overline{E_x}$。如果 S 不空，则从 S 中选择一个值 $v\in S$，向当前部分解 P 中添加 (x,v)。回到第(2)步。

(4)如果 S 为空集，即 $\overline{E_x}=dom(x)$，设 E 是变量 x 的所有 eliminating explanation 构成的集合。

(5)如果 E 为空集，则该 CSP 问题无解，返回。否则，设 (j,v_j) 满足 j 在 E 中出现，且该元组在 P 中最靠后。首先从 P 中删除 (j,v_j)。然后对于每个在 j 后面被赋值的变量 k，从 E_k 中删除所有涉及变量 j 的 eliminating explanation。向 E_j 中添加元组 $(v_j,E\cap\overline{P})$。转到第(2)步。

3 动态回溯的缺陷

动态回溯在发生回溯时保留全部当前已赋值变量的值，对于具有明显子问题结构的问题，这可以避免对同一子问题重复赋值。然而，对于没有明显子结构的问题，保留全部变量的值可能与随后赋值的变量产生冲突，从而增加回溯的次数，严重影响了效率。文献[4]通过对随机测试用例的分析，指出动态回溯对于某些问题的执行效率反而比普通回溯算法执行效率更低，甚至相差指数倍。

下面通过一个实例比较动态回溯算法与普通回溯的效率，以直观地说明动态回溯存在的上述问题。考虑如下的 2-SAT 问题：

$$a\rightarrow x \quad (1)$$

$$a\rightarrow b \quad (2)$$

$$b\rightarrow\neg a \quad (3)$$

$$\neg a\rightarrow c \quad (4)$$

$$c\rightarrow d \quad (5)$$

$$x\rightarrow\neg d \quad (6)$$

设动态回溯算法按变量出现顺序选择变量。首先算法设 a 为 1(TRUE)，然后考虑 x，由于 a 为 1，根据式(1)，x 不能取 0，因此 x 被赋值 1。然后考虑 b，由于 a 为 1，因此根据式(2)，b 不能取 0，又根据式(3)，b 不能为 1，此时 b 可以选择的值域集合为空，发生第 1 次回溯。此时情况如表 1 所示。

表 1 第 1 次回溯时各变量及其 eliminating explanation 的情况

变量	数值	eliminating explanation	
		1(TRUE)	0(FALSE)
a	1	-	-
x	1	-	a
b	-	a	a
c	-	-	-
d	-	-	-

根据算法，此时舍弃 a 的值，保留 x 的值。随着算法的执行，a 被赋值 0，b 被赋值 1，c 被赋值 1。当处理到变量 d 时，由 c 为 1，根据式(5)，d 不能为 0，又由式(6)，d 不能为 1。此时 d 可以选择的值域集合为空，发生第 2 次回溯。此时情况如表 2 所示。

表 2 第 2 次回溯时各变量及其 eliminating explanation 的情况

变量	数值	eliminating explanation	
		1(TRUE)	0(FALSE)
a	0	a	-
x	1	-	-
b	1	-	-
c	1	-	a
d	-	x	c

此时将舍弃 c 的值。并在 c 的 eliminating explanation 中添加元组$(1,x)$，即 x 当前的值使 c 不能为 1。然后选择处理变量 c，但此时 c 的可取值集合为空，进行第 3 次回溯。此时各变量及其 eliminating explanation 情况见表 3。

表 3 第 3 次回溯时各变量及其 eliminating explanation 的情况

变量	数值	eliminating explanation	
		1(TRUE)	0(FALSE)
a	0	a	-
x	1	-	-
b	1	-	-
c	-	x	a
d	-	x	-

这时算法将改变 x 的值为 0。随后为 b、c、d 均赋值 1，算法结束，见表 4。

表 4 算法结束时各变量及其 eliminating explanation 的情况

变量	数值	eliminating explanation	
		1(TRUE)	0(FALSE)
a	0	a	-
x	0	a	-
b	1	-	-
c	1	-	a
d	1	-	c

从上面分析可以看出，采用动态回溯算法要进行 3 次回溯。而如果采用普通的回溯算法，当发现无法为 b 赋值时，将直接回溯到 a，同时删除 x 的值。其后将不会发生回溯。因此，用普通的回溯算法解决该问题只会发生 1 次回溯。问题的根源就在于动态回溯算法在第 1 次回溯时保留了 x 的值，从而导致与后面的 c、d 均发生冲突，增加了回溯次数。

4 图分割技术

图分割的目的是将图分割成一些区域集合，使这些集合中都具有大致相同数目的节点，且连接不同区域的边数最少。该技术已被广泛应用于许多领域[6]。Metis[7]软件包可以对大规模的图在短时间内给出高质量的划分。

分割区域数 k 取决于变量及约束条件的个数和域的大小，还可能依赖于问题约束性的强弱。文献[8]中指出分割的数目主要依赖于变量数 n、域的大小 d 和影响相对较小的约束条件数 gd。在求解过程中，一个合适的分割数是实现最优化求解所必需的。一个近似的确定分割数 k 的公式如下：

$$k = \lfloor \ln(n) + \ln(d) + \lg(gd) \rfloor$$

其中，n 表示变量的个数；d 表示域的大小；gd 表示约束个数。

5 基于图分割的动态回溯算法

为克服动态回溯的不足，本文提出将图分割技术应用于动态回溯算法。其核心思想是在发生回溯时，不保留全部已赋值变量的值，而是只保留部分变量的值。图分割技术可以用来确定保留哪些变量的值，舍弃哪些变量的值。

一个二元约束系统可以建模为一个图：其中的每一个变量作为一个节点，如果 2 个变量之间存在约束关系，则在图中存在连接相应节点的一条边。图分割算法可以快速地将节点分成 k 个集合，每个集合中的节点数大致相等，且连接这些集合的边数最少。这也是选择图分割算法作为划分依据的重要原因。因为每个节点数基本相等可以保证每个子问题的规模相当，有利于降低算法的平均时间复杂度；连接这些集合的边数最少，可以保证各子问题间的相互影响最小。

算法的基本步骤是：首先根据各约束建立约束网络图。然后在该图上执行图分割算法将变量分成 k(k 值的确定在第 4 节中给出)个集合。在算法执行过程中，当发生回溯时，舍弃 culprit variable 及其后出现的且与其在同一集合中的变量的值，保留其余变量的值。

算法的详细描述如下：

(1)建立约束网络图 G。每个变量作为一个节点，如果 2 个变量之间存在约束关系，则在图中添加一条连接相应节点的边。

(2)计算 k 值。

(3)对图 G 执行图分割算法，将变量分成 k 个集合。

(4)设当前部分解 $P=\phi$，对于每个变量 x，设 $E_x=\phi$。

(5)如果 $\overline{P}=X$，则 P 是一个解，算法结束。否则，选择一个变量 $x \in X-\overline{P}$，设 $E_x = E_x \cup e(P,x)$。

(6)令 $S = dom(x) - \overline{E_x}$。如果 S 不空，则从 S 中选择一个值 $v \in S$，在当前部分解 P 中添加 (x,v)。回到第(5)步。

(7)如果 S 为空集，即 $\overline{E_x} = dom(x)$，设 E 是变量 x 的所有 eliminating explanation 构成的集合。

(8)如果 E 为空集，则该 CSP 问题无解，返回。否则，设 (j,v_j) 满足 j 在 E 中出现，且该元组在 P 中最靠后。首先从 P 中删除 (j,v_j) 及出现在 j 后面的且与 j 在同一集合的各变量及其当前值。然后对于在删除变量后面被赋值的每个变量 k，从 E_k 中删除所有涉及被删除变量的 eliminating explanation。在 E_j 中添加元组 $(v_j, E \cap \overline{P})$。转到第(5)步。

6 实验比较与分析

下面用应用图分割的动态回溯算法对第 3 节中的实例进行分析，以比较改进算法与原动态回溯算法的效率。

首先建立约束网络图，如图 1 所示。该问题涉及 5 个变量和 6 个约束，每个变量的论域大小为 2。根据第 4 节的讨论，确定集合划分数为 3。应用 Metis 图分割软件包将变量分为 3 个集合，结果为 $\{a, x\},\{b\},\{c, d\}$。

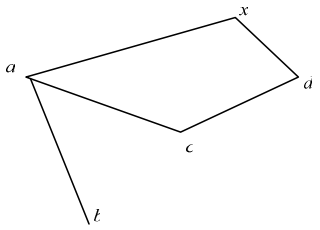


图 1 约束网络图

在发生第 1 次回溯时，各变量情况与表 1 相同。根据算法，将删除 a 的值，由于 a 与 x 在同一集合，因此 x 的值也将删除。随后将 a 赋值为 0，将 x 赋值为 0。后面算法的执行不会发生回溯。这个过程只进行一次回溯。

从对以上实例的分析可以看出，应用图分割的动态回溯算法比原动态回溯算法具有更少的回溯次数。

下面通过实验比较原动态回溯算法及应用图分割改进后算法的执行效率。采用随机问题对算法进行测试。本文主要通过回溯次数对两者的性能进行比较。

随机约束满足问题由随机约束满足问题生成器生成(http://www.lirmm.fr/~bessiere/generator.html)。其产生的问题具有随机性和通用性，多数约束满足问题的算法都使用该方法进行测试。问题的难度和规模可以通过参数进行设置。二元随机约束满足问题有 4 个描述参数(n,m,$p1$,$p2$)，其中，n 表示变量的个数；m 表示变量论域的大小(假设所有变量的论域大小相同)；$p1$ 表示此约束满足问题的密度，即约束图中边的个数(小于等于完全图中边的个数 $n\times(n-1)/2$)的比值；$p2$ 为每个二元约束的松紧度。

本文选择 2 组参数进行比较：(1)n=30，m=30，$p1$=220；(2)n=100，m=20，$p1$=250。2 组参数对于不同 $p2$ 的回溯次数对比如表 5、表 6 所示。

表 5 n=30、m=30、$p1$=220 时随机问题回溯次数对比

$p2$	回溯次数	
	动态回溯	带有图分割的动态回溯
300	31 558	19 330
350	29 722	13 762
400	15 706	9 690
450	17 560	10 147
500	9 720	7 157
550	7 387	6 522
600	8 628	5 442

表 6 n=100、m=20、$p1$=250 时随机问题回溯次数对比

$p2$	回溯次数	
	动态回溯	带有图分割的动态回溯
150	57	16
175	11 430	38
200	143 321	28 028
225	87 439	32 762
250	65 889	30 255
275	30 456	17 363
300	27 774	17 553

从上述实验比较可以看出，采用图分割的动态回溯算法，无论对于大、小规模的随机约束满足问题，效率都得到了大幅提高。这是因为新算法在回溯时没有保留当前变量的全部值，而只保留了与危害变量不在同一集合的变量的值，从而防止了新冲突的引入，减少了回溯次数，提高了效率。

7 结束语

本文通过分析动态回溯算法存在的问题，对普通动态回溯进行优化。通过对约束网络应用图分割，将变量分为若干个集合。在发生回溯时，不同于普通动态回溯那样保留全部变量的值，而是只保留那些与 culprit variable 不在同一集合的变量的值，这样有效地避免了新冲突的引入，减少了回溯次数。实验结果表明，该算法在求解没有明显子问题结构的约束满足问题时具有明显优势。

参考文献

[1] 孙吉贵, 朱兴军, 张永刚, 等. 一种基于预处理技术的约束满足问题求解算法[J]. 计算机学报, 2008, 31(6): 919-926.

[2] 李占山, 王孜文, 艾 阳, 等. 基于 AC-4 的动态值启发式约束满足问题求解算法[J]. 吉林大学学报: 工学版, 2011, 41(5): 1378-1382.

[3] Ginsberg M L. Dynamic Backtracking[J]. Journal of Artificial Intelligence Research, 1993, 1(1): 25-46.

[4] Baker A B. The Hazards of Fancy Backtracking[C]//Proceedings of the 12th National Conference on Artificial Intelligence. Seattle, USA: [s. n.], 1994: 288-293.

[5] Stallman R M, Sussman G J. Forward Reasoning and Dependency-directed Backtracking in a System for Computer-aided Circuit Analysis[J]. Artificial Intelligence, 1977, 9(1): 135-146.

[6] 杜会盈, 李占山, 李宏博, 等. 图分割在Singleten弧相容算法中的应用[J]. 吉林大学学报: 理学版, 2010, 48(6): 981-986.

[7] Karypis G, Kumar V. Multilevel Algorithms for Multi-constraint Graph Partitioning[C]//Proceedings of 1998 ACM/IEEE Conference on Supercomputing. Washington D. C., USA: ACM Press, 1998.

[8] Salido M A, Barber F. Distributed CSPs by Graph Partitioning[J]. Applied Mathematics and Computation, 2006, 183(1): 491-498.

编辑 张正兴

(上接第 184 页)

[3] 周 勇, 巩敦卫, 郝国生, 等. 交互式遗传算法基于 NN 的个体适应度分阶段估计[J]. 控制与决策, 2005, 20(2): 234-236.

[4] Lora X, Sastry K, Goldberg D E, et al. Combating User Fatigue in Igas: Partial Ordering, Support Vector Machines, and Synthetic Fitness[C]//Proc. of Genetic and Evolutionary Computation Conference. Washington D. C., USA: ACM Press, 2005: 1363-1370.

[5] Wang Shangfei, Wang Xufa. An Improved Interactive Genetic Algorithm Incorporating Relevant Feedback[C]//Proc. of the 4th International Conference on Machine Learning and Cybernetics. Guangzhou, China: [s. n.], 2005: 2996-3001.

[6] Zhang Tong, Oles F J. A Probability Analysis on the Value of Unlabeled Data for Classification Problems[C]//Proc. of the 17th International Conference on Machine Learning. San Francisco, USA: [s. n.], 2000: 1191-1198.

[7] Chapelle O, Zien A. Semi-supervised Classification by Low Density Separation[C]//Proc. of the 10th International Workshop on Artificial Intelligence and Statistics. Piscataway, USA: IEEE Press, 2005: 57-64.

[8] Sindhwani V, Niyogi P, Belkin M. Beyond the Point Cloud: From Transductive to Semi-supervised Learning[C]//Proc. of ICML'05. Bonn, Germany: [s. n.], 2005.

编辑 张正兴