计算机科学 1998 Vol. 25 №. 1

约束满足问题求解途径之比较与分析

The Comparison and Analysis of Solving Approachs to Constrain Satisfaction problems

陈 源 史忠植

（中国科学院计算技术研究所智能计算机科学开放实验室 北京 100080）

摘 要 本文从逻辑、自动机理论、代数方法、连接主义框架和遗传算法的角度深入地探讨了 CSP 问题的不同表示框架和求解风范，详细分析和讨论了不同表示和求解方法的特点以及它们之间的内在联系和可能的结合。

关键字 约束满足，逻辑，自动机，整数线性规划，连接主义，遗传算法

人工智能和计算机科学领域中的许多问题都能看作一类约束满足问题(CSP)，CSP 问题由一组变量 $x_1, x_2, \cdots, x_n$ 对应于各个变量的值域 $R_1, R_2, \cdots, R_n$ 和一组约束条件 $C_1, \cdots, C_m$ 组成。其中每个约束 $C_i(X_{i1}, X_{i2}, \cdots, X_{ij})$ 是笛卡尔积 $R_{i1} \times \cdots \times R_{ij}$ 的一个子集，它指定了相容的变量元组值。CSP 问题的一个解就是满足所有约束的一种变量赋值方案。CSP 问题的求解就是找到所有变量的一个或多个赋值，使约束得到满足。CSP 问题广泛地应用于计算机视觉，电路设计与分析，故障诊断推理，信念维护，任务调度，科学实验规划，CAD 系统和自然语言理解等领域。

一般情况下，CSP 问题的求解是一个 NP 完全问题。经典的 CSP 问题的求解方法是基于树的搜索算法。在此基础上人们提出了各种改进方法，包括弧一致性，路径一致性，前向检查，回跳，向后标记以及一些变量赋值次序和变量值选择的启发式策略。这些方法基本上从元结构(约束图的拓扑结构)，宏观结构(约束之间的关系)以及微观结构(一个变量不同值之间的关系)三方面出发，试图通过缩小搜索空间和寻找最佳搜索路径来提高 CSP 问题的求解效率。

从逻辑角度，可以将 CSP 问题表示成一组等价的一阶谓词公式或命题公式，通过定理证明方法或模型创建寻找问题的解。Vempaty 使用自动机理论表示 CSP 问题。根据约束构造一个最小确定状态有限自动机(MDFA)，使 CSP 问题的求解转换为寻找相应自动机所接收的语言。Davenport 等人提出一个连接主义结构 GENET，将变量-值对作为网络的节点，通过神经网络的演化求解 CSP 问题，由于采取相应的学习策略，GENET 可能跳出局部极小点。Rivin 则将 CSP 问题转化为一个等价的整数规划问题，通过代数方法求出解的数目，并对一些 CSP 问题算法的时间复杂度上界给出了更好的结果。De Kleer 从真值维护系统 ATMS 的框架研究了 CSP 问题的求解，特别是它与一般 CSP 概念：弧、节点、k 一致性、有向 k 一致性的联系和对应。其它的方法包括用遗传算法和爬山法等局部搜索法来求解 CSP。最常见的做法是选取未被满足的约束个数作为目标函数，目标函数为零时，即求得一个解。

下面我们将从逻辑，自动机，整数线性规划，连接主义框架和遗传算法的角度探讨 CSP 问题的不同表示和求解风范，以及不同实现方法的特点和它们之间的联系和交叉。

1. 约束满足问题的逻辑方法

CSP 问题可以用一阶逻辑定理证明，命题逻辑定理证明(SAT)，Prolog 方法，约束逻辑程序设计语言 CLP 以及命题逻辑的模型创建等不同的逻辑框架加以刻画，在不同的逻辑框架内，使用各种标准和非标准的方法进行 CSP 问题求解。

1.1 CSP 问题的一阶谓词逻辑框架

CSP 问题可以严格地用一阶谓词逻辑公式描述如下：

$$\text{Query}: \exists X_1 \exists X_2 \cdots \exists X_n Q(X_1, X_2, \cdots, X_n)$$

$$Q(X_1, X_2, \cdots, X_n) \Leftrightarrow \exists X_1 \exists X_2 \cdots \exists X_n$$
$$p_{x1}(X_1) \wedge p_{x2}(X_2) \wedge \cdots \wedge p_{xn}(X_n) \wedge$$
$$p_{x1x2}(X_1, X_2) \wedge p_{x1x3}(X_1, X_3) \wedge \cdots \wedge$$
$$p_{x1x2x3}(X_1, X_2, X_3) \wedge \cdots \wedge$$
$$\cdots$$
$$p_{x1x2\cdots xn}(X_1, X_2, X_3, \cdots, X_n)$$

约束被表示为基本原子公式集合 Constraints：

$\{p_{xi1i2,\cdots,xim}(c_{i1},c_{i2},\cdots,c_{im})|1\leq i_k\leq i_{k+1}\leq n\}$，$c_i$ 是常量，p 指定了约束所允许的元组值，Constraints 包括所有一致的变量赋值元组。CSP 问题的求解等价于找出使 Constraint⊢Query 成立的 $X_1,X_2,\cdots,X_n$ 值。

这样一个 CSP 问题被形式化为一个(Constraints，Query)二元组，这个理论的 Herbrand 域是 $H=\{c|p(\cdots,c,\cdots)\in Constraints\}$，显然这个 H 域是有限的，存在一个终止的算法 DA 确定 Constraint⊢Query 是否成立：只需对每个元组 $(c_1,c_2,\cdots,c_n)\in H^n$，判断是否有 Constraints⊢$Q(c_1,c_2,\cdots,c_n)$，Constraints⊢$Q(c_1,c_2,\cdots,c_n)$成立当且仅当 $Q(c_1,c_2,\cdots,c_n)$中的每个原子公式 $p\in$Constraints。这个简单算法相当于 CSP 求解中的穷举算法。

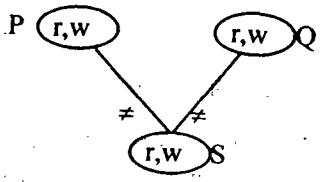

图 1　一个着色问题

图 1 的着色问题可以形式化为：

Query：∃ X∃ Y∃ Z p(X)∧q(Y)∧s(Z) ∧ne(X,Z) ∧ne(Y,Z)
Constraints：{p(r),p(w),q(r),q(w),s(r),s(w),ne(r,w),ne(w,r)}
H={r,w}
H^3={(r,r,r),(r,r,w),…,(w,w,w)}

使用 DA 算法很容易得到当(X,Y,Z)=(r,r,w)时，Constraint⊢Query 成立，即(r,r,w)是问题的一个解。如果将 Query 看作是待证明的定理，则 CSP 有解存在当且仅当 Constraints∪¬Query 不可满足或者说 Constrainst∪¬Query 没有 Herbrand 模型。这样我们就将 CSP 问题转化为一阶逻辑下标准的定理证明问题，可以使用各种定理证明方法进行求解。

1.2　CSP 问题的命题逻辑框架

对 CSP 问题的另一种不同刻画是将它作为命题逻辑的的模型创建问题。在这个框架下，一个与原始 CSP 问题相对应的合取范式 F 被创建，F 中每个析取子式包括三类事实：一个变量有多个可能结合的值；一个变量必须结合且只能结合到一个值；变量之间必须满足相应的约束关系。图1着色问题的一种编码如下：

F={P=r∨P=w,Q=r∨Q=w,S=r∨S=w,
¬P=r∨¬P=w,¬Q=r∨¬Q=w,¬S=r∨¬S=w,
¬P=r∨¬S=r,¬P=w∨¬S=w,¬Q=r∨¬S=r,¬Q=w∨¬S=w}

显然在这个框架下，F 的一个模型对应 CSP 问题的一个解。为了找出 F 的模型，我们可以使用 Davis-Putnam 算法。考虑到在这种编码方法下，F 中没有混合子句，我们可以利用 AC-归结策略得到简化公式 F′。AC-归结有如下性质：F 和归结后 F′具有同样的模型，但 F′的子句数目和子句长度小于等于 F。

由于这实际上是一个 SAT 问题，我们可以在 CSP 和 SAT 问题之间建立一些有意义的联系。如果 CSP 问题每个变量至多有两个值，根据上面的编码方法，我们得到了一个2-SAT 问题(图1的 CSP 问题即是这样的例子)，由于2-SAT 问题可以在多项式时间求解，可以得出相应的 CSP 问题也可以在多项式时间内求解。

对于不同结构的 CSP 问题，可以使用不同的编码方法。对于约束比较稀疏的 CSP 问题和有向约束网络问题(约束涉及的变量分为两类：输入变量和输出变量，约束对输入变量值不加限制)，另外一种混合编码方法被证明是更有效的。

可以看出，逻辑方法通过把 CSP 问题编码为相应逻辑框架下的一组公式，将 CSP 问题的求解转化为在一阶谓词逻辑下的定理证明或者命题逻辑中理论的模型创建，特别地，可以将 CSP 问题转化为一个 SAT 问题。这种方法使我们可以充分利用数理逻辑中的方法和理论来研究 CSP 问题，得出有用的结论。

2. 约束满足问题的有限自动机方法

Vempaty 等人将 CSP 问题表示成一个具有最小状态的确定有限自动机 MDFA，把 CSP 的解看作是自动机所接受的语言，运用相应的自动机理论进行求解。一个有限状态自动机是一个五元组 DFA(Q,D,δ,S,F)，其中：(1)Q 是有限的内部状态集合；(2)D 是输入符号的字母表；(3)δ 是状态转化函数；$Q\times D\to Q$；(4)S 是输入状态；(5)F 是接受状态集合。

给定一个 CSP 问题，我们可通过下面的规则创建有限自动机 DFA：(1)空约束对应空 DFA，空 DFA 没有状态，接收语言为空集；(2)对约束 C 允许的每个元组$(a_1,a_2,\cdots a_n)$，生成一个相应的 DFA，它接收唯一的串 $a_1a_2\cdots a_n$。约束 C 对应的自动机通过对约束中每个元组对应的自动机进行 OR 操作得

到;(3)已知两个约束C_1,C_2和它们对应的DFAs:$M_1(Q_1,D_1,\delta_1,S_1,F_1)$和$M_2(Q_2,D_2,\delta_2,S_2,F_2)$,可以构造$C_1 \vee C_2$,$C_1 \wedge C_2$和$C_1$—$C_2$对应的自动机$M_p(Q,D,\delta,S,F)$,$M_p=\{(q_1,q_2)\mid q_1\in Q_1 \text{ and } q_2\in Q_2\}$,$\delta((q_1,q_2),a)=(\delta_1(q_1,a),\delta_2(q_2,a))$,这相当于将a同时作用于$M_1$和$M_2$上。开始状态$S=S_1\cup S_2$,接收状态F根据具体操作符决定。OR:$F=\{(q_1,q_2)\mid q_1\in F_1 \text{ or } q_2\in F_2\}$;AND:$F=\{(q_1,q_2)\mid q_1\in F_1 \text{ and } q_2\in F_2\}$;Diff:$F=\{(q_1,q_2)\mid q_1\in F_1 \text{ and } q_2\notin F_2\}$;(4)给定约束C和它对应的DFA $M(Q,D,\delta,S,F)$,可以构造¬C对应的自动机$M'(Q,D,\delta,S,F')$,M'与M唯一不同是接收状态,$F'=\{q\mid q\notin F\}$。(6)在每一步生成相应的DFA后,我们都通过状态等价算法得到等价的具有最小状态的确定有限状态自动机MDFA。由于接收同一种语言的最小DFA都是同构的,通过这种构造方法,一个CSP问题得到唯一的一个MDFA,相应的CSP问题转化为自动机问题:

可满足性　CSP可满足等价于MDFA有接收状态

有效性　求CSP问题的所有解可以通过深度优先遍历MDFA得到

等价性　两个CSP等价⇔相应的MDFA同构

下面将给出利用有限自动机求解CSP的步骤:

1. 对于约束允许的每个元组值构造一个最小有限自动机MDFA;

2. 对于每个约束C,通过AND操作构造出约束对应的最小有限自动机;

3. 根据约束之间的与或关系,由约束对应的MDFA通过AND,OR,DIFF等操作构造出CSP问题对应的有限自动机DFA,最小化得到MDFA;

4. 在MDFA上进行CSP问题求解。

图2是一个简单的例子:利用自动机方法求解图1的CSP问题。

自动机方法将约束表示和约束推理统一在有限自动机中,不仅提供了CSP问题的一种规范表示,而且它允许渐增地创建和操作约束。一旦建立起相应的自动机,我们可以对相似约束问题进行复用,充分利用以前的计算结果。比如上面例子中,在P与Q之间加入了新的约束关系C_{PQ},我们只需要建立C_{PQ}对应的MDFA,然后将它与以前得到的MDFA进行AND操作,这样就生成了新CSP问题对应的MDFA。这种计算的复用是传统的方法所不具备的,它使自动机方法非常适合于渐增地加入约束。对于同一个CSP问题的不同查询,自动机方法也避免了代价高昂的重新计算和回溯。另外,自动机方法允许约束间有任意的逻辑关系,而不仅仅是简单的AND关系。所有这些特点使MDFA方法很适合于用来建造知识库,用于存储约束和进行约束推理。

自底向上创建的有限自动机在结构上对应着一个自顶向下的回溯搜索树。MDFA实际上是将搜索树中等价状态规约为一个单一状态,将失败节点删除而得到的图。不同的自动机合成次序将导致不同的时间和空间消耗,传统搜索方法中一些变量赋值次序的启发式策略完全可以应用到自动机方法中。进一步我们可以研究和探讨什么样的CSP问题适合于自底向上的方法,什么样的CSP问题适合于自顶向下的方法。

3. 约束满足问题的代数方法

Rivin等人给出了一种将搜索问题与代数方法相结合的CSP问题求解方法:将一个CSP问题形式化为一个整数线性规划问题,通过多项式乘积进行求解。这种方法的一个优点是它可以方便地决定解的数目,并且可以对一些CSP问题给出更简单的时间复杂度上限估计。

1. 每个可能的赋值$V_i\leftarrow d_j$,我们引进一个新的变量X_{ij}。对于每个变量V_i,有方程:

$$\sum_{1\le i\le m} X_{i,j}=1 \qquad (1)$$

它表达了这样一个事实:每个解中变量有且仅

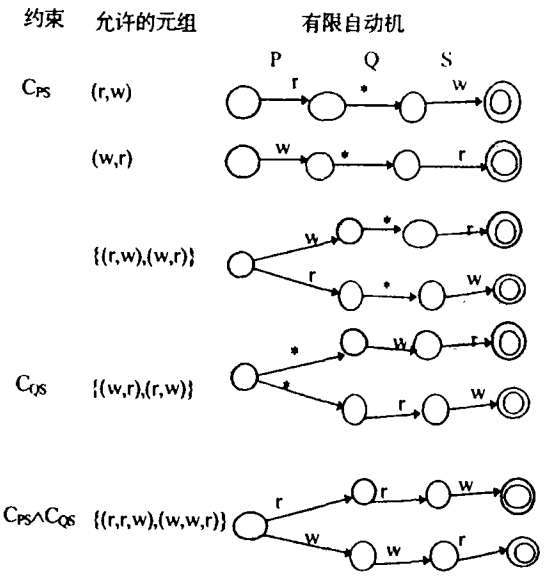

图2　使用MDFA求解图着色问题

有一个值。

2.对于每个不一致的赋值元组，比如$\{V_i \leftarrow d_j, V_k \leftarrow d_l\}$有方程：

$$X_{i,j}+X_{k,l}\leq 1 \qquad (2)$$

显然，方程有 n·m 个变量 $X_{i,j}$，方程组的所有解 $X_{i,j}$为1或0，解 $X_{i,j}=1$对应于 CSP 解中有赋值 $V_i \leftarrow d_j$。这样一个 CSP 问题就被转化成了一个标准的整数线性规划。对0-1规划的求解，运筹学中有方法和技巧，在这里就不介绍了，我们仅给出几个结果。

·CSP 问题解的数目。n 个变量，每个变量取值数为 m 的 CSP 问题对应于线性方程组：

$$\sum_{j=1}^{N}\omega_{i,j}x_j=s_i, i=1,\cdots,M$$

其中有 $s_i=1$ or 0，$\omega_{i,j}=1$ or 0，$N=mn$

定义生成函数 $\Phi(Y_1,\cdots,Y_M)$

$$\Phi(Y_1,\cdots,Y_m)=\prod_{j=1}^{N}(1+i\prod_{i=1}^{M}Y_i^{\omega_{i,j}})$$

方程的解数目就是生成函数 Φ 中项 $\prod_{i=1}^{M}Y_i^{S_i}$ 的系数。

对于形如 $\sum_{1\leq i\leq m}X_{i,j}\leq 1$的方程组，它实际上蕴含了两个等式方程，一是 $\sum_{1\leq i\leq m}X_{i,j}=1$，二是 $\sum_{1\leq i\leq m}X_{i,j}=0$，不等式的解的个数就是两个等式方程解数目的和，由于它们的生成函数是一样的，我们可以避免不必要的计算。如果方程组中有 α 个不等式，我们用等式替换不等式，计算各种组合情况下生成函数项 $\prod_{i=1}^{M}Y_i^{s_i}$ 的系数值，方程组的解数目就是 2^{α} 个系数和。

·找出 CSP 问题的解。引进 $N=mn$ 个不同的变量 α_i，定义另一个不同的生成函数 $\Phi(Y_1,\cdots,Y_M)=\prod_{j=1}^{N}(1+\alpha_i\prod_{i=1}^{M}Y_i^{\omega_{i,j}})$。计算出生成函数中项 $\prod_{i=1}^{M}Y_i^{s_i}$ 的系数，这时它不再是整数，而是不同 α_i 乘积项的组合，每个乘积项对应 CSP 问题的一个解。

算法的时间复杂度是 $O(nm2^{M-n})$，M 是 CSP 问题转换得到的方程个数。对于 n 皇后问题，有 $N=n^2$，2n 个方程对应行和列约束，2(2n−1)个方程对应对角线约束，总的方程个数 $M=6n-2$，可以估计出算法时间复杂度是 $O(n^232^n)$。这种方法在搜索问题和代数求解方法之间建立了联系，它使我们可能利用复杂的代数技巧来求解和分析 CSP 问题。例如，可以根据生成函数的性质来估计解的大概数目或得到解数目的上界，这对于估计 CSP 问题的难度是特别有用的。

4. 约束满足问题的连接主义方法

在求解象百万皇后这类问题时，传统的搜索方法常常显得无能为力，一些迭代改进方法通常有更好的效果。其基本思想是先产生一个初始解，它很可能是一个不一致的赋值，然后在可能的赋值空间中对这个解进行修正，尽量减少冲突约束数目，最后达到冲突数为0。这种方法实际上将 CSP 问题转化为一个优化问题，它包括下面介绍的神经网络进化方法，遗传算法以及 Minton 的最小冲突爬山搜索法。尽管这种方法是不完备的，但实验证明这种方法能够在较短的时间内求解一些大规模的 CSP 问题，对于这些 CSP 问题，完备的搜索是不可能的，象 Minton 的方法能在几秒钟内求出一百万皇后问题的一个解。局部搜索方法的一个主要问题是它可能陷入局部极小点，尤其是对于约束比较密集和解的数目相对较少的情况。Davenport 等人提出了一种解决 CSP 问题的联接主义结构 GENET，它在一般网络结构中加入一些启发式学习策略来跳出局部极小点。

·GENET 网络结构。每个变量由一族标记节点表示，每个标记节点对应变量的一个赋值，标记节点有两种状态 on 和 off。一个标记节点的状态 $S_{(i,j)}$表示是否将值 d_j 赋给变量 V_i($S_{(i,j)}$=on，则 $V_i=d_j$)，显然一族中只能有一个标记节点的状态为 on。一个标记节点输出为1如果它的状态是 on，否则输出为0。标记节点之间有连接当且仅当两个标记节点所代表的赋值不相容，每个标记节点的输入是连接到它的所有标记节点的输出加权和：

$$I_{(i,j)}=\sum_{k\in V,l\in D}W_{(i,j)(k,l)}V_{(k,l)}$$

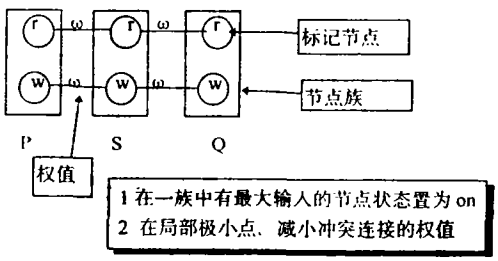

图3 着色问题对应的 GENET 网络

其中 $W_{(i,j)}$是标记节点(i,j)和(k,l)间连接的权值。初始状态时，所有权值被置为−1。根据这种构造，一个标记节点的输入说明了当这个标记节点状态为 on 时，有多少冲突的约束。如果没有冲突，输入将是最大值0。当网络中所有 on 节点的输入都为0时(全局最小点)，相应的 CSP 问题得到求解，状态为 on 的标记节点对应的赋值组成系统的一个解。在一族节点中，GENET 置具有最大输入的节点状态为 on，

当有两个节点输入值都为最大时，随机选择一个。

象大多数 Hill 爬山搜索一样，GENET 可能陷入局部最小点。在一些时候，GENET 能够通过跳到一些具有同等目标值的点来逃脱局部极小点。GENET 中权值可以看作是冲突约束的代价值，一个 GENET 状态的代价值是在这个状态下所有被冲突约束的代价值总和。GENET 在冲突节点间使用如下的权值修改规则：

$$W_{(i,j)(k,l)}(t+1)=W_{(i,j)(k,l)}(t)-V_{(i,j)}V_{(k,l)}$$

GENET 的学习策略可以通过增加局部极小点冲突约束的代价值来逃脱极小点。为了跳出局部极小，在学习后，在局部极小点冲突的约束很少再被冲突。下面给出相应的算法：

1. 随机地产生一个初始状态，每个族中只有一个节点状态为 on

2. 对所有族，同时对族中节点进行状态更新

3. 如果步骤2中没有节点改变状态，则

(a)如果所有 on 节点输入为0，找到一个解，算法终止

(b)否则，激活学习算法

4. 回到步骤2

实验结果表明具有学习策略的 GENET 性能超过简单的基于最小冲突的 Hill 爬山搜索方法。

5. 约束满足问题的遗传算法

前面提到可以将约束满足问题转化为一个优化问题，目标函数为冲突约束的数目，我们可以考虑使用遗传算法来解决 CSP 问题。一种最直接的表示方法如下，将 CSP 问题的解直接编码为串 $a_1a_2\cdots a_n$，代表相应赋值：变量 $V_i \leftarrow a_i$，$i=1,\cdots,n$。串的适应值定义为对应赋值所冲突的约束数目。通过杂交，突变等遗传算法，最终得到适应值为0的串，即问题的解。

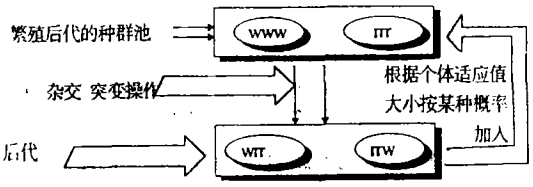

图4 遗传算法求解着色问题示意图

总结 上面我们讨论了 CSP 问题的几种不同的求解途径，包括基于逻辑的方法，自动机方法，整数线性规划方法，连接主义方法和遗传算法等。每种方法基本上都涉及两个问题：①问题表示。将 CSP 问题表示为一个特定领域的问题；②问题求解。一旦得到 CSP 问题的表示，我们可以充分应用相应领域里的方法和技巧进行求解。

不同的方法有着各自不同的特点，逻辑方法由于将 CSP 问题转化为定理证明或模型寻找，可以利用逻辑理论得到一些有用的方法和结论，比如说将 CSP 问题转化为 SAT 问题，由于变量域小于2的 CSP 问题对应的2-SAT 问题有多项式时间算法，我们可以得到相应的 CSP 问题可以在多项式时间内求解。自动机方法将表示和计算统一为最小确定有限自动机，允许动态地创建约束，计算结果可以复用，并且允许约束间任意的逻辑关系，这些特点使它很适合用来构建基于约束表示的知识库。整数线性规划方法使我们能够使用各种巧妙的代数方法来有效地求解和估计问题解的数目以及算法的复杂度，这对于分析 CSP 问题的难度很有意义。连接主义、遗传算法等方法则将 CSP 问题转化为一个优化问题进行求解，尽管不完备，但是对求解问题规模相对比较大的 CSP 问题特别有效。

各种方法之间也存在一些内在的联系。连接主义和整数线性规划等方法都涉及到(变量，值)二元组，它们的基本单元：神经网络中的标记节点和方程中的变量都是(变量，值)对。为了减小问题的规模，可以在求解前利用变量值可交换性进行预处理，删除变量域中冗余的等价值。自底向上构造的自动机方法在结构上对应于一棵自顶向下的搜索树，利用树搜索算法中变量和值的启发式策略来指导自动机构造顺序可以有效地提高求解效率。研究这些不同方法所适合求解的 CSP 问题以及这些方法之间的结合是一个非常有意义的课题。

参考文献

[1] Alan K. Mackworh, The logic of constraint satisfaction, Artificial Intelligence 58, 1992

[2] Nageshwars Rao Vempaty, Solving constraint satisfaction problems using finite state automata, AAAI-92

[3] Igor Rivin and Ramin Zabinh, An algebraic approach to constraint satisfaction problems, IJCAI-89

[4] A. Davenport et al., A connectionist architecture for solving constraint satisfaction problems by iterative improvement, AAAI-94

[5] Euegene C. Freuder, Eliminating interchangeable values in constraint satisfaction problems, in Proc. of AAAI Conf., 1991

[6] 廖乐健，约束满足系统研究，中科院计算所博士学位论文，1994

[7] 刘涛，约束满足问题算法和复杂性，中科院计算所博士学位论文，1994