* **人工智能的不同研究流派**：符号主义/逻辑主义学派--符号智能；连接主义--计算智能；行为主义-低级智能。

人工智能的主要**研究领域**

（一）自动推理（二）专家系统（三）机器学习（四）自然语言理解（五）机器人学和智能控制（六）模式识别（七）基于模型的诊断

**产生式系统**是人工智能系统中常用的一种程序结构，是一种知识表示系统。

**三部分组成**： 综合数据库:**存放问题的状态描述的数据结构, 动态变化的**。产生式规则集、控制系统。

/ 产生式规则集/ 控制系统

**产生式规则形式：** IF<前提条件> THEN<操作>

**八数码难题的产生式系统表示**

综合数据库：以状态为节点的有向图。

状态描述： 3×3矩阵

产生式规则：

* IF<空格不在最左边>Then<左移空格>；依次

控制系统：

选择规则:按左、上、右、下的顺序移动空格。

终止条件：匹配成功。

**产生式系统的基本过程:**

Procedure PROCUCTION

1. DATA←初始状态描述
2. until DATA 满足终止条件，do：
3. begin
4. 在规则集合中，选出一条可用于DATA的规则R（**步骤4是不确定的，只要求选出一条可用的规则R，至于这条规则如何选取，却没有具体说明。）**
5. DATA←把R应用于DATA所得的结果
6. End

**产生式系统的特点：**1.模块性强，2.产生式规则相互独立，3.规则的形式与逻辑推理相近，易懂。

**产生式系统的控制策略**：1.不可撤回的控制策略：优点是空间复杂度小、速度快；缺点是多数情况找不到解 2.试探性控制策略：回溯方式：占用空间小，多数情况下能找到解；缺点是如果深度限制太低就找不到解；和图搜索方式：优点总能找到解，缺点时间空间复杂度高。

**产生式系统工作方式**：正向、反向和双向产生式系统

**可交换产生式系统**：1.可应用性，每一条对D可应用的规则，对于对D应用一条可应用的规则后，所产生的状态描述仍是可应用的。2.可满足性，如果D满足目标条件，则对D应用任何一条可应用的规则所产生的状态描述也满足目标条件。3.无次序性，对D应用一个由可应用于D的规则所构成的规则序列所产生的状态描述不因序列的次序不同而改变。

**可分解的产生式系统**：能够把产生式系统综合数据库的状态描述分解为若干组成部分，产生式规则可以分别用在各组成部分上，并且整个系统的终止条件可以用在各组成部分的终止条件表示出来的产生式系统，称为可分解的产生式系统。**基本过程**：

Procedure SPLIT

1.DATA ← 初始状态描述

2.{Di} ← DATA的分解结果；每个Di看成是独立的状态描述

3.until 对所有的Di ∈{Di}， Di都满足终止条件，do:

4.begin

5. 在{Di}中选择一个不满足终止条件的D\*

6. 从{Di}中删除D\*

7.从规则集合中选出一个可应用于D\*的规则R

8.D ← 把R应用于D\*的结果

9.{di} ← D的分解结果

10.把{di}加入{Di}中

11.end

**回溯算法BACKTRACK过程**：Recursive Procedure BACKTRACK(DATA)

1.if TERM(DATA),return NIL;

2.if DEADEND(DATA),return FAIL;

3.RULES←APPRULES(DATA);

4.LOOP:if NULL(RULES),return FAIL;

5.R←FIRST(RULES);

6.RULES←TAIL(RULES);

7.RDATA←R（DATA）;

8.PATH←BACKTRACK(RDATA);

9．if PATH=FAIL,go PATH;

10.return CONS(R,PATH).

**Procedure GRAPHSEARCH**

1．G←{s}， OPEN ←（s）．

2．CLOSED ←NIL．

3．LOOP：IF OPEN=NIL,THEN FAIL．

4． n ← FIRST(OPEN)，OPEN ←TAIL(OPEN),CONS(n, CLOSED) ．

5． IF TERM(n)，THEN 成功结束

（解路径可通过追溯G中从n到s的指针获得）。

6． 扩展节点n，

令M={m︱ m是n的子节点，且m不是n的祖先} ， G ←G ∪M

7． （设置指针，调整指针）对于mM,

(1)若mCLOSED, mOPEN, 建立m到n的指针，并CONS(m, OPEN).

(2)(a)mOPEN, 考虑是否修改m的指针.

(b)mCLOSED,考虑是否修改m及在G中后裔的指针。

8． 重排OPEN表中的节点（按某一任意确定的方式或者根据探索信息）。

9． GO LOOP

**无信息的图搜索过程：深度优先搜索**：排列OPEN表中的节点时按它们在搜索树中的深度递减排序 。深度最大的节点放在表的前面，深度相等的节点以任意方式排序。**宽度优先搜索**：在排列OPEN表中节点时按它们在搜索图中的深度递增顺序，深度最小的节点放在表的前面。

**A算法**: 使用估价函数f(n)=g(n)+h(n) 排列OPEN表中节点顺序的 GRAPHSEARCH算法。

其中， g(n)：对g\*(n)的一个估计 是当前的搜索图G中s到n的最优路径费用 g(n)≥g\*(n)

h(n)：对h\*(n)的估计，称为启发函数。（Note:若h(n)=0，g(n)=d，则 f(n)=d，为宽度优先）。

**A\*算法**:对任何节点n都有h(n)≤h\*(n)的A算法。

* ***定义：*如果一个搜索算法对于任何具有解路径的图都能找到一条最佳路径，则称此算法为可采纳的。**

**可以证明：A\*算法是可采纳的（如果解路径存在，A\*一定由于找到最佳解路径而结束）**

**A\*算法的可采纳性**：定理1 GRAPHSEARCH对有限图必然终止。定理2 若存在s到目标的路，则算法A\*终止前的任何时刻，OPEN表中总存在一个节点n’， n’在从s到目标的最佳路径上，且满足f(n’) ≤f\*(s) 定理3 若存在从s到目标的解路，则算法A\*必终止。

定理4 算法A\*是可采纳的（即如果解路径存在，A\*一定找到最佳解路径而终止）．

定理5 算法A\*选择的任意扩展点都有f(n)≤f\*(s)

可采纳的条件：1.与或图有解图，2.对图中所有节点n有h(n)≤h\*(n)，3.启发函数满足单调性。则AO\*必然终止并找出最佳解路径。

**影响算法A启发能力的三个重要因素：**

（1）算法A所找到的解路径的费用。（2）算法A在寻找这条解路径的过程中所需要 扩展的节点数。

（3）计算启发函数所需要的计算量。

**启发能力的度量：渗透度P = L / T**  其中，L是算法发现的解路径的长度，

T是算法在寻找这条解路径期间所产生的节点数（不包括初始节点，包括目标节点）

**有效分枝系数是B，则有 B＋B2十…＋BL=T或 B（BL-1）/（B-1）=T**

**8数码启发函数**h(n)=P(n)+3S(n),p(n)是每个硬纸片离开目标位置的和，S(n)是如果一个硬纸片后面的纸片不是它的目标后继则记2，否则记0，如果中心有硬纸片记1，否则记0，然后求和。

渗透度，搜索算法的性能的度量：P = L / T，L是算法发现的解路径的长度，T是算法在寻找这条解路径期间所产生的节点数（不包括初始节点，包括目标节点） 。

有效分枝数B，反映目标搜索的集中程度：设搜索树的深度是L，算法所产生的总节点数为T，则B＋B2十…＋BL=T或B（BL-1）/（B-1）=T

**与/或图**是一种超图．在超图中父亲节点和一组后继节点用超弧连接． 超弧又叫 k-连接符．

k-连接符: 一个父节点指向一组k个有与关系的后继节点，这样一组弧线称为一个k-连接符．

* 极小极大原则：MAX节点在其MIN子节点的倒推值中选max；MIN节点在其MAX子节点的倒推值中选min
* 剪枝规则：（1）α剪枝：如果一个MIN节点的β值小于或等于它的某一个MAX祖先节点的α值，则剪枝发生在该MIN节点之下：中止这个MIN节点以下的搜索过程。这个MIN节点最终的倒推值就确定为这个β值。  
  （2）β剪枝：如果一个MAX节点的α值大于或者等于它的某一个MIN祖先节点的β值，则剪枝发生在该MAX节点之下．中止这个MAX节点以下的搜索过程。该MAX节点的最终返回值可以置成它的α值．
* ND=2(B^D/2)-1（D为偶数）ND=B^(D+1)/2+B^(D-1)/2-1（D为奇数）D为深度，B为平均后继。

定理1 任意公式G都等价于一个前束范式．

证明 通过如下四个步骤即可将公式G化为前束范式．

步骤1：使用基本等价式

F↔H=(F→H) ∧(H→F) F→H=～F∨H

可将公式G中的↔和→删去。

步骤2：使用～(～F)=F和De. Morgan律及引理1，

可将公式中所有否定号～放在原子之前。

步骤3：如果必要的话，则将约束变量改名．

步骤4：使用引理1和引理2又将所有量词都提到公式的最左边。

G=∃x∀y∀z∃u∀v∃wP(x,y,z,u,v,w) 则用a代替x，用f(y，z)代替u，用g(y，z，v)代替w，得公式G的Skolem范式： ∀y∀z∀vP(a,y,z,f(y,z),v,g(y,z,v))

定理2 设S是公式G的子句集．于是，G是不可满足的，当且仅当S是不可满足的．

医生骗子问题：S＝｛P(a),～D(y)∨L(a,y),～P(x)∨～Q(y)∨～L(x, y),D(b),Q(b)｝引理1 设G是仅含有自由变量x的公式，记以G（x），H是不含变量x的公式，于是有

(1) ∀x(G(x)∨H)= ∀xG(x )∨H

(1)’∃x(G(x)∨H)= ∃xG(x )∨H

(2) ∀x(G(x)∧H)= ∀xG(x ) ∧H

(2)’∃x(G(x) ∧H)= ∃xG(x ) ∧H

(3) ～(∀xG(x))= ∃x (～G(x ))

(4) ～(∃xG(x))= ∀x (～G(x ))

引理2 设H，G是两个仅含有自由变量x的公式，分别记以H(x)，G(x)，于是有：

(1) ∀xG(x)∧∀x H(x)= ∀x(G(x )∧H(x))

(2) ∃xG(x)∨∃x H(x)= ∃x(G(x )∨H(x))

(3) ∀xG(x)∨∀x H(x)= ∀x∀y (G(x )∨H(y))

(4) ∃xG(x)∧∃x H(x)= ∃x∃y (G(x )∧H(y))

* 基本等价式

1) (G↔H)=(G→H)∧(H→G)；

2) (G→H)=(～G∨H)；

3) G∨G=G，G∧G=G； (等幂律)

4) G∨H=H∨G，G∧H=H∧G； (交换律)

5) G∨(H∨S)=(G∨H)∨S，  
 G∧(H∧S)=(G∧H)∧S； (结合律)

6) G∨(G∧H)=G，G∧(G∨H)=G； (吸收律)

7) G∨(H∧S)=(G∨H)∧(G∨S)，  
 G∧(H∨S)=(G∧H)∨(G∧S)； (分配律)

8) G∨F=G，G∧T=G； (同一律)

9) G∧F=F，G∨T=T； (零一律)

10) ～(G∨H)= ～G∧～H，  
 ～(G∧H)= ～G∨～H。 (De Morgan律)

11) G∨～G=T；G∧～G=F （互补律）

12) ～～G=G （双重否定律）

将公式∀x∀y（A(x) →B(x,y)）→(∃yC(y) →∃zD(z))化为前束范式

**解：（1）消去→联结词。**

**∀x∀y（A(x) →B(x,y)）→(∃yC(y) →∃zD(z))**

**=** ～**∀x∀y（**～**A(x) ∨ B(x,y)）∨（**～**∃yC(y) ∨∃zD(z))**

**（2）将公式中所有否定号**～**放在原子之前。**

～**∀x∀y（**～**A(x) ∨ B(x,y)）∨（**～**∃yC(y) ∨∃zD(z))**

**= ∃x∃y(A(x) ∧**～**B(x,y)) ∨(∀y**～**C(y) ∨∃zD(z))**

**（3）将约束变量改名.**

**∃x∃y(A(x) ∧**～**B(x,y)) ∨ (∀y**～**C(y) ∨∃zD(z))**

**= ∃x∃y(A(x) ∧**～**B(x,y)) ∨ (∀t**～**C(t) ∨∃zD(z))**

**（4）将量词提到整个公式前。**

**∃x∃y(A(x) ∧**～**B(x,y)) ∨ (∀t**～**C(t) ∨∃zD(z))**

**= ∃x∃y∀t∃z （(A(x) ∧**～**B(x,y)) ∨** ～**C(t) ∨D(z)）**

**= ∃x∃y∀t∃z((A(x)∨**～**C(t)∨D(z))∧(**～**B(x,y)∨**～**C(t)∨D(z)))**

**用a代替x，用b代替y，用f(t)代替z，**

**得公式的Skolem范式：**

**∀t((A(a) ∨** ～**C(t) ∨D(f(t))) ∧(**～**B(a,b) ∨** ～**C(t) ∨D(f(t))))**

**例：1) G=∃x(P(f(x))∧Q(x，f(a)))  
2) H=∀x(P(x)∧Q(x，a))**

**设解释I： D={2，3}，  
 a   
 2  
 f(2) f(3)   
 3 2**

**P(2) P(3) Q(2, 2) Q(2, 3) Q(3, 2) Q(3, 3)   
 F T T T F T**

**TI(G) = TI((P(f(2))**∧**Q(2，f(2)))**∨ **P(f(3))**∧**Q(3，f(2))))  
 = TI((P(3)**∧**Q(2，3))**∨**(P(2)**∧**Q(3，3)))  
 =(T**∧**T)**∨**(F**∧**T)  
 =T**

* **TI(H) = TI(P(2)**∧**Q(2，2)**∧**P(3)**∧**Q(3，2))  
   =F**∧**T**∧**T**∧**F  
   =F**

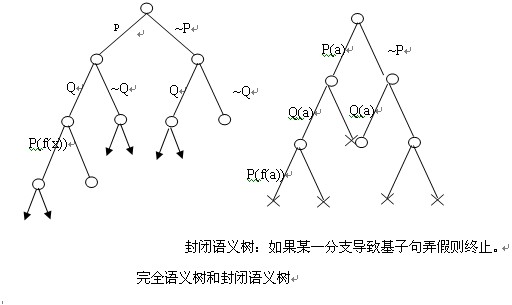
（Herbrand域）设S为子句集，令H0是出现于子句集S的常量符号集。如果S中无常量符号出现，则H0由一个常量符号a组成。 对于i＝1，2，…，令Hi = Hi-1∪｛所有形如f(t1，…，tn)的项｝其中f(t1，…，tn)是出现在S中的所有n元函数符号，tj∈ Hi-1，j＝1，…，n．Hi为S的i级常量集，H∞ 称为S的Herbrand域，简称S的H域。

{P(x)，Q（f(y)VR(y)）}，于是S的H域{a,f(a),f(f(a))…}原子集{P(a),Q(a),R(a),P(f(a))..} 子句的一个基例={Q(f(a))VR(a),Q(f(f(a)))VR(F(a))..};该S的H解释与原子集相同。

**H解释与普通解释的关系**：1、子句集S的H解释是S的普通解释。2、S的普通解释不一定是S的H解释：普通解释不是必须定义在H域上，即使定义在H域上，也不一定是一个H解释。3、任取普通解释I，依照I，可以按如下方法构造S的一个H解释I\*，使得若 S在 I下为真则 S在I\*下也为真。

定理：如果某区域D上的解释I满足子句集S，则对应于I的任意一个H解释I\*也满足S。

语义树：S={～P(x)∨Q(x),P(f(x)), ～Q(f(x))}分别画出S的完全语义树与 封闭语义树。

D-P过程：单文字规则：若S中有一个单元基子句L，令S’为删除S中包含L的所有基子句所剩子句集，则：1) 若S’为空集，则S可满足。(2) 否则，令S’’为删除S’中所有文字～L所得子句集（若S’ 中有单元基子句～L，则删文字～L 得空子句），于是， S恒假 iff S’’恒假。定义（纯文字）：称S的基子句中文字L是纯的，如果～L不出现在S中。纯文字规则设L是S中纯文字，且S’为删除S中所有包含L的基子句所剩子句集，则(1)若S’为空集，则S可满足。(2) 否则，S恒假 iff S’恒假。

分裂规则若S=(A1 ∨L) ∧… ∧ (Am ∨L) ∧(B1 ∨ ～L) ∧ … ∧ (Bn ∨ ～L) ∧R其中A i , Bi ,R都不含L或～L，令 S1 =A1 ∧… ∧ Am ∧R，S2= B1 ∧ … ∧ Bn ∧R 则S恒假 iff S1 ， S2同时恒假。

**归结式** 对任意两个基子句C1和C2。如果C1中存在文字L1，C2中存在文字L2，且L1＝～L2，则从C1和C2中分别删除L1和L2，将C1和C2的剩余部分析取起来构成的子句，称为C1和C2的归结式，记为R(C1, C2)。

**（归结演绎）** 设S是子句集。从S推出子句C的一个归结演绎是如下一个有限子句序列：

C1，C2，…，Ck其中Ci或者是S中子句，或者是Cj和Cr的归结式 (j<i, r< i)；并且Ck＝C。定理：如果基子句集S是不可满足的，则存在从S推出空子句的归结演绎。

归结式简化:若S={P∨C1’, … ,P∨Ci’ , ～P∨Ci+1’,… ,～P∨Cj’,Cj+1 ,… , Cn}则

S’={Ci+1’,… ,Cj’,Cj+1 ,… , Cn} S’’={C1’, … ,Ci’ , Cj+1 ,… , Cn}

合一算法：定义(替换)一个替换是形如{t1/v1, … , tn/vn }的一个有限集合，其中vi是变量符号，ti是不同于vi的项。

合一算法步骤：W={Q(f(a), g(x)), Q(y, y)}, 求W的mgu。

* 步骤1： k=0, W0=W, σ0=ε。
* 步骤2： D0 ={f(a), y}。
* 步骤3：有v0= y ∈D0，v0不出现在t0＝f(a)中。
* 步骤4：令 σ1=σ0⋅{t0/v0}={f(a)/y},

W1={Q(f(a), g(x)), Q(f(a), f(a))}

* 步骤2： D1 ={g(x), f(a) }。
* 步骤3：D1中无变量符号，算法停止，W不可合一。

归结定理的几种改进：支架集归结：子句集S的子集T称为S的支架集，如果（S-T）是可满足的。一个支架集归结是一个不同时属于（S-T）的两个子句的归结。

语义归结：(1)～P∨～Q∨R(2)P∨R(3)Q∨R(4)～R 令 I={～P, ～Q, ～R}，P>Q>R。于是在I下为假的子句只有两个：Si ={P∨R, Q∨R}我们可得如下

PI-演绎：(5)R 由(2)、(3) 、(1) (6) 由(5)、(4)

（线性归结）设S是一个子句集，C0是S中的一个子句。以C0为顶子句，从S到Cn的线性归结演绎是如下一个演绎：对于 i=0, 1, … ,n-1，Ci+1是Ci和Bi的归结式。每个Bi或者属于S，或者是一个Cj，其中j< i。

输入归结：边子句都属于S的线性归结演绎称为输入归结演绎.

（锁归结式） 将子句中的每个文字配上一个整数。最小锁原则（归结最小锁的子句），合并规则（保留最小锁原则）

基于规则的产生式系统：1、如果子表达式E1,…,Ek的父亲节点是（E1∨…∨Ek）,则用k-连接符（带弧）把这些子节点与父节点连接起来，否则用1-连接符（不带弧）。

（替换的相容性集合、合一复合替换）设有替换集合{σ1 ，σ2 ，……，σn}，每一σi 具有如下形式：σi = {ti1/vi1 ，……，tim(i)/vim(i)}其中ti1，……，tim(i) 是项，vi1，……，vim(i)是变量；我们用这些替换构造两个表达式U1和U2如下：

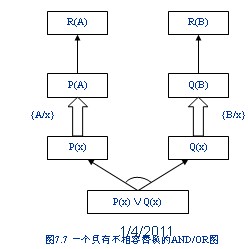
U1 ={v11，……，v1m(1)，……，vn1，……，vnm(n) }

U2 ={t11，……，t1m(1)，……，tn1，……，tnm(n)）}

如果U1 和U2是可合一的，则替换集合{σ1 ，σ2 ，……，σn}称作是相容的，否则称作是不相容的，U1 和U2的最一般合一替换也叫做{σ1 ，σ2 ，……，σn}的合一复合替换。

事实:P(x)∨Q(x)

规则:P(A)→R(A) Q(B)→R(B)

(R(A)∨R(B))不是该AND/OR图的一个子句表示。