# 当前进度汇报：21-01-19

## 1. 搜索空间

### 1.1. 目标空间

期望搜索的目标是符合特定性质的Horn Rules，因此目标空间是符合特定性质的规则的空间。这些性质包括：

1. Atomic：每个谓词的参数都只能是变量

把目标空间记作：$\Omega$

### 1.2. 优化的目标空间

$\Omega$中包含了一些实际意义不大的元素，需要排除：

1. Trivial的元素及其consequence：

   Travial的元素是指Head和Body完全相同的元素，例如：$p(X,Y) \leftarrow p(X,Y)$。这样的规则不包含任何有用的信息，需要排除。同时其所有的consequence也都需要一并排除，例如：$p(X,Y) \leftarrow q(W,X), p(X,Y), r(Y,W)$。

   另外，表达对称性或者参数旋转等价性的规则在压缩的过程中弊大于利（可能造成很多环），也需要连同其consequence一起排除，例如：$p(X,Y,Z) \leftarrow p(Y,Z,X)$。

   因此，这条要求实际排除的标准是Body中不能出现与Head相同functor且**参数集**相同的谓词。

2. Body中包含无意义片段的元素：

   Body中的无意义片段是指Body中的这样的一个子序列：其变量集和剩余规则片段（包括Head）的变量集没有交集。这种片段是没有任何意义的，它们的引入只有两种结果：

   - 片段在$B$中为true，则对剩余部分完全没有影响
   - 片段在$B$中为false，则整条规则不会推出任何结果

   比如规则：$h(X,Y) \leftarrow p(X), q(Y), n(Z,W), m(W,Z)$

   其中的谓词子序列$n(Z,W), m(W,Z)$就是完全无关的片段，它们的存在毫无必要，虽然整条规则是符合约束的，但是却毫无意义。

3. Head中存在重复变量的元素：

   存在重复参数的谓词是没有意义的，目前也没有足够的motivation去挖掘以这样模式作为结论的规则。

排除了以上元素的优化空间记作：$\Omega_m$(m for "modified")

## 1.3. One-step Extension of Rules

定义一些操作将$\Omega_m$中的元素按照特定顺序连接在一起，形成一个可以按照一定顺序遍历的搜索空间。这些操作是对某一个$r \in \Omega_m$进行的一步拓展（$r$的所有一步拓展的元素集合记为：$ext(r)$）

1. 拓展一个已知的变量（此类拓展的元素集记为：$ext_k(r)$，k for "known"）：

   即：1) 在$r$的body中当前还没有确定变量的参数设置为已知变量; 2) 或添加一个新的谓词，并在新谓词中的未确定参数位置尝试已知变量。

   例：

假设$r$为：$h(X,Y) \leftarrow p(?,X)$

其中"?"表示还未确定变量的参数（即自由变量），假设$B$中还存在一个谓词$q/1$，则$r$的已知变量的拓展有：

- $h(X,Y) \leftarrow p(X,X)$
- $h(X,Y) \leftarrow p(Y,X)$
- $h(X,Y) \leftarrow p(?,X), h(X,?)$
- $h(X,Y) \leftarrow p(?,X), h(Y,?)$
- $h(X,Y) \leftarrow p(?,X), h(?,X)$
- $h(X,Y) \leftarrow p(?,X), h(?,Y)$
- $h(X,Y) \leftarrow p(?,X), q(X)$
- $h(X,Y) \leftarrow p(?,X), q(Y)$

2. 拓展一个新的变量（此类拓展的元素集记为：$ext_u(r)$，u for "unknown"）：

   拓展一个新的变量的时候，如果仅在一个未知参数处拓展，则和原本的规则在查询时等价，因为未确定的参数其实际上代表一个独立的自由变量。

   还是举上述的例子$r$，在考虑其效果的时候做的查询等价于以下规则：

   $h(X,Y) \leftarrow p(Z,X)$

   其中$Z$就是一个独立的自由变量。此时在增加新变量的时候新的规则就和原规则等价，不存在梯度，因此一定要在两处未知参数处进行拓展。这两处未知参数可以是$r$中的，也可以是在添加了一个新的谓词之后的。在上例中有以下新变量拓展：

   - $h(X,Y) \leftarrow p(Z,X), h(Z,?)$
   - $h(X,Y) \leftarrow p(Z,X), h(?,Z)$
   - $h(X,Y) \leftarrow p(?,X), h(Z,Z)$

- $h(X,Y) \leftarrow p(Z,X), q(Z)$

在这个搜索空间中，定义两个元素的相邻关系：对于$\forall r, r_e \in \Omega_m, r_e \in ext(r)$，则称$r_e$是$r$的后继，记作$r_e \in sc(r)$；称$r$是$r_e$的前驱，记作$r \in pd(r_e)$。

## 1.4. One-step Extension的性质：Completeness

即：可以通过一步扩展操作遍历整个搜索空间

Proof: 对于任意规则$r$，可以找到这样的构造序列：在序列分为两部分，第一部分逐次将head中的变量在body中的出现的位置构造出来，剩下的位置暂时未知；在序列的第二部分，逐次添加仅在body中出现的变量，当这样的变量第一次出现的时候，同时添加两个，之后可以每次添加一个。按照这样的构造方式，即可构造出$r$，且这个序列中的每一步操作都在前文关于拓展的定义范围之内。

例如规则$h(X,Y) \leftarrow p(X,Z), q(Z,Z,W), r(W,Y)$按照上述流程可以找到序列：

- 第一部分：
  - $h(X,Y) \leftarrow$
  - $h(X,Y) \leftarrow p(X,?)$
  - $h(X,Y) \leftarrow p(X,?), r(?,Y)$
- 第二部分：
  - $h(X,Y) \leftarrow p(X,Z), r(?,Y), q(Z,?,?)$
  - $h(X,Y) \leftarrow p(X,Z), r(?,Y), q(Z,Z,?)$
  - $h(X,Y) \leftarrow p(X,Z), r(W,Y), q(Z,Z,W)$

□

其实，任意规则$r$都可以看做是一条SQL查询，这条查询中的所有条件即为对规则中同名变量所对应的列的等价性约束。例如规则：$h(X,Y) \leftarrow p(X,Z), q(Z,Y)$，转换为SQL查询得到：

```
SELECT * FROM h
WHERE h[0] = p[0] AND h[1] = q[1] AND p[1] = q[0]
```

其中`h`，`p`，`q`分别是三张table，方括号内的数字表示列的位置。需要说明的是，表示Head的表h为：$E_h = E_h^+ \cup E_h^-$；而表示Body中谓词的表只考虑正例，即$E_p^+$和$E_q^+$。设查询的结果为$R$，则：

- $E_r^+ = R \cap E_h^+$
- $E_r^- = R \cap E_h^-$

可以发现，上述的遍历方式其实就是在WHERE语句中逐次添加等价性条件。WHERE语句中的条件数量可以看做是规则的长度，记作$|r|$。

# 2. Evaluation Metric

评价任意规则的函数定义为：$eval : \Omega \rightarrow R$。符合这个要求的Metric有以下几种：

## 2.1. 压缩比

$$\tau(r) = \frac{|E_r^+|}{|E_r^+| + |E_r^-| + |r|}$$

其中$E_r^+$表示$r$可以推出的正例，$E_r^-$表示$r$推出的负例（也就是$r$的副作用），$|r|$表示规则的长度。当$E_r^+ = E_r^- = \emptyset$时，定义$\tau(r) = 0$，则：$\tau(r) \in [0, 1]$。当$\tau(r) \leq 0.5$时，$r$不会起到压缩的效果。

### 2.1.1. 压缩比的性质：Local Optimal

即：Ground Truth在搜索空间中一定是局部最优。其中Ground Truth指：$r_{max} = \arg\max_r \tau(r)$。

Proof: 当$B$足够大的时候，$|r|$对$\tau(r)$的影响可以忽略不计，此时$r_{max} = \arg\max_r \frac{|E_r^+|}{|E_r^+| + |E_r^-|}$。对于任意$r_{sc} \in sc(r_{max})$来说，其添加的条件最多使得$\frac{|E_r^+|}{|E_r^+| + |E_r^-|}$不会减小，但是$|r|$会增大，因此$\tau(r)$还是会减小，因此$r_{max}$是使得$\frac{|E_r^+|}{|E_r^+| + |E_r^-|}$最大的最短规则。因此对于任意$r_{pd} \in pd(r_{max})$，$eval(r_{pd}) < eval(r_{max})$，从而$r_{max}$是局部最优。

□

### 2.1.2. 压缩比的性质：Necessity

即：$r_{max}$在这种搜索方式中一定存在一条路径，使得路径上的每一步都有一个向上的梯度。

Proof: 根据2.1.1，$\forall r_{pd} \in pd(r_{max}), eval(r_{pd}) < eval(r_{max})$，说明$r_{max}$的Body中的每一个等价性条件的丢弃都会使得$\frac{|E_r^+|}{|E_r^+| + |E_r^-|}$减小，因此向一个空的WHERE语句中逐次添加这些等价性条件的时候都会使得$\frac{|E_r^+|}{|E_r^+| + |E_r^-|}$增大，而逐次增

加这些等价性条件即是构造了Ω中的一条搜索路径。

□

### 2.1.3. 压缩比的性质：Length Sensitive

即：在这种搜索方式下，理论上不需要预设一个关于规则长度的阈值限制。

Proof: 一条搜索路径的长度一定小于所有可以向WHERE语句中添加的等价性条件的数量，即为：$\left(\sum_{p\in B}\phi(p)\right)^2$。而$B$是有限的，因此这个上界是有限的。

□

需要说明的是，如果不设置搜索长度限制，AMIE是有可能不停机的（一直在添加dangling term）。

## 2.2. 压缩量

$$\delta(r) = |E_r^+| - |E_r^-| - |r|$$

其中$E_r^+$表示$r$可以推出的正例，$E_r^-$表示$r$推出的负例（也就是$r$的副作用）。$\delta(r) \in [-|E_r^-|, |E_r^+|]$，当$\delta(r) \leq 0$时，$r$不会起到压缩的效果。

（这个Metric是不是也有2.1的同样的性质？）

## 2.3. 覆盖率

$$\rho(r) = \frac{|E_r^+|}{|E_p^+|}$$

其中：$p = r.head$，$E_p^+$表示谓词$p$的所有正例

# 3. 一个新的搜索方法

根据上述分析，可以设计一种在$\Omega$中进行类似“梯度下降”的方式，这里的梯度就是指的$eval(r)$上升最快的路径，算法如下：

> **Input**: 知识库$B$
>
> **Output**: $r_{max}$
>
> ---
>
> $r \leftarrow \underset{r \in \Omega, r\text{仅有}Head}{\arg\max} \; eval(r)$
>
> **While** true **do**:
>
> > $L \leftarrow \emptyset$
> >
> > **For** each $r_e \in sc(r) \cup pd(r)$ **do**:
> >
> > > **If** $eval(r_e) > eval(r)$ **then**:
> > >
> > > > $L \leftarrow L \cup r_e$
> > >
> > > **End If**
> >
> > **End For**
> >
> > **If** $L = \emptyset$ **then**:
> >
> > > **Return** $r$
> >
> > **Else**:
> >
> > > $r \leftarrow \underset{r_e \in L}{\arg\max}\, eval(r_e)$
> >
> > **End If**

> End While

# 4. 这个搜索过程的缺点

## 4.1. Local Optimal

这个问题根本原因是当前的梯度只能作为目标规则的必要条件而不是充分条件。虽然$r_{max}$是Local Optimal，但是反之不成立，因此在当前实验的运行过程中出现了一些质量并不太好的局部最优解。

例如，在手动构造的家庭关系知识库中，有$sibling, brother, sister$三个谓词。手动解释$sibling$的最优解为：

- $sibling(X,Y) \leftarrow brother(X,Y)$
- $sibling(X,Y) \leftarrow sister(X,Y)$

但是在实际运行中，则得出了局部最优解：

- $sibling(X,Y) \leftarrow aunt(Z,X), brother(X,Y), aunt(Z,Y)$

或

- $sibling(X,Y) \leftarrow aunt(Z,X), aunt(Z,Y)$

这种情况就是在取得了很高的压缩率的情况下，覆盖率却很低。

## 4.2. 尝试次数多

虽然当前的这个过程已经将一个纯剪枝的过程提升为了可以按照梯度的方式进行有向搜索的过程，但是不得不说对于一个较长的$r$的扩展是很多的，即使通过梯度排除了一些，但是剩下的也不少，而且在计算梯度的过程中的数据库查询操作开销很高。

尝试次数多的另外一个可能原因是搜索存在重复，因为一条规则只要足够长，就存在不止一条符合搜索条件的构造序列，实际可以构造它的序列不会比这些更少。

目前考虑解决这个问题的思路有两条：

1. 寻找对得分的估计方法
2. 寻找一种更好的搜索过程，保持当前的性质，同时还能减少分支，减少重复

## 4.3. 搜索路径虽然不是无限的，但是有时候也很长

这个问题对于一些存在“传递”性质的谓词尤其明显，比如$brother$关系，在实验中就曾出现过这样的规则：

$$brother(X,Y) \leftarrow brother(X,Z), brother(Z,W), brother(W,Y)$$

这个问题还没想好怎么解决。

## 4.4. 参数不能设置为常量

目前对于规则中可能出现的常量参数还没有考虑，后续可以考虑加入对这种参数的考察。但是常量对于知识库来说过多（比如我用来测试的数据库，1100个facts的时候常量就有142个，对于任意一条$r$来说，这都是比可以尝试的变量数量多好几个数量级的存在），可能需要通过一些预处理的方式进行剪枝才不

会引入过多的分支。比如，对于一些参数的对应常量列表分布相对集中于少数几个常量的情况下才考虑把常量纳入考虑的范围。例如$gender(X,Y)$的第二个参数基本只有$male$和$female$两个（其他的基本都可以看做数据的噪音），此时可以考虑这里的$Y$被替换为二者其一。而$sibling(Z,W)$的两个参数的分布都很分散，不需要考虑替换为常量的可能性。

## 4.5. 对递归规则的处理还需要继续探索

现在的试验中还没有考虑通过递归的规则进行压缩的状况，这种情况还有待继续探索。