ALC计算开销上界下界都不友好

EL表达力比ALC弱，生物医学本体都用表达力很弱的EL

> **role之间的包含关系称为 role inclusion，用字母 H 表示**
>
> **EL中加入 role inclusion称为 ELH；同样ALC称为 ALCH。**

ALC = EL + negation

> **Tableau算法不再适用，因为不能取反**

表面上 ALC 比 EL 多了 negation、disjunction、forall、bottom concept 四个构造元素，实际上只多了 negation，因为 disjunction = negation + conjunction，forall = negation + exists，bottom concept = negation + top concept，所以我们只需要考虑 negation，conjunction，exists，top。

EL 语言下的 concept 都是 satisfiable 的，ontology 都是 consistent的

一、定义

1、基础符号

$\mathcal{EL}$ concepts are defined inductively as follows:

- all concept names are $\mathcal{EL}$ concepts
- $\top$ is a $\mathcal{EL}$ concept
- if C and D are $\mathcal{EL}$ concepts and r is a role name, then

$$C \sqcap D, \quad \exists r.C$$

  are $\mathcal{EL}$ concepts.
- nothing else is a $\mathcal{EL}$ concept.

举例

- $\exists$**hasChild**.$\top$ (somebody who has a child),
- **Human** $\sqcap$ $\exists$**hasChild**.$\top$ (a human who has a child),
- **Human** $\sqcap$ $\exists$**hasChild.Human** (a human who has a child that is human),
- **Human** $\sqcap$ $\exists$**gender.Female** (a woman),
- **Human** $\sqcap$ $\exists$**hasChild**.$\top$ $\sqcap$ $\exists$**hasParent**.$\top$ (a human who has a child and has a parent),
- **Human** $\sqcap$ $\exists$**hasChild**.$\exists$**gender.Female** (a human who has a daughter),
- **Human** $\sqcap$ $\exists$**hasChild**.$\exists$**hasChild**.$\top$ (a human who has a grandchild).

2、Concept definition

Let A be a concept name and C a $\mathcal{EL}$ concept. Then

- $A \equiv C$ is called a **concept definition**. C describes necessary and sufficient conditions for being an A. We sometimes read this as "A is equivalent to C".
- $A \sqsubseteq C$ is a **primitive concept definition**. C describes necessary conditions for being an A. We sometimes read this as "A is subsumed by C".

- 要求左边必须是concept name，同一概念只能定义一次（很难矛盾）
- 但可以循环定义（没有循环定义题目会特殊说明是acyclic）

However, we can have cyclic definitions such as

$$\mathsf{Human\_being} \equiv \exists \mathsf{has\_parent}.\mathsf{Human\_being}$$

A **acyclic $\mathcal{EL}$ terminology $\mathcal{T}$** is a $\mathcal{EL}$ terminology that does not contain (even indirect) cyclic definitions.

3、Concept inclusion&equation

We generalise $\mathcal{EL}$ concept definitions and primitive $\mathcal{EL}$ concept definitions. Let C and D be $\mathcal{EL}$ concepts. Then

- $C \sqsubseteq D$ is called a **$\mathcal{EL}$ concept inclusion**. It states that every C **is-a** D. We also say that C is subsumed by D or that D subsumes C. Sometimes we also say that C is included in D.
- $C \equiv D$ is is called a **$\mathcal{EL}$ concept equation**. We regard this as an abbreviation for the two concept inclusions $C \sqsubseteq D$ and $D \sqsubseteq C$. We sometimes read this as "C and D are equivalent".

Examples:

- $\mathsf{Disease} \sqcap \exists \mathsf{has\_location}.\mathsf{Heart} \sqsubseteq \mathsf{NeedsTreatment}$
- $\exists \mathsf{student\_of}.\mathsf{ComputerScience} \sqsubseteq \mathsf{Human\_being} \sqcap \exists \mathsf{knows}.\mathsf{Programming\_Language}$

- 更宽泛，不要求左面一定是单个concept

> 除了concept name 就是乱七八糟的EL concepts

区别

- Every $\mathcal{EL}$ concept definition is a $\mathcal{EL}$ concept equation, but not every $\mathcal{EL}$ concept equation is a $\mathcal{EL}$ concept definition.
- Every primitive $\mathcal{EL}$ concept definition is a $\mathcal{EL}$ concept inclusion, but not every $\mathcal{EL}$ concept inclusion is a primitive $\mathcal{EL}$ concept definition.

二、EL Tbox

A $\mathcal{EL}$ **TBox** is a finite set $\mathcal{T}$ of $\mathcal{EL}$ concept inclusions and $\mathcal{EL}$ concept equations.
Observe:

- Every acyclic $\mathcal{EL}$ terminology is a $\mathcal{EL}$ terminology;
- every $\mathcal{EL}$ terminology is a $\mathcal{EL}$ TBox.

Example:

$$\begin{array}{rcl}
\textsf{Pericardium} & \sqsubseteq & \textsf{Tissue} \sqcap \exists \textsf{cont_in}.\textsf{Heart} \\
\textsf{Pericarditis} & \sqsubseteq & \textsf{Inflammation} \sqcap \exists \textsf{has_loc}.\textsf{Pericardium} \\
\textsf{Inflammation} & \sqsubseteq & \textsf{Disease} \sqcap \exists \textsf{acts_on}.\textsf{Tissue} \\
\textsf{Disease} \sqcap \exists \textsf{has_loc}.\exists \textsf{cont_in}.\textsf{Heart} & \sqsubseteq & \textsf{Heartdisease} \sqcap \textsf{NeedsTreatment}
\end{array}$$

> Interpretation就像每个人对一件事物（式子）的不同认知（例子）

- 不能有环

1、concept hierarchy

The **concept hierarchy** induced by a TBox $\mathcal{T}$ is defined as

$$\{A \sqsubseteq B \mid A, B \text{ concept names in } \mathcal{T} \text{ and } \mathcal{T} \text{ implies } A \sqsubseteq B\}$$

2、concept inclusion true in I

Let $\mathcal{I}$ be an interpretation, $C \sqsubseteq D$ a concept inclusion, and $\mathcal{T}$ a TBox.

- We write $\mathcal{I} \models C \sqsubseteq D$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. If this is the case, then we say that
  - $\mathcal{I}$ satisfies $C \sqsubseteq D$ or, equivalently,
  - $C \sqsubseteq D$ is true in $\mathcal{I}$ or, equivalently,
  - $\mathcal{I}$ is a model of $C \sqsubseteq D$.
- We write $\mathcal{I} \models C \equiv D$ if $C^{\mathcal{I}} = D^{\mathcal{I}}$
- We write $\mathcal{I} \models \mathcal{T}$ if $\mathcal{I} \models E \sqsubseteq F$ for all $E \sqsubseteq F$ in $\mathcal{T}$. If this is the case, then we say that
  - $\mathcal{I}$ satisfies $\mathcal{T}$ or, equivalently,
  - $\mathcal{I}$ is a model of $\mathcal{T}$.

3、concept inclusion follow from a TBox

Let $\mathcal{T}$ be a TBox and $C \sqsubseteq D$ a concept inclusion. We say that $C \sqsubseteq D$ **follows from** $\mathcal{T}$ if, and only if, every model of $\mathcal{T}$ is a model of $C \sqsubseteq D$.

Instead of saying that $C \sqsubseteq D$ follows from $\mathcal{T}$ we often write

- $\mathcal{T} \models C \sqsubseteq D$ or
- $C \sqsubseteq_{\mathcal{T}} D$.

三、预处理pre-processing

> polynomial time算法（tractable便宜解决性 ）

相当于把乱七八糟的EL concept变成全是concept name的个体（但还不是definition）

normal form转换（不再转为NNF）

> 都可以转换成这四个

- 不能有equal
- 右侧不能有conjuction
- r.后面只能是concept name
- 左侧的conjuction两边如果复杂就用X替换
- 三连conjuction，一层一层用X替换
- 两边都复杂，插一个X在中间

(sform) $A \sqsubseteq B$, where A and B are concept names;

(cform) $A_1 \sqcap A_2 \sqsubseteq B$, where A_1, A_2, B are concept names;

(rform) $A \sqsubseteq \exists r.B$, where A, B are concept names;

(lform) $\exists r.A \sqsubseteq B$, where A, B are concept names.

6个方法

> 1：《==》拆成两个
> 3：非原子的先用原子X代替，然后再加上X《==》非原子

- Replace each $C_1 \equiv C_2$ by $C_1 \sqsubseteq C_2$ and $C_2 \sqsubseteq C_1$;
- Replace each $C \sqsubseteq C_1 \sqcap C_2$ by $C \sqsubseteq C_1$ and $C \sqsubseteq C_2$;
- If $\exists r.C$ occurs in $\mathcal{T}$ and C is complex, replace C in $\mathcal{T}$ by a fresh concept name X and add $X \sqsubseteq C$ and $C \sqsubseteq X$ to $\mathcal{T}$;
- If $C \sqsubseteq D$ in $\mathcal{T}$ and $\exists r.B$ occurs in C (but $C \neq \exists r.B$), then remove $C \sqsubseteq D$, take a fresh concept name X, and add

$$X \sqsubseteq \exists r.B, \quad \exists r.B \sqsubseteq X, \quad C' \sqsubseteq D$$

to $\mathcal{T}$, where C' is the concept obtained from C by replacing $\exists r.B$ by X.

- If $A_1 \sqcap \cdots \sqcap A_n \sqsubseteq D$ in $\mathcal{T}$ and $n > 2$, then remove it, take a fresh concept name X, and add

$$A_2 \sqcap \cdots \sqcap A_n \sqsubseteq X, \quad X \sqsubseteq A_2 \sqcap \cdots \sqcap A_n, \quad A_1 \sqcap X \sqsubseteq D$$

to $\mathcal{T}$.

- If $\exists r.B \sqsubseteq \exists s.E$ in $\mathcal{T}$, then remove it, take a fresh concept name X, and add

$$\exists r.B \sqsubseteq X, \quad X \sqsubseteq \exists s.E$$

to $\mathcal{T}$.

预处理例题

> 本例只用了规则2，4

Pre-Processing: Example

Consider $\mathcal{T}$:

$$A_0 \sqsubseteq B \sqcap \exists r.B', \quad A_1 \sqcap \exists r.B \sqsubseteq A_2$$

Step 1 gives:

$$A_0 \sqsubseteq B, \quad A_0 \sqsubseteq \exists r.B', \quad A_1 \sqcap \exists r.B \sqsubseteq A_2$$

Step 4 gives:

$$\begin{aligned} A_0 &\sqsubseteq B \\ A_0 &\sqsubseteq \exists r.B' \\ A_1 \sqcap X &\sqsubseteq A_2 \\ \exists r.B &\sqsubseteq X \\ X &\sqsubseteq \exists r.B \end{aligned}$$

四、判断intuition的算法

1、S函数（针对concept）

找出与每一个conceptA满足subsumption关系的集合B

> 每个都是A的父类
>
> 每走一轮就添加一次，直到添完

2、R函数（针对role name）

算出所有pair：（B1，B2）满足r关系

> 辅助计算S用的

Given $\mathcal{T}$ in normal form, we compute functions S and R:

- S maps every concept name A from $\mathcal{T}$ to a set of concept names B;
- R maps every role name r from $\mathcal{T}$ to a set of pairs (B_1, B_2) of concept names.

We will have $A \sqsubseteq_{\mathcal{T}} B$ if, and only if, $B \in S(A)$.

3、初始化

- concept初始化为自身，role初始化为空集

> A永远是自己的父类
> 一开始不知道谁满足r关系

Input: $\mathcal{T}$ in normal form. Initialise: $S(A) = \{A\}$ and $R(r) = \emptyset$ for A and r in $\mathcal{T}$.

4、添加规则

- 把应有的关系加进去，直到exhaustively（闭包，饱和）

> 1：最简单的sform直接添加
> 2：cform，交集非空时，把B加到交集的S中
> 3：rform，唯一对R（r）升级
> 4：lform，利用R的升级再升级S

(simpleR) If $A' \in S(A)$ and $A' \sqsubseteq B \in \mathcal{T}$ and $B \notin S(A)$, then

$$S(A) := S(A) \cup \{B\}.$$

(conjR) If $A_1, A_2 \in S(A)$ and $A_1 \sqcap A_2 \sqsubseteq B \in \mathcal{T}$ and $B \notin S(A)$, then

$$S(A) := S(A) \cup \{B\}.$$

(rightR) If $A' \in S(A)$ and $A' \sqsubseteq \exists r.B \in \mathcal{T}$ and $(A, B) \notin R(r)$, then

$$R(r) := R(r) \cup \{(A, B)\}.$$

(leftR) If $(A, B) \in R(r)$ and $B' \in S(B)$ and $\exists r.B' \sqsubseteq A' \in \mathcal{T}$ and $A' \notin S(A)$, then

$$S(A) := S(A) \cup \{A'\}.$$

5、example

Example

$$A_0 \sqsubseteq \exists r.B$$

$$B \sqsubseteq E$$

$$\exists r.E \sqsubseteq A_1$$

Initialise: $S(A_0) = \{A_0\}$, $S(A_1) = \{A_1\}$, $S(B) = \{B\}$, $S(E) = \{E\}$, $R(r) = \emptyset$.

- Application of (rightR) and axiom 1 gives: $R(r) = \{(A_0, B)\}$;
- Application of (simpleR) and axiom 2 gives: $S(B) = \{B, E\}$;
- Application of (leftR) and axiom 3 gives: $S(A_0) = \{A_0, A_1\}$;
- No more rules are applicable.

Thus, $R(r) = \{(A_0, B)\}$, $S(B) = \{B, E\}$, $S(A_0) = \{A_0, A_1\}$ and no changes for the remaining values. We obtain $A_0 \sqsubseteq_{\mathcal{T}} A_1$.

> 1：rightR
> 2：simpleR
> 3：leftR

- 算法目的：得到闭包后可以用于查阅是否存在包含关系