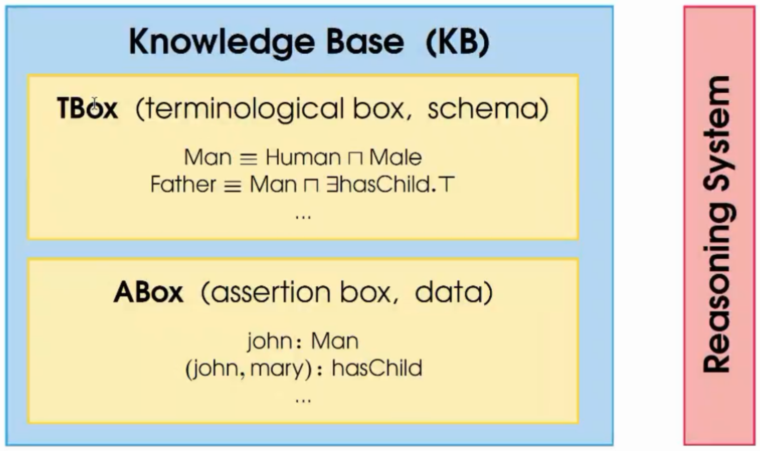
一、定义

- They are descendants of **semantic networks** and **KL-ONE** from the 1960-70s.
- They describe a domain of interest in terms of
  - **concepts** (also called classes),
  - **roles** (also called relations or properties),
  - **individuals**
- Modulo a simple translation, they are subsets of predicate logic.
- Distinction between terminology and data (see next slide).

二、架构



- KB：建立知识库

> **一个 TBox 和 ABox 共同组成一个 Knowledge Base，简称 KB**
>
> **一个本体就应该等同于一个 KB**

- Tbox**concept hierarchy**：建立概念骨架
- Abox：概念建立好后，把数据归类，insert
- Reasoning：建立后的推理，是演绎推理（非归纳推理）

ALC TBox

> possibly complex有可能是复杂概念
>
> **类层面的知识的集合，里面都是类与类之间的包含关系**
>
> 是有限的GCIs

- **TBox axioms 或者 TBox inclusions，即**GCI=general concept inclusion包含关系
- 等价关系可以转化为两个GCI
- GCI和等价关系都是axioms
- 能让GCI成立的interpretion才可以叫model

Definition (Semantics)

Let $\mathcal{I}$ be an interpretation. $\mathcal{I}$ satisfies a GCI $C \sqsubseteq D$ ($C \sqsubseteq D$ is true in $\mathcal{I}$), written $\mathcal{I} \models C \sqsubseteq D$, if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$.
In this case, $\mathcal{I}$ is called a model of $C \sqsubseteq D$.
$\mathcal{I}$ is a model of $\mathcal{T}$ iff it satisfies each GCI in $\mathcal{T}$.

> I是Tbox的model说明I满足其中所有GCI
>
> 判断I是否是model就看是否满足所有的包含关系

- 知识越多，model越少

ALC ABox

> **实体层面的知识的集合**

- **两种Assertion**

Definition (Syntax)

Let a and b be two individuals, C a possibly complex $\mathcal{ALC}$ concept, and r a role name:

- $a : C$ (concept assertion, sometimes written $C(a)$)
- $(a, b) : r$ (role assertion, sometime written $r(a, b)$)

Both are called Assertions.
An $\mathcal{ALC}$ ABox $\mathcal{A}$ is a finite set of concept and role assertions.

Definition (Semantics)

Let $\mathcal{I}$ be an interpretation. $\mathcal{I}$ satisfies a concept assertion $a : C$ ($a : C$ is true in $\mathcal{I}$), written $\mathcal{I} \models a : C$, if $a^{\mathcal{I}} \in C^{\mathcal{I}}$. $\mathcal{I}$ satisfies a role assertion $(a, b) : r$, written $\mathcal{I} \models (a, b) : r$, if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$.
In this case, $\mathcal{I}$ is called a model of the assertion.
$\mathcal{I}$ is a model of $\mathcal{A}$ iff it satisfies each assertion in $\mathcal{A}$.

> concept assertion声称为哪个类
>
> role assertion二元关系

- complete ABox：发现clash或 none of the expansion rules is applicable
- clash-free ABox：无clash

ALC RBox

除了 TBox 和 ABox，本体有时还会包含一个 Role Box（简称 RBox）。Role Box 顾名思义，是关于 role 的性质。比如说，role 之间也可以有 inclusion 关系：hasFather 是 hasAncester 的子关系。意味着，a 如果是 b 的父亲，a 就一定是 b 的祖先。role 之间的包含关系称为 role inclusion，用字母 H 表示。EL 语言中如果加入 role inclusion，则称为 ELH；同样地，ALC 称为 ALCH。除此之外，RBox 还可以加入 role equivalence，表示两个 role 等价（同样地，a role equivalence 可以表示成两个 role inclusions）。一些 role 拥有其它的性质，比如 role 的传递性（transitivity），对称性（symmetry），自反性（reflexivity），函数性（functional）等等。这里，传递性是我们在这门课上介绍的一个代表性性质，其字母表示规则很特殊：ALC + transitivity = S，ALCHI + transitivity = SHI。一个 role r 是 transitive 的，其语义为：对于 domain 任意元素 x, y, z 来说，$\forall x \forall y \forall z((r(x, y) \wedge r(y, z)) \rightarrow r(x, z))$。比如，hasAncester 就是一个 transitive role；但 hasFriend 和 hasParent 不是。

如果有 RBox，则 KB = TBox + ABox + RBox
如果没有，则 KB = TBox + ABox

三、Basic Reasoning Problems

1.satisfiable

能找到解释使C非空，domain非空

C is satisfiable with respect to $\mathcal{T}$ if there exists a model $\mathcal{I}$ of $\mathcal{T}$ and some $d \in \Delta^{\mathcal{I}}$ with $d \in C^{\mathcal{I}}$;

2.subsumed

如果T的每一model都让C包含于D成立

C is subsumed by D with respect to $\mathcal{T}$, written $\mathcal{T} \models C \sqsubseteq D$ or $C \sqsubseteq_{\mathcal{T}} D$, if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for every model of $\mathcal{T}$;

3.consistent

整个知识库K是否有model可满足，否则知识库无意义

$\mathcal{K}$ is consistent (satisfiable) if there exists a model of $\mathcal{K}$;

4.instance

个体属于集合

a is an instance of C with respect to $\mathcal{K}$, written $\mathcal{K} \models a : C$, if $a^{\mathcal{I}} \in C^{\mathcal{I}}$ for every model of $\mathcal{K}$.

5.转化

- 检测c是否包含于否D：检测互斥

- 都可以转化成（8）用Tableau

（1）验证两个 concepts 之间是否存在 inclusion（subsumption）关系（equivalence 关系验证可以转化成两个 inclusions 关系验证，下面涉及 equivalence 的地方也是一样）
（2）验证一个 individual 是否属于某个 concept（membership）
（3）验证两个 individual 之间是否存在某二元关系（membership）

（7）验证一个 ontology 是否 consistent（部分文献也叫 satisfiable）
（8）验证一个 concept 是否 satisfiable

无TBox推理

C包含于D是永真式

技巧：先拆解存在，再forall，因为forall是“如果有则一 定”

四、Tableau方法

> concept层面，不涉及包含关系
>
> ALC扩展的Tableau不涉及
>
> Ontology consistent 等价于 Concept satisfiable，用的都是tableau

一、算法

Algorithm consistent()
Input: a normalised $\mathcal{ALC}$ ABox $\mathcal{A}$
if expand($\mathcal{A}$) $\neq \emptyset$ **then**
  return "consistent"
else
  return "inconsistent"

Algorithm expand()
Input: a normalised $\mathcal{ALC}$ ABox $\mathcal{A}$
if $\mathcal{A}$ is not complete **then**
  select a rule R that is applicable to $\mathcal{A}$ and an assertion
  or pair of assertions α in $\mathcal{A}$ to which R is applicable
  if there is $\mathcal{A}' \in \mathsf{exp}(\mathcal{A}, R, \alpha)$ with expand($\mathcal{A}'$) $\neq \emptyset$ **then**
    return expand($\mathcal{A}'$)
  else
    return $\emptyset$
else
  if $\mathcal{A}$ contains a clash **then**
    return $\emptyset$
  else
    return $\mathcal{A}$

If algorithm returns “consistent”，*A* is a complete and clash-free ABox.

二、解题步骤

（要求用Tableau不能举例子）

1、检测NNF：not只能在单个原子concept前

Aim: starting from $S_0 = \{x: C\}$ apply completion rules to construct a clash-free system S_n to which no completion rule is applicable

- If this is possible, then we can extract a **model satisfying** C
- Otherwise, C is **not satisfiable**.

2、假设存在x属于S0满足

3、找根结点（主运算符）

4、按照添加规则逐步扩充S，直至饱和

> **当全部 branches 都检查完了且都找到了 clash，C 才是 unsatisfiable。**

1、标准化NNF方法：

$$
\begin{aligned}
\neg\top &\equiv \bot \\
\neg\bot &\equiv \top \\
\neg\neg C &\equiv C \\
\neg(C \sqcap D) &\equiv \neg C \sqcup \neg D \quad \text{(De Morgan's law)} \\
\neg(C \sqcup D) &\equiv \neg C \sqcap \neg D \quad \text{(De Morgan's law)} \\
\neg\forall r.C &\equiv \exists r.\neg C \\
\neg\exists r.C &\equiv \forall r.\neg C
\end{aligned}
$$

反证不satisfiable

证satisfiable给出一个例子就行

2、S：constraint system：constraint集合

Constraint: expression of the form $x: C$ or $(x, y): r$, where C is a concept in NNF and r a role name

Constraint system: a finite non-empty set S of constraints

Completion rules: $S \rightarrow S'$, where S' is a constraint system containing S

3、clash

S contains **clash** if

$$\boxed{\{x: A, \; x: \neg A\} \subseteq S, \quad \text{for some } x \text{ and concept name } A}$$

饱和：再加入知识，S也不变（此时无clash即satisfiable）

4、添加规则

$$S \rightarrow_{\sqcap} S \cup \{ x\colon C,\ x\colon D \}$$

if (a) $x\colon C \sqcap D$ is in S
(b) $x\colon C$ and $x\colon D$ are not both in S

$$S \rightarrow_{\sqcup} S \cup \{ x\colon E \}$$

if (a) $x\colon C \sqcup D$ is in S
(b) neither $x\colon C$ nor $x\colon D$ is in S
(c) $E = C$ or $E = D$ **(branching!)**

左右都要验证，发现一个满足即satisfiable

$$S \rightarrow_{\forall} S \cup \{ y\colon C \}$$

if (a) $x\colon \forall r.C$ is in S
(b) $(x, y)\colon r$ is in S
(c) $y\colon C$ is not in S

NB: Only applicable if role successors can be found

$$S \rightarrow_{\exists} S \cup \{ (x, y)\colon r,\ y\colon C \}$$

if (a) $x\colon \exists r.C$ is in S
(b) y is a fresh individual
(c) there is no z such that both $(x, z)\colon r$ and $z\colon C$ are in S

NB: The only rule that creates new individuals in a constraint system

Tableau Example 1

We check whether $(A \sqcap \neg A) \sqcup B$ is satisfiable.
It is in NNF, so we can directly apply the tableau algorithm to

$$S_0 = \{x : (A \sqcap \neg A) \sqcup B\}$$

The only rule applicable is $\rightarrow_{\sqcup}$. We have two possibilities.
Firstly we can try

$$S_1 = S_0 \cup \{x : A \sqcap \neg A\}.$$

Then we can apply $\rightarrow_{\sqcap}$ and obtain

$$S_2 = S_1 \cup \{x : A, x : \neg A\}$$

We have obtained a clash, thus this choice was unsuccessful.
Secondly, we can try

$$S_1^* = S_0 \cup \{x : B\}.$$

No rule is applicable to S_1^* and it does not contain a clash. Thus, $(A \sqcap \neg A) \sqcup B$ is satisfiable.

A model $\mathcal{I}$ satisfying it is given by

$$\Delta^{\mathcal{I}} = \{x\}, \quad B^{\mathcal{I}} = \{x\}, \quad A^{\mathcal{I}} = \emptyset.$$

四、Reasoning with Tbox

> Tbox是背景知识

原来不成立的，可能有了Tbox之后成立了缩小限制范围）

转化成全集的方式

$$\mathcal{I} \models C \sqsubseteq D \quad \text{iff} \quad \mathcal{I} \models \top \sqsubseteq \neg C \sqcup D$$

为了TBox，新增rule：universal

一旦有新元素就要拿进来

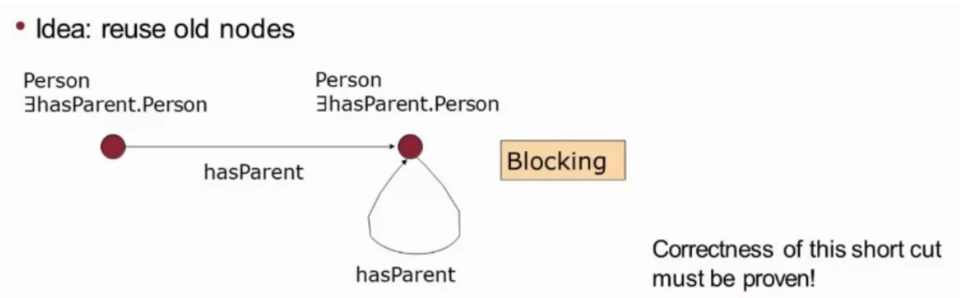
$$S \rightarrow_U S \cup \{ x : D \}$$

if (a) $\top \sqsubseteq D$ is in S
(b) x occurs in S
(c) $x : D$ is not in S

举例：可能导致不可终止---所以引入blocking

$$\begin{array}{ll}
S_0 & = \{ x_0 : \top, \quad \top \sqsubseteq \exists r.A \} \\
S_0 \rightarrow_U S_1 & = S_0 \cup \{ x_0 : \exists r.A \} \\
S_1 \rightarrow_\exists S_2 & = S_1 \cup \{ (x_0, x_1) : r, \quad x_1 : A \} \\
S_2 \rightarrow_U S_3 & = S_2 \cup \{ x_1 : \exists r.A \} \\
S_3 \rightarrow_\exists S_4 & = S_3 \cup \{ (x_1, x_2) : r, \quad x_2 : A \} \\
S_4 \rightarrow_U S_5 & = S_4 \cup \{ x_2 : \exists r.A \} \\
\quad \ldots & \quad \ldots
\end{array}$$

blocking



考试可能涉及有无Tbox时如何证明

抓取知识的方法：把“C包含于D”，变成全集满足”否C并D “

算法分析

Theorem 4.7. *The tableau algorithm presented in Definition 4.2 is a decision procedure for the consistency of $\mathcal{ALC}$ ABoxes.*

Proof. That the algorithm is a decision procedure for normalised $\mathcal{ALC}$ ABoxes follows from Lemmas 4.4, 4.5 and 4.6; and as we showed at the beginning of this subsection, an arbitrary $\mathcal{ALC}$ ABox can be transformed into an equivalent normalised ABox. □

分析可决定性的三要素：（适用于所有问题）

- 终止Termination
- 正确性soundness
- 完备性completeness

一、Termination

Lemma 4.4 (Termination). *For each $\mathcal{ALC}$ ABox $\mathcal{A}$,* $\mathsf{consistent}(\mathcal{A})$ *terminates.*

Extend the definition of subconcept to ABoxes and to knowledge bases:

$$\mathsf{sub}(\mathcal{A}) = \bigcup_{a:C\in\mathcal{A}} \mathsf{sub}(C)$$

and for $\mathcal{K} = (\mathcal{T}, \mathcal{A})$,

$$\mathsf{sub}(\mathcal{K}) = \mathsf{sub}(\mathcal{T}) \cup \mathsf{sub}(\mathcal{A}).$$

Set of concepts occurring in a concept assertion:

$$\mathsf{con}_{\mathcal{A}}(a) = \{C \mid a:C \in \mathcal{A}\}.$$

proof：封锁所有可扩展方式

1.运用规则增加的assertion有限

Proof. Let $m = |\mathsf{sub}(\mathcal{A})|$. Termination is a consequence of the following properties of the expansion rules:

(i) The expansion rules never remove an assertion from $\mathcal{A}$, and each rule application adds a new assertion of the form $a : C$, for some individual name a and some concept $C \in \mathsf{sub}(\mathcal{A})$. Moreover, we saw in Lemmas 3.11 and 4.3 that the size of $\mathsf{sub}(\mathcal{A})$ is bounded by the size of $\mathcal{A}$, and thus there can be at most m rule applications adding a concept assertion of the form $a : C$ for any individual name a, and $|\mathsf{con}_{\mathcal{A}}(a)| \leq m$.

2.新建的individual有限

(ii) A new individual name is added to $\mathcal{A}$ only when the $\exists$-rule is applied to an assertion of the form $a : C$ with C an existential restriction (a concept of the form $\exists r.D$), and for any individual name each such assertion can trigger the addition of at most one new individual name. As there can be no more than m different existential restrictions in $\mathcal{A}$, a given individual name can cause the addition of at most m new individual names, and the outdegree of each tree in the forest-shaped ABox is thus bounded by m.

（outdegree：往外延申r关系的）

3.单调递减

在2的基础上找到bound

(iii) The $\exists$- and $\forall$-rules are triggered by assertions of the form $a : \exists r.C$ and $a : \forall r.C$, respectively, and they only add concept assertions of the form $b : C$, where b is a successor of a; in either case, C is a strict subdescription of the concept $\exists r.C$ or $\forall r.C$ in the assertion to which the rule was applied, and it is clearly strictly smaller than these concepts. Further rule applications may be triggered by the presence of $b : C$ in $\mathcal{A}$, adding additional concept assertions $b : D$, but then D is a subdescription of C that is smaller than C, etc. Consequently, $\mathsf{sub}(\mathsf{con}_{\mathcal{A}}(b)) \subseteq \mathsf{sub}(\mathsf{con}_{\mathcal{A}}(a))$ and the size of the largest concept in $\mathsf{con}_{\mathcal{A}}(b)$ is smaller than the size of the largest concept in $\mathsf{con}_{\mathcal{A}}(a)$. The second fact shows that the inclusion stated by the first fact is actually strict; i.e., for any tree individual b whose predecessor is a, $\mathsf{sub}(\mathsf{con}_{\mathcal{A}}(b)) \subsetneq \mathsf{sub}(\mathsf{con}_{\mathcal{A}}(a))$. Consequently, the depth of each tree in the forest-shaped ABox is bounded by m.

二、soundness

Lemma 4.5 (Soundness). *If* $\mathsf{consistent}(\mathcal{A})$ *returns "consistent", then* $\mathcal{A}$ *is consistent.*

> **想证consistent，要找到&构造一个解释I是model，即符合所有A‘的assertions，因此也符合所有A的assertions**

proof

- 取expand后的Abox，构造其解释I

> con_A（a）:ABox A中所有包含a的assertion里面的concept集合

Proof. Let $\mathcal{A}'$ be the set returned by $\mathsf{expand}(\mathcal{A})$. Since the algorithm returns "consistent", $\mathcal{A}'$ is a complete and clash-free ABox.

The proof then follows rather easily from the very close correspondence between $\mathcal{A}'$ and an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ that is a model of $\mathcal{A}'$, i.e., that satisfies each assertion in $\mathcal{A}'$. Given that the expansion rules never delete assertions, we have that $\mathcal{A} \subseteq \mathcal{A}'$, so $\mathcal{I}$ is also a model of $\mathcal{A}$, and is a witness to the consistency of $\mathcal{A}$. We use $\mathcal{A}'$ to construct a suitable interpretation $\mathcal{I}$ as follows:

$$\begin{aligned}
\Delta^{\mathcal{I}} &= \{a \mid a : C \in \mathcal{A}'\}, \\
a^{\mathcal{I}} &= a \text{ for each individual name } a \text{ occurring in } \mathcal{A}', \\
A^{\mathcal{I}} &= \{a \mid A \in \mathsf{con}_{\mathcal{A}'}(a)\} \text{ for each concept name } A \text{ in } \mathsf{sub}(\mathcal{A}'), \\
r^{\mathcal{I}} &= \{(a,b) \mid (a,b) : r \in \mathcal{A}'\} \text{ for each role } r \text{ occurring in } \mathcal{A}'.
\end{aligned}$$

- 转化问题为验证：任意concept C满足下式

all role assertions in $\mathcal{A}'$. By induction on the structure of concepts, we show the following property (P1):

$$\text{if } a : C \in \mathcal{A}', \text{ then } a^{\mathcal{I}} \in C^{\mathcal{I}}. \qquad \text{(P1)}$$

Induction Basis C is a concept name: by definition of $\mathcal{I}$, if $a : C \in \mathcal{A}'$, then $a^{\mathcal{I}} \in C^{\mathcal{I}}$ as required.

- 归纳所有ALC符号（5证3）
- 具体步骤：

  1、转化assertion

  2、解释符号语义，简化assertion

  3、把assertion翻译到解释I的层面，如a^I不属于D^I，证明了解释满足要求

  4、再解释符号语义，变形成最终

Induction Steps

- $C = \neg D$: since $\mathcal{A}'$ is clash-free, $a : \neg D \in \mathcal{A}'$ implies that $a : D \notin \mathcal{A}'$. Since all concepts in $\mathcal{A}$ are in NNF, D is a concept name. By definition of $\mathcal{I}$, $a^{\mathcal{I}} \notin D^{\mathcal{I}}$, which implies $a^{\mathcal{I}} \in \Delta^{\mathcal{I}} \setminus D^{\mathcal{I}} = C^{\mathcal{I}}$ as required.
- $C = D \sqcup E$: if $a : D \sqcup E \in \mathcal{A}'$, then completeness of $\mathcal{A}'$ implies that $\{a : D, a : E\} \cap \mathcal{A}' \neq \emptyset$ (otherwise the $\sqcup$-rule would be applicable). Thus $a^{\mathcal{I}} \in D^{\mathcal{I}}$ or $a^{\mathcal{I}} \in E^{\mathcal{I}}$ by induction, and hence $a^{\mathcal{I}} \in D^{\mathcal{I}} \cup E^{\mathcal{I}} = (D \sqcup E)^{\mathcal{I}}$ by the semantics of $\sqcup$.
- $C = D \sqcap E$: this case is analogous to but easier than the previous one and is left to the reader as a useful exercise.
- $C = \forall r.D$: let $a : \forall r.D \in \mathcal{A}'$ and consider b with $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$. For $a^{\mathcal{I}}$ to be in $(\forall r.D)^{\mathcal{I}}$, we need to ensure that $b^{\mathcal{I}} \in D^{\mathcal{I}}$. By definition of $\mathcal{I}$, $(a, b) : r \in \mathcal{A}'$. Since $\mathcal{A}'$ is complete and $a : \forall r.D \in \mathcal{A}'$, we have that $b : D \in \mathcal{A}'$ (otherwise the $\forall$-rule would be applicable). By induction, $b^{\mathcal{I}} \in D^{\mathcal{I}}$, and since the above holds for all b with $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$, we have that $a^{\mathcal{I}} \in (\forall r.D)^{\mathcal{I}}$ by the semantics of $\forall$.
- $C = \exists r.D$: again, this case is analogous to and easier than the previous one and is left to the reader as a useful exercise.

As a consequence, $\mathcal{I}$ satisfies all concept assertions in $\mathcal{A}'$ and thus in $\mathcal{A}$, and it satisfies all role assertions in $\mathcal{A}'$ and thus in $\mathcal{A}$ by definition. Hence $\mathcal{A}$ has a model and thus is consistent. $\square$

三、completeness

Lemma 4.6 (Completeness). *If $\mathcal{A}$ is consistent, then* consistent($\mathcal{A}$) *returns "consistent".*

proof

- 首先若A是complete的则肯定返回consistent
- A非complete会递归调用expand自己直到complete，每次call都使用一个rule
- 下面证明rule的使用 preserves consistency

> 具体需要证明expand后I仍然是model

- The $\sqcup$-rule: If $a : C \sqcup D \in \mathcal{A}$, then $a^{\mathcal{I}} \in (C \sqcup D)^{\mathcal{I}}$ and Definition 2.2 implies that either $a^{\mathcal{I}} \in C^{\mathcal{I}}$ or $a^{\mathcal{I}} \in D^{\mathcal{I}}$. Therefore, at least one of the ABoxes $\mathcal{A}' \in \mathsf{exp}(\mathcal{A}, \sqcup\text{-rule}, a : C \sqcup D)$ is consistent. Thus, one of the calls of expand is applied to a consistent ABox.
- The $\sqcap$-rule: If $a : C \sqcap D \in \mathcal{A}$, then $a^{\mathcal{I}} \in (C \sqcap D)^{\mathcal{I}}$ and Definition 2.2 implies that both $a^{\mathcal{I}} \in C^{\mathcal{I}}$ and $a^{\mathcal{I}} \in D^{\mathcal{I}}$. Therefore, $\mathcal{I}$ is still a model of $\mathcal{A} \cup \{a : C, a : D\}$, so $\mathcal{A}$ is still consistent after the rule is applied.
- The $\exists$-rule: If $a : \exists r.C \in \mathcal{A}$, then $a^{\mathcal{I}} \in (\exists r.C)^{\mathcal{I}}$ and Definition 2.2 implies that there is some $x \in \Delta^{\mathcal{I}}$ such that $(a^{\mathcal{I}}, x) \in r^{\mathcal{I}}$ and $x \in C^{\mathcal{I}}$. Therefore, there is a model $\mathcal{I}'$ of $\mathcal{A}$ such that, for some new individual name d, $d^{\mathcal{I}'} = x$, and that is otherwise identical to $\mathcal{I}$. This model $\mathcal{I}'$ is still a model of $\mathcal{A} \cup \{(a, d) : r, d : C\}$, so $\mathcal{A}$ is still consistent after the rule is applied.
- The $\forall$-rule: If $\{a : \forall r.C, (a, b) : r\} \subseteq \mathcal{A}$, then $a^{\mathcal{I}} \in (\forall r.C)^{\mathcal{I}}$, $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$, and Definition 2.2 implies that $b^{\mathcal{I}} \in C^{\mathcal{I}}$. Therefore, $\mathcal{I}$ is still a model of $\mathcal{A} \cup \{b : C\}$, so $\mathcal{A}$ is still consistent after the rule is applied.