СПбГЭТУ «ЛЭТИ»
Кафедра Вычислительной техники

Магистерская программа
«Семантические технологии и многоагентные системы»

Дисциплина «*Семантический Web*»

Лекция 5
Дескриптивные логики

Содержание лекции

- Общая характеристика дескриптивных логик
- Архитектура системы ПЗ на основе ДЛ
- Простейшая ДЛ $\mathcal{AL}$: синтаксис и семантика
- Расширения ДЛ, конструкторы
- ДЛ $\mathcal{ALC}$ и $\mathcal{SHIQ}$
- Виды логического *вывода* над знаниями *о концептах*
- Виды логического *вывода* над знаниями *об индивидуумах*

Дескриптивные логики: общая характеристика

- Дескриптивные логики (ДЛ):
  - *семейство формальных языков* представления знаний, имеющих *точно определенную теоретико-модельную семантику*;
  - *явпяются формальной основой построения онтологий в Семантическом Web*;
  - происходят от *структурированных сетей с наследованием* (Structured Inheritance Networks)
    - *SIN* предложены с целью *придания формальной декларативной семантики* ранним моделям семантических сетей и фреймов и обеспечения возможности *создания эффективных инструментов реализации рассуждений*

Дескриптивные логики: общая характеристика

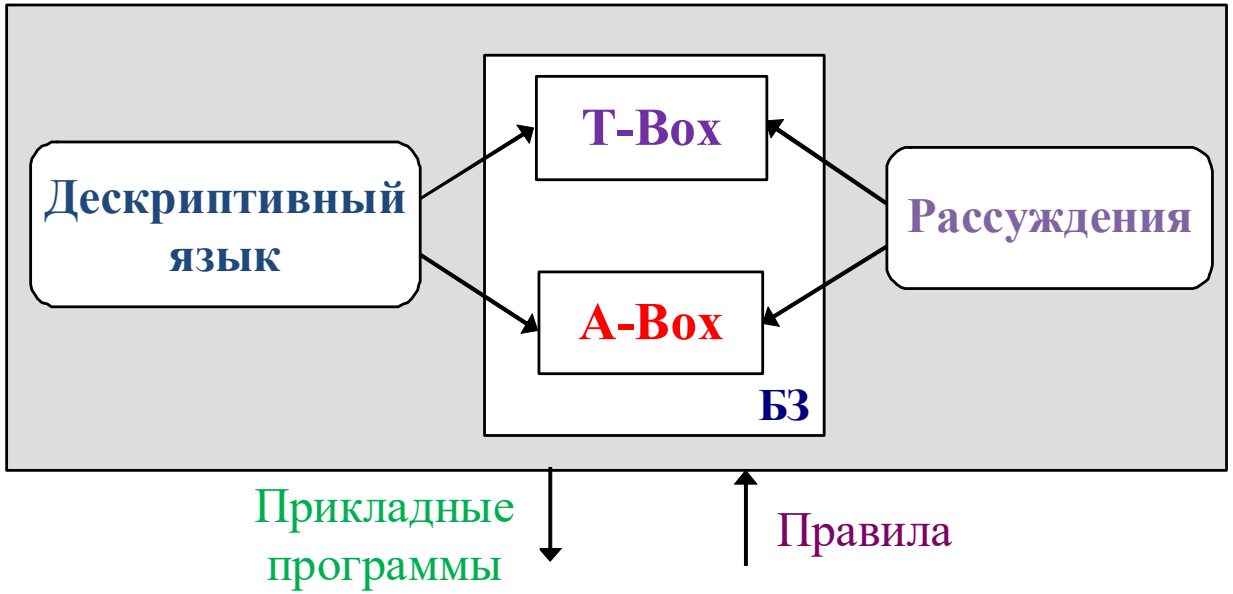
- В основе ДЛ лежат ***три идеи***, выдвинутые в работах по структурированным сетям с наследованием:

  1. *Базовыми синтаксическими единицами* (строительными блоками) являются:
     - *атомарные концепты* (унарные предикаты),
     - *атомарные роли* (бинарные предикаты) ,
     - *индивидуумы* (константы);

  2. *Более сложные концепты и роли* могут строиться с использованием (достаточно малого) *множества конструкторов, определяющих выразительную мощность языка*

  3. С помощью процедур логического вывода *автоматически могут быть выведены неявные знания* о концептах и индивидуумах

Архитектура системы ПЗ на основе ДЛ



- База знаний системы ДЛ состоит из двух компонент:
  - ***TBox*** (*Terminology Box*) – *вводит терминологию* (словарь) предметной области, содержащий ***классы*** и ***свойства***;
  - ***ABox*** (*Assertions Box*) – содержит *утверждения о конкретных индивидуумах* в терминах этого словаря;

Дескриптивные логики

- Системы, построенные на базе ДЛ, служат не только для хранения терминологии и утверждений, но также *предоставляют сервис логического вывода* на их основе
- Словарь состоит из
  - *понятий* (*концептов*) – задают *группы (множества) индивидуумов*
  - *ролей* – задают *бинарные отношения между индивидуумами*
  - *индивидуумов*
- *Атомарные концепты* и *атомарные роли* являются *элементарными описаниями*
- *Сложные описания* строятся из них с помощью *конструкторов концептов* (*операторов*)
- Разные дескриптивные языки *отличаются набором конструкторов* (операторов)

Дескриптивная логика $\mathcal{AL}$

- **Язык** $\mathcal{AL}$ (*Attributive Language* – атрибутивный язык):
- является *минимальным практически полезным языком*
- существует целое *семейство языков*, являющихся расширением $\mathcal{AL}$
- Далее используются обозначения:
  - *A* и *B* – *атомарные концепты* (понятия, соответствующие классам);
  - *R* – *атомарная роль*;
  - *C* и *D* – *описания концептов* (concept descriptions)

Синтаксис ДЛ $\mathcal{AL}$

- Правила построения концептов в языке $\mathcal{AL}$:

$C, D \rightarrow A$	|	// *атомарный концепт*
$\top$	|	// *концепт-универсум*
$\bot$	|	// *пустой концепт (bottom concept)*
$\neg A$	|	// ***атомарное*** (**ТОЛЬКО**!) *отрицание*
$C \sqcap D$	|	// *пересечение*
$\forall R.C$	|	// *ограничение значения (value restriction)*
$\exists R.\top$	|	// *ограниченная экзистенциальная квантификация*

- Особенности языка $\mathcal{AL}$:
  - *отрицание* может применяться ***только*** *к атомарным концептам*
  - среди конструкторов *отсутствует* ***объединение***;
  - *в области действия экзистенциальной квантификации* над ролью допускается *только концепт-универсум*

Подъязыки ДЛ $\mathcal{AL}$

- Подъязык, получаемый из $\mathcal{AL}$ *исключением атомарного отрицания*, назван $\mathcal{FL}$– (по историческим причинам)

- Подъязык $\mathcal{FL}$–, получаемый из $\mathcal{AL}$ *запрещением ограниченной экзистенциальной квантификации*, назван $\mathcal{FL}0$.

Примеры определений в ДЛ $\mathcal{AL}$

- Пусть заданы *атомарные концепты*:

  Person (Человек) и

  Female (Существо женского пола)

Тогда:

- **Person** $\sqcap$ **Female** – $\mathcal{AL}$-концепт, описывающий *женщин*
- **Person** $\sqcap$ $\neg$**Female** – $\mathcal{AL}$-концепт, описывающий людей, *не являющихся женщинами*

- Пусть задана *атомарная роль* **hasChild**. Тогда можно задать концепты:
  - **Person** $\sqcap$ $\exists$**hasChild**.$\top$ – описывает людей, *имеющих ребенка*;
  - **Person** $\sqcap$ $\forall$**hasChild**.$\bot$ – описывает людей, *не имеющих детей*;
  - **Person** $\sqcap$ $\forall$**hasChild.Female** – описывает людей, у которых все дети женского пола.

Формальная семантика $\mathcal{AL}$-концептов

- ДЛ имеют *теоретико-модельную семантику*
- Формальная семантика $\mathcal{AL}$ -концептов определяется через *интерпретации* $\mathcal{I}$
- *Интерпретация* $\mathcal{I} = (\Delta^{\mathcal{I}}, \bullet^{\mathcal{I}})$, где :
  - $\Delta^{\mathcal{I}}$ – *область интерпретации* (*непустое множество*),
  - $\bullet^{\mathcal{I}}$ – *функция интерпретации*, ставящая в соответствие:
    - каждому *атомарному концепту A* – множество $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
    - каждой *атомарной роли* – бинарное отношение $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
    - каждому *индивидууму i* – элемент из $\Delta^{\mathcal{I}}$
- Функция интерпретации *расширяется на описания концептов* с помощью правил

Интерпретация концептов в ДЛ

- *Функция интерпретации* расширяется на описания концептов с помощью следующих индуктивных *правил*:

$$\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$$

$$\bot^{\mathcal{I}} = \varnothing$$

$$(\neg A)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus A^{\mathcal{I}}$$

$$(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$$

$$(\forall R.C)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \forall\, b.\, (a, b) \in R^{\mathcal{I}} \rightarrow b \in C^{\mathcal{I}}\}$$

$$(\exists R.\top)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \exists\, b.\, (a, b) \in R^{\mathcal{I}}\}$$

- *Концепты C и D эквивалентны*, если для всех интерпретаций $\mathcal{I}$ справедливо $C^{\mathcal{I}} = D^{\mathcal{I}}$. Запись эквивалентности: $C \equiv D$

Пример:

$\forall$hasChild.Female $\sqcap$ $\forall$hasChild.Student $\equiv$ $\forall$hasChild.(Female $\sqcap$ Student)

Семейство языков $\mathcal{AL}$

- Языки *с б**о**льшими выразительными возможностями* получают *добавляя* к языку $\mathcal{AL}$ *новые конструкторы* (операторы)

- *Объединение* концептов записывается как $C \sqcup D$ (обозначается буквой $\mathcal{U}$) и интерпретируется так

$$(C \sqcup D)^I = C^I \cup D^I$$

- *Полная экзистенциальная квантификация* (обозначается буквой $\mathcal{E}$) записывается как $\exists R.C$ и интерпретируется так

$$(\exists R.C)^I = \{a \in \Delta^I \mid \exists b.(a, b) \in R^I \,\&\, b \in C^I\}$$

ДЛ: *числовые ограничения*

- *Числовые ограничения* (обозначаются буквой $\mathcal{N}$) записываются:

  $\geq nR$ (ограничение «*как минимум*»)

  $\leq nR$ (ограничение «*как максимум*»),

где n – неотрицательное целое число.

- Интерпретация ограничений:

$$(\geq nR)^I = \{a \in \Delta^I \mid |\{ b \mid (a, b) \in R^I \}| \geq n \}$$

  и

$$(\leq nR)^I = \{a \in \Delta^I \mid |\{b \mid (a, b) \in R^I \}| \leq n \},$$

  где $| \ldots |$ - мощность множества.

- *Отрицание* концепта (обозначается буквой $\mathcal{C}$) записывается как $\neg C$ и интерпретируется как

$$(\neg C)^I = \Delta^I \setminus C^I$$

Дескриптивные логики: Пример

- *Дополнительные конструкторы* (операторы) позволяют строить более сложные концепты
- Например,

$$\texttt{Person} \sqcap (\leq 1\ \texttt{hasChild} \sqcup (\geq 3\ \texttt{hasChild} \sqcap \exists\ \texttt{hasChild.Female}))$$

- Описывает *людей*, имеющих
  или *не более одного ребенка*
  или *не менее трех детей*, среди которых *есть девочки*

Дескриптивные логики: условные обозначения

- Все языки семейства $\mathcal{AL}$ получаются из языка $\mathcal{AL}$ путем его расширения определенным подмножеством вышеперечисленных конструкторов (операторов)
- Каждый язык $\mathcal{AL}$-семейства имеет название в форме

  $$\mathcal{AL}[\mathcal{U}][\mathcal{E}][\mathcal{N}][\mathcal{C}],$$

  где наличие определенной буквы в названии обозначает наличие соответствующего конструктора (оператора)
- Например, язык $\mathcal{ALEN}$ является расширением языка $\mathcal{AL}$, за счет добавления в него *полной экзистенциальной квантификации* ($\mathcal{E}$) и *числовых ограничений* ($\mathcal{N}$)
- ***Не все** эти **языки** **различны** **с семантической точки зрения***

Дескриптивные логики: *эквивалентность языков*

- Семантика обуславливает следующие эквивалентности:

$$C \sqcup D \equiv \neg (\neg C \sqcap \neg D)$$

$$\exists R.C \equiv \neg \forall R.\neg C.$$

- Таким образом, *объединение и полная экзистенциальная квантификация могут быть выражены с помощью отрицания*
  - поэтому в именах языков вместо букв $\mathcal{U}\mathcal{E}$ принято записывать $\mathcal{C}$
  - например, вместо $\mathcal{ALUE}$ писать $\mathcal{ALC}$, вместо $\mathcal{ALUEN}$ – $\mathcal{ALCN}$

- *И наоборот, комбинация объединения и полной экзистенциальной квантификации дает возможность выражать отрицание концептов*

- Следовательно, *все $\mathcal{AL}$-языки могут быть записаны с помощью букв $\mathcal{U}$, $\mathcal{E}$, $\mathcal{N}$*

- Нетрудно убедиться, что полученные таким образом *восемь языков попарно неэквивалентны*

ДЛ $\mathcal{ALC}$

- Правила построения концептов в языке $\mathcal{ALC}$:

$C, D \rightarrow A$ | // *атомарный концепт*

$\top$ | // *концепт-универсум*

$\bot$ | // *пустой концепт* (*bottom concept*)

$\neg C$ | // *отрицание,* ***не только атомарное!***

$C \sqcap D$ | // *пересечение*)

$C \sqcup D$ | // ***объединение***

$\forall R.C$ | // *ограничение значения* (*value restriction*)

$\exists R.C$ | // ***полная экзистенциальная квантификация***

Семейство языков ДЛ: язык $\mathcal{SHIQ}$

- $\mathcal{ALC}$ с *транзитивными ролями* (R_+) часто обозначается символом $\mathcal{S}$
- Другие расширения указываются *дополнительными символами*:
  - $\mathcal{H}$ – *иерархия ролей* (например, *hasDaughter* и *hasChild*);
  - $\mathcal{O}$ – *номиналы* – экстенсионально определенные понятия (oneOf)
    - EU= {France, Italy, …};
  - $\mathcal{I}$ – инверсные роли (например, *isChildOf* ´ *hasChild*$^-$);
  - $\mathcal{N}$ – числовые ограничения (например, ≥ 2*hasChild*, ≤ 3*hasChild*);
  - $\mathcal{Q}$ – *квалифицированные* числовые ограничения (например, ≥ 2*hasChild.Doctor*);
  - $\mathcal{F}$ – функциональность ролей – концепты вида (≤ 1 *R*), означающие: существует не более одного *R*–последователя, например, ≤ 1 *hasMother*)
  - (*D*) – расширение языка *конкретными типами данных* (доменами)
- $\mathcal{ALC} + R_+ + \mathcal{H} + \mathcal{I} + \mathcal{Q} = \mathcal{SHIQ}$
- $\mathcal{SHIQ}$ – основа OWL
  - OWL DL 1.1 – $\mathcal{SHOIQ}$(*D*) *расширение номиналами*
  - OWL Lite 1.1 – $\mathcal{SHIF}$(*D*) функциональность ролей

Пример описания класса на языках ДЛ и *OWL*

- *«Множество людей, все дети которых являются врачами или имеют детей, являющихся врачами»*

$$Person \sqcap \forall hasChild.(Doctor \sqcup \exists hasChild.Doctor)$$

```
<owl:Class>
  <owl:intersectionOf   rdf:parseType=" collection">     // класс является пересечением
    <owl:Class rdf:about="#Person"/>                     // класса «Люди»
    <owl:Restriction>                                    // и класса заданного ограничением
      <owl:onProperty rdf:resource="#hasChild"/>         // на отношение  «Имеет ребенка»,
      <owl:toClass>                                      // значением которого является
        <owl:unionOf rdf:parseType=" collection">        // объединение классов
          <owl:Class rdf:about="#Doctor"/>               // «Врач»
          <owl:Restriction>                              // и класса, заданного ограничением
            <owl:onProperty rdf:resource="#hasChild"/>   // на отношение «Имеет ребенка»
            <owl:hasClass rdf:resource="#Doctor"/>       // значением которого является «Врач»
          </owl:Restriction>
        </owl:unionOf>
      </owl:toClass>
    </owl:Restriction>
  </owl:intersectionOf>
</owl:Class>
```

Конструкторы *концептов* (классов)

Конструктор OWL	Синтаксис ДЛ	Пример	Синтаксис ЛППП
intersectionOf	$C_1 \sqcap \ldots \sqcap C_n$	$Human \sqcap Male$	$C_1(x) \wedge \ldots \wedge C_n(x)$
unionOf	$C_1 \sqcup \ldots \sqcup C_n$	$Doctor \sqcup Lawyer$	$C_1(x) \vee \ldots \vee C_n(x)$
complementOf	$\neg C$	$\neg Male$	$\neg C(x)$
oneOf	$\{o_1, \ldots, o_n\}$	$\{nick, mary\}$	$x = o_1 \vee \ldots \vee o_n$
allValuesFrom	$\forall R.C$	$\forall hasChild.Doctor$	$\forall y.R(x, y) \rightarrow C(y)$
someValuesFrom	$\exists R.C$	$\exists hasChild.Lawyer$	$\exists y.R(x, y) \wedge C(y)$
minCardinality	$\geq nR$	$\geq 3 hasChild$	$\exists^{\geq n} y.R(x, y)$
maxCardinality	$\leq nR$	$\leq 1 hasChild$	$\exists^{\leq n} y.R(x, y)$

- В $\forall R.C$ и $\exists R.C$ наряду с классами могут использоваться ***типы данных XMLS***. Например:
  - $\exists hasAge.nonNegativeInteger$
- *Произвольная вложенность* конструкторов:
  - $Person \sqcap \forall hasChild.(Doctor \sqcup \exists hasChild.Doctor)$

Конструкторы концептов: Пример

- *На языке ЛППП:*

  $\{x \mid Student(x)\} = \{x \mid Person(x) \;\&$
  $(\exists y.\mathrm{NAME}(x, y) \;\&\; \mathrm{String}(y)) \;\&$
  $(\exists z.\mathrm{ADDRESS}(x, z) \;\&\; \mathrm{String}(z)) \;\&$
  $(\exists w.\mathrm{ENROLLED}(x, w) \;\&\; \mathrm{Course}(w))\}$

- *На языке ДЛ*:

  $Student = Person \sqcap$
  $\exists \mathrm{NAME}.\mathrm{String} \sqcap$
  $\exists \mathrm{ADDRESS}.\mathrm{String} \sqcap$
  $\exists \mathrm{ENROLLED}.\mathrm{Course}$

Более компактный синтаксис !

Аксиомы онтологии в TBox

Синтаксис OWL	Синтаксис ДЛ	Пример
subClassOf	$C_1 \sqsubseteq C_2$	$Human \sqsubseteq Animal \sqcap Biped$
equivalentClass	$C_1 \equiv C_2$	$Man \equiv Human \sqcap Male$
subPropertyOf	$R_1 \sqsubseteq R_2$	$hasDaughter \sqsubseteq hasChild$
equivalentProperty	$R_1 \equiv R_2$	$cost \equiv price$
transitiveProperty	$R^+ \sqsubseteq R$	$ancestor^+ \sqsubseteq ancestor$

- Запись в ДЛ, ЛППП и модальной логики:
  - ДЛ: $C \sqsubseteq D$ ЛППП : $\forall x.C(x) \rightarrow D(x)$ МЛ: $C \rightarrow D$
- Часто различают два вида аксиом Tbox:
  - “Определения” $C \equiv D$ или $C \sqsubseteq D$ где C – имя концепта
  - Общие аксиомы включения концептов, где C может быть сложным

Аксиомы онтологии в ABox (факты)

Синтаксис OWL	Синтаксис ДЛ	Пример
rdf:type	$a : C$	$Nick : Happy\text{-}Father$
rdf:property	$\langle a, b \rangle : R$	$\langle Nick, Mary \rangle : hasChild$

- Аксиомы *ABox* могут быть сведены к ***аксиомам включения концептов*** с использованием *номиналов* (в $\mathcal{SHOIN}$):
  - $a : C$ эквивалентно $\{a\} \sqsubseteq C$
  - $\langle a, b \rangle : R$ эквивалентно $\{a\} \sqsubseteq \exists R.\{b\}$

Логический вывод в ДЛ

Семантика ДЛ: Пример

- $\mathcal{I} = (\Delta^{\mathcal{I}}, \bullet^{\mathcal{I}})$:
  - $\Delta^{\mathcal{I}} = \{Pete\_Ivanov, DL\_Reasoning\}$
  - $People^{\mathcal{I}} = Student^{\mathcal{I}} = \{Pete\_Ivanov\}$
  - $Topic^{\mathcal{I}} = \{DL\_Reasoning\}$
  - $Present^{\mathcal{I}} = \{(Pete\_Ivanov, DL\_Reasoning)\}$

- Интерпретацию удовлетворяющую всем аксиомам ДЛ также называют *моделью* онтологии

ДЛ: Сервисы рассуждения

- Система ДЛ не только хранит терминологию и утверждения, но также предлагает *сервисы для рассуждения* о них
- Над *концептуальными знаниями*, представленными в *TBox*, можно выполнять *четыре типа* полезных *логических выводов*:
  - *Подчиненность* (*Subsumption*);
  - *Совместимость* (*Satisfiability*);
  - *Эквивалентность* (*Equivalence*);
  - *Раздельность* (*Disjointness*)

Вывод в ДЛ над *TBox* : *Подчиненность*

- *Подчиненность* (*Subsumption*) – базовый вид вывода над понятиями
  - факт, что понятие *D* в *TBox* является *более общим*, чем понятие *C*, записывается так
    - $TBox \vdash C \sqsubseteq D$ (множество *C* является подмножеством *D)*
  - при выходе *за границы простой иерархии* понятий, например, когда надо рассмотреть множество отношений и ограничений кардинальности, **определение подчиненности становится *нетривиальным***
- Основная *цель подчиненности* – дать возможность *выводить отношения* между индивидами, *неочевидные из их определений в БЗ*

Вывод неявной иерархии

- Пример *TBox*:

  $Woman \equiv Person \sqcap Female$

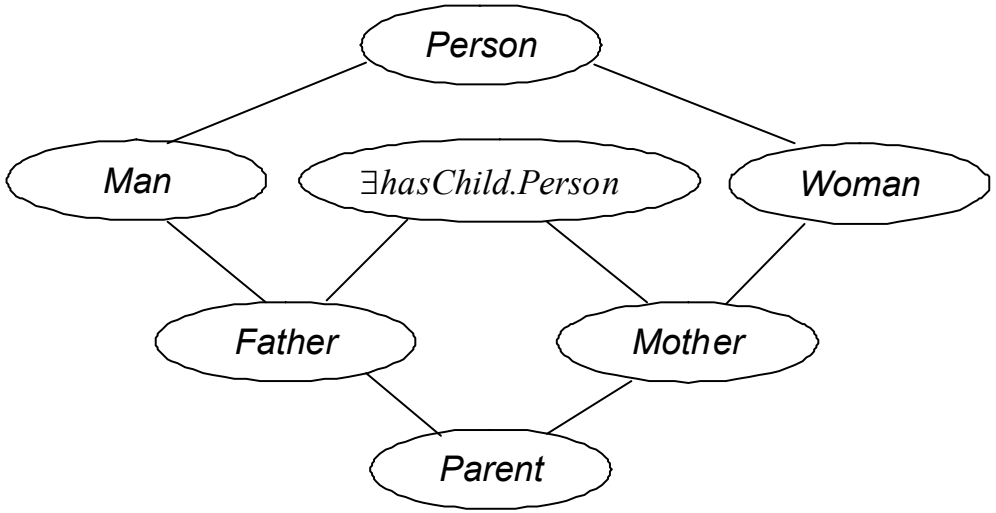
  $Man \equiv Person \sqcap \neg Woman$

  $Mother \equiv Woman \sqcap \exists hasChild.Person$

  $Father \equiv Man \sqcap \exists hasChild.Person$

  $Parent \equiv Mother \sqcup Father$

  Влечет следующую иерархию подчиненности (вывод неявных знаний):



Вывод в ДЛ над *TBox* : *Совместимость*

- При определении нового понятия в БЗ оно должно быть *совместимым*, т. е. *не противоречить* никаким *уже определенным понятиям*
- Понятие является совместимым, если *можно построить хотя бы одного индивидуума*, *удовлетворяющего данному понятию*
- В этом случае говорят, что понятие *выполнимо* (*satisfiable*), в противном случае – *невыполнимо* (*unsatisfiable*).
- *Невыполнимость* выразима через подчиненность следующим образом: $TBox \vdash C \sqsubseteq \bot$.

Вывод в ДЛ над *TBox*: *Эквивалентность* и *Раздельность*

- Проверка ***эквивалентности*** двух понятий полезна для устранения избыточности и неоднозначности БЗ.
  - Если *C* и *D* ***эквивалентны*** в TBox, это записывается так $TBox \vdash C \equiv D$
  - Эквивалентность также может быть *выражена в терминах подчиненности*: $TBox \vdash (C \sqsubseteq D \;\&\; D \sqsubseteq C)$

- Проверка *раздельности* понятий.
  - Понятия *раздельны* (*disjoint*), если между ними нельзя *вывести никаких отношений*
  - Раздельность также можно выразить через подчиненность:
    $TBox \vdash C \sqcap D \sqsubseteq \bot$.

Вывод в ДЛ над *ABox*: *типы вывода*

- Существует *четыре вида* важных *логических выводов*, которые могут быть выполнены *над знаниями о конкретных индивидуумах*, представленными в *ABox*:
  - *Проверка экземпляра* (*instance checking*);
  - *Поиск* (*retrieval*);
  - *Реализация* (*realization*);
  - *Проверка согласованности* (*consistency*)

Вывод в ДЛ над *ABox*: *Проверка экземпляра*

- *Проверка экземпляра* (*instance checking*) – основной вид логического вывода над знаниями в ABox,
  - *относится ли некоторый индивидуум* (экземпляр) *к конкретному понятию*;
  - Факт, что экземпляр *a* в *Abox* относится к понятию *C* записывается следующим образом:

    $ABox \vdash C(a)$.

- *Проверка экземпляра* важна, поскольку указывает, что данный экземпляр *классифицирован* в соответствии с терминологией данной БЗ

Вывод в ДЛ над *ABox*: *Поиск*

- *Поиск* (*retrieval*) – полезный на практике вид логического вывода
  - находит в БЗ *все экземпляры, относящиеся к заданному понятию*
- *Поиск* можно определить в терминах проверки экземпляра:

  $\{a \in ABox \mid ABox \vdash C(a)\}$.
- *Поиск* по существу является типом *запроса* к БЗ
- Дескриптивный язык, используемый для специфицирования представления знаний, в данном случае используется *непосредственно для определения образца запроса*

Вывод в ДЛ над *ABox*: *Реализация*

- *Реализация* (*realization*) – двойственный по отношению к нахождению вывод:
  - Задано: *индивидуум* ***a*** и *множество концептов*
  - Найти: *наиболее конкретные концепты C* из множества, такие что *ABox* $\vdash$ *C*(***a***)
  - *Наиболее конкретные концепты* – являются минимальными по отношению к упорядочению по подчиненности $\sqsubseteq$
- *Реализация* может, например, использоваться в естественно-языковых системах, если термины индексированы концептами и если для объекта дискурса надо *найти как можно более точный термин*

Вывод в ДЛ над *ABox*: *Проверка согласованности*

- Проверка *согласованности* (*consistency*) для *ABox* аналогична проверке выполнимости *TBox*
  - каждое понятие в БЗ допускает *по крайней мере одного индивидуума* из *ABox*
  - *Согласованность* можно определить в терминах проверки экземпляра:

    $\forall C \in TBox.\ \exists\, a \in ABox \mid ABox \vdash C(a).$

Вывод в ДЛ над *ABox*: сведение видов вывода

- С прагматической точки зрения полезны *все восемь* определенных выше *видов логического вывода*, т. к. они решают различные задачи рассуждений
- Однако, при построении машин логического вывода (модулей рассуждений) по соображениям простоты реализации *нецелесообразно реализовывать все виды вывода*
- В идеале желательно *реализовать только один вид вывода*, а остальные получить как специальные случаи
- Как было показано выше, *все выводы в TBox могут быть сведены к подчиненности*, а *все выводы в ABox – к проверке экземпляра*
- В действительности, *все выводы могут быть сведены к согласованности* для ABox