



Курс лекций по дисциплине Математические основы интеллектуальных систем

Шункевич Д.В.

Белорусский государственный университет информатики и радиоэлектроники



Лекция 1

Основные положения семантической технологии проектирования интеллектуальных компьютерных систем нового поколения

Шункевич Д.В.

Белорусский государственный университет информатики и радиоэлектроники



Понятие интеллектуальной компьютерной системы нового поколения



Кибернетическая система

кибернетическая система

- := [материальная сущность, способная целенаправленно воздействовать на среду своего обитания как минимум для сохранения своей целостности, жизнеспособности, безопасности]
- := [система, организация функционирования которой основано на обработке информации о той среде, в которой существует эта система]
- := [материальная сущность, способная к активной целенаправленной деятельности, которая на определенном уровне развития указанной сущности становится "осмысленной", планируемой, преднамеренной деятельностью]
- := [субъект, способный на самостоятельное выполнение некоторых "внутренних" и "внешних" действий либо порученных извне, либо инициированных самим субъектом]
- := [естественная или искусственно созданная система, способная мониторить и анализировать свое состояние и состояние окружающей среды, а также способная достаточно активно воздействовать на собственное состояние и на состояние окружающей среды]
- := [система, способная в достаточной степени самостоятельно взаимодействовать со своей средой, решая различные задачи]
- := [система, основанная на обработке информации]



Типология кибернетических систем

кибернетическая система

⇒ разбиение*:

Признак естественности или искусственности кибернетических систем

- = { • *естественная кибернетическая система*
:= [кибернетическая система естественного происхождения]
⊃ *человек*
- *компьютерная система*
:= [искусственная кибернетическая система]
:= [кибернетическая система искусственного происхождения]
:= [технически реализованная кибернетическая система]
- *симбиоз естественных и искусственных кибернетических систем*
:= [кибернетическая система, в состав которой входят компоненты как естественного, так и искусственного происхождения]
⊃ *сообщество компьютерных систем и людей*
- }

⊃ *интеллектуальная система*

Интеллектуальная компьютерная система



интеллектуальная компьютерная система

:= [интеллектуальная искусственная кибернетическая система]

\Rightarrow разбиение*:

- *индивидуальная интеллектуальная компьютерная система*
- *интеллектуальный коллектив интеллектуальных компьютерных систем*

:= [интеллектуальная многоагентная система, агенты которой являются интеллектуальными компьютерными системами]

\Rightarrow примечание*:

[Не каждый коллектив интеллектуальных компьютерных систем может оказаться интеллектуальным, поскольку уровень интеллекта такого коллектива определяется не только уровнем интеллекта его членов, но также и эффективностью (качеством) их взаимодействия.]

\Rightarrow разбиение*:

- *интеллектуальный коллектив индивидуальных интеллектуальных компьютерных систем*
- *иерархический интеллектуальный коллектив интеллектуальных компьютерных систем*

}

}



Интеллектуальная система (история понятия)

интеллектуальная система

:= [система, которая способна решать интеллектуальные задачи]

интеллектуальная задача

:= [задача, для которой отсутствует известный алгоритм ее решения]

интеллектуальная система

:= [система, которая способна обучаться]



Интеллектуальная система

интеллектуальная система

\coloneqq [система, которая может легко научиться решать новые задачи]

\Rightarrow *важно отличать**:

- { • способность обучаться более качественному решению задач одного ограниченного класса (как это делают нейросетевые модели)
 - способность обучаться решению задач разных классов (с ограничениями или без них)
- }

Интеллектуальная компьютерная система нового поколения



интеллектуальные компьютерные системы нового поколения

⇒ *предъявляемые требования**:

- *высокий уровень интероперабельности*
- *высокий уровень обучаемости*
- *высокий уровень гибридности*
- *высокий уровень способности решать интеллектуальные задачи (то есть задачи, методы решения которых и/или требуемая для их решения исходная информация априори неизвестны)*
- *высокий уровень синергетичности*

интероперабельность[^]

- := [способность к эффективному (целенаправленному) взаимодействию с другими самостоятельными субъектами]
- := [способность работать в коллективе (в команде)]
- := [уровень социализации]
- := [social skills]



Технология OSTIS



Комплексная задача

комплексная задача

\coloneqq [задача, для решения которой необходимо использовать различные виды знаний и различные модели решения задач]

\Rightarrow *примечание**:

[Для решения комплексных задач невозможно заранее определить набор моделей решения задач]

Примеры комплексных задач:

- задача понимания естественных языков, изображений, речевых сообщений
- задача планирования поведения интеллектуальных роботов
- задача комплексной автоматизации предприятий
- задача адаптивного обучения различным дисциплинам



Гибридная интеллектуальная система

гибридная интеллектуальная система

\coloneqq [интеллектуальная система, интегрирующая различные виды знаний и различные модели решения задач]

\Rightarrow *примечание**:

[ГИС ориентированны на решение комплексных задач]

\Rightarrow *недостатки современного состояния**:

- {• [монолитность]
 - [ориентированность на решение задачи одного класса (не нескольких различных классов задач)]
 - [требование колоссальных ресурсов (времени) для разработки систем]
 - [невозможность повторного использования компонентов систем для решения других задач (как следствие монолитности)]
- }



Технология OSTIS

OSTIS

:= [Open Semantic Technology for Intelligent Systems]

:= [открытая комплексная технология проектирования совместимых интеллектуальных систем]

⇒ *основные положения**:

- { • [база знаний OSTIS может описывать любой вид знаний]
- [решатель задач OSTIS основан на многоагентном подходе и позволяет легко комбинировать любые модели решения задач]
- [интерфейс ostis-системы представляет собой подсистему со своей БЗ и решателем задач (также может быть описан с помощью SC-кода)]
- [использование универсального способа представления (кодирования) информации, получившего название SC-код]
- }

OSTIS

⇒ *достоинства**:

- { • [унифицированность представления (любая информация представляется одинаково)]
- [удобство машинной обработки и восприятия человеком]
- [любые знания и модели решения задач легко интегрируются в ostis-систему (по принципу plug & play) и её всегда можно переобучить]
- [универсальность и совместимость компонентов (повторное использование позволяет сократить время разработки новых компонентов на 40-60%)]
- [система описывается с помощью SC-кода, поэтому она может анализировать себя, искать в себе ошибки и оптимизировать собственную работу (рефлексивность)]

}

OSTIS

⇒ *достоинства**:

- { • [платформенная независимость (разработка независима от архитектуры компьютера, платформа может быть реализована в программном варианте или в аппаратном)]
- [параллельная обработка информации]
- [ostis-система может включать в себя компоненты, разработанные на базе OSTIS, и объединяться с другими системами и интегрировать другие компоненты (с помощью JSON или API)]
- [производительность ostis-системы не хуже традиционной, а иногда может оказаться лучше за счёт параллельной обработки (при переходе на семантические компьютеры производительность будет ещё выше)]

}

Важно понимать:

- OSTIS – это не конкретная интеллектуальная система, а **технология разработки** интеллектуальных систем, каждая из которых будет решать задачи определённого класса
- ключевые преимущества OSTIS заключаются не в новых функциональных возможностях разрабатываемых систем (большинство функций ostis-систем можно реализовать с помощью традиционных средств), а в том, насколько легко **модифицировать и развивать** разрабатываемые системы, адаптировать их к новым задачам, а также насколько эффективно можно накапливать и использовать полученные компоненты при разработке новых систем, сокращая при этом время и трудоёмкость их разработки
- OSTIS – это способ решения проблемы **совместимости**, одной из важнейших проблем современных технологий



Архитектура ostis-системы





база знаний ostis-системы

:= [база знаний, описывающая любой вид знаний, при этом её легко дополнять новыми видами знаний]

решатель задач ostis-системы

:= [решатель задач, основанный на многоагентном подходе и позволяющий легко интегрировать и комбинировать любые модели решения задач]

интерфейс ostis-системы

:= [специализированная подсистема ostis-системы со своей базой знаний и решателем задач, предназначенная для общения ostis-системы с внешней средой]

⇒ *уточнение**:

[Интерфейс ostis-системы также может быть описан в SC-коде]



Информационные конструкции

Информационная конструкция

информационная конструкция

\coloneqq [конструкция (структура), содержащая некоторые сведения о некоторых сущностях]

\coloneqq [информация]

\Rightarrow *примечание**:

[Информационная конструкция имеет

- форму представления (текст, звук, изображение)
- форму структуризации (синтаксис)
- смысл (денотационную семантику)

]



информационная конструкция

\Rightarrow *разбиение**:

- $\{ \bullet$ *sc-множество*
 - $:=$ [внутренняя информационная конструкция ostis-системы, хранимая в её (внутренней) sc-памяти]
 - \bullet *файл*
 - $:=$ [внутренняя информационная конструкция ostis-системы, хранимая в её файловой (внешней) памяти]
 - \Rightarrow *примечание**:
 - [файл может храниться в памяти другой системы]
 - \bullet *внешняя информационная конструкция, не являющаяся ни файлом, ни sc-конструкцией*
- $\}$

\supset *дискретная информационная конструкция*

Дискретная информационная конструкция



дискретная информационная конструкция

⇒ *пояснение**:

[Каждая дискретная информационная конструкция — это информационная конструкция, смысл которой задается:

- множеством элементов (синтаксически атомарных фрагментов) этой информационной конструкции
- алфавитом этих элементов — семейством классов синтаксически эквивалентных элементов информационной конструкции
- принадлежностью каждого элемента информационной конструкции соответствующему классу синтаксически эквивалентных элементов информационной конструкции
- конфигурацией связей инцидентности между элементами информационной конструкции.

]

дискретная информационная конструкция

⇒ *примечание**:

[Форма представления элементов дискретной информационной конструкции для анализа её смысла не требует уточнения. Главным является:

- наличие простой процедуры выделения (сегментации) элементов дискретной информационной конструкции
- наличие простой процедуры установления синтаксической эквивалентности разных элементов дискретной информационной конструкции
- наличие простой процедуры установления принадлежности каждого элемента дискретной информационной конструкции соответствующему классу синтаксически эквивалентных элементов (т.е. соответствующему элементу алфавита).

]



Формальный язык

язык

\coloneqq [множество информационных конструкций, построенных по общим синтаксическим и семантическим правилам]

\Rightarrow разбиение*:

- { • *искусственный (построенный) язык*
- *естественный язык*
- }

\supset *формальный язык*

формальный язык

\coloneqq [язык, в котором значение каждого слова или знака, правила построения предложений и понимания их смысла однозначны]

\coloneqq [множество конструкций, составленных из некоторого конечного алфавита языка]

знак

:= [фрагмент информационной конструкции, обладающий свойством, обозначать некоторую сущность (объект), которая наряду с другими сущностями описывается указанной информационной конструкцией]

\Rightarrow *разбиение**:

- {
 - *знак, являющийся элементом дискретной информационной конструкции*
 - *знак, являющийся неатомарным фрагментом дискретной информационной конструкции*}



Под **знаком** понимается фрагмент *информационной конструкции*, который условно представляет (изображает) некоторую описываемую сущность, которую называют **денотатом знака**.

При этом отсутствие *знака*, обозначающего некоторую сущность, не означает отсутствие самой этой сущности. Это означает только то, что мы даже не догадываемся о ее существовании и, следовательно, не приступили к ее исследованию.

семиотика

:= [наука о знаках]

\Rightarrow *компоненты**:

- { • *синтаксис (изучает правильное построение текста)*
- *семантика (изучает значение знаков)*
- *прагматика (изучает отношение между знаком и субъектом)*
- }



Синтаксис и семантика

синтаксис

:= [наука, изучающая отношения между знаками]

\Rightarrow *примечание**:

[Синтаксис переводится как порядок, координация]

семантика

:= [наука, изучающая отношения между знаками и их значениями]

\Rightarrow *примечание**:

[Семантика переводится как обозначение, значение]

семантика знака

:= [отношение между знаком и сущностью (значением знака, денотатом), которую он обозначает]



Семантическая сеть

семантическая сеть

\equiv [граф, вершины которого являются знаками некоторых сущностей, а дуги (ребра) – знаками связей между этими сущностями]

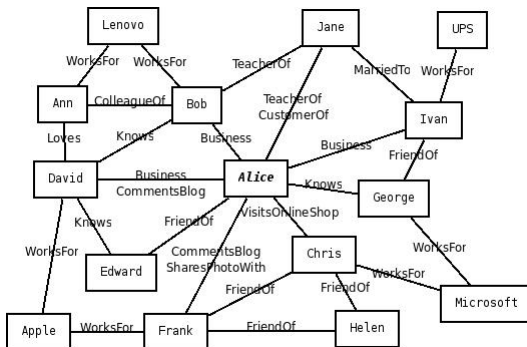


Рис.: Пример семантической сети



Семантическая память

семантическая память

:= [графодинамическая (нелинейная) память]

:= [смысловая память, обеспечивающая хранение семантических сетей]

⇒ *примечание**:

[Изменение семантической памяти происходит не только путем изменения состояния элементов памяти (вершин графа), но также и изменением конфигурации связей между элементами (ребер или дуг графа)]



Лекция 2

Базовый язык представления знаний

Шункевич Д.В.

Белорусский государственный университет информатики и радиоэлектроники

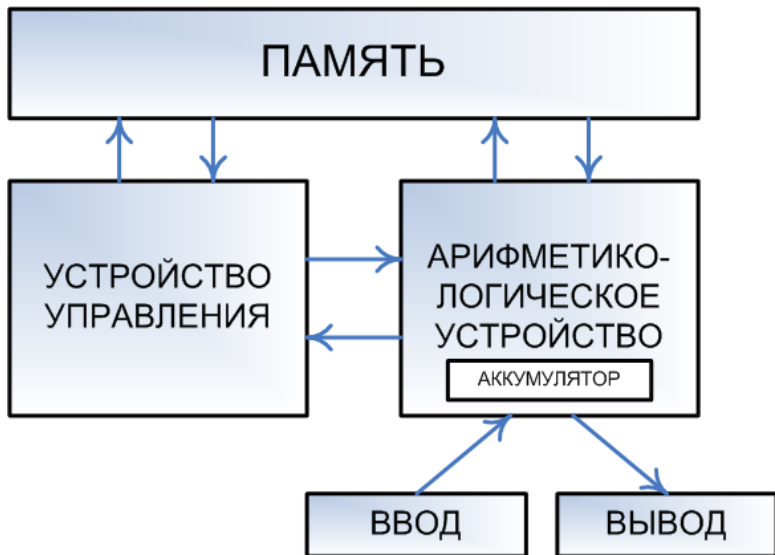


Содержание лекции

Основные положения базового языка представления знаний в интеллектуальных системах – SC-кода. Алфавит, синтаксис, базовая денотационная семантика SC-кода. Синтаксическая и семантическая классификация sc-элементов.



Архитектура фон-Неймана





Архитектура фон-Неймана

- память ЭВМ однородна (линейна)
- доступ к ячейкам памяти ЭВМ осуществляется по адресу
- алфавитом, используемым для представления данных и команд, является множество 0, 1
- каждая программа состоит из набора команд, выполняемых процессором последовательно
- возможно присутствие в программах команд условных переходов



Двоичное представление

```
1011010010011001100100110110011000010010011001001010100001100:
01010111001010110010110010101011100101010110010101100101010:
0100100110011001001101100110000100100110010010101000011001010:
1001100110010011011001100001001001001010100001100101011001:
100110110011000010010010010010101000011001011001011011001:
0100100110011001101100110000100100110010010101000011001010:
010011011001100001001001100100101010000110010101100101101101:
0011001100100110110011000010010011001001010100001100101011001:
1011010010011001100100110110011000010010011001001010100001100:
010100100110011001001101100110000100100110010010101000011001:
10010011001100100110011001100001001001100100101010000110010101:
1001100110010011011001100001001001100100101010000110010101100:
101010010011001100100110110011000010010011001001010100001100101:
10010011001100100110110011000010010011001001010100001100101011:
11001100100110110011000010010011001001010100001100101011001011:
11001001100110011000010010011001001010100001100101011001011011:
001001101100110000100100110010010101000011001010110010110101:
0011001001101100110000100100110010010101000011001010110010110:
10110100100110011001001101100110000100100110010010101000011001:
0100100110011001001101100110000100100110010010101000011001010:
10011011001100001001001100100101010000110010101100101101100:
0010011001100100110110011000010010011001001010100001100101011:
0010011001100100110110011000010010011001001010100001100101011:
0010011001100100110110011000010010011001001010100001100101011:
0110100100110011001001101100110000100100110010010101000011001:
1011010010011001100100110110011000010010011001001010100001100:
1101010010011001100100110110011000010010011001001010100001100:
1001001100110010011011001100001001001100100101010000110010101:
01001001100110010011011001100001001001100100101010000110010101:
01001001100110010011011001100001001001100100101010000110010101:
01001001100110010011011001100001001001100100101010000110010101:
```

Достоинства:

- легко реализовать
- удобно для машины

Недостатки:

- неудобно для человека
- невозможно интерпретировать без контекста



Предлагаемый подход

Разработать способ кодирования информации, который

- удобен для представления и обработки в компьютере
- удобен для восприятия человеком
- обладает «осмысленностью», может быть однозначно интерпретирован

Требуется переход от *данных* к *знаниям*.

Одна из ключевых отличительных особенностей знаний – интерпретируемость (Д. А. Пospelов).

SC-код

\coloneqq [способ универсального смыслового представления (кодирования) информации в памяти компьютерных систем]

\coloneqq [semantic computer code]

\Rightarrow *примечание**:

[SC-код основан на теории графов (синтаксис) и теории множеств (семантика), что обеспечивает универсальность и унифицированность (единообразие) представления информации, удобство машинной обработки и восприятия человеком]



- **Двоичное кодирование**
 - алфавит $\{0, 1\}$
 - не удобен для человека
 - удобен для обработки в компьютере
 - нельзя понять информацию без контекста
 - легко реализовать
- **SC-код**
 - базовый алфавит состоит из 5 элементов
 - удобен для человека
 - удобен для обработки в компьютере
 - обладает «осмысленностью»



Формы представления SC-кода

SC-код

:= [абстрактный язык, который находится в памяти компьютерной системы]

\Rightarrow *примечание**:

[С помощью SC-кода можно описывать базы знаний, решатели задач и интерфейс интеллектуальной системы. SC-код является абстрактным языком, но его можно визуализировать в различных формах]

Формы внешнего представления SC-кода

- SCs-код (текстовый линейный)
- SCn-код (гипертекстовый)
- SCg-код (графический)



Формы представления SC-кода

SCs

:= [Semantic Code string]

:= [язык линейного представления знаний]

```
concept_triangle <= nrel_subdividing:  
{  
  concept_acute_angle_triangle;  
  concept_rectangular_triangle;  
  concept_obtuse_triangle  
};
```

SCn

:= [Semantic Code natural]

:= [язык структурированного представления знаний]

```
треугольник  
= разбиение*: ...  
  ▸ остроугольный треугольник  
  ▸ прямоугольный треугольник  
  ▸ тупоугольный треугольник
```

SCg

:= [Semantic Code graphic]

:= [язык графического представления знаний]



Основные положения SC-кода

SC-код

⇒ *пояснение**:

[Информация содержится не в самих знаках, а в конфигурации связей между ними]

∈ *абстрактный язык*

∈ *графовый язык*

⇒ *основные положения**:

- { • [знаки сущностей – синтаксически элементарные фрагменты sc-текстов, такие знаки не имеют внутренней структуры]
- [информационные конструкции, не являющиеся sc-текстами (текстами SC-кода), могут быть включены в базу знаний в виде файлов]
- [тексты SC-кода имеют нелинейную структуру, так как каждый знак сущности входит в базу знаний однократно и имеет неограниченное количество связей с другими знаками сущностей]

}

SC-код

⇒ *основные положения**:

- { • [база знаний является графовой структурой, алфавит элементов которой включает в себя множество узлов, рёбер, дуг, базовых дуг, а также множество специальных узлов (файлов)]
- [структурная особенность графовой структуры базы знаний заключается в том, что её дуги и ребра могут связывать не только узел с узлом, но и узел с ребром или дугой и т.д.]
- [отсутствие синонимии осуществляется благодаря склеиванию одинаковых sc-элементов]
- [sc-узлы, sc-ребра и sc-дуги являются обозначениями различных сущностей]
- }



Синтаксис. Алфавит SC-кода

алфавит SC-кода

:= [семейство синтаксических меток, приписываемых sc-элементам и указывающих факт принадлежности sc-элемента к соответствующему классу sc-элементов]

минимальный алфавит SC-кода

\Rightarrow *пояснение**:

[Для понимания sc-конструкций, хранимых в sc-памяти, достаточно синтаксически выделить только *Класс константных постоянных позитивных sc-пар принадлежности (Класс базовых sc-дуг)*, с помощью которых каждый sc-элемент будет явно соединяться с sc-классами, которым этот sc-элемент принадлежит. Очевидно, что таким явным способом выделить базовые sc-дуги с помощью самих этих базовых sc-дуг невозможно.]



Минимальный алфавит SC-кода

Таким образом, любой класс sc-элементов можно выделить явно путём:

- введения sc-элемента, являющегося знаком этого класса sc-элементов (sc-класса);
- проведения постоянных позитивных sc-пар принадлежности во все sc-элементы, являющиеся элементами выделяемого sc-класса и хранимые (присутствующие) в текущем состоянии sc-памяти.

минимальный алфавит SC-кода

\coloneqq [*Класс базовых sc-дуг и Класс всех остальных sc-элементов* (по умолчанию)]

Алфавит Ядра SC-кода

Алфавит Ядра SC-кода

- = {
- *sc-узел, являющийся знаком файла*
 - *sc-узел, не являющийся знаком файла*
 - *базовая sc-дуга*
 - *sc-ребро*
 - *sc-дуга общего вида*
- }





Алфавит Ядра SC-кода

sc-ребро

:= [Класс *sc-элементов*, имеющих в рамках Ядра SC-кода синтаксическую метку обозначений неориентированных *sc-пар*]

sc-дуга общего вида

:= [Класс *sc-элементов*, имеющих в рамках Ядра SC-кода синтаксическую метку обозначений ориентированных *sc-пар*, не являющихся постоянными позитивными *sc-парами принадлежности*]

базовая sc-дуга

:= [Класс *sc-элементов*, имеющих в рамках Ядра SC-кода синтаксическую метку постоянных позитивных *sc-пар принадлежности*]

sc-узел, являющийся знаком файла

:= [*sc-элементов*, имеющий в рамках Ядра SC-кода синтаксическую метку *sc-элементов*, являющихся знаками файлов]

sc-узел, не являющийся знаком файла

:= [*sc-узел*, имеющий в рамках Ядра SC-кода синтаксическую метку *sc-элементов*, не являющихся ни знаками файлов, ни обозначениями *sc-пар*]



Семантическая классификация *sc*-элементов

К числу базовых признаков классификации *sc*-элементов относятся:

- структурный признак;
- логико-семантический признак;
- темпоральная характеристика сущностей, обозначаемых *sc* - элементами, которая в свою очередь, включает в себя:
 - постоянство или временность существования обозначаемой сущности;
 - статичность (стационарность) или динамичность (изменчивость) обозначаемой сущности.

sc-элемент

\Rightarrow разбиение*:

Структурный признак классификации sc-элементов

- = {
- обозначение sc-множества
 - обозначение внешней сущности

\Rightarrow разбиение*:

- {
- обозначение файла
 - обозначение информационной конструкции, не являющейся ни sc-множеством, ни файлом
 - обозначение внешней сущности, не являющейся информационной конструкцией

}

}

Структурная классификация sc-элементов



обозначение sc-множества

\Rightarrow разбиение*:

{ • *обозначение sc-связки*

\Rightarrow разбиение*:

{ • *обозначение sc-синглтона*

• *обозначение sc-пары*

\Rightarrow разбиение*:

{ • *обозначение неориентированной sc-пары*

• *обозначение ориентированной sc-пары*

}

• *обозначение sc-связки, не являющейся синглтоном и парой*

}

• *обозначение sc-класса*

• *обозначение sc-структуры*

}

обозначение ориентированной sc-пары

\Rightarrow разбиение*:

{ • *обозначение sc-пары принадлежности*

\Rightarrow разбиение*:

{ • *обозначение sc-пары нечеткой принадлежности*

• *обозначение sc-пары позитивной принадлежности*

• *обозначение sc-пары негативной принадлежности*

}

• *обозначение ориентированной sc-пары, не являющейся парой принадлежности*

}

Логико-семантическая классификация sc-элементов

$= \{$

sc-элемент

\Rightarrow разбиение*:

$\{ \bullet$ *sc-константа*

$:=$ [sc-элемент, логико-семантическим значением которого является он сам]

\bullet *sc-переменная*

$\}$

$\}$



Классификация sc-элементов по темпоральным характеристикам обозначаемых ими сущностей

$= \{$

sc-элемент

\Rightarrow разбиение*:

Признак постоянства существования сущностей, обозначаемых sc-элементами

$= \{ \bullet$ *обозначение постоянной сущности*
 $:=$ [обозначение постоянно существующей сущности]
 \bullet *обозначение временной сущности*
 $\}$

$\}$



Признак статичности сущностей

sc-элемент

\Rightarrow разбиение*:

Признак статичности сущностей, обозначаемых sc-элементами

$= \{ \bullet$ *обозначение статической сущности*
 $:=$ [обозначение сущности, изменения которой в рамках соответствующего отрезка времени считаются несущественными]
 \supset *обозначение статического sc-множества*
 \bullet *обозначение динамической сущности*
 $:=$ [обозначение сущности изменяющейся во времени]
 \supset *обозначение динамического sc-множества*
}



Обозначение временной сущности

обозначение временной сущности

⇒ разбиение*:

- { • *обозначение внешней временной сущности*
 - ⊃ *обозначение внешней ситуации*
 - ⊃ *обозначение внешнего события*
 - ⊃ *обозначение внешнего процесса*
- *обозначение внутренней временной сущности в sc-памяти*

обозначение внутренней временной сущности в sc-памяти

⇒ разбиение*:

- { • *обозначение ситуации в sc-памяти*
:= [обозначение ситуации, которая возникла или возникает в процессе обработки информации в sc-памяти]
- *обозначение события в sc-памяти*
:= [обозначение события, которое произошло или произойдет в процессе обработки информации в sc-памяти]
- *обозначение информационного процесса в sc-памяти*
:= [обозначение внутреннего процесса в sc-памяти, который происходит, произошёл или будет происходить]
- }



Темпоральные свойства sc-элементов

Когда речь идёт о темпоральных свойствах sc-элементов, следует чётко отличать:

- временный характер присутствия любого sc-элемента в составе той базы знаний, в которой он находится
- временный характер присутствия в sc-памяти всей заданной sc-конструкции (ситуации в sc-памяти);
- временный характер существования внешней сущности, которую sc-элемент обозначает;
- статичный (динам.) характер внешней сущности, обозначаемой sc-элементом; динамический характер внешней сущности, предполагает наличие в sc-памяти описания процесса изменения состояния или конфигурации указанной внешней сущности;
- динам. sc-множество, являющееся отражением соответствующего внешнего процесса;
- динам. sc-множество, являющееся отражением соответствующего внутреннего процесса

Структурная классификация sc-констант



Данная классификация полностью аналогична *Структурной классификации sc-элементов*, в отличие от которой она описывает структурную классификацию только константных sc-элементов (sc-констант)



Структурная классификация sc-констант
 $= \{$

sc-константа

\Rightarrow разбиение*:

$\{ \bullet \text{ } sc\text{-множество}$
 $\bullet \text{ } \text{внешняя сущность}$
 $\}$
 $\}$



внешняя сущность

внешняя сущность

\coloneqq [sc-элемент, являющийся знаком внешней сущности]

\coloneqq [знак внешней сущности]

\coloneqq [знак сущности, не являющейся sc-множеством (sc-конструкцией)]

\Rightarrow разбиение*:

- {• файл
 - внутренний файл
 - внешняя сущность, не являющаяся внутренним файлом
 - внешняя сущность, не являющаяся информационной конструкцией
 - информационная конструкция, не являющаяся ни sc-множеством, ни файлом



SC-МНОЖЕСТВО

sc-множество

\Rightarrow разбиение*:

{ • *sc-связка*

\Rightarrow разбиение*:

{ • *sc-синглетон*

• *sc-пара*

• *sc-связка, не являющаяся синглетоном и парой*

}

• *sc-класс*

• *sc-структура*

}

sc-пара

sc-пара

\Rightarrow разбиение*:

- {• неориентированная *sc-пара*
- ориентированная *sc-пара*

\Rightarrow разбиение*:

- {• *sc-пара* принадлежности

\Rightarrow разбиение*:

- {• *sc-пара* нечёткой принадлежности
- *sc-пара* позитивной принадлежности
- *sc-пара* негативной принадлежности
- }

- ориентированная *sc-пара*, не являющаяся *sc-парой* принадлежности

}

}



sc-пара позитивной принадлежности

sc-пара позитивной принадлежности

⊃ *sc-пара постоянной позитивной принадлежности*

⇐ *пересечение множеств**:

- {• *sc-константа*
- *постоянная сущность*
- *статическая сущность*
- *sc-пара позитивной принадлежности*

⊃ *sc-пара временной позитивной принадлежности*

⇐ *пересечение множеств**:

- {• *sc-константа*
- *временная сущность*
- *динамическая сущность*
- *sc-пара позитивной принадлежности*



Лекция 3

Внешние информационные конструкции

Шункевич Д.В.

Белорусский государственный университет информатики и радиоэлектроники



Содержание лекции

Понятие внешней информационной конструкции, классификация. Понятие внутреннего файла базы знаний, классификация. Понятие идентификатора, типология идентификаторов. Примеры формализации идентификаторов.



Информационная конструкция

информационная конструкция

- := [конструкция (структура), содержащая некоторые сведения о некоторых сущностях]
- := [информационная модель, состоящая из некоторого множества различных *знаков*, обозначающих моделируемые (описываемые) *сущности* любого вида и, в частности, *знаков*, обозначающих различного вида связи между знаками описываемых *сущностей* (такие связи чаще всего являются отражениями (моделями) связей между *сущностями*, которые обозначаются связываемыми *знаками*)]
- ⇒ *примечание**:
[Форма представления ("изображения", "материализации"), форма структуризации (синтаксическая структура), а также *смысл** (денотационная семантика) *информационных конструкций* могут быть самыми различными.]

Дискретная информационная конструкция



дискретная информационная конструкция

\subset *информационная конструкция*

\Rightarrow *пояснение**:

[Каждая дискретная информационная конструкция – это информационная конструкция, смысл которой задается

- множеством элементов (синтаксически атомарных фрагментов) этой информационной конструкции
- алфавитом этих элементов – семейством классов синтаксически эквивалентных элементов информационной конструкции
- принадлежностью каждого элемента информационной конструкции соответствующему классу синтаксически эквивалентных элементов информационной конструкции
- конфигурацией связей инцидентности между элементами информационной конструкции

]



Информационные конструкции

информационная конструкция

⇒ разбиение*:

- { • *внутренняя информационная конструкция*
:= [информационная конструкция, хранимая в памяти некоторой кибернетической системы, и непосредственно интерпретируемая (понимаемая) решателем задач этой системы]
- *внешняя информационная конструкция*
:= [информационная конструкция, представленная на каком-либо внешнем носителе или в памяти другой кибернетической системы]
- *файл*
:= [первичный электронный образ некоторой внешней информационной конструкции]
- }

файл

:= [информационная конструкция, которая не является *sc*-конструкцией и которая может храниться в файловой памяти *ostis*-системы]

\Rightarrow *примечание**:

[файловая память *ostis*-системы, хранящая различного рода *информационные конструкции* (образы, модели), не являющиеся *sc*-конструкциями должна быть тесно связана с *sc*-памятью этой же *ostis*-системы. Как минимум, каждый файл *ostis*-системы должен быть связан с тем *sc*-элементом, которых является знаком этого файла (точнее, знаком синглтона, элементом которого является указанный файл)]

файл

:= [sc-узел, обозначающий файл]

:= [знак файла]

\Rightarrow *разбиение**:

{ • *ея-файл*

:= [естественно-языковой файл]

- *файл, являющийся текстом формального языка*

\supset *sc.g-файл*

\supset *sc.n-файл*

\supset *sc.s-файл*

- *файл, отражающий процесс изменения sc.g-текста*
- *графический файл*
- *файл-изображение*
- *видео-файл*
- *аудио-файл*

}

файл

⇒ *разбиение**:

- { • *файл-экземпляр*
:= [файл, являющийся конкретным электронным документом или электронным образом конкретной внешней информационной конструкции]
- *файл-образец*
:= [файл-класс *ostis-системы*]
:= [файл, являющийся одновременно также и знаком множества всевозможных экземпляров (копий) этого файла]

⇒ *разбиение**:

- { • *внешний файл ostis-системы*
- *внутренний файл ostis-системы*

файл

⇒ *примечание**:

[Представление *информационных конструкций* в виде файлов ориентировано на представление *дискретных (!) информационных конструкций*. Поэтому "файловое" представление *недискретных информационных конструкций* (например, различного рода сигналов) предполагает "дискретизацию" таких конструкций, т.е. преобразование их в *дискретные*. Так преобразуются аудио-сигналы (в частности, речевые сообщения), изображения, видео-сигналы и др.]



Внутренний файл

внутренний файл ostis-системы

⇒ *примечание**:

[sc-узел может быть знаком файла, находящегося в памяти другой ostis-системы (не в той, в которой хранится этот sc-узел). Но в этом случае указанный sc-узел не будет принадлежать рассматриваемому классу sc-узлов.]

⇒ *примечание**:

[Следует отличать синтаксическую эквивалентность файлов, семантическую эквивалентность файлов и совпадение файлов (когда речь идет об одном и том же файле). Т.е. копия файла и один и тот же файл – это разные вещи]

Идентификатор

sc-идентификатор

\coloneqq [строка символов или пиктограмма, взаимно однозначно представляющая соответствующий *sc-элемент*, хранимый в *sc-памяти*]

\coloneqq [внешний идентификатор *sc-элемента*]

\Rightarrow разбиение*:

{ • *простой sc-идентификатор*

\coloneqq [простой внешний идентификатор *sc-элемента*]

• *sc-выражение*

\coloneqq [сложный внешний идентификатор *sc-элемента*, в состав которого входит один или несколько идентификаторов других *sc-элементов*]

}



Идентификатор

sc-идентификатор

\Rightarrow *разбиение**:

- { • *основной sc-идентификатор*
 - $:=$ [основной *sc-идентификатор* для носителей дополнительно указываемого языка общения (например, соответствующего естественного языка)]
 - \Rightarrow *примечание**:
 - [основной *sc-идентификатор* является уникальным (не имеет синонимов и омонимов) в рамках соответствующего естественного языка]
 - \supset *основной международный sc-идентификатор*
- *неосновной sc-идентификатор*
 - \supset (*неосновной sc-идентификатор* \cap *пояснение*)
 - $:=$ [неосновной *sc-идентификатор*, являющийся одновременно и пояснением обозначаемой сущности]
 - \supset (*неосновной sc-идентификатор* \cap *определение*)
 - $:=$ [неосновной *sc-идентификатор*, являющийся одновременно и определением обозначаемого понятия]
 - \supset *неосновной часто используемый sc-идентификатор*



Неосновной sc-идентификатор

С помощью неосновных sc-идентификаторов указываются возможные *синонимы** соответствующего *основного sc-идентификатора*, которые в частности, могут пояснять или даже определять обозначаемую сущность, указывает на важные свойства этой сущности.

Для некоторых sc-элементов могут часто использоваться не только основные, но и неосновные sc-идентификаторы (особенно в неформальных текстах – в пояснениях, примечаниях и т.п.). Явное выделение такого класса sc-идентификаторов позволяет упростить семантический анализ исходных текстов баз знаний.



Основной sc-идентификатор

основной sc-идентификатор

⊂ *файл-образец ostis-системы*

⇔ *семантическая эквивалентность**:

[Все основные идентификаторы *sc-элементов* в памяти *ostis-системы* оформляются в виде копируемых фалов-образцов *ostis-системы*.]

⇒ *пояснение**:

[Копии основных *sc-идентификаторов* входят в состав внешних текстов различных языков (SCg-кода, SCs-кода, SCn-кода), а также в различных падежах, склонения, спряжениях в состав файлов *ostis-систем*.]

Построение основных sc-идентификатор



В качестве *основных sc-идентификаторов* могут использоваться также общепринятые международные условные обозначения некоторых сущностей, например, обозначения часто используемых функций (\sin , \cos , tg , \log , и т.д.), единиц измерения, денежных единиц и многое другое. Формально каждый основной международный sc-идентификатор считается основным sc-идентификатором также и для каждого естественного языка, несмотря на то, что символы, используемые в основных международных sc-идентификаторах, могут не соответствовать алфавиту некоторых или даже всех естественных языков.

Идентификатор

sc-идентификатор

⇒ *разбиение**:

- *строковый sc-идентификатор*
 - := [*sc-идентификатор*, представленный строкой символов, которая является именем обозначаемой сущности]
 - := [имя сущности, обозначаемой идентифицируемым *sc-элементом*]
 - := [имя (термин, словосочетание), синонимичное соответствующему (идентифицируемому) *sc-элементу* и представленное в соответствующем алфавите символов]
- *нестроковый sc-идентификатор*
 - ⇒ *пояснение**:
[В общем случае в качестве *sc-идентификатора* некоторого *sc-элемента* может выступать произвольный *внутренний файл ostis-системы*, например, пиктограмма, условное обозначение или даже аудиофрагмент.]

⇒ *примечание**:

[Введенные нами *sc-идентификаторы* используются во всех внешних языках, близких SC-коду – в SCg-коде, в SCs-коде и в SCn-коде.]

Строковый sc-идентификатор

строковый sc-идентификатор

- \coloneqq [имя, приписываемое идентифицируемому *sc-элементу*]
- \coloneqq [имя сущности, обозначаемой идентифицируемым *sc-элементом*]
- \coloneqq [строка символов, синонимичная соответствующему идентифицируемому *sc-элементу*]
- ⊃ *основной строковый sc-идентификатор*
 - \coloneqq [уникальное для каждого естественного языка внешнее имя, приписываемое идентифицируемому *sc-элементу*]
 - ⊃ *основной русскоязычный sc-идентификатор*
 - ⊃ *системный sc-идентификатор*
 - ⊃ *основной англоязычный sc-идентификатор*
 - ⊃ *основной германоязычный sc-идентификатор*
 - ⊃ *основной франкоязычный sc-идентификатор*
 - ⊃ *основной италоязычный sc-идентификатор*
 - ⊃ *основной китайскоязычный sc-идентификатор*
- ⊃ *системный sc-идентификатор*



Системный *sc*-идентификатор

системный sc-идентификатор

\coloneqq [*sc-идентификатор*, являющийся уникальным в рамках всей базы знаний Экосистемы *OSTIS* (Глобальной базы знаний).]

\Rightarrow *пояснение**:

[Данный *sc-идентификатор*, как правило, используется в исходных текстах базы знаний, при обмене сообщениями между *ostis-системами*, а также для взаимодействия *ostis-системы* с компонентами, реализованными с использованием средств, внешних с точки зрения *Технологии OSTIS*, например, программ, написанных на традиционных языках программирования. Алфавит системных *sc-идентификаторов* максимально упрощен для того, чтобы обеспечить удобство автоматической обработки таких *sc-идентификаторов* с использованием современных технических средств, в частности, запрещены пробелы и различные специальные символы.]

Построение системных sc-идентификаторов



Символами, использующимися в *системном sc-идентификаторе*, могут быть буквы латинского алфавита, цифры, знак нижнего подчеркивания и знак тире. Для обеспечения интернационализации рекомендуется формировать *системные sc-идентификаторы* на основании основных англоязычных *sc-идентификаторов*.

Таким образом, наиболее целесообразно формировать *системный sc-идентификатор sc-элемента* из основного англоязычного путем замены всех символов, не входящих в описанный выше алфавит на символ нижнее подчеркивание (“_”). Кроме того, заглавные буквы чаще всего заменяются на соответствующие строчные.

Для именования *sc-элементов*, являющихся знаками *ролевых отношений*, вместо знака “/” в *системном sc-идентификаторе* используется приставка “trcl” и далее после нижнего подчеркивания записывается имя *ролевого отношения*.

Построение системных sc-идентификаторов



Для именования sc-элементов, являющихся знаками *неролевых отношений*, вместо знака “*” в *системном sc-идентификаторе* используется приставка “nrel” и далее после нижнего подчеркивания записывается имя *неролевого отношения*.

Для именования sc-элементов, являющихся знаками классов *понятий* в *системном sc-идентификаторе* используется приставка “concept” и далее после нижнего подчеркивания записывается имя *класса*.

Для именования sc-элементов, являющихся знаками *структур* в *системном sc-идентификаторе* используется приставка “struct” и далее после нижнего подчеркивания записывается имя *структуры*.



Нетранслируемый sc-идентификатор

нетранслируемый sc-идентификатор

\subset *системный sc-идентификатор*

$:=$ [*sc-идентификатор*, не представляемый в базе знаний *ostis-системы*]

$:=$ [*sc-идентификатор*, существующий только вне базы знаний *ostis-системы*]



Нетранслируемый sc-идентификатор

Нетранслируемые sc-идентификаторы используются только в рамках исходных текстов *баз знаний* (в том числе, *sc.s-текстов*) и при обмене сообщениями между *ostis-системами* в тех случаях, когда необходимо в нескольких фрагментах исходного текста *базы знаний* или передаваемого сообщения использовать имя одного и того же *sc-элемента*, но при этом указанный *sc-элемент* не имеет *системного sc-идентификатора* и вводить его нецелесообразно.

Использование *нетранслируемых sc-идентификаторов* позволяет повысить читабельность и структурированность исходных текстов *баз знаний*, а также позволяет обратиться к одному и тому же именованному (в рамках базы знаний) *sc-элементу* в разных файлах исходных текстов *баз знаний* или в разных сообщениях, передаваемых между *ostis-системами*. В качестве таких *sc-элементов* часто выступают знаки *структур* и *связок*.

Таким образом, *нетранслируемые sc-идентификаторы* существуют только вне *базы знаний ostis-системы* и при формировании базы знаний из исходных текстов или при погружении в базу знаний полученного сообщения соответствующий им *внутренний файл ostis-системы* не создается.



Отличия идентификаторов

Системные идентификаторы отличаются от основных, во-первых, требованием к уникальности в рамках всей базы знаний *Экосистемы OSTIS* (а, значит, независимостью от внешнего языка), а во-вторых, более простым алфавитом, удобным для автоматической обработки.

Системные sc-идентификаторы и нетранслируемые sc-идентификаторы выполняют схожие функции, связанные с именованием *sc-элементов* на уровне исходных текстов *баз знаний* или передаваемых между *ostis-системами* сообщений.

Каждый *системный sc-идентификатор* представляется в базе знаний в виде *внутреннего файла ostis-системы* и связан с соответствующим *sc-элементом* парой отношения *системный sc-идентификатор**. *Нетранслируемые sc-идентификаторы* не представляются в рамках *базы знаний*, не имеют соответствующих *внутренних файлов ostis-системы* и на уровне *базы знаний* никак не связаны с идентифицируемыми ими *sc-элементами*.



Простой sc-идентификатор

Простой sc-идентификатор – идентификатор sc-элемента, в состав которого идентификаторы других sc-элементов не входят и который не содержит *транслируемую в SC-код* информацию об обозначаемой им сущности.

Простой строковый sc-идентификатор – простой sc-идентификатор, представляющий собой строку (цепочку) символов, которая является именем (названием) той же сущности, что и идентифицируемый sc-элемент. Простые строковые sc-идентификаторы являются наиболее распространенным видом идентификаторов, приписываемых sc-элементам.



Простой sc-идентификатор

простой строковый sc-идентификатор

- ⊃ *системный sc-идентификатор*
- ⊃ *простой строковый идентификатор sc-переменной*
 - ⊃ *пример'*:
[_var1]
- ⊃ *простой строковый sc-идентификатор неролевого отношения*
 - \coloneqq [простой строковый идентификатор sc-узла, являющегося знаком неролевого отношения]
 - ⊃ *пример'*:
[включение множеств*]
- ⊃ *простой строковый sc-идентификатор ролевого отношения*
 - \coloneqq [простой строковый идентификатор sc-узла, являющегося знаком ролевого отношения]
 - ⊃ *пример'*:
[слагаемое']



Простой sc-идентификатор

- ⌋ *простой строковый sc-идентификатор класса классов*
 - \coloneqq [простой строковый идентификатор sc-узла, являющегося знаком класса классов]
 - \ni *пример'*:
[длина^]
- ⌋ *sc-идентификатор внешнего файла ostis-системы*
 - \coloneqq [URL-идентификатор]
 - \ni *пример'*:
["file:///home/user/image1.png"]
 - \Rightarrow *примечание**:
[Данный sc-идентификатор описывает абсолютный путь к файлу под названием "image1.png"]
 - \ni *пример'*:
["file://image1.png"]
 - \Rightarrow *примечание**:
[Данный sc-идентификатор описывает относительный путь к файлу под названием "image1.png"]
 - \ni *пример'*:
["https://conf.ostis.net/content/image1.png"]
 - \Rightarrow *примечание**:
[Данный sc-идентификатор описывает путь к файлу под названием "image1.png", расположенному на удаленном сервере.]



Простой sc-идентификатор

sc-идентификаторы внешних файлов ostis-систем предназначены для описания местоположения внешних файлов ostis-систем и представляют собой строку символов, которая строится в соответствии со стандартом URL, а затем берется в двойные кавычки. Кавычки нужны для однозначности определения того, где начинается и заканчивается данный sc-идентификатор, поскольку в общем случае в URL разрешены пробелы. Целесообразность этого обусловлена тем, что sc-идентификаторы данного типа часто используются в файлах исходных текстов баз знаний ostis-систем.

sc-идентификаторы внешних файлов ostis-систем с точки зрения Технологии OSTIS являются простыми строковыми sc-идентификаторами, хотя и могут содержать специальные символы, например "%" или "/". Это связано с тем, что указанные идентификаторы не несут в себе семантически значимой информации о свойствах самого sc-элемента, обозначаемого таким sc-идентификатором, а только информацию о его расположении в текущем состоянии внешнего мира ostis-системы.



Простой sc-идентификатор

Правила построения простых строковых sc-идентификаторов включают в себя:

- Алфавит символов, используемых в простых строковых sc-идентификаторах;
- Префиксы и суффиксы, используемые в простых строковых sc-идентификаторах;
- Разделители и ограничители, используемые в простых строковых sc-идентификаторах;
- Правила построения *имен собственных* и *имен нарицательных*, являющихся простыми строковыми sc-идентификаторами;
- Правила построения простых строковых sc-идентификаторов, определяемые различными классами идентифицируемых sc-элементов.



Сложный sc-идентификатор

sc-выражение – идентификатор, который не только обозначает соответствующую сущность, но также содержит информацию, представляющую собой по возможности однозначную спецификацию указанной сущности. Однозначную спецификацию сущности, которая является понятием, называют определением этого понятия



Сложный sc-идентификатор

sc-выражение

- := [имя соответствующей (именуемой) сущности построенное по принципу "та (тот), которая (который) указываемым образом связана с другими указываемыми сущностями"]
- := [выражение, идентифицирующее sc-элемент]
- := [идентификатор sc-элемента, в состав которого входят другие идентификаторы и денотационная семантика которого точно определяется конкретным набором правил построения таких сложных (комплексных) идентификаторов, состоящих из определенным образом связанных между собой других идентификаторов]
- := [сложный идентификатор, состоящий из других идентификаторов]
- := [идентификатор, который представляет собой конструкцию, состоящую из нескольких других идентификаторов, а также из некоторых разделителей и ограничителей, и денотационная семантика которого однозначно задается конфигурацией указанной конструкции]
- := [сложный sc-идентификатор]
- := [сложный (составной) внешний идентификатор sc-элемента]
- := [выражение, идентифицирующее sc-элемент]
- := [sc-идентификатор, в состав которого входит один или несколько простых sc-идентификаторов]



Сложный sc-идентификатор

sc-выражение разбивается на:

- *sc-выражение неориентированного множества* – *sc-выражение*, ограниченное фигурными скобками
- *sc-выражение структуры* – *sc-выражение*, обозначающее структуру, представленную на любом известном и легко определяемом языке (Русском, Английском, SCg-коде, SCs-коде, SCn-коде)

sc-выражение структуры обозначает структуру, содержащую *sc-текст*, семантически эквивалентный тому тексту (на некотором известном языке), который заключен в фигурные скобки. Чаще всего такой текст записывается на формальном языке, например, SCs-коде, и может быть автоматически однозначно интерпретирован. Возможна ситуация, когда указанный текст записан на менее неформальном языке, например, Русском, но в этом случае его автоматическая интерпретация значительно усложняется и в общем случае не всегда однозначна.

В текущей реализации средств разработки исходных текстов баз знаний в соответствии с более старой версией правил построения *sc-выражений* вместо фигурных скобок *sc-выражение структуры* ограничивается квадратными скобками со звездочками ("[" и "]*").

- *sc-выражение ориентированного множества* – *sc-выражение* кортежа, ограничиваемое угловыми скобками и обозначающее упорядоченное (ориентированное) множество *sc-элементов*, порядок которых задаётся последовательностью перечисляемых их *sc-идентификаторов*.

Сложный sc-идентификатор

- *sc-выражение внутреннего файла ostis-системы* – sc-выражение, обозначающее *внутренний файл ostis-системы*, визуальное представление (изображение) которого ограничивается квадратными скобками. sc-выражение внутреннего файла ostis-системы обозначает *внутренний файл ostis-системы*, содержимое которого заключено в квадратные скобки, ограничивающие данное sc-выражение.

Дополнительная спецификация *внутреннего файла ostis-системы* легко осуществляется с помощью *SC-кода*. Сюда входит язык, на котором представлена *информационная конструкция*, являющаяся содержимым *файла*, формат кодирования, *автор** и многое другое.

- *sc-выражение, обозначающее файл-образец ostis-системы* – sc-выражение, ограниченное квадратными скобками с восклицательными знаками.
- *sc-выражение, построенное на основе бинарного отношения* – sc-выражение, в состав которого входят либо (1) *sc-идентификатор*, обозначающий бинарное ориентированное отношение, и (2) в круглых скобках *sc-идентификатор* одного из элементов первого домена указанного бинарного ориентированного отношения, либо (1) *sc-идентификатор*, обозначающий бинарное неориентированное отношение и (2) в круглых скобках *sc-идентификатор* одного из элементов области определения указанного бинарного неориентированного отношения. sc-выражение, построенное путём указания некоторого бинарного отношения (обычно функционального) и одного из его аргументов (в круглых скобках).



Сложный sc-идентификатор

- *sc-выражение, построенное на основе алгебраической операции* – *sc-выражение*, ограниченное круглыми скобками и построенное путем указания *sc-идентификаторов*, разделенных знаком алгебраической операции.;
- *sc-выражение, идентифицирующее sc-коннектор* – *sc-выражение*, ограниченное круглыми скобками и идентифицирующее *sc-коннектор*, инцидентный двум указанным *sc-элементам* и имеющий тип, задаваемый путем изображения соответствующего *sc.s-коннектора*. Для упрощения восприятия и обработки *sc-выражений, идентифицирующих sc-коннектор* вводится следующее ограничение: первым и третьим компонентом такого *sc-выражения* может быть только *простой sc-идентификатор*. В рамках *sc.s-текстов* внутри *sc-выражений, идентифицирующих sc-коннектор* допускается также использование *sc.s-модификаторов*.



Лекция 4

Внешние языки представления информации

Шункевич Д.В.

Белорусский государственный университет информатики и радиоэлектроники



Содержание лекции

Графический язык внешнего представления конструкций SC-кода – SCg-код. Синтаксис и денотационная семантика SCg-кода. Линейный язык внешнего представления конструкций SC-кода – SCs-код. Синтаксис и денотационная семантика SCs-кода. Структурированный гипертекстовый язык внешнего представления конструкций SC-кода – SCn-код.



Понятие внешних языков

внешний язык

\coloneqq [язык представления информации, которую видит и понимает человек]

\coloneqq [язык обмена сообщениями]

\Rightarrow *примеры**:

- { • *SCn-код*
- *SCs-код*
- *SCg-код*
- }

внутренний язык

\coloneqq [язык представления информации в памяти кибернетической системы]



SCg-код

SCg-код

SCg-код

\coloneqq [Semantic Code graphical]

\coloneqq [Язык визуального (графического) представления баз знаний ostis-систем]

\in *графовый язык*

\Rightarrow *пояснение**:

[SCg-код представляет собой способ визуализации *sc-текстов* (информационных конструкций SC-кода) в виде рисунков этих абстрактных конструкций.

Подчеркнем, что абстрактная *графовая структура* и её рисунок (графическое изображение) – это не одно и то же даже если они изоморфны друг другу.]

\Rightarrow *пояснение**:

[SCg-код рассматривается нами как объединение *Ядра SCg-кода*, обеспечивающего изоморфное графическое изображение любого *sc-текста*, а также нескольких направлений расширения этого ядра, обеспечивающих повышение компактности и "читабельности" текстов SCg-кода (*sc.g-текстов*).]

Основная цель *SCg-кода* – иметь четкие синтаксические графические признаки изображения *sc.g-элементов*, позволяющие легко выделить и различать такие классы *sc.g-элементов*, как:

- *sc.g-константы* (знаки константных сущностей) и *sc.g-переменные* (изображения переменных, значениями которых являются соответствующие *sc-элементы*);
- *sc.g-переменные*, значениями которых являются *sc-константы*, и *sc.g-переменные*, значениями которых являются *sc-переменные*;
- знаки постоянных (стабильных) сущностей и знаки временных (нестабильных, временно существующих, ситуативных) сущностей;

- *sc.g-коннекторы* (знаки бинарных связей) и *sc.g-элементы*, не являющиеся *sc.g-коннекторами*;
- неориентированные *sc.g-коннекторы* (*sc.g-ребра*) и ориентированные (*sc.g-дуги*);
- *sc.g-дуги принадлежности* и *sc.g-дуги*, не являющиеся таковыми;
- *sc.g-дуги позитивной принадлежности*, негативной принадлежности и нечеткой принадлежности.

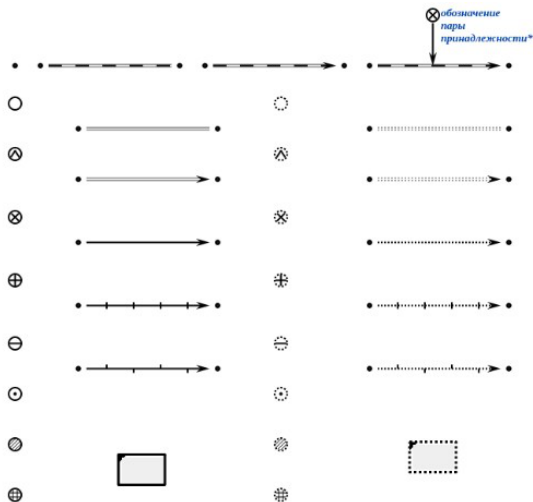


Синтаксис SCg-кода

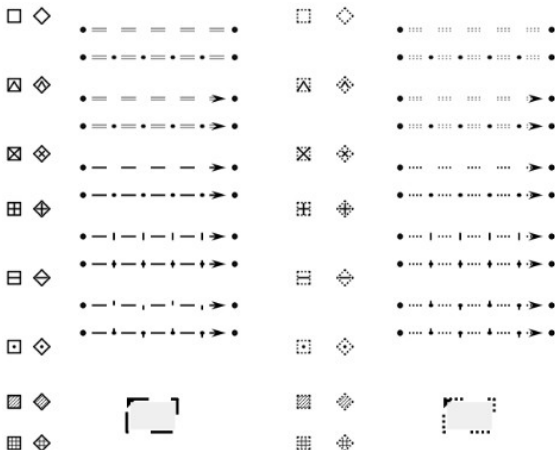
Алфавит Ядра SCg-кода

\coloneqq [Алфавит sc.g-элементов, графически изображающих sc-элементы]

Синтаксис SCg-кода



Синтаксис SCg-кода





Синтаксис SCg-кода

Первая группа (верхняя строка) включает в себя *sc.g-элементы*, для которых константность и постоянство обозначаемых ими сущностей требует дополнительно уточнения. Остальные четыре группы *sc.g-элементов* аналогичны друг другу и включают в себя соответственно:

- знаки **константных постоянных сущностей**;
- знаки **константных временных сущностей**;
- изображения *sc-переменных*, значениями которых или значениями значений которых (в случае, если значениями переменных являются переменные) являются знаки константных постоянных сущностей;
- изображения *sc-переменных*, значениями которых или значениями значений которых (в случае, если значениями переменных являются переменные) являются знаки константных временных сущностей.



Синтаксис SCg-кода

Особое место в SCg-коде занимает изображение *sc-элементов*, являющихся *обозначениями пар принадлежности**, путём явного использования этого семантически выделяемого класса *sc-элементов*. Данный *sc-элемент* используется тогда, когда нам необходимо изобразить *sc-дугу*, о которой известно, что она является обозначением *пары принадлежности**, но неизвестно о какой принадлежности идет речь – о константной или переменной, о постоянной или временной, о позитивной, негативной или нечеткой.



Синтаксис SCg-кода

Кроме *sc.g-элементов*, перечисленных в таблице “**SCg-текст. Алфавит SCg-кода**”, в состав *Алфавита SCg-кода* входят также следующие *sc.g-элементы*:

- внешние идентификаторы *sc-элементов*, идентичные (приписываемые) соответствующим *sc.g-элементам*;
- *sc.g-контура*, каждый из которых является знаком некоторого *sc-текста* (структуры, состоящей из *sc-элементов*);
- *sc.g-рамки* увеличенного размера являются ограничителями изображения различных файлов, хранимых в памяти *ostis-системы*;
- *sc.g-шины*, являющиеся обозначениями тех же сущностей, что и инцидентные им *sc.g-элементы*.



Синтаксис SCg-кода

Заметим также, что, кроме всех перечисленных элементов *Алфавита SCg-кода*, каждый из которых имеет вполне определенную денотационную семантику, для формализации синтаксиса SCg-кода необходимо ввести целый ряд более "мелких" синтаксических объектов, например:

- точек инцидентности *sc.g-коннекторов* с *sc.g-узлами*, с другими *sc.g-коннекторами*, с *sc.g-контурами*, с *sc.g-рамками*;
- точек инцидентности *sc.g-шин*;
- точек излома линейных *sc.g-элементов* (*sc.g-коннекторов*, *sc.g-контуров*, *sc.g-рамок*, *sc.g-шин*).

Денотационная семантика SCg-кода

В SCg-коде выделяются *Ядро SCg-кода* и его расширения. *Алфавит Ядра SCg-кода* – алфавит *sc.g-элементов*, графически изображаемых *sc-элементы*. *Алфавит Ядра SCg-кода* взаимно однозначно соответствует *Алфавиту SC-кода*. Денотационная семантика *Ядра SCg-кода* соответствует денотационной семантике SC-кода.

<i>Алфавит SC-кода</i>	<i>Алфавит Ядра SCg-кода</i>	<i>Изображение sc.g-элемента</i>
<i>sc-узел общего вида</i>	<i>sc.g-узел общего вида</i>	•
<i>sc-ребро общего вида</i>	<i>sc.g-ребро общего вида</i>	—
<i>sc-дуга общего вида</i>	<i>sc.g-дуга общего вида</i>	→
<i>базовая sc-дуга</i>	<i>базовая sc.g-дуга</i>	→
<i>внутренний файл ostis-системы</i>	<i>sc.g-узел с содержимым</i>	□

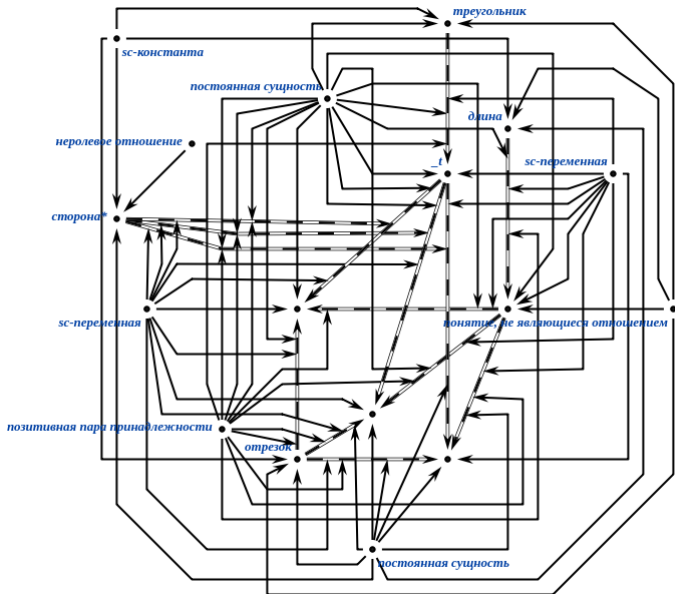


Денотационная семантика SCg-кода

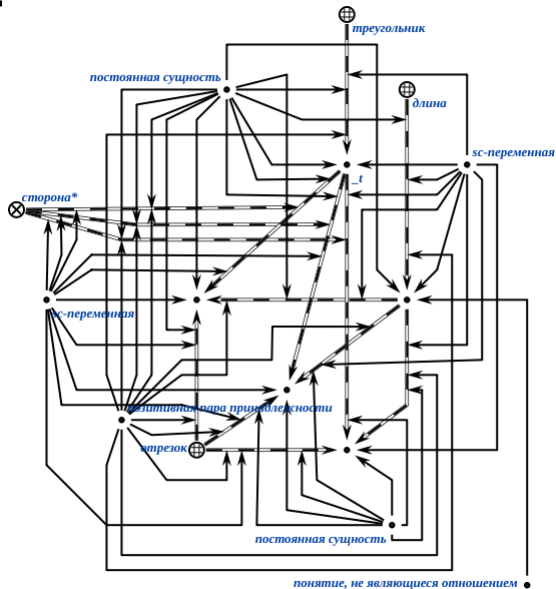
sc-узел с содержимым

- \coloneqq [*sc-узел*, имеющий содержимое]
- \coloneqq [*sc-узел*, являющийся знаком внутреннего файла ostis-системы]
- \coloneqq [*sc-знак* внутреннего файла ostis-системы]
- \coloneqq [*sc.g-рамка*, ограничивающая изображение (представление) внутреннего файла ostis-системы, обозначаемого этой *sc.g-рамкой*]

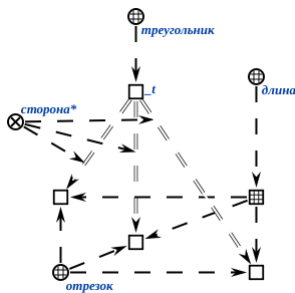
SCg-код: Примеры трансформации (1)



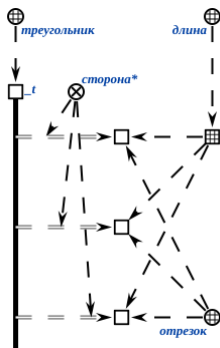
SCg-код: Примеры трансформации (1)



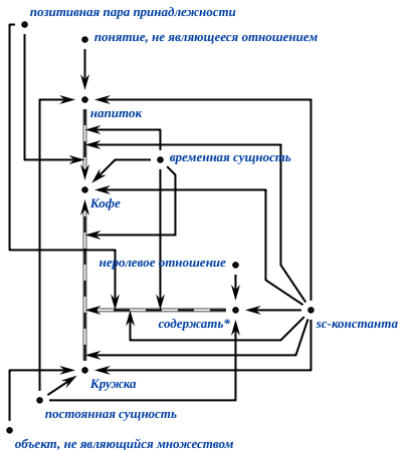
SCg-код: Примеры трансформации (1)



SCg-код: Примеры трансформации (1)

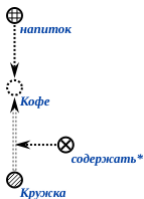


SCg-код: Примеры трансформации (2)



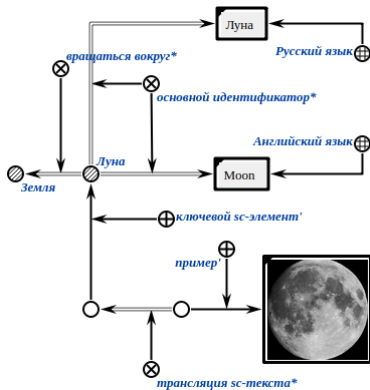


SCg-код: Примеры трансформации (2)





SCg-код: Примеры трансформации (3)







SCs-код

SCs-код

SCs-код

:= [Semantic Code string]

:= [Язык линейного представления знаний ostis-систем]

:= [Множество всевозможных текстов SCs-кода]

:= [Язык внешнего линейного представления конструкций внутреннего языка ostis-систем]

⇒ *пояснение**:

[SCs-код представляет собой множество линейных текстов (*sc.s-текстов*), каждый из которых состоит из предложений (*sc.s-предложений*), разделенных друг от друга *двойной точкой с запятой* (разделителем *sc.s-предложений*). При этом *sc.s-предложение* представляет собой последовательность *sc-идентификаторов*, являющихся именами описываемых сущностей и разделяемых между собой различными *sc.s-разделителями* и *sc.s-ограничителями*.]



Синтаксис SCs-кода

Алфавит SCs-кода

- := [Алфавит символов SCs-кода]
- := [множество символов SCs-кода]
- := [символ, используемый в текстах SCs-кода]
- := [Язык внешнего линейного представления конструкций внутреннего языка ostis-систем]
- ⇒ *пояснение**:

[*Алфавит SCs-кода* строится на основе современных общепринятых наборов символов, что позволяет упростить разработку средств для работы с *sc.s-текстами* с использованием современных технологий.

В состав *sc.s-текстов*, как и в состав текстов любых других языков, являющихся вариантами внешнего отображения текстов SC-кода, могут входить различные файлы, в том числе естественно-языковые или даже файлы, содержащие другие *sc.s-тексты*. В общем случае в таких файлах могут использоваться самые разные символы, в связи с чем будем считать, что в *Алфавит SCs-кода* эти символы не включаются.]



Синтаксис SCs-кода

Для упрощения процесса разработки исходных текстов баз знаний с использованием SCs-кода и создания соответствующих средств вводятся два алфавита символов:

- **Базовый алфавит символов**, используемых в *sc.s-коннекторах* включает только символы, входящие в переносимый набор символов и имеющиеся на стандартной современной клавиатуре. Таким образом, для разработки исходных текстов баз знаний, использующих только *Базовый алфавит* символов, используемых в *sc.s-коннекторах* достаточно обычного текстового редактора.
- **Расширенный алфавит символов**, используемых в *sc.s-коннекторах* включает также дополнительные символы, которые позволяют сделать *sc.s-тексты* (и *sc.n-тексты*) более читабельными и наглядными. Для визуализации и разработки *sc.s-текстов* с использованием расширенного алфавита требуется наличие специализированных средств.



Синтаксис SCs-кода

Как в *Базовом*, так и в *Расширенном Алфавитах sc.s-коннекторов* используются следующие общие признаки, характеризующие тип изображаемого *sc-коннектора*:

- знак подчеркивания как признак изображений переменных *sc-коннекторов* (один знак подчеркивания для *sc-коннекторов*, являющихся первичными *sc-переменными*, два знака подчеркивания для *sc-коннекторов*, являющихся вторичными *sc-переменными* (*sc-метапеременными*));
- вертикальная черта “|” как признак изображений *негативных sc-дуг принадлежности*;
- косая черта “/” как признак изображений *нечетких sc-дуг принадлежности*;
- тильда “~” как признак изображений *временных sc-дуг принадлежности*.



Синтаксис SCs-кода

Алфавит символов, используемых в sc.s-разделителях

- Э пробел; точка с запятой; двоеточие; круглый маркер; знак равенства
- ⊃ Алфавит символов, используемых в sc.s-разделителях, изображающих связь инцидентности sc-элементов

Э $![>]!$; $![<]!$; $![\setminus]!$; $![-]!$;

- ⊃ Алфавит символов, используемых в sc.s-коннекторах

⊃ Расширенный алфавит символов, используемых в sc.s-коннекторах
:= [Расширенный алфавит sc.s-коннекторов]

Э $![\in]!$; $![\ni]!$; $![\notin]!$; $![\not\subseteq]!$; $![\subseteq]!$; $![\supseteq]!$; $![\subset]!$; $![\supset]!$; $![\leq]!$; $![\geq]!$;
 $![\Leftarrow]!$; $![\Rightarrow]!$; $![\Leftrightarrow]!$; $![\leftarrow]!$; $![\rightarrow]!$; $![\leftrightarrow]!$

⊃ Базовый алфавит символов, используемых в sc.s-коннекторах
:= [Базовый алфавит sc.s-коннекторов]

Э $![\sim]!$; знак подчеркивания; знак равенства; $![>]!$; $![<]!$;
двоеточие; $![-]!$; $![\setminus]!$; $![/]!$;



Синтаксис SCs-кода






Алфавит символов, используемых в sc.s-ограничителях

⊃ `![(]!; ![()]!; ![*]!;`

Алфавит символов, используемых в неоднозначных sc.s-изображениях sc-узлов

⊃ `![{ }!; ![}]!; ![!]!; ![-]!; ![[]!; ![]]!;`

Денотационная семантика SCs-кода

Класс <i>sc</i> -элементов	Изображение <i>sc.g</i> - коннектора	Изображение <i>sc</i> -коннектора в Расширенном алфавите		Изображение <i>sc</i> -коннектора в Базовом алфавите	
константная постоянная позитивная <i>sc</i> -дуга принадлежности		Э	Є	->	<-
константная постоянная негативная <i>sc</i> -дуга принадлежности		Э̇	Є̇	- >	< -
константная постоянная нечеткая <i>sc</i> -дуга принадлежности		/Э	Є/	-/>	</-
константная временная позитивная <i>sc</i> -дуга принадлежности		~Э	Є~	~>	<~
константная временная негативная <i>sc</i> -дуга принадлежности		~Э̇	Є̇~	~ >	< ~

Денотационная семантика SCs-кода

константная временная нечеткая sc-дуга принадлежности	•▶ •	\sim/\exists	\in/\sim	$\sim/>>$	$</\sim$
переменная постоянная позитивная sc-дуга принадлежности	• — — — —▶ •	$_ \exists$	$_ \in$	$_ >$	$< _$
переменная постоянная негативная sc-дуга принадлежности	• — — — —▶ •	$_ \exists$	$_ \notin$	$_ >$	$< _$
переменная постоянная нечеткая sc-дуга принадлежности	• — ' — , — ' —▶ •	$_ / \exists$	$_ \in$	$_ / >$	$< / _$
переменная временная позитивная sc-дуга принадлежности	•▶ •	$_ \sim \exists$	$_ \sim \in$	$_ \sim >$	$< \sim _$
переменная временная негативная sc-дуга принадлежности	•▶ •	$_ \sim \exists$	$_ \sim \notin$	$_ \sim >$	$< \sim _$

Денотационная семантика SCs-кода

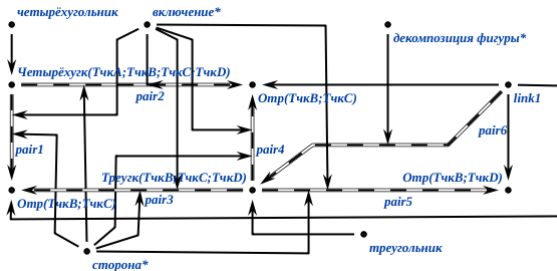
переменная временная нечеткая sc-дуга принадлежности	• * , * , ➤ •	\sim/\exists	\in/\sim	\sim/ \rightarrow	$</\sim$
метаварiable постоянная позитивная sc-дуга принадлежности	• - - - - - ➤ •	$_ \exists$	$\in _$	$_ \rightarrow$	$< _$
метаварiable постоянная негативная sc-дуга принадлежности	• - - - - - ➤ •	$_ \exists$	$\notin _$	$_ \rightarrow$	$< _$
метаварiable постоянная нечеткая sc-дуга принадлежности	• - - - - - , - - - - - , ➤ •	$_/\exists$	$\in/_$	$_ \sim/ \rightarrow$	$</\sim _$
метаварiable временная позитивная sc-дуга принадлежности	• * * * * ➤ •	$_ \sim \exists$	$\in \sim _$	$_ \sim \rightarrow$	$< \sim _$
метаварiable временная негативная sc-дуга принадлежности	• * * * * ➤ •	$_ \sim \exists$	$\notin \sim _$	$_ \sim \rightarrow$	$< \sim _$
метаварiable временная нечеткая sc-дуга принадлежности	• * * * * ➤ •	$_ \sim/\exists$	$\in/\sim _$	$_ \sim/ \rightarrow$	$</\sim _$



SCs-код: Примеры трансформации

```
[включение*  $\ni$  pair1;;  
  включение*  $\ni$  pair2;;  
  включение*  $\ni$  pair3;;  
  включение*  $\ni$  pair4;;  
  включение*  $\ni$  pair5;;  
  сторона*  $\ni$  pair1;;  
  сторона*  $\ni$  pair2;;  
  сторона*  $\ni$  pair3;;  
  сторона*  $\ni$  pair4;;  
  сторона*  $\ni$  pair5;;  
  Четырехугк(ТчкА;ТчкВ;ТчкС;ТчкD)  $\vdash$  pair1  $\triangleright$  | Отр(ТчкВ;ТчкС);;  
  Четырехугк(ТчкА;ТчкВ;ТчкС;ТчкD)  $\vdash$  pair2  $\triangleright$  | Отр(ТчкС;ТчкD);;  
  Треугк(ТчкВ;ТчкС;ТчкD)  $\vdash$  pair3  $\triangleright$  | Отр(ТчкВ;ТчкС);;  
  Треугк(ТчкВ;ТчкС;ТчкD)  $\vdash$  pair4  $\triangleright$  | Отр(ТчкС;ТчкD);;  
  Треугк(ТчкВ;ТчкС;ТчкD)  $\vdash$  pair5  $\triangleright$  | Отр(ТчкВ;ТчкD);;  
  четырехугольник  $\ni$  Четырехугк(ТчкА;ТчкВ;ТчкС;ТчкD);;  
  треугольник  $\ni$  Треугк(ТчкВ;ТчкС;ТчкD);;  
  link1  $\vdash$  pair6  $\triangleright$  | Треугк(ТчкВ;ТчкС;ТчкD);;  
  декомпозиция фигуры*  $\ni$  pair6;;  
  link1  $\ni$  Отр(ТчкВ;ТчкС);;  
  link1  $\ni$  Отр(ТчкС;ТчкD);;  
  link1  $\ni$  Отр(ТчкВ;ТчкD);;]
```

SCs-код: Примеры трансформации





SCs-код: Примеры трансформации

[*сторона** \ni (*Четырехуголк*(*ТчкаА*; *ТчкаВ*; *ТчкаС*; *ТчкаD*) \Rightarrow *Отр*(*ТчкаВ*; *ТчкаС*));;
*сторона** \ni (*Четырехуголк*(*ТчкаА*; *ТчкаВ*; *ТчкаС*; *ТчкаD*) \supseteq *Отр*(*ТчкаС*; *ТчкаD*));;
*сторона** \ni (*Треуголк*(*ТчкаВ*; *ТчкаС*; *ТчкаD*) \supseteq *Отр*(*ТчкаВ*; *ТчкаС*));;
*сторона** \ni (*Треуголк*(*ТчкаВ*; *ТчкаС*; *ТчкаD*) \supseteq *Отр*(*ТчкаС*; *ТчкаD*));;
*сторона** \ni (*Треуголк*(*ТчкаВ*; *ТчкаС*; *ТчкаD*) \supseteq *Отр*(*ТчкаВ*; *ТчкаD*));;
Четырехуголк(*ТчкаА*; *ТчкаВ*; *ТчкаС*; *ТчкаD*) \supseteq *Отр*(*ТчкаВ*; *ТчкаС*));;
Четырехуголк(*ТчкаА*; *ТчкаВ*; *ТчкаС*; *ТчкаD*) \supseteq *Отр*(*ТчкаС*; *ТчкаD*));;
Треуголк(*ТчкаВ*; *ТчкаС*; *ТчкаD*) \supseteq *Отр*(*ТчкаВ*; *ТчкаС*));;
Треуголк(*ТчкаВ*; *ТчкаС*; *ТчкаD*) \supseteq *Отр*(*ТчкаС*; *ТчкаD*));;
Треуголк(*ТчкаВ*; *ТчкаС*; *ТчкаD*) \supseteq *Отр*(*ТчкаВ*; *ТчкаD*));;
четырёхугольник \ni *Четырехуголк*(*ТчкаА*; *ТчкаВ*; *ТчкаС*; *ТчкаD*));;
треугольник \ni *Треуголк*(*ТчкаВ*; *ТчкаС*; *ТчкаD*));;
*декомпозиция фигуры** \ni (*link1* \Rightarrow *Треуголк*(*ТчкаВ*; *ТчкаС*; *ТчкаD*));;
link1 \ni *Отр*(*ТчкаВ*; *ТчкаС*));;
link1 \ni *Отр*(*ТчкаС*; *ТчкаD*));;
link1 \ni *Отр*(*ТчкаВ*; *ТчкаD*));]



SCs-код: Примеры трансформации

[*Четырехугол*к(*ТчкА*; *ТчкВ*; *ТчкС*; *ТчкD*) \supseteq *сторона**: *Отр*(*ТчкВ*; *ТчкС*);;
*Четырехугол*к(*ТчкА*; *ТчкВ*; *ТчкС*; *ТчкD*) \supseteq *сторона**: *Отр*(*ТчкС*; *ТчкD*);;
*Треугол*к(*ТчкВ*; *ТчкС*; *ТчкD*) \supseteq *сторона**: *Отр*(*ТчкВ*; *ТчкС*);;
*Треугол*к(*ТчкВ*; *ТчкС*; *ТчкD*) \supseteq *сторона**: *Отр*(*ТчкС*; *ТчкD*);;
*Треугол*к(*ТчкВ*; *ТчкС*; *ТчкD*) \supseteq *сторона**: *Отр*(*ТчкВ*; *ТчкD*);;
*четырёхугольн*ик \ni *Четырехугол*к(*ТчкА*; *ТчкВ*; *ТчкС*; *ТчкD*);;
*треугольн*ик \ni *Треугол*к(*ТчкВ*; *ТчкС*; *ТчкD*);;
link1 \Rightarrow *декомпозиция фигуры**: *Треугол*к(*ТчкВ*; *ТчкС*; *ТчкD*);;
link1 \ni *Отр*(*ТчкВ*; *ТчкС*);;
link1 \ni *Отр*(*ТчкС*; *ТчкD*);;
link1 \ni *Отр*(*ТчкВ*; *ТчкD*);;]



SCs-код: Примеры трансформации

[*Четырехугол*(ТчкА;ТчкВ;ТчкС;ТчкD) \supseteq сторона*: *Отр*(ТчкВ;ТчкС); *Отр*(ТчкС;ТчкD);;
Треугол(ТчкВ;ТчкС;ТчкD) \supseteq сторона*: *Отр*(ТчкВ;ТчкС); *Отр*(ТчкС;ТчкD); *Отр*(ТчкВ;ТчкD);;
четыреугольник \ni *Четырехугол*(ТчкА;ТчкВ;ТчкС;ТчкD);;
треугольник \ni *Треугол*(ТчкВ;ТчкС;ТчкD);;
link1 \Rightarrow декомпозиция фигуры*: *Треугол*(ТчкВ;ТчкС;ТчкD);;
link1 \ni *Отр*(ТчкВ;ТчкС); *Отр*(ТчкС;ТчкD); *Отр*(ТчкВ;ТчкD);;]



SCs-код: Примеры трансформации

[*четыреугольник* \ni *Четыреуголк*(*ТчкА*; *ТчкВ*; *ТчкС*; *ТчкD*)
($\ast \supseteq$ *сторона* \ast : *Отр*(*ТчкВ*; *ТчкС*); *Отр*(*ТчкС*; *ТчкD*);,; \ast);;
треугольник \ni *Треуголк*(*ТчкВ*; *ТчкС*; *ТчкD*)
($\ast \supseteq$ *сторона* \ast : *Отр*(*ТчкВ*; *ТчкС*); *Отр*(*ТчкС*; *ТчкD*); *Отр*(*ТчкВ*; *ТчкD*);,; \ast);;
Треуголк(*ТчкВ*; *ТчкС*; *ТчкD*) \Leftarrow *декомпозиция фигуры* \ast : *link1* ($\ast \ni$ *Отр*(*ТчкВ*; *ТчкС*); *Отр*(*ТчкС*; *ТчкD*);
Отр(*ТчкВ*; *ТчкD*);,; \ast);,;]



SCn-код



SCn-код

SCn-код

\coloneqq [Semantic Code natural]

\coloneqq [Язык структурированного представления знаний ostis-систем]

\coloneqq [Язык внешнего форматированного представления конструкций внутреннего языка ostis-систем]

\Rightarrow *пояснение**:

[*SCn-код* является языком структурированного внешнего представления текстов *SC-кода* и представляет собой синтаксическое расширение *SCs-кода*, направленное на повышение наглядности и компактности текстов *SCs-кода*.

SCn-код позволяет перейти от линейных текстов *SCs-кода* к форматированным и фактически двумерным текстам, в которых появляется декомпозиция исходного линейного текста *SCs-кода* на строки, размещенные "по вертикали". При этом начало всех строчек текста фиксировано и определяется известным и ограниченным набором правил, что дает возможность использовать это при форматировании *sc.n-текста*.]



SCn-код

Важной особенностью *SCn-кода* является "двухмерный" характер его текстов. Это проявляется в том, что для каждого фрагмента текста *SCn-кода* важное значение имеет величина отступа от левого края строчки.

В тексте *SCn-кода* в отличие от текста *SCs-кода* для каждого фрагмента текста важное значение имеет не только то, как этот фрагмент связан с другими фрагментами "по горизонтали" (какой фрагмент находится левее и какой правее на одной и той же *строчке*), но и то, как он связан с другими фрагментами "по вертикали" (какой фрагмент находится выше на предыдущей строчке и какой находится ниже на следующей строчке). Так, например, если в тексте *SCn-кода* некоторый *sc-идентификатор* (внешний идентификатор *sc-элемента*) размещен сразу после вертикальной табуляционной линии и точно под ним размещен некоторый *sc.s-коннектор*, то это означает, что указанный *sc-элемент* инцидентен *sc-коннектору*, изображенному указанным *sc.s-коннектором*.



Синтаксис SСn-кода

Алфавит символов *SСs-кода* является также алфавитом символов и *SСn-кода*, т.е. *алфавиты** этих языков совпадают. *SСn-код* – язык, каждый *текст* которого задается:

- множеством входящих в него *символов*;
- отношением порядка (последовательности) *символов* по "горизонтали";
- отношением порядка(последовательности) *символов* по "вертикали".



Синтаксис SCn-кода

sc.n-текст

- :=** [текст SCn-кода]
- :=** [последовательность предложений SCn-кода]
- :=** [последовательность предложений SCn-кода, каждое из которых не является частью какого-либо другого предложения из этой последовательности]

страница sc.n-текста

- :=** [страница, на которой размещается sc.n-текст]

линия разметки sc.n-текста

- :=** [табуляционная линия sc.n-текста]
- :=** [вертикальная линия разметки sc.n-текста]
- :=** [вертикальная линия, используемая для упрощения восприятия sc.n-текстов и показывающая уровень отступа для компонентов sc.n-предложений]



Синтаксис SСn-кода

Все компоненты *sc.s-текстов* используются также и в *sc.n-текстах*:

- *sc-идентификаторы*, *sc.s-коннекторы*, модификаторы *sc.s-коннекторов* с соответствующими разделителями (двоеточиями);
- разделители, используемые в *sc-выражениях*, обозначающих *sc-множества*, заданные перечислением элементов с соответствующими разделителями (точкой с запятой или круглым маркером);
- круглые маркеры в перечислениях идентификаторов *sc-элементов*, связанных однотипными *sc-коннекторами* с однотипными модификаторами с заданным *sc-элементом*;
- разделители предложений (двойные точки с запятой) (опускаются при преобразовании *sc.s-предложений* в *sc.n-предложения*);
- ограничители присоединенных *sc.s-предложений* (опускаются при преобразовании *sc.s-предложений* в *sc.n-предложения*).



Синтаксис SСn-кода

В отличие от *sc.s-текстов* в *sc.n-текстах*:

- добавляются новые виды *sc-выражений* (а именно – *sc-выражений*, имеющих двухмерный характер);
- добавляется новый вид разделителей предложений – пустая строка;
- меняется размещение предложений с учетом двухмерного характера такого размещения.



Денотационная семантика SСn-кода

В отличие от *sc.s-текстов*: в *sc.n-текстах* *sc.s-коннектор* может быть инцидентен предшествующему *sc-идентификатору* (как простому, так и *sc-выражению*) не только "по горизонтали", но и "по вертикали". Для этого *sc.s-коннектор* размещается строго под предшествующим ему *sc-идентификатором*.

Кроме того "по вертикали" *sc-идентификатор* может быть инцидентен не одному, а нескольким *sc.s-коннекторам*, которые последовательно "по вертикали" размещаются под указанным *sc-идентификатором*. Это позволяет в рамках одного *sc.n-предложения* представлять произвольное число "ответвлений" от каждого *sc-идентификатора*, т.е. произвольное число *sc.s-коннекторов*, инцидентных этому *sc-идентификатору*.

Каждый *sc-идентификатор*, включая *sc-выражение*, должен размещаться сразу правее вертикальной разметочной линии, если под ним размещается *sc.s-коннектор*.

SCn-код: Операция преобразования sc.s-предложения в sc.n-предложение



$[Треугк(ТчкВ;ТчкС;ТчкD) \Rightarrow \text{сторона}^*: Отр(ТчкВ;ТчкС) (* \in \text{отрезок};;$
 $*);;]$

$Треугк(ТчкВ;ТчкС;ТчкD)$

$\Rightarrow \text{сторона}^*:$

$Отр(ТчкВ;ТчкС)$

$\in \text{отрезок}$

SCn-код: Операция присоединения sc.n-предложения



Треугк(ТчкВ;ТчкС;ТчкD)

\Rightarrow сторона*:

Отр(ТчкВ;ТчкС)

Отр(ТчкВ;ТчкС)

\in отрезок

Треугк(ТчкВ;ТчкС;ТчкD)

\Rightarrow сторона*:

Отр(ТчкВ;ТчкС)

\in отрезок



Лекция 5

Представление в базе знаний множеств и операций над ними

Шункевич Д.В.

Белорусский государственный университет информатики и радиоэлектроники



Содержание лекции

Типология множеств, рефлексивное множество. Ориентированное множество, декартово произведение множеств. Атрибуты, кортежи и классические кортежи. Мощность, бесконечные и конечные множества (без представления в базе знаний). Отношения над множествами, равенство. Операции над канторовскими множествами. Операции над мультимножествами. Булеан множества, его мощность. Представление в базе знаний.



Понятие множества

Под *множеством* понимается соединение в некое целое M определённых хорошо различимых предметов m нашего созерцания или нашего мышления (которые будут называться “элементами” множества M).

множество – мысленная сущность, которая связывает одну или несколько сущностей в целое.

Более формально *множество* – это абстрактный математический объект, состоящий из элементов. Связь множеств с их элементами задается бинарным ориентированным отношением *принадлежность**.

В Теории множеств *множество* считается неопределяемым понятием, его можно только пояснить.



Задание множеств

множество может быть полностью задано следующими тремя способами:

- путем перечисления (явного указания) всех его элементов (очевидно, что таким способом можно задать только конечное множество)

\Rightarrow *изображение**:

$$A = \{1, 3, 7, 9\}$$

$$D = \{0, 1\}$$

$$K = \{A, H, L, Q, W, Z\}$$



- с помощью определяющего высказывания, содержащего описание общего характеристического свойства, которым обладают все те и только те объекты, которые являются элементами (т.е. принадлежат) задаваемого множества.

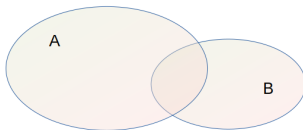
\Rightarrow изображение*:

$$K = \{x : -10 \leq x \leq 15\}$$
$$W = \{x : 0 \leq x \leq 12, x \in N\}$$

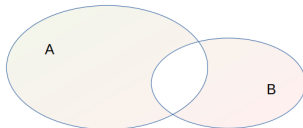


- с помощью теоретико-множественных операций, позволяющих однозначно задавать новые множества на основе уже заданных (это операции объединения, пересечения, разности множеств и др.)

\Rightarrow изображение*:



\Rightarrow изображение*:





Принадлежность

принадлежность*

$:=$ [принадлежность элемента множеству*]

\in *бинарное отношение*

\in *ориентированное отношение*

принадлежность* – это бинарное ориентированное отношение, каждая связка которого связывает множество с одним из его элементов.

Элементами отношения ***принадлежность**** в *SC-коде* по умолчанию являются ***базовые sc-дуги*** (позитивные постоянные константные sc-дуги принадлежности).



Классификация множеств

множество

⇒ *разбиение**:

- *конечное множество*
- *бесконечное множество*

⇒ *разбиение**:

- *множество без кратных элементов*
- *мультимножество*

⇒ *разбиение**:

- *кортеж*
- *неориентированное множество*



Конечность множеств

конечное множество

$:=$ [множество с конечным числом элементов]

конечное множество – это множество, количество элементов которого конечно, т.е. существует неотрицательное целое число k , равное количеству элементов этого множества.

бесконечное множество

$:=$ [множество с бесконечным числом элементов]

\Rightarrow разбиение*:

- { • *счетное множество*
- *несчетное множество*
- }

бесконечное множество – это множество, в котором для любого натурального числа n найдётся конечное подмножество из n элементов.



Счетность множеств

счетное множество – это *бесконечное множество*, для которого существует *взаимно однозначное соответствие* с натуральным рядом чисел.

несчетное множество

$:=$ [континуальное множество]

несчетное множество – это *бесконечное множество*, элементы которого невозможно пронумеровать натуральными числами.



Кратность множеств

множество без кратных элементов

$:=$ [классическое множество]

$:=$ [множество, состоящее из разных элементов]

\Rightarrow *пояснение**:

[*множество без кратных элементов* – это *множество*, для каждого элемента которого существует только одна пара принадлежности, выходящая из знака этого множества в указанный элемент.]



мультимножество

- \coloneqq [множество, имеющее кратные вхождения хотя бы одного элемента]
- \coloneqq [множество, по крайней мере один элемент которого входит в его состав многократно]
- \Rightarrow *пояснение**:
[*мультимножество* – это *множество*, для которого существует хотя бы одна кратная пара принадлежности, выходящая из знака этого множества.]



кратность принадлежности

$:=$ [кратность принадлежности элемента]

$:=$ [кратность вхождения элемента во множество]

\in *параметр*

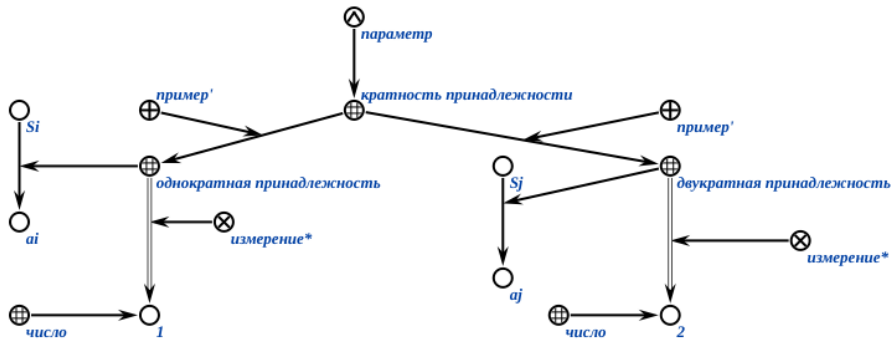
\Rightarrow *пояснение**:

[кратность принадлежности – параметр, значением которого являются числовые величины, показывающие сколько раз входит тот или иной элемент в рассматриваемое множество. Элементами данного параметра являются классы *позитивных sc-дуг принадлежности*, связывающих данное множество с элементом, кратность вхождения которого в данное множество мы хотим задать.

Таким образом, кратное вхождение элемента в мультимножество может быть задано как явным указанием *позитивных sc-дуг принадлежности* этого элемента данному множеству, так и "склеиванием" этих дуг в одну и включением ее в некоторый класс ***кратности принадлежности.***]



\Rightarrow описание примера*:





Нечеткое и четкое множество

нечеткое множество

⇒ *пояснение**:

[*нечеткое множество* – это *множество*, которое представляет собой совокупность элементов произвольной природы, относительно которых нельзя точно утверждать – обладают ли эти элементы некоторым характеристическим свойством, которое используется для задания этого нечеткого множества. Принадлежность элементов такому множеству указывается при помощи *нечетких позитивных sc-дуг принадлежности*.]

четкое множество

⇒ *пояснение**:

[*четкое множество* – это *множество*, принадлежность элементов которому достоверна и указывается при помощи *четких позитивных sc-дуг принадлежности*.]



Семейство множеств

семейство множеств

$:=$ [множество множеств]

\supset класс классов

\Rightarrow пояснение*:

[*семейство множеств* – это *множество*, элементами которого являются знаки множеств.]



Свойства отношений на множестве

нерефлексивное множество

⇒ *пояснение**:

[*нерефлексивное множеств* – это множество, знак которого не является элементом этого множества]

рефлексивное множество

⇒ *пояснение**:

[*рефлексивное множеств* – это множество, знак которого является элементом этого множества]



Мощность

мощность множества

- \coloneqq [кардинальное число]
- \coloneqq [общее число вхождений элементов в заданное множество]
- \coloneqq [класс эквивалентности, элементами которого являются знаки всех тех и только тех множеств, которые имеют одинаковую мощность]
- \coloneqq [класс эквивалентности, соответствующий отношению быть парой множеств, имеющих одинаковую мощность (равномощность множеств)]
- \coloneqq [величина мощности множеств]
- \coloneqq [трансфинитное число]
- \coloneqq [мощность по Кантору]
- \in *параметр*



Мощность

мощность множества

⇒ *пояснение**:

[*мощность множества* – это *параметр*, элементами которых являются *множества*, имеющие одинаковое количество элементов. Значением данного параметра является числовая величина, задающая количество элементов, входящих в данный класс множеств, т.е. по сути, количество *позитивных sc-дуг принадлежности*, выходящих из данного множества.]

множество

- ⊃ *пустое множество*
- ⊃ *синглетон*
- ⊃ *пара*
- ⊃ *тройка*



Пустое множество

пустое множество

\in *мощность множества*

\Rightarrow *пояснение**:

[*пустое множество* – это *множество*, которому не принадлежит ни один элемент.]



Синглетон

синглетон

\in *мощность множества*

$:=$ [множество мощности 1]

$:=$ [одноэлементное множество]

$:=$ [одномощное множество]

$:=$ [множество, мощность которого равна 1]

$:=$ [множество, имеющее мощность равную единице]

$:=$ [синглетон из sc-элемента]

$:=$ [sc-синглеон]

\subset *конечное множество*

\Rightarrow *пояснение**:

[**синглетон** – это множество, состоящее из одного элемента.]



Пара

пара

\in *мощность множества*

$:=$ [множество мощности два]

$:=$ [двухэлементное множество]

$:=$ [двумощное множество]

\subset *конечное множество*

\Rightarrow *пояснение**:

[*пара* – это *множество*, состоящее из двух элементов.]



Тройка

тройка

\in *мощность множества*

$:=$ [тройка]

$:=$ [множество мощности три]

\subset *конечное множество*

\Rightarrow *пояснение**:

[**тройка** – это множество, состоящее из трех элементов.]

кортеж

$:=$ [вектор]

\Rightarrow *пояснение**:

[*кортеж* – это множество, представляющее собой упорядоченный набор элементов, т.е. такое множество, порядок элементов в котором имеет значение. Пары принадлежности элементов *кортежу* могут дополнительно принадлежать каким-либо *ролевым отношениям*, при этом, в рамках каждого *кортежа* должен существовать хотя бы один элемент, роль которого дополнительно уточнена *ролевым отношением*.]



Отношения на множествах



Отношения на множествах

*включение**

$:=$ [быть подмножеством*]

\in *бинарное отношение*

\in *ориентированное отношение*

\in *транзитивное отношение*

\Rightarrow *область определения**:
множество

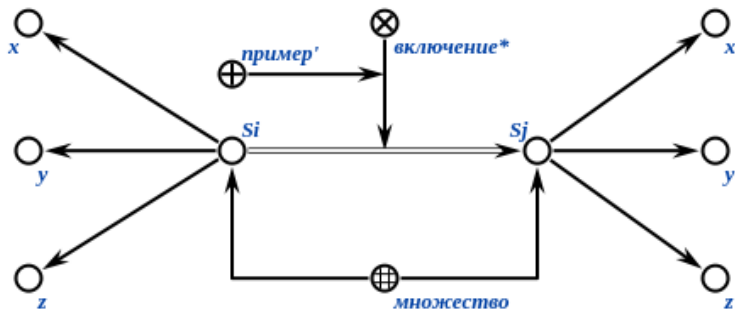
\supset *строгое включение**

\Rightarrow *определение**:

[***включение**** – это бинарное ориентированное отношение, каждая связка которого связывает два множества. Будем говорить, что *Множество S_i включает** в себя *Множество S_j* в том и только том случае, если каждый элемент *Множества S_j* является также и элементом *Множества S_i* .]



⇒ описание примера*:



⇒ пояснение*:

[Множество S_j включается во множество S_i .]



строгое включение*

$:=$ [строгое включение множеств*]

\subset включение*

\in бинарное отношение

\in ориентированное отношение

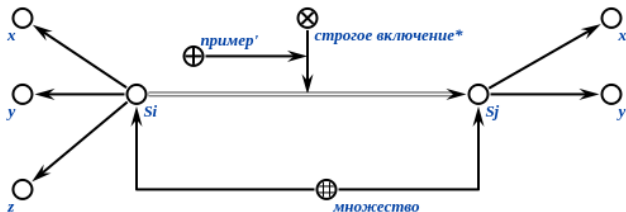
\Rightarrow область определения*:
множество

\Rightarrow определение*:

[строгое включение* – это бинарное ориентированное отношение, областью определения которого является семейство всевозможных множеств. Будем говорить, что *Множество S_i строго включает** в себя *Множество S_j* в том и только том случае, если каждый элемент *Множество S_j* является также и элементом *Множество S_i* , при этом существует хотя бы один элемент *Множество S_i* , не являющийся элементом *Множество S_j* .]



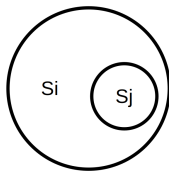
⇒ описание примера*:



⇒ пояснение*:

[Множество S_j строго включается во множество S_i .]

⇒ изображение*:





равенство множеств*

∈ *бинарное отношение*

∈ *неориентированное отношение*

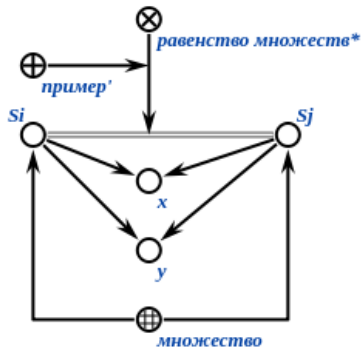
:= **[быть равными множествами*]**

⇒ *определение*:*

[равенство множеств* – бинарное неориентированное отношение, выражающее отношение равенства множеств. Любые два множества являются равными множествами тогда и только тогда, когда первое является включением второго и второе является включением первого.]



\Rightarrow описание примера*:



\Rightarrow пояснение*:

[Множество S_i равно множеству S_j .]

булеан*

$\text{булеан}^* := [\text{булеан множества}^*]$

$\text{булеан}^* := [\text{семейство всевозможных подмножеств заданного множества}^*]$

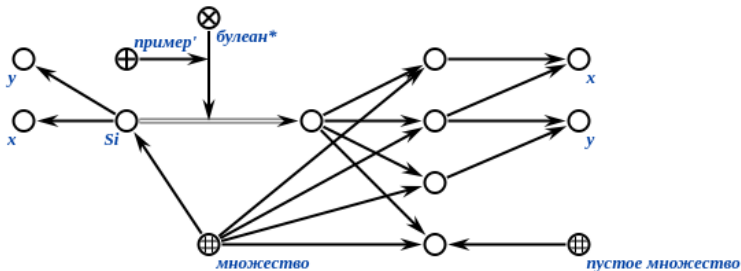
\in бинарное отношение

\in ориентированное отношение

\Rightarrow определение*:

[**булеан*** – это бинарное ориентированное отношение между множеством и некоторым семейством множеств, каждое из которых является подмножеством первого множества.]

\Rightarrow описание примера*:



*семейство подмножеств**

\coloneqq [семейство подмножеств заданного множества*]

\in бинарное отношение

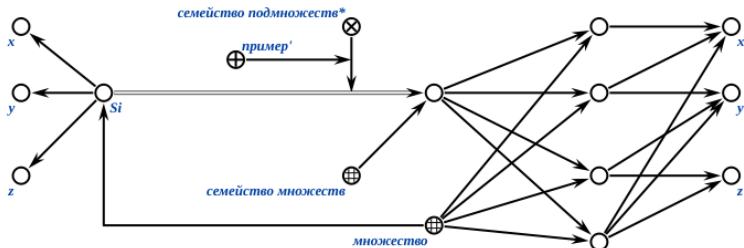
\in ориентированное отношение

\supset булеан*

\Rightarrow определение*:

[*семейство подмножеств** – это бинарное ориентированное отношение между множеством и некоторым семейством множеств, каждое из которых является подмножеством первого множества.]

\Rightarrow описание примера*:





Операции на множествах

объединение*

$:=$ [объединение множеств*]

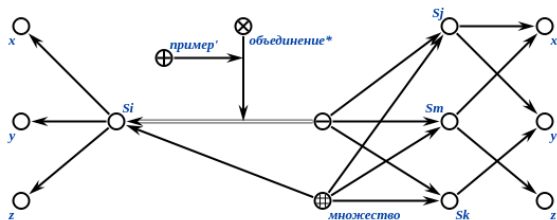
\in квазибинарное отношение

\in ориентированное отношение

\Rightarrow определение*:

[объединение* – это квазибинарное ориентированное отношение, областью определения которого является семейство всевозможных множеств. Будем говорить, что *Множество S_i* является объединением *Множество S_j* и *Множество S_k* тогда и только тогда, когда любой элемент *Множество S_i* является элементом или *Множество S_j* или *Множество S_k* .]

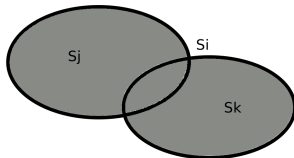
⇒ описание примера*:



⇒ пояснение*:

[Множество S_i является объединением множеств S_j , S_k и S_m .]

⇒ изображение*:





*разбиение**

- $:=$ [разбиение множества*]
- $:=$ [объединение попарно непересекающихся множеств*]
- $:=$ [декомпозиция множества*]
- \in *квазибинарное отношение*
- \in *ориентированное отношение*
- \in *отношение декомпозиции*



*разбиение**

⇒ *определение**:

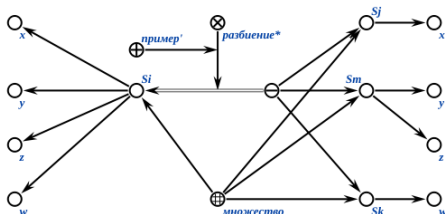
[разбиение* – это квазибинарное ориентированное отношение, областью определения которого является семейство всевозможных множеств. В результате разбиения множества получается множество попарно непересекающихся множеств, объединение которых есть исходное множество.

Семейство множеств $\{S_1, \dots, S_n\}$ является разбиением множества S_i в том и только том случае, если:

- семейство $\{S_1, \dots, S_n\}$ является семейством *попарно непересекающихся множеств*;
- семейство $\{S_1, \dots, S_n\}$ является покрытием множества S_i (множество S_i является *объединением* множеств, входящих в указанное выше семейство)

]

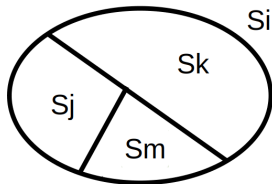
⇒ описание примера*:



⇒ пояснение*:

[Множество S_i разбивается на множества S_j , S_k и S_m .]

⇒ изображение*:





пересечение*

$:=$ [пересечение множеств*]

\in квазибинарное отношение

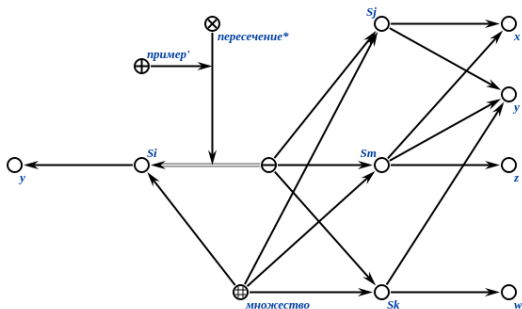
\in ориентированное отношение

\Rightarrow определение*:

[пересечение* – это операция над множествами, аргументами которой являются два или большее число множеств, а результатом является множество, элементами которого являются все те и только те сущности, которые одновременно принадлежат каждому множеству, которое входит в семейство аргументов этой операции.

Будем говорить, что *Множество S_i* является пересечением *Множество S_j* и *Множество S_k* тогда и только тогда, когда любой элемент *Множество S_i* является элементом *Множество S_j* и элементом *Множество S_k* .]

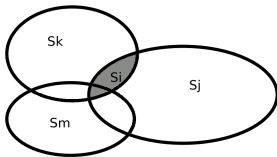
⇒ описание примера*:



⇒ пояснение*:

[Множество S_i является результатом пересечения множеств S_j , S_k и S_m .]

⇒ изображение*:





разность множеств*

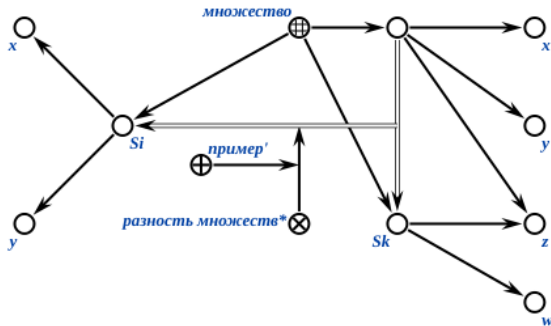
∈ *бинарное отношение*

∈ *ориентированное отношение*

⇒ *определение**:

[разность множеств* – это бинарное ориентированное отношение, связывающее между собой ориентированную пару, первым элементом которой является уменьшаемое множество, вторым – вычитаемое множество, и множество, являющееся результатом вычитания вычитаемого из уменьшаемого, в которое входят все элементы первого множества, не входящие во второе множество.]

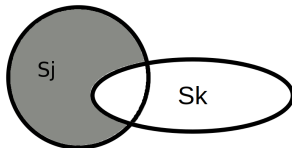
⇒ описание примера*:



⇒ пояснение*:

[Множество S_i является результатом разности множеств S_j и S_k .]

⇒ изображение*:





симметрическая разность множеств*

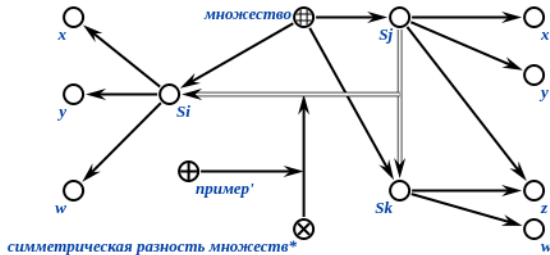
∈ бинарное отношение

∈ ориентированное отношение

⇒ определение*:

[симметрическая разность множеств* – это бинарное ориентированное отношение, связывающее между собой *пару* множеств и множество, являющееся результатом симметрической разности элементов указанной пары. Будем называть *Множество* S_i результатом симметрической разности *Множества* S_j и *Множества* S_k тогда и только тогда, когда любой элемент *Множества* S_i является или элементом *Множества* S_j или *Множества* S_k , но не является элементом обоих множеств.]

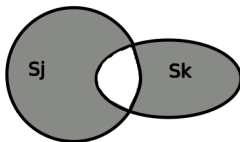
⇒ описание примера*:



⇒ пояснение*:

[Множество S_i является результатом симметрической разности множеств S_j и S_k .]

⇒ изображение*:





декартово произведение*

\coloneqq [декартово произведение множеств*]

\coloneqq [прямое произведение множеств*]

\in *бинарное отношение*

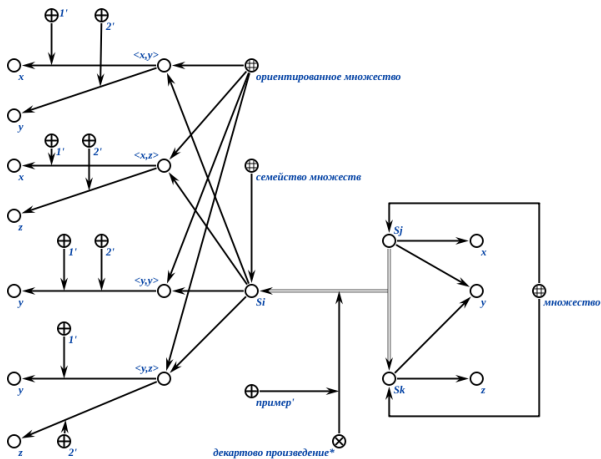
\in *ориентированное отношение*

\Rightarrow *определение*:*

[декартово произведение* – это бинарное ориентированное отношение между ориентированной парой множеств и множеством, элементами которого являются всевозможные упорядоченные пары, первыми элементами которых являются элементы первого из указанных множеств, вторыми – элементы второго из указанных множеств.]



⇒ описание примера*:



⇒ пояснение*:

[Множество S_i является результатом декартова произведения множеств S_j и S_k .]



Пересекающиеся множества

*пара пересекающихся множеств**

∈ бинарное отношение

∈ неориентированное отношение

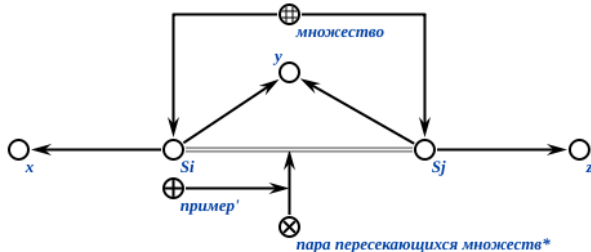
⇒ пояснение*:

[пара пересекающихся множеств* – бинарное неориентированное отношение между двумя множествами, имеющими непустое пересечение*.]

⇒ определение*:

[пара пересекающихся множеств* – бинарное неориентированное отношение между двумя множествами, имеющими, по крайней мере, один общий для этих двух множеств элемент.]

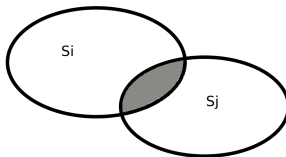
⇒ описание примера*:



⇒ пояснение*:

[Множество S_i и множество S_j являются парой пересекающихся множеств.]

⇒ изображение*:





попарно пересекающиеся множества*

:= [семейство попарно пересекающихся множеств*]

\supset *пересекающиеся множества**

\in *отношение*

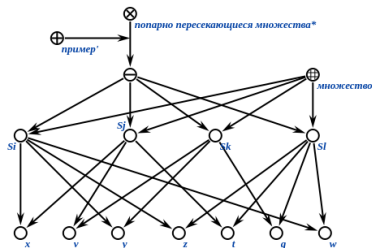
\Rightarrow *определение*:*

[*попарно пересекающиеся множества – семейство множеств, каждая пара которых является парой пересекающихся множеств, т.е. каждая пара которых имеет хотя бы один общий элемент]**

\Rightarrow *примечание*:*

[Не каждое семейство попарно пересекающихся множеств* является семейством пересекающихся множеств*, хотя обратное верно.]

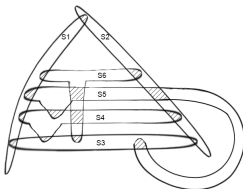
⇒ изображение*:



⇒ пояснение*:

[Множества S_i, S_j, S_k и S_l являются попарно пересекающимися множествами.]

⇒ изображение*:



пересекающиеся множества*

\coloneqq [семейство пересекающихся множеств*]

\coloneqq [быть семейством пересекающихся множеств*]

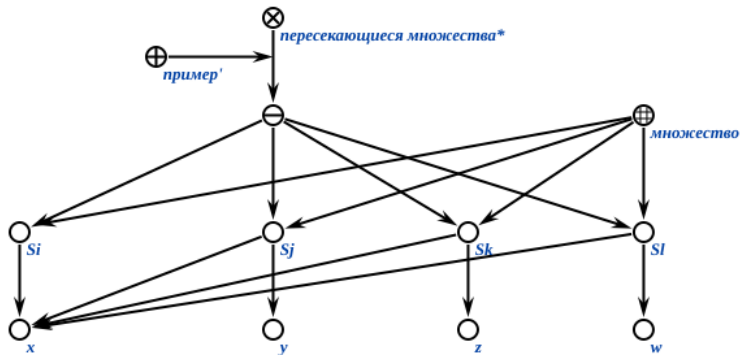
\coloneqq [семейство множеств, имеющих по крайней мере один элемент, являющийся общим для всех этих множеств*]

\supset *попарно пересекающиеся множества**

\Rightarrow *определение**:

[*пересекающиеся множества – это семейство множеств, имеющих по крайней мере один общий для всех этих множеств элемент]**

⇒ описание примера*:



⇒ пояснение*:

[Множества S_i , S_j , S_k и S_l являются пересекающимися множествами.]



пара непересекающихся множеств*

∈ бинарное отношение

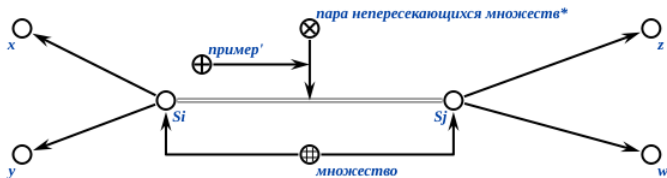
∈ неориентированное отношение

⇒ определение*:

[пара непересекающихся множеств* – это бинарное неориентированное отношение между множествами, результатом пересечения* которых есть пустое множество.]



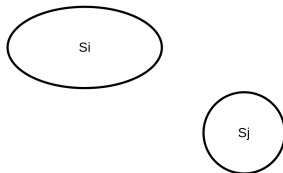
⇒ описание примера*:



⇒ пояснение*:

[Множества S_i и S_j являются парой непересекающихся множеств.]

⇒ изображение*:





попарно непересекающиеся множества*

\coloneqq [семейство попарно непересекающихся множеств*]

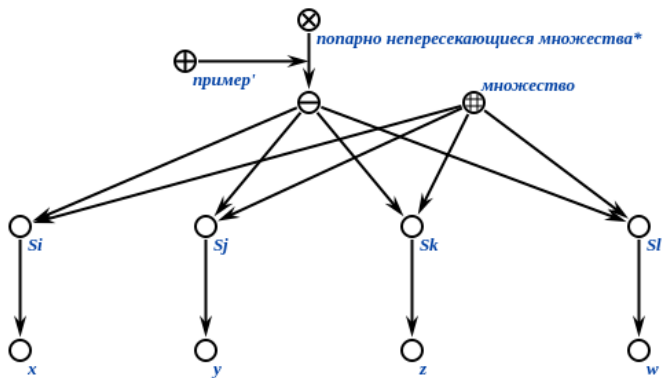
\subset *непересекающиеся множества**

\Rightarrow *определение**:

[попарно непересекающиеся множества* – семейство множеств, каждая пара которых является парой непересекающихся множеств, т.е. каждая пара которых не имеет ни одного общего элемента]



\Rightarrow изображение*:



\Rightarrow пояснение*:

[Множества S_i , S_j , S_k и S_l являются попарно непересекающимися множествами.]



непересекающиеся множества*

\coloneqq [семейство непересекающихся множеств*]

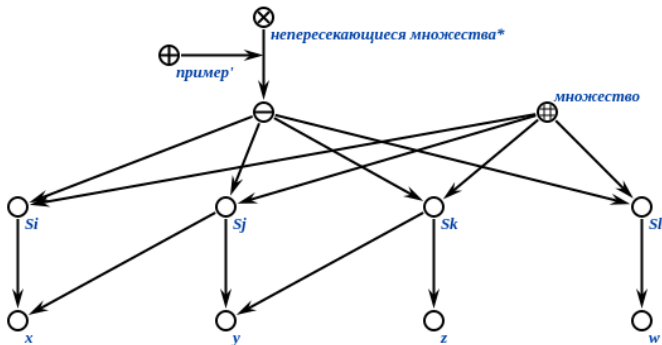
\coloneqq [быть семейством непересекающихся множеств*]

\Rightarrow *определение**:

[*непересекающиеся множества – это семейство множеств, не имеющих ни одного общего элемента для всех этих множеств]**



\Rightarrow изображение*:



\Rightarrow пояснение*:

[Множества S_i , S_j , S_k и S_l являются непересекающимися множествами.]



Лекция 6

Представление в базе знаний отношений и их свойств

Шункевич Д.В.

Белорусский государственный университет информатики и радиоэлектроники



Содержание лекции

Бинарное отношение и способы его задания. Рефлексивное и антирефлексивное бинарное отношение. Симметричное и антисимметричное бинарное отношение. Транзитивное бинарное отношение. Отношения строгого и нестрогого порядка. Отношения полного (линейного) и частичного порядка. Отношения эквивалентности и толерантности. Квазибинарное отношение. n -арное отношение, схема отношения. Область определения отношения, домен. Операции над отношениями (проекция, соединение, композиция). Метаотношения, примеры. Представление в базе знаний.



Понятие отношений

Под *отношением* понимается подмножество декартовой степени множества.

Отношение, заданное на множестве M – это подмножество *декартового произведения* этого множества самого на себя некоторое количество раз.

отношение

\Rightarrow *изображение**:

$$R \subseteq A \times A \dots \times A = A^n$$

В более широком смысле *отношение* – это математическая структура, которая формально определяет свойства различных объектов и их взаимосвязи.



Задание отношений

В теории отношений используются следующие основные типы представления бинарных отношений.

- *Путем перечисления элементов.* Отношение R между элементами множеств A и B задается перечислением пар, которые принадлежат R .
- *Матричное представление.* Бинарное отношение представляется в виде *булевой* (двоичной) матрицы.
- *Графическое представление.* В графическом виде бинарное отношение представляется направленным двудольным графом.



Классификация отношений

отношение

⇒ *разбиение**:

- *класс равномощных связей*
- *класс связей разной мощности*

⇒ *разбиение**:

- *бинарное отношение*
- *небинарное отношение*

⇒ *разбиение**:

- *ориентированное отношение*
- *неориентированное отношение*

⇒ *разбиение**:

- *ролевое отношение*
- *неролевое отношение*



Класс равномоощных связей

класс равномоощных связей

$:=$ [класс связей фиксированной арности]

$:=$ [отношение, обладающее свойством арности]

\supset *унарное отношение*

\supset *бинарное отношение*

\supset *тернарное отношение*

\Rightarrow *определение*:*

[**класс равномоощных связей** – это класс связей, имеющих одинаковую мощность.]



Класс связок разной мощности

класс связок разной мощности

$:=$ [отношение нефиксированной арности]

\subset *небинарное отношение*

\Rightarrow *определение*:*

[*класс связок разной мощности* – это класс связок, имеющих разную мощность.]



Унарное отношение

унарное отношение

$:=$ [отношение аности один]

$:=$ [одноместное множество]

$:=$ [множество синглетонов]

\Rightarrow *определение**:

[унарное отношение – это множество таких отношений на множестве M , являющихся любым подмножеством множества M .]



Бинарное отношение

бинарное отношение

$:=$ [отношение arity два]

$:=$ [двухместное отношение]

\supset *квазибинарное отношение*

\supset *отношение порядка*

\supset *отношение толерантности*

\Rightarrow *разбиение**:

- {
 - *рефлексивное отношение*
 - *антирефлексивное отношение*
 - *частично рефлексивное отношение*}

\Rightarrow *разбиение**:

- {
 - *симметричное отношение*
 - *антисимметричное отношение*
 - *частично симметричное отношение*}



бинарное отношение

⇒ разбиение*:

- транзитивное отношение
- антитранзитивное отношение
- частично транзитивное отношение

⇒ разбиение*:

- ролевое отношение
- неролевое отношение

⇒ определение*:

[бинарное отношение – это множество таких отношений на множестве M , являющихся подмножеством *декартова произведения* множества M .

Если **бинарное отношение** R задано на множестве M и два элемента этого множества a и b связаны данным отношением, то будем обозначать такую связь как aRb .]



Квазибинарное отношение

квазибинарное отношение

⇒ *пояснение**:

[*квазибинарное отношение* – множество ориентированных пар, хотя бы одним из компонентов которых является связка.

Таким образом, *sc-дуги*, принадлежащие *квазибинарным отношениям*, всегда выходят из связок либо входят в связку.]

⊃ *разбиение**

⊃ *пересечение**

⊃ *объединение**



Небинарное отношение

небинарное отношение

⇒ *пояснение**:

[*небинарное отношение* – это множество отношений, хотя бы одна из связок каждого из которых имеет значение мощности больше двух.]



Ориентированность отношений

ориентированное отношение

⇒ *определение**:

[*ориентированное отношение* – это множество таких отношений, каждая связка которых является кортежем.]

неориентированное отношение

⇒ *определение**:

[*неориентированное отношение* – это множество таких отношений, каждая связка которых является неориентированным множеством.]



Свойства бинарных отношений

рефлексивное отношение

⇒ *определение**:

[**рефлексивное отношение** на множестве A – это бинарное отношение, в котором все элементы множества A находятся в отношении R к самому себе.]

⇒ *равенство**

антирефлексивное отношение

⇒ *определение**:

[**антирефлексивное отношение** R на множестве A – это бинарное отношение, в котором все элементы множества A не находятся в отношении R к самому себе.]

⇒ *больше**

⇒ *выше**

частично рефлексивное отношение

⇒ *определение**:

[**частично рефлексивное отношение** R на множестве A – это бинарное отношение, в котором хотя бы один (но не все) элемент множества A находится в отношении R к самому себе.]



Свойства бинарных отношений

симметричное отношение

⇒ *определение**:

[*симметричное отношение R на множестве A – это бинарное отношение, в котором для каждой пары элементов a и b этого множества выполнение отношений aRb влечёт выполнение bRa .*]

⊃ *параллельность**

⊃ *перпендикулярность**

⊃ *родственник**

антисимметричное отношение

⇒ *определение**:

[*антисимметричное отношение R на множестве A – это бинарное отношение, в котором для каждой пары элементов a и b этого множества выполнение отношений aRb и bRa влечёт равенство a и b .*]

⊃ *больше**

⊃ *выше**

частично симметричное отношение

⇒ *определение**:

[*частично симметричное отношение R на множестве A – это бинарное отношение, в котором для каждой пары элементов a и b (но не для всех таких пар) этого множества выполнение отношений aRb влечёт выполнение bRa .*]



Свойства бинарных отношений

транзитивное отношение

⇒ *определение**:

[**транзитивное отношение R** на множестве A – это бинарное отношение, в котором для любых трёх элементов этого множества a, b, c выполнение отношений aRb влечёт выполнение bRc не влечёт выполнение отношения aRc .]

⇒ *параллельность**

⇒ *родственник**

антитранзитивное отношение

⇒ *определение**:

[**антитранзитивное отношение R** на множестве A – это бинарное отношение, в котором для любых трёх элементов этого множества a, b, c выполнение отношений aRb и bRc влечёт выполнение отношения aRc .]

⇒ *перпендикулярность**

частично транзитивное отношение

⇒ *определение**:

[**частично транзитивное отношение R** на множестве A – это бинарное отношение, в котором для любых трёх элементов этого множества a, b, c (но не для всех таких троек) выполнение отношений aRb влечёт выполнение bRc влечёт выполнение отношения aRc .]



Отношение порядка

отношение порядка

\Rightarrow разбиение*:

- *отношение строгого порядка*
- *отношение нестрогого порядка*

\Rightarrow разбиение*:

- *отношение полного порядка*
- *отношение частичного порядка*

\Rightarrow определение*:

[*отношение порядка* – это бинарное отношение, обладающее свойством транзитивности и антисимметричности.]



Отношение строгого и нестрого порядка

отношение строгого порядка

\Rightarrow *определение**:

[*отношение строгого порядка* – это *отношение порядка*, обладающее свойством антирефлексивности.]

\ni *больше**

отношение нестрогого порядка

\Rightarrow *определение**:

[*отношение нестрогого порядка* – это *отношение порядка*, обладающее свойством рефлексивности.]

\ni *больше или равно**



Отношение толерантности

отношение толерантности

⇒ *определение**:

[отношение толерантности – это бинарное отношение, принадлежащее классам симметричное отношение и рефлексивное отношение.]



Отношение эквивалентности

отношение эквивалентности

$:=$ [максимальное семейство отношений эквивалентности]

\subset *отношение толерантности*

\ni *равенство**

\Rightarrow *определение*:*

[*отношение эквивалентности* – это *отношение толерантности*, принадлежащее классу *транзитивных отношений*]

\Rightarrow *примечание*:*

[Каждое отношение эквивалентности уточняет то, что мы считаем эквивалентными сущностями, т.е то, на какие сходства этих сущностей мы обращаем внимание и какие их отличия мы игнорируем (не учитываем).]



Ролевое отношение

ролевое отношение

$:=$ [атрибут]

$:=$ [атрибутивное отношение]

$:=$ [отношение, которое задаёт роль элементов в рамках некоторого множества]

$:=$ [отношение, являющееся подмножеством отношения принадлежности]

\Leftarrow *семейство подмножеств**:
*принадлежность**

\subset *бинарное отношение*

\supset *числовой атрибут*

\Rightarrow *пояснение**:

[*ролевое отношение* – это отношение, являющееся подмножеством отношения принадлежности.]



⇒ *правило идентификации экземпляров**:

[В конце каждого *идентификатора*, соответствующего экземпляра класса ***ролевое отношение***, не являющегося системным, ставится знак "/".

Например:

ключевой экземпляр'

Из-за ограничений в разрешенном алфавите символов, в системном идентификаторе не может быть использовать знак “/”, поэтому в начале каждого *системного идентификатора*, соответствующего экземплярам класса ***ролевое отношение*** ставится префикс “rrel_”.

Например:

rrel_key_sc_element]



Числовой атрибут

числовой атрибут

$:=$ [порядковый номер]

$:=$ [номер компонента ориентированной связки]

$\ni 1'; 2'; 3'; 4'; 5'; 6'; 7'; 8'; 9'; 10'$

; ... \Rightarrow *пояснение**:

[*числовой атрибут* – ролевое отношение, задающее порядковый номер элемента некоторой ориентированной связки, не уточняя при этом семантику такой принадлежности. Во многих случаях бывает достаточно использовать числовые атрибуты, чтобы различать компоненты связки, семантика каждого из которых дополнительно оговаривается, например, при определении отношения, которому данная связка принадлежит.]



Нероловое отношение

нероловое отношение

⇒ *разбиение**:

- { • *небинарное отношение*
- *нероловое бинарное отношение*

⇒ *пояснение**:

[*нероловое отношение* – отношение, не являющееся подмножеством отношения принадлежности.]



⇒ *правило идентификации экземпляров**:

[В конце каждого *идентификатора*, соответствующего экземплярам класса ***неролевое отношение***, не являющегося системным, ставится знак “*”.

Например:

*включение**

Из-за ограничений в разрешенном алфавите символов, в системном идентификаторе не может быть использовать знак “*”, поэтому в начале каждого *системного идентификатора*, соответствующего экземплярам класса ***неролевое отношение*** ставится префикс “nrel_”.

Например:

nrel_inclusion]



Неролевое бинарное отношение

неролевое бинарное отношение

⇒ *пояснение**:

[неролевое бинарное отношение – бинарное отношение, не являющееся ролевым отношением.]

арность

$:=$ [арность отношения]

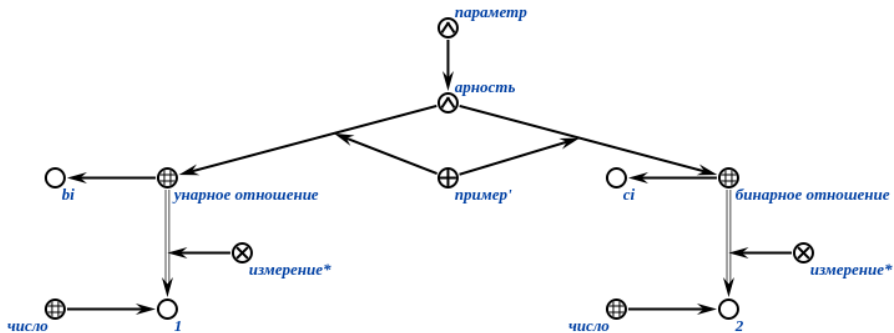
\in *параметр*

\Rightarrow *пояснение**:

[*арность* – это параметр, каждый элемент которого представляет собой класс *отношений*, каждая связка которых имеет одинаковую *мощность*. Значение данного *параметра* совпадает со значением *мощности* каждой из таких связок.]



⇒ описание примера*:





Область определения

*область определения**

$:=$ [область определения отношения*]

\in *бинарное отношение*

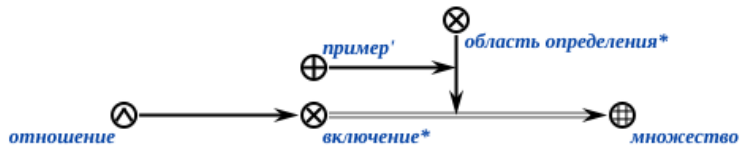
\Rightarrow *пояснение**:

[область определения* – это *бинарное отношение*, связывающее отношение со множеством, являющимся его областью определения.

Областью определения отношения будем называть результат теоретико-множественного объединения всех связок этого отношения, или, другими словами, результат теоретико-множественного объединения всех множеств, являющихся доменами данного отношения.]



\Rightarrow описание примера*:





Атрибут отношения

*атрибут отношения**

\coloneqq [ролевой атрибут, используемый в связках заданного отношения*]

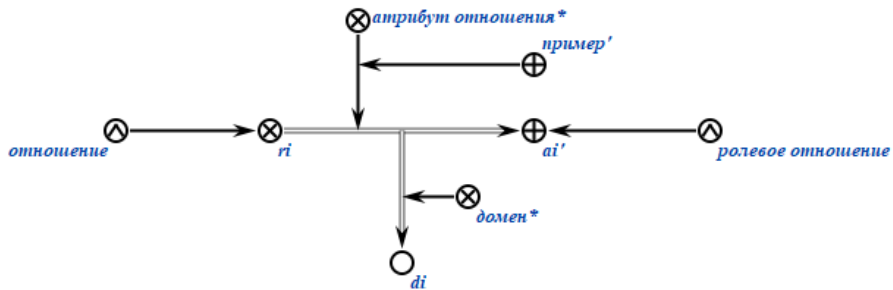
\in бинарное отношение

\Rightarrow пояснение*:

[*атрибут отношения** – это бинарное отношение, связывающее заданное отношение с ролевым отношением, используемым в данном отношении для уточнения роли того или иного элемента связок данного отношения.]



\Rightarrow описание примера*:



*домен**

$:=$ [домен отношения по заданному атрибуту*]

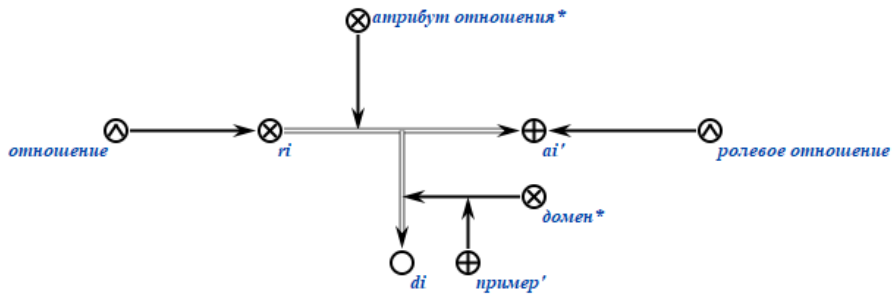
\in *бинарное отношение*

\Rightarrow *пояснение**:

[**домен** – это *бинарное отношение*, связывающее связку отношения *атрибут отношения** со множеством, являющимся доменом заданного отношения по заданному атрибуту. Множество ***di*** является доменом отношения ***ri*** по атрибуту ***ai*** в том и только том случае, если элементами этого множества являются все те и только те элементы связок отношения ***ri***, которые имеют в рамках этих связок атрибут ***ai***.]



\Rightarrow описание примера*:





Первый домен

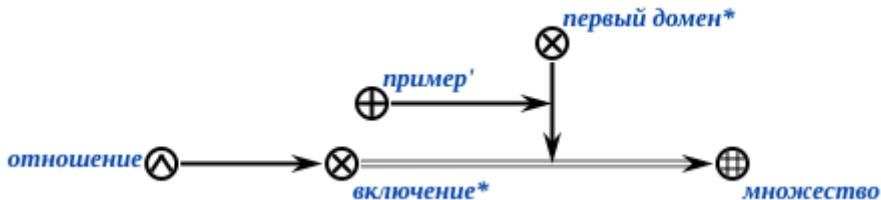
*первый домен**

∈ *бинарное отношение*

⇒ *определение**:

[*первый домен** – это бинарное отношение, связывающее отношение с множеством, являющимся доменом по атрибуту I' данного отношения.]

⇒ *описание примера**:





Второй домен

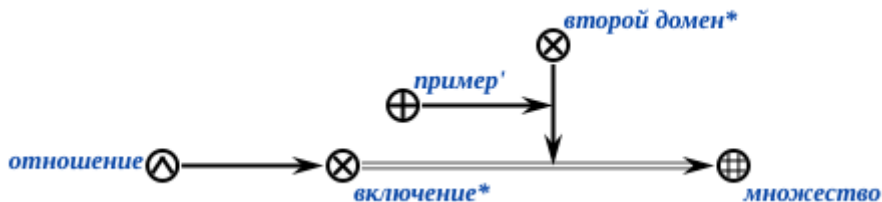
*второй домен**

\in бинарное отношение

\Rightarrow определение*:

[*второй домен** – это бинарное отношение, связывающее отношение с множеством, являющимся доменом по атрибуту $2'$ данного отношения.]

\Rightarrow описание примера*:





Композиция отношений

*композиция отношений**

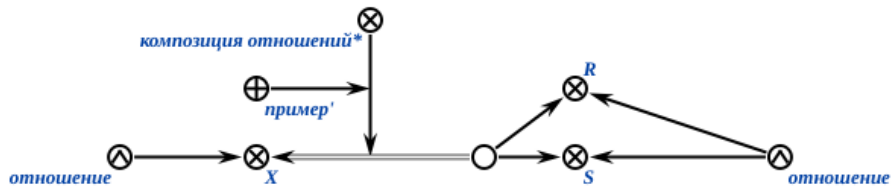
∈ *квазибинарное отношение*

⇒ *определение**:

[*композиция отношений** – это квазибинарное отношение, связывающее два бинарных отношения с отношением, являющимся их композицией. Под композицией бинарных отношений R и S будем понимать множество $\{(x, y) | \exists z (xSz \wedge zRy)\}$]



\Rightarrow описание примера*:



Фактор-множество

*фактор-множество**

$:=$ [быть фактор-множеством*]

$:=$ [множество всевозможных максимальных множеств из попарно эквивалентных элементов*]

$:=$ [множество всевозможных классов эквивалентности для заданного отношения эквивалентности*]

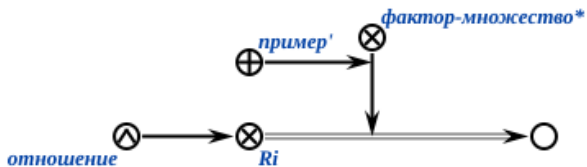
\in *бинарное отношение*

\Rightarrow *определение**:

[**фактор множество*** – это бинарное ориентированное отношение, каждая связка которого связывает некоторое отношение эквивалентности со множеством всех соответствующих этому отношению классов эквивалентности. Каждый такой класс представляет собой максимальное множество сущностей, каждая пара которых принадлежит указанному выше отношению эквивалентности.]



\Rightarrow описание примера*:



метаотношение

⇒ *определение**:

[метаотношение – это *отношение*, в каждой связке которого есть по крайней мере один компонент, являющийся знаком некоторого *отношения*.]

⇒ *домен**

⇒ *атрибут отношения**

⇒ *область определения**



Лекция 7

Представление в базе знаний параметров и величин

Шункевич Д.В.

Белорусский государственный университет информатики и радиоэлектроники



Содержание лекции

Понятие шкалы, величины и параметра. Измеряемые и неизмеряемые параметры. Точные, неточные и интервальные величины. Связь с понятием мощности множества, бесконечного и конечного множества.



Определение параметра

параметр – класс, являющийся семейством всевозможных классов эквивалентности или толерантности, задаваемых либо отношением эквивалентности, либо отношением толерантности.



Определение параметра

параметр

- $:=$ [характеристика]
- $:=$ [свойство]
- $:=$ [признак]
- $:=$ [измеряемое свойство]
- $:=$ [класс классов]
- $:=$ [признак классификации или покрытия некоторого класса сущностей]
- $:=$ [признак разбиения или покрытия некоторого класса сущностей]
- $:=$ [семейство множеств, разбивающих или покрывающих некоторый класс сущностей]
- $:=$ [семейство классов сущностей, обладающих одинаковым соответствующим свойством]
- \supset *ориентированный параметр*



Типы параметров. Измеряемый параметр

параметр

⇒ *разбиение**:

- { • *измеряемый параметр*
- *неизмеряемый параметр*
- }

измеряемый параметр

:= [количественный параметр]

:= [семейство измеряемых величин]

⇒ *пояснение**:

[Каждый *измеряемый параметр* представляет собой *параметр*, значение (элемент, экземпляр) которого трактуется как *величина*, которой можно поставить в соответствие ее числовое значение на основании выбранной единицы измерения и/или точки отсчета (нулевой отметки выбранной шкалы).]

⊃ *параметр, измеряемый по шкале*



Неизмеряемый параметр

неизмеряемый параметр

\coloneqq [качественный параметр]

неизмеряемый параметр – это параметр, для которого нельзя подобрать числовой эквивалент и единицу измерения. Примеры: цвет, вкус.



параметр

⇒ *пояснение**:

[Каждый ***параметр*** представляет собой класс, являющийся семейством всевозможных классов эквивалентности или толерантности, задаваемых либо *отношением эквивалентности*, либо *отношением толерантности* (симметричным, рефлексивным, но частично транзитивным).

Так, например, элементами (значениями, величинами) ***параметра*** *длина* являются либо классы эквивалентности, задаваемые отношением эквивалентности “*иметь точно одинаковую длину**”, либо классы толерантности, задаваемые отношением вида “*иметь приблизительно одинаковую длину с указываемой точностью**”, либо интервальные классы, задаваемые бинарными отношениями вида “*иметь длину, находящуюся в одном и том же указываемом интервале**” (например, от 1 метра до 2 метров).]



Параметрическое пространство

параметр

⇒ *пояснение**:

[Заметим, что среди элементов (значений, величин) параметра могут встречаться пересекающиеся множества (классы), но объединение всех элементов каждого параметра есть не что иное, как класс всевозможных сущностей, обладающих этим параметром (свойством, характеристикой).

Каждый конкретный параметр (характеристика), т.е. каждый элемент класса всевозможных параметров (характеристик) есть, по сути, признак классификации сущностей, обладающих этой характеристикой, по принципу эквивалентности (одинаковости значения) этой характеристики.

Параметр может разбиваться на классы для уточнения некоторого свойства, например элементами параметра цвет будут классы, соответствующие конкретным цветам (синий, красный и т.д.), в свою очередь каждый конкретный цвет может включать более частные классы, уточняющие данное свойство, например, темно-синий, светло-красный и т.д.]



Параметрическое пространство

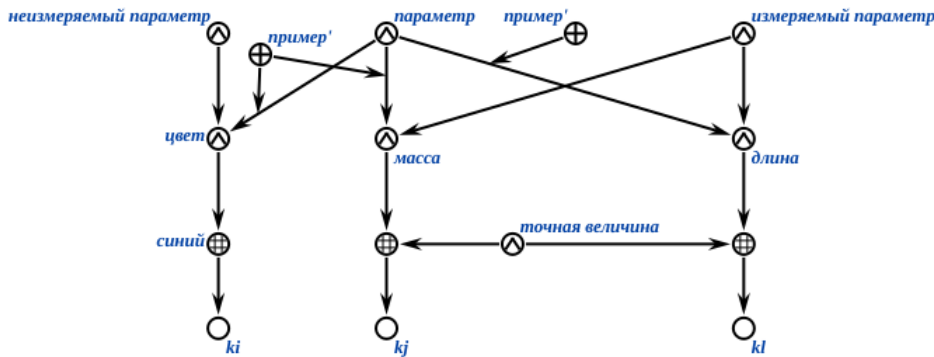
параметр

⇒ *пояснение**:

[Другими словами, каждому множеству сущностей может ставиться в соответствие набор (семейство) параметров (параметрическое пространство), которыми обладают сущности этого множества – аналог семейства отношений, определенных (заданных) на этом множестве. Часто бывает важно построить такое параметрическое пространство, "точки" которого взаимнооднозначно соответствуют параметризуемым сущностям (например, набор параметров, позволяющих однозначно идентифицировать, установить личность каждого человека).

Таким образом, для каждого используемого элемента (значения) какого-либо параметра, необходимо явно указывать спецификацию этого значения (точное значение, неточное значение, интервальное значение, точность, интервал).]

Параметр. Примеры



величина

\coloneqq [значение количественного параметра]

\coloneqq [значение измеряемого параметра]

\coloneqq [класс сущностей, имеющих одинаковое значение соответствующего параметра]

⊃ *точная величина*

⊃ *неточная величина*

⊃ *интервальная величина*



величина

⇒ *пояснение**:

[Каждая **величина** представляет собой однозначный и независящий от шкалы измерения результат измерения некоторой характеристики у некоторой сущности.

Каждой **величине** можно поставить в соответствие ее числовое значение на основании выбранной единицы измерения и точки отсчета (нулевой отметки выбранной шкалы, в случае, если измерение осуществляется по шкале).

Нельзя путать значение параметра (**величину**) и значение величины по некоторой шкале, которое может быть скалярным и векторным.]



Отношение *измерение**

*измерение**

- := [значение параметра*]
- := [значение заданной величины заданного параметра*]
- := [измерение как соответствие*]
- := [результат измерения заданной величины в заданной единице измерения и по заданной шкале*]
- := [бинарное ориентированное отношение, связывающее различные величины с результатами их измерения в различных единицах измерения и по различным шкалам*]



Отношение *измерение**

*измерение**

⇒ *пояснение**:

[Связки отношения *измерение** связывают величину и ее значение в некоторой единице измерения (в том числе, в интервале) или по некоторой шкале.

Конкретная единица измерения или шкала указывается дополнительно при помощи соответствующего отношения. Одной величине может соответствовать только одно значение в каждой возможной единице измерения или одна точка на некоторой шкале.]



Отношение *точность**

*точность**

$:=$ [отклонение*]

$:=$ [степень точности неточного значения параметра*]

\in *бинарное отношение*

\Rightarrow *пояснение**:

[Связки отношения *точность** связывают *неточную величину* и *точную величину* того же класса, задающую максимальное возможное отклонение указанной *неточной величины* от своего значения.]



Отношение *единица измерения**

*единица измерения**

∈ бинарное отношение

:= [единица по шкале*]

:= [единичная отметка по шкале*]

⇒ пояснение*:

[Связки отношения *единица измерения** связывают знак конкретного *измерения* с *фиксированной единицей измерения* и некоторую *точную величину*, входящую в тот же конкретный *параметр*, что и первый компонент связок этого конкретного измерения, и которая используется в данном случае в качестве единицы измерения.]



Измерение с фиксированной единицей измерения

измерение с фиксированной единицей измерения

\Leftarrow семейство подмножеств*:

*измерение**

\Rightarrow пояснение*:

[Каждая *измерение с фиксированной единицей измерения* представляет собой подмножество отношения *измерение** и характеризуется некоторой *единицей измерения**, которая является элементом того же параметра (семейством сущностей, имеющих значение данного параметра, совпадающее с этой единицей измерения).]

Измерение по шкале

измерение по шкале

$:=$ [шкала]

\Leftarrow семейство подмножеств*:
*измерение**

\Rightarrow пояснение*:

[Каждое *измерение по шкале* представляет собой подмножество отношения *измерение** и характеризуется не единицей измерения, а некоторой точкой отсчета для данной *шкалы*.

Результатом *измерения по шкале* будет некоторая точка шкалы, отстоящая от точки отсчета на определенное расстояние в нужную сторону (меньшую или большую). Понятно, что это расстояние может быть измерено любыми единицами измерения, но его величина при этом останется неизменной.]



Измерение по шкале

измерение по шкале

⇒ *пояснение**:

[Не стоит путать измерение по *измерение по шкале*, которое зависит от *нулевой отметки**, с измерением изменения того же *параметра*, которое характеризуется единицей измерения и не зависит от точки отсчета. Например, не стоит путать дату по некоторому календарю, соответствующую *началу* какого-либо процесса, и *длительность* этого процесса, которая не зависит от выбранного календаря.]



Измерение по шкале. Нулевая отметка

*нулевая отметка**

$:=$ [нуль по шкале*]

$:=$ [начало отсчета*]

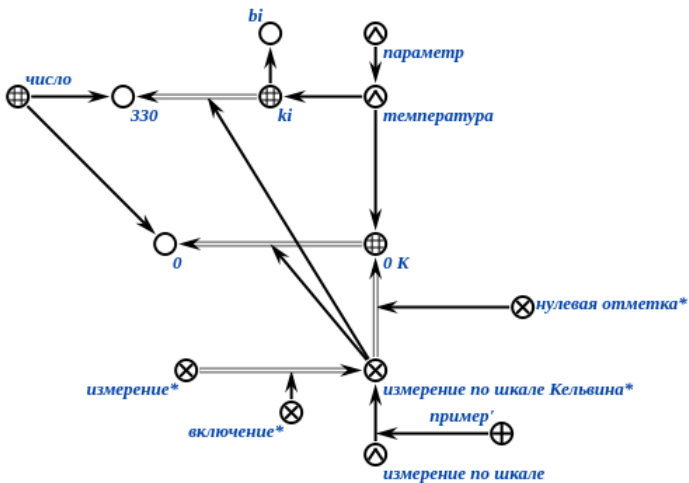
$:=$ [точка отсчета*]

\in *бинарное отношение*

\Rightarrow *пояснение**:

[Связки отношения *нулевая отметка** связывают знак некоторого измерения по шкале со знаком точной величины того же параметра, которая в рамках данной шкалы принимается за точку отсчета.]

Измерение по шкале. Пример





Точная величина

точная величина

\coloneqq [точное значение параметра]

\coloneqq [множество всех точных значений параметра]

\coloneqq [значение параметра, являющееся семейством классов эквивалентности, соответствующим некоторому отношению эквивалентности]

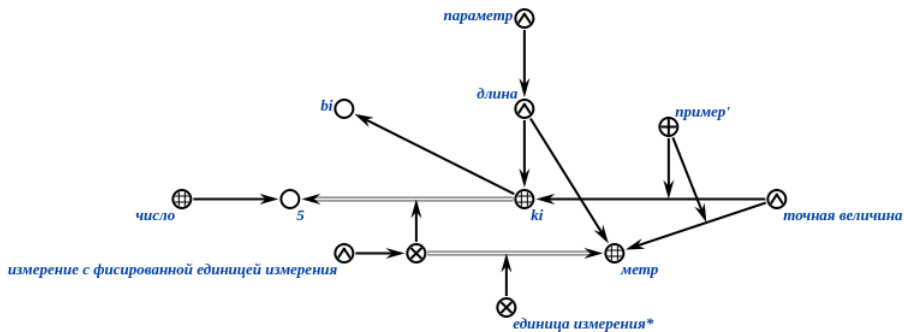
\coloneqq [класс эквивалентности]

\Rightarrow *пояснение**:

[Каждая *точная величина* имеет одно фиксированное значение в некоторой единице измерения или по какой-либо шкале. При этом считается, что все элементы такого класса имеют одинаковое значение данного параметра и отклонениями можно пренебречь.

Каждой *точной величине* можно поставить в соответствие группу *неточных величин*, являющихся не разбиениями, а покрытиями того же множества, но с разной степенью точности.]

Точная величина. Пример





Неточная величина

неточная величина

\coloneqq [множество неточных значений параметра]

\coloneqq [приблизительная величина]

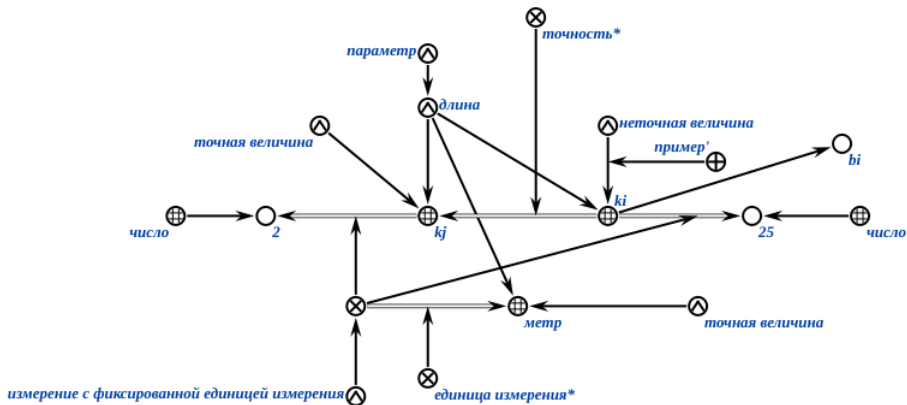
\coloneqq [приблизительное значение параметра]

\coloneqq [значение параметра в интервале с нефиксированными границами]

\Rightarrow *пояснение**:

[Каждой *неточной величине* ставится в соответствие ее значение в некоторой единице измерения или по какой-либо шкале, а также дополнительно указывается *точность**, т.е. возможное отклонение от данного значения.]

Неточная величина. Пример





Интервальная величина

интервальная величина

:= [интервальное значение параметра]

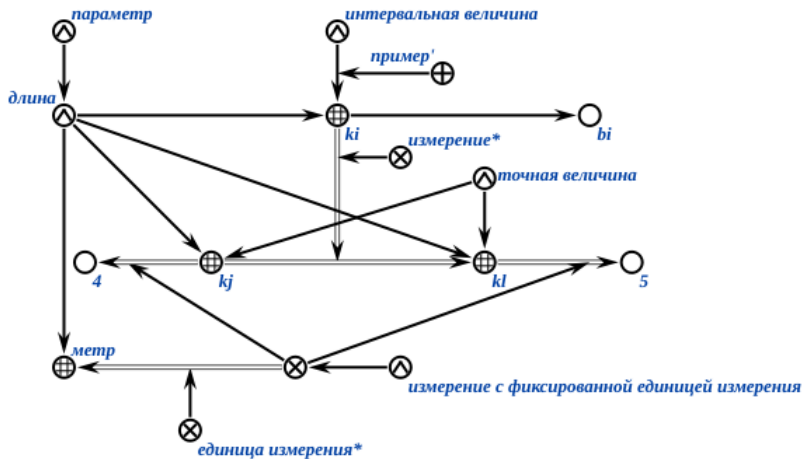
:= [значение параметра в интервале с фиксированными границами]

:= [интервал значения параметра из множества пересекающихся интервалов разной длины, имеющих нефиксированные границы]

⇒ *пояснение**:

[Каждая *интервальная величина* представляет собой класс сущностей, находящихся в рамках точно заданного интервала, минимальная и максимальная точка которого являются *точными величинами*. Результатом *измерения** такой величины является ориентированная пара, первым компонентом которой является левая (меньшая) граница интервала, вторым компонентом – правая (большая) граница интервала.]

Интервальная величина. Пример





Измерение как действие

действие. измерение

\coloneqq [измерение как действие]

\coloneqq [действие, направленное на установление связи, принадлежащей отношению измерение* и связывающей величину, которая принадлежит заданному параметру, и которой принадлежит заданная сущность, и соответствующее значение этой величины на некоторой шкале]

\coloneqq [действие, направленное на решение задачи измерения заданного параметра у заданной сущности]

\subset *действие*



Измерение как задача

задача. измерение

:= [спецификация действия измерения]

:= [спецификация действия, целью которого является измерение заданного параметра у заданной сущности]

\subset *задача*



Лекция 8

Представление структур и семантических окрестностей в базе знаний

Шункевич Д.В.



Содержание лекции

Понятие структуры как фрагмента базы знаний. Типология структур, роли элементов структуры. Отношения на структурах. Представление в базе знаний метаинформационных конструкций. Понятие семантической окрестности, типология семантических окрестностей.

Понятие структуры как фрагмента базы знаний



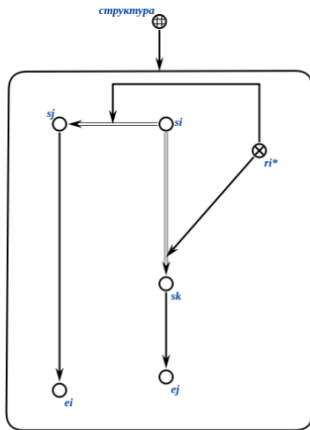
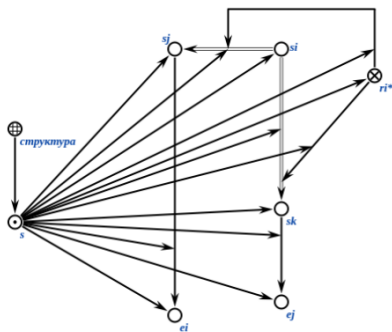
При накоплении больших объемов информации в базе знаний возникает необходимость выделять **целые фрагменты** базы знаний и иметь возможность их специфицировать, рассматривая **как отдельные сущности**.

Такой фрагмент базы знаний назван структурой (sc-структурой).

Под *структурой* будем понимать **множество sc-элементов**, удаление одного из которых может привести к нарушению целостности этого множества.

Структура может изображаться путем явного указания всех пар принадлежности элементов этой структуре, а также в виде контура, содержащего все элементы, входящие в состав этой структуры.

Пример представления структуры как фрагмента базы знаний





Типология структур

структура

\Rightarrow разбиение*:

- *связная структура*
- *несвязная структура*

\Rightarrow разбиение*:

- *тривиальная структура*
:= [структура, не содержащая связок]
- *нетривиальная структура*
:= [структура, содержащая хотя бы одну связку]



Типология структур

Структуре, представленной в SC-коде, поставим в соответствие оргграф, вершинами которого являются sc-элементы, а дугами – связи отношений инцидентности, связывающие sc-коннекторы с инцидентными им sc-элементами, которые являются компонентами указанных sc-коннекторов. Если полученный таким способом оргграф является связным оргграфом, то исходную структуру будем считать связной структурой. Если полученный таким способом оргграф не является связным оргграфом, то исходную структуру будем считать несвязной структурой.



Типология структур

По признаку стационарности выделяются динамические структуры (процессы), состав которых меняется с течением времени, и статические структуры, состав которых не меняется с течением времени.

структура

⇒ *разбиение**:

- { • *процесс*
 - := [динамическая структура]
 - := [нестационарная структура]
- *статическая структура*
 - := [стационарная структура]
 - := [структура, не изменяющаяся во времени]



Типология структур

По признаку времени существования выделяются временные структуры и постоянно существующие структуры.

структура

\Rightarrow разбиение*:

- $\{$
 - *временная структура*
 - *постоянно существующая структура* $\}$



Роли элементов структуры

Для формального представления структур используются понятия, описывающие роли элементов в рамках структуры. *элемент структуры'* – неосновное понятие, ролевое отношение, указывающее на все элементы каждой структуры.

элемент структуры'

\Rightarrow разбиение*:

- { • *непредставленное множество'*
- *полностью представленное множество'*
- *частично представленное множество'*
- *элемент структуры, не являющийся множеством'*
- }



Роли элементов структуры

элемент структуры'

\Rightarrow разбиение*:

- *максимальное множество'*

\Rightarrow пояснение*:

[Ролевое отношение, связывающее структуру со знаком множества, для которого не существует множества, которое было бы надмножеством указанного множества и знак которого был бы элементом этой же структуры]

- *немаксимальное множество'*

\Rightarrow пояснение*:

[Ролевое отношение, связывающее структуру со знаком множества, для которого в рамках данной структуры существует множество, являющееся надмножеством указанного множества.]

}

Роли элементов структуры

элемент структуры'

⇒ разбиение*:

{ • *первичный элемент'*

⇒ *пояснение**:

[Ролевое отношение, указывающее на (атомарный) элемент структуры, который не имеет элементов, ему принадлежащих. При этом соответствующая пара принадлежности может существовать, но в состав данной структуры не входить.]

• *вторичный элемент'*

:= [элемент данной структуры имеющий семантический уровень более 2']

:= [непервичный элемент']

⇒ *пояснение**:

[Ролевое отношение, указывающее на элемент структуры, обозначающий множество элементов, где обязательно хотя бы один элемент указанного множества входил бы в указанную структуру.]

}



Роли элементов структуры

элемент структуры'

\Rightarrow *разбиение**:

{ • *структура первого уровня'*

\Rightarrow *пояснение**:

[связка первичных элементов, тривиальная структура из первичных элементов или класс первичных элементов]

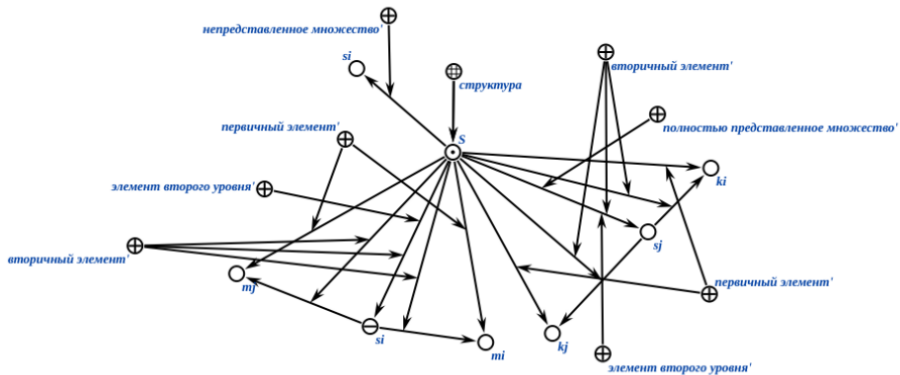
• *вторичная структура'*

\Rightarrow *пояснение**:

[структура, среди элементов которой есть хотя бы один вторичный элемент']

}

Пример ролей элементов структуры





Отношения на структурах

*бинарное отношение**

- \ni полиморфность*
- \subset соответствие*
- \ni полиморфизм*
- \ni гомоморфность*
- \subset соответствие*
- \ni гомоморфизм*
- \ni изоморфность*
- \subset соответствие*
- \ni изоморфизм*
- \ni автоморфность*
- \subset соответствие*
- \ni автоморфизм*



Отношения на структурах

*полиморфность**

⇒ *пояснение**:

[полиморфность* – это соответствие, заданное на структурах, при котором каждому элементу из области определения соответствия (первой структуры) ставится в соответствие один или более элемент из области значения соответствия (второй структуры), при этом существует хотя бы один элемент области определения соответствия, которому соответствуют два или более элемента из области значения соответствия.]



Отношения на структурах

*гомоморфность**

\Rightarrow *пояснение**:

[гомоморфность* – это соответствие, заданное на структурах, при котором каждому элементу из области определения соответствия (первой структуры) ставится в соответствие только один элемент из области значения соответствия (второй структуры).]



Отношения на структурах

*изоморфность**

\Rightarrow *пояснение**:

[изоморфность* – это гомоморфность*, при которой для каждого элемента из области значения существует ровно один соответствующий элемент из области определения.]

*автоморфность**

\Rightarrow *пояснение**:

[автоморфность* – это изоморфность*, у которой область определения соответствия и область значения соответствия совпадают.]



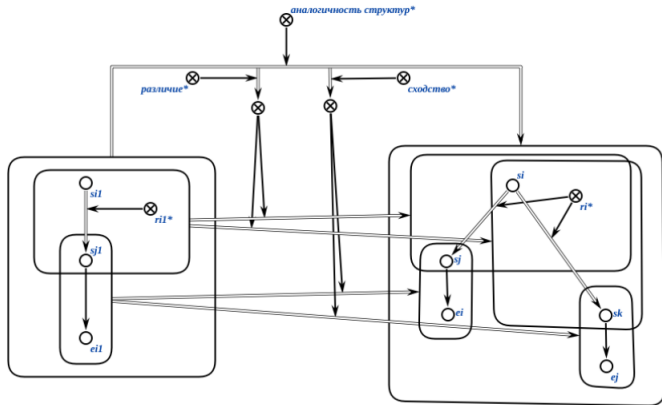
Отношения на структурах

*аналогичность структур**

- \subset *соответствие**
- \in *бинарное отношение**
- \Rightarrow *пояснение**:

[аналогичность структур* – соответствие*, задаваемое на структурах, и фиксирующее факт наличия некоторой аналогии на подструктурах (подмножествах) указанных структур. Каждой ориентированной паре, принадлежащей аналогичности структур* может быть поставлено в соответствие множество пар, задающих сходства* некоторых подструктур и различия* некоторых подструктур исходных структур.]

Пример





Понятие семантической окрестности

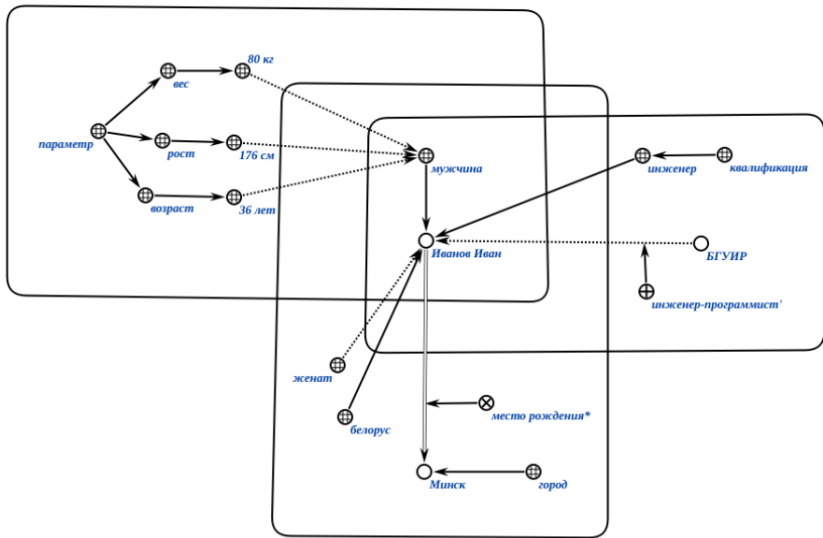
Для спецификации отдельных сущностей в рамках базы знаний вводится понятие *семантической окрестности*.

Семантическая окрестность представляет собой спецификацию заданной сущности, знак которой указывается как ключевой элемент этой спецификации. В отличие от других видов знаний, семантическая окрестность имеет только один ключевой элемент.

Набор признаков, по которым можно специфицировать сущности, различен. Кроме того, может возникнуть необходимость специфицировать одну и ту же сущность в различных аспектах и явно фиксировать эти аспекты в базе знаний.

Так, например, одну и ту же персону можно описывать с профессиональной, медицинской, гражданской и других точек зрения.

Пример





Понятие семантической окрестности

семантическая окрестность

:= [описание заданной сущности, знак которой указывается как ключевой элемент этой спецификации]

- ⊂** *знание*
- ⊃** *семантическая окрестность по инцидентным коннекторам*
- ⊃** *полная семантическая окрестность*
- ⊃** *базовая семантическая окрестность*
- ⊃** *специализированная семантическая окрестность*



Типология семантических окрестностей

семантическая окрестность по инцидентным коннекторам

⊃ *семантическая окрестность по выходящим дугам*

⊃ *семантическая окрестность по входящим дугам*

⇒ *пояснение**:

[вид семантической окрестности, в которую входят все коннекторы, инцидентные заданному элементу, а также все элементы, инцидентные указанным коннекторам.]



Типология семантических окрестностей

полная семантическая окрестность

:= [полная спецификация некоторой описываемой сущности]

базовая семантическая окрестность

:= [минимально достаточная семантическая окрестность]

специализированная семантическая окрестность

:= [вид семантической окрестности, набор связей для которой уточняется отдельно для каждого типа такой окрестности.]



Типология семантических окрестностей

специализированная семантическая окрестность

- ⌋ *пояснение*
- ⌋ *примечание*
- ⌋ *правило идентификации экземпляров*
- ⌋ *терминологическая семантическая окрестность*
- ⌋ *теоретико-множественная семантическая окрестность*
- ⌋ *логическая семантическая окрестность*
- ⌋ *описание типичного экземпляра*
- ⌋ *описание декомпозиции*



Лекция 9

Представление логических знаний

Шункевич Д.В.

Белорусский государственный университет информатики и радиоэлектроники



Содержание лекции

Формальные логические языки. Алфавит и синтаксис языка SCL. Предикаты и булевы функции. Логические связки (операторы), таблицы истинности. Логическая формула, равносильные логические формулы, логические законы. Классы логических формул. Кванторы, законы двойственности. Связанные и свободные переменные. Открытые и замкнутые формулы. Представление в базе знаний.



Формальный язык

Формальный язык – множество конечных слов (строк) над конечным алфавитом. Если L – формальный язык, а A – его алфавит, то $L \subseteq A^*$, где

A^* – операция замыкания множества A . Формальными логическими языками являются язык **логики высказываний** и язык **логики предикатов первого порядка**.



Формальный логический язык SCL

Формальный логический язык SCL построен на базе языка SC как его подъязык путем фиксации определенного набора специальных ключевых узлов, т.е. узлов, семантика которых должна быть априори известна и согласована. Логический язык SCL является языком теоретико-множественного типа, в основе которого лежит трактовка логических связок и кванторов через понятия множества, кортежа, атрибута, отношения, т.е. трактовка формальных теорий и неатомарных логических формул как реляционных структур над высказываниями.



Предикат. Булева функция

Предикат – функция с областью значений $\{\top, \perp\}$ и областью определений M^n , где M – множество объектов предметной области.

Булева функция – функция с областью определения B^n и областью значений B , где $B = \{\top, \perp\}$.

Булева функция задаётся конечным набором значений, что позволяет представить её в виде таблицы истинности.



Таблица истинности

Таблица истинности – таблица, устанавливающая соответствие между всеми возможными наборами логических переменных, входящих в логическую функцию, и значениями функции.

X	Y	$F(X, Y)$
0	0	0
0	1	1
1	0	1
1	1	0

Высказывание

высказывание

⇒ *пояснение**:

[Под **высказыванием** понимается некоторая *структура* (в которую входят *sc-константы* из некоторой предметной области и/или *sc-переменные*) или *логическая связка*, которая может трактоваться как истинная или ложная в рамках какой-либо *предметной области*.]

⇐ *разбиение**:

- { • *атомарное высказывание*
- *неатомарное высказывание*
- }

⇐ *разбиение**:

- { • *фактографическое высказывание*
- *логическая формула*
- }



Высказывание

Истинность **высказывания** задается путем указания принадлежности знака этого высказывания *формальной теории*, соответствующей данной *предметной области*. Ложность высказывания задается путем указания принадлежности знака *отрицания** этого высказывания данной *формальной теории*.

Явно указанная непринадлежность **высказывания** *формальной теории* может говорить как о его ложности в рамках данной теории (если это указано рассмотренным выше образом), так и о том, что данное **высказывание** вообще не рассматривается в данной *формальной теории* (например, использует понятия, не принадлежащие данной *предметной области*).

Одно и то же **высказывание** может быть истинно в рамках одной *формальной теории* и ложно в рамках другой.



Высказывание формальной теории

высказывание формальной теории'

\in неосновное понятие

\Leftarrow разбиение*:

- { • истинное высказывание'
- ложное высказывание'
- нечеткое высказывание'
- бессмысленное высказывание'
- }



Истинное высказывание. Ложное высказывание.

истинное высказывание'

- := [высказывание, истинное в рамках данной формальной теории']
- := [высказывание, знак которого принадлежит данной формальной теории']

ложное высказывание'

- := [высказывание, ложное в рамках данной формальной теории']
- := [высказывание, знак отрицания которого принадлежит данной формальной теории']



Нечёткое высказывание. Бессмысленное высказывание

нечеткое высказывание'

- := [гипотетическое высказывание']
- := [высказывание, возможно истинное или ложное в рамках данной формальной теории']
- := [высказывание, истинное или ложное в рамках данной формальной теории с некоторой вероятностью']

бессмысленное высказывание'

- := [высказывание, бессмысленное в рамках данной формальной теории']
- := [высказывание, не рассматриваемое в рамках данной формальной теории']
- ⇒ *пояснение**:

[Высказывание является бессмысленным в рамках заданной формальной теории, если в какое-либо *атомарное высказывание* в его составе (или в само это высказывание, если оно является атомарным) входит какая-либо *sc-константа*, не являющаяся элементом предметной области, описываемой указанной *формальной теорией*.]



Атомарное высказывание. Неатомарное высказывание

атомарное высказывание

\subset структура

\Leftarrow разбиение*:

- { • атомарное фактографическое высказывание
- атомарная логическая формула

\Rightarrow пояснение*:

[**атомарное высказывание** – это высказывание, которое содержит хотя бы один *sc*-элемент, не являющийся знаком другого высказывания.]

неатомарное высказывание

\Rightarrow пояснение*:

[**неатомарное высказывание** – это высказывание, в состав которого входят только знаки других высказываний. Следует отметить, что мы не можем говорить об истинности либо ложности **неатомарного высказывания** в рамках какой-либо *формальной теории*, в случае, когда невозможно установить истинность либо ложность любого из его элементов в рамках этой же *формальной теории*.]



Фактографическое высказывание

фактографическое высказывание

⊃ *атомарное фактографическое высказывание*

Под *фактографическим высказыванием* понимается:

- *атомарное высказывание*, в состав которого не входит ни одна *с-переменная*
- *неатомарное высказывание*, все элементы которого также являются *фактографическими высказываниями*



Логическая формула

Под **логической формулой** понимается:

- *атомарное высказывание*, в состав которого входит хотя бы одна *sc-переменная*;
- *неатомарное высказывание*, хотя бы один элемент которого является *логической формулой*.

логическая формула

⇐ *разбиение**:

- *атомарная логическая формула*
- *неатомарная логическая формула*

⇐ *разбиение**:

- *открытая логическая формула*
- *замкнутая логическая формула*



Логические связки (операторы)

*логическая связка**

$:=$ [неатомарная логическая формула]

$:=$ [логический оператор*]

$:=$ [пропозициональная связка*]

\in *класс связок разной мощности*

\Leftarrow *семейство подмножеств**:

неатомарное высказывание

\Rightarrow *пояснение**:

[**логическая связка*** – это отношение (класс связок), связками которого являются *высказывания*. **логическая связка*** – это *отношение*, областью определения которого является множество *высказываний*, при этом само это отношение и некоторые его подмножества могут быть *классами связок разной мощности*.]



Конъюнкция

*конъюнкция**

$:=$ [логическое и*]

$:=$ [логическое умножение*]

\subset логическая связка*

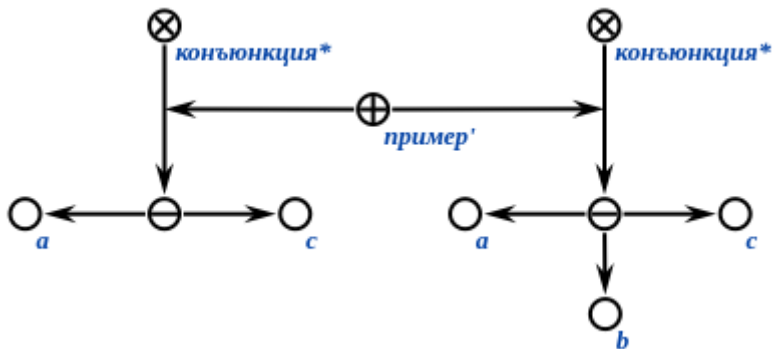
\in неориентированное отношение

\in класс связей разной мощности

\Rightarrow пояснение*:

[**конъюнкция*** – это множество конъюнктивных *высказываний*, каждое из которых истинно в рамках некоторой *формальной теории* только в том случае, когда все его компоненты истинны в рамках этой же *формальной теории*. **конъюнкция*** атомарных формул может быть заменена на атомарную формулу, полученную путём объединения исходных атомарных формул.]

Конъюнкция





Дизъюнкция

*дизъюнкция**

$:=$ [логическое или*]

$:=$ [логическое сложение*]

$:=$ [включающее или*]

\subset *логическая связка**

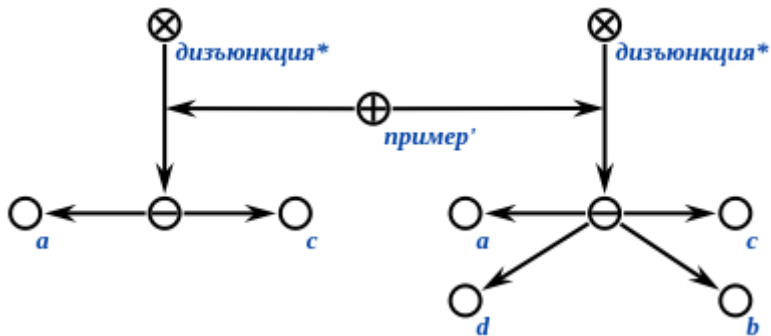
\in *неориентированное отношение*

\in *класс связок разной мощности*

\Rightarrow *пояснение**:

[*дизъюнкция** – это множество дизъюнктивных *высказываний*, каждое из которых истинно в рамках некоторой *формальной теории* только в том случае, когда хотя бы один его компонент является истинным в рамках этой же *формальной теории*.]

Дизъюнкция





Отрицание

*отрицание**

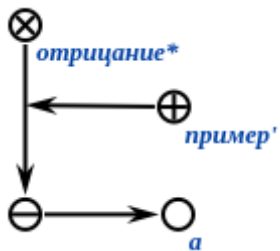
⊂ *логическая связка**

⊂ *синглетон*

⇒ *пояснение**:

[*отрицание** – это множество *высказываний* об отрицании, каждое из которых истинно в рамках некоторой *формальной теории* только в том случае, когда его единственный элемент является ложным в рамках этой же *формальной теории*.]

Отрицание





Строгая дизъюнкция

*строгая дизъюнкция**

$:=$ [сложение по модулю 2*]

$:=$ [исключающее или*]

$:=$ [альтернатива*]

\subset *логическая связка**

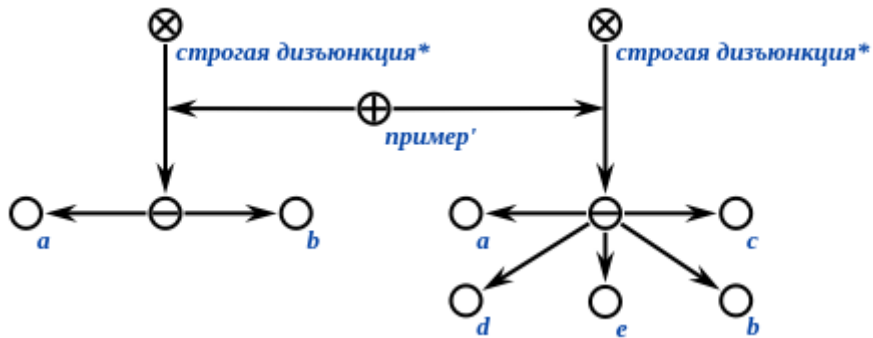
\in *неориентированное отношение*

\in *класс связок разной мощности*

\Rightarrow *пояснение**:

[*строгая дизъюнкция** – это множество строго дизъюнктивных высказываний, каждое из которых истинно в рамках некоторой формальной теории только в том случае, когда ровно один его компонент является истинным в рамках этой же формальной теории.]

Строгая дизъюнкция





Импликация

*импликация**

$:=$ [логическое следование*]

\subset логическая связка*

\in бинарное отношение

\in ориентированное отношение

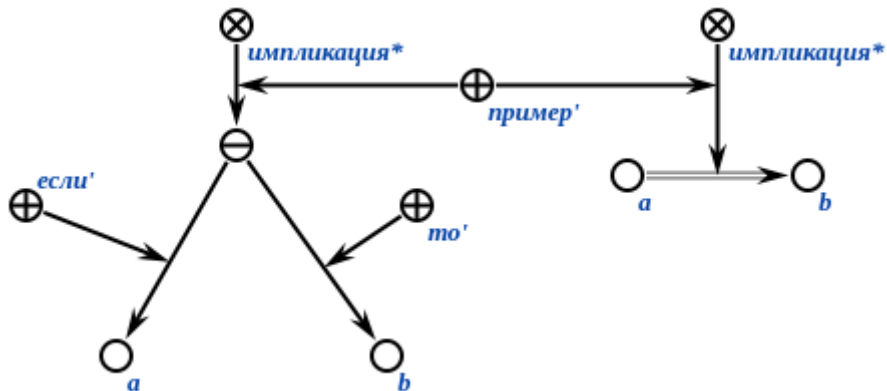
\Rightarrow пояснение*:

[импликация* – это множество имплицативных *неатомарных высказываний*, каждое из которых состоит из посылки (первый компонент *высказывания*) и следствия (второй компонент *высказывания*).

Каждое имплицативное *высказывание* ложно в рамках некоторой *формальной теории* в том случае, когда его посылка истинна, а заключение ложно в рамках этой же *формальной теории*. В других случаях такое *высказывание* истинно.

По умолчанию на все переменные, входящие в обе части высказывания об **импликации*** (или хотя бы одну из *подформул** каждой части) неявно накладывается квантор *всеобщности**, при условии, что эти переменные не связаны другим *квантором*, указанным явно.]

Импликация





Эквиваленция

эквиваленция*

$:=$ [эквивалентность*]

\subset логическая связка*

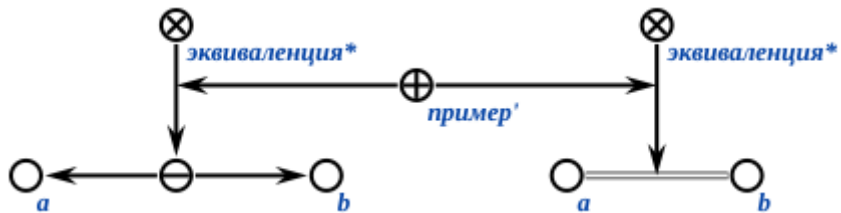
\in бинарное отношение

\in неориентированное отношение

\Rightarrow пояснение*:

[**эквиваленция*** – это множество *высказываний* об эквивалентности, каждое из которых истинно в рамках некоторой *формальной теории* только в тех случаях, когда оба его компонента одновременно либо истинны в рамках этой же *формальной теории*, либо ложны.]

По умолчанию на все переменные, входящие в обе части высказывания об **эквиваленции*** (или хотя бы одну из *подформул** каждой части) неявно накладывается квантор *всеобщности**, при условии, что эти переменные не связаны другим *квантором*, указанным явно.]



Таблицы истинности логических функций



X	Y	$\neg X$	$X \wedge Y$	$X \vee Y$	$X \oplus Y$	$X \rightarrow Y$	$X \leftrightarrow Y$
0	0	1	0	0	0	1	1
0	1	1	0	1	1	1	0
1	0	0	0	1	1	0	0
1	1	0	1	1	0	1	1



Равносильность логических формул

Две формулы логики A и B называются равносильными, если они принимают одинаковые логические значения при любом наборе значений входящих в формулы элементарных высказываний (переменных).

Обозначение: $A \iff B$

$X \wedge X \iff X$	$X \vee X \iff X$
$X \wedge \top \iff X$	$X \vee \top \iff \top$
$X \wedge \perp \iff \perp$	$X \vee \perp \iff X$
$X \wedge \neg X \iff \perp$	$X \vee \neg X \iff \top$
$\neg(\neg X) \iff X$	$X \vee (Y \wedge X) \iff X$
$X \wedge (Y \vee X) \iff X$	$X \leftrightarrow Y \iff (X \leftarrow Y) \wedge (Y \leftarrow X)$
$X \leftarrow Y \iff \neg X \vee Y$	$\neg(X \wedge Y) \iff \neg X \vee \neg Y$
$\neg(X \vee Y) \iff \neg X \wedge \neg Y$	$X \wedge Y \iff \neg(\neg X \vee \neg Y)$
$X \vee Y \iff \neg(\neg X \wedge \neg Y)$	



Законы де Моргана

$$\neg(A \wedge B) \iff (\neg A \vee \neg B)$$

$$\neg(A \vee B) \iff (\neg A \wedge \neg B)$$

тавтология

⇒ *пояснение**:

[*тавтология* – это логическая формула, которая является либо только истинной, либо только ложной в рамках всех формальных теорий, в которых можно установить ее истинность или ложность.

тавтология – это такая логическая формула, которая является либо общезначимой логической формулой, либо противоречивой логической формулой.]



Общезначимая логическая формула

общезначимая логическая формула

\coloneqq [тождественно истинная логическая формула]

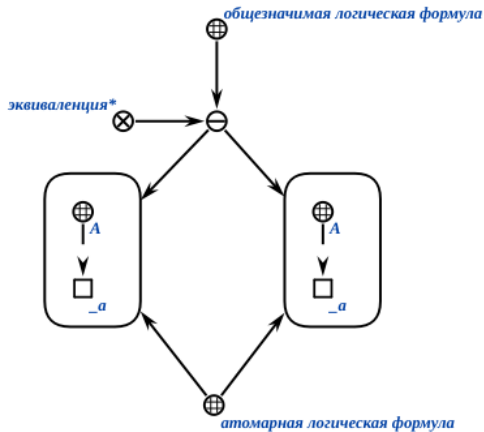
\subset *выполнимая логическая формула*

\subset *тавтология*

\Rightarrow *пояснение**:

[*общезначимая логическая формула* – это логическая формула, для которой не существует *формальной теории*, в рамках которой она была бы ложной с учетом истинности и ложности всех ее *под-формул** в рамках этой же *формальной теории*.]

Общезначимая логическая формула





Противоречивая логическая формула

противоречивая логическая формула

\coloneqq [тождественно ложная логическая формула]

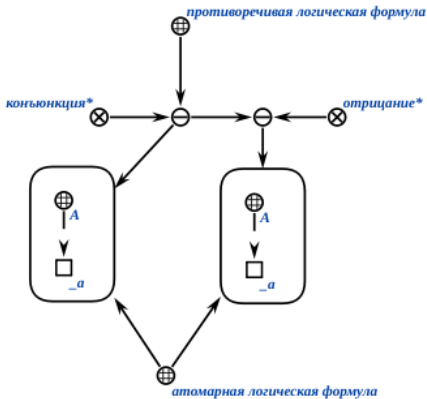
\subset невыполнимая логическая формула

\subset тавтология

\Rightarrow пояснение*:

[*противоречивая логическая формула* – это логическая формула, для которой не существует *формальной теории*, в рамках которой она была бы истинной с учетом истинности и ложности всех ее *под-формул** в рамках этой же *формальной теории*.]

Противоречивая логическая формула





Нейтральная логическая формула

нейтральная логическая формула

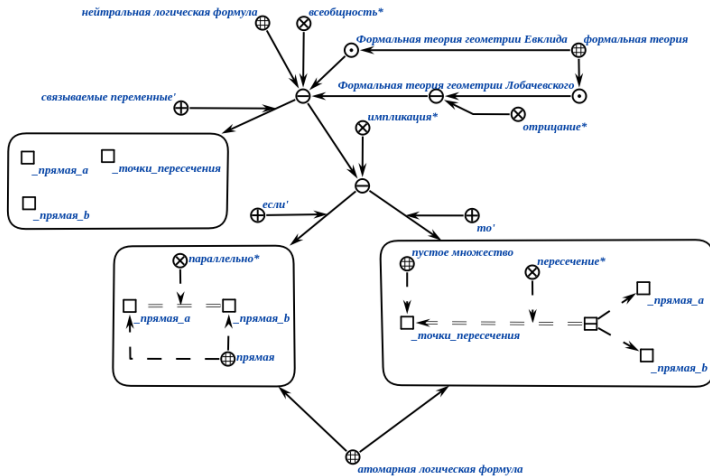
\subset *выполнимая логическая формула*

\Rightarrow *пояснение**:

[*нейтральная логическая формула* – это логическая формула, для которой существует хотя бы одна *формальная теория*, в рамках которой эта формула ложна, и хотя бы одна *формальная теория*, в рамках которой эта формула истинна.]



Нейтральная логическая формула



Непротиворечивая логическая формула



непротиворечивая логическая формула

$:=$ [выполнимая логическая формула]

\Rightarrow *пояснение**:

[*непротиворечивая логическая формула* – это логическая формула, для которой существует хотя бы одна *формальная теория*, в рамках которой эта формула истинна.]

\Leftarrow *объединение**:

- { • *нейтральная логическая формула*
- *общезначимая логическая формула*
- }



Необщезначимая логическая формула

необщезначимая логическая формула

\coloneqq [невыполнимая логическая формула]

\Rightarrow *пояснение**:

[*необщезначимая логическая формула* – это логическая формула, для которой существует хотя бы одна *формальная теория*, в рамках которой эта формула ложна.]

\Leftarrow *объединение**:

- *нейтральная логическая формула*
- *противоречивая логическая формула*

квантор

\subset логическая связка*

\Rightarrow пояснение*:

[квантор – это *отношение*, каждая связка которой задает истинность множества *логических формул*, входящих в ее состав, при выполнении дополнительных условий, связанных с некоторыми из переменных, входящих в состав этих *логических формул*.

Будем говорить, что переменные связаны **квантором** или попадают под область действия данного **квантора** (имея в виду конкретную связку конкретного **квантора**).

В состав каждой связки каждого **квантора** входит *атомарная формула*, являющаяся *тривиальной структурой*, в которой перечислены переменные, связанные данным **квантором**.]



Квантор всеобщности

*всеобщность**

$:=$ [квантор всеобщности*]

$:=$ [квантор общности*]

\in *квантор*

\in *ориентированное отношение*

\in *класс связок разной мощности*

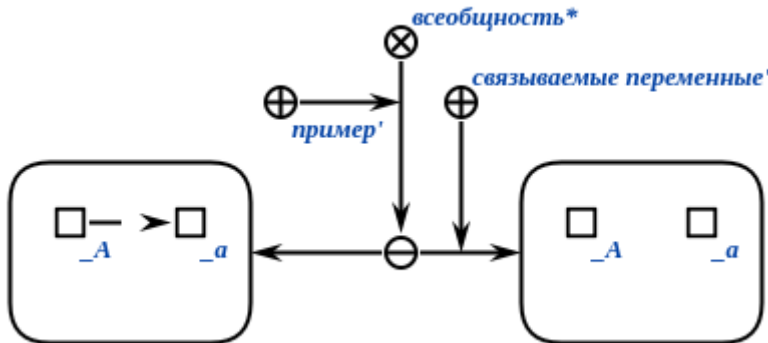
\Rightarrow *пояснение**:

[всеобщность – это *квантор*, для каждой связки которого, истинной в рамках некоторой *формальной теории*, выполняется следующее утверждение: все формулы, входящие в состав этой связки истинны в рамках этой же *формальной теории* при всех (любых) возможных значениях всех элементов множества *связываемых переменных'* входящего в эту связку.

Каждая связка *квантора всеобщности** может быть представлена как *конъюнкция** (потенциально бесконечная) исходных *логических формул*, входящих в состав этой связки, в каждой из которых все *связанные переменные'* заменены на их возможные значения.

Квантор *всеобщности** зачастую обозначается " \forall " и читается как "для всех", "для каждого", "для любого" или "все", "каждый", "любой".]

Квантор всеобщности





Квантор существования

формула существования

$:=$ [существование*]

\Leftarrow разбиение*:

- { • атомарная логическая формула
- неатомарное существование*
- }



Неатомарное существование

*неатомарное существование**

$:=$ [квантор неатомарного существования*]

\in *квантор*

\in *ориентированное отношение*

\in *класс связок разной мощности*

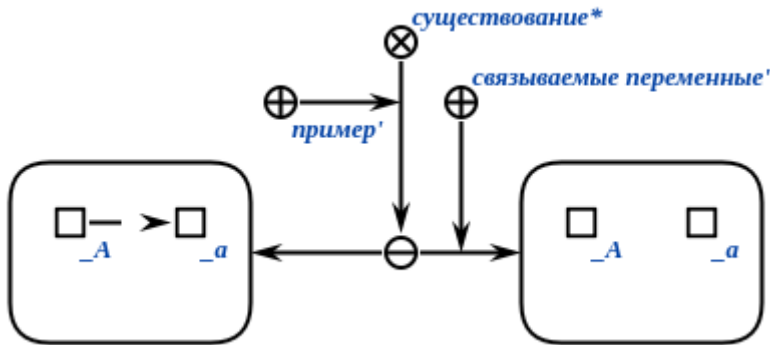
\Rightarrow *пояснение**:

[неатомарное существование* – это *квантор*, для каждой связки которого, истинной в рамках некоторой *формальной теории*, выполняется следующее утверждение: существуют значения всех элементов множества *связываемых переменных*['] входящего в эту связку, такие, что все формулы, входящие в состав этой связки истинны в рамках этой же *формальной теории*.

Каждая связка *квантора неатомарное существование** может быть представлена как *дизъюнкция** (потенциально бесконечная) исходных *логических формул*, входящих в состав этой связки, в каждой из которых все *связанные переменные*['] заменены на их возможные значения.

Квантор *существования** зачастую обозначается "∃" и читается как "существует", "для некоторого", "найдется".]

Неатомарное существование





Единственное существование

единственное существование

$:=$ [однократное существование]

$:=$ [формула существования и единственности]

\Rightarrow *пояснение**:

[*единственное существование* зачастую обозначается " $\exists!$ " и читается как "существует и единственный".]



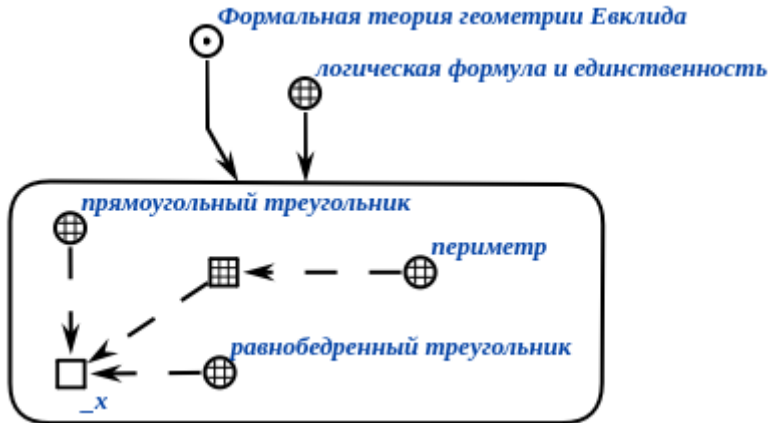
Единственное существование

логическая формула и единственность

- \subset логическая формула
- \subset единственное существование
- \Rightarrow пояснение*:

[Каждый элемент множества *логическая формула и единственность* представляет собой *логическую формулу* (атомарную или неатомарную), для которой дополнительно уточняется, что при ее интерпретации на некоторой предметной области существует только один набор значений переменных, входящих в эту формулу (или ее подформулы*), при котором указанная логическая формула истинна в рамках *формальной теории*, в которую входит данная *предметная область*.]

Единственное существование





Двойственность кванторов

Двойственность – содержательное понятие, применяемое в логике и математике всякий раз, когда между двумя группами понятий установлено взаимно-однозначное соответствие так, что замена понятий одной группы на соответствующие понятия др. группы каждый раз переводит истинные высказывания в истинные высказывания.

Двойственность кванторов:

$$\neg(\forall x A(x)) \iff \exists x(\neg A(x))$$

$$\neg(\exists x A(x)) \iff \forall x(\neg A(x))$$



Связываемые переменные

связываемые переменные'

\in ролевое отношение

\Rightarrow пояснение*:

[*связываемые переменные'* – это ролевое отношение, которое связывает связку конкретного квантора с множеством переменных, которые связаны этим квантором.]

открытая логическая формула

\Rightarrow пояснение*:

[*открытая логическая формула* – это логическая формула, в рамках которой (и всех ее подформул*) существует хотя бы одна переменная, не связанная никаким квантором.]

замкнутая логическая формула

\Rightarrow пояснение*:

[*замкнутая логическая формула* – это логическая формула, в рамках которой (и всех ее подформул*) не существует переменных, не связанных каким-либо квантором.]



Примеры