

# Модальная логика и Логики описания понятий

Евгений Евгеньевич Золин

старший научный сотрудник  
Кафедра математической логики и теории алгоритмов  
Механико математический факультет  
МГУ им. М.В.Ломоносова

Встреча со студентами 2 курса, 15 марта 2018 года

Как пишутся **модальные формулы**:

- начинаем с переменных:  $p_0, p_1, \dots$
- используем обычные (булевы) связки:

$$\neg A, \quad (A \wedge B), \quad (A \vee B), \quad (A \rightarrow B), \quad \dots$$

Как пишутся **модальные формулы**:

- начинаем с переменных:  $p_0, p_1, \dots$

- используем обычные (булевы) связки:

$$\neg A, \quad (A \wedge B), \quad (A \vee B), \quad (A \rightarrow B), \quad \dots$$

- используем две «модальности»:

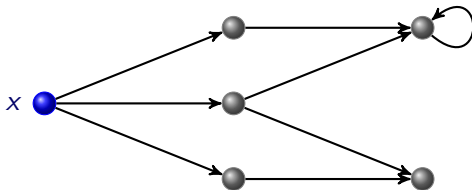
$\Box A$  (читается: «необходимо  $A$ »)

$\Diamond A$  (читается: «возможно  $A$ »)

Как понимаются **модальные формулы**:

- Граф (не обязательно конечный):

$F = (W, R)$ , где  $W$  — вершины,  $R \subseteq W \times W$  — рёбра.

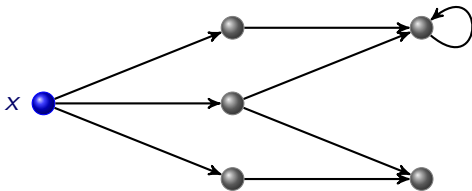


Как понимаются **модальные формулы**:

- Граф (не обязательно конечный):

$F = (W, R)$ , где  $W$  — вершины,  $R \subseteq W \times W$  — рёбра.

- Задаем: в каких вершинах какие переменные верны.

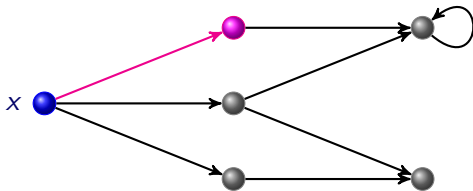


Как понимаются **модальные формулы**:

- Граф (не обязательно конечный):

$F = (W, R)$ , где  $W$  — вершины,  $R \subseteq W \times W$  — рёбра.

- Задаем: в каких вершинах какие переменные верны.
- формула  $\diamond A$  верна в точке  $x$ ,  
если  $A$  верна в **некоторой** точке  $y$ , такой что  $xRy$



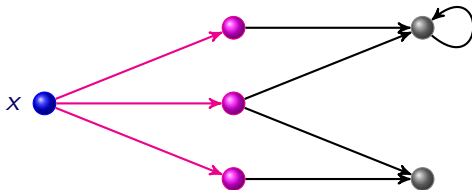
Как понимаются **модальные формулы**:

- Граф (не обязательно конечный):

$F = (W, R)$ , где  $W$  — вершины,  $R \subseteq W \times W$  — рёбра.

- Задаем: в каких вершинах какие переменные верны.

- формула  $\Box A$  верна в точке  $x$ ,  
если  $A$  верна в **каждой** точке  $y$ , такой что  $xRy$



# Модальная логика: Ключевые вопросы

- Какие модальные формулы **верны всегда**? (во всех графах)
  - Есть ли алгоритм, распознающий это свойство формул?
  - Каковы аксиомы и правила, из которых они все выводятся?
  - А если ограничиться лишь графами определенного вида?
- Какие **свойства графов** задаются модальными формулами?



- Какие модальные формулы **верны всегда**? (во всех графах)
  - Есть ли алгоритм, распознающий это свойство формул?
  - Каковы аксиомы и правила, из которых они все выводятся?
  - А если ограничиться лишь графами определенного вида?
- Какие **свойства графов** задаются модальными формулами?
- Те же вопросы, если **обогащаем** модальный язык, например:
  - формула вида  $\diamond^{\geq n} A$  верна в точке  $x$ ,  
если  $A$  верна в  $\geq n$  точках  $y$ , таких что  $xRy$
  - формула вида  $\diamond^{\infty} A$  верна в точке  $x$ ,  
если  $A$  верна в  $\infty$  числе точек  $y$ , таких что  $xRy$
  - формула вида  $\boxplus A$  верна в точке  $x$ ,  
если  $A$  верна во **всех** точках  $y$ , достижимых из точки  $x$   
за несколько переходов по  $R$ -рёбрам:  $xR \dots Ry$

- Верификация (доказательство правильности) программ
  - используется **логика времени**, **динамическая логика**

- Верификация (доказательство правильности) программ  
– используется **логика времени**, **динамическая логика**
- Описание законов доказуемости — **логики доказуемости**  
 $\Box A$  читаем: «утверждение  $A$  доказуемо в теории  $T$ »

- Верификация (доказательство правильности) программ  
– используется **логика времени**, **динамическая логика**
- Описание законов доказуемости — **логики доказуемости**  
 $\Box A$  читаем: «утверждение  $A$  доказуемо в теории  $T$ »
- **Пространственная логика**:  
формула  $\Box A$  верна в точке  $x$ , если формула  $A$  верна  
во всех точках  $y$  некоторой **окрестности** точки  $x$   
(в топологическом пространстве)

- Верификация (доказательство правильности) программ – используется **логика времени**, **динамическая логика**
- Описание законов доказуемости — **логики доказуемости**  
 $\Box A$  читаем: «утверждение  $A$  доказуемо в теории  $T$ »
- **Пространственная логика**:  
формула  $\Box A$  верна в точке  $x$ , если формула  $A$  верна во всех точках  $y$  некоторой **окрестности** точки  $x$  (в топологическом пространстве)
- **Логики описания понятий** — для представления знаний

# Логика описания понятий (description logics)

- Большое семейство языков, похожих на модальные
- Для записи знаний некоторой предметной области
- Для «вывода» новых знаний из уже имеющихся

# Логики описания понятий (description logics)

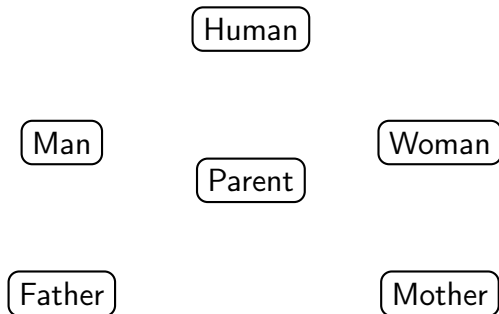
- Большое семейство языков, похожих на модальные
- Для записи знаний некоторой предметной области
- Для «вывода» новых знаний из уже имеющихся

Совокупность записанных знаний — **база знаний** или **онтология**.  
Их записывают на XML-языке **OWL** (Web Ontology Language)

## Примеры онтологий:

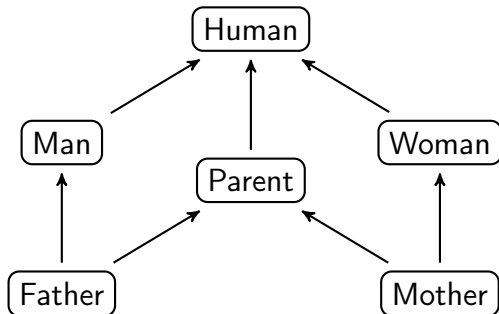
- **SNOMED CT** — **1,3 млн** связей между **300 000** мед. терминов (симптомы, диагнозы, мед. процедуры, структуры тела, типы организмов, хим. вещества, фармацевтические продукты и т.д.)
- **FMA** (Foundational Model of Anatomy) — более **2.1 млн** связей между **120 000** терминами анатомии
- **GO** (Gene Ontology) — около **50 000** терминов для аннотации генов всех биол. видов, миллионы связей между терминами

# Простейший пример онтологии

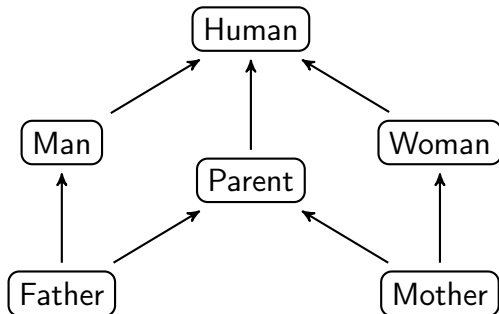




# Простейший пример онтологии

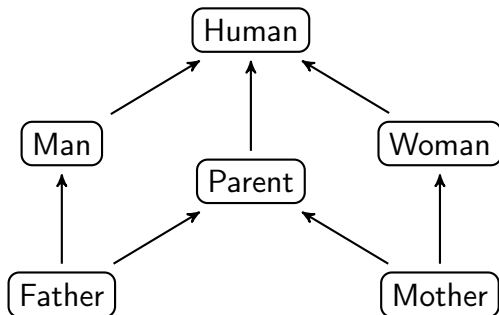


# Простейший пример онтологии



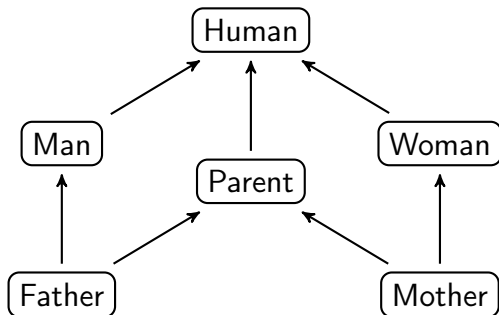
- Используя лишь  $\subseteq$ , пишем:  $\text{Parent} \subseteq \text{Human}$

# Простейший пример онтологии



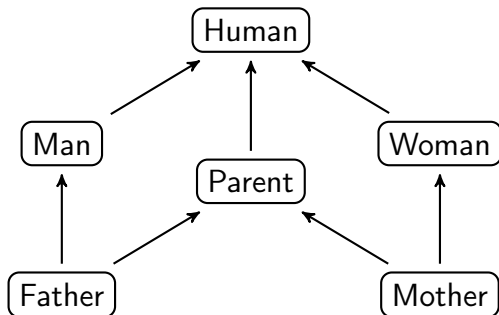
- Используя лишь  $\subseteq$ , пишем:  $\text{Parent} \subseteq \text{Human}$
- Используя  $\cap$ , можно писать:  $\text{Father} = \text{Man} \cap \text{Parent}$

# Простейший пример онтологии



- Используя лишь  $\subseteq$ , пишем:  $\text{Parent} \subseteq \text{Human}$
- Используя  $\cap$ , можно писать:  $\text{Father} = \text{Man} \cap \text{Parent}$
- Используя  $\exists$ , можно писать:  $\text{Father} = \text{Man} \cap \exists \text{hasChild.Human}$

# Простейший пример онтологии



- Используя лишь  $\subseteq$ , пишем:  $\text{Parent} \subseteq \text{Human}$
- Используя  $\cap$ , можно писать:  $\text{Father} = \text{Man} \cap \text{Parent}$
- Используя  $\exists$ , можно писать:  $\text{Father} = \text{Man} \cap \exists \text{hasChild.Human}$
- А это уже фактически модальная формула:  $F \leftrightarrow (M \wedge \diamond H)$ .

# Редактор онтологий Protégé

The screenshot displays the Protégé ontology editor interface. The browser address bar shows the URL: <http://www.co-ode.org/ontologies/pizza/pizza.owl>. The menu bar includes File, Edit, Reasoner, Tools, Refactor, Tabs, View, Window, and Help. The toolbar contains navigation and editing icons. The main window is divided into two panes:

- Left Pane: Asserted Class Hierarchy: Pizza**
  - Thing
    - DomainConcept
      - Country
      - Food
        - IceCream
        - Pizza**
          - CheeseyPizza
          - InterestingPizza
          - MeatyPizza
          - NamedPizza
          - NonVegetarianPizza
          - RealltalianPizza
          - SpicyPizza
          - SpicyPizzaEquivalent
          - ThinAndCrispyPizza
          - VegetarianPizza
          - VegetarianPizzaEquivalent
          - VegetarianPizzaEquivalent1
        - PizzaBase
        - PizzaTopping
      - ValuePartition

- Right Pane: OWL Viz: Pizza**
- Buttons: Show class, Show children, Show parents, Show all classes, Hide class, Hide children, Hide classes
- Model tabs: Asserted model, Inferred model
- Diagram: A network of nodes representing classes and their relationships. The 'Pizza' node is highlighted with a blue box. It is connected to various other nodes via 'is-a' relationships, including: Hot, Medium, Mild, IceCream, PizzaBase, PizzaTopping, CheeseyPizza, ThinAndCrispyPizza, NamedPizza, SpicyPizzaEquivalent, VegetarianPizzaEquivalent1, VegetarianPizza, NonVegetarianPizza, InterestingPizza, QuattroFormaggi, Napolitana, Rosa, PolloAdAstra, Soho, Margherita, Fiorentina, and LaReine.

# Редактор онтологий Protégé

The screenshot displays the Protégé 3.3 beta ontology editor. The main window title is "newspaper Protégé 3.3 beta (file:C:\Program%20Files\Protege\_3.3\_beta%20(build%20414)\examples\newspaper\news...". The menu bar includes File, Edit, Project, Window, Tools, and Help. The toolbar contains various icons for file operations and editing. The interface is divided into several panes:

- CLASS BROWSER:** Shows the class hierarchy for the project "newspaper". The hierarchy includes: :THING, :SYSTEM-CLASS, Author, Content, Layout\_info, Library (1), Newspaper (6), Organization (1), and Person.
- INSTANCE BROWSER:** Shows the instance "San Jose Mercury News" for the class "Organization".
- INSTANCE EDITOR:** Displays the instance "San Jose Mercury News" (instance of Organization, internal...). It shows a list of roles: Columnist, Editor, Reporter, Salesperson, Manager, and Director. A legend on the left maps colors to these roles: Columnist (pink), Editor (blue), Reporter (green), Salesperson (yellow), Manager (cyan), and Director (grey).
- Diagram:** A network diagram showing relationships between instances. The instances are: Tim (Director, grey), Mary (Manager, cyan), Joe (Manager, cyan), Jane (Manager, cyan), Mr. Science (Editor, blue), Chief Honcho (Editor, blue), Kelly (Columnist, pink), Kim (Columnist, pink), Joe Schmo (Reporter, green), Sports Nut (Reporter, green), and Ms Gardiner (Editor, blue). Relationships are: Tim directs Mary, Joe, and Jane; Mary manages Kelly and Kim; Joe manages Mr. Science; Jane manages Chief Honcho; Mr. Science is responsible for Joe Schmo; Chief Honcho is responsible for Sports Nut and Ms Gardiner.

# Редактор онтологий Protégé

The screenshot displays the Protégé ontology editor interface. The top menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, and Help. The address bar shows the URL: <http://www.semanticweb.org/ontologies/2011/9/Ontology131958817251.owl>. The toolbar contains various icons for navigation and editing.

The left sidebar shows the **Class Hierarchy** for **ResearchProfile**, listing the following classes:

- Thing
- ResearchProfile
- GrantSponsor
- ProposalDueDate
- RequiredFacilities
- RequiredFunding
- ResearchTopic
- Researcher

The main workspace displays a complex ontology graph. The central node is **ResearchProfile**, which is highlighted with a blue border. It is connected to several other nodes:

- RequiredFunding**: connected to values like \$150,000, \$4,200,000, and \$1,000,000.
- Researcher**: connected to individuals like Eric and David.
- GrantSponsor**: connected to entities like National Science Foundation, Fusion, and Department of Energy.
- RequiredFacilities**: connected to entities like Biological Facilities and Microsystems Technology Office.
- ResearchTopic**: connected to entities like Computational Science and Biological Sciences.
- ProposalDueDate**: connected to dates like January 17, 2012, Jan 20, 2012, Dec 04, 2011, and Sep 30, 2008.

At the bottom right, there is a status bar with the text: "To use the reasoner click Reasoner->Start reasoner" and a checked checkbox for "Show Inferences".



# Редактор онтологий Protégé

The screenshot displays the Protégé ontology editor interface. The main window shows a large, complex network graph with numerous nodes and directed edges. The nodes are color-coded according to a legend on the left, representing different types of ontology elements. The legend includes:

- Class Node (Yellow)
- Instance Node (Purple)
- Datatype Property Node (Green)
- Object Property Node (Blue)
- Property Node (Pink)
- Collection Node (Gold)
- Literal Node (Light Blue)
- Default Node (Red)

The graph is centered around a large purple node, with many other nodes radiating outwards, connected by pink arrows. A smaller overview graph is visible in the bottom-left corner of the main window. The interface includes a menu bar (File, Edit, Ontologies, Reasoner, Tools, Acl2actor, Tabs, View, Window, Help), a toolbar, and a left-hand pane showing a class hierarchy. The active ontology is 'doap' (http://usefulinc.com/ns/doap#), and the current view is 'doap:doap'. The status bar at the bottom indicates 'No Reasoner set: Select a reasoner from the Reasoner menu.' and 'Show Inferences'.

# Контактная информация

Пишите:

`ezolin@yandex.ru`

Читайте — включая задачи (темы курсовых) по МЛ и ДЛ:

<http://lpcs.math.msu.su/~zolin/>

Заходите:

спецкурс «[Дескрипционная логика](#)»  
(пятница 16:45, ауд. 12-26 Б)

Просеминар для студентов 1–2 курса:

пятница 16:45, ауд. 16-22.