Механизмы искусственного интеллекта

Е. А. Черкашин

ИДСТУ им. В. М. Матросова СО РАН

6 июля 2023, Иркутск

К определению понятия «Искусственный интеллект»

К ИИ относятся программы, реализующие задачи, решение которых потребовало бы от человека использования его *когнитивных способностей*, выполнения *творческих функций*[1].

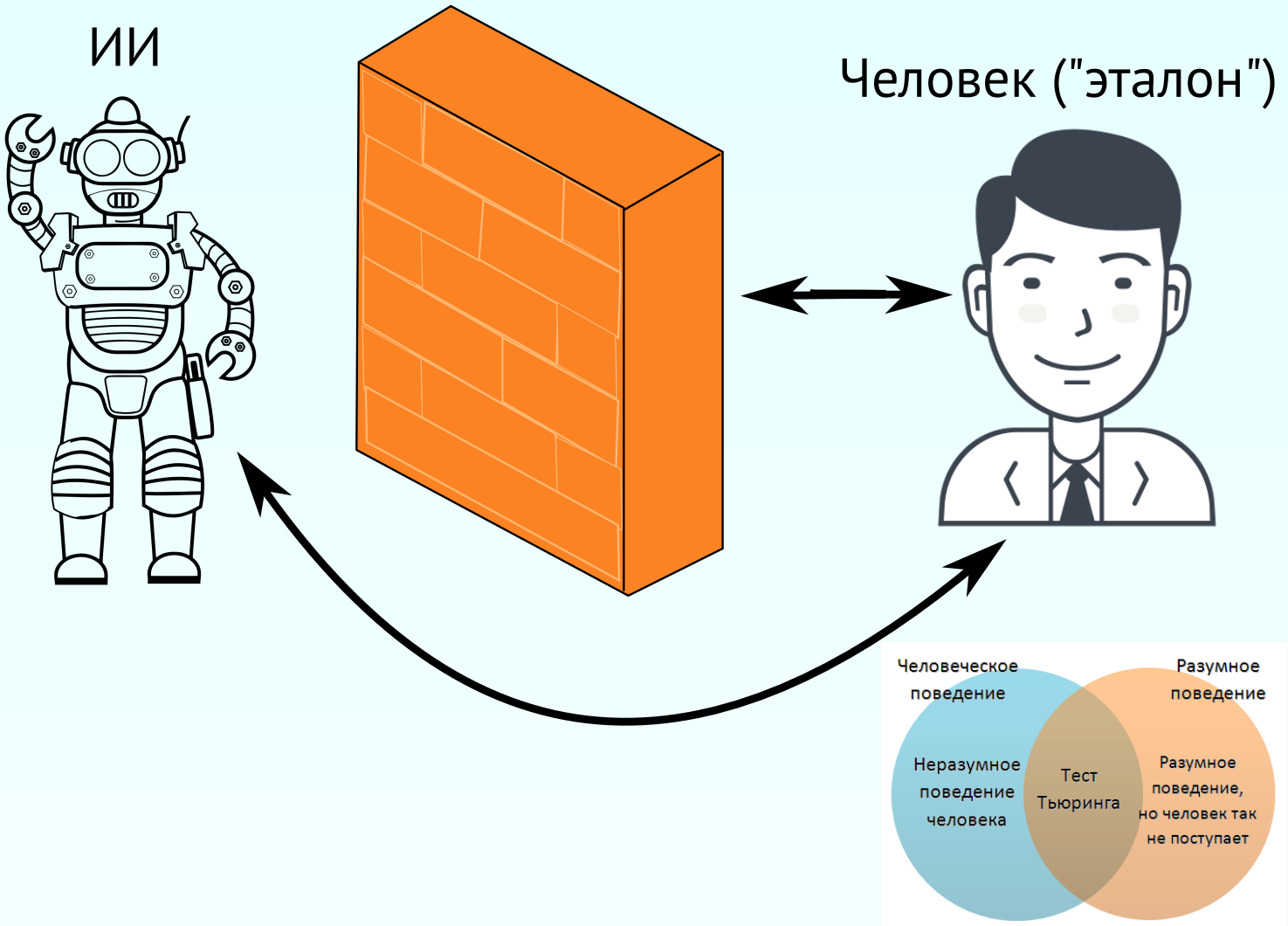
Определение из книги AIMA[2]:

	Рационально	**Как человек**
Рассуждать	○	○
Вести себя	○	○

[1] Справочник ИИ, Т.1, 1990

[2] P.Norvig, S.Russell. Artificial Intelligence Modern Approach

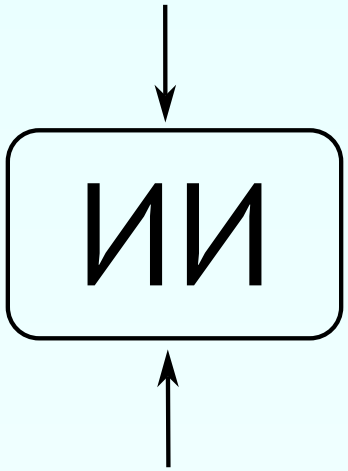
Тест Тьюринга



Если Человек в процессе общения с ИИ будет думать, что общается с Человеком, то ИИ обладает требуемым свойством.

ИИ как модель человека

Модулирование рассуждений ⟶ моделирование «математики»



Моделирование мозга ⟶ моделирование самоорганизующихся систем

Свойства и классы задач ИИ

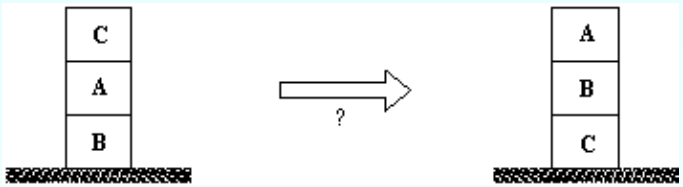
Свойства задач

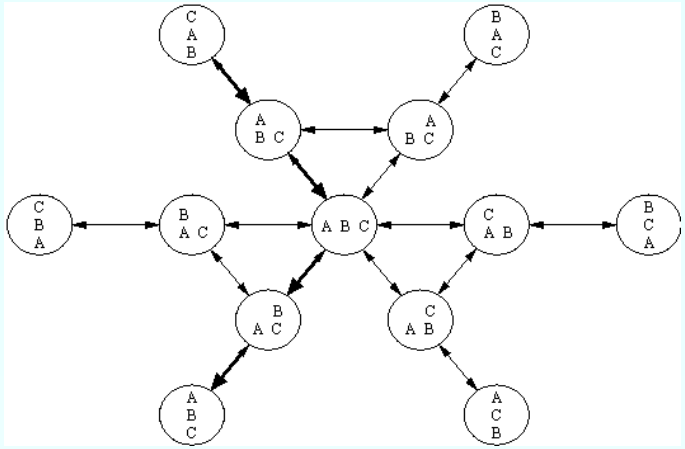
- Не существует алгоритма решения
- Обработка символьной информации
- Автоматизация принятия решения
- Решение – комбинация «возможностей»
- Обработка неполной и противоречивой информации

Классы задач

1. Восприятие и распознавание образов
2. Автоматическое доказательство теорем
3. Игры
4. Планирование действий (Problem Solving)
5. Понимание естественного языка, перевод
6. Логическое программирование
7. Экспертные системы
8. Интеллектные информационные системы
9. Восприятие и усвоение знаний (Machine learning)
10. Интеллектное управление [производством] ...
11. Робототехника (Robotics)
12. Системы поддержки принятия решений

Планирование действий (Problem Solving)





Формализованная постановка (SSG, State Space Graph)

- Структура данных для представления **состояния**
- Формализация **правил перехода** из состояния в состояние G
- Программирование **распознавателя *целевого состояния*** $R(v)$
- Задание **исходного состояния** I

$$SSG =< G, R(v), I >, G =< V, E >, v \in V.$$

Представление знаний

Естественно определить данные как некоторые сведения об отдельных объектах, а знания – о мире в целом.

Данные представляют информацию о существовании объектов, представляемых значениями признаков, а **знания** – информацию о существующих в мире закономерных связях между признаками и запрещающих некоторые другие сочетания свойств у объектов.

Данные – это информация о существовании объектов с некоторым набором свойств, **знания** – информация о несуществовании объектов с некоторым набором свойств.

Пусть $H(x) \Leftrightarrow$ «x является человеком», а $M(x)$ – «x – смертен».

«существует x, x является человеком»:

$$A_1 = \exists x H(x),$$

«не существует бессмертных людей»):

$$A_2 = \neg\exists x\big(H(x)\&\neg M(x)\big),$$

преобразуем во «все люди смертны»:

$$A_3 = \forall x\big(H(x) \to M(x)\big).$$

...все объекты x, обладающие свойством H, будут обладать свойством M.

Обработка информации «в математике»

Добавим информатики. Утверждение «Сократ – человек» представим как $H(s)$, где s – это Сократ. Теперь из **знания** A_3 и **исходного данного** $H(s)$ получим новую информацию о Сократе:

$$B = \left(H(s)\& \overbrace{\forall x(H(x) \to M(x))}^{A_3}\right) \to M(s).$$

Из того, что Сократ – человек и что все люди смертны, следует, что Сократ тоже смертен $M(s)$. Доказательство «от противного»:

1. Требуется **опровергнуть высказывание** $\neg B = \text{True}$
2. $B = \text{False} \vdash H(s)\&A_3 = \text{True}$, $M(s) = \text{False}$
3. $A_3 = \text{True} \vdash H(s) = \text{True}$, $A_3 = \text{True}$
4. Подставим s вместо x в A_3, получим $H(s) \to M(s)$
5. $H(s) = \text{True}, M(s) = \text{False} \vdash H(s) \to M(s) = \text{False}$
6. $H(s) \to M(s) = \text{False} \vdash A_3 = \text{False}$
7. Из п. 3 следует $A_3 = \text{True}$, а из п. 6 – $A_3 = \text{False}$
8. В формальной (математической) логике в п. 7 получено **противоречие**
9. Следовательно, $\neg B \neq \text{False}$, $B = \text{True}$
10. Из п. 9 следует $M(s) = \text{True}$ □ **(ЧТД, QED[3])**

[3] *Quod Erat Demonstrandum*, «что и требовалось показать», “which was to be demonstrated“

Сократим доказательство

Будем выписывать только **истинные высказывания**, т.е.
$M(s) \Leftrightarrow M(s) =$ True («Сократ смертен»),
$\neg M(s) \Leftrightarrow M(s) =$ False («Сократ бессмертен»)

$$B = \Big(H(s) \& \overbrace{\forall x(H(x) \to M(x))}^{A_3}\Big) \to M(s).$$

1. **Опровергаем** $\neg B$
2. $B \vdash H(s) \& A_3, M(s)$
3. $A_3 \vdash H(s), A_3$
4. $A_3\{s/x\} = H(s) \to M(s)$
5. $H(s), \neg M(s) \vdash \neg(H(s) \to M(s))$
6. $\neg(H(s) \to M(s)) \vdash \neg A_3$
7. Из п. 3 следует A_3, а из п. 6 – $\neg A_3$
8. В п. 7 получено **противоречие**
9. Следовательно, B
10. Из п. 9 $M(s)$ $\square$

Программа (теория)

```
h(s).
m(X) :- h(X).   % h(x) -> m(x).
```

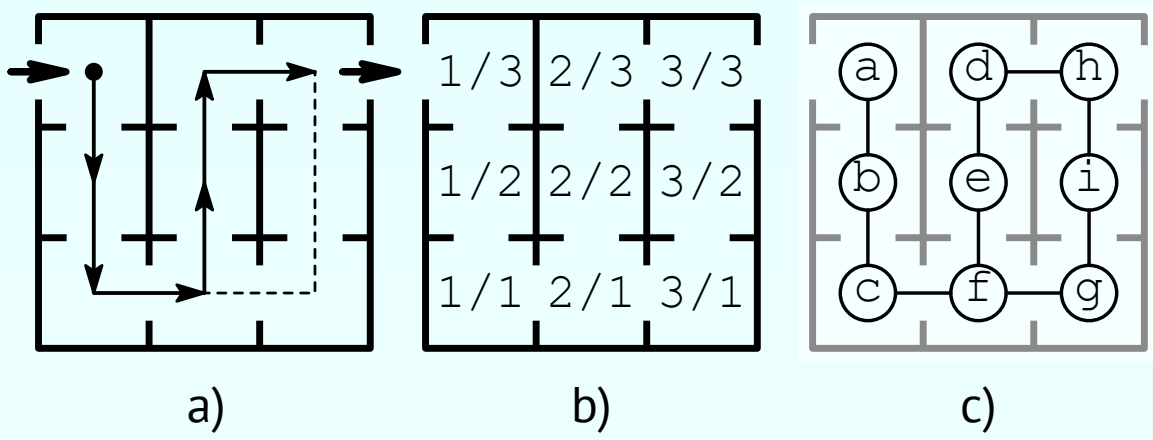
Консультация

```
?- [socrates].  % Загрузка теории из файла
true.

?- m(s).        % Запрос
true.
```

Объяснение вывода

```
?- trace.
true.

[trace]  ?- m(s).
   Call: (10) m(s) ?     % Запрос пользователя
   Call: (11) h(s) ?     % Применение m(X):-h(X)
                         % при {s/X}
   Exit: (11) h(s) ?     % Найден факт h(s).
   Exit: (10) m(s) ?     % следовательно m(s).
true.
```

Планирование действий



Теория лабиринта

```
% "Данные"

e(a,b). e(b,c). e(c,f).
e(f,e). e(e,d). e(d,h).
e(f,g). e(g,i). e(i,h).

% "Знания"
% Что такое "путь".

path(A,B) :- e(A,B).
path(A,B) :- e(A,C), path(C,B).
```

Запрос

```
?- path(a,h).
true ;
true ;
false.
```

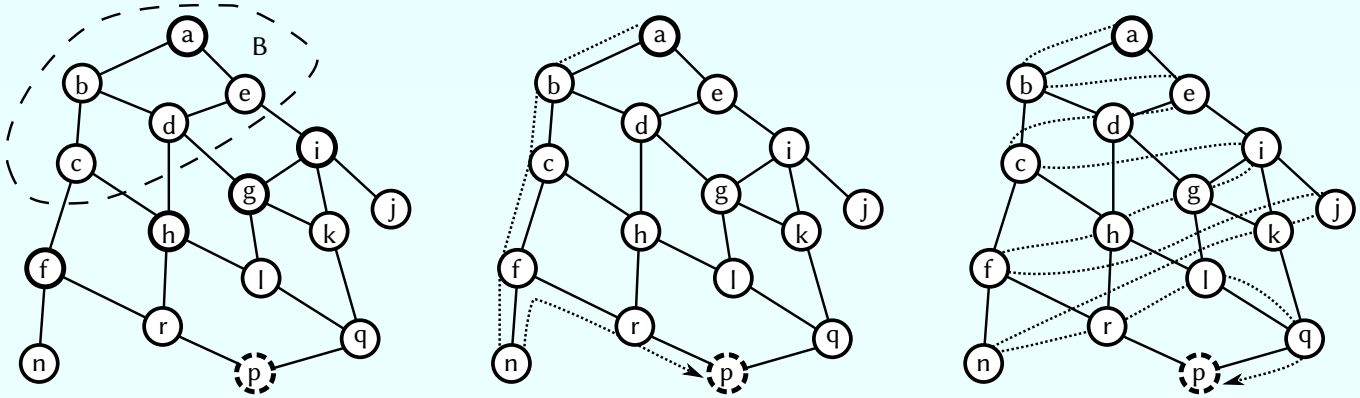
Конструктивное решение

```
path(A,B, [A-B]) :- e(A,B).
path(A,B, [A-C|T]) :- e(A,C), path(C,B,T).
```

Запрос

```
?- path(a,h,L).
L = [a-b, b-c, c-f, f-e, e-d, d-h] ;
L = [a-b, b-c, c-f, f-g, g-i, i-h] ;
false.
```

Стратегии поиск в SSG



a) b) c)

Поиск в глубину

```
dfs(V,[]):- r(V).
dfs(V,[V-N|T]):- \+ r(V), after(V,N), dfs(N,T).
r(h).
after(X,Y):- e(X,Y); e(Y,X).
```

```
?- dfs(a, S).
S = [a-b, b-c, c-f, f-e, e-d, d-h] ;
S = [a-b, b-c, c-f, f-e, e-d, d-e, e-d, d-h] ;
S = [a-b, b-c, c-f, f-e, e-d, d-e, e-d, d-e, ...]
```

Поиск в ширину

```
bfs([[X|T]|_],[X|T]):-r(X),!.    % (1)
bfs([[X|T]|Ways], S):-
    findall([Y,X|T],             % (2)
      (after(X,Y), \+ member(Y,[X|T])),
    L),
    append(Ways, L, NWays),      % (3)
    bfs(NWays, S).
```

```
?- bfs([[a]],S).
S = [h, d, e, f, c, b, a].
```

Использование дополнительной информации

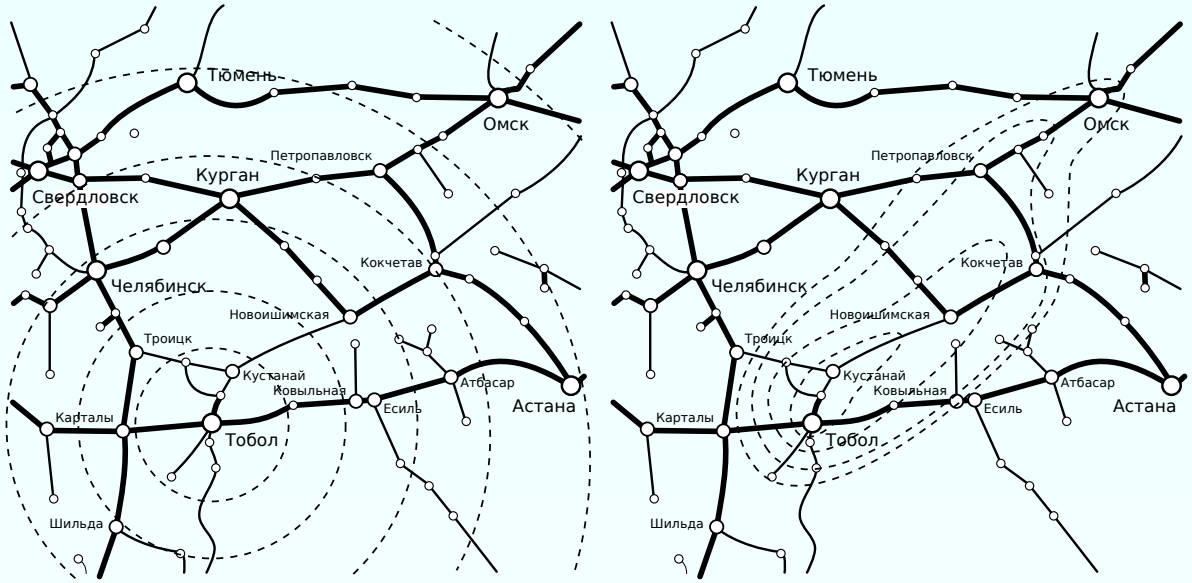
Зададим функцию f: $V \to R$ следующего вида:

$$f(x) = g(x) + r(x)$$

– стоимость кратчайшего пути (КП) через x, $g(x)$ – до x, $r(x)$ – КП до целевой вершины (не известен).
Оценка $r(x)$ снизу – $h(x)$, $h(x) \leqslant r(x)$.
Теперь

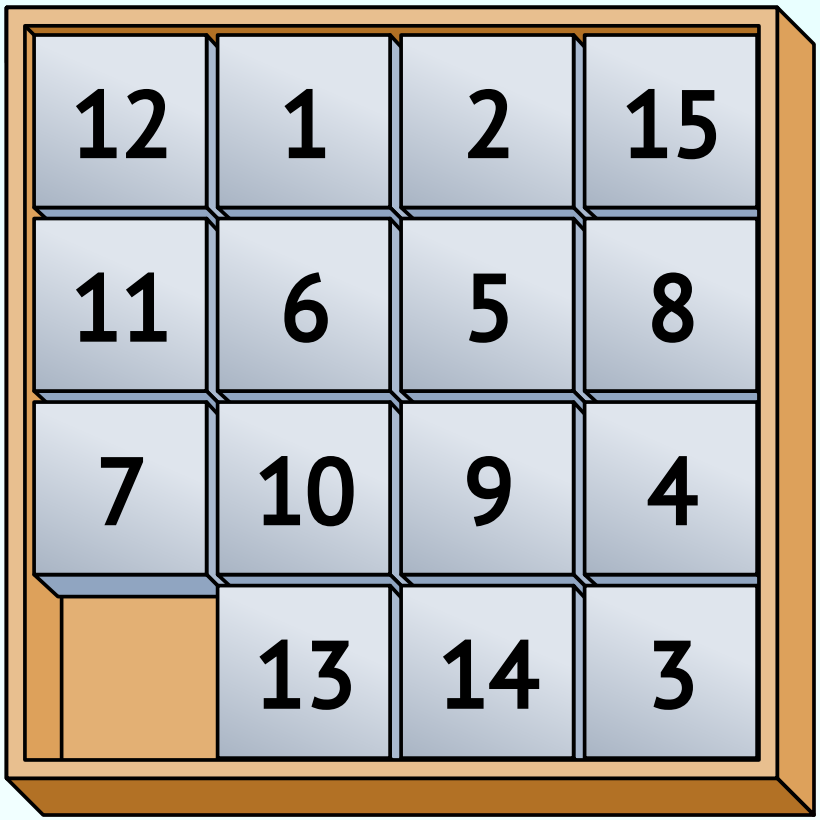
$$f(x) \leqslant g(x) + h(x).$$

```
bf1([_-s(G,[Target|T]) |_],% (1)
    Target-_,s(G,[Target|T])):-!.
bf1([_-s(G,[X|T])|Ways], Target-GPS, S):-
    Target\=X,
    findall(F1-s(G1,[Y,X|T]),
      after([X|T],G,GPS, Y,G1,F1), L),
    append(L, Ways, NWays), keysort(NWays,SNWays),
    bf1(SNWays,Target-GPS,S).
after([S|R],SG, GPS, T,TG, F):-
    transdist(S,T,D), % \+ member(T,[S|R]),
    TG is SG + D,
    geodist(T, GPS, GDist), % (2)
    F is TG + GDist.
bf(Start, Target, Sol):-
    geocode(Target, Lon, Lat, _),
    bf1([0-s(0,[Start])],
        Target-ll(Lon,Lat), Sol). % (3)
```



a) *b*)

Игра 15



Поиск решения с использованием эвристики

```
center% ./15-solve 100 20 1
Environment:
USE_HEURISTIC=1
Puzzle 15 solving program
State(00,00[00])
<---solution--->
0 Step ------------
 5  1  3  4
 9  2  6  8
13 10 15  0
14 12  7 11
x=2, y=3
State(00,16[16])
. . . . . .
17 Step ------------
 1  2  3  4
 5  6  7  8
 9 10 11 12
13 14 15  0
x=3, y=3
State(17,00[17])
Tested 111 states.
```

Без эвристики

```
. . . . . .
17 Step ------------
 1  2  3  4
 5  6  7  8
 9 10 11 12
13 14 15  0
x=3, y=3
State(17,00[17])
Tested 921299 states.
```

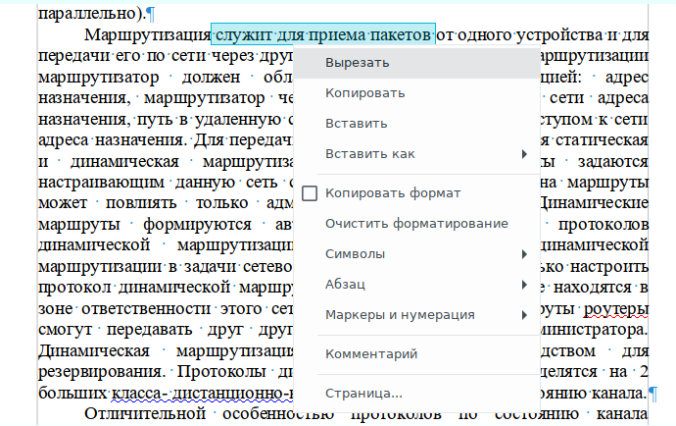
Понимания естественного языка: актуальность

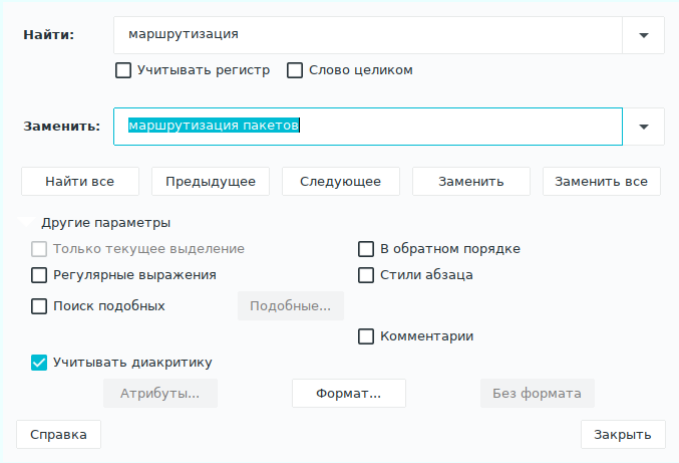
Понимание естественного языка (ЕЯ), перевод из одного ЕЯ на другой – **Направление ИИ**. Решаются следующие задачи:

1. Анализ текстов, помещение изъятой информации в базу данных:
   - изготовление шаблонов документов, отчетов;
   - синтез структур данных для ИС;
   - заполнение баз данных ИС и т.п.;
2. Ведение диалога с пользователем:
   - идентификация моделей и планирование действий (интеллектные пользовательские интерфейсы);
   - приобретение знаний (оболочки экспертных систем);
3. Управление приложением:
   - запросы на естественном языке к базам данных;
   - внесение изменений в данные;
   - командное управление («Проветрить квартиру»).

Графический пользовательский интерфейс

Специализирован на **операциях над отдельными частями** информационного объекта.

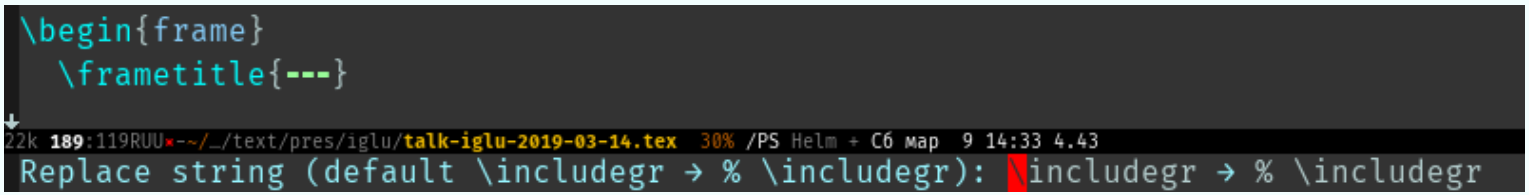




В операциях аргументы вводится в диалоговом окне (для операций с аргументами).

Более «умные» редакторы не используют контекстные операции (EMACS, VI, Visual Studio Code, Sublime, AutoCAD).

`Alt-X replace-string`, *Набирается как «Alt-X repl str»*



Язык как математическая модель

Семиотика (наука о знаках) делится на три раздела (Моррис)

- **Семантика** — отношение знака к объекту:
  Что значит *знак*?
- **Синтаксис** — отношение знаков между собой –
  как создаются *новые смыслы* (термины, суждения)
  комбинированием *знаков*.
- **Прагматика** — отношение знака к субъекту:
  Что *обозначает предложение*, что надо дальше делать? На какой конкретно вопрос и как надо отвечать?

Язык программирования

Синтаксис на уровне грамматики определяет корректные последовательности символов (операторы, структуры). Но синтаксическая правильность не гарантирует даже осмысленности программы.

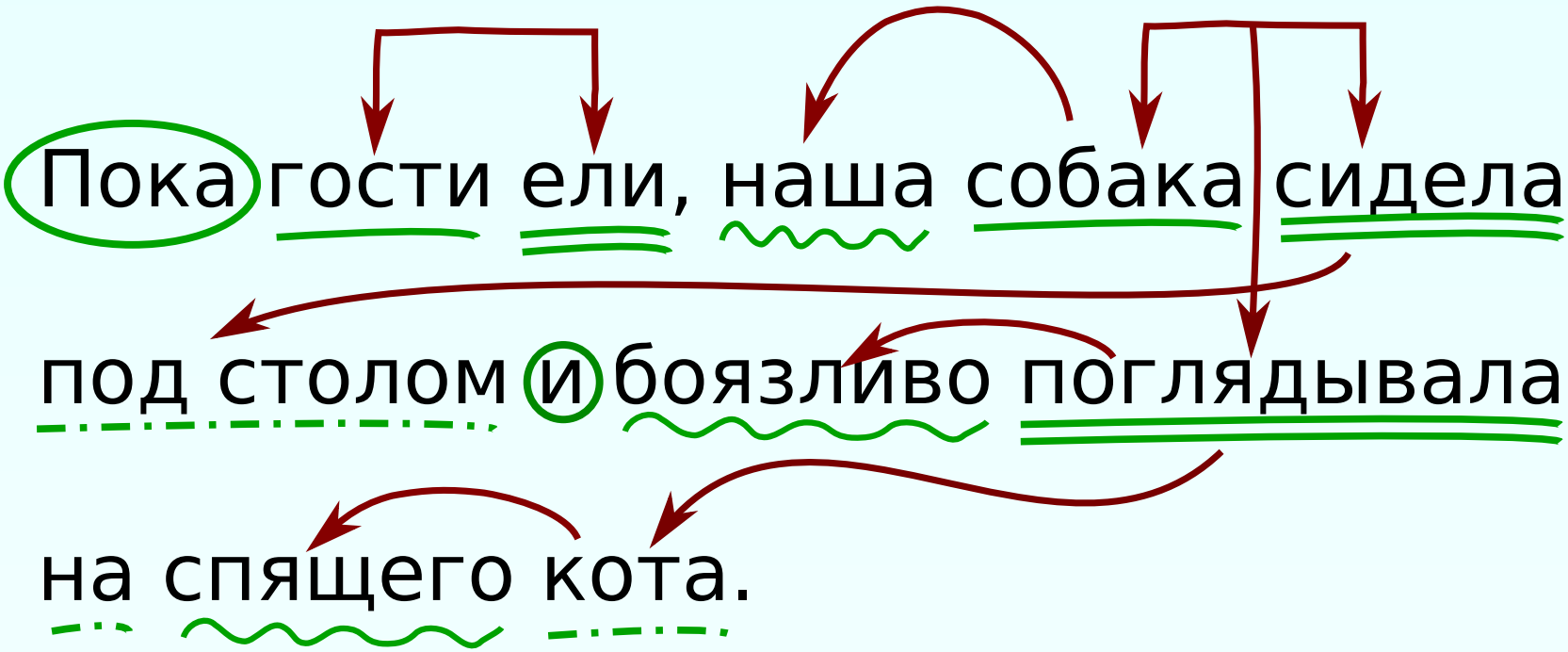
Семантика — это соответствие между синтаксически правильными программами и [вариантами] действий абстрактного исполнителя, то есть это смысл синтаксических конструкций.

Прагматика задает конкретизацию абстрактного вычислителя (конкретный процессор и др. ресурсы) для вычислительной системы. Стандарт языка программирования задаёт поведение вычислителя не полностью, конкретный транслятор языка переводит программу в конкретной машинный код на конкретную программно-аппаратную платформу.

Реализованный язык является прагматическим опосредованием абстрактной модели вычислений и ее реализацией на конкретном компьютере.

Цель программиста — получить нужный ему эффект в результате исполнения программы на конкретном оборудовании: трансляция и исполнение осуществляется на конкретных вычислителях.

Синтаксический разбор предложения



Грамматика

$$G = \langle T, N, \Sigma, R \rangle$$

T – множество терминальных символов (слова, буквы, IF, ELSE),
N – множество нетерминальных символов (обозначения, A, B, <noun>, <verb>), $T \cap N = \emptyset$,
Σ – стартовый символ (<программа>, <предложение>), $\Sigma \in N$,
R – множество правил грамматики $\mathrm{A} \to \mathrm{B}$,
$R \subset ((T \cup N)^* N (T \cup N)^*) \times (T \cup N)^*$.

Язык $L(G) = \{\Omega \in T^* | \Sigma \to^* \Omega\}$.

Вывод $\Sigma \to^* \Omega$

$$\Sigma \to \Sigma\mathrm{A} \quad \Sigma \to \mathrm{A}$$

$$\mathrm{A} \to b\Sigma e \quad \mathrm{A} \to be$$

Пример: $a =$ ` $(', \quad b =$ ` $)'$.

Σ	Σ
A	A
$b\Sigma e$	(Σ)
$b\Sigma\mathrm{A}e$	$(\Sigma\mathrm{A})$
$b\mathrm{A}\mathrm{A}e$	(AA)
$bbe\mathrm{A}e$	$(()\mathrm{A})$
$bbebee$	$(()())$

Типы грамматик

По иерархии Ноама Хомского, грамматики делятся на **четыре** типа, каждый последующий является более ограниченным подмножеством предыдущего (но и легче поддающимся анализу):

- Тип 0. Неограниченные грамматики — возможны любые правила;
- Тип 1. Контекстно-зависимые грамматики — левая часть может содержать один нетерминал, окруженный «контекстом»; сам нетерминал заменяется непустой последовательностью символов в правой части;
- Тип 2. Контекстно-свободные грамматики — левая часть состоит из одного нетерминала;
- Тип 3. Регулярные грамматики — более простые, распознаются конечными автоматами.

Грамматика языка программирования C

```
<translation-unit> ::= {<external-declaration>}*

<external-declaration> ::= <function-definition>
                    | <declaration>

<function-definition> ::= {<declaration-specifier>}* <declarator> {<declaration>}* <compound-statement>
                    | union

<struct-declaration> ::= {<specifier-qualifier>}* <struct-declarator-list>
. . . . . .
<specifier-qualifier> ::= <type-specifier>
                    | <type-qualifier>

<struct-declarator-list> ::= <struct-declarator>
                    | <struct-declarator-list> , <struct-declarator>

<struct-declarator> ::= <declarator>
                    | <declarator> : <constant-expression>
                    | : <constant-expression>
. . . . . .
<selection-statement> ::= if ( <expression> ) <statement>
                    | if ( <expression> ) <statement> else <statement>
                    | switch ( <expression> ) <statement>

<iteration-statement> ::= while ( <expression> ) <statement>
                    | do <statement> while ( <expression> ) ;
                    | for ( {<expression>}? ; {<expression>}? ; {<expression>}? ) <statement>

<jump-statement> ::= goto <identifier> ;
                    | continue ;
                    | break ;
                    | return {<expression>}? ;
```

Пример трансляции

```c
#include <stdio.h>

typedef unsigned long int ulint;

ulint fact (ulint n) {
    if (n==0) return 1;
    if (n==1) return 1;
    return n*fact(n-1);
}

int main() {
    ulint n = 10;
    printf("Factorial of %lu = %lu.\n",
        n, fact(n));
    return 0;
}
```

```
        .file   "fact.c"
        .text
        .globl  fact
        .type   fact, @function
fact:
.LFB11:
        movl    $1, %eax
        cmpq    $1, %rdi
        jbe     .L4
.L3:
        imulq   %rdi, %rax
        subq    $1, %rdi
        cmpq    $1, %rdi
        jne    .L3
.L4:
        ret
.LFE11:
        .section    .rodata.str1.1,"aMS",@progbits,1
.LC0:
        .string     "Factorial of %lu = %lu.\n"
        .section    .text.startup,"ax",@progbits
        .globl  main
        .type   main, @function
main:
.LFB12:
        ;; . . . . . .
        ret
        .cfi_endproc
.LFE12:
        .size   main, .-main
        .ident  "GCC: (GNU) 8.2.1 20181127"
        .section    .note.GNU-stack,"",@progbits
```

Синтаксический разбор предложения

Корова трясет хвостом.

A cow shakes the tail.

```
% [a, cow, shakes, the, tail]
```

Грамматика:

- Множество **терминальных** символов – {a,b,c,...,z}. На самом деле, $\Sigma = \{a, cow, shakes, walks, ...\}$.
- Множество **нетерминальных** символов – `<sentence>`, `<noun>`, `<verb>`, …
- Стартовый символ – `<sentence>`.
- **Правила** упрощенного английского языка:

$$\langle sentence \rangle \rightarrow \langle noungroup \rangle \langle verbgroup \rangle \quad (1)$$

$$\langle noungroup \rangle \rightarrow \langle determinant \rangle \langle noun \rangle \quad (2)$$

$$\langle verbgroup \rangle \rightarrow \langle verb \rangle \langle noungroup \rangle \quad (3)$$

$$\langle noun \rangle \rightarrow cow \mid tail \mid \ ... \quad (4)$$

$$\langle verb \rangle \rightarrow walks \mid shakes \mid \ ... \quad (5)$$

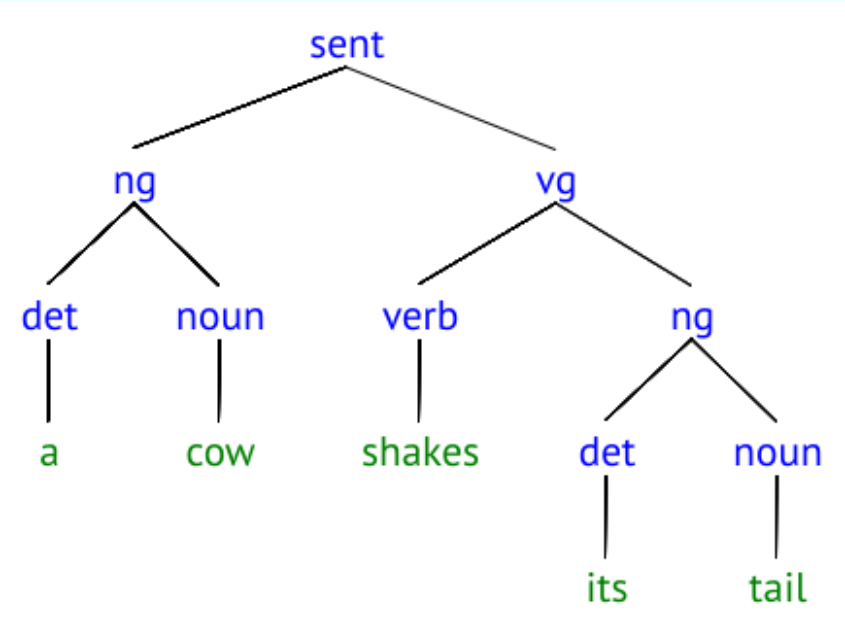
$$\langle determinant \rangle \rightarrow a \mid the \mid \varepsilon \mid my \mid \ ... \quad (6)$$

Разбор предложения на Prolog

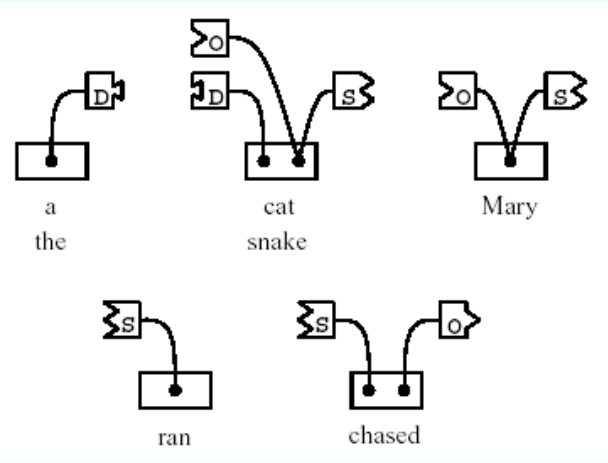
```
% <sent> ::= <noun group> <verb group>
% <noun group> ::= <det> <noun>
% <verb group> ::= <verb> <noun group>
% <det> ::= a | the | my | yours | its
% <noun> ::= cow | tail | body
% <verb> ::= walks | shakes | moves

% ?- t(sent, [a,cow,shakes,its,tail], [], Tree).
```

```
noun(cow). noun(tail). noun(body).
%noun(X):-member(X,[cow, tail, body]).
det(X):-member(X,[a, the, my, yours, its]).
verb(X):-member(X,[walks, shakes, moves]).

t(sent, I, O, sent(NG,VG)):-
    t(ng, I, R, NG),
    t(vg, R, O, VG).
t(ng, I, O, ng(Det,N)):-
    t(det,I,R, Det),
    t(noun,R,O,N).
t(vg, I, O, vg(V,NG)):-
    t(verb, I,R, V),
    t(ng, R,O, NG).
t(det, [X|I],I, det(X)):-
    det(X).
t(verb,[X|I],I, verb(X)):-
    verb(X).
t(noun, [X|I],I, noun(X)):-
    noun(X).
```

```
?- [lp].
true.

?- t(sent, [a,cow,shakes,its,tail], [], T).
T = sent(ng(det(a), noun(cow)), vg(verb(shakes),
         ng(det(its), noun(tail)))) ;
false.
```



Linked grammar (грамматики связей)



words	formula
a the	D+
snake cat	D- & (O- or S+)
Mary	O- or S+
ran	S-
chased	S- & O+

Последовательность слов находится в linked grammar, если существует способ нарисовать связи между словами, такие что

Связи не пересекаются (**планарный граф**);

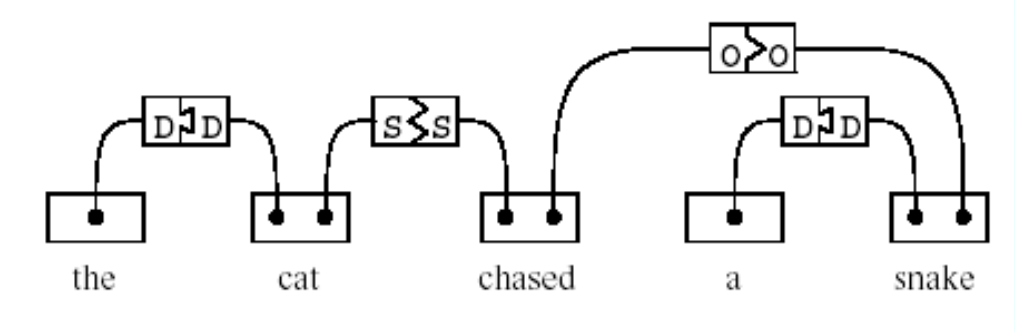
Все слова последовательности соединены связями (**связность**);

Все связи удовлетворяют ограничениям (**непротиворечивость**)

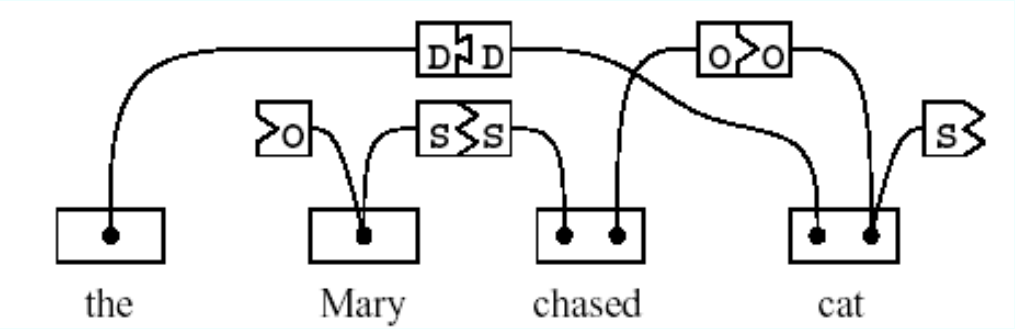
Два слова соединены одной и только одной связью (**исключительность**).

Примеры разбора Linked grammar

“The cat chased a snake”

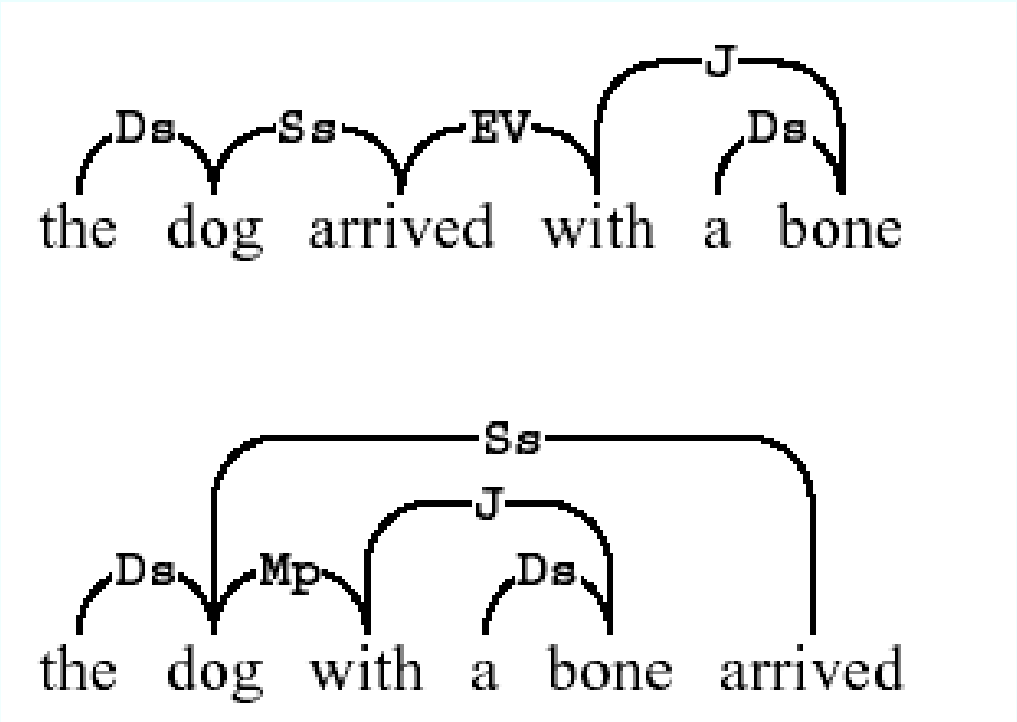


“The Mary chased cat”



Примеры разбора Linked grammar

“A dog arrived with a bone”
“A dog with a bone arrived”



Примеры разбора Linked grammar

```
linkparser> GABA mediates the inhibitory effect of NO on the AVP and OXT respons
es to insulin-induced hypoglycemia.
++++Time                                          0.47 seconds (0.47 total)
Found 129 linkages (75 had no P.P. violations)
  Linkage 1, cost vector = (UNUSED=0 DIS=0 AND=0 LEN=39)

    +-----------------------------------------------------------------Xp----------
    |               +-----------------------------------MVp--------------------
    |               +-----------------MVp-----------------+
    |               +------------Os------------+          |
    |               |      +--------D*u--------+          +--Js--+
    +---Wd--+---Ss--+      |        +-----A----+--Mp-+Js+  |  +-DG+
    |       |       |      |        |          |     |  |  |  |   |
LEFT-WALL GABA mediates.v the inhibitory.a effect.n of NO on the AVP and OXT


-------------------------------------------------+
-------------+                                   |
             |                                   |
             |                                   |
             +------------Jp-----------+         |
             |         +-------Ah-------+         |
             |         |                |         |
responses.n to insulin-induced hypoglycemia[?].n .


    +-----------------------------------------------------------------Xp----------
    |               +-----------------------------------MVp--------------------
    |               +-----------------MVp-----------------+
    |               +------------Os------------+          |
    |               |      +--------D*u--------+          +----------Jp-----
    +---Wd--+---Ss--+      |        +-----A----+--Mp-+Js+  |               +--
    |       |       |      |        |          |     |  |  |               |
LEFT-WALL GABA mediates.v the inhibitory.a effect.n of NO on the AVP and OXT


-------------------------------------------------+
-------------+                                   |
             |                                   |
             |                                   |
-----+       +------------Jp-----------+         |
-AN--+       |         +-------Ah-------+         |
     |       |         |                |         |
responses.n to insulin-induced hypoglycemia[?].n .
```

Информационная система GeoBase. База данных

GeoBase – программа, позволяющая делать запросы на ЕЯ к базе данных по географии США, Borland, 1988.

```
state('alabama','al','montgomery',3894e3,51.7e3,22,'birmingham','mobile','montgomery','huntsville').
state('alaska','ak','juneau',401.8e3,591e3,49,'anchorage','fairbanks','juneau','sitka').

city('alabama','al','birmingham',284413).
city('alabama','al','mobile',200452).

border('florida','fl',['georgia','alabama']).

highlow('alabama','al','cheaha mountain',734,'gulf of mexico',0).

mountain('alaska','ak','mckinley',6194).
mountain('alaska','ak','st. elias',5489).

road('66',['district of columbia','virginia']).

lake('huron',59570,['michigan']).
```

```
  The database contains the following information:

Information about states:
  Area of the state in square kilometers
  Population of the state in citizens
  Capital of the state
  Which states border a given state
  Rivers in the state
  Cities in the state
  Highest and lowest point in the state in meters

Information about rivers:
  Length of river in kilometers

Information about cities:
  Population of the city in citizens
```

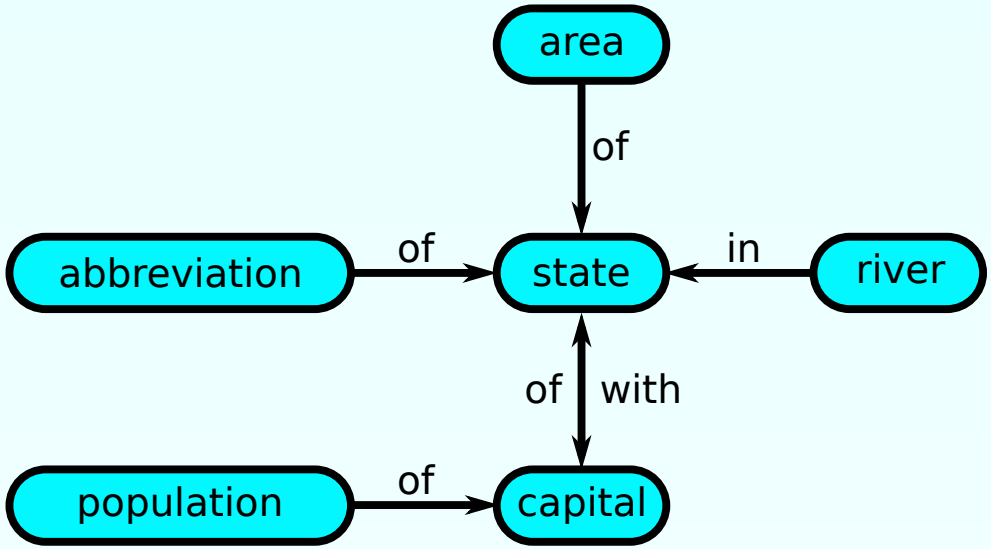
Примеры запросов

Some sample queries:

- states
- give me the cities in california.
- what is the biggest city in california ?
- what is the longest river in the usa?
- which rivers are longer than 1 thousand kilometers?
- what is the name of the state with the lowest point?
- which states border alabama?
- which rivers do not run through texas?
- which rivers run through states that border the state with the capital austin?

Схемы (спецификации) интерпретации

```
schema('abbreviation','of','state').
schema('state','with','abbreviation').
schema('capital','of','state').
schema('state','with','capital').
schema('population','of','state').

schema('area','of','state').
schema('city','in','state').

schema('length','of','river').
schema('state','with','river').
schema('river','in','state').

schema('capital','with','population').
schema('point','in','state').

schema('height','of','point').
schema('mountain','in','state').

schema('height','of','mountain').
schema('lake','in','state').

schema('name','of','river').
schema('name','of','capital').

schema('road','in','continent').
```



Программа

Основной цикл программы

```
geobase(STR, X, E):-
        STR \= "",
        atom_string(ATOM,STR),
        tokenize_atom(ATOM,LIST),      /* Returns a list of words(symbols)           */
        filter(LIST,LIST1),            /* Removes punctuation and words to be ignored*/
        pars(LIST1,E,Q),               /* Parses queries                             */
        findall(A,eval_interp(Q,A),L),
        unik(L,L1),
        % unit(E,U),
        member(X,L1).
```

Синтаксический анализатор (часть)

```
  pars(LIST,E,Q):-s_attr(LIST,OL,E,Q),check(OL),!.
  pars(LIST,_,_):-error(LIST),fail.
% . . . . . .
  /* How big is the biggest city -- BIG QUERY */
  s_attr([BIG|S1],S2,E1,q_eaq(E1,A,E2,Q)):-
                size(_,BIG),s_minmax(S1,S2,E2,Q),
                size(E2,BIG),entitysize(E2,E1),
                schema(E1,A,E2),!.

  s_attr(S1,S2,E,Q):-s_minmax(S1,S2,E,Q).
% . . . . . .

/*  ... the shortest river in texas -- MIN QUERY */
  s_assoc1([MIN|S1],S2,E1,A,q_eaq(E1,A,E2,q_min(E2,Q))):-minn(MIN),!,
                s_nest(S1,S2,E2,Q),schema(E1,A,E2).

/*  ... the longest river in texas -- MAX QUERY */
  s_assoc1([MAX|S1],S2,E1,A,q_eaq(E1,A,E2,q_max(E2,Q))):-maxx(MAX),!,
                s_nest(S1,S2,E2,Q),schema(E1,A,E2).
```

Корпус, источник данных

```
population of Washington
```

Население штата или города?

```
?- schema('population','of','city').
?- schema('population','of','state').
```

Корпус реализован при помощи реструктуризации базы данных.

```
% . . . . . .
/* Relationships about states */
db(abbreviation,of,state,ABBREVIATION,STATE):-  state(STATE,ABBREVIATION,_,_,_,_,_,_,_,_).
db(state,with,abbreviation,STATE,ABBREVIATION):-state(STATE,ABBREVIATION,_,_,_,_,_,_,_,_).
db(area,of,state,AREA,STATE):-        state(STATE,_,_,_,AREA1,_,_,_,_,_),str_real(AREA,AREA1).
db(capital,of,state,CAPITAL,STATE):-  state(STATE,_,CAPITAL,_,_,_,_,_,_,_).
db(state,with,capital,STATE,CAPITAL):-state(STATE,_,CAPITAL,_,_,_,_,_,_,_).
db(population,of,state,POPULATION,STATE):-state(STATE,_,_,POPUL,_,_,_,_,_,_),str_real(POPULATION,POPUL).
db(state,border,state,STATE1,STATE2):-border(STATE2,_,LIST),member(STATE1,LIST).

/* Relationships about rivers */
db(length,of,river,LENGTH,RIVER):-    river(RIVER,LENGTH1,_),str_real(LENGTH,LENGTH1).
db(state,with,river,STATE,RIVER):-    river(RIVER,_,LIST),member(STATE,LIST).
db(river,in,state,RIVER,STATE):-      river(RIVER,_,LIST),member(STATE,LIST).

/* Relationships about points */
db(point,in,state,POINT,STATE):-      highlow(STATE,_,POINT,_,_,_).
db(point,in,state,POINT,STATE):-      highlow(STATE,_,_,_,POINT,_).
db(state,with,point,STATE,POINT):-    highlow(STATE,_,POINT,_,_,_).
db(state,with,point,STATE,POINT):-    highlow(STATE,_,_,_,POINT,_).
db(height,of,point,HEIGHT,POINT):-    highlow(_,_,_,_,POINT,H),str_real(HEIGHT,H),!.
db(height,of,point,HEIGHT,POINT):-    highlow(_,_,POINT,H,_,_),str_real(HEIGHT,H),!.
% . . . . . .
```

Программа

Интерпретация запроса

…`findall(A,eval_interp(Q,A),L)`, …

```
eval_interp(Q,IAns):-
              eval(Q,A),
              e_i(A,IAns).

eval(q_min(ENT,TREE),ANS):-
              findall(X,eval(TREE,X),L),
              entitysize(ENT,ATTR),
              sel_min(ENT,ATTR,99e99,'',ANS,L).

eval(q_max(ENT,TREE),ANS):-
              findall(X,eval(TREE,X),L),
              entitysize(ENT,ATTR),
              sel_max(ENT,ATTR,-1,'',ANS,L).

eval(q_sel(E,gt,ATTR,VAL),ANS):-
              schema(ATTR,ASSOC,E),
              db(ATTR,ASSOC,E,SVAL2,ANS),
              str_real(SVAL2,VAL2),
              VAL2>VAL.
% . . . . . .
eval(q_eaq(E1,A,E2,TREE),ANS):-
              eval(TREE,VAL),db(E1,A,E2,ANS,VAL).

eval(q_eaec(E1,A,E2,C),ANS):-db(E1,A,E2,ANS,C).

eval(q_e(E),ANS):-        ent(E,ANS). % EVAL "ATOM"

eval(q_or(TREE,_),ANS):- eval(TREE,ANS).

eval(q_or(_,TREE),ANS):- eval(TREE,ANS).

eval(q_and(T1,T2),ANS):- eval(T1,ANS1),eval(T2,ANS),ANS=ANS1.
```

Пример запуска программы Geobase

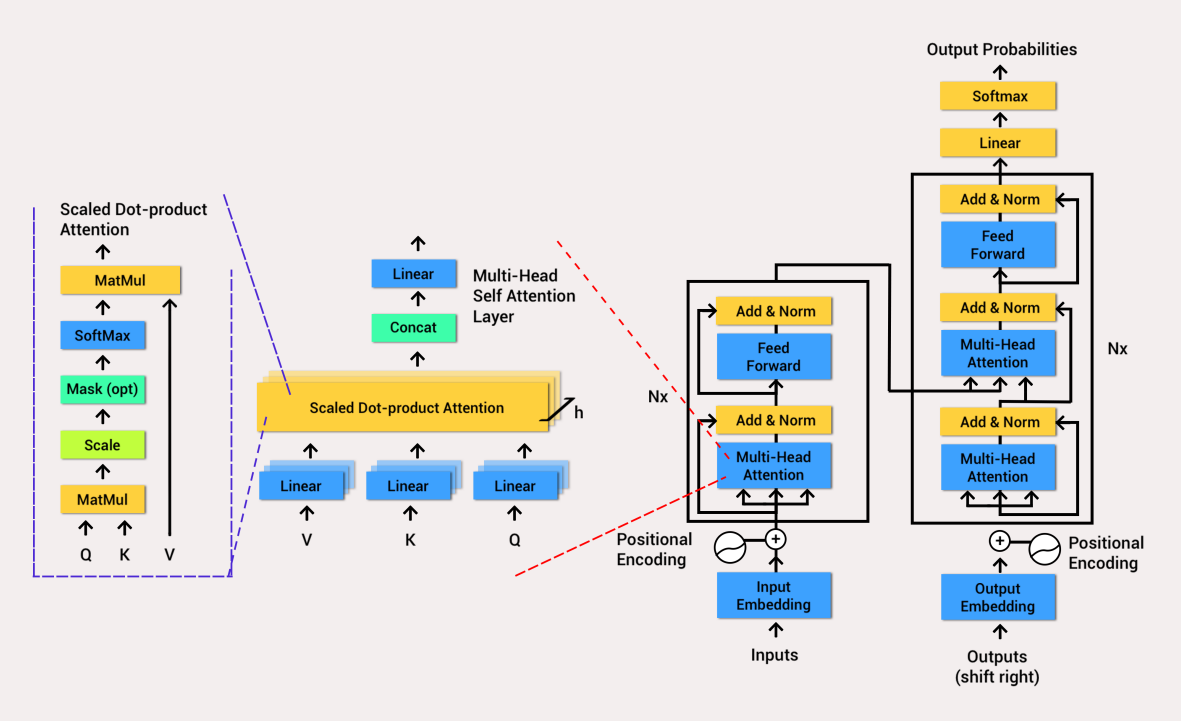
```
Welcome to SWI-Prolog (threaded, 64 bits, version 9.0.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- [geobase].
true.

?- loaddba.
Loading database file - please wait
true.

?- geobase:geobase("states").
alabama alaska arizona arkansas california
colorado connecticut delaware florida georgia
hawaii idaho illinois indiana iowa
kansas kentucky louisiana maine maryland
massachusetts michigan minnesota mississippi missouri
montana nebraska nevada new hampshire new jersey
new mexico new york north carolina north dakota
ohio oklahoma oregon pennsylvania rhode island
south carolina south dakota tennessee texas
utah vermont virginia washington west virginia
wisconsin wyoming

50 Solutions
true.

?- geobase:geobase("which rivers run through states that border the state with the capital austin?").
neosho washita arkansas st. francis white
mississippi ouachita pearl red canadian
cimarron rio grande san juan gila pecos

15 Solutions
true.
```

Chat GPT 4

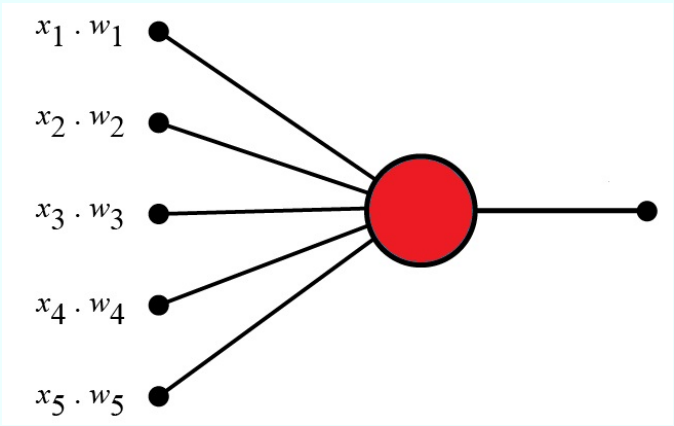
ChatGPT (Generative Pretrained Transformer, Порождающий [пред]треннированный преобразователь) – чат-бот OpenAI, запущенный в ноябре 2022.

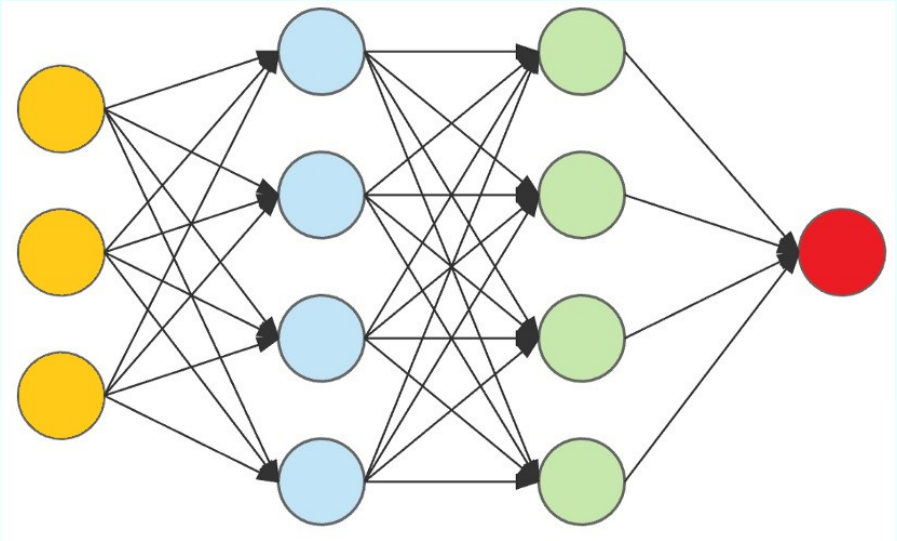
...настроен при помощи обучения «с учителем» и «с подкреплением». Именно обучение с подкреплением делает ChatGPT **уникальным**.

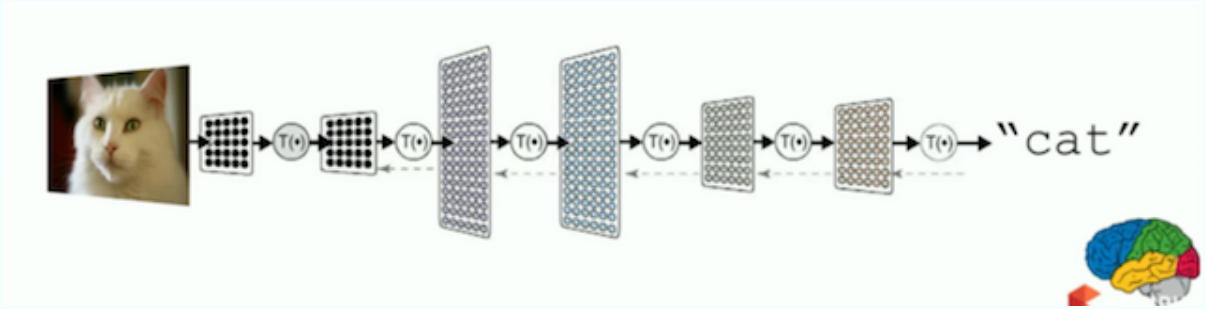
...предназначен для реагирования на входные данные, представленные на естественном языке.



Нейронные сети







Анализ входного текста

GPT-3 Codex

GPT-3 (Generative Pre-trained Transformer 3) uses a process called tokenization to break down text. Many words map to single tokens, though longer or more complex words often break down into multiple tokens. On average, tokens are roughly 4 characters long.

Clear Show example

Tokens 56

Characters 258

GPT-3 (Generative Pre-trained Transformer 3) uses a process called token ization to break down text. Many words map to single tokens, though longer or more complex words often break down into multiple tokens. On average, tokens are roughly 4 characters long.

TEXT TOKEN IDS

Воспитание нейронной сети

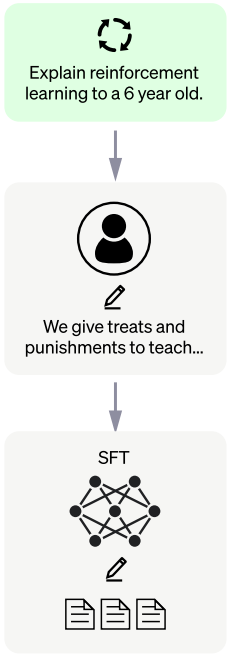
Step 1

Collect demonstration data and train a supervised policy.

A prompt is sampled from our prompt dataset.

A labeler demonstrates the desired output behavior.

This data is used to fine-tune GPT-3.5 with supervised learning.



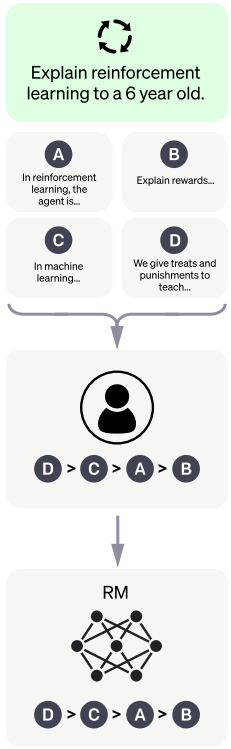
Step 2

Collect comparison data and train a reward model.

A prompt and several model outputs are sampled.

A labeler ranks the outputs from best to worst.

This data is used to train our reward model.



Step 3

Optimize a policy against the reward model using the PPO reinforcement learning algorithm.

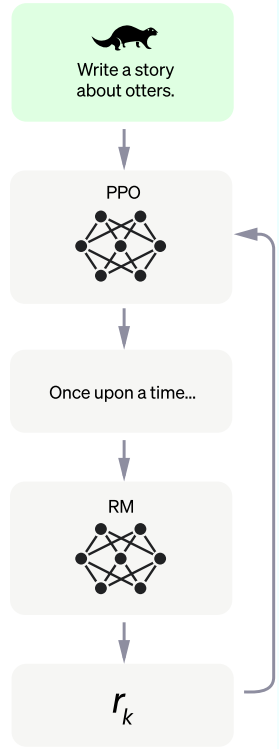
A new prompt is sampled from the dataset.

The PPO model is initialized from the supervised policy.

The policy generates an output.

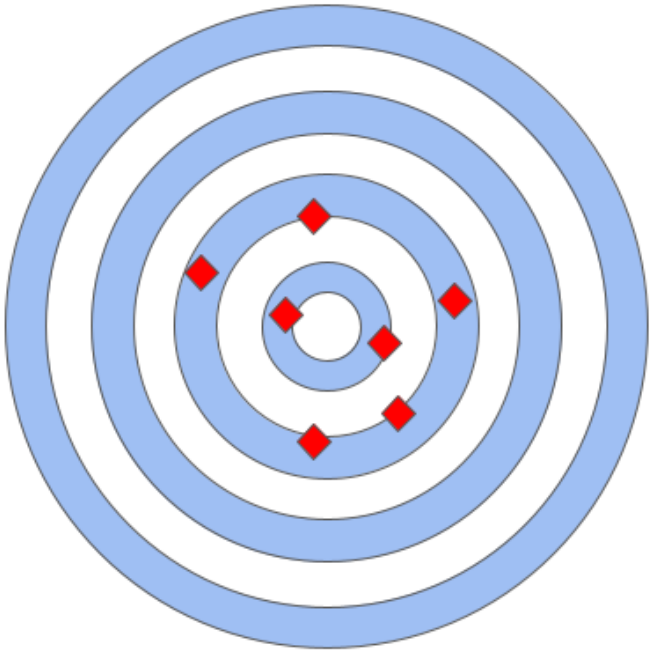
The reward model calculates a reward for the output.

The reward is used to update the policy using PPO.



Цель воспитания

High capability
Low alignment



Low capability
High alignment

QR-код презентации