

# Úvod do umělé inteligence, jazyk Prolog

Aleš Horák

E-mail: [hales@fi.muni.cz](mailto:hales@fi.muni.cz)  
<http://nlp.fi.muni.cz/uui/>

Obsah:

- ▶ Organizace předmětu PB016
- ▶ Co je “umělá inteligence”
- ▶ Stručné shrnutí Prologu

# Organizace předmětu PB016

## Hodnocení předmětu:

- ▶ **průběžná písemka** (max 32 bodů)
  - v 1/2 semestru – v rámci 6. přednášky, pro všechny jediný termín
- ▶ **závěrečná písemka** (max 96 bodů)
  - dva řádné a jeden opravný termín
- ▶ hodnocení – součet bodů za obě písemky (max 128 bodů)
- ▶ známka A za více než 115 bodů známka E za více než 63 bodů
- ▶ rozdílly **zk**, **k**, **z** – různé limity
- ▶ někteří můžou získat body za **studentské referáty**
  - až 20 bodů – za kvalitní text (cca 5 stran) + 10–20 minut referát
  - nutné *před průběžnou písemkou* domluvit **téma** – projekt/program, algoritmus z Náplně předmětu
  - domluva *e-mailem* – návrh tématu, který musí projít schválením
- ▶ kdo opraví chybu nebo vylepší **příklady z přednášek**, může dostat 1–5 bodů (celkem max 5)

# Základní informace

- ▶ cvičení – samostudium, v rámci “třetího kreditu”
- ▶ web stránka předmětu – <http://nlp.fi.muni.cz/uui/>
- ▶ <http://nlp.fi.muni.cz/uui/priklady/> – příklady z přednášek
- ▶ slajdy – průběžně doplňovány na webu předmětu
- ▶ kontakt na přednášejícího – Aleš Horák <[hales@fi.muni.cz](mailto:hales@fi.muni.cz)>  
(Subject: PB016 ...)
- ▶ literatura:
  - Russell, S. a Norvig, P.: [Artificial Intelligence: A Modern Approach](#), 3rd ed., Prentice Hall, 2010. (prezenčně v knihovně)
  - Bratko, I.: [Prolog Programming for Artificial Intelligence](#), Addison-Wesley, 2001. (prezenčně v knihovně)
  - slajdy na webu předmětu
  - Jirků, Petr: [Programování v jazyku Prolog](#), Praha : Státní nakladatelství technické literatury, 1991.

# Náplň předmětu

- 1 úvod do UI, jazyk Prolog (19.9.)
- 2 operace na datových strukturách (26.9.)
- 3 prohledávání stavového prostoru (3.10.)
- 4 heuristiky, best-first search, A\* search (10.10.)
- 5 dekompozice problému, AND/OR grafy (17.10.)
- 6 problémy s omezujícími podmínkami, [průběžná písemka](#) (24.10.)
- 7 hry a základní herní strategie (31.10.)
- 8 logický agent, výroková logika (7.11.)
- 9 logika prvního řádu a transparentní intenzionální logika (14.11.)
- 10 reprezentace a vyvozování znalostí (21.11.)
- 11 učení, rozhodovací stromy, neuronové sítě (28.11.)
- 12 zpracování přirozeného jazyka (5.12.)

# Co je “umělá inteligence”

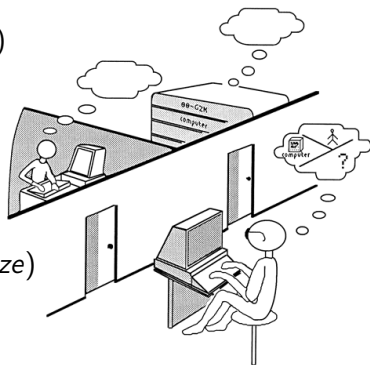
- ▶ systém, který se chová jako člověk

**Turingův test** (1950) zahrnuje:

- ▶ zpracování přirozeného jazyka (NLP)
- ▶ reprezentaci znalostí (KRepresentation)
- ▶ vyvozování znalostí (KReasoning)
- ▶ strojové učení
- ▶ (počítačové vidění)
- ▶ (robotiku)


od 1991 – **Loebnerova cena** (*Loebner Prize*)

→ každý rok \$4.000 za “nejlidštější”  
program, nabízí \$100.000 a zlatá  
medaile za složení celého Turingova testu



A comic panel showing two men sitting at a table in a bar. The man on the left has dark hair and a beard, wearing a grey shirt. The man on the right has blonde hair, a cigarette in his mouth, and is wearing a red shirt. They both have mugs of beer in front of them. A large white speech bubble originates from the man on the left.


Můj počítač udělal ten Turingův test.

A comic panel showing the same two men at the table. A large white speech bubble originates from the man on the right.

To znamená, že je stejně inteligentní jako já.

A comic panel showing the same two men at the table. A large white speech bubble originates from the man on the left.

Nebo ty.

A comic panel showing the same two men at the table. A large white speech bubble originates from the man on the right.

No on to zas tak velkej úspěch není.

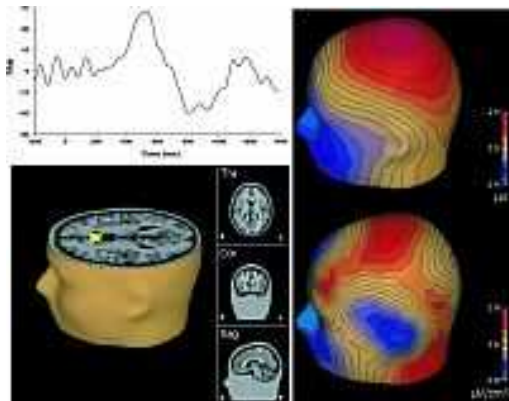
► systém, který myslí jako člověk

- snaha porozumět postupům lidského myšlení – **kognitivní (poznávací) věda**
- využívá poznatků neurologie, neurochirurgie, ... např.

*COLING 2000 – Angela Friederici: Language Processing in the Human Brain*

*Max Planck Institute of Cognitive Neuroscience, Leipzig*

měření "Event Related Potentials" (ERP) v mozku – jako potvrzení oddělení syntaxe a sémantiky při zpracování věty



- ▶ systém, který myslí rozumně
  - od dob Aristotela (350 př.n.l.)
  - ▶ náplň studia **logiky**
  - ▶ problém – umět najít řešení teoreticky × prakticky (složitost a vyčíslitelnost)
  - ▶ problém – neúplnost a nejistota vstupních dat
- ▶ systém, který se chová rozumně
  - inteligentní **agent** – systém, který
  - ▶ jedná za nějakým účelem
  - ▶ jedná samostatně
  - ▶ jedná na základě vstupů ze svého prostředí
  - ▶ pracuje delší dobu
  - ▶ adaptuje se na změny



# Čím se budeme zabývat?

- ▶ základní **struktury** a **algoritmy** běžně používané při technikách **programování pro inteligentní agenty**
- ▶ **strategie** řešení, **prohledávání** stavového prostoru, **heuristiky**, ...
- ▶ s příklady v jazyce **Prolog** a **Python**

# Stručné shrnutí Prologu

## Historie:

- ▶ 70. I. Colmerauer, Kowalski; D.H.D. Warren (WAM); → CLP, paralelní systémy
- ▶ PROgramování v LOGice; část predikátové logiky prvního řádu (logika Hornových klauzulí)
- ▶ deklarativnost (specifikace programu je přímo programem)
- ▶ řešení problémů týkajících se objektů a vztahů mezi nimi

## Prology na FI:

- ▶ SICStus Prolog (modul sicstus)
- ▶ SWI (modul pl)
- ▶ ECLiPSe (modul eclipse)
- ▶ stroje aisa, erinys, oreias, nymfe
- ▶ verze

# Příklad

jednoduchý příklad – DB rodinných vztahů:

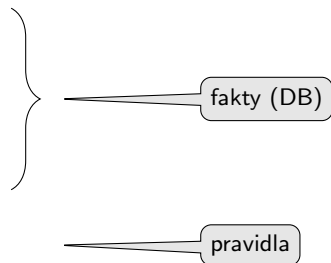
otec(milan,dana).  
otec(milan,petr).  
otec(jan,david).

matka(pavla,dana).  
matka(pavla,petr).  
matka(jana,david).

rodic(X,Y):- otec(X,Y).  
rodic(X,Y):- matka(X,Y).

?- otec(X,dana).  
X = milan  
Yes

?- rodic(X,david).  
X = jan ;  
X = jana ;



# Principy

- ▶ **backtracking** řízený **unifikací**, hojně využívá **rekurzi**
- ▶ spojitost s **logikou**:
  - důkaz pravdivosti cíle; cíl je dokázán, unifikuje-li s hlavou nějaké klauzule a všechny podcíle v těle této klauzule jsou rovněž dokázány. Strategie výběru podcíle: shora dolů, zleva doprava.
- ▶ **unifikace**:
  - řídicí mechanismus, hledání nejobecnějšího unifikátoru dvou termů.  
`info(Manzel,dana,Deti,svatba('20.12.1940')) = info(petr,dana,[jan,pavel],Info).`  
po unifikaci: **Manzel=petr**, **Deti=[jan,pavel]**, **Info=svatba('20.12.1940')**
- ▶ **backtracking**:
  - standardní metoda prohledávání stavového prostoru do hloubky (průchod stromem → nesplnitelný cíl → návrat k nejbližšímu minulému bodu s alternativní volbou)
- ▶ **rekurze**
  - `potomek(X,Y):- rodic(Y,X).`
  - `potomek(X,Y):- rodic(Z,X), potomek(Z,Y).`

# Syntax jazyka Prolog

- ▶ logický (prologovský) program – seznam klauzulí (pravidel a faktů) – nikoli množina
- ▶ klauzule – seznam literálů
  - ▶ Literál před :- je hlava, ostatní literály tvoří tělo klauzule.
  - ▶ Význam klauzule je implikace:
    - hlava:-tělo1, tělo2, ....
    - $\text{tělo1} \wedge \text{tělo2} \wedge \dots \Rightarrow \text{hlava}$
    - Pokud je splněno tělo1 a současně tělo2 a současně ..., pak platí také hlava.
  - ▶ 3 možné typy klauzulí:
    - fakt: hlava bez těla. Zápis v Prologu:  $p(X,Y)$ . (ekv.  $p(X,Y):-true$ .)
    - pravidlo: hlava i tělo. Prolog:  $p(Z,X) :- p(X,Y), p(Y,Z)$ .
    - cíl: tělo bez hlavy. Prolog:  $?- p(g,f)$ .
- ▶ predikát – seznam (všech) klauzulí se stejným funktorem a aritou v hlavovém literálu.
  - ▶ Zapisuje se ve tvaru *funktor/arita* – **potomek/2**.

- ▶ **literál** – atomická formule, nebo její negace
- ▶ **atomická formule** – v Prologu zcela odpovídá složenému termu (syntaktický rozdíl neexistuje)
- ▶ **term**:
  - ▶ konstanta: **a**, **1**, **'.'**, **[]**, **sc2**  
**atomic/1** (metalogické testování na konstantu)  
**atom/1**, **number/1**
  - ▶ proměnná: **X**, **Vys**, **\_**  
**var/1** (metalogické testování na proměnnou)
  - ▶ složený term: **f(a,X)**  
funktor, argumenty, arita  
**functor/3** dává funktor termu, **arg/3** dává  $n$ -tý argument  
zkratka pro zápis seznamů:  
**[1,a,b3]** odpovídá struktuře **'.'(1, '.'(a, '.'(b3, [])))**

## Příklad

predikát **sourozenci(X,Y)** – je **true**, když **X** a **Y** jsou (vlastní) sourozenci.

```
sourozenci(X,Y):- otec(O,X), otec(O,Y), X\=Y, matka(M,X), matka(M,Y).
```

```
1 otec(milan,dana).
2 otec(milan,petr).
3 otec(jan,david).
4 matka(pavla,dana).
5 matka(pavla,petr).
6 matka(jana,david).
7 rodic(X,Y):- otec(X,Y).
8 rodic(X,Y):- matka(X,Y).
```

```
?- sourozenci(dana,Y).
1, otec(O,dana) % O = milan
2, otec(milan,Y) % Y = dana
3, dana \= dana % fail → backtracking
2*, otec(milan,Y) % Y = petr
3, dana \= petr % true
4, matka(M,dana) % M = pavla
5, matka(pavla,petr) % true
```

Y = petr

Yes

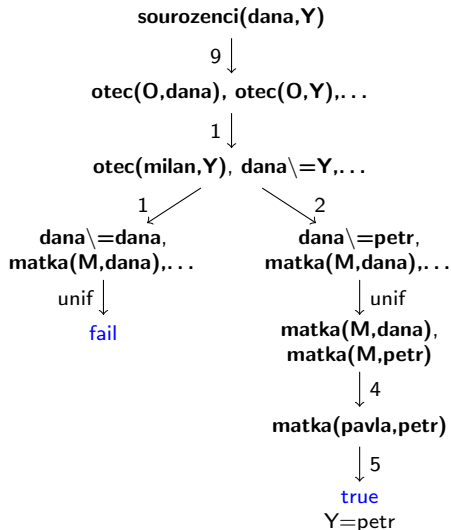
# Strom výpočtu

```

1 otec(milan,dana).
2 otec(milan,petr).
3 otec(jan,david).
4 matka(pavla,dana).
5 matka(pavla,petr).
6 matka(jana,david).
7 rodic(X,Y):- otec(X,Y).
8 rodic(X,Y):- matka(X,Y).
9 sourozenci(X,Y):- otec(O,X), otec(O,Y),
10   X\=Y,
11   matka(M,X), matka(M,Y).

```

Dotaz ?- **sourozenci(dana,Y).**





## Rozdíly od procedurálních jazyků

- ▶ **single assignment**
- ▶ **=** (unifikace) vs. přiřazovací příkaz, **==** (identita), **is** (vyhodnocení aritm. výrazu).      rozdíl:
  - ?- A=1, A=B. % B=1 Yes
  - ?- A=1, A==B. % No
  - ?- A=1, B is A+1. % B=2 Yes
- ▶ vícesměrnost predikátů (omezená, obzvláště při použití řezu)
  - ?- otec(X,dana).
  - ?- otec(milan,X).
  - ?- otec(X,Y).
 (rozdílení vstupních/výstupních proměnných: + - ?)
- ▶ cykly, podmíněné příkazy
  - tiskniseznam(S) :- **write**('seznam=[',nl,tiskniseznam(S,1).
  - tiskniseznam([],\_) :- **write**(']'),nl.
  - tiskniseznam([H|T],N):- **tab**(4),**write**(N),**write**(': '),**write**(H),nl,N1 is N+1,  
tiskniseznam(T,N1).

# Programujeme

```
consult('program.pl'). % “kompiluj” program.pl  
[‘program.pl’,program2]. % “kompiluj” program.pl, program2.pl  
listing. % vypiš programové predikáty  
trace, rodic(X,david). % trasuj volání predikátu  
notrace. % zruš režim trasování  
halt. % ukonči interpret
```

## Fibonacciho čísla

Fibonacciho čísla jsou čísla z řady: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, ...

Rekurenční vzorec této řady je:  $\text{fib}_0 = 0$

$$\text{fib}_1 = 1$$

$$\text{fib}_i = \text{fib}_{i-1} + \text{fib}_{i-2}, \text{ pro } i \geq 2$$

Přepis do Prologu je přímočarý:

`fib(0,0).`

`fib(1,1).`

`fib(X,Y) :- X1 is X-1, X2 is X-2, fib(X1,Y1), fib(X2,Y2), Y is Y1+Y2.`

## Fibonacciho čísla II

Předchozí program – **exponenciální** časová **složitost** (konstatní paměťová)  
Využití extralogických predikátů – **lineární** časová **složitost** (a lineární paměťová)

```
fib(0,0).
```

```
fib(1,1).
```

```
fib(X,Y) :- X1 is X-1, X2 is X-2, fib(X1,Y1), fib(X2,Y2), Y is Y1+Y2,  
           asserta(fib(X,Y)).
```