

# Mesterséges Intelligencia MI

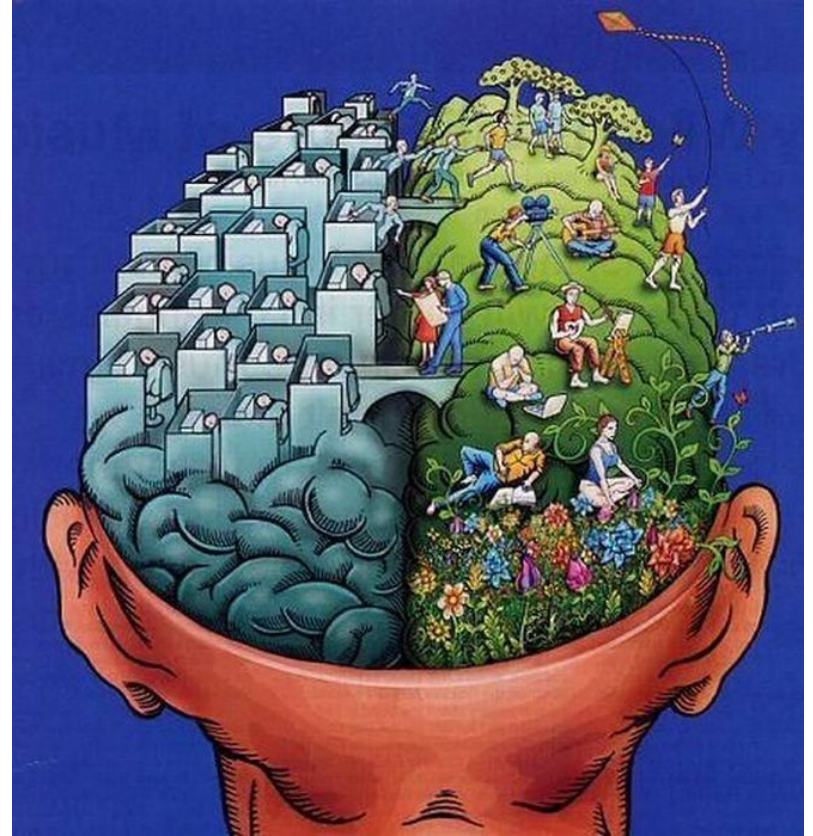
Produkción és  
tételbizonyító  
rendszerek

Dobrowiecki Tadeusz  
Eredics Péter, és mások

BME I.E. 437, 463-28-99

[dobrowiecki@mit.bme.hu](mailto:dobrowiecki@mit.bme.hu),

<http://www.mit.bme.hu/general/staff/tade>



Szabály alapú rendszerek = Tudásalapú rendszerek  
= Szakértő rendszerek

**IF** [(Blood pressure systolic > 140 OR Blood pressure diastolic > 90)  
AND Blood cholesterol total > 200] OR  
[Arrhythmia AND Temperature(F) > 99 AND Not Flu infection]  
**THEN** Assign Investigation: Echocardiography  
**IF** (Cough OR Wet nose OR Sore throat) AND  
Temperature(F) > 99 AND (Joint pain AND Not Arthritis) AND  
Headache AND Eyes movement pain  
**THEN** Make Diagnosis: Flu infection with 85% probability

**IF enemy AND health\_low THEN retreat**

**IF enemy AND NOT distance\_close THEN pursue**

**IF enemy THEN dodge**

**IF enemy AND distance\_far THEN use\_rocketlauncher AND use\_railgun**

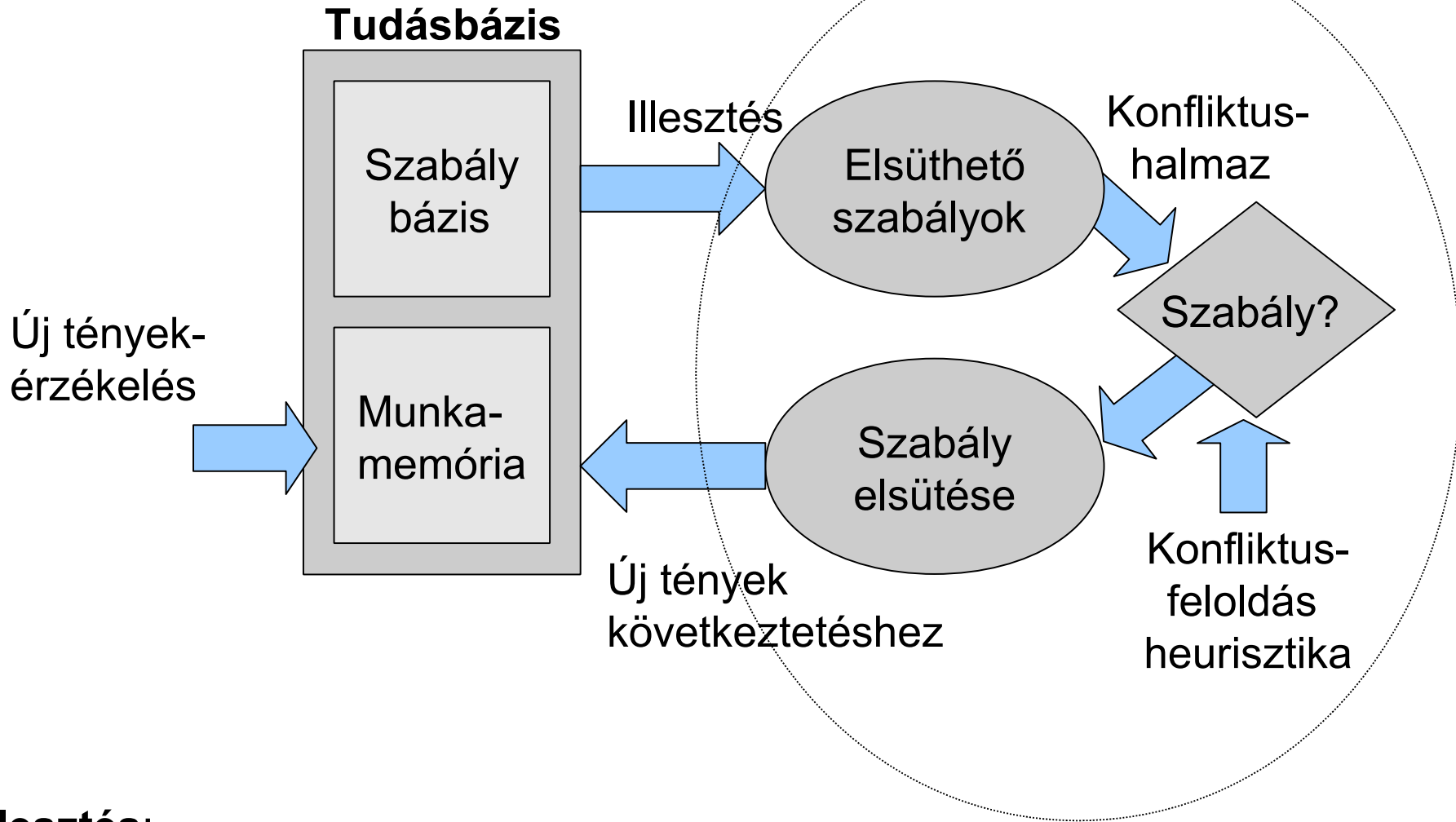
# **Mycin**, Stanford, ca. 1972, kb. 500 szabály, szakorvos szintje

**IF** The site of the culture blood, and  
The gram stain of the organism is gramneg, and  
The morphology of the organism is rod, and  
The portal of entry of the organism is urine, and  
The patient has not had a genito-urinary manipulative procedure, and  
Cystitis is not a problem for which the patient has been treated  
**THEN** There is suggestive evidence (.6) that  
the identity of the organism is e.coli

**IF** The identity of the organism is bacteroides  
**THEN** Recommend therapy chosen from the following drugs:  
1. clindamycin (0.99)      2. chloramphenicol (0.99)  
3. erythromycin (0.57)    4. tetracycline (0.28)  
5. carbenicillin (0.27)

**R1, XCON**, CMU, 1978, DEC VAX rendszerek rendelés utáni konfigurálása,  
kb. 2500 szabály, 1980-1986 között kb. 80,000 rendelés kezelése,  
95-98% pontosság, évi \$25M profit (szerelés sebessége, kevesebb  
anyagi kár, nagyobb elégedettség)

# Következtetőgép



## Illesztés:

Premisszára = **előre**-következtetés, forward chaining

Következményre = **hátra**-következtetés, backward chaining

# Előre/ Hátraláncolt működés (következtetés)

Sz1:  $P1 \wedge P2 \wedge P3 \rightarrow K1$

Sz2:  $P1 \wedge P2 \wedge P4 \wedge P5 \rightarrow K2$

Sz3:  $P1 \wedge P2 \wedge P3 \wedge P4 \rightarrow K3$

Tények

Szabályok

P1

Sz1? Sz2? Sz3?

P1, P2

Sz1? Sz2? Sz3?

P1, P2

**Sz1!** Sz2? Sz3?

P3, **K1**

P1, P2, **K1**

**Sz1!** Sz2? **Sz3!**

P3, P4, **K3**

P1, P2, **K1**

**Sz1! Sz2! Sz3!**

P3, P4, **K3**

P5, ...

## Illesztés.

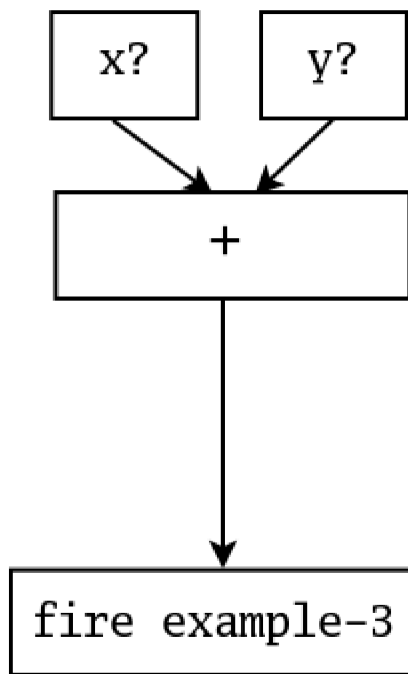
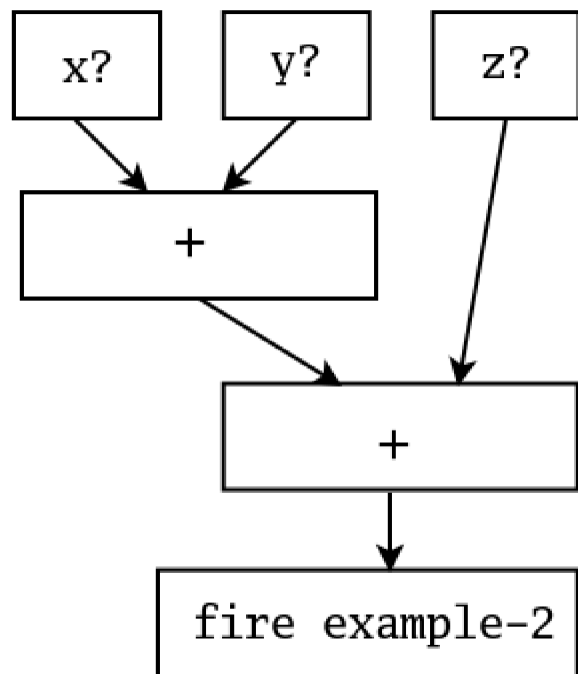
Valamennyi szabályt összehasonlítjuk a ténybázissal és meghatározzuk a sikeres illeszkedéseket;

## A konfliktus feloldása.

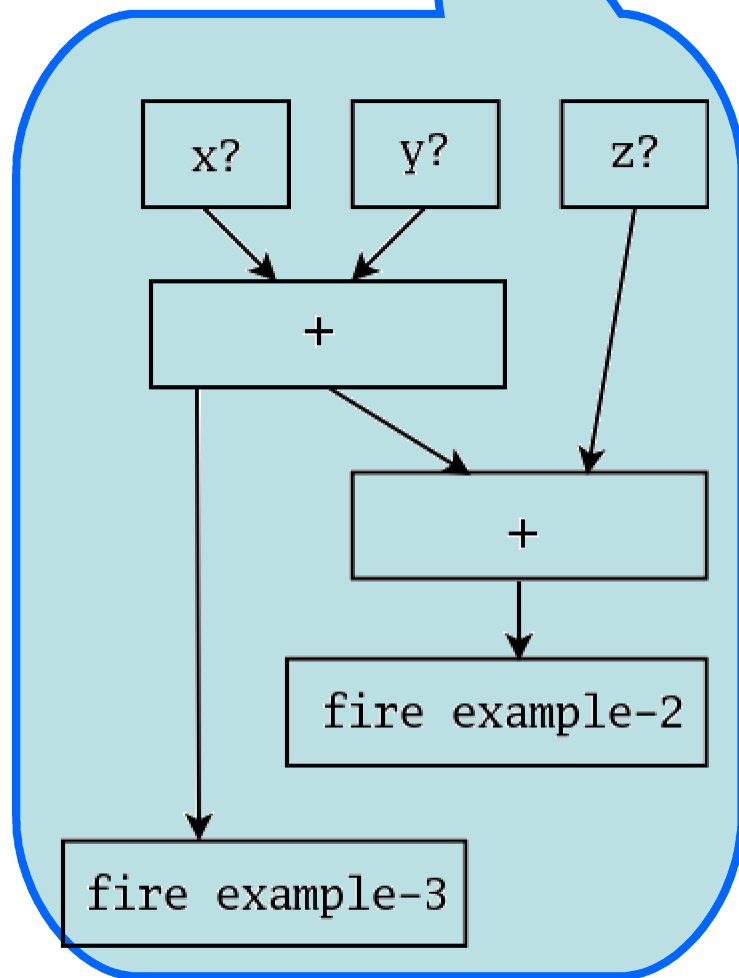
Több illeszkedő szabály esetén egy, több vagy valamennyi szabályt kiválaszthatjuk a stratégia függvényében;

## Végrehajtás.

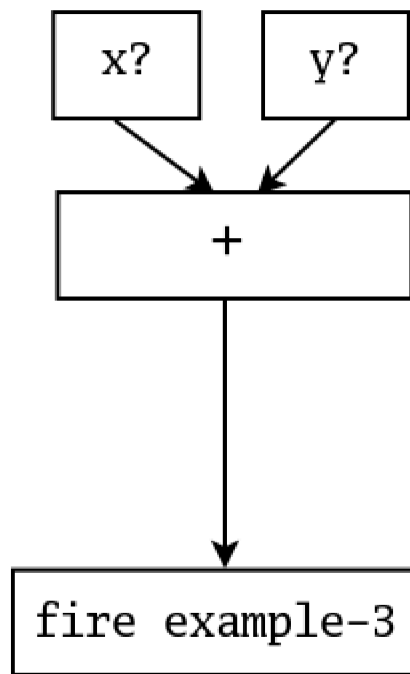
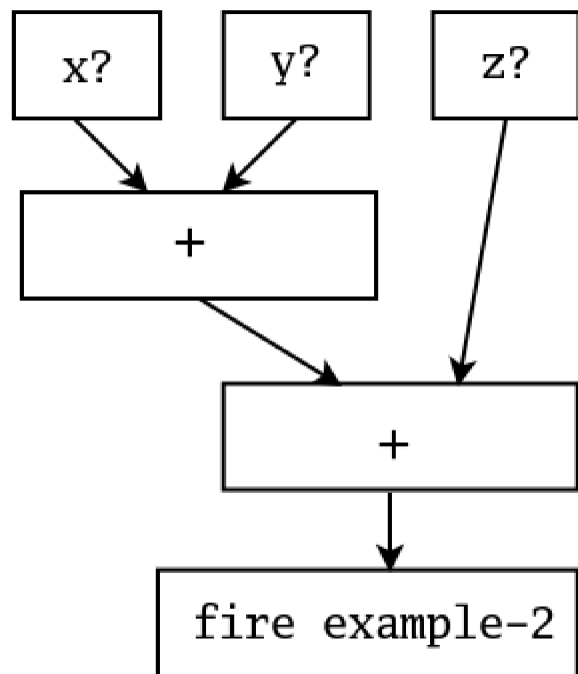
Hozzáadjuk a kiválasztott szabály(ok) következtetés(eit) a ténybázishoz.



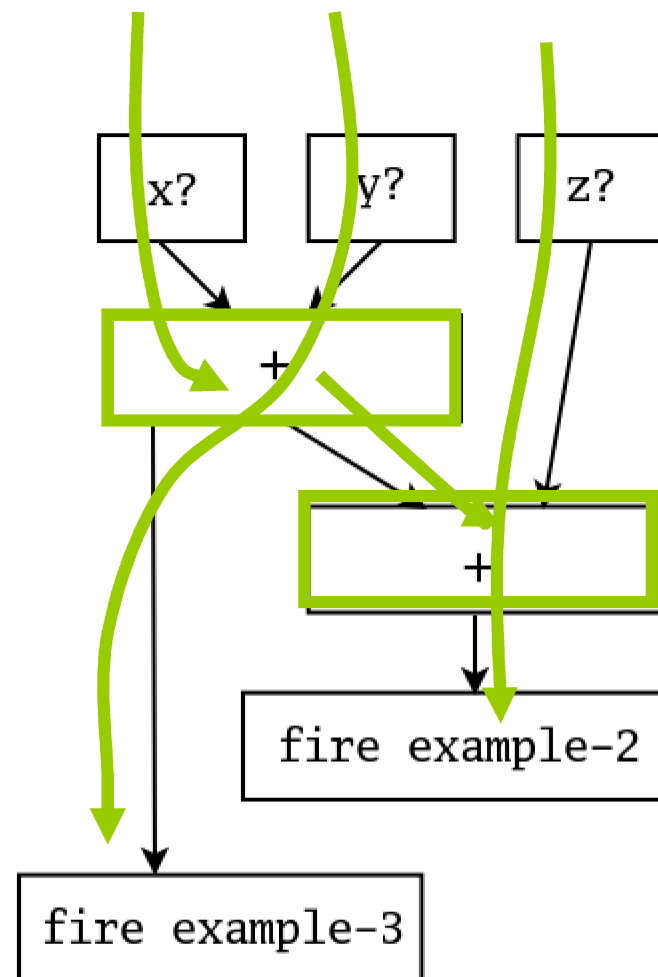
## RETE háló



Feltétel részek gráffá való „lefordítása”  
- Knowledge base compilation



## RETE háló



Feltétel részek gráffá való „lefordítása”  
- Knowledge base compilation

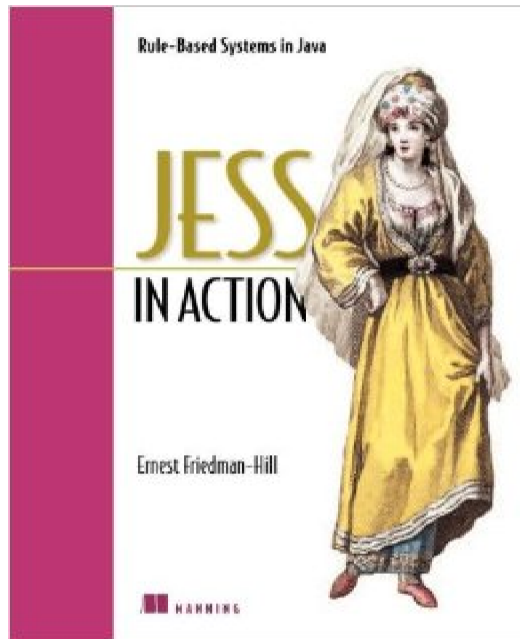


**CLIPS** = C Language Integrated Production System  
NASA's Johnson Space Center (80-as évek)  
C nyelv megvalósítás, LISP szintaktika

```
(defrule class-B-fire-emergency
  (emergency fire) (fire-class B)
  =>
  (printout t "Use carbon dioxide extinguisher" crlf))
```

<http://clipsrules.sourceforge.net/>

Konfliktus feloldási stratégia  
Frissesség  
Specifikusság  
Ciklusmentesség



**Jess®** The Rule Engine for the Java™ Platform

Jess in Action: Java  
Rule-Based Systems  
by Ernest Friedman-Hill



<http://herzberg.ca.sandia.gov/>

# Logikai programozás      Prolog

Prolog program = határozott klóz halmaz

mélységi keresés

beépített aritmetikai függvények = a kód kiszámít, „bizonyít”

predikátumok mellékhatásokkal

sikertelenségi tagadás     $\neg A$  bizonyított, ha A nem bizonyítható

korlátozott egyenlőségi operátor

nincs előfordulási próba

**nem teljes**

## Tételbizonyítók

**Otter** (Organized Techniques for Theorem proving and Effective Research)  
rezolúciós bizonyítás támogató halmaz (set of support) stratégiával

Prolog kiterjesztése – **PTTP Prolog Technology Theorem Prover**

+ előfordulási próba

+ legjobb először keresés mélységi keresés helyett

+ engedélyezett negált literál

+ zárás:     $A \Leftarrow B \wedge C$        $\neg B \Leftarrow C \wedge \neg A$        $\neg C \Leftarrow B \wedge \neg A$

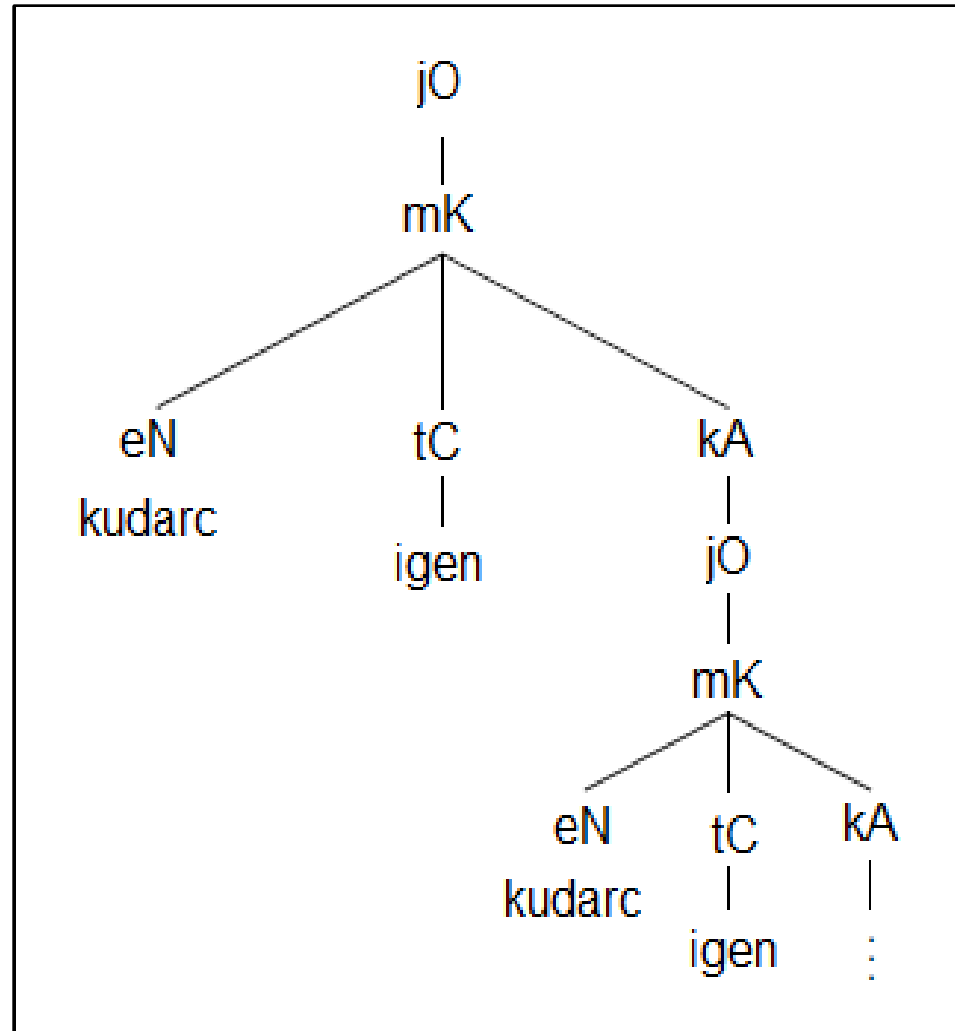
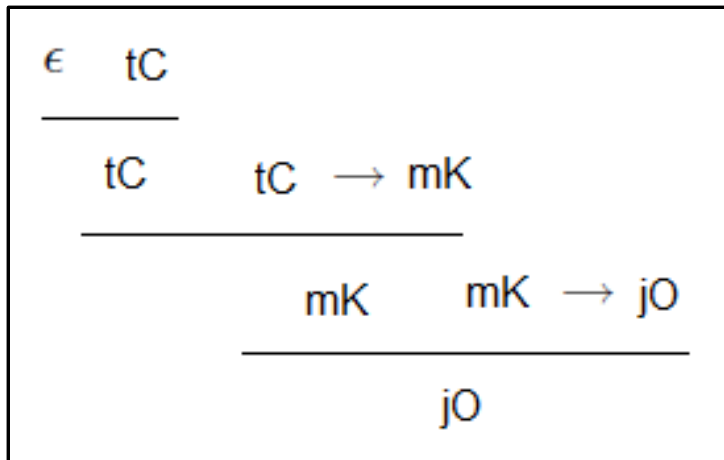
+ következtetés teljessé tétele: bemeneti rezolúció  
gyors, nem követhető

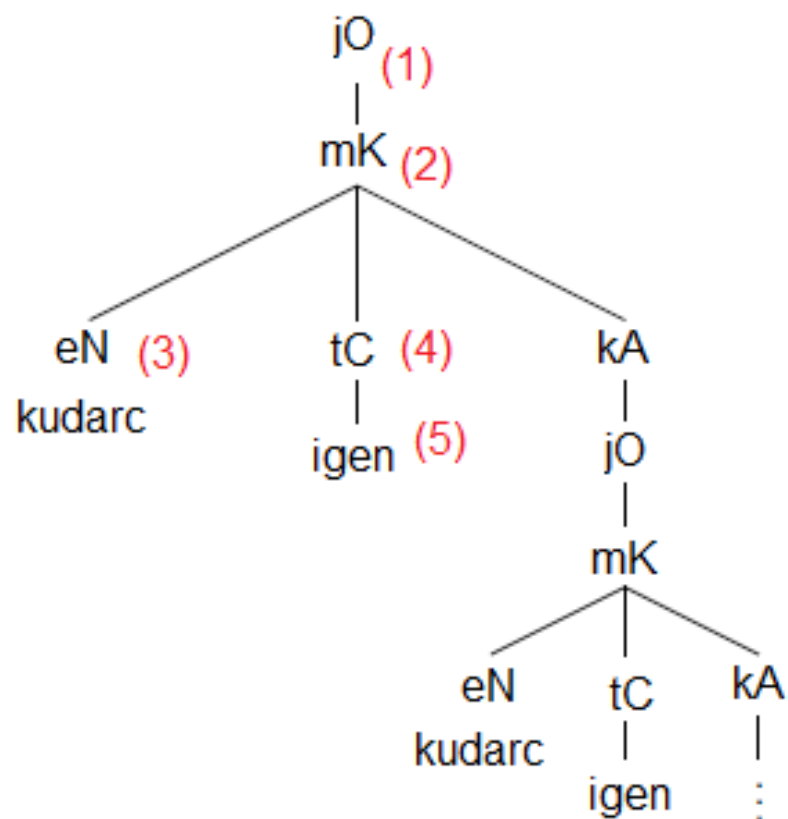
**Prover9**

# Logikai programozás      Prolog

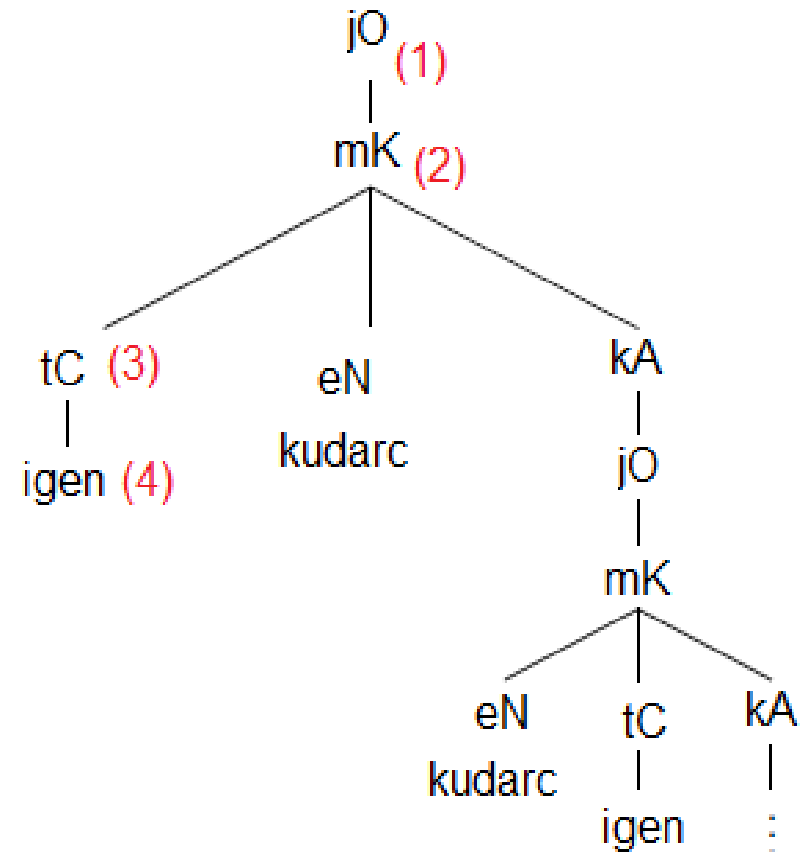
erősebbNapsütés → melegebbKlima  
 többCO2 → melegebbKlima  
 melegebbKlima → jégOlvasás  
 jégOlvasás → kisebbAlbedo  
 kisebbAlbedo → melegebbKlima  
 többCO2

JégOlvasás ?





|                 |   |               |
|-----------------|---|---------------|
| többCO2         | → | melegebbKlima |
| erősebbNapsütés | → | melegebbKlima |
| melegebbKlima   | → | jégOlvadás    |
| jégOlvadás      | → | kisebbAlbedo  |
| kisebbAlbedo    | → | melegebbKlima |
| többCO2         |   |               |



```

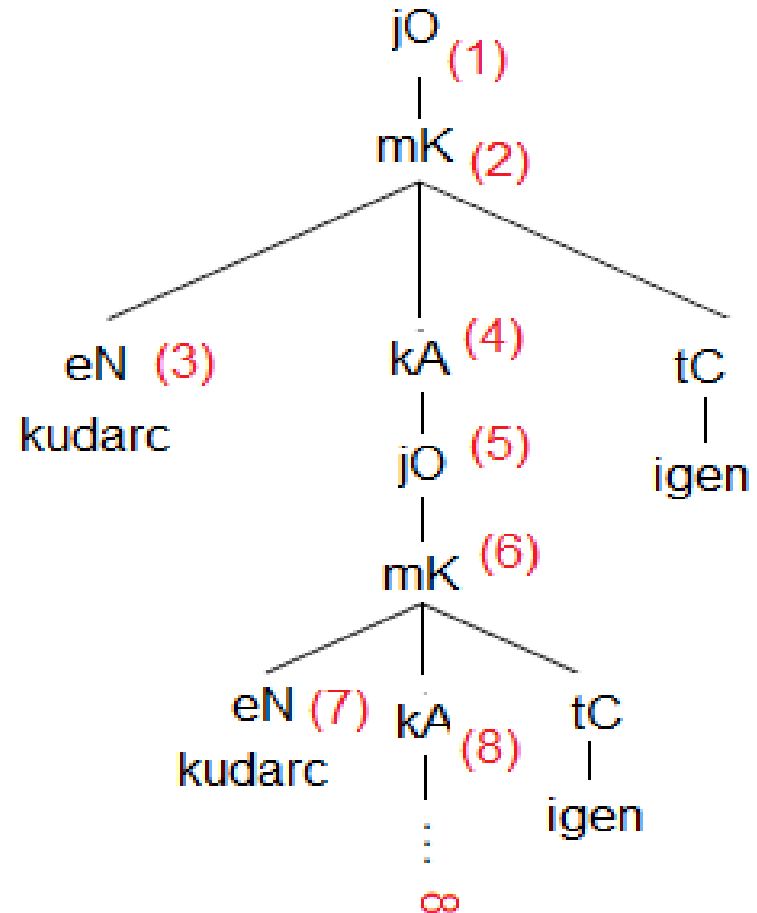
erősebbNapsütés → melegebbKlima
kisebbAlbedo    → melegebbKlima
többCO2          → melegebbKlima
melegebbKlima    → jégOlvasás
jégOlvasás       → kisebbAlbedo
többCO2

```

Query:

| ?- jegOlvadas.

# ! Resource error: insufficient memory



$$F_1, \dots, F_n \vdash_{\text{Prolog}} g_1, \dots, g_m \quad \Rightarrow \quad F_1, \dots, F_n \vdash g_1, \dots, g_m .$$

$$F_1, \dots, F_n \vdash g_1, \dots, g_m \quad \Rightarrow \quad F_1, \dots, F_n \models g_1 \wedge \dots \wedge g_m .$$

$$F_1, \dots, F_n \models g_1 \wedge \dots \wedge g_m \quad \Rightarrow \quad F_1, \dots, F_n \vdash g_1, \dots, g_m .$$

$$F_1, \dots, F_n \models g_1 \wedge \dots \wedge g_m \quad \not\Rightarrow \quad F_1, \dots, F_n \vdash_{\text{Prolog}} g_1, \dots, g_m .$$

# Tételbizonyítók

**Otter** (Organized Techniques for Theorem proving and Effective Research)

<http://www.cs.unm.edu/~mccune/otter/>

<http://www.cs.unm.edu/~mccune/prover9/>

```
formulas(assumptions).
```

```
  all x all y all z (amerikai(x) & fegyver(y) &  
    nemzet(z) & ellenseges(z) & elad(x,y,z) -> bunos(x)).
```

```
  exists x (van(nono,x) & raketa(x)).
```

```
  all x (van(nono,x) & raketa(x) -> elad(awest,x,nono)).
```

```
  all x (raketa(x) -> fegyver(x)).
```

```
  all x (ellensege(x,amerika) -> ellenseges(x)).
```

```
  amerikai(awest).
```

```
  nemzet(nono).
```

```
  ellensege(nono,amerika).
```

```
  nemzet(amerika).
```

```
end_of_list.
```

```
formulas(goals).
```

```
  bunos(awest).
```

```
end_of_list.
```

# Prover9

**Automated Reasoning  
and the  
Discovery of Missing and  
Elegant Proofs**

**Larry Wos  
Gail W. Pieper**

 Rinton Press

CD-ROM  
Included

Prover9 Manual - Windows Internet Explorer

http://www.cs.unm.edu/~mccune/prover9/manual/2009-02A/

File Edit View Favorites Tools Help

Favorites Hirszerzö - Külföld - Sosem l... Papers Suggested Sites . PET . Arboretum.lap.hu BME OMIKK, 1111 Buda...

Prover9 Manual

*Prover9 Manual*

[Introduction](#)

[Installation](#)

[Running Prover9](#)

[Input Files](#)

[Clauses & Formulas](#)

[Search Prep](#)

Auto Modes

Term Ordering

More Prep

Search Limits

[Inference](#)

The Loop

Select Given

Inference Rules

Process Inferred

[Output Files](#)

[More Features](#)

Weighting

Attributes

Goals and Denials

Production Mode

[Advanced Features](#)

[Mace4](#)

Introduction

Input

Options

Arithmetic

Interpformat

Isosfilter

[Related Programs](#)

Prooftrans

FOF-Prover9

More Programs

[Ending](#)

All Prover9 Options

Glossary

References

*Prover9 Manual*

# Prover9

*Version 2009-02A*

## Introduction

[Prover9](#) is a resolution/paramodulation automated theorem prover for first-order and equational logic. Prover9 is a successor of the [Otter](#) Prover [[McCune-Otter33](#)].

## Getting Started

Prover9 has a fully [automatic mode](#) in which the user simply gives it formulas representing the problem. See the Section [Clauses and Formulas](#).

An good way to learn about Prover9 is to browse and study the [example input and output files](#). *Users are encouraged to contribute examples from their own work with Prover9 (and Mace4).*

## Related Programs

Several programs come bundled with Prover9. The most important is [Mace4](#), which looks for finite models and counterexamples. Mace4 can help avoid wasting time searching for a proof with Prover9 by first finding a counterexample or by first helping to debug logical specifications.

Another useful program is [Prooftrans](#), which can transform proofs found by Prover9 in various ways, including producing more detailed proofs, simplifying the justifications, renumbering the steps, producing proofs in XML, and producing proofs for input to other programs.

## Terms of Use

Prover9, Mace4, related programs, and the LADR libraries (with which they were all constructed) are distributed under the terms of the [GNU General Public License \(v2\)](#).

## Other Theorem Provers

- [E](#) is a very good all-around prover.
- [Waldmeister](#) is a fast prover for equational logic.
- [Vampire](#) has lately been winning the MIX category of [CASC](#).
- [Decadix](#) is an excellent program for finding finite models and counterexamples.