Experiments with an Agent-oriented Reasoning System

Christoph Benzmüller

joint work with: Mateja Jamnik, Manfred Kerber, Volker Sorge

Fachrichtung Informatik
Universität des Saarlandes
Saarbrücken, Germany

School of Computer Science
The University of Birmingham
Birmingham, England

Motivation – Cognitive Perspective

Ωmega

To solve complex mathematical problems

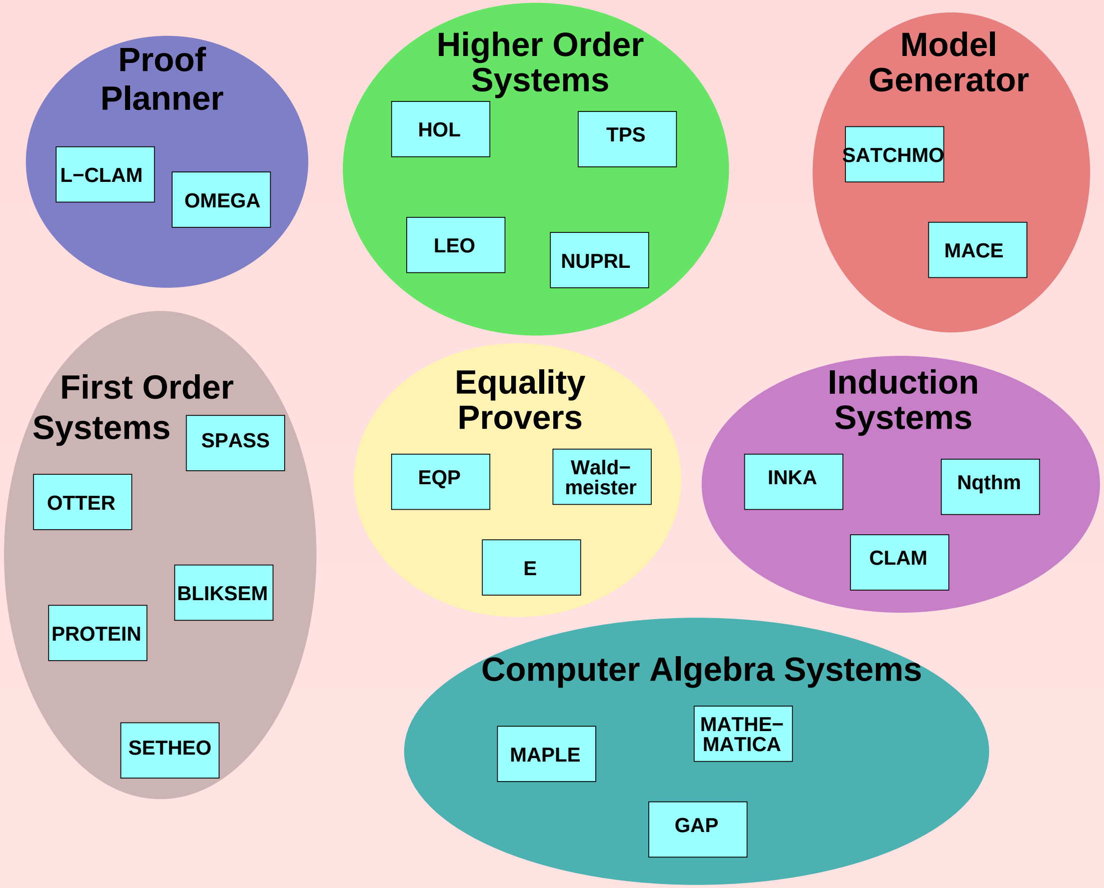
- different specialists may have to bring in their expertise and cooperate
- a communication language is required

A single mathematician

- possesses a large repertoire of specialised reasoning and problem solving techniques
- uses experience and intuition to flexibly combine them in an appropriate way

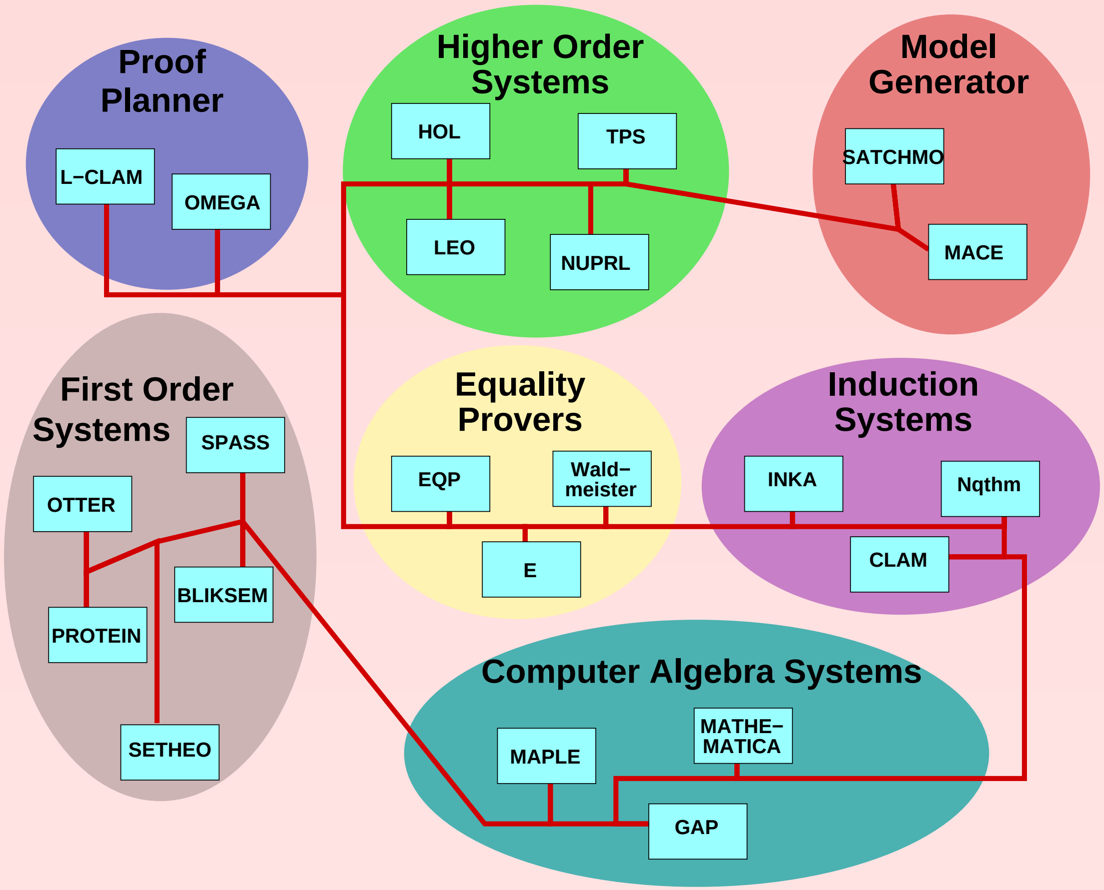
Motivation – Existing Systems

Ωmega



- heterogeneous
- different niches

Motivation – Existing Systems



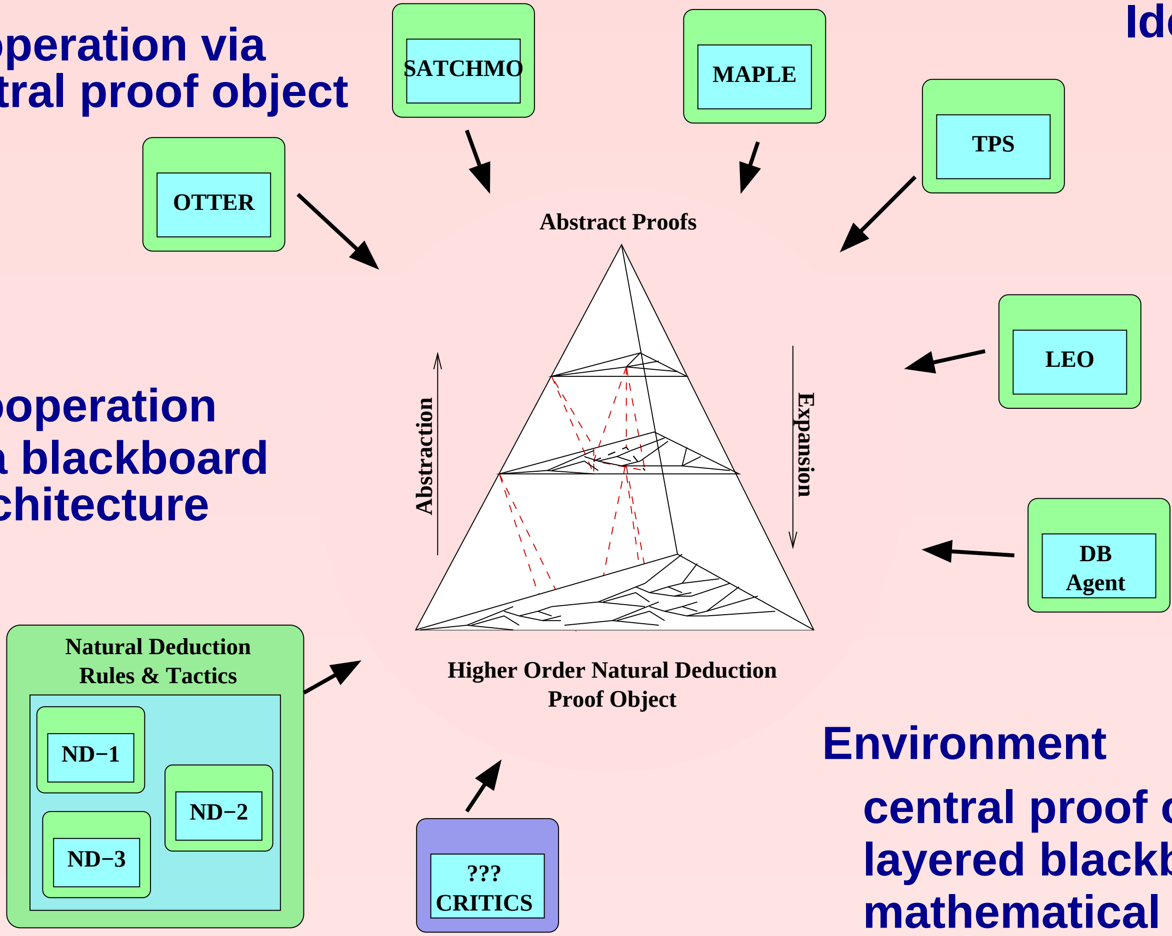
- heterogeneous
- different niches
- hardwired integrations
- system networks: MATHWEB, PROSPER integration-infrastructure

How to realise a flexible interplay?

Motivation – Flexible Integration

Cooperation via
central proof object

Idea
decentralised control



Cooperation
via blackboard
architecture

ND−Backbone

Agents
- reactive
- proactive
- heterogenous
- simple & complex
- cooperative & competetive
- distributed via MathWeb
- run−time definable
- resource adapted

Environment
- central proof object
- layered blackboards (local communication)
- mathematical database

Example – HO- and FO-ATP

Higher Order ATP with LEO

C_1: favourite-numbers($\lambda x.$ odd$(x) \wedge$ square(x))

unifies (semantically) with

C_2: $\neg$favourite-numbers($\lambda x.$ square$(x) \wedge$ (square$(x) \Rightarrow$ odd(x)))

iff the following first order clauses can be contradicted

First Order ATP with OTTER

square(N)

$\neg$odd$(N) \vee \neg$square(N)

odd$(N) \vee \neg$square(N)

odd$(N) \vee$ square(N)

Example – HO- and FO-ATP

fn $=$ favourite-numbers

o $=$ odd

s $=$ square

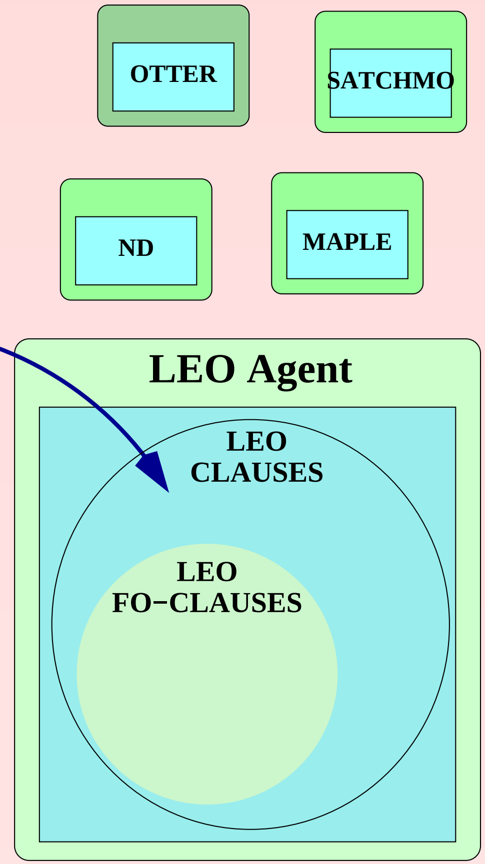
C_1 : $\mathsf{fn}(\lambda x.\, \mathsf{o}(x) \wedge \mathsf{s}(x))$

C_2 : $\neg\mathsf{fn}(\lambda x.\, \mathsf{s}(x) \wedge (\mathsf{s}(x) \Rightarrow \mathsf{o}(x)))$

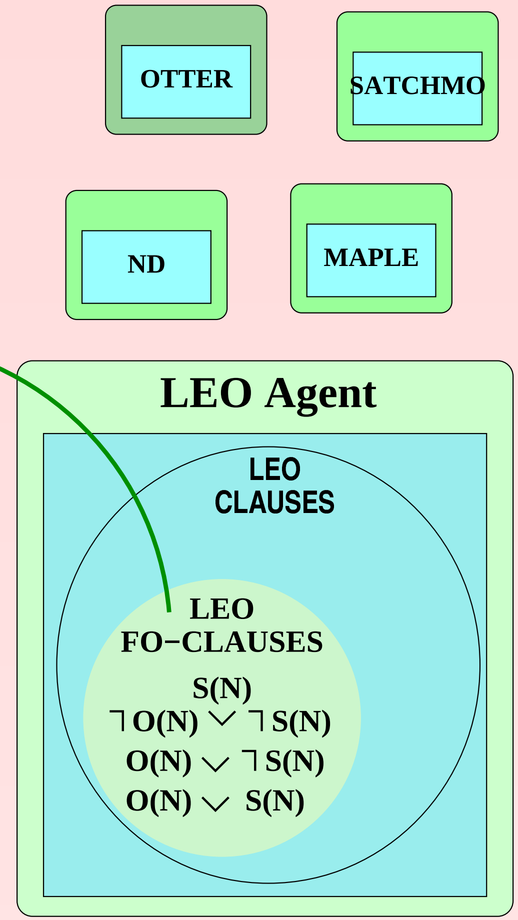
$$\mathsf{fn}(\lambda x.\, \mathsf{o}(x) \wedge \mathsf{s}(x))$$

$$\vdots \; ?$$

$$\mathsf{fn}(\lambda x.\, \mathsf{s}(x) \wedge (\mathsf{s}(x) \Rightarrow \mathsf{o}(x)))$$



Example – HO- and FO-ATP



$$\dfrac{\mathsf{fn}(\lambda x.\, \mathsf{o}(x) \wedge \mathsf{s}(x)) \qquad \neg\mathsf{fn}(\lambda x.\, \mathsf{s}(x) \wedge (\mathsf{s}(x) \Rightarrow \mathsf{o}(x)))}{\mathsf{s}(N) \wedge (\neg\mathsf{o}(N) \vee \neg\mathsf{s}(N)) \wedge (\mathsf{o}(N) \vee \neg\mathsf{s}(N)) \wedge (\mathsf{o}(N) \vee \mathsf{s}(N))}\ \mathit{LEO}$$

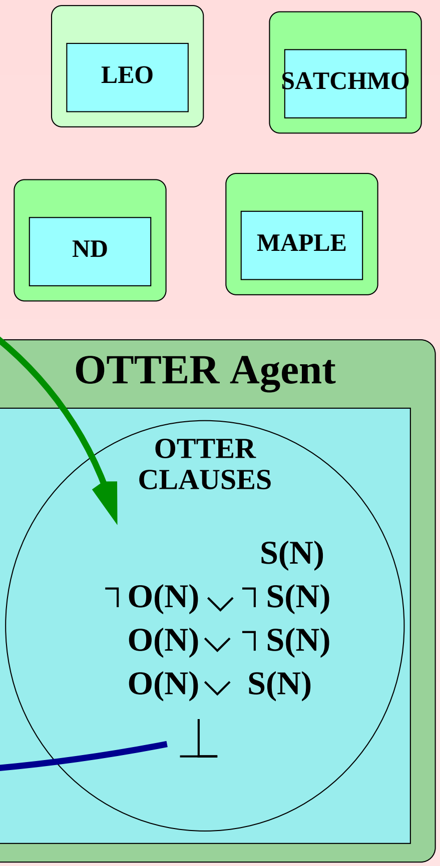
$$\vdots\ ?$$

$$\dfrac{\bot}{\mathsf{fn}(\lambda x.\, \mathsf{s}(x) \wedge (\mathsf{s}(x) \Rightarrow \mathsf{o}(x)))}\ \textbf{indirect}$$

Example – HO- and FO-ATP

$$\dfrac{\dfrac{\begin{array}{c}\mathsf{fn}(\lambda x.\, \mathsf{o}(x) \land \mathsf{s}(x)) \\ \neg\mathsf{fn}(\lambda x.\, \mathsf{s}(x) \land (\mathsf{s}(x) \Rightarrow \mathsf{o}(x)))\end{array}}{\begin{array}{r}\mathsf{s}(N) \land (\neg\mathsf{o}(N) \lor \neg\mathsf{s}(N)) \land (\mathsf{o}(N) \lor \neg\mathsf{s}(N)) \\ \land\, (\mathsf{o}(N) \lor \mathsf{s}(N))\end{array}}\,LEO}{\dfrac{\bot}{\mathsf{fn}(\lambda x.\, \mathsf{s}(x) \land (\mathsf{s}(x) \Rightarrow \mathsf{o}(x)))}}\,OTTER$$

indirect



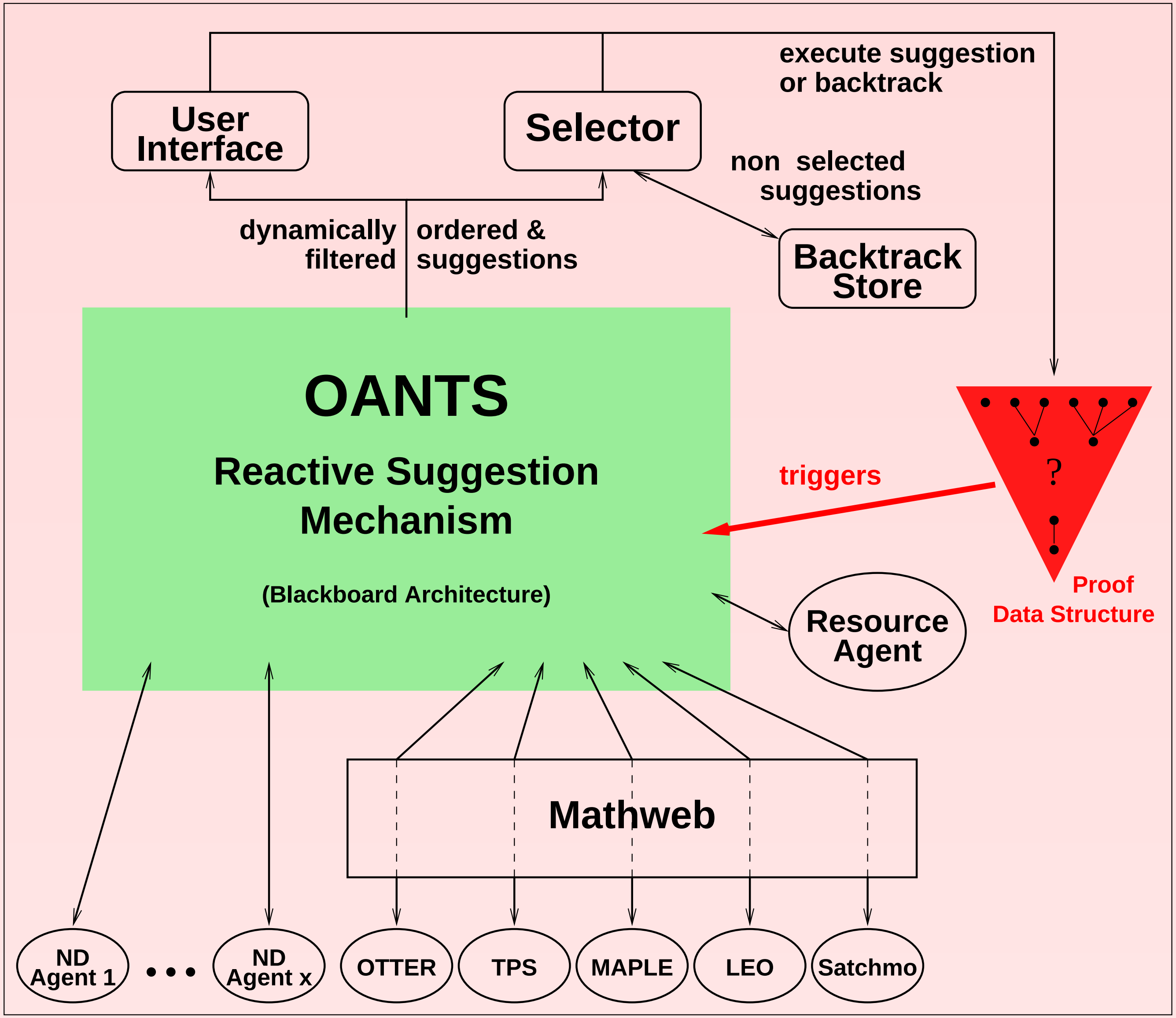
Realisation – Ω-ANTS

$$\frac{\texttt{Left}:A \quad \texttt{Right}:B}{\texttt{Conj}:A \land B}\ \land\texttt{I}$$

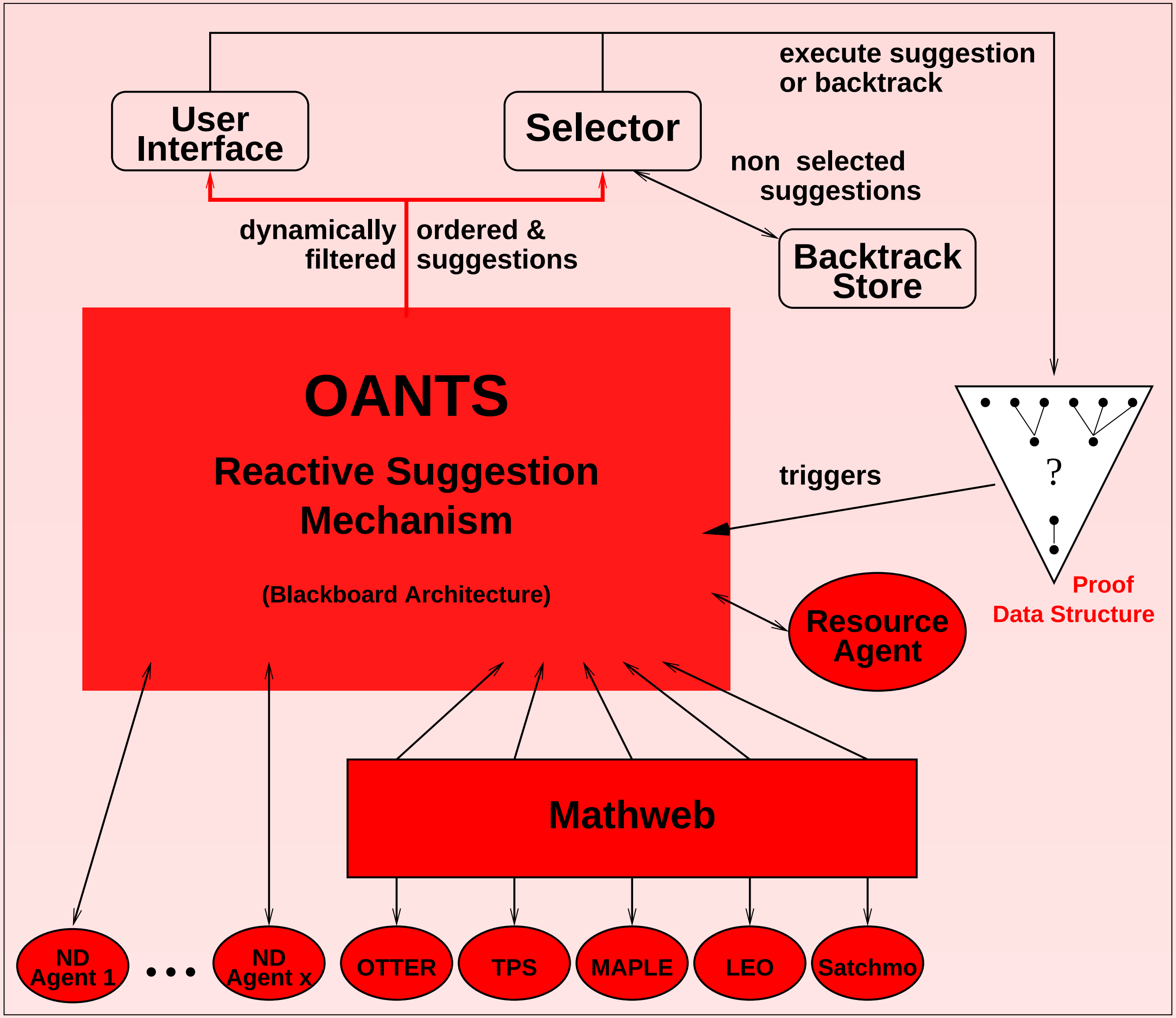
$$\frac{\texttt{Ass}_1:A_1 \quad \ldots \quad \texttt{Ass}_n:A_n}{\texttt{Conc}:C}\ \texttt{OTTER}(\texttt{P}_1:f_1,\ldots,\texttt{P}_m:f_m)$$

- Distributed applicability checks for:
  proof rules, tactics, external systems, etc.
- Applicability checks further distributed in various sub-processes
- Declarative specification/modification at run-time
- Currently more than 400 distributed processes

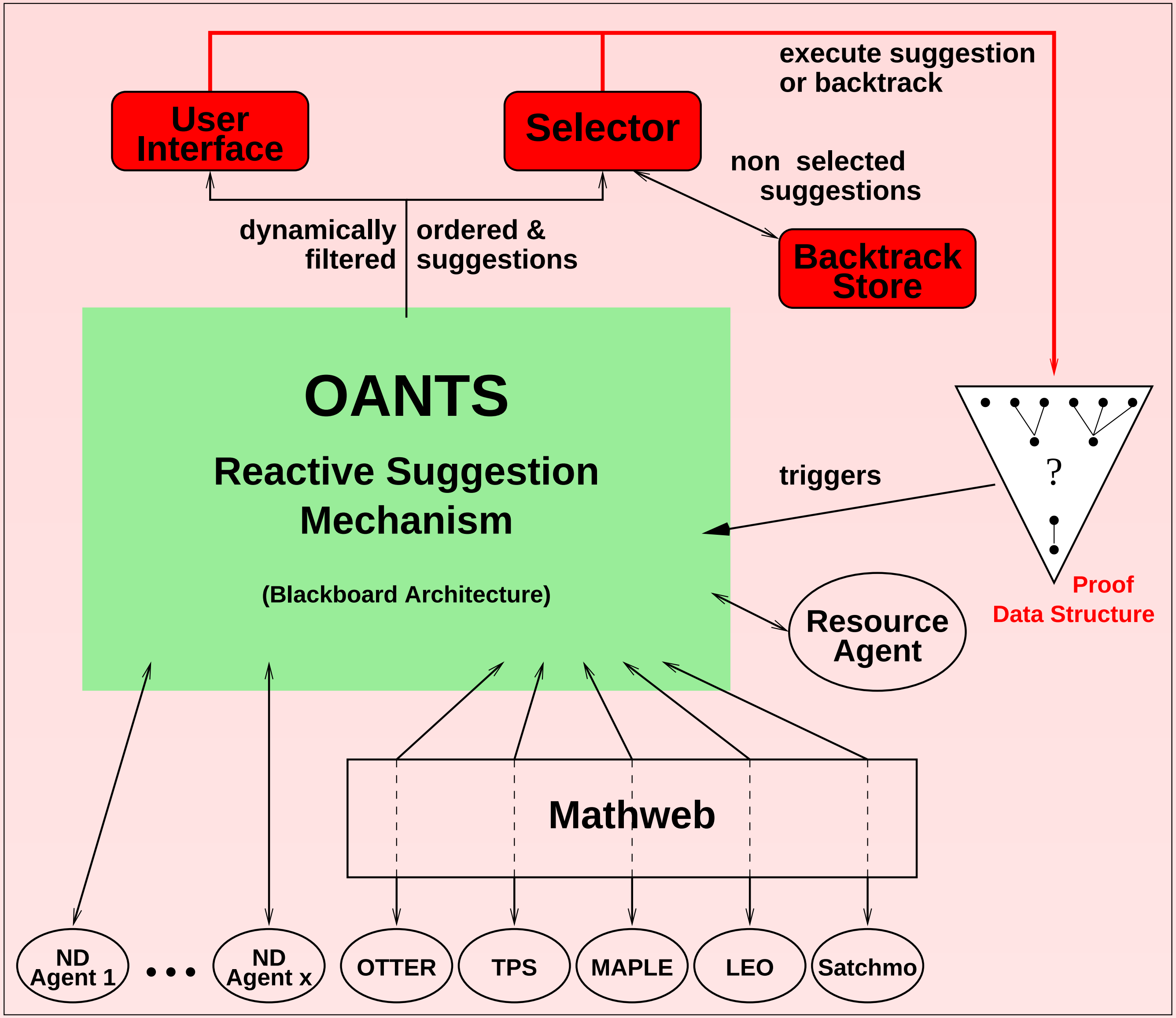
Realisation – Ω-ANTS



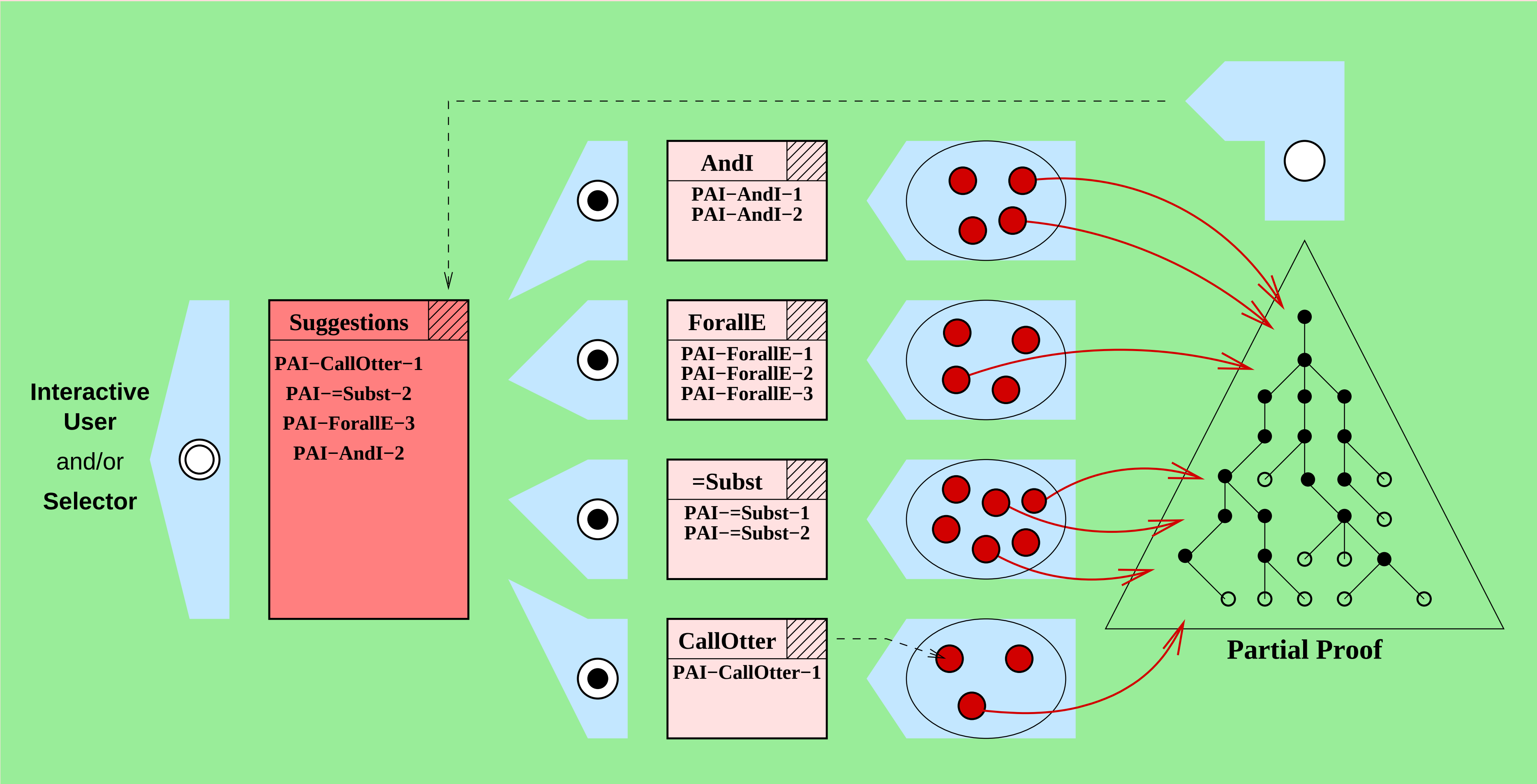
Realisation – Ω-ANTS



Realisation – Ω-ANTS



Realisation – Ω-ANTS



Realisation – Resources

Resource adapted behaviour

- clock speed **low** — automatic proof by single ATP
- clock speed **medium** — cooperative proof
- clock speed **high** — attack at ND level

Experiments: resource adaptivity (in interactive sessions)

agents decide to be inactive/active wrt varying clock speed

Results – Example Classes

Ex1 Higher order ATP and first order ATP

$$\forall x, y, z. (x = y \cup z) \Leftrightarrow (y \subseteq x \wedge z \subseteq x \wedge \forall v. (y \subseteq v \wedge z \subseteq v) \Rightarrow (x \subseteq v))$$

Ex2 ND based TP, propositional ATP, and model generation

$$\forall x. \forall y. \forall z. ((x \cup y) \cap z) = (x \cap z) \cup (y \cap z)$$ **10000 Examples**

$$\forall x. \forall y. \forall z. ((x \cup y) \cup z) = (x \cap z) \cup (y \cap z)$$ **988 valid / 9012 invalid**

Ex3 CAS and higher order ATP

$$\{x | x > gcd(10, 8) \wedge x < lcm(10, 8)\} = \{x | x < 40\} \cap \{x | x > 2\}$$

Ex4 Tactical TP, first-order ATP, CAS, and higher order ATP

. . . *group-definition-1* . . . ⇔ . . . group-definition-2 . . .

Results – ND, PL-ATP, Models

Ωmega

Conc	$\vdash \forall x\boldsymbol{.}\, \forall y\boldsymbol{.}\, \forall z\boldsymbol{.}\, ((x \cup y) \cap z) = (x \cap z) \cup (y \cap z)$	Forall-I L1
	$\ldots$	
L3	$\vdash ((X \cup Y) \cap Z) = (X \cap Z) \cup (Y \cap Z)$	Set-Ext L4
L4	$\vdash \forall e\boldsymbol{.}\, e \in ((X \cup Y) \cap Z) \leftrightarrow e \in (X \cap Z) \cup (Y \cap Z)$	Forall-I L5
L5	$\vdash E \in ((X \cup Y) \cap Z) \leftrightarrow E \in (X \cap Z) \cup (Y \cap Z)$	Def L6
	$\ldots$	
L8	$\vdash ((E \in X \vee E \in Y) \wedge E \in Z) \leftrightarrow ((E \in X \wedge E \in Z) \vee (E \in Y \wedge E \in Z))$	OTTER

Theorem

Results – ND, PL-ATP, Models

Conc	$\vdash \forall x_\bullet \forall y_\bullet \forall z_\bullet ((x \cup y) \cup z) = (x \cap z) \cup (y \cap z)$	Forall-I L1
	$\ldots$	
L3	$\vdash ((X \cup Y) \cup Z) = (X \cap Z) \cup (Y \cap Z)$	Set-Ext L4
L4	$\vdash \forall e_\bullet e \in ((X \cup Y) \cup Z) \leftrightarrow e \in (X \cap Z) \cup (Y \cap Z)$	Forall-I L5
L5	$\vdash E \in ((X \cup Y) \cup Z) \leftrightarrow E \in (X \cap Z) \cup (Y \cap Z)$	Def L6
	$\ldots$	
L8	$\vdash ((E \in X \vee E \in Y) \vee E \in Z) \leftrightarrow ((E \in X \wedge E \in Z) \vee (E \in Y \wedge E \in Z))$	**SATCHMO**

Countermodel: $G \in Z \wedge G \notin X \wedge G \notin Y$

Related Work

- Parallel & distributed theorem proving [Bonacina 2000]
- TECHS & TEAMWORK approach [Denzinger/Fuchs 1999]
  - filtered exchange of clauses between first-order provers
  - no higher-order systems and no CAS
  - no explicit proof object
  - no user orientation
- Concurrent theorem proving [Fisher 1997]
  METATEM (temporal logics) [Fisher 1994]
- Multi agent proof-planning [Fisher/Ireland 1998]
- . . . agent based architectures, layered architectures . . .

Conclusion

Employ agent paradigm

to flexibly combine
very heterogeneous reasoning components
working on conceptually different layers
in a sceptical, centralised approach
that uses an expressive proof representation format.

Architecture is not restricted to theorem proving

Problems and Future Work

- Short-term goals
  - counterexamples: illustration (Venn-diagrams) & early backtracking
  - more & better agents; more case studies
- Long-term goals
  - decentralisation
  - dynamic clustering
  - communication bottleneck
  - agent interlingua
  - or-parallelism
  - integration with proof planning
  - critical (reflecting) agents

Results – ND, CAS, HO-ATP

$$\{x|x > gcd(10,8) \wedge x < lcm(10,8)\} = \{x|x < 40\} \cap \{x|x > 2\}$$

Conc	$\vdash (\lambda x. x > gcd(10,8) \wedge x < lcm(10,8)) = (\lambda x. x < 40) \cap (\lambda x. x > 2)$	**CAS** L1
L1	$\vdash (\lambda x. x > 2 \wedge x < 40) = (\lambda x. x < 40) \cap (\lambda x. x > 2)$	Def L3
L3	$\vdash (\lambda x. x > 2 \wedge x < 40) = (\lambda x. x < 40 \wedge x > 2)$	**LEO**