Three Approaches for Guiding the Cooperation of Mathematical Reasoning Systems:

Proof Planning, Agent-based Reasoning, and Automated Composition of Reasoning Web Services

Serge Autexier, Christoph Benzmüller

{serge|chris}@ags.uni-sb.de

CS Department, Saarland University, Saarbrücken, Germany

QSL Day, February 10, 2005

LORIA, Nancy, France

Overview

- Context & Motivation
- Hierarchical PDS: supports flexible
  1. represention of results obtained from external systems, and
  2. their verification
- Orchestration of external systems via
  - ...deliberative knowledge-based proof-planning MULTI
  - ...reactive agent-based reasoning OANTS
- Framework for offering and accessing external systems

  MATHWEB-SB

  + the next generation MATHSERV

Integration of Reasoning Systems: Motivation

Source: Autexier, Benzmüller

Motivation – Cognitive Perspective

To solve complex mathematical problems

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

A single mathematician

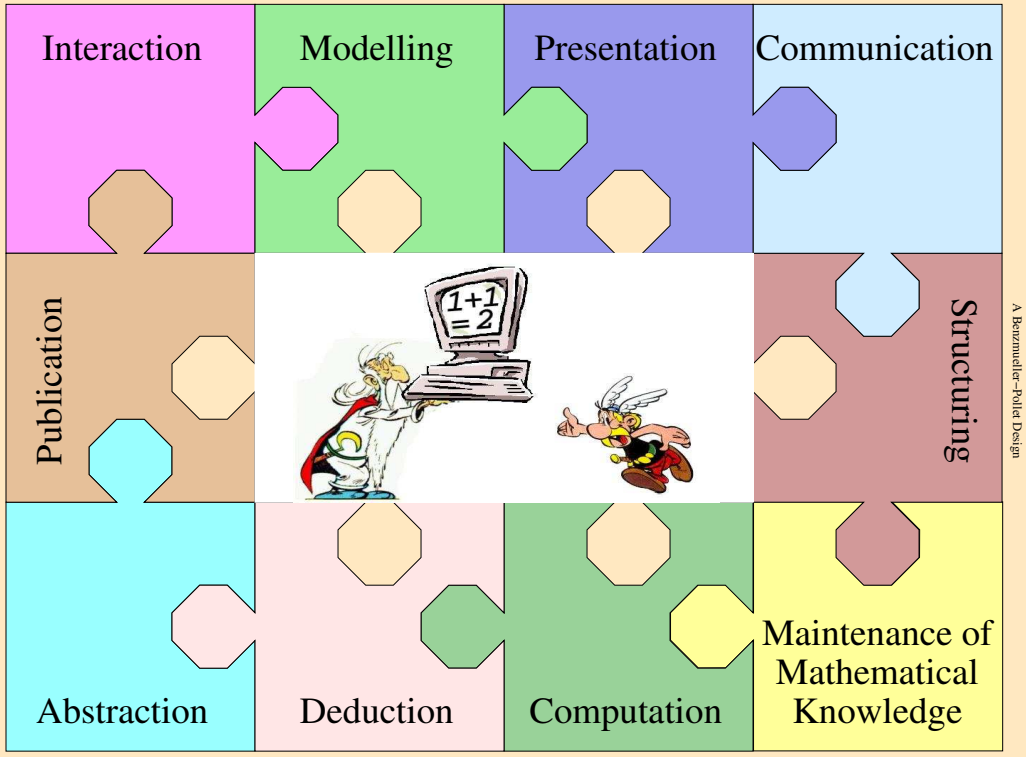
- possesses a large repertoire of specialized reasoning and problem solving techniques
- uses experience and intuition to flexibly combine them in an appropriate way

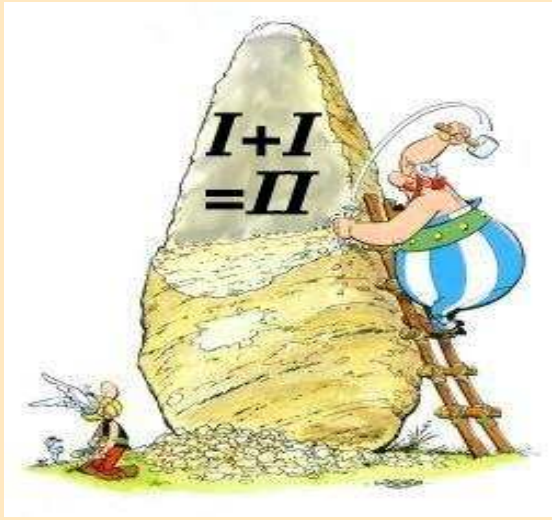
Mathematics Assistance Systems

Integrated Mathematics Assistance Environment

vs.

'Pen-and-Paper' Mathematics





Applications

Mathematics research
Mathematics education
Formal methods

Join of ressources necessary

System level: Coq, NuPrl, Isabelle/HOL, PVS, Theorema, ΩMEGA, CLam, ...

Research Networks: Calculemus, MKM, Monet, MoWGLI, ...

Source: Autexier, Benzmüller

CALCULEMUS Network Partners

USAAR	Jörg Siekmann, Christoph Benzmüller, Serge Autexier
UED1	Alan Bundy, Ewen MacLean
UKA	Jacques Calmet, Regine Endsuleit
RISC	Bruno Buchberger, Wolfgang Windsteiger, Tudor Jebelean
TUE	Arjeh Cohen, Henk Barendregt, Herman Geuvers Freek Wiedejk
ITC-IRST/DIT	Fausto Giunchiglia, Roberto Sebastiani, Alessandro Cimatti, Marco Bozzano
UWB	Andrzej Trybulec, Czeslaw Bylinski, Grzegorz Bancerek
UGE	Alessandro Armando, Enrico Giunchiglia
UBIR	Manfred Kerber, Volker Sorge

CALCULEMUS Sociological Goal

Early stage training of young researchers

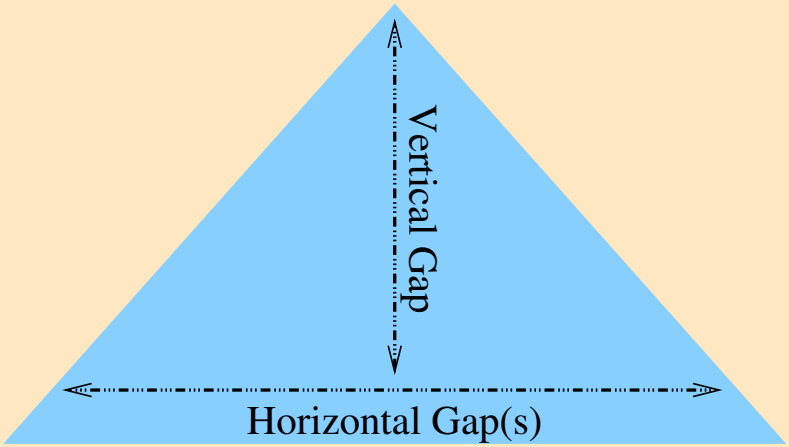
Measures:

- The CALCULEMUS Autumn School 2002
- Symposia and Network Meetings
- Training at an Individual Level at the Network Nodes
- Local Courses, Workshops, Talks, and Seminars
- Exchange of YVRs between Network Nodes
- Industry Internships

Source: Autexier, Benzmüller

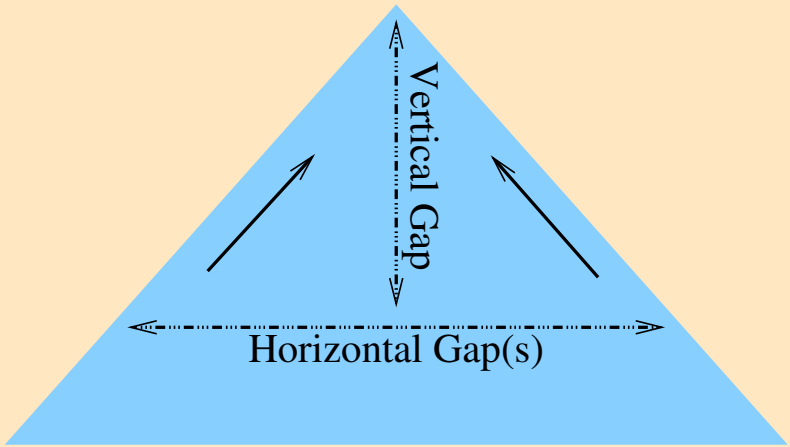
CALCULEMUS Methodology

Vision:
Powerful mathematical assistant system(s)



Reality:
heterogeneous frameworks, systems, and tools
individual strengths and weaknesses

Vision:
powerful mathematical assistant system(s)



Bottom-up from: CAS, DS, KBs, . . .
When to integrate modules and when to re-implement?

CALCULEMUS Results

Training

- 43 Young Researchers
- 15 Nations

Dissemination

- Related Publications of Consortium: approx. 350
- With Young Researchers as Co-Authors: approx. 150
- Joint Publications: approx. 100

Results

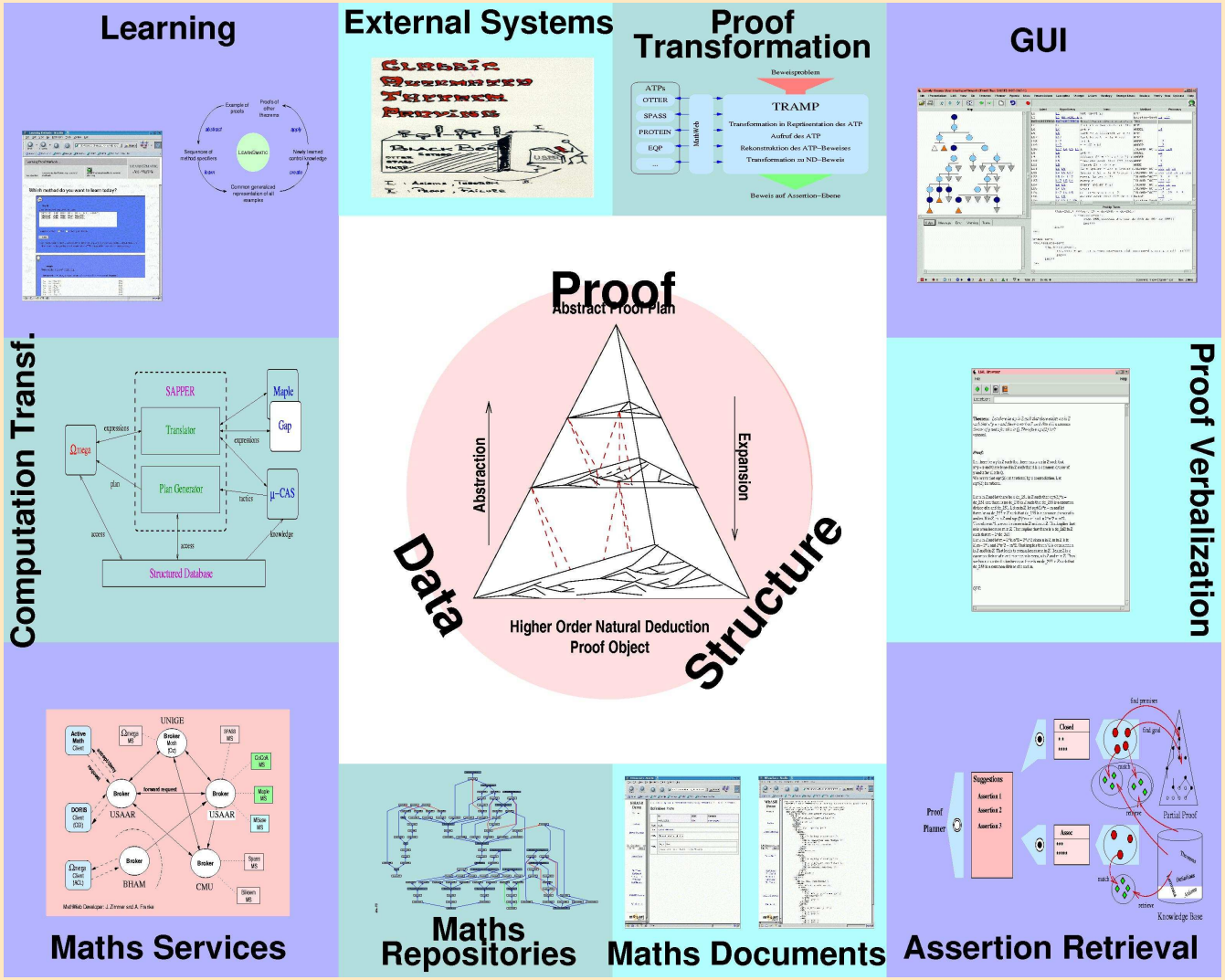
- Single predominant theoretical framework for the integration of DS and CAS is currently not possible.
- One reason: Diverging approaches for DSs and CASs.
- Focus: Integration of CASs and DSs at the systems layer.

Source: Autexier, Benzmüller

Philosophy of ΩMEGA: White Box Integration supported by the PDS and proof transformation

Source: Autexier, Benzmüller

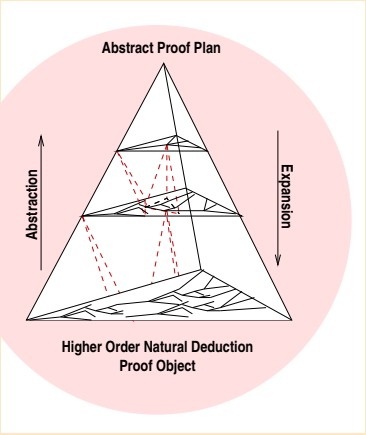
The ΩMEGA System



External Specialist Reasoners

Usually required in OMEGA:

- white box integration of external specialist reasoners
- tools for extraction and transformation of results



problem

proof(e.g. ND)

Transformation Tools

Proof Transformation

Proof Reconstruction

problem'

result(cryptic)

External Specialist Reasoner

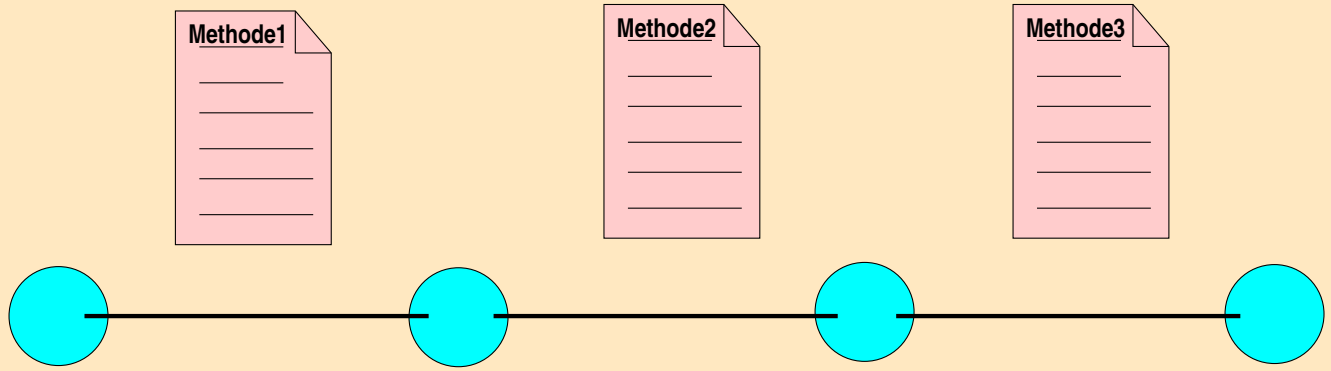
available for:

FOL ATPs (TRAMP), CASs (SAPPER), TPS, constraint solving

Deliberative Orchestration: Knowledge-Based Proof-Planning

Source: Autexier, Benzmüller

Knowledge Based Proof Planning



ΩMEGA aspects (inspired by and extending [Bundy88]):

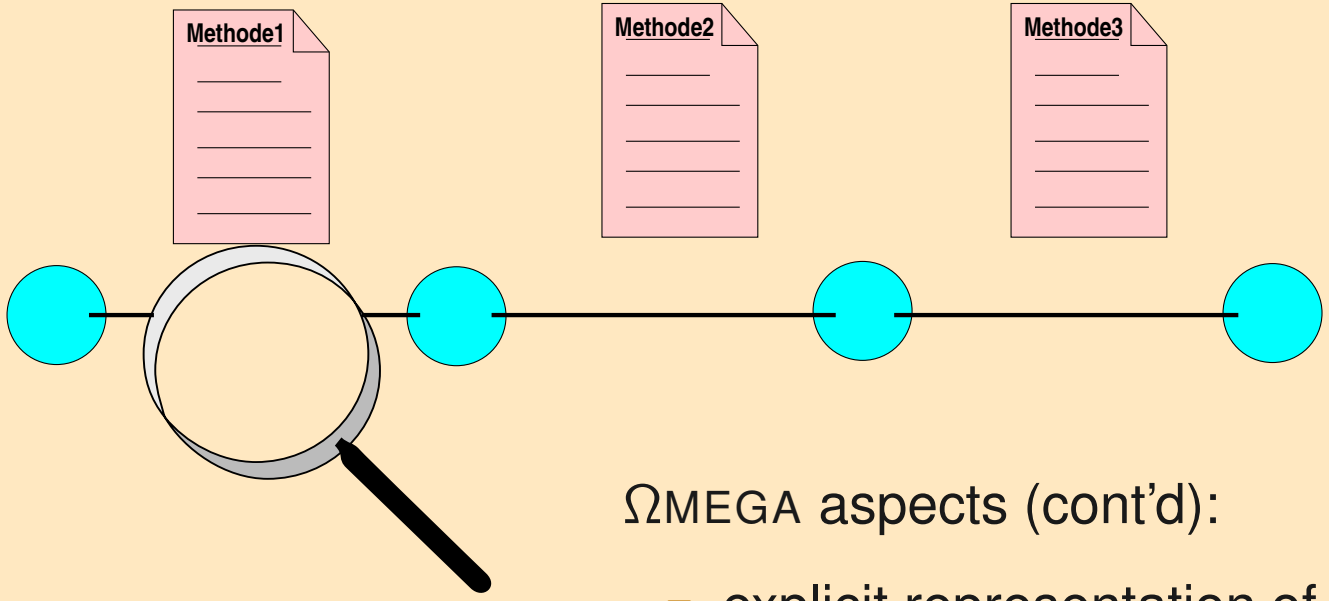
- declarative, domain specific control layer
- strategy = domain specific instantiation of a general proof search algorithm with set of proof methods and control information
- multi-strategy proof planning

Use of External Reasoners (since 1995): [ECAI98,AI99,JSC02,MeierPHD04]

- in proof methods + at control layer + in plan expansion

Source: Autexier, Benzmüller

Knowledge Based Proof Planning

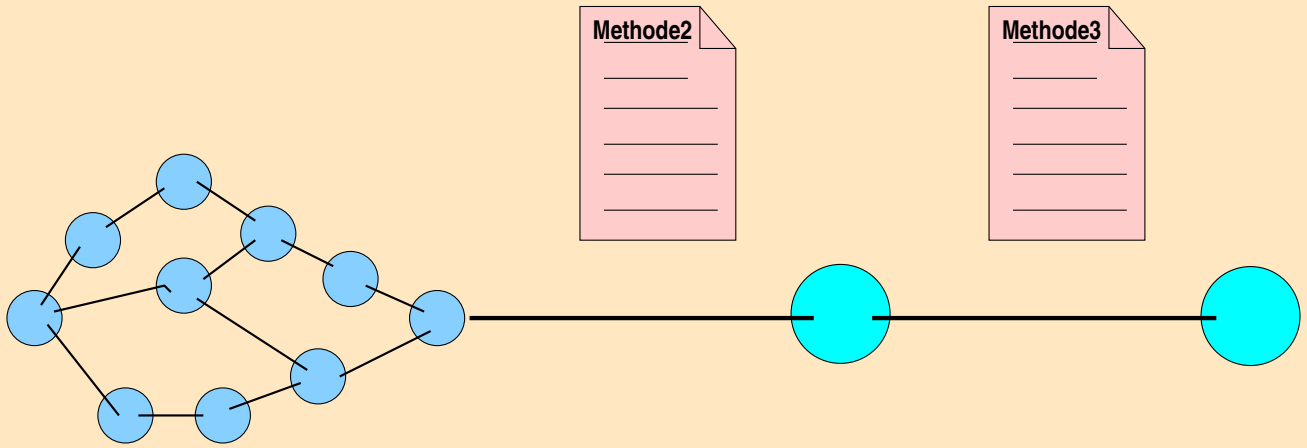


ΩMEGA aspects (cont'd):

- explicit representation of proof plans
- under-specification of pre-conditions: potentially non-sound proof plans
- soundness guaranteed via . . .

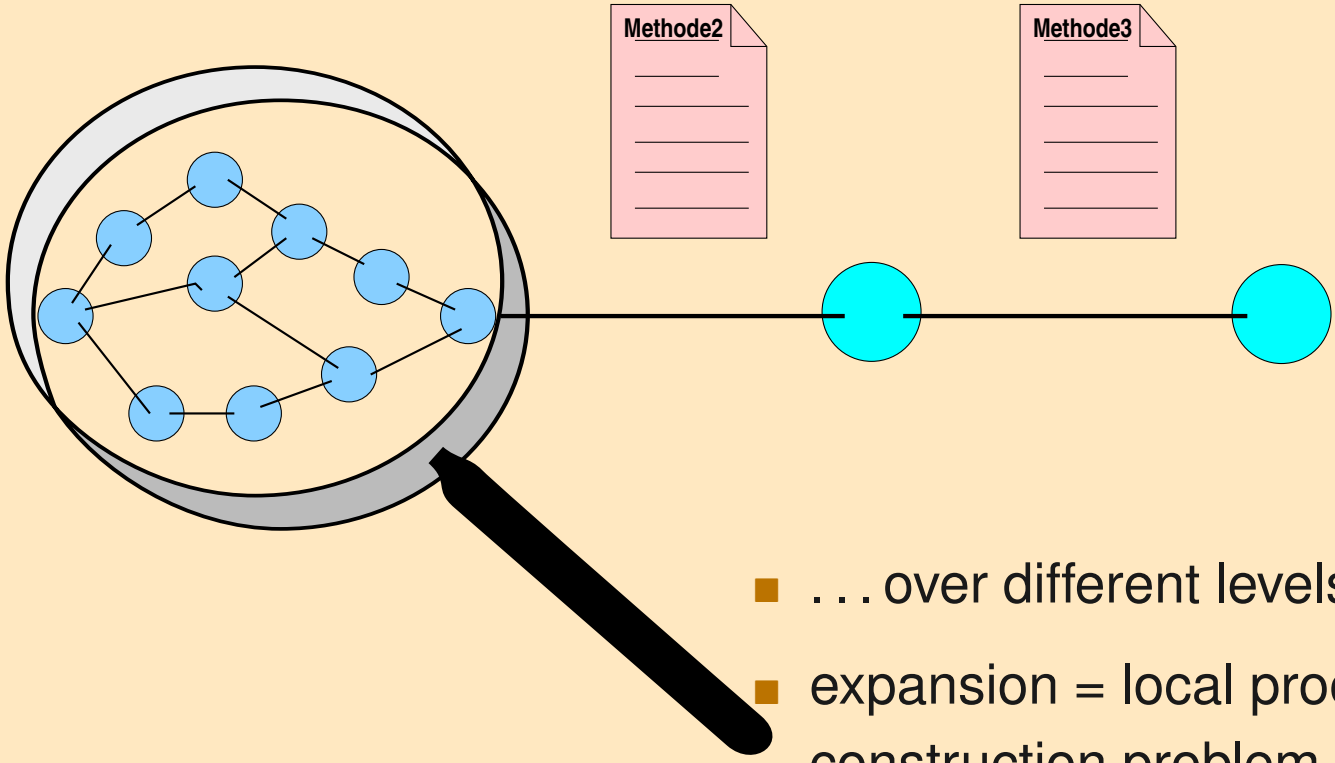
Source: Autexier, Benzmüller

Knowledge Based Proof Planning



- . . . proof (plan) expansion over . . .

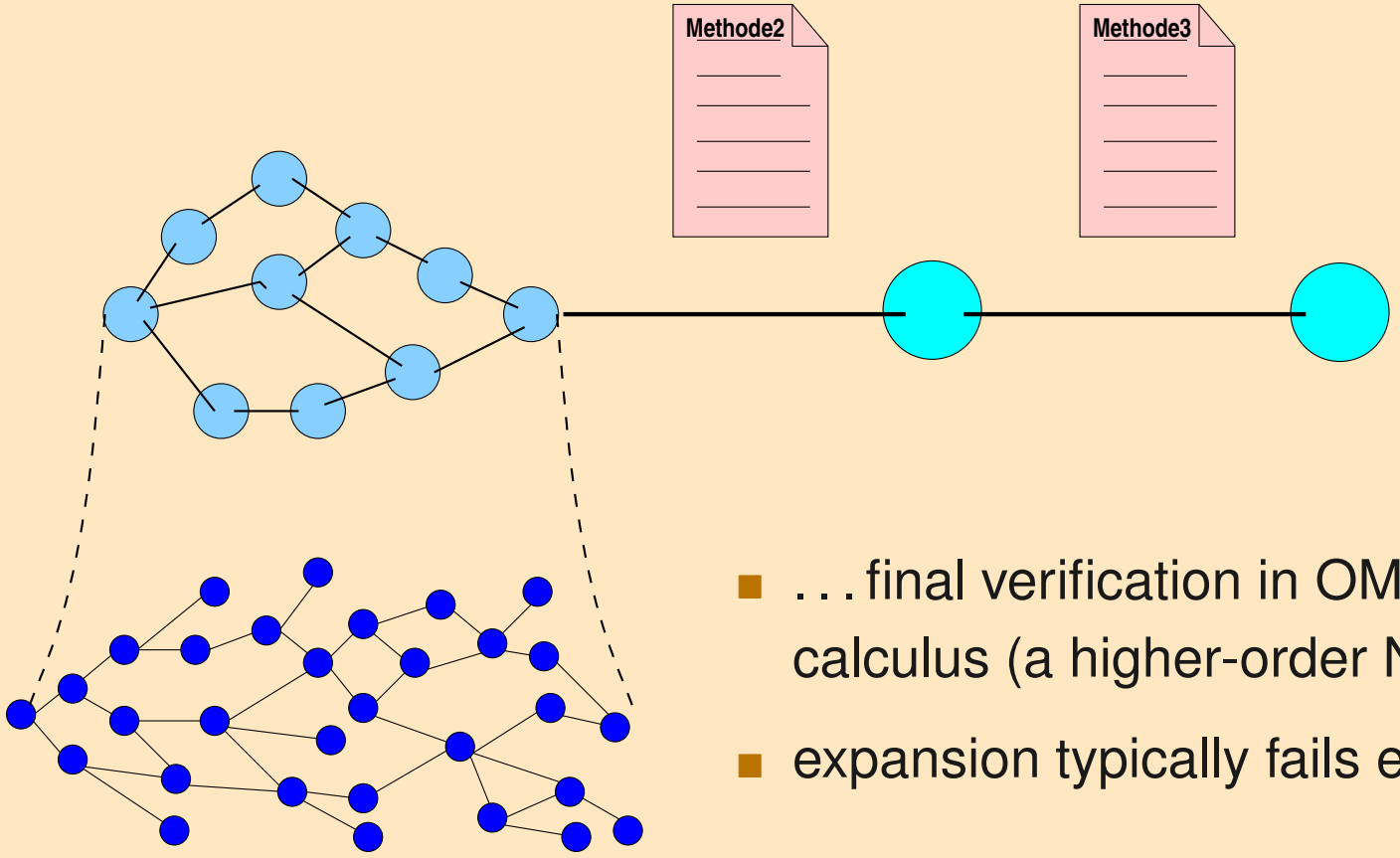
Knowledge Based Proof Planning



- . . . over different levels of granularity
- expansion = local proof search / proof construction problem
- may employ external reasoners

Source: Autexier, Benzmüller

Knowledge Based Proof Planning



- . . . final verification in OMEGAs base calculus (a higher-order ND variant)
- expansion typically fails early!

Source: Autexier, Benzmüller

Knowledge Based Proof Planning

Case studies

ε-δ-proofs: Use of constraint-solver and computer algebra system

Residue class properties: combination of CASs and theory formation system HR, classification of $\sim$18.000 structures

Verification of GAP computations on permutation groups: Verification by proof search instead of hard-wired scripts

Discussion

+ Automatic proofs for problem classes in specific domains

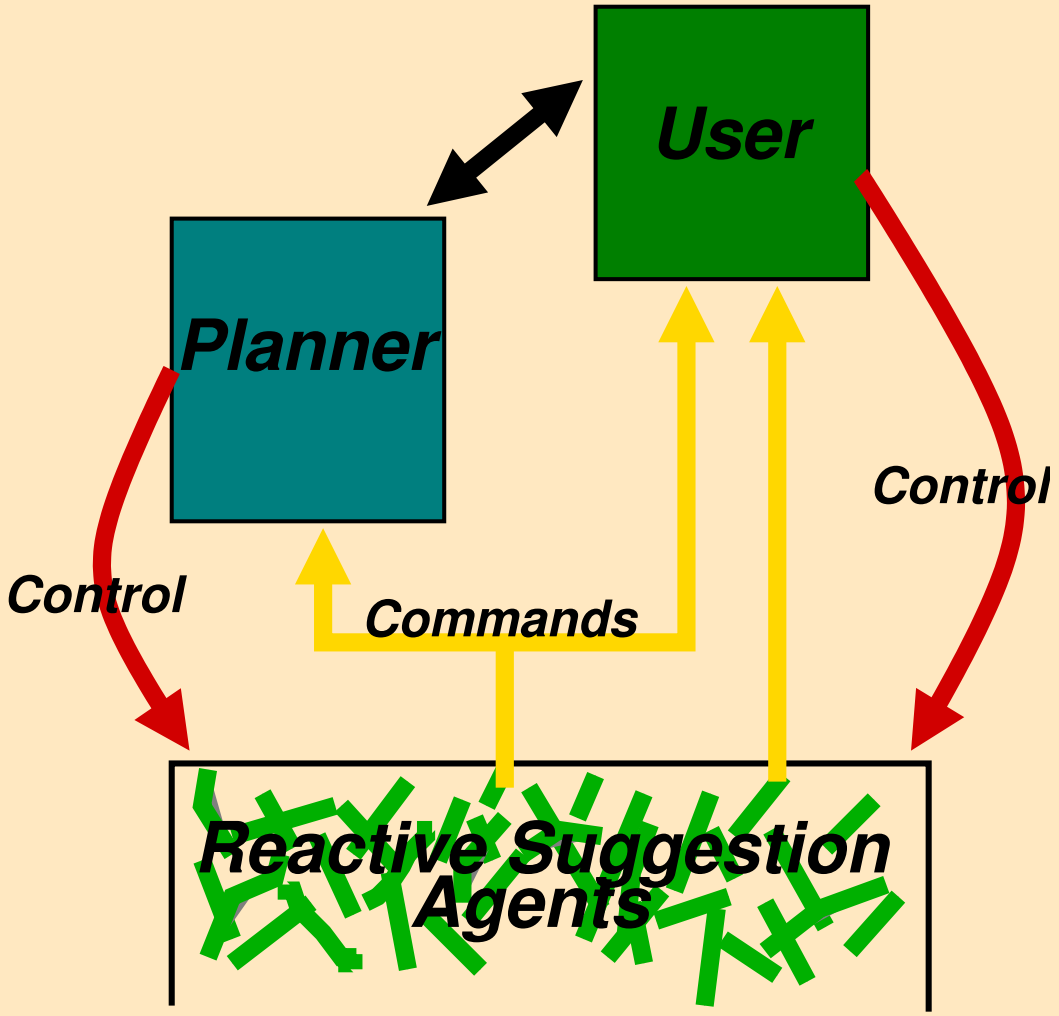
+ Coordination of systems

− Brittleness and logic layer dependency

Reactive Orchestration: Agent-based Reasoning

Source: Autexier, Benzmüller

Agent-based Reasoning: OANTS



- pro-active support versus passive support
- concurrent versus sequential

(since 1997)

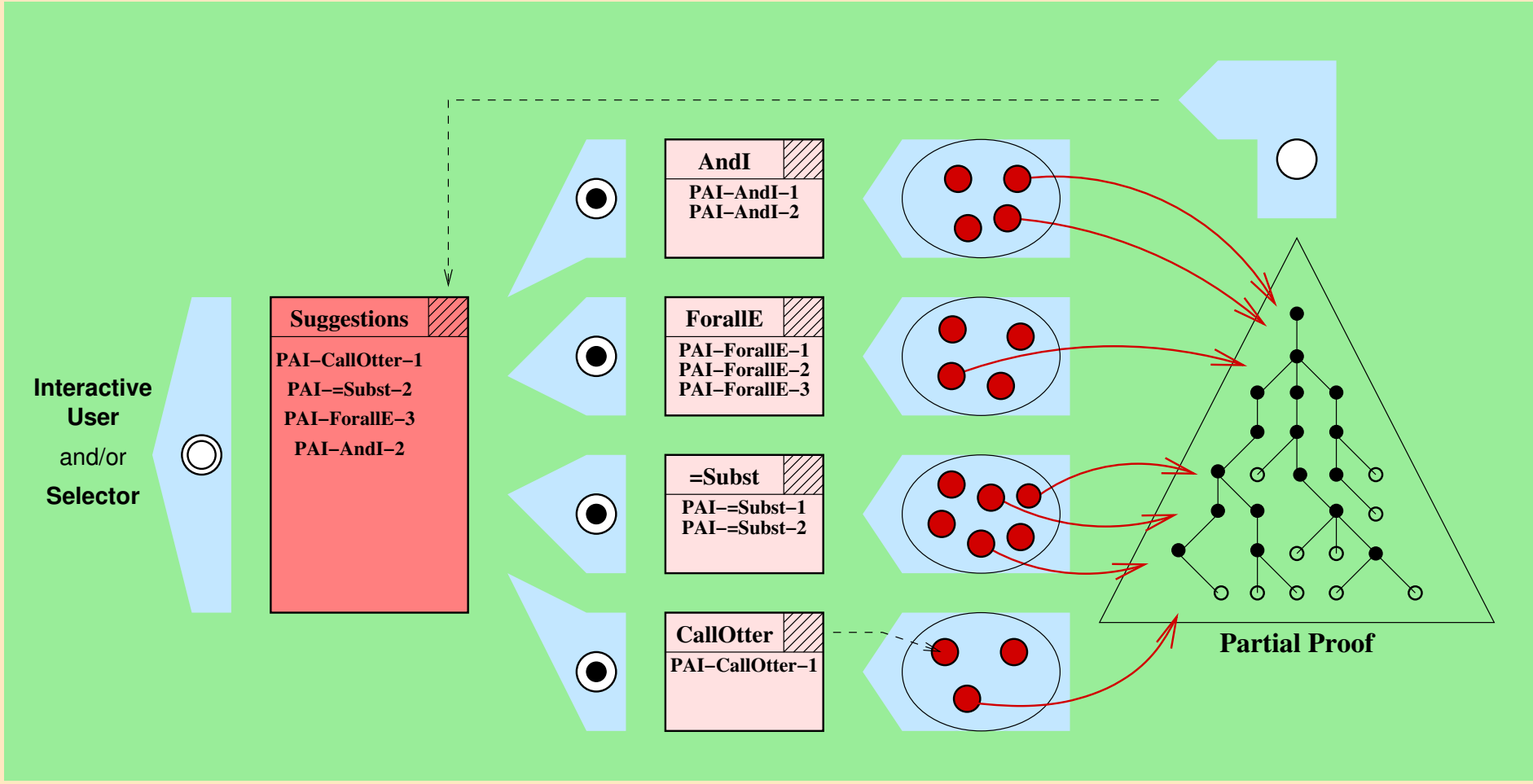
Source: Autexier, Benzmüller

Agent-based Reasoning: OANTS

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

- Distributed applicability checks for:
  proof rules, tactics, external systems, etc.
- Further distributed processes for single parameter or parameter combinations
- Two layered blackboard architecture
- Anytime character
- Publications: [AIMSA98,EPIA99,CALCULEMUS00,KI01,LPAR05]

Agent-based Reasoning: OANTS



Agent-based Reasoning: OANTS

Adding an *new* external system (or rule, tactic, method)

- simply provide a new command

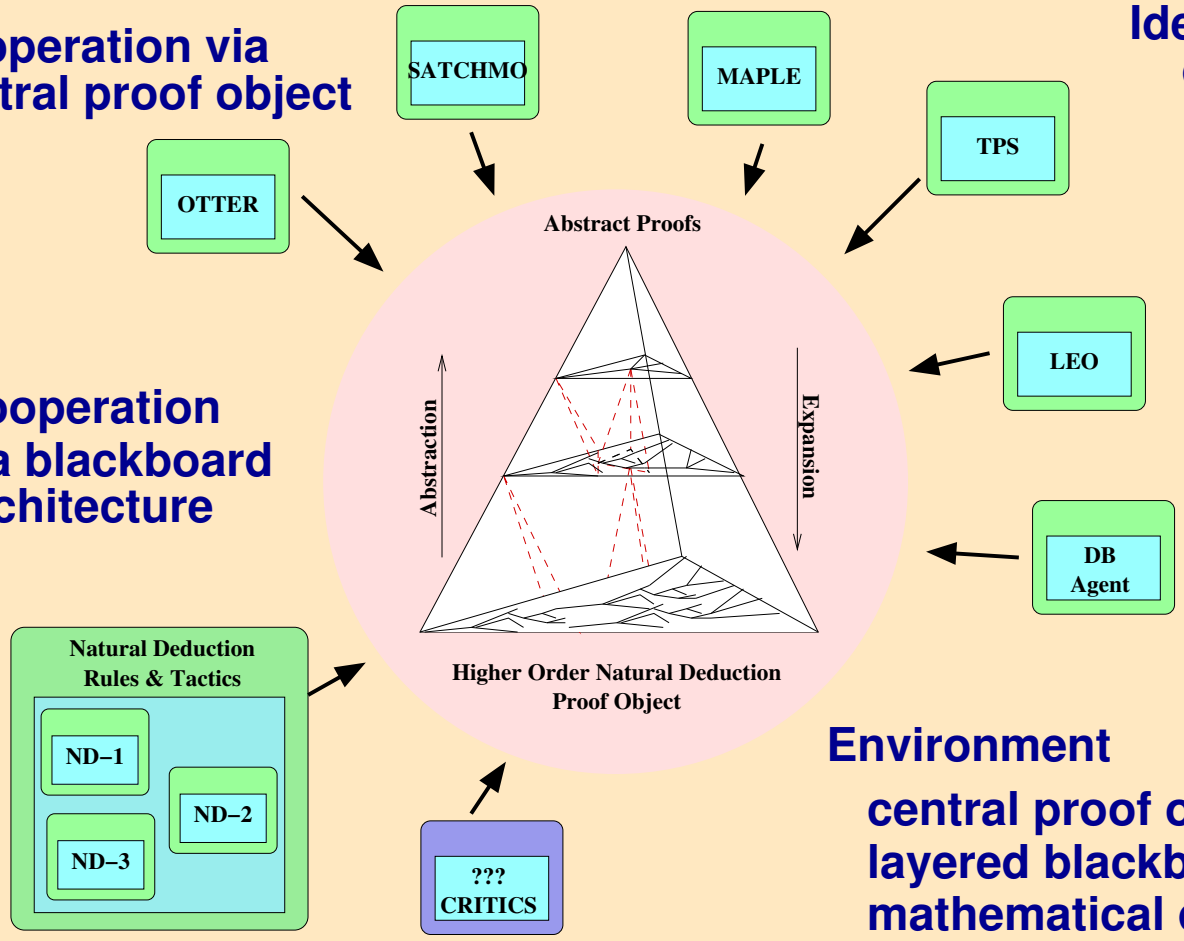
$$\frac{\texttt{Ass-1:}A_1 \quad \ldots \quad \texttt{Ass-n:}A_n}{\texttt{Conc:}C}\ \texttt{OTTER}(\texttt{P-1:}f_1,\ldots,\texttt{P-m:}f_m)$$

- define parameter agents
- adapt heuristic filter (utility function)
- $\Rightarrow$ command agent & blackboard, etc. are created automatically
- optional (but important for sceptical approach): proof/computation transformation module

Agent-based Reasoning: OANTS

Cooperation via central proof object

Cooperation via blackboard architecture



ND–Backbone

Idea
- decentralised control

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

Source: Autexier, Benzmüller

Example – HO- and FO-ATP

fn = favorite-numbers

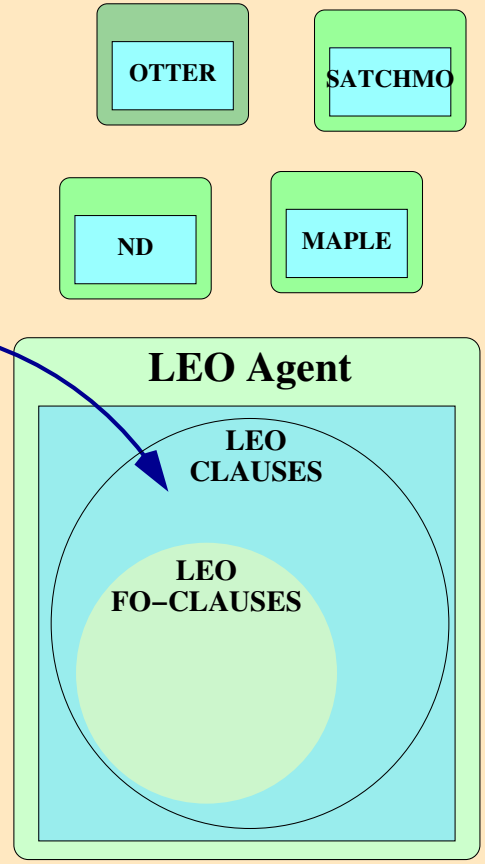
o = odd

s = square

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

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

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

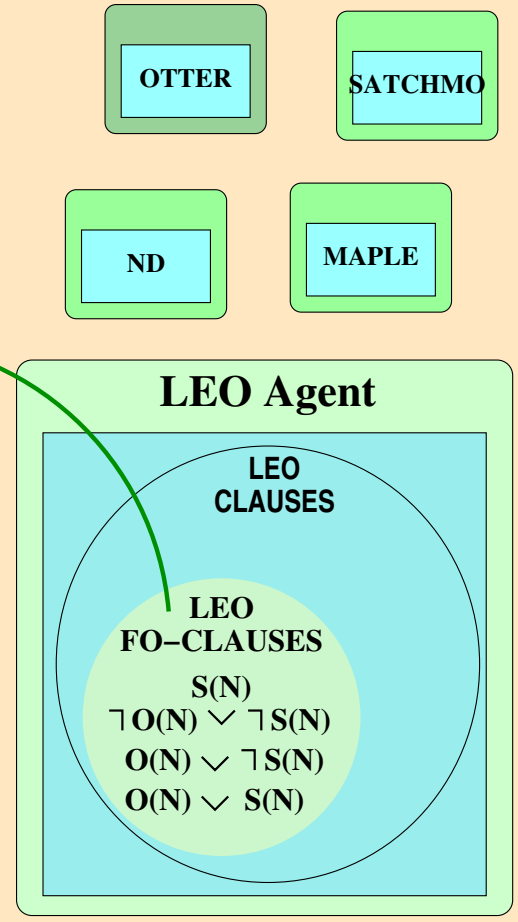


Example – HO- and FO-ATP

$$\dfrac{\begin{array}{c}\mathsf{fn}(\lambda x.\, o(x) \wedge s(x)) \\ \neg\mathsf{fn}(\lambda x.\, s(x) \wedge (s(x) \Rightarrow o(x)))\end{array}}{\begin{array}{l}s(N) \wedge (\neg o(N) \vee \neg s(N)) \wedge (o(N) \vee \neg s(N)) \\ \qquad\qquad \wedge\ (o(N) \vee s(N))\end{array}}\ \text{LEO}$$

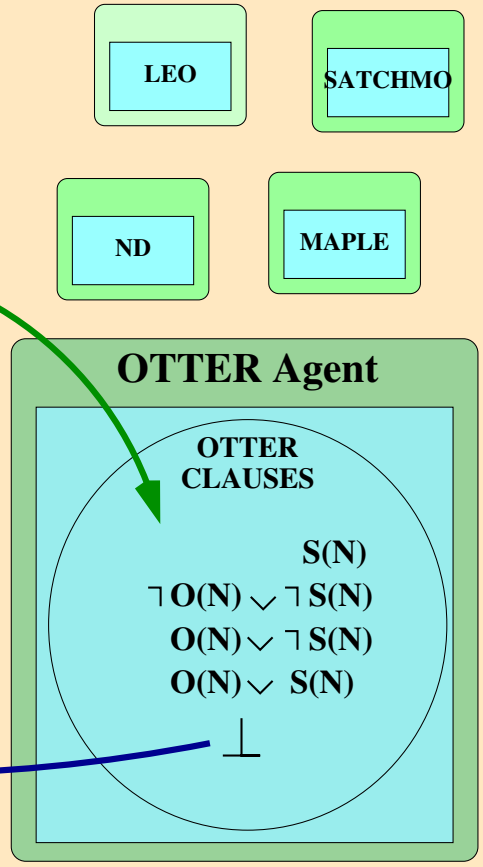
$$\vdots\ ?$$

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



Example – HO- and FO-ATP

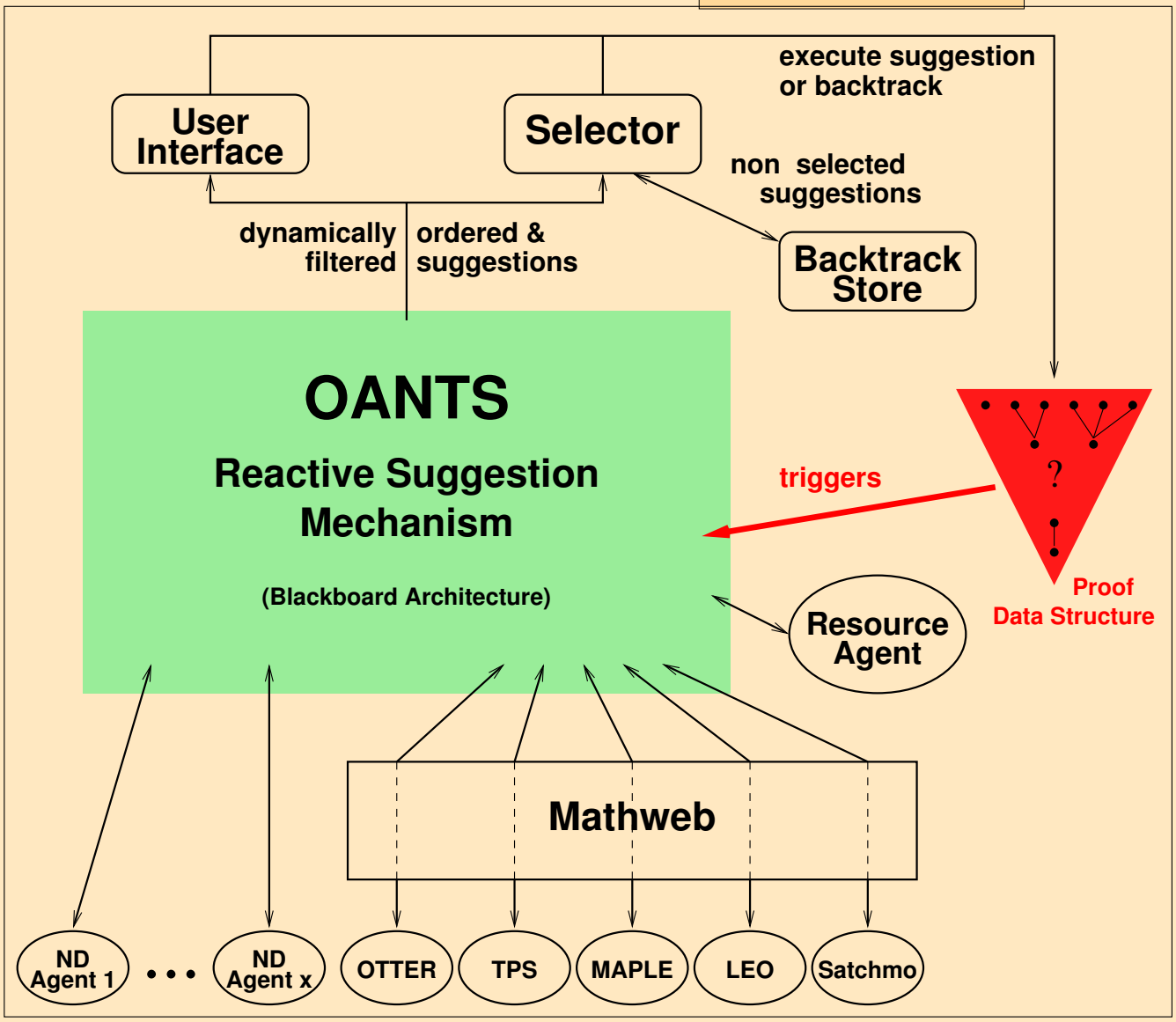
$$\dfrac{\dfrac{\dfrac{\mathsf{fn}(\lambda x.\, o(x) \wedge s(x)) \qquad \neg\mathsf{fn}(\lambda x.\, s(x) \wedge (s(x) \Rightarrow o(x)))}{s(N) \wedge (\neg o(N) \vee \neg s(N)) \wedge (o(N) \vee \neg s(N)) \wedge (o(N) \vee s(N))}\ \text{LEO}}{\bot}\ \text{OTTER}}{\mathsf{fn}(\lambda x.\, s(x) \wedge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



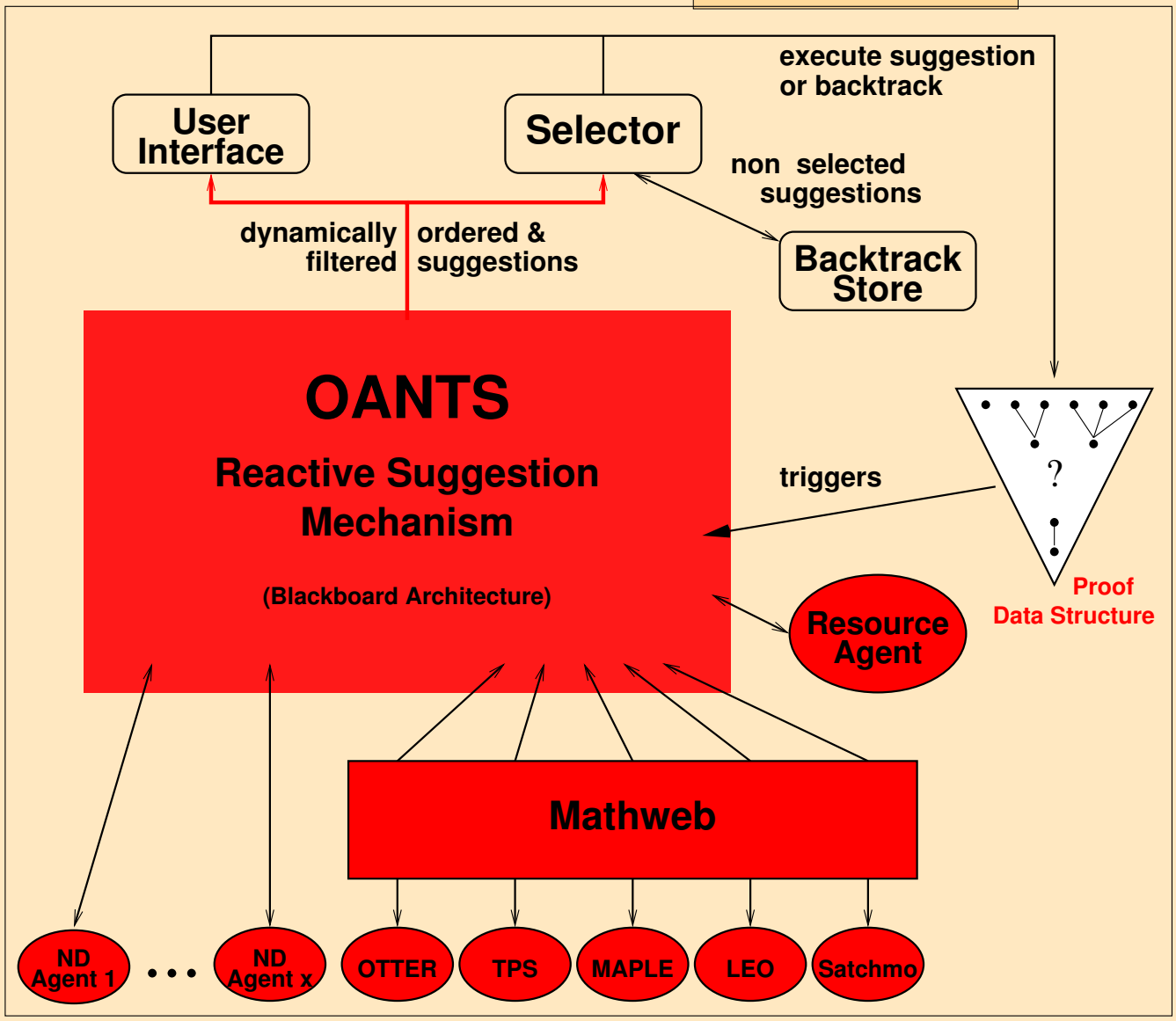
Realization – Main Components

- ΩMEGA-system data structures
  ΩMEGA's proof data structure PDS (natural deduction with abstraction facilities)
- OANTS blackboard architecture (developed since 1997)
- MATHWEB-system
- Various external systems integrated via MATHWEB
- Translation modules like
  - TRAMP (machine oriented first-order ATP $\Longrightarrow$ natural deduction)
  - SAPPER (symbolic computation $\Longrightarrow$ natural deduction)
- Calculus NIC (CMU) for efficient ND proof search

Realization – Automating OANTS

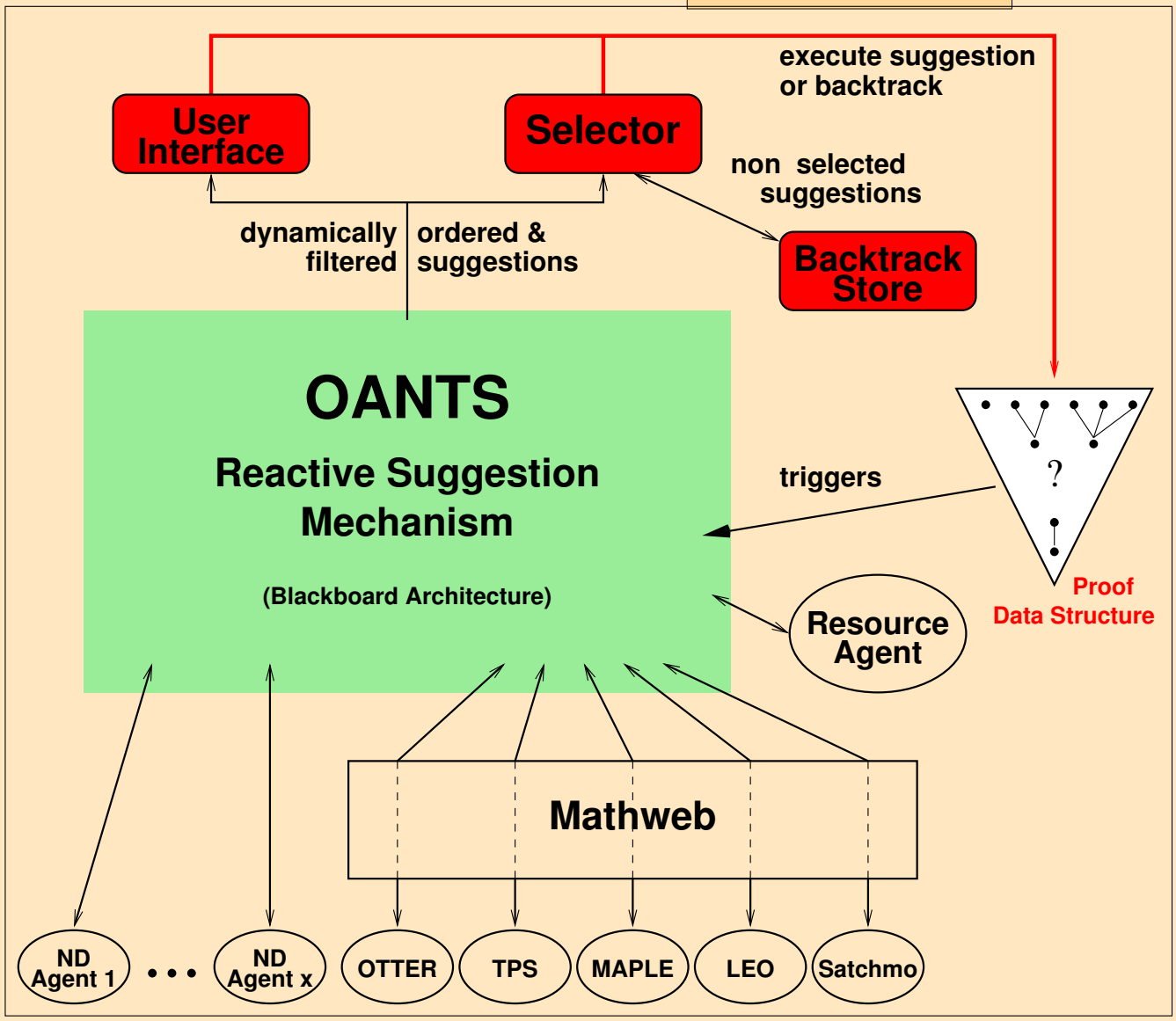


Realization – Automating OANTS



Source: Autexier, Benzmüller

Realization – Automating OANTS



Realization – Resource Aspects

Resource adapted behavior

- 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 via multiple inference rules

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

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

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

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

Ex5 Higher order ATP and first order ATP via a single inference rule

45 examples for SET (TPTP); outperforms all FO-ATPs [LPAR05]

The MathWeb Software Bus for Distributed Mathematical Reasoning

(`http://www.mathweb.org/mathweb/`)

Source: Autexier, Benzmüller

Goal

- Use external (mathematical) systems from within a mathematical assistance systems (MASs)
- Examples of systems:
  - External, automated theorem provers *TPS, Vampire, Otter, Spass, Bliksem, Waldmeister, EPROVER, EQP, Fdpll, Protein*
  - Computer Algebra Systems *MAPLE, MAGMA, GAP, COCOA*
  - Model generators *Mace, Kimba*
  - Theory exploration tool *HR*
  - Mathematical Knowledge Bases *MBASE*

The Situation in 1995

- For *each* MAS and *each* external system a specific integration must be implemented
- Need to know how to call the external system, etc.
- A lot of work if you want to integrate an external system
- Consequence for MAS: Simply downloading and installing the MAS is *not* sufficient to be able to use the MAS!

Monolithic systems with heavy-tailored integrations of external systems

Source: Zimmer, Autexier, Benzmüller

Difficulties

- *Installation:*
  - The system must be available for your OS
  - You have to install the individual systems
  - You may need to get a license
- *Configuration:*
  - You must configure the MAS to your local installation
- *Invocation:*
  - Need to know how and with which parameter to call each system
  - What are the “services” a system can provide?
  - How to interpret the output returned by the systems;
    With which exit status did the service terminate?
    What are its answers?

Source: Zimmer, Autexier, Benzmüller

Difficulties (cont'd)

- *Language:*
  - Need to know the interface syntax for each individual system
  - Need to write syntax transformations towards the system and backwards into the MAS
  - Are the transformations correct?

Source: Zimmer, Autexier, Benzmüller

Steps Towards MathWeb-SB

- Step 1: Instead of having to do the integrations for *many* systems, do them once for one system B via which we can call the other systems

Steps Towards MathWeb-SB

- Step 1: Instead of having to do the integrations for *many* systems, do them once for one system B via which we can call the other systems
- Step 2: Provide remote access to system B via the internet

  (like a web-server)

Source: Zimmer, Autexier, Benzmüller

Steps Towards MathWeb-SB

- Step 1: Instead of having to do the integrations for *many* systems, do them once for one system B via which we can call the other systems
- Step 2: Provide remote access to system B via the internet

  (like a web-server)
- Step 3.a (Availability): Make system B available at different locations

Steps Towards MathWeb-SB

- Step 1: Instead of having to do the integrations for *many* systems, do them once for one system B via which we can call the other systems
- Step 2: Provide remote access to system B via the internet

  (like a web-server)
- Step 3.a (Availability): Make system B available at different locations
- Step 3.b (Robustness):

Steps Towards MathWeb-SB

- Step 1: Instead of having to do the integrations for *many* systems, do them once for one system B via which we can call the other systems
- Step 2: Provide remote access to system B via the internet

  (like a web-server)
- Step 3.a (Availability): Make system B available at different locations
- Step 3.b (Robustness):
  - Make the different instances of system B know each other

Steps Towards MathWeb-SB

- Step 1: Instead of having to do the integrations for *many* systems, do them once for one system B via which we can call the other systems
- Step 2: Provide remote access to system B via the internet

  (like a web-server)
- Step 3.a (Availability): Make system B available at different locations
- Step 3.b (Robustness):
  - Make the different instances of system B know each other
  - If some external system E is currently not available at location *SB*, then enable the system B_{SB} to access the system E via a *known* system B at a different location. *Networking*

Steps Towards MathWeb-SB

- Step 1: Instead of having to do the integrations for *many* systems, do them once for one system B via which we can call the other systems
- Step 2: Provide remote access to system B via the internet

  (like a web-server)
- Step 3.a (Availability): Make system B available at different locations
- Step 3.b (Robustness):
  - Make the different instances of system B know each other
  - If some external system E is currently not available at location *SB*, then enable the system B_{SB} to access the system E via a *known* system B at a different location. *Networking*
  - No need to install *all* external systems at *each* location

Simplifying Life Step by Step

- Step 1: Instead of having to do the integrations for *many* systems, do them once for one system B via which we can call the other systems
- Use standards for specifying system calls
- Only need to know the syntax how to execute system B

Source: Zimmer, Autexier, Benzmüller

Service Specifications

All first-order ATPs...

- . . . offer the same interface: `prove(ProblemString Syntax Timeout)`
- . . . accept standard problem formats TPTP and OMDoc
- . . . return one of specified exit states

exit state	For input formula F reasoning system R has
`proof`	found a proof for the problem
`unsatisfiable`	determined that the problem is unsatisfiable
`determined`	determined one of the above
`error`	detected an error (e.g., incorrect problem description)
`search-exhausted`	the prover cannot go on with the search (e.g. SoS empty)
`syntax-error`	could not understand the syntax of the problem
`timeout`	used up a given time resource and is not yet `determined`
`undetermined`	could not determine the state but there was no `error` or c`timeout`

Simplifying Life Step by Step

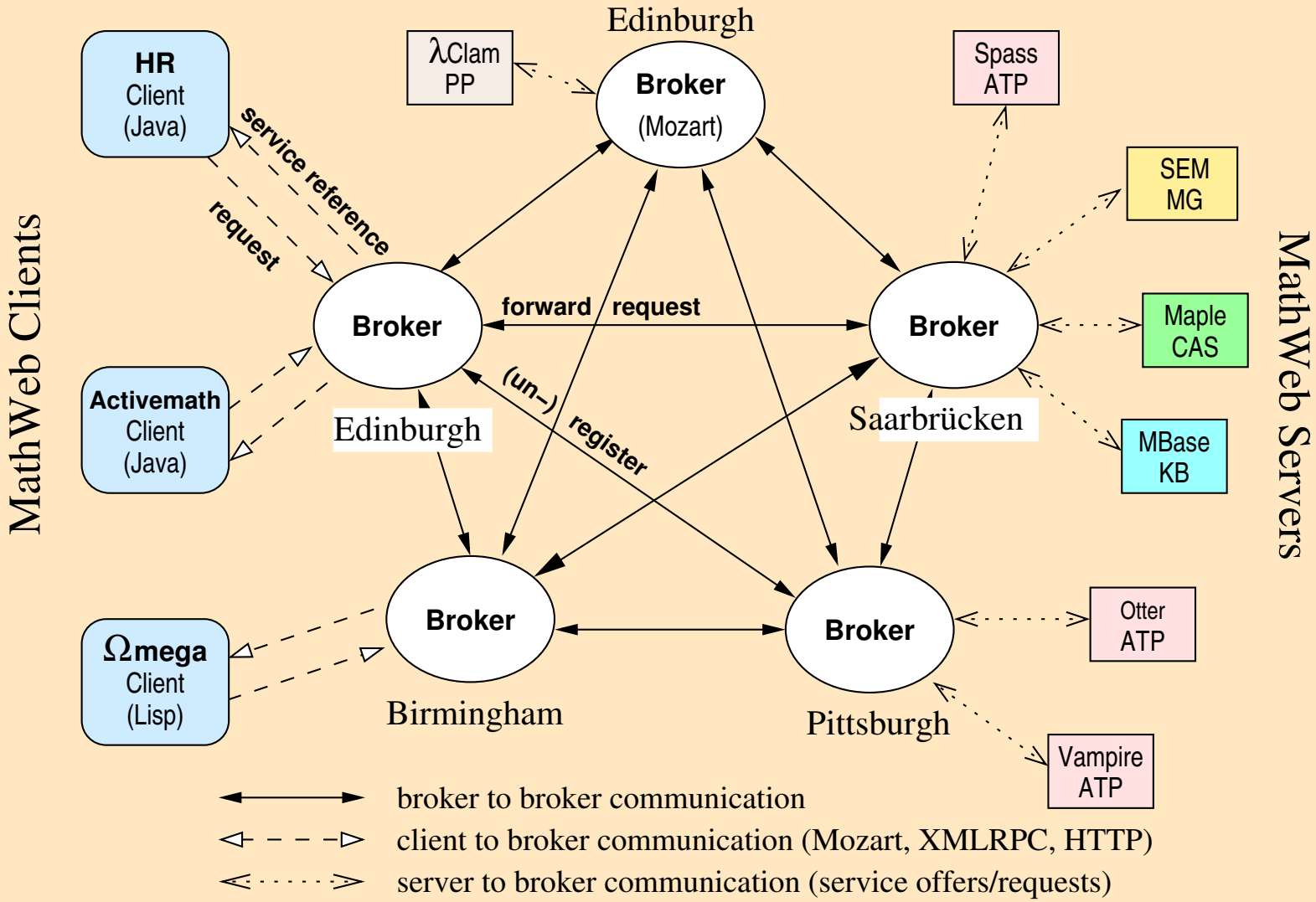
- Step 2: Provide remote access to system B via the internet
- Use standards to make offering and accessing of services language independent!
- MatHWeb-SB HTTP server
  $\Rightarrow$ Service access via HTML forms human consumption
- MatHWeb-SB XML-RPC interface (XML-encoded Remote Procedure Calls):
  - Simple & easy to use,
  - $>$50 implementations in many programming languages

Sample XML-RPC to MathWeb-SB


```
<methodCall><methodName> Broker.getService</methodName>
<params><param><value><string> SPASS</string></value></param></params>
</methodCall>

<methodCall><methodName> prove</methodName>
<params><param><struct>
<member><name>1</name><value><string>
 include('Axioms/EQU001+0.ax').
  include('Axioms/GRP004+0.ax').
  input_formula(conjecture118,conjecture,(! [B,C,D] :
  ((equal(inverse(B),C) & equal(multiply(C,B),D) ) <=>
  (equal(multiply(B,C),D) & equal(multiply(C,B),D) & equal(inverse(C),B))))).
</string></value></member>
<member><name> syntax</name><value><string> tptp</string>
<member><name> timeout</name><value><int> 40</int></value></member>
</struct></param></params>
</methodCall>
```

The MathWeb-SB



Source: Zimmer, Autexier, Benzmüller

NEW: Translation Services

- Still need to know the interface syntax of services and define transformations
- $\Rightarrow$ Define external systems that provide service of translating between representations
- Example: The *Tramp* System [A. Meier, 1997]
  - Translation from HOL to Otter, Spass, etc.
  - Translation of resolution proofs back into HOL ND proofs

Features/Systems in MathWeb-SB

- Client-broker-server architecture
- Dynamic web of brokers (robustness, scalability)
- Integrates many different external systems
  - Reasoning systems *TPS, Vampire, Otter, Spass, Bliksem, Waldmeister, EPROVER, EQP, Fdpll, Protein, PVS, λ-Clam*
  - Computer Algebra Systems *MAPLE, MAGMA, GAP, COCOA*
  - Model generators *Mace, Kimba*
  - Theory exploration tool *HR*
  - Mathematical Knowledge Bases *MBASE*
  - Translation services *for OMDoc (OPENMATH) & TPTP, HOL→FOL, FOLRes → HOL-ND*
- Uniform service specification for served external systems

Applications using the MathWeb-SB

- Proof assistant ΩMEGA [ΩMEGA Group '02,]
- Proof Planner λ-Clam [Richardson et al. '98]
- Theory formation system HR [Colton '00]
- Interactive Textbook Activemath [Melis et al. '01]
- NLP System DORIS [Bos '01]

MAS communicate with a single, system-oriented and passive MathWeb

MathWeb-SB: The Next Generation

- *Motivation:*

MathWeb-SB: The Next Generation

- *Motivation:*
  - Still need to know *which* external system provides *what* kind of service

MathWeb-SB: The Next Generation

- *Motivation:*
  - Still need to know *which* external system provides *what* kind of service
  - Still need to know input (and output) syntax of external systems

MathWeb-SB: The Next Generation

- *Motivation:*
  - Still need to know *which* external system provides *what* kind of service
  - Still need to know input (and output) syntax of external systems
  - For complex operations, still need to combine yourself the calls to the different external systems

Source: Zimmer, Autexier, Benzmüller

MathWeb-SB: The Next Generation

- *Motivation:*
  - Still need to know *which* external system provides *what* kind of service
  - Still need to know input (and output) syntax of external systems
  - For complex operations, still need to combine yourself the calls to the different external systems
- *Goal:*

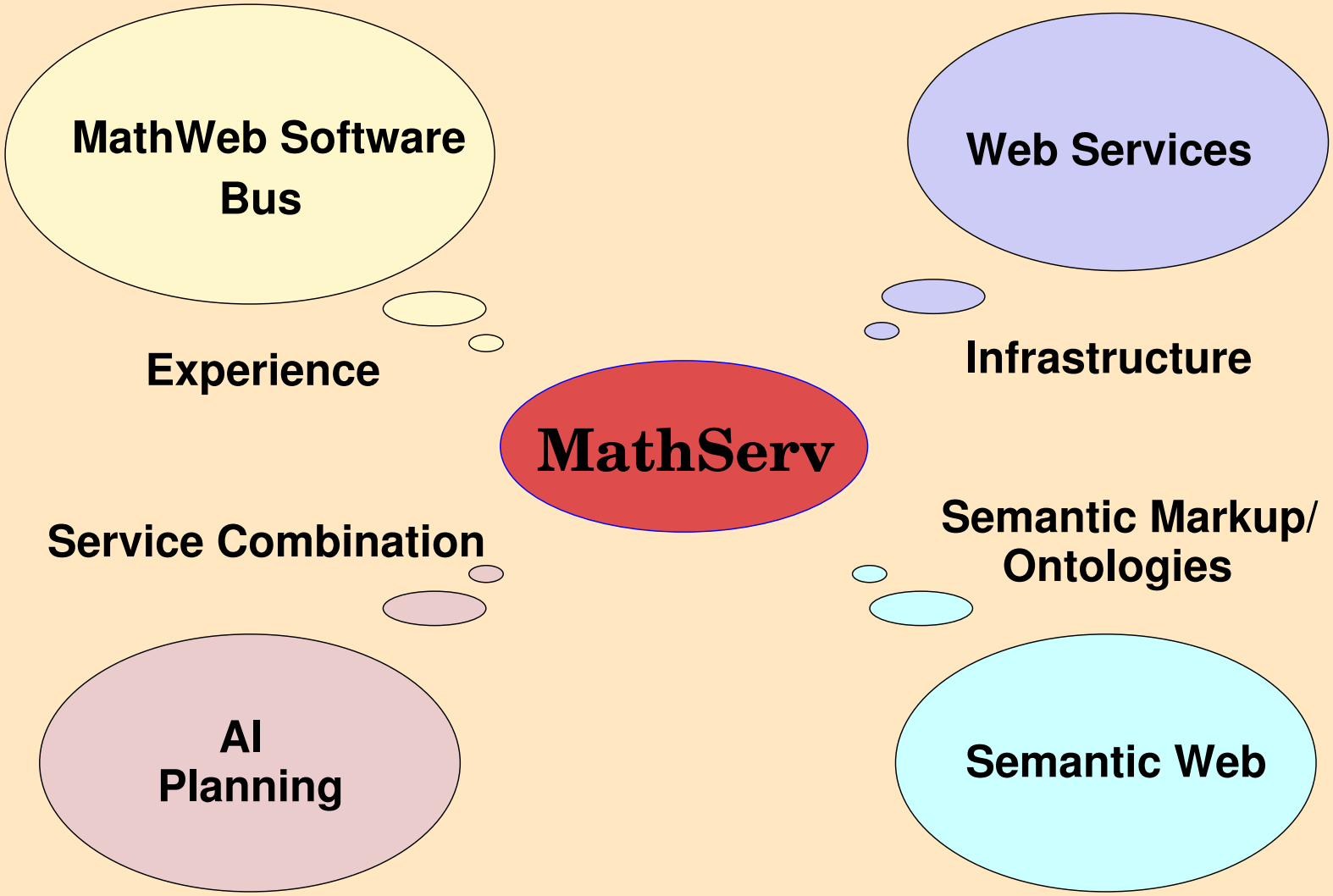
MathWeb-SB: The Next Generation

- *Motivation:*
  - Still need to know *which* external system provides *what* kind of service
  - Still need to know input (and output) syntax of external systems
  - For complex operations, still need to combine yourself the calls to the different external systems
- *Goal:*

Away from the
system-oriented and passive MathWeb
towards a
problem-oriented and active MathServ

Source: Zimmer, Autexier, Benzmüller

The MathServ Framework



Source: Zimmer, Autexier, Benzmüller

Advantages of MathServ

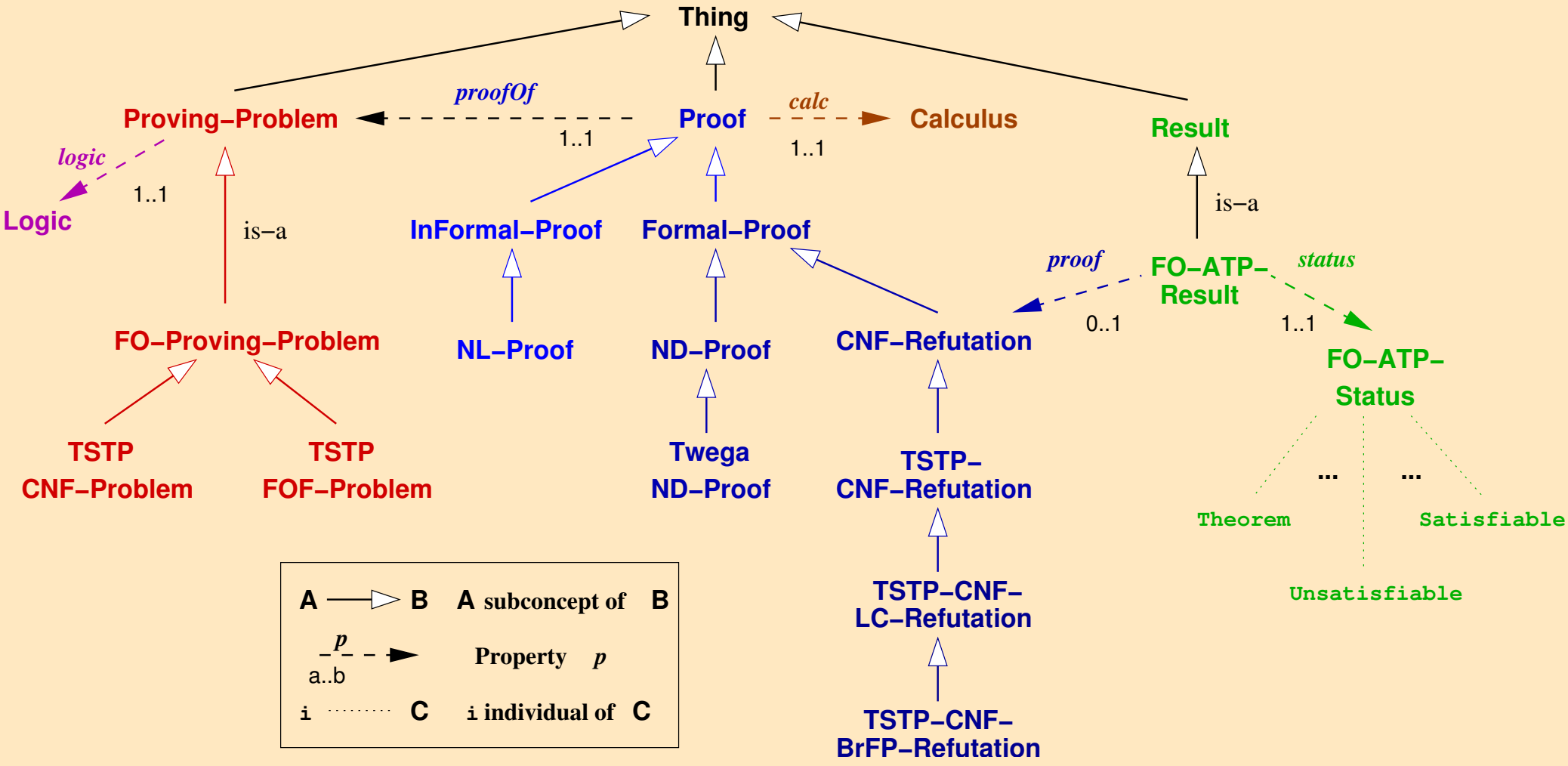
The MathServ framework can be used by humans or machines to...

- retrieve reasoning services (by human $\vee$ machine) given a semantic description of a problem.
- automatically combine services to tackle a problem.
- tackle subproblems in automatic or interactive theorem proving.

No need to know technical details of underlying system!

Source: Zimmer, Autexier, Benzmüller

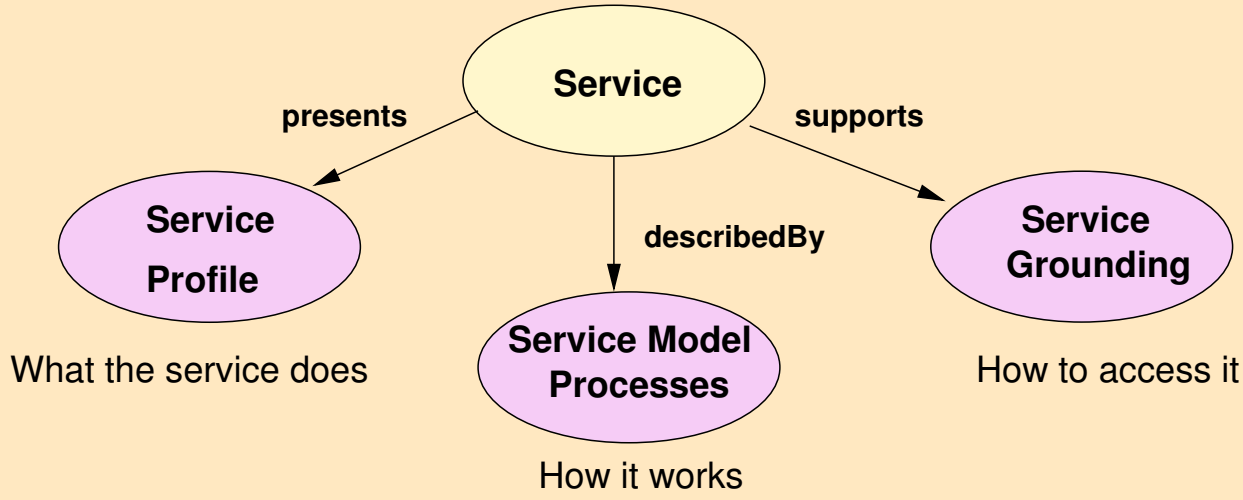
An OWL Ontology for Service Descriptions



OWL-S Ontology for Semantic Web Services

- Upper OWL ontology for service descriptions
- Developed by W3C working group.
- IOPE: Inputs/Outputs/Preconditions/Effects
- Atomic & Composite Processes

  (`Sequence, Split, If-Then-Else, Repeat-Until`)



Source: Zimmer, Autexier, Benzmüller

OWL-S Atomic Process for ATP Service

The central part of a service description

Service: EpATP	
input parameters:	*?problem*::TSTP-CNF-Problem (Concept)
output parameters:	*?result*::FO-ATP-Result
pre-conditions:	$\top$
post-conditions:	*resultFor(?result, ?problem)*

- We completely omit XML details.
- Conditions in Semantic Web Rule Language (SWRL) (RDF-triples, conjunction, implications).

Source: Zimmer, Autexier, Benzmüller

The Service of Tramp

Service: Tramp-NDforFOF	
input parameters:	*?fofProblem*::TSTP-FOF-Problem *?atpResult*::FO-ATP-Result
output parameters:	*?ndProof*::Twega-ND-Proof
pre-conditions:	*resultFor(?atpResult, ?cnfProblem)* $\wedge$ *deltaRelated(?cnfProblem, ?fofProblem)*
post-conditions:	*proofOf(?ndProof, ?fofProblem)*

"*deltaRelated*": $\exists$ *mapping* δ*: Literals* $\mapsto$ *FOF-Atoms*

Composition of Semantic Web Services

Problem: Sometimes one service is not sufficient.

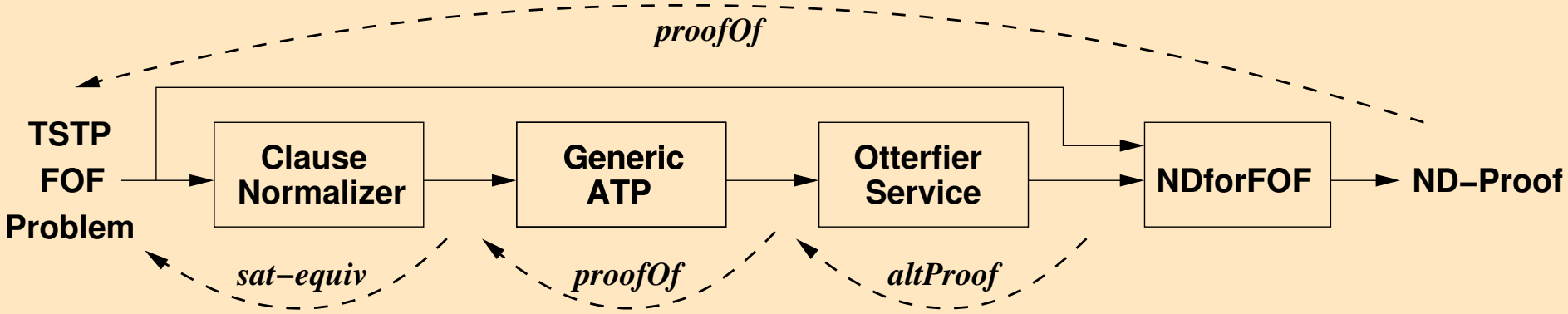
Brokering: Use AI planning to combine services automatically.

Planners used so far:

- Java POP
  (No-name, easy to use, coupling with Ontology reasoner).
- Prodigy [Veloso et. al. '96]
  (Control rules, simple type system, abstraction)

Difficulty: Planners assume static set of objects!

A Service Combination



Best-First Planning for ATPs

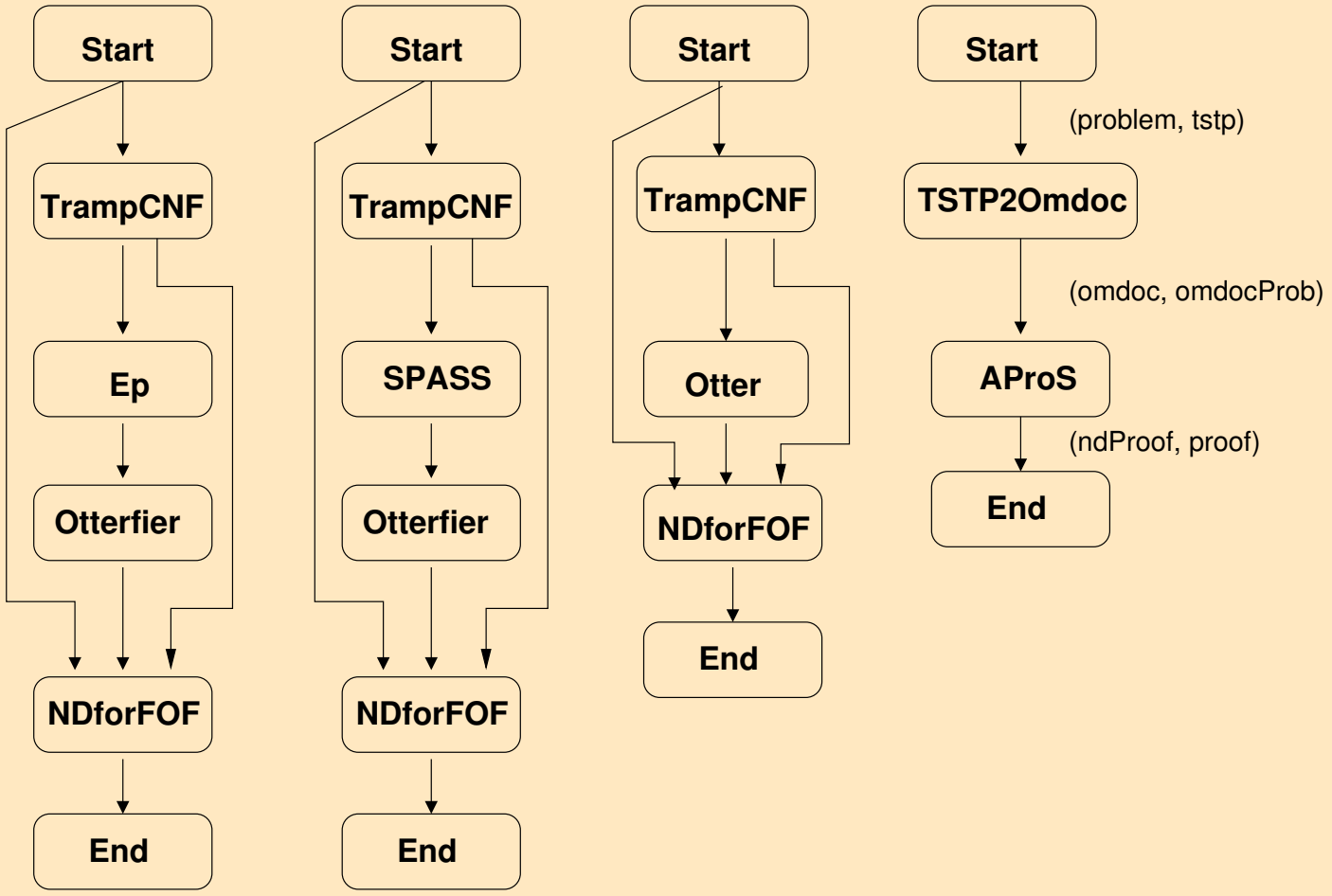
- ATP systems perform better on certain problem classes
  ⇒ Specialists Problem Classes (SPCs) [Sutcliffe, Suttner]
- Idea:
  - Classify proving problem before planning.
  - Use classification for best-first planning.
- Also applicable to TPS.

Prodigy control rule for best-first planning:

```
(Control-Rule prefer-ep-for-equality-problems
      (if (and (current-goal (resultFor <x> <y>))
               (true-in-state (problem-class <y> pure-equality-class))))
      (then prefer operator EpATP <anything>))
```

Source: Zimmer, Autexier, Benzmüller

Best-First Planning



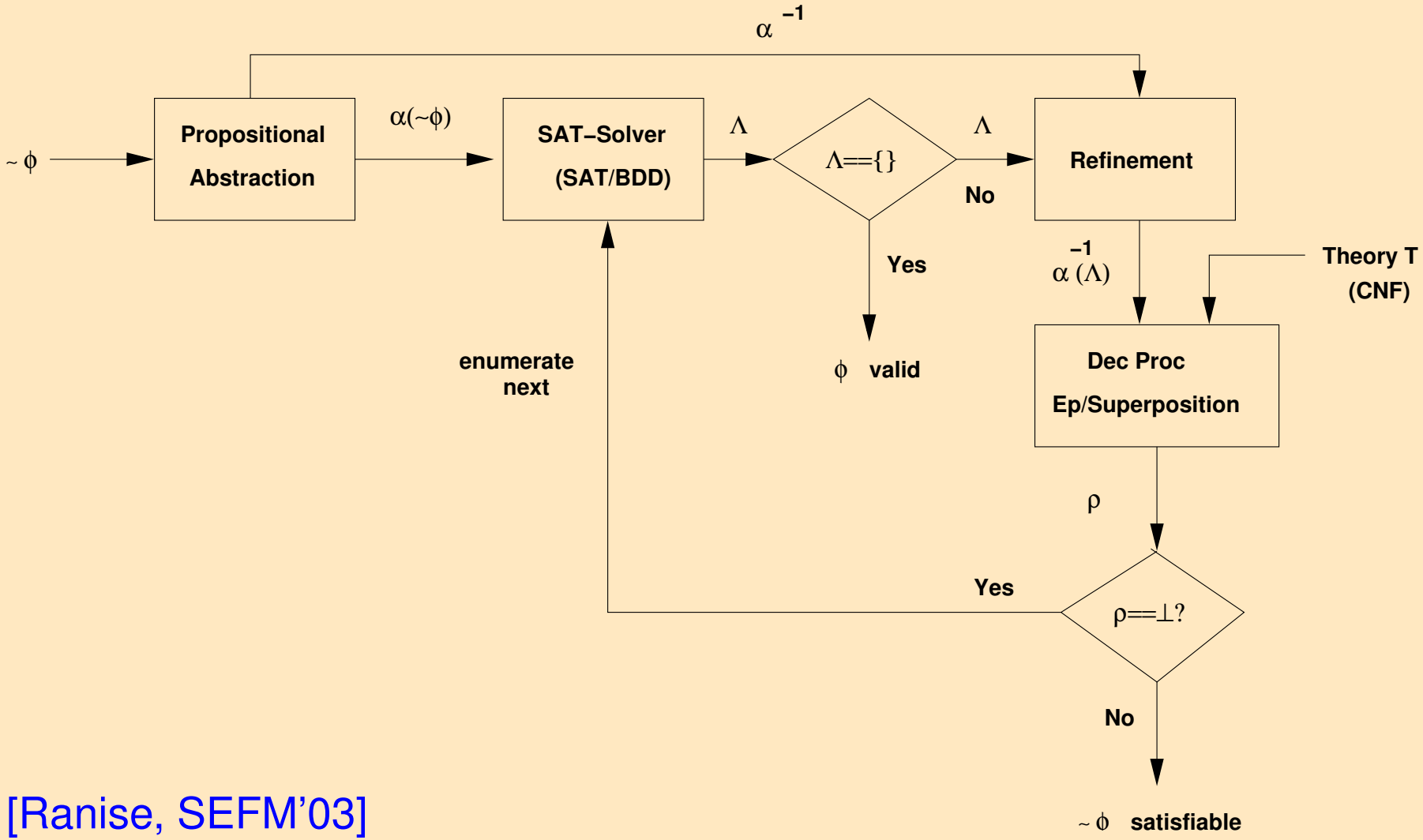
Complex Service Compositions

Context: Combinations of decision procedures

- **Question:** Formula φ valid in background theory T?
- Efficiency gain by combination with SAT techniques.
- Complex service combinations.
  (conditional branching, loops)
- AI planning not sufficient but expressible in OWL-S.
  (`If-Then-Else, Repeat-Until,...`)

Source: Zimmer, Autexier, Benzmüller

Decision Procedure with SAT Solving



[Ranise, SEFM'03]

Source: Zimmer, Autexier, Benzmüller

Summary about MATHSERV

- MathServ will provide possibility to submit problems in a uniform syntax and get an answer
  - given n in $\mathbb{N}at$ give me a proof that n is prime.
  - given an finite algebraic structure $(G, \circ)$, prove that $(G, \circ)$ is a group.
- Still ongoing work and many little details and problems need to be fixed. *PhD J. Zimmer*

Conclusions

Conclusions

- Independent from all logical problems of integrating external systems, a good infrastructure is important to support the integration process
  - From first experiments with integrating external systems
  - Via making them available for interactive theorem proving
  - Until the integration into standard search procedures
- Infrastructure along four lines:
  - MATHWEB: simplifies integration of new external systems by standardized interfaces (Even better if you only have to state your problem)
  - Hierarchical PDS: quick assessment of benefits of external system integration by delayed verification
  - OANTS: reactively provide external system services
  - MULTI: integration into standard/domain specific search procedures


OMEGA GROUP

Systems Integrated as Web Services

1) Automated theorem proving systems:

- EPROVER, SPASS, OTTER
- For classical first-order predicate logic with equality.

2) "Otterfier Tool for proof transformation [Sutcliffe'04]":

- CNF refutation $\longmapsto$ CNF refutation (BrFP) calculus
  BrFP = Binary resolution, Factoring, Paramodulation
- Calls OTTER to replace "alien" inference steps

More Systems Integrated

3) The Tramp system [A.Meier, 1997]

- FOF problem + CNF refutation (BrFP) $\longmapsto$ ND proof at assertion level.

assertion level step [Huang'94]: $\dfrac{F \subset G \qquad c \in F}{c \in G} \subset DEF$

$(\subset DEF) : \forall s_1.\forall s_2.(s_1 \subset s_2 \Leftrightarrow \forall x.(x \in s_1 \Rightarrow x \in s_2))$

4) Soon: The *P.rex* system [A. Fiedler, 2001]

- Formal proof $\longmapsto$ Natural Language (NL) proof.
- Proof quality depends on linguistic knowledge.

Prodigy Planning Operator for the Tramp Service


```
(OPERATOR  Tramp-NDforFOF
 (params  <fofProblem>  <atpResult>  <ndProof>)
 (preconds
(( <fofProblem> OWL-TSTPFOFPROBLEM) ( <atpResult> OWL-FOBRFPRESULT)
 ( <ndProof> OWL-NDPROOF)           (<cnfProblem> OWL-TSTPCNFPROBLEM))
 (and (bound <fofProblem>)
      (bound <atpResult>)
      (bound <cnfProblem>)
      (unbound <ndProof>)

      (resultFor <atpResult> <cnfProblem>)
      (deltaRelated <cnfProblem> <fofProblem>)))
 (effects ()
 ((del (unbound <ndProof>))
  (add (bound <ndProof>))
  (add (proofOf <ndProof> <fofProblem>))
  )))
```