Modular logic & Argumentation in Arg-tuProlog

Roberta Calegari* Giuseppe Contissa* *Giuseppe Pisano**
Galileo Sartor° Giovanni Sartor*

*Alma-AI - Alma Mater Studiorum – Università di Bologna
{roberta.calegari, giuseppe.contissa, g.pisano, giovanni.sartor}@unibo.it

°Università di Torino
galileo.sartor@unito.it

20th International Conference of the Italian Association for AI
December 1, 2021

The CompuLaw project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (Grant agreement No. 833647)

Context & motivation I

Dealing with multiple normative systems

Application and enforcement of the law makes it necessary to take into account the interplay of multiple normative systems

→ international contracts and other commercial and social interactions involving different countries

→ coexistence of national legal systems and of various transnational or international laws and conventions

Focus on the field of private international law (PIL):

- deals with the coexistence of multiple normative systems
- its logical analysis has highlighted how can be suitably modelled by modular argumentation [Dung and Sartor, 2011]

  ! not yet been captured and implemented in ready-to-use technology

Context & motivation II

Creating a ready-to-use technology

An extension for the Arg2P framework [Pisano et al., 2020b, Calegari et al., 2020] enabling the use of modular knowledge bases

(i) seamlessly integrate different legal knowledge bases

(ii) support for standard Prolog (*deductive reasoning*) evaluation

(iii) support for argumentation models and semantics (*defeasible reasoning*)

In this paper:

- presentation of the technology
- evaluation in two PIL case studies

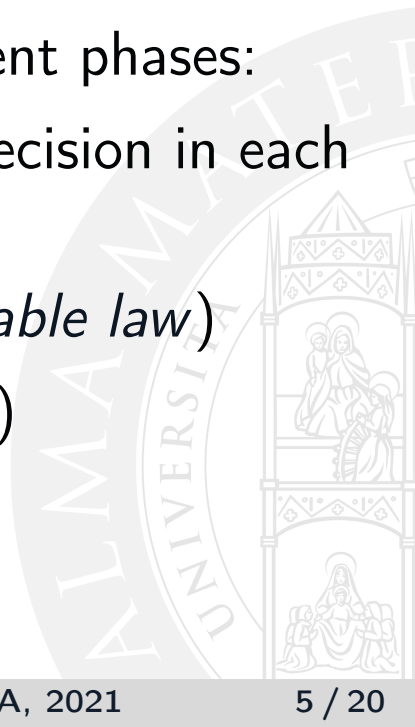
Private International Law I

Private international law (PIL)

Deals with the coexistence of multiple normative systems
$\rightarrow$ tries to establish priorities between them

The decision-making to solve *conflicts* is distributed in different phases:

- identifying which authority is responsible for making a decision in each given case (*jurisdiction*)
- identifying which set of norms should be applied (*applicable law*)
- applying the identified law to the case (*substantive laws*)



Private International Law II

PIL Model

Can be suitably modelled by modular argumentation [Dung and Sartor, 2011]
→ provide a formal model of the interaction among multiple legal systems

- PIL not concerned with specific inconsistencies between rules of different legal systems
- only one legal system will be selected and applied
  - ! regardless of how the others would regulate the same case
- modules where different legal systems are represented separately
  - → legal-system module is further split in separate modules
    1. determining jurisdiction
    2. establishing the law to be applied
    3. providing substantive legal outcomes

Arg2P

Arg2P

A lightweight Prolog-based argumentation framework that combines logic programming and legal reasoning [Pisano et al., 2020b].

It makes it possible to:

- represent legal sources
- argument on conditional norms featuring obligations, prohibitions, and permissions
- consider the burden-of-persuasion constraints that may apply

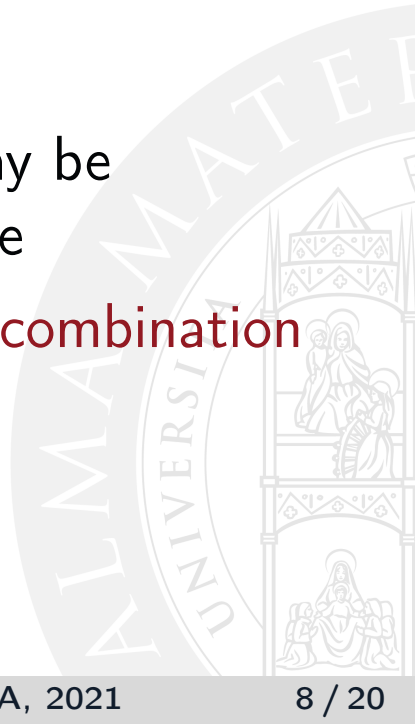
Enabling Modularity I

Modularity support

Arg2P[Pisano et al., 2020a] supports modularity according to the concepts introduced in the PIL logical model [Dung and Sartor, 2011]
→ knowledge organised in distinct modules that can "call" one another

The modularity support enables:

- organization of the knowledge: distinct modules that may be dependent on and/or pertinent to a hierarchical structure
- creation of compound-modules and dynamic evaluation/combination of knowledge from different sources
- modular argumentation



Enabling Modularity II

call_module/2

```
call_module([Module1, ..., ModuleN], Goal).
```

- modules are identified by distinct Prolog files (`.pl` files)
- `Modules` is an input parameter containing the list of the required modules
- `Query` is the query that must be evaluate

The location of the modules must be included in the *root* theory (main module) the predicate `modulesPath(++FileSystemPath)`

→ in case of nested calls, it is not necessary to re-specify the path in the submodules

Enabling Modularity III

call_module/2

```
call_module([Module1, ..., ModuleN], Goal).
```

In details, the predicate:

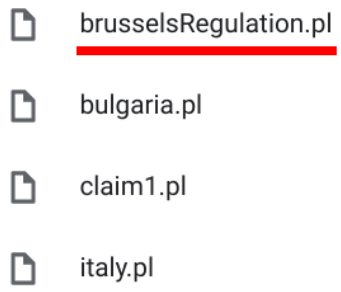
1. creates a new environment containing *only* the required modules data
2. executes the query in the newly created environment
3. feeds the result to the caller module
   - ! the original caller environment is not altered by the procedure

Reasoning on Jurisdiction

Brussels Regulation

Provides common EU rules on jurisdiction
→ some cases where it points to national legislation for the relevant answer

We can transalte the case exploiting the Arg2P modular extension...




```
hasJurisdiction(Article, Country, Court, ClaimId):-
        personRole(PID, ClaimId, defendant), memberState(ML),
        call_module([ML, ClaimId],
          hasJurisdiction(Article, Country, Court, ClaimId)).
hasJurisdiction(Article, Country, Court, ClaimId):-
        claimObject(ClaimId, rightsInRem), memberState(ML),
        call_module([ML,ClaimId],
          hasJurisdiction(Article,Country,Court,ClaimId)).
```

General jurisdiction rule I

The case

Let us consider the case of Marius, an Italian citizen with:

- his primary *residence* in the city of Rome
- his *place of business* in Sofia

```
personRole(marius, claim2, defendant).
personDomicile(marius, italy, rome).
personPlaceOfBusiness(marius,bulgaria,sofia).
memberState(bulgaria).
memberState(italy).
```

General jurisdiction rule II

We consider article 3.1 of the Italian Law No. 218 of 31 May 1995 (Reform of the Italian System of Private International Law)...

```
hasJurisdiction(art3_1, italy, Court, ClaimId) :-
        personRole(PersonId, ClaimId, defendant),
        personDomicile(PersonId, italy, Court).

hasJurisdiction(art3_1, italy, Court, ClaimId):-
        personRole(PersonId, ClaimId, defendant),
        personAgent(AgentId, PersonId),
        personDomicile(AgentId, italy, Court).
```

General jurisdiction rule III

...and article 4 of the Bulgarian Law DB, bp. 42 ot 17.05.2005 r. (Private International Law Code).

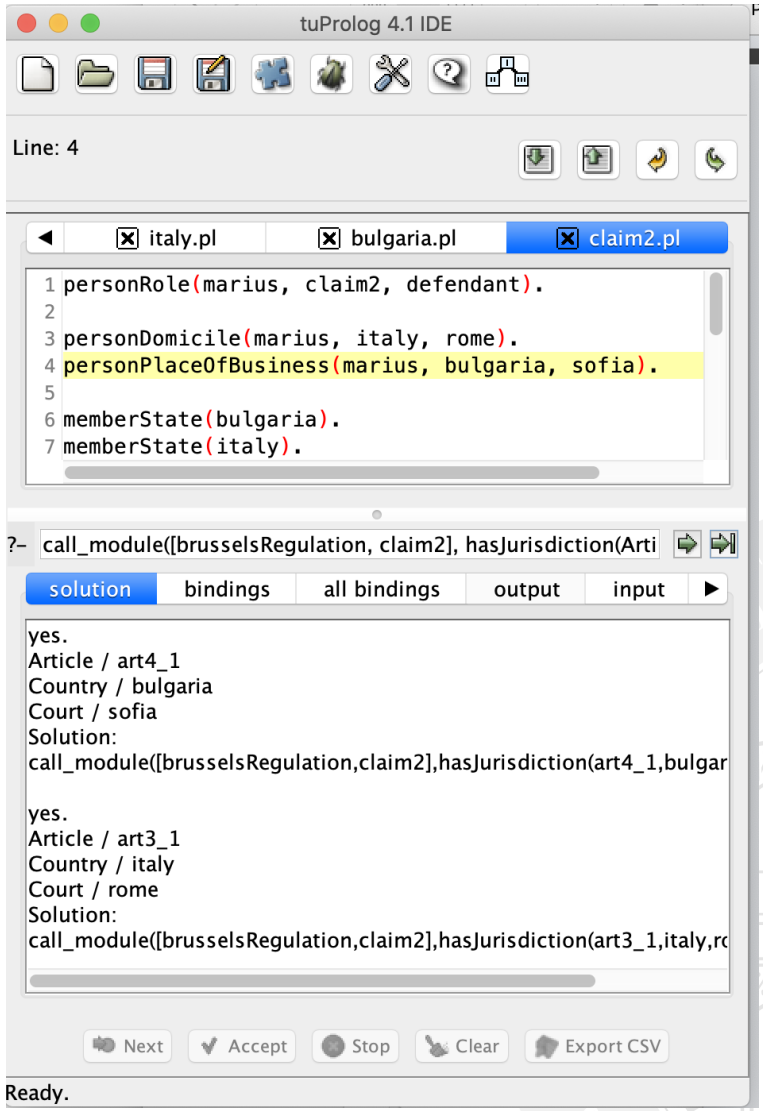
```
hasJurisdiction(art4_1, bulgaria, Court, ClaimId):-
        personRole(PersonId, ClaimId, defendant),
        personDomicile(PersonId, bulgaria, Court).

hasJurisdiction(art4_1, bulgaria, Court, ClaimId):-
        personRole(PersonId, ClaimId, defendant),
        personPlaceOfBusiness(PersonId, bulgaria, Court).
```

General jurisdiction rule IV

We require the evaluation of the case

```
call_module(
  [brussels, claim2],
  hasJurisdiction(Article,
  Country, Court, claim2))
```

Results show how the courts to which the case is assigned are both Rome and Sofia.



Jurisdiction related to rights in rem I

The Case

Let us consider the case of Marius, an Italian citizen:

- owner of two houses, one in Italy (Milan) and the other in Bulgaria (Sofia)
- a claim is brought against Marius with an action concerning his Bulgarian property

```
claimObject(claim3, rightsInRem).
immovableProperty(claim3, bulgaria, sofia).
memberState(bulgaria).
memberState(italy).
```

Jurisdiction related to rights in rem II

We consider article 5 (Actions concerning rights in rem in immovables situated abroad) of Italian PIL...

```
hasJurisdiction(art51, italy, Court, ClaimId):-
        claimObject(ClaimId, rightsInRem),
        immovableProperty(ClaimId, italy, Court).
```

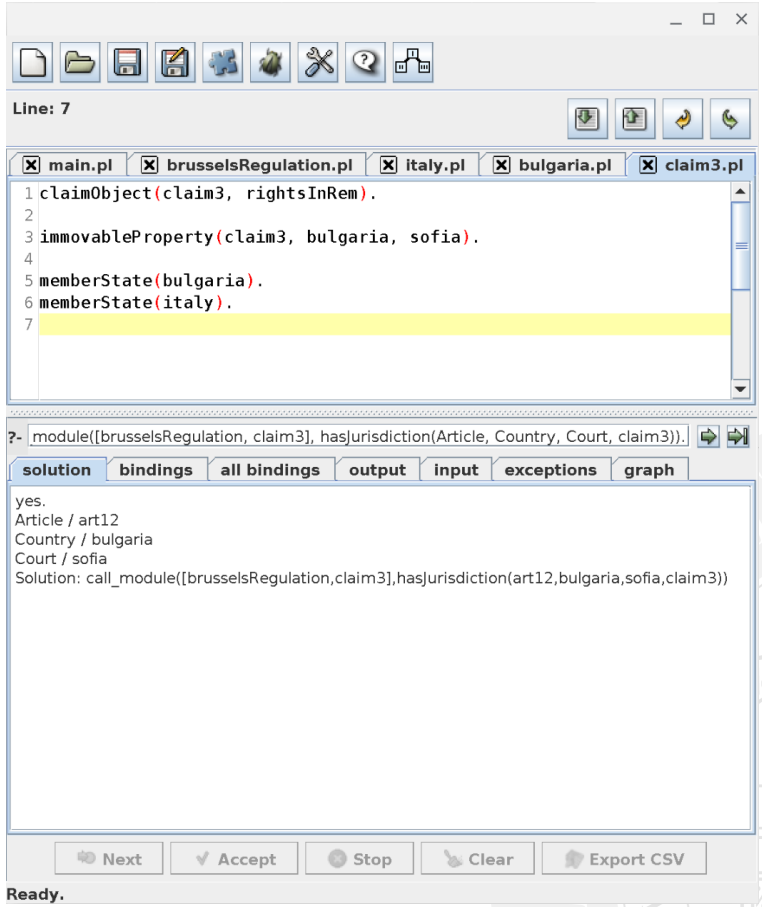
...and article 12 of the Bulgarian Law (Private International Law Code).

```
hasJurisdiction(art12, bulgaria, Court, ClaimId):-
        claimObject(ClaimId, rightsInRem),
        immovableProperty(ClaimId, bulgaria, Court).
```

Jurisdiction related to rights in rem III

We require the evaluation of the case

```
call_module(
  [brussels, claim1],
  hasJurisdiction(Article,
  Country, Court, claim3))
```



Results show how the court to which the case is assigned is Sofia $\rightarrow$ since the immovable property is in Bulgaria, the Brussels Regulation defers to Bulgarian legislation to determine the jurisdiction

Future work

The work needs further analysis w.r.t. the themes of:

- use of argumentation in case of more than one applicable jurisdictions
  - ! determining the most favorable jurisdiction
- reason also on applicable law
- further model legal domains presenting characteristics and issues similar to PIL
  - → internet law
  - → aviation law

Modular logic & Argumentation in Arg-tuProlog

Roberta Calegari* Giuseppe Contissa* *Giuseppe Pisano**
Galileo Sartor° Giovanni Sartor*

*Alma-AI - Alma Mater Studiorum – Università di Bologna
{roberta.calegari, giuseppe.contissa, g.pisano, giovanni.sartor}@unibo.it

°Università di Torino
galileo.sartor@unito.it

20th International Conference of the Italian Association for AI
December 1, 2021

The CompuLaw project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (Grant agreement No. 833647)

References I

[Calegari et al., 2020] Calegari, R., Contissa, G., Pisano, G., Sartor, G., and Sartor, G. (2020).
Arg-tuProlog: a modular logic argumentation tool for PIL.
In Villata, S., Harašta, J., and Křemen, P., editors, *Legal Knowledge and Information Systems. JURIX 2020: The Thirty-third Annual Conference*, volume 334 of *Frontiers in Artificial Intelligence and Applications*, pages 265–268.

[Dung and Sartor, 2011] Dung, P. M. and Sartor, G. (2011).
The modular logic of private international law.
Artificial Intelligence and Law, 19(2-3):233–261.

[Pisano et al., 2020a] Pisano, G., Calegari, R., Omicini, A., and Sartor, G. (2020a).
Arg-tuProlog: A tuProlog-based argumentation framework.
In Calimeri, F., Perri, S., and Zumpano, E., editors, *CILC 2020 – Italian Conference on Computational Logic. Proceedings of the 35th Italian Conference on Computational Logic*, volume 2710 of *CEUR Workshop Proceedings*, pages 51–66, Aachen, Germany. Sun SITE Central Europe, RWTH Aachen University, CEUR-WS.

[Pisano et al., 2020b] Pisano, G., Calegari, R., Omicini, A., and Sartor, G. (2020b).
Arg-tuprolog: a tuprolog-based argumentation framework.
In *Proceedings of the 35th Italian Conference on Computational Logic - CILC 2020*, volume 2710 of *CEUR Workshop Proceedings*, pages 51–66. CEUR-WS.org.