Legal Conflict Detection in Interacting Legal Systems

Tingting LI [a], Tina BALKE [b,a], Marina DE VOS [a], Julian PADGET [a] and Ken SATOH [c]

[a] *Dept. of Computer Science, University of Bath, UK*
[b] *Centre for Research in Social Simulation, University of Surrey, UK*
[c] *Principles of Informatics Res. Division, National Institute of Informatics, Japan*

Keywords. legal conflict detection, interacting legal systems, answer set programming

1. Introduction

Social rules such as laws, conventions and contracts prescribe and regulate human behaviour. It is also possible for us to break these rules at our discretion and face the consequences. In the normative multiagent systems community, normative frameworks are a way to identify a set of norms that describe the ideal behaviour of agents by specifying what is permitted, obliged and empowered within a given normative context. In [1] it was demonstrated how these frameworks could be applied to the legal domain. In [2], using the same technology, we showed that conflicts between legal specifications can be detected and resolved. Two types of conflicts are distinguished: (i) *weak* conflicts capture situations where an event/action is permitted by one legal system but prohibited by another, and (ii) *strong* conflicts between an obligation to perform an action in one and a prohibition on the action in another. An important assumption in [2] is that the specifications are independent of one another in that there is no connection between their respective transition rules. This is a strong assumption which is rare in reality, where such interactions will surely occur, intended or otherwise. This is why we here extend the work to the detection of conflicts between *interacting* legal specifications.

2. Methodology

We begin by modelling individual legal specification using an event-driven approach [1] in which a legal specification is defined over a trace of exogenous events. Starting from an initial state, each event brings about a state change, through the initiation and termination of fluents (i.e. permission, power, obligation and domain). From such a trace, we can compute a sequence of states that constitute the model of the legal specification. This process is automated through the encoding of the formal model in a computational framework built on Answer Set Programming, which is then combined with a similar translation of the different legal specifications.

Consequently, we can address the matter of the combination of a set of legal specifications – *interacting legal specification*, denoted C_I – to examine how individual specifications may interact with one another, such that either an event or a state change in one institution can trigger an event or state change, respectively, in another. We introduce two special rules in order to describe this *interaction*: (i) cross-specification generation rules provide a bridge for event generation between specifications; (ii) cross-specification consequence rules update the states of different specifications. By means of these rules, the occurrence of an external event may trigger all the constituent individual specifications to compute their next state – or if the event is not recognised by a specification, its next state is the same as its current state. Therefore, a sequence of *combined models* can be obtained from a given trace, where the combined model comprises the models of each individual specification.

We view each event trace as characterising a particular case that an interacting legal specification C_I may encounter. Conflicts are then detectable by comparing fluents from the individual models at a given time point. The whole detection procedure is implemented as an *AnsProlog* program and each generated answer set that contains an atom `conflict(X,Y,I,F)` represents the occurrence of a conflict between specification `X` and specification `Y` at time `I` with respect to fluent `F`. Furthermore, by testing all possible cases a C_I may encounter, we can identify whether the C_I is in general conflict-free.

We demonstrate this mechanism with a case study from a topical issue related to digital civil rights in Europe. The Irish Data Protection Authority (ODPC) has recently ruled that the Irish subsidiaries of Facebook are not breaking EU laws by sharing data with the NSA. The subjects involved are: Facebook Ireland, EU privacy law and US surveillance law. The data sharing activities of Facebook triggered a legal conflict between EU privacy law and US surveillance law. On the one hand, EU law states that exporting data to another country is legal only if adequate protection is provided. On the other hand, US law requires US companies to cooperate when data collection for surveillance purposes. As a subsidiary of Facebook, Facebook Ireland is placed in a dilemma, as it should abide by both US and EU law. The discussion of the ruling is out of the scope of this paper, but this case itself fits the characteristics of interacting legal specifications, in that a data sharing event without user's consent by Facebook generates the event of data exportation for EU privacy law and the event of data collection for US surveillance law respectively, leading to a state change for both EU and US legal positions. The resulting states of the EU and US in turn influence the state of Facebook with regard to the permission and obligation of sharing data. The initial state specifies that the NSA is a trusted party but there is no acceptable protection can be guaranteed by the NSA. Therefore, EU privacy law and US surveillance law disagree on the permission and obligation of Facebook's data-sharing action, resulting in a legal conflict. Such a conflict can be detected by our system automatically when given an event describing the scenario.

References

[1] M. De Vos, J. Padget, and K. Satoh. Legal modelling and reasoning using institutions. In T. Onoda, D. Bekki, and E. McCready, editors, *New Frontiers in Artificial Intelligence (JSAI-isAI 2010 Workshop)*, pages 129–140. Springer, 2011.

[2] T. Li, T. Balke, M. De Vos, J. Padget, and K. Satoh. A model-based approach to the automatic revision of secondary legislation. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Law*, ICAIL '13, pages 202–206, New York, NY, USA, 2013. ACM.

Legal Conflict Detection in Interacting Legal Systems

Tingting Li [1], Tina Balke [2,1], Marina De Vos [1], Julian Padget [1] and Ken Satoh [3]

{t.li, mdv, jap}@cs.bath.ac.uk, t.balke@surrey,ac,uk, ksatoh@nii.ac.jp

[1]Department of Computer Science, University of Bath, UK

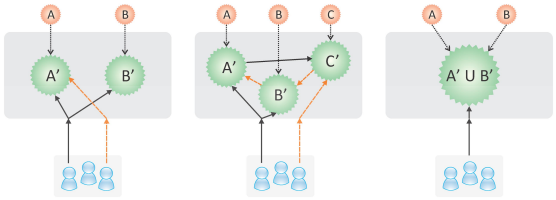
[2]Centre for Research in Social Similautaion, University of Surrey, UK

[3]Principles of Informatics Res. Division, National Institute of Informatics, Japan

Abstract

Acting under several jurisdictions at the same time is becoming the norm rather than the exception, certainly for companies but also (sometimes without knowing) for individuals. In these circumstances disparities among the different laws are inevitable. Here, we present a mathematical and a computational model of ***interacting legal specifications***, along with a mechanism to find conflicts between them. We illustrate the approach by a case study using European Privacy law.

Cooperating Legal Specifications



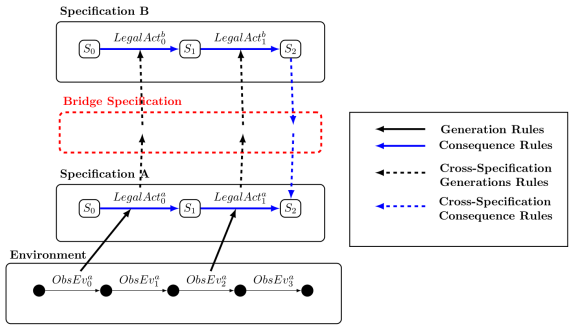
(a) (b) (c)

Three Forms of Combining Legal Specifications

(a) Comparative[2]: a set of peer legal specifications combined together to form a common governance scope, but states evolve independently.
(b) Interacting: an interlinked structure of a set of legal specifications, in which one might change the states of another. *(Focus of this work)*
(c) Merged: all the laws of each legal specification are merged to form a completely new specification.

Interacting Legal Specification

- **Bridge Specification:** separates connecting rules from the main specifications to make individuals oblivious to their interaction partners. Maintaining the flexibility and reusability of the structure.
- **Cross-specification Generation Relation:** an event in one specification triggers one or more events in one or more legal specifications:
  * $\mathbb{G}^x : \mathcal{X}^m \times \mathcal{E}^m \rightarrow \langle 2^{\mathcal{E}_1}, \ldots, 2^{\mathcal{E}_n}, 2^{\mathcal{E}^m} \rangle$
  * generation power: *gpow(source, event, destination)*
- **Cross-specification Consequence Relation:** a state change of one specification may result in a state change of another specification:
  * $\mathbb{C}^x : \mathcal{X}^m \times \mathcal{E}^m \rightarrow \langle 2^{\mathcal{F}_1}, \ldots, 2^{\mathcal{F}_n}, 2^{\mathcal{F}^m} \rangle \times \langle 2^{\mathcal{F}_1}, \ldots, 2^{\mathcal{F}_n}, 2^{\mathcal{F}^m} \rangle$.
  * initiation power: *ipow(source, fluent, destination)*
  * termination power: *tpow(source, fluent, destination)*



Legal Conflict Detection

1. Formalize and model individual legal specifications
2. Combine them to form an interacting specification
3a. *User-lead detection*
- give a particularly interesting event trace
- trigger states change of all participating specifications.
- obtain combined state model - a sequence of states.

3b. *Full-diagnose detection*
- compute all possible event traces
- for each trace, run **3a.**
- obtain combined state model in response to each trace

4. find conflicts by comparing fluent values between specifications at each state, by means of the detection program below:

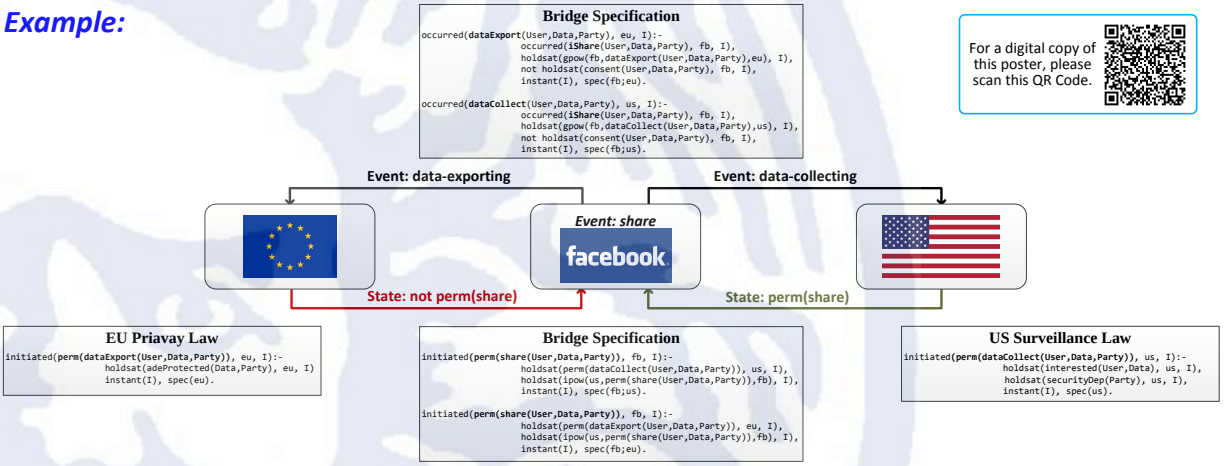
```
weakConflict(SpecX, SpecY, I, F):- holdsat(F,SpecX,I), not holdsat(F,SpecY, I),
                                   ifluent(F,SpecX), ifluent(F,SpecY),
                                   instant(I), spec(SpecX; SpecY).

strongConflict(SpecX, SpecY, I, E):- holdsat(obl(E,D,V), SpecX, I),
                                     not holdsat(perm(E), SpecY, I),
                                    ifluent(obl(E,D,V),SpecX),ifluent(perm(E),SpecY),
                                    instant(I), spec(SpecX; SpecY).
```

Reference:

[1] M. De Vos, J. Padget, and K. Satoh. Legal modelling and reasoning using institutions. In T. Onoda, D. Bekki, and E. McCready, editors, New Frontiers in Artificial Intelligence, pages 129–140. Springer, 2011.
[2] T. Li, T. Balke, M. De Vos, J. Padget, and K. Satoh. A model-based approach to the automatic revision of secondary legislation. In Proceedings of the Fourteenth International Conference on Artificial Intelligence and Law, ICAIL'13, pages 202–206. ACM, 2013.

Example:



For a digital copy of this poster, please scan this QR Code.

Tingting Li
t.li@cs.bath.ac.uk
University of Bath