

# Lex: A Language for Formalizing Legal Provisions

Practical Work by Jonas Degelo

# Motivation

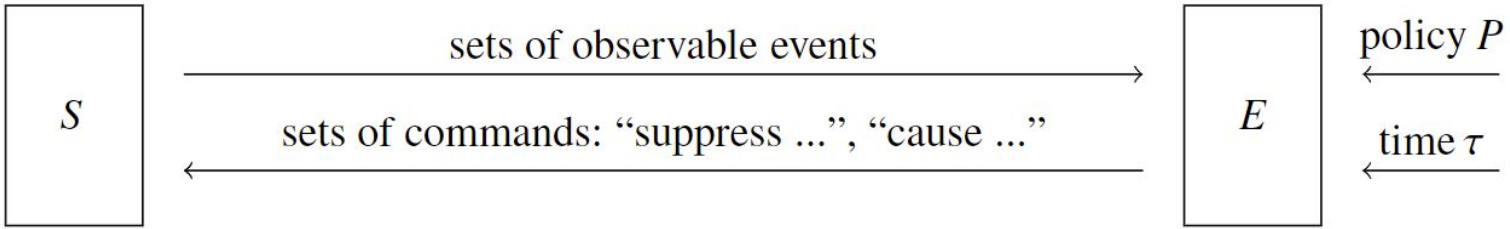
# Motivation for Legal Formalization

- Illustrate logical structure of legal texts
  - Eliminate (some) ambiguity
  - Allow for clearer reasoning about the logical meaning of the law
- Access to the law for a broader audience
  - Make certain aspects of a law more understandable
  - Automated tools can improve improve accessibility to the law
- **Automatization**
  - Runtime Verification
  - Check that the behavior of a computer system is compliant with the law



# Formalizing the Law for Runtime Verification

- Runtime monitoring and runtime enforcement
  - Monitoring: check that a system's behavior complies with a policy (online or offline)
  - Enforcement: check and possible "correct" a system's behavior (inherently online)
- State of the art enforcement tools require the law to be encoded as logical formulas
- Direct translation of legal texts to logical formulas is a difficult and error-prone task
- A "good" formalization provides improved readability




Hublet, François, Alexander Kvamme, and Srđan Krstić. "Towards an Enforceable GDPR Specification." arXiv preprint arXiv:2402.17350 (2024).

# Presentation Outline & Main Contributions

- Formalizing the Law: Some Theory
  - Vagueness
  - Logical Structure
- Temporal Logic
  - (Enforceable) Metric First Order Temporal Logic
- The Lex Language
  - Hierarchy
  - Rules
  - Type System

- Intermediate language between:
  - The law in natural language
  - Runtime verification policies (Logical Formulas)
- High readability by keeping hierarchical structure of the law
- Mechanisms to make logical structure explicit

# Formalizing the Law: Some Theory

# Vagueness of the Law

The Human Element

- Vagueness is inherent to the law
- Debate among scholars whether or not it could be eliminated entirely
- We aim to offer a solution to formalize the logical structure of legal texts
- Ambiguities can be decided by a human and the decisions can be provided as inputs to our system

# Logical Structure of the Law

- Exceptions specifying when some other section of the law does not apply
- Institutional vs. brute facts
  - Brute facts are “facts of nature”
    - a person's age, a car’s speed, etc.
  - Institutional facts are derived from brute facts
    - A person is allowed to vote in Switzerland if they are over 18 and a Swiss citizen
    - “Having the right to vote” is an institutional fact
- Structure is different from common programming languages

# Rules

Toy Example

- Obligation Rules
  - *(1) To enter the building authorization is required*
- Constitutive Rules
  - *(2) Employees are authorized to enter the building*
- Exception Rules
  - *(3) During time off (2) does not apply*

# Temporal Logic

# Metric First Order Temporal Logic (MFOTL)

First order predicate logic $\phi \wedge \psi,\ \phi \vee \psi,\ \neg\phi,\ P(x),$ etc.



| | | | |
|---|---|---|---|
| Always | $\Box_I\ \phi$ | Historically | $\blacksquare_I\ \phi$ |
| Until | $\phi\ \mathcal{U}_I\ \psi$ | Since | $\phi\ \mathcal{S}_I\ \psi$ |
| Next | $\bigcirc_I\ \phi$ | Previous | $\bullet_I\ \phi$ |
| Eventually | $\Diamond_I\ \phi$ | Once | $\blacklozenge_I\ \phi$ |

# Enforceable MFOTL

$$\frac{}{\Gamma \vdash \top : \mathbb{C}}\ \top^{\mathbb{C}} \qquad \frac{}{\Gamma \vdash \bot : \mathbb{S}}\ \bot^{\mathbb{S}} \qquad \frac{e \in \mathbb{C} \quad \Gamma(e) = \mathbb{C}}{\Gamma \vdash e(t_1, \ldots, t_k) : \mathbb{C}}\ \mathbb{E}^{\mathbb{C}} \qquad \frac{e \in \mathbb{S} \quad \Gamma(e) = \mathbb{S}}{\Gamma \vdash e(t_1, \ldots, t_k) : \mathbb{S}}\ \mathbb{E}^{\mathbb{S}}$$

$$\frac{\Gamma \vdash \varphi : \mathbb{S}}{\Gamma \vdash \neg\varphi : \mathbb{C}}\ \neg^{\mathbb{C}} \qquad \frac{\Gamma \vdash \varphi : \mathbb{C}}{\Gamma \vdash \neg\varphi : \mathbb{S}}\ \neg^{\mathbb{S}} \qquad \frac{\Gamma \vdash \varphi : \mathbb{C}}{\Gamma \vdash \exists x.\ \varphi : \mathbb{C}}\ \exists^{\mathbb{C}} \qquad \frac{\Gamma \vdash \varphi : \mathbb{S} \quad \vdash \varphi : \mathsf{PG}(x)^+}{\Gamma \vdash \exists x.\ \varphi : \mathbb{S}}\ \exists^{\mathbb{S}}$$

$$\frac{\Gamma \vdash \varphi : \mathbb{C} \quad \Gamma \vdash \psi : \mathbb{C}}{\Gamma \vdash \varphi \wedge \psi : \mathbb{C}}\ \wedge^{\mathbb{C}} \qquad \frac{\Gamma \vdash \varphi : \mathbb{S}}{\Gamma \vdash \varphi \wedge \psi : \mathbb{S}}\ \wedge^{\mathbb{SL}} \qquad \frac{\Gamma \vdash \psi : \mathbb{S}}{\Gamma \vdash \varphi \wedge \psi : \mathbb{S}}\ \wedge^{\mathbb{SR}}$$

$$\frac{0 \in I \quad \Gamma \vdash \psi : \mathbb{C}}{\Gamma \vdash \varphi\, \mathsf{S}_I\, \psi : \mathbb{C}}\ \mathsf{S}^{\mathbb{C}} \qquad \frac{0 \notin I \quad \Gamma \vdash \varphi : \mathbb{S}}{\Gamma \vdash \varphi\, \mathsf{S}_I\, \psi : \mathbb{S}}\ \mathsf{S}^{\mathbb{SL}} \qquad \frac{0 \in I \quad \Gamma \vdash \varphi : \mathbb{S} \quad \Gamma \vdash \psi : \mathbb{S}}{\Gamma \vdash \varphi\, \mathsf{S}_I\, \psi : \mathbb{S}}\ \mathsf{S}^{\mathbb{SLR}}$$

$$\frac{\Gamma \vdash \psi : \mathbb{S}}{\Gamma \vdash \varphi\, \mathsf{U}_I\, \psi : \mathbb{S}}\ \mathsf{U}^{\mathbb{S}} \qquad \frac{b \neq \infty \quad \Gamma \vdash \psi : \mathbb{C}}{\Gamma \vdash \varphi\, \mathsf{U}_{[0,b]}\, \psi : \mathbb{C}}\ \mathsf{U}^{\mathbb{CR}} \qquad \frac{b \neq \infty \quad \Gamma \vdash \varphi : \mathbb{C} \quad \Gamma \vdash \psi : \mathbb{C}}{\Gamma \vdash \varphi\, \mathsf{U}_{[a,b]}\, \psi : \mathbb{C}}\ \mathsf{U}^{\mathbb{CLR}}$$

Typing of formulae as causable/suppressable

$$\frac{\Gamma \vdash \varphi : \mathbb{C} \quad b > 0}{\Gamma \vdash \bigcirc_{[0,b)}\, \varphi : \mathbb{C}}\ \bigcirc^{\mathbb{C}} \qquad \frac{\Gamma \vdash \varphi : \mathbb{S}}{\Gamma \vdash \bigcirc_I\, \varphi : \mathbb{S}}\ \bigcirc^{\mathbb{S}}$$

- Policy language of WhyEnf
- Subset of MFOTL
- Predicates are marked with capabilities
  - Causable - “the system can create an event”
    - E.g. “Send notification to user”
  - Suppressable - “the system can prevent an action”
    - E.g. “A user makes a public comment”
  - Observable - “the system can take no action with regards to those events”

Hublet, François, et al. "Proactive Real-Time First-Order Enforcement." International Conference on Computer Aided Verification. Cham: Springer Nature Switzerland, 2024.

# Enforceable MFOTL - Example

$$\frac{}{\Gamma \vdash \top : \mathbb{C}}\ \top^{\mathbb{C}} \qquad \frac{}{\Gamma \vdash \bot : \mathbb{S}}\ \bot^{\mathbb{S}} \qquad \frac{e \in \mathbb{C} \quad \Gamma(e) = \mathbb{C}}{\Gamma \vdash e(t_1, \ldots, t_k) : \mathbb{C}}\ \mathbb{E}^{\mathbb{C}} \qquad \frac{e \in \mathbb{S} \quad \Gamma(e) = \mathbb{S}}{\Gamma \vdash e(t_1, \ldots, t_k) : \mathbb{S}}\ \mathbb{E}^{\mathbb{S}}$$

$$\frac{\Gamma \vdash \varphi : \mathbb{S}}{\Gamma \vdash \neg\varphi : \mathbb{C}}\ \neg^{\mathbb{C}} \qquad \frac{\Gamma \vdash \varphi : \mathbb{C}}{\Gamma \vdash \neg\varphi : \mathbb{S}}\ \neg^{\mathbb{S}} \qquad \frac{\Gamma \vdash \varphi : \mathbb{C}}{\Gamma \vdash \exists x.\ \varphi : \mathbb{C}}\ \exists^{\mathbb{C}} \qquad \frac{\Gamma \vdash \varphi : \mathbb{S} \quad \vdash \varphi : \mathsf{PG}(x)^+}{\Gamma \vdash \exists x.\ \varphi : \mathbb{S}}\ \exists^{\mathbb{S}}$$

$$\frac{\Gamma \vdash \varphi : \mathbb{C} \quad \Gamma \vdash \psi : \mathbb{C}}{\Gamma \vdash \varphi \wedge \psi : \mathbb{C}}\ \wedge^{\mathbb{C}} \qquad \frac{\Gamma \vdash \varphi : \mathbb{S}}{\Gamma \vdash \varphi \wedge \psi : \mathbb{S}}\ \wedge^{\mathbb{S}\mathrm{L}} \qquad \frac{\Gamma \vdash \psi : \mathbb{S}}{\Gamma \vdash \varphi \wedge \psi : \mathbb{S}}\ \wedge^{\mathbb{S}\mathrm{R}}$$

$$\frac{0 \in I \quad \Gamma \vdash \psi : \mathbb{C}}{\Gamma \vdash \varphi\, \mathsf{S}_I\, \psi : \mathbb{C}}\ \mathsf{S}^{\mathbb{C}} \qquad \frac{0 \notin I \quad \Gamma \vdash \varphi : \mathbb{S}}{\Gamma \vdash \varphi\, \mathsf{S}_I\, \psi : \mathbb{S}}\ \mathsf{S}^{\mathbb{S}\mathrm{L}} \qquad \frac{0 \in I \quad \Gamma \vdash \varphi : \mathbb{S} \quad \Gamma \vdash \psi : \mathbb{S}}{\Gamma \vdash \varphi\, \mathsf{S}_I\, \psi : \mathbb{S}}\ \mathsf{S}^{\mathbb{S}\mathrm{LR}}$$

$$\frac{\Gamma \vdash \psi : \mathbb{S}}{\Gamma \vdash \varphi\, \mathsf{U}_I\, \psi : \mathbb{S}}\ \mathsf{U}^{\mathbb{S}} \qquad \frac{b \neq \infty \quad \Gamma \vdash \psi : \mathbb{C}}{\Gamma \vdash \varphi\, \mathsf{U}_{[0,b]}\, \psi : \mathbb{C}}\ \mathsf{U}^{\mathrm{CR}} \qquad \frac{b \neq \infty \quad \Gamma \vdash \varphi : \mathbb{C} \quad \Gamma \vdash \psi : \mathbb{C}}{\Gamma \vdash \varphi\, \mathsf{U}_{[a,b]}\, \psi : \mathbb{C}}\ \mathsf{U}^{\mathrm{CLR}}$$

Typing of formulae as causable/suppressable

$$\frac{\Gamma \vdash \varphi : \mathbb{C} \quad b > 0}{\Gamma \vdash \bigcirc_{[0,b)}\, \varphi : \mathbb{C}}\ \bigcirc^{\mathbb{C}} \qquad \frac{\Gamma \vdash \varphi : \mathbb{S}}{\Gamma \vdash \bigcirc_I\, \varphi : \mathbb{S}}\ \bigcirc^{\mathbb{S}}$$

2 Options to “cause” (make true) the following formula

$$\phi \implies \psi$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\cdots}{\Gamma \vdash \phi : \mathbb{S}}}{\Gamma \vdash (\neg\phi) : \mathbb{C}}\ \neg^{\mathbb{C}}}{\Gamma \vdash (\neg\phi) \vee \psi : \mathbb{C}}\ \vee^{\mathbb{C}L}}{\Gamma \vdash \phi \implies \psi : \mathbb{C}}\ \text{rewriting}$$

Suppress the LHS

$$\dfrac{\dfrac{\dfrac{\cdots}{\Gamma \vdash \psi : \mathbb{C}}}{\Gamma \vdash (\neg\phi) \vee \psi : \mathbb{C}}\ \vee^{\mathbb{C}R}}{\Gamma \vdash \phi \implies \psi : \mathbb{C}}\ \text{rewriting}$$

Cause the RHS

Hublet, François, et al. "Proactive Real-Time First-Order Enforcement." International Conference on Computer Aided Verification. Cham: Springer Nature Switzerland, 2024.

# The Lex Language

# Goals for Lex

- High readability (subject of planned user study)
- Close correspondence to legal texts in structure and form
- Relatively easy auditing of whether or not a formalization in Lex accurately represents the law it formalizes
- Compiles to:
  - Documentation-like HTML representation of the formalization
  - Logical formula that is enforceable by recent runtime enforcers
    - The Lex compiler specifically checks the enforceability requirements

# Section Labels

Lex Code: US Tax Code

```
law "U.S. Code"

title "26"

title[1] "A"

chapter "1"

chapter[1] "B"

section "III"

article "121"
```




Lex documentation: US Tax Code

| | |
|---|---|
| law U.S. Code | |
| title 26 | law U.S. Code |
| title[1] A | law U.S. Code / title 26 |
| chapter 1 | law U.S. Code / title 26 / title[1] A |
| chapter[1] B | law U.S. Code / title 26 / title[1] A / chapter 1 |
| section III | law U.S. Code / title 26 / title[1] A / chapter 1 / chapter[1] B |
| article 121 | law U.S. Code / title 26 / title[1] A / chapter 1 / chapter[1] B / section III |

# Event and Type Definitions

Lex Code: GDPR (EU)

```
type entity   is string
type person   is string
type task     is int
type action   is int
type data     is int
type consent  is int

event Nominates
  """ Controller {c} nominates processor {p} to perform data processing task {t} """
  c: entity
  p: entity
  t: task
```

# GDPR Example - Data Processing

# Obligation Rules

Lex Documentation: GDPR (EU)

article 5

law GDPR / chapter II

XML: GDPR Grundsätze für die Verarbeitung personenbezogener Daten

paragraph 1

law GDPR / chapter II / article 5

point a

law GDPR / chapter II / article 5 / paragraph 1

```
rule
  whenever
    PersonalDataProcessing(ep, x, z)
  oblige
    lawfulness(ep)
    fairness(ep)
    transparency(ep)
enforceable
suppressing PersonalDataProcessing
```

Reading

Whenever all of the following happen:

- x is processing personal data z as part of processing operation ep

Then the following **shall be obligatory**:

- processing ep is lawful
- processing ep is fair
- processing ep is transparent

XML: GDPR *Personenbezogene Daten müssen [...] a) auf rechtmäßige Weise, nach Treu und Glauben und in einer für die betroffene Person nachvollziehbaren Weise verarbeitet werden ( Rechtmäßigkeit, Verarbeitung nach Treu und Glauben, Transparenz );*

# Constitutive Rules

Lex Documentation: GDPR (EU)

> 1. Processing shall be lawful only if and to the extent that at least one of the following applies:
>
> (a) the data subject has given consent to the processing of his or her personal data for one or more specific purposes;

article 6 — law GDPR / chapter II

XML: GDPR Rechtmäßigkeit der Verarbeitung

paragraph 1 — law GDPR / chapter II / article 6

point 1 — law GDPR / chapter II / article 6 / paragraph 1

subpoint a — law GDPR / chapter II / article 6 / paragraph 1 / point 1

```
rule
  whenever
    PersonalDataProcessing(ep, x, z)
    HasPurpose(ep, prp)
    ONCE (GiveConsent(w, c) AND Authorizes(c, prp))
    ONCE Nominates(y, x)
    PersonalData(z, w)
  constitute
    lawfulness(ep)
```

Reading

Whenever all of the following happen:

- x is processing personal data z as part of processing operation ep
- processing operation ep has purpose prp
- at some point in the past, the following happened:
  - all of the following are the case:
    - data subject w gives consent c
    - consent c authorizes usage of personal data for purpose prp
- at some point in the past, the following happened:
  - controller y nominates processor x to process personal data on y's behalf
- z is personal data for data subject w

Then the following **shall be constituted**:

- processing ep is lawful

# Exception Rules

Lex Documentation: GDPR (EU)

article 8 — law GDPR / chapter II

XML: GDPR Bedingungen für die Einwilligung eines Kindes in Bezug auf Dienste der Informationsgesellschaft

paragraph 1 — law GDPR / chapter II / article 8

```
rule
  whenever
   isMinor(w)
  except
   article 6 paragraph 1 point 1 subpoint a
```

Reading

Whenever all of the following happen:

- data subject w is a minor

Then article 6 paragraph 1 point 1 subpoint a **shall not apply**.

# Lifting the EMFOTL Type System to Lex

Lex Documentation: GDPR (EU)

```
rule
  whenever
    PersonalDataProcessing(ep, x, z)
  oblige
    lawfulness(ep)
    fairness(ep)
    transparency(ep)
enforceable
```

3 ways to make an obligation compliant (i.e. true)

1. "Suppress its conditions"
2. "Cause its effects"
3. "Cause potential exceptions to the obligation rule"

$$\frac{r : \phi \Rightarrow \psi \in R \quad \Gamma \vdash \phi : \mathbb{S}}{\Gamma \vdash \mathrm{comply}(r) : \mathbb{C}}\ \textsc{RegL} \qquad \frac{r : \phi \Rightarrow \psi \in R \quad \Gamma \vdash \psi : \mathbb{C}}{\Gamma \vdash \mathrm{comply}(r) : \mathbb{C}}\ \textsc{RegR} \qquad \frac{\Gamma \vdash \mathrm{exc}(r) : \mathbb{C}}{\Gamma \vdash \mathrm{comply}(r) : \mathbb{C}}\ \textsc{RegE}$$

# Lifting the EMFOTL Type System to Lex - Temporal Patterns

| Description $patt_i$ | $form_i(\phi)$ | Type |
|---|---|---|
| – | $\phi$ | Present |
| `eventually` $[itv]$ | $\texttt{EVENTUALLY}_I\ \phi$ | Future |
| `eventually delaying if` $\psi$ $[itv]$ | $\psi\ \texttt{UNTIL}_I\ \phi$ | Future |
| `always in the future` $[itv]$ | $\texttt{ALWAYS}_I\ \psi$ | Future |
| `once` $[itv]$ | $\texttt{ONCE}_I\ \psi$ | Past |
| `always in the past` $[itv]$ | $\texttt{HISTORICALLY}_I\ \psi$ | Past |
| `always since` $\psi$ $[itv]$ | $\phi\ \texttt{SINCE}_I\ \psi$ | Past |

$$\frac{\Gamma \vdash \phi_0 : \mathbb{C} \quad \dots \quad \Gamma \vdash \phi_n : \mathbb{C}}{\Gamma \vdash \bigwedge_{i=0}^{n} \phi_i : \mathbb{C}}\ \text{Present}^{\mathbb{C}} \qquad \frac{\Gamma \vdash \phi_j : \mathbb{S}}{\Gamma \vdash \bigwedge_{i=0}^{n} \phi_i : \mathbb{S}}\ \text{Present}^{\mathbb{S}_j}$$

$$\frac{\Gamma \vdash \phi_0 : \mathbb{C} \quad \dots \quad \Gamma \vdash \phi_n : \mathbb{C}}{\Gamma \vdash \texttt{EVENTUALLY}_{[a,b]} \bigwedge_{i=0}^{n} \phi_i : \mathbb{C}}\ \text{Eventually}^{\mathbb{C}} \qquad \frac{\Gamma \vdash \phi_j : \mathbb{S}}{\Gamma \vdash \texttt{EVENTUALLY}_{[a,b]} \bigwedge_{i=0}^{n} \phi_i : \mathbb{S}}\ \text{Eventually}^{\mathbb{S}_j}$$

$$\frac{\Gamma \vdash \phi_0 : \mathbb{C} \quad \dots \quad \Gamma \vdash \phi_n : \mathbb{C}}{\Gamma \vdash \texttt{ALWAYS}_{[a,b]} \bigwedge_{i=0}^{n} \phi_i : \mathbb{C}}\ \text{Always}^{\mathbb{C}} \qquad \frac{\Gamma \vdash \phi_j : \mathbb{S}}{\Gamma \vdash \texttt{ALWAYS}_{[a,b]} \bigwedge_{i=0}^{n} \phi_i : \mathbb{S}}\ \text{Always}^{\mathbb{S}_j}$$

$$\frac{b \neq \infty \quad \Gamma \vdash \phi_0 : \mathbb{C} \quad \dots \quad \Gamma \vdash \phi_n : \mathbb{C}}{\Gamma \vdash \psi\ \texttt{Until}_{[0,b]} \bigwedge_{i=0}^{n} \phi_i : \mathbb{C}}\ \text{Until}^{\mathbb{C}R} \qquad \frac{\Gamma \vdash \phi_j : \mathbb{S}}{\Gamma \vdash \psi\ \texttt{Until}_{[a,b]} \bigwedge_{i=0}^{n} \phi_i : \mathbb{S}}\ \text{Until}^{\mathbb{S}_j}$$

$$\frac{b \neq \infty \quad \Gamma \vdash \psi : \mathbb{C} \quad \Gamma \vdash \phi_0 : \mathbb{C} \quad \dots \quad \Gamma \vdash \phi_n : \mathbb{C}}{\Gamma \vdash \psi\ \texttt{Until}_{[a,b]} \bigwedge_{i=0}^{n} \phi_i : \mathbb{C}}\ \text{Until}^{\mathbb{C}LR}$$

$$\frac{\Gamma \vdash \phi_j : \mathbb{S}}{\Gamma \vdash \texttt{HISTORICALLY}_{[a,b]} \bigwedge_{i=0}^{n} \phi_i : \mathbb{S}}\ \text{Historically}^{\mathbb{S}_j}$$

$$\frac{0 \in [a,b] \quad \Gamma \vdash \phi_0 : \mathbb{C} \quad \dots \quad \Gamma \vdash \phi_n : \mathbb{C}}{\Gamma \vdash \texttt{Once}_{[a,b]} \bigwedge_{i=0}^{n} \phi_i : \mathbb{C}}\ \text{Once}^{\mathbb{C}} \qquad \frac{0 \in [a,b] \quad \Gamma \vdash \psi : \mathbb{C}}{\Gamma \vdash \left(\bigwedge_{i=0}^{n} \phi_i\right)\ \texttt{Since}_{[a,b]}\ \psi : \mathbb{C}}\ \text{Since}^{\mathbb{C}}$$

$$\frac{0 \notin [a,b] \quad \Gamma \vdash \phi_j : \mathbb{S}}{\Gamma \vdash \left(\bigwedge_{i=0}^{n} \phi_i\right)\ \texttt{Since}_{[a,b]}\ \psi : \mathbb{S}}\ \text{Since}^{\mathbb{S}L_j} \qquad \frac{0 \in [a,b] \quad \Gamma \vdash \phi_j : \mathbb{S} \quad \Gamma \vdash \psi : \mathbb{S}}{\Gamma \vdash \left(\bigwedge_{i=0}^{n} \phi_i\right)\ \texttt{Since}_{[a,b]}\ \psi : \mathbb{S}}\ \text{Since}^{\mathbb{S}LR_j}$$

# Compiled Result

MFOTL Formula: GDPR (EU)

```
let Exception2(w) = isMinor(w)
let lawfulness(_p1) = ∃c. ∃ep. ∃prp. ∃w. ∃x. ∃y. ∃z.
                        ((((((PersonalDataProcessing(ep, x, z)
                                ∧ HasPurpose(ep, prp))
                                ∧ ♦ (GiveConsent(w, c)
                                ∧ Authorizes(c, prp)))
                                ∧ ♦ Nominates(y, x))
                                ∧ PersonalData(z, w))
                            ∧ ¬Exception2(w))
                        ; _p1 <- ep)

□ ∀ep. ∀x. ∀z. (PersonalDataProcessing(ep, x, z) ∧:L ⊤)
    →:L ((lawfulness(ep) ∧ fairness(ep)) ∧ transparency(ep))
```

MFOTL Signature: GDPR (EU)

```
Authorizes(c__: int, prp__: string)
GiveConsent(w__: int, c__: int)
HasPurpose(ep__: int, prp__: string)
Nominates(y__: int, x__: int)
PersonalData(z__: int, w__: int)
PersonalDataProcessing(ep__: int, x__: int, z__: int)-
fairness(ep__: int)
isMinor(w__: int)
lawfulness(ep__: int)
transparency(ep__: int)
ts(~return_value__: time)
fun day(t__: time) -> time
Exception2(w__: int)
```

# Lex Compilation Pipeline




| | | | | |
|---|---|---|---|---|
| Parse lex code | Type terms<br>Compute labels<br>Collect exceptions | Topologically sort rules<br>Check enforceable | Prepare signature and MFOTL formula | Compile and linearize as strings |

# Future Work

- Refinement
  - Refinement will allow users to take an existing formalization and specify how the high-level events in the formalization map to specific events in a given system
  - This will allow for formalizations to be reused across different use cases
- Tooling
  - VS Code Extension
  - Language Server
- Formalization of GDPR
- User Study on the usability of Lex

# References


[1] Hublet, François, et al. "Proactive Real-Time First-Order Enforcement." International Conference on Computer Aided Verification. Cham: Springer Nature Switzerland, 2024.

[2] https://www.law.cornell.edu/uscode/text/26/121

[3] https://gdpr-info.eu/art-6-gdpr/

[4] https://www.vecteezy.com/vector-art/534940-gdpr-general-data-protection-regulation-icon-line-style

[5] Hublet, François, Alexander Kvamme, and Srđan Krstić. "Towards an Enforceable GDPR Specification." arXiv preprint arXiv:2402.17350 (2024).

# BGG Example

# Event and Type Definitions

```
type recht is string
type gericht is string
type gebiet is string
type gegenstand is string
type beschwerde is int
  """ eine Beschwerde """
type akt is int
  """ ein Akt """
type person is string
type staat is string

predicate anfechtungsObjekt
  """ Beschwerde {b} fechtet Akt {a} an """
  b: beschwerde
  a: akt
```

# Obligation Rules

article 82

law BGG / chapter 3 / section 3

XML: BGG Grundsatz

```
rule principle
  whenever
   beurteilt(g, b)
  oblige
   zulaessig(g, b)
 transparently enforceable
```

Reading

Whenever all of the following happen:

- Gericht g beurteilt Beschwerde b

Then the following **shall be obligatory**:

- Beschwerde b ist bei Gericht g zulässig

# Exception Rules

point b

law BGG / chapter 3 / section 3 / article 83 / paragraph 1

```
rule
  whenever
   betrifft(a, "ordentliche Einbürgerung")
  except
   article 82 paragraph 1
```

Reading

Whenever all of the following happen:

- Akt a betrifft ordentliche Einbürgerung

Then article 82 paragraph 1 **shall not apply**.

XML: BGG *Die Beschwerde ist unzulässig gegen: [...] b. Entscheide über die ordentliche Einbürgerung;*

# Constitutive Rules

point b

law BGG / chapter 3 / section 3 / article 82 / paragraph 1

```
rule
  whenever
   anfechtungsObjekt(b, a)
   kantonalerErlass(a)
  constitute
   zulaessig("Bundesgericht", b)
```

Reading

Whenever all of the following happen:

- Beschwerde b fechtet Akt a an
- Akt a ist ein kantonaler Erlass

Then the following **shall be constituted**:

- Beschwerde b ist bei Gericht Bundesgericht zulässig

XML: BGG *Das Bundesgericht beurteilt Beschwerden: [...] b. gegen kantonale Erlasse;*

# Compiled Output

```
let Exception14(a) = betrifft(a, "Verweigerung der Ermächtigung zur Strafverfolgung von Behördenmitgliedern oder von Bundespersonal")
let Exception13(a, bew) = ((((gebiet(a, "Asyl") ∧ ♦ trifft("kantonale Vorinstanz", a)) ∧ betrifft(a, bew)) ∧ bewilligung(bew))
        ∧ ¬raeumtAnspruchEin("Bundesrecht", bew)) ∧ ¬raeumtAnspruchEin("Völkerrecht", bew)
let Exception12(a) = (gebiet(a, "Asyl") ∧ ♦ trifft("Bundesverwaltungsgericht", a))
        ∧ ¬∃p. ∃s. (betrifftPerson(a, p) ∧ suchtSchutz(p, s) ∧ ♦ stelltAuslieferungsersuchen(s, p))
let Exception11(a) = gebiet(a, "Ausländerrecht") ∧ (betrifft(a, "Verlängerung der Grenzgängerbewilligung")
        ∨ betrifft(a, "Kantonswechsel") ∨ betrifft(a, "Stellenwechsel von Personen mit Grenzgängerbewilligung")
        ∨ betrifft(a, "Erteilung von Reisepapieren an schriftenlose AusländerInnen"))
let Exception10(a) = gebiet(a, "Ausländerrecht") ∧ betrifft(a, "Abweichungen von den Zulassungsvoraussetzungen")
let Exception9(a) = gebiet(a, "Ausländerrecht") ∧ (betrifft(a, "Ausweisung (nach Art. 121 Abs. 2 BV)") ∨ betrifft(a, "Wegweisung"))
let Exception8(a) = gebiet(a, "Ausländerrecht") ∧ betrifft(a, "vorläufige Aufnahme")
let Exception7(a, bew) = (((gebiet(a, "Ausländerrecht") ∧ betrifft(a, bew)) ∧ bewilligung(bew))
        ∧ ¬raeumtAnspruchEin("Bundesrecht", bew)) ∧ ¬raeumtAnspruchEin("Völkerrecht", bew)
let Exception6(a) = gebiet(a, "Ausländerrecht") ∧ betrifft(a, "Einreise")
let Exception5(a) = betrifft(a, "ordentliche Einbürgerung")
let Exception4(a) = gebiet(a, "auswärtige Angelegenheiten") ∧ ¬beurteilungsAnspruch("Völkerrecht", a)
let zulaessig(_p1, _p2) = ∃a. ∃b. ∃bew. ((((((((((((((anfechtungsObjekt(b, a) ∧ (betrifft(a, "politische Stimmberechtigung")
        ∨ betrifft(a, "Volkswahlen") ∨ betrifft(a, "Volksabstimmungen")))
        ∧ ¬Exception4(a)) ∧ ¬Exception5(a)) ∧ ¬Exception6(a)) ∧ ¬Exception7(a, bew)) ∧ ¬Exception8(a))
        ∧ ¬Exception9(a)) ∧ ¬Exception10(a)) ∧ ¬Exception11(a)) ∧ ¬Exception12(a))
        ∧ ¬Exception13(a, bew)) ∧ ¬Exception14(a)); _p1 <- "Bundesgericht", _p2 <- b)
    ∨ ((((((((((((anfechtungsObjekt(b, a) ∧ kantonalerErlass(a)) ∧ ¬Exception4(a))
        ∧ ¬Exception5(a)) ∧ ¬Exception6(a)) ∧ ¬Exception7(a, bew)) ∧ ¬Exception8(a)) ∧ ¬Exception9(a))
        ∧ ¬Exception10(a)) ∧ ¬Exception11(a)) ∧ ¬Exception12(a)) ∧ ¬Exception13(a, bew))
        ∧ ¬Exception14(a)); _p1 <- "Bundesgericht", _p2 <- b))
    ∨ ((((((((((((anfechtungsObjekt(b, a) ∧ oeffentlichesRecht(a)) ∧ ¬Exception4(a))
        ∧ ¬Exception5(a)) ∧ ¬Exception6(a)) ∧ ¬Exception7(a, bew)) ∧ ¬Exception8(a)) ∧ ¬Exception9(a))
        ∧ ¬Exception10(a)) ∧ ¬Exception11(a)) ∧ ¬Exception12(a)) ∧ ¬Exception13(a, bew))
        ∧ ¬Exception14(a)); _p1 <- "Bundesgericht", _p2 <- b))

□ ∀b. ∀g. (beurteilt(g, b) ∧:L ⊤) →:L zulaessig(g, b)
```

```
anfechtungsObjektpred(b__: int, a__: int)
betrifftpred(a__: int, bet__: string)
betrifftPersonpred(a__: int, p__: string)
beurteilt(g__: string, b__: int)-
beurteilungsAnspruchpred(r__: string, a__: int)
bewilligungpred(g__: string)
gebietpred(a__: int, geb__: string)
kantonalerErlasspred(a__: int)
oeffentlichesRechtpred(a__: int)
raeumtAnspruchEinpred(r__: string, g__: string)
stelltAuslieferungsersuchen(s__: string, p__: string)
suchtSchutz(p__: string, s__: string)
trifftpred(g__: string, a__: int)
ts(~return_value__: time)
zulaessigpred(g__: string, b__: int)
fun day(t__: time) -> time
Exception14(a__: int)
Exception13(a__: int, bew__: string)
Exception12(a__: int)
Exception11(a__: int)
Exception10(a__: int)
Exception9(a__: int)
Exception8(a__: int)
Exception7(a__: int, bew__: string)
Exception6(a__: int)
Exception5(a__: int)
Exception4(a__: int)
```