A Methodology for Encoding Regulatory Rules

Hanif Bhuiyan[1,2], Francesco Olivieri[1], Guido Governatori[1], Mohammad Badiul Islam[1], Andy Bond[2], and Andry Rakotonirainy[2]

[1] Data61, CSIRO
{hanif.bhuiyan,francesco.olivieri,guido.governatori,badiul.islam}@data61.csiro.au
[2] Queensland University of Technology (QUT), Centre for Accident Research and Road Safety (CARRS-Q), Queensland, Australia
{h.bhuiyan,andy.bond,r.andry}@qut.edu.au

Abstract. This paper introduces a methodology for the encoding of rules into a semantic logical format to facilitate the automated reasoning process. We demonstrate how to identify, capture, combine, and thus formulate all the components from rules into a computationally-oriented formalism. The need for the methodology is motivated by the desire for automated reasoning of automated vehicle information regarding traffic rules. We use Defeasible Deontic Logic as a formal foundation of our methodology. The overtaking traffic rule is our use-case to illustrate the usefulness of our methodology. Through this use-case, it is seen that the logical semantic representation of the traffic rules seems conceivable to support automated reasoning. This paper includes the source materials, the use cases, proposed methodology, and the example of encoding.

Keywords: Rules · Terms · Legal norms · Defeasible Deontic Logic.

1 Introduction

Road crash is a major concern of global public health due to the epidemic growth of road fatalities. Every day, more than 3,700[1] people die due to road crashes. From world road crash statistics, it was found that the driver's behaviour is the main contributing factor for 90% of these crashes [1]. In Australia, around 30% of road crashes occurred due to speeding[2]. From 2013-2017[3], in Queensland, the average death due to high speed was 58 per year. To overcome these driver's behaviour related errors, Automated Vehicles (AVs) can be introduced to follow traffic rules properly, which can reduce road fatalities and injuries, and improve road safety [17]. As AVs are designed and programmed to follow traffic rules [20],

[1] https://www.who.int/violence_injury_prevention/road_traffic/en/
[2] https://www.budgetdirect.com.au/car-insurance/research/car-accident-statistics.html
[3] https://streetsmarts.initiatives.qld.gov.au/speeding/factsheet

therefore, it is suggested that AVs would be the solution to traffic violations [17]. However, traffic rules are expressed in natural language, therefore it is required to encode them into a machine-computable format to be processable by AVs. Only then it might be effective for monitoring or validating AV driving action through automatic traffic rule reasoning [2,29,26].

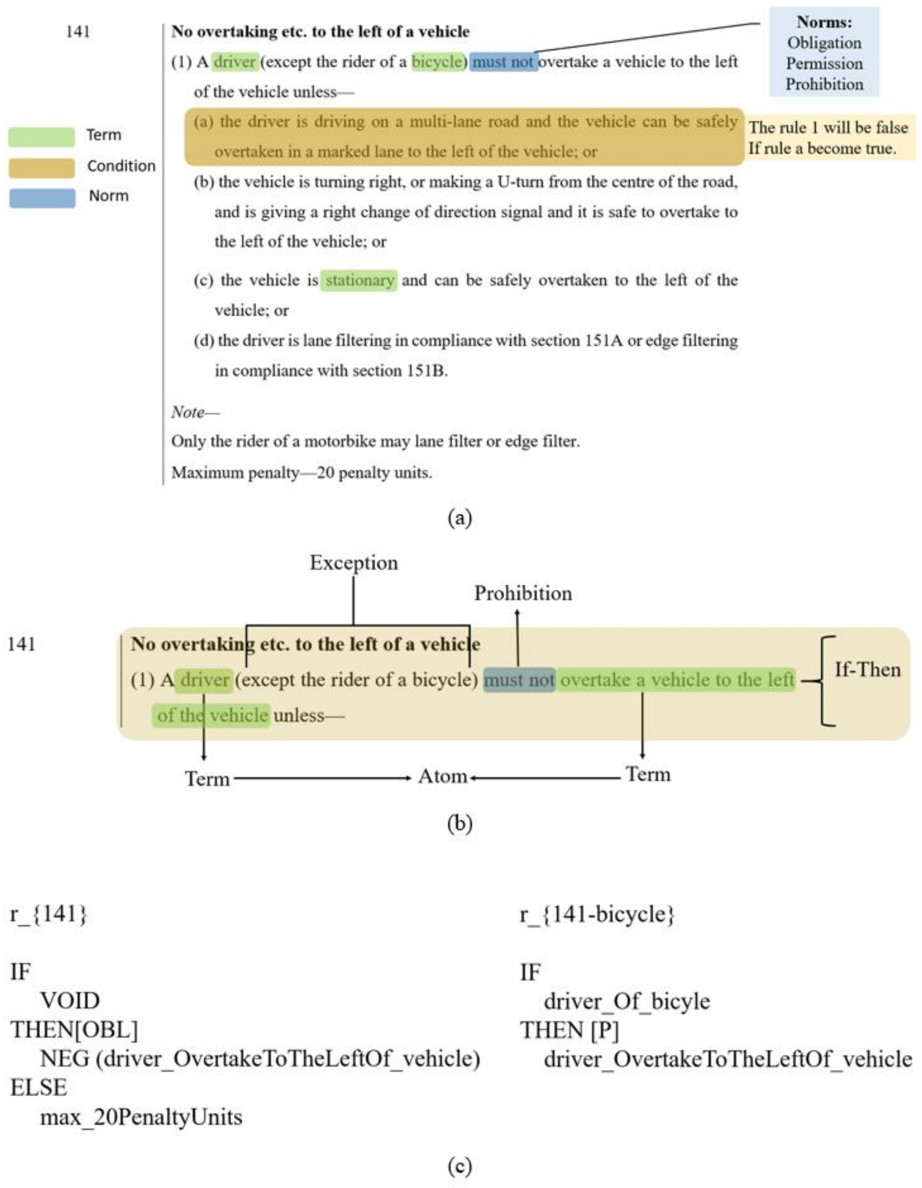
Rule encoding is one of the important requirements for compliance checking, automated reasoning, and legal validation [31,30]. However, rule encoding is a complex task due to its domain-specific, sentence length, clause embedding, and structure. Moreover, rules include thousands of provisions and complex norms, [4], which make the encoding task more challenging. Several research works have been done to address these issues [28,21]. Standards for the representation of norms and legal knowledge have been proposed [24]. For example, XML serialisations have been presented for business contracts [10], business process compliance [13], GDPR regulations [5,25] and building regulations [8]. There are also some commercial products available (Oracle Policy Automation [4]) that provide services to translate regulations into executable language and also served with an interactive natural language interface on the web.

Traffic rules are often detailed and complex and, therefore, it is a big challenge to encode them. There have been several efforts to encode traffic rules for different purposes [7,26]. Costescu [6] proposed a traffic rules formalisation method using Higher Order Language (HOL) to keep the AV accountable. Shardin et al [29] presented an expert system to formalise the traffic rules for controlling the autonomous vehicle in certain situations. However, none of the previous research considers how to resolve the conflicts and how to handle norms and exceptions in the rules, which are the most important variant features of the traffic rules. Therefore, here we propose a Defeasible Deontic Logic (DDL) based encoding methodology to translate traffic rules (natural text) into a semantic logical format (machine-computable). The integration of DDL makes our rule encoding methodology more applicable and useful for handling the exceptions, situation conflicts, ambiguities, and various forms of norms. An example of translating a traffic rule into semantic logical format is shown in Figure 1. The use case we used in this work is derived from the Queensland Traffic Rules, Australia[5]. The motivation for the use of DDL is provide below.

Legal reasoning has some special features, which are norms and exceptions. Usually, norms set baseline conditions but later, they are open to exceptions. These exceptions are also expressed as norms. Norms prescribe behaviour using permissions, obligations, and prohibitions. Therefore, we need a logical approach to handle such exceptions in the traffic rules. In legal reasoning, DDL has been successfully proposed [12] to handle the exceptions and it is also seen that it does not undergo from problems affecting other logics used for reasoning about compliance and norms [11]. For the representation of rules, DDL is a conceptually profound approach, and at the same time, it exhibits a computationally feasible

[4] `https://www.oracle.com/technetwork/apps-tech/policy-automation/overview/index.html`

[5] `https://www.legislation.qld.gov.au/view/html/inforce/current/sl-2009-0194#sec.20`



141 **No overtaking etc. to the left of a vehicle**

(1) A driver (except the rider of a bicycle) must not overtake a vehicle to the left of the vehicle unless—

(a) the driver is driving on a multi-lane road and the vehicle can be safely overtaken in a marked lane to the left of the vehicle; or

(b) the vehicle is turning right, or making a U-turn from the centre of the road, and is giving a right change of direction signal and it is safe to overtake to the left of the vehicle; or

(c) the vehicle is stationary and can be safely overtaken to the left of the vehicle; or

(d) the driver is lane filtering in compliance with section 151A or edge filtering in compliance with section 151B.

Note—

Only the rider of a motorbike may lane filter or edge filter.

Maximum penalty—20 penalty units.

(a)

141 **No overtaking etc. to the left of a vehicle**

(1) A driver (except the rider of a bicycle) must not overtake a vehicle to the left of the vehicle unless—

(b)

```
r_{141}                                        r_{141-bicycle}

IF                                             IF
    VOID                                          driver_Of_bicyle
THEN[OBL]                                      THEN [P]
    NEG (driver_OvertakeToTheLeftOf_vehicle)      driver_OvertakeToTheLeftOf_vehicle
ELSE
    max_20PenaltyUnits
```

(c)

Fig. 1. (a) A snippet of Overtaking Traffic Rules (b) Terms, norms, and if-then structure Identification (c) Encoding of traffic rules 141 (1) using Defeasible Deontic Logic.

environment to reason about them. Furthrmore, Defeasible Deontic Logic (DDL) is computationally feasible since we can compute the extension of a theory in linear time [15,14] and it is suitable to efficient implementations [19,16].

There are several situations in conflict within the Queensland traffic rules, which may create problems for automatic traffic rule reasoning regarding AV. For example, in Traffic rules, Part 12, Division 3, Rule 170 says

> (2) A driver must not stop on a road within 20m from the nearest point of an intersecting road at an intersection with traffic lights, unless the driver— (a) stops at a place on a length of road, or in an area, to which a parking control sign applies; and (b) is permitted to stop at that place under this regulation.

Also, in Part 12, Division 1, Rule 165 recites

> the driver may stops at a particular place, or in a particular way because the condition of the driver, a passenger, or the driver's vehicle makes it necessary for the driver to stop in the interests of safety, and the driver stops for no longer than is necessary in the circumstances.

These two rules can contradict each other regarding stopping on a road. The main purpose of these examples is not to show the difference in traffic behaviour but rather to find the solution to follow the appropriate rule in such different situations. These type of conflicts often may be solved by the explicit priority rules. However, Leens and Lucivero [20] noticed that subtle conflict arises in the rules that may not be solved by rule priorities. To solve such issues, there is one effective approach by [10], which works based on logic using a suitable variant. Regarding such issues, approaches by [18], [22] are also well known in nonmonotonic logic areas.

2 Traffic Rules Encoding

The proposed methodology consists of four modules as shown in Figure 2. Here, we introduce our methodology to translate the traffic rules into the semantic logical format. The input of the system is traffic rules (natural text) as shown in Figure 1(a). In the first module, we define atoms from the rules. In the second module, we determine the norms. The if-then structure is identified and generated from the rules in the third module. Finally, we use Defeasible Deontic Logic (DDL) on the atoms, norms and if-then structure to make the semantic logical format. The modules are explained below. An example of making a semantic logical format of traffic rules is shown in Figure 1.

Define Atoms This section outlines in brief that how we define atoms from rules. An atom is a predicate symbol including constants or variables that contains no logical connectives. Here to define atoms, we use terms of the rule sentence. A term is a variable or an individual constant in the sentence. This work deals with those variables and constants that refer to subject (s), predicate (p), property (pr), object (o), and qualifier (q) (Figure 3) in the rule sentence.

In natural language a subject is refers to that term about which something is said in the sentence. The something which is said about something is the predicate

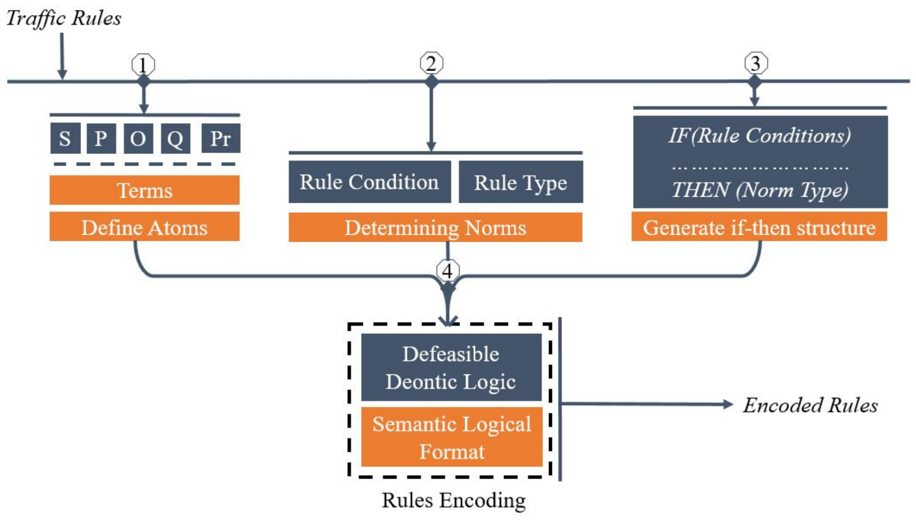


Fig. 2. Workflow for traffic rules encoding.

of the sentence. The predicate of a subject-predicate sentence indicates relation or a property. The object is what that subject does something to. In another word, object are the results of action. Qualifiers are that terms which usually enhance or limits the other word meaning. In one sense it can be said as an adverb of the sentence. Before generating terms, some article preprocessing are done on the text. Like, we are not considering the verbs (auxiliary, principal and modal) for this task. In logic, Subjects are variable or constant in the rule sentence that refers to the entity always. Predicate always refers to the properties or action of entities. Properties indicate the relation between subject and predicate. Object refers to

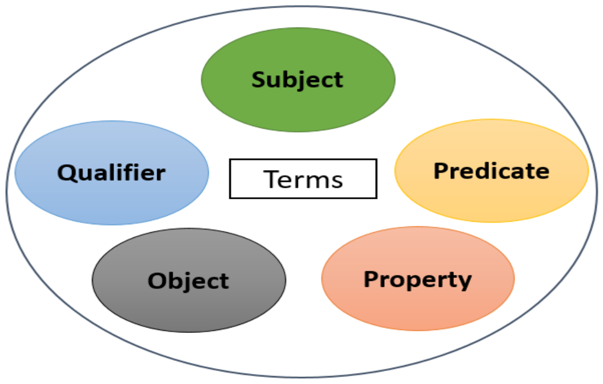


Fig. 3. Terms of the traffic rules.

the properties of the entities. Qualifiers refer to the variable that enhances or limits the entities.

Atom is a combination of these terms and composed as a statement which only can be evaluated as true or false. For example "The bus breaks the traffic rule". According to the linguistic perspective, the term 'bus' is the subject of the sentence as this sentence is about this. The term 'traffic rule' is the object of the sentence as the subject is doing something to it. The term (verb) 'breaks' is the predicate of the sentence as it expresses the relation between the subject (bus) and object (traffic rule). In logical approach, 'bus' is that variable (subject) which refers the entity of the sentence. 'breaks' is the predicate constant which refers the action of the entity. 'traffic rule' is an individual of the sentence which refers the properties of the entity. So in a logical way, we can represent the above example as Predicate (subject, object): B(b,t): Breaks (bus, truck). Therefore the atom can be represented as Subject-Predicate-Object: bus_Breaks_theTrafficRule.

In traffic rules, the rule structure are not equally structured. Due to this heterogeneity of the rules information, the atom structure varies. An example of identifying terms and defining atom from the traffic rule is shown in Figure 1 (b) & (c). Throughout the empirical study of the Queensland Traffic Rules, we semantically define the atom in terms of five aspects which are: Subject-Predicate-Object; Subject-Predicate-Qualifier-Object; Subject-Property, Subject-Predicate-Object-Object; Subject-Qualifier-Predicate-Object. Some examples are shown below.

Example 1: Overtaking traffic rule section 151A: 3 (c) "the rider rides in a school zone".

Subject	Predicate	Object
the rider	rides	in school zone

Generated Atom: rider_IsRidingIn_schoolZone.

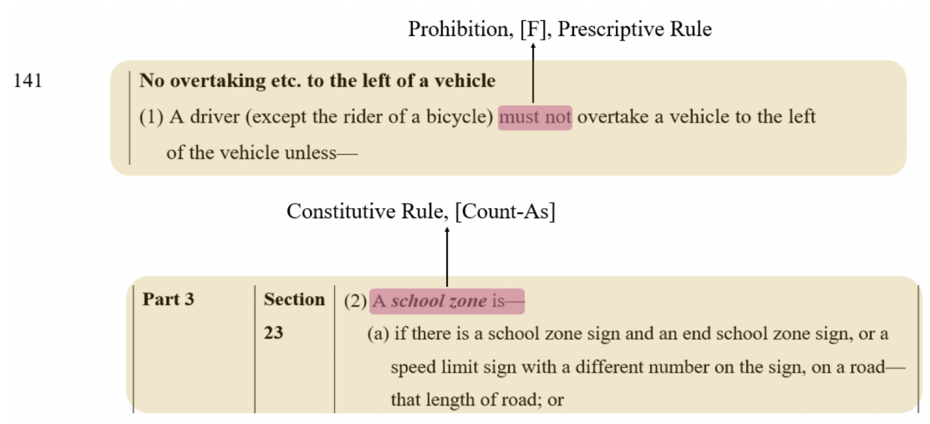
Example 2:Overtaking traffic rule section 140: b "the driver can safely overtake the vehicle".

Subject	Predicate	Qualifier	Object
the driver	overtake	safely	the vehicle

Generated Atom: driver_CanSafelyOvertake_vehicle.

Determining Norms Norms stipulate the conditions in the rule to perform specific actions. Every norm is represented by one or more rules, which could either constitutive or prescriptive rules [9]. Constitutive rules define the terms specific to legal documents, whereas prescriptive rules are used to encode the obligation, permission, and prohibition, ..., and the conditions under which they are entering into force to follow according to legal document. An obligation is a type of legal requirement where the subject has to perform an action otherwise a violation is triggered, whereas prohibition is about an action that cannot be performed; if it is performed, then the result represents a violation.

Prescriptive rules are determined based on conceptual semantic understanding and some special keywords, which are “must”, “must not”, “should”, “ought”, etc. For prescriptive rules, we consider only obligation, prohibition and permission norms. We identify the constitutive rules through our knowledge of understanding and descriptive notions in the sentences. Some examples of descriptive notions are “it is”, “means”, “does”, “does not”, etc. An example of determining prescriptive and constitutive norms from overtaking traffic rules, is shown Figure 4.


Prohibition, [F], Prescriptive Rule
Constitutive Rule, [Count-As]

141 **No overtaking etc. to the left of a vehicle**

(1) A driver (except the rider of a bicycle) must not overtake a vehicle to the left of the vehicle unless—

Part 3	**Section 23**	(2) A ***school zone*** is— (a) if there is a school zone sign and an end school zone sign, or a speed limit sign with a different number on the sign, on a road—that length of road; or

Fig. 4. An example of determining norms.

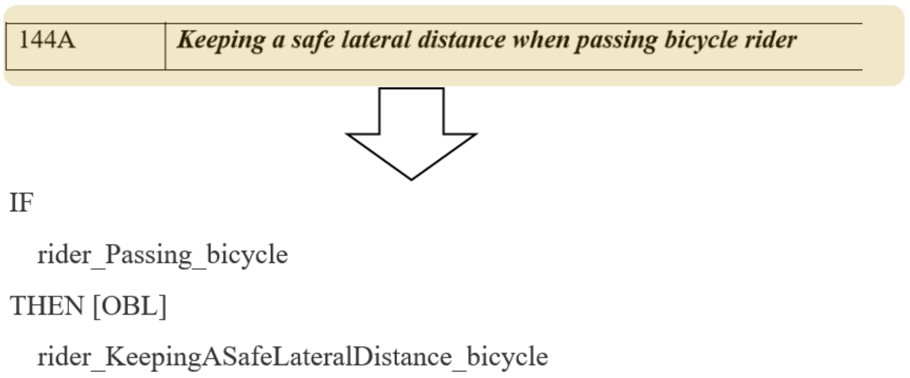
Generate if-then Structure Typically rules contain conditions and norms, which control the behaviour of the subject. From a legal perspective, rules use conditions on some actions to achieve particular behaviours. Rules can be analysed based on the subject’s behaviour and the circumstances [9]. In AI & Law domain it is widely accepted that norms have if-then conditional structure [27]. Therefore, for encoding traffic rules, we make the conditional structure using the atoms and norms (prescriptive or constitutive).

$$\text{if } (X_1, \ldots, X_n)$$

$$\text{then [norms]}$$

$$\{\ Y\ \}$$

Here, $(X_1, \ldots, X_n)$ is the antecedent (premises), and Y is the conclusion (consequent) of the rule. In a traffic rules context, norms represent the situation of actions. Some actions are mandatory to follow; some are not. For example, “Rule 144A: Keeping a safe lateral distance when passing bicycle rider”. The rule expresses an obligation for the driver while passing a bicycle rider. A simple example of generating an if-then structure is shown in Figure 5.



144A	***Keeping a safe lateral distance when passing bicycle rider***

IF

rider_Passing_bicycle

THEN [OBL]

rider_KeepingASafeLateralDistance_bicycle

Fig. 5. An example of making if - then structure from rules.

Rules Encoding Defeasible Deontic Logic (DDL) is the combined form of defeasible logic and deontic logic. DDL can deal with both normative and defeasible reasoning based on defeasible logic [23,3] and deontic logic. We identify and combine atoms, norms, and if-then structure using DDL to create semantic logical format (machine-computable) of the rules. A brief overview of how DDL is used to represent traffic rules is given below.

Defeasible theory consists of five distinct knowledge foundations: strict rules, facts, defeasible rules, superiority relations, and defeaters [3]. The theoretical notion of defeasible logic (DL) is D $(F, R, \succ)$ where, F is a set of facts, R is a set of rules, and $\succ$ is a superiority relation over R.

DL consists of a finite set of literals, where a literal is either an atomic proposition or its negation. Given a literal X, $\sim X$ denotes its complement. That is, if $X = Y$ then $\sim X = \neg Y$, and if $X = \neg Y$ then $\sim X = Y$.

Facts (F) are conclusive and unambiguous statements. A fact is represented either in the form of state affairs (literal or modal literal) or actions that have been performed and are considered as always true. For example, "Honda is a Motorbike", is represented by: Motorbike(Honda).

A rule $(r \in R)$ describes the relationship between premises and conclusion and we can specify the strength of this relationship. Based on the relationship strength of the rules, we can differentiate defeasible rules, strict rules and defeaters [12]. These rules are represented by the following expressions $X_1, \ldots, X_n \rightarrow Y$ (Strict Rules), $X_1, \ldots, X_n \Rightarrow Y$ (Defeasible Rules) and $X_1, \ldots, X_n \rightsquigarrow Y$ (Defeaters), where $X_1, \ldots, X_n$ is the antecedent or premises (clauses) and Y is the consequent or conclusion (effect) of the rule. Besides these, rules also contain free variables which are interpreted as the ground instances.

Strict rules are rules in the classical sense: whenever the premises are indisputable (e.g. a fact) so is the conclusion. For example, "a motorbike is vehicle," formally: Motorbike(Honda) $\rightarrow$ Vehicle(Honda).

Defeasible rules are rules that can be defeated by contrary evidence. An example of a traffic rule is: "Part 11 Division 4 Section 151 B (3): the rider is taken to have unlawfully edge filtered along the length of road—(a) the rider does not hold an O type licence for the class of the motorbike”. In summary, a motorbike can edge filter if the rider holds an O type licence, so formally we can write: Motorbike(Honda) $\Rightarrow$ EdgeFilteringVehicle(Honda). From this information, it can be concluded that a motorbike can edge filter on the road unless there is any evidence is provided that the motorbike cannot edge filter. A defeasible rule with empty premises would be considered as a presumption.

Table 1. A snippet of Turnip about defeasible reasoning.

Atoms:	
Atom Motorbike “Honda is a motorbike”	
Atom Vehicle “Honda is a vehicle”	
Atom EdgeFilteringVehicle “Honda can Edge Filter”	
Atom rider_HoldOTypeLicence “The rider hold O Type Licence”	
Atom Learner “Who is just learning driving”	
Atom Provisional “Who know driving but not yet become professional driver”	
Atom Professional “Who is the professional driver”	
Rules:	
r0:$\Rightarrow$ rider_HoldOTypeLicence	
Motorbike $\rightarrow$ Vehicle	
Motorbike $\Rightarrow$ EdgeFilteringVehicle	
$\neg$ rider_HoldOTypeLicence & Vehicle $\rightsquigarrow$ $\neg$ EdgeFilteringVehicle	
r1: Learner | Provisional $\rightarrow$ $\neg$ rider_HoldOTypeLicence	
r2: Professional $\rightarrow$ rider_HoldOTypeLicence	
r1 >>**r0**	
r2 >>**r1**	
Facts:	**Result**:
Motorbike	$\neg$(rider_HoldOTypeLicence)
Learner	Learner
Provisional	Provisional
Professional	Vehicle
	Motorbike
	Professional
	rider_HoldOTypeLicence

Defeaters are rules, that are used to prevent the conclusion. As an example, $\neg$(rider_HoldOTypeLicence_Vehicle(Honda)) $\rightsquigarrow$ $\neg$EdgeFilteringVehicle(Honda). From this rule we can state that Honda is an edge filtering vehicle but if the rider of a Honda does not hold a O type licence, then it cannot edge filter on the road. This statement can prevent the conclusion of edge filtering. This is not also supporting the no edge filtering.

Defeasible Logic is a non-monotonic, skeptical approach that does not support a contradictory conclusion. It aims to resolve the conflicts between knowledge. For example, suppose there is information, and it has support to conclude A, but also there is information which does not support A and prevents it from concluding A. If the support for A has priority over $\neg A$, then it might be possible to conclude A. In such scenarios no conclusion can be made unless the rules are prioritized. The superiority relation ($\succ$) used the priority set among the rules, where one rule may override the conclusion of other rules. An example of edge filtering concept using the Turnip Engine is shown in Table 1.

In Table 1, we can see that, no conclusive decision can be made until we use the superiority relation $\succ$. Without using the superiority relation, we can only conclude that Honda is an Edge Filtering vehicle. After using the superiority relation like $r1 \succ r0$ and $r2 \succ r1$, then we can conclude that (as in the result section of Table 1) a professional rider can Edge Filter if he/she has a O Type Licence but learner or provisional rider cannot edge filtering.

In addition to defeasibility, traffic rules also engage with deontic concepts, which are obligation (O), permission (P), and prohibition (F). For example, considering the concept "overtake", we can define these notions as:

$$\begin{gathered}
[\mathrm{F}]\text{overtake} \equiv [\mathrm{O}]\neg\text{overtake} \\
[\mathrm{O}]\text{overtake} \equiv [\mathrm{F}]\neg\text{overtake} \\
[\mathrm{P}]\text{overtake} \equiv \neg[\mathrm{O}]\neg\text{overtake}
\end{gathered}$$

An example of representing traffic rules using DDL is given below. For this example, we use the Queensland Overtaking traffic rules (Part 11 Division 3 rule 140 in Traffic Rules).

$\emptyset$ (Empty Set) $\Rightarrow$ [F] Overtake

driver (HasClearViewOf_approachingTraffic)
$\wedge$ driver (CanSafelyOvertake_vehicle) $\Rightarrow$ [P] Overtake

A simple and complete example of encoding Overtaking Traffic Rules: 141 (1) using Defeasible Deontic Logic (DDL) is shown in Figure 1.

3 Limitation

This work uses Defeasible Deontic Logic (DDL) to define a logical semantic representation of traffic rules; however, there are some issues regarding the accuracy and completeness of the representation. Given the sophisticated and varied nature of traffic rules, identifying all the terms, norms, rule types and conditions is a challenging task, as these components found in explicit or implicit linguistic forms. As noted, the generic interpretation is not represented here. Only five different aspects of combination between subject, predicate, object, property, and qualifier are considered for defining atoms. In terms of norms determination from rules, only explicit types of norms (obligation, permission, and prohibition) are considered, although there might be different types of permission and other normative effects [14]. There are some questions which may arise regarding the use of defeasible deontic logic in this work. As we did not evaluate the methodology based on any gold standard or any other approaches. Despite the above issues, there are significant advantages of our proposed encoding methodology, which are domain independence and scope of applicability. This methodology can be used in other domains such as anti-money laundering rules and regulations, university rules and regulations, etc.

4 Conclusion

Encoding the complex and varying nature of traffic rules is a challenging task. Any wrong encoding of the rule can adversely effect on the reasoning process. Therefore, we proposed a Defeasible Deontic Logic (DDL) based encoding methodology for translating traffic rules into a semantic logical format (machine-computable), which can be used for automatic traffic rule reasoning to validate automated vehicle legal behaviour. The methodology incorporates the components and behaviour of regulations such as atoms (defined from terms), norms, and if-then structure to analyse the rule content as well as identify the actions and activities of the rules. DDL is applied to the characteristics of the rules to resolve conflicts and understand the norms and exceptions more explicitly. In the future, we plan to enhance the scale and scope of this proposed methodology. We intend to cover all possible combinations of terms and normative effects for this task. Besides work on the overall traffic rules, we also plan to work on other domains to make this encoding methodology more efficient and standard in the field of Law and AI research.

References

1. Åberg, L.: Traffic rules and traffic safety. Safety science **29**(3), 205–215 (1998)
2. Aladin, D., Varlamov, O., Chuvikov, D., Chernenkiy, V., Smelkova, E., Baldin, A.: Logic-based artificial intelligence in systems for monitoring the enforcing traffic regulations. In: Materials Science and Engineering. vol. 534, pp. 12–25. IOP Publishing (2019)
3. Antoniou, G., Billington, D., Governatori, G., Maher, M.J.: Representation results for defeasible logic. ACM Transactions on Computational Logic (TOCL) **2**(2), 255–287 (2001)
4. Banks, T.L., Banks, F.Z.: Corporate legal compliance handbook. Aspen Publishers Online (2010)
5. Bartolini, C., Lenzini, G., Robaldo, L.: The DAta Protection REgulation COmpliance Model. IEEE Secur. Priv. **17**(6), 37–45 (2019)
6. Costescu, D.M.: Keeping the autonomous vehicles accountable: Legal and logic analysis on traffic code. In: Conference Vision Zero for Sustainable Road Safety in Baltic Sea Region. pp. 21–33. Springer (2018)

7. Damm, W., Peter, H.J., Rakow, J., Westphal, B.: Can we build it: formal synthesis of control strategies for cooperative driver assistance systems. Mathematical Structures in Computer Science **23**(4), 676–725 (2013)
8. Ghannad, P., Lee, Y.C., Dimyadi, J., Solihin, W.: Automated bim data validation integrating open-standard schema with visual programming language. Advanced Engineering Informatics **40**, 14–28 (2019)
9. Gordon, T.F., Governatori, G., Rotolo, A.: Rules and norms: Requirements for rule interchange languages in the legal domain. In: RuleML 2009. pp. 282–296. LNCS 5858, Springer (2009)
10. Governatori, G.: Representing business contracts in RuleML. International Journal of Cooperative Information Systems **14**(2-3), 181–216 (2005)
11. Governatori, G.: The Regorous approach to process compliance. In: 2015 IEEE 19th International Enterprise Distributed Object Computing Workshop. pp. 33–40. IEEE (2015)
12. Governatori, G.: Practical normative reasoning with defeasible deontic logic. In: d'Amato, C., Theobald, M. (eds.) Reasoning Web 2018, pp. 1–25. LNCS 11078, Springer International Publishing, Cham (2018)
13. Governatori, G., Hashmi, M., Lam, H.P., Villata, S., Palmirani, M.: Semantic business process compliance checking using LegalRuleML. In: Knowledge Engineering and Knowledge Management. pp. 746–761. LNAI 10024, Springer International (2016)
14. Governatori, G., Olivieri, F., Rotolo, A., Scannapieco, S.: Computing strong and weak permissions in defeasible logic. Journal of Philosophical Logic **42**(6), 799–829 (2013)
15. Governatori, G., Rotolo, A.: BIO logical agents: Norms, beliefs, intentions in defeasible logic. Autonomous Agents and Multi-Agent Systems **17**(1), 36–69 (2008)
16. Governatori, G., Rotolo, A., Rubino, R.: Implementing temporal defeasible logic for modeling legal reasoning. In: JSAI International Symposium on Artificial Intelligence. pp. 45–58. Springer (2009)
17. Khorasani, G., Tatari, A., Yadollahi, A., Rahimi, M.: Evaluation of intelligent transport system in road safety. International Journal of Chemical, Environmental & Biological Sciences (IJCEBS) **1**(1), 110–118 (2013)
18. Kowalski, R.A., Toni, F.: Abstract argumentation. In: Logical Models of Legal Argumentation, pp. 119–140. Springer (1996)
19. Lam, H.P., Governatori, G.: The making of SPINdle. In: RuleML 2009. pp. 315–322. LNCS 5858, Springer, Heidelberg (2009)
20. Leenes, R., Lucivero, F.: Laws on robots, laws by robots, laws in robots: regulating robot behaviour by design. Law, Innovation and Technology **6**(2), 193–220 (2014)
21. de Maat, E., Winkels, R.: Automated classification of norms in sources of law. In: Semantic processing of legal texts, pp. 170–191. Springer (2010)
22. McCarty, L.T., Cohen, W.W.: The case for explicit exceptions. In: LPNMR. pp. 81–94 (1990)
23. Nute, D.: Defeasible deontic logic, vol. 263. Springer Science & Business Media (2012)
24. OASIS: LegalRuleML Core Specification Version 1.0. OASIS Committee Specification (2018)
25. Palmirani, M., Governatori, G.: Legal knowledge modelling for GDPR compliance checking. In: JURIX 2018. pp. 101–110. IOS Press (2018)
26. Rizaldi, A., Althoff, M.: Formalising traffic rules for accountability of autonomous vehicles. In: 2015 IEEE 18th International Conference on Intelligent Transportation Systems. pp. 1658–1665. IEEE (2015)

27. Sartor, G.: Legal reasoning. Springer (2005)
28. Sergot, M.J., Sadri, F., Kowalski, R.A., Kriwaczek, F., Hammond, P., Cory, H.T.: The british nationality act as a logic program. Communications of the ACM **29**(5), 370–386 (1986)
29. Shadrin, S.S., Varlamov, O.O., Ivanov, A.M.: Experimental autonomous road vehicle with logical artificial intelligence. Journal of advanced transportation **2017** (2017)
30. Wyner, A., Governatori, G.: A study on translating regulatory rules from natural language to defeasible logics. In: Fodor, P., Roman, D., Anicic, D., Wyner, A., Palmirani, M., Sottara, D., Lévy, F. (eds.) RuleML Challenge 2013. vol. 1004. CEUR-WS.org (2013)
31. Wyner, A.Z., Peters, W.: On rule extraction from regulations. In: JURIX 2011. pp. 113–122 (2011)