





Run-Time Norms Synthesis in Dynamic Environments with Changing Objectives

Maha Riad^(✉) , Saeedeh Ghanadbashi , and Fatemeh Golpayegani 

School of Computer Science, University College Dublin, Dublin, Ireland
{maha.riad,saeedeh.ghanadbashi}@ucdconnect.ie, fatemeh.golpayegani@ucd.ie
<https://mas3.ucd.ie>

Abstract. Normative Multi-Agent Systems (NorMAS) can model real-world applications as multi-agent systems and facilitate the coordination of the social behaviour of various entities (agents) interacting in an environment using norms. Aligning such norms with the objectives of the agents is crucially important to ensure that applying the norms would not affect the achievement of their objectives. However, when the environment is dynamic, agents can face unseen situations and might need to change their objectives accordingly. Therefore, it becomes more challenging to understand the change, synthesise norms, and align them with such dynamic objectives. This paper introduces a Dynamic Objectives and Norms Synthesizer and Reasoner (DONSR) model to align objectives and norms using a utility-based approach. An ontology-based schema, forward reasoning, and backward reasoning are used to identify the change in the environment and synthesise new objectives. Case-based reasoning enables the dynamic changing and reasoning of previously created objectives and synthesising norms. DONSR is evaluated using multiple simulated traffic scenarios, including different unseen situations (emergency events).

Results show that norms can be synthesised and maintained efficiently while the objectives are being created and changed. Further, DONSR showed its efficacy in handling unseen situations, creating new objectives, and aligning them with the created norms.

Keywords: Normative multi-agent systems · Norms synthesis · Dynamic objectives

1 Introduction

Multi-agent systems (MAS) model complex systems that consist of autonomous agents with various objectives [4, 7]. These objectives are achieved by agents interacting together, competing, or cooperating [5]. Normative multi-agent systems (NorMAS) can coordinate the behaviour of agents [12, 14] using social norms that prohibit, obligate, and give permission for actions that would prompt the effective interaction of a social group of agents in a multi-agent system [3, 10].

For example, in a traffic scenario, if vehicles are considered as agents, if an ordinary vehicle is aware of the norm of giving priority to emergency vehicles, this will avoid accidents.

While it is important to coordinate the system behaviour by applying norms, it is essential to ensure that the norms appliance does not affect the achievement of the objectives of the agents. For example, if the system objective is to minimise the average waiting (stopping) time of vehicles, this objective should still be reachable while the norms are applied. Therefore, several recent works proposed various techniques for aligning norms and objectives. [1, 2] used reasoning techniques to guide the agents to align with objectives. [18] used formal argumentation techniques to reason about the system's objectives and norms. However, in these approaches, the agents need to have reasoning capabilities. In [14, 15], we developed a model that coordinates the system's objectives with norms using a utility-based approach. However, the model does not address dynamic objectives. For choosing the best set of norms aligned with specific objectives, [17] proposed using quantitative approaches (e.g., optimisation technique) and [16] proposed using qualitative techniques (e.g., ranking technique). In contrast, [11] finds the objective with the best performance based on pre-defined norms. These approaches have a significant disadvantage of matching norms with a single objective or having a subset of preferred objectives. However, in reality, all objectives need to be aligned (coordinated) with the whole societal norms regardless of their internal compatibility. Moreover, these models do not consider heterogeneous environments where agents may support varying objectives.

Despite these efforts, there is a gap in having a technique for aligning and reformulating norms and objectives simultaneously in a dynamic environment where agents, network, and situations keep changing. Operating in such an ever-changing environment, agents need to evolve their objectives to cope with the unseen situations [8], and adapt their norms and behaviour to match these changes [14, 15]. For example, in the context of a traffic network, if roadworks occur on one of the roads, the system's objective can evolve to minimise the number of vehicles travelling on this road. Subsequently, norms can change at this time to avoid entering this road.

To solve these problems, we introduce a Dynamic Objectives and Norms Synthesizer and Reasoner (DONSR) model. DONSR represents a normative multi-agent system that is responsible to: (1) Operate in a dynamic environment with unseen situations. (2) Reason objectives and reformulate the objective set based on the changing situations online. (3) Synthesise efficient norms online. (4) Ensure that the process of objectives reasoning does not affect the process of the norms synthesising and appliance and their effectiveness. (5) Align multiple norms with the evolving objectives formulated from the unseen situation. To reach these previously listed goals, DONSR includes the following components:

- An Objective Reasoner Component: which is responsible for reasoning the changes and deciding whether to change the objective, leave it, or create a new one.

- An Objective Formulator Component: which uses backward reasoning to create new objectives [when needed] when an unseen situation occurs.
- A Norm Synthesizer Component: which is responsible for online norms synthesising using case-based reasoning technique.
- A Norm Reasoner Component: which is responsible for aligning the objectives with norms, using a utility-based technique that transforms the current objective chosen by the objectives reasoner component to decide which norm to apply in case of multiple applicable norms.

The remainder of this paper is as follows. Section 2 covers the relevant background related to the techniques used to formulate DONSR. In Sect. 3, an example of a dynamic normative multi-agent system is stated to assist in elaborating DONSR and to be used as the evaluation scenario. DONSR is illustrated in detail in Sect. 4 and then evaluated in Sect. 5. Finally, the conclusion is covered in Sect. 6.

2 Background

2.1 Ontology

Ontologies provide machine-understandable semantics and augment human intelligence [6]. An ontology describes concepts C , properties P , relationships R in a specific environment [20]. It is possible to use a relationship for a particular type of instance (domain) with a particular value (range). An inference rule is an implication of the form: If J_1, J_2 up to J_n are inferable, then J is inferable (see Eq. (1)). Using Semantic Web Rule Language (SWRL), ontology engineers express the inference rules manually [9].

$$J_1, J_2, \dots, J_n \rightarrow J \quad (1)$$

In **forward reasoning**, state observations are used as inputs, and inference rules are applied to extract additional facts until the goal is reached. For example, we can conclude from “A” and “A implies B” to “B”. **Backward reasoning** is based on starting with the goal and chaining through inference rules to find the facts that support it. For example, we can conclude from “not B” and “A implies B” to “not A”.

2.2 Case-Based Reasoning

Case-based reasoning [13] algorithm defines new problems (situations) as cases, and then it searches for similar cases collected from old experiences to find the best solution that was used before.

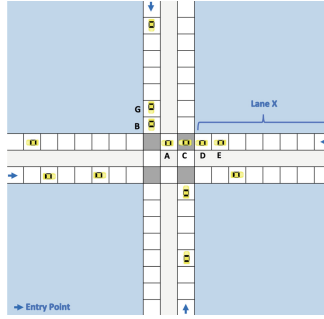


Fig. 1. Traffic grid

3 Running Example (Traffic Scenario)

To facilitate the illustration of concepts in our proposed model and for evaluation purposes, a traffic scenario is used. This scenario includes two main roads with 4 intersections, each with two lanes in opposite directions (see Fig. 1). In this scenario, the vehicles are modelled as agents, and the norms are used to avoid accidents, e.g., a norm is created to avoid going (moving forward) if there is another vehicle in front. The vehicles are of two types, emergency vehicles (e.g., ambulance, police vehicle) and ordinary vehicles. We assume that the intersections are unsignalized, and a traffic manager guides the vehicles to decide their subsequent actions based on the environment. To model the concepts in the traffic environment, we use the ontology shown in Fig. 2. The traffic manager tries to avoid accidents by communicating the synthesised norms to vehicles and, at the same time, aims to reach its current objective. The traffic manager’s objectives can be to:

- **(Objective 1) minimise the average waiting time of vehicles:** minimising vehicles’ waiting (stopping) time is the default objective of the system, as it is used to avoid congestion and maximise the flow of vehicles.
- **(Objective 2) minimise the waiting time in a specific lane:** This might be the case when a road is a bottleneck, and it is downstream, so we would like to minimise the queue length in it.
- **(Objective 3) minimise the waiting time of emergency vehicles.** When an incident happens, we are expected to minimise the waiting time of emergency vehicles so that they can get to the incident location as soon as possible.

4 DONSR: Dynamic Objectives and Norms Synthesizer and Reasoner Model

We propose the Dynamic Objectives and Norms Synthesizer and Reasoner (DONSR) Model to represent dynamic normative multi-agent systems. DONSR

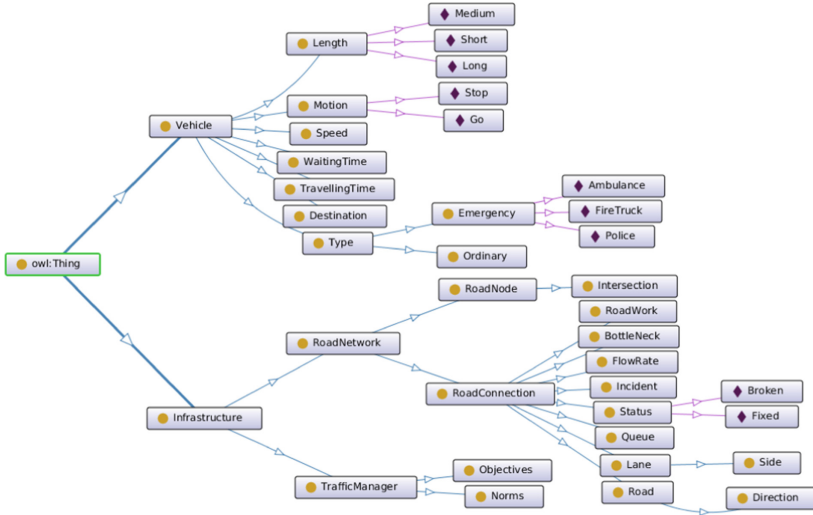


Fig. 2. Ontology for traffic environment, as represented by OntoGraf.

aims to enable online norms and objectives synthesising and reasoning, in addition to aligning the synthesised norms and objectives, and ensuring that none of the processes negatively affect the other processes’ effectiveness. To reach this, DONSR carries out three main functionalities in every time-step: (A) Objectives reasoning and formulation: this is the process of reviewing the current objective and changing it if required (Algorithm 1). (B) Norms synthesising: this is the process of creating a new norm if a new ‘behaviour’ conflict was detected. For example, in the traffic scenario, accidents will be the result of behaviour conflicts, and when DONSR detects a new accident, it will create a new norm (Sect. 4.3). (C) Norm reasoning: this process takes place when there are unmatchable norms. Unmatchable norms is the result of having two(or more) applicable norms that can be applied in the same context, however, their application would result in a conflict. For example, in the traffic scenario, in Fig. 1, if there are two norms defined, n1: stop when there is a vehicle on the right of the intersection. n2: stop when there is a vehicle on the left of the intersection. If both vehicles A and B apply n1 and n2, respectively, both vehicles would not move, resulting in a deadlock. In this case, the norm reasoning process takes place to decide which of these norms (n1 and n2) to be applied and which to ignore (Sect. 4.4). DONSR meets functionalities (A), (B), and (C) using four main components coloured in grey in Fig. 3. The components are:

4.1 The Objective Reasoner Component

We assume the traffic manager models its observation using a schema described by an ontology Ont_D . This schema is used when semantic descriptions are needed (e.g., to explain unexpected events) and is composed of surrounding concepts and

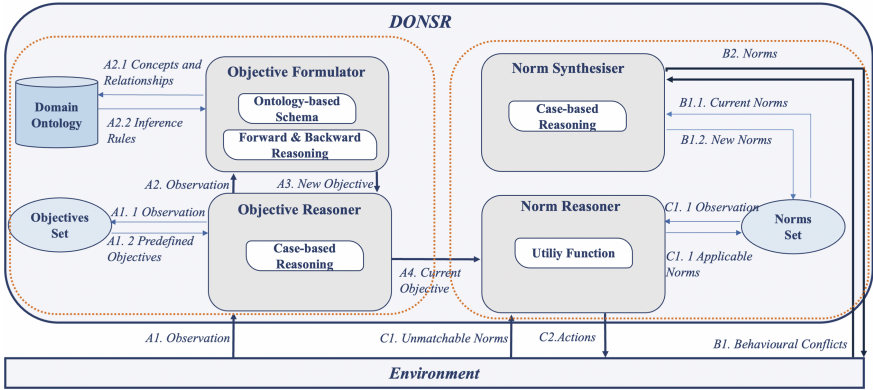


Fig. 3. Dynamic objectives and norms synthesizer and reasoner model

relationships between concepts perceived by the traffic manager. For instance, the concept “Vehicle” can be defined along with the relationship “hasWaiting-Time”. These relationships enable automated reasoning.

The traffic manager has an objective set O in the form of (if situation X happens \rightarrow objective Y should be applied). The traffic manager continuously reasons about the objectives it is pursuing, and when an objective needs to be changed or generated, two cases are possible [8]:

- **Choosing a predefined objective.** Using a case-based reasoning technique, the traffic manager can choose a predefined objective o_c from an objective set based on its current observation obs_i (see lines 6–7 of Algorithm 1).
- **Creating a new objective.** In the absence of a suitable objective in an objective set, the traffic manager uses backward reasoning over inference rules to create a new objective (see lines 8–10 of Algorithm 1).

4.2 The Objective Formulator Component

When an unseen situation is observed, and the traffic manager can not find a suitable objective from an objective set, it uses backward reasoning over inference rules to extract new objectives. The ontology-based schema may include the relationship “hasFlowRate($?r$, Decreased)”. It means there is a decreasing flow rate in the road r . Also, the traffic manager has no predefined objective to handle this new situation. However, inference rules may declare that “hasEnteringVehicles($?r$, Increased) \rightarrow hasFlowRate($?r$, Decreased)”. So to increase the flow rate (as a new objective), the traffic manager needs to decrease the number of entering new vehicles into the road (hasFlowRate($?r$, Increased) \rightarrow hastEnteringVehicles($?r$, Decreased)). That means that a traffic manager can add the new objective “if road r has decreased flow rate \rightarrow the objective is minimising the number of vehicles entering the road r ” to the set of objectives, although that was not part of the original one. One can also say that the new objective was “discovered via inferencing”. We have given two

Algorithm 1. Objectives reasoning & formulation

```

1: for each  $t$  do
2:   Input:  $O, Ont_D, Obs$ 
3:   Output:  $o_c$ 
4:   for each  $obs_i \in Obs$  do
5:      $case \leftarrow obs$ 
6:     if  $case_i \in Cases$  then
7:        $o_c \leftarrow sol_{case_i}$ 
8:     else //comment: Objective Formulator
9:        $IR \leftarrow Query(obs)$ 
10:       $o_c \leftarrow Reason(IR)$ 
11:     end if
12:   end for
13: end for

```

examples of unseen situations and their relevant inference rules in the traffic environment in the following.

- **Example 1:** When a bottleneck happens in a specific road, the traffic manager infers that to reduce the bottleneck in road $r1$, the waiting time of all instances of vehicle b on the road $r1$ should be decreased. This is reached by applying backward reasoning over the inference rules shown in Table 1. Afterwards, the traffic manager uses this inference to create a new objective in the objective set: if road r has bottleneck – $>$ the objective is minimising the waiting time of vehicles on the road r .

Table 1. An example of inference rules, inferring minimising the waiting time of all instances of vehicle b through backward reasoning.

Inference rules
TrafficManager(? i), Intersection(? s), Road(? r), Lane(? l), Vehicle(? b), isOn(? b , ? l), consistOf(? r , ? l), hasWaitingTime(? b , <i>Increased</i>)
– $>$
hasBottleNeck(? r)

- **Example 2:** Suppose an unseen situation occurs when ambulance a enters intersection s , according to the inference rules shown in Table 2, the traffic manager infers through backward reasoning that it should minimise the waiting time of ambulance a until it passes through the intersection. Then, the new objective is added to the traffic manager’s objective set: if the vehicle a with unknown type is at the intersection s – $>$ the objective is minimising the waiting time of vehicle a .

4.3 The Norm Synthesizer Component

The Norm Synthesizer component monitors behavioural conflicts (accidents in case of the traffic scenario) and uses it to create norms. We used the norms

Table 2. An example of inference rules, inferring minimising the waiting time of the ambulance a through backward reasoning.

Inference rules
TrafficManager(?i), Intersection(?s), Road(?r), Lane(?l), Vehicle(?a), isOn(?a, ?l) consistOf(?r, ?l), hasType(?a, <i>Emergency</i>), hasWaitingTime(?a, <i>Increased</i>)
– >
atIntersection(?a, ?s)

synthesising algorithm used in [15], which is based on case-based reasoning. The norms synthesizer checks the accidents (behavioural conflicts) at each time-step and checks if it is similar to a previous case (context of the accident and action taken primary to the accident). The same solution is implemented if an identical case was found with a successful solution. If it is a new case, a new norm is created. A norm is defined as $n_i = (\alpha, \theta(a_i))$, where α is the pre-condition that should exist for the norm to be applied, while θ is a denotic operator to be applied on an action a_i . A denotic operator is either prohibition, obligation, or permission. In the traffic context, α will include the directions of the neighbouring vehicles in the three cells in front of the reviewed vehicle. For example, in Fig. 1, to define a norm for Vehicle A to prohibit its movement in its current context to avoid accidents, the norm will be $n_a = (left(<), front(-), right(<), Proh(Go))$.

4.4 The Norm Reasoner Component

This component resolves the unmatchable norms problem, and aligns the norms and the objectives. When there are two applicable unmatchable norms in the same context, the Norm Reasoner is responsible to decide which norm to apply and which to ignore. For example, if both vehicle A and B will apply the norms $n_a = (left(<), front(-), right(<), Proh(Go))$, and $n_b = (left(<), front(-), right(-), Proh(Go))$ respectively in the next time-step, the Norm Reasoner calculates the utility gained from applying each of the norms, and apply the norm that gives the highest utility. The utility calculated is not only concerned with the directly benefiting agents (Vehicle A & B) from the decision, but also the utility of indirect agents (Vehicle C, D, D, E & G) is calculated. This approach of calculating direct and indirect utility is named 'Accumulated Utility' in [15]. To align the norms and the objectives, the utility is constructed based on the current objective specified by the Objective Reasoner (Sect. 4.1), and sent to the Norm Reasoner Component in step A4 in Fig. 3. We used the same technique used in [15] for converting objectives to the utility function, in which the utility is calculated by getting the inverse of the minimisation objective and the exact value of the maximisation objective. In the traffic example, the utility of each of the four objectives in Sect. 3 sequentially will be:

$$u_1 = -1 * ((wt_e + wt_{ord})/|V|) \quad (2)$$

wt_e : waiting time of emergency vehicles

wt_{ord} : waiting time of ordinary vehicles

$|V|$: number of vehicles

$$u_2 = -1 * wt_{LaneX} \quad (3)$$

wt_{LaneX} : waiting time at of vehicles at lane X

$$u_4 = -1 * wt_e \quad (4)$$

In DONSR, to ensure that none of the norms processes or the objectives processes affect each other, we implement each process in separate components. As represented in Fig. 3 by the orange dotted frames, none of the objectives' components interfere with the norms' components except to notify the Norms Reasoner of the new objective, if the old objective evolved. However, to align the norms and the objectives, we build the utility function used in the Norms Reasoner based on the system's objective to ensure its achievement as well.

5 Empirical Evaluation

5.1 Simulated Environment

We simulate the traffic scenario (in Sect. 3) represented in Fig. 1 by a 19×19 grid using SUMO [19]. The ratio of creating emergency vehicles compared to the number of ordinary vehicles is 12:100. The destination and route of the vehicles were chosen randomly by the simulator while they are created. Every time-step, the simulator prepare (2 to 4) vehicles to start their trip, however, only if there are available entry points they can enter (indicated with blue arrows in Fig. 1). In each time-step, the vehicle can only move 1 cell *Go* or *Stop*.

5.2 Experimental Scenarios

We simulated four scenarios and compared the results with UNS [15], which uses a utility-based approach for aligning norms and objectives, but does not detect environmental changes and evolve objectives. The used utility-function in the current evaluation of UNS is based on minimising the average waiting time (objective 1). In our four scenarios as well, the default objective is minimising the average waiting time of vehicles. However, depending on the scenario, if a change was recognised in the environment, DONSR starts its objective reasoning process and may change the objective accordingly. Also, if an ambulance was seen at an intersection (at any of the scenarios), the objective changes at this time-step (at this intersection only) to minimise the waiting time of emergency vehicles (objective 3). Accordingly, The norm reasoner will use u_2 defined in Eq. 3 in the case of norms reasoning. The simulated scenarios are:

- **Scenario A:** This is the basic scenario, only two objectives (objective 1 and objective 3) are used. objective 3 is applied whenever an ambulance is recognised and gets back to the default objective (objective 1) afterwards.

- **Scenario B:** In this scenario, the default objective changes to objective 2 from time-step 500 to 1000 only at the intersection following Lane X (Check Fig. 1).
- **Scenario C:** The objective used for the intersection at the end of Lane X in Fig. 1, changes every 300 steps, switching between the default objective and objective 2.
- **Scenario D:** This is a faster version of scenario C, where the objective used for the intersection at the end of Lane X, changes every 50 steps, switching between the default objective and objective 2.

5.3 Results

Run-Time Norm Synthesising of Efficient Norms. Figure 4 shows the ability of DONSR to synthesise norms, through reflecting how the effectiveness of norms resulted in zero collisions after the norms set was synthesised. Moreover, it can be seen in all scenarios, even in scenario D (in which the objectives are changing while the norms are still being synthesised), how the norms synthesising process is not affected by the objectives changing process.

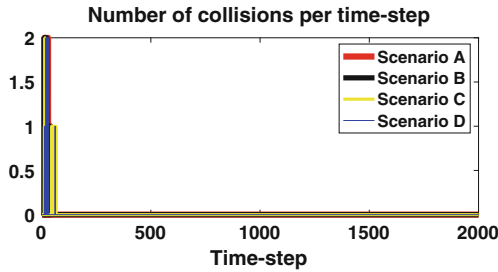


Fig. 4. Number of collisions per time-step

Objective 1: Minimising the Average Waiting Time of Vehicles. This is the default objective used in all of the scenarios and in UNS. As it is used in DONSR interchangeable with other objectives, it is important to analyse the extent its original performance was negatively affected. As seen in Table 3, the highest effect, although insignificant, is in Scenario C and D, which is expected as objective 2 was used more frequently compared to Scenario B (which used it from time-step 500 to 1000 only). Nevertheless, in Scenario A, where this objective was the default objective, and objective 3 was only applied when an ambulance is at the intersection, the average waiting time was improved by 0.093% compared to UNS.

Objective 2: Minimising the Waiting Time in Lane X. In Table 3, the waiting time in Lane X was decreased in the scenarios that involve objective 2 (Scenario B, C and D). Scenario C and D have higher improvement compared to B, as scenario C and D used it as the main objective to be applied several times, while in scenario B objective 2 was used only once between time-step 500 and 1000.

Table 3. Objectives results comparison versus UNS

Objective	UNS	Scenario A	Scenario B	Scenario C	Scenario D
Average waiting time of all vehicles (Obj1)	49.432	49.386	49.331	49.580	49.580
Improvement in Obj1	–	0.093%	0.205%	–0.299%	–0.299%
Average waiting time in Lane X of all vehicles (Obj2)	1.955	1.978	1.683	1.597	1.564
Improvement in Obj2	–	–1.165%	13.917	18.327%	20%
Average waiting time of emergency vehicles (Obj3)	48.308	45.666	46.669	47.893	45.037
Improvement in Obj3	–	5.468%	3.393%	0.858%	6.771%

Objective 3: Minimising the Waiting Time of Emergency Vehicles. As seen in Table 3, all of DONSR scenarios improved the waiting time of emergency vehicles compared to UNS, because it is assumed that UNS can only apply a fixed utility, which is assumed to be formulated based on objective 1 (minimise the average waiting time) only.

6 Conclusion

In this paper, we proposed DONSR, a novel Dynamic Objectives and Norms Synthesizer and Reasoner model, used for run-time norms and objectives alignment, synthesising, and reasoning. DONSR aims to operate in a dynamic environment in which new situations appear that can result in changing objectives. In such an environment, DONSR can formulate new objectives, if needed, using ontology-based schema, forward reasoning, and backward reasoning. Moreover, DONSR synthesises norms and reasons objectives online depending on the situation using case-based reasoning. Furthermore, DONSR ensures that objectives are aligned when applying norms by using utility functions constructed based on the system’s objectives. We evaluated DONSR with several traffic scenarios with different changing objectives. Results showed that DONSR was able to synthesise effective norms that can avoid collisions, evolve three objectives, and further reach these objectives. As future work, we look forward to defining a decentralised mechanism in which the objectives and norms of the agents are reasoned by the agents themselves, in addition to DONSR (central unit).

References

1. Aydođan, R., Kafali, Ö., Arslan, F., Jonker, C.M., Singh, M.P.: NOVA: value-based negotiation of norms. *ACM Trans. Intell. Syst. Technol. (TIST)* **12**(4), 1–29 (2021)
2. Bench-Capon, T., Modgil, S.: Norms and value based reasoning: justifying compliance and violation. *Artif. Intell. Law* **25**(1), 29–64 (2017). <https://doi.org/10.1007/s10506-017-9194-9>
3. Cranefield, S., Savarimuthu, B.T.R.: Normative multi-agent systems and human-robot interaction (2021)

4. Dorri, A., Kanhere, S.S., Jurdak, R.: Multi-agent systems: a survey. *IEEE Access* **6**, 28573–28593 (2018)
5. Edenhofer, S., Stifter, C., Madkour, Y., Tomforde, S., Kantert, J., Müller-Schloer, C., Hähner, J.: Bottom-up norm adjustment in open, heterogeneous agent societies. In: 2016 IEEE 1st International Workshops on Foundations and Applications of Self* Systems (FAS* W), pp. 36–41. IEEE (2016)
6. Fong, A.C.M., Hong, G., Fong, B.: Augmented intelligence with ontology of semantic objects. In: International Conference on Contemporary Computing and Informatics (IC3I), pp. 1–4. IEEE (2019)
7. Ghanadbashi, S., Golpayegani, F.: An ontology-based intelligent traffic signal control model. In: 2021 IEEE International Intelligent Transportation Systems Conference (ITSC), pp. 2554–2561. IEEE (2021)
8. Ghanadbashi, S., Golpayegani, F.: Using ontology to guide reinforcement learning agents in unseen situations. *Appl. Intell.* **52**(2), 1808–1824 (2022). <https://doi.org/10.1007/s10489-021-02449-5>
9. Horrocks, I., Patel-Schneider, P.F., Boley, H., Tabet, S., Grosz, B., Dean, M., et al.: SWRL: a semantic web rule language combining OWL and RuleML. *W3C Member Submission* **21**(79), 1–31 (2004)
10. Mashayekhi, M., Ajmeri, N., List, G.F., Singh, M.P.: Prosocial norm emergence in multi-agent systems. *ACM Trans. Auton. Adapt. Syst. (TAAS)* **17**(1–2), 1–24 (2022)
11. Montes, N., Sierra, C.: Value-guided synthesis of parametric normative systems (2021)
12. Morales, J., Wooldridge, M., Rodríguez-Aguilar, J.A., López-Sánchez, M.: Off-line synthesis of evolutionarily stable normative systems. *Auton. Agent. Multi-Agent Syst.* **32**(5), 635–671 (2018). <https://doi.org/10.1007/s10458-018-9390-3>
13. O’Mahony, E., Hebrard, E., Holland, A., Nugent, C., O’Sullivan, B.: Using case-based reasoning in an algorithm portfolio for constraint solving. In: Irish Conference on Artificial Intelligence and Cognitive Science, pp. 210–216 (2008)
14. Riad, M., Golpayegani, F.: A Normative multi-objective based intersection collision avoidance system. In: Jezic, G., Chen-Burger, Y.H.J., Kusek, M., Sperka, R., Howlett, R.J., Jain, L.C. (eds.) *Agents and Multi-Agent Systems: Technologies and Applications 2022. Smart Innovation, Systems and Technologies*, vol 306. Springer, Singapore (2022). https://doi.org/10.1007/978-981-19-3359-2_25
15. Riad, M., Golpayegani, F.: Run-time norms synthesis in multi-objective multi-agent systems. In: Theodorou, A., Nieves, J.C., De Vos, M. (eds.) *International Workshop on Coordination, Organizations, Institutions, Norms, and Ethics for Governance of Multi-Agent Systems*, pp. 78–93. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-16617-4_6
16. Serramia, M., Lopez-Sanchez, M., Rodriguez-Aguilar, J.A.: A qualitative approach to composing value-aligned norm systems. In: *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 1233–1241 (2020)
17. Serramia, M., López-Sánchez, M., Rodríguez-Aguilar, J.A., Morales, J., Wooldridge, M., Ansotegui, C.: Exploiting moral values to choose the right norms. In: *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, pp. 264–270 (2018)
18. Shams, Z., Vos, M.D., Oren, N., Padget, J.: Argumentation-based reasoning about plans, maintenance goals, and norms. *ACM Trans. Auton. Adapt. Syst. (TAAS)* **14**(3), 1–39 (2020)
19. SUMO: www.eclipse.org/sumo/

20. Zouaq, A., Nkambou, R.: A survey of domain ontology engineering: Methods and tools. In: Nkambou, R., Bourdeau, J., Mizoguchi, R. (eds.) *Advances in Intelligent Tutoring Systems*, pp. 103–119. Springer, Berlin (2010). https://doi.org/10.1007/978-3-642-14363-2_6

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

