Computer Science • 19(4) 2018 https://doi.org/10.7494/csci.2018.19.4.3007

Łukasz Szymański
Bartłomiej Śnieżyński
Bipin Indurkhya

MULTI-AGENT BLACKBOARD ARCHITECTURE FOR SUPPORTING LEGAL DECISION MAKING

Abstract *Our research objective is to design a system to support legal decision making using the multi-agent blackboard architecture. Agents represent experts that may apply various knowledge-processing algorithms and knowledge sources. Experts cooperate with each other using the blackboard to store facts about a current case. Knowledge is represented as a set of rules. The inference process is based on bottom-up control (forward chaining). The goal of our system is to find rationales for arguments that support different decisions for a given case by using precedents and statutory knowledge. Our system also uses top--down knowledge from statutes and precedents to interactively query the user for additional facts when such facts could affect the judgment. The rationales for various judgments are presented to the user, who may choose the most appropriate one. We present two example scenarios in Polish traffic law to illustrate the features of our system. Based on these results, we argue that the blackboard architecture provides an effective approach to modeling situations where a multitude of possibly conflicting factors must be taken into account in the decision making.*

Keywords agent-based legal decision-making model, blackboard architecture, legal decision support

Citation Computer Science 19(4) 2018: 457–477

1. Introduction

Generating arguments supporting a particular decision from a legal point of view in a given scenario is a complex process: what is required is much more than a logic--based inference system that generates judgements from the given facts. Bench-Capon, Prakken, and Sartor [6] provide an overview of the various approaches AI researchers have taken to model the different aspects of legal arguments. Our goal in this research project is to develop a tool that generates different legal arguments (possibly with different outcomes) for a given case. In particular, we would like to design our framework in such a way that human cognitive factors can be incorporated into the system (including affective biases). We aim to realize the following vision of John Wisdom: "(...) The process of argument is not a chain of demonstrative reasoning. It is a presenting and re-presenting of those features of the case which severally cooperate in favor of the conclusion, in favor of saying what the reasoner wishes said, in favor of calling the situation by the name which he/she wishes to call it. The reasons are like the legs of a chair, not the links of a chain" [33] (Quoted in [31], p. 268). With this goal in mind, we aim to incorporate the following features into our design:

1. To be able to combine rules and cases in a mixed-paradigm reasoning.
2. To take into account the desired outcome. So, if we are using the system from the defendant's point of view, it should generate plausible arguments that support the decision in favor of the defendant.
3. To elicit from user relevant facts based on statutory knowledge and precedents that the user may not have initially considered to be relevant.
4. To take into account affective and extra-legal factors in legal decision making [19].

In this paper, we present our current research efforts in incorporating the first three of these features into a legal decision-making system focused on Polish traffic law.

Towards this goal, we decided to use the blackboard architecture, which allows a number of independently acting agents (experts) to compete and cooperate to generate different heuristic arguments. We would like to apply our recent experience in implementing a blackboard-based architecture for a poetry-generating system that incorporates an emotional personality [21]. We would also like to employ our insights from applying this architecture to develop a context-aware recommender system (CARE) that provides detailed textual explanations to support the user in the decision-making process [22]. CARE incorporates a hierarchical structure in which independent modules embodying different aspects of the context compete and cooperate to generate recommendations for the user with accompanying rationales. We contend that the legal decision-making process can also be seen as generating recommendations based on precedents, statutes, various supra-legal factors, and so on.

On a more general level, we would like to argue that our experience in building the preliminary system described here suggests that the blackboard architecture is particularly suited for modeling decision making in domains where a multitude of

possible conflicting factors must be considered. We will briefly describe two such situations at the end of the paper.

2. Related research

In designing our system, we incorporated several ideas from past research. The earliest attempt to incorporate different kinds of knowledge sources in a legal expert system was the PROLEXS system [23]. It was an advisory system implemented in the domain of Dutch landlord-tenant laws. Later on, two other systems (GREBE [7] and CABARET [29]) attempted to integrate rules with cases in modeling legal reasoning. In GREBE, the features of any given case (or a precedent) were subdivided into the smallest groups that justified the individual inference steps. This allowed for flexibility in comparing cases for parts of different cases could be combined to justify an inference for a new case. In CABARET, if most preconditions of a rule were found to be applicable to a given case, the system switched to case-based reasoning to generalize the missing preconditions, thereby generalizing the rule based on the precedents.

Anne Gardner carried out a detailed analysis of a real case (a lawsuit over the ownership of land just off Alaska's north coast) to identify four factors that need to be addressed to incorporate human-like reasoning into a computational legal decision--making system: "(1) Combining rules that were adopted for differing purposes but that all have application to the problem at hand; (2) allowing for argument over the logical structure of rules, and managing to reason with them even when unsure what the logical structure is; (3) allowing cases to be used mainly for their facts and outcome, mainly for their reasoning, or mainly for the rules they lay down, and employing each technique when appropriate; and (4) extending the legal sources that are treated as cases." [14]. More recently, there have been a number of studies that explore the psychological aspects of how judges and juries come to their decisions [5, 9, 16, 24, 26, 32]. However, these insights have not been incorporated into a computational system to the best of our knowledge.

For example, one psychological factor is that people are not always aware of which facts are relevant and which ones are not. So, a system that started out by asking the user to input all of the facts and then worked to build an argument based on those facts (and taking into account relevant legal factors) is not going to be very useful if the user did not input some relevant information. This feature was incorporated into one of the early expert systems (Mycin [27]), which was based on backward chaining. In Mycin, if some fact was needed to make an inference and it was not available, then the user was queried for it. Sometimes, this required the user to carry out some new test in order to generate the needed factual data.

This issue is also one of the main motivating factors behind the evolution of iterative and participatory design methodologies, for the user is not usually aware of all of the design requirements at the beginning of a project. So, the current approach is to involve the user interactively as the design evolves [2, 10, 28]. In the same spirit,

our aim is to design an advisory system that constructs legal arguments interactively with the user, occasionally querying them for more information as needed.

We also incorporate the results from past research on Case-Based Reasoning (CBR), which keeps past cases in a knowledge base and uses them as needed. Each case is represented by a list of facts, the judgement, and the rationale for the judgment. Given the facts of the current case, a CBR system looks for analogical cases, analyses their rationales, and tries to come up with similar arguments for the current case. Any differences are adopted to the problem. In the last step, the system evaluates the correctness of the new solution. The correct solutions are saved in the knowledge base so that the system can continuously learn new solutions and use them in the future. Typical research problems facing the design of a CBR system are knowledge-representation techniques, searching and using precedents, testing the adaptation to a problem, and storing new solutions. Descriptions of these problems and suggestions on how to solve them are presented in [1, 4].

Another line of research that is relevant in the design of our system is ontology, which refers to a well-defined set of concepts and relationships that are used to represent some real-world environment [15]. For our system, an appropriate ontology can be used as a basis for CBR to find analogies between cases by comparing concepts [3]; for example, to support case-based comparisons, distinguish deep and shallow analogies, induce/test hypotheses, and so on.

3. Introduction to blackboard architecture

We decided to implement our system to generate multi-pronged arguments using the blackboard architecture [11]. The blackboard architecture was initially proposed in the 1970s for speech-understanding systems [13]. In this architecture, a number of independent agents interact through a shared memory space called the blackboard to arrive at an interpretation (or multiple interpretations) of a given situation. This architecture is often visualized by the metaphor of a group of independent human experts with diverse knowledge sitting in a room with one blackboard. When the initial problem is written on the blackboard, the experts start to add their contributions incrementally while monitoring the blackboard until they find some information that they could use for their contribution. The experts continue to add their contributions to the blackboard until the problem is solved. The blackboard can have multiple levels of abstractions, and an expert can work at the same level or between adjacent levels. Since its initial proposal, the blackboard architecture has been successfully applied to model problem solving in several other domains [8].

There are many advantages of using the blackboard architecture for realizing our vision for an assistive system for legal decision making. The main factor is that it allows for a number of heterogeneous agents: each agent can use a different internal representation and a different processing strategy to interact. The only thing that needs to be homogenized is the interface between an agent and the blackboard. This makes it particularly suitable for incorporating different kinds of supra-legal principles

as well as statutory knowledge, case-based knowledge, world-knowledge, statistical knowledge, psychological knowledge, and so on. The blackboard system also makes it natural to have a combination of top-down and bottom-up processing [5] and allows for multiple constraints to be satisfied in different ways [25].

In our earlier work on modeling creativity in legal reasoning [17, 18], we had also proposed using the blackboard architecture; however, those ideas were not implemented in a working system. We have recently implemented two systems based on the blackboard architecture: a system for generating poetry [21], and a context-aware recommender system [22]. In the research presented here, we would like to use our experience with those systems to build an assistive system for legal decision making.

A classical solution for such a system consists of a knowledge base (e.g., rules) and a single inference system (e.g., forward/backward chaining inference engine). The blackboard-based solution has several advantages. Instead of one centralized inference algorithm, the reasoning is performed by agents that encapsulate their inference engines that operate on the agent-specific internal knowledge representation. Therefore, it is possible to mix various types of reasoning. The reasoning process may also be distributed and parallelized. Such a system is very flexible and can be easily extended. It is enough to add a new agent with its own knowledge representation and reasoning mechanism (e.g., Bayesian). One may also add other types of processing algorithms; for example, it is possible to add agents that apply machine learning. Such an agent could be executed when other agents would not be able to apply their inference engines. Rule induction could add the new rules learned from the experience that allow it to break the impasse (see [30] for a non-blackboard solution).

4. Architecture for generating legal arguments

Our proposed blackboard architecture for legal decision making is shown below in Figure 1.

The information on the blackboard is organized at different levels of abstraction and can be in different formats as long as the appropriate modules can interface with it. At any point in time, several modules may be active, so there is a control component implemented as an independent module that manages the scheduling of the modules. Based on heuristics, it is responsible for selecting the most beneficial module to run at each step of the problem solving.

As the solution in a blackboard system is built incrementally, this architecture is effective for problems that require numerous steps or the use of many diverse sources of knowledge. The blackboard architecture also facilitates searching in a large space of possible solutions, including many paths of reasoning with incomplete or imprecise knowledge. The search strategy may be flexibly adapted to different stages of problem solving, allowing it to work on several levels of abstraction and explore multiple paths of reasoning. We feel that all of these aspects make the blackboard architecture particularly appealing for modeling cognitive and affective factors in legal decision making.

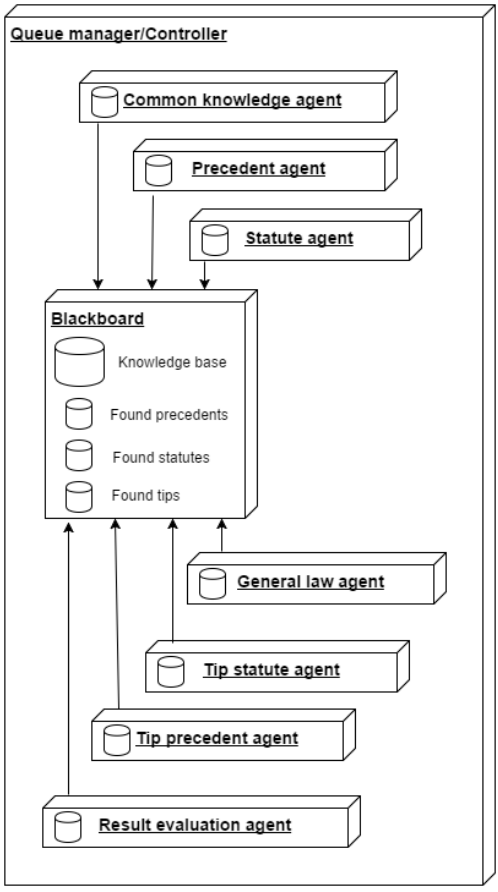

Figure 1. Blackboard architecture for legal decision making

The blackboard stores knowledge about the current case. It is hierarchically organized into multiple abstraction levels. At the bottom level are the facts of the current case, and the top level contains a decision for the given facts of the case in terms of statutory concepts. The middle layers have intermediate concepts that connect the statutory concepts with the facts of the case. The blackboard also stores precedents, legal rules, and tips that the agents found that match the case.

There can be three types of agents: top-down, bottom-up, and intra-level, as all three modes of reasoning occur in legal decision making [5]. Top-down agents work as follows: if, according to the traffic laws, a driver who causes an accident can be found not guilty if the authorities failed to fulfill their obligations, then the top-down expert posts this as an agenda at the next lower level. Other experts will then try to verify or refute it based on the facts of the case. Bottom-up agents work as follows: if someone was injured and hospitalized for more than seven days, then a bottom-up expert will

post the assumption that he/she suffered 'serious' injuries at the next higher level. Intra-level agents connect facts and make inferences at the same hierarchical level. For instance, they can convert speed from mph to km/h.

In our system, we have implemented the following agents:

1. Precedent Agent – applies precedent knowledge: checks whether the precedents are similar to the current case. If each premise of the precedent is the same or can be applied by categorization to the premises of the current case, the precedent is saved to the blackboard.
2. Statute agent – applies statute: checks whether a statute rule can be applied to the current case.
3. General-law Agent – processes law knowledge: currently generalizes law premises and gives them categories to help the statute agent apply the statute rules (independently).
4. Common-knowledge Agent – processes common-sense knowledge: generalizes and specializes categories of the premises not linked with the law.
5. Tip-precedent Agent – suggests facts that should be checked (they impact the inference result according to the precedents). The number of possible missing facts is defined by a special parameter. The user can modify this parameter in the GUI.
6. Tip-statute Agent – suggests facts to check because of the statute rules (in the same way as the tip-precedent agent).
7. Result-evaluation Agent – sorts the results (applied precedents, statutes, and tips) saved on the blackboard in the proper order. If the result is more similar (relevant) to the current case, it is presented higher on the list.

All of the experts are independent and operate on different levels of the blackboard. They have their own knowledge bases and reasoning rules. The precedent, statute, and general-law agents are bottom-up agents. The common-knowledge agent works both in a top-down and bottom-up fashion. We can assign the tip-precedent and tip-statute agents to the group of top-down agents. The result-evaluation agent is an intra-level agent.

The blackboard architecture allows us to extend the system in a simple manner. In the future, additional agents may be implemented by taking statistical knowledge, affective and extra-legal factors, judges preferences, etc. into account.

The queue-manager (controller) module determines the order in which the activated agents are executed. It uses the Drools library to create rules of action for the agents and represent the knowledge base saved on the blackboard. Drools implements the agenda and an algorithm resolving conflicts while ordering agents and giving them access to the blackboard data. We used a simple algorithm to order the agents. Each expert has a priority parameter that determines its priority in the order. If the agent with the highest priority cannot modify the blackboard, the agent with the next-highest priority value gains access to the blackboard, and so on. If an agent

can modify the blackboard knowledge base, it saves the changes in the data on the blackboard, and the algorithm is started from the beginning (Fig. 2).

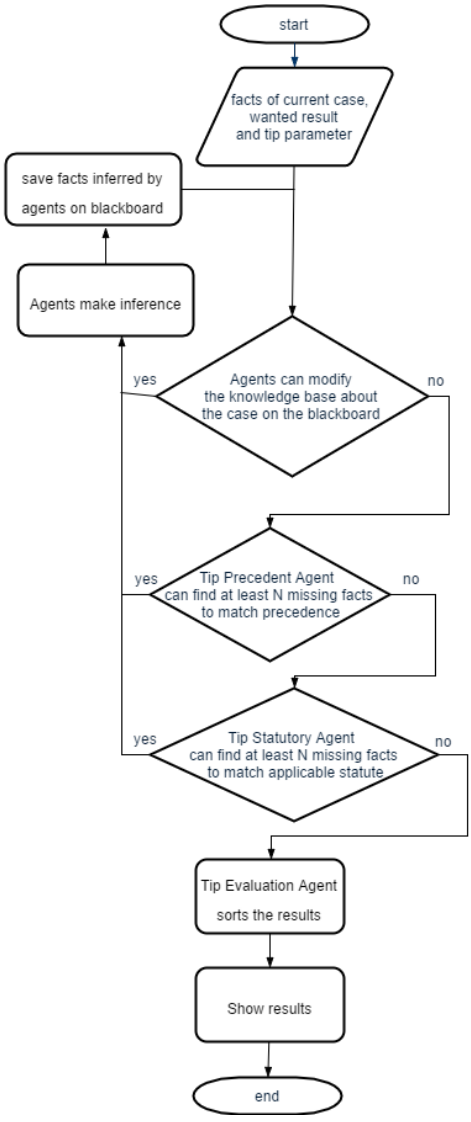

Figure 2. Queue-manager algorithm

Each agent has a rule that defines how its knowledge is used to modify the blackboard. Most agents apply the bottom-up reasoning method (forward reasoning): if the premises of the rules stored in their knowledge are met, they update the blackboard knowledge, adding new inferences from the facts already on the blackboard.

As a result, the argument for a decision is represented as a reasoning path from the facts to the conclusions. Each reasoning step is the result of applying a precedent or statute (mostly on the top level) or a result of the generalization or specialization of given facts.

The Precedent Agent and Statute Agent check to see whether all of the premises of the precedents/statutes were declared in the current case or could be inferred (Alg. 1). If so, they write their conclusions on the blackboard. A conclusion may be inferred by using some precedent or law code.

```
Data: agent's knowledge base, current blackboard state
Result: new conclusions reasoned from premises, precedents/statutes
initialization;
while not all rules from knowledge base checked do
    read current rule;
    if all premises from the rule are applied then
        connect new conclusions with possible premises from the rule (for
          argumentation) save new conclusions on the blackboard;
        add the rule to results (found precedents/statutes);
    end
end
```

Algorithm 1. Algorithm of Precedent Agent and Statute Agent

The generalization and specialization operators are performed by the Common-knowledge and General-law Agents (Alg. 2). For example, they can generalize the category 'you hit a car' with the category 'you hit a vehicle.' These operators are also defined in the form of rules.

```
Data: agent's knowledge base, current blackboard state
Result: new conclusions reasoned from premises
initialization;
while not all rules from knowledge base checked do
    read current rule;
    if all premises from the rule are applied then
        connect new conclusions with possible premises from the rule (for
          argumentation) save new conclusions on the blackboard;
    end
end
```

Algorithm 2. Algorithm of Common-knowledge Agent and General-law Agent

The Tip-precedent and Tip-statute Agents use the top-down method to check if, at most, tipEpsilon (a number defined by the user) of the facts are missing to declare an analogy between the current case and a precedent/statute (Alg. 3). If so, they write these precedents/statutes in the form of tips on the blackboard. Precedents and statutes are stored in the knowledge base of these agents in the form of cases. The tips are presented to the user, where missing facts are marked in red. A mix of the

top-down and bottom-up approaches allows us to ask the user for more information that may be relevant for the desired decision.

Data: tipEpsilon, Agent's knowledge base, current blackboard state
Result: list of tips
initialization;
while *not all rules from knowledge base checked* **do**
  read current rule;
  if *all except [1..tipEpsilon] premises from the rule are applied* **then**
    add the rule to results (found tips);
  end
end

Algorithm 3. Algorithm of Tip-precedent and Tip-statute Agents

Moreover, the Result-evaluation Agent sorts the applied precedents, statutes, and tips (Alg. 4). It is helpful when the user has many possible arguments or searches for relevant tips. There is a simple rank algorithm that works as follows: the more category changes are needed to apply the result to the current case, the greater the cost value of the result; it is presented to the user lower on the list. When we evaluate the tips, we also take into consideration the constant cost of adding a new fact to the knowledge base of the current case.

Data: current blackboard state
Result: sorted result list on the blackboard
initialization;
rank all results from the blackboard;
sort results separately (precedents/statutes/tips)

Algorithm 4. Algorithm of Result-evaluation Agent

A simplified class diagram of the system is presented in Figure 3. When the system starts, the agents and blackboard are created and initialized. The agents load the knowledge base from their JSON files. CurrentCaseLoader transforms the user input data (facts of the case) into LegalFormulas and saves it on the blackboard. When everything is ready, QueueManager sets the order of agents in the queue to modify the blackboard. Agents operate on LegalFormulas and LegalFacts, writing new conclusions on the blackboard. If all of the agents cannot modify the blackboard, the system prints the results to the user. The package utils contains helper classes that serialize and deserialize the JSON files.

We decided to represent the knowledge bases with the model described in Figure 4. There is a basic data type processed by agents (called LegalFormula) that is used to represent a legal rule. Each agent has a list of LegalFormula objects. Each rule contains the following:

- premises – a list of facts that should be applied to prove and use the conclusions,
- conclusions – a list of facts that can be added to the knowledge base (blackboard) when all of the premises of LegalFormula are proven.

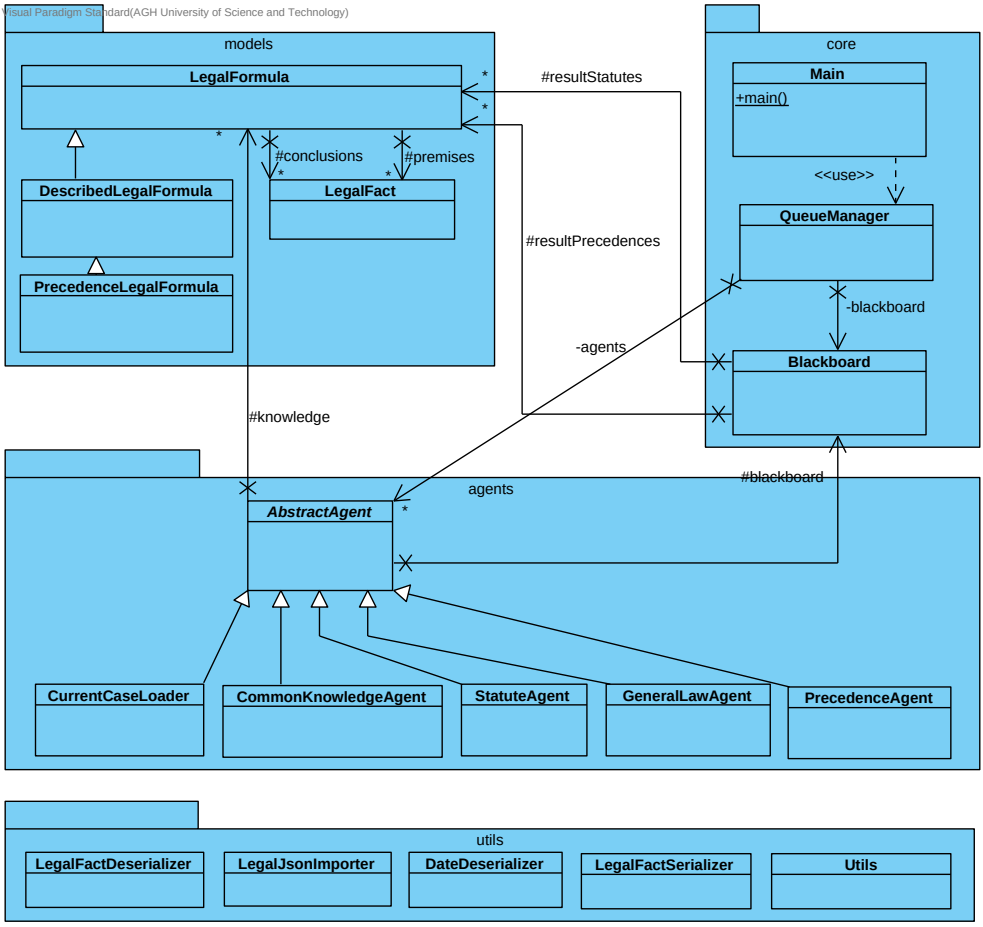

Figure 3. Class diagram of system

Each fact (LegalFact) has a name (description of the scenario element), a status (true if the fact happened, false otherwise), and premises representing a list of arguments that can be used to infer this fact. LegalFormula is used to represent the common-knowledge base and general-law knowledge base. No additional information is needed on this level (such as the signature of the case or the date of the judgement in the court). To represent the statutes, we use DescribedLegalFormula (which has a signature). A signature helps to identify the law rule. To store the precedents, we use PrecedentLegalFormula; it is extended with a signature and the date of the judgement in the court, which makes it possible to find the precedent (for example, on the Internet).

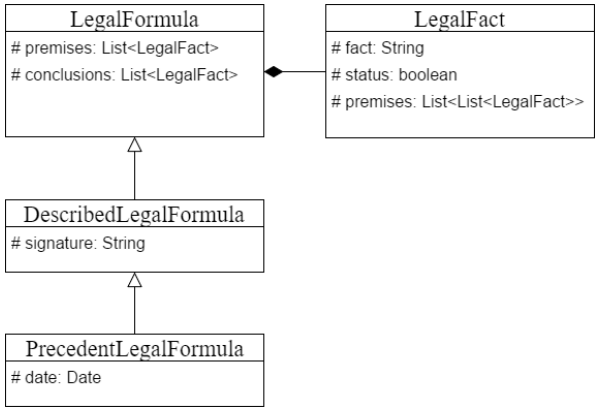

Figure 4. Class diagram of knowledge-base model

The system is written in Java language and consists of 18 core classes (without GUI). The inference engine is based on the Drools library (https://www.drools.org/). The user interface uses the Swing widget toolkit.

5. Test runs

The goal of our test runs was to verify where our system based on a blackboard architecture can help users to find arguments for the desired outcomes and find missing information that would allow them to extend the range of their arguments. For the test runs, we prepared a knowledge base with the following:

- 30 precedents (5 based on press news and our experiences with traffic law, 25 based on a page with Polish court rulings `http://orzeczenia.ms.gov.pl/`),
- 10 statutes (based on Polish Code),
- 25 generalizations/specializations (6 based on common knowledge, and 19 based on general-law knowledge).

The agent knowledge base is stored in JSON files. Examples of Legal Formulas are presented in Figure 5. Each agent has a separate file with its knowledge. In each file is a list of rules. Each rule contains at least one premise and at least one conclusion. When the system starts, the agents load data from concrete files with a knowledge base prepared for each of them. Afterwards, they use their knowledge to modify the blackboard.

Below, we present two scenarios that demonstrate the main features of our system. The system was tested in the domain of Polish traffic law. In our experiments, we did not filter the results by the desired outcome to show the many possible arguments with different judgements. The exclamation mark before a fact name indicates its negation.

```
1. Common knowledge
{
    "conclusions": [{"fact": "you_hit_a_vehicle"}],
    "premises": [{"fact": "you_hit_a_car"}]
}

2. Statute nowledge
{
    "signature" : "art. 156 § 1 k.k.",
    "conclusions": [{"fact": "imprisoned_[12..120]_months"}],
    "premises": [{"fact": "heavy_damage_on_health"}, {"fact": "intentional"}]
}

3. Precedent knowledge
{
    "signature" : "Signature 4",
    "date": "07.01.2016",
    "conclusions": [{"fact": "innocent"}],
    "premises": [{"fact": "!you_give_a_way"}, {"fact":
    "accident_at_the_crossroads"},
        {"fact": "you_hit_a_car"}, {"fact": "!there_were_casualties"},
        {"fact": "there_were_injuries"}, {"fact": "!you_were_drunk"},
        {"fact": "!authorities_fulfill_their_obligations"}]
}
```

Figure 5. Examples of knowledge base representation in JSON files

Scenario 1

Assume that the defendant failed to comply with the traffic signs and did not cede the right of way to another vehicle. The event took place at a crossroad, when the defendant improperly changed lanes. His/her vehicle inadvertently hit another car. As a result, the driver of the other car suffered permanent injuries and died in the hospital. The defendant ran away from the accident place. He/she had no previous convictions.

For this example, the system found four similar precedents (Fig. 6). The selected case appears to be the most similar to the current case. Several agents cooperated to generate the results. The Precedent Agent checked whether all of the facts of the precedents were declared in the current case or could be deduced. Among others, it could find a match with Case 'VI Ka 132/15'. The General-law Agent applied the rule that, if someone had 'extensive damage to their health,' it implies that someone was 'injured for more than seven days.' Even though the defendant hit a car, the Common-knowledge Agent inferred the fact 'you hit a pedestrian' using the generalization ('you hit a car' → 'you hit a vehicle' → 'you hit another road user') and the specialization ('you hit another road user' → 'you hit a pedestrian'). Although there is a difference between the facts 'you hit a car' and 'you hit a pedestrian,' they can be generalized to the same category, as this difference may not be relevant at all times. The entire argumentation chain is presented to the user so he/she can decide whether this argument seems reasonable and can be used in court. Based on the statutes, the Statute Agent inferred that the defendant can possibly face up to eight years in prison (Fig. 7).

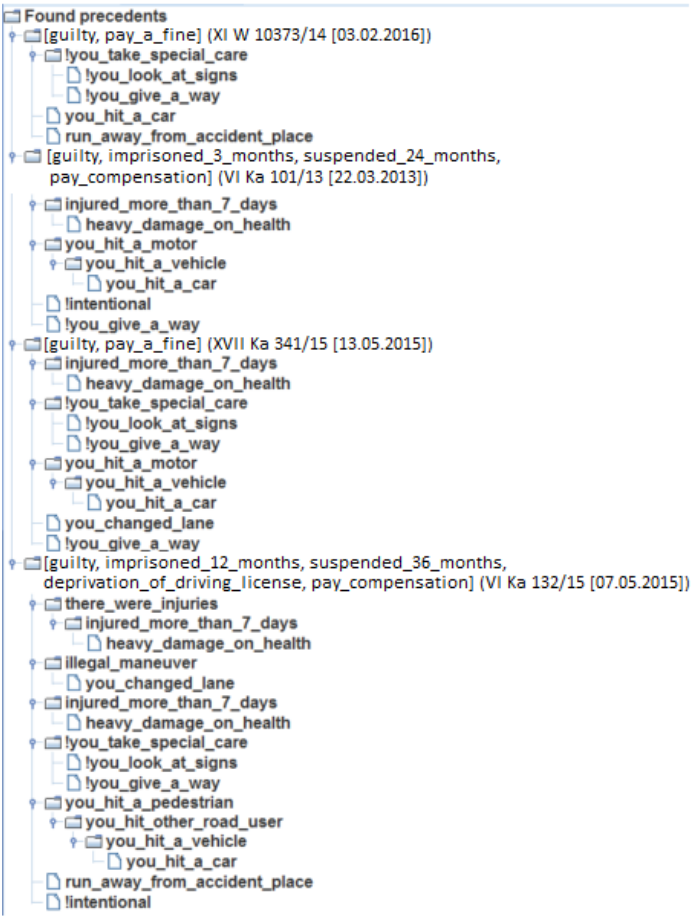

Figure 6. Precedents found in Scenario 1

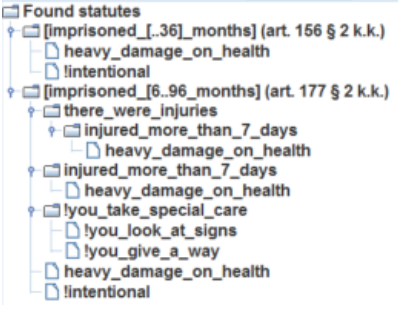

Figure 7. Statutes found in Scenario 1

Some arguments have several levels of nesting, which may indicate the lack of precise mapping for the precedents or statues to the current case. When a tree has several levels, more operations are needed to match the case.

The tip agents were able to find a few tips (Fig. 8). From the defendant's point of view, tips with an acquittal judgment are especially interesting. One precedent in the list has such a judgement (Tip 5).

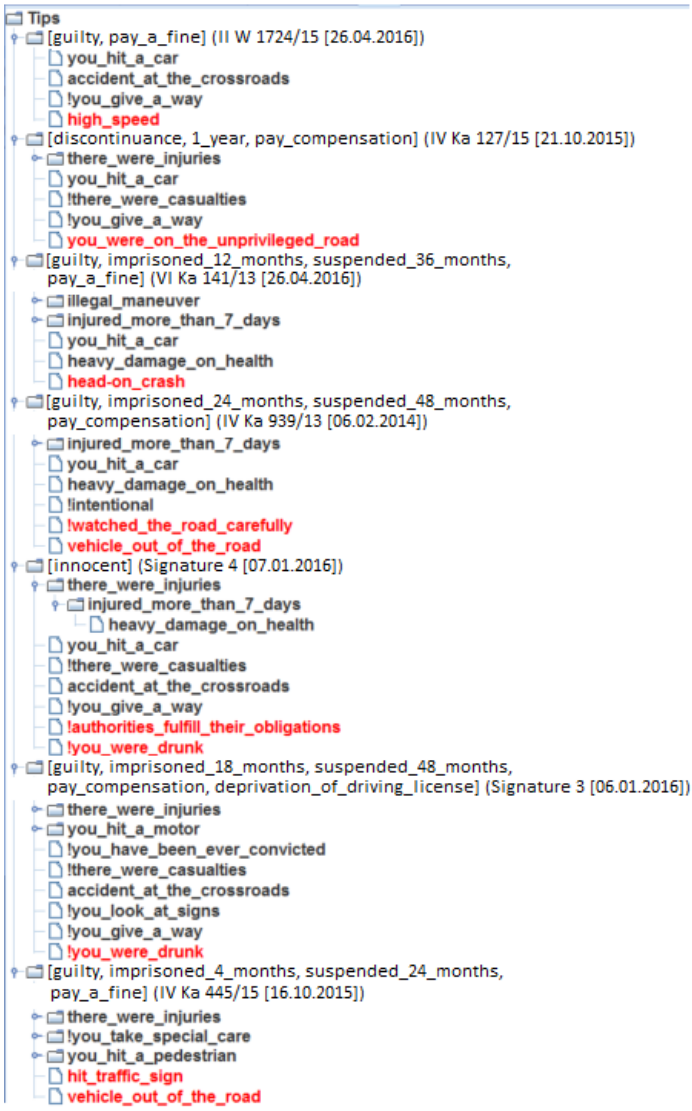

Figure 8. Tips found in Scenario 1

To be innocent, the defendant needs to declare, additionally, that he/she was sober and the authorities failed to fulfill their obligations; e.g., they did not replace a road sign that indicated who had the right of way (the sign was previously destroyed by an act of vandalism).

The tips can be used by interactively the user. The missing facts suggested by the tip agents can be added to the current case description, and the reasoning can be executed once more possibly leading to a different judgement.

Scenario 2

In this scenario, the defendant failed to comply with the traffic signs and did not cede the right of way to another vehicle. The defendant was sober. His/her vehicle hit another car because he/she (intentionally) ignored the right-of-way scheme. As a result of the accident, the driver of the car that was hit suffered permanent injuries and had to be hospitalized for more than seven days. The defendant ran away from the scene of the accident. When the police stopped him/her, he/she did not show the necessary documents (driver's license, etc.).

The system found one similar precedent and displayed its rationale for the decision (Fig. 9). The facts 'ran away from the scene of the accident' and 'you hit a car' were also declared in the current case. The system can infer the fact 'you did not take special care' from the facts 'you ignored the signs' and 'you did not cede the right of way.' However, this precedent does not include some important facts, like if somebody was injured.

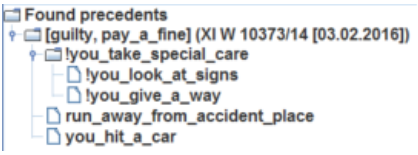

Figure 9. Precedents found in Scenario 2

In this case, the Statute Agent found a rule that fits better. The defendant may be imprisoned for one to ten years (Fig. 10).

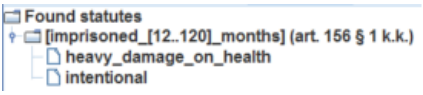

Figure 10. Statutes found in Scenario 2

When we look at the tips (Fig. 11), we can predict that the defendant could be deprived of his/her driver's license if he/she had previous convictions for similar offenses. There is another precedent, which differs in the fact that the accident

took place at a crossroad. If the defendant caused somebody's death, he/she could be imprisoned for 2–12 years. Paying a fine seems to be the minimal punishment. The details of the precedent should be analyzed manually in the case description (a reference number is provided by the system).

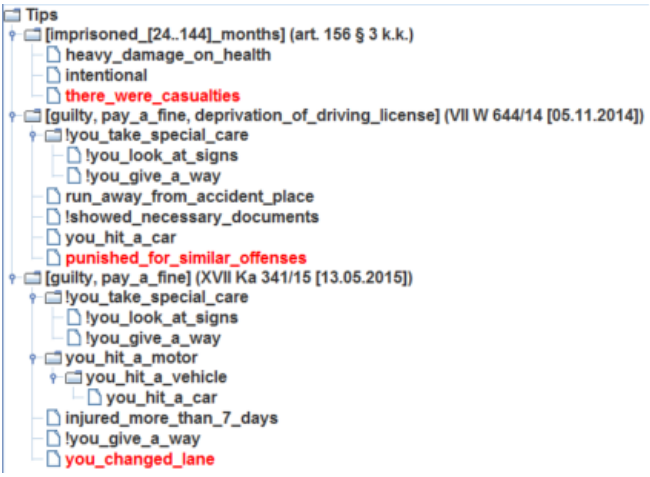

Figure 11. Tips found in Scenario 2

6. Conclusions and further research

We have proposed a blackboard architecture that integrates rule-based reasoning with case-based reasoning. Our system constructs arguments that support a given decision. In constructing an argument, it prompts the user for additional facts that the user may not have initially considered relevant. The approach combines top-down control (which starts from the desired decisions and looks for supporting facts) and bottom-up control (which makes inferences from the available facts).

Though the concepts on which our system is based (namely, the blackboard architecture, combining top-down and bottom-up controls and combining rule-based and case-based approaches) were all applied to legal reasoning in the 1980s and 90s, we are now redeploying these ideas in the context of the current technology to create a practical usable system that scales up to a large amount of online legal data. This paper presents our initial step in this direction.

In our future research, we plan to create a module to look for new precedents and laws on the Internet and transform the text into a format that can be integrated in a knowledge base for agents. In this way, the knowledge base with precedents, common knowledge, and law knowledge would be continuously updated. As the law changes and more cases are decided, the knowledge base would automatically keep pace with the current developments. We would also like to extend the knowledge base of the system to other domains besides traffic law.

Another interesting improvement is to create an agent that would learn which arguments are the most convincing for each judge and try to use them in generating an effective argument for a particular court or judge. This would be helpful because, in addition to the applicable law, judges' decisions are often influenced by psychological aspects, prejudices, personal experiences, social standards, political events, common sense, and moral principles.

Finally, we would like to submit that the architecture proposed here is effective for other domains that feature a multitude of conflicting factors that affect the decision-making process. For example, as autonomous systems are becoming more and more pervasive, they often must make decisions concerning moral and ethical values. However, in order for the decision making to seem persuasive to humans, it needs to reflect human values and judgments. In our earlier work [20], we proposed a blackboard architecture to implement a moral decision-making system that generates rationales that are persuasive to humans. Our vision is that such a system can be used as an advisory system to consider a situation from different moral perspectives and generate ethical pros and cons of taking a particular course of action in a given context.

Another domain is that of modeling mobility behavior in a smart city. We are working on developing a multi-agent architecture for modeling, operationalizing, and quantifying the well-being value of mobility. These well-being factors are based on eudaimonic principles and are, thus, subtended within an ethical framing. This is done to best understand how an individual's ethical framework affects his/her decisions and how these decisions relate to the individual's well-being (and to the well-being of society as a whole) [12].

References

[1] Aamodt A., Plaza E.: Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches, *IOS Press*, vol. 7(1), pp. 39–59, 1994.

[2] Arcia A., Suero-Tejeda N., Bales M.E., Merrill J.A., Yoon S., Woollen J., Bakken S.: Sometimes more is more: iterative participatory design of infographics for engagement of community members with varying levels of health literacy, *Journal of the American Medical Informatics Association*, vol. 23(1), pp. 174–183, 2016. https://doi.org/10.1093/jamia/ocv079.

[3] Ashley K.D.: *An Approach to Legal Ontologies: the Case-Based Reasoning Perspective*, Jurix 2008 Workshop on Approaches to Legal Ontologies, ITTIG-CNR, EUI, Florence, Italy December, 2008. http://www.lrdc.pitt.edu/Ashley/OntologiesTalk2Final.pdf.

[4] Atkinson K., Bench-Capon T.: Legal case-based reasoning as practical reasoning, *Artificial Intelligence and Law*, vol. 13(1), pp. 93–131, 2005. https://doi.org/10.1007/s10506-006-9003-3.

[5] Bartels B.L.: Top-Down and Bottom-Up Models of Judicial Reasoning. In: Klein D., Mitchell G. (eds.), *The Psychology of Judicial Decision Making*, Oxford University Press, pp. 41–55, 2010. https://doi.org/10.1093/acprof:oso/9780195367584.003.0003.

[6] Bench-Capon T., Prakken H., Sartor G.: Argumentation in Legal Reasoning. In: Simari G., Rahwan I. (eds.), *Argumentation in Artificial Intelligence*, Springer, Boston, MA 2009, pp. 363–382. https://doi.org/10.1007/978-0-387-98197-018.

[7] Branting L.K.: Building explanations from rules and structured cases, *International Journal of Man-Machine Studies*, vol. 34(6), pp. 797–837, 1991. https://doi.org/10.1016/0020-7373(91)90012-V.

[8] Carver N., Lesser V.: *The Evolution of Blackboard Control Architectures*. Technical Report, University of Massachusetts Amherst, MA, USA, 1992.

[9] Cohen J.: *Blindfolds Off: Judges on How They Decide*. American Bar Association, 2014.

[10] Cooper A., Reimann R., Cronin D., Noessel C.: *About Face: The Essentials of Interaction Design*. Wiley Publishing, 4th ed., 2014.

[11] Corkill D.D.: Blackboard Systems, *AI Expert*, vol. 6(9), pp. 40–47, 1991.

[12] De Waal A.W., Indurkhya B., Vanderschuren M.: Re-interpreting motility constructs within a Mobility as a Service framing, the case of eudaimonic and hedonic well-being. In: *Proceedings of the T2M Conference, Montreal, 24–27 October*. 2018.

[13] Erman L.D., Hayes-Roth F., Lesser V.R., Reddy D.R.: The Hearsay-II Speech-Understanding System: Integrating Knowledge to Resolve Uncertainty, *ACM Computing Surveys (CSUR)*, vol. 12(2), pp. 213–253, 1980. https://doi.org/10.1145/356810.356816

[14] v.d.L. Gardner A.: Rules and Cases for Legal Reasoning: Notes on Some Neglected Aspects. In: *AAAI Technical Report SS-98-04*, pp. 64–68, 1998. https://www.aaai.org/Papers/Symposia/Spring/1998/SS-98-04/SS98-04-012.pdf.

[15] Guarino N.: Formal ontology and information systems. In: *Formal Ontology in Information Systems: Proceedings of the First International Conference (FOIS'98), June 6-8, Trento, Italy*, pp. 3–15. 1998.

[16] Healy T.: *The Great Dissent: How Oliver Wendell Holmes Changed His Mind and Changed the History of Free Speech in America*. Metropolitan Books, New York, 2013.

[17] Indurkhya B.: Using set-of-support control strategy to deal with indeterminacy in legal reasoning. In: van Kralingen R.W., van den Herik H.J., Prins J.E.J., Sergot M., Zeleznikow J. (eds.), *Legal Knowledge Based Systems (Jurix'96)*, pp. 123–133, Tilburg University Press, The Netherlands, 1996.

[18] Indurkhya B.: On modeling creativity in legal reasoning. In: *ICAIL'97 Proceedings of the 6th international conference on artificial intelligence and law, Melbourne, Australia, June 30 – July 03, 1997*, pp. 180–189, 1997. https://doi.org/10.1145/261618.261651.

[19] Indurkhya B., Misztal-Radecka J.: On Modeling Cognitive and Affective Factors in Legal Decision-Making. In: Rotolo A. (ed.), *Legal knowledge and information systems. JURIX'2015: the twenty-eight annual conference*, pp. 157–160, IOS Press, Amsterdam, 2015.

[20] Indurkhya B., Misztal-Radecka J.: Incorporating Human Dimension in Autonomous Decision-Making on Moral and Ethical Issues. In: *Proceedings of the AAAI Spring Symposium on Ethical and Moral Considerations in Non-Human Agents*. Palo Alto: California, pp. 226–230, 2016.

[21] Misztal J., Indurkhya B.: A Blackboard System for Generating Poetry, *Computer Science*, vol. 17(2), pp. 265–294, 2016.

[22] Misztal J., Indurkhya B.: Explaining Contextual Recommendations: Interaction Design Study and Prototype Implementation. In: O'Donovan J., Felfernig A., Tintarev N., Brusilovsky P., Semeraro G., Lops P. (eds.), *Proceedings of the Joint Workshop on Interfaces and Human Decision Making for Recommender Systems, IntRS 2015, co-located with ACM Conference on Recommender Systems (RecSys 2015), Vienna, Austria, September 19, 2015*. pp. 13–20, 2015.

[23] Oskamp A., Walker R.F., Schrickx J.A., van den Berg P.H.: PROLEXS divide and rule: a legal application. In: *ICAIL '89. Proceedings of the 2nd international conference on artificial intelligence and law*, pp. 54–62. ACM, New York, NY, USA, 1989. https://doi.org/10.1145/74014.74022.

[24] Posner R.A.: *How Judges Think*. Harvard University Press, Cambridge (MA), 2010.

[25] Robbennlot J.K., MacCoun R.J., Darley J.M.: Multiple constraint satisfaction in judging. In: Klein D.E., Mitchell G. (eds.), *The Psychology of Judicial Decision Making*. Oxford University Press, 2010.

[26] Schweizer M.D.: *Kognitive Täuschungen vor Gericht: eine empirische Studie*, Dissertation der Rechtswissenschaftlichen Fakultät der Universität Zürich, 2005.

[27] Shortliffe E.: *Computer-based medical consultations: MYCIN*, Elsevier 1976.

[28] Simonsen J., Hertzum M.: Sustained Participatory Design: Extending the Iterative Approach, *Design Issues*, vol. 28(3), pp. 10–21, 2012. https://doi.org/10.1162/DESIa00158.

[29] Skalak D.B., Rissland E.L.: Arguments and cases: An inevitable intertwining, *Artificial Intelligence and Law*, vol. 1(1), pp. 3–44, 1992.

[30] Śnieżyński B., Legień G., Wilk-Kołodziejczyk D., Kluska-Nawarecka S., Nawarecki E., Jaśkowiec K.: Creative Expert System: Comparison of Proof Searching Strategies. In: Nguyen N.T., Tojo S., Nguyen L.M., Trawinski B. (eds.), *Intelligent Information and Database Systems*, pp. 400–409. Springer International Publishing, Cham, 2017.

[31] Twining W., Miers D.: *How To Do Things With Rules: A Primer of Interpretation* (2nd ed.). Weidenfeld and Nicolson, London, 1982.

[32] Vidmar N.: The psychology of Trial Judging, *Current Directions in Psychological Science*, vol. 20(1), pp. 58–62, 2011. https://doi.org/10.1177%2F0963721410397283.

[33] Wisdom J.: Gods, *Proceedings of the Aristotelian Society*, vol. 45, pp. 185–206, 1944.

Affiliations

Łukasz Szymański
AGH University of Science and Technology, Krakow, Poland, lukasz.szymansk@gmail.com

Bartłomiej Śnieżyński
AGH University of Science and Technology, Faculty of Computer Science, Electronics and Telecommunications, al. A. Mickiewicza 30, 30-059 Krakow, Poland, Bartlomiej.Sniezynski@agh.edu.pl, ORCID ID: https://orcid.org/0000-0002-4206-9052

Bipin Indurkhya
AGH University of Science and Technology, Krakow, Poland, bipin@agh.edu.pl, ORCID ID: https://orcid.org/0000-0002-3798-9209

Received: 07.08.2018
Revised: 31.10.2018
Accepted: 31.10.2018