Data & Policy (2024), 6: e8
doi:10.1017/dap.2023.49

COMMENTARY

On the use of smart hybrid contracts to provide flexibility in algorithmic governance

Carlos Molina-Jimenez[1] and Sandra Milena Felizia[2]

[1]Department of Computer Science and Technology, University of Cambridge, Cambridge, United Kingdom
[2]Facultad de Ciencias Económicas, Universidad Abierta Interamericana, Rosario, Santa Fe, Argentina
Corresponding author: Carlos Molina-Jimenez; Email: carlos.molina@cl.cam.ac.uk

Received: 26 May 2022; **Revised:** 17 July 2023; **Accepted:** 28 November 2023

Keywords: algorithmic governance; blockchains; computational law; hybrid contracts; law automation; monopolistic competition; preventive law; smart contracts

Abbreviations: FSM, finite state machine; HC, hybrid contract

Abstract

The use of computer technology to automate the enforcement of law is a promising alternative to simplify bureaucratic procedures. However, careless automation might result in an inflexible and dehumanized law enforcement system driven by algorithms that do not account for the particularities of individuals or minorities. In this article, we argue that hybrid smart contracts deployed to monitor rather than blindly enforce regulations can be used to add flexibility. Enforcement is a suitable alternative only when prevention is strictly necessary; however, we argue that in many situations a corrective approach based on monitoring is more flexible and suitable. To add more flexibility, the hybrid smart contract can be programmed to stop to request the intervention of a human or of a group of them when human judgment is needed.

Policy Significance Statement

The article assumes that algorithmic governance will be gradually adopted by governments, which implies that we are heading to a society where the law is enforced automatically by computer-executable programs called smart contracts (digital contracts, programmable contracts, etc.). The authors argue that smart contracts are inflexible, likely to suffer from gaps and, more importantly, lack human judgment. Therefore, there is a risk of creating a dehumanized law enforcement system driven by algorithms that do not account for the particularities of individuals or minorities. To address the problem, they suggest that smart contracts should work in tandem with humans to be involved in situations where human sense is needed.

1. Introduction

The strong relationship between logic and law has been acknowledged since ancient times and the subject of interest for decades (Lacock, 1964). It is widely acknowledged that regulations, at least partially, can be modeled by logical statements like *if event_occurs and condition_holds then execute_action* that can be

This research article was awarded Open Data and Open Materials badges for transparent practices. See the Data Availability Statement for details.

https://doi.org/10.1017/dap.2023.49 Published online by Cambridge University Press

expressed as computer code that can be executed mechanically. For example, *if transaction executed and amount larger than 10000 then report to government.*

Recent progress in computer technology has generated excitement about the possibility of building systems that enable law automation. This is a recently emerging concept referred to by different terms. In Filippi and Wright (2018), it is called Lex Cryptographia; in Law.MIT.edu (2021), Law (2005), and Genesereth (2021), it is called Computational Law; in Hazard and Haapio (2017) and Micheler and Whaley (2020), it is called Replacement of Law with Computer Code; and in Surden (2012), it is called Computable Contracts. It can fairly be called Programmable Law because it is a law implemented by computer programs or Algorithmic Governance (Werbach, 2020; Gamito and Ebers, 2021; Gasser and Almeida, 2022) to emphasize that law enforcement will follow algorithms, that is, strict mathematical procedures.

The general idea behind all these terminologies is to use computer code to automate the enforcement of regulations in different fields of our society, ranging from regulations within private companies to governments. This would require translating laws (civil code, codes covering corporate law, administrative law, tax law, constitutional law, etc.) which are currently written in natural language for human interpretation, into a code that computers can read, interpret, and execute automatically. In addition, this code needs to be protected against accidental and malicious threats. Some authors use the term smart contract to refer to this and similar code that can be used for the automation of regulations.

We can define a smart contract (also known as digital contract, executable contract, and automatic contract) as a piece of executable computer code that a software engineer implements from the translation of normative statements written in natural language into a computer language such as Python, Solidity, Go, and so on.

Current governments are infamous for their cumbersome and unnecessarily slow bureaucratic procedures; for example, it takes months or even years for courts to dictate a sentence and involve scores of printed documents. Fortunately, algorithmic governance promises to simplify and speed up bureaucratic procedures. More importantly, the adoption of computer technology by the legal system opens opportunities to ameliorate the drawbacks that afflict current political systems, such as minority exclusion (Emerson, 2018). However, algorithmic governance raises new challenges (Felizia et al., 2022a). In our opinion, one of the most important challenges is the adoption of automatic preventive laws, as we explained in Section 5.

In this work, we raise the question about the lack of flexibility that algorithmic governance can potentially introduce. There are concerns that smart contracts are inflexible (Sklaroff, 2018) software mechanisms, consequently, their use in the implementation of computational law would compromise the flexibility of the current law which is based on the intervention of humans to provide human judgment. To ameliorate the problem, we suggest the use of incomplete hybrid smart contracts that can be deployed to monitor and enforce as necessary, rather than only to mechanically enforce regulations. In addition, we suggest that in borderline situations where human judgment is needed, the incomplete hybrid smart contract stops requesting human intervention. The response can be produced by a single individual or by a group of them after reaching a consensus.

Human judgment is needed where the decision to be taken would have an irreversible effect on an individual or society. At the top of the list, we would place situations that have been documented to be challenging to handle with computer technology. For example, it has been widely documented that algorithm-based image recognition is not reliable; therefore, it is too risky to use medical images in life-threatening surgeries without human examination for final approval. This and other examples of situations where computers fall short and therefore need human help are discussed in Choi (2021).

The rest of this article is developed as follows: In Section 2, we introduce concepts related to normative statements, including contracts, rights, obligations, and prohibitions. In Section 3, we explain how an automatic contract can be deployed for monitoring and enforcement. In Section 4, we explain the solution that we suggest for providing flexibility in algorithmic governance. In Section 5, we discuss preventive law as a future research topic. We suggest that automatic preventive law can be used as a measure to prevent the execution of criminal acts, as opposed to criminal punishment. For example, it can be used to

https://doi.org/10.1017/dap.2023.49 Published online by Cambridge University Press

deter monopolistic practices. In Section 6, we close the discussion with some remarks that reflect lawyers' concerns about the invasion of computer technology of a field that has been for centuries lawyers' exclusive domain.

2. Contracts, rights, obligations, prohibitions, and operations

From a technical perspective, a contract is conceived as a set of clauses that stipulate rights, obligations, and prohibitions that the signatories are expected to comply with rights, obligations, and prohibitions are associated with at least one operation. An operation is a business action executed by a party that changes the state of the contract development, for example, paying bills, delivering items, etc.

In simple contracts, each right, obligation, and prohibition is associated with a single operation. In these situations, one can regard a right as an operation that a party is entitled to execute. Likewise, an obligation can be regarded as an operation that a party is expected to execute. Finally, a prohibition can be regarded as an operation that a party is not expected to execute. As an example, we can think of a contract where Bob has the obligation to pay Alice 100.00 under certain conditions (by December 31, 2020). To honor this obligation, Bob needs to successfully execute through some mechanism the corresponding operation "pay 100.00 to Alice" by the deadline. In practice, contracts include several obligations. For example, a buyer has the obligation to pay and a seller the obligation to deliver or the obligation to refund. Therefore, to comply with the whole contract, the signatory parties need to honor each obligation by means of executing the corresponding operation—pay, deliver, and refund in this example. The motivation for using digital contracts is that they automate the execution of operations in compliance with the rights, obligations, and prohibitions stipulated in the contract. Automatic execution frees the signatory parties from the hassle of performing them manually to honor the corresponding obligations.

To understand the enforcement of a whole digital contract, it helps to regard the execution of a digital contract as the execution of several interrelated operations where the execution of one of them exercises a right, honors an obligation, or violates a prohibition and might enable or disable other rights, obligations and prohibitions. Some authors regard each right, obligation, and prohibition as an individual contract. In their model, a contract is composed of one or more interrelated subcontracts.

3. Contract execution

In automatic contracts, operations are executed through the execution of the executable code that implements them. Figure 1 shows the execution of a payment operation. Pay 100 is assumed to be stipulated in a contract agreed upon between Bob (the payer) and Alice (the payee). Bob's application is installed on his laptop and Alice's is installed on her mobile phone. The executable code that implements the pay operation is assumed to be implemented in a programming language and deployed on a computer, a local one, in a cloud server or on a blockchain. To illustrate the architecture that we suggest in this article for providing flexibility (Figure 4), we will assume that the contract is deployed on a blockchain, for example, on the Ethereum blockchain (Ethereum Foundation, 2018).

In principle, smart contracts can be executed in centralized systems (e.g., Carta, https://carta.com/) following the traditional client-server model which is far simpler and better understood than the other alternative—the use of decentralized systems like blockchains. We acknowledge that blockchains are far

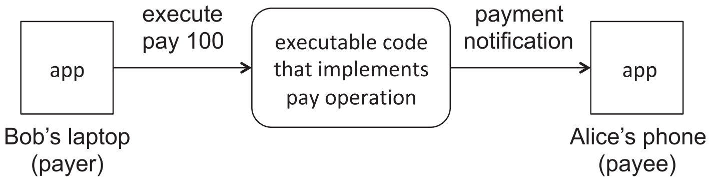

Figure 1. *Execution of pay operation without the involvement of a smart contract.*

https://doi.org/10.1017/dap.2023.49 Published online by Cambridge University Press

more complex and still under test. However, we argue that in some situations, the advantages that decentralized systems bring, outweigh complexity. For example, in some government applications (say, budget expenditure), transparency, traceability, and indelible records are essential requirements. These properties are naturally provided by blockchains and are difficult to implement in centralized systems.

Returning to the example of Figure 1, to pay Alice, Bob's application issues the operation "pay 100" against the executable code. As a response, the executable code executes the operation and as a result, Alice's application receives a notification of payment, for example, bank evidence of the payment. Notice that Figure 1 shows only the execution of the pay operation. There are no digital mechanisms in place to detect Bob's failure to execute the pay operation or to enforce him to execute it automatically. Automatic enforcement can be achieved with the help of a digital contract. From the perspective of the level of interference that the digital contract causes in the execution of the contractual operations, digital contracts can be deployed to either monitor or enforce. The two alternatives are shown, respectively, in Figures 2 and 3. The advantage of automatic enforcement and monitoring is fundamental in law automation. Unfortunately, existing literature focuses only on enforcement and fails to appreciate the advantages of monitoring. See, for example, Lex Cryptographia (Filippi and Wright, 2018).

3.1. *Contract monitoring*

Contract monitoring is a technique where a smart contract is deployed to observe the development of the action passively and to store records of the operations executed by the signatory parties. Monitoring is passive in the sense that the smart contract does not interfere with the development of the action; it only observes and keeps records for potential postmortem examination, for example, if a dispute is raised.

Figure 2 shows how a smart contract can be deployed for monitoring the execution of a payment operation. Notice that the smart contract is directly interrelated with the executable code that implements the payment operation. In fact, in existing literature, the two components are frequently discussed as a single one. We separate them to help understand how smart contracts work.

1. Bob's application places the operation "execute pay 100" against the executable code.
2. The executable code executes the operation and as a result, Alice's application receives "payment notification", for example, a bank receipt.
3. The executable code provides the smart contract that is responsible for monitoring with records of the execution of the pay 100 operation placed by Bob.

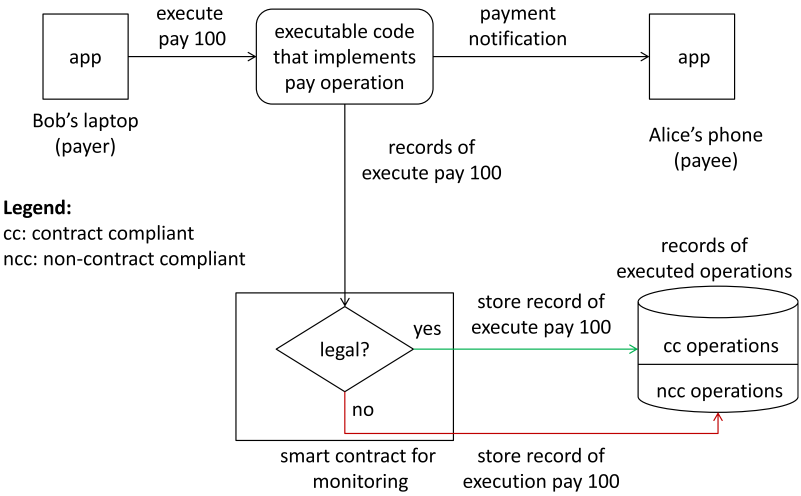

Figure 2. *Monitoring of the execution of a pay operation.*

https://doi.org/10.1017/dap.2023.49 Published online by Cambridge University Press

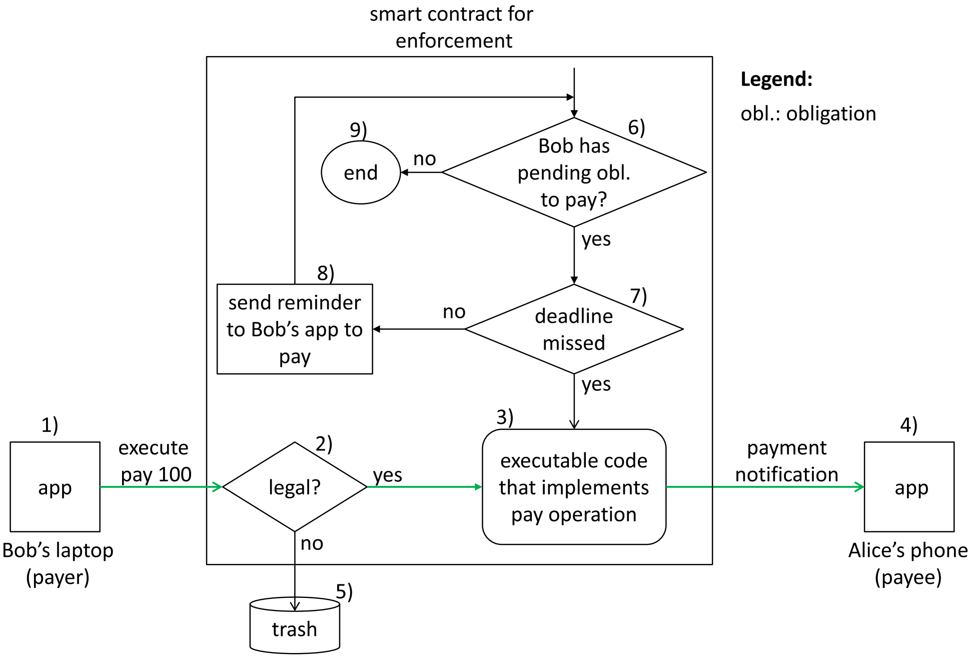

Figure 3. *Enforcement of the execution of a pay operation.*

4. The monitoring contract analyses the records, determines if "pay 100" operation is contract compliant (legal) or non-contract compliant and sends its verdict (red and green lines, respectively) to the database. The records accumulated in the database can be used for conducting postmortem (offline) examination of the contract development.

3.2. Contract enforcement

Contract enforcement is a technique where a smart contract is deployed to prevent contract breaches. As shown in Figure 3, to be preventive, enforcement operates intrusively (rather than nonintrusively like in contract monitoring) in the sense that it interferes with the execution of each operation.

In the example of the figure, a smart contract is deployed for enforcing the execution of the "pay 100" operation shown in Figures 1 and 2. The smart contract is responsible for ensuring that the "pay 100" operation is executed as agreed upon, for example, within the deadline. Though not shown explicitly in the figure, a finite state machine (FSM) operates inside the contract. The FSM keeps track of the current state of the contract, for instance, it keeps records about what obligations have been fulfilled. Its records can help the contract to determine what operations are currently pending and what are legal or illegal, that is, contract compliant or noncontract compliant.

1. The ideal execution path is shown by the green line: boxes 1–4.
2. Bob's application tries to place the execution of the "pay 100" operation against the executable code that implements the pay operation (box 3).
3. The operation is intercepted (box 2) by the digital contract and analyzed for contract compliance. If it is, the contract forwards the operation to the code that implements the pay operation, otherwise (if the operation is illegal) the contract trashes the operation (box 5) so that its execution is denied.
4. If the "pay 100" operation reaches the executable code, it is executed and Alice is notified. Alice's application does not necessarily receive the actual money, it might receive only a payment notification as shown in the figure by box 4, for example, a bank receipt.

https://doi.org/10.1017/dap.2023.49 Published online by Cambridge University Press

5. The enforcement contract is responsible for assuring that Bob complies with his obligation to pay. Accordingly, it includes an enforcing mechanism (boxes 6–9) that is programmed to send reminders to Bob and to collect the payment if Bob fails to honor his obligation.
6. Box 6 checks if Bob has a pending obligation to pay. If the answer is "yes" the smart contract verifies (box 7) if the deadline has been missed.
7. If the deadline has not been missed yet, the smart contract sends reminders (box 8) to Bob's app.
8. If the deadline has been missed by Bob, the smart contract triggers the execution of the executable code that implements the pay operation (box 3) to collect Bob's payment automatically. A pay notification is sent to Alice's app (box 4).
9. Box 2 allows the execution only of legal operations. For instance, it will not allow Bob's application to execute a "pay 100" operation outside the pay window, that is, before or after the agreed-upon paydays. Neither will it allow executing "pay 100" after the payment has been provided by Bob or enforced by the smart contract.

The power to enforce depends on the particularities to execute the operations; for instance, a contract is able to enforce Bob's "pay 100" operation if it is provided with money in advance, such as in escrow, or linked to Alice's accounts via some API; otherwise, the contract can only send a warning message to Bob's application to remind him of his pending obligation; next it is up to Bob's application to honor or violate the contract.

4. Hybrid contracts can provide flexibility

A hybrid contract is a smart contract that is executed and enforced automatically by computer executable code in collaboration with humans (Law Commission, 2020; Law Commission 2021) Some authors use the term Ricardian contract (Grigg, 2015) instead of hybrid and remark that these contracts consist of two parts that are cryptographically bound: executable code and tagged text that is human-readable. Human intervention is required because the assumption is that hybrid contracts are incomplete. Thus, at some point, their execution reaches a gap left accidentally or intentionally by its designers. Incomplete contracts are frequently used in business because they offer several advantages, including simplicity and flexibility (Hadfield, 1994; Ayrest and Gertner, 2018; Rodrigues, 2018).

We assume that the hybrid contract is designed (drawn up, specified) by multidisciplinary professionals with a deep understanding of issues that lie at the intersection of fundamental computer science, policymaking, jurisprudence, and human rights. Also, we assume that programmers are responsible for translating the designed contract into computer executable code. We assume that the programmers are multidisciplinary professionals with a similar background as the designers. We return to this question in Section 6.

We will use Figure 4 to explain how a hybrid contract can be used in algorithmic governance to provide flexibility. The figure makes no assumptions about the particularities of law that the hybrid contract is meant to enforce. It can be any legal obligation like the enforcement of tax payment, a right to claim insurance or pension, or the enforcement of a prison sentence. The bottom part of the figure is the component of a hybrid contract that executes automatically. It is deployed on a conventional blockchain platform (e.g., Ethereum) composed of N nodes that host the executable code of the hybrid contract. Only four miner nodes are shown; each of them runs an instance (HC_i) of the executable code. These instances are responsible for participating in a consensus protocol (say proof of work, proof of stake, etc.) to agree on the execution of an action (called transaction by the blockchain community) that leads to the next stage of the hybrid contract. The figure assumes that the designers intentionally or accidentally left the specification of the hybrid contract incomplete, thus, at some points, it stops and requests human interventions, and the top part of the figure is activated. At each stop the decisions to progress the hybrid contract are taken either by a single human (Clare in the figure) unilaterally or, in borderline situations, by a committee composed of M humans (only three are shown) after running a protocol to reach a consensus. The figure makes no assumptions about the consensus protocol run by Alice, Bob, and Clare.

https://doi.org/10.1017/dap.2023.49 Published online by Cambridge University Press

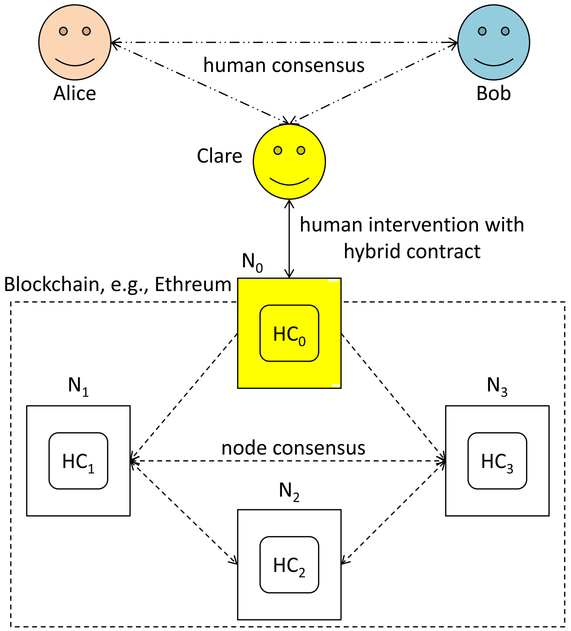

Figure 4. *A hybrid contract driven by majority (miners') consensus and human consensus.*

In practice, the automatic (bottom) part of the figure will encode laws that are suitable for the majority and therefore they are likely to be enforced (see Figure 3). Complementary, the nonautomatic (top) part of the figure will account for the minorities and their odd (exceptional) cases that are too risky to solve without human judgment.

For example, imagine that the hybrid contract is responsible for enforcing the execution of a prison sentence to be served. An enforcing smart contract programmed to operate without human intervention will act like in Figure 3, that is, impose its algorithm-based verdict independently. The hybrid contract shown in Figure 4 is more flexible because it will stop to request human intervention to determine if the sentence is fair or unfair before enforcing it.

Smart contracts deployed to monitor (see Figure 2) are inherently flexible because they only observe and collect records about the activities executed by individuals. The records collected can be examined either programmatically by computers or manually by humans. Therefore, the smart contract is not responsible for making critical decisions. For example, it is the responsibility of whoever examines the records (not of the smart contract) to classify a homicide as murder or manslaughter.

5. Future work and preventive law

Technology provides the opportunity to transform not just the judicial system but the legal system in general. We consider that the most significant aspect of algorithmic governance does not lie in the automation of existing laws to apply them faster, at lower costs, or increase efficiency. This is undoubtedly helpful. However, in our opinion, the most valuable benefit of algorithmic governance is innovation: algorithmic governance opens the opportunity to introduce radical changes to the long-standing systems that have been used to govern societies for centuries. It is widely acknowledged that the democracy that we know suffers from numerous shortcomings. The availability of technology presents us with the opportunity to include changes that without technology were unattainable.

Technology can help algorithmic governance to innovate in several fields such as e-democracy, participatory budgets, online voting, and the implementation of automatic preventive laws. The latter is particularly challenging and a topic in our research agenda to progress the discussion presented in this

https://doi.org/10.1017/dap.2023.49 Published online by Cambridge University Press

article. The second author has been studying preventive laws as a central topic of her in-progress PhD research. Preliminary results are available in an unpublished manuscript (Felizia, 2023). We will discuss the main ideas in this section.

We define automatic preventive laws as an algorithmic system that aims at the prevention of the execution of criminal acts, as opposed to criminal punishment. The latter is the prevalent practice in current judicial systems, where the perpetrator is punished when the act is already committed. We understand that this topic is likely to generate controversial discussion because at first glance it seems that it attempts against freedom, in fact, careless implementation and abuse can result in surveillance and repression. It is important to note that the creation of automatic preventive laws is not intended to be repressive. The challenge is implementing preventive law under the observance of human rights as we currently know them and as they might evolve.

In this context, we use the term law to refer to legal norms in general (rules, decrees, wills, and contracts). These laws would be challenging to circumvent as they are applied ex ante. Additionally, they do not require the participation of third parties, such as police or judges (who do not always ensure the fairness of proceedings and the proper administration of justice). Consequently, a judicial process would not be necessary. Digital tools such as sensors and artificial intelligence can help in the development of these systems.

Existing government laws are punitive. They resort to punishing criminals and restoring (if possible) the harm inflicted upon victims. For instance, a law that penalizes the actions of a thief who commits a robbery. Digital technology can enable the implementation of preventive laws to deter individuals from engaging in criminal activities. For example, the tax system could automatically collect taxes, eliminating both accidental and deliberate taxpayers' evasions. A more complex and contentious example would be to arrest a potential murderer before harming the intended victim.

Preventive law can be used to prevent petty crime (e.g., traffic offenses and underage drinking). However, their main benefit would be preventing catastrophic and often irreversible criminal actions, such as murders, government frauds, and large bank frauds. Reasons for implementing preventive laws may lie in the potential to prevent further harm or even the loss of innocent lives, which would constitute a greater injustice.

We suggest that automatic preventive laws be gradually introduced in various sectors of society and at different regulatory levels. However, societies should not adopt automatic preventive laws unless they consider human participation in extreme circumstances through flexible hybrid smart contracts that stop and request human intervention when necessary.

In our forthcoming analysis, we will explore the application of automatic preventative laws to help monitor potential monopolistic behavior online. In a preventive law system, artificial intelligence can be used to detect and signal if a marketplace manipulates search results to favor specific sellers and products. If this happens, corresponding legal measures can be enforced automatically before the occurrence of illegal acts. This approach ensures fair competition and consumer protection. The general idea is that automatic preventive laws can act as a powerful deterrent against the creation of monopolies.

6. Conclusions

Algorithmic governance is all about law automation, which is an alternative that has the potential to ameliorate several problems that afflict current legal systems, such as its unacceptable slow pace and lack of impartiality introduced by judges that succumb to corruption. However, if technology is adopted, care should be taken not to create a computer-driven system that is unnecessarily rigid and dehumanized. Automated law should not be embraced unless it accounts for human involvement in borderline situations where human intelligence is likely to produce fairer decisions.

Some authors refer to algorithmic governance as Lex Algomata (Molina-Jimenez, 2023) to emphasize the risk that algorithmic governance can bring. Algomata is a term that possesses profound meaning and is formed by combining two words: the word *algo* which is the prefix of algorithm, and the word *mata* which in English means to kill and is also the suffix of the word automata (an automaton is the graphical

https://doi.org/10.1017/dap.2023.49 Published online by Cambridge University Press

representation of an algorithm). Therefore, *algomata* carries a negative connotation and emphasizes that careless use of automation in the field of Law can have undesirable and irreversible consequences.

Let us not forget ancient traditions and the human factor in discussions of law governance. It is essential to bear in mind that the transformation of legal practice by the use of automation of law will not be necessarily welcome by the lawyer community. One of the problems is that Law automation can lead to job losses for many legal professionals who do not have sufficient knowledge of technology. However, it may be a good opportunity for younger or newly graduated attorneys who adapt faster to technological changes. A further important point to consider is the opportunities and challenges that the automation of law will bring to courts, law firms, lawyers, and people who need their services. In relation to procedural law, on the one hand, the vision is that automated law will help judges to resolve cases that need minimal human intervention in less time. On the other hand, it will force judges to become familiar with programming languages to understand code and, in general, with computer technology. A more general and fundamental question here is which professionals will be responsible for the implementation of the programs (say, the smart contracts) that are needed to automate the law, to certify that they are correct, and to interpret their results when human intervention is needed. We are asking for multidisciplinary professionals with Law and Computer Science backgrounds, that is, lawyers with the knowledge to read programming code and software engineers with the knowledge to read civil codes (Grimmelmann, 2022). Currently, such professionals are missing, and it is not clear who and where they can be trained. A possible solution is to create a branch of traditional software engineering to cover the existing gap (Felizia et al., 2022b). In fact, some authors have already suggested the creation of Blockchain-Oriented Software Engineering (BOSE) (Porru et al., 2017; Destefanis et al., 2018; Fahmideh et al., 2021). We agree with their views but suggest that BOSE also covers legal aspects thoroughly. These professionals will help to design and program the smart contracts and to react to requests for human intervention placed by hybrid contracts.

Another important issue is that total automation can generate rigidity in a system and inflexibility in decision making. For this reason and as argued in this article, we believe that sometimes unilateral decisions of a computer system or a single individual do not offer a fair solution. In these situations, consensus that emerges from the agreement between several people could provide greater certainty about decisions.

Let us not forget that so far, technologists have not been able to produce technology that is 100% reliable. Their current technology is embarrassingly brittle to rely on it for serious matters such as dictation of a prison sentence or, in some countries, executions.

The recent outage of Facebook, WhatsApp, and Instagram on October 4, 2022 that lasted for about 6 h can help to illustrate the argument. Apparently, the outage left billions of users without the services was caused by an internal configuration issue (https://www.bbc.co.uk/news/technology-58793174) and suggests that failures are unavoidable. As a second example, we can mention the failures of artificial intelligence (AI) technologies that have been already used to assist in law automation. There are examples that have shown that AI algorithms are not infallible. For example, facial recognition used in the criminal sphere has led to biased (i.e., racial) decisions. This example shows that careless use of unsound technology can result in systematic discrimination of minority groups (Hassan and Filippi, 2017; Choi, 2021; Norori et al., 2021).

Perfecting this technology will take time. Therefore, we consider that while these technological problems are solved, it is not convenient to fully automate or make decisions based entirely on the orders issued by a machine. Human intervention is essential to protect the life, equality, dignity, and fundamental rights of people.

Data availability statement. An open data version of this document can be found in the arxiv repository: https://arxiv.org/abs/2312.07565.

Acknowledgments. For the purpose of open access, the author Carlos Molina-Jimenez has applied a Creative Commons Attribution (CC BY) license to any author accepted manuscript version arising from this submission.

Author contribution. The first author was responsible for the technical arguments and suggested solutions. The second author was responsible for the legal arguments, in particular, the conclusions and remarks.

Funding statement. The first author has been supported by UKRI grant EP/X015785/1(G115169), the Department of Computer Science and Technology of the University of Cambridge (www.cst.cam.ac.uk) and TODAQ (todaq.net).

Competing interest. The authors declare none.

Ethical standard. The research complies with the ethical guidelines of the University of Cambridge and the Inter-American Open University where the research was conducted.

References

Ayrest I and Gertner R (2018) Filling gaps in incomplete contracts: An economic theory of default rules. *The Yale Law Journal 99* (1), 87–130.

Choi CQ (2021) 7 revealing ways ais fail: Neural networks can be disastrously brittle, forgetful, and surprisingly bad at math. *IEEE Spectrum 58*(10), 42–47.

Destefanis G, **Marchesi M**, **Ortu M**, **Tonelli R**, **Bracciali A and Hierons R** (2018) Smart contracts vulnerabilities: A call for blockchain software engineering? In *Proceedings of International Workshop on Blockchain Oriented Software Engineering (IWBOSE)*. IEEE, pp. 19–25.

Emerson P (2018) *From Majority Rule to Inclusive Politics*. Switzerland: Springer.

Ethereum Foundation (2018) Ethreum: Blockchain App Platform. Available at https://www.ethereum.org (accessed 23 October 2027).

Fahmideh M, **Fahmideh M**, **Fahmideh M**, **Shen J**, **Yan J**, **Mougouei D**, **Wang P**, **Ghose A**, **Gunawardana A**, **Aickelin U and Abedin B** (2021) Software engineering for blockchain based software systems: Foundations, survey, and future directions. arXiv: 2105.01881 [cs.SE].

Felizia SM (2023) Implementation of Automatic Preventive Law under Algorithmic Governance. PhD thesis, Faculty of Law, National University of Rosario.

Felizia SM, **Molina-Jimenez C**, **Frantz RZ**, **Reina-Quintero AM and Valente AD** (2022a) Fortalecimiento del derecho a la confidencialidad en la gobernanza algorítmica. In Navarro E (ed.), *Actas de las XVII Jornadas de Ingeniería de Ciencia e Ingeniería de Servicios (JCIS 2022)*. Sistedes.

Felizia SM, **Molina-Jimenez C and Valente AD** (2022b) ISLA: Ingeniería de software para leyes automáticas. In Navarro E (ed.), *Actas de las XVII Jornadas de Ingeniería de Ciencia e Ingeniería de Servicios (JCIS 2022)*. Sistedes.

Filippi PD and Wright A (2018) *Blockchain and the Law: The Rule of Code*. Cambridge, MA: Hardvard University Press.

Gamito MC and Ebers M (2021) Algorithmic governance and governance of algorithms: An introduction. In Ebers M and Gamito MC (eds.), *Algorithmic Governance and Governance of Algorithms Legal and Ethical Challenges. Data Science, Machine Intelligence, and Law*, Vol. *1*. Cham: Springer, pp. 2–18.

Gasser U and Almeida V (2022) Futures of digital governance. *Communications of the ACM 65*(3), 30–32.

Genesereth M (2021) What is computational law? Stanford University. Available at https://law.stanford.edu/2021/03/10/what-is-computational-law/.

Grigg I (2015) On the Intersection of Ricardian and Smart Contracts. Available at https://iang.org/papers/intersection_ricardian_smart.html (accessed 24 December 2020).

Grimmelmann J (2022) Programming languages and law. In *Proceedings of the 2022 Symposium on Computer Science and Law (CSLAW'22)*. New York, NY: Association for Computing Machinery.

Hadfield GK (1994) Judicial competence and the interpretation of incomplete contracts. *The Journal of Legal Studies 23*(1), 159–184.

Hassan S and Filippi PD (2017) The expansion of algorithmic governance: From code is law to law is code. *The Journal of Field Actions 17*(17), 88–90.

Hazard J and Haapio H (2017) Wise contracts: Smart contracts that work for people and machines. In Schweighofer E, *et al.* (eds.), *Trends and Communities of Legal Informatics. Proceedings of the 20th International Legal Informatics*. Wien: Österreichische Computer Gesellschaft, pp. 425–432.

Lacock DD (1964) The relevance of logic to law. *Modern Uses of Logic in Law 5*(2), 13–23.

Law C (2005) Nathaniel love and michael genesereth. In *Proceedings of 10th International Conference on Artificial Intelligence and Law (ICAIL 2005)*. New York, NY: ACM, pp. 205–206.

Law Commission (2020) Smart Contracts Call for Evidence. Available at https://s3-eu-west-2.amazonaws.com/lawcom-prod-storage-11jsxou24uy7q/uploads/2020/12/201216-Smart-contracts-call-for-evidence.pdf (accessed 19 December 2020).

Law Commission (2021) Smart Legal Contracts Advice to Government. Available at https://s3-eu-west-2.amazonaws.com/lawcom-prod-storage-11jsxou24uy7q/uploads/2021/11/Smart-legal-contracts-accessible.pdf (accessed 28 June 2022).

Law.MIT.edu (2021) Computational Law. Available at https://law.mit.edu (accessed 5 July 2021).

Micheler E and Whaley A (2020) Regulatory technology: Replacing law with computer code. *European Business Organization Law Review 21*, 349–377.

Molina-Jimenez C (2023) Ley Algomata. Available at https://github.com/carlos-molina/LeyAlgomata (accessed 1 April 2023).

Norori N, **Hu Q**, **Aellen FM**, **Faraci FD and Tzovara A** (2021) Addressing bias in big data and AI for health care: A call for open science. *Patterns 2*(10), 100347.

https://doi.org/10.1017/dap.2023.49 Published online by Cambridge University Press

Porru S, **Pinna A**, **Marchesi M and Tonelli R** (2017). Blockchain-oriented software engineering: Challenges and new directions. In *Proceedings of 39th International Conference on Software Engineering Companion*. IEEE, pp. 169–171.

Rodrigues U (2018) Law and the blockchain. *Iowa Law Review 104*, 679–729.

Sklaroff JM (2018) Smart contracts and the cost of inflexibility. *University of Pennsylvania Law Review 166*, 263.

Surden H (2012) Computable contracts. *UC Davis Law Review 46*, 72.

Werbach K (2020) The siren song: Algorithmic governance by blockchain. In Werbach K (ed.), *After the Digital Tornado: Networks, Algorithms, Humanity, Chapter 9*. Cambridge: Cambridge University Press, pp. 215–239.

Cite this article: Molina-Jimenez C and Felizia SM (2024). On the use of smart hybrid contracts to provide flexibility in algorithmic governance. *Data & Policy*, 6: e8. doi:10.1017/dap.2023.49

https://doi.org/10.1017/dap.2023.49 Published online by Cambridge University Press