The Auto-Learning Framework For Dealing With Network Optimization Problems in Wireless Communication Systems

[1]PillaVinay Kumar, [2]B Maha Lakshmi Rao
[1,2]Dept. of Computer Science & Engineering, KIET, Kakinada, AP, India

Abstract
In Wireless Communication Systems (WCSs), the network optimization problems (NOPs) play an important role in maximizing system performance by setting appropriate network configurations. When dealing with NOPs by using conventional optimization methodologies, there exist the following three problems: human intervention, model invalidity, and high computation complexity. As such, in this article we propose an auto-learning framework to achieve intelligent and automatic network optimization by using machine learning (ML) techniques. We review the basic concepts of ML, and propose their rudimentary employment models in WCSs, including automatic model construction, experience replay, efficient trial and error, RL-driven gaming, complexity reduction, and solution recommendation. We hope these proposals can provide new insights and motivation in future research for dealing with NOPs in WCSs by using ML techniques.

I. Introduction

In Wireless Communication Systems (WCSs), network optimization problems (NOPs) have been extensively studied to maximize system performance by setting appropriate network configuration settings. NOP contains a broad range of research aspects in wireless networking; typical applications include resource allocation and management, system parameter provision, task scheduling, and user quality of service (QoS) optimization. Figure 1 shows the basic process of solving a NOP in WCSs, which includes the following four steps. Data Collection: the collection of essential information of the system and the surrounding environment. The collected data can be channel state information (CSI), interference, noise, user location, spectrum and time slot occupations, and so on. Some QoS information, such as delay and energy consumption rates and mobility state, can also be the input data to support the following optimization process. Model Construction: in which the expert constructs an optimization model that contains an objective function and several constraints. The objective of the optimization model can be throughput, spectrum utilization, user-perceived delay, energy consumption/gain, facility deploy (cment cost, and so on. Typically, model construction is conducted by using a mathematical formulation process, and experts are required to master the domain knowledge and theories involved in the model. Optimization: The most commonly used methodologies for solving optimization problems are mathematical derivation-based methods (DBMs) and heuristic algorithms. The former adopt a mathematical derivation process to find the solution, such as the Lagrangian duality, Karush-Kuhn-Tucker (KKT) conditions, and gradient descendent methodologies. The latter adopt a heuristic neighborhood searching process to approach the optimal solution, including genetic algorithm, simulated annealing, particle swarm optimization, firefly algorithms, and so on. In general, DBMs are quite suitable for solving problems with explicit and convex objective functions, while heuristic algorithms do not require the derivatives of the objective functions, and are generally able to produce high-quality solutions for complex optimization problems if the optimization complexity is suitably high. Besides the above two optimization methods, game theoretical techniques, including non-cooperative games, cooperative games, and Bayesian games, also have been successfully applied to solve the optimization problem by learning automatic configuration strategies from interactions with other functional nodes.

A. Configuration

With the optimization results, the system then reconfigures the settings of the system to improve the performance. Possible reconfigurations may include transmission power allocation, energy harvesting scheduling, routing decision, and spectrum resource allocation, to name a few. After configuration, the system then repeats the optimization process to keep the system in suitable working conditions. Although NOPs have been extensively studied in WCSs, existing optimization methodologies still face the following three dilemmas.

B. Human Intervention

The optimization models in NOPs are always constructed by experts with domain knowledge, and this knowledge-driven process is expensive and inefficient in practical implementations. If we can conduct the optimization operations automatically, network optimization will be easier to conduct in real world applications. However, how to reduce human intervention in solving NOPs is still an unexplored field in WCSs.

II. Literature Survey

1. Label-less Learning for Traffic Control in an Edge Network AUTHORS: Min Chen, YixueHao, Kai Lin, Zhiyong Yuan, Long Hu

With the development of intelligent applications (e.g., self-driving, real-time emotion recognition, etc), there are higher requirements for the cloud intelligence. However, cloud intelligence depends on the multi-modal data collected by user equipments (UEs). Due to the limited capacity of network bandwidth, offloading all data generated from the UEs to the remote cloud is impractical. Thus, in this article, we consider the challenging issue of achieving a certain level of cloud intelligence while reducing network traffic. In order to solve this problem, we design a traffic control algorithm based on label-less learning on the edge cloud, which is dubbed as LLTC. By the use of the limited computing and storage resources at edge cloud, LLTC evaluates the value of data, which will be offloaded. Specifically, we first give a statement of the problem and the system architecture. Then, we design the LLTC algorithm in detail. Finally, we set up the system testbed. Experimental results show that the proposed LLTC can guarantee the required cloud intelligence while minimizing the amount of data transmission.

2. Coalitional Game Theory for Communication Networks: A Tutorial AUTHORS: W. Saad et al

Game theoretical techniques have recently become prevalent

in many engineering applications,notably in communications. With the emergence of cooperation as a new communication paradigm, and the need for selforganizing, decentralized, and autonomic networks, it has become imperative to seek suitable game theoretical tools that allow to analyze and study the behavior and interactions of the nodes in future communication networks. In this context, this tutorial introduces the concepts of cooperative game theory, namely coalitional games, and their potential applications in communication and wireless networks. For this purpose, we classify coalitional games into three categories: Canonical coalitional games, coalition formation games, and coalitional graph games. This new classification represents an application-oriented approach for understanding and analyzing coalitional games. For each class of coalitional games, we present the fundamental components, introduce the key properties, mathematical techniques, and solution concepts, and describe the methodologies for applying these games in several applications drawn from the state-of-the-art research in communications. In a nutshell, this article constitutes a unified treatment of coalitional game theory tailored to the demands of communications and network engineers.

3. Machine Learning for Networking: Workflow, Advances and OpportunitiesAUTHORS: M. Wang et al;Vladimir I. Yaropolov

The paper shows historical aspects of the application of computer-based information systems, providing the process of cosmonaut training for a space flight at the Gagarin Cosmonaut Training Center. These systems include: systems ensuring the operation of simulation complexes for cosmonaut training, computer-assisted instruction systems (computer-assisted simulators), databases for storing the results of cosmonaut selection and training, information-management systems of cosmonaut training, cosmonaut training planning systems, data retrieval systems (electronic libraries, electronic catalogs), multimedia complexes, etc.

4. Extreme learning machines: a survey AUTHORS: G. B. Huang, Jorge D. Cambab

Computational intelligence techniques have been used in wide applications. Out of numerous computational intelligence techniques, neural networks and support vector machines (SVMs) have been playing the dominant roles. However, it is known that both neural networks and SVMs face some challenging issues such as: slow learning speed, trivial human intervene, and/or poor computational scalability. Extreme learning machine (ELM) as emergent technology which overcomes some challenges faced by other techniques has recently attracted the attention from more and more researchers. ELM works for generalized single-hidden layer feedforward networks

III. Existing System

In Wireless Communication Systems (WCSs), network optimization problems (NOPs) have been extensively studied to maximize system performance by setting appropriate network configuration settings. In recent years, machine learning (ML) techniques have shown powerful magic in dealing with networking problems, such as traffic prediction, point-to-point regression, and signal detection. However, the application of ML in dealing with NOPs has not been fully discussed in existing works.

Disadvantages of Existing System

- The optimization models in NOPs are always constructed by experts with domain knowledge, and this knowledge-driven process is expensive and inefficient in practical implementations.
- With the development of hardware and software techniques, the WCS is becoming an increasingly complex system with more users, more access ways, and more complex functions and relationships among the network entities.

B. Proposed System

Network Optimization Problems (NOPs) in WCSs, and propose an auto-learning framework (ALF) that employs ML techniques to achieve intelligent and automatic network optimization in WCSs. Within ALF, we propose several potential paradigms, including automatic model construction, experience replay, efficient trial and error, reinforcement learning (RL)-driven gaming, complexity reduction, and solution recommendation. The basic workflows, applications, and challenges of these models are discussed. Collecting the experience data is the prerequisite for conducting ML-based models and must be properly addressed. Besides the system and environment state information, in ALF the output solution data of an optimization process is also collected as historical experience.

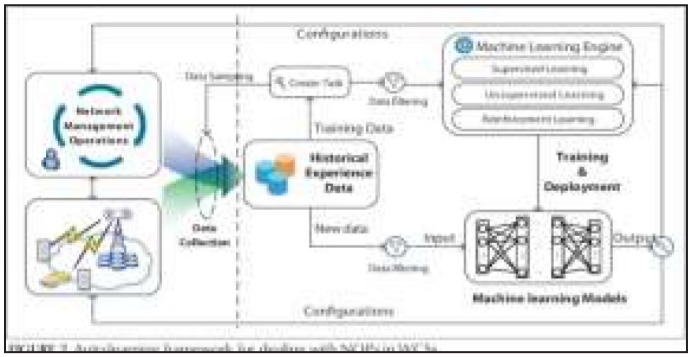
Advantages Of Proposed System:

- Model Deployment. The deployment of the mapping model is very easy to achieve. The calculation process mainly includes matrix multiplications and nonlinear transforms with activation functions, and both of them can be easily calculated.
- The black-box auto-learning model may need to be refined due to the change of wireless systems and environments, and imperfect training data. The dynamic adjusting of an ML model can be regarded as an incremental learning problem, and the key step is the proper updating of training data instances.

Therefore, it is suggested to update the training dataset periodically to guarantee that the obtained model performs well when the system model is changed.

IV. System Architecture

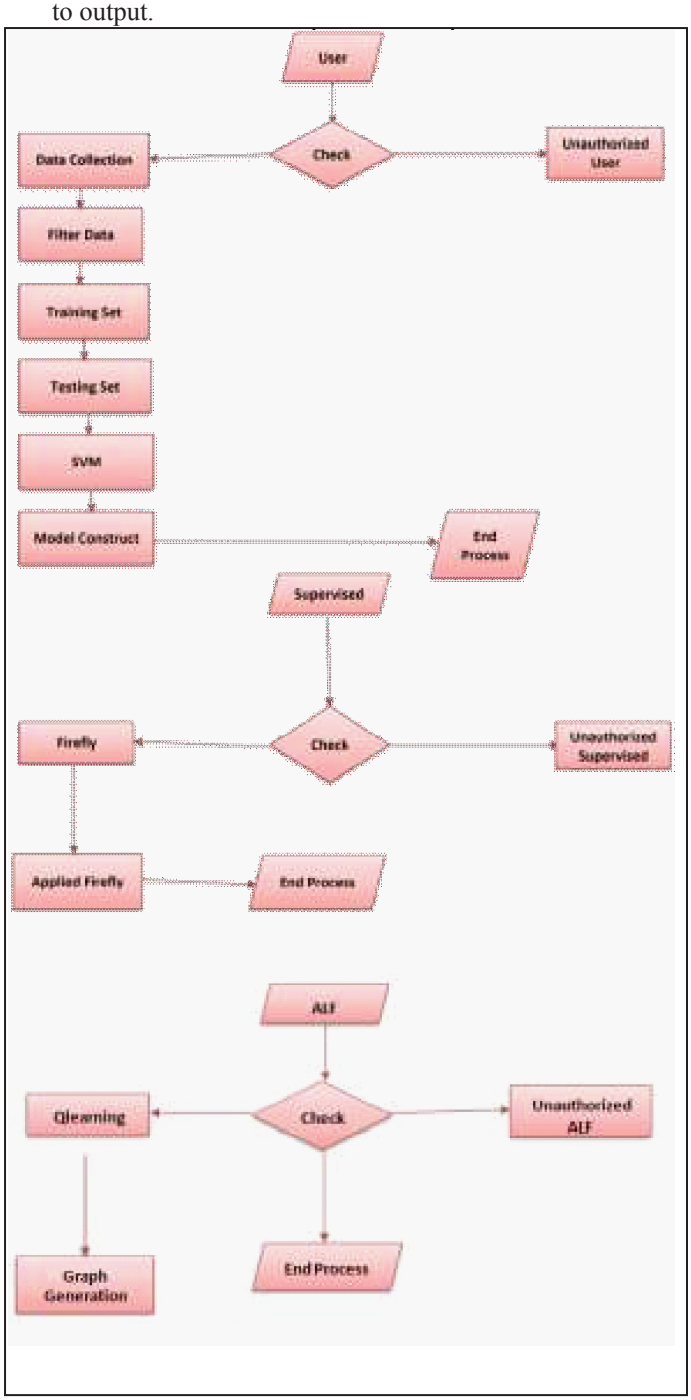
Below diagram depicts the whole system architecture of The most trending articles every year using NLP technique.



Data Flow Diagram

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.

2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.



V. System Implementation

Modules

- Q-learning model
- Firefly algorithm
- KKT Condition

A. Q-learning model

Q-learning is amodel-freereinforcementlearningalgorithm. The goal of Q-learning is to learn a policy, which tells an agent what action to take under what circumstances. It does not require a model (hence the connotation "model-free") of the environment, and it can handle problems with stochastic transitions and rewards, without requiring adaptations. For any finiteMarkov decision process(FMDP), Q-learning finds a policy that is optimal in the sense that it maximizes the expected value of the total reward over any and all successive steps, starting from the current state. Qlearning can identify an optimal action-selection policy for any given FMDP, given infinite exploration time and a partly-random policy. "Q" names the function that returns the reward used to provide the reinforcement and can be said to stand for the "quality" of an action taken in a given state.

B. Firefly algorithm

Firefly Algorithm (FA) is a metaheuristic algorithm for global optimization, which is inspired by flashing behavior of firefly insects. This algorithm is proposed by Xin-She Yang in 2008. Fireflies use the flashing behavior to attract other fireflies.

Fireflies are winged beetles or insects that produce light and blinking at night. The light has no infrared or an ultraviolet frequency which is chemically produced from the lower abdomen is called bioluminescence. They use the flash light especially to attract mates or prey. The flash light also used as a protective warning mechanism to remind the fireflies about the potential predators. Firefly algorithm formulated by Yang is a metaheuristic algorithm that is inspired by the flashing behavior of fireflies and the phenomenon of bioluminescent communication. formulated the Firefly Algorithm with the following assumptions:

A firefly will be attracted to each other regardless of their sex because they are unisexual.

Attractiveness is proportional to their brightness whereas the less bright firefly will be attracted to the brighter firefly.

However, the attractiveness decreased when the distance of the two fireflies increased.

Fireflies are winged beetles or insects that produce light and blinking at night. The light has no infrared or an ultraviolet frequency which is chemically produced from the lower abdomen is called bioluminescence. They use the flash light especially to attract mates or prey. The flash light also used as a protective warning mechanism to remind the fireflies about the potential predators. Firefly algorithm formulated by Yang [10] is a metaheuristic algorithm that is inspired by the flashing behavior of fireflies and the phenomenon of bioluminescent communication. [10] formulated the Firefly Algorithm with the following assumptions:

1. A firefly will be attracted to each other regardless of their sex because they are unisexual.
2. Attractiveness is proportional to their brightness whereas the less bright firefly will be attracted to the brighter firefly.

However, the attractiveness decreased when the distance of the two fireflies increased.

Fireflies are winged beetles or insects that produce light and blinking at night. The light has no infrared or an ultraviolet frequency which is chemically produced from the lower abdomen is called bioluminescence. They use the flash light especially to attract mates or prey. The flash light also used as a protective warning mechanism to remind the fireflies about the potential predators.

Firefly algorithm formulated by Yang [10] is a metaheuristic algorithm that is inspired by the flashing behavior of fireflies and the phenomenon of bioluminescent communication. [10] formulated the Firefly Algorithm with the following assumptions: 1) A firefly will be attracted to each other regardless of their sex because they are unisexual. 2) Attractiveness is proportional to their brightness whereas the less bright firefly will be attracted to the brighter firefly. However, the attractiveness decreased when the distance of the two fireflies increased.

C. KKT Condition

The Karush–Kuhn–Tucker (KKT) conditions, also known as the Kuhn–Tucker conditions, are first derivative tests (sometimes called first-order) necessary conditions for a solution in nonlinear programming to be optimal, provided that some regularity conditions are satisfied.Allowing inequality constraints, the KKT approach to nonlinear programming generalizes the method of Lagrange multipliers, which allows only equality constraints. Similar to the Lagrange approach, the constrained maximization (minimization) problem is rewritten as a Lagrange function whose optimal point is a saddle point, i.e. a global maximum (minimum) over the domain of the choice variables and a global minimum (maximum) over the multipliers, which is why the Karush–Kuhn– Tucker theorem is sometimes referred to as the saddle-point theorem.The KKT conditions were originally named after Harold W. Kuhn and Albert W. Tucker, who first published the conditions in 1951. Later scholars discovered that the necessary conditions for this problem had been stated by William Karush in his master's thesis in 1939.

VI. System Test

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assembliesand/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

A. Types Oftests

1. Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

2. Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

3. Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals. Functional testing is centered on the following items:

- Valid Input: identified classes of valid input must be accepted.
- Invalid Input: identified classes of invalid input must be rejected.
- Functions: identified functions must be exercised.
- Output: identified classes of application outputs must be exercised.

Systems/Procedures:interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

4. System Test

System testing ensures that the entire integrated software system meets requirements. It tests a
configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre- driven process links and integration points.

5. White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

6. Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. you cannot "see" into it. The test provides inputs and responds to outputs without considering how the software works.

B. Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

All field entries must work properly.

Pages must be activated from the identified link. The entry screen, messages and responses must not be delayed. Features to be tested

Verify that the entries are of the correct format No duplicate entries should be allowed All links should take the user to the correct page.

1. Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error. Test Results: All the test cases mentioned above passed successfully. No defects encountered.

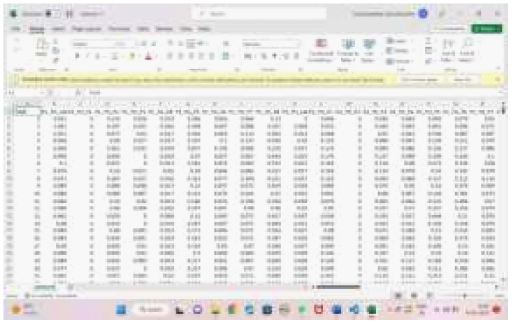
2. Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

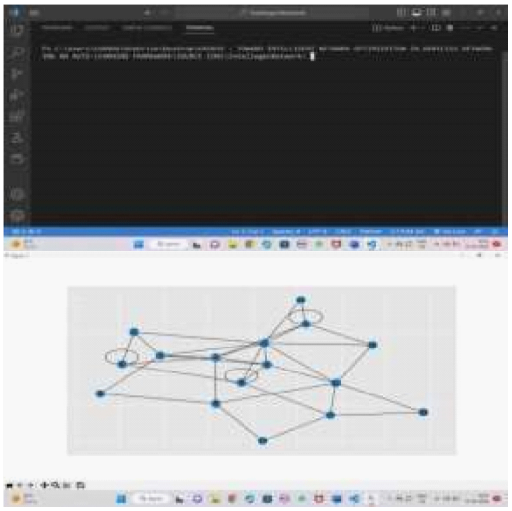
Test Results: All the test cases mentioned above passed successfully. No defects encountered.

VII. Screen Shots

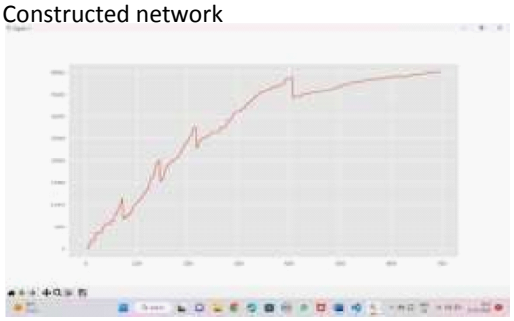
DATA SET



Running code



Constructed network



Scoreeach link

VIII. Conclusion & Future Work

This article recalls the models of network optimization in WCSs and proposes an ALF that employs the advantages of powerful ML techniques to deal with the human intervention, model invalidity, and high complexity problems in conventional optimization models. We review the basic concepts of supervised learning, reinforcement learning, and unsupervised learning, and then propose several potential models to deal with NOPs, including automatic model construction, experience replay, efficient trial and error, RL-driven gaming, complexity reduction, and solution recommendation. We encourage readers to test and modify these proposals, and further design more new ML-based methods for dealing with NOPs in WCSs.

REFERENCES

[1] David Tse and PramodViswanath, Fundamentals of Wireless Communication, Cambridge Univ. Press, 2005.

[2] E. K. P. Chong and S. H. Zak, An Introduction to Optimization, 3rd ed., 2011.

[3] W. Saad et al., "Coalitional Game Theory for Communication Networks: A Tutorial," IEEE Signal Processing Mag., vol. 26, no. 5, 2009, pp. 77–97.

[4] M. Wang et al., "Machine Learning for Networking: Workflow, Advances and Opportunities," IEEE Network, vol. 31, no. 2, Mar./Apr. 2017.

[5] M. Chen et al., "Cognitive-LPWAN: Towards Intelligent Wireless Services in Hybrid Low Power Wide Area Networks", IEEE Trans. Green Commun. and Networking, vol. 3, no. 2, June 2019, pp. 407–17.

[6] M. Chen and V. Leung, "From Cloud-Based Communications to Cognition-Based Communications: A Computing Perspective," Comp. Commun., vol. 18, 2018, pp. 74–79.

[7] M. Chen et al., "Labelless Learning for Traffic Control in an Edge Network," IEEE Network, vol. 32, no. 6, Nov./Dec. 2018, pp. 8–14.

[8] K. Hwang et al., Big Data Analytics for Cloud/IoT and Cognitive Computing, Wiley. ISBN: 9781119247029, 2017.

[9] E Bjornson, M Kountouris, and M Debbah, "Massive MIMO and Small Cells: Improving Energy Efficiency by Optimal Soft-Cell Coordination," Int'l. Conf. Telecommun., 2013, pp. 1–5.

[10] G. B. Huang, "Extreme Learning Machines: A Survey," Int'l. J. Machine Learning & Cybernetics, vol. 2, no. 2, 2011, pp. 107–22.