mathematics

MDPI

Article

Analysis of Nature-Inspired Algorithms for Long-Term Digital Preservation

Andres El-Fakdi *,†,‡ **and Josep Lluis de la Rosa** †,‡

TECNIO Centre EASY Research Group, VICOROB Institute, University of Girona, 17004 Girona, Spain; peplluis@eia.udg.edu

* Correspondence: aelfakdi@eia.udg.edu

† Current address: Department of Electrical Engineering, Electronics and Automation (EEEA), Campus Montilivi, Polytechnic School of the University of Girona, Building P4, 17003 Girona, Spain.

‡ These authors contributed equally to this work.

Abstract: Digital preservation is a research area devoted to keeping digital assets preserved and usable for many years. Out of the many approaches to digital preservation, the present research article follows a new object-centered digital preservation paradigm where digital objects share part of the responsibility for preservation: they can move, replicate, and evolve to a higher-quality format inside a digital ecosystem. In the new framework, the behavior of digital objects needs to be modeled in order to obtain the best preservation strategy. Thus, digital objects are programmed with the mission of their own long-term self-preservation, which entails being accessible and reproducible by users at any time in the future regardless of frequent technological changes due to software and hardware upgrades. Three nature-inspired computational intelligence algorithms, based on the collective behavior of decentralized and self-organized systems, were selected for the modeling approach: multipopulation genetic algorithm, ant colony optimization, and a virus-based algorithm. TiM, a simulated environment for running distributed digital ecosystems, was used to perform the experiments. The results map the relation between the models and the expected object diversity obtained in short- and mid-term digital preservation scenarios. Comparing the results, the best performance corresponded to the multipopulation genetic algorithm. The article aims to be a first step in the digital self-preservation field. Building nature-inspired model behaviors is a good approach and opens the door to future tests with other AI-based methods.

Keywords: computational intelligence; digital object preservation; distributed architectures

Citation: El-Fakdi, A.; de la Rosa, J.L. Analysis of Nature-Inspired Algorithms for Long-Term Digital Preservation. *Mathematics* **2021**, *9*, 2279. https://doi.org/10.3390/math9182279

Academic Editors: Ximeng Liu and Radi Romansky

Received: 20 July 2021
Accepted: 10 September 2021
Published: 16 September 2021

1. Introduction

Long-Term Digital Preservation (LTDP) or simply Digital Preservation (DP) is an increasing focus of businesses and public sector agencies, as well as scientists and citizens. The challenge in preserving valuable digital information—consisting of text, music, images, multimedia documents, web pages, sensor data, etc. generated throughout all areas of our society—is real and growing at an exponential pace. In 2010, a study by the International Data Corporation (IDC) had already found and warned that the dimension of the digital information already created has reached zettabyte magnitude, and it keeps growing at at enormous rate [1]. The LTDP of all this information is already a major concern for institutions, governments, and the scientific community. We all have the responsibility to preserve our knowledge, and it is now a challenge to find the most efficient and democratic system to achieve such a goal [2,3].

The problems associated with the scalability of the DP solutions are obvious. A shift away from large-scale digital archiving systems towards intelligent objects in environments with learning abilities is a trend in recent LTDP research projects. The intelligence of computer systems should be harnessed more fully to contribute to more powerful and less costly LTDP. Furthermore, most of current preservation approaches give solutions or

define novel architectures at the repository level, when, in fact, preservation is first related to objects and solutions should be found at the object level, making digital objects self-preserving. In this line, very few works have been carried out, but it is worth mentioning the data-centered perspective study using the Unsupervised Small World (USW) graph creation algorithm presented in [4]. The behavior of the objects must be simple and highly adaptable to an uncertain environment: LTDP is uncertain in the long term or perhaps simply unpredictable. A solution that today might be efficient might not work in a distant future time. Evolution has proven to be resilient and adaptive to unforeseen changes well into the future. That is why different Computational Intelligence (CI) techniques are considered for self-preserving objects, as explained in [5]. We focus on Evolutionary Computation (EC) [6], a family of algorithms for global optimization inspired by biological evolution, and Swarm Intelligence (SI), based on intelligence emerging from the collective behavior of individuals interacting inside a decentralized ecosystem. Both CI techniques aim to solve optimization problems through collective behavior and cooperation among the actors of a system. Such methodologies give theoretical and practical hints to help with the design of the tools and the behaviors of digital objects and provide models to anticipate the emergent outcome.

The main principle behind the presented approach is thinking of a digital object as a living thing that, understanding the network as an interacting environment shared with other objects, it moves in, from node to node, duplicating and evolving, cooperating with other files, with the final objective of guaranteeing the persistence of the information contained within it along its descendants. Our challenge is to design the behaviors of Self-Preserving Digital Objects (SPDOs), for which the responsibility of preservation belongs to the file itself instead of to people or to any centralized or distributed system. This approach, object-centered, self-organized by nature, is expected to prove to be the best preservation approach, as it will solve the core problems of DP with yet another benefit: digital objects will try to keep themselves *alive* in terms of being accessible, useful, and authentic. By natural selection, the best of them (the fittest) will survive the longest, which means they will enjoy a greater chance of preservation. Under this expectation, the research presented in this article is focused on self-organization paradigms, including those related to learning in Multi-Agent Systems (MAS).

The article is about digital preservation. Our proposal is about the definition of a new paradigm, a new scenario where part of the responsibility of preservation relies on the files themselves: they can move, replicate, and evolve to a higher-quality format inside a digital ecosystem. In the proposed framework, the behavior of digital objects would need to be modeled in order to obtain the best preservation strategy. Nature-inspired methods offer a good initial approach and open the door to test other AI-based models on digital object behavior inside a self-preservation network environment. The main objective of the article is to present a new paradigm of the solution to the digital preservation problem where the objects are programmed for self-preservation. The article investigates computational intelligence algorithms as a possible programming behavior for objects living in the proposed distributed environment. For that purpose, this article focuses on building CI models with interesting features as potential behaviors for SPDOs, aiming to find those leading to a higher diversity of digital objects in the LTDP context. The objectives of the article, which also represent the main contributions of the work, can be summarized as follows:

- The proposal of a new paradigm for the solution of the digital preservation problem. The proposed approach describes an environment where the digital objects are programmed for self-preservation and manage the resources available in a distributed network to guarantee their survival in the long-term;
- The article investigates the performance of Computational Intelligence (CI) programming as an initial approach for developing self-preserving behaviors for digital objects. The features present in the Evolutionary Computation (EC) and Swarm Intelligence

(SI) methodologies make them good candidates to be used for optimizing SPDO environments and open the door to future tests with other AI-based algorithms;
- In order to demonstrate the feasibility of the proposed paradigm and the programmed CI behaviors, a simulated environment was designed to test the environment and compare the performance offered by the three algorithms put to test.

The article is structured as follows: Section 2 details the current digital preservation platforms and the former work related to SPDOs. Section 3 gives a detailed explanation of the self-preserving objects paradigm. Section 4 describes the main features of CI algorithms and presents the three CI methods selected to carry out the simulated experiments; all three methodologies used for comparison are well-known metaheuristics from the literature and offer a solid continuous benchmark for testing. Section 5 presents the simulation software platform developed for the testing of multiple SPDO behaviors and strategies. Section 6 provides the simulated results obtained during the experiments. Section 7 discusses the results obtained and comparatively analyzes the performance of the methods, and finally, Section 8 concludes this study and addresses future research directions.

2. Background and Related Work

Currently, most of the initiatives, open source and private, related to digital preservation exist in the form of repositories that try to solve the preservation problem from a centralized, uniform point of view. That is the case, at the institutional level, of the networking for digital preservation developed by the International Federation of Library Associations and Institutions (IFLA) [7] or the Digital Preservation Program carried out by UNESCO [8]. By the dimension and continuous support of their communities and the size of the files stored, open-source initiatives stand out as the most relevant ones for institutions looking to implement long-term digital repositories. The most important ones are the Repository of Authentic Digital Objects (RODA) [9], the Dynamic Digital Repository (DSpace) [10], the Flexible and Extensible Digital Object Repository Architecture (FEDORA) [11], and EPrints Institutional Repository Software [12]. These platforms offer multiple storage features, and all of them implement the standard OAIS preservation model [13] and the Dublin Core metadata standard [14]. The OAIS model standard is widely accepted and utilized by various organizations and disciplines, both national and international, and was designed to ensure preservation. It is considered the optimum standard to create and maintain a digital repository over a long period of time. With respect to metadata, Dublin Core conforms a 15-element set of descriptors emerging from interdisciplinary and international consensus, and such elements are organized into categories such as provenance, authenticity, preservation activity, technical environment, or rights management. All these repositories and databases share the current main strategies for the preservation of digital information being implemented, which are replication or redundancy, auditing, refreshment, emulation, migration or conversion, encapsulation, and federation. All these strategies have a common goal: they are a set of actions or methods designed to ensure digital information remains usable over time; they are centralized and managed by preservation and metadata standard cores of each repository. The current state-of-the-art of centralized systems shares some common advantages: The integrity of digital objects is maximized, and data redundancy is minimized, as the single storing place of all the data also implies that a given set of data only has one primary record. This aids in the maintaining of data as accurately and as consistently as possible and enhances data reliability. The security needs of centralized systems are lower, as the single location of the digital objects is easier to protect against attacks. Single database designs are also simpler and more user-friendly than distributed environments, either for clients or administrators. Data kept in the same location is easier to maintain, organize, or mirror, and any update on the data is instantly received by the clients. On the other hand, centralized repositories are highly dependent on network connectivity. The slower the speed, the longer the time needed to access them, and bottlenecks can occur as a result of high traffic. Hardware failures may prove catastrophic in centralized systems. If a set of data is unexpectedly

lost, it is very hard to retrieve it again. As stated in [15], in order to be prepared for much larger volumes of digital objects to come in the future, it is not enough to make backups on storage devices: our actions must go further, elaborating distributed strategies and autonomous intelligent behaviors to share preservation efforts and, at the same time, democratize knowledge, making it available to everybody.

Currently, the available storage capacity of connected-to-Internet devices is superior to the space available at central repository structures. Decentralized structures offer many advantages: increased reliability, higher availability, higher performance levels, better scalability and sustainability, simplicity, and advanced data features. The main differences between centralized and distributed databases arise due to their respective basic characteristics. In a distributed database, all the information is stored at multiple physical locations, as distributed systems rely on replication and duplication within the multiple subdatabases in order to keep records up to date, and they are composed of multiple database files. As stated before, if data are lost in a centralized system, retrieving them would be much harder. If, however, data are lost in a distributed system, retrieving them would be very easy, because there is always a copy of the data in a different location of the database. One of the weaknesses of distributed environments, the excess of data redundancy, is, at the same time, one of its strengths in a data loss or hack attack scenario. Distributed control and management of digital objects is a rather complex task, and such a type of architectures needs specific Database Management Systems (DBMSs), making decentralized repositories harder to program, maintain, and update, and the geographically spread distribution tends to make them inefficient compared to simple centralized repositories. Lately, decentralized digital ecosystems, together with blockchain technologies have promoted the appearance of distributed storage systems where users participate in the sharing and storing of their own devices, Filecoin [16], Storj [17], and Sia [18] being the most notable examples. Following this philosophy, recent approaches search for infrastructure independence by involving high-level conceptual metadata models and knowledge representations as an integral part of digital objects and their associated migration processes. At this point, it is clear that decentralized systems are key components of a future DP global model where files respond, demand, and care for their own preservation.

Self-Preserving Digital Objects (SPDOs) constitute the basic pillar for solving the future scalability and sustainability concerns of current structure-centered repositories.

The very first evidence of SPDOs can be found in buckets [19]. Buckets represent aggregative, intelligent, WWW-accessible digital objects that were optimized for publishing in digital libraries, which existed within the Smart Objects, Dumb Archives (SODA) digital library model. Buckets implement the philosophy that information itself is more important than the systems used to store and access the information. Buckets were designed to imbue information objects with certain responsibilities, such as the display, dissemination, protection, and maintenance of their contents. Works in that line such as [20,21] leave the door open to digital objects that have embedded behavior to fight digital obsolescence with a promising degree of success.

Before 2000, there were a number of projects that had similar aggregation goals to SPDOs and buckets, namely the Warwick Framework containers [22] and the previously mentioned FEDORA. Interactions with FEDORA objects occur through a Common Object Request Broker Architecture (CORBA) interface. Multivalent documents [23] appear similar to buckets at first glance. However, the focus of multivalent documents is more on expressing and managing the relationships of differing semantic layers of a document, including language translations, derived metadata, and annotations. The AURORA architecture defines a framework for using container technology to encapsulate content, metadata, and usage [24], which is also developed in CORBA. Some of the mentioned projects are from the DP community, and others are from e-commerce and computational science. Most did not have the SODA-inspired motivation of freeing the information object from the control of a single server. The mobility and independence of buckets as SPDOs is not seen in any other DP projects. Most of the DP projects that focus on intelligence are

mainly centered on giving aid to the digital library user or creator, the intelligence being machine-to-human based. As a novelty in the current state-of-the-art, our SPDO paradigm is unique because the digital object itself is intelligent, providing machine-to-machine (i.e., SPDO-to-SPDO and SPDO-to-agentified/web services) intelligence.

Synergy works on computational ecologies [25] show how this approach might work: the SPDOs ask themselves how much preservation is necessary (appraise) and, according to a DP budget that would be regularly assigned to them, compete with each other for the preservation services.

There are also works on the information ecological approach to repositories, including the work in [26] that used the word ecology to evoke the sense of dynamism and fragility inherent in a biological systems, as well as in information systems. Currently, although there is no universally accepted concept of what exactly can be defined as an information society, we all agree that the character of information has transformed the way that we live today. Inside this scenario, the term *information ecology* relates ecological ideas to the highly dense and complex informational environment our society has turned into, viewing the informational space as an ecosystem [27] where interacting entities show collective intelligence and share knowledge. Thus, info-ecological approaches focus on knowledge or information organization and 2.0 approaches, but there is still no connection with DP. This article explores a step further into this connection as more intelligent digital objects need more intelligent and participative environments in which users actively cooperate and provide resources for DP.

The proposal presented in this article takes advantage of both centralized and decentralized systems to create a network environment where digital objects are gifted with collective intelligence and imprinted with a sense for self-preservation. The advantages of decentralization, such as scalability, data redundancy, and network performance accessibility, are combined with the utilization simplicity, efficiency, and reliability demonstrated by centralized databases.

3. The Self-Preserving Digital Objects Paradigm

The future is digital. The amount of information, official or legal documents, stored or simply transaction vouchers grows exponentially every day. The longevity of digital information is a concern that may not be obvious at first glance. While we all agree that digitalized information has many advantages over old printed information, such as the ease of duplication, transmission, and storage, digital data suffer unique longevity concerns that hard copies do not, including short lifecycles through multiple file formats [28]. Repositories, either centralized or decentralized, together with the technology used to program them are necessary for information management, but we think that the information content is more important than the system used to store it. Digital information should have the same long-term survivability prospects as traditional hard copy information and should be protected against Software Adoption Waves (SAWs). Digital objects and digital repository systems should evolve on independent paths in the search for the best design strategies. The problem, which is the focus of this article, is to set up a sustainable strategy to preserve digital information over time, ensuring accessibility, readability, and execution independently of the software and hardware used to access it. The world is transforming, from centralized structures and linear chains of command, to become a distributed environment, more flexible, agile, and adaptable. Digital structures such as the Internet are ahead in this transformation. Global networks are becoming not only a place to occasionally share and communicate, but to sustain a great part of our daily activities, either professional or leisure. Artificial intelligence is playing a great role in the whole process, currently, several major digital structures and services are being monitored and maintained by AI-based programs, some of them making extraordinary complex decisions. We present a plan to create, design, simulate, and test a software infrastructure that supports long-term digital preservation of data based on the idea that the digital data should have an active role in their own preservation. The presented article explores a new paradigm in the field of DP,

taking advantage of both decentralization and artificial intelligence, to provide the digital objects with the tools for self-preservation, a new context where digital objects share a part of the responsibility for preservation: given a set of rules, constraints, and resources, digital objects can move, replicate, and evolve to a higher-quality format inside a digital ecosystem, thus becoming SPDOs.

There are two main strategies for DP: preserving the technological environment (emulation) and overcoming the obsolescence of the file formats (migration). This article focuses on the second one to simulate the digital preservation activities of migration and copy. Migration is considered as a transformation of files in different formats, whereas copy is a replication of the file in the same format [29,30]. In the new framework, the behavior of SPDOs needs to be modeled in order to obtain the best preservation strategy. A design goal of the proposed paradigm is that digital objects be autonomous and active, separate from the server or repository where they are stored. Storage systems are similar to temporary living places, whereas the most important factor is the information contained in the digital object itself. SPDOs should be able to perform and respond to actions on their own and be active participants in their own state and existence. Making files intelligent opens the door for many potential applications. Thus, digital objects are programmed by means of selected CI algorithms with the mission, once distributed over a network of computers or devices, of their own long-term self-preservation, which entails being accessible and reproducible by users at any time in the future regardless of frequent technological changes due to software and hardware upgrades. For maximum autonomy, the proposed environment configuration allows files to contain all their code, data, and metadata. Related to self-sufficiency, the proposed preservation architecture makes digital objects fully mobile agents. That is, they can physically move from place to place in the network since they contain all the code, data, and support files they need. In order to demonstrate the feasibility of the proposal, the described environment was simulated using a specially designed software called Simulation Time Machine (TiM) for DP studies.

For the experimental simulation setup, migrated copies of SPDOs were created in various formats following a CI-based replication and migration strategy to try to guarantee their future persistence and survival in front of software and hardware changes and updates. At this point, we define a Software Adoption Wave (SAW) as big format changes that force digital objects to change their internal structure in order to remain accessible and readable by users. SAW refers to both software or hardware shifts, and the consequences of a SAW usually entails the *disappearance* of a percentage of the digital objects, as they become obsolete (unreadable or inaccessible) in the new technological paradigm shift after the SAW.

In our simulation, such a changing environment is represented by a network of interconnected nodes, each node representing sites. The connections between nodes represent mobility possibilities, opening the door to file migration and copy, resulting in a DP open network [31].

3.1. SPDO Behavior

According to previous definitions, SPDOs are digital objects equipped with the necessary tools and programmed for self-preservation. The proposed simulated environment considers SPDOs as different file types coexisting in various formats. Digital objects belong to particular nodes of the network, users, or open storage sites. In a given environment, digital objects can take various actions: they can make a copy, move to other nodes of the network, or remain at the site. Depending on the CI method used to program the SPDO behavior, different strategies and combinations of actions will arise, resulting in digital objects and their descendants moving through sites and making digital copies of themselves, aiming at self-preservation.

All the simulation experiments assumed that, according to [1], 75% of information is replicated every year, which means digital objects travel through the network, making copies of themselves. The copies keep links to predecessor files because, indeed, they

happen to be the same object. Each node representing a user's computer might only be able to read a specific format after suffering a SAW. Formats range from oldest to newest to simulate processing in a computer when the digital objects can only be accessed by special software installed in the computer (the site). The files in older formats become unreadable when they are no longer compatible with the new software versions installed on a given computer after several migrations. Those formats are taken as a representation of preweb and nonweb formats because there is no danger of forgetting how to read a file format that was in use after the web became available; there is a migration path forward.

3.2. Software Adoption Waves

A wave simulates a massive update of old software for file management (editing, transformation, etc.) to new software, normally linked to technology (hardware or software) or market shifts. Today, most of the volume occupied by digital information is obsolete. The life expected from media is short, and there are many digital objects stored on dead websites. If we take a look at which digital products and services are the most used, a large portion of them did not exist five years ago. A particular format of a given medium can be expected to become obsolete within no more than five years [28]. This is the reason why we assumed that a typical period for a software update is every five years, although longer or shorter periods could be proposed without loss of generality. Thus, for simulation-related purposes, this article fixed the SAWs occurrence at the same five-year pace.

Normally, a new version of software provides support for its previous software products, but this seldom exceeds three versions, so older versions lose their compatibility or retain it, but with losses in the quality and integrity of digital objects. Therefore, the estimated time a digital object is likely to be obsolete is 15 y (3×5 y). The total time of TiM experimental simulations was set to 35 y, time enough to analyze several SAWs and the performance reactions of the selected CI models. The total 35 y period was divided into two terms: short-term and mid-term. Short-term represents a 15 y period, that is a defined period of time where usage is predicted, but does not extend beyond the foreseeable future and/or until it becomes inaccessible because of changes in technology (the problems with accessing the digital objects start after 3 versions, 3 SAWs, or 15 years). Mid-term was set to 30 y (35 y in the simulation to see how the system responds after the last 30 y sixth SAW), that is continued access to digital materials beyond changes in technology for a long period of time, but not indefinitely.

If we assume that from now, the digital universe will nearly double itself every two years [1], the annual growth factor of the information is 1.41. With this growth factor, after 15 y, the volume of digital information that is created would $1.41^{15} = 180.99$-times the volume of the first year. Therefore, the volume of digital objects that have to remain accessible and readable after the 15 y period is 0.6% (1/180.99) of the total information after 15 y. We took $r = 0.6\%$, r being the percentage of original digital objects that remain from this time out of the total volume of information. Another assumption is that all new digital objects are created with the newest software available. This is represented in the formula of Equation (1), which calculates the percentage of digital objects of the network that remain (r) after three SAWs.

$$r = (1 - a)^3 \tag{1}$$

where a represents the percentage of digital objects that are affected in each SAW. Hence, the percentage of digital objects that remain is $1 - a$, to a power of three (one for each wave). The last assumption is that for each digital object, there are on average three copies of it made yearly. As stated in [32], from the total volume of data created every year, 25% of it is new, and the other 75% is a copy of existing data. Such an assumption is represented in Equation (2) as:

$$r = (1 - \frac{a}{4})^3 \tag{2}$$

Combining Equations (1) and (2), the percentage of digital objects of the network that remain after three SAWs is inside the range shown in Equation (3).

$$(1 - \frac{a}{4})^3 < r < (1 - a)^3 \quad (3)$$

Finally, assuming $r = 0.6\%$ (the volume of digital objects that remain after a 15 y period), the percentage of digital objects that are affected in each SAW a must be between 72% and 82%, as computed in Equation (4):

$$72\% < a < 82\% \quad (4)$$

Regarding specific choices of format shifts, these were selected randomly out of the new formats, and these format shifts were common for all the experiments performed. It is noteworthy that the environment itself would also be vulnerable to SAWs, but this is out of the scope of this article, as it focuses on the preservation of digital objects and their behavior.

3.3. The Network

The users and their computers conform the network scenario where SPDOs exist, move, and evolve. Users, sites, or repositories are connected among them asymmetrically. The resulting network is the living world for our SPDOs, and depending on the node, they can host, duplicate, or migrate from the current format to a newer one. The present article refers to repositories, users, or sites as nodes.

On the basis of previous works on the topology of network graphs for long-term storage of digital objects [20], the experimental study presented in this article focused on the Watts and Strogatz *small world* [33], as it is one of the most referenced network structures found in the literature. Watts and Strogatz's proposed network is characterized by nodes that are easily attainable in a few steps, and the nodes tend to cluster locally, to a much larger degree than if the networks were generated randomly. All TiM simulations used the same Watts and Strogatz topology, with $n = 1000$ (number of nodes), $p = 0.05$ (rewiring probability), and $k = 5$ (initial connectivity). A more realistic simulation would be to make the topology vary with time, but for simplicity, the experimentation was performed as a fixed topology. There are also recent works that presented real social network topologies and the interaction between the users obtained from Facebook, Google+, and Twitter, as shown in [34], but these social networks consist of many more nodes, making them unfeasible to reproduce in the current version of the developed TiM simulator.

The R package igraph function *watts.strogatz.game()* was selected to create the simulation environment, consisting of 1000 nodes. Tests were performed to ensure the resulting graph was simple and connected. The graph had a clustering coefficient equal to 0.48 and an average path length equal to 4.41.

3.4. The Measure of Expected Resilience

In order to measure SPDOs' preservation capabilities, we needed a way to quantify and measure the *health* of digital objects in terms of the diversity and number of accessible copies. If preservation is sharing, migrating to several formats, and duplicating into a given number of sites, it can be concluded that preservation success is tied to spreading, disorder, randomness, and uncertainty. These concepts drove us to think that entropy levels inside a finite network ecosystem would be a good measure for object diversity. Therefore, the Shannon entropy formula represented in Equation (5) was selected as a measure of the level of resilience inside the simulated system and was used for the evaluation of the results.

$$H(X) = -\sum_{i=1}^{n} p(x_i) log_2 p(x_i) \quad (5)$$

where n indicates the total number of possible outcomes x (copies) of variable X (object) and $p(x)$ its probability of happening. In the context of our article, the high entropy of a digital object $H(X)$ indicates the high resilience (preservation) after a SAW according to a given CI strategy for format migrations. As explained before, several copies existing in various formats provide better preservation guarantees and recovery capabilities in the case of a SAW. Equation (6) is the resultant application of Equation (5) to the DP context, where N is the total number of original digital objects, x represents the copies, j indicates the different formats where digital objects can migrate to (with a value ranging from 1–5), and $p(x_{i,j})$ is the percentage of the copies in format j among the total number of copies i of an original digital object. $\overline{H}(X)$ represents the average entropy (diversity) of original digital objects at any given time.

$$\overline{H}(X) = \frac{-\sum_{i=1}^{N}\left[\sum_{j=1}^{5} p(x_{i,j}) log_2 p(x_{i,j})\right]}{N} \quad (6)$$

Table 1 shows the entropy measures for three example cases. For demonstration purposes, files can exist in five different formats, Format 1 being the newest, to Format 5, the oldest. The column named *Copies* shows the number of digital objects available in the network for each format. As can be seen, more copies and more formats lead to the highest values of the entropy. Higher entropy represents higher resilience.

Table 1. Random examples of entropy as a measure of diversity.

Example	Formats	Copies	$p(x_{i,j})$	$p(x_{i,j}) log_2 p(x_{i,j})$	$H(X)$
Example 1	1	3	0.231	−0.4881	2.0758
	2	1	0.077	−0.2846	
	3	5	0.385	−0.5301	
	4	1	0.077	−0.2846	
	5	3	0.231	−0.4881	
Example 2	1	13	1	0	0
	2	0	0	0	
	3	0	0	0	
	4	0	0	0	
	5	0	0	0	
Example 3	1	0	0	0	0.9182
	2	1	0.333	−0.5283	
	3	2	0.667	−0.3899	
	4	0	0	0	
	5	0	0	0	

In Example 1, resilience is higher than in Example 3 and, at the same time, greater than in Example 2. The reason is due to the fact that, although Example 1 and Example 2 both have the same number of resultant copies (thirteen), the copies of Example 1 are more diversified among the five possibilities. More randomness means more *surprise* and thus more entropy. Furthermore, the resultant entropy of Example 2 is zero as the *uncertainty* inherent to the possible outcome formats is nonexistent because migration to Format 1 has a probability of happening equal to one.

4. Computational Intelligence Algorithms

The expression CI usually refers to the ability of a computer to learn a specific task from data or experimental observation. Even though it is commonly considered a synonym of soft computing, there is still no commonly accepted definition of computational intelligence. Generally, computational intelligence is a set of nature-inspired computational methodologies and approaches to address complex real-world problems for which mathe-

matical or traditional modeling can be useless for a few reasons: the processes might be too complex for mathematical reasoning; it might contain some uncertainties during the process; the process might simply be stochastic in nature. Such uncertainties in computer programming open a wide field of metaheuristic algorithms that try to simulate real-world phenomena. These algorithms can be classified into several categories, where evolution-based and swarm-intelligence-based methods take a prominent position. Evolutionary Computation (EC) involves combinatorial optimization mechanisms and is inspired by biological evolution, whereas Swarm Intelligence (SI) is based on the collective behavior of decentralized and self-organized systems. These techniques solve problems in an implicit way through collective behavior and cooperation between the actors in a system, and several recent examples of successful applications of these techniques can be found in the literature, alone or using hybrid approaches where two or more CI techniques are combined, regarding resource allocation [35], bioinformatics [36], energy management optimization [37], the Internet of Things [38], scheduling and routing [39], and social community network analysis [40], among others.

Features present in EC and SI systems make them good candidates to be used for optimizing SPDOs environments. Therefore, this article explores the behavior of three specific algorithms in a simulated environment: the Multipopulation Genetic Algorithm (MPGA) as an improved variant of an EC system and the Ant Colony Optimization (ACO) as the SI method. A third more recent methodology, called the Virus-Based Optimization Algorithm (VBA), is also studied in this article. VBA algorithms share features from both EC and SI. The VBA is an iteratively population-based method that imitates the behavior of viruses attacking a living cell. The number of viruses grows at each replication and is controlled by an immune system (a so-called *antivirus*) to prevent the explosive growth of the virus population. In computer programming language, VBAs behave similar to a computer worm. We think that these three nature-inspired methodologies have good synergies with our proposal for self-preserving digital objects. The next subsections give further details on each of the mentioned approaches studied.

4.1. Multipopulation Genetic Algorithm

The Genetic Algorithm (GA) [41] is one of the best evolutionary algorithms used to solve optimization problems. The GA is a metaheuristic inspired by the process of natural selection and belongs to the larger class of Evolutionary Algorithms (EAs). Genetic algorithms are commonly used to generate high-quality solutions to optimization and search problems by relying on biologically inspired operators such as mutation, crossover, and selection. In a genetic algorithm, a population of candidate solutions (called individuals, creatures, or phenotypes) to an optimization problem is evolved toward better solutions. Each candidate solution has a set of properties (its chromosomes or genotype) that can be mutated and altered.

The evolution usually starts from a population of randomly generated individuals and is an iterative process, with the population in each iteration called a generation. In each generation, the fitness of every individual in the population is evaluated; the fitness is usually the value of the objective function in the optimization problem being solved. The more fit individuals are stochastically selected from the current population, and each individual's genome is modified (recombined and possibly randomly mutated) to form a new generation. The new generation is then used in the next iteration of the algorithm, and the process keeps repeating till either a predefined number of generations has been reached or a good fitness level has been achieved.

For our approach, the selected method is a variant of GA called the multipopulation genetic algorithm. The MPGAs are usually used to overcome one of the most important limitations of GAs: premature convergence. To do so, MPGAs divide the population into several subpopulations (subpopulation number) with the same number of individuals (subpopulation size). In our experiment, every individual is a SPDO. Each site is a subpopulation, and the conjunction of the network is the multiple populations. According to

Figure 1, the algorithm starts by generating random populations of SPDOs in every site (subpopulation). After fitness evaluation of all individuals (or just a selection, depending on the method), the next population of *children* is obtained by combining two selected *parents* using the next operators:

- *Reproduction:* This performs the survival of the fittest within the subpopulation (at each site). There are many ways to achieve effective reproduction. One simple way is to select individuals for reproduction according to their fitness value. Individuals with higher fitness have a higher probability of being selected for mating (crossover) and subsequent genetic actions (exchange and mutation). In the DP context implementation, there is always a fashionable format that appears randomly during the simulation. Depending on the format that becomes fashionable, formats newer than the fashionable format will have a random fitness value between zero and one because, theoretically, their true value is unknown due to their novel nature. Older formats will have zero as the fitness value, and the random appearing fashionable format will have a fitness value of one;
- *Exchange:* Two individuals selected by the reproduction operator create two new individuals. In this implementation, there is no chromosome representation. The percentage of selected individuals for exchange range from 0.2% to 5% of the total number of SPDOs in the network with a good fitness value for reproduction. For each selected SPDO, an additional SPDO with a good fitness value from a random neighbor subpopulation is paired (parents). Then, a copy of each SPDO is created (child), and those copies are exchanged (subpopulation swap). This operation makes subpopulations (sites) interact, taking advantage of the multipopulation algorithm features;
- *Mutation:* This operation was implemented as a random migration of an SPDO to a different format. As illustrated in the GA flowchart of Figure 1, this operation is performed after the exchange operation. Once the digital copy of each SPDO is created, that copy is migrated to a random format, becoming a mutation of the recently created digital object. Again, the percentage of selected individuals for mutation range from 0.2% to 5% of the total number of SPDOs in the network that have been created and exchanged with the previous operation.

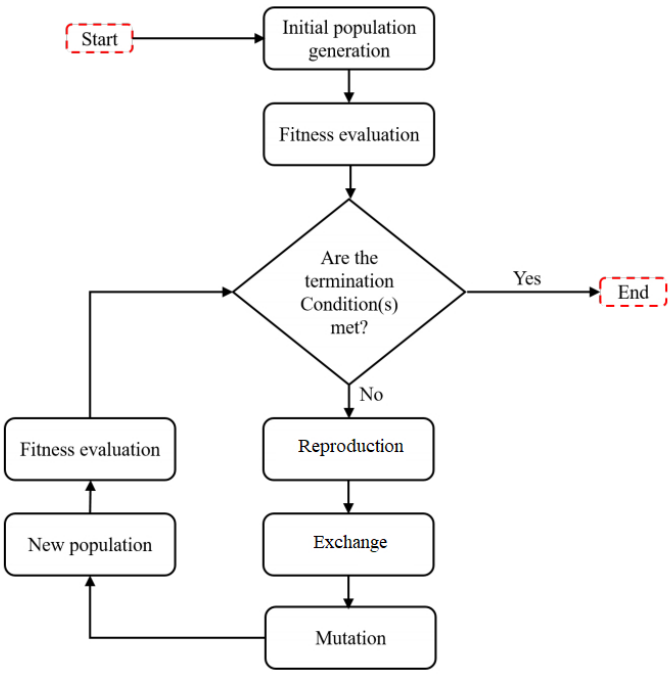

Figure 1. Genetic algorithm flowchart. Each subpopulation evolves according to the graph.

The combination of operations and fitness evaluation cycles grants the evolution of those SPDO individuals with the best strategies for format migrations, and at optimal or near-optimal levels we should expect evolved SPDOs with high expected preservation capabilities and resilience. The initial conditions and other considerations regarding TiM simulation parameters for MPGA are detailed in Section 5.

4.2. Ant Colony Optimization

This algorithm is a well-known swarm intelligence method, falling inside the category of metaheuristic optimization algorithms. Initially proposed by Marco Dorigo in 1992 in his Ph.D. thesis [42], the first algorithm aimed to search for an optimal path in a graph, based on the behavior of ants seeking a path between their colony and a source of food. The original idea has since diversified to solve a wider class of numerical problems, and as a result, several problems have emerged, drawing on various aspects of the behavior of ants. From a broader perspective, ACO performs a model-based search and shares some similarities with the estimation of distribution algorithms.

For the experiments presented in this article, the algorithm is structured as follows: SPDOs behave similarly to ants; the users represent the sources of food (needed to keep the copies of SPDOs alive), and users' computers (or sites of the network) are the habitats for SPDOs. As illustrated in Figure 2, the SPDOs (ants) seek their self-preservation by jumping over the habitats and replicate by making copies of themselves (descendants) in several formats. There is a limitation when making copies whereby SPDOs can only make a copy or be replicated in a certain percentage of nodes of the network (see the percentages in Table 2). In these nodes, there is food for the ants, and they are selected randomly every three years, which is an estimation of the average lifetime of a PC before being upgraded.

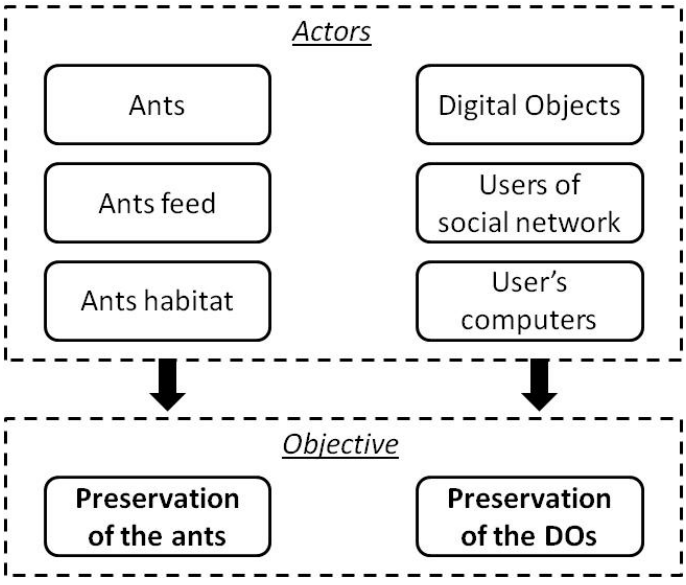

Figure 2. Ant colony model for digital preservation.

With regard to SPDO behaviors, at the beginning of the experiment the ants start to explore the graph (network). To do so, they initially choose nodes randomly until they find one with a pheromone trail. Then, the ant starts to copy its path back to its starting point. While backtracking, the ant drops pheromones. When a new route is started, ants calculate the probability of choosing an edge to continue its route. The probability of choosing an edge p_e in each step is computed as follows in Equation (7),

$$p_e = \frac{\pi_e * \eta_e}{\sum_{i=1}^{n} \pi_i * \eta_i} \tag{7}$$

where:

- π_e is the value of the pheromone at edge e;
- η_e is the quality of the route. This value can be estimated as $\eta_e = \frac{1}{d_e}$, where d_e is the length of the edge. In this simulation, the length is the number of nodes between two edges;
- n is a set of all the edges that the ant can use for its next step. It includes all the edges except the one for which we compute the probability p_e.

Table 2. Definition of the initial parameters used in the TiM simulations for every CI method.

Parameters	**Common to All**		
Total simulation time	35 years		
Equivalence of 1 simulation step	1 month		
Total waves	6		
Percentage of objects involved in a SAW	72% and 82%		
Number of objects associated with any user	1 to 5		
Preservation service for each user	100%		
Percentage of statistical stability	99.3%		
Random seed	123,456,789		
	MPGA	**ACO**	**VBA**
% of mutation % of exchange	0.2% to 5% 0.2% to 5%		
% of nodes of the network where ants can make copies		20%, 50%, 70% and 90%	
Period in which nodes are changed where ants can make copies		3 years	
Maximum number of copies that a digital object can have at each site			1

To simplify the application of the algorithm to the LTDP problem, some algorithm coefficients were discarded. Thus, the impact of the length of a finished route affecting ant decisions and the amount of pheromone given to an edge to amplify the impact were not taken into consideration for the presented experiments. Initial conditions and other considerations regarding TiM simulation parameters for ACO are detailed in Section 5.

4.3. Virus-Based Algorithm

The idea of a virus attacking a host cell has been previously proposed to enhance optimization power, which is nature-inspired as it shares common features with evolutionary systems [43]. The Virus-Based Algorithm (VBA) is a metaheuristic technique and population-based solution recommended to solve optimization problems in continuous domain problems. Although the development of VBA algorithms and software programming tools is relatively new compared to the rest of methods, there are some successful applications of such methodologies to solve problems such as the one presented in this article.

In general terms, the VBA's progression is quite similar to other EA methods. Here, the population of individuals are viruses, which aim to persist by infecting a living cell. Once individuals enter the cell, they will start replicating and alter the genetic material of the host. As a result, more viruses will be produced, and ultimately, the host cell will die. In this algorithm, the solution space is taken as the cell itself, and global optima can be found inside the cell. Many viruses can coexist within a host cell, and each virus represents a solution in the solution space. The viruses are divided into two categories: *strong* and *weak*, which correspond to the exploration and exploitation capability of the virus optimization algorithm. Strong viruses will have a high objective function value compared to weak

viruses, and they will replicate faster than them. The algorithm has mainly four types of phases: initialization, evaluation, replication, and updating, as depicted in Figure 3.

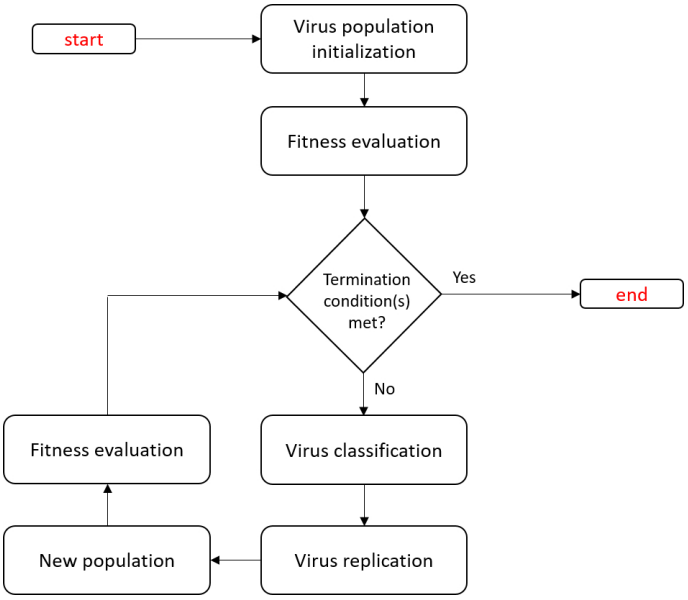

Figure 3. Virus-based flowchart.

An analogy for the digital object preservation problem presented in this article would be to consider the biological virus to have a behavior similar to a computer worm virus. There are several types of computer virus based on biological viruses or worms. A computer worm is a standalone malware computer program that replicates itself to spread to other computers. It often uses the computer network to spread itself. Worms do not infect files or damage them, but they replicate themselves so fast that the entire network may collapse. The behavior implemented in this article follows a worm pattern, but instead of a computer virus, we have an SPDO.

The main idea of the proposed approach is that SPDOs will act as viruses, moving through the entire network, selecting connections randomly as they attempt to spread as far away as possible as they replicate themselves. The limitations are: the capacity of the sites, as they can host a limited number of SPDOs in every step of the simulation, and the fact that each SPDO can only replicate once in each site without collapsing the network (the antivirus keeping the network stable). Initial conditions and other considerations regarding TiM simulation parameters for ACO are detailed in the Section 5.

5. Simulation Time Machine for DP Studies

In the last few years, the TECNIO Centre EASY has been developing a software tool for environment simulation of Multi-Agent Systems (MASs), the Simulation Time Machine (TiM) [44]. Initially conceived of as a simulated environment for running a wide number of agent-based ecosystems, today, it is being iterated to a simulation environment platform for DP studies for the SPDOs.

Figure 4 shows the framework that was applied in TiM. In this framework, the communication of the individuals with each other and with the environment is performed through JAVA methods. The *simulation* class can be executed under predefined setup conditions and contains all the actors within a list. All the individuals, our SPDOs, have a main method, in which they perform their activities in every step of the simulation. The platform, programmed in JAVA, was designed with the aims of:

- Studying CI methods and their application to the digital self-preservation paradigm;
- Finding the most efficient environments for SPDOs;
- Studying the network topologies and finding which architectures better support digital preservation contexts.

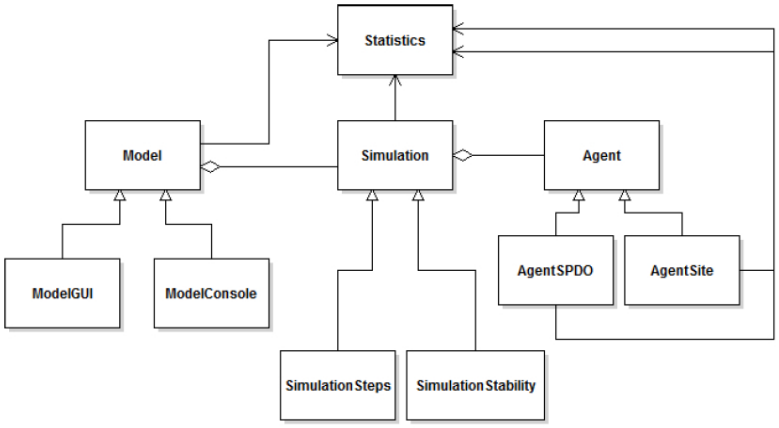

Figure 4. Excerpt of the simulation framework.

The TiM application has two running modes: graphical user interface mode and console mode. Both options give the user freedom to set personalized network topologies or select an existent one from a default folder. It also allows users to set up the initialization simulation parameters and to see the simulation progress in real time (through graphical user interface), as depicted in Figure 5. Executions in console mode need to obtain the initialization simulation parameters from an imported file.

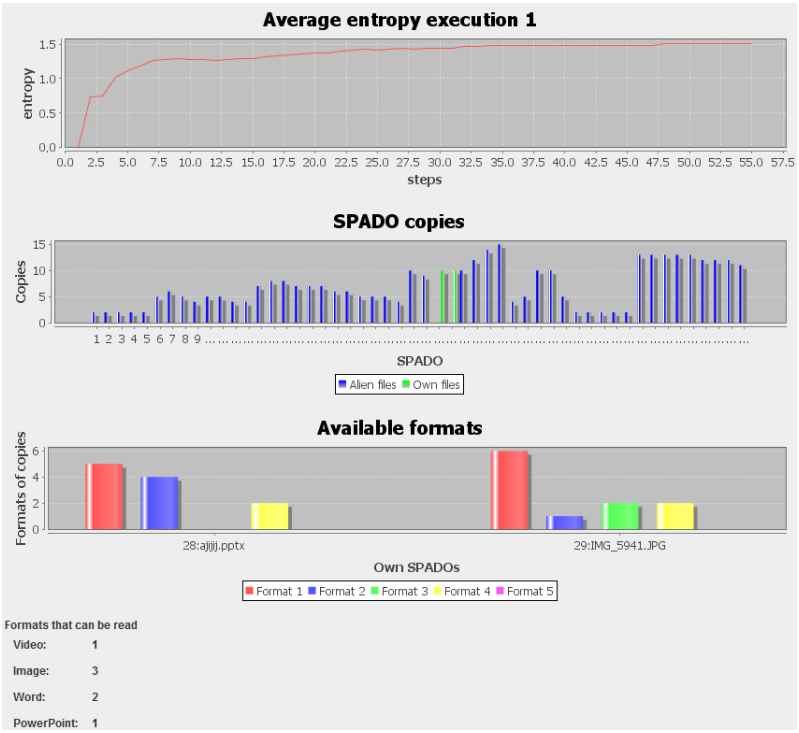

Figure 5. TiM graphical user interface for the system initialization and real-time execution control.

The results of the simulation are saved in a file. The statistics stored in the results file include:

- Initialization setup parameters for the simulation;
- Step entropy and the average entropy of all executions;
- Accessible and readable SPDOs in each execution and the final average;
- Site occupation in each execution and the final average;
- The number of copies and formats of each SPDO for each execution and the final average.

The TiM software is currently in development. Once finished, it will be open and free to use by the community. The TiM simulator was used as a simulation platform to test the behavior of SPDOs evolving under the rules proposed by the three nature-inspired methods described in Section 4, the MPGA, the ACO, and the VBA.

Experimental Setup

As detailed in Section 3.2, the TiM simulations ran for a 35 y period, with SAWs hitting every 5 y. A *month* was used as the temporal unit at every step for our simulations, and thus, there was a SAW every 60 steps (12 mos $\times$ 5 y), and the simulation time had a total length of 420 steps (12 mos $\times$ 35 y).At any given simulation step n, the average resulting entropy S_n was calculated after the differential steady entropy $\overline{H}(X)$ between one step and the previous one meeting a convergence stability requirement ε. The reliability of the experiment was measured using Equation (8), which defines $\overline{H}(X)_n$ as the average of the steady entropy in step n, such that the sum converges with an increasing number of experiments to yield a reliability of $1-\varepsilon = 99.3\%$.

$$S_n = \sum_{i=1}^{n} (\overline{H}(X)_n - \overline{H}(X)_{n-1}) < \varepsilon \tag{8}$$

Table 2 shows the initial parameter configuration to run on TiM for every CI algorithm tested.

These parameters were introduced as the initial setup conditions for each of the algorithms tested during the simulated experiments detailed in the next section of the article. The final selected values presented in the table above were tuned experimentally. All experimental tests were performed on a laptop with the following configuration: Intel(R) Core(TM) i7-4500U CPU @ 1.80 GHz and 8 GB of RAM memory. The simulations always took under 20 s to execute. Timing was not a constraint for the development of the article. In the future, if the complexity of the algorithms increases, better hardware settings would improve the simulation times.

6. Results

In the presented experimental study, we were interested in: (1) understanding the results obtained from each CI algorithm; (2) finding which CI algorithms provided the maximum resilience (better entropy); (3) finding which CI algorithms contributed the maximum recovery (effective resilience) after every SAW. In order to see how the behaviors based on the different CI proposals worked as solutions for DP, the results are presented as a concise numerical study based on the following items, which we considered key to characterize the obtained results:

- The presented experiments analyzed how the system responded to short- and mid-term simulations. The short-term was fixed as a period of 15 y (180 mos/180 simulation steps), and the mid-term was fixed as 30 y (360 mos/360 simulation steps). The simulations were extended to 35 y to see how the system responded after a 30 y period, as explained in Section 3.2;
- We analyzed the evolution of the entropy value as it is related to the existent degree diversity within the system, and studied the evolution of the number of copies of each digital object along simulations, and finally, the resilience obtained after each new SAW.

These items were evaluated individually for each CI algorithm presented in this article. After that, a comparison between them was also carried out. The executions were performed for the mid-term, applying 72% and 82% as the percentages of digital objects that were affected at each SAW (lower and upper ranges discussed in Section 3.2). For this, 72% was considered to be a mild SAW, and 82% was considered to be a severe SAW. The simulation results presented in the next sections showed, presented in a graphic form and for each methodology, the evolution of the average entropy S_n (Y-axis in the graphs)

over a simulation execution period of 420 steps, which corresponds to 35 y (X-axis in the graphs). The short- and mid-term are separated in the graphs by a black vertical line at the exact mark of 180 simulation steps (15 y). The measure of the average entropy S_n indicates, as was explained in Section 3.4, a measure of the preservation capabilities of existing digital objects: the higher the entropy, the higher is the expected resilience and the diversity of the digital objects.

6.1. Results on the Multipopulation Genetic Algorithm

In the simulations of the MPGA, four independent executions were carried out with four different percentages of exchange and mutation operators within the fixed range: 0.2% of the population, 1% of the population, 2% of the population, and 5% of the population; these simulations were repeated twice applying 72% and 82% as the percentages of digital objects that were affected at each SAW. Figure 6 shows the simulation results obtained for the MPGA method under severe SAWs of 82% of the digital objects' population.

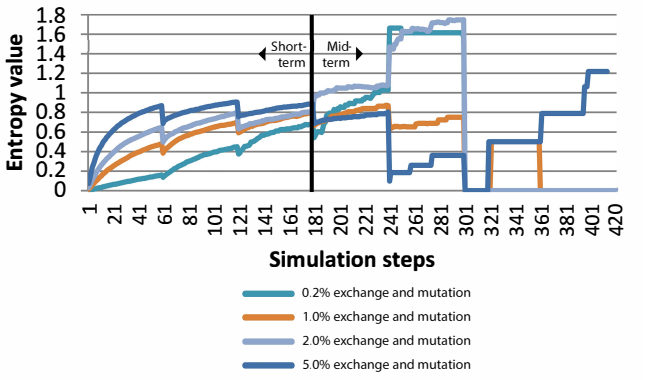

Figure 6. MPGA results applying severe SAWs of 82%.

As depicted in Figure 6, the plot illustrates the average resulting entropy for each simulation when the model was subject to severe SAWs for both the short- (1–15 y) and mid-term (15–35 y). If we take a closer look to the short-term results (Steps 1–180), the more exchanges and mutations, the higher the entropy that was achieved. However, after each SAW, the entropy values obtained with the different percentages of operations became closer. From a long-term point of view (Steps 181–420), one can see that after 35 y of simulation, the only execution that had a positive entropy was the one corresponding to an operation rate of 5%. After the fifth SAW (Simulation Step 300), the entropy of all executions fell to zero, and from here, the only execution able to recover and resist the sixth wave (Simulation Step 360) was the one with 5% exchange and mutation rates.

Figure 7 shows the results obtained in the case of a milder SAW of 72%. As can be appreciated, from the point of view of the short-term period, the results were similar to the ones obtained in Figure 6 with severe SAWs. The differences appeared in the mid-term analysis. Here, all the executions maintained positive entropy at the end of the last SAW (Simulation Step 360), except for the execution with a rate of 5% of the GA operators, which fell to zero after the last SAW. In general terms, the results showed that, in both the short- and mid-term, SPDOs were more likely to be preserved when more exchanges and mutations were performed.

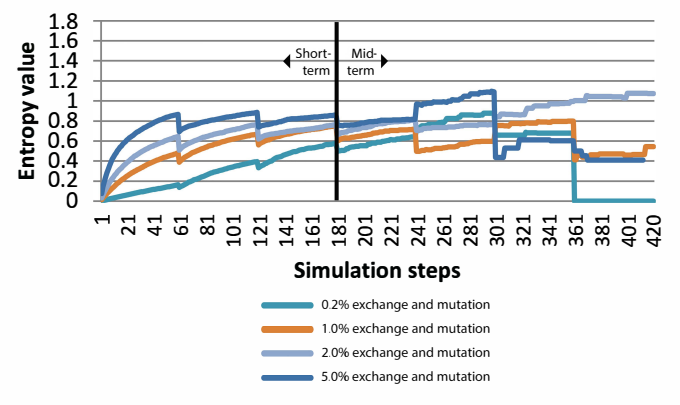

Figure 7. MPGA results applying mild SAWs of 72%.

6.2. Results on the Ant Colony Optimization Algorithm

For the ACO simulations, four runs were executed with different percentages of nodes (sites) in the network in which SPDOs could make copies of themselves: 20%, 50%, 70%, and 90%. These simulations were repeated twice applying 72% and 82% as the percentages of digital objects that were affected at each SAW. In Figure 8, the plot illustrates the average resulting entropy for each simulation when the model was subject to severe SAWs for the both short- (1–15 y) and mid-term (15–35 y).

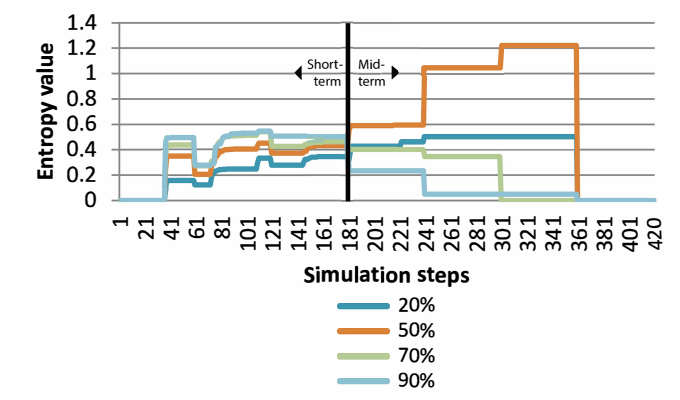

Figure 8. ACO results applying severe SAWs of 82%.

As can be appreciated in Figure 8, in the short-term (Steps 1–180), the higher the percentage of nodes in which SPDOs could make copies of themselves, the higher the average entropy value was. Similar to the MPGA, after each SAW, the entropy values obtained with the different percentages became closer. From the long-term point of view (Steps 181–420), by contrast, the entropy fell for all executions. When SPDOs were allowed to copy themselves in 20% and 50% of the nodes, both executions seemed to resist the SAWs and even grew in terms of resilience, but both of them abruptly fell to zero when the sixth and last SAW hit.

Figure 9 shows the results obtained in the case of a milder SAW of 72%. Analyzing the short-term period, it can be seen that the results were similar to the ones obtained in Figure 8, but the more resilient ones were no longer proportional to the percentages of available nodes; in this case, the best ones corresponded to allowing sites to copy 50%, 70%, 90%, and 20%, in that order. On the other hand, when applying mild SAWs, the results were much more encouraging in the mid-term, with all executions showing positive levels of average entropy after the last SAW. The best average entropy value was achieved by the lowest percentage of nodes in which SPDOs could make copies of themselves (20%).

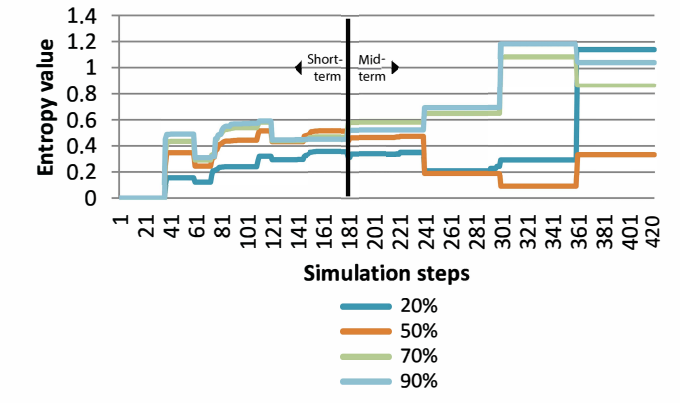

Figure 9. ACO results applying mild SAWs of 72%.

Despite the short-term simulations with severe SAWs showing that higher percentages of sites in which SPDOs could make copies had higher resilience, in the mid-term, those executions eventually fell, suffering from instability problems. It can also be appreciated that in the short-term and for mild SAWs, the lower percentages of nodes showed higher resilience, and such a tendency was amplified in the mid-term, where the 20% case showed the best performance. However, and generally speaking, the higher percentages of nodes in which SPDOs could make copies had better values of entropy during almost all of the time of the simulation.

6.3. Results on the Virus-Based Algorithm

For the VBA simulations, with no particular setup changes in the initialization conditions, only two executions were performed applying mild (72%) and severe (82%) percentages of digital objects that were affected at each SAW, and both of them can be compared directly in Figure 10. The plot illustrates the average resulting entropy for each simulation when the VBA model was subject to severe and mild SAWs for both the short- (1–15 y) and mid-term (15–35 y).

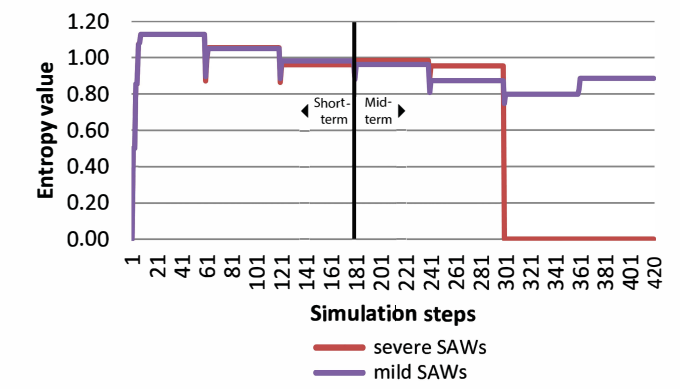

Figure 10. VBA results applying mild (72%) and severe (82%) SAWs.

As can be observed in the short-term (Steps 1–180), independent of the percentages of SAWs, the both executions reacted in the same way and showed good resilience in front of the SAWs despite the entropy value decreasing a bit after each SAW. The graph shows how the average entropy evolution grew rapidly at the beginning up to a value that was then maintained until there was a new SAW, losing entropy after the SAW, but recovering quickly till the next SAW. The plot has a flat shape due to the initialization constraint, which made the SPDOs unable to create more than one copy per site. Therefore, after each SAW, the entropy value was a little less than before because some SPDOs died, and sometimes, both the original copy and the descendants died, making this digital object unrecoverable.

In the mid-term (Steps 180–420), one can see that average good results of entropy were obtained as mild SAWs hit, whereas severe SAWs completely depleted SPDOs' existence at the fifth SAW, bringing the population to zero. The instability showed by the severe fifth SAW may suggest further experimentation with other SAW severity percentages between 72% and 82% and different constraint values as well.

6.4. Results on the Final Comparison of the MPGA, ACO, and VBA

In this section, the simulated results with all three nature-inspired algorithms are compared. The best executions of each algorithm were selected to be subject to severe and mild SAWs for both the short- (1 –15 y) and mid-term (15–35 y).

Figure 11 depicts the results obtained with a comparative analysis of the best execution of the three algorithms in a series of severe SAWs context. In the short-term, it can be observed that the value of the entropy, and therefore resilience, attained with the VBA method was the best of the three, followed very closely by the MPGA. At this term, the performance shown by the ACO algorithm, despite growing at a steady pace, was far behind the other two. Regarding the mid-term, the VBA started in the lead, but once the fourth SAW hit, the VBA's performance fell behind that obtained with ACO, which maintain the lead until the sixth and last SAW wave, then falling abruptly. The MPGA method, which grew steady from the fifth wave, ended up as the only algorithm with a positive average entropy at the end of the simulation, after the 35 y period. If we take a general overview of all three algorithms' behavior through the full simulation time, the best option is MPGA, as it was the only algorithm that maintained a positive entropy at the end of the simulation, despite having a short period of time after the fifth wave where it had no entropy value at all. It can be concluded that the MPGA algorithm is, on average, the most resilient of the three in the face of severe SAWs, both in the short- and mid-term.

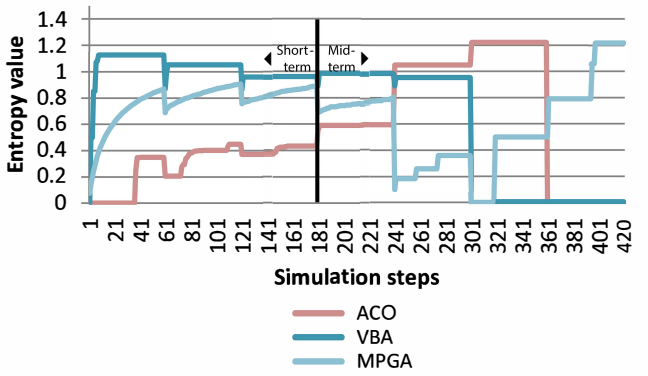

Figure 11. Algorithm comparison under severe (82%) SAWs.

Figure 12 depicts the results obtained with a comparative analysis of the best execution of the three algorithms in a series of mild SAWs context. In the short-term, the performance depicted by the three methods was quite similar to the one demonstrated in the severe context, only MPGA and ACO showing slightly lower entropy values. At the end of the short-term, the VBA algorithm was the one with the best resilience. Once the mid-term began, the best entropy values were still for the VBA, but after the fifth SAW, it was surpassed by the MPGA. Finally, once the sixth and last SAW hit, both the VBA and MPGA were surpassed by the ACO method. After observing the results obtained with mild SAWs along the mid-term, at the end of the simulation, the highest entropy value was obtained by ACO. On the contrary, the ACO model showed very poor performance during all the simulation, only showing good performance at the end of the simulation. In a DP context, SPDOs must be accessible and reproducible by users at any time, and, in the case of this particular simulation, both the VBA and MPGA offer a better solution, on average, than the ACO algorithm. It can be concluded that, for the case of mild SAW

waves, the VBA model was the one with the highest average entropy, indicating higher expected preservation capabilities.

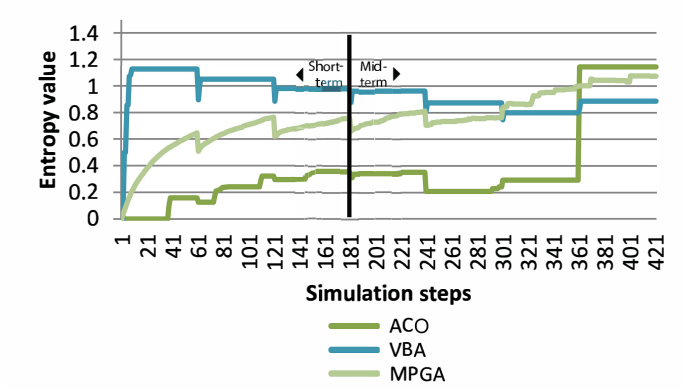

Figure 12. Algorithm comparison under mild (72%) SAWs.

The analysis of the comparative results demonstrated that, in the context of severe or mild SAWs, on average, the best algorithm was the MPGA, as it was the only one keeping positive values at the end of a period of a severe SAW chain, and it also maintained good average and positive ending values of entropy at the end of a period of a mild SAW chain.

7. Discussion

The Results Section showed several simulations with different self-preservation strategies (CI methods) using the TiM simulator in two different scenarios, one affected by waves of mild SAWs and another one affected by waves of severe SAWs. Furthermore, the results analyzed the response of the models in two different time frame windows, short-term (0–15 y) and mid-term (15–35 y). The evaluation of each CI algorithm tested in this paper gave the following conclusions:

- *Multipopulation genetic algorithm:* The more severe the SAW was, the better the algorithm responded. Several executions with different search parameters were performed with this algorithm. It was observed that in the mid-term, SPDOs were more likely to be preserved when there were higher percentages of mutations and exchanges. In general terms, despite being better in the presence of severe SAWs, the results obtained with milder SAWs were also very good;
- *Ant colony optimization:* The results showed that none of the execution models programmed for the ACO algorithm survived in the mid-term in a severe SAW scenario. Furthermore, the entropy values for the short-term were quite low. For mild SAWs, higher percentages of nodes in which SPDOs could make copies had better values of entropy during almost all the simulation. In general terms, the ACO algorithm, despite not having the best entropy values in the short-term simulations, responded very well in the mid-term, with the entropy value increasing at the fourth SAW and after;
- *Virus-based algorithm:* In the short-term, independently of the severity of the SAW, the VBA exhibited good values of entropy. On the one hand, for the long run, the model was very sensitive to severe SAWs, falling abruptly. The results were very good in the case of a mild chain of SAWs, with the algorithm showing high and stable levels of entropy during almost all the simulation time

Qualitatively, comparing the results obtained with the three CI algorithms tested in the short- and mid-term, under both severe and mild SAW episodes, the best performance, on average, corresponded to the MPGA. This model was the only one that maintained a positive/growing entropy as the proper sign of strong resilience, and it had the most resistant behavior in front of a severe SAW scenario, both in the short- and mid-term. In the case of milder SAWs, the highest entropy value at the end of the simulation was obtained by the ACO model, but its performance before arriving at the last SAW was very poor

(SPDOs must be as accessible and reproducible as possible by users at any given time), and in that case, again, the MPGA offered better overall results than ACO.

8. Conclusions and Future Work

The aim of this research article was to propose a novel object-centered paradigm to address the DP problem in modern society. In our approach, digital objects share part of the responsibility for self-preservation. Therefore, the behavior of digital objects needs to be modeled in order to obtain the best preservation strategy. Digital objects were programmed with the mission of their own long-term self-preservation, which entails being accessible and reproducible by users at any time in the future regardless of frequent technological changes due to software and hardware upgrades. Analyzing the performance of the proposed context required experimentation approaches of DP based on precise measures of preservability over an accurate range of situations deployed by means of Software Adoption Waves (SAWs). The model behaviors selected for the simulated experiments were nature-inspired computational intelligence algorithms. Thus, the following CI methods were implemented in the TiM simulator: Multipopulation Genetic Algorithm (MPGA), Ant Colony Optimization (ACO), and Virus-Based Algorithm (VBA). These artificial models were implemented, tested, and compared using a specifically designed software platform called Simulation Time Machine (TiM). TiM simulations are powered by a network structure as an environment that enables the behavior of the digital objects under the policy of *preservation is to share*. Digital objects are mainly files, and the best strategy for those files to persist is through migration and copy to overcome the obsolescence of the file formats. The migration strategy of a digital object is designed as an adaptation of the file into a different format to be distributed over the network in an attempt to be preserved. The copy strategy is designed as a replication of the same file in the same file format in another site.

Three CI algorithms were tested, in the short- and mid-term, under both severe and mild SAW episodes. The selected CI methods were studied taking into account two basic indicators: how well they performed in terms of resilience (diversity) and how well they behaved in terms of stability or sensitivity to the severity of SAWs. The results showed that the best performance, on average, corresponded to the MPGA. The simulated results obtained demonstrated that a digital preservation environment with objects programmed for self-preservation is possible. The results presented here are a first step towards creating real structured environments where models coexist under a predefined system of rules and constraints. The problem needs a full characterization of the desirable properties that could be required from a CI model for its application to DP through the newly proposed object-centered self-preservation paradigm. This article did not intend to focus on CI research, nor graph design or software architectures, as we see all these fields as tools for the study of strategies and behaviors for the self-preservation of digital objects. The results corroborated that nature-inspired models offer a good initial approach and open the door to test other AI-based behaviors.

Future work will include more variations in the initialization setup of the selected CI algorithms, such as adding more operators to the MPGA or implementing a better strategy to select the best options for network nodes in which ants can make copies in ACO. Regarding the VBA, it would be interesting to add a feature that allows testing the number of copies that a digital object can make at each node (each SPDO and its descendants), to determine the number for which the best results (in terms of preservation) would be obtained and the number of copies at which the entropy starts to collapse. Further research will also look ahead to better exploit the features of the selected CI models, in order to reduce their observed sensitivity to the SAW intensity.

With respect to the TiM simulator, we are currently working on more robust versions able to extend the long-term goal (100+ y). Furthermore, we are currently preparing the simulator to automatically run simulations for all the severity range (from 72% to 82%), so we can directly evaluate the performance of the tested models for all the range at once.

Extensive experimentation is planned ahead in the future. With more robust results and knowledge on the table, we plan to combine the properties from different models to obtain the best advantages of each algorithm and develop a new behavior resulting from the combination of various methods.

Author Contributions: Conceptualization, A.E.-F. and J.L.d.l.R.; Data curation, A.E.-F. and J.L.d.l.R.; Formal analysis, A.E.-F. and J.L.d.l.R.; Funding acquisition, A.E.-F. and J.L.d.l.R.; Investigation, A.E-F. and J.L.d.l.R.; Methodology, A.E.-F. and J.L.d.l.R.; Project administration, A.E.-F. and J.L.d.l.R.; Resources, A.E.-F. and J.L.d.l.R.; Software, A.E.-F. and J.L.d.l.R.; Supervision, A.E.-F. and J.L.d.l.R.; Validation, A.E.-F. and J.L.d.l.R.; Visualization, A.E.-F. and J.L.d.l.R.; Writing—original draft, A.E.-F. and J.L.d.l.R.; Writing—review & editing, A.E.-F. and J.L.d.l.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the PRESERVA 2019 PROD 00024 and VoteVote DEMOC00001 of the AGAUR.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

LTDP	Long-Term Digital Preservation
DP	Digital Preservation
IDC	International Data Corporation
USW	Unsupervised Small World
CI	Computational Intelligence
EC	Evolutionary Computation
SI	Swarm Intelligence
SPDO	Self-Preserving Digital Object
MAS	Multi-Agent System
SODA	Smart Objects, Dumb Archives
FEDORA	Flexible and Extensible Digital Object Repository Architecture
CORBA	Common Object Request Broker Architecture
TiM	Simulation Time Machine
SAW	Software Adoption Wave
MPGA	Multipopulation Genetic Algorithm
ACO	Ant Colony Optimization
VBA	Virus-Based Algorithm
GA	Genetic Algorithm
EA	Evolutionary Algorithm

References

1. The Digital Universe in 2020: Big Data, Bigger Digital Shadows, and Biggest Growth in the Far East. Available online: https://www.speicherguide.de/download/dokus/IDC-Digital-Universe-Studie-iView-11.12.pdf (accessed on 28 June 2021).
2. Library of Congress. Available online: https://www.digitalpreservation.gov/ (accessed on 12 August 2021).
3. Netpreserve. Available online: https://netpreserve.org/ (accessed on 12 August 2021).
4. Cartledge, C.L.; Nelson, M.L. When Should I Make Preservation Copies of Myself? *Digit. Libr.* **2014**, 109–118. [CrossRef]
5. Olvera, J.A. Digital Preservation: A New Approach from Computational Intelligence. *Bull. IEEE Tech. Comm. Digit. Libr.* **2013**, *9*, 22–26. Available online: https://bulletin.jcdl.org/Bulletin/v9n2/papers/olvera.pdf (accessed on 25 August 2021).
6. Fogel, L.J.; Owens, A.J.; Walsh, M.J. *Artificial Intelligence through Simulated Evolution*, 1st ed.; John Wiley & Sons: Hoboken, NJ, USA, 1966; ISBN 978-0471265160.
7. Networking for Digital Preservation: Current Practice in 15 National Libraries. Available online: https://www.ifla.org/ES/publications/ifla-publications-series-119 (accessed on 11 August 2021).
8. UNESCO: Digital Preservation Programme. Available online: https://en.unesco.org/themes/information-preservation/digital-heritage/digital-preservation-programmes (accessed on 11 August 2021).

9. Ramalho, J.; Ferreira, M.; Faria, L.; Castro, R.; Barbedo, F.; Corujo, L. RODA and CRiB a Service-Oriented Digital Repository. In Proceedings of the Fifth International Conference on Preservation of Digital Objects, London, UK, 29–30 September 2008. Available online: http://repositorium.uminho.pt/bitstream/1822/8226/1/RodaAndCrib.pdf (accessed on 25 August 2021).
10. Smith, M.; Bass, M.; McClellan, G.; Tansley, R.; Barton, M.; Branschofsky, M.; Stuve, D.; Walker, J.H. DSpace: An Open Source Dynamic Digital Repository. *D-Lib Mag.* **2003**, *9* [CrossRef]
11. Daniel, R.; Lagoze, C. Distributed active relationships in the Warwick framework. In Proceedings of the Second IEEE Metadata Workshop, Silver Spring, MD, USA, 16–17 September 1997. Available online: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.50.1685&rep=rep1&type=pdf (accessed on 25 August 2021).
12. Beazley, M.R. Eprints Institutional Repository Software: A Review. *Partnersh. Can. J. Libr. Inf. Pract. Res.* **2011**, *5*. [CrossRef]
13. Giaretta, D. Introduction to OAIS Concepts and Terminology. In *Advanced Digital Preservation*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 13–30. [CrossRef]
14. Weibel, S.; Kunze, J.; Lagoze, C.; Wolf, M. Dublin Core Metadata for Resource Discovery. *RFC* **1998**, *2413*, 1–8. [CrossRef]
15. Digital Preservation Coaltion. Digital Preservation Handbook. Available online: https://www.dpconline.org/docs/digital-preservation-handbook2/1552-dp-handbook-digital-preservation-briefing/file (accessed on 12 August 2021).
16. Filecoin Is a Decentralized Storage Network Designed to Store Humanity's Most Important Information. Available online: https://filecoin.io/ (accessed on 12 August 2021).
17. Storj: Decentralized Cloud Storage. Available online: https://www.storj.io/ (accessed on 12 August 2021).
18. Sia: Decentralized Storage for the Post-Cloud World. Available online: https://sia.tech/ (accessed on 12 August 2021).
19. Nelson, M. Buckets: Smart Objects for Digital Libraries. Ph.D. Thesis, Old Dominion University, Norfolk, VI, USA, 2001. [CrossRef]
20. Cartledge, C.L.; Nelson, M.L. Analysis of graphs for digital preservation suitability. In Proceedings of the 21st ACM Conference on Hypertext and Hypermedia, Toronto, ON, Canada, 13–16 June 2010; pp. 109–118. [CrossRef]
21. De la Rosa, J.L.; Olvera, J.A. First Studies on Self-Preserving Digital Objects. In Proceedings of the 15th International Conference of the Catalan Association of Artificial Intelligence, Alacant, Spain, 25–26 October 2012; pp. 213–222. [CrossRef]
22. Lagoze, C.; Lynch, C.A.; Daniel, R. *The Warwick Framework: A Container Architecture for Aggregating Sets of Metadata*; Technical Report TR-96-1593; Cornell University: Ithaca, NY, USA, 1996; ISSN 1082-9873.
23. Phelps, T.; Wilensky, R. Multivalent documents. *Commun. Acm* **2000**, *43*, 83–90. [CrossRef]
24. Marazakis, M.; Papadakis, D.; Papadakis, S.A. A framework for the encapsulation of value-added services in digital objects. In Proceedings of the 2nd European Conference on Research and Advanced Technology for Digital Libraries, Crete, Greece, 21–23 September 1998; pp. 75–94; ISSN 0302-9743.
25. De la Rosa, J.L.; Hormazábal, N.; Aciar, S.; Lopardo, G.; Trias, A.; Montaner, M. A Negotiation Style Recommender Based on Computational Ecology in Open Negotiation Environments. *IEEE Trans. Ind. Electron.* **2011**, *58*, 2073–2085. [CrossRef]
26. Nardi, B.; O'Day, V. *Information Ecologies—Using Technology with Heart*, 1st ed.; MIT Press: London, UK, 1999; ISBN 978-0262640428.
27. Garcia-Marco, F.J. Libraries in the digital ecology: Reflections and trends. *Electron. Libr.* **2011**, *29*, 16. [CrossRef]
28. Rothenberg, J. Ensuring the Longevity of Digital Information. *Int. J. Leg. Inf.* **1998**, *26*, 1–22. [CrossRef]
29. Garret, J.; Waters, D. *Preserving Digital Information: Report of the Task Force on Archiving of Digital Information*; The Commission on Preservation and Access and The Research Libraries Group Inc.: Washington, DC, USA, 1996; ISBN 978-1887334501.
30. Rothenberg, J. *Avoiding Technological Quicksand: Finding a Viable Technical Foundation for Digital Preservation*, 1st ed.; Council on Library and Information Resources: Arlington, VA, USA, 1999; ISBN 1-887334-63-7.
31. Jin, X.; Jiang, J.; de la Rosa, J.L. PROTAGE: Long-Term Digital Preservation Based on Intelligent Agents and Web Services. *ERCIM News* **2010**, *80*, 15–16; ISSN 0926-4981.
32. De la Rosa, J.L.; Trias, A. (IIiA Research Report 2010, University of Girona, Girona, Catalonia, Spain). Agents and Social Networks Environments for Digital Preservation. 2011. Available online: https://www.scribd.com/document/101207916/Research-Report-2010 (accessed on 25 August 2021).
33. Watts, D.J.; Strogatz, S.H. Collective dynamics of 'small-world' networks. *Nature* **1998**, *393*, 440–442. [CrossRef] [PubMed]
34. McAuley, J.; Leskovec, J. Learning to Discover Social Circles in Ego Networks. In Proceedings of the NIPS'12: 25th International Conference on Neural Information Processing Systems, Lake Tahoe, NV, USA, 3–8 December 2012; Pereira, F., Burgers, C.J.C., Bottou, L., Weinberger, K.Q., Eds.; Curran Associates Inc.: Red Hook, NY, USA, 2012; Volume 1, pp. 539–547. Available online: https://proceedings.neurips.cc/paper/2012/file/7a614fd06c325499f1680b9896beedeb-Paper.pdf (accessed on 25 August 2021).
35. Ari, A.A.A.; Gueroui, A.; Titouna, C.; Thiare, O.; Aliouat, Z. Resource allocation scheme for 5G C-RAN: A Swarm Intelligence based approach. *Comput. Netw.* **2019**, *165*, 106957. [CrossRef]
36. Hallen, M.A.; Donald, B.R. Protein design by provable algorithms. *Commun. ACM* **2019**, *62*, 76. [CrossRef] [PubMed]
37. Rodemann, T. A Comparison of Different Many-Objective Optimization Algorithms for Energy System Optimization. In *Applications of Evolutionary Computation*; Kaufmann, P., Castillo, P.A., Eds.; Springer Nature: Cham, Switzerland, 2019; pp. 3–18. [CrossRef]
38. Zedadra, O.; Guerrieri, A.; Jouandeau, N.; Spezzano, G.; Seridi, H.; Fortino, G. Swarm Intelligence and IoT-Based Smart Cities: A Review. In *The Internet of Things for Smart Urban Ecosystems*; Cicirelli, F., Ed.; Springer International Publishing: Cham, Switzerland, 2019; pp. 177–200. [CrossRef]

39. Lin, Y.; Hu, Y. Residential consumer-centric demand-side management based on energy disaggregation-piloting constrained swarm intelligence: Towards edge computing. *Sensors* **2018**, *18*, 1365. [CrossRef] [PubMed]
40. Lyu, C.; Shi, Y.; Sun, L. A Novel Local Community Detection Method Using Evolutionary Computation. *IEEE Trans. Cybern.* **2021**, *51*, 3348–3360. [CrossRef] [PubMed]
41. Holland, J. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*, 1st ed.; University of Michigan Press: Ann Arbor, MI, USA, 1975; ISBN 978-0262275552.
42. Dorigo, M. Optimization, Learning and Natural Algorithms. Ph.D. Thesis, Politecnico di Milano, Milan, Italy, 1992.
43. Liang, Y.C.; Juarez, J.R.C. A novel metaheuristic for continuous optimization problems: Virus optimization algorithm. *Eng. Optim.* **2016**, *48*, 73–93. [CrossRef]
44. Olvera, J.A.; de la Rosa, J.L. Time Machine: Projecting the Digital Assets onto the Future Simulation Environment. In *Advances in Practical Applications of Agents, Multi-Agent Systems, and Sustainability: The PAAMS Collection, Proceedings of the PAAMS 2015, Salamanca, Spain, 3–4 June 2015*; Demazeau, Y., Decker, K., Bajo Pérez, J., de la Prieta, F., Eds.; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2015; Volume 9086, pp. 175–186. [CrossRef]