When Should I Make Preservation Copies of Myself?

Charles L. Cartledge
Old Dominion University
Department of Computer Science
Norfolk, VA 23529 USA
ccartled@cs.odu.edu

Michael L. Nelson
Old Dominion University
Department of Computer Science
Norfolk, VA 23529 USA
mln@cs.odu.edu

ABSTRACT

We investigate how different replication policies ranging from least aggressive to most aggressive affect the level of preservation achieved by autonomic processes used by web objects (WOs). Based on simulations of small-world graphs of WOs created using the Unsupervised Small-World algorithm, we report quantitative and qualitative results for graphs ranging in order from 10 to 5000 WOs. Our results show that a moderately aggressive replication policy makes the best use of distributed host resources by not causing spikes in CPU resources nor spikes in LAN activity while meeting preservation goals.

Categories and Subject Descriptors

H.3.7 [**Information Storage and Retrieval**]: Systems issues

General Terms

Algorithms, Design, Experimentation

1. MOTIVATION

Much of our current cultural heritage exists only in digital format and digital preservation approaches rely on the long term commitment of individuals, institutions and companies to preserve this heritage. The length of time that an individual will be engaged in preservation activities is, by definition, limited to their lifetime (and probably just the middle part of that life). Even those few years may be longer than institutions and companies would be willing to undertake digital preservation. Institutions and companies may cease to exist or be unwilling or unable to meet their original preservation commitments. If this happens then the digital files and their information (our heritage) may become irretrievably lost. The acknowledgment that much of our personal and cultural heritage exists only in digital format, and the recognition that there is a real risk of total loss through accident [1] or change in business goals [2] has been recognized in academic reports and papers [3] and is starting to be recognized in the popular press [4, 5]. While recognizing that companies and institutions claim that they will preserve web objects for a long time, there are acknowledgments that this is a service offered at a cost, and that the cost must be paid by someone willing to have the data preserved.

Our motivation is to change the focus from preservation services administered by repositories or institutions (a repository centric perspective) to one where the data preserves itself (a data centric perspective). We continue to investigate this data-centric perspective through the use of the Unsupervised Small-World (USW) graph creation algorithm [6, 7, 8, 9] where we have shown that web objects instrumented with just a few rules can form into small-world graphs. The focus of this work is to augment the prior work of forming networks between disparate WOs with the capability of individual WOs to create a number of copies of themselves for preservation purposes. Additionally, we extend this body of work by focusing on determining when copies should be created during the USW process and what are the communication impacts of different preservation policies.

2. RELATED WORK

This work is at the convergence of digital library repositories and network theory. To provide the context of understanding the contributions of this research, we first briefly review the status of how objects are stored in repositories, as well as the nature and types of various networks or graphs.

2.1 Repositories

Repositories range from theoretical to ready-to-download. They include frameworks or architectural proposals such as SAV [10]. Others, are middleware systems, ready to be the core repository technology in a local deployment, like Fedora [11]. Some systems are complete and ready to deploy. These include DSpace [12], sponsored by MIT and HP Laboratories, LOCKSS [13], sponsored by the Stanford University Libraries and aDORe [14] sponsored by the LANL Research Laboratory. DSpace is an institutional repository, intended to archive the intellectual output of a university's faculty and students. LOCKSS allows libraries to create "dark archives" of publishers' websites. If the publishers' contents are lost, the dark archives are activated and the content is available again. CLOCKSS (Controlled Lots of Copies Keeps Stuff Safe) is a not-for-profit organization that is the embodiment of LOCKSS for that content whose originators are no longer available [15]. The total risk of loss using LOCKSS is mitigated through many sites archiving content of their own choosing.

Depending on an institution's requirements, the systems described above can be quite attractive. But there is an implicit assumption in any repository system: that there is a person, community, or institution that exists to tend to the repository. What happens when the responsible organization no longer exists? There are repository trading and synchronization provisions (e.g., [16]), but most are specific to a particular repository architecture.

Cooperative File Systems (CFS) [17], Internet Backplane Protocol (IBP) [18], Storage Resource Broker (SRB) [19] and OceanStore [20] are among several generic network storage systems and APIs that have also been proposed. CFS and OceanStore rely on distributed hash tables and an overlay network to locate content in the Internet. Systems with such additional levels of shared infrastructure have not been widely deployed. IBP and SRB are more traditional in their repository design and have enjoyed greater deployment. SRB (and its follow-on, iRODs [21]) has a user community similar in size to LOCKSS and Fedora.

Each of the approaches listed above inherently rely on human and institution intervention in the digital preservation activities of refreshing and migration [22, 23]. Digital preservation activities of emulation and metadata attachment are outside our context in this paper. Over time humans die and their personal archives can become lost, institutions may lose funding or have a change in ownership and therefore be unable to continue their preservation activities. As the amount of digital data continues to grow (at potentially an exponential rate), the organizational and human cost to keep up with traditional approaches can become overwhelming. An alternative approach is to revisit the definition of a WO and to incorporate into that definition the idea that the WO is empowered to make replication copies of itself for the purposes of preservation.

We have proposed placing the preservation requirements on the web objects and imbuing them with internal controls and data to enable them to undertake their own preservation. We have proposed fundamentals of these ideas and their mechanisms in [6, 7, 8, 9]. We continue to expand on these ideas by focusing on when a web object (WO) should endeavor to make preservation copies of itself. The timing of when to create these preservation copies has impacts on host CPU and LAN bandwidth utilization.

2.2 Graph Construction

Our approach for the construction of a small-world network of WOs for self preservation is different than others have used or proposed. We define a small-world graph as one that has a high *clustering coefficient* when compared to a randomly created graph and an average path length is proportional to the number of nodes in the graph [24]. The Watts-Strogatz approach to constructing such a graph is to take a lattice graph of some degree k and order n and perturb the links to create a graph with small-world characteristics. Some approaches make connections between nodes (or WOs) based on the proportion of the destination node's degree count [25, 26, 27], a kind of preferential attachment or fitness policy. Yet another type of approach takes an existing graph and then grows a small-world by the addition of new links [28, 29]. Or, by connecting a node to a fixed number of vertices based on their degree [30], or even creating a small-world graph from a random one [31]. Newman provides a survey of small-world graph construction techniques in [32]. Our USW approach, can use preferential attachment to select the first node when adding a new node to an existing graph, but after the first node selection, the USW algorithm controls where the node fits into the graph and how many edges are created to other nodes in the system. USW is the only small-world graph creation algorithm that we know of where connections are made between nodes based only on information that the node gleans from the existing graph prior to making its first connection.

3. SELF-PRESERVING WEB OBJECTS

We consider web objects (WOs) to be in the tradition of Kahn-Wilensky and related implementations [33, 34]. For the research presented here, the WO implementation is not specified; WOs are postulated to have the capability to carry a "payload" (e.g., one or more HTML, PDF, images, or other files), associated descriptive, structural, and administrative metadata, and the necessary code for initiating HTTP connections, and making copies of itself to other repositories on the web as per the rules described below (section 3.2).

This paper expands the theories from our previous work and informs our reference implementation [35] using the HTTP mailbox to provide WO to WO communication [36].

3.1 Various Perspectives of a Collection of WOs

A collection of WOs can be viewed from a graph theoretical perspective and be made up of vertices and undirected edges. Each WO is a vertex and each link is an undirected edge. Each WO contains the identities of WOs that are in its k-neighborhood ($k = 1$) and each WO uncovers the identities of its k-neighborhood through a discovery process based on WO's local knowledge of the graph.

A collection of WOs exist as a series of bits on a physical medium and require a finite amount of space. WOs communicate between themselves using a local LAN infrastructure when they are activated (a user browses to them, or as the result of a local maintenance activity). A system administrator will be willing to allocate effectively an infinite amount of space to locally created WOs, but only a finite amount of space to remotely created WOs that are attempting to put their preservation copies on local resources.

Just as there are orthogonal views of a collection of WOs; within a simulation there can be different views of time. In our event driven simulation, a simulation event is equivalent to time $S_{\mathrm{e}} \equiv S_{\mathrm{t}}$. A time slice $T_{\mathrm{slice}} = 10S_{\mathrm{e}}$. A time step $T_{\mathrm{step}} = S_{\mathrm{e}}3500 + T_{\mathrm{slice}} * \mathrm{step}$.

3.2 Flocking for Preservation

Craig Reynolds's seminal paper on "boids" (his term for bird-like objects) [37], demonstrated that three simple rules were sufficient to simulate the complex behaviors of schools of fish, flocks of birds, herds of animals and the like. The rules themselves are simple and yet the behaviors that emerge are complex and realistic. The remarkable feature about these rules is that they are scale-free and knowing the entire order of the network is not required. We believe these rules can be adapted to create self-preserving WOs with similarly complex emergent behaviors. The transcription of Reynolds's rules from a boid to a WO perspective are discussed below. *Collision avoidance* WOs flocking to a new repository cannot overwrite each other (collide in physical storage), nor collide in namespaces (have the same URI). This is orthogonal to the naming mechanism used: URIs, URN handles, DOIs, globally unique identifiers (GUIDS) or content addressable naming schemes [38].

Velocity matching All members of a herd, or school, or flock move at roughly the same speed. With boids, the idea is to travel the same speed as your neighbors. Interpreting velocity as resource consumption (i.e., storage space) enables this rule to be applied to a WO environment. Specifically, a WO should try to consume as much, and only as much, storage as everyone else. In resource-rich environments (lots of storage space available on lots of hosts), making as many copies of yourself as you would like is easy. When storage becomes scarce, this becomes more difficult. WOs must be able to delete copies of themselves from different archives to make room for late arriving WOs in low-storage situations. WOs will never delete the last copy of themselves to make room for new WOs, but they will delete copies of themselves to come down from a soft threshold (e.g., 10 copies) down to a hard threshold (e.g., 3). When resources become plentiful again, new copies can be made.

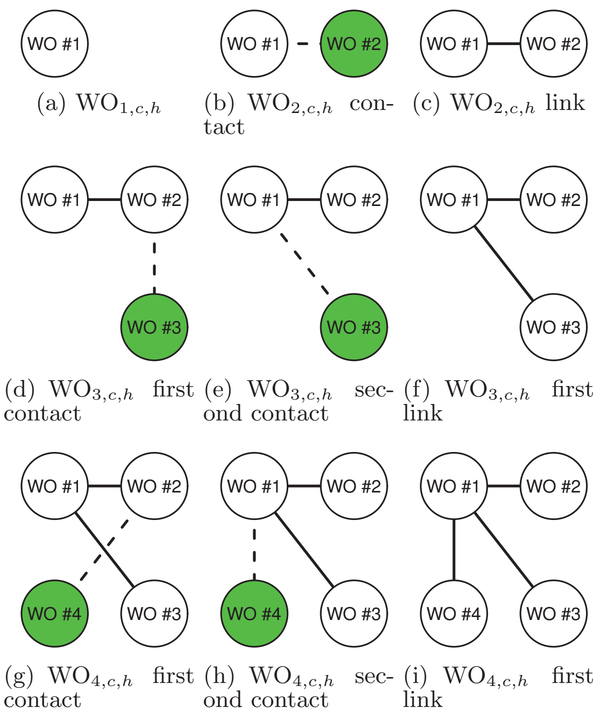
Flock centering For boids means staying near (but not colliding with) other flockmates. We interpret this in a manner similar to *velocity matching*, with WOs attempting to stay near other WOs as they make copies of themselves at new repositories. In essence, when a WO learns of a new repository and makes a copy of itself there, it should tell the other WOs it knows so they will have the opportunity to make copies of themselves at the new location. Announcing the location of a new repository will thus cause WOs at other repositories that have not reached their soft threshold to create copies that "flow" to the new repository.

At the macro level; in much the same way that flocks self-navigate to new locations that have the resources they need, we envision WOs self-preserving in a loose confederation of cooperating archives each with varying levels of resources and availability. Making copies in new repositories is performed in an opportunistic model, within the guidelines imbued in the WOs at creation time. From time to time an archivist may steer the entire collection (or parts of it) to new archives, but for the most part the WOs replicate themselves.

3.3 USW Graph Creation

We introduce some terminology to discuss how WOs can self-arrange. A *graph* is a composite object composed of two types of objects: *nodes* and *edges*. An *edge* connects exactly two *nodes* and the edge can be directional or not. The nodes directly connected to particular WO (i.e., they share an edge) are considered its friends. A *Family* is all WOs that are replicas of each other. A *Parent* is the family member that was first inserted into the graph and is responsible for ensuring that enough family members are created to meet its preservation goals.

Based on a review of graph structures, their characteristics and attributes, small-world graphs appear to be the most practical choice for minimizing a graph's size, communication costs and construction effort. Small-world graphs also emulate natural processes and occur often in nature and human endeavors, whereas regular and random graphs are relatively infrequent.



(a) $WO_{1,c,h}$ (b) $WO_{2,c,h}$ contact (c) $WO_{2,c,h}$ link

(d) $WO_{3,c,h}$ first contact (e) $WO_{3,c,h}$ second contact (f) $WO_{3,c,h}$ first link

(g) $WO_{4,c,h}$ first contact (h) $WO_{4,c,h}$ second contact (i) $WO_{4,c,h}$ first link

Figure 1: The USW growth algorithm with 4 WOs. The "wandering" WO symbol is filled. The dashed lines are communications and the solid lines are friendship links.

A *wandering* WO is introduced to an existing WO in the graph. The wanderer gets a list of friends from the WO it is introduced to. If the *wandering* WO does not form a friendship link with the initial WO, it will select another candidate WO that it has learned of from all its conversations with other WOs it has encountered. When the *wandering* WO finally makes a friendship link, it will then look back at all the WOs that it did not connect with as well as some that it was intending to communicate with and make friendship links with some of them. This process with 4 WOs is shown in Figure 1.

Friendship links are separate from HTML navigation links (i.e., `<link>` instead of `<a>` HTML elements). These links serve as a way for WOs to send messages from one to another, such as when new storage locations are available or message concerning the scope and migration of file formats (cf. the semi-automated alert system described in Panic [39]). Friendship links support the replication process. If a WO needs to replicate itself and it has a friend that lives on a different host and if there is room on that host for an additional WO then the WO can replicate itself onto the new host. Replication is how a family grows from a single copy (a parent) to multiple copies (a family).

A WO's family members will be spread across a collection of hosts. A complete description of a WO's position in a family structure and the host that it is living on is given by $WO_{n,c,h}$. Where:

$$\text{definitions: } \begin{cases} c_{\text{soft}} = \text{min. preservation copies} \\ c_{\text{hard}} = \text{max. preservation copies} \\ n_{\max} = \text{max. WOs} \\ h_{\max} = \text{max. hosts} \\ h_{\text{cap}} = \text{host capacity} \end{cases}$$

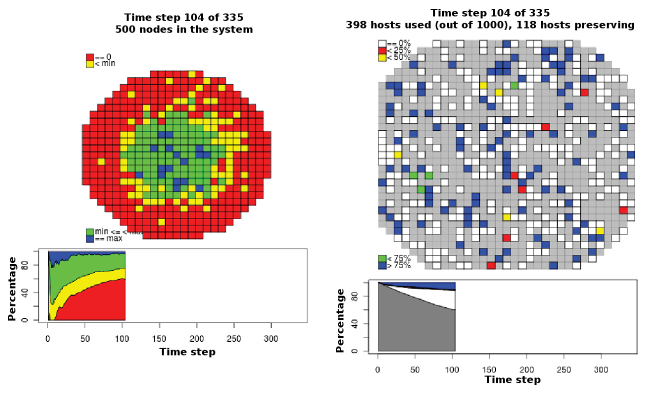


Figure 2: A snapshot of the least aggressive replication policy. WOs are shown on the left and hosts are shown on the right. The colors show the state of the WO's preservation copies, or host's preservation capacity used at the time of the measurement. Under each circular plot is a T_{step} histogram. Above each circular plot is a status line showing T_{step}, how many WOs are in the system or how many hosts are active and preserving data.

$$n, c, h \text{ constraints: } \begin{cases} n = 1, \ldots, n_{\text{max}} \\ c = 0, \ldots, c_{\text{hard}} \\ h = 1, \ldots, h_{\text{max}} \end{cases}$$

$$\text{subject to: } \begin{cases} (n, h) \text{ unique } \forall \text{ n and } \forall \text{ h} \\ c = \begin{cases} 0 & \text{if parent WO} \\ > 0 & \text{otherwise} \end{cases} \\ 0 \leq c_{\text{soft}} \leq c_{\text{hard}} \end{cases}$$

4. POLICIES FOR SELF-REPLICATION

We model three different replication policies to quantify and qualify their effects on the system as measured in two different areas. The first being how effective the replication policy is at having as many WOs as possible achieve c_{hard} copies. The second area is the communication costs associated with each replication policy as the system grows.

We focus on the following replication policies (assuming that the WO has defined values for c_{soft} and c_{hard}):

1. **Least aggressive** — a WO will only make a single replication copy at a time, regardless of how many copies are needed and how many opportunities are available to the WO at a particular time. It will continue to make single copies until it reaches c_{hard}.

2. **Moderately aggressive** — a WO will make as many copies as it can to reach c_{soft} when it makes its first connection. There upon it will fall back to *least aggressive* behavior.

3. **Most aggressive** — a WO will make as many copies as it can to reach c_{hard} at every opportunity.

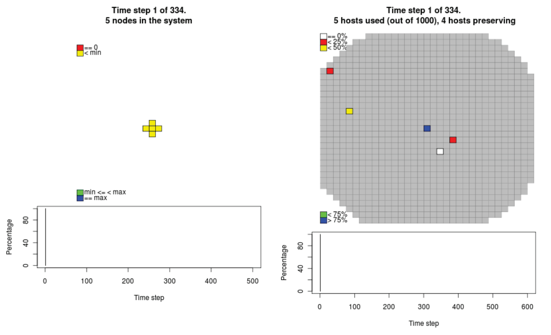
Figure 2 serves as a legend for the sub-figures in Figures 3 and 4 and shows WO and host replication status as a function of T_{step}. Figure 2 is divided into four areas. The left half of Figure 2 shows WO related data, while the right half shows host data. WOs are created and added sequentially to the model. In Figure 2, WOs are added in a spiral fashion starting at the center of the left-hand plot, and newer WOs are plotted in a circular manner from the center (see Figure 3). This presentation is much the same as the rings of a tree, in that the oldest are in center and the youngest are on the outer edge.

The preservation status of a WO is reported by the color assigned to the WO. Initially the WO has $c = 0$ copies and is colored red. As the WO creates copies, the color changes to yellow. When the WO reaches c_{soft}, the color changes to green. When c_{hard} is reached, the WO turns blue. The rules of the model (based on our interpretation of Reynolds's "boids") permit the killing of one WO's replication copies for the sake of creating a copy of a WO that needs to reach its c_{soft} (i.e., if a $\text{WO}_{1,c,h}$ has more than its c_{soft} and $\text{WO}_{2,c,h}$ has not reached its c_{soft}, then $\text{WO}_{1,c,h}$ will sacrifice one of its copies so that the other WO can move closer to c_{soft}). Sacrificing a preservation copy for the betterment of the whole is the embodiment of *velocity matching*. The effect of this behavior is that a WO can change color from red to yellow to green and then possibly to blue. If the WO changes to blue, it might oscillate between green and blue as its preservation copies oscillate between c_{soft} and c_{hard}. A WO will never sacrifice a copy if it has not exceeded its c_{soft}. The histogram under the WO circular plot shows the percentage of WOs in each of the different preservation copy states as a function of T_{step}.

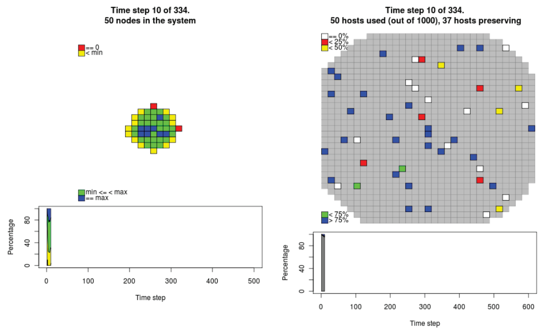
The host preservation utilization status is shown in the right half of Figure 2. The universe of possible hosts is constant and is represented by the entire right half plot. Hosts that are not being used are shown in grey. The placement of the host in the figure is based on the host's sequential number in the model. Those hosts that are used are drawn in one of five colors. If the host is used in the model, but is not hosting any preservation copies then it is colored white. If less than 25% of the host's capacity is used then it is colored red. Similarly, it is yellow if less than 50% is used, green if less than 75% and blue if greater than 75%. The histogram on the host's side shows the percentage of the hosts that are in any of the particular states.

The model has n_{max}=500, c_{soft}=3, c_{hard}= 5, h_{max}= 1000, h_{cap}= 5. The model runs until it reaches a steady state. A steady state is defined as: all WOs are unable to locate candidate hosts on which to store preservation copies. Steady state is reached at different times based on the replication policy. In all cases, all n_{max} WOs have been introduced into the model by T_{step} = 100. c_{soft}=3 was chosen as a lower limit in order to have at least 0.999% probability that one or more WO would remain after 1 year based on random measured loss rate of 11% in the first year [40]. Losses can come from failures, or threats [41], and may be compensated by using techniques based from self-preserving digital objects (SPDOs) [42], or an outside process such as a Crew Intelligence System [43]. A future area of work could be to include the ability of WOs to modify their responses to attack, thereby enhancing their collective behavior [44].

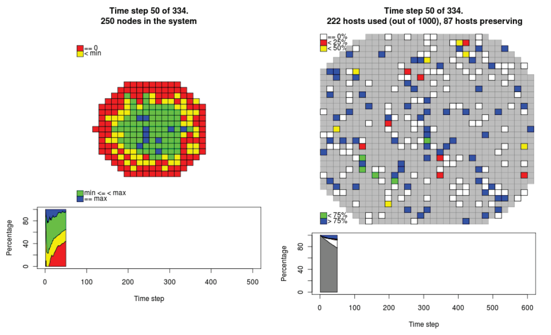
The initial WO is plotted in the center of the left hand upper quadrant of each composite, Figure 3(a) shows the first 5 WOs in the system. The one in the center is the oldest WO, while the others are younger. Younger WOs are plotted further and further from the eldest WO in much the same way as new tree growth is added to the outer edge of the trunk. The five WOs currently in the system (see Figure 3(a)), live on hosts in the system. Hosts can live



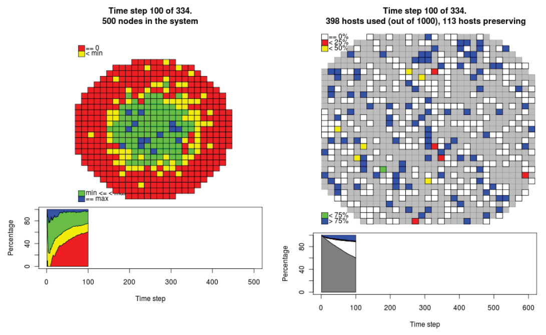
(a) $T_{\text{step}} = 1$



(b) $T_{\text{step}} = 10$



(c) $T_{\text{step}} = 50$



(d) $T_{\text{step}} = 100$

Figure 3: The growth of a $n_{\text{max}} = 500$ WO system captured at various timesteps. The left half of each sub-figure shows the "tree ring" growth of the WO's portion of the system. The WO and host histograms show the percentage of WO and hosts that are in their respective states as a function of time. All WOs have been created and assigned to a host by $T_{\text{step}} = 100$.

anywhere on the network and where a particular host lives does not carry any additional information. Plotting of the hosts in the same manner as the WOs (i.e., tree rings) could even be misleading because the assignment of WOs to hosts can be a completely random process. The hosts in Figure 3 have a finite store capacity that their respective system administrators have allocated to the preservation of copies of "foreign" WOs. Foreign copies are copies of WOs that originated on another host are are being preserved on the local host.

At any point in time during the simulation, there will likely be a difference in the number of preservation copies that the WOs want to create and the preservation capacity of all the hosts. Reynolds's rules attempt to balance these two requirements over time. Figure 3(a) indicates that the WOs have each made some number of copies (they are colored yellow vice red) and those copies are spread across some of the hosts in a non-even manner. One host has used all its capacity (as shown in blue), while one has not used any (as shown in white). The remaining hosts have used something in between those two extremes (they are yellow and red). Below both the WO "tree ring presentation" and the host "field" is a histogram showing how the system evolves over time. In Figure 3(a), the histograms do not show too much information because the figure shows the $T_{\text{slice}} = 1$ of system growth.

In Figure 3(b), $T_{\text{slice}} = 10$. The tree ring growth of the WOs is becoming more apparent. Older WOs have had more opportunities to make preservation copies of themselves, therefore there is more green and blue in the center of the WO plot. Many of the hosts are have reached h_{cap}, as indicated by the number of blue hosts. The histograms are starting to become filled with data. The WO histogram is starting to show that the percentage of the WOs that have made some, but not all their preservation copies (those in yellow) is starting to grow, while the percentage of those that have reached their goals is lessening. The hosts histogram is starting to show that the percentage of the hosts that have been discovered and added to the system (the grey area), is starting to decrease. A WO will be local to exactly one host. A host may have more than one WO local to it. A WO will not put a preservation copy on any host that it lives on, or that already has a preservation copy of itself.

In Figure 3(c), $T_{\text{slice}} = 50$. The tree ring presentation of the WO success at preservation is becoming more pronounced. Younger WOs are struggling to make copies, while the old ones are maintaining their copies. More of the hosts are being brought into the system (the percentage of grey hosts is decreasing), but a significant percentage of the hosts are not being used for preservation (those shown in white).

In Figure 3(d), $T_{\text{slice}} = 100$. All WOs have been introduced into the system. The tree ring preservation effect is still evident, and some of the new WOs have been fortunate enough to make some number of preservation copies (as shown by the yellow markers in the sea of red). The percentage of hosts that are still not preserving any WOs is still significant, and the percentage of hosts that have reached $h_{\text{cap}}$is holding constant. The system will continue to evolve until it reaches a steady state, when those WOs that have preserved as many copies of themselves as they can be based on their knowledge of hosts that have excess preservation capacity. The final time slice for this particular graph is shown in Figure 4(a).

5. REPLICATION POLICIES

5.1 Preservation Status When the System Reaches Steady State

Figure 4 shows the steady state condition of the same system using the three different replication policies. All WOs have been introduced into the system by $T_{\text{step}} = 100$ (as shown by the "kink" in the percentage of hosts that are used histogram). Each replication policy resulted in a significantly different time to reach a steady state. A steady state in the system is achieved when the WOs have made as many preservation copies as they are able to based on the number of friends that they have acquired when the system was in a growth phase and the number of unique hosts that those friends live on. The WOs are programmed to attempt to achieve between 3 and 5 preservation copies, while the hosts are limited to 5 have a preservation capacity of 5. The hosts have enough preservation capacity to accommodate the preservation needs of all the WOs. If the WO can locate enough unique hosts via its friends, then it will be able to meet its preservation goals.

The least aggressive policy reaches steady state after T_{step} = 334 (see Figure 4(a)) and there are a significant percentage of the WOs that have not been able to make any preservation copies (as shown by the lower-most (red) band in the histogram. As shown in the node half of the figure, many of the hosts are not preserving any WOs and those hosts that are preserving have reached their capacity.

The moderately aggressive policy reaches steady state after $T_{\text{step}} = 554$ (see Figure 4(b)). Prior to $T_{\text{step}} = 100$, most of the WOs have made most of their preservation copies. After $T_{\text{step}} = 100$, the percentage achieving c_{hard} slowly increases until steady state at $T_{\text{step}} = 554$. The hosts' preservation capacity is used by the WOs in the system almost as quickly as the hosts come on line. This is indicated by the very narrow white region between the unused host region and the totally used region. At steady state, only a very few of the hosts have not been totally used (as shown by the few host usage squares that are neither blue or grey).

The most aggressive policy reaches steady state after T_{step} = 300 (see Figure 4(c)). Close examination of the host histograms in Figures 4(b) and 4(c) show almost identical behavior both prior to $T_{\text{step}} = 100$ and at steady state. Comparing the host usage plot in the two figures show that slightly more hosts have unused capacity based on a most aggressive policy than a moderately aggressive policy (390 versus 397). Based on n_{max} WOs in the system, the difference between the two policies host under utilization does not appear to be significant.

5.2 Communication Phases While the System Strives to Reach Steady State

From the WO's perspective, there are two distinct phases of communication. The first is when the WO is *wandering* through the graph and collecting information from WOs that are already connected into the graph, called the *growth* phase. The second is after the WO is connected into the graph and based on the current replication policy has made some number of preservation copies, called the *maintenance* phase. During the growth phase, the WO is aggressively communicating with other WOs. While in the maintenance phase, the WO is responding to queries and communications from other WOs. This change in communication patterns occurs at $T_{\text{step}} = 100$ in Figure 4. $T_{\text{step}} = 100$ in Figure 4 corresponds to approximately $S_{\text{t}} = 3500$ in Figure 5. Figure 5 shows the communications for 2 different WOs and the system in total as a function of the replication policy. $\text{WO}_{1,c,h}$ and $\text{WO}_{250,c,h}$ were selected to represent the messaging profiles of all WOs in order to see if the profile changes as a function of when a WO enters the system. Time in Figure 5 runs until $S_{\text{t}} = 15000$ and messages are counted in bins sized to 100 simulation events.

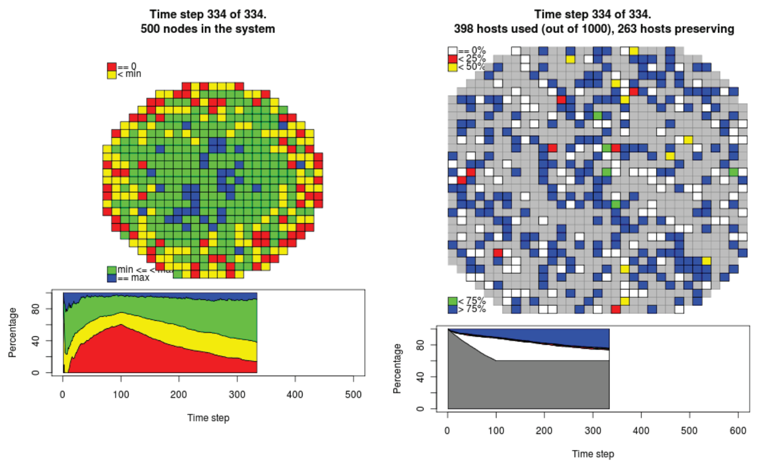
Looking at figures 5(a), 5(b), 5(d), 5(e), 5(g) and 5(h), there is a marked difference in the communication curves between $\text{WO}_{1,c,h}$ and $\text{WO}_{250,c,h}$. These curves (with only minor differences) are consistent across all replication policies. $\text{WO}_{1,c,h}$ (the earliest WO introduced into the system), sends a rather modest number of messages $O(2n)$ to WOs that are also in the system as $\text{WO}_{1,c,h}$ attempts to create preservation copies. Under the least aggressive policy (see Figure 5(a)), $\text{WO}_{1,c,h}$ sends a few messages per time bin until the system enters the maintenance phase. The number of messages sent during the moderately aggressive policy is nominally the same (see Figure 5(d)). While the most aggressive policy results in messages for just a couple of time bins and then virtually no messages are sent (see Figure 5(g)). Regardless of the replication policy, the number of messages that $\text{WO}_{1,c,h}$ receives is about the same.

Comparing the message curves for $\text{WO}_{1,c,h}$ and $\text{WO}_{250,c,h}$ indicates that the system discovered by the later WO is very different than the one discovered by the earliest WO. The late arriving node has more than enough opportunities to satisfy its preservation goals within the $T_{\text{slice}} = 0$. $\text{WO}_{250,c,h}$ sends all of its messages in one time bin and virtually nothing thereafter. This behavior is constant across all replication policies and indicates that the late arriving WOs are able to connect with another WO in very short order (within one time bin) and almost immediately enter into the maintenance phase of their existence. The maintenance phase of the system corresponds to a combination of the *velocity matching* and *flocking centering*.

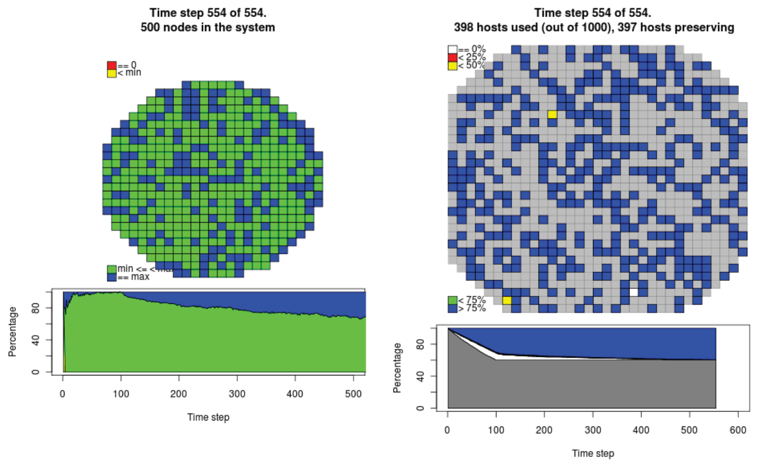
The system is in a growth phase from about $T_{\text{slice}} = 1500$ to $T_{\text{slice}} = 3500$ as shown by the rising curves in the "Sum of all WOs" sub-figures. During the growth phase, the *wandering* node is sending and receiving a lot of messages while attempting to make its initial connection into the graph. After $T_{\text{slice}} = 3500$, the system is in a maintenance phase when the system is attempting to balance the preservation needs of the WOs with the capacity of the hosts. Comparing the messages curves for the entire system Figures 5(c), 5(f) and 5(i) shows that there is no qualitative difference between the number of messages sent and received in the system based on replication policy. The nuances of the message curves for early WOs is lost as the order and size of the system increases.

5.3 Messages Sent and Received as the System Grows in Size

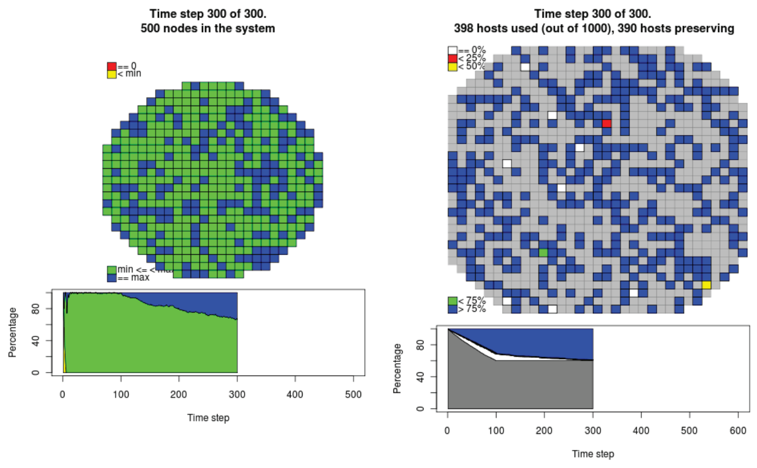
Figures 4 and 5 showed the efficacy and communication costs associated with a system with $n_{\text{max}} = 500$ and $h_{\text{max}} =$



(a) Least aggressive replication policy. System stabilization at $T_{\text{step}} = 334$.

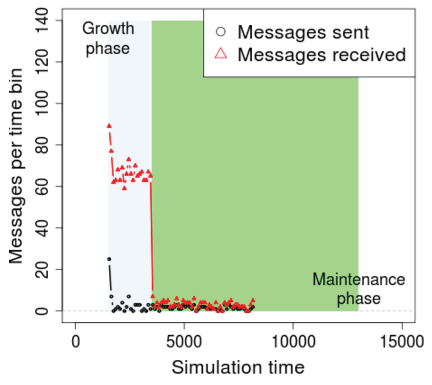


(b) Moderately aggressive replication policy. System stabilization $T_{\text{step}} = 554$.

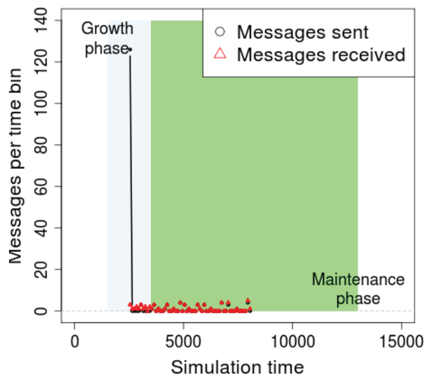


(c) Most aggressive replication policy. System stabilization at $T_{\text{step}} = 300$.

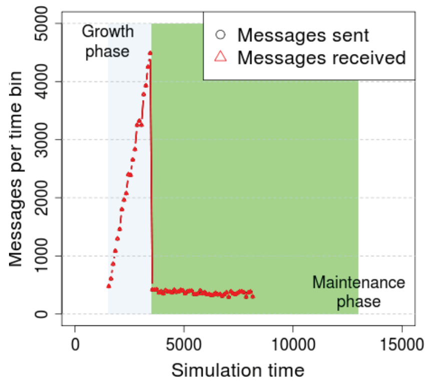
Figure 4: Time lapsed comparison of different replication policies. Using a moderately aggressive policy results in a higher percentage of WOs meeting their preservation goals sooner and makes more efficient use of limited host resources sooner.



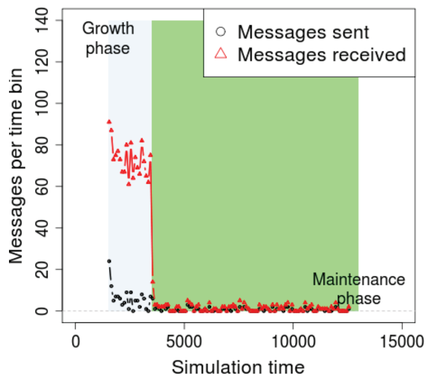
(a) $\mathrm{WO}_{1,c,h}$, replication policy 1.



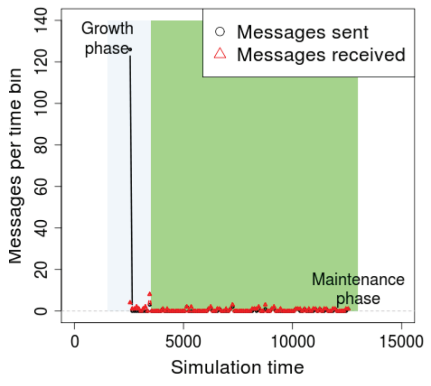
(b) $\mathrm{WO}_{250,c,h}$, replication policy 1.



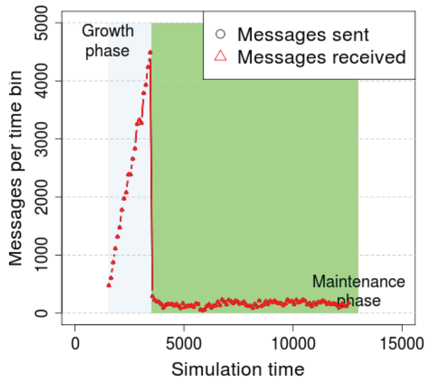
(c) Sum of all WOs, replication policy 1.



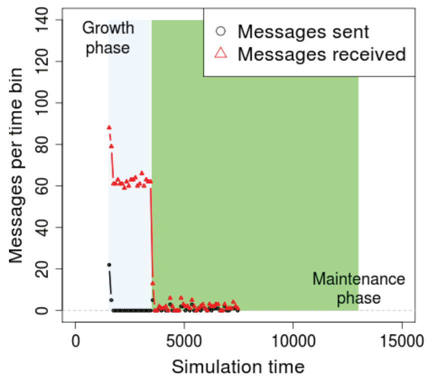
(d) $\mathrm{WO}_{1,c,h}$, replication policy 2.



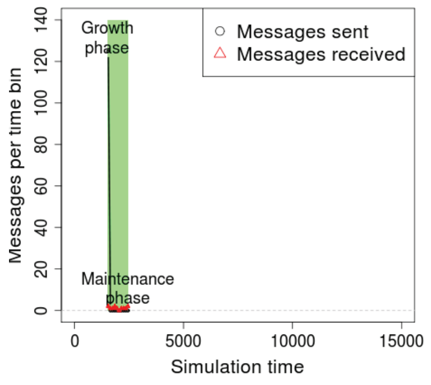
(e) $\mathrm{WO}_{250,c,h}$, replication policy 2.



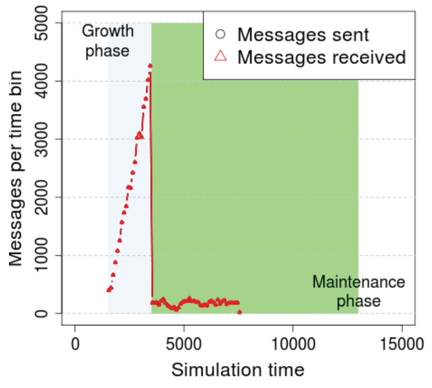
(f) Sum of all WOs, replication policy 2.



(g) $\mathrm{WO}_{1,c,h}$, replication policy 3.



(h) $\mathrm{WO}_{250,c,h}$, replication policy 3.



(i) Sum of all WOs, replication policy 3.

Figure 5: Showing total messages sent and received by an early node, a mid-simulation node and all WOs. The shape of the message sent curves (in black) for the early node is different based on the replication policy (see Figures 5(a), 5(d) and 5(g)). While the shape of messages received curve (in red) remains almost the same. This behavior is contrasted with the mid-simulation node (see Figures 5(b), 5(e) and 5(h)). The mid-simulation node message sent curve is constant regardless of the replication policy. The growth and maintenance phases are shown in light blue and light green respectively.

1000. These values allowed the simulation to execute quickly, therefore enabling more options and combinations to be investigated. After determining that a moderately aggressive replication policy enabled a high percentage of WOs to meet at least their c_{soft} goals, the next area of investigation was to determine how the total number of messages changes as a function of system size. Figure 5 clearly shows that there are two different types of communication curves reflecting the different types of communication during the growth and maintenance phases. During the maintenance phase, the WOs are attempting to spread their copies out across all the unique hosts in their friend's network. One of the contributing factors to this spreading is the limited capacity of the hosts to support preservation. In order to remove the effects of maintenance communications and focus purely on the effect of the number of WOs in the system, a series of simulations were run where $h_{\text{cap}} = 2 * n_{\text{max}}$. This ensured that there would be room on the host for any WO that discovered the host via one of their friends. Based on the simulations, the total number of messages exchanged during the growth phase approximates $O(n^2)$ and the incremental messaging cost of each new WO to the system is $O(2n)$.

6. CONCLUSION

Implementing Reynolds's "boid" premise that a limited number of rules in an autonomic manner can result in WOs behaving in a manner that works towards the betterment of the whole by occasionally sacrificing an individual. Using simulations, we investigated different policies that WOs could use when make preservation copies of themselves. The policies were:(1) be least aggressive and only attempt to make a single copy at a time, (2) be moderately aggressive and initially make at least a minimum number of copies and then revert to policy (1), or (3) be most aggressive and make as many copies as possible at every opportunity.

There are two distinct communication message curves; one prior to all the WOs being introduced into the system and one after. The growth period is characterized by many messages being sent from the *wandering* WO and few being received while the WO attempts to make its appropriate number of preservation copies. The maintenance period is characterized by a relatively few number of messages as the WO is directed to sacrifice its preservation copies for the greater good of the graph, and subsequently having to create copies anew. There are distinct differences between the growth message curves of new and late arriving WOs. The number of messages exchanged between WOs is virtually independent of the replication policy used. The difference between the maximum and minimum number of messages was only 18% when the USW graph had 100 WOs. As the order of the graph increased to 5000 WOs, the percent difference between varied from 1 to 8%.

Based on simulations of 500 WOs and potentially 500 hosts with limited preservation capacity; a replication policy of moderate aggression enabled the WOs to attain the same preservation percentage in the same time frame as the most aggressive policy and to more slowly exhaust the preservation capacity of the supporting hosts.

7. ACKNOWLEDGMENT

This work supported in part by the NSF, Project 370161.

8. REFERENCES

[1] S. Yin, "Flickr Permanently Deletes User's Account, 4,000 Photos by Accident," *PC Magazine*, February 2011.

[2] M. Milian, "GeoCities' time has expired, Yahoo closing the site today," *Los Angeles Times*, October 26, 2009.

[3] F. McCown and M. L. Nelson, "What happens when facebook is gone?," in *Proc. of the 9th ACM/IEEE-CS Joint Conf. on Digital Libraries*, pp. 251 – 254, 2009.

[4] R. Walker, "Cyberspace When You're Dead," *The New York Times*, January 11, 2011.

[5] M. Wohlsen, "Digital data that never dies," *Associated Press*, January 16, 2011.

[6] C. L. Cartledge and M. L. Nelson, "Unsupervised Creation of Small World Networks for the Preservation of Digital Objects," in *Proc. of the 9th ACM/IEEE-CS Joint Conf. on Digital Libraries*, pp. 349 – 352, 2009.

[7] C. L. Cartledge and M. L. Nelson, "Self-Arranging Preservation Networks," in *Proc. of the 8th ACM/IEEE-CS Joint Conf. on Digital Libraries*, pp. 445 – 445, 2008.

[8] C. L. Cartledge and M. L. Nelson, "Connectivity Damage to a Graph by the Removal of an Edge or Vertex," tech. rep., arXiv 1103.3075, Old Dominion University, Computer Science Department, Norfolk, VA, 2011.

[9] C. L. Cartledge and M. L. Nelson, "Analysis of Graphs for Digital Preservation Suitability," in *Proc. of the 21st ACM conference on Hypertext and hypermedia*, pp. 109 – 118, ACM, 2010.

[10] B. Cooper, A. Crespo, and H. Garcia-Molina, "Implementing a Reliable Digital Object Archive," in *Proc. of the 4th European Conf. on Research and Advanced Technology for Digital Libraries*, pp. 128 – 143, 2000.

[11] S. Payette and T. Staples, "The Mellon Fedora Project," in *Proc. of the 6th European Conf. on Research and Advanced Technology for Digital Libraries*, pp. 406 – 421, 2002.

[12] M. Smith, "DSpace: An Institutional Repository from the MIT Libraries and Hewlett Packard Laboratories," in *Proc. of the 6th European Conf. on Research and Advanced Technology for Digital Libraries*, pp. 543 – 549, 2002.

[13] P. Maniatis, M. Roussopoulos, T. J. Giuli, D. S. H. Rosenthal, and M. Baker, "The LOCKSS Peer-to-Peer Digital Preservation System," *ACM Transactions on Computer Systems*, vol. 23, no. 1, pp. 2 – 50, 2005.

[14] H. Van de Sompel, J. Bekaert, X. Liu, L. Balakireva, and T. Schwander, "aDORe: A Modular, Standards-Based Digital Object Repository," *The Computer Journal*, vol. 48, no. 5, pp. 514 – 535, 2005.

[15] V. Reich, "CLOCKSS-It Takes a Community," *The Serials Librarian*, vol. 54, no. 1-2, pp. 135 – 139, 2008.

[16] B. F. Cooper and H. Garcia-Molina, "Peer-to-Peer Data Trading to Preserve Information," *ACM Transactions on Information Systems*, vol. 20, no. 2, pp. 133 – 170, 2002.

[17] F. Dabek, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica, "Wide-Area Cooperative Storage with CFS," in *Proc. of the 18th Annu. ACM Symposium on Operating Systems Principles*, October 2001.

[18] M. Beck, T. Moore, and J. S. Plank, "An End-to-End Approach to Globally Scalable Network Storage," in *Proc. of the 2002 Conf. on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pp. 339 – 346, 2002.

[19] A. Rajasekar, M. Wan, and R. Moore, "MySRB and SRB: Components of a Data Grid," in *Proc. of the 11th IEEE Int. Symposium on High Performance Distributed Computing*, pp. 301 – 310, 2002.

[20] S. Rhea, C. Wells, P. Eaton, D. Geels, B. Zhao, H. Weatherspoon, and J. Kubiatowicz, "Maintenance-Free Global Data Storage," *IEEE Internet Computing*, vol. 5, no. 5, pp. 40 – 49, 2001.

[21] A. Rajasekar, M. Wan, R. Moore, and W. Schroeder, "A prototype rule-based distributed data management system," in *HPDC workshop on Next Generation Distributed Data Management*, 2006.

[22] D. Waters and J. Garrett, *Preserving Digital Information. Report of the Task Force on Archiving of Digital Information.* The Commission on Preservation and Access, 1996.

[23] J. Rothenberg, *Avoiding Technological Quicksand: Finding a Viable Technical Foundation for Digital Preservation. A Report to the Council on Library and Information Resources.* Council on Library and Information Resources, 1999.

[24] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small world' networks," *Nature*, vol. 393, pp. 440 – 442, June 1998.

[25] V. Nguyen and C. Martel, "Analyzing and characterizing small-world graphs," in *ACM-SIAM Symposium on Discrete Algorithms*, pp. 311 – 320, 2005.

[26] K. Klemm and V. M. Eguíluz, "Growing Scale-Free Networks with Small-World Behavior," *Physical Review E*, vol. 65, no. 5, 2002.

[27] A.-L. Barabási, R. Albert, and H. Jeong, "Scale-Free Characteristics of Random Networks: The Topology of the World Wide Web," *Physica A*, vol. 281, pp. 69 – 77, June 2000.

[28] P. Duchon, N. Hanusse, E. Lebhar, and N. Schabanel, "Towards Small World Emergence," in *ACM Symposium on Parallelism in Algorithms and Architectures*, pp. 225 – 232, 2006.

[29] J. Kleinberg, "The Small-World Phenomenon: An Algorithmic Perspective," in *Proc. of the 32nd ACM Symposium on Theory of Computing*, vol. 32, pp. 163 – 170, 2000.

[30] B. Bollobás, O. Riordan, J. Spencer, and G. Tusnády, "The Degree Sequence of a Scale-Free Random Graph Process," *Random Structures and Algorithms*, vol. 18, no. 3, pp. 279 – 290, 2001.

[31] B. Gaume and F. Mathieu, "From Random Graph to Small World by Wandering," Tech. Rep. 6489, Unité de recherche INRIA Rocquencourt, 2008.

[32] M. E. J. Newman, "Models of the Small World: A Review," *Journal of Statistical Physics*, vol. 101, p. 819, 2000.

[33] R. Kahn and R. Wilensky, "A Framework for Distributed Digital Object Services," *Int. Journal on Digital Libraries*, vol. 6, no. 2, pp. 115 – 123, 2006.

[34] M. L. Nelson and H. Van de Sompel, "IJDL special issue on complex digital objects: Guest editors' introduction," *Int. Journal on Digital Libraries*, vol. 6, no. 2, pp. 113 – 114, 2006.

[35] C. Cartledge, "Preserve Me! (... if you can, using Unsupervised Small-World graphs.)." http://ws-dl.blogspot.com/2013/10/2013-10-23-preserve-me-if-you-can-using.html/, 2013.

[36] S. Alam, "MS Thesis: HTTP Mailbox - Asynchronous RESTful Communication." http://ws-dl.blogspot.com/2013/09/2013-09-09-ms-thesis-http-mailbox.html, 2013.

[37] C. W. Reynolds, "Flocks, Herds and Schools: A Distributed Behavioral Model," *SIGGRAPH Computer Graphics*, vol. 21, no. 4, pp. 25 – 34, 1987.

[38] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker, "A Scalable Content-Addressable Network," in *Proc. of the 2001 Conf. on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pp. 161 – 172, 2001.

[39] J. Hunter and S. Choudhury, "A Semi-Automated Digital Preservation System Based on Semantic Web Services," in *Proc. of the 4th ACM/IEEE-CS Joint Conf. on Digital Libraries*, pp. 269 – 278, 2004.

[40] H. M. Salaheldeen and M. L. Nelson, "Resurrecting My Revolution: Using Social Link Neighborhood in Bringing Context to the Disappearing Web," in *Proc. of Theory and Practice of Digital Libraries*, pp. 333–345, 2013.

[41] D. S. H. Rosenthal, T. S. Robertson, T. Lipkis, V. Reich, and S. Morabito, "Requirements for Digital Preservation Systems: A Bottom-up Approach," *Dlib Magazine*, vol. 11, 2005.

[42] J. L. de la Rosa and J. A. Olvera, "First Studies on Self-Preserving Digital Objects," in *Artificial Intelligence Research and Development: Proc. of the 15th Int. Conf. of the Catalan Association for Artificial Intelligence and Applications*, pp. 213 – 222, 2012.

[43] J. L. de la Rosa, E. Del Acebo, A. Trias, S. Aciar, and H. Quisbert, "Crew Intelligence Systems for Digital Objects Preservation," in *The 2nd Swarm Intelligence Algorithms and Applications Symposium-SIAAS*, vol. 9, 2009.

[44] L. Spector, J. Klein, C. Perry, and M. Feinstein, "Emergence of Collective Behavior in Evolving Populations of Flying Agents," in *Genetic and Evolutionary Computation Conf.*, pp. 61–73, Springer, 2003.