16th ICCRTS

"Collective C2 in Multinational Civil-Military Operations"

On Optimizing Command and Control Structures

Topics

Primary:
Topic 9: Networks and Networking

Alternates:
Topic 5: Collaboration, Shared Awareness, and Decision Making
Topic 8: Architectures, Technologies, and Tools

David Scheidt
The Johns Hopkins University Applied Physics Lab, 11100 Johns Hopkins Rd, Laurel MD 20723

Kevin Schultz
The Johns Hopkins University Applied Physics Lab, 11100 Johns Hopkins Rd, Laurel MD 20723

Point of Contact
David Scheidt
The Johns Hopkins University
Applied Physics Laboratory
11100 Johns Hopkins Rd
Laurel, MD 20724
David.Scheidt@jhuapl.edu

On Optimizing Command and Control Structures

David Scheidt, Kevin Schultz
Johns Hopkins University Applied Physics Laboratory
Laurel, MD
{david.scheidt, kevin.schultz}@jhuapl.edu

Abstract— **This paper uses information theory to investigate the utility of alternative Command and Control (C2) structures and strategies. Specifically we use information theoretic metrics *complexity, entropic drag and evolutionary entropy.* Complexity measures the amount of information required to fully express a situation. Entropic drag measures the rate at which situational information becomes obsolete. These metrics are used to characterize situations managed by C2 systems. The performance of C2 designs is measured as a function of these characteristics; performance that is characterized as the C2 system's impact on mission performance, situation awareness and controllability. This paper provides a comparative analysis of C2 topologies including: centralized control, scale-free hierarchies, scale-invariant hierarchies and heterarchical "small world" topologies. Dominance plots that map optimal topologies to scenario characteristics are provided as results. To support our analysis of training and innovation we introduce the concept of *evolutionary entropic drag*, which is the rate at which adversaries are able to adapt their behavior.**

Keywords- Concepts, Theory and Policy; Collaboration, Shared Awareness and Decision Making; Networks and Networking

I. INTRODUCTION

A healthy tension exists in military command and control (C2) between the desire to maximize coordination between units through centralized planning and the desire to rapidly respond to changing conditions with decentralized "agile" units. In *Power to the Edge*, Alberts and Hayes [1] provide compelling evidence that agile C2 can provide a decisive advantage in military engagements. In *Planning Complex Endeavors* [2] Alberts and Hayes showed how multi-unit planning can be performed by "flattened" decentralized organizations. In his highly influential paper *Network-Centric Warfare: Its Origin and Future* [3], Vice Admiral Cebrowski hypothesized that future military conflict will be won by the side that achieves "information dominance" by producing and distributing information throughout a force more effectively than one's adversary. The Net-centric Warfare vision suggests that modern, internet-like, tele-communication networks would be the key enabler to information dominance. It is widely assumed that net-centric warfare and agile C2 visions are synergistic. When net-centric systems use internet-like networks the net-centric infrastructure may not be conducive to agile C2. As shown in this paper, agile systems based upon flattened structure dissimilar to the internet can outperform internet-like structures.

Internet style networks are scale free, relying on quasi-centralized "supernodes" to rapidly connect leaf nodes. Barabasi [4] showed that the scale free topologies are highly effective at connecting massively parallel distributed systems. Barabasi's findings appear to contradict Alberts' assertion that flat, decentralized organizations outperform centralized and quasi-centralized organizations. Further, recent U.S. military experiences in Iraq and Afghanistan support Alberts' argument, as experiences show that large interconnected networked force with superior intelligence, surveillance and reconnaissance (ISR) assets will not always achieve information dominance over forces that use human spotters communicating with manual signals. Failure to achieve information dominance over less sophisticated adversaries is not due to a lack of sensing capabilities but due to an inability to process and deliver information to the appropriate war fighter without overloading and/or over-tasking the war fighter. The reason Barabasi's discovery that scale free networks outperform other structures does always hold for C2 systems is found in the differences between internet and military requirements. Whereas the Internet is commonly used to facilitate information exchanges between arbitrary users, C2 networks are designed to positively impact a game-theoretic, time-varying system, specifically a military engagement. As such, C2 networks are more contextual and more time-sensitive than their civilian counterparts. A specific difference is that the value of military C2 information is more dynamic as the value of the information is more susceptible to change over time.

This paper explores the impact of information dynamics on command and control structure. We do this by using information theory to characterize the underlying environment, the C2 system and C2 measurements.

II. DEFINITION OF A COMMAND AND CONTROL SYSTEM

In *Power to the Edge*, Command and Control (C2) is defined as the "common military term for management of personnel and resources." They go on to discuss some of the approaches taken to differentiate command and control, but note that many of these approaches are focused on a single command entity, despite their primary thesis that command and control has become a distributed responsibility. If one thinks of the structure of the C2 process as a network, on the "edge" of this network (to use the terminology of Alberts and Hayes) are information sources and actors. Information sources include sensors, humans, databases, etc. Actors include mechanical systems, humans, and weapons systems; essentially any entity that can change its state due to a control action is an actor. There is considerable (if not complete) overlap between information sources and actors in a given situation. These two types of entities, along with non-combatants and adversaries are called agents.

One focus of this manuscript is the structure of the C2 network. The interconnections between agents in the C2 process form networks, and we show that the structure of these networks can affect the performance of the agents in the C2 process. In hierarchical C2 structures, the sources and actors are at the bottom of the hierarchy. Sources provide information that is passed and fused through various intermediaries in the hierarchy, who in turn are responsible for passing information to superior entities and relaying control actions to their subordinates based on instructions from their superiors. Additionally, if this hierarchy is centralized and all agents share a common representation scheme, then the role of the intermediary entities is purely to relay information, all the decision making is performed at the top of the hierarchy. For centralized hierarchies that do not share a representation scheme, intermediate agents take on the responsibility of translating information across representation schemes. The concept of "power to the edge," however, includes the notion of sources and actors being able to share information and also to have some authority to use local information to dictate their own control actions. If the sources and actors are to communicate and coordinate in a non-hierarchical fashion, this implies a degree of decentralization in the decision making process, potentially in a collaborative fashion between actors.

Boyd's Observe, Orient, Decide, Act (OODA) loop [5] is a basic model for a C2 process. In the context of a single entity, this is a self-contained cycle based on that entity's knowledge and decision-making abilities. This line of thinking can be extended to a centralized C2 process of many entities by thinking of the observation stage as including communication of observation towards the centralized decision maker, and the act stage including the communication of control actions from the decision maker to the actors. In a more decentralized C2 process, each entity is responsible for its own OODA loop, while (potentially asynchronously) receiving communications containing observations, decisions, action requests, etc., from other entities.

III. Information Theory System Characterization

For a C2 process, the continuous world in which the process evolves is encoded in a C2 system that discretizes the world into a discrete and finite number of states so that the inherently discrete sensors, communications, computers, etc., can interact with the process. This discretization is fundamentally an engineering decision based on the communication and computational resources available. For our purposes, we will assume that the C2 process $x(t)$ operates in a discrete state space $X=\{x_i\}$ with finite cardinality $|X|$. Then, we call the number of bits required to describe the state of the C2 process the *descriptive complexity*, which is equal to $\log_2|X|$ bits. The descriptive complexity can be thought of as the fidelity at which the process is described, for instance the location of an adversary agent to the nearest meter as compared which city block the agent currently inhabits.

To quantify the notions of uncertainty and information in the process, we use Shannon information entropy [6]. For each state $x_1 \in X$ the information entropy (in bits) of the process x at time t is

$$H(x(t)) = -\sum P(x_i,t)\log_2\left(P(x_i,t)\right),$$

where $P(x_i,t)$ is the probability that the process x is in state x_i at time t. Additionally, the information, I, of a particular observation $x(t)=x_i$ is

$$I(x_i,t) = -\log_2 P(x_i,t).$$

Thus, one interpretation of entropy is the expected information gain of an observation. It is known that entropy is maximized when all states are equally probable, i.e., $P(x_i,t)=|X|^{-1}$ for all states x_i, so in this sense the notion of descriptive complexity outlined above is the maximum possible entropy of the system with a fixed discretization.

Consider an observation $S=(x_i,t)$ with the interpretation that $x(t)=x_i$, then as mentioned above, (allowing for an abuse of notation) the information content of the observation is

$$I(S,t) = -\log_2 P(x_i,t).$$

As time elapses, however, the relevance of the observation S that occurred at time t should decrease. Indeed, if the system is not uniquely determined by a single observation, then the conditional expected information content of a second observation $S' = (x_j,t')$ of the same sensor for some $t' > t$ is $H(x(t')|S)$ would be nonzero. The very fact that repeating the same observation (i.e., polling the same sensor) results in gaining new information beyond the original observation indicates that the previous observation's information content has in some sense decayed. For a sequence of k observations $S_{1:k}=\{(x_j,t_j)\}$ of the process x ending at time $t_k = t$ we define the *entropic drag* (Γ) of the system on the observations at time $t'>t$ by

$$\Gamma(S_{1:k},t,t') = \frac{H(x(t')|S_{1:k})}{t'-t}.$$

Conceptually, this can be thought of as the time-derivative of conditional entropy, but in a strict mathematical sense the assumed discrete space of the sensors will not admit a derivative. For observations that occur with fixed sampling time $\Delta t>0$ the quantity $\Gamma(S_{1:k},t,t+\Delta t)$ is effectively the expected rate of information generation of the system at time t. Entropic drag should not be interpreted solely as being due to the motion of the underlying system, for example a pendulum or train moving at a fixed speed have predictable trajectories and would have considerably lower entropic drag than systems whose motion is not as constrained.

The information sources and actors in a C2 process have access to their own local repositories of information in the form of local observations and information accumulated through communication. We call the sum of all information over the set of sources and actors at any specified point in time the *information volume* (at that time). Since there will generally be overlap in the information known between different actors and sources, the information volume can be larger than the information content of the state of the system, i.e., the descriptive complexity. However, the information volume being larger than the descriptive complexity does not mean that fusion of all the information comprising the information volume would result in enough unique information to be able to determine the state of the system. Since the information about the system from a particular agent's point of view is bounded by the system complexity, it is clear that the information volume must be bounded by the descriptive complexity times the number of unique agents that are sources and/or actors.

It is assumed that actor agents are able to take control actions. Here, we refer to a specific control action that an actor takes as an actuation, in much the same way that a sensing action results in an observation. While an actuation is a deterministic event and as such contains no information per se, we call the information that went into the decision that determined the control process the *actuation information*. If an actuation is taken and held without reprocessing new information, the actuation information of that actuation can decrease via entropic drag on the information that the decision was based upon. In this way, even if an actuation was based on perfect knowledge of the system state, unless the actuation eliminated all future uncertainty in the process, the actuation could eventually be rendered suboptimal due to the effects of entropic drag.

IV. INFORMATION FITNESS

For a C2 process, it can be tempting to believe that more information results in better, more effective decisions and thus more effective control actions. Indeed, this is not inconsistent with notions from information theoretic control theory. Touchette and Lloyd [7,8] showed that the maximum decrease in uncertainty of some controlled system was equal to the decrease possible without any information (i.e., open-loop control) plus the information gathered by the controller observing the system state. On its face, this indicates that the descriptive complexity should be as high as possible, in order to reduce the uncertainty as much as possible. This strategy would certainly work for a system without entropic drag, as the time spent gathering, fusing, processing, and communicating observations and decisions would have no effect on the information content of the observations.

When entropic drag is considered, however, there is a cost to increasing the descriptive complexity. As the descriptive complexity increases, the number of bits required to fully encode the state increases. If the communications bandwidth is fixed, more time is required to communicate observations. Additionally, as the descriptive complexity increases, the amount of time required to process these observations after they are communicated will likely increase as well. As the time to manipulate information increases, the actual information content will be reduced due the decay induced by entropic drag and there is likely a point where the information will be decaying faster than it can be used [9], see Figure 1.

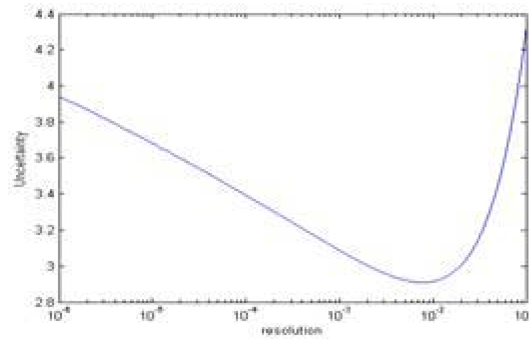


Figure 1: Uncertainty of a sample two-dimensional world showing effects of entropic drag as unit resolution is increased. Increasing resolution increases complexity, which increases communication time, resulting in more entropic drag [9].

Beyond information content and its decay via entropic drag, there are other factors that determine the utility of information. Different actors have different tasks to accomplish, and the nature of these tasks determines the relevance of an observation to these actors. Much as the relevance of an observation to an agent decays relative to localization in time, it is natural to suggest that for certain types of agents there is likely a decrease in relevance of an observation with respect to spatial localization as well. If an agent is not fast or far reaching enough to affect entities far away in space, that information will likely be less relevant to that agent. There are also likely cases where the utility of information to an agent is a nonlinear function of the specific observations and not just the sum of the information content. Because of this, a C2 process is, as has often been said, about *getting the right information to the right agents at the right time*, and there are tradeoffs between individual utility of information and mission objectives.

The following sections we describe some simulation-based experiments that initially focus on situational awareness, the "observe orient" portion of the OODA loop and subsequently extend to include the execution or "act" part of the OODA loop. All of these experiments assume a uniform utility of information.

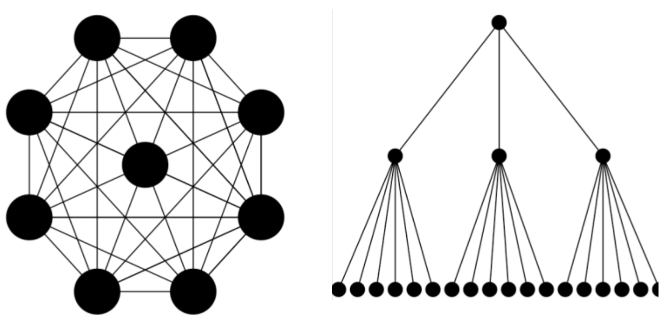
V. METHODOLOGY

A. *Graph Theoretic Preliminaries*

Here, we define the command and control network topology using the mathematical concept of a graph. A (finite) graph $G = (V,E)$ is a collection of two sets, where $V=\{1,\ldots,N\}$ is the set of N vertices or nodes of the graph, and $E \subseteq V\times V$ is the set of edges of the graph. In this manuscript, we assume that G is an undirected graph with no self-loops, thus $(i,j) \in E \Leftrightarrow (j,i) \in E,\ \forall i,\ j\in V$ and $(i,i) \notin E,\ \forall i,j\in V$. Let $\deg(i)=|\{i,j\} \in E : j \in V\}|$ be the degree of the vertex i. A graph is connected if for any $i, j \in V$, $i \neq j$, there is a sequence of vertices a_k, $k=0,\ldots,K$, where $(a_{l-1},a_l) \in E$ for $l=1,\ldots K$, $a_0 =$

i, and $a_K = j$. We call this set of vertices a path, and the length of the path is K, and the minimum length of all such paths between i and j is called the geodesic distance between i and j. We denote this length $d(i, j)$ when i, j are connected, and set $d(i, j)=\infty$ if they are not connected. A path is simple if it repeats no vertices from start to end (thus all minimum length paths are simple, but not vice-versa). A graph is acyclic if for every $i \in V$, there does not exist a simple path of nonzero length from i to i.

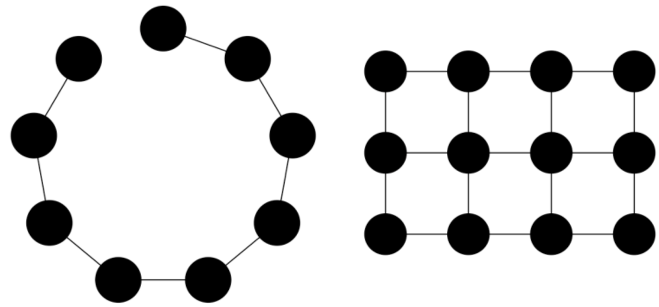
A number of different graph topologies are studied here, including randomly generated graphs. All graphs are assumed to be finite and connected, as well as undirected and without self-loops, as described above. An all-to-all or fully connected graph is a graph where each vertex is connected to all other vertices, i.e., $\deg(i)=|V|-1$, $\forall i \in V$ (see Figure 2a). A tree is a connected acyclic graph. If we choose one vertex i in a tree and call it the "root" of the tree, then we have a rooted tree. The parent of a vertex i in a rooted tree is the vertex j such that $(i, j) \in E$ and (i, j) is in the simple path to the root, and every vertex except the root has a unique parent. The children of a vertex i are the set of vertices for which i is a parent, and a leaf is a vertex without any children. Vertices of the same geodesic distance from the root are said to be of the same generation. Here, we look at two particular classes of rooted tree, m-ary trees and regular trees. An m-ary tree is a rooted tree where each node has at most m children. Regular trees are described by a vector $[a_1,a_2,\ldots,a_n]$, where a_i are positive integers, that define the maximum number of children per node in a generation (see Figure 2b). For a given regular tree described by $[a_1,a_2,\ldots,a_n]$, the root has at most a_1 children, the root's children have at most $a_2$children, and so on. Clearly an m-ary tree is also a $[m,\ldots,m]$ regular tree. Unless otherwise noted, all m-ary and regular trees are "full" in the sense that they have the maximum number of children per generation, and the number of generations is fixed.



(a) (b)

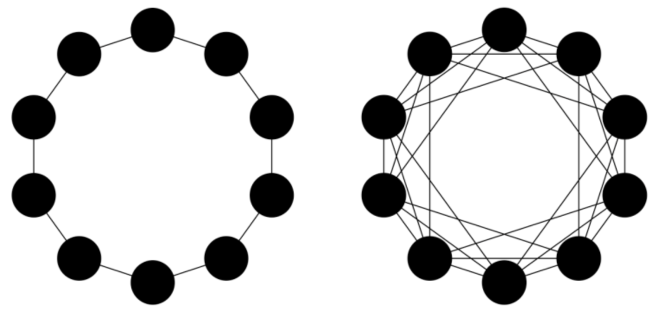
Figure 2: Sample graphs. (a) Fully connected. (b) [3,6] tree.

A path graph is a tree with two vertices of degree one, and the remaining vertices of degree two (see Figure 3a). A 1-ring is formed by taking a path graph and adding an edge between the two vertices of degree one (assuming $|V| \geq 3$). A k-ring (for $k > 1$) can then be defined from a 1-ring by connecting each vertex to vertices with geodesic distance $\leq k$ along the 1-ring, up to $\lfloor |V|/2 \rfloor$, where $\lfloor x \rfloor$ is the greatest integer $\leq x$ (see Figure 4). The final class of non-random graphs that we consider is the two-dimensional grid graph. A grid graph G can be defined as the Cartesian product of two path graphs $P_1=(V_1,E_1)$ and $P_2=(V_2,E_2)$, where the vertices of $G=V_1 \times V_2$ and two vertices (i,i') and (j,j') are adjacent in G if $i=i'$ and $(j,j') \in E_2$ or $j=j'$ and $(i,i') \in E_1$ (see Figure 3b).



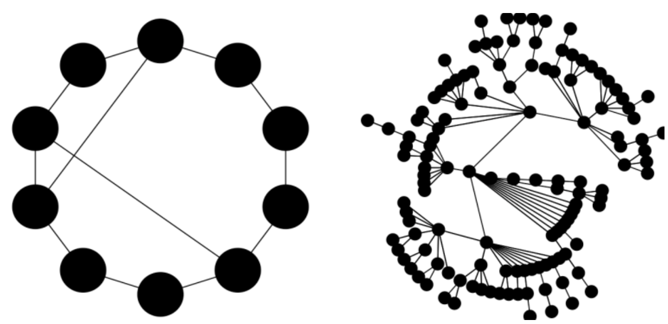
(a) (b)

Figure 3: Sample graphs. (a) Path. (b) 4 by 3 grid.



(a) (b)

Figure 4: Sample graphs. (a) 1-ring. (b) 3-ring.



(a) (b)

Figure 5: Sample graphs. (a) 1-ring with two additional links. (b) (1,1) scale-free.

Two different classes of random graphs are investigated here as well. Small world graphs were introduced by Watts and Strogatz [10], but we use the variation used by Newman and Watts [11] and also by Monasson [12]. In this variation, a small-world graph is generated by adding additional edges to the graph, instead of "swapping" a vertex in an existing edge. In this paper, we use k-rings as defined above for the base lattice, and denote the resulting small-world graph as a k-ring with m additional links (k-ring + m for short, see Figure 5a). The other class of random graph investigated here is the scale-free graph generated using Barabasi's preferential attachment mechanism [13]. Here, we use the term (l,m) scale-free to denote a graph generated starting from a fully connected graph of l vertices and adding l vertices each with degree m at each iteration of the construction process (see Figure 5b).

B. Information Flow Simulation

To explore the flow of entropic information on a network of C2 entities, we constructed simulations that would allow us to study network and system parameters. We use a given graph $G=(V,E)$ to model a specific instance of a networked collection of entities observing an external system. The simulation models a distributed set of sensors and intermediary nodes connected through the network that are observing portions of a common system and communicating and processing these observations. This network is assumed to be time invariant; so that the edge set E is fixed over the course of an experiment. In addition to the sets V and E, we define a sensor set $S \subseteq V$, denoting the entities that are equipped with a sensor that observes the system. Each of the sensors observes one bit of information, i.e., each sensor partitions the state space in to two equally probable outcomes. Additionally, we assume that the set of sensors is such that the largest common refinement of the sensors' corresponding partitions results in $2^{|S|}$ equally probable outcomes, so that the maximum amount of unique information in the system is $|S|$ bits. Here, the state space of the simulation does not represent a collection of sensor readings, but rather the state space is a collection of each node's information content (i.e., one bit of information per sensor less the current entropy). To model entropic drag, we use a function that decays the information content (i.e., increase the entropy) of an observation from the time that it was actually observed. For a specific observation $I(t_0)$ initially sensed at time t_0, we calculate the information content of that observation at a later time by $I(t_0+t)=I(t_0)(1-\Gamma)^t$, and we call Γ the (geometric) decay rate.

We implemented a discrete event system (DES) simulation that accommodates variable delays in computation and communication. This simulation models the "observe-orient" portion of the OODA loop. In this model, a node on the graph G represents an entity. Entities are abstractions that represent either automated or human elements. This entity is in one of four states: **SEND**, **CHECK**, **COMPUTE**, or **SENSE**, representing the node sending, checking, processing, and sensing new information, respectively. In the sensing state, a node $i \in S$ reads one bit of perfect (no initial uncertainty) information and associates with this information a time-stamp at the current simulation time. In our experiments we assign a delay of 1 sec to the sensing state. From the sensing state, a node immediately enters into a computation state whose time length is a function of the amount of information to be processed. Computation delay is determined by a linear delay that is proportional to the number of new sensor readings, counting each sensor no more than once (i.e., only the most recent observation from a given sensor is "processed"). A simulation parameter of β sec per unprocessed observation is used to determine the length of the **COMPUTE** state. The term β is used to quantify the complexity of the computations. After **COMPUTE** has been completed, the node enters **SEND** and communicates with its neighbors in the graph by sending information that they have processed that the neighboring node has not yet received. This communication stage takes 1 sec of simulation time regardless of the number of observations sent, and all neighbors are communicated with simultaneously and without interaction on the part of the neighbor. From **SEND** the node enters **SENSE** again if the node is in S and there are still sensor readings to perform, otherwise, it goes into **CHECK**. The state **CHECK** is a holding state where the node remains until it receives new information, when the process transitions to **COMPUTE** on the new information.

To test the hypothesis that certain network topologies would perform better than others under different decay rates and complexities, we compared simulations of thirty different graphs at a number of points in the decay-complexity (Γ,β) plane (see Appendix A). Included in this experiment are a fully connected graph, a grid, a number of trees, small-worlds, rings, scale-free graphs. All of the graphs had 127 vertices with 64 sensors (the number of vertices and leaves in binary tree of seven generations, respectively). The non-binary trees and the grid graphs were truncated from their full size to 127 by removing rows and columns from the adjacency matrix. For the trees, node reduction removed leaf vertices such that all siblings of that vertex are removed before removing a leaf node from another group of siblings. For the grid graph, this amounts to removing vertices from the last "row" of the grid starting at a corner and moving in the same direction. The sensor set for all graphs were vertices 64-127. Each sensor performed exactly one observation at the beginning of the simulation. The primary metric used in this simulation is the sum of the information of all processed observations at a given time step, denoted the processed information volume.

To explore the relationship of complexity and entropic drag on irregular, random graphs (both small-world and scale-free), an intermediate step was taken to first find representative graphs. For each type of random graph (e.g., 3-ring with 10 links, or scale-free generated by adding 3 nodes per iteration), a total of twenty sample graphs were generated, and running an experiment on the collection of twenty samples produced information volume metrics. From the results of these experiments two (in one case three) graphs were selected based on the results. For most of the samples, one sample tended to dominate in the lower-left (Γ,β) ranges and another sample dominated in the upper-right (Γ,β) ranges. In the case of the small-world graph constructed from a 1-ring with 15 links, there were three dominant samples, one in the lower-left and upper-right range, but also an additional sample graph that dominated in the middle range between the other dominant samples. These dominant sample graphs were then compared with other topologies.

C. Information Flow with Actuation

In order to begin an investigation system that includes both observation and actuation (control), we implemented a second DES simulation. We chose to first study the simpler case of hierarchical C2 processes. This restricts the topologies studied to trees. Specifically, there is a designated root node, and distance from that root node determines order in the hierarchy, i.e., child nodes are subordinate to their parent node. Nodes in the tree are equipped with sensors as above, but a node with a sensor is also equipped with an actuator that can take a one-bit control action. Information is pushed up the hierarchy, and actuation decisions are pushed down the hierarchy. Additionally, vertices at each level are able to process information received from sensors below them in the hierarchy, and relay the resulting control actions based on information from this subset of sensors to the actuators below them.

In this simulation, each node is in one of seven high-level states at any time step: **SENSE**, **SEND_I**, **REC_I**, **PROC**, **SEND_A**, **REC_A**, **ACT**, and **HOLD**. As above, in the **SENSE** state, a node that is equipped with a sensor reads one bit of perfect information. If it is a child, it will then transition in one time step to **HOLD** to attempt communication; otherwise it will go into the **PROC** state. In the **SEND_I** state, a node is sending all of the sensor observations for sensors below it in the tree to its parent node, which must be in the **REC_I** state. Mirroring these two states are the states **SEND_A**, where a parent is communicating actuation commands to a child for all of the actuators below the child in the hierarchy, and the state **REC_A** where a child receives actuation commands from its parent. All of the communication states last for one time step per bit communicated (i.e., one time step per sensor observation or actuator command). From the **SEND_I** state, the node will go in to the **PROC** state. From the **REC_I** state the node will go into **PROC** if it is the root and it has received information from all of its children since the last time it was in **PROC**. From **SEND_A**, the node goes into **HOLD**. From **REC_A**, if the node is an actuator, it will enter **ACT**, otherwise it will enter **HOLD**. In the state **PROC**, the node will delay for the number of sensors below it in the hierarchy (representing a processing delay of one time step per bit of observation), and from **PROC** it will go to **ACT** if it is an actuator, or to **HOLD** if it is not. The **ACT** state takes one time step to represent the application of the control command to the actuator, and the node will then transition to **HOLD** if it has actuation commands to relay to children, or to **SENSE** if it does not. In the **HOLD** state, the node first looks among its children for nodes who are also in **HOLD** who haven't yet received its portion of the most recent actuation command (whether this was received from a parent or the result of processing at the node). If such a child exists, the child transitions to **REC_A** and the node transitions to **SEND_A**. If no such child exists, the node will send information to its parent if it has received information from all of the node's children since the last time the node communicated with its parent. If this is the case, the node transitions to **SEND_I** and the parent transitions to **REC_I**. If neither of the above steps results in a state transition, the node remains in **HOLD** until the next time step when it tries again to communicate.

Clearly, the communication process (for both observations and actuations) is significantly different from the process in the previous simulation. For this simulation, a vertex can only communicate with a single neighbor at a time, and must wait for a neighbor to be ready to initiate communication. Additionally, communication between neighbors is now serial, so that the communication time is linearly proportional to the number of bits in the communication (whether observation or actuation). Another simulation parameter that changed between the two simulations is the use of repeated observations for this simulation, as compared to the single round of observations used in the previous simulation. Rather than information volume, the metric used to evaluate results is total actuated information, which we define as the sum over all time steps and actuators of the information content behind an actuation decision at a given time step and actuator.

As above, the number of sensors (and thus actuators) is 64 across all topologies tested; however, the total number of nodes was varied to explore the effect of intermediary (non-sensor/actuator) nodes on the network (see Appendix B). Additionally, the root node of the hierarchy was determined by finding the node with the minimum average geodesic distance to all other nodes in the graph, using a deterministic method to break ties (lower node number in our adjacency matrix based enumeration). For regular trees, this corresponds to the root as defined in the regular tree definition, for path graphs this is the median node, and for the scale-free graphs the initial node in the construction algorithm (although this is not guaranteed in general). In addition to the algorithm given above, we also ran each graph through two slightly modified simulations. In the first modification, a sensor node in **HOLD** will perform another observation instead of a repeated **HOLD** and then enter **HOLD** again (i.e., the observations will be more recent when the node finally communicates with its parent). We call this resensing. In the second modification, a sensor node in **HOLD** will transition to the **SENSE** state instead of a repeated **HOLD**, resulting in a re-sense, re-process, and re-actuation before it enters the **HOLD** state and attempts communication again. We call this reprocessing.

VI. RESULTS

A. *C2 Topology impact on Situational Awareness (Information Flow Simulation)*

To test the hypothesis that different topologies would perform better (in terms of information volume) under different computational complexity and entropic drag conditions, thirty different graphs were compared at a number of points in the (Γ,β) plane using the information flow simulation as described in Section 5.1. Specifically, the decay rate varied from 0.001 to 0.999 in increments of 0.002 bits/step, and the computational complexity varied from 0 to 16 steps/bit in increments of one. To display the results of this simulation, we use a dominance plot (see Figure 6), which shows the topology with the greatest processed peak information volume at each tested point in the (Γ,β) plane.

There are a number of interesting features present in the dominance plot. First, not every graph has its own region of dominance, and there are also regions where multiple graphs co-dominate. At the far right of the plot is a region marked "All Bad" where every topology performed equally poorly, due to the extremely high entropic drag and computational complexities. Along the majority of $\beta=0$ and for a small range of $\beta=1$ is the region dominated by the all to all graph. The lower left region of the plane is dominated by various scale-free graphs, with the two (5,5) scale-free graphs closer to the origin, followed by the two (3,3) scale-free graphs. One of the (1,1) scale-free graphs forms much of the boundary between the scale-free graphs and the large region dominated by the binary tree. To the right of the binary tree region of dominance is the region dominated by one of the 3-rings with 10 additional links. Adjacent to this region is a region where a 3-ring with 15 links is co-dominant with the same 3-ring with 10 that dominated the region to the left. Adjacent to this co-dominant region is another co-dominant region that includes the grid graph in addition to the co-dominant graphs in the region to the left. In the lower right corner is a small region that is co-dominated by the regular trees. There are also a few very small disconnected regions that appear in the right of the dominance plot that are likely related to numerical issues.

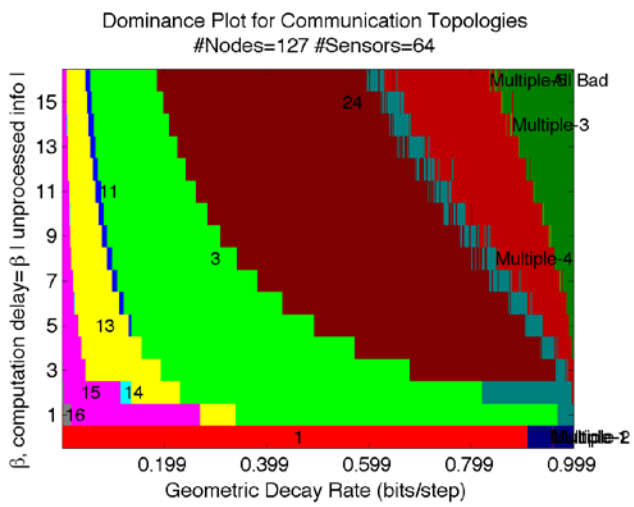


FIGURE 6: DOMINANCE PLOT. 1- ALL TO ALL. 3- BINARY TREE. 11- (1,1) SCALE-FREE. 13 - (3,3) SCALE-FREE. 14- (3,3) SCALE-FREE. 15- (5,5) SCALE-FREE. 16- (5,5) SCALE-FREE. 24- 3 RING+10. MULTIPLE 1&2- REGULAR TREES. MULTIPLE 3- 3 RING+10, 3 RING +15, GRID. MULTIPLE 4- 3-RING+10, 3-RING+15. MULTIPLE 5- SEVERAL GRAPHS. ALL BAD – ALL GRAPHS EQUALLY POOR.

There appears to be some connection between neighboring regions of topological dominance that appeals to intuition. Starting from the far right of the dominance plot, the region where all topologies performed equally poorly indicates a region where communication is of no value, since the information content of any additional observations is lost during the time that it takes to communicate and process the observations. The regions dominated by the different small world graphs and the grid graphs correspond to regions where it is efficient to communicate with a small number of neighbors. To see why the grid graph is co-dominant in one of the regions, note that it is only one edge short of being isomorphic to a 1-ring with 132 additional links, so it also balances local communication with links across the graph. The region dominated by the binary tree corresponds to parameters that continue to favor communication among a small number of neighbors, but also favor gathering this information and distributing hierarchically up and down the tree. Adjacent to the binary tree dominated region is a narrow region dominated by a (1,1) scale-free graph. This graph is also a tree, but lacks the regular structure of the binary tree and instead has the large range of degrees from which the scale-free graph gets its name. The regions from the binary tree dominated region moving left each represent a decrease in average path length of the graph at the cost of an increase in average vertex degree. A vertex with higher degree tends to serve as a conduit of information to its neighbors, so they spend a lot of time processing and communicating, and thus serve as bottlenecks.

The power law distribution of vertex degree in scale-free graphs necessarily results in so called super-nodes of high degree [4]. In networks where these super-nodes are responsible for increased traffic and information processing (such as the Internet), the computational power of these nodes should be greater than nodes with little or no computational needs (such as end-users) to maintain efficient operation of the network. The design of this simulation, however, is that the communication and processing capabilities of all nodes are assumed to be identical, beyond network topology differences and sensing ability. This results in areas in the (Γ,β) where scale-free topologies are not dominant. In fact, the region dominated by the binary-tree is larger than all tested scale-free graphs combined.

B. C2 Topology impact on Control Efficacy (Information Flow with Actuation)

To test the efficacy of different hierarchical control structures (in terms of total actuated information), eleven different trees were compared using the information flow with actuation simulation. Each graph was run using the base behavior, as well as the resensing and reprocessing modifications as outlined above, for a total of 33 different experiments over 501 logarithmically spaced geometric decay points between 0 and 1. Logarithmic spacing of decay was chosen because initial simulations using linear spacing indicated more variation in relative performance over the lower decay regions than the vast majority of the higher decay regions. Results from the comparison of the 33 different experiments are shown in Figure 7. These figures show the ranking of the 33 experiments in ascending order (i.e., rank 33 had the most total actuated information at that decay value), rather than the total actuated info. These plots can be thought of as a slice of a dominance plot similar to Figure 6. Again, there are a number of regions of dominance, with four different dominant topologies. The dominant topologies, from lowest decay to highest are: [64] regular tree with resensing, [2,2,2,2,2,2] regular tree with resensing, (1,1) scale-free with reprocessing, and a path graph of length 64 with reprocessing.

For the vast majority of the decay ranges, i.e., $\Gamma \in [0.1,1)$, the reprocessing modification outperformed all other candidates. The interpretation here is that when the decay rates are higher, the actuation commands received from nodes above in the hierarchy are based on observations that have little remaining information content, and the only way to accumulate actuations based on any information content is to use repeated local observations. Nearly the exact opposite phenomenon occurs for the low decay regions. With the exception of the dominant [64] regular tree using resensing, the next eleven dominant are candidates that are not reprocessing or resensing. This indicates that when decay rates are low, for the actuations received from agents above are based on much more information, making the reprocessing strategy less effective. Furthermore, at low decay rates, even the additional time taken to perform resensing reduces the amount of total actuated information by decreasing the frequency at observations can be sent upward and actuations sent downward.

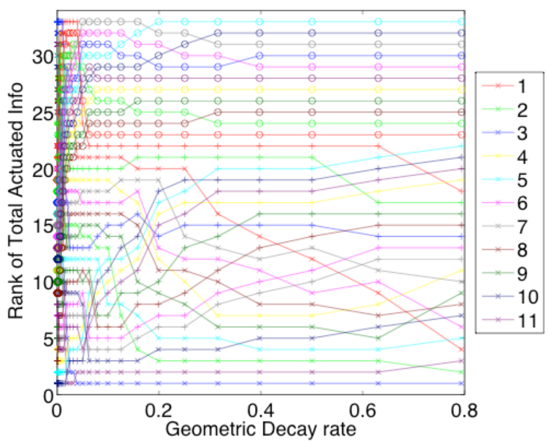


Figure 7: Rank of total actuated information for 11 graphs with linear scale in decay. The graphs are indicated by color – (1) [2, 2, 2, 2, 2, 2] (binary) tree, (2) [4, 4, 4] tree, (3) [64] tree (star), (4) truncated binary, (5) 64 line, (6,7) (1,1) scale-free graphs, (8,9) truncated (1,1) scale-free graphs, (10) 127 line, (11) [63] tree. Graphs with the base algorithm are denoted with an 'x', resensing with a '+', and reprocessing with a 'o'.

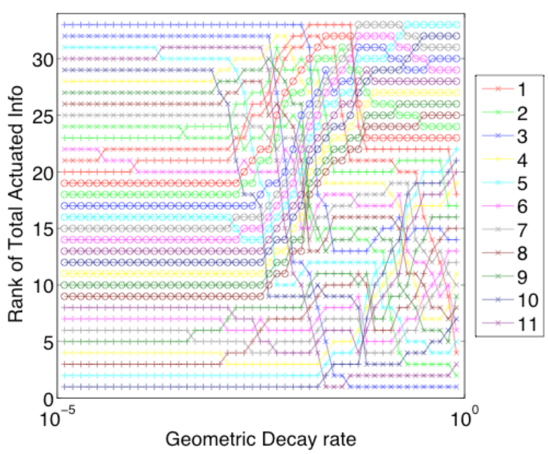


Figure 8: Rank of total actuated information for 11 graphs with logarithmic decay scale. The graphs are indicated by color – (1) [2, 2, 2, 2, 2, 2] (binary) tree, (2) [4, 4, 4] tree, (3) [64] tree (star), (4) truncated binary, (5) 64 line, (6,7) (1,1) scale-free graphs, (8,9) truncated (1,1) scale-free graphs, (10) 127 line, (11) [63] tree. Graphs with the base algorithm are denoted with an 'x', resensing with a '+', and reprocessing with a 'o'.

VII. Control

Control actuations change the state probabilities within a system. Ideal actuations reduce the probability of states that are inconsistent with the actuation to zero. Actuations within a fault-prone system reduce the probability of inconsistent states in proportion to the actuator's probability of failure. This change in probability can be represented as an increase in order, called negentropy [14] for a system with the possible state set X associated with actuation, I_A.

$$I_A(t) = H(x(t)|A),$$

where t is the time at which actuation(s) take effect and t is the time immediately prior to actuation. Likewise a sequence of actuations $A_{1:k}$ will produce negentropic drag Γ_A

$$\Gamma_A(A_{1:k}, t, t') = \frac{H(x(t')|A_{1:k})}{t'-t}.$$

Note that negentropy and negentropic drag measure the increase in order but do not measure the quality of the decision.

VIII. Fitness Revisited

For the simulations presented here the utility of all information is assumed to be equal, however this is not the only utility measure for information. In general, information varies in contextual importance and in time and space. For the purposes of this discussion, assume that the C2 process lasts for finite time, and that we have some utility function $U : X \rightarrow \mathbb{R}$ that assigns a numerical value to the desirability of each system state, based on mission objectives. Then, the expected utility $E[U]$ and conditional expectations $E[U \mid x(t) \in S]$ for some set S are well defined based on the transition probabilities of the process. Furthermore, for some sequence of actuation $A_{1:k}=\{(u_i,t_i)\}$ treated as inputs to the system, $E[U \mid x(t_1), A_{1:k}]$ defines the expected utility of that sequence of actuations.

In this sense, it is the actuations that actually have utility, since they are responsible for restricting the future states of the system, or, for fault-prone systems, increasing the probability of desirable states while decreasing the probability of undesirable states. Thus, the utility of a particular piece of information is dependent upon the influence the information has on actuation. In turn, the value of actuation resulting from an information-based control action is equal to the difference in expected utilities of the control actuations that would be chosen after receiving that information (including time costs and entropic drag) less the utility of the actuation that would have been taken without that information.

If there is no time cost associated with communicating and processing information, the impact on utility of the next control action of additional information is nonnegative. However, if there is entropic drag present in the system, then the decay due to entropic drag in both the communication and processing stages of an incremental piece of information can result in reduced control efficacy, particularly if the information would have minor (or no) impact on the decision process that decided upon to use a specific actuation. This can easily be seen using a uniform utility of information on control actuations, as noted above (see Figure 1). Control can be used to reduce entropy by modifying the probabilities of succeeding system states, for example by restricting the future states of the system so some subset of the state space. However, if the control actuation is chosen so as to maximize the change in expected utility, then by preventing a low utility state of very high probability in favor of multiple high utility, low probability states the entropy of the system could increase as a result of an actuation, even as the expected utility of remaining states increased.

IX. Adaptation and Learning

Another aspect of a C2 process that may be able to be analyzed using information theory is the notion of adaptation and learning in adversary agents. This would result in non-stationary (time-varying) probabilities when conditioned upon prior states, e.g.,

$$P(x_j, t_2 | x(t_1) = x_i) \neq P(x_j, t_2 + t | x(t_1 + t) = x_i).$$

This is problematic, as decisions based on the identical observations (up to a shift in time) would result in identical actuations if a stationary system model is used to predict the effects of possible actuations. However, since the underlying system behavior has changed, the effect of the chosen actuation on the system will be different. Assuming that the adversary agents are learning and adapting to the behavior of the actors, these actuations will be less effective. Similarly, if adversary agents are able to predict the behavior of the information sources, they may be able to reduce the information produced by the sources' observations. Many actuations are designed to reduce uncertainty in the system, and observations (potentially) provide information that necessarily reduces uncertainty. In this sense, the adaptation of the adversary agents serves to reduce the expected reduction in uncertainty (i.e., expected gain in information) when an actuation or observation is performed. We call this phenomenon *evolutionary entropic drag.*

This notion is prevalent in game theory, specifically in the context of competitive repeated games [15][16], where the (mixed) strategies of opponents are allowed to vary with time, and are thus non-stationary. In this context, even if the behavior of the adversary agents is stationary, there is still a notion of uncertainty in the strategy of the adversary that appears as a non-stationary strategy (*strategic entropy*) that can occur due to finite memory not being able to store all past actions of the adversary, for example, if the strategy is dependent on a number of previous states of greater quantity than can be stored in the actors' memory. In the machine learning community, the notion of a non-stationary environment is known as *concept drift* or *concept shift* depending on whether it is a gradual or sudden change. This is of particular concern in the field of online learning, where the target concept to be learned can change (e.g., classifying multiple shapes of varying color by shape changes to classifying them by color [17]. Alternatively, the target concept could stay the same, but the underlying distribution of data could change (*data drift* or *data shift*), resulting in previously learned rules to become less effective [18].

Methods have been proposed for detecting these non-stationary changes using information theoretic criterion, typically based on approximations to the entropy rate [19][20]. The entropy rate of a process is

$$H(X) = lim_{n\to\infty} H(X(t_0), X(t_1), \ldots, X(t_n)),$$

when the limit exists. Techniques from the tracking domain can also be used to deal with non-stationarity, including adaptive filtering (e.g., [21]) and non-stationary models (e.g., jump-Markov or switched linear systems [22]). Since entropy depends only on the probabilities, and not the outcomes, it may be more appropriate to quantify and study the notion of evolutionary entropic drag using a distance metric on the probabilities (taken as a subset of $\mathbb{R}^n$ there are *n* distinct outcomes, or as an element $\mathcal{L}^p$ space for continuous probabilities) or a divergence (e.g., Renyi, Kullback-Leibler).

X. Future Work

This paper describes an approach for using information theory to assess all aspects of coordinated decision-making (the OODA loop) facilitated by C2 structures. Simulation-based experimentation has been used to generate quantitative results that show a relationship between complexity, entropy and entropic drag and situational awareness (the "observe-orient" portion of the OODA loop) produced by C2 structures with varying topologies. Additional experiments produced quantitative results that include "actions", resulting in an "OO-A" loop. A formalism for extending these experiments to include the decision-making process has been introduced. Future work will further refine the information-theoretic models of decision-making strategies and use the models as a basis for additional experimentation that includes the entire OODA loop.

Modeling decision-making processes across a distributed system will require information-theoretic models of multiple decision-making strategies. Of particular interest is a comparison between deliberative planning and training. The concept of evolutionary entropy has been introduced in this paper as an information-theoretic metric for exploring this relationship. Also of interest is a comparison between human in-the-loop decision-making, human on-the-loop decision-making and fully autonomous decision-making (machine intelligence). A series of mixed human-machine experiments are proposed to produce data sets from which models may be constructed.

Abstract, context-independent, models formed the basis of the experiments discussed in this paper. To demonstrate that the experimental results hold for real-world systems additional experiments are planned. In these experiments militarily relevant real-world systems are characterized through information-theoretics and C2 efficacy will be measured through a series of simulation-based experiments. Finally, a prototype *agile* C2 system is proposed. In this proposed system the C2 system uses information- theoretic metrics to conduct real-time assessments of C2 performance and, based upon those metrics, dynamically changes the C2 structure and information exchange strategy.

XI. Conclusion

This paper described a method for evaluating C2 structures based upon information theory. Specifically, complexity, entropy and entropic drag were used to characterize dynamic engagements and C2 performance. Evolutionary entropy, a novel information theoretic metric that we believe can be used to assess the comparative benefits of training vs. planning, was also defined. Simulation-based results demonstrated that correlations do exist between information theoretic measurements of a situation and a C2 structure's effectiveness in addressing that situation. In these experiments it was found that the optimal amount of information used to describe a dynamic scene and the optimal topology used to coordinate units operating in that scene varied as the information-theoretic metrics of the scene varied. In particular, for certain information-theoretic characteristics, binary trees and lattice-based topologies out-performed the scale-free topologies, and vice-versa. Based on these results we can confidently state that the structure used by an organization to effect command and control should vary as the situation changes and, further, that information-theoretic metrics can be used as a basis for selecting a C2 strategy.

Acknowledgment

This research was performed for the Office of Naval Research under contract N00014-0-C-0396 and under Johns Hopkins University Applied Physics Laboratory internal research grant X9RSTAE2. The authors would like to thank Frank Ferrese, Matt Johns, and Evan Sultanik for their comments and insight.

Appendix A

List of Topologies Used in Information Flow Simulation

1. Fully connected
2. Star ([126] tree)
3. Binary tree ([2,2,2,2,2,2] tree)
4. [3,3,3,3,3,3] tree truncated to 127 vertices
5. [4,4,4,4] tree truncated to 127 vertices
6. [2,3,4,5] tree truncated to 127 vertices
7. [11,11] tree truncated to 127 vertices
8. 1-ring
9. 2-ring
10. 3-ring
11. (1,1) scale-free
12. (1,1) scale-free
13. (3,3) scale-free
14. (3,3) scale-free
15. (5,5) scale-free
16. (5,5) scale-free
17. 1-ring+10
18. 1-ring+10
19. 1-ring+15
20. 1-ring+15
21. 1-ring+15
22. 1-ring+5
23. 1-ring+5
24. 3-ring+10
25. 3-ring+10
26. 3-ring+15
27. 3-ring+15
28. 3-ring+5
29. 3-ring+5
30. 11 by 12 grid truncated to 127 vertices

Appendix B

List of Topologies Used in Information Flow With Actuation Simulation

1. [2,2,2,2,2,2] tree
2. [4,4,4] tree
3. [64] tree (star)
4. [2,2,2,2,2,2] tree truncated to 64 vertices
5. 64 vertex path graph
6. (1,1) scale-free
7. (1,1) scale-free
8. (1,1) scale-free truncated to 64 vertices
9. (1,1) scale-free truncated to 64 vertices
10. 127 vertex path graph
11. [63] tree

References

[1] Alberts, D. & Hayes, R. (2003). *Power to the Edge*, CCRP Press.

[2] Alberts, D. & Hayes, R. (2007). *Planning Complex Endeavors,* CCRP Press.

[3] Cebrowski, A. K. & Garstka, J. J. (1998). Network-Centric Warfare: Its Origin and Future, *Proc. of the U.S. Naval Institute* Jan.

[4] Barabasi, A., Albert, R. (1999). Emergence of Scaling in Random Networks, *Physica A*, 272, 173–187.

[5] Boyd, J., *The Essence of Winning and Losing*, http://belisarius.com/modern_business_strategy/boyd/essence/eowl_frameset.htm

[6] Shannon C. E. (1948). A Mathematical Theory of Communication. *The Bell Systems Technical Journal*, vol. 27, pp. 379–423 and 623–656, July and October 1948.

[7] Touchette, H., & Lloyd, S. (2004). Information-theoretic approach to the study of control systems. *Physica A: Statistical Mechanics and its Applications , 331*, 140–172.

[8] Touchette, H., & Lloyd, S. (2000). Information-theoretic limits of control. *Physical Review Letters , 84*, 1156–1159.

[9] Scheidt, D., & Pekala, M. (2007). The impact of Entropic Drag on Command and Control. *12th ICCRTS : Adapting C2 to the 21st Century*. Newport, RI: DoD Command and Control Research Program.

[10] Watts, D. J., & Strogatz, S. H. (1998). Collective dynamics of 'small-world' networks. *Nature* , 440–442.

[11] Newman, M. E., & Watts, D. J. (1999). Renormalization group analysis of the small-world network model. *Physics Letters A* , 341–346.

[12] Monasson, R. (1999). Diffusion, localization and dispersion relations on "small-world" lattices. *The European Physical Journal B - Condensed Matter and Complex Systems , 12*, 555–567.

[13] Barabasi, A. L., & Albert, R. (1999). Emergence of scaling in random networks. *Science , 286*, 509–512.

[14] Brillion, L. *Science and Information Theory* 2nd Ed., Dover Publications Inc.

[15] Neyman, A. & Okada, D. (1999). Strategic entropy and complexity in repeated games, *Games and Economic Behavior*, 29 191–223.

[16] Neyman, A. & Okada, D. (2009). Growth of strategy sets, entropy, and nonstationary bounded recall, *Games and Economic Behavior,* 66 404–425.

[17] Nishida, K., Shimada, S., Ishikawa, S., Yamauchi, K. (2008) Detecting sudden concept drift with knowledge of human behavior, in:.*IEEE International Conference on Systems, Man and Cybernetics, 2008*, pp. 3261–3267.

[18] Kivinen, J., Smola, A., Williamson, R. (2004). Online learning with kernels, *IEEE Transactions on Signal Processing*, 52, 2165–2176.

[19] Vorburger, P., & Bernstein, A. (2006). Entropy-based concept shift detection, in: *Proceedings of the Sixth International Conference on Data*

Mining, ICDM '06, IEEE Computer Society, Washington, DC, USA, pp. 1113–1118.

[20] Bollt, E. M., & Skufca J. D. (2009). Control entropy: A complexity measure for nonstationary signals, *Mathematical Biosciences and Engineering*, 6, 1–25.

[21] Myers, K. & Tapley B. (1976). Adaptive sequential estimation with unknown noise statistics, *IEEE Transactions on Automatic Control*, 21,, 520–523.

[22] Doucet, A., Gordon, N., Krishnamurthy V. (2001). Particle filters for state estimation of jump markov linear systems, *IEEE Transactions on Signal Processing*, 49, 613–624.