Using Cognitive Communications to Increase the Operational Value of Collaborative Networks of Satellites

Ryan B. Linnabary
Department of Electrical and Computer Engineering
The Ohio State University
Columbus OH, USA
linnabary.24@osu.edu

Andrew J. O'Brien
Department of Electrical and Computer Engineering
The Ohio State University
Columbus OH, USA
obrien.200@osu.edu

Graeme E. Smith
Department of Electrical and Computer Engineering
The Ohio State University
Columbus OH, USA
smith.8347@osu.edu

Christopher Ball
Department of Electrical and Computer Engineering
The Ohio State University
Columbus OH, USA
ball.51@osu.edu

Joel T. Johnson
Department of Electrical and Computer Engineering
The Ohio State University
Columbus OH, USA
johnson.1374@osu.edu

***Abstract*—Future Earth-observing systems will involve distributed missions of autonomous and heterogeneous spaceborne sensor networks. New systems will occupy a more complex decision space and will be capable of adaptive decisions and collaborative networking. These capabilities must be utilized to increase a network's operational value and to reduce human involvement in high-level communications decisions. Developing the required technology for both ground operations and space deployment may be greatly accelerated by machine learning techniques and cognitive communications algorithms. The following research describes the development and analysis of simulated sensor networks which employ a cognitive communications model. Results from these simulations provide training data for machine learning models which are used to explore the expanded decision space.**

***Keywords*—Autonomy, Sensor Network, Software, Remote Sensing, OSSE**

I. Introduction

It is envisioned that NASA's future space systems will be composed of large, inhomogeneous networks of small satellites and autonomous platforms. These resource constrained systems, carrying an array of different instruments, will be expected to operate autonomously and collaboratively to achieve mission and science goals. Unfortunately, current and near-future inter-satellite communications are highly constrained in terms of link availability, reliability, power and bandwidth. Although future technologies (such as free space optical links) may alleviate some constraints, it is expected that future instruments will rapidly expand in both data volume and sensor reconfigurability. In this way, it is not sufficient to simply increase the capabilities of the communication links. Rather, it is also necessary to improve the complex decision making that communication systems perform, such as deciding when to transmit, what information is valuable to nodes of the network, and how to adapt local operations following the reception of new information.

Recently, cognitive space communication algorithms have been proposed as a solution to address the complexity of future inter-satellite communication systems. Typically, these cognitive algorithms have tried to address communications at a low level and include decision making regarding modulation, power and bandwidth, and error rate. However, it is reasonable to expect that cognition may also offer an improvement in the complex, higher level decisions of communication in the context of mission and science objectives. At this level, cognition is applied to the operation of the network with the decision making primarily influenced by the constraints of the space communication network links.

In this work, we show results of simulation studies to explore the advantages that cognition could offer for collaborative small-satellite networks. Under a NASA Advanced Information System Technology program, we are currently developing an open-source C++ library for the simulation of autonomous and collaborative networks of adaptive sensors. This library and accompanying utilities allow for the efficient simulation of networks of satellites with realistic constraints in communication, power, and measurements. A key focus of this software is the simulation of sensors that operate adaptively. Adaptive sensors must make intelligent decisions regarding their configuration based on their own measurements as well as the measurements provided by other sensors in a network. However, the extreme complexity of the decision space makes the development of optimal decision-making systems very

This research was sponsored by the NASA Advanced Information Systems Technology Program NNH16ZDA001N-AIST.

978-1-7281-0048-7/19/$31.00

difficult. Thus, an approach based on cognition could offer an appealing solution. We investigate how our simulation tools could be useful for production of large training datasets that capture the operation of collaborative, adaptive networks of small satellites. We then investigate how such a dataset could be combined with machine learning techniques to train neural networks that could make intelligent decisions about when and what to communicate. Results from our investigation will be presented, and the applicability of these methods to future cognitive space communication will be discussed.

II. Overview Of Collaborative Networks Of Adaptive Sensors

- Standard stuff from the AIST proposal and IGARSS papers

III. Cognition and High-Level Communications

- What it means for low-level and high-level communications
- Cognitive communications communications
- What are high-level aspects of comm: when and where to move information given large scale constraints on the network. What are tuning parameters at this high level?
- Table of low level and high level comm issues
- Summary of different methods for dealing with this problem. Can ML help?

Machine learning provides computers the ability to learn without being explicitly programmed. Such intelligent machines are capable of accurately predicting and classifying new data. Computers can be programmed to learn autonomously with or without supervision, and often require significant quantities of training data and careful adjustment of model parameters. These techniques are useful for the problems which: require many manual adjustments or long lists of rules; operate in a fluctuating environment; need to process a large amount of data; and/or have no known "good" solution.

TABLE I
Low vs. high level communication issues

Low Level	**High Level**
Bandwidth	Scheduling
Error Rate	Packet Contents
Latency	Satellite Health
Path Loss	
Center Frequency	
Doppler	
Modulation	
Power	

Machine learning techniques are currently being applied in the development of scheduling algorithms, sensor network design, and other applicable areas. An opportunity exists to use machine learning algorithms to optimize the flow of information within a collaborative network of satellites. Optimization will increase communications efficiency by increasing the value of data contents and reducing power consumption.

All of the common machine learning algorithms apply to optimization of high-level communication parameters. Regression tasks enable satellites to autonomously adjust parameters for communication, sensing, and on-board data processing. Classification of network nodes based on proximity and capabilities could increase efficiency in communications decisions.

This research involves the application of existing machine learning tools as well as the implementation of new machine learning algorithms. A group of third-party Python packages is used, including SkikitLearn and Tensorflow. The custom algorithms discussed are written in C++ .

A. Cognitive Communications

A cognitive entity is capable of taking action based on its goals and perception of the environment, potentially learning from the result of its action. It is an intelligent entity which posesses perception, learning, reasoning, and making decisions.

In the current context, cognitive communication involves the intelligent routing of information within an autonomous satellite sensor network for collaborative improvement in mission science return. This concept adds collaboration to "adaptive remote sensing" and feedback to traditional inter-satellite "communication".

- Example references of how it has been applied
- Insert figure showing flow chart of problem formation and solution using the ML approach
- Generation of training data, training of the NN, application of the NN to the system. What are input and outputs. What are the steps? List of steps in proposed procedure if needed
- Proposed cognitive communcaitons flow chart.

IV. Sensor Network Simulations to Support Machine Learning Research

Research in applying machine learning to sensor networks will rely on simulations to validate algorithms both deployed on spacecraft and on the ground. Simulations must provide the means for basic cognitive communication as well as the production of training data for post-processing by neural networks. Training data should include simulation variables which are suspected of correlation to the parameter being optimized. Variables may capture time-series data involving satellite position, health, communication hardware details, sensor hardware details, network connectivity, or other similar parameters.

A software tool-set COLLABORATE is under development which is capable of producing the described training data. The toolset has two main components: first, a C++ development library for observing system simulation experiments; and second, a Python visualization and analysis package for post-processing of data. The project is published to a Git repository under the GPLv3.0 license.

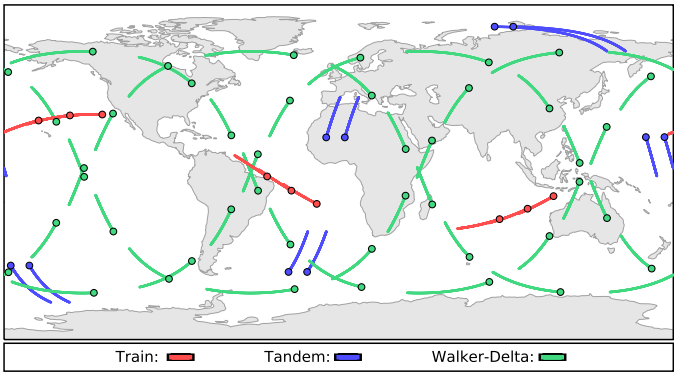
The COLLABORATE library offers a number of unique features valuable to future observing system simulation experiments. At its core, it is a physics engine for satellite position, velocity, and attitude. Power and RF accessories may be attached to satellites and individually oriented. The next level involves rapid constellation design. Standard orbit models described by two-line-element (TLE) sets are provided, copied, and modified to generate novel and interesting constellation patterns. Examples are illustrated in Fig.1(a).

Sensor hardware is attached to satellites as an interface to truth data (NetCDF Nature Run data). This provides a custom modeling environment for real sensor hardware and enables heterogeneous sensor constellations with different capabilities. As a satellite orbits, its pointing vector intersects Earth's surface or an atmospheric layer and samples the underlying data, as shown in Fig.1(b-d).

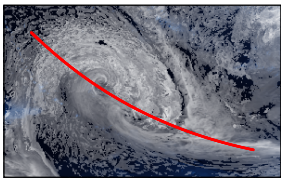
COLLABORATE is named for its ability to manage collaborative networks of satellites. Its implementation focuses on the high-level communication decision space previously discussed. The library employs, in addition to standard C++ components, advanced data structures including trees and graphs to execute predictive route-finding algorithms for efficient communications. For example, line-of-sight wireless channels are captured in a graph, as illustrated in Fig.1(e), the minimum spanning tree.

A predictive scheduling algorithm is illustrated in Fig.1(f). Satellite "A" measures cloud depth in the Pacific Ocean (blue line). It then predicts the arrival of a follow-up satellite and relays a message through satellites "B" and "C" to queue "D" for a measurement with a different sensor (red line). Significant work has been done to optimize this algorithm, because it is the main consumer of CPU time and is run regularly in simulation to find routes. It is also the foundation for deployed cognitive communications, as discussed in Section [TODO].

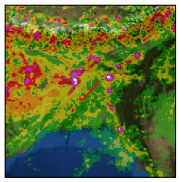
Presently, the included algorithms are iterative and often take minutes to conclude. It may be possible to reduce runtime or optimize routes based on alternative parameters using machine learning. An option is to replace these algorithms with predictive neural networks for advanced regression or classification algorithms. COLLABORATE facilitates this by producing data in standard formats for post-processing.



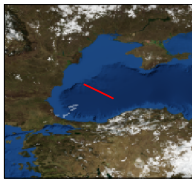
(a) Various constellations defined by orbit patterns



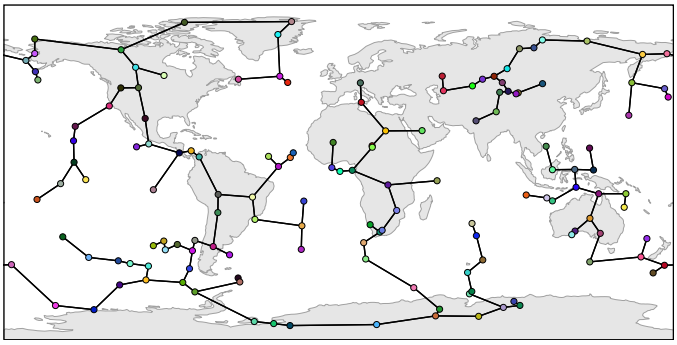
(b) Cloud Depth



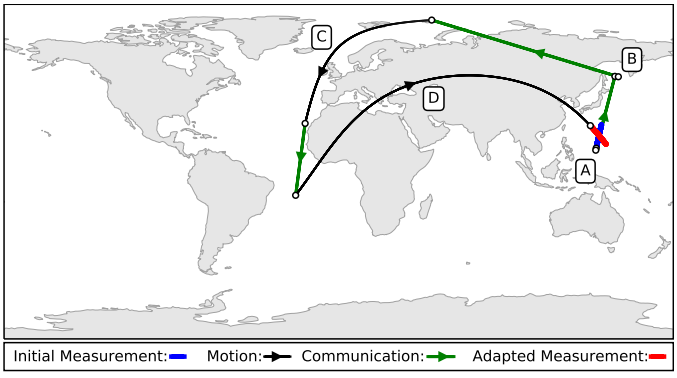
(c) Precipitation



(d) Optical Images

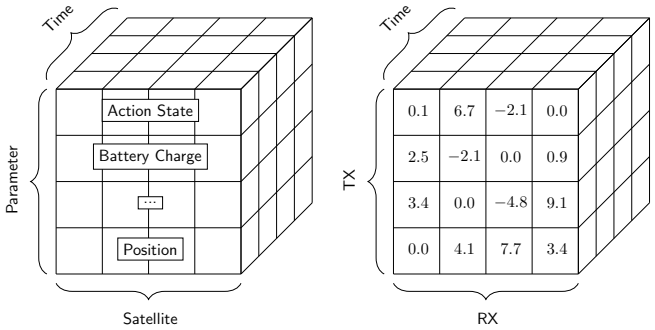


(e) Network graph structure (minimum spanning tree)



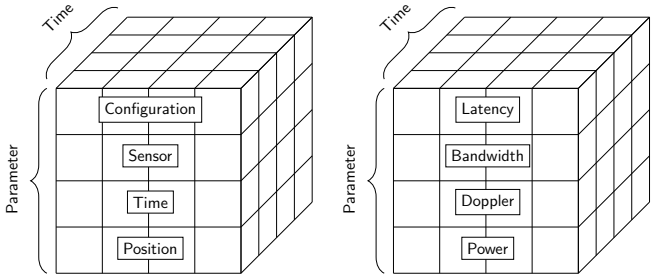
(f) Collaborative sequence in time

Fig. 1. Collaborate software library features



(a) Satellite parameter data

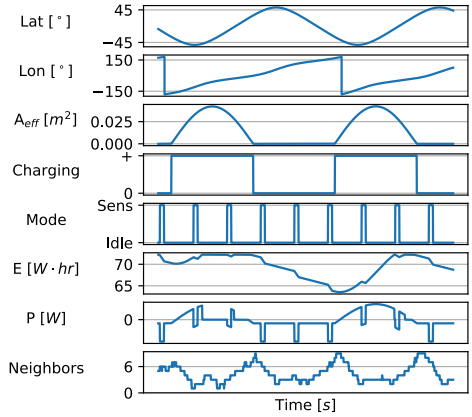
(b) Network parameter data



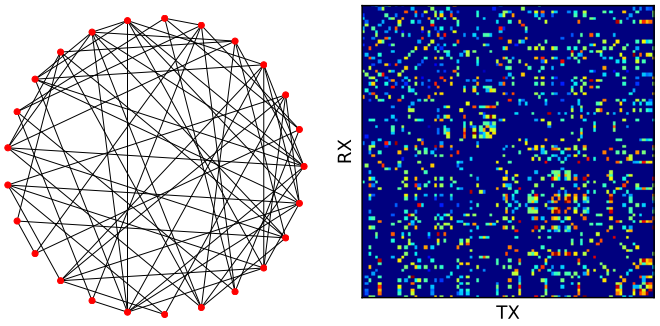
(c) Satellite measurement data

(d) Comm. channel data

Fig. 2. Simulation training data formats



(a) Satellite parameter data



(a) Line-of-sight connections

(b) Channel RF gain

Fig. 3. Visualized simulation data

The software logs simulation data to files accessible by external machine learning tools. COLLABORATE was developed around simple data formats for portability and to promote development of custom analysis tools. Primarily, data is serialized and written to binary files. These formats are well documented and easy to parse in Python or other scripting languages. Included Python packages understand the data formats and can read and store the data for later use as Numpy or Pandas data structures. Examples include time-series data frames or network adjacency matrices (weighted and unweighted). Fig.2 shows several common data structures in memory.

Python scripts are provided not only for receiving simulation data at a low level, but also many high level analysis tasks. In fact, all figures in this document were produced using the tools provided by the library. Primary third-party packages used include the following: Numpy, Pandas, Cython, NetCDF4, Matplotlib, Cartopy, Scikitlearn, TensorFlow, and SciPy. These enable post processing for plots and animations or to train machine learning algorithms. Numpy and Pandas provide powerful linear algebra and statistics operations. Cartopy provides extensive map projections and transformations which support visualizing satellite positions and truth data. Machine learning algorithms are available in the Scikitlearn and Tensorflow packages, which interface well with Numpy and Pandas structures.

For example, satellite parameter data (Fig.2(a)) is plotted in Fig.3(a) to expose and potentially exploit correlations. Several of these seem strongly correlated and many are also periodic. Potential high-level communications optimizations may involve predicting when a satellite has the most visible neighbors (available line-of-sight links). An algorithm for power management scheduling may use the instantaneous charge or power to plan efficient sensor operation.

Another visualization involves network structure. Fig.3(b) and (c) contain plots of unweighted and weighted adjacency matrices. A built-in Python library called NetworkX produced these graphs and works well with the adjacency matrix format.

Example plots.

Example plots.

Example plots.

V. Example Case Studies

Description of examples and why they were chosen.

A. Example 1: Cognitive Feedback For Deployed Regression Algorithm

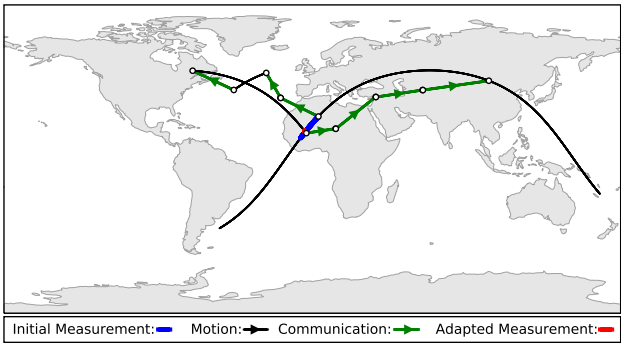


Fig. 4. Various constellations defined by orbit patterns

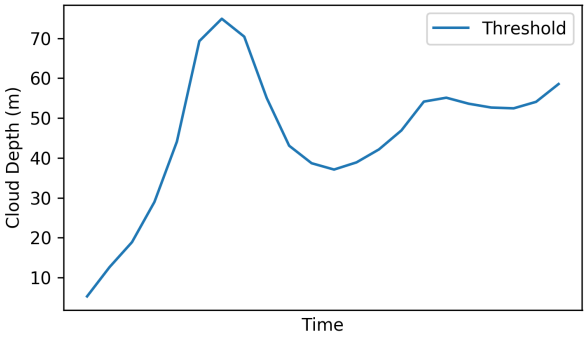


Fig. 5. Various constellations defined by orbit patterns

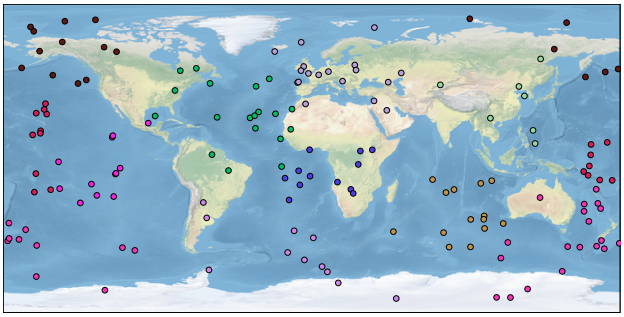


Fig. 6. Various constellations defined by orbit patterns

B. Example 2: Spectral Clustering Using Simulated Network Data

Something

VI. Summary and Next Steps

...

VII. References

...