Proceedings of the 2010 Winter Simulation Conference
B. Johansson, S. Jain, J. Montoya-Torres, J. Hugan, and E. Yücesan, eds.

MODELS AND METRICS OF GEOMETRIC COOPERATION

Chris Arney
Kristin Arney
Elisha Peterson

Department of Mathematics
United States Military Academy
West Point, NY 10996, USA

ABSTRACT

A basic way that entities can cooperate with one another is by sharing of tasks through synchronized movement to balance their geometric load. For example, players of a team defending a goal may be assigned equal-spaced zones to defend or units in a military force may be assigned equal-spaced sectors to control. As the dynamics of the situation unfold and as entities move, withdraw, or enter the space; the other entities cooperate by adjusting their positions to retain load balance. Various ways that this geometric cooperation can be accomplished, both from the perspectives of central and local control, are developed, analyzed, and simulated. This problem is related to other geometric cooperation problems such as movements in multi-player pursuit-evasion games and balancing loads for other generally non-geometric algorithms. The authors use the metrics to establish a framework for a theory of geometric cooperation. Simulations, metrics, and results of the algorithms' performance in various scenarios are presented.

1 INTRODUCTION

The major theme of this paper is cooperation – specifically, the way entities cooperate by sharing tasks through synchronized movement to balance their geometric load. This kind of cooperation is motivated by military units assigned equal-spaced sectors to control or sports players assigned equal areas to cover or defend. This paper provides the background, problem, algorithms, results, and analysis of an investigation in geometric cooperation in the form of equi-distribution of dynamic entities in a geometrical situation. The essence of the problem is to have mobile entities maintain equal-area spacing in an assigned geometry as the dynamics of the situation unfold -- entities move, withdraw, or enter the space.

The Army's myriad networks connect battlespace entities (not only humans and soldiers, but also machines, computers, and robots). Many of the social/biological/physical/informational connections in these networks are in the form of cooperation – entities working together to achieve a common goal. A major component of net-centric warfare is to "interact and collaborate in the virtual (informational) domain" (Alberts, Garstka, and Stein 1999). At the most basic level, cooperation can be traced to geometric synchronization of entities (Strogatz 2003). At the highest levels of application, businesses are embracing the concept with the hope of inventing "organizations for the twenty-first century that will not only be more economically productive but also more humanly desirable" (Malone, Laubacher, and Morton 2003; Tapscott and Williams 2006). So how does this complex web of connections, relationships, collaborations, and communities of diverse entities work? What are the metrics for a successful cooperative network? What is cooperation and what makes a cooperative system or organization effective? What forms of communication within a network can enhance cooperation? How do we measure trust and selflessness? Can a cooperative autonomous system be as effective as a centrally controlled system? Can net-

work structures, processes, tools, connections, communications, and languages enhance cooperation, achieve synergy, and optimize networks?

Cooperation is not yet well-understood from a mathematical point-of-view. The most classical mathematical construct in this area is John von Neumann's "cooperative game theory", introduced in the early 20th century (Neumann 1928). However, this concept applies primarily to a group of entities working together for selfish reasons. In contrast, most forms of cooperation in social and military networks involve a team working together for a common good. In reality, von Neumann's theory only applies to certain kinds of cooperation. This work starts at the root level of cooperation to establish the basic framework.

The framework of subset team games (Arney and Peterson 2008) provides a foundational tool that can be used to study some of these questions about networks and cooperative system. The framework builds upon existing metrics of "team success" to provide complementary metrics of "altruistic" or team-oriented cooperation and "selfish" cooperation. The result is a tool that can provide some powerful insight into altruism, teamwork and cooperation. (Axelrod 1996, Axelrod and Cohen 2000)

In this paper, we demonstrate several examples in basic geometric cooperation. We seek to build a foundation by establishing a simulation tool to study these questions about networks and cooperative systems. In this paper, we demonstrate several examples in basic geometric cooperation. Using a game theoretic construct, the basic assumption is that the players (or entities) join together as a team, working for the best team payoff or metric performance. In this case, the goal of the entities (or players) is to maintain their equal spacing as the situation changes. There are many examples of such situations: players on sports teams assigned to defend areas that shift in importance as the game is played; groups in organizations assigned to perform tasks that can be assigned geometrically; military units assigned to control or observe assigned areas of operations where assignments need to shift as the situation changes; and mobile sensors placed on the battlefield to monitor and sense equally the entire battlespace.

2 GEOMETRIC EQUI-DISTRIBUTION PROBLEM

2.1 Equal Areas

In this paper, we focus on cooperation for the simple task of determining geometric assignments in a polygonal region to establish regions of equal area. There are some static algorithms that try to use establish shapes (triangles, polygons) to divide the region (Ben et al. 2006, Snyder 1992) Each entity (player or agent) is assigned a particular region within the polygon, and the goal is for the entities (players) to sense their situation and move autonomously to equalize their assigned areas. Several agent-based algorithms are developed and run in simulations to test the performance and convergence of the algorithm. Metrics are computed to determine how successful the entity distribution is at any stage of the dynamic situation.

2.2 Metrics

The metrics that are used are normalized with respect to the mean area of the sub-regions and the number of entities to capture the deviations in the areas of responsibilities as ratios. Let $\bar{A}$ represent the mean sub-region area, and let $A_1, A_2, \ldots, A_n$ represent the areas assigned to individual players. We track three measures of deviation:

1) The maximum deviation from the mean as a ratio with mean sub-region area: $\max_i |A_i - \bar{A}| / \bar{A}$.
2) The average deviation from the mean as a ratio with mean sub-region area: $\frac{1}{n} \sum |A_i - \bar{A}| / \bar{A}$.
3) The sum of the mean-squared deviations as a ratio with the square of the mean sub-region area: $\frac{1}{n} \sum (A_i - \bar{A})^2 / \bar{A}^2$.

By tracking these normalized utility metrics (ratios or percentages), we are able to normalize the algorithms behavior independent of the size of the region or the number of points used in the simulation.

2.3 Basic Geometry

The simulations are performed by establishing a polygonal region and initially randomly dispersing a specified number of points (representing entities or players) in the region. A Voronoi diagram then partitions the overall region into sub-regions where each point is responsible for the area where it is the closest point. (Ohyama 2007) This geometric framework is shown as the "meshed" region in Figure 1. In a dynamic assignment game, the areas would then be assigned to each given point as it cooperates to maximize the utility function. The simulations implement various algorithms to dynamically adjust the point locations to make the areas equal.

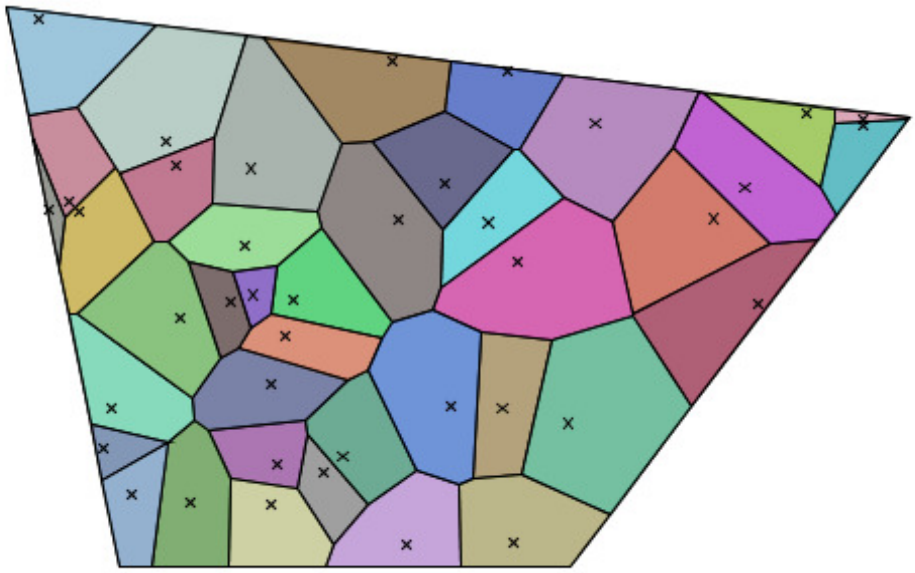

Figure 1: Initial "meshed" region with 40 points (unequal areas). This region has poor utility metrics --- e.g., the maximum assigned area for the largest entity is 140% of the mean area.

3 ALGORITHMS

We begin with the premise that when the autonomous entities in a region sense they are no longer in balance in terms of equal areas of responsibility, they attempt to move to equalize the areas using only local information. To do this, the entity determines local factors (distances or areas or loads of neighboring entities) and seeks to move in a way to reduce the imbalance. For organization purposes, we classify the balancing algorithms as static (based on local static factors -- areas of neighboring sub-regions) or dynamic (based on how neighboring elements are moving – distances from neighbors). We present two algorithms of both types for a total of four algorithms and label them S-1, S-2, D-1, and D-2.

3.1 Static Algorithms

The two static (local, area-based) algorithms are described as follows:

- S-1: Each entity moves toward the neighboring sub-region with largest area in an attempt to increase its own area and decrease the area of the larger-area neighbor.
- S-2: Each entity moves in a direction found by weighting the differences in the region's area with all its adjacent neighbor areas. If the neighbor has area A and position $\vec{x}$, and its neighbors have areas A_i and positions $\vec{x}_i$, then the entity will move its position by

  $$\gamma \sum_i (A_i - A) \frac{\vec{x}_i - \vec{x}}{||\vec{x}_i - \vec{x}||},$$

  where γ is a constant parameter. This is equivalent to gradient-based local optimization of the function $\sum_i (A_i - A)(\vec{x}_i - \vec{x})$.

In these static algorithms, an entity uses information about its neighbors' positions and areas only.

3.2 Dynamic Algorithms

The two dynamic (local distance- and movement-based) algorithms are described as follows:

- D-1: Each entity moves with the average of all neighbor movements (iterative scheme) if its area is close to the mean sub-region area. If the area is too much larger than this target area, or too much smaller, then its behavior depends upon whether or not its own region is part of the boundary of the larger polygonal region. If it is a boundary point, then it will move toward the boundary if its area is too large or away from the boundary if its area is too small. If it is not a boundary point, then it will move toward its farthest neighbor if the area is too large or away from the closest neighbor if its area is too small.
- D-2: Each entity moves with the average of its *two* closest neighbors if they are far enough away, but moves directly away from its nearest neighbor if it is too close. After this initial movement stage, entities that are too close to the boundary move directly away from the boundary.

In these dynamic algorithms, an entity has information about its target area, the mean sub-region area. But it does not use information about its neighbors' areas, but rather information about its neighbors' positions and movement in past iterations. Each parameter in the algorithm, e.g. how close is "too close" for two neighbors and the threshold areas, remain fixed throughout a simulation.

3.3 Equi-Distribution Platform

The simulations are managed through our Java-programmed Equi-Distribution Platform that enables us to establish the initial polygonal region, set the number of sub-regions/entities/points, set the parameters, and establish graphic images through Voronoi diagrams. Once the initial system is established the platform enables us to select and run the algorithm (either continuously or step by step) and watch the movement of the entities and monitor the metrics of the simulation (to include plots of metrics over time). A visual image of the complete graphic platform interface is shown in Figure 2.

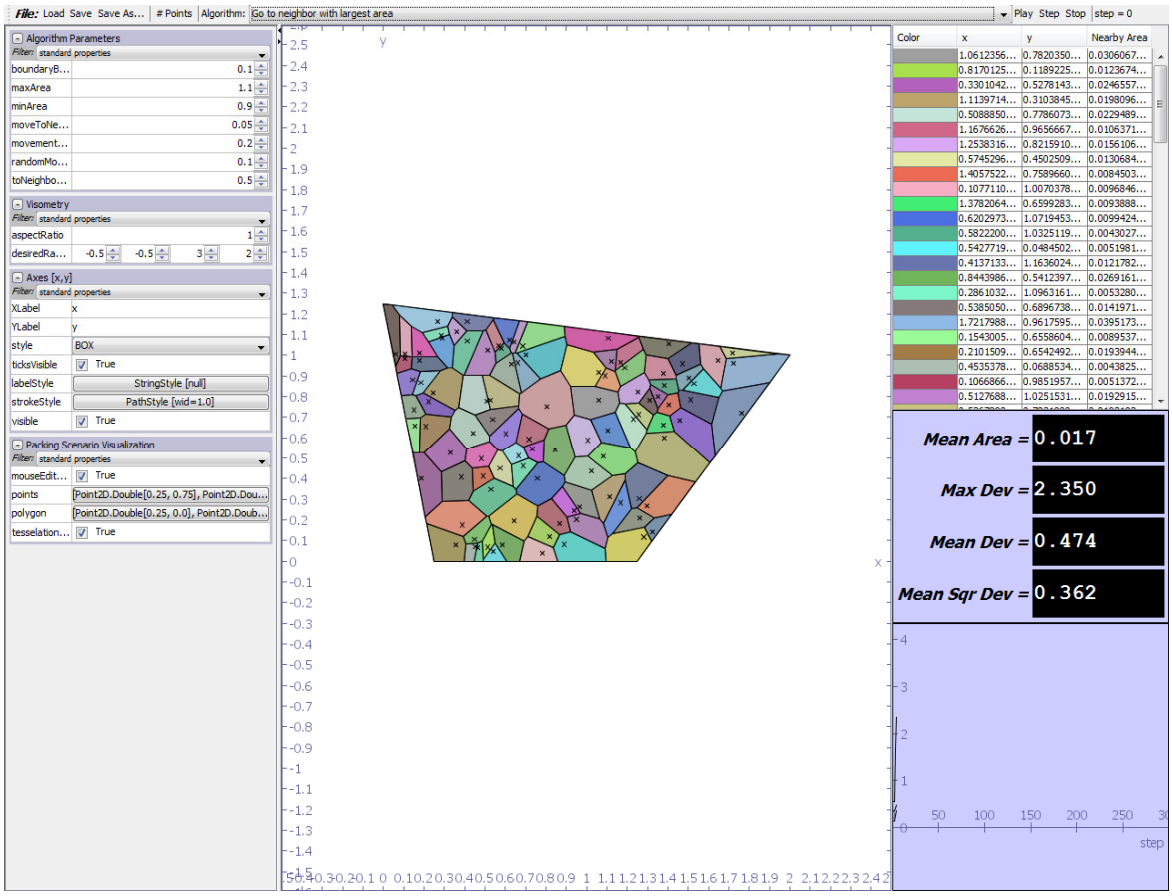

Figure 2: JAVA-based simulation platform for geometric distribution algorithms

Our perspective was that because these algorithms are only locally based, convergence to an optimal mesh would be slow and difficult to achieve even using the areas as input parameters in algorithms S-1 and S-2. More in keeping with the idea that the entities should be able to directly sense the imbalance, algorithms D-1 and D-2 use the distances to neighbors as their primary input parameters. While there is some correspondence between distance and area, we were unsure that these local algorithms would lead to adequate or reasonable behavior.

4 RESULTS

4.1 Basic Convergence Results

We have run the four algorithms to test their performance in various scenarios. We did this for 40 entities (with two different initial conditions), 100 entities, and 400 entities, using all four of our algorithms in each case. An image of the region with a sample converged mesh (40 areas or sub-regions that are equal) is shown in Figure 3. The results of the simulations for the four algorithms for 40 entities (2 different initial meshes), 100 entities, and 400 entities are provided in Table 1. Additionally, the performance and convergence data for the simulations for algorithm S-2 are provided in Table 2.

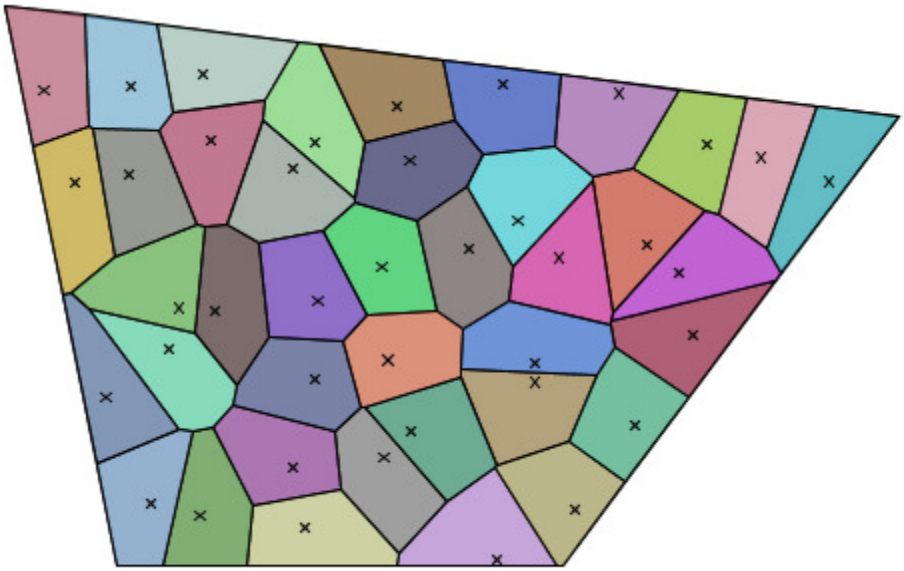

Figure 3: Converged region with 40 points arrayed in locations to produce equal areas of responsibility.

Table 1: Data for all three deviation metrics for the simulations for all four algorithms (S-1, S-2, D-1, D-2) and all four initial meshes (40, 40, 100, 400 points).

			After 500 steps			
		Initial	**S1**	**S2**	**D1**	**D2**
CASE 1 : 40 points	**Mean Area**	0.043				
	Maximum Deviation	1.417	0.955	1.198E-04	0.641	1.154
	Mean Deviation	0.429	0.205	2.18E-05	0.117	0.376
	Mean Squared Deviation	0.292	0.07	1.25E-09	0.028	0.208
CASE 2: 40 points	**Mean Area**	0.043				
	Maximum Deviation	1.191	1.061	2.18E-07	0.208	1.038
	Mean Deviation	0.303	0.308	6.47E-08	0.084	0.252
	Mean Squared Deviation	0.147	0.156	7.58E-15	0.011	0.11
100 points	**Mean Area**	0.017				
	Maximum Deviation	2.35	0.895	5.20E-02	0.583	1.231
	Mean Deviation	0.474	0.212	4.00E-03	0.124	0.268
	Mean Squared Deviation	0.362	0.073	7.18E-05	0.027	0.125
400 points	**Mean Area**	0.004				
	Maximum Deviation	2.572	0.904	1.90E-01	0.614	4.281
	Mean Deviation	0.428	0.149	1.40E-02	0.132	0.351
	Mean Squared Deviation	0.314	0.045	5.04E-04	0.029	0.274

S-2 performance (converging at order $n^{-0.5}$ using local neighbor area data) is impressive and shows that the entities are cooperating in an efficient manner. Likewise, D-1's performance using only local distance data was effective as well. While this algorithm did not converge, it controls the deviations of the sub-regions and reaches an oscillatory equilibrium. Overall, these two algorithms appear to exhibit cooperative behavior that provides for geometric synchronization of the system.

Table 2: Performance metrics for algorithm S-2.

			Number of Steps				
		Initial	**10**	**50**	**100**	**250**	**500**
CASE 1: 40 points	**Maximum Deviation**	1.417	0.332	0.057	0.018	0.002	1.20E-04
	Mean Deviation	0.429	0.094	0.011	0.004	2.97E-04	2.18E-05
	Mean Sq. Deviation	0.292	0.014	2.47E-04	2.62E-05	2.48E-07	1.25E-09
CASE 2: 40 points	**Maximum Deviation**	1.191	0.167	0.114	0.069	1.66E-04	2.18E-07
	Mean Deviation	0.303	0.051	0.012	0.009	4.65E-05	6.47E-08
	Mean Sq. Deviation	0.147	0.004	5.96E-04	2.88E-04	3.70E-09	7.58E-15
100 points	**Maximum Deviation**	2.35	0.521	0.288	0.064	0.037	0.052
	Mean Deviation	0.474	0.152	0.063	0.024	0.005	0.004
	Mean Sq. Deviation	0.362	0.037	0.006	7.93E-04	5.42E-05	7.18E-05
400 points	**Maximum Deviation**	2.572	1.028	0.437	0.389	0.257	0.190
	Mean Deviation	0.428	0.218	0.093	0.054	0.014	0.014
	Mean Sq. Deviation	0.314	0.078	0.013	0.005	0.001	5.04E-04

4.2 Cooperation Simulation

In the following simulation, different individual entities were programmed to follow different algorithms forming teams whose goal was to converge (the utility function measures how close each entity is to the mean or equi-distribution). The entities were scored as to their altruistic and selfish contributions and teams were scored as to their overall performance. Figure 4 shows the evolution of average team scores for five teams of various player (algorithm) composition, with a maximum/ideal score of 20 and the average taken over 200 sets of initial positions.

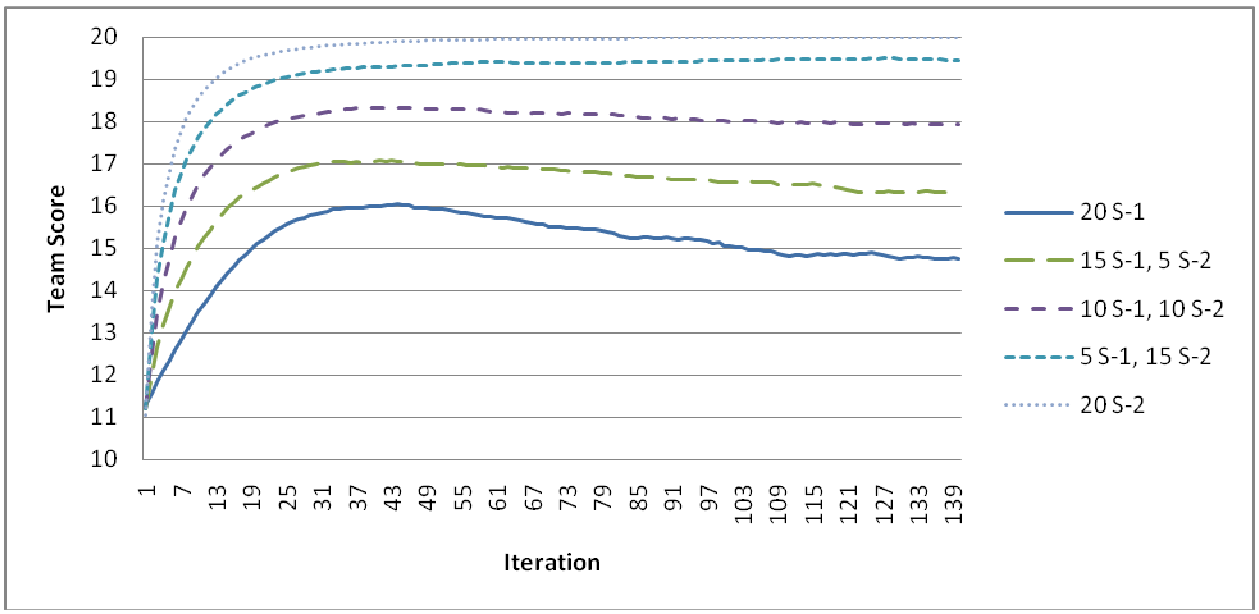

Figure 4: Teams scores reflecting average closeness to equi-distribution for five teams of 20 players each with various compositions of players using the S-1 and S-2 algorithms (as indicated in the margin). Each line represents the average of 200 runs for different sets of initial conditions, with each run consisting of 140 iterations.

It is clear from Figure 4 that the S-2 algorithm performs better on a homogeneous team than the S-1 algorithm, but it is less clear to what extent the individual players (algorithms) contribute to the overall team score on a heterogeneous team. The framework of subset team games (Arney/Peterson 2008) provides one way to compare the contributions of the different algorithms to the common team. Within this framework, a player's overall contribution is determined by the difference between team score with and without that player. That contribution is further subdivided into a *selfish* contribution, indicating to what extent that player's contribution helps itself, and an *altruistic* contribution, indicating to what extent that player's contribution helps out the other players on the team.

On this individual level, the results indicate that the S-2 algorithms perform better than the S-1 algorithms on heterogeneous teams, in *both* their selfish and altruistic contribution. Figure 5 shows a comparison of the altruistic and selfish scores of the S-1 and S-2 players on a common team, in this case a team of 20 entities with 10 S-1 and 10 S-2 players. The selfish contribution of S-2 is better on average, and the gap widens over time. The altruistic contribution of S-2 is also better on average, and increases over time. Similar differences in cooperation utility were also apparent in all other team compositions of players performing the S-1 and S-2 algorithms.

These results indicate that the S-2 algorithm, which weights individual movements based on differences in neighbor areas, improves the overall team score beyond what is expected from a single individual gain. Even in the scenario where almost all players follow S-1, a few S-2 players can have a positive impact on their fellow S-1 players.

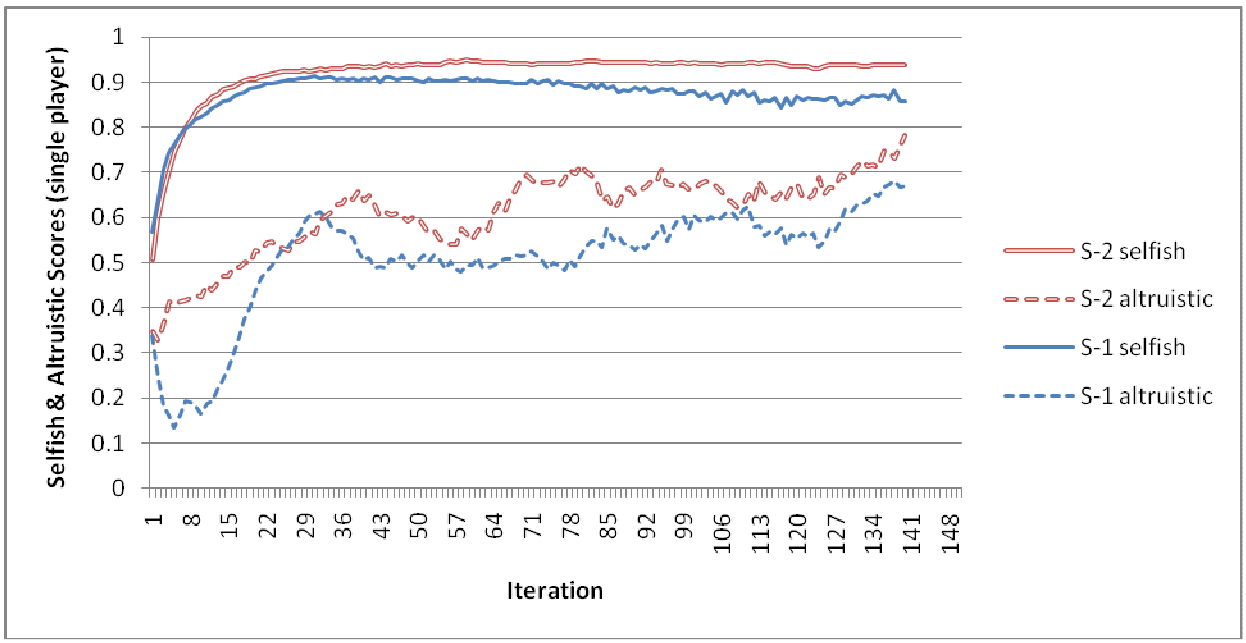

Figure 5: Selfish and altruistic scores for players operating with S-1 and S-2 algorithms on a team with 10 S-1 players and 10 S-2 players. Each line represents the average of 200 runs for different sets of initial conditions, with each run consisting of 140 iterations. Despite some noise in the results, there is clear evidence that the S-2 algorithm is more altruistic.

5 APPLICATION

One well-known application involving this problem is the positioning of satellites over the earth so they have equal area coverage. Other applications involve geo-spatial allocations, air traffic control situations, parallel computing allocations, and various needs of mapping and geographical information systems. We provide a sample of the power of the S-2 algorithm by showing the graphs in Figures 6 and 7 of equi-distributing 10 and 25 satellites, respectively, over the coverage region of New York State.

The dynamic power of these algorithms enable positions to be autonomously adjusted as satellites shift or drift or are incapacitated or added to the region. For example, programming the satellites with the S-2 algorithm would allow each satellite to alter their own coverage area in response to the change in the operability or coverage of a neighbor. This could be done locally by the satellites and would not require an outside controller to recalculate the necessary coverage and then program each satellites to move to a new designated location to ensure equal area coverage.

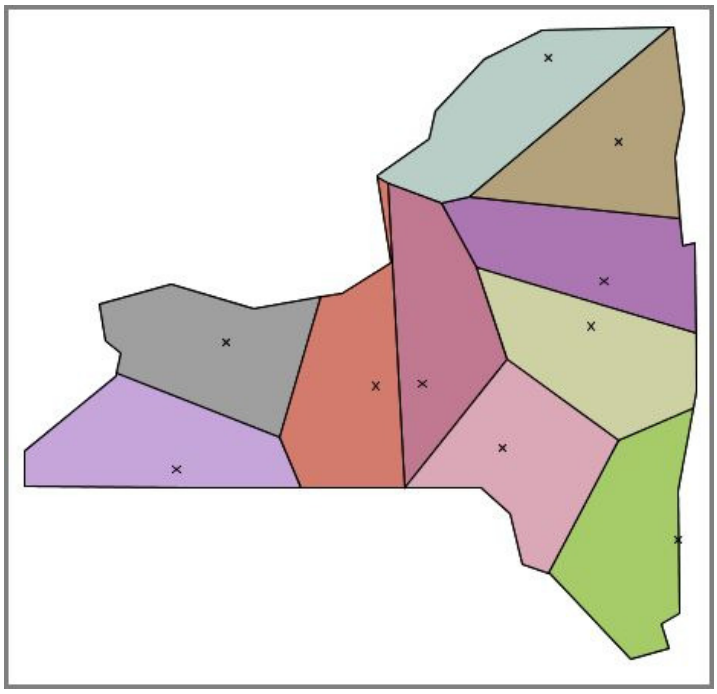

Figure 6: Result of the S-2 algorithm distributing 10 satellites over New York State.

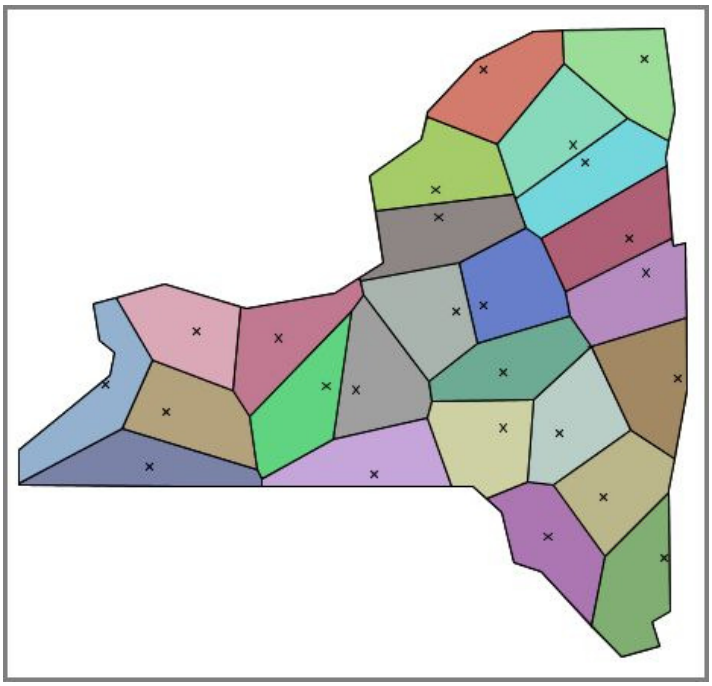

Figure 7: Result of the S-2 algorithm distributing 25 satellites over New York State.

6 CONCLUSION

The simulation results clearly indicate that local algorithms are sufficient to achieve a global balance in areas assigned to individual entities, and in the case of the weighted-area algorithm (S-2), the convergence of these local solutions can be surprisingly quick. It is even more surprising that algorithms that do not directly use information about area also show good performance, although they do not converge.

The results obtained here using the Equi-Distribution Platform inform our understanding of how metrics for a basic cooperation algorithm can be used to understand the power of cooperation. Underlying the simulation is a geometry network comprised of individual entities and their neighboring entities, and we have demonstrated that this kind of geometric system can solve global problems using only local information.

We also seek to understand this system as a cooperative geometric network. The main idea behind this development is that better networks with increased performance metrics will reduce communications load and increase efficiency. Our results indicate that certain algorithms perform better in heterogeneous teams, both from a selfish perspective and from an altruistic perspective. This quantitative analysis of the cooperative nature of algorithms could prove valuable in assessing situations where certain algorithms are expensive but perform better than others.

The Army has many networks and organizations that operate primarily on the basis of cooperation. The Army's culture is based on highly technical, cooperative teams. Many of the new emerging technologies of net-centric warfare involve using cooperation in the form of hybrid systems – specialized teams of people, machines, computers, and robots (Alberts, Garstka, and Stein 1999). At the next layer of detail the major elements of cooperation are the mix of trust and autonomy of the agents (i.e., lack of strict control). The American Army is known for its trust and autonomy and this culture will continue in our future doctrine; and therefore, understanding, designing for and using cooperation are critical elements in meeting the goals of the future highly networked Army. This fundamental research in the basic mathematical principles of cooperation can contribute greatly to that effort and these important Army goals.

7 REFERENCES

Alberts, D. S., J. J. Garstka, and F. P. Stein. 1999. *Network Science Warfare: Developing and Leveraging Information Superiority*. Washington, DC: CCRP.

Arney, D. C. and E. Peterson. 2008. Cooperation in Social Networks: Communication, Trust, and Selflessness. *Proceedings of the 2008 Army Science Conference*.

Axelrod, R. 1997. *The Complexity of Cooperation: Agent-Based Models of Competition and Collaboration*, Princeton, New Jersey: Princeton University Press.

Axelrod, R. and M. D. Cohen. 2000. *Harnessing Complexity: Organizational Implications of a Scientific Frontier*. New York: Basic Books.

Ben, J., X. Tong, Y. Zhang, and H. Zhang, 2006. Discrete Global Grid Systems: Generating Algorithm and Software Model. *Geoinformatics 2006* 6421: 1-13.

Malone, T. W., R. Laubacher, and M. S. S. Morton. 2003. *Inventing the Organizations of the 21st Century*. MIT Press.

Ohyama, T. 2007. Division of a region into equal areas using additively weighted power diagrams. In *IEEE Proceeding of the 4th International Symposium of Voronoi Diagrams in Science and Engineering*.

Snyder, J. 1992. An Equal-Area Map Projection for Polyhedral Globes. *Cartographica* 29: 10-21.

Strogatz, S. 2003. *Sync: The Emerging Science of Spontaneous Order*. Hyperion.

Tapscott, D. and A. D. Williams. 2006. *Wikinomics: How Mass Collaboration Changes Everything*. Portfolio.

v. Neumann, J. 1928. Zur Theorie der Gesellschaftsspiele. *Math Annalen* 100: 295-320.

AUTHOR BIOGRAPHIES

CHRIS ARNEY is a Professor of Mathematics at the United States Military Academy. His graduate studies led to a PhD in mathematics from Rensselaer Polytechnic Institute. Chris spent most of his 30-year military career as a mathematics professor at West Point (NY). He also served as the Dean of Mathematics and Sciences and as Interim Vice President for Academic Affairs at the College of Saint Rose in Albany (NY) and the division chief of the Mathematical Sciences Division of the Army Research Office (NC). There he managed and performed research in the area of cooperative systems, with particular interest in information networks, pursuit-evasion modeling, intelligence processing, artificial intelligence, and language for robots. Chris has authored 22 books, written over 120 technical articles, made over 240 presentations, and reviewed over 200 books. His technical areas of interest include mathematical modeling and cooperative systems. His primary teaching interests are in modeling and inquiry.
His email address is <david.arney@usma.edu>.

KRISTIN ARNEY is an Instructor of Mathematics at the United States Military Academy. She is currently a Major in the United States Army serving in the Military Intelligence branch. She received a Bachelor of Science in Mathematics from Lafayette College in Easton, PA and a Master of Science in Operations Research from North Carolina State University in Raleigh, NC. She directs a course in discrete modeling and conducts her research in resource allocation. Her email address is <kristin.arney@usma.edu>.

ELISHA PETERSON is an Assistant Professor and Davies Postdoctoral Fellow at the United States Military Academy (West Point). He received a BS in Mathematics from Harvey Mudd College, and completed graduate work at Oxford University and the University of Maryland, College Park, where he was awarded a PhD in 2006. His research interests include game theory, pursuit-evasion modeling, the mathematics of cooperation, multi-agent systems, diagrammatic algebras, and graph theory. Elisha has authored numerous Java applets for visualizing mathematics and for algorithm simulation, as well as a Java-based platform called Blaise designed to streamline the creation of such applets. His email address is <elisha.peterson@usma.edu>.