

# A Greedy Randomized Algorithm for the Cooperative Communication Problem on Ad Hoc Networks

Clayton Commander, *University of Florida*\*

Carlos A.S. Oliveira, *Oklahoma State University*<sup>†</sup>

Panos M. Pardalos, *University of Florida*\*

Mauricio G.C. Resende, *AT&T Labs Research*<sup>‡</sup>

Marco Tsitselis, *Università degli Studi di Napoli FEDERICO II*<sup>§</sup>

Paola Festa, *Università degli Studi di Napoli FEDERICO II*<sup>§</sup>

**Keywords:** cooperative communication, ad hoc networks, metaheuristics, GRASP.

## 1. Introduction

An ad hoc network is a system of mobile nodes that communicate through the use of wireless links. Due to the lack of central authority, ad hoc networks have non-fixed topology, and their communication pattern depends on the actual position of individual nodes. This characteristic presents a great challenge for some fundamental design tasks such as ensuring connectivity, robustness, and proper routing over the network.

Applications of ad hoc networks are pervasive, including loosely coupled systems of personal digital assistants (and other equipment using wireless technology such as bluetooth) as well as sensor networks. Applications occur on urban and rural settings, including rescue missions and military operations. Optimization techniques have been applied on several ad hoc network problems, such as routing, correction of measurement errors, and ensuring general connectivity [5]. In this paper we propose an algorithm for group communication, which is a vital resource when a collaborative task is defined. In this case, we assume that members of the network must keep in contact for as long as possible, in order to exchange vital information.

Consider a graph  $G = (V, E)$ , where the set  $V$  of nodes corresponds to candidate positions in a given area. The set  $U$  contains elements representing members of the ad hoc network. Each element  $u \in U$  has associated position  $p_t(u)$ , at each instant  $t \in 1, \dots, T$ , where  $T$  is the given time horizon. Assume that the initial and final positions for each  $u \in U$  are known, and given respectively by  $s_u$  and  $d_u$ , for each  $u \in U$ . Each edge  $e \in E$  has an associated distance given by function  $\text{dist} : E \rightarrow R_+$ . Elements of the wireless network move according to simple rules. At each time step  $t$  an element can stay at the same position or move to one of the positions in  $N(p_t(u))$  – where  $N : V \rightarrow 2^V$  is a function returning the neighborhood of a node.

Two elements of the wireless network are connected at a given time instant if the distance between them is less than the radius  $\tau$ , which is determined by the range of the wireless equipment. Total communication is measured as the summation of the number of connections achieved by the system over time. To formalize this, let  $r : V^2 \rightarrow \{0, 1\}$  be a function returning 1 if the distance between nodes is less than  $\tau$ . Then, the objective function for our problem can be represented as  $\sum_{t=1, \dots, T} \sum_{u, v \in U} r(p_t(u), p_t(v))$ . Moreover, we require that each path  $P$  from  $s_u$  to  $d_u$ , for each  $u \in U$ , have total distance  $\sum_{e \in P} \text{dist}(e)$  of at most  $D_u$  units, where  $D_u$  is a natural limit on the travel autonomy of wireless agent  $u$ .

It has been determined [4] that the problem described above is NP-hard. This can be shown by a reduction from the well known 3SAT problem. Moreover, it is NP-hard even to find an optimal solution for one stage of the problem at a given time  $t$ . To see this, consider an algorithm that maximizes the number of connections

---

\*Department of Industrial and Systems Engineering. Address: 303 Weil Hall, Gainesville, FL 32611 USA. Email: {clayton8, pardalos}@ufl.edu

<sup>†</sup>School of Industrial Engineering and Management. Address: 322 EN, Stillwater, OK 74078, USA. Email: coliv@okstate.edu

<sup>‡</sup>Internet and Network Systems Research Center. Address: Room C241, 180 Park Avenue, Florham Park, NJ 07932 USA. Email: mgcr@research.att.com

<sup>§</sup>Dipartimento di Matematica e Applicazioni. Address: Compl. Monte S. Angelo, via Cintia, 80126 Napoli, Italy. Email: {marco.tsitselis, paola.festa}@unina.it

at time  $t$ , by defining the positions for members of the network. Run this algorithm for different sets  $U^i$ , with  $|U^i| = i$  and  $i$  varying from 1 to  $T$ . Then, the algorithm stops when the number of connections is less than  $\binom{i}{2}$ , and the value returned is  $i - 1$ ; clearly, the algorithm described above computes the value of the maximum clique on the underlying unit graph. However, finding the maximum clique on a unit graph is known to be NP-hard [1].

In this paper, we propose a heuristic for maximizing connectivity, based on the greedy randomized adaptive search algorithm (GRASP) of Feo and Resende [3]. The algorithm provides a quick way of determining solutions for the problem without the necessity of a full scale enumerative algorithm. The solutions provided by the resulting algorithm are close to the optimal for most of the tested instances.

## 2. Greedy Randomized Adaptive Search Procedure

Greedy randomized adaptive search procedure is a metaheuristic for combinatorial optimization that aims to find very good solutions through the controlled use of random sampling, greedy selection, and local search. The metaheuristic has been used in the last decade in several optimization problems with very good results in practice. Let  $F$  be the set of feasible solutions for the problem  $\Pi$ , where each solution  $S \in F$  is composed of  $k$  discrete components  $a_1, \dots, a_k$ . GRASP constructs a sequence  $\{S\}_i$  of solutions for  $\Pi$ , such that each  $S_i$  is feasible for  $\Pi$ . At the end, the algorithm returns the best of the solutions found.

Each GRASP solution is built in two stages, called *greedy randomized construction* and *intensification* phases. The construction phase receives as parameter an instance of the problem, a ranking function  $g : A(S) \rightarrow R$  (where  $A(S)$  is the domain of feasible components  $a_1, \dots, a_k$  for a partial solution  $S$ ), and a constant  $0 < \alpha < 1$ . It starts with an empty partial solution  $S$ . Assuming that  $|A(S)| = l$ , the algorithm creates a list of the best ranked  $\alpha l$  components in  $A(S)$ , and returns a uniformly chosen element  $x$  from this list. The current partial solution is augmented with  $x$ , and the procedure is repeated until the solution is feasible, i.e.,  $S \in F$ .

The intensification phase consists of the implementation of a hill-climbing procedure. Given a solution  $S \in F$ , let  $N(S)$  be the set of solutions that can be found from  $S$  by changing one of the components  $a \in S$ . Then,  $N(S)$  is called the neighborhood of  $S$ . The improvement algorithm consists of finding, at each step, the element  $S^*$  such that

$$S^* = \arg \max_{S' \in N(S)} f(S'),$$

where  $f : F \rightarrow R$  is the objective function of the problem. At the end of each step we make  $S \leftarrow S^*$  if  $S < S^*$ . The algorithm will eventually achieve a local optimum, in which case the procedure above will generate a solution  $S^*$  such that  $S \geq S^*$ .

To employ GRASP for solving the cooperative communication problem in ad hoc networks, we need to specify the set  $A$ , the greedy function  $g$ , the parameter  $\alpha$ , and the neighborhood  $N(S)$ , for  $S \in F$ . The components of each solution  $S$  are feasible moves of a member of the ad hoc network from a node  $v$  to a node  $w \in N(v) \cup \{v\}$ . The complete solution is constructed according to the following procedure. Start with a random  $u \in U$  and find the shortest path  $P$  from  $s_u$  to  $d_u$ . If the total distance of  $P$  is more than  $D_u$ , then the instance is clearly infeasible, and the algorithm ends. Else, the algorithm considers each feasible move. A feasible move connects the final node of a sub-path  $P_v$ , for  $v \in U \setminus \{u\}$ , to another node  $w$ , such that the shortest path from  $w$  to  $d_v$  has distance at most  $D_v - \sum_{e \in P_v} \text{dist}(e)$ . The set of all feasible moves in a solution is defined as  $A(S)$ .

The greedy function  $g$  returns for each move in  $A(S)$  the number of additional connections created by that move. As described above, the construction procedure will rank the elements of  $A(S)$  according to  $g$ , and return one of the best  $\alpha l$  elements. This is repeated until a complete solution for the problem is obtained.

The improvement phase is defined by the perturbation function, which consists of selecting a wireless agent  $u \in U$  and rerouting it, i.e., finding a complete path using the procedure described above for each time step 1

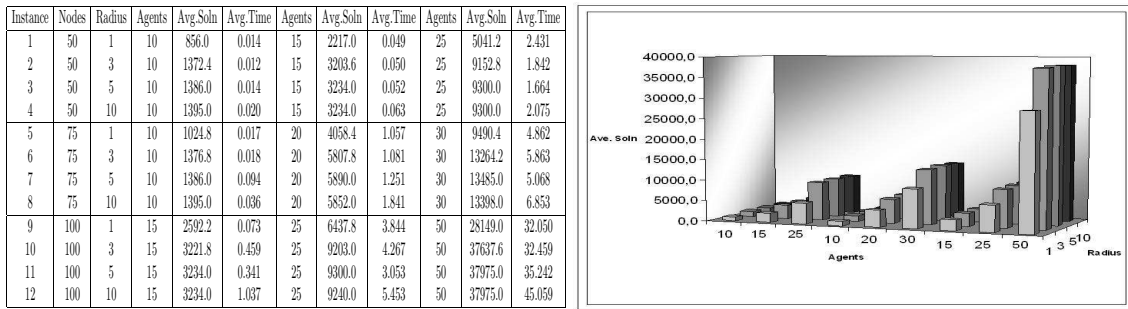


Figure 1: Left: numerical results of the algorithm. Right: evolution of solution values.

to  $T$ . The set of all perturbations of a solution  $S$  is its neighborhood  $N(S)$ . At each step, all elements  $u \in U$  are tested, and the procedure stops when no such element  $u$  that improves the current solution can be found.

### 3. Computational Experiments

The described procedure has been implemented in the C programming language using the Linux operating system. It was tested on a Dell Power Edge 2600 with twin 3.2GHz Pentium 4 processors with 1M cache and 6GB of RAM running in hyperthreading mode. The proposed GRASP was tested on 60 random unit graphs with varying densities ranging from 50 to 100 nodes. The radii of communication vary from 1 to 10 units (miles). We tested each case with different sets of mobile agents in order to achieve better comparisons and model real-world scenarios. The graphs were created by a generator used by Butenko et al. on the BROADCAST SCHEDULING problem [2].

Numerical results can be seen in Figure 1. The results summarize the solutions from the average of 5 graphs having the same number of nodes, radius of communication, and number of mobile agents. As expected, we obtain better average solutions as we increase the communication range and the number of agents. Subsequently, as the complexity of the instance gets larger, this leads to longer computation times. Nevertheless, the average solution was found in 1.71s not counting the 100 node graphs where 50 agents were routed. The average time including these instances was 5.54s. We see that the proposed GRASP is very robust in that it is able to solve a variety of instances in a fraction of the time that would be required by a pure IP solver.

1

### References

- [1] B.N. Clark, C.J. Colbourn, and D.S. Johnson. Unit disk graphs. *Discrete Mathematics*, 86:165–177, 1990.
- [2] Clayton W. Commander, Sergiy I. Butenko, and Panos M. Pardalos. On the performance of heuristics for broadcast scheduling. In D. Grundel, R. Murphey, and P. Pardalos, editors, *Theory and Algorithms for Cooperative Systems*, pages 63–80. World Scientific, 2004.
- [3] Thomas A. Feo and Mauricio G.C. Resende. Greedy randomized adaptive search procedures. *Journal of Global Optimization*, 6:109–133, 1995.
- [4] Carlos A.S. Oliveira and Panos M. Pardalos. An optimization approach for cooperative communication in ad hoc networks. Technical report, School of Industrial Engineering and Management, Oklahoma State University, 2005.
- [5] Carlos A.S. Oliveira, Panos M. Pardalos, and Mauricio G.C. Resende. Optimization problems in multicast tree construction. In *Handbook of Optimization in Telecommunications*, pages 701–733. Kluwer, Dordrecht, 2005.